Clustering

Frederick Jones

2024-04-16

import data

grocery

```
library(factoextra)
```

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(readr)
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
grocery <- read.transactions("~/Downloads/GroceryDataSet.csv")
```

```
## Warning in asMethod(object): removing duplicated items in transactions
```

```
head(grocery)
```

```
## transactions in sparse format with
##  6 transactions (rows) and
##  8219 items (columns)
```

```
summary(grocery)
```

```
## transactions as itemMatrix in sparse format with
##  9835 rows (elements/itemsets/transactions) and
##  8219 columns (items) and a density of 0.0004422899
##
## most frequent items:
## vegetables,whole            whole         tropical            other
##              940              717              482              460
##           citrus          (Other)
##              453            32700
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 1380 2733 1774 1257  910  601  415  293  166   95   75   44   39   19   11    9
##   17   18   19   20   21   23
##    2    3    3    3    1    2
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.635   5.000  23.000
##
## includes extended item information - examples:
##                       labels
## 1    ,,,,,,,,,,,,,,,,,,,,,,,
## 2   ,,,,,,,,,,,,,,,,,,,,,,,,
## 3 ,,,,,,,,,,,,,,,,,,,,,,,,,,
```

eploring the data

```
library(tibble)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tidyr     1.3.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x tidyr::pack()   masks Matrix::pack()
## x dplyr::recode() masks arules::recode()
## x tidyr::unpack() masks Matrix::unpack()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(arules)


grocery_frequency <- tibble(Items = names(itemFrequency(grocery)),
                            Frequency = itemFrequency(grocery))
head(grocery_frequency)
```

```
## # A tibble: 6 x 2
##   Items                          Frequency
##   <chr>                              <dbl>
## 1 ,,,,,,,,,,,,,,,,,,,,,,,         0.000102
## 2 ,,,,,,,,,,,,,,,,,,,,,,,,        0.000203
## 3 ,,,,,,,,,,,,,,,,,,,,,,,,,,      0.000305
## 4 ,,,,,,,,,,,,,,,,,,,,,,,,,,,     0.000102
## 5 ,,,,,,,,,,,,,,,,,,,,,,,,,,,,    0.000712
## 6 ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,   0.000712
```

The table below represents the support or actual frequency with which each item is purchased. For example, whole milk is purchased with actual frequency of 0.255 or 26% of the time. Stated differently, about 1 in 4 transactions (0.255 of the time) include whole milk.

```
grocery_frequency %>%
  arrange(desc(Frequency)) %>%
  slice(1:20)
```

```
## # A tibble: 20 x 2
##    Items                                   Frequency
##    <chr>                                       <dbl>
##  1 vegetables,whole                           0.0956
##  2 whole                                      0.0729
##  3 tropical                                   0.0490
##  4 other                                      0.0468
##  5 citrus                                     0.0461
##  6 cheese                                     0.0397
##  7 beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,        0.0386
##  8 vegetables,other                           0.0383
##  9 bakery                                     0.0374
## 10 life                                       0.0374
## 11 fruit,other                                0.0368
## 12 fruit,root                                 0.0356
## 13 bottled                                    0.0354
## 14 fruit,whole                                0.0346
## 15 canned                                     0.0328
## 16 root                                       0.0293
## 17 fruit,pip                                  0.0287
## 18 pip                                        0.0282
## 19 fruit,tropical                             0.0199
## 20 hamburger                                  0.0170
```

```
grocery_frequency %>%
  select(Frequency) %>%
  summary()
```

```
##    Frequency
##  Min.   :0.0001017
##  1st Qu.:0.0001017
##  Median :0.0001017
##  Mean   :0.0004423
##  3rd Qu.:0.0002034
##  Max.   :0.0955770
```

Extract the rules

```
frequency_per_day <- 5
days_per_period <- 30
total_transactions <- length(grocery)

support_value <- (frequency_per_day * days_per_period)/total_transactions

grocery_rules <- apriori(grocery,
                         parameter = list(
                           support = support_value,
                           confidence = 0.25,
                           minlen = 2
                         ))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime    support minlen
##        0.25    0.1    1 none FALSE            TRUE       5 0.01525165      2
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 150
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[8219 item(s), 9835 transaction(s)] done [0.04s].
## sorting and recoding items ... [21 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [6 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(grocery_rules)
```

```
## set of 6 rules
##
## rule length distribution (lhs + rhs):sizes
## 2
## 6
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2       2       2       2       2       2
##
## summary of quality measures:
##     support          confidence        coverage            lift
##  Min.   :0.01708   Min.   :0.4642   Min.   :0.03284   Min.   : 4.857
##  1st Qu.:0.01995   1st Qu.:0.5311   1st Qu.:0.03604   1st Qu.: 8.975
```

```
##  Median :0.02644   Median :0.7446   Median :0.03742   Median :20.833
##  Mean   :0.02710   Mean   :0.7389   Mean   :0.03671   Mean   :17.499
##  3rd Qu.:0.03467   3rd Qu.:0.9512   3rd Qu.:0.03810   3rd Qu.:25.253
##  Max.   :0.03742   Max.   :1.0000   Max.   :0.03864   Max.   :26.726
##      count
##  Min.   :168.0
##  1st Qu.:196.2
##  Median :260.0
##  Mean   :266.5
##  3rd Qu.:341.0
##  Max.   :368.0
##
## mining info:
##     data ntransactions    support confidence
##  grocery          9835 0.01525165       0.25
##                                                                                                call
##  apriori(data = grocery, parameter = list(support = support_value, confidence = 0.25, minlen = 2))
```

Assess the rules

```
grocery_rules %>%
  sort(by = "confidence") %>%
  head(n = 10) %>%
  inspect()
```

```
##     lhs
## [1] {bakery}                                =>
## [2] {life}                                  =>
## [3] {canned}                                =>
## [4] {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} =>
## [5] {fruit,root}                            =>
## [6] {vegetables,other}                      =>
##     rhs                                     support    confidence coverage
## [1] {life}                                  0.03741739 1.0000000  0.03741739
## [2] {bakery}                                0.03741739 1.0000000  0.03741739
## [3] {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} 0.02643620 0.8049536  0.03284189
## [4] {canned}                                0.02643620 0.6842105  0.03863752
## [5] {vegetables,whole}                      0.01708185 0.4800000  0.03558719
## [6] {vegetables,whole}                      0.01779359 0.4641910  0.03833249
##     lift      count
## [1] 26.725543 368
## [2] 26.725543 368
## [3] 20.833469 260
## [4] 20.833469 260
## [5]  5.022128 168
## [6]  4.856722 175
```

```
grocery_rules %>%
  sort (by = "lift") %>%
  head(n = 10) %>%
  inspect()
```

```
##     lhs
## [1] {bakery}                                 =>
## [2] {life}                                   =>
## [3] {canned}                                 =>
## [4] {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} =>
## [5] {fruit,root}                             =>
## [6] {vegetables,other}                       =>
##     rhs                                      support    confidence coverage
## [1] {life}                                   0.03741739 1.0000000  0.03741739
## [2] {bakery}                                 0.03741739 1.0000000  0.03741739
## [3] {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} 0.02643620 0.8049536  0.03284189
## [4] {canned}                                 0.02643620 0.6842105  0.03863752
## [5] {vegetables,whole}                       0.01708185 0.4800000  0.03558719
## [6] {vegetables,whole}                       0.01779359 0.4641910  0.03833249
##     lift      count
## [1] 26.725543 368
## [2] 26.725543 368
## [3] 20.833469 260
## [4] 20.833469 260
## [5]  5.022128 168
## [6]  4.856722 175
```

Clustering

```
grocery_rules_df <- as(grocery_rules, "data.frame")
head(grocery_rules_df)
```

```
##                                                rules    support confidence
## 1 {canned} => {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} 0.02643620  0.8049536
## 2 {beer,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,} => {canned} 0.02643620  0.6842105
## 3                {fruit,root} => {vegetables,whole} 0.01708185  0.4800000
## 4          {vegetables,other} => {vegetables,whole} 0.01779359  0.4641910
## 5                             {bakery} => {life} 0.03741739  1.0000000
## 6                             {life} => {bakery} 0.03741739  1.0000000
##     coverage      lift count
## 1 0.03284189 20.833469   260
## 2 0.03863752 20.833469   260
## 3 0.03558719  5.022128   168
## 4 0.03833249  4.856722   175
## 5 0.03741739 26.725543   368
## 6 0.03741739 26.725543   368
```

```
grocery_rules_df %>%
  select(lift, count) %>%
  summary()
```

```
##       lift            count
##  Min.   : 4.857   Min.   :168.0
##  1st Qu.: 8.975   1st Qu.:196.2
##  Median :20.833   Median :260.0
```

```
##  Mean   :17.499   Mean   :266.5
##  3rd Qu.:25.253   3rd Qu.:341.0
##  Max.   :26.726   Max.   :368.0
```

```
grocery_rules_df <- na.omit(grocery_rules_df)
```

```
grocery_rules_scaled <- grocery_rules_df %>%
  select(lift, count) %>%
  scale()

head(grocery_rules_scaled)
```

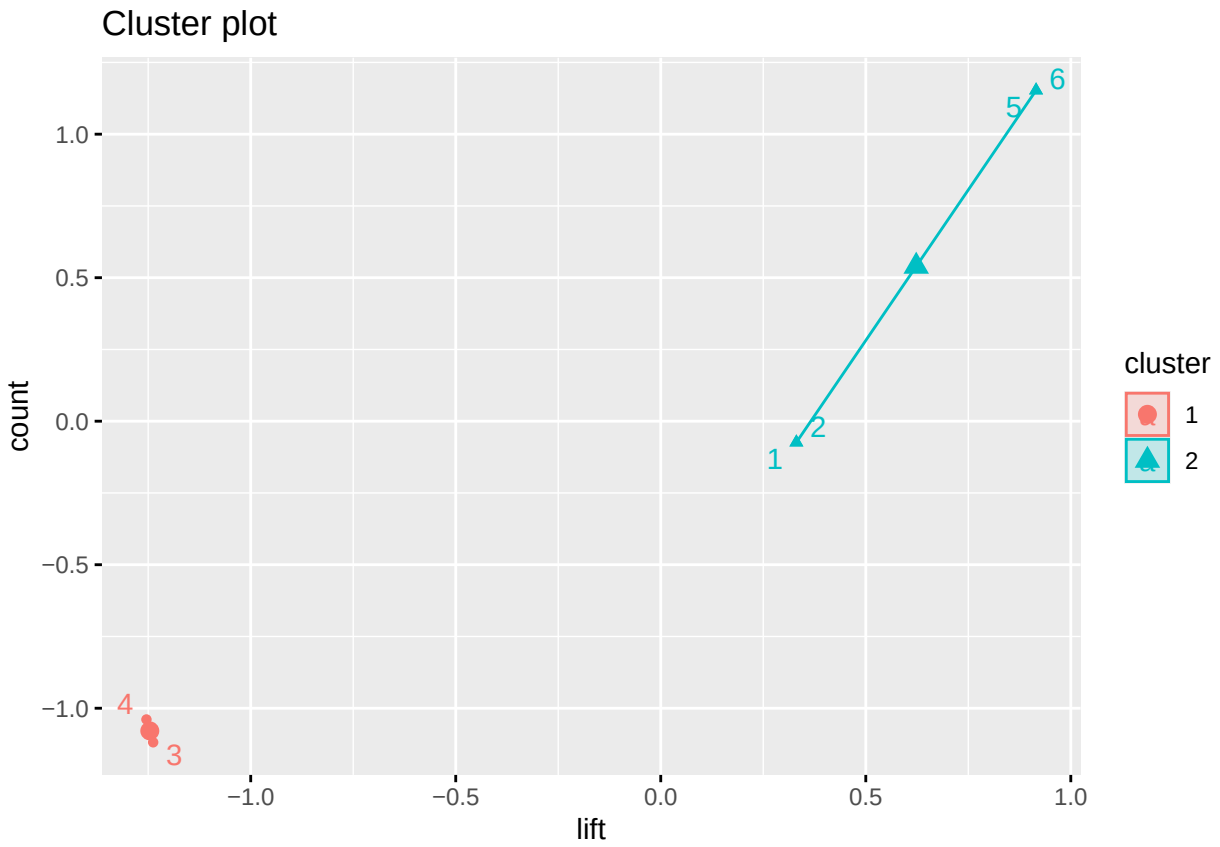
```
##         lift       count
## 1  0.3307650 -0.07382218
## 2  0.3307650 -0.07382218
## 3 -1.2378774 -1.11868996
## 4 -1.2542873 -1.03918915
## 5  0.9153174  1.15276174
## 6  0.9153174  1.15276174
```

```
set.seed(101)

k_2 <- kmeans(grocery_rules_scaled, centers = 2, nstart = 5)
k_2$size
```

```
## [1] 2 4
```

```
fviz_cluster(k_2, data = grocery_rules_scaled, repel = TRUE)
```

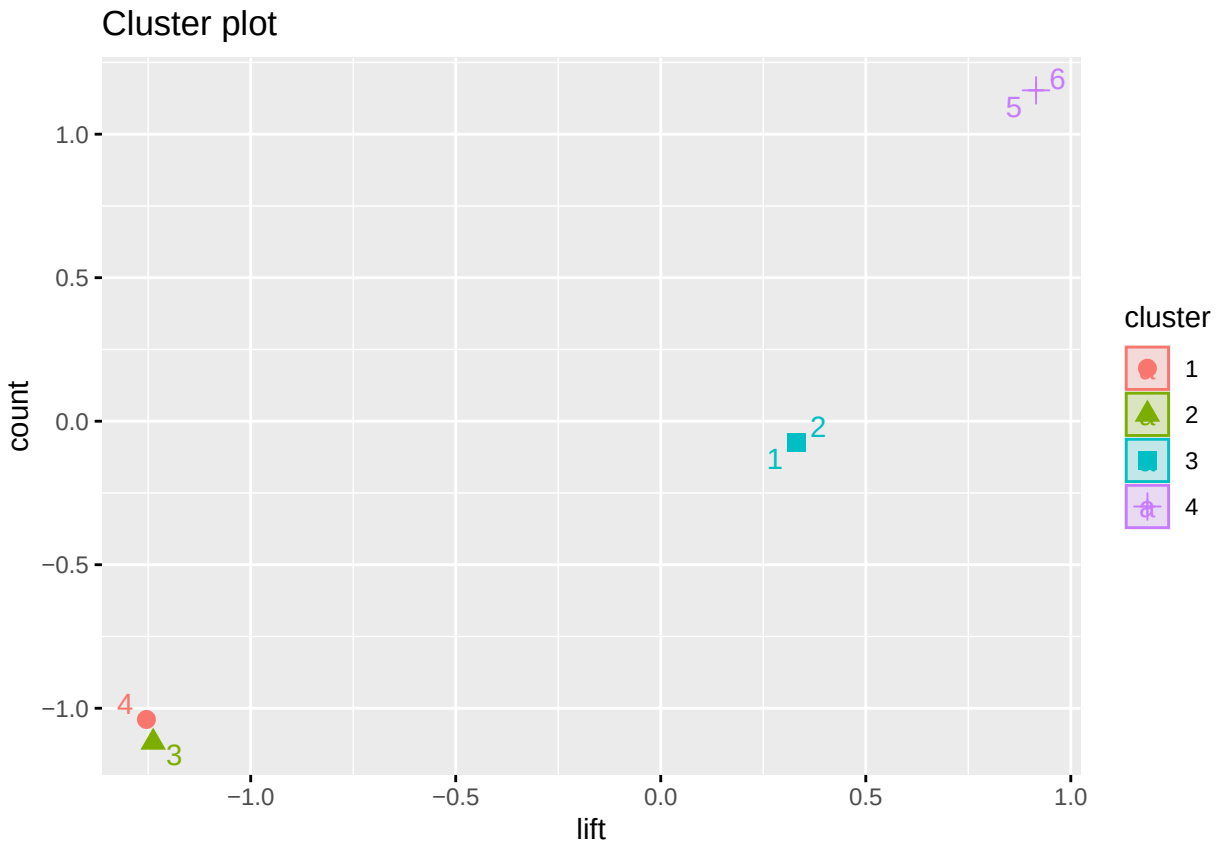



```
set.seed(102)

k_4 <- kmeans(grocery_rules_scaled, centers = 4, nstart = 25)
k_4$size
```

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

```
fviz_cluster(k_4, data = grocery_rules_scaled, repel = TRUE)
```


Cluster plot
count
lift
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
1
2
3
4
5
6
cluster
1
2
3
4