k-means exercise using Wine Data

Tom Thorpe

Aug 15, 2018

Objective

Use clustering for wine data

```
# This mini-project is based on the K-Means exercise from 'R in Action'
# Go here for the original blog post and solutions
# http://www.r-bloggers.com/k-means-clustering-from-r-in-action/

# Exercise 0: Install these packages if you don't have them already

# install.packages(c("cluster", "rattle.data","NbClust"))

# Now load the data and look at the first few rows

data(wine, package="rattle.data")
str(wine)
```

```
## 'data.frame':    178 obs. of  14 variables:
##  $ Type           : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Alcohol        : num  14.2 13.2 13.2 14.4 13.2 ...
##  $ Malic          : num  1.71 1.78 2.36 1.95 2.59 1.76 1.87 2.15 1.64 1.35 ...
##  $ Ash            : num  2.43 2.14 2.67 2.5 2.87 2.45 2.45 2.61 2.17 2.27 ...
##  $ Alcalinity     : num  15.6 11.2 18.6 16.8 21 15.2 14.6 17.6 14 16 ...
##  $ Magnesium      : int  127 100 101 113 118 112 96 121 97 98 ...
##  $ Phenols        : num  2.8 2.65 2.8 3.85 2.8 3.27 2.5 2.6 2.8 2.98 ...
##  $ Flavanoids     : num  3.06 2.76 3.24 3.49 2.69 3.39 2.52 2.51 2.98 3.15 ...
##  $ Nonflavanoids  : num  0.28 0.26 0.3 0.24 0.39 0.34 0.3 0.31 0.29 0.22 ...
##  $ Proanthocyanins: num  2.29 1.28 2.81 2.18 1.82 1.97 1.98 1.25 1.98 1.85 ...
##  $ Color          : num  5.64 4.38 5.68 7.8 4.32 6.75 5.25 5.05 5.2 7.22 ...
##  $ Hue            : num  1.04 1.05 1.03 0.86 1.04 1.05 1.02 1.06 1.08 1.01 ...
##  $ Dilution       : num  3.92 3.4 3.17 3.45 2.93 2.85 3.58 3.58 2.85 3.55 ...
##  $ Proline        : int  1065 1050 1185 1480 735 1450 1290 1295 1045 1045 ...
```

```
nrow(wine)
```

```
## [1] 178
```

```
head(wine)
```

```
##   Type Alcohol Malic  Ash Alcalinity Magnesium Phenols Flavanoids
## 1    1   14.23  1.71 2.43       15.6       127    2.80       3.06
## 2    1   13.20  1.78 2.14       11.2       100    2.65       2.76
## 3    1   13.16  2.36 2.67       18.6       101    2.80       3.24
## 4    1   14.37  1.95 2.50       16.8       113    3.85       3.49
## 5    1   13.24  2.59 2.87       21.0       118    2.80       2.69
## 6    1   14.20  1.76 2.45       15.2       112    3.27       3.39
##   Nonflavanoids Proanthocyanins Color  Hue Dilution Proline
## 1          0.28            2.29  5.64 1.04     3.92    1065
## 2          0.26            1.28  4.38 1.05     3.40    1050
## 3          0.30            2.81  5.68 1.03     3.17    1185
## 4          0.24            2.18  7.80 0.86     3.45    1480
## 5          0.39            1.82  4.32 1.04     2.93     735
## 6          0.34            1.97  6.75 1.05     2.85    1450
```

```
#knitr::knit_exit()

# Exercise 1: Remove the first column from the data and scale
# it using the scale() function

# the following are some calculations to understand the scale function
avgAlc=mean(wine$Alcohol)
sdAlc=sd(wine$Alcohol)
avgAlc
```

```
## [1] 13.00062
```

```
sdAlc
```

```
## [1] 0.8118265
```

```
(wine$Alcohol[1]-avgAlc)/sdAlc
```

```
## [1] 1.514341
```

```
# scale(x, center = TRUE, scale = TRUE)
# wine[-1] removes the first column, so the remaining columns are scaled
# center with default = true, subracts the mean of the column
# scale with default = true, divides by the standard deviation
df <- scale(wine[-1])  # remove first column and scale

str(df)
```

```
##  num [1:178, 1:13] 1.514 0.246 0.196 1.687 0.295 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  - attr(*, "scaled:center")= Named num [1:13] 13 2.34 2.37 19.49 99.74 ...
##   ..- attr(*, "names")= chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  - attr(*, "scaled:scale")= Named num [1:13] 0.812 1.117 0.274 3.34 14.282 ...
##   ..- attr(*, "names")= chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
```

```
head(df)
```

```
##        Alcohol       Malic        Ash Alcalinity  Magnesium   Phenols
## [1,] 1.5143408 -0.56066822  0.2313998 -1.1663032 1.90852151 0.8067217
## [2,] 0.2455968 -0.49800856 -0.8256672 -2.4838405 0.01809398 0.5670481
## [3,] 0.1963252  0.02117152  1.1062139 -0.2679823 0.08810981 0.8067217
## [4,] 1.6867914 -0.34583508  0.4865539 -0.8069748 0.92829983 2.4844372
## [5,] 0.2948684  0.22705328  1.8352256  0.4506745 1.27837900 0.8067217
## [6,] 1.4773871 -0.51591132  0.3043010 -1.2860793 0.85828399 1.5576991
##      Flavanoids Nonflavanoids Proanthocyanins      Color        Hue
## [1,]  1.0319081    -0.6577078       1.2214385  0.2510088  0.3611585
## [2,]  0.7315653    -0.8184106      -0.5431887 -0.2924962  0.4049085
## [3,]  1.2121137    -0.4970050       2.1299594  0.2682629  0.3174085
## [4,]  1.4623994    -0.9791134       1.0292513  1.1827317 -0.4263410
## [5,]  0.6614853     0.2261576       0.4002753 -0.3183774  0.3611585
## [6,]  1.3622851    -0.1755994       0.6623487  0.7298108  0.4049085
##       Dilution     Proline
## [1,] 1.8427215  1.01015939
## [2,] 1.1103172  0.96252635
## [3,] 0.7863692  1.39122370
## [4,] 1.1807407  2.32800680
## [5,] 0.4483365 -0.03776747
## [6,] 0.3356589  2.23274072
```

```
# Now we'd like to cluster the data using K-Means.
# How do we decide how many clusters to use if you don't know that already?
# We'll try two methods.

# Method 1: A plot of the total within-groups sums of squares against the
# number of clusters in a K-means solution can be helpful. A bend in the
# graph can suggest the appropriate number of clusters.

wssplot <- function(data, nc=15, seed=1234){
    wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
      set.seed(seed)

    wss[i] <- sum(kmeans(data, centers=i)$withinss)
  }

    plot(1:nc, wss, type="b", xlab="Number of Clusters",
                            ylab="Within groups sum of squares")
}

sumVal=sum(apply(df,2,var))
sumVal
```

```
## [1] 13
```

```
str(sumVal)
```

```
##  num 13
```

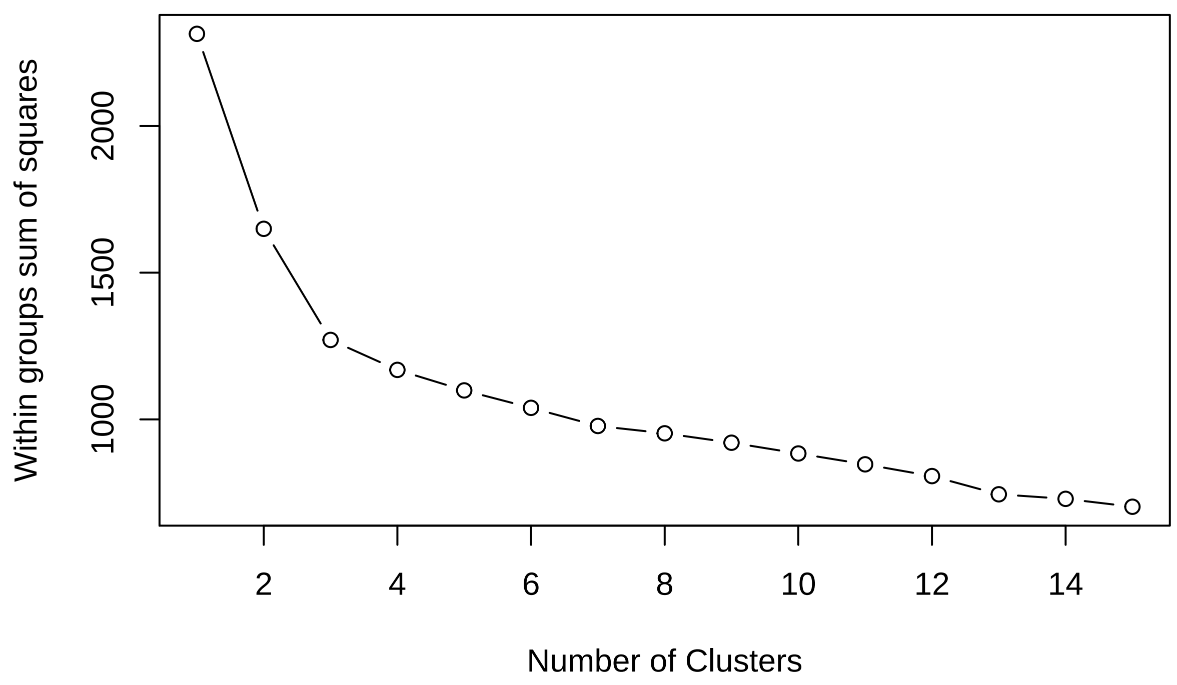
```
wss<-(nrow(df-1))*sumVal
wss
```

```
## [1] 2314
```

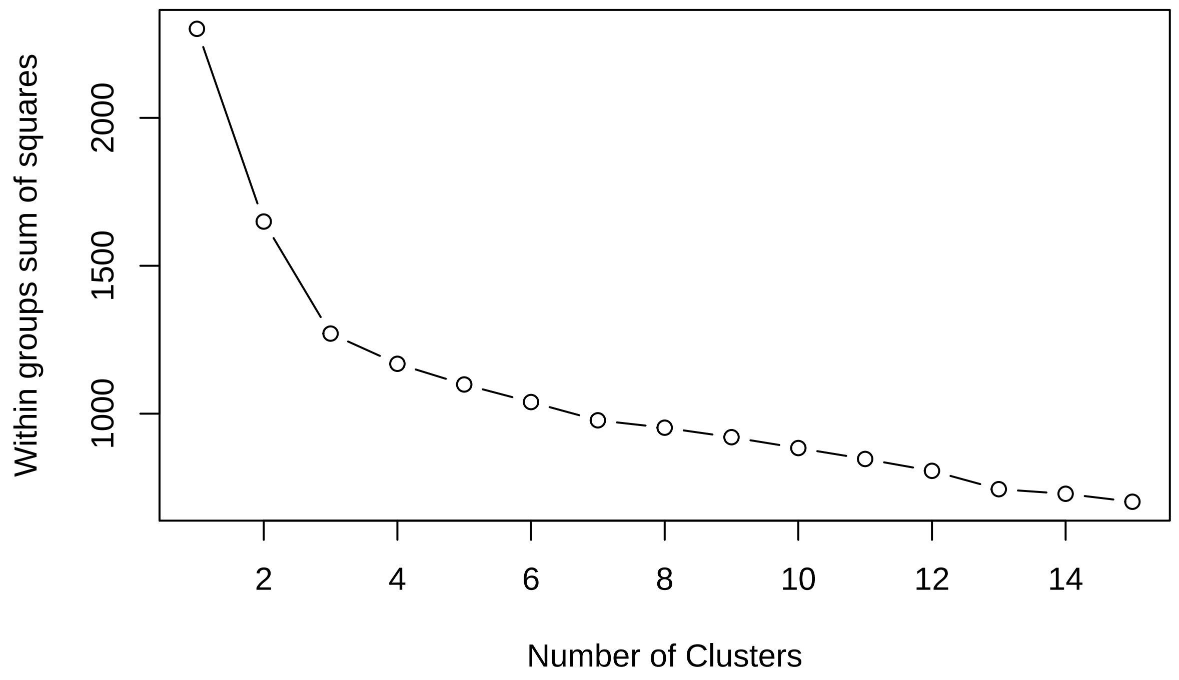
```
for (i in 2:15) {
  set.seed(1234)
  km = kmeans(df, centers=i)
  str(km)
  km
  wss[i] <- sum(km$withinss)
  print(wss[i])
}
```

```
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:2, 1:13] 0.3249 -0.3106 -0.3529 0.3374 0.0521 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "1" "2"
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:2] 765 884
##  $ tot.withinss: num 1649
##  $ betweenss   : num 652
##  $ size        : int [1:2] 87 91
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 1649.44
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 1 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:13] 0.833 -0.923 0.164 -0.303 -0.393 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:3] 386 559 326
##  $ tot.withinss: num 1271
##  $ betweenss   : num 1030
##  $ size        : int [1:3] 62 65 51
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 1270.749
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 2 1 1 1 1 1 ...
##  $ centers     : num [1:4, 1:13] 0.958 -0.787 0.186 -0.905 -0.377 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "1" "2" "3" "4"
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:4] 269 307 303 290
##  $ tot.withinss: num 1169
##  $ betweenss   : num 1132
##  $ size        : int [1:4] 56 28 49 45
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 1168.614
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 4 1 1 1 1 1 ...
##  $ centers     : num [1:5, 1:13] 0.982 -0.772 -0.909 -0.669 0.186 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:5] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:5] 243 201 247 105 303
##  $ tot.withinss: num 1099
##  $ betweenss   : num 1202
##  $ size        : int [1:5] 54 30 37 8 49
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 1098.739
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 4 1 1 1 1 1 ...
##  $ centers     : num [1:6, 1:13] 0.982 -0.758 -0.926 -0.669 -0.105 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:6] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:6] 243 193 236 105 177 ...
##  $ tot.withinss: num 1039
##  $ betweenss   : num 1262
##  $ size        : int [1:6] 54 29 36 8 33 18
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 1039.296
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 4 1 1 1 1 1 ...
##  $ centers     : num [1:7, 1:13] 0.9998 -0.7509 -1.0165 -0.3738 -0.0639 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:7] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:7] 227.2 130.2 132.6 73.4 169 ...
##  $ tot.withinss: num 978
##  $ betweenss   : num 1323
##  $ size        : int [1:7] 52 20 24 7 32 19 24
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 977.541
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 4 1 1 1 1 1 ...
##  $ centers     : num [1:8, 1:13] 0.9683 -0.9171 -1.0269 -0.592 -0.0639 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:8] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:8] 209.9 115.5 128.1 63.6 169 ...
##  $ tot.withinss: num 953
##  $ betweenss   : num 1348
##  $ size        : int [1:8] 49 18 23 6 32 19 24 7
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 952.5328
## List of 9
##  $ cluster     : int [1:178] 1 1 1 1 4 1 1 1 1 1 ...
##  $ centers     : num [1:9, 1:13] 0.968 -0.951 -0.895 -0.592 -0.054 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:9] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:9] 209.9 53.4 84.5 63.6 159.8 ...
##  $ tot.withinss: num 920
##  $ betweenss   : num 1381
##  $ size        : int [1:9] 49 8 16 6 31 18 14 7 29
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 920.4558
## List of 9
##  $ cluster     : int [1:178] 1 7 1 1 4 1 7 1 7 7 ...
##  $ centers     : num [1:10, 1:13] 1.145 -0.951 -0.895 -0.592 -0.054 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:10] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:10] 101.1 53.4 84.5 63.6 159.8 ...
##  $ tot.withinss: num 884
##  $ betweenss   : num 1417
##  $ size        : int [1:10] 26 8 16 6 31 18 24 7 32 10
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 883.7607
## List of 9
##  $ cluster     : int [1:178] 1 7 1 1 4 1 7 1 7 7 ...
##  $ centers     : num [1:11, 1:13] 1.145 -0.951 -0.526 -0.592 -0.042 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:11] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:11] 101.1 53.4 75.6 63.6 137.4 ...
##  $ tot.withinss: num 847
##  $ betweenss   : num 1454
##  $ size        : int [1:11] 26 8 12 6 29 18 24 7 19 15 ...
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 846.7963
## List of 9
##  $ cluster     : int [1:178] 1 7 1 1 4 1 7 1 7 7 ...
##  $ centers     : num [1:12, 1:13] 1.145 -0.951 0.086 -0.417 -0.203 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:12] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:12] 101.1 53.4 90.1 48.2 53.1 ...
##  $ tot.withinss: num 807
##  $ betweenss   : num 1494
##  $ size        : int [1:12] 26 8 23 5 11 15 24 7 22 12 ...
##  $ iter        : int 4
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 806.6972
## List of 9
##  $ cluster     : int [1:178] 8 7 13 13 4 13 7 7 8 8 ...
##  $ centers     : num [1:13, 1:13] -0.818 -0.812 0.086 -0.358 -0.203 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:13] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:13] 9.16 59.02 90.08 24.64 53.09 ...
##  $ tot.withinss: num 745
##  $ betweenss   : num 1556
##  $ size        : int [1:13] 3 9 23 4 11 15 22 14 18 12 ...
##  $ iter        : int 6
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 744.7018
## List of 9
##  $ cluster     : int [1:178] 8 7 13 13 4 13 7 8 7 7 ...
##  $ centers     : num [1:14, 1:13] -0.818 -0.951 0.18 -0.358 0.138 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:14] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:14] 9.16 53.45 65.57 24.64 55.23 ...
##  $ tot.withinss: num 729
##  $ betweenss   : num 1572
##  $ size        : int [1:14] 3 8 18 4 16 6 21 15 18 12 ...
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 729.1297
## List of 9
##  $ cluster     : int [1:178] 8 8 15 15 4 15 8 7 8 8 ...
##  $ centers     : num [1:15, 1:13] -0.8178 -0.984 0.0675 -0.358 0.4575 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:15] "1" "2" "3" "4" ...
##   .. ..$ : chr [1:13] "Alcohol" "Malic" "Ash" "Alcalinity" ...
##  $ totss       : num 2301
##  $ withinss    : num [1:15] 9.16 50.79 87.51 24.64 48.71 ...
##  $ tot.withinss: num 702
##  $ betweenss   : num 1599
##  $ size        : int [1:15] 3 11 22 4 15 2 20 18 6 13 ...
##  $ iter        : int 5
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
## [1] 702.2454
```

```r
plot(1:15, wss, type="b", xlab="Number of Clusters",
                              ylab="Within groups sum of squares")
```


Within groups sum of squares
1000
1500
2000
2
4
6
8
10
12
14
Number of Clusters

```r
wssplot(df)
```

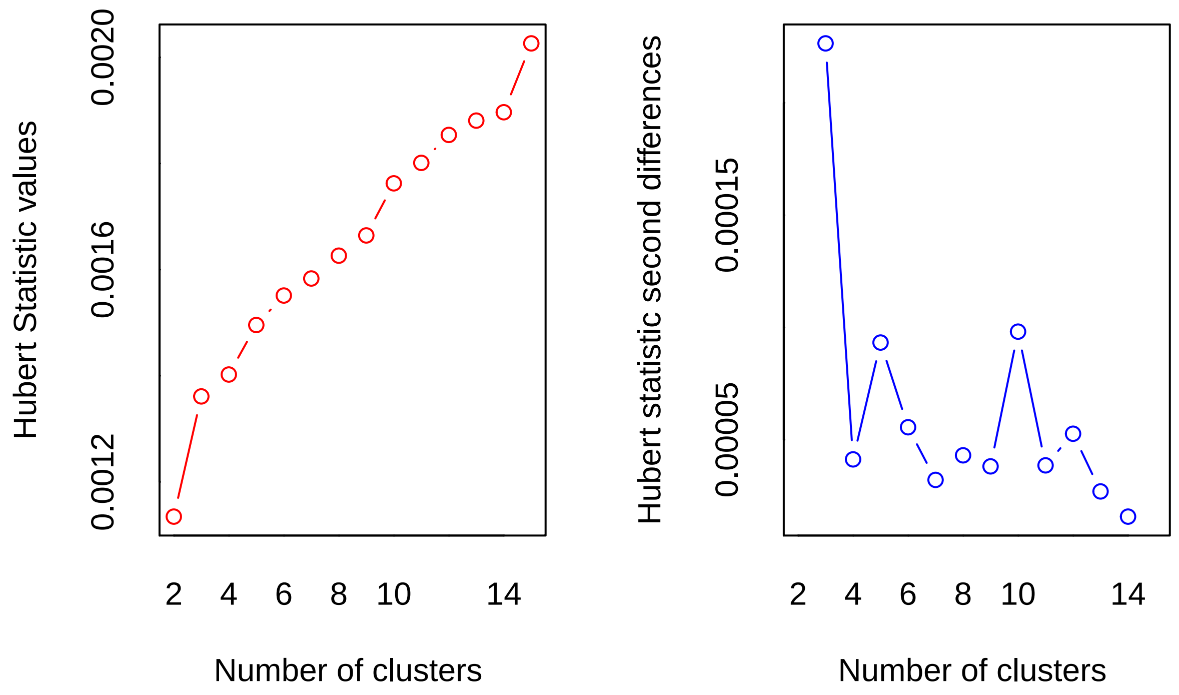


```
# Exercise 2:
#   * How many clusters does this method suggest?
#   * Why does this method work? What's the intuition behind it?
#   * Look at the code for wssplot() and figure out how it works
```

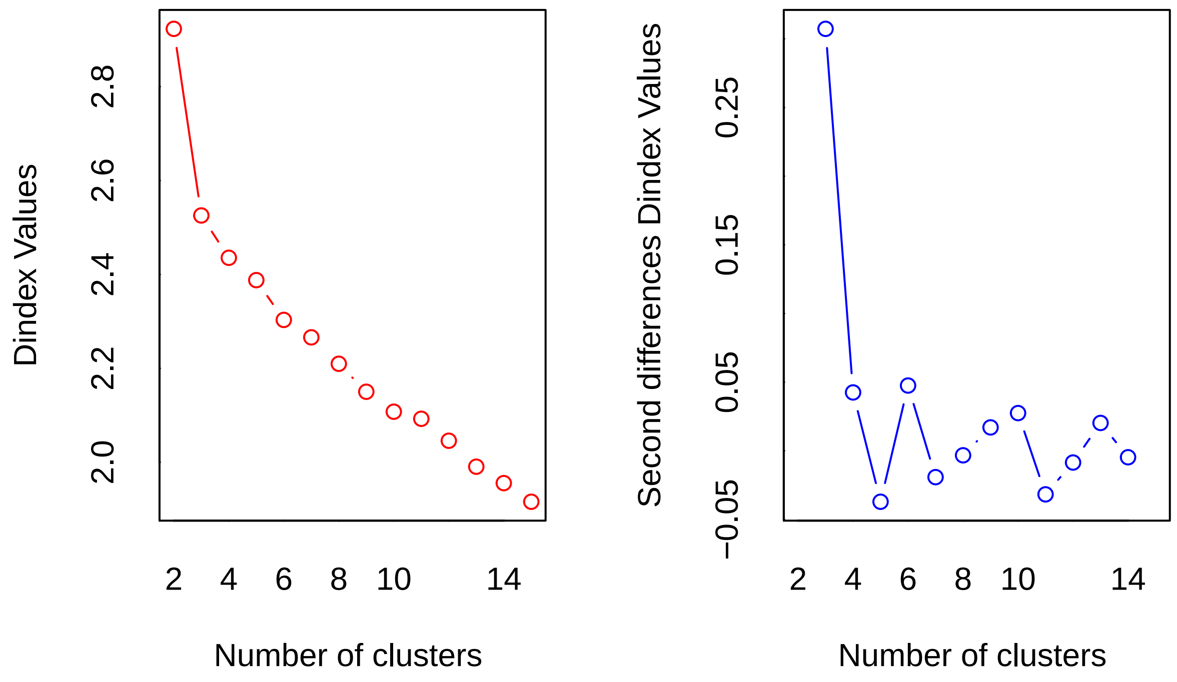
This method suggests 3 clusters. I am not really sure how it works. Need to investigate.

```
# Method 2: Use the NbClust library, which runs many experiments
# and gives a distribution of potential number of clusters.

library(NbClust)
set.seed(1234)
nc <- NbClust(df, min.nc=2, max.nc=15, method="kmeans")
```



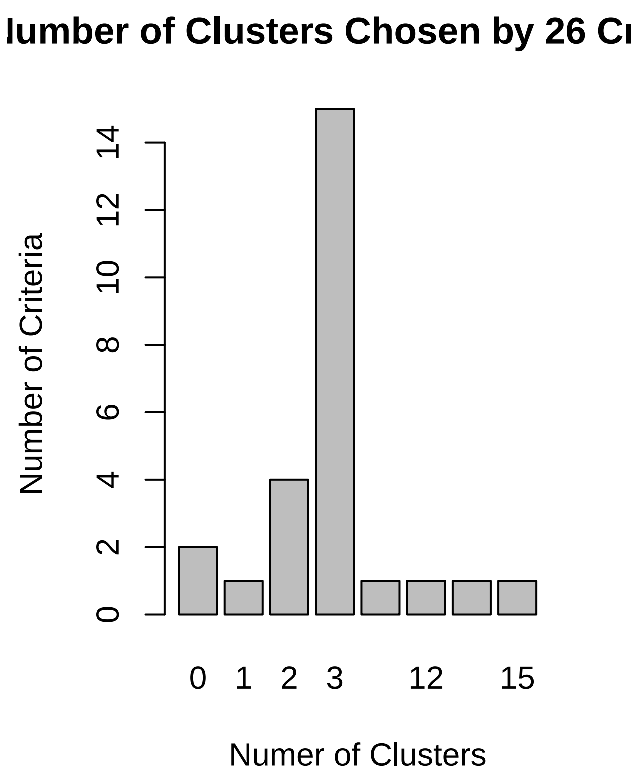
```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 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:
## * 4 proposed 2 as the best number of clusters
## * 15 proposed 3 as the best number of clusters
## * 1 proposed 10 as the best number of clusters
## * 1 proposed 12 as the best number of clusters
## * 1 proposed 14 as the best number of clusters
## * 1 proposed 15 as the best number of clusters
##
##                    ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************
```

```r
barplot(table(nc$Best.n[1,]),
              xlab="Numer of Clusters", ylab="Number of Criteria",
                    main="Number of Clusters Chosen by 26 Criteria")


# Exercise 3: How many clusters does this method suggest?
```



This also suggests 3 clusters.

```
# Exercise 4: Once you've picked the number of clusters, run k-means
# using this number of clusters. Output the result of calling kmeans()
# into a variable fit.km

# fit.km <- kmeans( ... )

fit.km <- kmeans(df,centers=3)

fit.km
```

```
## K-means clustering with 3 clusters of sizes 51, 65, 62
##
## Cluster means:
##      Alcohol      Malic         Ash Alcalinity   Magnesium     Phenols
## 1  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548
## 2 -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891
## 3  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724
##    Flavanoids Nonflavanoids Proanthocyanins      Color        Hue
## 1 -1.21182921    0.72402116     -0.77751312  0.9388902 -1.1615122
## 2  0.02075402   -0.03343924      0.05810161 -0.8993770  0.4605046
## 3  0.97506900   -0.56050853      0.57865427  0.1705823  0.4726504
##     Dilution    Proline
## 1 -1.2887761 -0.4059428
## 2  0.2700025 -0.7517257
## 3  0.7770551  1.1220202
##
## Clustering vector:
##   [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 1 2 2 2 2 2 2 2
##  [71] 2 2 2 3 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2
## [106] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 3 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
## [141] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [176] 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 326.3537 558.6971 385.6983
##  (between_SS / total_SS =  44.8 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

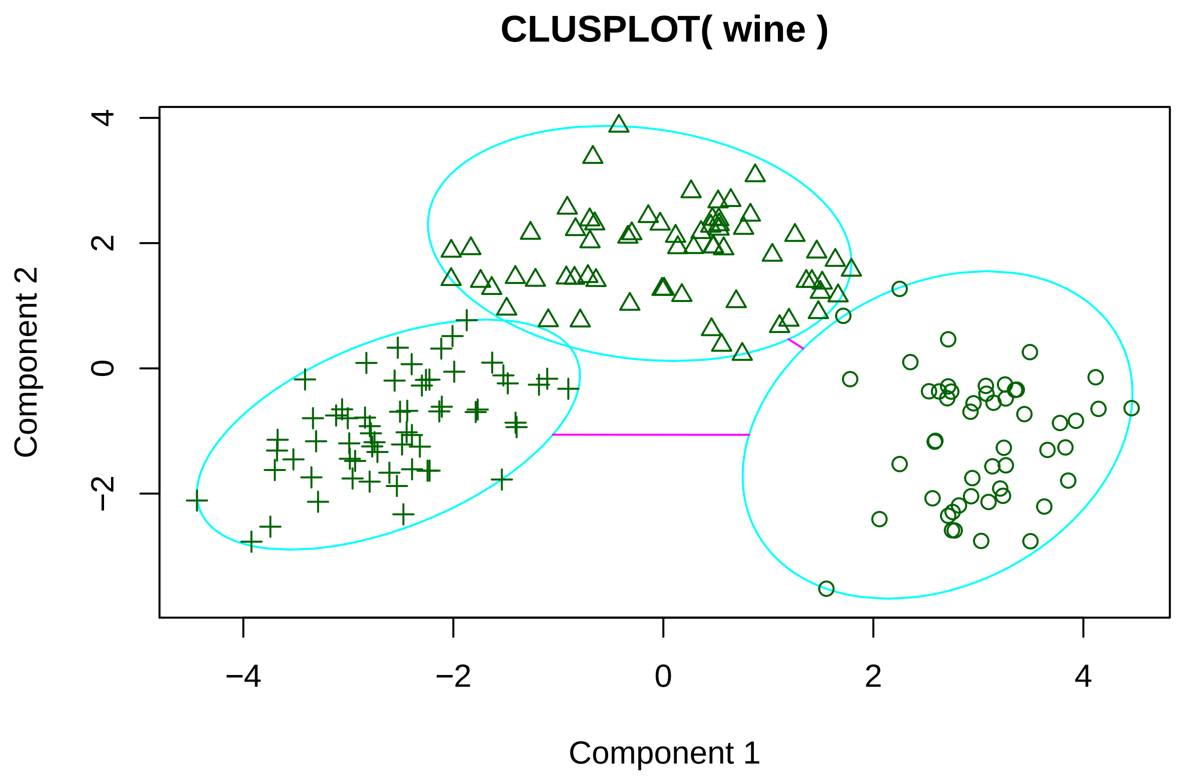
```
# Now we want to evaluate how well this clustering does.

# Exercise 5: using the table() function, show how the clusters in fit.km$clusters
# compares to the actual wine types in wine$Type. Would you consider this a good
# clustering?

table(wine$Type,fit.km$cluster)
```

```
##
##      1  2  3
##   1  0  0 59
##   2  3 65  3
##   3 48  0  0
```

```
# Exercise 6:
# * Visualize these clusters using  function clusplot() from the cluster library
# * Would you consider this a good clustering?
library(cluster)
clusplot(wine, fit.km$cluster )
```



These two components explain 57.38 % of the point variability.