Movie Recommender

Ross Jackson

2023-03-07

```
library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'arules'

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

```
movies <- read.transactions("data_movielens_hw1.csv", format = "single",
                            sep = ",", cols = c("userId", "title"), header = TRUE)
```

Obtaining a set of rules of manageable size. In doing so, taking into consideration that the company is mainly interested in developing a system that recommends a movie according to its association with at least two other movies. PLotting the data in order to get a grasp of its characteristics.

```
inspect(movies[1:3])
```

```
##     items                                       transactionID
## [1] {Logan,
##      The Fundamentals of Caring}                        1000
## [2] {Arrival,
##      Baby Driver,
##      Blade Runner 2049,
##      Call Me by Your Name,
##      First Reformed,
##      Get Out,
##      Hereditary,
##      Lady Bird,
##      Mandy,
##      Mother!,
##      Phantom Thread,
##      Piper,
##      The Handmaiden,
##      The Neon Demon,
##      Three Billboards Outside Ebbing, Missouri}         10000
## [3] {Blade Runner 2049,
##      Get Out,
##      Lovesong}                                          100020
```

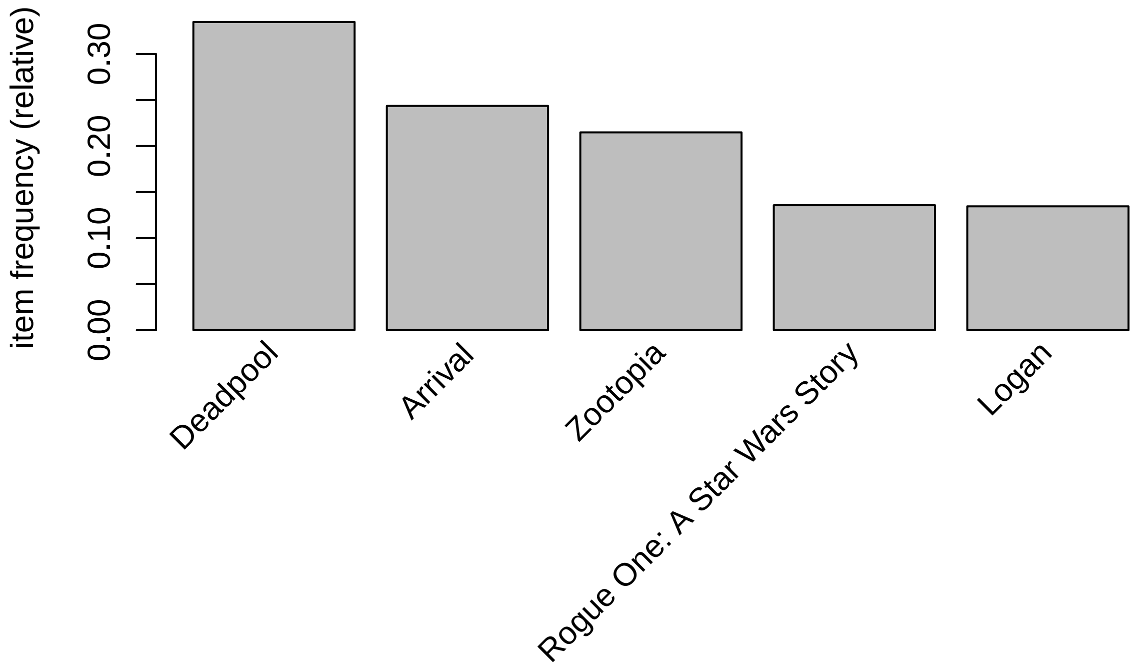
```
summary(movies)
```

```
## transactions as itemMatrix in sparse format with
##  24857 rows (elements/itemsets/transactions) and
##  3222 columns (items) and a density of 0.002046879 
## 
## most frequent items:
##                     Deadpool                      Arrival 
##                         8322                         6055 
##                     Zootopia Rogue One: A Star Wars Story 
##                         5341                         3374 
##                        Logan                      (Other) 
##                         3344                       137497 
## 
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16 
## 7648 3962 2460 1807 1249  916  843  643  568  442  395  386  308  281  245  236 
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32 
##  197  183  158  126  152  113  102  101   95   85   82   71   65   59   62   52 
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48 
##   47   45   35   37   39   27   37   25   27   33   17   27   28   15   17   11 
##   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64 
##   22   15    8    7   15   13   13    9   16   12    9    8    8    9    5    3 
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80 
##    3    9    9    4    3    4    5    7    4    2    3    1    4    4    6    3 
##   81   82   83   84   85   86   87   88   90   91   93   94   96   97  100  103 
##    1    3    1    5    2    5    3    1    3    2    3    2    1    2    2    1 
##  104  106  108  109  111  112  113  117  119  122  123  138  161  182  183  193 
##    1    1    1    1    1    3    1    1    1    1    1    1    1    1    1    1 
## 
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   1.000   1.000   3.000   6.595   7.000 193.000 
## 
## includes extended item information - examples:
##          labels
## 1    '63 Boycott
## 2  #realityhigh
## 3 1 Mile to You
## 
## includes extended transaction information - examples:
##   transactionID
## 1          1000
## 2         10000
## 3        100020
```

```
movie_freq <- itemFrequency(movies)
head(movie_freq)
```

```
##          '63 Boycott        #realityhigh       1 Mile to You             1 Night 
##         4.023012e-05        2.816108e-04        4.023012e-05        1.609205e-04 
##                 1:54 10 Cloverfield Lane 
##         8.046023e-05        6.521302e-02
```

```
itemFrequencyPlot(movies, topN = 5)
```



Parameter choice explanation: support: too many rules returned when support= 0.01, had to be increased confidence: Will return rules with higher association with eachother minlen = 2, as it is specified that "take into consideration that the company is mainly interested in developing a system that recommends a movie according to its association with at least two other movies." maxlen = 8 as warning message returned otherwise

```
movie_rules <- apriori(movies, parameter = list(support = 0.02, confidence = 0.8, minlen = 2, maxlen =
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5    0.02      2
##  maxlen target  ext
##       8  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 497
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[3222 item(s), 24857 transaction(s)] done [0.06s].
## sorting and recoding items ... [71 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 5 done [0.01s].
## writing ... [12 rule(s)] done [0.00s].
## creating S4 object  ... done [0.01s].
```

```
summary(movie_rules)
```

```
## set of 12 rules
##
```

```
## rule length distribution (lhs + rhs):sizes
##  4
## 12
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       4       4       4       4       4       4
##
## summary of quality measures:
##     support           confidence        coverage             lift
##  Min.   :0.02003   Min.   :0.8002   Min.   :0.02450   Min.   :2.390
##  1st Qu.:0.02035   1st Qu.:0.8054   1st Qu.:0.02512   1st Qu.:2.406
##  Median :0.02124   Median :0.8068   Median :0.02639   Median :2.459
##  Mean   :0.02216   Mean   :0.8133   Mean   :0.02724   Mean   :3.815
##  3rd Qu.:0.02362   3rd Qu.:0.8203   3rd Qu.:0.02879   3rd Qu.:6.519
##  Max.   :0.02639   Max.   :0.8325   Max.   :0.03243   Max.   :6.701
##      count
##  Min.   :498.0
##  1st Qu.:505.8
##  Median :528.0
##  Mean   :550.8
##  3rd Qu.:587.0
##  Max.   :656.0
##
## mining info:
##    data ntransactions support confidence
##  movies         24857    0.02        0.8
##                                                                                                 call
##  apriori(data = movies, parameter = list(support = 0.02, confidence = 0.8, minlen = 2, maxlen = 8))
```

```r
rules1 <- sort(movie_rules, by = "support") # sort by support
inspect(rules1[1:5]) #inspect the top 5 rules
```

```
##     lhs                              rhs                            support confidence   coverage
## [1] {Doctor Strange,
##      Guardians of the Galaxy 2,
##      Logan}                       => {Deadpool}                  0.02639096  0.8324873 0.03170133 2.4
## [2] {Captain America: Civil War,
##      Doctor Strange,
##      Guardians of the Galaxy 2}   => {Deadpool}                  0.02594842  0.8002481 0.03242547 2.3
## [3] {Captain America: Civil War,
##      Doctor Strange,
##      Logan}                       => {Deadpool}                  0.02458060  0.8301630 0.02960937 2.4
## [4] {Captain America: Civil War,
##      Guardians of the Galaxy 2,
##      Logan}                       => {Deadpool}                  0.02329324  0.8166432 0.02852315 2.4
## [5] {Deadpool,
##      Doctor Strange,
##      Thor: Ragnarok}              => {Guardians of the Galaxy 2} 0.02212656  0.8064516 0.02743694 6.5
```

Below I have computed the standardized lifts and bounds for the rules. This allows us to compare the rules on a level playing field.

```r
## Standardized Lift:
rules1 <- apriori(movies, parameter = list(support = 0.02, confidence = 0.8, minlen = 2, maxlen = 8))
```

```
## Apriori
```

```
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5    0.02      2
##  maxlen target  ext
##       8  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 497
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[3222 item(s), 24857 transaction(s)] done [0.05s].
## sorting and recoding items ... [71 item(s)] done [0.00s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 5 done [0.01s].
## writing ... [12 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
qual <- quality(rules1) # extract quality measures

# compute p(A) and p(B)
pA <- qual$coverage
pB <- qual$confidence/qual$lift
# compute lift upper and lower bounds
U <- apply(cbind(1/pA, 1/pB), 1, min)
L <- apply(cbind(1/pA + 1/pB - 1/(pA*pB), 0.01/(pA*pB), 0.5/pB, 0), 1, max)
std_lift <- (qual$lift - L)/(U - L) # standardized lift
data.frame(rule = labels(rules1),
           lift = qual$lift, L, U, std_lift) # print rules and associated metrics
```

```
##                                                                                          rule
## 1        {Deadpool,Thor: Ragnarok,Untitled Spider-Man Reboot} => {Guardians of the Galaxy 2}
## 2        {Doctor Strange,Guardians of the Galaxy 2,Untitled Spider-Man Reboot} => {Deadpool}
## 3        {Deadpool,Doctor Strange,Untitled Spider-Man Reboot} => {Guardians of the Galaxy 2}
## 4  {Captain America: Civil War,Doctor Strange,Thor: Ragnarok} => {Guardians of the Galaxy 2}
## 5                    {Deadpool,Doctor Strange,Thor: Ragnarok} => {Guardians of the Galaxy 2}
## 6       {Captain America: Civil War,Doctor Strange,Guardians of the Galaxy 2} => {Deadpool}
## 7                            {Captain America: Civil War,Doctor Strange,Logan} => {Deadpool}
## 8                         {Captain America: Civil War,Doctor Strange,Zootopia} => {Deadpool}
## 9              {Captain America: Civil War,Logan,Rogue One: A Star Wars Story} => {Deadpool}
## 10                {Captain America: Civil War,Guardians of the Galaxy 2,Logan} => {Deadpool}
## 11                         {Doctor Strange,Logan,Rogue One: A Star Wars Story} => {Deadpool}
## 12                            {Doctor Strange,Guardians of the Galaxy 2,Logan} => {Deadpool}
##        lift        L        U  std_lift
## 1  6.517054 4.045736 8.091471 0.6108453
## 2  2.406644 1.493451 2.986902 0.6114650
## 3  6.700957 4.045736 8.091471 0.6563011
## 4  6.616671 4.045736 8.091471 0.6354680
## 5  6.525380 4.045736 8.091471 0.6129032
## 6  2.390263 1.493451 2.986902 0.6004963
## 7  2.479616 1.493451 2.986902 0.6603261
## 8  2.405340 1.493451 2.986902 0.6105919
## 9  2.410822 1.493451 2.986902 0.6142626
## 10 2.439233 1.493451 2.986902 0.6332863
## 11 2.402896 1.493451 2.986902 0.6089552
## 12 2.486558 1.493451 2.986902 0.6649746
```

As we can see from this, rules 12, 7 and 3 are 3 of the most interesting, as there standardized lifts are the highest obtained from the set of 12 rules.