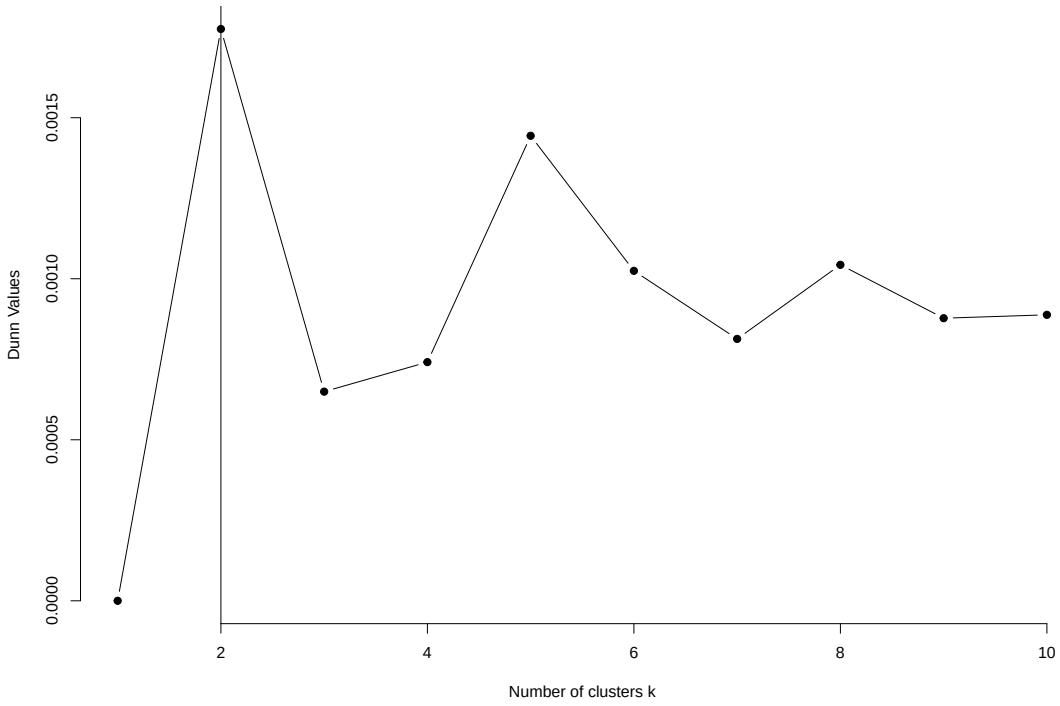
Universidade Estadual de Campinas - Instituto de Computação (IC/Unicamp)
Disciplina Aprendizado de Máquina (MO444)
Professor: Jacques Wainer, PhD. **Aluno: Luiz Alberto Ferreira Gomes**
Atividade 4, 30 de outubro de 2016 **RA:007275**

Questão 1:

Use alguma metrica interna (algum Dunn, Silhouette, Calinski-Harabaz index) - apenas uma -para escolher o k entre 2 e 10.

Solução: A medida interna adotada no script (ver anexo I) foi a Dunn. Ela foi extraída pela função cluster.stats para selecionar o valores de **k**. A comportamento dessa medida para os valores de k (de 2 até 10) é apresentado no gráfico abaixo.

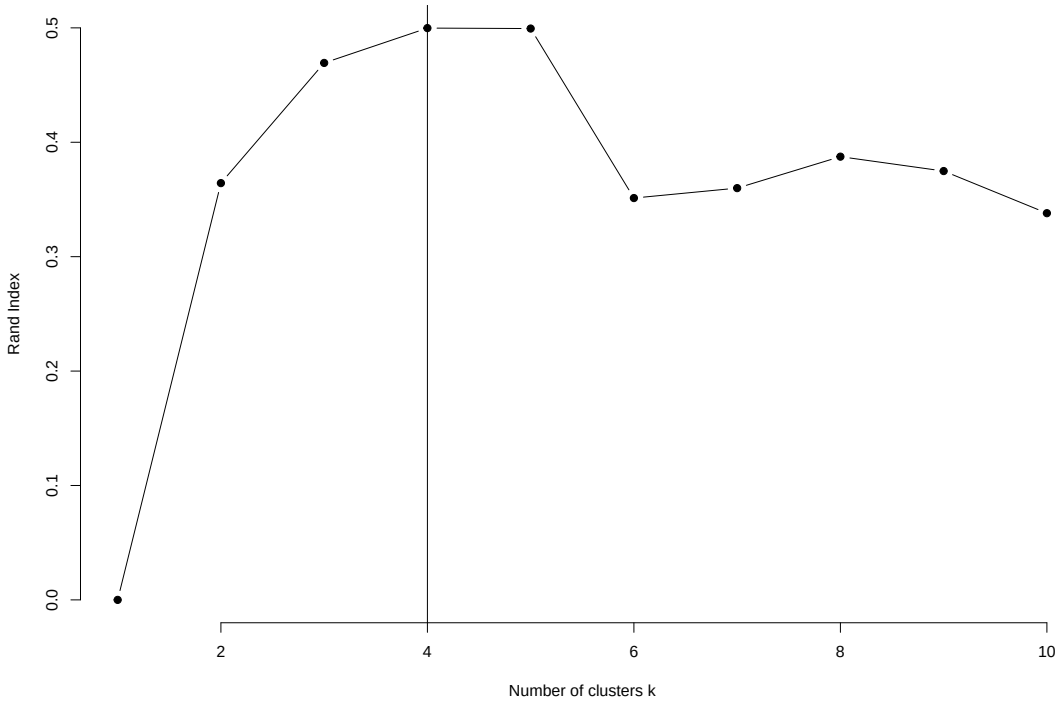


Universidade Estadual de Campinas - Instituto de Computação (IC/Unicamp)
Disciplina Aprendizado de Máquina (MO444)
Professor: Jacques Wainer, PhD. **Aluno: Luiz Alberto Ferreira Gomes**
Atividade 4, 30 de outubro de 2016 **RA:007275**

Questão 2:

Use alguma medida externa (Normalized/adjusted Rand, Mutual information, variation of information) para decidir no k.

Solução: A medida externa adotada no script (ver anexo I) foi a Ajusted Rand. Ela foi extraída pela função cluster.stats. O valor de **k** selecionado através dela foi igual a **4**, conforme o gráfico abaixo.



Anexo I: Script fonte em R

```
# ------------------------------------------------------------------------
# Description:
#   solutions for activity 4 (MO444)
#
# Version: 1.0
#
# Author:
#   Luiz Alberto, gomes.luiz@gmail.com
#
# History:
#   Oct 20th,   2016   started
#
# To do:
#  -
# ------------------------------------------------------------------------

if (!require(caret))
  install.packages(caret)
```

```
19 if (!require(cluster))
20   install.packages(cluster)
21 if (!require(fpc))
22   install.packages(fpc)
23
24 library(fpc)
25 library(caret)
26 library(cluster)
27
28 # cleans up execution environment.
29 rm(list = ls())
30
31 # sets up path to data files.
32 setwd('~/Workspace/doutorado/disciplinas/mo444b/atividades/4')
33
34 # ------------------------------------------------------------------------
35 # common functions
36 # ------------------------------------------------------------------------
37 ReadDataFile <- function(name) {
38   # Reads a data file in csv format.
39   #
40   # Args:
41   #   name: file name to be read.
42   #
43   # Returns:
44   #   the data frame with rows and columns from file.
45   #
46   result <-
47     read.csv(
48       file = paste('./data/', name, sep = ''),
49       header = TRUE,
50       sep = ','
51     )
52   return(result)
53 }
54
55
56 # ------------------------------------------------------------------------
57 # main function
58 # ------------------------------------------------------------------------
59 main <- function() {
60   # reads raw data from files.
61   cluster.data <- ReadDataFile('cluster-data.csv')
62   cluster.data.class <- ReadDataFile('cluster-data-class.csv')
63   cluster.data.scaled <- scale(cluster.data)
64
65   set.seed(123)
66   dd <- dist(cluster.data.scaled, method = "euclidean")
67
68
69   # external cluster validation
70   output <- data.frame(line=character())
71   e.folds <-
72     createFolds(
73       cluster.data.class$x,
74       k = 5,
75       list = TRUE,
76       returnTrain = TRUE
77     )
78   max.rand <- 0
79   for (e in 1:5) {
80     ## internal cluster validation.
81     i.folds <-
82       createFolds(
83         cluster.data.class$x,
84         k = 3,
```

```
85          list = TRUE,
86          returnTrain = TRUE
87        )
88      max.dunn <- 0
89      for (k in 2:10) {
90        dunn <- 0
91        for (i in 1:3) {
92          kmi <- kmeans(cluster.data.scaled[i.folds[[i]], ], k, nstart = 5)
93          kmi.stats <-
94            cluster.stats(dd, kmi$cluster, silhouette = FALSE)
95          dunn <- dunn + kmi.stats$dunn
96
97          output <- rbind(output, data.frame(line=sprintf(' i = %d, k = %d, dunn = %.10f\n', i, k,
                kmi.stats$dunn)))
98        }
99        dunn <- dunn / 3
100       if (dunn > max.dunn) {
101         max.dunn <- kmi.stats$dunn
102         max.ki <- k
103       }
104     }
105
106     output <- rbind(output, data.frame(line=sprintf(' max.ki = %d, max.dunn = %.10f\n', max.ki,
          kmi.stats$dunn)))
107
108     ## external cluster validation
109     classes <- as.numeric(cluster.data.class$x[e.folds[[e]]])
110     kme <-
111       kmeans(cluster.data.scaled[e.folds[[e]],], max.ki, nstart = 5)
112     kme.stats <- kme.stats <- cluster.stats(dd, classes, kme$cluster, silhouette = FALSE)
113
114     output <- rbind(output, data.frame(line=sprintf(' e = %d, max.ki = %d, max.dunn = %.10f\n',
          e, max.ki, kme.stats$corrected.rand)))
115     if (kme.stats$corrected.rand > max.rand) {
116       max.rand <- kme.stats$corrected.rand
117       max.ke <- max.ki
118     }
119   }
120
121   ## write the final k.
122   output <- rbind(output, data.frame(line=sprintf(' max.ke = %d, max.rand = %.10f\n', max.ke,
        max.rand)))
123
124   write.csv(output, file='./data/result.csv', row.names = FALSE, quote = FALSE)
125
126 }
127
128 main()
```