Exam 1 – Fall 2019 (Johnston)

Section 1: Clustering

```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

Microsoft R Open 3.5.1
The enhanced R distribution from Microsoft
Microsoft packages Copyright (C) 2018 Microsoft Corporation

Using the Intel MKL for parallel mathematical computing (using 4 cores).

Default CRAN mirror snapshot taken on 2018-08-01.
See: https://mran.microsoft.com/.

> ###
> #
> # BUAN 6357 2019 Fall (Johnston)
> #
> # Exam 1: section 1 - clustering
> #
> #
> options(scipen=10)           # avoid exponential notation
> setwd("c:/data/exams/exam1") # change as needed
>
> byRows <- 1
> byCols <- 2
>
> require(tidyverse)
Loading required package: tidyverse
-- Attaching packages ---------------------------------------- tidyverse 1.2.1 --
v ggplot2 3.0.0     v purrr   0.2.5
v tibble  1.4.2     v dplyr   0.7.6
v tidyr   0.8.1     v stringr 1.3.1
v readr   1.1.1     v forcats 0.3.0
-- 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 1.11.4  Latest news: http://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(train1)
     V1  V2  V3  V4
1   4.8 3.0 1.4 0.3
2   4.7 3.2 1.6 0.2
3   5.4 3.0 4.5 1.5
4   5.2 2.7 3.9 1.4
5   5.1 3.8 1.6 0.2
6   4.3 3.0 1.1 0.1
7   6.3 3.3 4.7 1.6
8   5.6 3.0 4.5 1.5
9   5.7 2.8 4.5 1.3
10  7.3 2.9 6.3 1.8
11  6.1 3.0 4.6 1.4
12  6.0 2.2 4.0 1.0
13  6.1 2.8 4.7 1.2
14  6.3 3.4 5.6 2.4
15  6.7 3.3 5.7 2.1
16  7.9 3.8 6.4 2.0
17  4.9 3.1 1.5 0.1
18  5.1 3.8 1.5 0.3
19  5.1 3.3 1.7 0.5
20  5.0 3.5 1.3 0.3
21  6.7 3.0 5.0 1.7
22  6.2 2.8 4.8 1.8
23  5.8 2.7 5.1 1.9
24  7.1 3.0 5.9 2.1
25  6.1 3.0 4.9 1.8
26  6.4 2.9 4.3 1.3
27  5.5 4.2 1.4 0.2
28  5.8 2.6 4.0 1.2
29  5.4 3.9 1.3 0.4
30  4.8 3.4 1.9 0.2
31  5.0 3.0 1.6 0.2
32  4.9 3.1 1.5 0.2
33  5.8 2.7 5.1 1.9
34  7.4 2.8 6.1 1.9
35  5.8 4.0 1.2 0.2
```

115	36	6.3	2.5	5.0	1.9
116	37	6.9	3.1	4.9	1.5
117	38	6.6	2.9	4.6	1.3
118	39	7.2	3.0	5.8	1.6
119	40	6.7	2.5	5.8	1.8
120	41	6.5	2.8	4.6	1.5
121	42	5.4	3.4	1.5	0.4
122	43	6.3	2.9	5.6	1.8
123	44	5.8	2.7	4.1	1.0
124	45	5.0	3.6	1.4	0.2
125	46	5.6	2.7	4.2	1.3
126	47	5.7	2.5	5.0	2.0
127	48	6.1	2.6	5.6	1.4
128	49	6.2	2.2	4.5	1.5
129	50	5.0	3.4	1.5	0.2
130	51	6.0	3.4	4.5	1.6
131	52	5.0	2.3	3.3	1.0
132	53	4.4	3.2	1.3	0.2
133	54	5.1	2.5	3.0	1.1
134	55	6.2	3.4	5.4	2.3
135	56	4.9	3.6	1.4	0.1
136	57	5.5	2.3	4.0	1.3
137	58	5.7	2.8	4.1	1.3
138	59	5.4	3.9	1.7	0.4
139	60	5.9	3.0	5.1	1.8
140	61	5.4	3.7	1.5	0.2
141	62	5.7	2.9	4.2	1.3
142	63	6.1	2.8	4.0	1.3
143	64	7.2	3.2	6.0	1.8
144	65	7.7	2.6	6.9	2.3
145	66	6.7	3.0	5.2	2.3
146	67	6.5	3.0	5.8	2.2
147	68	6.7	3.1	4.7	1.5
148	69	5.5	2.5	4.0	1.3
149	70	7.0	3.2	4.7	1.4
150	71	6.1	2.9	4.7	1.4
151	72	6.7	3.3	5.7	2.5
152	73	6.3	3.3	6.0	2.5
153	74	4.5	2.3	1.3	0.3
154	75	5.2	3.4	1.4	0.2
155	76	6.3	2.5	4.9	1.5
156	77	5.8	2.8	5.1	2.4
157	78	4.6	3.6	1.0	0.2
158	79	4.6	3.2	1.4	0.2
159	80	6.4	2.8	5.6	2.1
160	81	5.8	2.7	3.9	1.2
161	82	6.0	3.0	4.8	1.8
162	83	7.7	2.8	6.7	2.0
163	84	4.7	3.2	1.3	0.2
164	85	6.8	3.2	5.9	2.3
165	86	5.1	3.8	1.9	0.4
166	87	4.6	3.1	1.5	0.2
167	88	6.5	3.0	5.2	2.0
168	89	6.4	3.1	5.5	1.8
169	90	6.8	3.0	5.5	2.1
170	91	5.6	3.0	4.1	1.3
171	92	5.5	2.6	4.4	1.2
172	93	4.9	3.0	1.4	0.2
173	94	6.4	3.2	5.3	2.3

```
95  5.5 3.5 1.3 0.2
96  5.0 3.3 1.4 0.2
97  5.7 4.4 1.5 0.4
98  5.0 2.0 3.5 1.0
99  4.4 2.9 1.4 0.2
100 5.1 3.4 1.5 0.2
101 4.8 3.0 1.4 0.1
102 6.5 3.2 5.1 2.0
103 4.4 3.0 1.3 0.2
104 6.7 3.1 4.4 1.4
105 4.9 2.5 4.5 1.7
106 6.4 3.2 4.5 1.5
107 6.9 3.2 5.7 2.3
108 5.2 3.5 1.5 0.2
109 5.7 3.0 4.2 1.2
110 6.9 3.1 5.1 2.3
111 4.6 3.4 1.4 0.3
112 5.7 3.8 1.7 0.3
113 6.3 2.7 4.9 1.8
114 6.0 2.7 5.1 1.6
115 5.7 2.6 3.5 1.0
116 5.2 4.1 1.5 0.1
117 5.4 3.4 1.7 0.2
118 7.7 3.8 6.7 2.2
119 4.8 3.4 1.6 0.2
120 6.7 3.1 5.6 2.4
121 5.9 3.0 4.2 1.5
122 5.5 2.4 3.7 1.0
123 4.8 3.1 1.6 0.2
124 5.0 3.5 1.6 0.6
125 5.1 3.5 1.4 0.2
126 5.6 2.5 3.9 1.1
127 5.3 3.7 1.5 0.2
128 4.9 2.4 3.3 1.0
129 7.6 3.0 6.6 2.1
130 6.2 2.9 4.3 1.3
131 7.7 3.0 6.1 2.3
132 5.0 3.2 1.2 0.2
133 6.9 3.1 5.4 2.1
134 7.2 3.6 6.1 2.5
135 6.0 2.2 5.0 1.5
> as.data.frame(test1)
    V1  V2  V3  V4
1  5.9 3.2 4.8 1.8
2  6.5 3.0 5.5 1.8
3  6.0 2.9 4.5 1.5
4  6.4 2.7 5.3 1.9
5  5.6 2.8 4.9 2.0
6  6.4 2.8 5.6 2.2
7  5.6 2.9 3.6 1.3
8  5.0 3.4 1.6 0.4
9  5.1 3.7 1.5 0.4
10 5.1 3.5 1.4 0.3
11 6.8 2.8 4.8 1.4
12 6.3 2.8 5.1 1.5
13 6.3 2.3 4.4 1.3
14 5.5 2.4 3.8 1.1
15 6.6 3.0 4.4 1.4
>
```

```
> maxGrp     <- 10
> starts     <- 10
> seed       <- 359703212
> set.seed(seed)
>
> myKmeans  <- function(seed,df,k,ns) {
+              set.seed(seed)
+              return(kmeans(df,k,ns)$tot.withinss)
+              }
>
> (dt       <- data.table(idx=1:maxGrp, k=1:maxGrp)  )
     idx  k
 1:    1  1
 2:    2  2
 3:    3  3
 4:    4  4
 5:    5  5
 6:    6  6
 7:    7  7
 8:    8  8
 9:    9  9
10:   10 10
> (kmss     <- dt[,
+                 .(wgss=myKmeans(seed,train1,k,starts)),
+                 by=.(idx)])
     idx      wgss
 1:    1 637.85274
 2:    2 142.69367
 3:    3  72.05550
 4:    4  64.75030
 5:    5  45.31727
 6:    6  43.07146
 7:    7  33.53043
 8:    8  32.79122
 9:    9  28.23622
10:   10  24.00620
>
> dif1      <- function(df) {
+              n  <- length(df)
+              t1 <- df[1:(n-1)]-df[2:n]
+              t2 <- t1/max(t1)
+              return(list(d1=t1,d1scaled=t2))
+              }
>
> plot(1:maxGrp,kmss$wgss)
>
> (tkm      <- dif1(kmss$wgss)  )
$`d1`
[1] 495.1590702  70.6381696   7.3052055  19.4330249   2.2458061   9.5410320
[7]   0.7392165   4.5550000   4.2300115

$d1scaled
[1] 1.000000000 0.142657529 0.014753250 0.039246024 0.004535525 0.019268620
[7] 0.001492887 0.009199064 0.008542733

>
> 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$grps    <- NULL
+      n          <- nrow(df)
+      df2        <- scale(df, center=T, scale=T)
+      sscp       <- t(df2) %*% df2
+      vcvinv     <- solve((1/(n-1)) * sscp)
+      return( list(n      = n,
+                   avg    = attr(df2, "scaled:center"),
+                   sdev   = attr(df2, "scaled:scale"),
+                   vcvinv = vcvinv )
+             )
+      }
>
> t          <- train1
> t$cluster  <- km3clust
> kmMHwk     <- t                    %>%
+               group_by(cluster)    %>%
+               do(desc=prepMHD(select(.,V1,V2,V3,V4)))
> kmMHwk     <- as.data.frame(kmMHwk)
>
> 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,]  47.2634655    2.1719877  84.0092113
  [2,]  43.7013323    2.0124306  84.8708025
  [3,]   4.8011632  376.8928751  13.4096495
  [4,]   3.1530915  274.6794835  17.4423733
  [5,]  64.4520437    2.5509449 100.0040591
  [6,]  54.2861145    6.5841119  98.7801471
  [7,]   3.6990388  418.2421800   6.6602830
  [8,]   2.6440837  374.4670342  10.8132427
  [9,]   2.7091283  350.6261580  12.1619321
 [10,]  25.3578589  862.7131477   2.6628397
 [11,]   1.5205152  377.3731003   8.1539782
 [12,]   6.4996840  270.4788699  23.2580652
 [13,]   5.7162126  381.1309577  10.6867257
 [14,]  12.4364028  793.0525160   3.7526263
```

```
      [15,]   7.7770212  733.5711021   0.9428458
      [16,]  28.6185400  893.2432196  10.6941051
      [17,]  43.2978488    3.0305853  89.0130100
      [18,]  69.3377482    1.6812409  98.5655012
      [19,]  48.7664014    7.7058402  74.7888433
      [20,]  66.2912286    1.7490667  95.6435370
      [21,]   3.8304543  510.9533332   3.0043024
      [22,]   2.3212518  493.4621238   4.1990802
      [23,]   3.5185404  576.2039485   6.2879356
      [24,]  11.0805105  802.9989808   0.4493040
      [25,]   1.7964123  503.3328454   3.4869566
      [26,]   2.6887754  322.6298086  11.3454862
      [27,]  92.3053123    4.9436509 122.0764657
      [28,]   0.7948920  265.0138089  16.0050515
      [29,]  88.2759229    5.9178114 106.0000635
      [30,]  42.6581386    9.1764105  82.8006794
      [31,]  39.1300935    3.5211150  81.9504140
      [32,]  44.2291697    1.1499728  85.6036064
      [33,]   3.5185404  576.2039485   6.2879356
      [34,]  19.7709342  835.4828973   2.4644094
      [35,]  97.1021235    9.6845169 124.4262045
      [36,]   6.0491568  569.6857458   6.6847804
      [37,]   5.9668296  459.9936632   6.1220340
      [38,]   3.9513894  379.5019383   9.1365009
      [39,]  17.3056678  690.6068211   2.7796060
      [40,]  14.7716893  735.4976886   4.5200764
      [41,]   2.8168591  408.0219095   7.2507822
      [42,]  60.7567883    4.9122804  87.3278204
      [43,]   8.2702602  658.5836092   3.4347230
      [44,]   3.3417216  257.6800192  18.0961601
      [45,]  63.4156911    0.6899885  98.7943986
      [46,]   0.5801140  302.5901148  13.7499347
      [47,]   5.7285732  589.2776700   9.3348228
      [48,]  24.5852747  611.5622609  11.5099545
      [49,]   7.3164933  415.7160718  15.5405888
      [50,]  52.7962495    0.2756412  91.1390279
      [51,]   5.9513920  380.2664384   9.9144338
      [52,]   5.4334994  157.4551131  31.9828585
      [53,]  54.0674632    3.9557662  93.0258342
      [54,]  11.2131265  135.5458986  32.4937167
      [55,]  11.1820906  720.1165491   3.7660490
      [56,]  62.3146681    2.8707562 102.7030496
      [57,]   2.3001383  286.6041038  19.2299413
      [58,]   0.5293536  284.2342397  13.5276938
      [59,]  68.5337949    3.9105618  93.8527189
      [60,]   3.4848675  543.3537830   5.2219692
      [61,]  65.2379982    1.7310790 100.5023257
      [62,]   1.0967723  296.6490641  13.0833082
      [63,]   2.6200006  274.8528682  14.1553701
      [64,]  16.2508081  760.0735116   1.9840450
      [65,]  37.6542281 1176.1913053   9.7025366
      [66,]  15.5389170  696.3337285   3.8491020
      [67,]   7.8460927  794.3653447   1.4868169
      [68,]   4.2505691  416.6293764   6.8068306
      [69,]   0.9708422  278.6540785  16.7667414
      [70,]   8.2363605  407.7241130  10.2304183
      [71,]   1.7853598  399.5602827   7.5703912
      [72,]  14.9558947  849.2777297   2.3491845
      [73,]  13.8800763  921.6567981   5.7746744
```

```
   [74,]  41.7906550   11.7077651  86.7603501
   [75,]  57.7901672    1.2078580  94.8172332
   [76,]   4.0750632  474.3693121   7.2961915
   [77,]  14.7614730  716.6220536   7.9532491
   [78,]  79.9471716   10.5231339 109.6516912
   [79,]  50.0429106    1.3736670  89.8582308
   [80,]   6.3607965  730.1187669   2.1739476
   [81,]   1.1524991  246.8409176  16.4827702
   [82,]   1.9790512  484.5147509   4.2519544
   [83,]  34.9269353 1036.6913700   5.0584421
   [84,]  53.5927925    1.1844302  92.3256813
   [85,]  10.0210498  838.9110416   0.7869444
   [86,]  56.5261746    8.6270624  85.3564567
   [87,]  44.1198660    1.5721688  85.7966944
   [88,]   4.6523844  610.9136857   1.1923423
   [89,]   6.4782371  627.5375941   2.6188450
   [90,]   7.4729977  703.4475457   0.6300740
   [91,]   2.3573547  278.4470680  14.7151219
   [92,]   4.4948562  327.2697153  16.1252964
   [93,]  45.7003868    1.9226384  87.2471859
   [94,]  10.6901695  704.6780165   1.9098200
   [95,]  68.9201902    5.3840367 102.9704337
   [96,]  53.5354982    0.4253948  91.8699843
   [97,] 104.5095656    7.7038476 121.5316604
   [98,]   7.0927740  191.1050485  34.7478798
   [99,]  42.9397387    2.7925856  86.3637279
  [100,]  53.1671701    0.5055118  91.4047671
  [101,]  43.9085081    2.7873479  90.2086552
  [102,]   5.7331099  582.6258821   1.6156298
  [103,]  48.5114834    3.0590713  89.9165418
  [104,]   5.9976417  350.9830478  11.0496559
  [105,]   8.1335834  436.3559170  19.7375179
  [106,]   3.4634961  371.1680217   8.1055829
  [107,]  10.9463278  793.4051308   0.8064923
  [108,]  56.8698299    0.7305255  94.0831559
  [109,]   3.0199091  281.6784503  15.3711869
  [110,]  21.0457388  676.4196087   5.7470530
  [111,]  57.9234462    3.4578863  90.4121349
  [112,]  65.2591556    4.8847199  96.3587827
  [113,]   2.9343460  518.0404495   4.3735931
  [114,]   4.1988032  516.6578397   6.1742161
  [115,]   3.6006580  176.6208126  24.6992164
  [116,]  81.4339414    7.2179691 118.5147286
  [117,]  47.8891775    4.9922252  87.3032269
  [118,]  29.4174300 1014.8965652  10.3068937
  [119,]  49.6302994    1.8589229  88.8016861
  [120,]  12.5809998  803.8410646   1.8642643
  [121,]   2.2151864  325.0652800   9.9581523
  [122,]   2.4908674  206.1398995  24.0556883
  [123,]  40.8676400    1.8398249  82.9936905
  [124,]  61.8025179   13.3993064  78.2105292
  [125,]  60.2497499    0.3858738  96.5016513
  [126,]   1.3759382  240.7004539  19.4855313
  [127,]  64.5325917    1.1306058  99.9064845
  [128,]   5.6770666  154.6484828  31.7343173
  [129,]  26.4349029 1007.4452780   3.1178975
  [130,]   1.3053205  318.0532919  10.8698082
  [131,]  22.7585004  923.6445235   5.5555328
  [132,]  59.4855849    3.0998615  96.3615610
```

```
[133,]   8.8350687  678.9197130   1.1142330
[134,]  18.0436102  933.1789099   4.0219352
[135,]   8.8725540  504.3698938  12.4785926
>
> kmNew          <- apply(kmTr, byRows, which.min)
> train1$mhCl    <- kmNew
> train1$grp     <- t1Grp
> train1$clust   <- km3clust
> table(train1$grp,   train1$clust)

     1  2  3
  1  0 47  0
  2 40  0  3
  3 12  0 33
> table(train1$clust, train1$mhCl)

     1  2  3
  1 51  0  1
  2  0 47  0
  3  2  0 34
>
> 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$grp     <- t2Grp
> table(test1$grp, test1$mhCl)

    1 2 3
  1 0 3 0
  2 7 0 0
  3 2 0 3
>
> hcDat      <- train1
> hcDat$mhCl <- NULL
> hcDat$grp  <- NULL
> hcDat$clust<- NULL
>
> hc         <- hclust(dist(hcDat)^2,method="ward.D")
>
> 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:   1 638.85274
 2:   2 148.83436
 3:   3  78.40481
 4:   4  65.19046
 5:   5  58.88102
 6:   6  57.70422
 7:   7  60.40264
 8:   8  64.27314
 9:   9  71.54127
10:  10  79.83770
>
> plot(1:maxGrp,hcss$wgss)
>
> (thc       <- dif1(hcss$wgss) )
$`d1`
[1] 490.018384  70.429543  13.214352   6.309439   1.176803  -2.698423  -3.870500
[8]  -7.268124  -8.296429

$d1scaled
[1]  1.000000000  0.143728369  0.026967053  0.012875922  0.002401549
[6] -0.005506778 -0.007898683 -0.014832350 -0.016930852

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