Метод k-средних

Примеры. Многомерное шкалирование

Сегментация потребителей безалкогольных напитков

```r
#  Читаем данные примера
beverage.01 <- read.table("week_03/data/beverage.csv",
                          header=T, sep=";")
# Вспомним имена переменных
names(beverage.01)
```

```
## [1] "numb.obs" "COKE"     "D_COKE"   "D_PEPSI"  "D_7UP"    "P
## [7] "SPRITE"   "TAB"      "SEVENUP"
```

```r
# Убираем ненужный 1-й столбец
beverage.01[,1] <- NULL
# Удачно угадаем, что кластеров три
summ.3 = kmeans(beverage.01, 3, iter.max = 100)
# Вспомним поля списка
names(summ.3)
```

```
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

```
# К каким кластерам принадлежат объекты?
summ.3$cluster
```

```
##  [1] 1 3 3 2 3 3 2 1 1 1 3 2 1 3 2 3 2 3 1 3 3 3 2 3 2 2 2 1
```

Важная опция `nstart` — число стартовых попыток задать центры кластеров. По результатам процедура выберет наилучшее с точки зрение `tot.withinss` решение.

```
summ.31 = kmeans(beverage.01, 3, iter.max = 100,
                 nstart = 10)
```

```
summ.31$cluster
```

```
##  [1] 3 1 1 2 1 3 2 3 3 1 3 2 3 3 2 3 3 1 1 1 1 1 2 1 2 2 2 3
```

У вас результат может отличаться: разное число стартовых попыток, другие номера кластеров.

Координаты центров кластеров — основной источник вдохновения для интерпретации

```
options(digits=2) # иначе результат неудобно читать.
# summ.3$centers    # координаты центров кластеров
# Транспонируем
t(summ.3$centers)
```

```
##              1     2     3
## COKE      0.89 0.000 0.923
## D_COKE    0.22 1.000 0.231
## D_PEPSI   0.11 0.500 0.077
## D_7UP     0.11 0.500 0.000
## PEPSI     0.56 0.000 0.846
## SPRITE    0.56 0.083 0.385
## TAB       0.11 0.833 0.000
## SEVENUP   1.00 0.000 0.000
```

Другие результаты кластеризации

```
# Сумма квадратов расстояний от объектов кластера до центра кластера
summ.3$withinss
```

```
## [1] 9.555556 8.583333 8.923077
```

```
# Сумма элементов предыдущего вектора
summ.3$tot.withinss
```

```
## [1] 27.06197
```

```
# Полная сумма квадратов: sum(33*(apply(beverage.01, 2, sd))^2)
summ.3$totss
```

```
## [1] 58.38235
```

```
# Межкластерная сумма квадратов: betweenss = totss - tot.withinss
summ.3$betweenss
```

```
## [1] 31.32039
```

```
# Размеры кластеров
summ.3$size
```

```
## [1]  9 12 13
```

Попробуем определить “правильное” число кластеров

```
# Максимальное число кластеров
n.clust <- 15
# Вектор для хранения результатов
wcss <- vector(mode="numeric", length=n.clust)
# Запускаем kmeans для k от 1 до 15
for(i in 1:n.clust){
  wcss[i] <- kmeans(beverage.01, centers=i)$tot.withinss
}
plot(1:n.clust, wcss, type="b", xlab="Number of Clusters",
     ylab="Within clusters sum of squares")
```

Scree plot

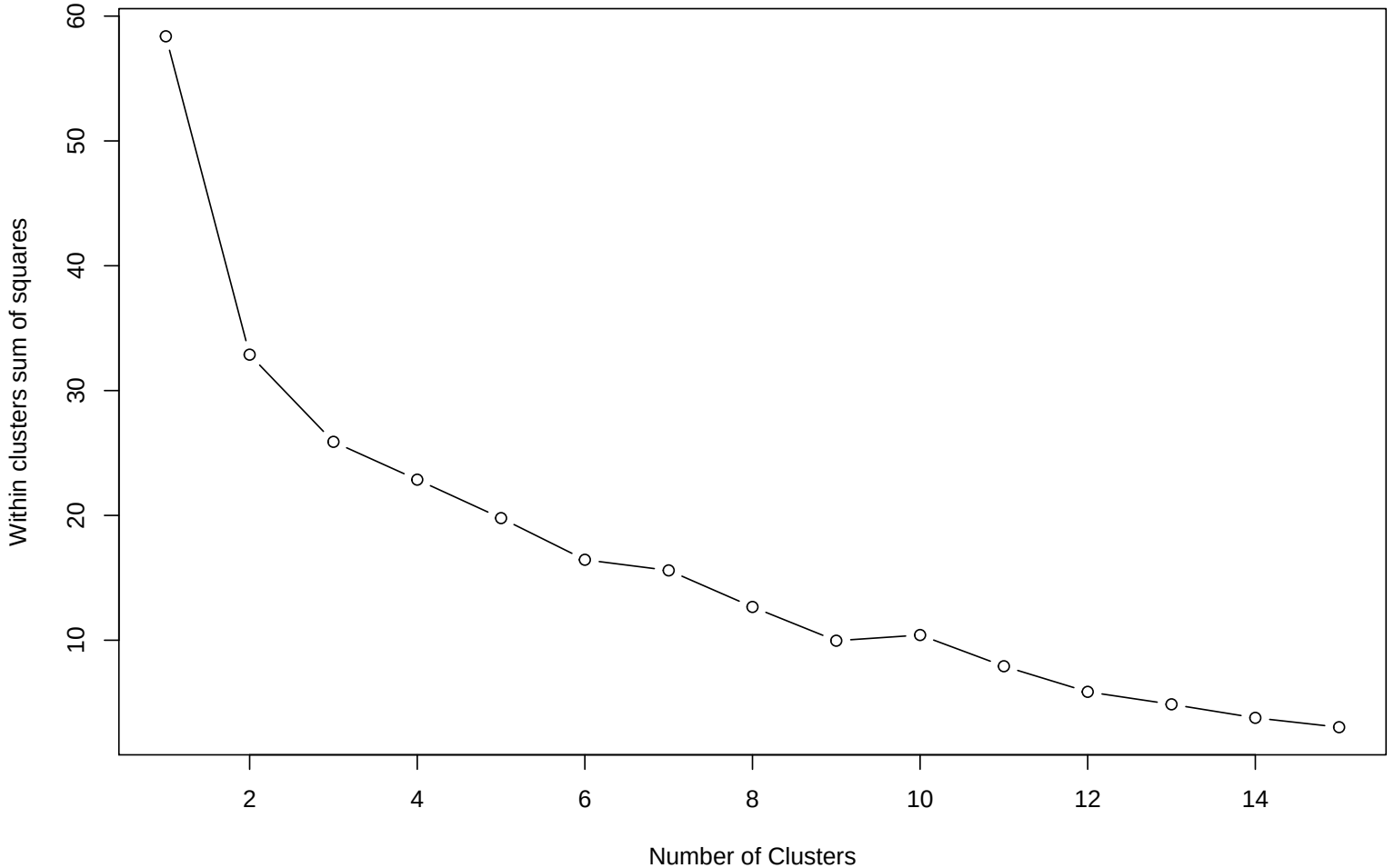

Within clusters sum of squares
10
20
30
40
50
60
2
4
6
8
10
12
14
Number of Clusters

Таблица сопряженности (contingency table)

Реализуется функцией `table`.

```
#  Попробуем решение с 4 кластерами и сравним результаты
summ.4 = kmeans(beverage.01, 4, iter.max = 100)
summ.3$size
```

```
## [1]  9 12 13
```

```
summ.4$size
```

```
## [1] 11  5 11  7
```

```
#  Для сравнения использовать команду
table(summ.3$cluster, summ.4$cluster)
```

```
##
##      1  2  3  4
##   1  3  2  0  4
##   2  0  0 11  1
##   3  8  3  0  2
```

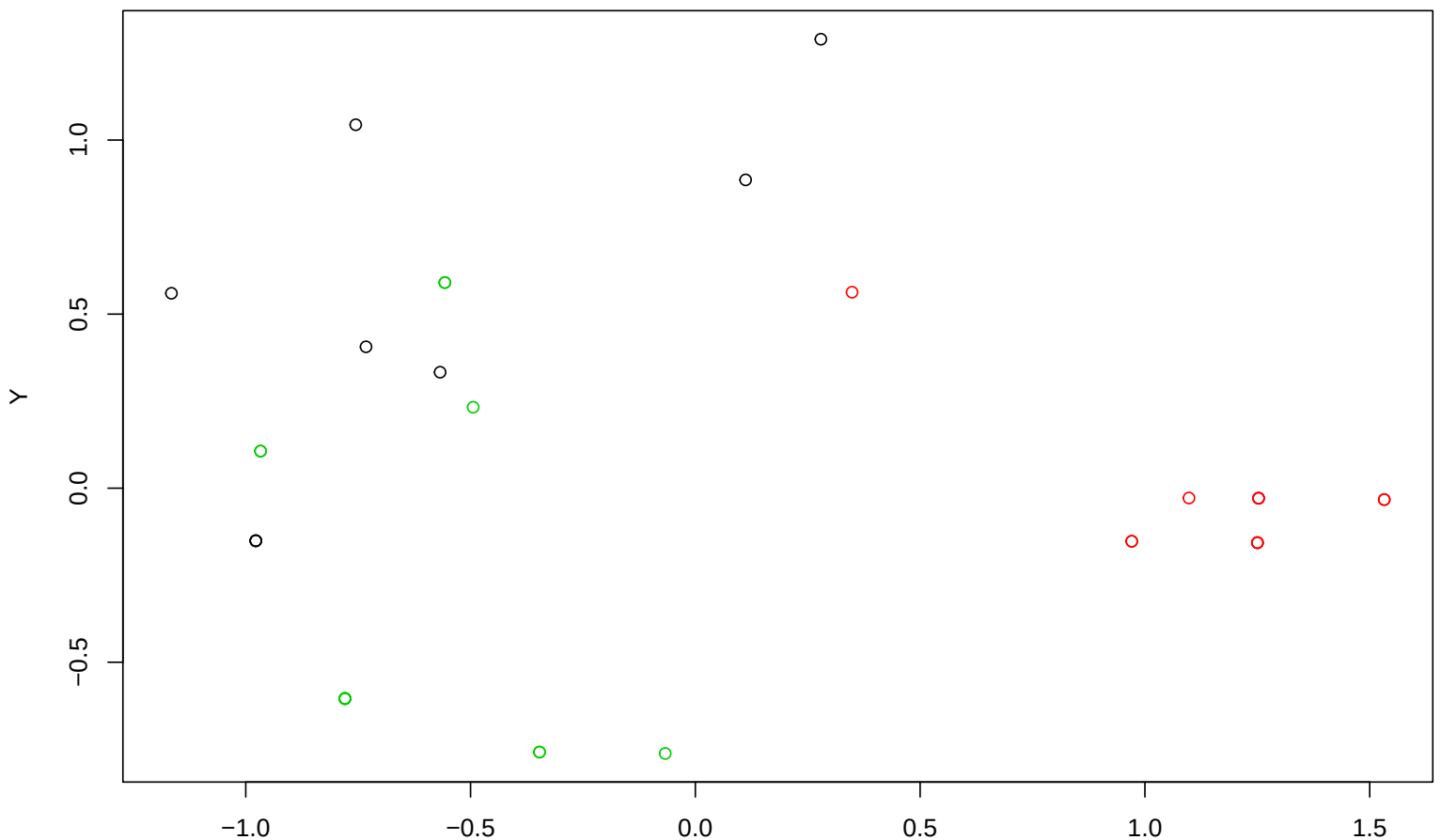
Сравним две кластеризации: hclust и kmeans

```
dist.beverage <- dist(beverage.01)
clust.beverage <- hclust(dist.beverage, "ward.D")
groups <- cutree(clust.beverage, k=3)
table(summ.3$cluster, groups)
```

```
##    groups
##       1  2  3
##    1  6  3  0
##    2  1  0 11
##    3  5  8  0
```

Многомерное шкалирование в деле (cmdscale)

```
# Используем матрицу попарных расстояний
beverage.mds <- cmdscale(dist.beverage)
# и результаты k-means кластеризации
plot(beverage.mds, col=summ.3$cluster, xlab="Index", ylab="Y")
```



Многомерное шкалирование (Multidimentional scaling)

Визуализацию многомерного пространства выполнить трудно. Есть много способов, но они значительно уступают в выразительности двух- и трехмерным графикам.

Идея: давайте спроецируем данные на плоскость (или двумерное подпространство).

Проблема: как выбрать плоскость проекции?

Мы знаем расстояния между точками в пространстве и пытаемся расположить точки на плоскости так, чтобы расстояния ними были такими же как в исходном многомерном пространстве. Точнее, мы будем располагать точки на плоскости так, чтобы отклонения расстояний между ними от "правильных" расстояний в многомерном пространстве было минимальными.

Что это дает. Удовлетворительный результат кластерного анализа, отображенный через многомерное шкалирование, успокаивает, а неудовлетворительный дает повод призадуматься о качестве кластеризации в вашем исследовании.

Потребление белков в Европе

Читаем, стандартизируем

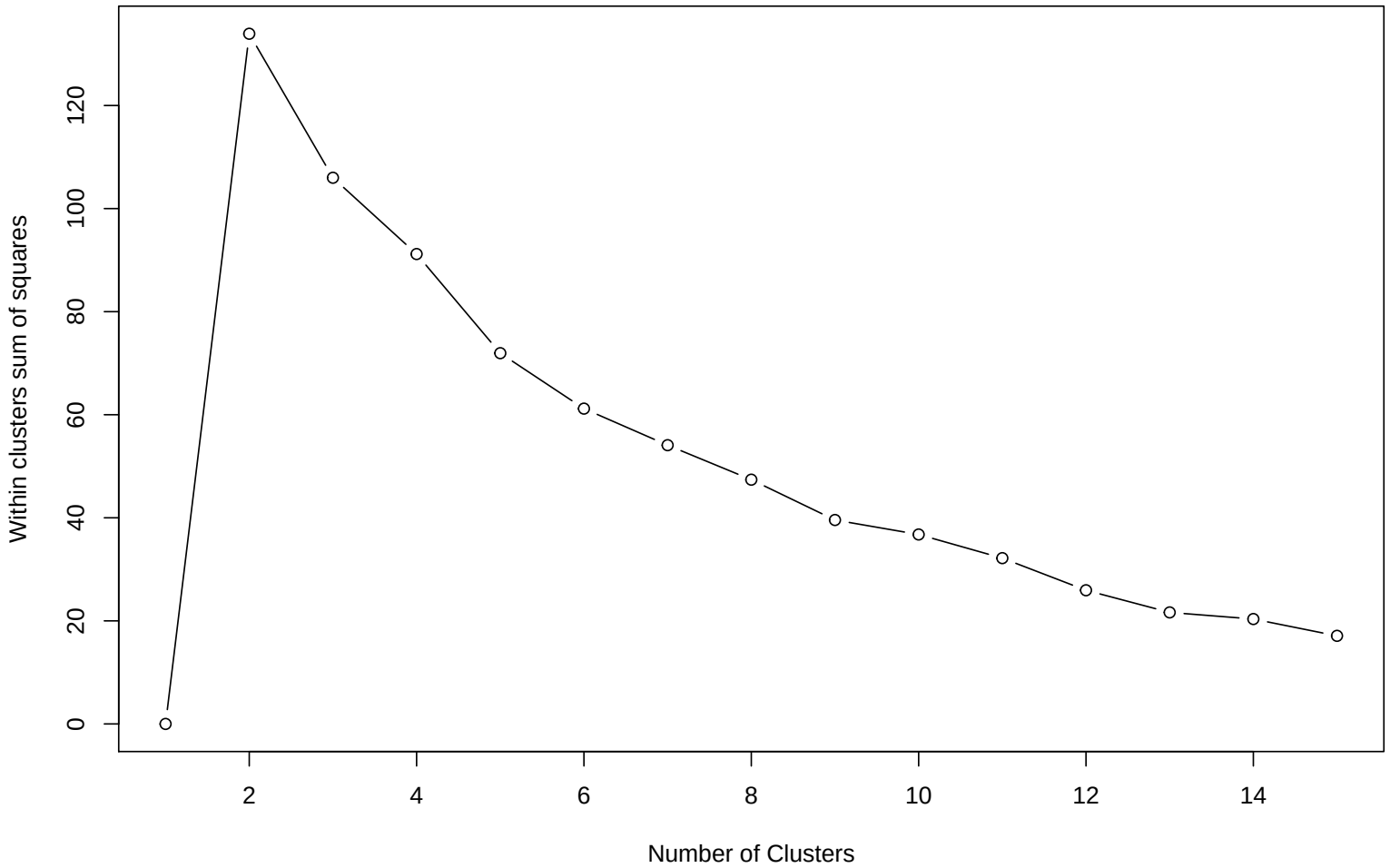
```
## Шаг 1. Чтение данных
setwd("week_04/data")
data.01 <- read.table("Protein Consumption in Europe.csv", header=T,
                      sep=";", dec = ",", row.names = 1)
# Проверим структуру таблицы: не получились ли строки вместо цифр?
str(data.01)
```

```
## 'data.frame':    25 obs. of  9 variables:
##  $ RedMeat  : num  10.1 8.9 13.5 7.8 9.7 10.6 8.4 9.5 18 10.2 ...
##  $ WhiteMeat: num  1.4 14 9.3 6 11.4 10.8 11.6 4.9 9.9 3 ...
##  $ Eggs     : num  0.5 4.3 4.1 1.6 2.8 3.7 3.7 2.7 3.3 2.8 ...
##  $ Milk     : num  8.9 19.9 17.5 8.3 12.5 25 11.1 33.7 19.5 17.6 ...
##  $ Fish     : num  0.2 2.1 4.5 1.2 2 9.9 5.4 5.8 5.7 5.9 ...
##  $ Cereals  : num  42.3 28 26.6 56.7 34.3 21.9 24.6 26.3 28.1 41.7 ...
##  $ Starch   : num  0.6 3.6 5.7 1.1 5 4.8 6.5 5.1 4.8 2.2 ...
##  $ Nuts     : num  5.5 1.3 2.1 3.7 1.1 0.7 0.8 1 2.4 7.8 ...
##  $ Fr.Veg   : num  1.7 4.3 4 4.2 4 2.4 3.6 1.4 6.5 6.5 ...
```

```
## Шаг 2. Удаление пропущенных значений
summary(data.01)
# В данной задаче пропущенных значений нет.
```

```
## Шаг 3. Стандартизация переменных.
# к среднему 0 и ст. отклонению 1
data.02 <- scale(data.01, center = TRUE, scale = TRUE)
```

Выбираем число кластеров k



Предположительно кластеров k=5

Процедура кластерного анализа

```
# Проводим кластерный анализ с выбранным k
clust = kmeans(data.02, 5, iter.max = 100, nstart = 10)
```

Интерпретируем результаты

```
t(clust$centers)
```

```
##                  1      2     3      4     5
## RedMeat     0.0066 -0.570 -0.81  1.011 -0.51
## WhiteMeat -0.2290  0.580 -0.87  0.742 -1.11
## Eggs        0.1915 -0.086 -1.55  0.941 -0.41
## Milk        1.3459 -0.460 -1.08  0.570 -0.83
## Fish        1.1583 -0.454 -1.04 -0.267  0.98
## Cereals   -0.8723  0.318  1.72 -0.688  0.13
## Starch      0.1677  0.786 -1.42  0.229 -0.18
## Nuts      -0.9553 -0.268  1.00 -0.508  1.31
## Fr.Veg    -1.1148  0.069 -0.64  0.022  1.63
```

Само по себе это мало что дает. Нужна дополнительная информация: какие страны попали в какие кластеры? Будем делать так:

```
# Какие страны попали в 1-й кластер?
row.names(data.01[clust$cluster==1,])
# Их диета
colMeans(data.01[clust$cluster==1,])
```

Страны и диеты

```
## [1] "Denmark" "Finland" "Norway"  "Sweden"

##   RedMeat WhiteMeat      Eggs      Milk      Fish   Cereals    Starch
##       9.8       7.1       3.2      26.7       8.2      22.7       4.5
##      Nuts    Fr.Veg
##       1.2       2.1

## [1] "Czechoslovakia" "E_Germany"      "Hungary"        "Poland"
## [5] "USSR"

##   RedMeat WhiteMeat      Eggs      Milk      Fish   Cereals    Starch
##       7.9      10.0       2.8      13.8       2.7      35.7       5.6
##      Nuts    Fr.Veg
##       2.5       4.3

## [1] "Albania"    "Bulgaria"   "Romania"    "Yugoslavia"

##   RedMeat WhiteMeat      Eggs      Milk      Fish   Cereals    Starch
##      7.12      4.67      1.20      9.45      0.75     51.12      1.95
##      Nuts    Fr.Veg
##      5.05      2.98
```

Страны и диеты 2

```
## [1] "Austria"     "Belgium"     "France"      "Ireland"     "Netherlands"
## [6] "Switzerland" "UK"          "W_Germany"
```

```
##   RedMeat WhiteMeat      Eggs      Milk      Fish   Cereals    Starch
##      13.2      10.6       4.0      21.2       3.4      24.7       4.6
##      Nuts    Fr.Veg
##       2.1       4.2
```

```
## [1] "Greece"   "Italy"    "Portugal" "Spain"
```

```
##   RedMeat WhiteMeat      Eggs      Milk      Fish   Cereals    Starch
##       8.1       3.8       2.5      11.2       7.6      33.7       4.0
##      Nuts    Fr.Veg
##       5.7       7.1
```

Многомерное шкалирование

```
data.dist <- dist(data.02)
mds <- cmdscale(data.dist)
plot(mds, col = clust$cluster, xlab = "Index", ylab = "Y")
# Рисуем метки-названия стран
text(mds, labels = rownames(data.02), pos = 1, cex = .7)
```