Which Beauty Product Combinations do Customers Often Buy Together?

Mining Association Rules for a Beauty Products Shop

John Karuitha

12/7/22

Table of Contents

1 Background **2**

2 Data **3**

3 Most Popular Items **5**

4 Items often Bought Together **7**

5 Interpreting the Results **10**

6 Implications of the Analysis **11**

7 Conclusion **12**

8 Technology and Packages Utilised **13**

References **17**

Chapter 1

Background

In this mini-project, I explore association rules using data from a beauty product shop.

> Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness.In any given transaction with a variety of items, association rules are meant to discover the rules that determine how or why certain items are connected (Kotsiantis and Kanellopoulos 2006).

The association rules that we explore in this analysis are;

- Which beauty products have the highest demand? (See section 3)
- Which combinations of beauty products are often purchased together? (See section 4)
- How best can the owner of this beauty shop utilise this analysis? (See section 6)

Read More of my Work

Please visit my rpubs site to see more data projects. Alternatively, copy and paste the link https://www.rpubs.com/Karuitha into your browser.
My data visualizations projects are available in my Tableau Public profile page or copy and paste the link https://public.tableau.com/app/profile/john.karuitha.

Tools Utilized & Skills Applied

R (R Core Team 2022), arules, arulesViz, Quarto, Data Science, Association Rules Mining.

Note, however, many rules generated using this technique maybe trivial. It would require domain expertise sift through the output to spot actionable rules.

Chapter 2

Data

The file Cosmetics.csv contains 1000 transaction information about cosmetics (1: purchased; 0: no purchased). Each transaction is tied to an invoice so that the data shows the basket of items that each consumer bought on the transaction date.

```
## read in the data
cosmetics <- read_csv("Cosmetics.csv") %>%
    clean_names() %>%
    mutate(invoice_number = paste0("c", 1:nrow(.))) %>%
    pivot_longer(cols = -invoice_number,
                 names_to = "item",
                 values_to = "bought") %>%
    filter(bought == 1) %>%
    mutate(invoice_number = factor(invoice_number)) %>%
    group_by(invoice_number) %>%
    filter(!duplicated(item)) %>%
    ungroup()

head(cosmetics, 10) %>% formatting_function(caption = "A Peek into the Data")
```

Table 2.1: A Peek into the Data

invoice_number	item	bought
c1	blush	1
c1	nail_polish	1
c1	brushes	1
c1	concealer	1
c1	bronzer	1
c1	lip_liner	1
c1	mascara	1
c1	eyeliner	1
c2	nail_polish	1
c2	concealer	1

Chapter 3

Most Popular Items

After reading in the data and converting the data into a transactions object, I build an item frequency plot. The plot shows the most popular items in all transactions contained in the data. The 2 most popular items are `foundation` and `lip gross`.

```
## Split the data by invoice number
my_bundles <- split(cosmetics$item, cosmetics$invoice_number)
my_bundles <- sapply(my_bundles, unique)
```

```
trans <- as(my_bundles, "transactions")
summary(trans)
```

```
transactions as itemMatrix in sparse format with
 957 rows (elements/itemsets/transactions) and
 14 columns (items) and a density of 0.33

most frequent items:
foundation  lip_gloss   eyeliner  concealer eye_shadow    (Other)
       536        490        457        442        381       2080

element (itemset/transaction) length distribution:
sizes
  1   2   3   4   5   6   7   8   9  10  11  12  13
 67 116 166 158 156 107  79  53  17  18  16   3   1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    1.0     3.0     4.0     4.6     6.0    13.0

includes extended item information - examples:
   labels
```

```
1     bag
2   blush
3 bronzer

includes extended transaction information - examples:
  transactionID
1            c1
2           c10
3          c100
```

```r
itemFrequency(trans)
```

```
          bag            blush          bronzer          brushes        concealer
        0.056            0.379            0.292            0.156            0.462
   eye_shadow eyebrow_pencils         eyeliner       foundation        lip_gloss
        0.398            0.044            0.478            0.560            0.512
    lip_liner         lipstick          mascara      nail_polish
        0.245            0.336            0.373            0.293
```

```r
itemFrequencyPlot(trans, topN = 10)
```

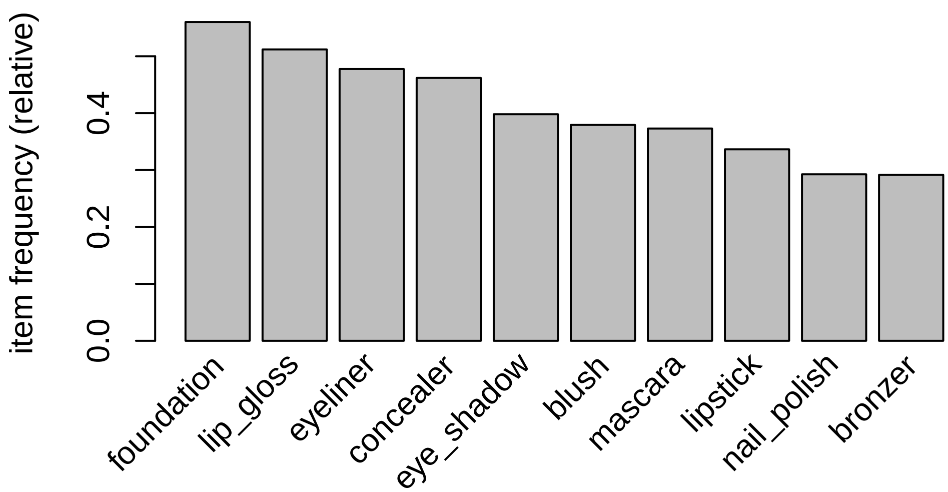

Figure 3.1: Most Frequently Bought Items

Chapter 4

Items often Bought Together

Next, I build an association rule model, setting the support value to 0.01 and the confidence value to 0.1. Based on the association rule results, I show the first eight rules after sorting the rules by their lift values.

> ❗ Important
>
> I define the terms support, confidence and lift later in section 6.

```
my_rules <- apriori(data = trans,
                    parameter = list(support = 0.01,
                                     confidence = 0.8,
                                     minlen=2))
```

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.8    0.1    1 none FALSE            TRUE       5    0.01      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: 9

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[14 item(s), 957 transaction(s)] done [0.00s].
```

```
sorting and recoding items ... [14 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.00s].
writing ... [4433 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
summary(my_rules)
```

```
set of 4433 rules

rule length distribution (lhs + rhs):sizes
   2    3    4    5    6    7    8    9   10
   3   53  338  862 1324 1149  549  140   15

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0     5.0     6.0     6.2     7.0    10.0

summary of quality measures:
    support       confidence      coverage         lift          count
 Min.   :0.01   Min.   :0.80   Min.   :0.01   Min.   :1.4   Min.   : 10
 1st Qu.:0.01   1st Qu.:0.88   1st Qu.:0.01   1st Qu.:2.1   1st Qu.: 12
 Median :0.02   Median :0.94   Median :0.02   Median :2.5   Median : 15
 Mean   :0.02   Mean   :0.93   Mean   :0.02   Mean   :2.7   Mean   : 21
 3rd Qu.:0.02   3rd Qu.:1.00   3rd Qu.:0.03   3rd Qu.:3.2   3rd Qu.: 23
 Max.   :0.34   Max.   :1.00   Max.   :0.40   Max.   :6.2   Max.   :321

mining info:
  data ntransactions support confidence
 trans           957    0.01        0.8
                                                                                  call
 apriori(data = trans, parameter = list(support = 0.01, confidence = 0.8, minlen = 2))
```

I convert the rules into a dataframe for ease of printing.

```
as(my_rules, "data.frame") %>%
    arrange(desc(lift)) %>%
    head(8) %>%
    gt(caption = "Top Rules by Lift")
```

rules	support	confidence	coverage
{bronzer,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.026	0.96	0.027
{bronzer,concealer,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.026	0.96	0.027
{bronzer,eye_shadow,eyeliner,lip_liner,nail_polish} =¿ {brushes}	0.024	0.96	0.025
{bronzer,concealer,eye_shadow,eyeliner,lip_liner,nail_polish} =¿ {brushes}	0.023	0.96	0.024
{bronzer,eye_shadow,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.022	0.95	0.023

{bronzer,concealer,eye_shadow,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.022	0.95	0.023
{blush,bronzer,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.018	0.94	0.019
{blush,bronzer,concealer,eyeliner,lip_liner,mascara,nail_polish} =¿ {brushes}	0.018	0.94	0.019

```
#inspect(sort(my_rules, by = "lift"))
```

Chapter 5

Interpreting the Results

I use the first rule from the table in section 4 (above) to illustrate the meaning of these rules. The first rule is as follows.

(bronzer,eyeliner,lip_liner,mascara,nail_polish) => {brushes}

This rule means that consumers who buy the basket of goods on the left (bronzer,eyeliner,lip_liner,mascara,nail_polish) often buy brushes. But How often do they buy brushes? To answer this question we explore the values for support, confidence and lift for this rule.

Support: First bronzer, eyeliner, lip_liner, mascara, and nail_polish constitute 3% of all sales.

Confidence: When a customer buys bronzer, eyeliner, lip_liner, mascara, nail_polish, there is a 96% chance of buying brushes.

Lift: Having bronzer, eyeliner, lip_liner, mascara, nail_polish in the shopping basket raises the probability that a customer will buy brushes sixfold (6.2).

Chapter 6

Implications of the Analysis

In this section, I give the precise meaning of “support”, “confidence”, and “lift”.I then discuss the implications of the results above in the setting of association rules and business.

Support:

Support is the percentage of groups that contain all of the items listed in the association rule. In the first rule above, the support is 0.03, meaning that 3% of the transactions contained the items listed on the right (bronzer,eyeliner,lip_liner,mascara,nail_polish => brushes).

Implication of support: The support shows the volume of transactions that a product or groups of products as a percent of total transactions. When combined with other confidence and lift, we can focus on selling products that generate biggers sales. In other words, support allows us to filter for meaningful rules. It would not be very useful to have a confidence of 99% but that constitutes only0.001% of sales.

Confidence:

Confidence is the proportion of times that a customer buys item X given that she/he buys item Y. In our first rule, given that the customer buys bronzer,eyeliner,lip_liner,mascara,nail_polish => brushes then that customer bought brushes 96% of the time.

The implication of confidence: We can target customers buying bronzer,eyeliner,lip_liner,mascara,nail_polish ith bushes as they have a 96% chance of buying. Alternatively, in a store, these products could be placed close to each other.

Lift:

The lift value captures rule importance. Usually, its the confidence of a rule divided by the support of a product. It is the rise in probability of the purchase of product Y once we know that product X is in the basket. Again, the implication is that it helps managers decide on product placements in stores.

Chapter 7

Conclusion

In this mini-project, I have explored association rules using data from a beauty product shop. The analysis has implications for the conduct of business. However, many rules generated using this technique maybe trivial. It would require domain expertise to spot actionable rules.

Chapter 8

Technology and Packages Utilised

In this analysis, I have utilised Zorin OS, R, Quarto and the following R packages.

```
sessioninfo::session_info()
```

```
- Session info ---------------------------------------------------------------
 setting  value
 version  R version 4.2.2 Patched (2022-11-10 r83330)
 os       Zorin OS 16.2
 system   x86_64, linux-gnu
 ui       X11
 language (EN)
 collate  en_US.UTF-8
 ctype    en_US.UTF-8
 tz       Africa/Nairobi
 date     2022-12-07
 pandoc   NA (via rmarkdown)

- Packages -------------------------------------------------------------------
 package       * version    date (UTC) lib source
 Amelia        * 1.8.1      2022-11-19 [1] CRAN (R 4.2.2)
 arules        * 1.7-5      2022-10-21 [1] CRAN (R 4.2.2)
 arulesViz     * 1.5-1      2021-11-19 [1] CRAN (R 4.2.2)
 assertthat      0.2.1      2019-03-21 [1] CRAN (R 4.2.2)
 backports       1.4.1      2021-12-13 [1] CRAN (R 4.2.2)
 base64enc       0.1-3      2015-07-28 [1] CRAN (R 4.2.2)
 BiocManager     1.30.19    2022-10-25 [1] CRAN (R 4.2.2)
 bit             4.0.5      2022-11-15 [1] CRAN (R 4.2.2)
 bit64           4.0.5      2020-08-30 [1] CRAN (R 4.2.2)
 broom         * 1.0.1      2022-08-29 [1] CRAN (R 4.2.2)
```

```
cellranger       1.1.0      2016-07-27 [1] CRAN (R 4.2.2)
class            7.3-20     2022-01-13 [4] CRAN (R 4.1.2)
cli              3.4.1      2022-09-23 [1] CRAN (R 4.2.2)
codetools        0.2-18     2020-11-04 [4] CRAN (R 4.0.3)
colorspace       2.0-3      2022-02-21 [1] CRAN (R 4.2.2)
corrplot       * 0.92       2021-11-18 [1] CRAN (R 4.2.2)
crayon           1.5.2      2022-09-29 [1] CRAN (R 4.2.2)
DBI              1.1.3      2022-06-18 [1] CRAN (R 4.2.2)
dbplyr           2.2.1      2022-06-27 [1] CRAN (R 4.2.2)
dials          * 1.1.0      2022-11-04 [1] CRAN (R 4.2.2)
DiceDesign       1.9        2021-02-13 [1] CRAN (R 4.2.2)
digest           0.6.30     2022-10-18 [1] CRAN (R 4.2.2)
dplyr          * 1.0.10     2022-09-01 [1] CRAN (R 4.2.2)
ellipsis         0.3.2      2021-04-29 [1] CRAN (R 4.2.2)
evaluate         0.18       2022-11-07 [1] CRAN (R 4.2.2)
extrafont        0.18       2022-04-12 [1] CRAN (R 4.2.2)
extrafontdb      1.0        2012-06-11 [1] CRAN (R 4.2.2)
fansi            1.0.3      2022-03-24 [1] CRAN (R 4.2.2)
farver           2.1.1      2022-07-06 [1] CRAN (R 4.2.2)
fastmap          1.1.0      2021-01-25 [1] CRAN (R 4.2.2)
firatheme      * 0.2.4      2022-11-25 [1] Github (vankesteren/firatheme@006d4d0)
forcats        * 0.5.2      2022-08-19 [1] CRAN (R 4.2.2)
foreach          1.5.2      2022-02-02 [1] CRAN (R 4.2.2)
foreign          0.8-84     2022-12-06 [1] CRAN (R 4.2.2)
fs               1.5.2      2021-12-08 [1] CRAN (R 4.2.2)
furrr            0.3.1      2022-08-15 [1] CRAN (R 4.2.2)
future           1.29.0     2022-11-06 [1] CRAN (R 4.2.2)
future.apply     1.10.0     2022-11-05 [1] CRAN (R 4.2.2)
gargle           1.2.1      2022-09-08 [1] CRAN (R 4.2.2)
generics         0.1.3      2022-07-05 [1] CRAN (R 4.2.2)
ggforce          0.4.1      2022-10-04 [1] CRAN (R 4.2.2)
ggplot2        * 3.4.0      2022-11-04 [1] CRAN (R 4.2.2)
ggraph           2.1.0      2022-10-09 [1] CRAN (R 4.2.2)
ggrepel          0.9.2      2022-11-06 [1] CRAN (R 4.2.2)
globals          0.16.2     2022-11-21 [1] CRAN (R 4.2.2)
glue             1.6.2      2022-02-24 [1] CRAN (R 4.2.2)
googledrive      2.0.0      2021-07-08 [1] CRAN (R 4.2.2)
googlesheets4    1.0.1      2022-08-13 [1] CRAN (R 4.2.2)
gower            1.0.0      2022-02-03 [1] CRAN (R 4.2.2)
GPfit            1.0-8      2019-02-08 [1] CRAN (R 4.2.2)
graphlayouts     0.8.4      2022-11-24 [1] CRAN (R 4.2.2)
gridExtra        2.3        2017-09-09 [1] CRAN (R 4.2.2)
gt             * 0.8.0      2022-11-16 [1] CRAN (R 4.2.2)
gtable           0.3.1      2022-09-01 [1] CRAN (R 4.2.2)
hardhat          1.2.0      2022-06-30 [1] CRAN (R 4.2.2)
haven            2.5.1      2022-08-22 [1] CRAN (R 4.2.2)
```

```
hms           1.1.2      2022-08-19 [1] CRAN (R 4.2.2)
htmltools     0.5.3      2022-07-18 [1] CRAN (R 4.2.2)
httr          1.4.4      2022-08-17 [1] CRAN (R 4.2.2)
igraph        1.3.5      2022-09-22 [1] CRAN (R 4.2.2)
infer       * 1.0.4      2022-12-02 [1] CRAN (R 4.2.2)
ipred         0.9-13     2022-06-02 [1] CRAN (R 4.2.2)
iterators     1.0.14     2022-02-05 [1] CRAN (R 4.2.2)
janitor     * 2.1.0      2021-01-05 [1] CRAN (R 4.2.2)
jsonlite      1.8.4      2022-12-06 [1] CRAN (R 4.2.2)
kableExtra  * 1.3.4      2021-02-20 [1] CRAN (R 4.2.2)
knitr         1.41       2022-11-18 [1] CRAN (R 4.2.2)
lattice       0.20-45    2021-09-22 [4] CRAN (R 4.2.0)
lava          1.7.0      2022-10-25 [1] CRAN (R 4.2.2)
lhs           1.1.5      2022-03-22 [1] CRAN (R 4.2.2)
lifecycle     1.0.3      2022-10-07 [1] CRAN (R 4.2.2)
listenv       0.8.0      2019-12-05 [1] CRAN (R 4.2.2)
lubridate     1.9.0      2022-11-06 [1] CRAN (R 4.2.2)
magrittr      2.0.3      2022-03-30 [1] CRAN (R 4.2.2)
MASS          7.3-58.1   2022-08-03 [1] CRAN (R 4.2.2)
Matrix      * 1.5-3      2022-11-11 [1] CRAN (R 4.2.2)
modeldata   * 1.0.1      2022-09-06 [1] CRAN (R 4.2.2)
modelr        0.1.10     2022-11-11 [1] CRAN (R 4.2.2)
munsell       0.5.0      2018-06-12 [1] CRAN (R 4.2.2)
nnet          7.3-18     2022-09-28 [4] CRAN (R 4.2.1)
pacman      * 0.5.1      2019-03-11 [1] CRAN (R 4.2.2)
parallelly    1.32.1     2022-07-21 [1] CRAN (R 4.2.2)
parsnip     * 1.0.3      2022-11-11 [1] CRAN (R 4.2.2)
pillar        1.8.1      2022-08-19 [1] CRAN (R 4.2.2)
pkgconfig     2.0.3      2019-09-22 [1] CRAN (R 4.2.2)
polyclip      1.10-4     2022-10-20 [1] CRAN (R 4.2.2)
prodlim       2019.11.13 2019-11-17 [1] CRAN (R 4.2.2)
purrr       * 0.3.5      2022-10-06 [1] CRAN (R 4.2.2)
R6            2.5.1      2021-08-19 [1] CRAN (R 4.2.2)
Rcpp        * 1.0.9      2022-07-08 [1] CRAN (R 4.2.2)
readr       * 2.1.3      2022-10-01 [1] CRAN (R 4.2.2)
readxl        1.4.1      2022-08-17 [1] CRAN (R 4.2.2)
recipes     * 1.0.3      2022-11-09 [1] CRAN (R 4.2.2)
repr          1.1.4      2022-01-04 [1] CRAN (R 4.2.2)
reprex        2.0.2      2022-08-17 [1] CRAN (R 4.2.2)
rlang         1.0.6      2022-09-24 [1] CRAN (R 4.2.2)
rmarkdown     2.18       2022-11-09 [1] CRAN (R 4.2.2)
rpart       * 4.1.19     2022-10-21 [4] CRAN (R 4.2.1)
rpart.plot  * 3.1.1      2022-05-21 [1] CRAN (R 4.2.2)
rsample     * 1.1.0      2022-08-08 [1] CRAN (R 4.2.2)
rstudioapi    0.14       2022-08-22 [1] CRAN (R 4.2.2)
Rttf2pt1      1.3.11     2022-10-08 [1] CRAN (R 4.2.2)
```

```
 rvest          1.0.3      2022-08-19 [1] CRAN (R 4.2.2)
 scales       * 1.2.1      2022-08-20 [1] CRAN (R 4.2.2)
 sessioninfo    1.2.2      2021-12-06 [1] CRAN (R 4.2.2)
 skimr        * 2.1.4      2022-04-15 [1] CRAN (R 4.2.2)
 snakecase      0.11.0     2019-05-25 [1] CRAN (R 4.2.2)
 stringi        1.7.8      2022-07-11 [1] CRAN (R 4.2.2)
 stringr      * 1.5.0      2022-12-02 [1] CRAN (R 4.2.2)
 survival       3.4-0      2022-08-09 [4] CRAN (R 4.2.1)
 svglite        2.1.0      2022-02-03 [1] CRAN (R 4.2.2)
 systemfonts    1.0.4      2022-02-11 [1] CRAN (R 4.2.2)
 tibble       * 3.1.8      2022-07-22 [1] CRAN (R 4.2.2)
 tidygraph      1.2.2      2022-08-22 [1] CRAN (R 4.2.2)
 tidymodels   * 1.0.0      2022-07-13 [1] CRAN (R 4.2.2)
 tidyr        * 1.2.1      2022-09-08 [1] CRAN (R 4.2.2)
 tidyselect     1.2.0      2022-10-10 [1] CRAN (R 4.2.2)
 tidyverse    * 1.3.2      2022-07-18 [1] CRAN (R 4.2.2)
 timechange     0.1.1      2022-11-04 [1] CRAN (R 4.2.2)
 timeDate       4021.106   2022-09-30 [1] CRAN (R 4.2.2)
 tune         * 1.0.1      2022-10-09 [1] CRAN (R 4.2.2)
 tweenr         2.0.2      2022-09-06 [1] CRAN (R 4.2.2)
 tzdb           0.3.0      2022-03-28 [1] CRAN (R 4.2.2)
 utf8           1.2.2      2021-07-24 [1] CRAN (R 4.2.2)
 vctrs          0.5.1      2022-11-16 [1] CRAN (R 4.2.2)
 viridis        0.6.2      2021-10-13 [1] CRAN (R 4.2.2)
 viridisLite    0.4.1      2022-08-22 [1] CRAN (R 4.2.2)
 vroom          1.6.0      2022-09-30 [1] CRAN (R 4.2.2)
 webshot        0.5.4      2022-09-26 [1] CRAN (R 4.2.2)
 withr          2.5.0      2022-03-03 [1] CRAN (R 4.2.2)
 workflows    * 1.1.2      2022-11-16 [1] CRAN (R 4.2.2)
 workflowsets * 1.0.0      2022-07-12 [1] CRAN (R 4.2.2)
 xfun           0.35       2022-11-16 [1] CRAN (R 4.2.2)
 xml2           1.3.3      2021-11-30 [1] CRAN (R 4.2.2)
 xtable       * 1.8-4      2019-04-21 [1] CRAN (R 4.2.2)
 yaml           2.3.6      2022-10-18 [1] CRAN (R 4.2.2)
 yardstick    * 1.1.0      2022-09-07 [1] CRAN (R 4.2.2)

 [1] /home/karuitha/R/x86_64-pc-linux-gnu-library/4.2
 [2] /usr/local/lib/R/site-library
 [3] /usr/lib/R/site-library
 [4] /usr/lib/R/library

------------------------------------------------------------------------------
```

References

Kotsiantis, Sotiris, and Dimitris Kanellopoulos. 2006. "Association Rules Mining: A Recent Overview." *GESTS International Transactions on Computer Science and Engineering* 32 (1): 71–82.

R Core Team. 2022. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.