Abschlussbeispiel

UK Computational Statistics

Sicherstellen, dass keine alten Objekte im File sind:

```
rm(list=ls())


## libraries
library (cluster)
#install.packages("randomcoloR")
library(randomcoloR)

## default daten:
daten <- faithful
```

VARIABLES FOR TESTING

k <- 3

x <- faithful

trace <- FALSE

maxiter <- 10

change <- 0.01

————————————————————

FUNCTION KMEANS:

————————————————————

input-parameter:

x … dataframe (dimension nx2, d.h. n rows, 2 cols)

k … anzahl der zu bildenden teilgruppen

trace=FALSE … falls TRUE, dann soll für jeden zwischenschritt eine grafik produziert werden

maxiter=10 … maximale anzahl von iterationsschritten

change=0.001 … abbruchswert für die relative änderung

output (list):

iter … anzahl der durchgeführten iterationsschritte

zentren … matrix der dimension kx2 (enthält für jede teilmenge die mittelwerte der beiden variablen)

index … vektor der länge n (enthält für jeden datenpunkt info, zu welcher teilmenge er gehört)

distanz … vektor der länge n (enthält für jeden datenpunkt die distanz zum zentrum seiner teilmenge)

```
KMEANS <- function(x=daten, k=3, trace=FALSE, maxiter=10, change=0.001){

  # OUTPUT INITIALISIERUNG:
  iter <- 0
  zentren <- NULL
  index <- rep(-1,nrow(x))
```

```r
distanz <- rep(-1,nrow(x))

# weitere intialisierungen:
distanzensumme <- 0
relativeAenderung_DistanzenSumme <- change+1
group_means <- data.frame(xvalue=rep(0,k), yvalue=rep(0,k))

if (trace == TRUE) {
  #x11(680, 380)

  par(mfcol = c(3, 4))
  par(bg = "gray97")
  library(randomcoloR)
  anz <- k
  palette <- distinctColorPalette(anz)

}


# SCHRITT 1:
# waehle zufaellig k punkte aus beobachtungen als startloesung fuer die
  gruppenmittelwerte
# stelle sicher, dass keine identischen beobachtungspaare unter den k
  ausgewaehlten punkten

if(k > nrow(x)){
  stop("k muss kleiner als anzahl der beobachtungen sein!")
}

x_unique <- x[!duplicated(x), ] # duplikate herausnehmen
randomStartingRowIndices <- sample(nrow(x_unique), size=k, replace=FA
LSE)
randomStartingPoints <- x_unique[randomStartingRowIndices, ]
rownames(randomStartingPoints) <- 1:k

# plot wenn trace ist TRUE
if(trace == TRUE) {
  plot(x, xlab = '', ylab = '', main = 'Start', col = "lightcyan3")
  points(randomStartingPoints, pch = 4, col = palette[1:k], lwd = 3)
}


# SCHRITT 2:
# bestimme faer jeden punkt die euklidschen distanzen zu den aktuellen
```

```
   gruppenmittelwerten

  while(iter < maxiter && relativeAenderung_DistanzenSumme >= change){

    # check abbruchbedingungen (i.e. schritt 4)

    for(i in 1:nrow(x)){ # i ... anzahl der beobachtungen


      distanzenZuClustern_proBeobachtung <- rep(0,k)

      for(j in 1:k){ # j ... anzahl der k zu bildenden gruppen

        ## berechne euklidsche distanzen
        if(iter==0){

          distanzenZuClustern_proBeobachtung[j] <- sqrt(

          (x[i,1]-randomStartingPoints[j,1])^2 +
          (x[i,2]-randomStartingPoints[j,2])^2

          )
        }
        else{

          distanzenZuClustern_proBeobachtung[j] <- sqrt(

          (x[i,1]-group_means[j,1])^2 +
          (x[i,2]-group_means[j,2])^2

         )
        }

      }


      ## setze distanz (distanz zum gewaehlten cluster-mittelpunkt)
      distanz[i] <- min(distanzenZuClustern_proBeobachtung)

      ## setze index (clusterzuordnung gemaess minimaler distanz)
      index[i] <- which(distanzenZuClustern_proBeobachtung == distanz[i])
[1]

    }


    ## distanzsumme und relative aenderung davon in laufender iteration
```

```
    if(iter!=0){
      relativeAenderung_DistanzenSumme <- abs(distanzensumme - sum(dist
      anz))  / distanzensumme
    }

    distanzensumme <- sum(distanz)


    ## SCHRITT 3: bestimme aufgrund von aktueller gruppenzugehoerigkeit
                  der datenpunkte fuer jede der k gruppen durch anwendu
                  ng von "mean" neue gruppenmittelwerte

    for(a in 1:k){
      rowIndex_groupK <- which(index==a)
      group_means[a,1] <- mean(x[rowIndex_groupK,1]) # xvalue means
      group_means[a,2] <- mean(x[rowIndex_groupK,2]) # yvalue means
    }

    if (trace == TRUE) {
      plot(x, xlab = '', ylab = '', main = paste("Iteration", iter + 1)
      , col = palette[index])
      points(group_means, col = palette[1:k], pch = 4, lwd = 4)
    }

    ## iterationsschritte-anzahl erhoehen
    iter <- iter + 1

  }


  ## RETURN liste mit iter, zentren, index, distanz
  list(iter=iter, zentren=group_means, index=index, distanz=distanz)

}
```

TEST AUFRUF:

FALL: Parameter trace ist FALSE

```r
ergebnis <- KMEANS(daten, 10, FALSE, 10, 0.001)
ergebnis
```

```
## $iter
## [1] 7
##
## $zentren
##       xvalue   yvalue
## 1  4.304755 82.42857
## 2  4.056853 72.58824
## 3  4.600000 93.60000
## 4  4.339148 77.92593
## 5  1.930250 46.25000
## 6  4.330857 85.71429
## 7  2.039848 54.96970
## 8  1.990571 50.52381
## 9  4.527000 89.12500
## 10 2.314400 62.00000
##
## $index
##   [1]  4  7  2 10  6  7  9  6  8  6  7  1  4  5  1  8 10  1  8  4  8
5  4
##  [24]  2  2  1  7  4  4  4  2  4 10  4  2  8  5  4 10  9  4  7  1  7
2  1
##  [47] 10  7  1 10  2  9  7  4  7  1  2 10  4  1 10  1  5  1 10  3  4
4 10
##  [70]  2  1  7  4  2 10  4 10  4  4  1  2  1  2 10  2  9  4  4  5  6 1
0  9
##  [93]  8  4 10  2  1  2  8  1 10  9  8  1  1  5  1  8  6  1  2 10  9
4 10
## [116]  1  8  6 10  6  7  2  4  7  9  1  5  1  7  9  5  1  7  9  5  1
8  6
## [139]  7  4  1 10  1  4  4 10  4  8  3  7  4  4 10  1  2  2  1  3  7
9  5
## [162]  6  7  4 10  4 10  9  8  3  8  7  4  2  1  1  2  8  6  2  7  4
1  1
## [185]  8  4  1  5  1  7  1  7  4  1  4  1  6  4  8  4 10  1  9  7  4
5  4
## [208]  1  8  1  2  4  8  2 10  4  7  3  7  4  8  1  7  2  4  4  4  4
2  4
## [231]  2  7  6  8  9  7  7  4  4 10  2  5  6 10  6  1  7  1 10  2  7
1  2
## [254]  2  9  4  2  1  7  4  4  1  7  1  5 10  2  1  5  9  5  2
##
## $distanz
##   [1] 1.30383093 0.99891917 1.58651904 0.03140000 0.74233807 0.8436958
```

```
9
##   [7] 1.13822406 1.02193750 0.47791569 0.71454218 0.99151324 1.6185616
4
##  [13] 0.15763621 0.77135599 0.69480151 1.48669612 0.56440000 1.6476210
9
##  [19] 1.52698538 1.07776737 0.51290822 0.77135599 0.89222833 3.7222629
4
##  [25] 1.48989785 0.90730941 0.07889978 1.94288511 0.49472505 1.0781666
3
##  [31] 0.47819522 0.93471114 4.13617780 2.09654715 1.42940188 1.4764270
4
##  [37] 1.75114264 2.13205838 3.03837884 0.91168032 2.07410246 3.0343595
5
##  [43] 1.59316043 3.04413347 0.62949678 1.14113573 2.51120409 1.9706152
2
##  [49] 0.53983162 3.01642957 2.52366332 0.89517931 0.99151324 2.1320583
8
##  [55] 1.01708810 0.81295619 1.62418945 2.10217191 0.95354879 1.4286239
1
##  [61] 3.00110412 1.58351139 1.75925838 0.65493586 2.06092376 1.6124515
5
##  [67] 0.18740852 0.36837620 3.01018384 0.76366767 0.50746853 1.0328752
3
##  [73] 1.08605176 1.58925253 0.33140000 2.05887323 2.02199079 0.2395901
4
##  [79] 1.97920737 0.90730941 2.41296651 0.42950116 2.58859491 3.0168702
3
##  [85] 0.41188971 1.19601881 1.96484782 2.08168551 1.76594183 0.4371488
3
##  [91] 2.00326917 0.89624829 0.53818799 0.48355906 1.10984051 0.6365070
7
##  [97] 1.61264042 2.43120705 0.49196267 0.73347799 0.16860000 1.1363208
2
## [103] 1.52773364 0.60386355 1.45110871 0.75266232 1.62037227 1.4907126
6
## [109] 0.59257232 1.55801018 2.50475225 3.00003456 0.39338785 1.0768918
4
## [115] 3.06226834 1.46579707 0.61719691 0.76330987 3.04095491 1.2885968
4
## [121] 2.05251310 3.58824964 0.93020762 1.03287523 1.12736596 1.5264327
3
## [127] 1.25007022 0.47095015 0.22916388 0.88360285 1.25159920 0.5877985
0
## [133] 1.28037286 0.23078345 0.26824907 0.43565552 0.48818949 0.6664898
1
## [139] 1.96970888 1.23330884 1.43037237 2.00165580 0.48556071 1.0419602
```

```
7
## [145] 1.92593574 3.01824882 2.09478690 1.52403869 2.45153013 1.9842463
2
## [151] 1.15705195 0.98608331 3.00122098 1.45876176 1.66206115 2.5888596
3
## [157] 1.44185190 0.79201578 1.98424632 0.57378132 1.27877483 0.3381448
8
## [163] 3.03056502 0.51153975 4.17200759 1.94130224 1.00138242 1.2203909
2
## [169] 1.47731269 0.60024079 1.52558455 2.03076154 0.95749807 4.6449829
1
## [175] 1.43519782 1.42885062 0.60492106 0.67543893 0.78719155 1.4160550
7
## [181] 0.15974893 0.95749807 0.57404593 0.78467265 0.47807693 0.1195622
8
## [187] 1.58699813 0.26824907 0.58234829 0.14632372 1.51198010 2.0408127
5
## [193] 1.98029672 1.58471209 0.99828771 1.43037237 1.53081182 0.9263152
0
## [199] 0.54227350 0.33611576 2.01145901 0.43095310 1.91594911 1.9772664
8
## [205] 0.27116537 0.29014231 0.92634472 1.63590646 1.52489670 0.6038635
5
## [211] 2.30743906 2.10523094 1.52881175 2.42213111 2.28379657 1.9288489
1
## [217] 2.00235243 0.44721360 0.05006172 1.93519190 0.53818799 0.4302312
4
## [223] 1.01208911 2.44912440 0.34714325 1.09680669 0.26664366 0.1034036
9
## [229] 2.59201095 1.09457463 2.58836736 1.04045926 0.32170544 0.5706542
9
## [235] 0.87838147 0.98230019 0.98810660 0.92762678 1.14239722 2.0000864
9
## [241] 2.41356280 0.85947080 0.66648981 1.15884743 0.75748274 0.6373588
8
## [247] 2.03076154 0.43306800 5.00328951 1.44187891 0.98283301 0.5895987
6
## [253] 0.63992662 0.60492106 1.18648810 2.13878983 1.59438082 0.5895987
6
## [259] 1.03107334 1.07554067 0.43421674 1.58791797 3.03624421 0.5740459
3
## [265] 3.25042806 2.00103657 2.50939472 1.44085680 0.33285141 0.8818871
8
## [271] 0.27445503 1.47013611
```

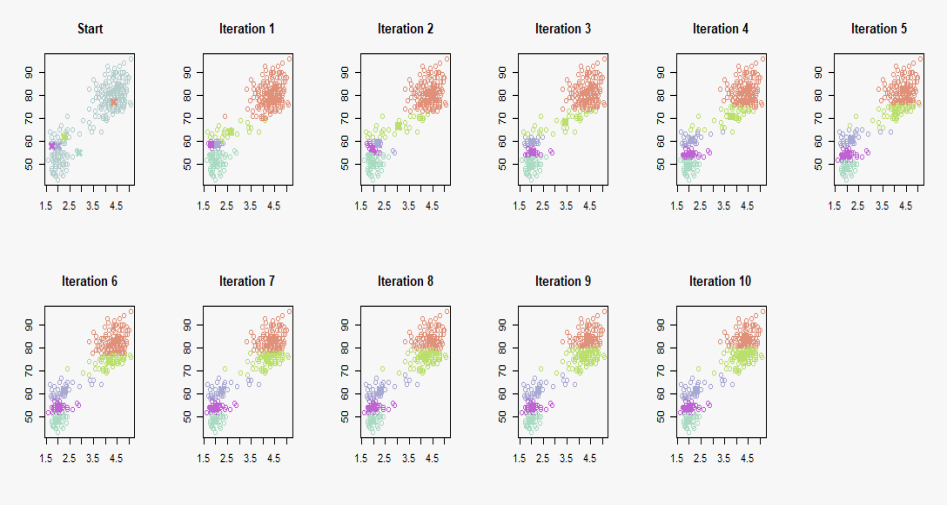
FALL: Parameter trace ist TRUE:

```
ergebnis <- KMEANS(daten, 5, TRUE, 10, 0.001)
ergebnis
```

```
## $iter
## [1] 8
##
## $zentren
##     xvalue   yvalue
## 1 4.439578 87.42222
## 2 2.008238 50.98413
## 3 4.293972 80.54167
## 4 4.187218 74.10909
## 5 2.240919 61.16216
##
## $index
##   [1] 3 2 4 5 1 2 1 1 2 1 2 1 3 2 3 2 5 1 2 3 2 2 3 4 4 3 2 4 3 3 4 4
5 3 4
##  [36] 2 2 3 5 1 3 5 1 5 4 3 5 2 3 5 4 1 2 3 2 3 4 5 4 3 5 1 2 3 5 1 3
3 5 4
##  [71] 3 2 3 4 5 4 5 3 4 3 4 3 4 5 4 1 4 3 2 1 5 1 2 3 5 4 1 4 2 3 5 1
2 3 3
## [106] 2 1 2 1 3 4 5 1 3 5 3 2 1 5 1 2 4 4 2 1 3 2 3 2 1 2 3 2 1 2 3 2
1 2 3
## [141] 3 5 3 4 4 5 3 2 1 2 4 4 5 3 4 4 3 1 2 1 2 1 5 3 5 4 5 1 2 1 2 5
4 4 3
## [176] 3 4 2 1 4 2 4 3 3 2 3 1 2 3 2 3 5 4 1 4 3 1 4 2 3 5 3 1 2 3 2 4
1 2 3
## [211] 4 3 2 4 5 4 2 1 2 4 2 3 2 4 3 3 3 3 4 3 4 2 1 2 1 2 2 4 3 5 4 2
1 5 1
## [246] 3 5 3 5 4 2 3 4 4 1 3 4 3 2 3 3 1 5 3 2 5 4 3 2 1 2 4
##
## $distanz
##   [1] 1.69066069 3.02305361 0.86115593 0.83889395 2.42402314 4.1100419
1
##   [7] 0.63375618 2.56360124 0.06036247 2.42387802 3.02095985 3.4618914
6
##  [13] 2.54340328 3.99248729 2.49163828 1.02820393 0.97106829 3.4411493
9
##  [19] 1.09483164 1.54229364 0.20884218 3.99248729 2.67812594 5.2304587
5
##  [25] 0.36258226 2.55440800 4.01608474 1.89377893 2.58015131 1.5479227
5
##  [31] 1.11481047 2.90441616 4.96716554 0.60125642 0.37063641 1.0159108
```

```
##  [37] 2.98746750 0.76416865 2.20030522 2.60055323 0.54455660 3.1823537
7
##  [43] 3.42459361 3.20004233 1.16174339 2.64535017 3.25392774 2.0179604
2
##  [49] 1.49722274 2.17554295 1.08130503 2.59255617 3.02095985 0.7641686
5
##  [55] 3.02840652 2.52791545 3.14444771 2.89529040 2.91574855 0.4589114
5
##  [61] 2.16217666 3.42275558 2.99527975 1.54363215 1.23706432 4.5779488
6
##  [67] 2.54483622 2.57389355 3.84177655 1.22189518 1.48150012 5.0160425
3
##  [73] 1.55537248 3.11472261 0.87663813 2.08555826 1.18353731 2.5562890
3
##  [79] 1.91522476 2.55440800 0.89255735 1.45885547 4.11001644 3.8578137
4
##  [85] 1.11558731 0.75979777 1.90573074 0.58578509 2.98834724 1.4886049
4
##  [91] 1.16288230 2.57998006 0.99421030 2.59492341 1.88256361 2.1121042
1
##  [97] 3.42977052 0.99241057 0.14212724 1.57924215 0.87210979 0.5823183
8
## [103] 1.98624775 2.46695161 0.51922239 3.98662966 3.43211665 1.0405432
2
## [109] 1.48025756 0.76377778 1.04479510 2.16296921 1.64358490 1.5465678
0
## [115] 2.22879754 0.57009585 1.03142612 2.42752874 2.20332759 0.4228254
5
## [121] 2.10578609 5.11050510 2.89159073 5.01604253 0.59963526 0.6984047
3
## [127] 5.98482248 1.47281484 4.02420101 2.58635183 5.98579351 2.4616101
9
## [133] 5.07797883 1.58137331 4.98720665 1.46104827 0.12623998 1.5053841
8
## [139] 2.01602509 1.64055660 0.46237112 1.16218914 1.47779240 2.9587126
4
## [145] 1.89652037 2.17749108 0.63901691 1.98414633 8.60316390 2.0265998
9
## [151] 3.01209267 2.89696497 3.84113343 0.55111019 3.17034965 4.1133537
1
## [157] 0.50251059 5.58916386 2.02659989 1.64703141 5.98719871 1.4514032
3
## [163] 3.17132645 2.58313082 4.99899591 1.93188520 1.84215753 0.8049225
0
## [169] 1.01865537 5.58059886 1.98622362 4.16515693 2.91787570 6.1685233
```

```
6
## [175] 0.47559583 0.45999197 1.15235199 1.06564169 2.46178576 0.1109486
4
## [181] 4.01782536 2.91787570 2.45872657 2.51418028 0.02941266 2.5454662
0
## [187] 3.44074885 4.98720665 2.46140988 4.01967384 0.68273974 4.1821037
4
## [193] 1.98772200 3.43902864 2.89936077 0.46237112 1.03008641 2.8964320
0
## [199] 0.24228242 2.56889454 1.17067461 1.45940920 3.59088899 2.0208147
4
## [205] 2.56002391 4.98921377 2.89649389 3.47263688 1.98555299 2.4669516
1
## [211] 3.59466960 0.67695002 1.98914758 0.95874372 3.07188709 1.8914632
3
## [217] 2.05358745 6.58764485 4.01588147 1.89127533 0.99421030 1.4585827
4
## [223] 3.02690881 0.93872567 2.55861078 1.55179099 2.55040756 2.5418097
8
## [229] 4.11796624 1.56278160 4.11041233 3.04344810 1.44518103 1.0060255
7
## [235] 2.57779885 3.01847223 3.02002141 2.89249538 1.57957368 2.8393313
5
## [241] 0.89168616 3.99875844 1.50538418 1.95244354 2.42646459 1.5294546
4
## [247] 4.16515693 1.46016066 5.83883525 0.19595598 3.02196338 2.4632798
1
## [253] 1.27072941 1.15235199 0.64628357 0.72173768 3.12081146 2.4632798
1
## [259] 5.01587978 1.54170571 2.58530941 3.42349714 3.18623401 2.4587265
7
## [265] 7.98416687 1.16219764 1.05377530 0.49131315 4.98614262 2.5778766
5
## [271] 4.98779448 0.30029767
```



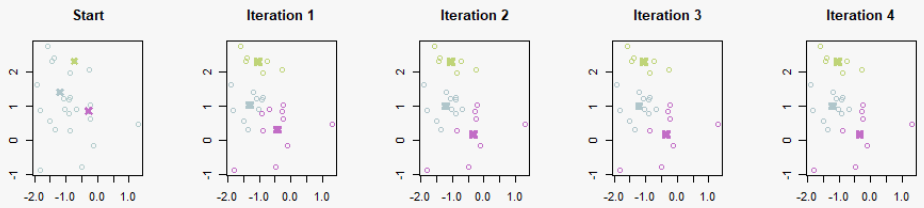
SIMULIERTER DATENSATZ:

generieren Sie 4 Stichproben mit je 25 Beobachtungen (insgesamt n=100), und folgenden Mittelwerten (-1,1), (-1,-1), (1,1), (1,-1), die Werte für die beiden Variablen sollen jeweils um den Mittelwert normalverteilt mit Standardabweichung 1 sein.

```
set.seed(100)

stichprobe1 <- data.frame(xvalue = rnorm(25, mean = -1, sd = 1),yvalue= r
norm(25, mean = 1, sd =1))
```

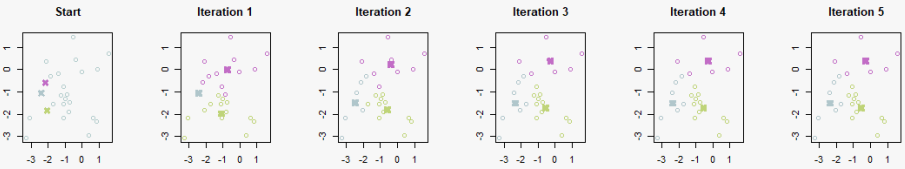
```
KMEANS(stichprobe1, 3,TRUE, 10, 0.001)

## $iter
## [1] 5
##
## $zentren
##       xvalue     yvalue
## 1 -1.0245436 -0.1103079
## 2 -1.1710525  2.2086211
## 3 -0.6625892  0.9643872
##
```

```
## $index
##  [1] 1 1 3 1 3 3 2 3 3 1 3 3 3 2 2 3 2 1 2 3 2 3 2 3 1
##
## $distanz
##  [1] 0.82434255 0.42015079 0.49434978 0.91256134 0.35847806 0.05859226
##  [7] 0.68544684 0.39075526 1.16512424 0.53766647 0.30972889 0.32541961
## [13] 0.70407408 0.92208337 0.37885865 0.37262211 0.29206146 0.85488724
## [19] 0.94593987 2.03202548 0.29020027 0.53805295 0.44687695 0.44316324
## [25] 1.10190691
```



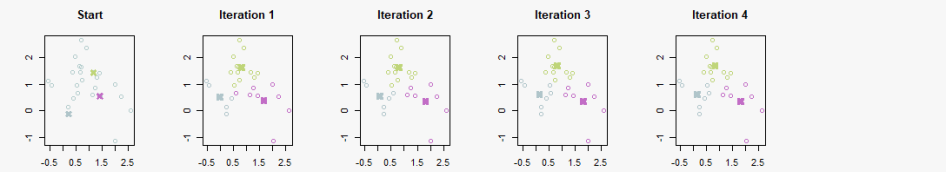
Formatiert: Absatz-Standardschriftart, Schriftart: +Textkörper (Calibri), Englisch (Vereinigtes Königreich)

```
stichprobe2 <- data.frame(xvalue = rnorm(25, mean = -1, sd = 1),yvalue= r
norm(25, mean = -1, sd =1))
KMEANS(stichprobe2, 3,TRUE, 10, 0.001)
```



```
## $iter
## [1] 6
##
## $zentren
##      xvalue      yvalue
## 1  1.239712  0.36781381
## 2 -1.164342 -1.85252456
## 3 -1.183646  0.03634394
##
## $index
##  [1] 1 2 3 2 2 2 3 2 3 3 1 2 2 3 2 2 2 3 2 3 2 2 2 3 2
##
## $distanz
##  [1] 0.4962598 2.4361653 1.1724410 1.4724142 2.0493531 1.8848261 0.732
1672
##  [8] 0.7014504 0.8279735 0.2933784 0.4962598 0.8044702 0.6231501 0.910
3422
## [15] 0.3682595 0.3193266 0.3301418 1.5447829 0.9002370 1.1601326 1.841
9698
## [22] 1.9247964 0.5489506 0.9611360 0.5797410
```

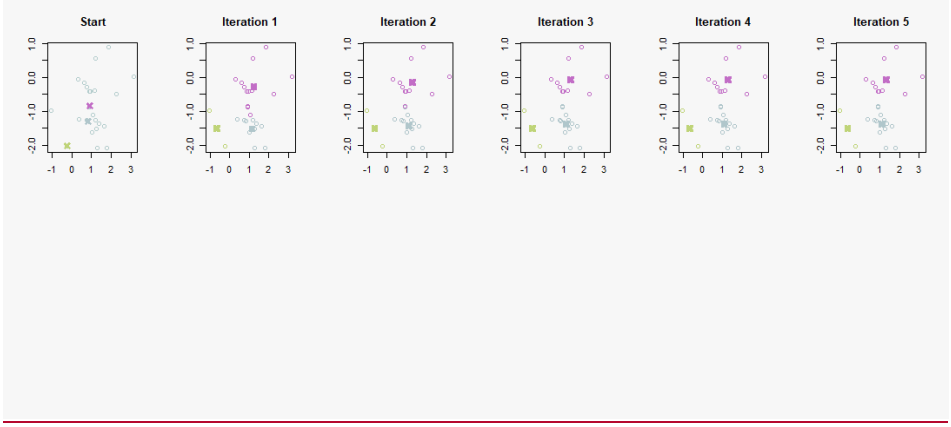
```
stichprobe3 <- data.frame(xvalue = rnorm(25, mean = 1, sd = 1),yvalue= rn
orm(25, mean = 1, sd =1))
KMEANS(stichprobe3, 3, TRUE, 10, 0.001)
```




```
## $iter
## [1] 5
##
## $zentren
```

```
##      xvalue      yvalue
## 1 1.6172517 0.850626439
## 2 0.6932496 0.009249176
## 3 0.4342780 1.593068946
##
## $index
##  [1] 2 3 1 3 1 1 3 3 3 1 2 3 3 1 2 1 2 1 3 3 3 1 3 1 2
##
## $distanz
##  [1] 0.4901946 0.2186198 0.6960981 0.8833048 0.7407509 0.3626538 0.145
9259
##  [8] 1.0707906 0.2717732 0.5532606 1.7515732 0.5352602 0.2715705 1.299
5666
## [15] 0.4828634 0.3468060 0.5496550 0.6032705 1.0817114 0.4441086 1.098
3065
## [22] 0.4117444 0.6247161 0.5115505 0.6629918
```

```
stichprobe4 <- data.frame(xvalue = rnorm(25, mean = 1, sd = 1),yvalue= rn
orm(25, mean = -1, sd =1))
KMEANS(stichprobe4,3, TRUE, 10, 0.001)
```

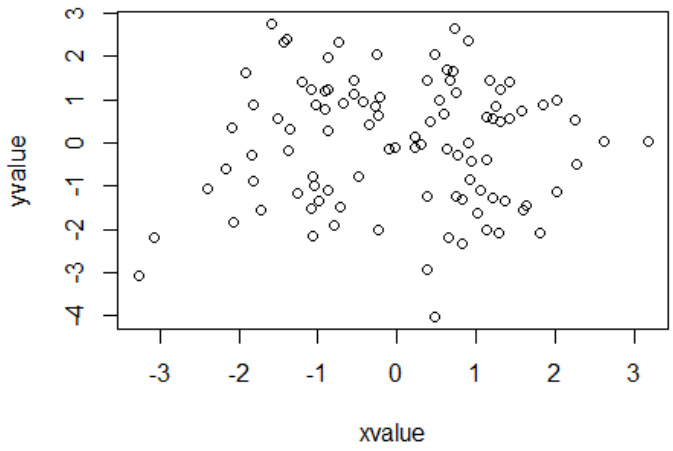



```
## $iter
## [1] 4
##
## $zentren
##       xvalue      yvalue
## 1  1.3626345 -0.03174447
## 2  1.1466300 -1.47178858
## 3 -0.2684268 -2.33804054
```

```
##
## $index
##  [1] 1 1 2 2 2 2 2 2 1 2 1 2 1 2 2 1 3 2 1 1 1 3 2 2 3
##
## $distanz
##  [1] 0.4136941 0.6584273 0.3885924 0.3689134 0.8927474 0.2073329 0.812
3086
##  [8] 0.4562073 1.0531546 0.2132755 1.0456472 0.4872402 0.6283272 0.676
8419
## [15] 0.6300724 0.5720887 1.5699424 0.2409940 0.7442722 1.0134946 1.806
6564
## [22] 0.3178601 0.4522649 0.5412400 1.8401994
```

Alle Stichproben in einem Data Frame erfassen:

```
Random_data <- rbind(stichprobe1,stichprobe2, stichprobe3, stichprobe4)
plot(Random_data)
```

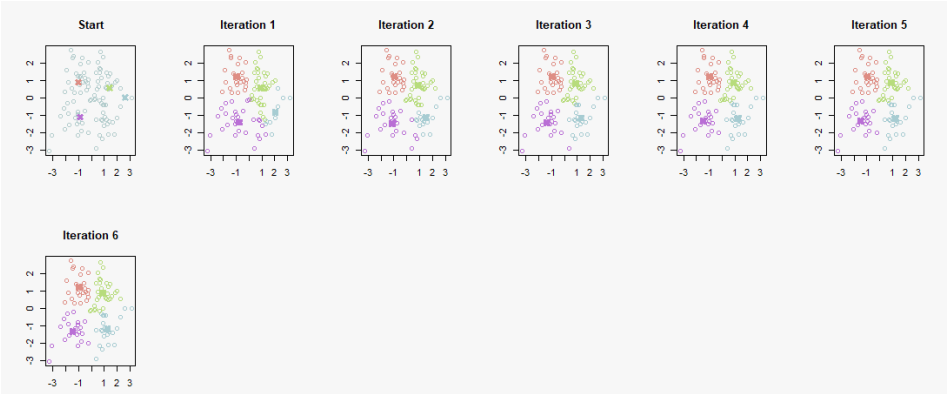



```
KMEANS(Random_data,4,TRUE,10,0.001)
```

```
## $iter
## [1] 10
##
```

```
## $zentren
##       xvalue     yvalue
## 1 -1.4676955 -1.3412266
## 2  0.9423250  0.8189418
## 3  1.2171138 -1.5007759
## 4 -0.9678602  1.2322188
##
## $index
##   [1] 4 4 4 2 4 4 4 4 4 4 4 4 4 4 4 4 4 1 4 2 4 4 4 4 1 2 1 2 1 3 3 1
1 1 1
##  [36] 2 1 1 4 1 1 1 4 1 1 3 1 1 4 1 2 2 2 2 2 2 2 4 2 2 3 2 2 2 2 2 2
2 4 2
##  [71] 2 2 2 2 2 3 2 3 3 3 3 3 3 2 3 2 3 2 3 3 3 1 3 2 3 3 1 3 3 3
##
## $distanz
##   [1] 0.85750118 0.95761232 0.11106417 1.44717118 0.08612269 0.4319960
0
##   [7] 1.64408449 0.77631824 0.92361540 1.00208757 0.45767075 0.0809002
4
##  [13] 0.29818419 1.09317659 0.74360195 0.33945757 1.24436370 1.1296762
4
##  [19] 1.02344280 0.58167994 1.18711517 0.94367260 1.11087989 0.7648418
7
##  [25] 0.57806679 0.87421941 2.50182981 1.34315907 0.97561197 1.0508496
8
##  [31] 1.77475117 1.11443030 0.27553433 0.68556114 1.16198764 0.7065625
7
##  [37] 0.64298301 0.32679888 1.00664932 0.86745948 0.90810040 0.4201507
6
##  [43] 0.46825064 0.77288343 1.02580866 0.99794271 1.79880029 0.4805169
5
##  [49] 1.41739212 0.74923601 0.99251486 0.84788670 1.39032257 1.4655260
0
##  [55] 0.62641372 0.36344033 0.76518356 0.60359417 0.61554632 0.3745262
8
##  [61] 0.80205475 0.31169773 0.80153685 1.92724333 1.19864337 0.6202170
0
##  [67] 0.62146988 0.74864937 0.43382714 1.22330394 1.77922745 1.1191638
4
##  [73] 0.37406664 0.54707677 0.38042247 1.01413964 1.18216872 0.3539361
9
##  [79] 0.44313517 0.88211082 0.34459717 0.89931823 0.52665743 1.0966544
2
##  [85] 0.13250163 0.95374891 0.37828611 0.43544411 0.64967233 0.6991326
2
##  [91] 1.01294435 0.56373722 0.10657351 1.06365359 1.35468468 2.3732014
```

```
7
##  [97] 1.40380208 0.37655130 0.63926455 2.74675911
```



Optionale Mehrleisung.

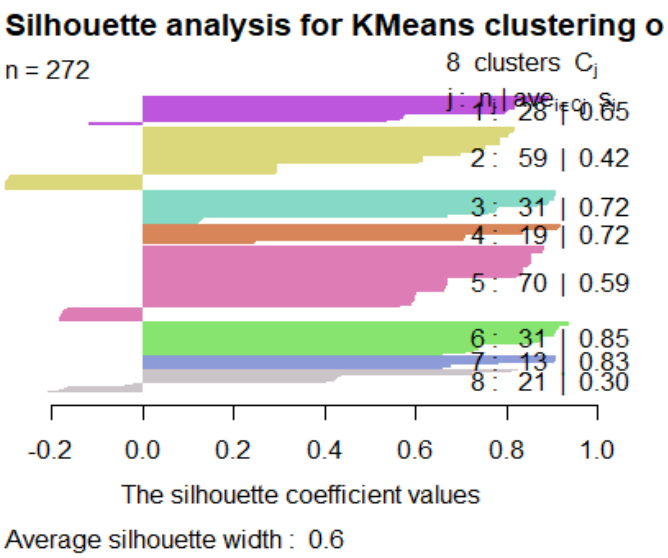
- Ermittlung der Silhouetten-Werte für jede Beobachtung und der Silhouettenkoeffizienten für jeden Cluster
- Implementierung eines Silhouetten-Plots

TEST-CODE FUER OPTIONALE ZUSATZLEISTUNG

```
#install.packages(('cluster'))
library(cluster)
daten <- faithful
```

Silhouettenplot und -koeffizient

```
anz <- 8
palette <- distinctColorPalette(anz)
dis = dist(daten)^2
anz <- 8
res = KMEANS(daten, k = anz)
sil = silhouette (res$index, dis)
windows()
plot(sil, col = palette, xlab= "The silhouette coefficient values",main =
paste("Silhouette analysis for KMeans clustering on sample data with n_cl
uster =", max(res$index)))
```



FUNCTION silhouetten:

Diese Funktion berechnet die Silhouettenwerte & Silhouettenkoeffizienten von einem gruppierten Datensatz. Die Silhouettenwerte werden mithilfe eines Silhouettenplots dargestellt.

Silhouettenwerte werden folgenderweise berechnet:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$, wobei

a ... durchschnittliche distanz zwischen jedem punkt und allen restlichen punkten im selben cluster

b ... minimale mittlere distanz von jedem punkt i zu allen anderen punkten in einem anderen cluster, in dem i nicht liegt.

input-parameter:

x … dataframe (dimension nx2, d.h. n rows, 2 cols)

erg … list mit Ergebnissen vom KMEANS-Algorithmus

output (list):

Werte … vektor der länge n mit Silhouettenwerten

Summary … summary von Silhouettenwerten

Koeffizienten … Silhouettenkoeffizienten von Clusters

Dazu werden auch folgende Plots ausgegeben:

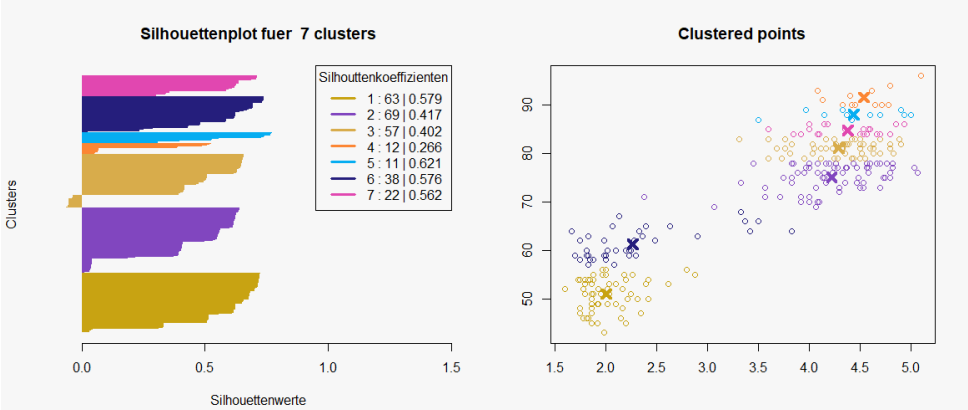
- barplot von Silhouettenwerten
- plot von geclusterten Punkten

```
silhouetten <- function(x = daten, erg = ergebnis) {
  # wieviele clusters
  k <- length(unique(erg$index))
  # Initialisiere die Silhouettenwerte: Vektor lauter 0
  silhouetten_werte <- numeric(nrow(x))

  for (i in 1:nrow(x)) {
    # mittlere Distanz von einem Punkt bis zu anderen in demselben Cluste
r
    bool <- erg$index == erg$index[i]
    pkt <- cbind(rep(x[i, 1], times = nrow(x)), rep(x[i, 2], times = nrow
(x)))

    dist_within <- sqrt(rowSums((pkt[bool,] - x[bool, ])^2))
    a <- mean(dist_within[dist_within != 0])

    # kleinste mittlere Distanz von einem Punkt bis zu anderen in restlic
hen Clustern
    dist_nextgroup <- numeric(k)
    vek <- 1:k
    vek <- vek[vek != erg$index[i]] # alle Clustern ausser eigener
    for (j in vek) {
      bool <- erg$index == j
      dist_nextgroup[j] <- mean(sqrt(rowSums((pkt[bool, ] - x[bool, ])^2)
))
    }
```

```
    b <- min(dist_nextgroup[dist_nextgroup != 0])

    silhouetten_werte[i] <- (b - a)/max(a, b)
  }

  # Dataframe fuer Plot und Berechnung von Koeffizienten
  yy <- cbind(erg$index, silhouetten_werte)
  yy <- yy[order(yy[, 1], yy[, 2]),] # nach Cluster und Wert sortieren
  yy <- as.data.frame(yy)

  # Silhouettenkoeffizienten
  n_in_cluster <- tapply(yy[, 2], yy[, 1], length)
  sil_koef <- tapply(yy[, 2], yy[, 1], mean)

  # Vorbereitung fuer Plot
  #x11(160, 70)
  par(mfrow = c(1, 2))
  par(bg = "gray97")
  farben <- vector(mode = "character", length = k)
  vek <- c(LETTERS[1:6], 0:9)
  for (i in 1:k) {
    farben[i] <- paste0("#", paste0(sample(vek, size = 6), collapse = "")
)
  }
  # Silhouettenplot
  barplot(yy[, 2],  space = 0, horiz = TRUE, col = farben[yy[, 1]], borde
r = NA,
          main = paste("Silhouettenplot fuer ", k, "clusters"),
          xlab = "Silhouettenwerte", ylab = "Clusters",
          xlim = c(min(yy[, 2]), 1.5))
  legend("topright", legend = paste(1:k, ":", n_in_cluster, "|", round(si
l_koef, digits = 3)),
         col = farben[1:k], lwd = 3, title = "Silhouettenkoeffizienten")

  # Geclusterter Datensatz
  plot(x, xlab = '', ylab = '', main = "Clustered points", col = farben[e
rg$index])
  points(erg$zentren, col = farben[1:k], pch = 4, lwd = 4, cex = 1.5)

  return(list(Werte = sihlouetten_werte, Summary = summary(silhouetten_we
rte),
              Koeffizienten = sil_koef))
}
```

BEISPIEL 1:

```
ergebnis <- KMEANS(daten, 7, FALSE, 10, 0.0001)
ergebnis$iter

## [1] 10

silhouetten(x = daten, erg = ergebnis)
```




```
## $Werte
## [1]  0.39320040  0.51601693  0.61469266  0.74001584  0.71287640  0.31165585
0.76895937
## [8]  0.62723217  0.71576520  0.71334242  0.50858758  0.59750486  0.04141607
0.67732083
## [15] -0.05256505  0.68194338  0.73458377  0.60913254  0.67881976  0.41812369
0.71466112
## [22]  0.67732083  0.01209863  0.18533448  0.63728672 -0.05287924  0.33242911
0.55969195
## [29]  0.03065587  0.41724900  0.62828868  0.40519267  0.43567881  0.64784927
0.63394814
## [36]  0.68324756  0.70399211  0.63073054  0.62220780  0.05380915  0.65419530
0.48151580
## [43]  0.63653722  0.47885098  0.62649180 -0.03829039  0.63983806  0.62226837
0.49422969
## [50]  0.62427001  0.61443484  0.06390138  0.50858758  0.63073054  0.51452222
-0.06293721
## [57]  0.54783419  0.67291469  0.40283647  0.65882890  0.62218051  0.63819957
0.70338127
## [64]  0.47624470  0.69181732  0.50792133  0.04087808  0.03854404  0.58586315
0.62247861
## [71]  0.49447799  0.02968031  0.41539065  0.55215370  0.73856589  0.53190018
0.69592274
## [78]  0.04142380  0.55214438 -0.05287924  0.62444977  0.50956882  0.41158806
0.58355767
```

```
## [85]  0.62715508  0.74140074  0.55520130  0.65238200  0.70327454  0.34101228
0.69661901
## [92]  0.05158673  0.71993565  0.03368011  0.73336786  0.59891843  0.62847847
0.61361978
## [99]  0.71545730  0.46317326  0.73859486  0.76555952  0.72004145 -0.05673795
0.65057166
## [106]  0.68287078  0.62458345  0.68210066  0.34183813  0.61772796  0.6172200
7  0.62071042
##[113]  0.52852041  0.41752727  0.61796557  0.64531907  0.72292071  0.70869607
0.62182997
## [120]  0.39700600  0.61001258  0.21866792  0.40477209  0.02968031  0.7733628
9  0.62674867
## [127]  0.62351871  0.50450426  0.33011147  0.06848864  0.62344315 -0.0360145
2  0.03740258
## [134]  0.55604183  0.65088544  0.50929627  0.71557517  0.30115137  0.6227178
9  0.40085597
## [141]  0.65244198  0.69632035  0.50244749  0.38946918  0.56189920  0.6242172
6  0.64873597
## [148]  0.72051245  0.38753160  0.61541698  0.37388040  0.39576469  0.5860469
0  0.64770011
## [155]  0.54424228  0.41015734  0.65328382  0.51740365  0.61541698  0.5260283
6  0.62276044
## [162]  0.35550804  0.48162103  0.02990839  0.43095038  0.55705266  0.7377527
7  0.72879950
## [169]  0.68328100  0.52608757  0.72053652  0.23395865  0.39337112  0.0231503
8  0.65674795
## [176]  0.65862709  0.62149380  0.72122916  0.68789634  0.63968784  0.3316054
9  0.39337112
## [183] -0.05232715 -0.03686573  0.71552834  0.04216224  0.61919629  0.6508854
4 -0.03675196
## [190]  0.33141458  0.63006273  0.25266441  0.54816260  0.62076182  0.3940158
9  0.65244198
## [197]  0.30894885  0.40622766  0.71311732  0.03953196  0.69667114  0.5097199
6  0.39026560
## [204]  0.62235963  0.04098323  0.65626836  0.40622750  0.58616899  0.7205853
8 -0.05673795
## [211]  0.49871081  0.64559969  0.72021027  0.61693474  0.65649731  0.5620297
9  0.61684451
## [218]  0.48765281  0.33251889  0.56113601  0.71993565  0.50781605  0.5149718
1  0.62345537
## [225]  0.03625283  0.41579417  0.03884996  0.04196522  0.40850327  0.4134596
2  0.41140482
## [232]  0.50767445  0.35706484  0.72403278  0.06787562  0.51660818  0.5166051
8  0.40543882
## [239]  0.41061443  0.67786130  0.62464368  0.68027511  0.30115137  0.7321892
5  0.71021486
## [246]  0.47596021  0.23395865  0.50962195  0.28792145  0.63982272  0.5131561
4 -0.05446609
## [253]  0.61684386  0.62149380  0.74007739  0.63659735  0.55125260 -0.0544660
9  0.04523385
```

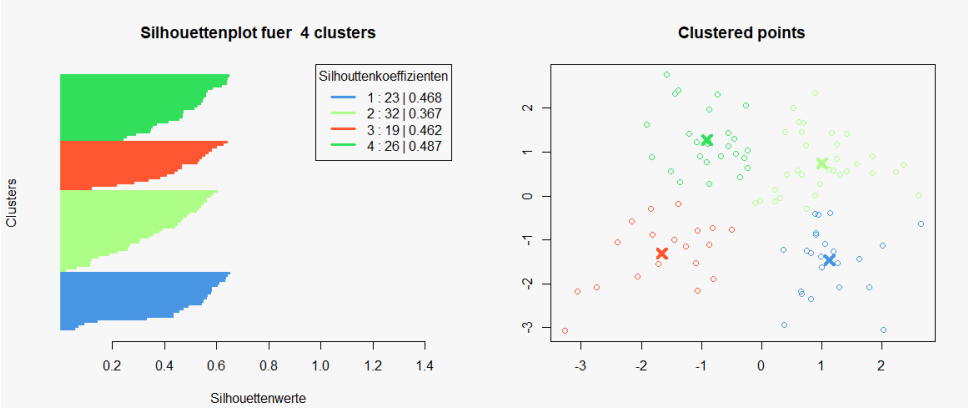
```
## [260]  0.41831732  0.03598794  0.63798419  0.48111896 -0.05232715  0.5576796
6  0.69604870
## [267]  0.61659457  0.65454204  0.65538629  0.06527323  0.65646409  0.6385634
2
##
## $Summary
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.06294  0.39670  0.56196  0.48702  0.65089  0.77336
##
## $Koeffizienten
##         1         2         3         4         5         6         7
## 0.5791851 0.4168477 0.4023371 0.2656498 0.6213368 0.5764689 0.5616605
```

BEISPIEL 2:

```
ergebnis <- KMEANS(Random_data, 4, FALSE, 15, 0.001)
ergebnis$iter
```

```
## [1] 10
```

```
silhouetten(x = Random_data, erg = ergebnis)
```




```
## $Werte
## [1] 0.41612889 0.37293329 0.65166710 0.06431753 0.64013309 0.56549757 0.4719
9120 0.35612289
## [9] 0.45853937 0.29219699 0.58498968 0.64176973 0.64521058 0.34889316 0.5591
9052 0.62015865
## [17] 0.52948552 0.21768431 0.56134996 0.54003074 0.54215437 0.24337478 0.473
08972 0.34776701
## [25] 0.55886182 0.58388465 0.57077866 0.38406672 0.30618848 0.45308862 0.121
74961 0.54377634
```

```
## [33] 0.56570090 0.33334388 0.28434836 0.60262221 0.46818693 0.12212120 0.562
68041 0.46693148
## [41] 0.64381477 0.25758105 0.33953005 0.40978514 0.52922610 0.54703961 0.626
72489 0.43893536
## [49] 0.54939373 0.53371702 0.48709177 0.39431448 0.11698036 0.46300852 0.514
70733 0.45324899
## [57] 0.26030942 0.26232762 0.37988329 0.19009306 0.49776699 0.60540573 0.148
16152 0.47192883
## [65] 0.34237263 0.58773081 0.43591326 0.44166075 0.30419496 0.11667416 0.221
15200 0.55178993
## [73] 0.35764636 0.52455267 0.54979398 0.49399782 0.14393034 0.63332006 0.470
41641 0.56313310
## [81] 0.54473918 0.07136504 0.45780982 0.64444917 0.05786061 0.02376838 0.564
47355 0.60420773
## [89] 0.58352405 0.65469210 0.47406884 0.58127872 0.26933611 0.61052382 0.534
18055 0.58208939
## [97] 0.58030655 0.43669912 0.63788800 0.09446339
##
## $Summary
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02377 0.34166 0.47254 0.43940 0.56473 0.65469
##
## $Koeffizienten
##         1         2         3         4
##  0.4676405 0.3669523 0.4620045 0.4870652
```