A Handbook of Statistical Analyses Using **R** — 2nd Edition

Brian S. Everitt and Torsten Hothorn

Cluster Analysis: Classifying Romano-British Pottery and Exoplanets

18.1 Introduction

18.2 Cluster Analysis

18.3 Analysis Using **R**

18.3.1 Classifying Romano-British Pottery

We start our analysis with computing the dissimilarity matrix containing the Euclidean distance of the chemical measurements on all 45 pots. The resulting 45×45 matrix can be inspected by an *image plot*, here obtained from function `levelplot` available in package **lattice** (Sarkar, 2009, 2008). Such a plot associates each cell of the dissimilarity matrix with a color or a grey value. We choose a very dark grey for cells with distance zero (i.e., the diagonal elements of the dissimilarity matrix) and pale values for cells with greater Euclidean distance. Figure~18.1 leads to the impression that there are at least three distinct groups with small inter-cluster differences (the dark rectangles) whereas much larger distances can be observed for all other cells.

We now construct three series of partitions using single, complete, and average linkage hierarchical clustering as introduced in subsections~**??** and **??**. The function `hclust` performs all three procedures based on the dissimilarity matrix of the data; its `method` argument is used to specify how the distance between two clusters is assessed. The corresponding `plot` method draws a dendrogram; the code and results are given in Figure~18.2. Again, all three dendrograms lead to the impression that three clusters fit the data best (although this judgement is very informal).

From the `pottery_average` object representing the average linkage hierarchical clustering, we derive the three-cluster solution by cutting the dendrogram at a height of four (which, based on the right display in Figure~18.2 leads to a partition of the data into three groups). Our interest is now a comparison with the kiln sites at which the pottery was found.

```
R> pottery_cluster <- cutree(pottery_average, h = 4)
R> xtabs(~ pottery_cluster + kiln, data = pottery)

                kiln
pottery_cluster  1  2  3  4  5
```

```
R> pottery_dist <- dist(pottery[, colnames(pottery) != "kiln"])
R> library("lattice")
R> levelplot(as.matrix(pottery_dist), xlab = "Pot Number",
+            ylab = "Pot Number")
```
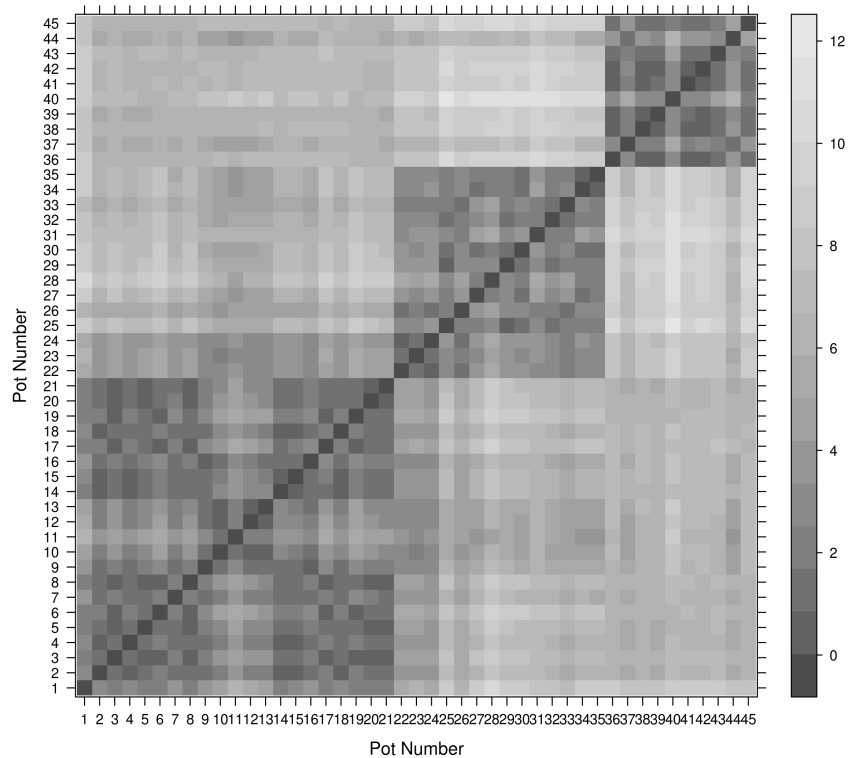

Figure 18.1   Image plot of the dissimilarity matrix of the `pottery` data.

```
    1 21   0   0   0   0
    2  0  12   2   0   0
    3  0   0   0   5   5
```

The contingency table shows that cluster 1 contains all pots found at kiln site number one, cluster 2 contains all pots from kiln sites number two and three, and cluster three collects the ten pots from kiln sites four and five. In fact, the five kiln sites are from three different regions defined by one, two and three, and four and five, so the clusters actually correspond to pots from three different regions.

```
R> pottery_single <- hclust(pottery_dist, method = "single")
R> pottery_complete <- hclust(pottery_dist, method = "complete")
R> pottery_average <- hclust(pottery_dist, method = "average")
R> layout(matrix(1:3, ncol = 3))
R> plot(pottery_single, main = "Single Linkage",
+       sub = "", xlab = "")
R> plot(pottery_complete, main = "Complete Linkage",
+       sub = "", xlab = "")
R> plot(pottery_average, main = "Average Linkage",
+       sub = "", xlab = "")
```
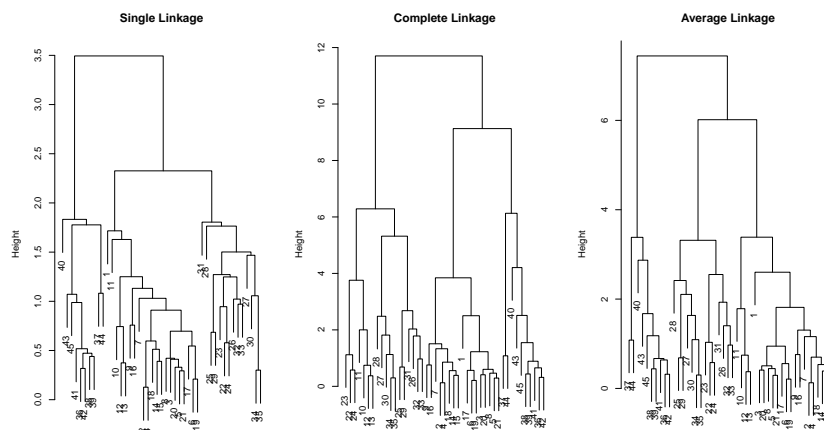

Figure 18.2   Hierarchical clustering of `pottery` data and resulting dendrograms.

18.3.2 Classifying Exoplanets

Sadly Figure~18.4 gives no completely convincing verdict on the number of groups we should consider, but using a little imagination 'little elbows' can be spotted at the three and five group solutions. We can find the number of planets in each group using

```
R> planet_kmeans3 <- kmeans(planet.dat, centers = 3)
R> table(planet_kmeans3$cluster)
```

```
 1   2   3
28  10  63
```

The centres of the clusters for the untransformed data can be computed using a small convenience function

```
R> ccent <- function(cl) {
+      f <- function(i) colMeans(planets[cl == i,])
+      x <- sapply(sort(unique(cl)), f)
```

```
R> data("planets", package = "HSAUR2")
R> library("scatterplot3d")
R> scatterplot3d(log(planets$mass), log(planets$period),
+       log(planets$eccen + ifelse(planets$eccen == 0, 0.001, 0)),
+       type = "h", angle = 55, pch = 16,
+       y.ticklabs = seq(0, 10, by = 2),
+       y.margin.add = 0.1, scale.y = 0.7,
+       xlab = "log(mass)", ylab = "log(period)",
+       zlab = "log(eccen)")
```
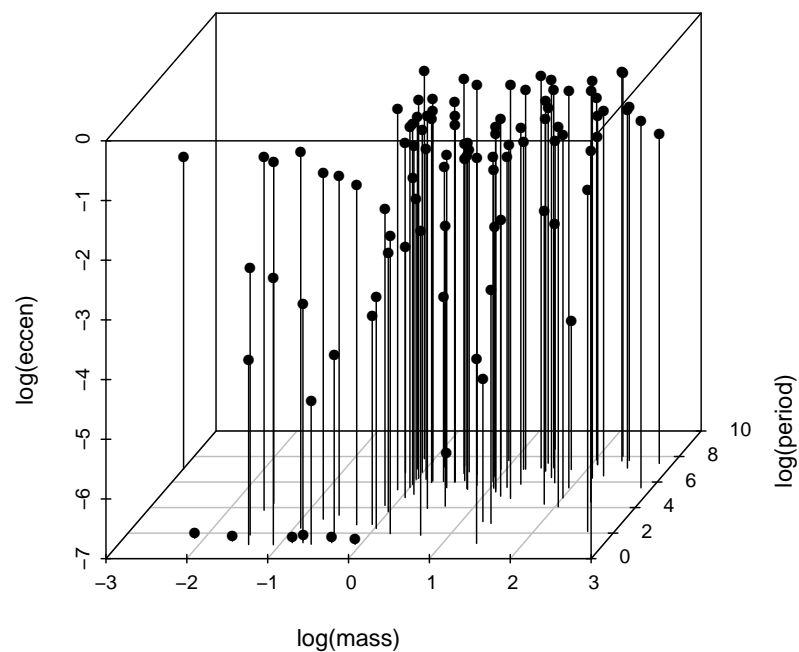

Figure 18.3   3D scatterplot of the logarithms of the three variables available for each of the exoplanets.

```
R> rge <- apply(planets, 2, max) - apply(planets, 2, min)
R> planet.dat <- sweep(planets, 2, rge, FUN = "/")
R> n <- nrow(planet.dat)
R> wss <- rep(0, 10)
R> wss[1] <- (n - 1) * sum(apply(planet.dat, 2, var))
R> for (i in 2:10)
+       wss[i] <- sum(kmeans(planet.dat,
+                            centers = i)$withinss)
R> plot(1:10, wss, type = "b", xlab = "Number of groups",
+       ylab = "Within groups sum of squares")
```
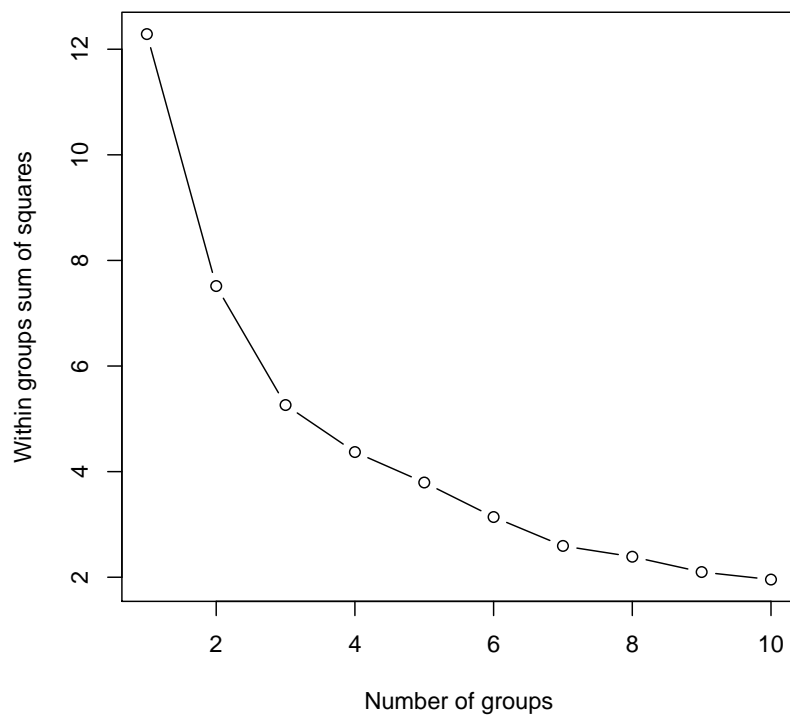

Figure 18.4   Within-cluster sum of squares for different numbers of clusters for the exoplanet data.

```
+        colnames(x) <- sort(unique(cl))
+        return(x)
+   }
```

which, applied to the three-cluster solution obtained by k-means gets

```
R> ccent(planet_kmeans3$cluster)
```

```
             1        2        3
mass      7.053    3.436    1.654
period  839.164 2420.550  311.390
eccen     0.518    0.272    0.178
```

for the three-cluster solution and, for the five cluster solution using

```
R> planet_kmeans5 <- kmeans(planet.dat, centers = 5)
R> table(planet_kmeans5$cluster)
```

```
 1  2  3  4  5
28  5  7 49 12
```

```
R> ccent(planet_kmeans5$cluster)
```

```
             1        2        3        4        5
mass      2.262   14.348    2.186    1.685    8.595
period  580.683  659.398 2557.643  282.269 1335.740
eccen     0.491    0.327    0.199    0.122    0.473
```

18.3.3 Model-based Clustering in R

We now proceed to apply model-based clustering to the planets data. R functions for model-based clustering are available in package **mclust** (Fraley et~al., 2009, Fraley and Raftery, 2002). Here we use the Mclust function since this selects both the most appropriate model for the data *and* the optimal number of groups based on the values of the BIC computed over several models and a range of values for number of groups. The necessary code is:

```
R> library("mclust")
R> planet_mclust <- Mclust(planet.dat)
```

and we first examine a plot of BIC values using the R code that is displayed on top of Figure~18.5. In this diagram the different plotting symbols refer to different model assumptions about the shape of clusters:

  EII: spherical, equal volume,

  VII: spherical, unequal volume,

  EEI: diagonal, equal volume and shape,

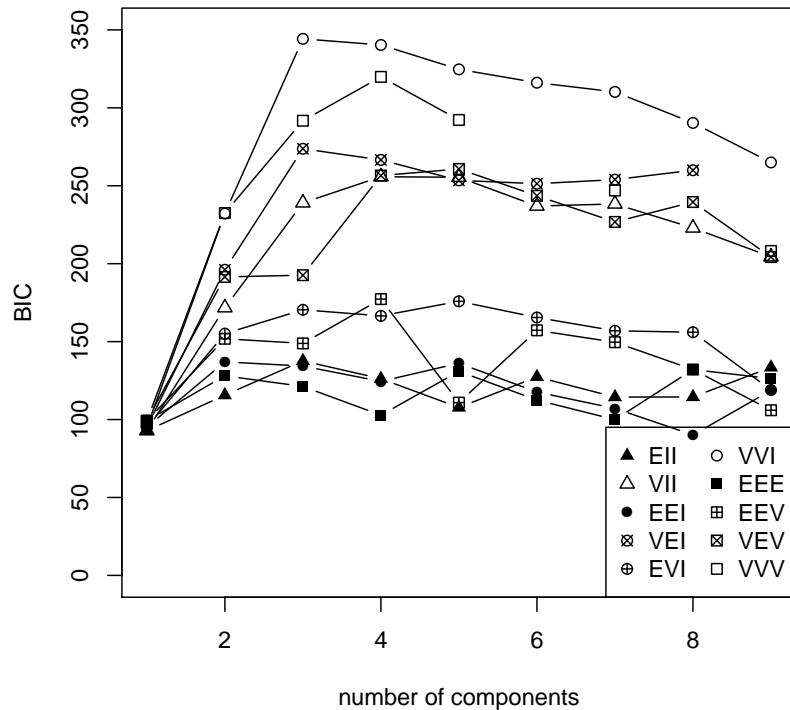 VEI: diagonal, varying volume, equal shape,

  EVI: diagonal, equal volume, varying shape,

  VVI: diagonal, varying volume and shape,

  EEE: ellipsoidal, equal volume, shape, and orientation,

  EEV: ellipsoidal, equal volume and equal shape,

```
R> plot(planet_mclust, planet.dat, what = "BIC", col = "black",
+       ylab = "-BIC", ylim = c(0, 350))
```


Figure 18.5 Plot of BIC values for a variety of models and a range of number of clusters.

VEV: ellipsoidal, equal shape,

VVV: ellipsoidal, varying volume, shape, and orientation

The BIC selects model VVI (diagonal varying volume and varying shape) with three clusters as the best solution as can be seen from the `print` output:

```
R> print(planet_mclust)
```

best model: diagonal, varying volume and shape with 3 components

This solution can be shown graphically as a scatterplot matrix. The plot is shown in Figure~18.6. Figure~18.7 depicts the clustering solution in the three-dimensional space.

The number of planets in each cluster and the mean vectors of the three clusters for the untransformed data can now be inspected by using

```
R> clPairs(planet.dat,
+       classification = planet_mclust$classification,
+       symbols = 1:3, col = "black")
```
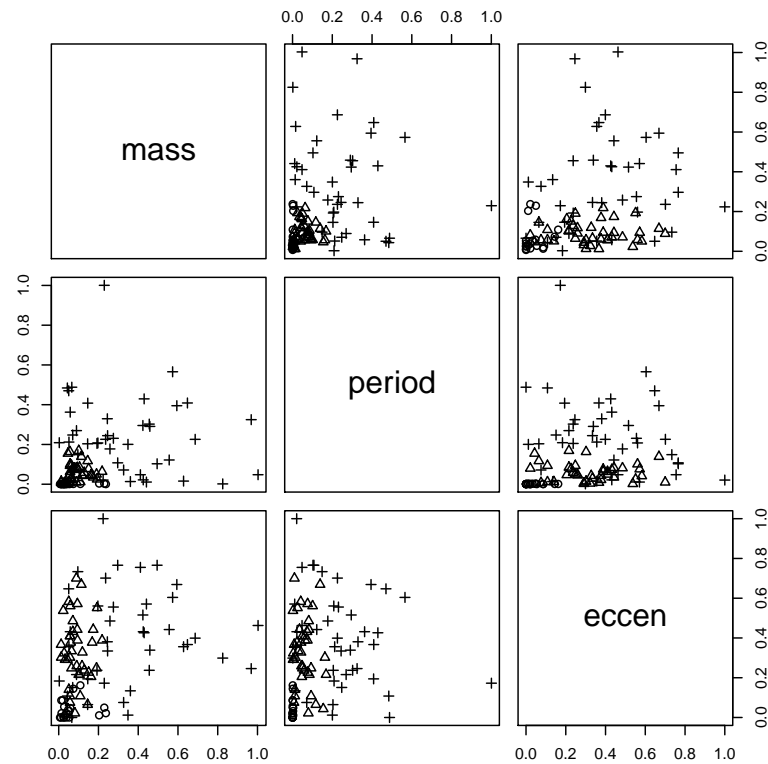

Figure 18.6   Scatterplot matrix of planets data showing a three-cluster solution
from `Mclust`.

```
R> table(planet_mclust$classification)

 1  2  3
19 41 41

R> ccent(planet_mclust$classification)

            1        2        3
mass   1.1665    1.580     6.08
period 6.4718  313.413  1325.53
eccen  0.0365    0.306     0.37
```

Cluster 1 consists of planets about the same size as Jupiter with very short
periods and eccentricities (similar to the first cluster of the k-means solution).

```
R> scatterplot3d(log(planets$mass), log(planets$period),
+       log(planets$eccen + ifelse(planets$eccen == 0, 0.001, 0)),
+       type = "h", angle = 55,  scale.y = 0.7,
+       pch = planet_mclust$classification,
+       y.ticklabs = seq(0, 10, by = 2), y.margin.add = 0.1,
+       xlab = "log(mass)", ylab = "log(period)",
+       zlab = "log(eccen)")
```
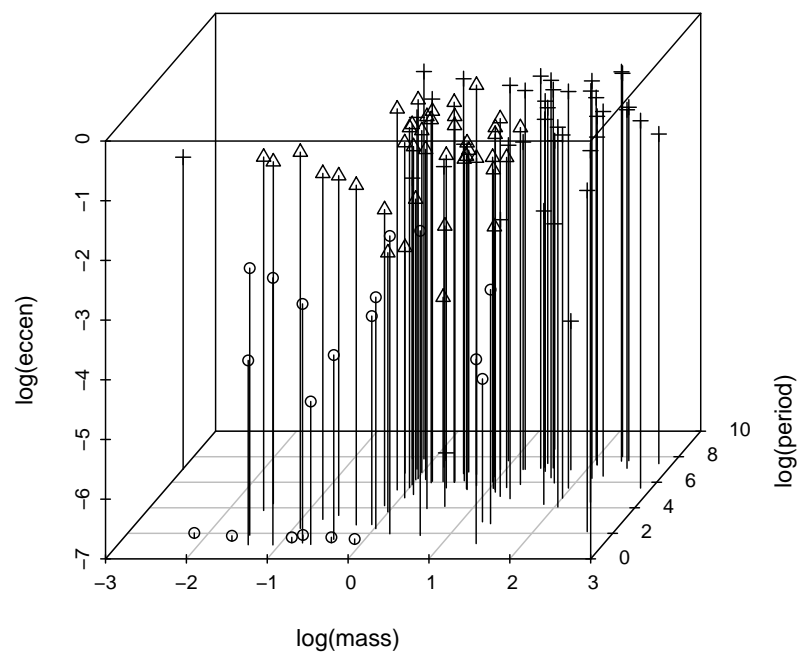

Figure 18.7   3D scatterplot of planets data showing a three-cluster solution from
                  `Mclust`.

Cluster 2 consists of slightly larger planets with moderate periods and large eccentricities, and cluster 3 contains the very large planets with very large periods. These two clusters do not match those found by the k-means approach.

Bibliography

Fraley, C. and Raftery, A.~E. (2002), "Model-based clustering, discriminant analysis, and density estimation," *Journal of the American Statistical Association*, 97, 611–631.

Fraley, C., Raftery, A.~E., and Wehrens, R. (2009), **mclust**: *Model-based Cluster Analysis*, URL `http://www.stat.washington.edu/mclust`, R package version 3.1-10.3.

Sarkar, D. (2008), *Lattice: Multivariate Data Visualization with R*, New York, USA: Springer-Verlag.

Sarkar, D. (2009), **lattice**: *Lattice Graphics*, URL `http://CRAN.R-project.org/package=lattice`, R package version 0.17-22.