BUAN 6357 Exam 1 Clustering (Johnston) Spring 2023

```
> ###
> #
> # BUAN 6357 2023 Spring (Johnston)
> #
> # Exam 1: section 1 - clustering
> #
> # A run log of this code is provided as a PDF file.
> # You may run this code, explore its actions, and add
> #  comments as you wish.
> #
> # Based on class discussions and homework assignments,
> # You should extend this code as needed in preparation
> #  for answering questions about the process presented
> #  here.
> #
> ###
>
> options(width=70,scipen=10)                    # avoid exponential
notation
> setwd("c:/data/BUAN6357/exams/exam1") # change as needed
>
> byRows <- 1
> byCols <- 2
>
> require(tidyverse)
Loading required package: tidyverse
-- Attaching packages ----------------------------- tidyverse 1.3.2
--
v ggplot2 3.4.0      v purrr   0.3.5
v tibble  3.1.8      v dplyr   1.0.10
v tidyr   1.2.1      v stringr 1.5.0
v readr   2.1.3      v forcats 0.5.2
-- Conflicts -------------------------------- tidyverse_conflicts()
--
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
> require(data.table)
Loading required package: data.table
data.table 1.14.6 using 4 threads (see ?getDTthreads).  Latest news:
r-datatable.com

Attaching package: ‘data.table’
```

```
The following objects are masked from ‘package:dplyr’:

    between, first, last

The following object is masked from ‘package:purrr’:

    transpose

>
> t1          <- fread(file="train1.dat")
> t2          <- fread(file="test1.dat")
> t1Grp       <- t1$grp; t1$grp      <- NULL
> t2Grp       <- t2$grp; t2$grp      <- NULL
> train1      <- t1
> test1       <- t2
> as.data.frame(t1)
      V1  V2  V3  V4
1    4.8 3.0 1.4 0.3
2    4.7 3.2 1.6 0.2
3    5.2 2.7 3.9 1.4
4    5.1 3.8 1.6 0.2
5    4.3 3.0 1.1 0.1
6    6.3 3.3 4.7 1.6
7    5.7 2.8 4.5 1.3
8    6.1 3.0 4.6 1.4
9    6.0 2.2 4.0 1.0
10   6.1 2.8 4.7 1.2
11   6.3 3.4 5.6 2.4
12   6.7 3.3 5.7 2.1
13   7.9 3.8 6.4 2.0
14   4.9 3.1 1.5 0.1
15   5.1 3.8 1.5 0.3
16   5.1 3.3 1.7 0.5
17   5.0 3.5 1.3 0.3
18   6.7 3.0 5.0 1.7
19   5.8 2.7 5.1 1.9
20   7.1 3.0 5.9 2.1
21   6.4 2.9 4.3 1.3
22   5.5 4.2 1.4 0.2
23   5.8 2.6 4.0 1.2
24   5.4 3.9 1.3 0.4
25   4.9 3.1 1.5 0.2
26   5.8 2.7 5.1 1.9
27   7.4 2.8 6.1 1.9
28   5.8 4.0 1.2 0.2
29   6.3 2.5 5.0 1.9
30   6.9 3.1 4.9 1.5
31   6.6 2.9 4.6 1.3
```

32	7.2	3.0	5.8	1.6
33	6.7	2.5	5.8	1.8
34	6.5	2.8	4.6	1.5
35	5.4	3.4	1.5	0.4
36	6.3	2.9	5.6	1.8
37	5.8	2.7	4.1	1.0
38	5.0	3.6	1.4	0.2
39	5.6	2.7	4.2	1.3
40	5.7	2.5	5.0	2.0
41	6.1	2.6	5.6	1.4
42	6.2	2.2	4.5	1.5
43	5.0	3.4	1.5	0.2
44	6.0	3.4	4.5	1.6
45	5.0	2.3	3.3	1.0
46	4.4	3.2	1.3	0.2
47	6.2	3.4	5.4	2.3
48	4.9	3.6	1.4	0.1
49	5.5	2.3	4.0	1.3
50	5.7	2.8	4.1	1.3
51	5.4	3.9	1.7	0.4
52	5.9	3.0	5.1	1.8
53	5.4	3.7	1.5	0.2
54	5.7	2.9	4.2	1.3
55	6.1	2.8	4.0	1.3
56	7.2	3.2	6.0	1.8
57	7.7	2.6	6.9	2.3
58	6.7	3.0	5.2	2.3
59	6.5	3.0	5.8	2.2
60	6.7	3.1	4.7	1.5
61	5.5	2.5	4.0	1.3
62	7.0	3.2	4.7	1.4
63	6.1	2.9	4.7	1.4
64	6.7	3.3	5.7	2.5
65	6.3	3.3	6.0	2.5
66	4.5	2.3	1.3	0.3
67	5.2	3.4	1.4	0.2
68	6.3	2.5	4.9	1.5
69	5.8	2.8	5.1	2.4
70	4.6	3.6	1.0	0.2
71	4.6	3.2	1.4	0.2
72	6.4	2.8	5.6	2.1
73	5.8	2.7	3.9	1.2
74	6.0	3.0	4.8	1.8
75	7.7	2.8	6.7	2.0
76	4.7	3.2	1.3	0.2
77	6.8	3.2	5.9	2.3
78	5.1	3.8	1.9	0.4
79	4.6	3.1	1.5	0.2
80	6.5	3.0	5.2	2.0

81	6.4	3.1	5.5	1.8
82	6.8	3.0	5.5	2.1
83	5.6	3.0	4.1	1.3
84	5.5	2.6	4.4	1.2
85	4.9	3.0	1.4	0.2
86	5.5	3.5	1.3	0.2
87	5.0	3.3	1.4	0.2
88	5.7	4.4	1.5	0.4
89	5.0	2.0	3.5	1.0
90	4.4	2.9	1.4	0.2
91	5.1	3.4	1.5	0.2
92	4.8	3.0	1.4	0.1
93	6.5	3.2	5.1	2.0
94	4.4	3.0	1.3	0.2
95	6.7	3.1	4.4	1.4
96	4.9	2.5	4.5	1.7
97	6.4	3.2	4.5	1.5
98	6.9	3.2	5.7	2.3
99	5.2	3.5	1.5	0.2
100	5.7	3.0	4.2	1.2
101	6.9	3.1	5.1	2.3
102	5.7	3.8	1.7	0.3
103	6.3	2.7	4.9	1.8
104	6.0	2.7	5.1	1.6
105	5.7	2.6	3.5	1.0
106	5.2	4.1	1.5	0.1
107	5.4	3.4	1.7	0.2
108	7.7	3.8	6.7	2.2
109	4.8	3.4	1.6	0.2
110	6.7	3.1	5.6	2.4
111	5.9	3.0	4.2	1.5
112	5.5	2.4	3.7	1.0
113	5.0	3.5	1.6	0.6
114	5.1	3.5	1.4	0.2
115	5.6	2.5	3.9	1.1
116	5.3	3.7	1.5	0.2
117	4.9	2.4	3.3	1.0
118	7.6	3.0	6.6	2.1
119	6.2	2.9	4.3	1.3
120	7.7	3.0	6.1	2.3
121	5.0	3.2	1.2	0.2
122	6.9	3.1	5.4	2.1
123	7.2	3.6	6.1	2.5
124	6.0	2.2	5.0	1.5
125	5.9	3.2	4.8	1.8
126	6.0	2.9	4.5	1.5
127	5.6	2.8	4.9	2.0
128	5.6	2.9	3.6	1.3
129	5.0	3.4	1.6	0.4

```
130 5.1 3.7 1.5 0.4
131 5.1 3.5 1.4 0.3
132 6.8 2.8 4.8 1.4
133 6.3 2.8 5.1 1.5
134 6.3 2.3 4.4 1.3
135 6.6 3.0 4.4 1.4
> as.data.frame(t2)
    V1  V2  V3  V4
1  6.4 3.2 5.3 2.3
2  5.0 3.0 1.6 0.2
3  6.4 2.7 5.3 1.9
4  7.3 2.9 6.3 1.8
5  4.8 3.1 1.6 0.2
6  6.1 3.0 4.9 1.8
7  4.6 3.4 1.4 0.3
8  5.6 3.0 4.5 1.5
9  6.5 3.0 5.5 1.8
10 5.4 3.0 4.5 1.5
11 5.1 2.5 3.0 1.1
12 6.4 2.8 5.6 2.2
13 5.5 2.4 3.8 1.1
14 4.8 3.4 1.9 0.2
15 6.2 2.8 4.8 1.8
>
> maxGrp     <- 10
> starts     <- 10
> seed       <- 838216542
> set.seed(seed)
>
> myKmeans  <- function(seed,df,k,ns) {
+              set.seed(seed)
+              return(kmeans(df,k,ns)$tot.withinss)
+              }
>
> (dt       <- data.table(idx=1:maxGrp, k=1:maxGrp)  )
    idx  k
 1:   1  1
 2:   2  2
 3:   3  3
 4:   4  4
 5:   5  5
 6:   6  6
 7:   7  7
 8:   8  8
 9:   9  9
10:  10 10
> (kmss     <- dt[, .(wgss=myKmeans(seed,train1,k,starts)), by=.
(idx)])
```

```
    idx      wgss
 1:   1 622.68519
 2:   2 138.90833
 3:   3  70.77430
 4:   4  51.53153
 5:   5  44.28929
 6:   6  37.98418
 7:   7  30.36137
 8:   8  28.78611
 9:   9  25.69147
10:  10  24.17375
>
> dif1      <- function(df) {
+              n  <- length(df)
+              t1 <- df[1:(n-1)]-df[2:n]
+              t2 <- t1/max(t1)
+              return(list(d1=t1,d1scaled=t2))
+              }
>
> plot(1:maxGrp,kmss$wgss)
>
> (tkm      <- dif1(kmss$wgss)  )
$d1
[1] 483.776852  68.134032  19.242776   7.242233   6.305115   7.622811
[7]   1.575254   3.094645   1.517715

$d1scaled
[1] 1.000000000 0.140837726 0.039776140 0.014970192 0.013033107
[6] 0.015756874 0.003256159 0.006396843 0.003137221

>
> plot(1:(maxGrp-1),tkm$d1)
>
> plot(1:(maxGrp-1),tkm$d1scaled)
>
> set.seed(seed)
>
> k        <- 3
> km3      <- kmeans(train1,k,nstart=10)
> km3clust <- km3$cluster
>
> prepMHD <- function(df) {
+      df$cluster <- NULL
+      df$grps    <- NULL
+      n          <- nrow(df)
+      df2        <- scale(df, center=T, scale=T)
+      vcvinv     <- solve(cov(df2))
+      return( list(n      = n,
+                   avg    = attr(df2, "scaled:center"),
```

```
+                    sdev   = attr(df2, "scaled:scale"),
+                    vcvinv = vcvinv )
+              )
+       }
>
> t           <- train1
> t$cluster   <- km3clust
> kmMHwk      <- t                       %>%
+                group_by(cluster)   %>%
+                do(desc=prepMHD(select(.,V1,V2,V3,V4)))
>
> kmDesc      <- kmMHwk$desc
> kmDF        <- 4
> nCl         <- 3
> kmTr        <- matrix(NA,nrow=nrow(train1),ncol=nrow(kmMHwk))
> for ( i in 1:nrow(kmMHwk) ) {
+     tD       <- kmDesc[[i]]
+     tdf      <- scale(select(t,V1,V2,V3,V4), center=tD$avg,
scale=tD$sdev)
+     kmTr[,i] <- mahalanobis(tdf, center=F, cov=tD$vcvinv,
inverted=T)
+     }
>
> kmTr
               [,1]        [,2]        [,3]
  [1,]    2.0952026 103.2189135  51.0569562
  [2,]    2.9831598 102.8632274  46.8127239
  [3,]  286.6562942  17.8624011   3.3867753
  [4,]    3.0810089 123.7345614  65.9048248
  [5,]    6.7170268 118.0855921  58.8363008
  [6,]  444.0603243   8.1819321   3.4197900
  [7,]  389.5768324  12.6191440   3.2514947
  [8,]  412.7762753   9.5218910   1.5765324
  [9,]  306.1118114  25.3582954   5.7452684
 [10,]  434.0091884  11.8859428   5.9422277
 [11,]  801.0453714   3.7443008  12.5292337
 [12,]  770.1506517   0.8642712   8.1506788
 [13,]  973.3262501  10.4406145  28.1091928
 [14,]    3.6831123 109.9734760  46.5811340
 [15,]    1.6024924 122.6828048  70.4369480
 [16,]    6.8324526  93.5006060  50.9459646
 [17,]    1.7348707 119.8773724  68.6196160
 [18,]  543.6772442   4.1057919   3.1020248
 [19,]  603.5267817   5.8351671   3.6237534
 [20,]  850.7078178   0.4000442  10.8746964
 [21,]  350.8897877  14.6624012   1.8751172
 [22,]    4.8320028 153.4123500  91.5728639
 [23,]  289.2673771  18.0859841   0.8525261
 [24,]    6.0654114 134.7355146  88.5867413
```

```
[25,]    1.5412425 105.8441476  47.4560876
[26,]  603.5267817   5.8351671   3.6237534
[27,]  912.9199598   2.4686421  18.4786607
[28,]   10.4767404 160.4791015  96.3500896
[29,]  591.4897456   6.2761095   6.3050402
[30,]  500.4778003   8.5399432   4.4665216
[31,]  420.0606776  11.9938757   2.7436332
[32,]  772.9959914   3.4382303  16.3784634
[33,]  806.4121010   4.2763070  14.2427989
[34,]  437.2653480   9.0424842   2.1074535
[35,]    4.6445960 111.6283198  62.7775420
[36,]  717.1481957   3.1854520   9.1217501
[37,]  295.4227554  20.5946544   3.5282641
[38,]    0.6917353 123.1031686  65.4623773
[39,]  329.4388902  14.6034782   0.8229716
[40,]  605.5605381   8.7526665   5.7813281
[41,]  703.3658598  10.8795051  26.1592950
[42,]  444.0643142  15.4188299   7.2081244
[43,]    0.5020362 113.5570597  55.2456840
[44,]  398.6824472  11.3074313   5.8078998
[45,]  170.4535452  33.4501614   6.4049850
[46,]    4.2331278 111.4164046  57.6603377
[47,]  727.2866970   3.6636490  11.0852013
[48,]    2.9894127 126.8209261  64.7995228
[49,]  307.3708306  19.4346590   2.5746654
[50,]  306.6453684  15.1683163   0.7121127
[51,]    3.5745090 118.0598605  68.5675209
[52,]  576.6594652   4.9034252   3.9467821
[53,]    1.7885922 127.7818994  66.2131952
[54,]  322.6166122  14.5626431   1.3782079
[55,]  292.8917347  17.4008588   2.4686108
[56,]  837.6590756   2.1874133  16.3064001
[57,] 1271.3741414  10.0532656  37.1555158
[58,]  694.0848599   3.7989374  16.2577578
[59,]  830.0647117   1.5387670   8.4248064
[60,]  448.4374713   9.3506238   3.1172526
[61,]  298.7934630  17.4863470   1.2445321
[62,]  444.7235946  14.1126910   6.1395404
[63,]  439.8862718   8.5479980   1.8726535
[64,]  852.0460961   2.3861585  15.2804486
[65,]  942.0341613   6.3109372  14.6682812
[66,]   11.9396537 100.7511248  47.9435592
[67,]    1.2667412 120.0814418  60.0842270
[68,]  519.0374083   7.4980164   3.4554944
[69,]  705.6700419   8.0294123  15.0513200
[70,]   11.7303873 134.7192557  82.7392030
[71,]    1.5910938 108.9791870  53.4301860
[72,]  764.4905716   2.0783722   6.6785100
[73,]  267.1922460  19.1753496   1.2827364
```

```
 [74,]  503.5358733   4.4734040   1.9821931
 [75,] 1145.3256335   5.1609783  33.9406508
 [76,]    1.2055337 113.3303946  57.0804030
 [77,]  869.4256979   0.7865262  10.4914644
 [78,]    9.7120481 104.5082221  57.5022098
 [79,]    2.1196959 103.5129197  47.5707817
 [80,]  631.9766039   1.3379654   4.6205695
 [81,]  679.1376160   2.5050513   7.2501517
 [82,]  730.4646699   0.6426731   7.4184279
 [83,]  300.6852812  16.3508197   2.7155051
 [84,]  368.8346984  16.1285493   5.0388275
 [85,]    2.1270409 107.9986590  49.3910827
 [86,]    5.7782586 132.5381036  70.6263294
 [87,]    0.4693593 114.9078496  56.3551958
 [88,]    7.3002649 152.5268777 102.0432029
 [89,]  210.9012090  34.5265179   7.6076914
 [90,]    3.1288281 102.5027086  46.9693870
 [91,]    0.7096160 114.7066381  55.4693650
 [92,]    3.0953290 110.8449326  47.6485721
 [93,]  598.4140113   2.0096599   5.5095711
 [94,]    3.2088708 107.4543477  52.5368825
 [95,]  376.3805494  14.9905484   4.6259098
 [96,]  455.2500459  18.3079134   7.5002926
 [97,]  394.8251611  10.6584144   2.8101977
 [98,]  812.9478602   0.7516609  11.0794474
 [99,]    0.8793015 118.7025760  58.7651905
[100,]  312.7066739  17.1730906   3.6083587
[101,]  669.4562173   5.8860709  21.7591024
[102,]    5.1742755 123.6747245  65.0420168
[103,]  541.5566578   4.4920401   2.9143792
[104,]  564.6788850   5.8616826   4.4760875
[105,]  193.5065283  29.0624959   4.0401013
[106,]    7.4817462 146.0868392  82.3866367
[107,]    6.1345428 111.1157444  49.4411760
[108,] 1097.6717651   9.5133495  31.0107863
[109,]    2.7123426 108.4189149  52.3192418
[110,]  811.0253848   1.8423949  13.0498719
[111,]  339.2307008  11.8002415   2.1634414
[112,]  230.4111332  26.1877520   2.8218366
[113,]   11.6305038  97.2356862  63.4760150
[114,]    0.3638602 121.3126943  62.4057385
[115,]  266.9420793  21.2609304   1.5734326
[116,]    1.1806149 126.2598187  65.7276018
[117,]  167.6774229  33.1244287   6.5731505
[118,] 1098.2011447   3.1538493  26.2038206
[119,]  346.6372960  13.5707680   0.8764014
[120,]  964.0847051   5.0565447  22.0409815
[121,]    3.3512353 121.0372418  63.0549902
[122,]  700.3292606   1.2929355   8.5815997
```

```
[123,]  953.5844328   3.6997375  18.1993670
[124,]  556.2990184  11.6274326   8.1676540
[125,]  498.0127190   5.9420972   3.9404677
[126,]  403.1553157   8.6273799   0.2874975
[127,]  571.8613464   6.9951160   4.9567429
[128,]  220.2688294  22.1319060   6.3413561
[129,]    2.4634542 101.8052733  54.6766509
[130,]    2.6131299 115.0505233  68.7818568
[131,]    0.5635393 117.1501908  64.4555643
[132,]  484.3361597   9.8209340   4.2993651
[133,]  553.8897794   5.2160832   4.7427270
[134,]  403.1579282  16.5567077   5.2846852
[135,]  378.6536976  13.6339073   3.4008513
>
> kmNew          <- apply(kmTr, byRows, which.min)
> train1$mhCl    <- kmNew
> train1$grp     <- t1Grp
> train1$clust   <- km3clust
> (tbl446        <-table(train1$grp,   train1$clust,
dnn=c("grp"  ,"clust")) )
   clust
grp  1  2  3
  1 46  0  0
  2  0  2 44
  3  0 31 12
> (tbl452        <-table(train1$clust, train1$mhCl,
dnn=c("clust","mhCl" )) )
     mhCl
clust  1  2  3
    1 46  0  0
    2  0 31  2
    3  0  2 54
>
> kmStat         <- apply(kmTr, byRows, min)
> kmP            <- pchisq(kmStat, df=kmDF, lower.tail=F)
>
> kmTst          <- matrix(NA,nrow=nrow(test1),ncol=nrow(kmMHwk))
> for ( i in 1:nrow(kmMHwk) ) {
+     tD        <- kmDesc[[i]]
+     tdf       <- scale(test1, center=tD$avg, scale=tD$sdev)
+     kmTst[,i] <- mahalanobis(tdf, center=F, cov=tD$vcvinv,
inverted=T)
+     }
>
> tstNew        <- apply(kmTst, byRows, which.min)
> test1$mhCl    <- tstNew
> test1$grp     <- t2Grp
> (tbl472       <- table(test1$grp, test1$mhCl,
dnn=c("grp","mhCl")) )
```

```
   mhCl
grp 1 2 3
  1 4 0 0
  2 0 0 4
  3 0 5 2
>
> hcDat          <- train1
> hcDat$mhCl     <- NULL
> hcDat$grp      <- NULL
> hcDat$clust    <- NULL
>
> hc             <- hclust(dist(hcDat)^2,method="complete")
>
> hcwgss         <- function(train,hc,i) {
+       t1 <- cutree(hc,i)
+       t2 <- data.table(idx=t1,j=1:nrow(train))
+       t3 <- t2[, .(ss=sum(scale(train[j,], center=T, scale=F)^2)),
by=.(idx)]
+       return(sum(t3))
+       }
>
> (hcss       <- dt[,.(wgss=hcwgss(hcDat,hc,k)),by=.(idx)] )
    idx      wgss
 1:   1 623.68519
 2:   2 226.98258
 3:   3  87.97811
 4:   4  65.89673
 5:   5  64.48006
 6:   6  57.89971
 7:   7  61.85718
 8:   8  64.84383
 9:   9  72.18458
10:  10  80.28636
>
> plot(1:maxGrp,hcss$wgss)
>
> (thc       <- dif1(hcss$wgss) )
$d1
[1] 396.702606 139.004468  22.081385   1.416670   6.580346  -3.957465
[7]  -2.986657  -7.340747  -8.101779

$d1scaled
[1]  1.000000000  0.350399684  0.055662313  0.003571114  0.016587605
[6] -0.009975899 -0.007528705 -0.018504408 -0.020422802

>
> plot(1:(maxGrp-1),thc$d1)
>
> plot(1:(maxGrp-1),thc$d1scaled)
```