kmeans

MA Mingyu 14110562D

4/4/2017

COMP4433 Assignment 2 Question 3 a, b and c

derek.ma@connect.polyu.hk derek.ma

Set Up

Import data, delete first column, set initial cluster centers to first two records

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#PREPARE DATA IN USE
data_original <- read.csv("~/Google Drive/_DM/2_Assignments/Ass2/data_q3.csv", stringsAsFactors = FALSE)
#delete first column for this specific case becasue it is not a data record
data_original <- data_original[,2:length(colnames(data_original))]
#SET VALUE OF K
data <- data_original
#SET INITIAL CLUSTER CENTERS
centers <- data[0,]
```

Data Preprocessing and Normalization

```
#Normalization
min1 <- min(data$B)
max1 <- max(data$B)
min2 <- min(data$C)
max2 <- max(data$C)
min3 <- min(data$D)
max3 <- max(data$D)
min4 <- min(data$E)
max4 <- max(data$E)
min5 <- min(data$F)
max5 <- max(data$F)
min6 <- min(data$G)
max6 <- max(data$G)
```

```
data_normalized <- data %>%
  mutate(BN = (B - min1)/(max1-min1)) %>%
  mutate(CN = (C - min2)/(max2-min2)) %>%
  mutate(DN = (D - min3)/(max3-min3)) %>%
  mutate(EN = (E - min4)/(max4-min4)) %>%
  mutate(FN = (F - min5)/(max5-min5)) %>%
  mutate(GN = (G - min6)/(max6-min6))

data_normalized<-data_normalized[,c(7,8,9,10,11,12)]
colnames(data_normalized) <- c("B","C","D","E","F","G")
data <- data_normalized
centers <- data[c(1,2),]
```

Basic Function 1 - Distance Function

In this case, Euclidean Distance is used to calculate the dissimilarities.

```
distance <- function(vector1, vector2){
  #Euclidean distance
  #Input: two vectors of data with same length
  #Input example: c(1,2,3); c(2,3,4)
  count <- 0
  for (i in 1:length(vector1)){
    count = count + (vector1[i] - vector2[i])^2
  }
  count^(1/2)
}
distance2 <- function(vector1, vector2){
  #Euclidean distance
  #Input: two vectors of data with same length
  #Input example: c(1,2,3); c(2,3,4)
  count <- 0
  for (i in 1:length(vector1)){
    count = count + (vector1[i] - vector2[i])^2
  }
  count
}
```

Basic Function 2 - Compare Similarity and Assign Objects to Clusters

In this function, each record can be decivded belong to which clusters.

```
assign <- function(objectsData, centersData){
  #OUTPUT a data frame with new cluster information
  #FOR EACH RECORDS
  result <- objectsData %>%
    mutate(cluster = NA)
  for (i in 1:nrow(objectsData)){
    whichCenter <- 0
```

```
    currentMinDist <- -1
    #COUNT DISSIMILARITY BETWEEN IT AND CENTERS
    for (j in 1:nrow(centersData)){
      distValue <- distance(
        as.numeric(objectsData[i,]),
        as.numeric(centersData[j,]))
      if (distValue < currentMinDist || whichCenter == 0){
        #FOUND CENTER WITH SMALLER DISSIMILARITY
        whichCenter <- j
        currentMinDist <- distValue
      }
    }
    #SET THIS CENTER AS CLUSTER
    result[i,"cluster"] <- whichCenter
  }
  result
}
```

Basic Function 3 - Calculate Mean Values of Objects and Update Centers

In this function, the centers will be updated to the mean of clustered objects.

```
update <- function(objectsData, centersData){
  #INPUT  objectsData: the data frame with original data and corresponding cluster information
  #INPUT  centersData: all last round data for all centers
  #OUTPUT a data frame with new centers
  #FOR EACH CENTER
  result <- centersData
  for (i in 1:nrow(centersData)){
    #GET ALL NODES IN THIS CLUSTER
    clusterData <- subset(objectsData, objectsData[,"cluster"] == i)
    #CALCULATE MEAN FOR EACH FEATURE & UPDATE CENTERS
    for (j in 1:ncol(centersData)){
      result[i,j] <- mean(clusterData[,j])
    }
  }
  result
}
```

Question 3a

First Round

Run the algorithm for the first time.

```
data1 <- assign(data, centers)
data1
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       1
```

```
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       1
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       1
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       2
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       2
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       2
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       2
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       2
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       2
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       2
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       1
```

```
centers1 <- update(data1, centers)
centers1
```

```
##           B         C         D         E         F         G
## 1 0.6377551 0.3358896 0.7170139 0.2758621 0.2538168 0.2500000
## 2 0.4021336 0.4771891 0.3232323 0.4440961 0.4559334 0.4545455
```

We can find that some data records are devided to belong to cluster 1 and others are belong to cluster 2. Update the centers.

Second Round

Run the alogrithm for the second time.

```
data2 <- assign(data, centers1)
data2
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       1
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       1
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       2
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       2
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       1
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       1
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       1
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       2
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       1
```

```
centers2 <- update(data2, centers1)
centers2
```

```
##           B         C         D          E          F    G
## 1 0.7798834 0.1501607 0.6200397 0.09031199 0.04907306 0.00
## 2 0.1894133 0.6926892 0.2604167 0.66954023 0.71087786 0.75
```

There are some changes for centers and cluster distribution. Update the centers again.

Third Round

Then run the algorithm for the third time.

```
data3 <- assign(data, centers2)
data3
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       1
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       1
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       2
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       2
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       1
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       1
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       1
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       2
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       1
```

```
centers3 <- update(data3, centers2)
centers3
```

```
##           B         C         D          E          F    G
## 1 0.7798834 0.1501607 0.6200397 0.09031199 0.04907306 0.00
## 2 0.1894133 0.6926892 0.2604167 0.66954023 0.71087786 0.75
```

We can find that the centers are not changed. Thus all objects are divided into two clusters and the final clustering result is already got. The clustering result is:

Result

```
result_2 <- data3
data3 %>% select(cluster)
```

```
##    cluster
## 1        2
## 2        2
## 3        1
## 4        1
## 5        2
## 6        2
## 7        2
## 8        2
## 9        1
## 10       1
## 11       1
## 12       2
## 13       1
## 14       2
## 15       1
```

```
centers3
```

```
##           B         C         D          E          F    G
## 1 0.7798834 0.1501607 0.6200397 0.09031199 0.04907306 0.00
## 2 0.1894133 0.6926892 0.2604167 0.66954023 0.71087786 0.75
```

Question 3b

First Round

```
centers <- data[c(13,14,15),]
data1 <- assign(data, centers)
data1
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       2
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       2
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       1
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       2
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers1 <- update(data1, centers)
centers1
```

```
##            B          C         D          E          F    G
## 13 0.6326531 0.30879346 0.3784722 0.19540230 0.14503817 0.00
## 14 0.1894133 0.69268916 0.2604167 0.66954023 0.71087786 0.75
## 15 0.8387755 0.08670757 0.7166667 0.04827586 0.01068702 0.00
```

Second Round

```
data2 <- assign(data, centers1)
data2
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       2
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       1
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
```

```
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       1
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       1
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers2 <- update(data2, centers1)
centers2
```

```
##            B          C         D          E          F         G
## 13 0.5280612 0.35276074 0.4322917 0.28160920 0.24141221 0.2500000
## 14 0.1113946 0.79134288 0.1851852 0.77011494 0.83524173 0.8333333
## 15 0.8387755 0.08670757 0.7166667 0.04827586 0.01068702 0.0000000
```

Third Round

```
data3 <- assign(data, centers2)
data3
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       1
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       1
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       1
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       1
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers3 <- update(data3, centers2)
centers3
```

```
##             B          C         D          E          F   G
## 13 0.47448980 0.40204499 0.4111111 0.31034483 0.29465649 0.3
## 14 0.08163265 0.82977505 0.1569444 0.83908046 0.90076336 0.9
## 15 0.83877551 0.08670757 0.7166667 0.04827586 0.01068702 0.0
```

Fourth Round

```
data4 <- assign(data, centers3)
data4
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
```

```
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       1
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       1
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       2
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       3
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       1
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers4 <- update(data4, centers3)
centers4
```

```
##             B         C         D          E         F     G
## 13 0.44132653 0.4253579 0.3715278 0.35344828 0.3463740 0.375
## 14 0.08163265 0.8297751 0.1569444 0.83908046 0.9007634 0.900
## 15 0.80017007 0.1237219 0.6921296 0.06321839 0.0235369 0.000
```

Fifth Round

```
data5 <- assign(data, centers4)
data5
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       1
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       1
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       1
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       3
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       1
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers5 <- update(data5, centers4)
centers5
```

```
##             B         C         D          E         F   G
## 13 0.37959184 0.4625767 0.3625000 0.40229885 0.3992366 0.4
## 14 0.06887755 0.8843558 0.1145833 0.89942529 0.9732824 1.0
## 15 0.80017007 0.1237219 0.6921296 0.06321839 0.0235369 0.0
```

Sixth Round

```
data6 <- assign(data, centers5)
data6
```

```
##             B          C         D          E          F   G cluster
## 1  0.07142857 1.00000000 0.3263889 1.00000000 0.96183206 1.0       2
## 2  0.00000000 0.87484663 0.0000000 0.85057471 0.96564885 1.0       2
## 3  0.73979592 0.09815951 1.0000000 0.00000000 0.00000000 0.0       3
## 4  0.78571429 0.12883436 0.7708333 0.08045977 0.02671756 0.0       3
## 5  0.26020408 0.59918200 0.3263889 0.42528736 0.50763359 0.5       1
## 6  0.27040816 0.47239264 0.5694444 0.19540230 0.38931298 0.5       1
## 7  0.13265306 0.61145194 0.3263889 0.59770115 0.61068702 0.5       1
## 8  0.05102041 0.83435583 0.0000000 0.80459770 0.96564885 1.0       2
## 9  1.00000000 0.00000000 0.4930556 0.13793103 0.00000000 0.0       3
## 10 0.60714286 0.30879346 0.5694444 0.13793103 0.08778626 0.0       3
## 11 0.71428571 0.08997955 0.5486111 0.00000000 0.00000000 0.0       3
## 12 0.15306122 0.82822086 0.1319444 0.94252874 1.00000000 1.0       2
## 13 0.65816327 0.30879346 0.1875000 0.25287356 0.20229008 0.0       1
## 14 0.57653061 0.32106339 0.4027778 0.54022989 0.28625954 0.5       1
## 15 0.95408163 0.11656442 0.7708333 0.02298851 0.02671756 0.0       3
```

```
centers6 <- update(data6, centers5)
centers6
```

```
##             B         C         D          E         F   G
## 13 0.37959184 0.4625767 0.3625000 0.40229885 0.3992366 0.4
## 14 0.06887755 0.8843558 0.1145833 0.89942529 0.9732824 1.0
## 15 0.80017007 0.1237219 0.6921296 0.06321839 0.0235369 0.0
```

We can found the coordinates for centers and cluster distrbution are not changed from round 5 to round 6. Thus we can terminate the iteration and find out the result:

Result

```
result_3 <- data6
data6 %>% select(cluster)
```

```
##    cluster
## 1        2
## 2        2
## 3        3
## 4        3
## 5        1
## 6        1
## 7        1
## 8        2
## 9        3
## 10       3
## 11       3
## 12       2
## 13       1
## 14       1
## 15       3
```

```
centers6
```

```
##             B         C         D          E         F   G
## 13 0.37959184 0.4625767 0.3625000 0.40229885 0.3992366 0.4
## 14 0.06887755 0.8843558 0.1145833 0.89942529 0.9732824 1.0
## 15 0.80017007 0.1237219 0.6921296 0.06321839 0.0235369 0.0
```

Question 3c

Calculate the CH index for the clustering result from (a) and (b), the performance should be better for the one with higher CH index.

Funcion - calculate within cluster variation

```
withinClusterVariation <- function(objectsData){
  result <- rep(NA,length(unique(objectsData$cluster)))
  for (clusterIndex in unique(objectsData$cluster)){
    temp <- subset(objectsData, cluster==clusterIndex)
    center <- rep(NA,ncol(objectsData)-1)
    for (i in 1:(ncol(objectsData)-1)){
      center[i] <- mean(temp[,i])
    }
    temp <- temp[,1:6]
    sum <- 0
    for (i in 1:nrow(temp)){
      sum <- sum + distance2(center,as.numeric(temp[i,]))
    }
    result[clusterIndex] <- sum
  }
  sum(result)
}
```

Funcion - calculate between cluster variation

```
betweenClusterVariation <- function(objectsData){
  result <- rep(NA,length(unique(objectsData$cluster)))
  center <- rep(NA,ncol(objectsData)-1)
  for (i in 1:(ncol(objectsData)-1)){
    center[i] <- mean(objectsData[,i])
  }
  for (clusterIndex in unique(objectsData$cluster)){
    temp <- subset(objectsData, cluster==clusterIndex)
    center1 <- rep(NA,ncol(objectsData)-1)
    for (i in 1:(ncol(objectsData)-1)){
      center1[i] <- mean(temp[,i])
    }
    sum <- 0
    for (i in 1:nrow(temp)){
      sum <- sum + distance2(center,center1)
```

```
    }
    result[clusterIndex] <- sum
  }
  sum(result)
}
```

Calculate CH Index

```
w2 <- withinClusterVariation(result_2)
w3 <- withinClusterVariation(result_3)
b2 <- betweenClusterVariation(result_2)
b3 <- betweenClusterVariation(result_2)
ch_2 <- ((b2/(2-1))/(w2/(15-2)))
ch_3 <- ((b3/(3-1))/(w3/(15-3)))
w2
```

```
## [1] 3.232459
```

```
w3
```

```
## [1] 1.296679
```

```
ch_2
```

```
## [1] 31.65498
```

```
ch_3
```

```
## [1] 36.42087
```

Result

CH index for `k=2` is 31.65 which is small than CH index for `k=3`'s 36.42. For within cluster variation, `k=2` is larger than `k=3`. These two index both show that the performance of `k=3` is better than `k=2`.