k-means Analysis of UCI Wine Dataset

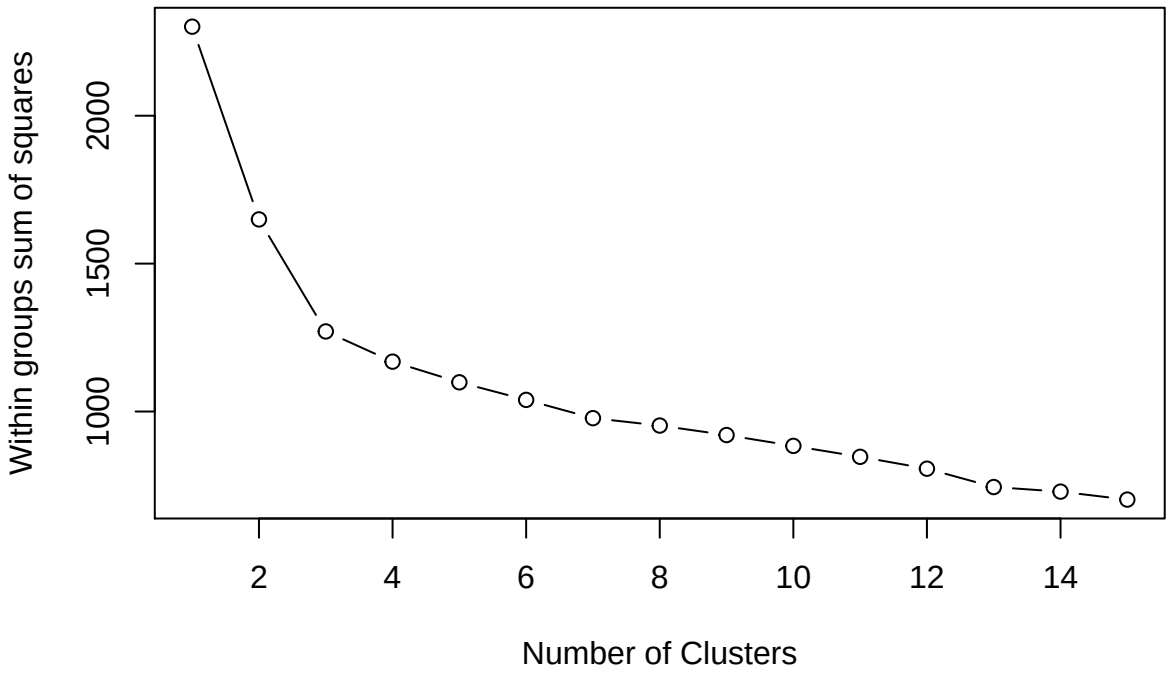
Jon Kinsey

Thu Dec 11 23:39:02 2014

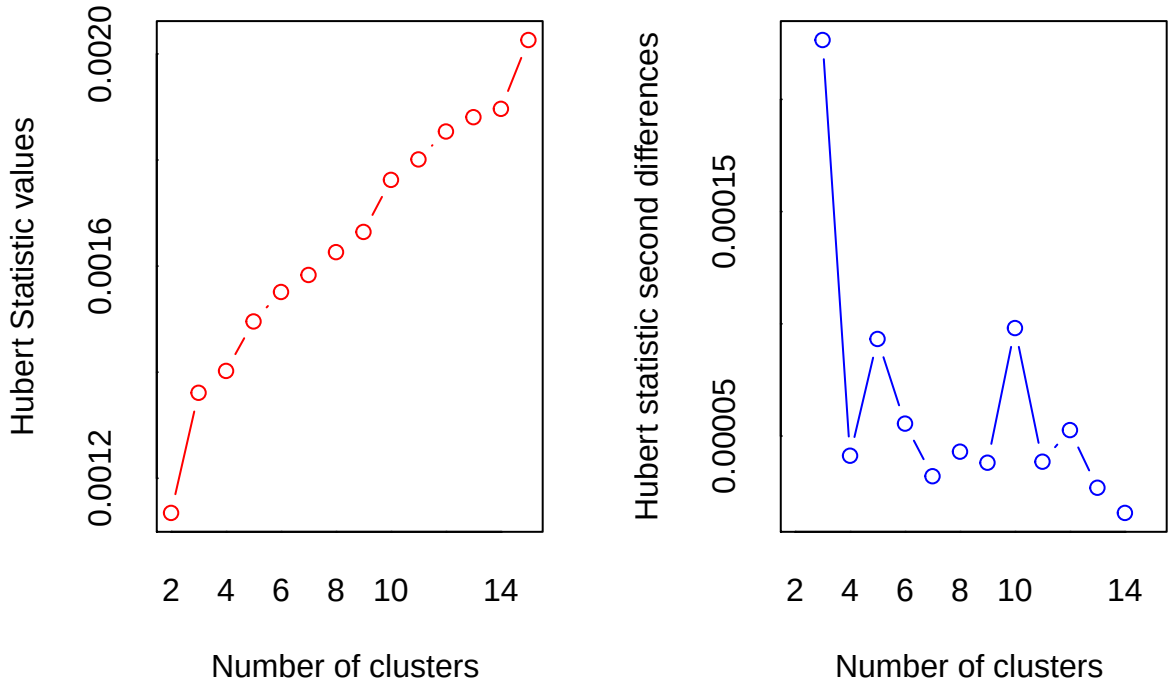
```
# The wine dataset containing 13 chemical measurements on 178 Italian wine
# samples is analyzed. In this data are the results of a chemical analysis of
# wines grown in the same region in Italy but derived from three different cultivars.
# The analysis determined the quantities of 13 constituents found in each of the
# three types of wines. The data originally come from the UCI Machine Learning
# Repository (http://www.ics.uci.edu/~mlearn/MLRepository.html) but we will
# access it via the rattle package. From R in Action by Rob Kabacoff
#
library(NbClust)
#
# First, define a function to plot the total within-groups sums of squares against
# the number of clusters in a K-means solution
# Here, the data parameter is the numeric dataset to be analyzed, nc is the
# maximum number of clusters to consider, and seed is a random number seed
# (see comments below).
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")}
#
data(wine, package="rattle")
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
```

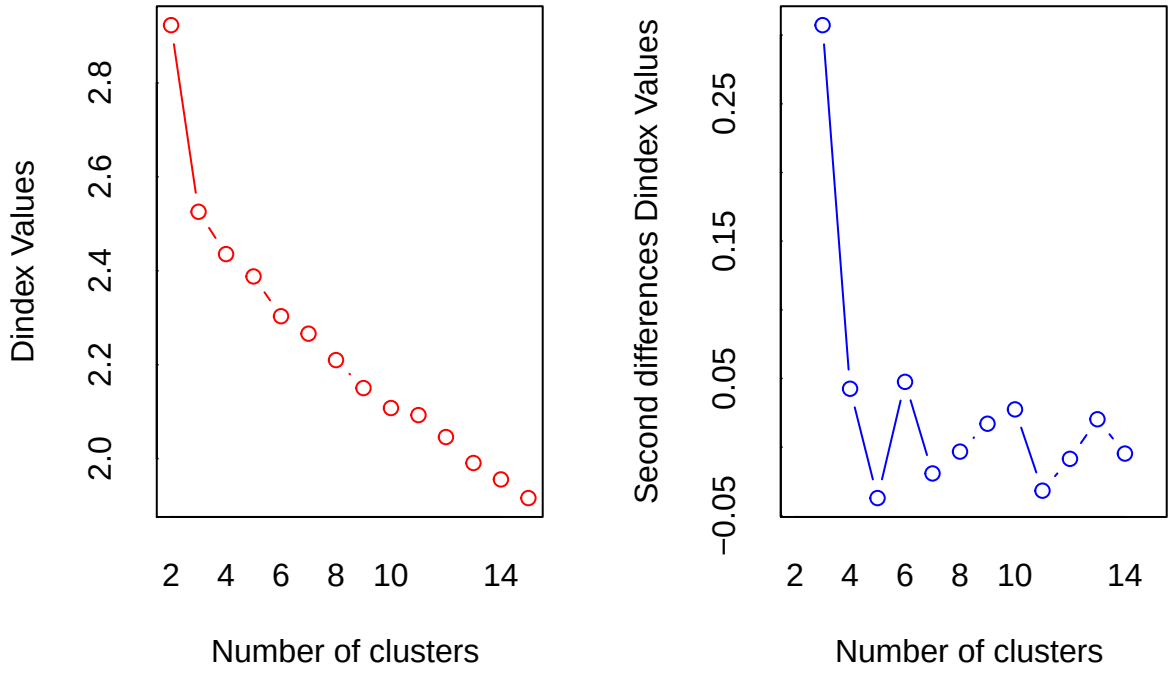
```
# Since the variables vary in range, they are standardized prior to clustering
df <- scale(wine[-1])
#
wssplot(df)
```




```
# The plot indicates that there is a distinct drop in within groups sum of
# squares going from 1 to 3 clusters. After three clusters, this decrease
# drops off, suggesting that a 3-cluster solution may be a good fit to the data.
#
# K-means clustering requires that the number of clusters to extract be
# specified in advance. So, we can use the NbClust package as a guide.
# Since K-means cluster analysis starts with k randomly chosen centroids,
# a different solution can be obtained each time the function is invoked.
# Use the set.seed() function to guarantee that the results are reproducible.
# Additionally, this clustering approach can be sensitive to the initial
# selection of centroids. The kmeans() function has an nstart option that
# attempts multiple initial configurations and reports on the best one.
# For example, adding nstart=25 will generate 25 initial configurations.
# This approach is often recommended.
#
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.
##
## All 178 observations were used.
##
## ******************************************************************
## * 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
##
##
## ******************************************************************
```

```
table(nc$Best.n[1,])
```

```
##
##  0  1  2  3 10 12 14 15
##  2  1  4 15  1  1  1  1
```

```
par(mfrow=c(1,1))
barplot(table(nc$Best.n[1,]),
        xlab="Numer of Clusters", ylab="Number of Criteria",
        main="Number of Clusters Chosen by 26 Criteria")
#
# This plot shows that 14 of 26 criteria provided by the NbClust package
# suggest a 3-cluster solution.
#
# Next, a final cluster solution is obtained with kmeans() function and
# the cluster centroids are printed.
set.seed(1234)
fit.km <- kmeans(df, 3, nstart=25)
fit.km$size
```

```
## [1] 62 65 51
```

```
fit.km$centers
```

```
##      Alcohol      Malic        Ash Alcalinity   Magnesium     Phenols
## 1  0.8328826 -0.3029551  0.3636801 -0.6084749  0.57596208  0.88274724
## 2 -0.9234669 -0.3929331 -0.4931257  0.1701220 -0.49032869 -0.07576891
## 3  0.1644436  0.8690954  0.1863726  0.5228924 -0.07526047 -0.97657548
##    Flavanoids Nonflavanoids Proanthocyanins      Color        Hue
## 1  0.97506900   -0.56050853      0.57865427  0.1705823  0.4726504
## 2  0.02075402   -0.03343924      0.05810161 -0.8993770  0.4605046
## 3 -1.21182921    0.72402116     -0.77751312  0.9388902 -1.1615122
```

```
##     Dilution    Proline
## 1  0.7770551  1.1220202
## 2  0.2700025 -0.7517257
## 3 -1.2887761 -0.4059428
```

```
#
# Since the centroids provided by the function are based on standardized data,
# the aggregate() function is used along with the cluster memberships to determine
# variable means for each cluster in the original metric.
aggregate(wine[-1], by=list(cluster=fit.km$cluster), mean)
```

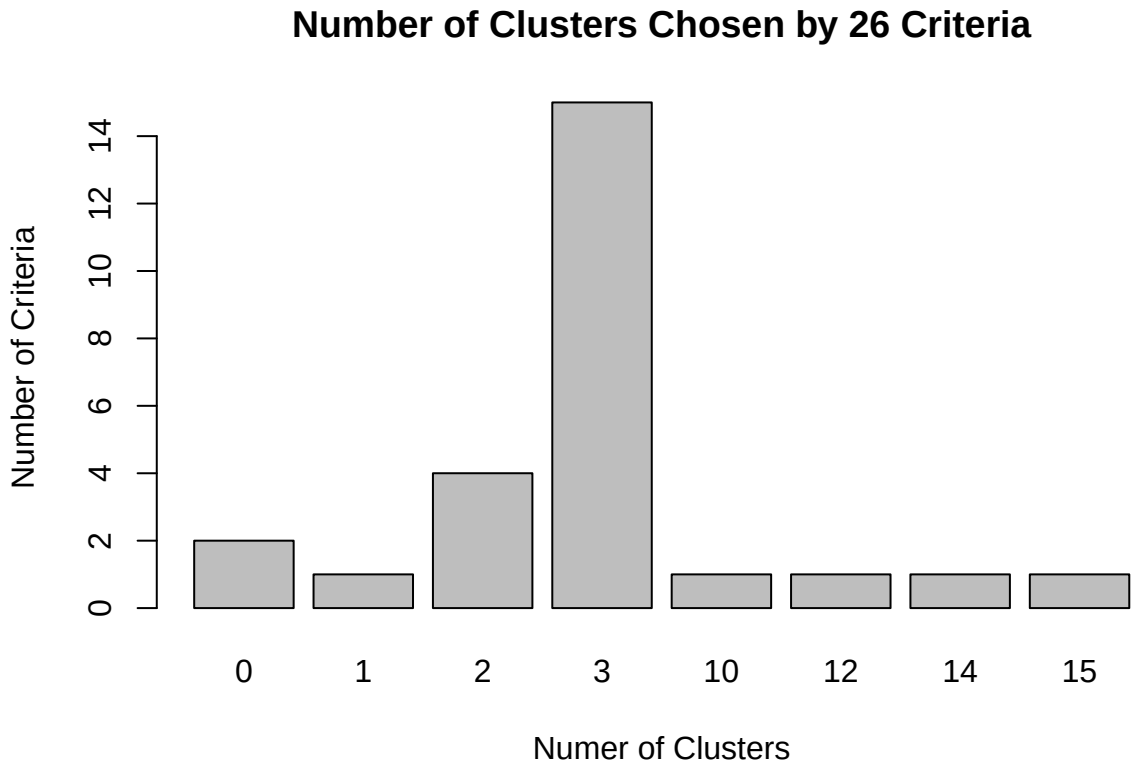
```
##   cluster  Alcohol    Malic      Ash Alcalinity Magnesium  Phenols
## 1       1 13.67677 1.997903 2.466290   17.46290 107.96774 2.847581
## 2       2 12.25092 1.897385 2.231231   20.06308  92.73846 2.247692
## 3       3 13.13412 3.307255 2.417647   21.24118  98.66667 1.683922
##   Flavanoids Nonflavanoids Proanthocyanins    Color       Hue Dilution
## 1  3.0032258     0.2920968        1.922097 5.453548 1.0654839 3.163387
## 2  2.0500000     0.3576923        1.624154 2.973077 1.0627077 2.803385
## 3  0.8188235     0.4519608        1.145882 7.234706 0.6919608 1.696667
##     Proline
## 1 1100.2258
## 2  510.1692
## 3  619.0588
```

```
#1 standardize data
#2 determine number of clusters
#3 K-means cluster analysis
#
# So how well did the K-means clustering uncover the actual structure of the
# data contained in the Type variable? A cross-tabulation of Type (wine variety)
# and cluster membership is given by
ct.km <- table(wine$Type, fit.km$cluster)
ct.km
```

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

```
#
# We can quantify the agreement between type and cluster, using an adjusted Rank
# index provided by the flexclust package.
library(flexclust)
```

```
## Loading required package: grid
## Loading required package: lattice
## Loading required package: modeltools
## Loading required package: stats4
```




```
randIndex(ct.km)
```

```
##      ARI
## 0.897495
```

```
#
# Here, the adjusted Rand index provides a measure of the agreement between
# two partitions, adjusted for chance. It ranges from -1 (no agreement) to
# 1 (perfect agreement). Agreement between the wine variety type and the cluster
# solution is 0.9. Not bad-shall we have some wine?
#
```