INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

Machine Learning & Data Mining Coursework

Module code: 5DATA002W

Module leader: Dr. V.S. Kontogiannis

Assignment: Coursework

Handed Out: 20/02/2023.

Due Date: 04/05/2022, Submission by 13:00

Name: B.K. Praveen Navanjana

IIT ID: 20200877

UOW ID: w1870590

Table of Contents

Partitioning Clustering Part

Dataset loading and checking the dataset

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
> library(readxl)
> library(dplyr)
> library(ggplot2)
> library(NbClust)
> library(factoextra)
> library(cluster)
> datasheet <- read_excel("C:/Users/my pc/Desktop/ML cw/vehicles.xlsx") # Add dataset path
> View(datasheet)
> View(datasheet)
> str(datasheet)
tibble [846 × 20] (S3: tbl_df/tbl/data.frame)
 $ Samples     : num [1:846] 1 2 3 4 5 6 7 8 9 10 ...
 $ Comp        : num [1:846] 95 91 104 93 85 107 97 90 86 93 ...
 $ Circ        : num [1:846] 48 41 50 41 44 57 43 43 34 44 ...
 $ D.Circ      : num [1:846] 83 84 106 82 70 106 73 66 62 98 ...
 $ Rad.Ra      : num [1:846] 178 141 209 159 205 172 173 157 140 197 ...
 $ Pr.Axis.Ra  : num [1:846] 72 57 66 63 103 50 65 65 61 62 ...
 $ Max.L.Ra    : num [1:846] 10 9 10 9 52 6 6 9 7 11 ...
 $ Scat.Ra     : num [1:846] 162 149 207 144 149 255 153 137 122 183 ...
 $ Elong       : num [1:846] 42 45 32 46 45 26 42 48 54 36 ...
 $ Pr.Axis.Rect: num [1:846] 20 19 23 19 19 28 19 18 17 22 ...
 $ Max.L.Rect  : num [1:846] 159 143 158 143 144 169 143 146 127 146 ...
 $ Sc.Var.Maxis: num [1:846] 176 170 223 160 241 280 176 162 141 202 ...
 $ Sc.Var.maxis: num [1:846] 379 330 635 309 325 957 361 281 223 505 ...
 $ Ra.Gyr      : num [1:846] 184 158 220 127 188 264 172 164 112 152 ...
 $ Skew.Maxis  : num [1:846] 70 72 73 63 127 85 66 67 64 64 ...
 $ Skew.maxis  : num [1:846] 6 9 14 6 9 5 13 3 2 4 ...
 $ Kurt.maxis  : num [1:846] 16 14 9 10 11 9 1 3 14 14 ...
 $ Kurt.Maxis  : num [1:846] 187 189 188 199 180 181 200 193 200 195 ...
 $ Holl.Ra     : num [1:846] 197 199 196 207 183 183 204 202 208 204 ...
 $ Class       : chr [1:846] "van" "van" "saab" "van" ...
>
```

First 18 columns

```
> data_c
# A tibble: 846 × 18
    Comp  Circ D.Circ Rad.Ra Pr.Axis.Ra Max.L.Ra Scat.Ra Elong Pr.Axis.Rect Max.L.Rect
   <dbl> <dbl>  <dbl>  <dbl>      <dbl>    <dbl>   <dbl> <dbl>        <dbl>      <dbl>
 1    95    48     83    178         72       10     162    42           20        159
 2    91    41     84    141         57        9     149    45           19        143
 3   104    50    106    209         66       10     207    32           23        158
 4    93    41     82    159         63        9     144    46           19        143
 5    85    44     70    205        103       52     149    45           19        144
 6   107    57    106    172         50        6     255    26           28        169
 7    97    43     73    173         65        6     153    42           19        143
 8    90    43     66    157         65        9     137    48           18        146
 9    86    34     62    140         61        7     122    54           17        127
10    93    44     98    197         62       11     183    36           22        146
# ℹ 836 more rows
# ℹ 8 more variables: Sc.Var.Maxis <dbl>, Sc.Var.maxis <dbl>, Ra.Gyr <dbl>, Skew.Maxis <dbl>,
#   Skew.maxis <dbl>, Kurt.maxis <dbl>, Kurt.Maxis <dbl>, Holl.Ra <dbl>
# ℹ Use `print(n = ...)` to see more rows
> |
```

Preprocessing tasks

Scaling

Scaling is a crucial component of machine learning because it ensures that all variables are handled similarly during the learning process, regardless of their original values or measurement units. We can improve the performance and speed of some machine learning algorithms and produce models that are more accurate and effective by scaling the features.

```
> Data_N_outliers <- data_c
> md <- mahalanobis(data_c, colMeans(data_c), cov(data_c))
> outliers <- which(md > qchisq(0.95, df = ncol(data_c)))
> Data_N_outliers <- Data_N_outliers[-outliers, ]
> D_Scaled <- scale(data_c)
> D_Scaled
             Comp        Circ       D.Circ      Rad.Ra  Pr.Axis.Ra   Max.L.Ra      Scat.Ra
 [1,]  0.16048541  0.50864931  0.057784334  0.270645678  1.30651856  0.31135767 -0.205722605
 [2,] -0.32527723 -0.62589728  0.121189713 -0.834749808 -0.59504361  0.09402385 -0.596759110
 [3,]  1.25345137  0.83280548  1.516108039  1.196787842  0.54589369  0.31135767  1.147865294
 [4,] -0.08239591 -0.62589728 -0.005621044 -0.296989842  0.16558126  0.09402385 -0.747157765
 [5,] -1.05392120 -0.13966303 -0.766485586  1.077285627  5.23641371  9.43937817 -0.596759110
 [6,]  1.61777335  1.96735207  1.516108039  0.091392356 -1.48243929 -0.55797761  2.591692388
 [7,]  0.40336674 -0.30174112 -0.576269450  0.121267909  0.41912288 -0.55797761 -0.476440185
 [8,] -0.44671789 -0.30174112 -1.020107099 -0.356740949  0.41912288  0.09402385 -0.957715883
 [9,] -0.93248054 -1.76044388 -1.273728613 -0.864625362 -0.08796036 -0.34064379 -1.408911850
[10,] -0.08239591 -0.13966303  1.008865011  0.838281197  0.03881045  0.52869149  0.425951748
[11,] -0.93248054 -1.43628771 -0.766485586 -0.774998701 -0.08796036  0.09402385 -1.078034808
[12,] -0.44671789 -1.76044388 -1.020107099 -0.984127577 -0.84858523 -0.55797761 -1.378832119
[13,] -0.68959922  0.18449314 -0.512864072  0.061516802  0.79943532 -0.55797761 -0.506519916
[14,] -0.56815856 -0.46381920  0.184595091 -0.745123147 -0.46827280  0.31135767 -0.506519916
[15,]  0.03904475  0.67072739 -0.195837180  1.017534519  1.17974775 -0.77531143  0.155234168
[16,]  0.28192607  1.64319590  1.325891903  0.957783412  0.41912288  0.09402385  1.057626101
[17,] -0.56815856 -1.43628771 -1.971187776 -1.790767526 -1.22889766 -0.55797761 -1.529230774
[18,]  0.64624806 -0.62589728 -0.322647936  0.838281197  0.92620613 -0.55797761  0.245473361
[19,]  1.25345137  1.48111782  1.135675768  0.509650107 -0.08796036  0.31135767  1.418582874
[20,]  0.88912938  1.80527399  1.135675768  1.376041164  0.92620613  0.31135767  1.177945026
[21,] -1.17536186  0.34657122 -0.449458693 -0.476243164  0.29235207 -0.55797761 -0.446360454
[22,] -1.17536186 -1.27420962 -1.844377019 -1.432260882 -0.34150198 -0.77531143 -1.378832119
[23,]  0.03904475 -0.30174112 -1.146917856  0.121267909  0.92620613 -0.34064379 -0.566679379
[24,] -0.81103988 -0.95005345 -0.766485586 -0.625620932 -0.08796036 -0.34064379 -0.777237496
[25,]  0.64624806  1.31903973  1.452702660  1.495543378  0.54589369  0.52869149  1.057626101
[26,] -1.05392120  0.02241505 -0.132431801 -0.446367610  0.29235207  0.09402385 -0.656918572
[27,] -1.29680252 -1.43628771 -1.780971641 -1.492011989 -0.59504361 -0.55797761 -1.228433463
[28,]  1.61777335  1.48111782  1.008865011  1.017534519  0.41912288  0.52869149  1.478742337
[29,]  1.01057004  0.02241505  0.184595091  0.718778983  0.29235207 -0.55797761  0.696669328
[30,] -1.66112451 -1.11213154 -1.210323235 -1.193256452 -0.84858523 -0.34064379 -0.686998303
[31,] -0.56815856 -0.30174112  0.184595091 -0.267114288  0.29235207  0.52869149 -0.416280723
[32,] -0.68959922 -0.46381920 -0.322647936 -0.535994271 -0.46827280 -0.12330997 -0.867476690
[33,] -0.08239591 -1.59836579 -1.020107099 -0.446367610 -0.34150198 -0.55797761 -0.807317228
[34,]  0.88912938  0.50864931  1.579513417  1.585170039  0.79943532  0.31135767  1.177945026
```

```
R 4.3.0 · ~/
[37,] -1.41824319 -0.13966303 -0.639674829 -1.521887543 -1.22889766 -0.34064379 -0.506519916
[38,] -0.44671789  0.50864931  0.248000470  4.094716548  8.15214237  8.78737671 -0.476440185
[39,]  1.49633269  1.31903973  1.008865011  0.210894570 -0.97535604  0.31135767  1.418582874
[40,] -1.53968385  0.02241505 -0.893296343  0.001765695  1.43328937 -0.55797761 -0.536599648
[41,]  0.16048541  0.50864931  1.389297282  1.346165610  0.67266451  0.09402385  1.087705832
[42,] -0.68959922 -1.27420962 -1.971187776 -1.910269740 -1.22889766 -0.77531143 -1.499151043
[43,]  0.03904475  0.67072739  0.311405848 -0.954252023 -0.97535604  0.52869149 -0.326041530
[44,] -0.08239591 -1.27420962 -0.386053315  0.420023446  0.16558126 -0.12330997 -0.145563143
[45,]  3.07506129  1.48111782  1.516108039  1.525418932  0.41912288  0.74602532  1.328343681
[46,] -0.08239591  0.18449314 -0.005621044 -0.715247593 -0.46827280  0.52869149 -0.295961799
[47,] -0.32527723 -0.30174112 -0.766485586 -1.073754238 -0.84858523 -0.12330997 -1.168274001
[48,] -1.05392120 -0.46381920 -1.020107099 -1.402385328 -0.97535604 -0.55797761 -0.626838841
[49,] -0.56815856  0.34657122 -0.069026423 -0.655496486  0.29235207  0.52869149 -0.386200992
[50,] -0.32527723  0.02241505 -0.195837180  0.210894570 -0.34150198  0.09402385 -0.175642874
[51,] -1.90400583 -1.11213154 -1.210323235 -1.611514204 -1.35566847 -0.55797761 -0.807317228
[52,] -0.20383657 -1.11213154 -0.703080207  0.151143463  0.54589369 -0.34064379 -0.446360454
[53,]  0.52480740  1.64319590  1.199081146  1.764423361  1.05297694  0.09402385  1.238104488
[54,]  0.88912938 -0.46381920 -1.273728613  0.181019017  0.67266451 -0.55797761 -0.596759110
[55,]  0.88912938  1.80527399  1.389297282  0.479774553 -1.10212685 -0.55797761  2.651851850
            Elong Pr.Axis.Rect    Max.L.Rect Sc.Var.Maxis  Sc.Var.maxis       Ra.Gyr
 [1,]  0.136489241   -0.2248114  7.578841e-01  -0.40214560 -0.3447305820  0.285643409
 [2,]  0.520535463   -0.6105933 -3.443743e-01  -0.59325983 -0.6220483430 -0.513213860
 [3,] -1.143664834    0.9325342  6.889930e-01   1.09491586  1.1041132308  1.391753474
 [4,]  0.648550870   -0.6105933 -3.443743e-01  -0.91178355 -0.7408988120 -1.465697528
 [5,]  0.520535463   -0.6105933 -2.754832e-01   1.66825855 -0.6503460737  0.408544527
 [6,] -1.911757278    2.8614436  1.446796e+00   2.91050104  2.9264870891  2.743665776
 [7,]  0.136489241   -0.6105933 -3.443743e-01  -0.40214560 -0.4466024125 -0.083059946
 [8,]  0.904581685   -0.9963752 -1.377009e-01  -0.84807881 -0.8993661040 -0.328862183
 [9,]  1.672674129   -1.3821571 -1.446633e+00  -1.51697861 -1.2276197804 -1.926576722
[10,] -0.631603204    0.5467523 -1.377009e-01   0.42601606  0.3683722322 -0.697565538
[11,]  1.160612500   -0.9963752 -1.239959e+00  -1.13475015 -0.9842592962 -1.465697528
[12,]  1.672674129   -1.3821571 -2.066653e+00  -1.29401201 -1.2219602342 -1.742225044
[13,]  0.264504648   -0.6105933  8.143163e-05  -0.27473612 -0.5145169663  0.531445646
[14,]  0.392520055   -0.6105933 -2.754832e-01  -0.49770272 -0.5371551508 -0.421038022
[15,] -0.503587796    0.1609705  4.134284e-01   0.23490183  0.1419903864  0.961599560
[16,] -1.143664834    0.9325342  1.240122e+00   1.22232535  1.0418582233  2.190610744
[17,]  2.056720352   -1.3821571 -1.308851e+00  -1.64438810 -1.3238320648 -1.527148087
[18,] -0.631603204    0.1609705 -6.199390e-01   0.42601606  0.2551813093 -0.728290818
[19,] -1.271680241    1.3183161  1.722360e+00   1.15862060  1.3927500842  1.391753474
[20,] -1.143664834    1.3183161  1.446796e+00   1.22232535  1.1946659691  1.483929313
[21,]  0.264504648   -0.6105933 -2.065920e-01  -0.43399798 -0.4862192355  0.285643409
[22,]  1.800689537   -1.3821571 -1.584415e+00  -1.51697861 -1.2389388727 -1.281345850
```

```
R 4.3.0 · ~/
[25,] -1.143664834    0.9325342  1.171231e+00   1.03121112  1.0361986771  1.514654593
[26,]  0.520535463   -0.6105933  8.143163e-05  -0.62511221 -0.6560056199 -0.021609387
[27,]  1.544658722   -0.9963752 -1.584415e+00  -1.45327387 -1.1427265882 -1.096994173
[28,] -1.271680241    1.7040980  1.309013e+00   1.28603009  1.4493455456  1.268852356
[29,] -1.015649426    0.5467523 -1.377009e-01   0.90380163  0.7362427315 -0.359587462
[30,]  0.648550870   -0.6105933 -1.239959e+00  -0.65696458 -0.7126010813 -0.513213860
[31,]  0.264504648   -0.6105933  2.067549e-01  -0.49770272 -0.4749001433 -0.021609387
[32,]  0.776566278   -0.9963752 -4.132655e-01  -0.75252169 -0.8314515503 -0.513213860
[33,]  0.648550870   -0.9963752 -1.377742e+00  -0.84807881 -0.7691965427 -1.680774485
[34,] -1.143664834    1.3183161  4.134284e-01   1.38158721  1.1380705077  0.900149001
[35,] -0.119541574   -0.2248114 -1.308851e+00  -0.08362189 -0.2145610206 -1.373521689
[36,]  0.264504648   -0.2248114  1.378637e-01  -0.59325983 -0.4352833203  0.285643409
[37,]  0.392520055   -0.6105933 -6.880972e-02  -0.46585035 -0.5654528816  0.070566452
[38,]  0.392520055   -0.6105933  5.512107e-01   2.65568207 -0.5314956047  0.777247882
[39,] -1.271680241    1.3183161  1.584578e+00   1.47714432  1.4210478149  1.330302915
[40,]  0.392520055   -0.6105933 -1.377009e-01  -0.49770272 -0.5880910661  0.347093968
[41,] -1.143664834    0.9325342  2.067549e-01   1.22232535  1.0644964078  0.838698442
[42,]  2.056720352   -1.3821571 -1.377742e+00  -1.70809284 -1.3181725187 -1.527148087
[43,]  0.264504648   -0.2248114  9.645576e-01  -0.33844086 -0.4183046818  0.347093968
[44,] -0.119541574   -0.2248114 -9.643947e-01   0.07563997 -0.1975823822 -1.096994173
[45,] -1.271680241    1.3183161  1.309013e+00   1.09491586  1.3304950766  1.760456829
[46,]  0.264504648   -0.2248114  8.267753e-01  -0.27473612 -0.3900069511  0.439269807
[47,]  1.288627907   -0.9963752 -1.377009e-01  -0.94363592 -1.0578333961 -0.574664420
[48,]  0.648550870   -0.6105933 -4.821567e-01  -0.52955509 -0.6956224429 -0.021609387
[49,]  0.392520055   -0.2248114  1.033449e+00  -0.59325983 -0.4975383278  0.408544527
[50,] -0.119541574   -0.2248114  8.143163e-05  -0.14732663 -0.2032419284  0.132017011
[51,]  0.776566278   -0.6105933 -1.239959e+00  -0.84807881 -0.7974942735 -0.881917215
[52,]  0.264504648   -0.6105933 -1.033286e+00  -0.24288375 -0.4805596894 -1.373521689
[53,] -1.271680241    1.3183161  1.377905e+00   1.50899669  1.2512614306  2.159885464
[54,]  0.264504648   -0.6105933 -6.199390e-01  -0.62511221 -0.5597933354 -0.298136903
[55,] -1.911757278    2.8614436  1.377905e+00   2.75123918  2.9208275430  1.699006270
       Skew.Maxis  Skew.maxis  Kurt.maxis  Kurt.Maxis     Holl.Ra
 [1,] -0.32886116 -0.07666562  0.38076562 -0.31353666  0.18384857
 [2,] -0.06173054  0.53329468  0.15683255  0.01093064  0.45270923
 [3,]  0.07183477  1.54989517 -0.40300011 -0.15130301  0.04941824
 [4,] -1.26381831 -0.07666562 -0.29103357  1.63326713  1.52815188
 [5,]  7.28436143  0.53329468 -0.17906704 -1.44917221 -1.69817606
 [6,]  1.67461847 -0.27998571 -0.40300011 -1.28693856 -1.69817606
 [7,] -0.86312239  1.34657507 -1.29873236  1.79550078  1.12486089
 [8,] -0.72955708 -0.68662591 -1.07479930  0.65986523  0.85600023
 [9,] -1.13025301 -0.88994601  0.15683255  1.79550078  1.66258221
```

R 4.3.0 · ~/

```
[13,] -0.19529585 -0.27998571 -0.17906704  0.01093064 -0.08501209
[14,] -0.06173054  0.32997458  0.04486602 -0.31353666  0.18384857
[15,] -0.19529585 -0.07666562 -1.18676583  1.30879983  0.45270923
[16,]  0.20540008 -0.07666562 -1.18676583 -0.47577031 -0.21944242
[17,]  1.00679193 -0.88994601  0.15683255 -1.28693856 -1.42931540
[18,] -0.06173054 -0.48330581 -0.29103357  1.47103348  0.45270923
[19,]  0.20540008 -0.27998571 -0.17906704 -0.63800396 -0.08501209
[20,]  0.20540008 -0.07666562 -0.85086623 -0.47577031 -0.35387275
[21,]  0.33896539 -1.29658621 -1.07479930 -0.63800396 -0.48830308
[22,]  1.27392255  0.12665448 -1.29873236 -1.61140586 -1.69817606
[23,] -0.59599177  0.53329468 -1.29873236  1.63326713  1.39372155
[24,] -0.46242646 -1.09326611 -1.18676583  0.49763158  0.45270923
[25,] -0.59599177 -1.29658621 -0.73889970  0.33539794  0.72156990
[26,] -0.19529585 -1.09326611 -0.96283277 -0.15130301  0.45270923
[27,]  1.27392255 -0.07666562 -1.07479930 -1.61140586 -1.69817606
[28,] -0.06173054 -1.09326611  1.72436400 -0.31353666  0.45270923
[29,]  0.47253070 -0.07666562 -0.62693317  0.98433253 -0.35387275
[30,]  1.40748785  0.53329468  0.82863175 -1.44917221 -1.42931540
[31,] -0.06173054 -0.27998571 -0.40300011 -0.63800396  0.04941824
[32,] -1.13025301  0.73661478 -0.17906704  1.47103348  1.25929122
[33,] -1.13025301 -0.27998571  0.04486602  1.30879983  0.85600023
[34,] -0.32886116 -0.27998571  2.84402932  0.17316429  0.85600023
[35,] -1.26381831 -1.09326611  1.38846441  1.47103348  1.25929122
[36,] -0.72955708  2.15985547 -0.62693317  0.49763158  0.58713957
[37,]  1.27392255 -0.88994601 -1.18676583 -1.44917221 -1.42931540
[38,]  6.08227365 -1.29658621  0.26879909 -0.63800396 -0.21944242
[39,]  0.20540008 -1.09326611 -0.40300011 -0.31353666  0.18384857
[40,]  0.33896539  0.12665448 -1.41069890 -0.96247126 -0.89159407
[41,]  0.20540008 -0.27998571 -0.40300011 -0.47577031 -0.35387275
[42,]  1.80818378  0.32997458  0.38076562 -1.61140586 -1.69817606
[43,]  0.33896539 -0.27998571 -0.85086623 -0.96247126 -0.21944242
[44,] -0.72955708 -0.48330581 -0.62693317  0.49763158  0.18384857
[45,] -0.86312239  2.76981576 -1.29873236  0.49763158  0.85600023
[46,]  0.60609600 -0.88994601 -0.96283277 -0.96247126 -0.21944242
[47,] -0.32886116 -1.09326611 -0.51496664  0.17316429 -0.21944242
[48,]  2.07531440 -0.07666562  0.15683255 -1.44917221 -1.83260639
[49,]  0.47253070 -0.07666562  0.04486602 -0.80023761 -0.35387275
[50,] -1.39738362 -1.29658621 -0.29103357  1.63326713  1.66258221
[51,]  0.60609600 -0.88994601 -0.96283277 -1.28693856 -1.42931540
[52,] -0.32886116 -0.48330581  1.27649787  0.01093064 -0.08501209
[53,] -0.06173054 -1.09326611 -0.73889970 -0.15130301  0.18384857
```

```
 [ reached getOption("max.print") -- omitted 791 rows ]
attr(,"scaled:center")
       Comp        Circ      D.Circ      Rad.Ra Pr.Axis.Ra   Max.L.Ra     Scat.Ra
  93.678487   44.861702   82.088652  168.940898  61.693853   8.567376  168.839243
      Elong Pr.Axis.Rect Max.L.Rect Sc.Var.Maxis Sc.Var.maxis  Ra.Gyr  Skew.Maxis
  40.933806   20.582742  147.998818  188.625296 439.911348 174.703310   72.462175
 Skew.maxis  Kurt.maxis  Kurt.Maxis     Holl.Ra
   6.377069   12.599291  188.932624  195.632388
attr(,"scaled:scale")
       Comp        Circ      D.Circ      Rad.Ra Pr.Axis.Ra   Max.L.Ra     Scat.Ra
   8.234474    6.169866   15.771533   33.472183   7.888251   4.601217   33.244978
      Elong Pr.Axis.Rect Max.L.Rect Sc.Var.Maxis Sc.Var.maxis  Ra.Gyr  Skew.Maxis
   7.811560    2.592138   14.515652   31.394837 176.692614  32.546490    7.486974
 Skew.maxis  Kurt.maxis  Kurt.Maxis     Holl.Ra
   4.918353    8.931240    6.163949    7.438797
>
```

Determining the number of clusters using four automated tools

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal × Background Jobs ×

R 4.3.0 · ~/

```
        }
        DB = mean(r[is.finite(r)])
        resul <- list(DB = DB, r = r, R = R, d = M, S = S, centers = centers)
        resul
    }
    Indice.S <- function(d, cl) {
        d <- as.matrix(d)
        Si <- 0
        for (k in 1:max(cl)) {
            if ((sum(cl == k)) <= 1)
                Sil <- 1
            else {
                Sil <- 0
                for (i in 1:length(cl)) {
                  if (cl[i] == k) {
                    ai <- sum(d[i, cl == k])/(sum(cl == k) -
                      1)
                    dips <- NULL
                    for (j in 1:max(cl)) if (cl[i] != j)
                      if (sum(cl == j) != 1)
                        dips <- cbind(dips, c((sum(d[i, cl ==
                          j]))/(sum(cl == j))))
                      else dips <- cbind(dips, c((sum(d[i, cl ==
                        j]))))
                    bi <- min(dips)
                    Sil <- Sil + (bi - ai)/max(c(ai, bi))
                  }
                }
            }
            Si <- Si + Sil
        }
        Si/length(cl)
    }
    Indice.Gap <- function(x, clall, reference.distribution = "unif",
        B = 10, method = "ward.D2", d = NULL, centrotypes = "centroids") {
        GAP <- function(X, cl, referenceDistribution, B, method,
            d, centrotypes) {
            set.seed(1)
            simgap <- function(Xvec) {
                ma <- max(Xvec)
                mi <- min(Xvec)
                set.seed(1)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
simgap <- function(Xvec) {
    ma <- max(Xvec)
    mi <- min(Xvec)
    set.seed(1)
    Xout <- runif(length(Xvec), min = mi, max = ma)
    return(Xout)
}
pcsim <- function(X, d, centrotypes) {
    if (centrotypes == "centroids") {
      Xmm <- apply(X, 2, mean)
    }
    for (k in (1:dim(X)[2])) {
      X[, k] <- X[, k] - Xmm[k]
    }
    ss <- svd(X)
    Xs <- X %*% ss$v
    Xnew <- apply(Xs, 2, simgap)
    Xt <- Xnew %*% t(ss$v)
    for (k in (1:dim(X)[2])) {
      Xt[, k] <- Xt[, k] + Xmm[k]
    }
    return(Xt)
}
if (is.null(dim(x))) {
    dim(x) <- c(length(x), 1)
}
ClassNr <- max(cl)
Wk0 <- 0
WkB <- matrix(0, 1, B)
for (bb in (1:B)) {
    if (reference.distribution == "unif")
      Xnew <- apply(X, 2, simgap)
    else if (reference.distribution == "pc")
      Xnew <- pcsim(X, d, centrotypes)
    else stop("Wrong reference distribution type")
    if (bb == 1) {
      pp <- cl
      if (ClassNr == length(cl))
        pp2 <- 1:ClassNr
      else if (method == "k-means") {
        set.seed(1)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
        if (ClassNr == length(cl))
          pp2 <- 1:ClassNr
        else if (method == "k-means") {
          set.seed(1)
          pp2 <- kmeans(Xnew, ClassNr, 100)$cluster
        }
        else if (method == "single" || method == "complete" ||
          method == "average" || method == "ward.D2" ||
          method == "mcquitty" || method == "median" ||
          method == "centroid" || method == "ward.D")
          pp2 <- cutree(hclust(dist(Xnew), method = method),
            ClassNr)
        else stop("Wrong clustering method")
        if (ClassNr > 1) {
          for (zz in (1:ClassNr)) {
            Xuse <- X[pp == zz, ]
            Wk0 <- Wk0 + sum(diag(var(Xuse))) * (length(pp[pp ==
              zz]) - 1)/(dim(X)[1] - ClassNr)
            Xuse2 <- Xnew[pp2 == zz, ]
            WkB[1, bb] <- WkB[1, bb] + sum(diag(var(Xuse2))) *
              (length(pp2[pp2 == zz]) - 1)/(dim(X)[1] -
              ClassNr)
          }
        }
        if (ClassNr == 1) {
          Wk0 <- sum(diag(var(X)))
          WkB[1, bb] <- sum(diag(var(Xnew)))
        }
      }
      if (bb > 1) {
        if (ClassNr == length(cl))
          pp2 <- 1:ClassNr
        else if (method == "k-means") {
          set.seed(1)
          pp2 <- kmeans(Xnew, ClassNr, 100)$cluster
        }
        else if (method == "single" || method == "complete" ||
          method == "average" || method == "ward.D2" ||
          method == "mcquitty" || method == "median" ||
          method == "centroid" || method == "ward.D")
          pp2 <- cutree(hclust(dist(Xnew), method = method),
            ClassNr)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
                   ClassNr)
                 else stop("Wrong clustering method")
                 if (ClassNr > 1) {
                   for (zz in (1:ClassNr)) {
                     Xuse2 <- Xnew[pp2 == zz, ]
                     WkB[1, bb] <- WkB[1, bb] + sum(diag(var(Xuse2))) *
                       length(pp2[pp2 == zz])/(dim(X)[1] - ClassNr)
                   }
                 }
                 if (ClassNr == 1) {
                   WkB[1, bb] <- sum(diag(var(Xnew)))
                 }
               }
           }
           Sgap <- mean(log(WkB[1, ])) - log(Wk0)
           Sdgap <- sqrt(1 + 1/B) * sqrt(var(log(WkB[1, ]))) *
               sqrt((B - 1)/B)
           resul <- list(Sgap = Sgap, Sdgap = Sdgap)
           resul
       }
       if (sum(c("centroids", "medoids") == centrotypes) ==
           0)
           stop("Wrong centrotypes argument")
       if ("medoids" == centrotypes && is.null(d))
           stop("For argument centrotypes = 'medoids' d can not be null")
       if (!is.null(d)) {
           if (!is.matrix(d)) {
               d <- as.matrix(d)
           }
           row.names(d) <- row.names(x)
       }
       X <- as.matrix(x)
       gap1 <- GAP(X, clall[, 1], reference.distribution, B,
           method, d, centrotypes)
       gap <- gap1$Sgap
       gap2 <- GAP(X, clall[, 2], reference.distribution, B,
           method, d, centrotypes)
       diffu <- gap - (gap2$Sgap - gap2$Sdgap)
       resul <- list(gap = gap, diffu = diffu)
       resul
   }
   Index.sdindex <- function(x, clmax, cl) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
    }
    Index.sdindex <- function(x, clmax, cl) {
        x <- as.matrix(x)
        Alpha <- Dis(clmax, x)
        Scatt <- Average.scattering(cl, x)$scatt
        Dis0 <- Dis(cl, x)
        SD.indice <- Alpha * Scatt + Dis0
        return(SD.indice)
    }
    Index.SDbw <- function(x, cl) {
        x <- as.matrix(x)
        Scatt <- Average.scattering(cl, x)$scatt
        Dens.bw <- density.bw(cl, x)
        SDbw <- Scatt + Dens.bw
        return(SDbw)
    }
    Index.Dindex <- function(cl, x) {
        x <- as.matrix(x)
        distance <- density.clusters(cl, x)$distance
        n <- length(distance)
        S <- 0
        for (i in 1:n) S <- S + distance[i]
        inertieIntra <- S/n
        return(inertieIntra)
    }
    Index.dunn <- function(md, clusters, Data = NULL, method = "euclidean") {
        distance <- as.matrix(md)
        nc <- max(clusters)
        interClust <- matrix(NA, nc, nc)
        intraClust <- rep(NA, nc)
        for (i in 1:nc) {
            c1 <- which(clusters == i)
            for (j in i:nc) {
                if (j == i)
                  intraClust[i] <- max(distance[c1, c1])
                if (j > i) {
                  c2 <- which(clusters == j)
                  interClust[i, j] <- min(distance[c1, c2])
                }
            }
        }
        dunn <- min(interClust, na.rm = TRUE)/max(intraClust)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
    }
    for (nc in min_nc:max_nc) {
        if (any(method == 1) || (method == 2) || (method == 3) ||
            (method == 4) || (method == 5) || (method == 6) ||
            (method == 7) || (method == 9)) {
            cl1 <- cutree(hc, k = nc)
            cl2 <- cutree(hc, k = nc + 1)
            clall <- cbind(cl1, cl2)
            clmax <- cutree(hc, k = max_nc)
            if (nc >= 2) {
                cl0 <- cutree(hc, k = nc - 1)
                clall1 <- cbind(cl0, cl1, cl2)
            }
            if (nc == 1) {
                cl0 <- rep(NA, nn)
                clall1 <- cbind(cl0, cl1, cl2)
            }
        }
        if (method == 8) {
            set.seed(1)
            cl2 <- kmeans(jeu, nc + 1)$cluster
            set.seed(1)
            clmax <- kmeans(jeu, max_nc)$cluster
            if (nc > 2) {
                set.seed(1)
                cl1 <- kmeans(jeu, nc)$cluster
                clall <- cbind(cl1, cl2)
                set.seed(1)
                cl0 <- kmeans(jeu, nc - 1)$cluster
                clall1 <- cbind(cl0, cl1, cl2)
            }
            if (nc == 2) {
                set.seed(1)
                cl1 <- kmeans(jeu, nc)$cluster
                clall <- cbind(cl1, cl2)
                cl0 <- rep(1, nn)
                clall1 <- cbind(cl0, cl1, cl2)
            }
            if (nc == 1) {
                stop("Number of clusters must be higher than 2")
            }
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
        }
        j <- table(cl1)
        s <- sum(j == 1)
        j2 <- table(cl2)
        s2 <- sum(j2 == 1)
        if (any(indice == 3) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 3] <- Indices.Traces(jeu, md,
                clall1, index = "hart")
        }
        if (any(indice == 4) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 4] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$ccc
        }
        if (any(indice == 5) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 5] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$scott
        }
        if (any(indice == 6) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 6] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$marriot
        }
        if (any(indice == 7) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 7] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$trcovw
        }
        if (any(indice == 8) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 8] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$tracew
        }
        if (any(indice == 9) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 9] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$friedman
        }
        if (any(indice == 10) || (indice == 31) || (indice ==
            32)) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            res[nc - min_nc + 1, 9] <- Indices.WBT(x = jeu, cl = cl1,
                P = TT, s = ss, vv = vv)$friedman
        }
        if (any(indice == 10) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 10] <- Indices.WBT(x = jeu,
                cl = cl1, P = TT, s = ss, vv = vv)$rubin
        }
        if (any(indice == 14) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 14] <- Indices.WKWL(x = jeu,
                cl1 = cl1, cl2 = cl2)$duda
        }
        if (any(indice == 15) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 15] <- Indices.WKWL(x = jeu,
                cl1 = cl1, cl2 = cl2)$pseudot2
        }
        if (any(indice == 16) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 16] <- beale <- Indices.WKWL(x = jeu,
                cl1 = cl1, cl2 = cl2)$beale
        }
        if (any(indice == 14) || (indice == 15) || (indice ==
            16) || (indice == 31) || (indice == 32)) {
            NM <- Indices.WKWL(x = jeu, cl1 = cl1, cl2 = cl2)$NM
            NK <- Indices.WKWL(x = jeu, cl1 = cl1, cl2 = cl2)$NK
            NL <- Indices.WKWL(x = jeu, cl1 = cl1, cl2 = cl2)$NL
            zz <- 3.2
            zzz <- zz * sqrt(2 * (1 - 8/((pi^2) * pp))/(NM *
                pp))
            if (any(indice == 14) || (indice == 31) || (indice ==
                32)) {
                resCritical[nc - min_nc + 1, 1] <- critValue <- 1 -
                  (2/(pi * pp)) - zzz
            }
            if ((indice == 15) || (indice == 31) || (indice ==
                32)) {
                critValue <- 1 - (2/(pi * pp)) - zzz
                resCritical[nc - min_nc + 1, 2] <- ((1 - critValue)/critValue) *
                  (NK + NL - 2)
            }
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            }
            if (any(indice == 16) || (indice == 31) || (indice ==
                32)) {
                df2 <- (NM - 2) * pp
                resCritical[nc - min_nc + 1, 3] <- 1 - pf(beale,
                  pp, df2)
            }
        }
        if (any(indice == 18) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 18] <- Indices.Traces(jeu, md,
                clall1, index = "ball")
        }
        if (any(indice == 19) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 19] <- Indice.ptbiserial(x = jeu,
                md = md, cl1 = cl1)
        }
        if (any(indice == 20) || (indice == 32)) {
            if (method == 1) {
                resultSGAP <- Indice.Gap(x = jeu, clall = clall,
                  reference.distribution = "unif", B = 10, method = "ward.D2",
                  d = NULL, centrotypes = "centroids")
            }
            if (method == 2) {
                resultSGAP <- Indice.Gap(x = jeu, clall = clall,
                  reference.distribution = "unif", B = 10, method = "single",
                  d = NULL, centrotypes = "centroids")
            }
            if (method == 3) {
                resultSGAP <- Indice.Gap(x = jeu, clall = clall,
                  reference.distribution = "unif", B = 10, method = "complete",
                  d = NULL, centrotypes = "centroids")
            }
            if (method == 4) {
                resultSGAP <- Indice.Gap(x = jeu, clall = clall,
                  reference.distribution = "unif", B = 10, method = "average",
                  d = NULL, centrotypes = "centroids")
            }
            if (method == 5) {
                resultSGAP <- Indice.Gap(x = jeu, clall = clall,
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            resultSGAP <- Indice.Gap(x = jeu, clall = clall,
              reference.distribution = "unif", B = 10, method = "mcquitty",
              d = NULL, centrotypes = "centroids")
        }
        if (method == 6) {
            resultSGAP <- Indice.Gap(x = jeu, clall = clall,
              reference.distribution = "unif", B = 10, method = "median",
              d = NULL, centrotypes = "centroids")
        }
        if (method == 7) {
            resultSGAP <- Indice.Gap(x = jeu, clall = clall,
              reference.distribution = "unif", B = 10, method = "centroid",
              d = NULL, centrotypes = "centroids")
        }
        if (method == 9) {
            resultSGAP <- Indice.Gap(x = jeu, clall = clall,
              reference.distribution = "unif", B = 10, method = "ward.D",
              d = NULL, centrotypes = "centroids")
        }
        if (method == 8) {
            resultSGAP <- Indice.Gap(x = jeu, clall = clall,
              reference.distribution = "unif", B = 10, method = "k-means",
              d = NULL, centrotypes = "centroids")
        }
        res[nc - min_nc + 1, 20] <- resultSGAP$gap
        resCritical[nc - min_nc + 1, 4] <- resultSGAP$diffu
    }
    if (nc >= 2) {
        if (any(indice == 1) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 1] <- Indices.Traces(jeu,
              md, clall1, index = "kl")
        }
        if (any(indice == 2) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 2] <- Indices.Traces(jeu,
              md, clall1, index = "ch")
        }
        if (any(indice == 11) || (indice == 31) || (indice ==
            32)) {
            res[nc - min_nc + 1, 11] <- Indice.cindex(d = md,
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            cl = cl1)
    }
    if (any(indice == 12) || (indice == 31) || (indice ==
        32)) {
        res[nc - min_nc + 1, 12] <- Indice.DB(x = jeu,
          cl = cl1, d = NULL, centrotypes = "centroids",
          p = 2, q = 2)$DB
    }
    if (any(indice == 13) || (indice == 31) || (indice ==
        32)) {
        res[nc - min_nc + 1, 13] <- Indice.S(d = md,
          cl = cl1)
    }
    if (any(indice == 17) || (indice == 31) || (indice ==
        32)) {
        res[nc - min_nc + 1, 17] <- Indices.Traces(jeu,
          md, clall1, index = "ratkowsky")
    }
    if (any(indice == 21) || (indice == 31) || (indice ==
        32)) {
        res[nc - min_nc + 1, 21] <- Index.15and28(cl1 = cl1,
          cl2 = cl2, md = md)$frey
    }
    if (any(indice == 22) || (indice == 31) || (indice ==
        32)) {
        res[nc - min_nc + 1, 22] <- Index.15and28(cl1 = cl1,
          cl2 = cl2, md = md)$mcclain
    }
    if (any(indice == 23) || (indice == 32)) {
        res[nc - min_nc + 1, 23] <- Index.sPlussMoins(cl1 = cl1,
          md = md)$gamma
    }
    if (any(indice == 24) || (indice == 32)) {
        res[nc - min_nc + 1, 24] <- Index.sPlussMoins(cl1 = cl1,
          md = md)$gplus
    }
    if (any(indice == 25) || (indice == 32)) {
        res[nc - min_nc + 1, 25] <- Index.sPlussMoins(cl1 = cl1,
          md = md)$tau
    }
    if (any(indice == 26) || (indice == 31) || (indice ==
        32)) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            if (any(indice == 26) || (indice == 31) || (indice ==
                32)) {
                res[nc - min_nc + 1, 26] <- Index.dunn(md, cl1,
                  Data = jeu, method = NULL)
            }
            if (any(indice == 27) || (indice == 31) || (indice ==
                32)) {
                res[nc - min_nc + 1, 27] <- Index.Hubert(jeu,
                  cl1)
            }
            if (any(indice == 28) || (indice == 31) || (indice ==
                32)) {
                res[nc - min_nc + 1, 28] <- Index.sdindex(jeu,
                  clmax, cl1)
            }
            if (any(indice == 29) || (indice == 31) || (indice ==
                32)) {
                res[nc - min_nc + 1, 29] <- Index.Dindex(cl1,
                  jeu)
            }
            if (any(indice == 30) || (indice == 31) || (indice ==
                32)) {
                res[nc - min_nc + 1, 30] <- Index.SDbw(jeu, cl1)
            }
        }
        else {
            res[nc - min_nc + 1, 1] <- NA
            res[nc - min_nc + 1, 2] <- NA
            res[nc - min_nc + 1, 11] <- NA
            res[nc - min_nc + 1, 12] <- NA
            res[nc - min_nc + 1, 13] <- NA
            res[nc - min_nc + 1, 17] <- NA
            res[nc - min_nc + 1, 21] <- NA
            res[nc - min_nc + 1, 22] <- NA
            res[nc - min_nc + 1, 23] <- NA
            res[nc - min_nc + 1, 24] <- NA
            res[nc - min_nc + 1, 25] <- NA
            res[nc - min_nc + 1, 26] <- NA
            res[nc - min_nc + 1, 27] <- NA
            res[nc - min_nc + 1, 28] <- NA
            res[nc - min_nc + 1, 29] <- NA
            res[nc - min_nc + 1, 30] <- NA
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
                res[nc - min_nc + 1, 30] <- NA
            }
        }
        nc.KL <- indice.KL <- 0
        if (any(indice == 1) || (indice == 31) || (indice == 32)) {
            nc.KL <- (min_nc:max_nc)[which.max(res[, 1])]
            indice.KL <- max(res[, 1], na.rm = TRUE)
            best.nc <- nc.KL
        }
        nc.CH <- indice.CH <- 0
        if (any(indice == 2) || (indice == 31) || (indice == 32)) {
            nc.CH <- (min_nc:max_nc)[which.max(res[, 2])]
            indice.CH <- max(res[, 2], na.rm = TRUE)
            best.nc <- nc.CH
        }
        nc.CCC <- indice.CCC <- 0
        if (any(indice == 4) || (indice == 31) || (indice == 32)) {
            nc.CCC <- (min_nc:max_nc)[which.max(res[, 4])]
            indice.CCC <- max(res[, 4], na.rm = TRUE)
            best.nc <- nc.CCC
        }
        nc.DB <- indice.DB <- 0
        if (any(indice == 12) || (indice == 31) || (indice == 32)) {
            nc.DB <- (min_nc:max_nc)[which.min(res[, 12])]
            indice.DB <- min(res[, 12], na.rm = TRUE)
            best.nc <- nc.DB
        }
        nc.Silhouette <- indice.Silhouette <- 0
        if (any(indice == 13) || (indice == 31) || (indice == 32)) {
            nc.Silhouette <- (min_nc:max_nc)[which.max(res[, 13])]
            indice.Silhouette <- max(res[, 13], na.rm = TRUE)
            best.nc <- nc.Silhouette
        }
        nc.Gap <- indice.Gap <- 0
        if (any(indice == 20) || (indice == 32)) {
            found <- FALSE
            for (ncG in min_nc:max_nc) {
                if ((resCritical[ncG - min_nc + 1, 4] >= 0) && (!found)) {
                    ncGap <- ncG
                    indiceGap <- res[ncG - min_nc + 1, 20]
                    found <- TRUE
                }
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal × Background Jobs ×

R 4.3.0 · ~/

```
        }
        if (found) {
            nc.Gap <- ncGap
            indice.Gap <- indiceGap
            best.nc <- nc.Gap
        }
        else {
            nc.Gap <- NA
            indice.Gap <- NA
        }
    }
    nc.Duda <- indice.Duda <- 0
    if (any(indice == 14) || (indice == 31) || (indice == 32)) {
        foundDuda <- FALSE
        for (ncD in min_nc:max_nc) {
            if ((res[ncD - min_nc + 1, 14] >= resCritical[ncD -
                min_nc + 1, 1]) && (!foundDuda)) {
                ncDuda <- ncD
                indiceDuda <- res[ncD - min_nc + 1, 14]
                foundDuda <- TRUE
            }
        }
        if (foundDuda) {
            nc.Duda <- ncDuda
            indice.Duda <- indiceDuda
            best.nc <- nc.Duda
        }
        else {
            nc.Duda <- NA
            indice.Duda <- NA
        }
    }
    nc.Pseudo <- indice.Pseudo <- 0
    if (any(indice == 15) || (indice == 31) || (indice == 32)) {
        foundPseudo <- FALSE
        for (ncP in min_nc:max_nc) {
            if ((res[ncP - min_nc + 1, 15] <= resCritical[ncP -
                min_nc + 1, 2]) && (!foundPseudo)) {
                ncPseudo <- ncP
                indicePseudo <- res[ncP - min_nc + 1, 15]
                foundPseudo <- TRUE
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            }
        }
        if (foundPseudo) {
            nc.Pseudo <- ncPseudo
            indice.Pseudo <- indicePseudo
            best.nc <- nc.Pseudo
        }
        else {
            nc.Pseudo <- NA
            indice.Pseudo <- NA
        }
    }
    nc.Beale <- indice.Beale <- 0
    if (any(indice == 16) || (indice == 31) || (indice == 32)) {
        foundBeale <- FALSE
        for (ncB in min_nc:max_nc) {
            if ((resCritical[ncB - min_nc + 1, 3] >= alphaBeale) &&
                (!foundBeale)) {
                ncBeale <- ncB
                indiceBeale <- res[ncB - min_nc + 1, 16]
                foundBeale <- TRUE
            }
        }
        if (foundBeale) {
            nc.Beale <- ncBeale
            indice.Beale <- indiceBeale
            best.nc <- nc.Beale
        }
        else {
            nc.Beale <- NA
            indice.Beale <- NA
        }
    }
    nc.ptbiserial <- indice.ptbiserial <- 0
    if (any(indice == 19) || (indice == 31) || (indice == 32)) {
        nc.ptbiserial <- (min_nc:max_nc)[which.max(res[, 19])]
        indice.ptbiserial <- max(res[, 19], na.rm = TRUE)
        best.nc <- nc.ptbiserial
    }
    foundNC <- foundIndice <- numeric(0)
    nc.Frey <- indice.Frey <- 0
    if (any(indice == 21) || (indice == 31) || (indice == 32)) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
        for (ncF in min_nc:max_nc) {
            if (res[ncF - min_nc + 1, 21] < 1) {
                ncFrey <- ncF - 1
                indiceFrey <- res[ncF - 1 - min_nc + 1, 21]
                foundFrey <- TRUE
                foundNC[i] <- ncFrey
                foundIndice[i] <- indiceFrey
                i <- i + 1
            }
        }
        if (foundFrey) {
            nc.Frey <- foundNC[1]
            indice.Frey <- foundIndice[1]
            best.nc <- nc.Frey
        }
        else {
            nc.Frey <- NA
            indice.Frey <- NA
            print(paste("Frey index : No clustering structure in this data set"))
        }
    }
    nc.McClain <- indice.McClain <- 0
    if (any(indice == 22) || (indice == 31) || (indice == 32)) {
        nc.McClain <- (min_nc:max_nc)[which.min(res[, 22])]
        indice.McClain <- min(res[, 22], na.rm = TRUE)
        best.nc <- nc.McClain
    }
    nc.Gamma <- indice.Gamma <- 0
    if (any(indice == 23) || (indice == 31) || (indice == 32)) {
        nc.Gamma <- (min_nc:max_nc)[which.max(res[, 23])]
        indice.Gamma <- max(res[, 23], na.rm = TRUE)
        best.nc <- nc.Gamma
    }
    nc.Gplus <- indice.Gplus <- 0
    if (any(indice == 24) || (indice == 31) || (indice == 32)) {
        nc.Gplus <- (min_nc:max_nc)[which.min(res[, 24])]
        indice.Gplus <- min(res[, 24], na.rm = TRUE)
        best.nc <- nc.Gplus
    }
    nc.Tau <- indice.Tau <- 0
    if (any(indice == 25) || (indice == 31) || (indice == 32)) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
    }
    if ((indice == 3) || (indice == 5) || (indice == 6) || (indice ==
        7) || (indice == 8) || (indice == 9) || (indice == 10) ||
        (indice == 18) || (indice == 27) || (indice == 11) ||
        (indice == 29) || (indice == 31) || (indice == 32)) {
        DiffLev <- array(0, c(max_nc - min_nc + 1, 12))
        DiffLev[, 1] <- min_nc:max_nc
        for (nc3 in min_nc:max_nc) {
            if (nc3 == min_nc) {
                DiffLev[nc3 - min_nc + 1, 2] <- abs(res[nc3 -
                  min_nc + 1, 3] - NA)
                DiffLev[nc3 - min_nc + 1, 3] <- abs(res[nc3 -
                  min_nc + 1, 5] - NA)
                DiffLev[nc3 - min_nc + 1, 4] <- abs(res[nc3 -
                  min_nc + 1, 6] - NA)
                DiffLev[nc3 - min_nc + 1, 5] <- abs(res[nc3 -
                  min_nc + 1, 7] - NA)
                DiffLev[nc3 - min_nc + 1, 6] <- abs(res[nc3 -
                  min_nc + 1, 8] - NA)
                DiffLev[nc3 - min_nc + 1, 7] <- abs(res[nc3 -
                  min_nc + 1, 9] - NA)
                DiffLev[nc3 - min_nc + 1, 8] <- abs(res[nc3 -
                  min_nc + 1, 10] - NA)
                DiffLev[nc3 - min_nc + 1, 9] <- abs(res[nc3 -
                  min_nc + 1, 18] - NA)
                DiffLev[nc3 - min_nc + 1, 10] <- abs(res[nc3 -
                  min_nc + 1, 27] - NA)
                DiffLev[nc3 - min_nc + 1, 12] <- abs(res[nc3 -
                  min_nc + 1, 29] - NA)
            }
            else {
                if (nc3 == max_nc) {
                  DiffLev[nc3 - min_nc + 1, 2] <- abs(res[nc3 -
                    min_nc + 1, 3] - res[nc3 - min_nc, 3])
                  DiffLev[nc3 - min_nc + 1, 3] <- abs(res[nc3 -
                    min_nc + 1, 5] - res[nc3 - min_nc, 5])
                  DiffLev[nc3 - min_nc + 1, 4] <- abs(res[nc3 -
                    min_nc + 1, 6] - NA)
                  DiffLev[nc3 - min_nc + 1, 5] <- abs(res[nc3 -
                    min_nc + 1, 7] - res[nc3 - min_nc, 7])
                  DiffLev[nc3 - min_nc + 1, 6] <- abs(res[nc3 -
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
          min_nc + 1, 9] - res[nc3 - min_nc, 9])
        DiffLev[nc3 - min_nc + 1, 8] <- abs(res[nc3 -
          min_nc + 1, 10] - NA)
        DiffLev[nc3 - min_nc + 1, 9] <- abs(res[nc3 -
          min_nc + 1, 18] - res[nc3 - min_nc, 18])
        DiffLev[nc3 - min_nc + 1, 10] <- abs(res[nc3 -
          min_nc + 1, 27] - NA)
        DiffLev[nc3 - min_nc + 1, 12] <- abs(res[nc3 -
          min_nc + 1, 29] - NA)
      }
      else {
        DiffLev[nc3 - min_nc + 1, 2] <- abs(res[nc3 -
          min_nc + 1, 3] - res[nc3 - min_nc, 3])
        DiffLev[nc3 - min_nc + 1, 3] <- abs(res[nc3 -
          min_nc + 1, 5] - res[nc3 - min_nc, 5])
        DiffLev[nc3 - min_nc + 1, 4] <- ((res[nc3 -
          min_nc + 2, 6] - res[nc3 - min_nc + 1, 6]) -
          (res[nc3 - min_nc + 1, 6] - res[nc3 - min_nc,
            6]))
        DiffLev[nc3 - min_nc + 1, 5] <- abs(res[nc3 -
          min_nc + 1, 7] - res[nc3 - min_nc, 7])
        DiffLev[nc3 - min_nc + 1, 6] <- ((res[nc3 -
          min_nc + 2, 8] - res[nc3 - min_nc + 1, 8]) -
          (res[nc3 - min_nc + 1, 8] - res[nc3 - min_nc,
            8]))
        DiffLev[nc3 - min_nc + 1, 7] <- abs(res[nc3 -
          min_nc + 1, 9] - res[nc3 - min_nc, 9])
        DiffLev[nc3 - min_nc + 1, 8] <- ((res[nc3 -
          min_nc + 2, 10] - res[nc3 - min_nc + 1, 10]) -
          (res[nc3 - min_nc + 1, 10] - res[nc3 - min_nc,
            10]))
        DiffLev[nc3 - min_nc + 1, 9] <- abs(res[nc3 -
          min_nc + 1, 18] - res[nc3 - min_nc, 18])
        DiffLev[nc3 - min_nc + 1, 10] <- abs((res[nc3 -
          min_nc + 1, 27] - res[nc3 - min_nc, 27]))
        DiffLev[nc3 - min_nc + 1, 12] <- ((res[nc3 -
          min_nc + 2, 29] - res[nc3 - min_nc + 1, 29]) -
          (res[nc3 - min_nc + 1, 29] - res[nc3 - min_nc,
            29]))
      }
    }
  }
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
    if (any(indice == 3) || (indice == 31) || (indice == 32)) {
        nc.Hartigan <- DiffLev[, 1][which.max(DiffLev[, 2])]
        indice.Hartigan <- max(DiffLev[, 2], na.rm = TRUE)
        best.nc <- nc.Hartigan
    }
    nc.Ratkowsky <- indice.Ratkowsky <- 0
    if (any(indice == 17) || (indice == 31) || (indice == 32)) {
        nc.Ratkowsky <- (min_nc:max_nc)[which.max(res[, 17])]
        indice.Ratkowsky <- max(res[, 17], na.rm = TRUE)
        best.nc <- nc.Ratkowsky
    }
    nc.cindex <- indice.cindex <- 0
    if (any(indice == 11) || (indice == 31) || (indice == 32)) {
        nc.cindex <- (min_nc:max_nc)[which.min(res[, 11])]
        indice.cindex <- min(res[, 11], na.rm = TRUE)
        best.nc <- nc.cindex
    }
    nc.Scott <- indice.Scott <- 0
    if (any(indice == 5) || (indice == 31) || (indice == 32)) {
        nc.Scott <- DiffLev[, 1][which.max(DiffLev[, 3])]
        indice.Scott <- max(DiffLev[, 3], na.rm = TRUE)
        best.nc <- nc.Scott
    }
    nc.Marriot <- indice.Marriot <- 0
    if (any(indice == 6) || (indice == 31) || (indice == 32)) {
        nc.Marriot <- DiffLev[, 1][which.max(DiffLev[, 4])]
        round(nc.Marriot, digits = 1)
        indice.Marriot <- max(DiffLev[, 4], na.rm = TRUE)
        best.nc <- nc.Marriot
    }
    nc.TrCovW <- indice.TrCovW <- 0
    if (any(indice == 7) || (indice == 31) || (indice == 32)) {
        nc.TrCovW <- DiffLev[, 1][which.max(DiffLev[, 5])]
        indice.TrCovW <- max(DiffLev[, 5], na.rm = TRUE)
        best.nc <- nc.TrCovW
    }
    nc.TraceW <- indice.TraceW <- 0
    if (any(indice == 8) || (indice == 31) || (indice == 32)) {
        nc.TraceW <- DiffLev[, 1][which.max(DiffLev[, 6])]
        indice.TraceW <- max(DiffLev[, 6], na.rm = TRUE)
        best.nc <- nc.TraceW
    }
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

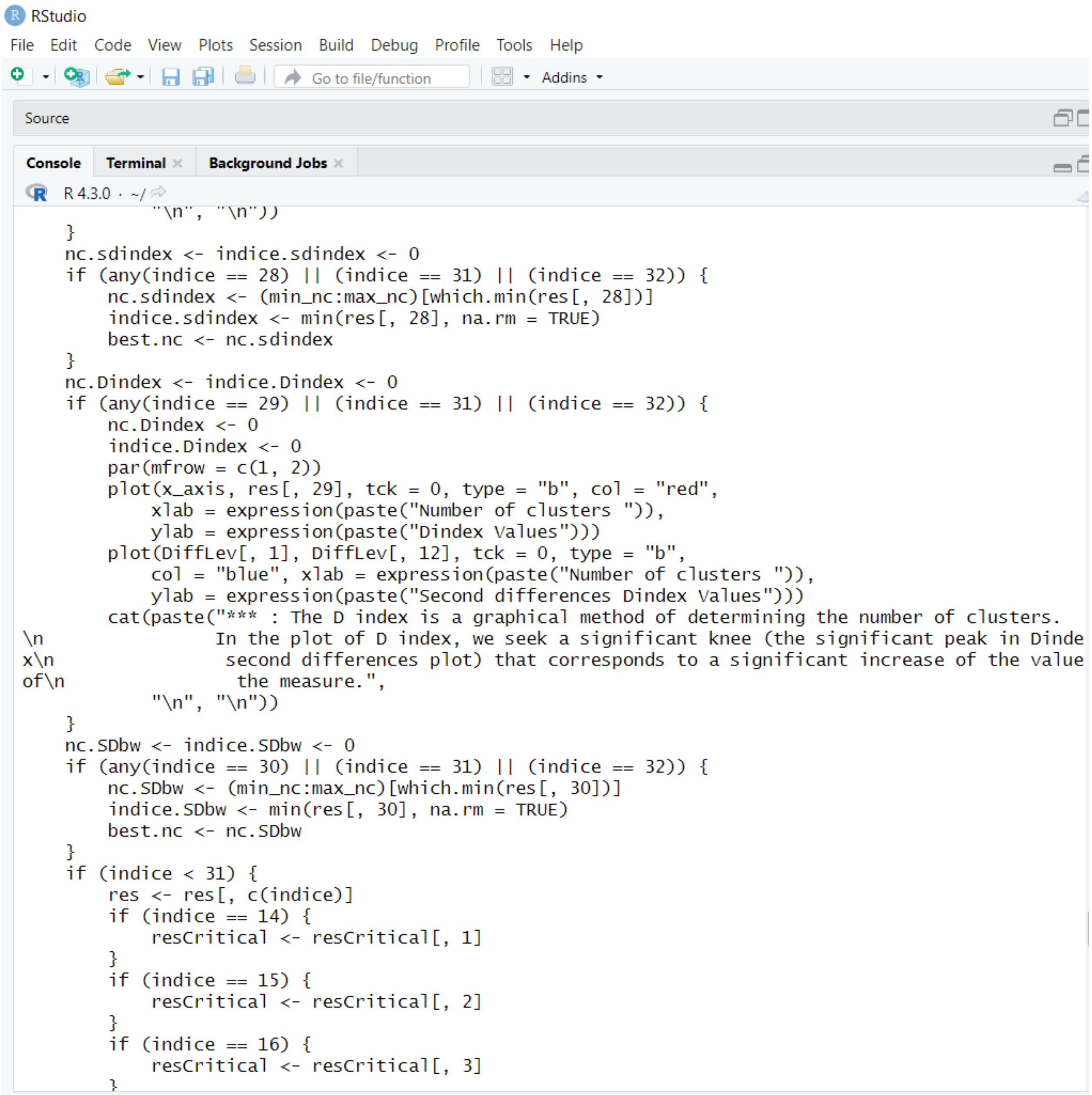
Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
        nc.Friedman <- DiffLev[, 1][which.max(DiffLev[, 7])]
        indice.Friedman <- max(DiffLev[, 7], na.rm = TRUE)
        best.nc <- nc.Friedman
    }
    nc.Rubin <- indice.Rubin <- 0
    if (any(indice == 10) || (indice == 31) || (indice == 32)) {
        nc.Rubin <- DiffLev[, 1][which.min(DiffLev[, 8])]
        indice.Rubin <- min(DiffLev[, 8], na.rm = TRUE)
        best.nc <- nc.Rubin
    }
    nc.Ball <- indice.Ball <- 0
    if (any(indice == 18) || (indice == 31) || (indice == 32)) {
        nc.Ball <- DiffLev[, 1][which.max(DiffLev[, 9])]
        indice.Ball <- max(DiffLev[, 9], na.rm = TRUE)
        best.nc <- nc.Ball
    }
    nc.Dunn <- indice.Dunn <- 0
    if (any(indice == 26) || (indice == 31) || (indice == 32)) {
        nc.Dunn <- (min_nc:max_nc)[which.max(res[, 26])]
        indice.Dunn <- max(res[, 26], na.rm = TRUE)
        best.nc <- nc.Dunn
    }
    nc.Hubert <- indice.Hubert <- 0
    if (any(indice == 27) || (indice == 31) || (indice == 32)) {
        nc.Hubert <- 0
        indice.Hubert <- 0
        par(mfrow = c(1, 2))
        plot(x_axis, res[, 27], tck = 0, type = "b", col = "red",
            xlab = expression(paste("Number of clusters ")),
            ylab = expression(paste("Hubert Statistic values")))
        plot(DiffLev[, 1], DiffLev[, 10], tck = 0, type = "b",
            col = "blue", xlab = expression(paste("Number of clusters ")),
            ylab = expression(paste("Hubert statistic second differences")))
        cat(paste("*** : The Hubert index is a graphical method of determining the number of cluste
rs.\n                In the plot of Hubert index, we seek a significant knee that corresponds to a
\n              significant increase of the value of the measure i.e the significant peak in Hube
rt\n              index second differences plot.",
            "\n", "\n"))
    }
    nc.sdindex <- indice.sdindex <- 0
    if (any(indice == 28) || (indice == 31) || (indice == 32)) {
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console
Terminal
Background Jobs
R 4.3.0 · ~/
"\n", "\n"))
}
nc.sdindex <- indice.sdindex <- 0
if (any(indice == 28) || (indice == 31) || (indice == 32)) {
nc.sdindex <- (min_nc:max_nc)[which.min(res[, 28])]
indice.sdindex <- min(res[, 28], na.rm = TRUE)
best.nc <- nc.sdindex
}
nc.Dindex <- indice.Dindex <- 0
if (any(indice == 29) || (indice == 31) || (indice == 32)) {
nc.Dindex <- 0
indice.Dindex <- 0
par(mfrow = c(1, 2))
plot(x_axis, res[, 29], tck = 0, type = "b", col = "red",
xlab = expression(paste("Number of clusters ")),
ylab = expression(paste("Dindex Values")))
plot(DiffLev[, 1], DiffLev[, 12], tck = 0, type = "b",
col = "blue", xlab = expression(paste("Number of clusters ")),
ylab = expression(paste("Second differences Dindex Values")))
cat(paste("*** : The D index is a graphical method of determining the number of clusters.
\n In the plot of D index, we seek a significant knee (the significant peak in Dinde
x\n second differences plot) that corresponds to a significant increase of the value
of\n the measure.",
"\n", "\n"))
}
nc.SDbw <- indice.SDbw <- 0
if (any(indice == 30) || (indice == 31) || (indice == 32)) {
nc.SDbw <- (min_nc:max_nc)[which.min(res[, 30])]
indice.SDbw <- min(res[, 30], na.rm = TRUE)
best.nc <- nc.SDbw
}
if (indice < 31) {
res <- res[, c(indice)]
if (indice == 14) {
resCritical <- resCritical[, 1]
}
if (indice == 15) {
resCritical <- resCritical[, 2]
}
if (indice == 16) {
resCritical <- resCritical[, 3]
}

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```r
        }
    }
    if (indice == 31) {
        res <- res[, c(1:19, 21:22, 26:30)]
        resCritical <- resCritical[, c(1:3)]
    }
    if (any(indice == 20) || (indice == 23) || (indice == 24) ||
        (indice == 25) || (indice == 32)) {
        results <- c(nc.KL, indice.KL, nc.CH, indice.CH, nc.Hartigan,
            indice.Hartigan, nc.CCC, indice.CCC, nc.Scott, indice.Scott,
            nc.Marriot, indice.Marriot, nc.TrCovW, indice.TrCovW,
            nc.TraceW, indice.TraceW, nc.Friedman, indice.Friedman,
            nc.Rubin, indice.Rubin, nc.cindex, indice.cindex,
            nc.DB, indice.DB, nc.Silhouette, indice.Silhouette,
            nc.Duda, indice.Duda, nc.Pseudo, indice.Pseudo, nc.Beale,
            indice.Beale, nc.Ratkowsky, indice.Ratkowsky, nc.Ball,
            indice.Ball, nc.ptbiserial, indice.ptbiserial, nc.Gap,
            indice.Gap, nc.Frey, indice.Frey, nc.McClain, indice.McClain,
            nc.Gamma, indice.Gamma, nc.Gplus, indice.Gplus, nc.Tau,
            indice.Tau, nc.Dunn, indice.Dunn, nc.Hubert, indice.Hubert,
            nc.sdindex, indice.sdindex, nc.Dindex, indice.Dindex,
            nc.SDbw, indice.SDbw)
        results1 <- matrix(c(results), nrow = 2, ncol = 30)
        resultats <- matrix(c(results), nrow = 2, ncol = 30,
            dimnames = list(c("Number_clusters", "Value_Index"),
                c("KL", "CH", "Hartigan", "CCC", "Scott", "Marriot",
                  "TrCovW", "TraceW", "Friedman", "Rubin", "Cindex",
                  "DB", "Silhouette", "Duda", "PseudoT2", "Beale",
                  "Ratkowsky", "Ball", "PtBiserial", "Gap", "Frey",
                  "McClain", "Gamma", "Gplus", "Tau", "Dunn",
                  "Hubert", "SDindex", "Dindex", "SDbw")))
    }
    else {
        results <- c(nc.KL, indice.KL, nc.CH, indice.CH, nc.Hartigan,
            indice.Hartigan, nc.CCC, indice.CCC, nc.Scott, indice.Scott,
            nc.Marriot, indice.Marriot, nc.TrCovW, indice.TrCovW,
            nc.TraceW, indice.TraceW, nc.Friedman, indice.Friedman,
            nc.Rubin, indice.Rubin, nc.cindex, indice.cindex,
            nc.DB, indice.DB, nc.Silhouette, indice.Silhouette,
            nc.Duda, indice.Duda, nc.Pseudo, indice.Pseudo, nc.Beale,
            indice.Beale, nc.Ratkowsky, indice.Ratkowsky, nc.Ball,
            indice.Ball, nc.ptbiserial, indice.ptbiserial, nc.Frey,
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
            indice.sdindex, nc.Dindex, indice.Dindex, nc.SDbw,
            indice.SDbw)
        results1 <- matrix(c(results), nrow = 2, ncol = 26)
        resultats <- matrix(c(results), nrow = 2, ncol = 26,
            dimnames = list(c("Number_clusters", "Value_Index"),
                c("KL", "CH", "Hartigan", "CCC", "Scott", "Marriot",
                  "TrCovW", "TraceW", "Friedman", "Rubin", "Cindex",
                  "DB", "Silhouette", "Duda", "PseudoT2", "Beale",
                  "Ratkowsky", "Ball", "PtBiserial", "Frey",
                  "McClain", "Dunn", "Hubert", "SDindex", "Dindex",
                  "SDbw")))
    }
    if (any(indice <= 20) || (indice == 23) || (indice == 24) ||
        (indice == 25)) {
        resultats <- resultats[, c(indice)]
    }
    if (any(indice == 21) || (indice == 22)) {
        indice3 <- indice - 1
        resultats <- resultats[, c(indice3)]
    }
    if (any(indice == 26) || (indice == 27) || (indice == 28) ||
        (indice == 29) || (indice == 30)) {
        indice4 <- indice - 4
        resultats <- resultats[, c(indice4)]
    }
    resultats <- round(resultats, digits = 4)
    res <- round(res, digits = 4)
    resCritical <- round(resCritical, digits = 4)
    if (any(indice == 31) || (indice == 32)) {
        cat("*******************************************************************",
            "\n")
        cat("* Among all indices:                                               ",
            "\n")
        BestCluster <- results1[1, ]
        c = 0
        for (i in min.nc:max.nc) {
            vect <- which(BestCluster == i)
            if (length(vect) > 0)
                cat("*", length(vect), "proposed", i, "as the best number of clusters",
                  "\n")
            if (c < length(vect)) {
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function   Addins

Source

Console   Terminal ×   Background Jobs ×

R 4.3.0 · ~/

```
        }
        cat("\n", "                  ***** Conclusion *****                           ",
            "\n", "\n")
        cat("* According to the majority rule, the best number of clusters is ",
            j, "\n", "\n", "\n")
        cat("*******************************************************************",
            "\n")
        if (any(method == 1) || (method == 2) || (method == 3) ||
            (method == 4) || (method == 5) || (method == 6) ||
            (method == 7) || (method == 9))
            partition <- cutree(hc, k = j)
        else {
            set.seed(1)
            partition <- kmeans(jeu, j)$cluster
        }
    }
    if (any(indice == 1) || (indice == 2) || (indice == 3) ||
        (indice == 4) || (indice == 5) || (indice == 6) || (indice ==
        7) || (indice == 8) || (indice == 9) || (indice == 10) ||
        (indice == 11) || (indice == 12) || (indice == 13) ||
        (indice == 14) || (indice == 15) || (indice == 16) ||
        (indice == 17) || (indice == 18) || (indice == 19) ||
        (indice == 20) || (indice == 21) || (indice == 22) ||
        (indice == 23) || (indice == 24) || (indice == 25) ||
        (indice == 26) || (indice == 28) || (indice == 30)) {
        if (any(method == 1) || (method == 2) || (method == 3) ||
            (method == 4) || (method == 5) || (method == 6) ||
            (method == 7) || (method == 9))
            partition <- cutree(hc, k = best.nc)
        else {
            set.seed(1)
            partition <- kmeans(jeu, best.nc)$cluster
        }
    }
    if ((indice == 14) || (indice == 15) || (indice == 16) ||
        (indice == 20) || (indice == 31) || (indice == 32)) {
        results.final <- list(All.index = res, All.CriticalValues = resCritical,
            Best.nc = resultats, Best.partition = partition)
    }
    if ((indice == 27) || (indice == 29))
        results.final <- list(All.index = res)
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
    }
    if (any(indice == 1) || (indice == 2) || (indice == 3) ||
        (indice == 4) || (indice == 5) || (indice == 6) || (indice ==
        7) || (indice == 8) || (indice == 9) || (indice == 10) ||
        (indice == 11) || (indice == 12) || (indice == 13) ||
        (indice == 14) || (indice == 15) || (indice == 16) ||
        (indice == 17) || (indice == 18) || (indice == 19) ||
        (indice == 20) || (indice == 21) || (indice == 22) ||
        (indice == 23) || (indice == 24) || (indice == 25) ||
        (indice == 26) || (indice == 28) || (indice == 30)) {
        if (any(method == 1) || (method == 2) || (method == 3) ||
            (method == 4) || (method == 5) || (method == 6) ||
            (method == 7) || (method == 9))
            partition <- cutree(hc, k = best.nc)
        else {
            set.seed(1)
            partition <- kmeans(jeu, best.nc)$cluster
        }
    }
    if ((indice == 14) || (indice == 15) || (indice == 16) ||
        (indice == 20) || (indice == 31) || (indice == 32)) {
        results.final <- list(All.index = res, All.CriticalValues = resCritical,
            Best.nc = resultats, Best.partition = partition)
    }
    if ((indice == 27) || (indice == 29))
        results.final <- list(All.index = res)
    if (any(indice == 1) || (indice == 2) || (indice == 3) ||
        (indice == 4) || (indice == 5) || (indice == 6) || (indice ==
        7) || (indice == 8) || (indice == 9) || (indice == 10) ||
        (indice == 11) || (indice == 12) || (indice == 13) ||
        (indice == 17) || (indice == 18) || (indice == 19) ||
        (indice == 21) || (indice == 22) || (indice == 23) ||
        (indice == 24) || (indice == 25) || (indice == 26) ||
        (indice == 28) || (indice == 30))
        results.final <- list(All.index = res, Best.nc = resultats,
            Best.partition = partition)
    return(results.final)
}
<bytecode: 0x000001c5be1ea998>
<environment: namespace:NbClust>
>
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

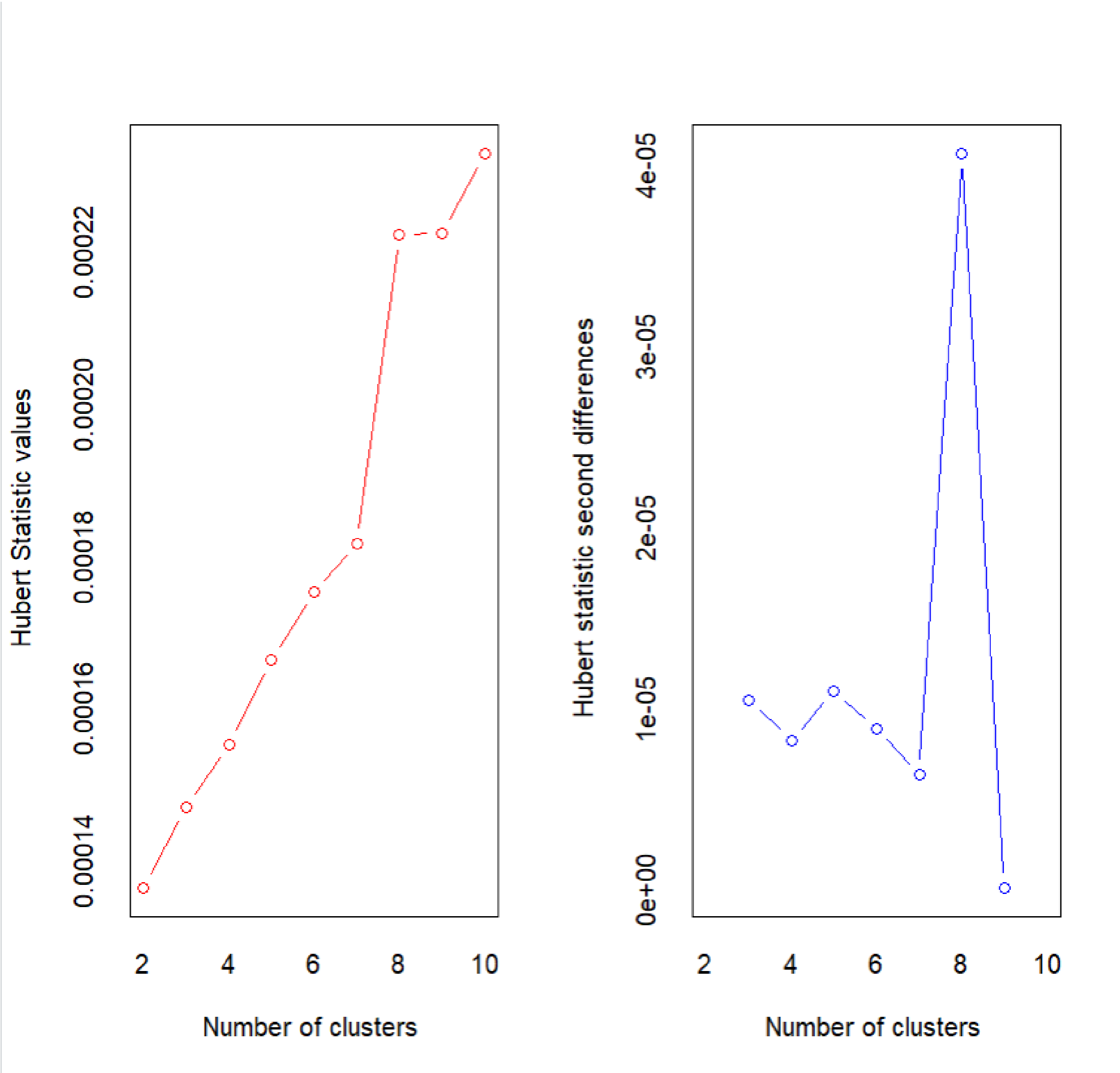
Go to file/function Addins

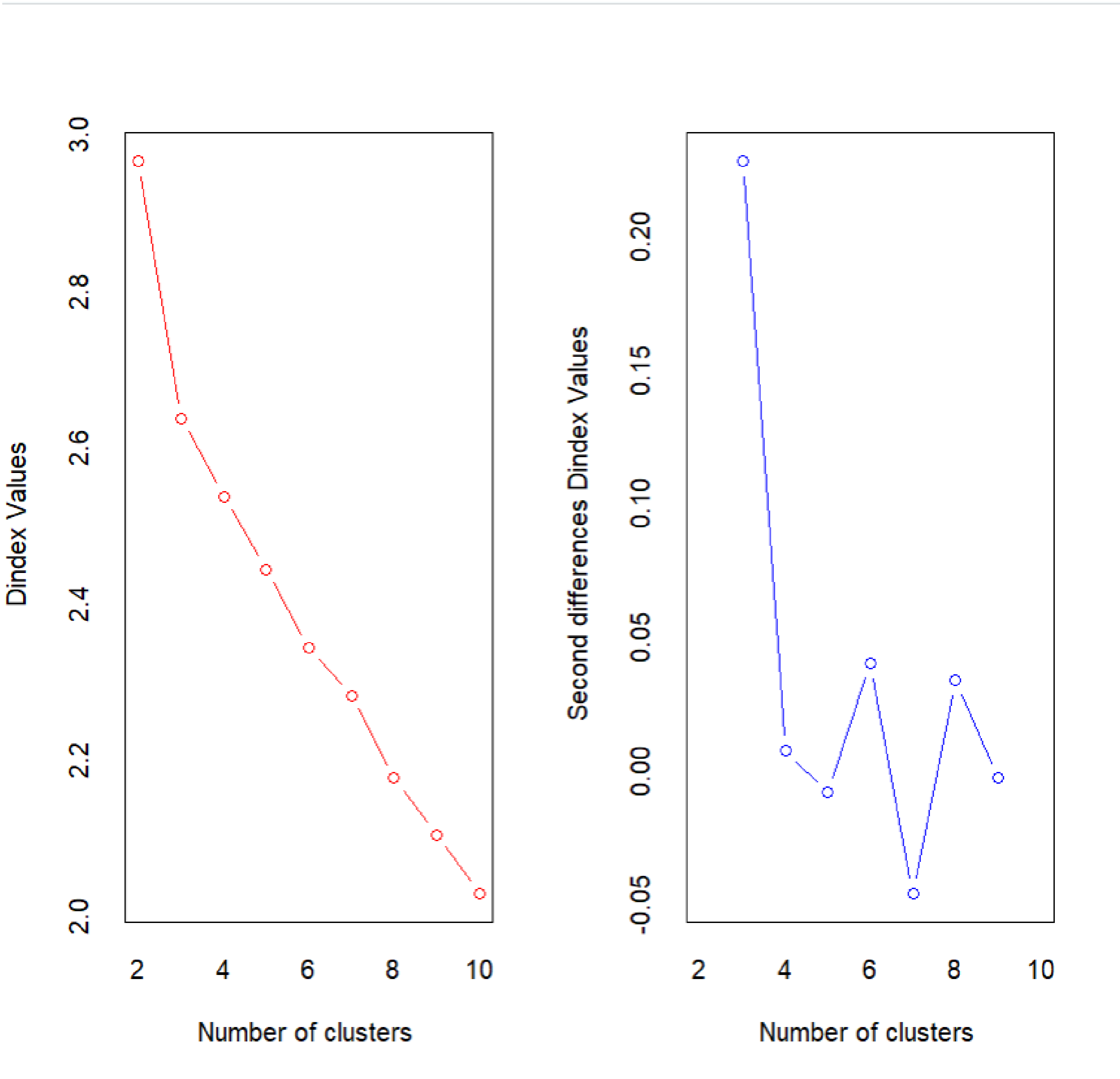
Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
        (indice == 21) || (indice == 22) || (indice == 23) ||
        (indice == 24) || (indice == 25) || (indice == 26) ||
        (indice == 28) || (indice == 30))
        results.final <- list(All.index = res, Best.nc = resultats,
            Best.partition = partition)
    return(results.final)
}
<bytecode: 0x000001c5be1ea998>
<environment: namespace:NbClust>
> set.seed(123)
> nb_clusters <- NbClust(D_Scaled, diss = NULL, distance = "euclidean", method = "kmeans", min.nc =
2, max.nc = 10, index = "all")
*** : The Hubert index is a graphical method of determining the number of clusters.
                In the plot of Hubert index, we seek a significant knee that corresponds to a
                significant increase of the value of the measure i.e the significant peak in Hubert
                index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the significant peak in Dindex
                second differences plot) that corresponds to a significant increase of the value of
                the measure.

*******************************************************************
* Among all indices:
* 10 proposed 2 as the best number of clusters
* 5 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 2 proposed 7 as the best number of clusters
* 4 proposed 8 as the best number of clusters
* 2 proposed 10 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2


*******************************************************************
> table(nb_clusters$Best.n[1,])

 0  2  3  4  7  8 10
 2 10  5  1  2  4  2
```

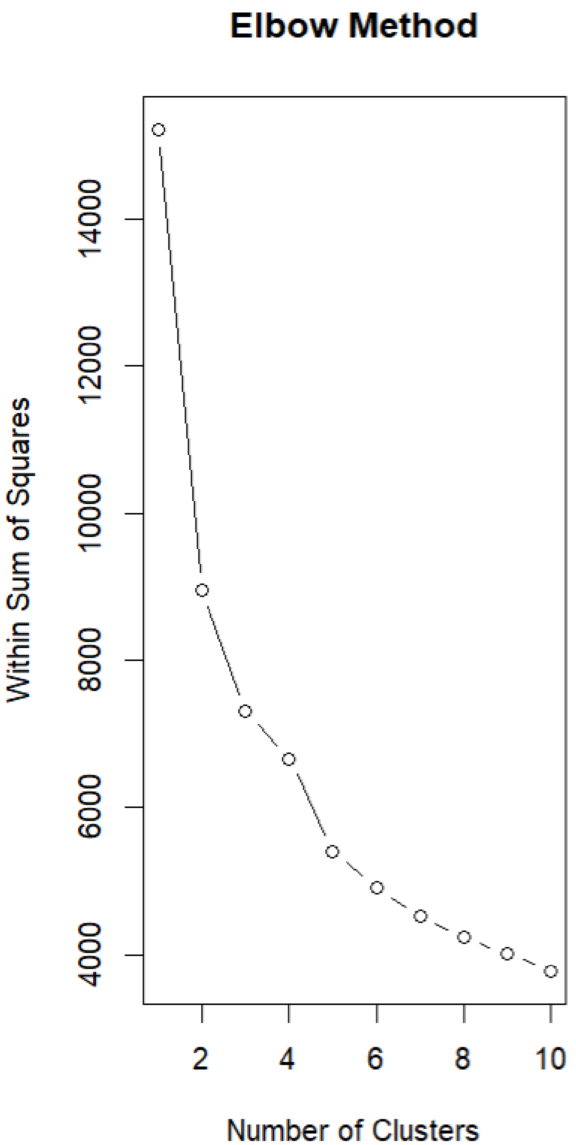

Hubert Statistic values
0.00014
0.00016
0.00018
0.00020
0.00022
2
4
6
8
10
Number of clusters
Hubert statistic second differences
0e+00
1e-05
2e-05
3e-05
4e-05
2
4
6
8
10
Number of clusters



Elbow method

The elbow method is a method used to calculate the ideal number of clusters for a given dataset in clustering algorithms like K-means. Finding the elbow point in the graph, when more clusters do not significantly increase the explained variance, involves graphing the explained variation versus the number of clusters. When the number of clusters is unknown beforehand, this technique aids in balancing the quality of fit and complexity of the model and finding the right number.

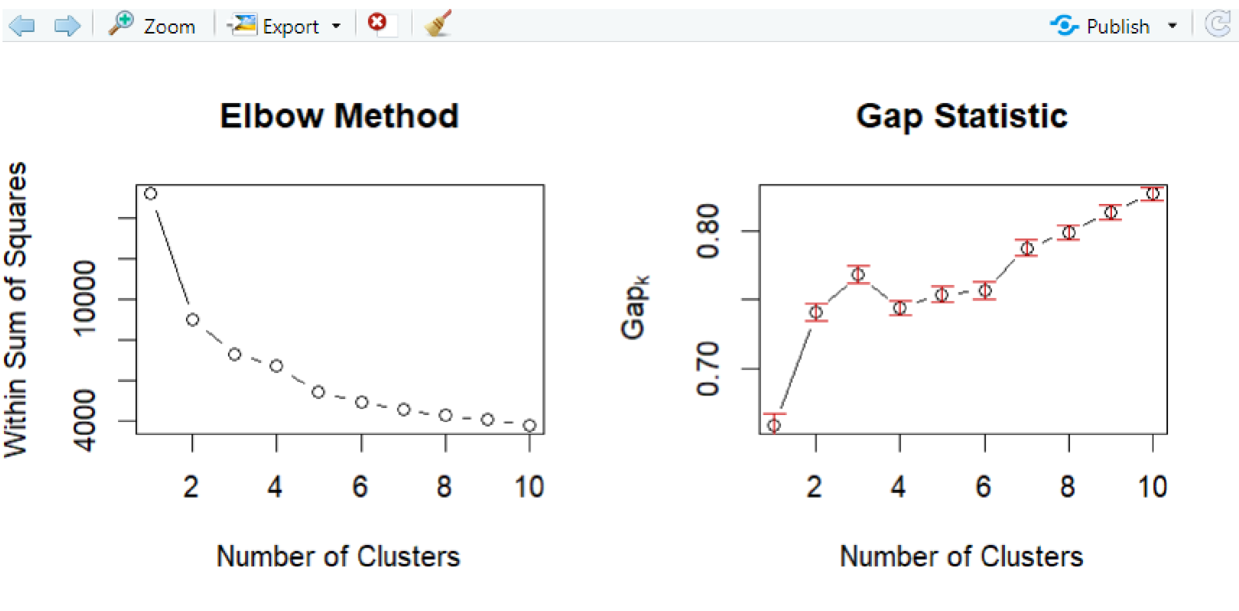
```
> table(nb_clusters$Best.n[1,])

 0  2  3  4  7  8 10 
 2 10  5  1  2  4  2 
> # Elbow method
> wss_ratio <- c()
> for (i in 1:10) {
+   set.seed(123)
+   kmeans_clusters <- kmeans(D_Scaled, centers = i, nstart = 10)
+   wss_ratio[i] <- kmeans_clusters$tot.withinss
+ }
> plot(1:10, wss_ratio, type = "b", main = "Elbow Method", xlab = "Number of Clusters", ylab = "Wit
hin Sum of Squares")
> 
```



Gap statics Method

The ideal number of clusters in a dataset for clustering algorithms can be found using gap statistics, a statistical technique. It contrasts the actual data's within-cluster variation with the variation anticipated by a null reference distribution. Applying a clustering technique on the dataset for a range of cluster numbers, computing the within-cluster variation, and creating a set of reference datasets with a random distribution are the steps required to employ this method. The gap statistic is then derived as the difference between the predicted variance of the reference datasets and the logarithm of the within-cluster variation of the actual data. The number of clusters that maximizes the gap statistic is considered to be the ideal number of clusters.

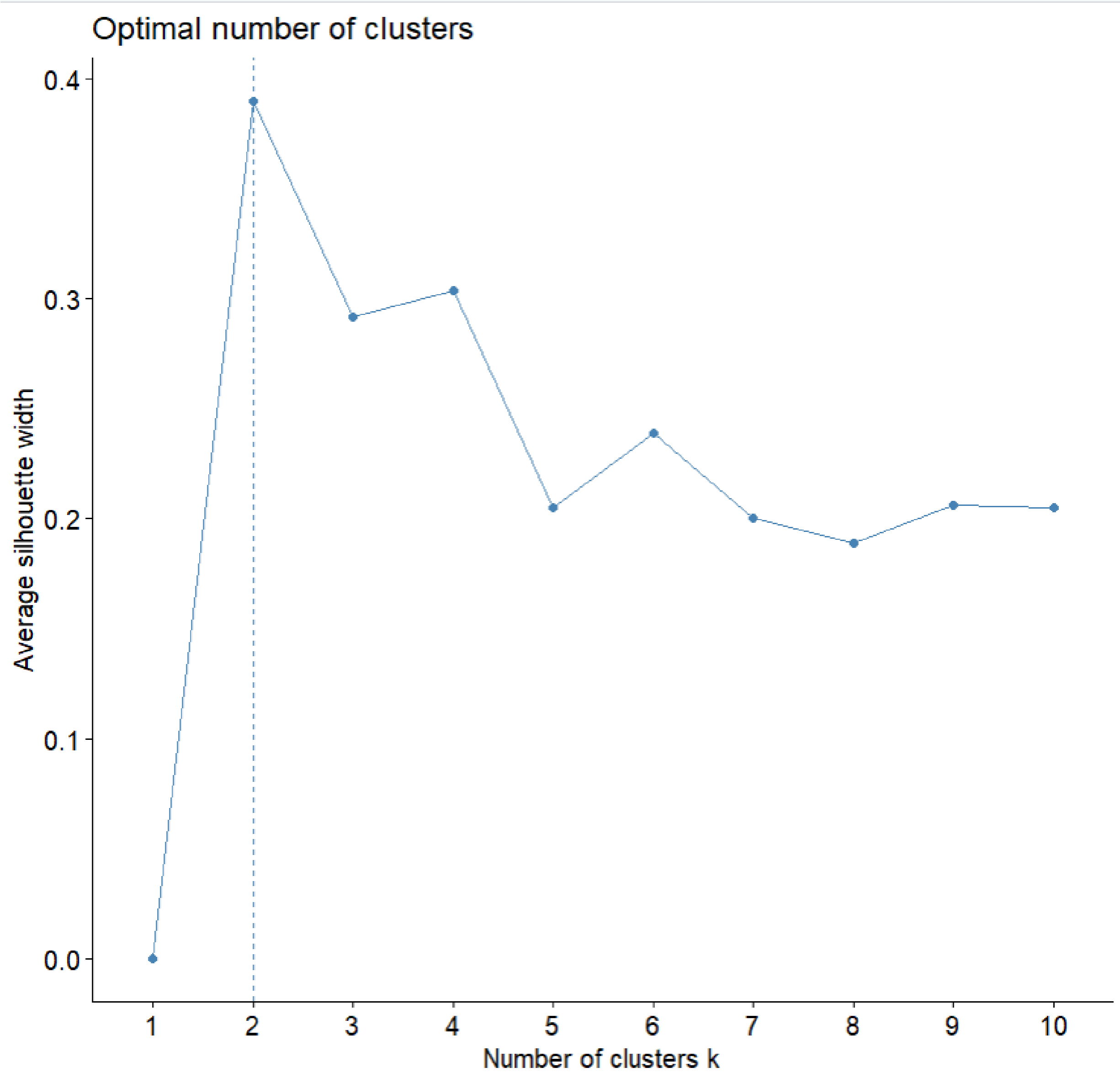
```
> # Gap statistics
> set.seed(123)
> gap <- clusGap(D_Scaled, FUN = kmeans, nstart = 2, K.max = 10, B = 100)
Clustering k = 1,2,..., K.max (= 10): .. done
Bootstrapping, b = 1,2,..., B (= 100)  [one "." per sample]:
.................................................. 50
.................................................. 100
> plot(gap, main = "Gap Statistic", xlab = "Number of Clusters")
> 
```



Silhouette method

A approach for assessing the quality of clusters in a dataset is the Silhouette method. For each object, it determines a Silhouette coefficient, which assesses how well the object fits into its designated cluster in relation to other clusters. The ideal number of clusters is then determined by calculating the average Silhouette coefficient for all items for each cluster number. This approach can identify overlapping or improperly separated clusters and offers a more thorough evaluation of cluster quality.

```
> # Silhouette method
> sil_ratio <- c()
> fviz_nbclust(D_Scaled, kmeans, method = 'silhouette')
> |
```



optimal number of clusters

```
> set.seed(123)
> kmeans_clusters <- kmeans(D_Scaled, centers = 2, nstart = 25)
> kmeans_summary <- as.data.frame(kmeans_clusters$centers)
> kmeans_summary$cluster <- c("Cluster 1", "Cluster 2")
> D_clustered <- datasheet %>% mutate(cluster = as.factor(kmeans_clusters$cluster))
> 
```

Evaluation metrics

```
> # Evaluation metrics
> BSS <- sum(kmeans_clusters$betweenss)
> TSS <- sum(kmeans_clusters$totss)
> WSS <- sum(kmeans_clusters$tot.withinss)
> cat("Ratio of BSS to TSS:", BSS/TSS, "\n")
Ratio of BSS to TSS: 0.4110022
> cat("BSS:", BSS, "\n")
BSS: 6251.343
> cat("WSS:", WSS, "\n")
WSS: 8958.657
> 
```

Silhouette plot

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

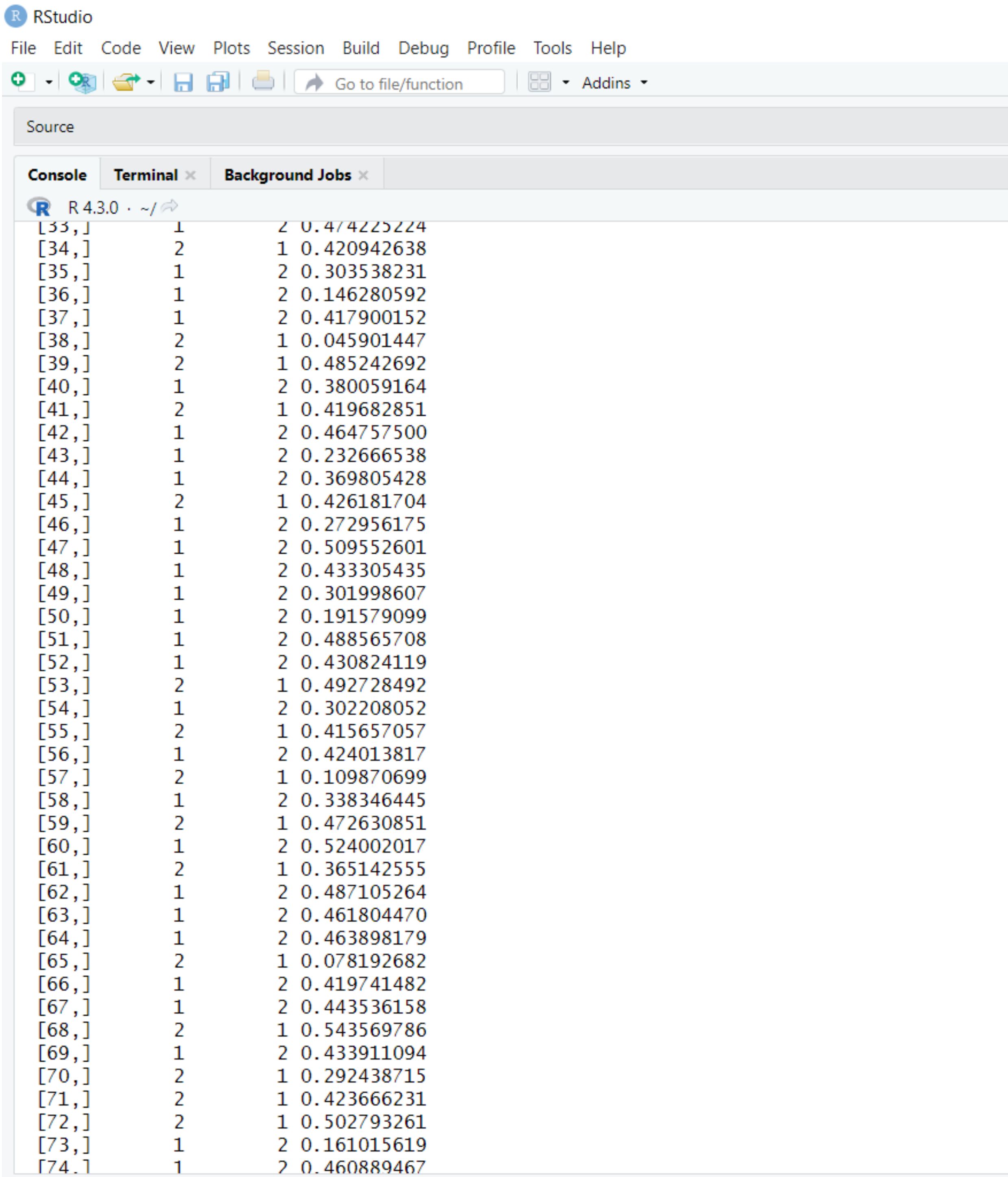
Go to file/function   Addins

Source

Console   Terminal   Background Jobs

R 4.3.0 · ~/

```
BSS: 6251.343
> cat("WSS:", WSS, "\n")
WSS: 8958.657
> # Silhouette plot
> silhouette(kmeans_clusters$cluster, dist(D_Scaled))
      cluster neighbor   sil_width
 [1,]       1        2 0.140451608
 [2,]       1        2 0.461810445
 [3,]       2        1 0.488575080
 [4,]       1        2 0.394383678
 [5,]       1        2 0.014286921
 [6,]       2        1 0.403942965
 [7,]       1        2 0.282578045
 [8,]       1        2 0.465562093
 [9,]       1        2 0.457076992
[10,]       2        1 0.078164075
[11,]       1        2 0.510528647
[12,]       1        2 0.462731287
[13,]       1        2 0.365459685
[14,]       1        2 0.453210784
[15,]       2        1 0.076725970
[16,]       2        1 0.462599423
[17,]       1        2 0.488233428
[18,]       1        2 0.110178692
[19,]       2        1 0.536817836
[20,]       2        1 0.513287875
[21,]       1        2 0.399807555
[22,]       1        2 0.486713032
[23,]       1        2 0.334102310
[24,]       1        2 0.508220687
[25,]       2        1 0.474723652
[26,]       1        2 0.426657813
[27,]       1        2 0.493044366
[28,]       2        1 0.513509277
[29,]       2        1 0.176125423
[30,]       1        2 0.467719919
[31,]       1        2 0.386814008
[32,]       1        2 0.434675999
[33,]       1        2 0.474225224
[34,]       2        1 0.420942638
[35,]       1        2 0.303538231
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console
Terminal
Background Jobs
R 4.3.0 · ~/
[33,] 1 2 0.474225224
[34,] 2 1 0.420942638
[35,] 1 2 0.303538231
[36,] 1 2 0.146280592
[37,] 1 2 0.417900152
[38,] 2 1 0.045901447
[39,] 2 1 0.485242692
[40,] 1 2 0.380059164
[41,] 2 1 0.419682851
[42,] 1 2 0.464757500
[43,] 1 2 0.232666538
[44,] 1 2 0.369805428
[45,] 2 1 0.426181704
[46,] 1 2 0.272956175
[47,] 1 2 0.509552601
[48,] 1 2 0.433305435
[49,] 1 2 0.301998607
[50,] 1 2 0.191579099
[51,] 1 2 0.488565708
[52,] 1 2 0.430824119
[53,] 2 1 0.492728492
[54,] 1 2 0.302208052
[55,] 2 1 0.415657057
[56,] 1 2 0.424013817
[57,] 2 1 0.109870699
[58,] 1 2 0.338346445
[59,] 2 1 0.472630851
[60,] 1 2 0.524002017
[61,] 2 1 0.365142555
[62,] 1 2 0.487105264
[63,] 1 2 0.461804470
[64,] 1 2 0.463898179
[65,] 2 1 0.078192682
[66,] 1 2 0.419741482
[67,] 1 2 0.443536158
[68,] 2 1 0.543569786
[69,] 1 2 0.433911094
[70,] 2 1 0.292438715
[71,] 2 1 0.423666231
[72,] 2 1 0.502793261
[73,] 1 2 0.161015619
[74,] 1 2 0.460889467

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

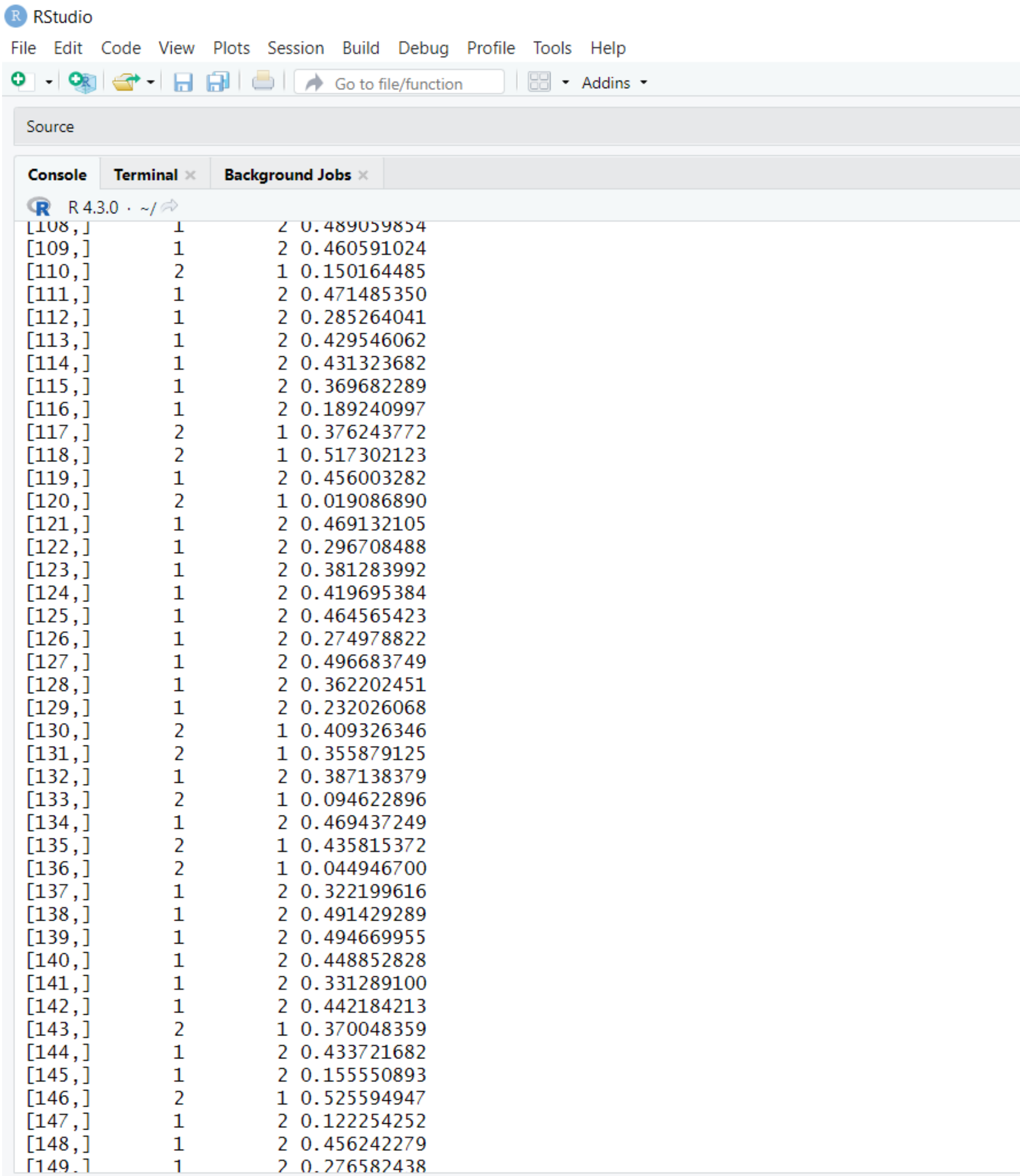
Go to file/function Addins

Source

Console Terminal × Background Jobs ×

R 4.3.0 · ~/

```
 [71,]        2         1 0.423666231
 [72,]        2         1 0.502793261
 [73,]        1         2 0.161015619
 [74,]        1         2 0.460889467
 [75,]        1         2 0.378094323
 [76,]        2         1 0.445340604
 [77,]        1         2 0.301219528
 [78,]        1         2 0.465171488
 [79,]        2         1 0.458852688
 [80,]        1         2 0.379879286
 [81,]        1         2 0.400101889
 [82,]        2         1 0.448969592
 [83,]        1         2 0.484386713
 [84,]        1         2 0.436690449
 [85,]        1         2 0.395254261
 [86,]        2         1 0.416592527
 [87,]        1         2 0.454464016
 [88,]        1         2 0.525811771
 [89,]        1         2 0.167909334
 [90,]        1         2 0.491322800
 [91,]        2         1 0.552114318
 [92,]        1         2 0.329263238
 [93,]        2         1 0.478089834
 [94,]        1         2 0.203998020
 [95,]        1         2 0.463642158
 [96,]        2         1 0.493051447
 [97,]        1         2 0.413445337
 [98,]        1         2 0.500004362
 [99,]        2         1 0.333117951
[100,]        1         2 0.428610439
[101,]        1         2 0.004580152
[102,]        1         2 0.334162411
[103,]        1         2 0.468550756
[104,]        1         2 0.492972916
[105,]        2         1 0.177570099
[106,]        2         1 0.559430635
[107,]        2         1 0.275413411
[108,]        1         2 0.489059854
[109,]        1         2 0.460591024
[110,]        2         1 0.150164485
[111,]        1         2 0.471485350
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Source
Console
Terminal
Background Jobs
R 4.3.0 · ~/
[108,] 1 2 0.489059854
[109,] 1 2 0.460591024
[110,] 2 1 0.150164485
[111,] 1 2 0.471485350
[112,] 1 2 0.285264041
[113,] 1 2 0.429546062
[114,] 1 2 0.431323682
[115,] 1 2 0.369682289
[116,] 1 2 0.189240997
[117,] 2 1 0.376243772
[118,] 2 1 0.517302123
[119,] 1 2 0.456003282
[120,] 2 1 0.019086890
[121,] 1 2 0.469132105
[122,] 1 2 0.296708488
[123,] 1 2 0.381283992
[124,] 1 2 0.419695384
[125,] 1 2 0.464565423
[126,] 1 2 0.274978822
[127,] 1 2 0.496683749
[128,] 1 2 0.362202451
[129,] 1 2 0.232026068
[130,] 2 1 0.409326346
[131,] 2 1 0.355879125
[132,] 1 2 0.387138379
[133,] 2 1 0.094622896
[134,] 1 2 0.469437249
[135,] 2 1 0.435815372
[136,] 2 1 0.044946700
[137,] 1 2 0.322199616
[138,] 1 2 0.491429289
[139,] 1 2 0.494669955
[140,] 1 2 0.448852828
[141,] 1 2 0.331289100
[142,] 1 2 0.442184213
[143,] 2 1 0.370048359
[144,] 1 2 0.433721682
[145,] 1 2 0.155550893
[146,] 2 1 0.525594947
[147,] 1 2 0.122254252
[148,] 1 2 0.456242279
[149,] 1 2 0.276582438

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
[146,]        2        1 0.525594947
[147,]        1        2 0.122254252
[148,]        1        2 0.456242279
[149,]        1        2 0.276582438
[150,]        1        2 0.424262592
[151,]        2        1 0.343680514
[152,]        2        1 0.075498621
[153,]        1        2 0.331205293
[154,]        2        1 0.469452083
[155,]        1        2 0.504191996
[156,]        2        1 0.440397581
[157,]        1        2 0.425403218
[158,]        1        2 0.518724039
[159,]        1        2 0.415487660
[160,]        1        2 0.348510777
[161,]        1        2 0.454084816
[162,]        2        1 0.394378728
[163,]        1        2 0.241499338
[164,]        1        2 0.541857399
[165,]        2        1 0.477263928
[166,]        2        1 0.439851339
[167,]        1        2 0.340956674
[168,]        2        1 0.423592387
[169,]        1        2 0.424777936
[170,]        1        2 0.477253114
[171,]        2        1 0.487139609
[172,]        2        1 0.384637548
[173,]        1        2 0.345200401
[174,]        2        1 0.393065445
[175,]        1        2 0.235571935
[176,]        1        2 0.357558274
[177,]        1        2 0.172617800
[178,]        1        2 0.433508496
[179,]        1        2 0.389664149
[180,]        1        2 0.489103350
[181,]        2        1 0.495778234
[182,]        1        2 0.491251577
[183,]        1        2 0.374673773
[184,]        1        2 0.477128876
[185,]        2        1 0.448241783
[186,]        1        2 0.084308552
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
[183,]         1         2 0.374673773
[184,]         1         2 0.477128876
[185,]         2         1 0.448241783
[186,]         1         2 0.084308552
[187,]         1         2 0.216363384
[188,]         1         2 0.261491324
[189,]         2         1 0.520327400
[190,]         1         2 0.408165141
[191,]         2         1 0.164845086
[192,]         1         2 0.453800682
[193,]         1         2 0.395642539
[194,]         2         1 0.453357549
[195,]         1         2 0.391921475
[196,]         1         2 0.460164189
[197,]         2         1 0.482655250
[198,]         1         2 0.418018168
[199,]         1         2 0.409295225
[200,]         1         2 0.448006071
[201,]         1         2 0.264847658
[202,]         1         2 0.391236808
[203,]         2         1 0.452889420
[204,]         2         1 0.540261835
[205,]         1         2 0.463660143
[206,]         1         2 0.391735434
[207,]         1         2 0.266028846
[208,]         1         2 0.435468428
[209,]         1         2 0.442222712
[210,]         2         1 0.440375686
[211,]         1         2 0.165037730
[212,]         1         2 0.529413108
[213,]         1         2 0.452408524
[214,]         2         1 0.342907322
[215,]         1         2 0.513390329
[216,]         1         2 0.356841329
[217,]         1         2 0.439943322
[218,]         2         1 0.525104005
[219,]         1         2 0.468255634
[220,]         1         2 0.444937319
[221,]         2         1 0.499102761
[222,]         1         2 0.356988401
[223,]         2         1 0.246551205
[224,]         1         2 0.427629913
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Source

Console Terminal Background Jobs

R 4.3.0 · ~/

```
[221,]         2         1 0.499102761
[222,]         1         2 0.356988401
[223,]         2         1 0.246551205
[224,]         1         2 0.427629913
[225,]         1         2 0.540339103
[226,]         1         2 0.434217442
[227,]         2         1 0.492249936
[228,]         1         2 0.485716700
[229,]         2         1 0.515214600
[230,]         1         2 0.485147639
[231,]         1         2 0.463130430
[232,]         1         2 0.436980095
[233,]         1         2 0.417894380
[234,]         2         1 0.478227879
[235,]         1         2 0.476581580
[236,]         1         2 0.275473611
[237,]         1         2 0.281102133
[238,]         1         2 0.426663548
[239,]         2         1 0.542894269
[240,]         1         2 0.400204515
[241,]         1         2 0.386822002
[242,]         1         2 0.281142168
[243,]         1         2 0.450901436
[244,]         2         1 0.539761476
[245,]         1         2 0.350704691
[246,]         1         2 0.489289116
[247,]         1         2 0.329806180
[248,]         1         2 0.307689268
[249,]         2         1 0.551347522
[250,]         1         2 0.502776398
[251,]         1         2 0.481886928
[252,]         2         1 0.526828587
[253,]         1         2 0.390958752
[254,]         1         2 0.391671963
[255,]         1         2 0.278598249
[256,]         2         1 0.437145691
[257,]         1         2 0.434920078
[258,]         1         2 0.352803563
[259,]         2         1 0.417743180
[260,]         2         1 0.498398484
[261,]         1         2 0.503301414
```

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

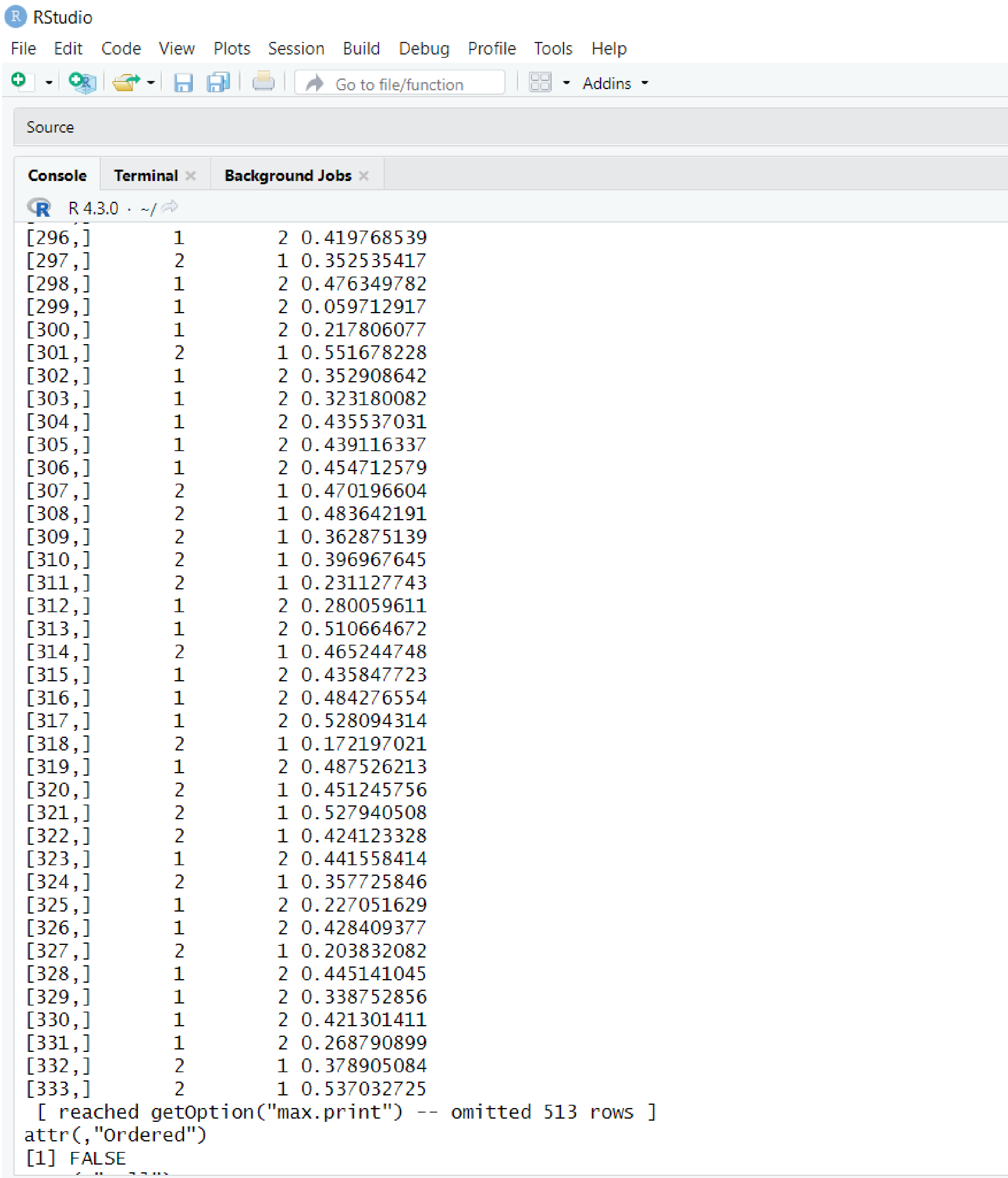
Go to file/function   Addins

Source

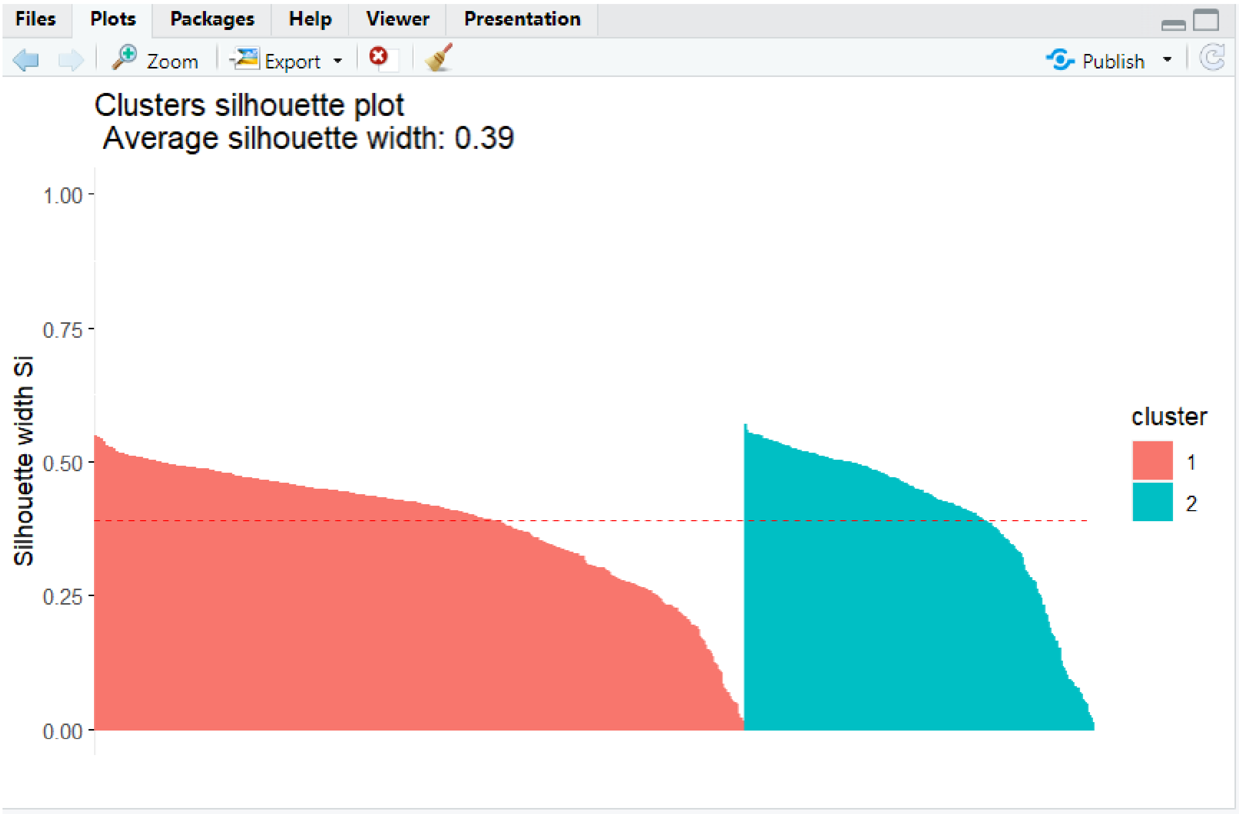
Console   Terminal   Background Jobs

R 4.3.0 · ~/

```
[258,]         1         2 0.352803563
[259,]         2         1 0.417743180
[260,]         2         1 0.498398484
[261,]         1         2 0.503301414
[262,]         1         2 0.200108255
[263,]         1         2 0.505291922
[264,]         1         2 0.459729742
[265,]         2         1 0.274366733
[266,]         1         2 0.447904278
[267,]         1         2 0.435832636
[268,]         1         2 0.055512964
[269,]         1         2 0.460515258
[270,]         1         2 0.500299997
[271,]         1         2 0.408723872
[272,]         2         1 0.054383234
[273,]         1         2 0.296245126
[274,]         1         2 0.337553646
[275,]         2         1 0.424302188
[276,]         1         2 0.373227204
[277,]         1         2 0.452064044
[278,]         1         2 0.514519117
[279,]         2         1 0.503506108
[280,]         1         2 0.370580996
[281,]         2         1 0.078812427
[282,]         1         2 0.480591666
[283,]         1         2 0.410125453
[284,]         2         1 0.469347935
[285,]         1         2 0.352702746
[286,]         1         2 0.067632704
[287,]         1         2 0.524061513
[288,]         1         2 0.420278234
[289,]         2         1 0.376155681
[290,]         1         2 0.452031221
[291,]         1         2 0.505806264
[292,]         2         1 0.012467862
[293,]         1         2 0.460333897
[294,]         1         2 0.471458283
[295,]         2         1 0.423246900
[296,]         1         2 0.419768539
[297,]         2         1 0.352535417
[298,]         1         2 0.476349782
[299,]         1         2 0.059712917
```




```
[296,]        1        2 0.419768539
[297,]        2        1 0.352535417
[298,]        1        2 0.476349782
[299,]        1        2 0.059712917
[300,]        1        2 0.217806077
[301,]        2        1 0.551678228
[302,]        1        2 0.352908642
[303,]        1        2 0.323180082
[304,]        1        2 0.435537031
[305,]        1        2 0.439116337
[306,]        1        2 0.454712579
[307,]        2        1 0.470196604
[308,]        2        1 0.483642191
[309,]        2        1 0.362875139
[310,]        2        1 0.396967645
[311,]        2        1 0.231127743
[312,]        1        2 0.280059611
[313,]        1        2 0.510664672
[314,]        2        1 0.465244748
[315,]        1        2 0.435847723
[316,]        1        2 0.484276554
[317,]        1        2 0.528094314
[318,]        2        1 0.172197021
[319,]        1        2 0.487526213
[320,]        2        1 0.451245756
[321,]        2        1 0.527940508
[322,]        2        1 0.424123328
[323,]        1        2 0.441558414
[324,]        2        1 0.357725846
[325,]        1        2 0.227051629
[326,]        1        2 0.428409377
[327,]        2        1 0.203832082
[328,]        1        2 0.445141045
[329,]        1        2 0.338752856
[330,]        1        2 0.421301411
[331,]        1        2 0.268790899
[332,]        2        1 0.378905084
[333,]        2        1 0.537032725
 [ reached getOption("max.print") -- omitted 513 rows ]
attr(,"Ordered")
[1] FALSE
```


Files
Plots
Packages
Help
Viewer
Presentation
Zoom
Export
Publish
Clusters silhouette plot
Average silhouette width: 0.39
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2

Dimensionality Reduction with PCA

```
R 4.3.0 · ~/
> # Perform PCA
> D_quality <- Data_N_outliers
> D_pca <- prcomp(D_Scaled)
> summary(D_pca)
Importance of components:
                          PC1    PC2    PC3     PC4     PC5     PC6     PC7     PC8     PC9
Standard deviation     3.0705 1.7386 1.3779 1.08728 0.95400 0.73033 0.59669 0.46989 0.39755
Proportion of Variance 0.5238 0.1679 0.1055 0.06568 0.05056 0.02963 0.01978 0.01227 0.00878
Cumulative Proportion  0.5238 0.6917 0.7972 0.86287 0.91344 0.94307 0.96285 0.97511 0.98389
                          PC10   PC11    PC12    PC13    PC14    PC15    PC16    PC17    PC18
Standard deviation     0.30218 0.2508 0.20940 0.18721 0.14610 0.12646 0.11355 0.07870 0.01904
Proportion of Variance 0.00507 0.0035 0.00244 0.00195 0.00119 0.00089 0.00072 0.00034 0.00002
Cumulative Proportion  0.98897 0.9925 0.99490 0.99685 0.99803 0.99892 0.99964 0.99998 1.00000
> D_pca.transform <- as.data.frame(-D_pca$x[, 1:10])
> D_pca.transform
           PC1          PC2          PC3         PC4          PC5          PC6         PC7
1   0.33008671 -0.214358296  -0.996691160  0.17188815  0.082605091 -0.726465844 -0.91461297
2  -1.59322354 -0.421883157   0.368657675  0.23214677  0.691903545 -0.528331092  0.35646058
3   3.75905753  0.187893015  -0.088626688  1.20046225  0.718805359  0.704361537 -0.01030921
4  -1.74165971 -2.821701010  -0.112458943  0.37385893 -0.368148386 -0.506501111  0.45118004
5   0.55099709  4.765977863 -11.679347547  0.16643526  3.242122566 -0.298604457  2.66105177
6   6.39853584  3.925694674   2.067951954 -0.38404114 -0.571226699  0.899195611  0.77730594
7  -0.77669551 -2.203235795   0.123030908  2.10964871 -0.212144346  0.777780808 -0.12203096
8  -2.14105741 -1.166269567  -0.651455750  0.81825415 -1.242264794 -0.657561654 -0.16034312
9  -4.45658043 -3.086803124  -0.104909036 -0.55053903 -0.571267072 -0.304295061  0.26871082
10  1.17001772 -1.996724845  -0.175795805 -0.67893639 -0.445301995 -0.140904054  0.48855979
11 -3.49477750 -1.758952296  -0.291257851 -0.42223407 -0.684765229 -0.324841271  0.47611904
12 -4.38320711 -2.424415301   0.774102875 -1.30578452  0.850273985 -0.053265511  0.32667541
13 -0.82545521  0.029209348  -0.445542089  0.29574946 -0.570224147 -0.031753070 -1.31812479
14 -1.41227892 -0.018430581   0.119183043  0.14372362  0.505877154 -0.642632229  0.26870471
15  1.19863657 -0.861076457  -0.536371241  1.18511362 -1.399177178  0.828170988 -1.17255668
16  3.79704179  1.295440361  -0.172146344  1.02505361 -1.119105668 -0.132572652 -0.70562160
17 -5.27747851  1.737605710   0.549555431 -0.86021046  0.158967860 -0.104805385  0.78667883
18  0.34588877 -1.593432266  -0.500656283 -0.40875915 -0.798262118  1.697273095  0.25736716
19  4.15858451  1.155933921   0.405031590  0.11053613 -0.489808556 -0.528537997  0.20508500
20  4.21076898  1.059629390  -0.593377459  0.72816543 -0.863905083  0.111037589 -0.42879745
21 -1.37701833  1.188201912  -0.328298736  0.06760925 -1.663956716 -0.240602060 -0.80455415
22 -5.10479548  2.260031736  -0.070717396  0.58683746 -0.075240822  0.895736601  0.29751049
23 -1.15871891 -2.050370801  -0.595831487  1.70168523 -0.770586984  0.556565573 -0.42291384
24 -2.75681565 -0.853744312  -0.146857237  0.07871385 -1.501249949  0.016529997  0.34957582
25  3.85713290 -0.401090743  -0.555210410 -0.00318032 -1.797477794 -0.586226026 -0.24779055
```

R 4.3.0 · ~/

```
22  -5.10479548  2.260031736  -0.070717396  0.58683746 -0.075240822  0.895736601  0.29751049
23  -1.15871891 -2.050370801  -0.595831487  1.70168523 -0.770586984  0.556565573 -0.42291384
24  -2.75681565 -0.853744312  -0.146857237  0.07871385 -1.501249949  0.016529997  0.34957582
25   3.85713290 -0.401090743  -0.555210410 -0.00318032 -1.797477794 -0.586226026 -0.24779055
26  -1.44672712 -0.016461828  -0.607235427  0.25049711 -1.383243726 -0.900679099 -0.38963485
27  -4.82856111  2.351690378   0.041113514  0.22011134 -0.053902843  0.755735582  0.44069768
28   4.59574320  0.140063696  -0.018160246 -1.69690838  0.029185716 -0.662545258 -0.15978039
29   1.54126791 -0.290080767   0.089666219 -0.01503813 -0.626356138  1.754587627  0.65063381
30  -3.21085890  2.204527239   0.323430804 -0.79119315  1.408821394  0.478766777 -0.25139872
31  -0.89165586  0.306095790  -0.676669881  0.21776713 -0.256248534 -0.777402054 -0.08048598
32  -2.01951090 -2.173003600   0.430173510  0.96629311  0.220384861 -0.626962133 -0.17906560
33  -3.01178099 -2.404288303   0.469415437 -0.38692089 -0.241362103  0.550823147  0.62307919
34   3.91370687 -1.114955675  -0.287043371 -2.29904555  1.255243956  0.007728381 -0.79591582
35  -1.16809525 -2.771683406   0.099808788 -2.05155331 -0.158977981  0.132409835  0.03348118
36   0.16272752 -1.205959591  -0.082277260  2.20822795  1.069596502  0.046329835 -0.21946839
37  -2.24245205  2.912416275   0.548085272  0.11086620 -0.995190764 -0.493626740  0.20942363
38   3.05995499  2.233875180 -13.750911492 -0.88247688  1.419253060  0.098634574  0.66414896
39   4.02169457  1.022518756   0.931121336 -0.28752496 -1.168966043 -0.725503471  0.95967244
40  -1.57144876  1.362050724  -1.132814744  1.12867923 -0.870948672  0.574469589 -1.51989870
41   2.95681526  0.642281618  -0.502465105 -0.32278971 -0.592938009  0.843267642 -0.20773379
42  -5.32641418  2.399977400   0.650299672 -0.35624969  1.276910832  0.342270933  0.54301642
43  -0.21797670  1.405048146   0.375257980  0.78975425 -0.442687716 -1.595287861  0.58752127
44  -1.10340896 -1.214077014  -0.234077691 -0.36146137 -0.743673233  1.030031494  0.68796843
45   5.23303708 -1.012502417   0.297129596  3.06444310  0.914277904  0.491685928  0.80860658
46  -0.39967124  1.464123755  -0.215137888  0.38641024 -0.928216071 -1.215021580  0.49150793
47  -2.76556752  0.007220952   0.384999187  0.06867854 -0.948463332 -1.067350852  0.38433741
48  -2.57854760  3.113261106   0.428419644 -0.41196628  0.557721669  0.231714003 -0.08065196
49  -0.48881672  1.186709672  -0.573122060  0.35013351  0.158593994 -1.305313378 -0.51150869
50  -0.13789740 -2.308104548   0.157813233 -0.18906932 -1.549805820 -0.913210015 -0.09373896
51  -3.85656480  2.099877910   0.764985351 -0.44175241 -0.822830104  0.117635631  0.42745909
52  -1.79957607 -0.916236926  -0.375539120 -1.56615340  0.495838111  0.712755576 -0.14948658
53   4.52401983  0.591050179  -0.732275575  0.09792532 -1.748703054 -0.011290308 -0.96589101
54  -1.11048194 -3.050015888  -0.121515624  0.72346585 -0.449537427  0.518720546 -0.17764643
55   6.03158515  3.302342949   1.811665156 -1.93081742  0.542330414  0.742002904  0.06914396
56  -3.10554406 -3.214139749   0.084175871  1.24413524 -0.351468069  0.158282871  0.90541923
57   1.20555089 -1.057956986  -0.005386846 -0.26526507 -0.813771679  0.418833766  0.53224650
58  -0.83818681  1.472776142   0.304435226  0.35397832 -0.150071093 -1.621870681  0.28661878
59   4.16787231 -0.639576293  -0.369489903  0.29930007 -2.094560207 -0.190331897  0.53877507
60  -4.02339031  1.397150443   1.006422303  0.37592917  0.328092146 -0.118339042  0.24666807
61   2.81501708 -1.895783141   0.104495261  0.67678642 -1.227315676 -0.024223560  0.34241859
62  -4.19759767  1.199638415   0.251300681 -0.15382439  1.893740600  0.471896506 -0.96468929
63  -3.03331630  1.778701635   0.908529352  0.26521327  1.230880571 -0.003856559  1.20789033
```

R 4.3.0 · ~/

```
66  -2.22085752  1.104624631   0.648634279  1.76364252  1.524861523  0.778923932  0.29410453
67  -2.40270691  2.713388023   0.086259703 -0.60599310  0.756494408 -0.463466705 -0.52520280
68   4.90976395  0.808828345  -0.575225081 -0.07168510 -0.308766165 -0.201262609 -0.61504679
69  -1.59876007 -0.707764659  -1.008164157 -0.17924636 -0.966867569 -0.855189542 -0.79905018
70   2.28994955 -0.352886739  -1.040916047  0.31311230 -1.203818894  1.172899890 -0.91260299
71   3.59690171  1.949081382   0.906397048  0.20266625 -1.093589232 -1.022034861  0.37242155
72   5.49324077  0.282838004  -0.360273159 -0.13751982  0.860814076 -0.990898835 -1.49246777
73   0.09878185 -1.524073223   0.065141386 -1.72915890  0.101631400  0.441430330  0.06649150
74  -5.27030626  2.760873997   0.188779779 -1.24916793 -0.007170271 -0.396001063 -0.45776668
75  -1.16510620 -0.405484304  -0.120519669 -0.17609162  1.646936376 -0.861766181 -0.18608130
76   3.02744946 -0.299999575   0.113507326  0.71785593 -1.091370898 -0.306891956 -0.10004913
77  -0.63482241 -0.018371674  -0.008139357 -1.24077517 -0.998037958  0.706607832  0.70721815
78  -2.67999494  2.364083891  -0.459893688 -0.13333343 -0.881151553  0.667303743  0.03541521
79   3.72504704  0.555840978   0.389952118  1.25426491 -0.380326512  1.563782077 -0.32655804
80  -2.15450951  3.557778097   0.744276184  0.48006264 -0.004762879 -0.174856548  0.48520682
81  -1.37238585  1.055198125  -0.425374477  0.64552976 -1.611948328  0.078648883 -0.78191495
82   3.37635619  1.058748050   0.274331638 -0.21352922 -0.718524450  1.288261979 -0.52670223
83  -3.83606125 -1.273252347   0.659376088  0.08861005 -1.302099094 -0.072939616  1.50673279
84  -2.64501635  0.865119627   0.605703814 -1.58725541  1.774483566 -0.538524839 -0.60664314
85  -1.53373322 -1.096394171  -1.564344846 -0.19406853 -0.912492694  1.052606945 -0.64458896
86   6.76702609  3.667714342   2.219229021 -0.17235008  0.246150690  1.061355556  0.68734665
87  -1.56911901 -0.343999529   0.173045700 -0.66651314  0.200129324 -0.944361241  0.07991113
88  -4.86546477  0.512946298   0.691108443 -0.23607198 -0.612290982 -0.311392987  0.41066857
89   0.05027942 -1.832498751  -1.140863209 -0.54855870 -0.938666173  1.216253049 -0.89267101
90  -4.98265862  2.299489520   0.272050225  0.28523393  0.381047960  0.490568076  0.31565849
91   4.95427838  0.202624867  -0.515643197  0.04640846 -1.032716695 -0.416902950 -0.26729233
92  -0.83209894 -0.605699876   0.702429753 -0.40671715  0.728234559  0.687389924  0.94516029
93   3.35707158 -0.055363119   0.715399574  0.96228200  0.050905565 -0.064463549 -0.01964542
94   0.06151440  0.790955692  -0.476361026 -0.09142817 -0.054420715 -1.163917489 -0.39380543
95  -2.07767545  1.306014243  -0.381956564  0.32313804 -1.323338519 -0.015416223 -0.85538971
96   3.99869127 -0.011584744  -0.004362434 -1.38395733  0.959126309  0.058899592 -0.76115842
97  -1.82752543  0.663529572   0.369236763  1.23893022  1.736389416  0.597026452 -0.64425562
98  -1.85177326  0.599374706  -0.221160988 -0.04954649 -0.545926744  0.158206871 -0.33164709
99   3.01390385 -1.320592191  -0.287997344 -2.35785450  1.197497615  0.109995104 -1.18602002
100 -2.40048449  3.163859481   0.269833393 -0.40680567  0.090163300  0.065347115 -0.07917751
             PC8          PC9         PC10
1   -0.400214338 -0.874065893 -0.097489202
2    0.251006040  0.129983525  0.089554116
3    0.466635984 -0.317045735  0.470495718
4   -0.018486013 -0.035221243 -0.459144754
5   -0.404142379  2.100507107  0.362529057
```

R 4.3.0 · ~/

```
2   [illegible]
3    0.466635984 -0.317045735  0.470495718
4   -0.018486013 -0.035221243 -0.459144754
5   -0.404142379  2.100507107  0.362529057
6   -0.184232335  0.968592826  0.426800687
7   -0.406425878  0.341578800  0.066457130
8   -0.423144489  0.004864284  0.059177489
9   -0.378198246  0.418893513 -0.206113855
10   0.992707363  0.193445702 -0.273979262
11   0.283220367  0.139816267 -0.025911626
12  -0.398428701  0.098556980  0.505846716
13  -0.201106667 -0.054330477  0.246228901
14   0.538564946  0.067047703  0.080985202
15  -0.506002548  0.394362031  0.004514205
16   0.483798790  0.201751661  0.550348072
17  -0.871223642 -0.598050702  0.043534610
18  -0.472160626  0.165223037 -0.344163719
19  -0.214810847 -0.174878534 -0.110745245
20   0.074761317 -0.435128171 -0.032971009
21   0.199559980  0.081197653  0.090651008
22   0.031802073 -0.644474478 -0.124904179
23  -0.678632926  0.486455260  0.012001102
24   0.223424299  0.155785147 -0.136415456
25   0.303315616  0.004822282  0.472309690
26   0.334984698  0.092030653  0.040924559
27   0.212316913 -0.358126325 -0.069027920
28  -0.819700206 -0.319421184  0.125335986
29  -0.372387379  0.212633648 -0.390817374
30   0.477125268  0.510855102 -0.070138785
31   0.506387389 -0.338862133  0.089266293
32   0.108719009  0.680084894  0.094872195
33  -0.143697282  0.021638624 -0.107231526
34   0.001553432 -0.144653925  0.436100119
35   0.706439989  0.599239225 -0.096617135
36   0.029517444 -0.709400987  0.300762015
37   0.389670053  0.552012682 -0.088999445
38  -0.874189275  0.643071178  0.567636806
39  -0.462071711  0.338696315  0.127473882
40   0.558913123 -0.347353822 -0.237484481
41   1.029713484 -0.173474234  0.243754429
42  -0.881897166 -0.485915652 -0.207248608
43  -0.045511305 -0.134901889  0.060133331
```

R 4.3.0 · ~/

```
46  -0.172235905 -0.085192746  0.106566638
47  -0.614871418 -0.099823738  0.251287892
48  -0.343786616  0.470526138 -0.002872660
49  -0.198231433 -0.178780198 -0.254356164
50  -0.090044196  0.884103999  0.244217106
51   0.941515700  0.430153150 -0.162974762
52  -0.065834996 -0.562713496 -0.226001415
53   0.116647642  0.006827807  0.463357994
54  -1.429132039  0.250817093  0.168469965
55   0.086381832  0.724768890 -0.407824563
56  -0.249828513  0.141460348 -0.350117829
57   0.833972071 -0.360571492  0.076075630
58   0.281308836 -0.165936045 -0.215037718
59   0.034698887 -0.578600513  0.407387723
60   0.307773462 -0.417758393 -0.015768001
61  -0.066027944  0.027758633 -0.055836373
62   0.483978629  0.037787753 -0.456535267
63  -0.553390339 -0.524138453  0.040193319
64   0.388791178 -0.294690415 -0.511054641
65   0.206716909  0.273801694 -0.029185133
66   0.758961352 -0.184745977 -0.088686787
67   0.098973697  0.300476127 -0.239286356
68   0.152687923 -0.069050874 -0.762986930
69  -0.043248769 -0.042145751  0.137750652
70  -0.565053368  0.092007197  0.100469158
71   0.185588382  0.755972574 -0.364935643
72   0.232372142 -0.434973037 -0.979674668
73   0.736098355 -0.006877523  0.042965762
74   0.396826275  0.072582877 -0.135708684
75   0.479727876 -0.028904811  0.089589596
76   0.174185610 -0.264983281  0.363796286
77   0.755237486  0.066882528  0.076253109
78   0.096852945 -0.133757143  0.235845105
79  -0.404773228  0.133134960 -0.078973508
80  -0.437475961  0.457192777  0.230503006
81   0.014497926 -0.199251472  0.239978873
82  -0.363347260  0.400436651 -0.110675665
83  -0.313487367 -0.208073930  0.060690747
84   0.877881780  0.205289689 -0.222131642
85   0.043709677 -0.430906320 -0.205181288
86  -0.267473209  0.564395476  0.200483617
87   0.128099318  0.170973983  0.146380018
```


85   0.043709677 -0.430906320 -0.205181288
86 -0.267473209 0.564395476 0.200483617
87   0.128099318 0.170973983 0.146380018


```
88  -0.275688572 -0.415958455  0.039636385
89   0.211610013 -0.121529113 -0.143601982
90   0.573201572 -0.313923339 -0.241439153
91  -0.056699583 -0.365312355  0.184004979
92   0.839280997  0.270990118  0.465741959
93  -0.073716851  0.239172452  0.903717839
94  -0.381188303 -0.565584935  0.032806248
95   0.126580977 -0.319181644  0.020387786
96   0.518848006  0.196283036  0.413013926
97   0.512552468 -0.470989556  0.166488244
98  -0.037498101  0.304608620 -0.148718765
99   0.383769873  0.121240996  0.497759928
100 -0.543689596  0.091948213 -0.131718380
 [ reached 'max' / getOption("max.print") -- omitted 746 rows ]
>
```

K2 cluster

```
> #k2 cluster
> k=2
> kmeans_clustered.pca = kmeans(D_pca.transform, centers = k, nstart = 10)
> kmeans_clustered.pca
K-means clustering with 2 clusters of sizes 295, 551

Cluster means:
        PC1         PC2         PC3         PC4          PC5         PC6         PC7
1  3.712440  0.05517768  0.09037270 -0.04544055  0.004748502  0.04182875  0.03018973
2 -1.987604 -0.02954159 -0.04838466  0.02432843 -0.002542301 -0.02239470 -0.01616328
          PC8         PC9        PC10
1 -0.007121053 -0.02886359  0.016831490
2  0.003812542  0.01545328 -0.009011415

Clustering vector:
  [1] 2 2 1 2 2 1 2 2 2 1 2 2 2 2 1 1 2 2 1 1 2 2 2 2 1 2 2 1 1 2 2 2 2 1 2 2 2 1 1 2 1 2 2 2 1
 [46] 2 2 2 2 2 2 2 1 2 1 2 1 2 1 2 1 2 2 2 1 2 2 1 2 1 1 1 2 2 2 1 2 2 1 2 2 1 2 2 2 1 2 2 2 2
 [91] 1 2 1 2 2 1 2 2 1 2 2 2 2 2 1 1 1 2 2 1 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 2 1 1 2 1 2 1
[136] 1 2 2 2 2 2 2 1 2 2 1 2 2 2 2 1 1 2 1 2 1 2 2 2 2 2 1 2 2 1 1 2 1 2 2 1 1 2 1 2 2 2 2 2 2
[181] 1 2 2 2 1 2 2 2 1 2 1 2 2 1 2 2 1 2 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 1 2 2 2 1 2 2 1 2 1 2 2
[226] 2 1 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 1 2 2 2 1 2 2 1 1 2 2 2 2 1 2 2 2 2 2
[271] 2 1 2 2 1 2 2 2 1 2 1 2 2 1 2 2 2 2 1 2 2 1 2 2 1 2 1 2 2 2 1 2 2 2 2 2 1 1 1 1 1 2 2 1 2
[316] 2 2 1 2 1 1 1 2 1 2 2 1 2 2 2 2 1 1 2 1 1 2 1 2 2 2 2 2 1 1 1 1 2 2 2 1 2 2 2 1 2 2 1 2 1
[361] 1 1 2 2 2 1 2 2 2 2 2 2 2 2 2 1 1 2 2 1 2 1 2 1 2 2 1 2 1 1 2 2 2 2 1 2 1 2 2 2 1 2 1 2 1
[406] 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 1 2 1 1 2 2 1 2 2 2 1 1 1 2 2 1 1 2 1 1 1 2
[451] 2 2 2 2 1 2 2 2 1 2 2 1 2 2 1 2 2 1 1 2 2 1 1 1 2 1 1 2 1 2 1 1 2 2 2 2 1 2 2 1 1 2 2 1 1
[496] 2 1 2 2 1 1 1 2 2 1 1 1 2 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2 2 1 2 1 1 2 2 1 1 1 2 2 2 1 2 1 1
[541] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 1 1 2 2 1 1 2 1 2 2 1 1 2 2 2 1 2 2 2 1 1 1 1 2 2 2 2 1 1 1
[586] 2 1 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 1 1 2 2 1 2 1 2 2 1 2
[631] 2 1 2 1 2 2 2 2 1 1 1 2 1 2 2 2 2 2 1 2 1 2 2 2 1 2 2 2 2 1 2 1 2 2 2 2 1 1 2 2 1 2 1 2 2
[676] 1 2 1 2 2 2 2 2 1 2 2 2 1 2 1 2 2 1 2 1 2 2 2 2 2 1 1 2 2 1 1 2 2 2 2 1 1 1 1 2 1 2 2 1 1
[721] 2 1 2 1 2 2 1 2 2 1 1 1 2 1 2 2 1 1 1 2 1 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 1 1 2 1 1
[766] 2 2 2 1 2 2 1 1 2 2 2 1 1 1 2 1 2 1 1 2 2 1 2 1 2 2 2 1 1 2 2 2 1 1 2 2 2 2 1 1 2 2 1 2 2
[811] 1 2 2 2 2 2 2 1 1 1 2 1 2 1 1 2 1 1 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 1 2 2
```

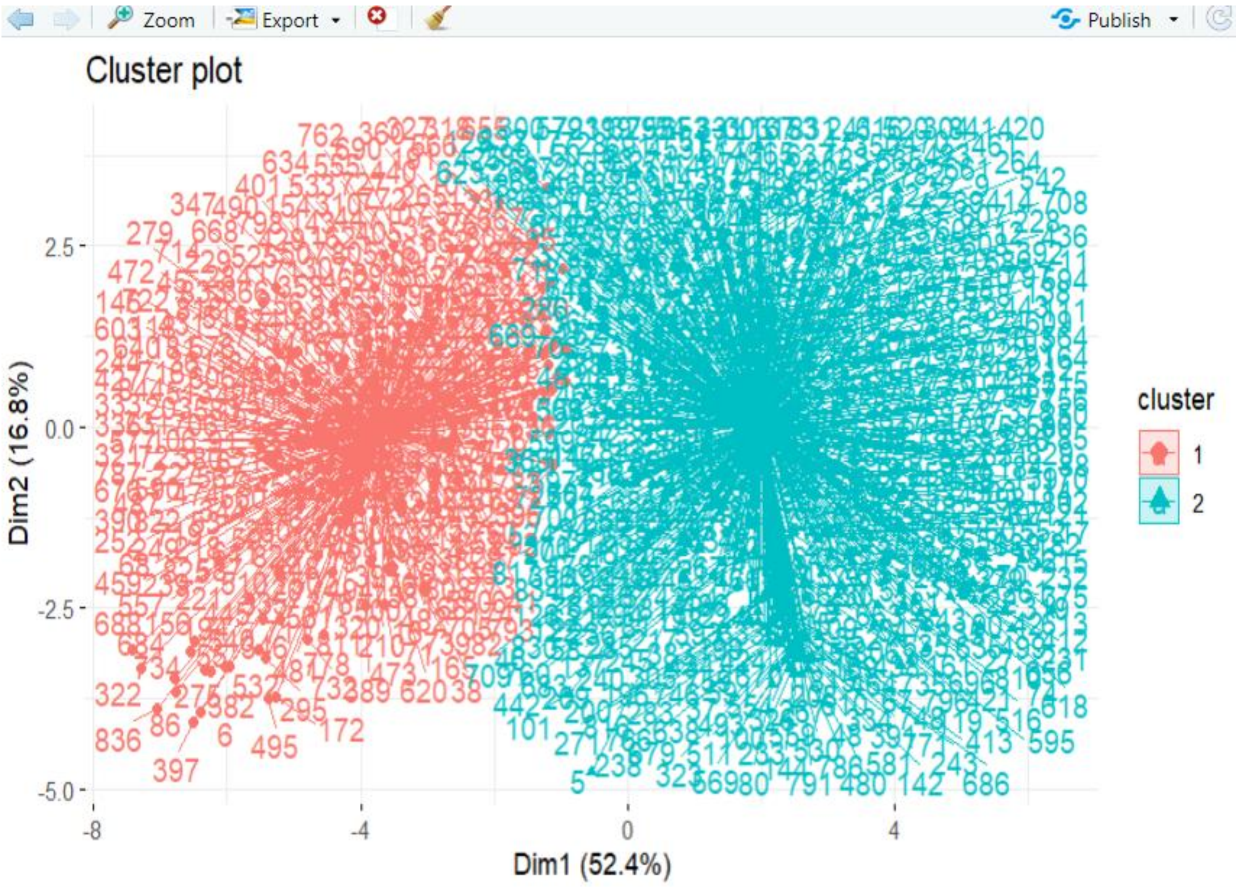
```
Within cluster sum of squares by cluster:
[1] 3113.806 5678.106
 (between_SS / total_SS =  41.6 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
>
```

Finding BSS and WSS

```
> #CM
> wss_ratio = kmeans_clustered.pca$tot.withinss
> bss = kmeans_clustered.pca$betweenss
> print(paste("Total within-cluster sum of square is", wss_ratio))
[1] "Total within-cluster sum of square is 8791.91164569894"
> print(paste("Between Sum of Squares is", bss))
[1] "Between Sum of Squares is 6250.27506960953"
>
```

Plot

```
> #plot
> par(mar = c(2, 2, 2, 2))
> fviz_cluster(kmeans_clustered.pca, data = D_Scaled, ellipse.type = "euclid", star.plot = TRUE, re
pel = TRUE, ggtheme = theme_minimal())
> |
```



Energy Forecasting Part

1st Subtask Objectives

A)

Electricity load forecasting is critical to assuring the reliability of power systems. Because of its capacity to describe complicated non-linear interactions between inputs and outputs, MLP models have been frequently employed in this domain. Defining the input vector for MLP models in electrical load forecasting, on the other hand, is a significant undertaking that necessitates careful consideration of several parameters.

The usage of autoregressive models is a standard method for defining the input vector. These models anticipate future load levels by using historical load values as inputs. This method has the benefit of capturing the temporal dependence between load levels, which is necessary for good forecasting. It does, however, presume that load levels follow a linear trend, which is not necessarily the case.

Another method for capturing periodicity in load data is to employ Fourier transformations. This entails breaking down the load data into its constituent frequency components and feeding them into the MLP model. This method has the benefit of capturing seasonal trends and periodic swings in load data. However, it assumes that the load data has a stationary frequency domain, which is not always the case.

Wavelet transformations have also been utilized to capture load data's temporal and frequency dependencies. The load data is decomposed into wavelet coefficients, which are then fed into the MLP model. This method has the benefit of capturing both short-term and long-term dependence in load data. However, it necessitates a significant amount of computation, which may be prohibitively expensive for large datasets.

To minimize the dimensionality of the input vector, principal component analysis (PCA) was also performed. This entails reducing the original input variables to a smaller collection of uncorrelated variables that explain the majority of the data variance. This method has the advantage of reducing the computational complexity of the MLP model and potentially improving its interpretability. It does, however, imply that the input variables are linearly connected, which is not always the case.

To summarize, there are numerous techniques for creating the input vector for MLP models in energy load forecasting, each with its own set of benefits and drawbacks. The technique of choosing should be determined by the unique characteristics of the load data as well as the needs of the forecasting activity. The trade-off between model accuracy and computational complexity, as well as the interpretability of the final model, should be carefully considered. More research is needed to compare the performance of these various approaches on various datasets and under various conditions.

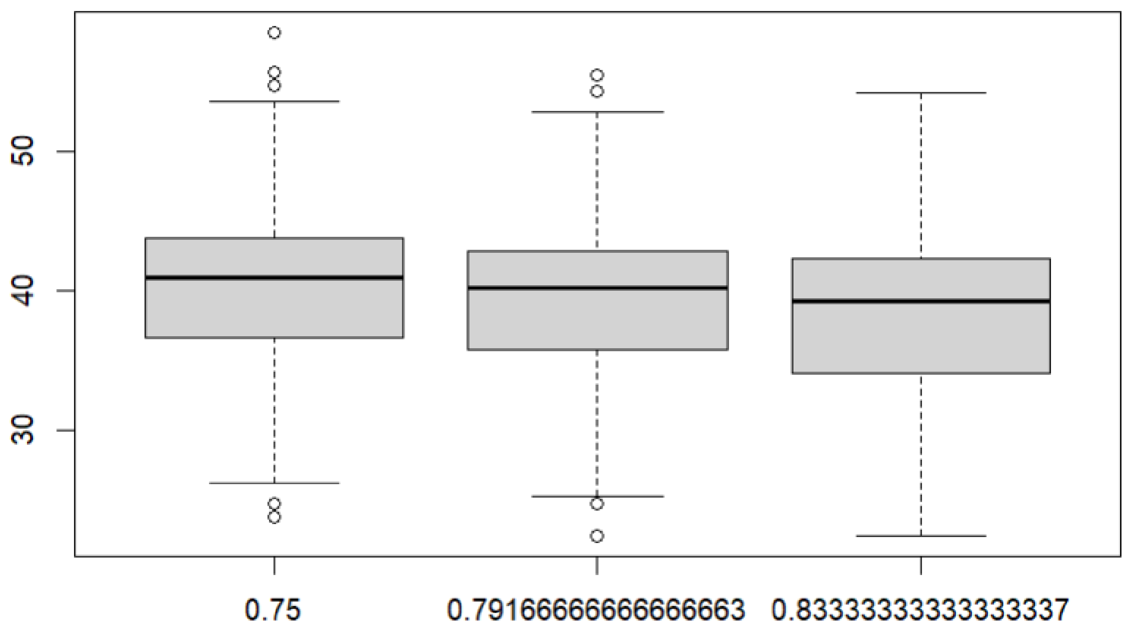
B)

Loading the dataset

```
R 4.3.0 · ~/
> #import libraries
> library(ggpubr)
> library(neuralnet)
> library(readxl)
> library(Metrics)
> Data_load <- read_excel("C:/Users/my pc/Desktop/ML cw/uow_consumption.xlsx")
> Data_load
# A tibble: 470 × 4
   date                `0.75` `0.791666666666663` `0.833333333333337`
   <dttm>               <dbl>                <dbl>                <dbl>
 1 2018-01-01 00:00:00   38.9                 38.9                 38.9
 2 2018-01-02 00:00:00   42.3                 41.9                 41.9
 3 2018-01-03 00:00:00   40.8                 40.5                 40.7
 4 2018-01-04 00:00:00   42.3                 41.9                 41.9
 5 2018-01-05 00:00:00   44                   44.1                 44
 6 2018-01-06 00:00:00   45.6                 44.5                 44.3
 7 2018-01-07 00:00:00   40.9                 41.2                 35.7
 8 2018-01-08 00:00:00   44.6                 44.3                 43.6
 9 2018-01-09 00:00:00   41.9                 42.9                 42.5
10 2018-01-10 00:00:00   44.9                 44.4                 43.9
# ℹ 460 more rows
# ℹ Use `print(n = ...)` to see more rows
> |
```

Summary of the dataset

```
> summary(Data_load)
      date                          0.75       0.791666666666663 0.833333333333337
 Min.   :2018-01-01 00:00:00   Min.   :23.80   Min.   :22.40     Min.   :22.40
 1st Qu.:2018-04-28 06:00:00   1st Qu.:36.60   1st Qu.:35.83     1st Qu.:34.10
 Median :2018-08-23 12:00:00   Median :40.90   Median :40.15     Median :39.20
 Mean   :2018-08-23 12:00:00   Mean   :40.18   Mean   :39.27     Mean   :38.36
 3rd Qu.:2018-12-18 18:00:00   3rd Qu.:43.80   3rd Qu.:42.77     3rd Qu.:42.30
 Max.   :2019-04-15 00:00:00   Max.   :58.50   Max.   :55.40     Max.   :54.20
> boxplot(Data_load[,-1]) #plot before normalizing data
> |
```

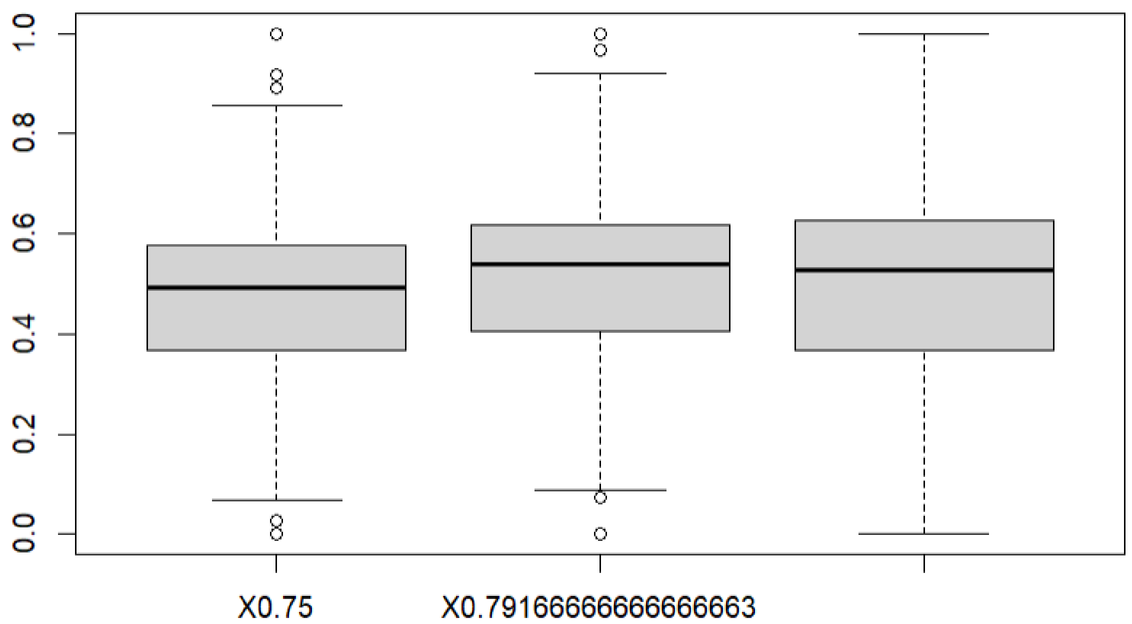


C)

Normalizing dataset

To guarantee that all input variables are on a same scale, data must be normalized before being used in a Multi-Layer Perceptron (MLP) structure. This is significant because, during training, MLPs utilize mathematical functions to adjust the weights of connections between neurons, and these functions might be sensitive to the size of input data. When data is not normalized, variables with greater values might dominate the training process, resulting in poor performance or difficulty with convergence. By normalizing the data, we guarantee that each input variable contributes evenly to the training process, hence improving the model's overall performance.

```
> #normalize function
> normalize <- function(x){
+   return ((x - min(x)) / (max(x) - min(x)))
+ }
> uow_load_norm <- as.data.frame(lapply(Data_load[-1,-1], normalize))# normalized data without
> View(uow_load_norm)
> summary(uow_load_norm)
     X0.75         X0.7916666666666663 X0.8333333333333337
 Min.   :0.0000   Min.   :0.0000      Min.   :0.0000
 1st Qu.:0.3689   1st Qu.:0.4061      1st Qu.:0.3679
 Median :0.4928   Median :0.5394      Median :0.5283
 Mean   :0.4721   Mean   :0.5113      Mean   :0.5018
 3rd Qu.:0.5764   3rd Qu.:0.6182      3rd Qu.:0.6258
 Max.   :1.0000   Max.   :1.0000      Max.   :1.0000
> boxplot(uow_load_norm) #plot after normalization
> 
```



Split dataset

```
> #split data
> traindata <- uow_load_norm[1:400,] #data used for training model
> testdata <- uow_load_norm[400:469,] #data used for testing model
> View(traindata)
> names(traindata)
[1] "X0.75"              "X0.7916666666666663" "X0.8333333333333337"
> testdata
        X0.75 X0.7916666666666663 X0.8333333333333337
400 0.4927954           0.5181818           0.4874214
401 0.3861671           0.4545455           0.4402516
402 0.3573487           0.4060606           0.4119497
403 0.3602305           0.3969697           0.4056604
404 0.3429395           0.4151515           0.4213836
405 0.2507205           0.2939394           0.2672956
406 0.4466859           0.4757576           0.4748428
407 0.4063401           0.4666667           0.4465409
408 0.4610951           0.4939394           0.5125786
409 0.3804035           0.4303030           0.4433962
410 0.4466859           0.5060606           0.5251572
411 0.3487032           0.4151515           0.4402516
412 0.3717579           0.4303030           0.3113208
413 0.3804035           0.4393939           0.4371069
414 0.4034582           0.4848485           0.4968553
415 0.3919308           0.4424242           0.4182390
416 0.4553314           0.5090909           0.5062893
417 0.4783862           0.5212121           0.5157233
418 0.5187320           0.5818182           0.5786164
419 0.4236311           0.4787879           0.4245283
420 0.5878963           0.6121212           0.6226415
421 0.6858790           0.7090909           0.6855346
422 0.8126801           0.7606061           0.7421384
423 0.4783862           0.5484848           0.5345912
424 0.5677233           0.5545455           0.5691824
425 0.4149856           0.4666667           0.4937107
426 0.4121037           0.4696970           0.3710692
427 0.5734870           0.5969697           0.5691824
428 0.7089337           0.7212121           0.7389937
429 0.6714697           0.7303030           0.7389937
430 0.6685879           0.7484848           0.7547170
431 0.6195965           0.6878788           0.7389937
```

```
430 0.6685879          0.7484848          0.7547170
431 0.6195965          0.6878788          0.7389937
432 0.6570605          0.7272727          0.6981132
433 0.5302594          0.5818182          0.4937107
434 0.5734870          0.6848485          0.6949686
435 0.5331412          0.6181818          0.6415094
436 0.6080692          0.6636364          0.6761006
437 0.5936599          0.6363636          0.5849057
438 0.5648415          0.6484848          0.6792453
439 0.5216138          0.5818182          0.6194969
440 0.5331412          0.6000000          0.4371069
441 0.6570605          0.6909091          0.7327044
442 0.6772334          0.7696970          0.7987421
443 0.6167147          0.6939394          0.7295597
444 0.6282421          0.6909091          0.7264151
445 0.5446686          0.6484848          0.6603774
446 0.6080692          0.6666667          0.7264151
447 0.5216138          0.5787879          0.4968553
448 0.6167147          0.6757576          0.6981132
449 0.5590778          0.5787879          0.6006289
450 0.6570605          0.7151515          0.7327044
451 0.6195965          0.6575758          0.6572327
452 0.8069164          0.7272727          0.7201258
453 0.5878963          0.6333333          0.6572327
454 0.5244957          0.6030303          0.5125786
455 0.4985591          0.5363636          0.5377358
456 0.2536023          0.2454545          0.2484277
457 0.5389049          0.6030303          0.5880503
458 0.5763689          0.6363636          0.6572327
459 0.5216138          0.5727273          0.6069182
460 0.5417867          0.6121212          0.6383648
461 0.4755043          0.4151515          0.4213836
462 0.4524496          0.5181818          0.5188679
463 0.2968300          0.3363636          0.3270440
464 0.5216138          0.5484848          0.5440252
465 0.5302594          0.5818182          0.5880503
466 0.5533141          0.6030303          0.6320755
467 0.4524496          0.5303030          0.5314465
468 0.4726225          0.4757576          0.4716981
469 0.5331412          0.5969697          0.6037736
>
```

E)

Indices explanation

RMSE

RMSE is an abbreviation for Root Mean Squared Error, which is a popular statistic for determining the accuracy of a prediction model. It computes the square root of the average squared difference between expected and actual values to calculate the difference between predicted and actual values.

The root mean square error (RMSE) is often used in regression analysis to calculate the difference between the anticipated and actual values of the dependent variable. It is chosen over Mean Absolute Error (MAE) because squaring the difference penalizes big mistakes more harshly than tiny errors. A lower RMSE number shows that the model is better fitted to the data, whereas a higher RMSE value indicates that the model has a bigger prediction error. RMSE, on the other hand, should always be viewed in the context of the issue domain and the range of values being forecasted.

MAE

MAE is an abbreviation for Mean Absolute Error, which is a popular statistic for determining the accuracy of a prediction model. It computes the average absolute difference between anticipated and actual values. MAE is frequently used in regression analysis to calculate the difference between the expected and actual values of the dependent variable.

In cases when big mistakes are not necessarily more relevant than tiny errors, it is chosen over Root Mean Squared Error (RMSE).A lower MAE number implies that the model fits the data better, whereas a higher MAE value shows that the model has a bigger prediction error. However, MAE should always be considered in the context of the problem domain and the range of predicted values.

MAPE

MAPE is an abbreviation for Mean Absolute Percentage Error, which is a popular statistic for determining the accuracy of a prediction model. It computes the percentage difference between expected and actual values.

MAPE is a regularly used metric in forecasting and time series analysis to assess the accuracy of predictions made across several time periods. It is an effective indicator for comparing the performance of various forecasting models. A lower MAPE number implies that the model fits the data better, whereas a higher MAPE value shows that the model has a bigger prediction error. MAPE does have certain limitations, such as being sensitive to extreme values or tiny data sets, and it may not be appropriate for instances when the real values are near to zero.

Overall, MAPE should be considered alongside other evaluation metrics and interpreted within the context of the specific problem domain and the range of values being predicted.

symmetric MAPE

The measure Symmetric Mean Absolute Percentage Error (sMAPE) is a variation on the Mean Absolute Percentage Error (MAPE) that tackles some of its shortcomings. sMAPE calculates the percentage difference between expected and real values, but unlike MAPE, it uses the absolute difference between actual and predicted values as the denominator rather than the average of the actual values.

The sMAPE formula is as follows:

sMAPE = (1/n) * (((|A_t| + |F_t|)/2)) * 100%

Where A_t represents the actual value, F_t represents the forecast value, and n is the number of observations.

sMAPE is a number that spans from 0% to 200%, with lower values suggesting higher model accuracy. A number of 0% represents the best possible forecast, while a value of 200% represents the worst possible prediction. sMAPE is widely used in time series forecasting, and it is especially beneficial when the actual and forecasted values have comparable scales. It is particularly important when dealing with intermittent demand since it allows for a more realistic comparison of various forecasting models.

References

Zhao, H., Zhang, Y., & Zhao, Y. (2020). Short-term electricity load forecasting using a hybrid model based on principal component analysis and extreme learning machine. IEEE Access, 8, 41379-41386.

Zhou, J., Xiao, F., & Zhou, Z. (2019). A novel load forecasting method based on Fourier transform and feature selection. Electric Power Systems Research, 167, 140-147.

Elnosh, M. H., Eisa, A. M., & Khalid, A. E. (2018). Short-term load forecasting in smart grids using principal component analysis and artificial neural networks. Journal of Intelligent & Fuzzy Systems, 34(5), 3275-3284.