Association Rule Task

```
#Load necessary package
library(ggplot2); library(ggiraph); library(ggiraphExtra); library(RColorBrewer); library(arules)
```

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

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

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

Description of Data

The file data being used `data_movielens_hw1.csv` contains viewing and liking patterns of 24,857 subscribers, concerning 3,222 movies over the period of 3 years (2016-2018). If a user has given a rating of 4 or more (out of 5), the user has liked the movie and it is included in the dataset.

The data contains 163938 observations and 2 columns (UserId and Title). Both columns will be used in the analysis, where:

- UserId, that is recorder as “transaction”
- Title, as “items”, which contains information about the liked movie from each user in a particular year

```
df1 <- read.csv("data_movielens_hw1.csv")  # load the data and store it as "df1"
dim(df1) # present data dimension
```

```
## [1] 163938      2
```

```
head(df1) # print first 6 row of the data
```

```
##   userId    title
## 1     14 Deadpool
## 2     14 Zootopia
## 3     14    Piper
## 4     14     Coco
## 5     34 Zootopia
## 6     36       It
```

```
length(unique(df1$userId)) # compute the total count of unique subscriber
```

```
## [1] 24857
```

```
length(unique(df1$title)) # compute the total count of unique movie
```

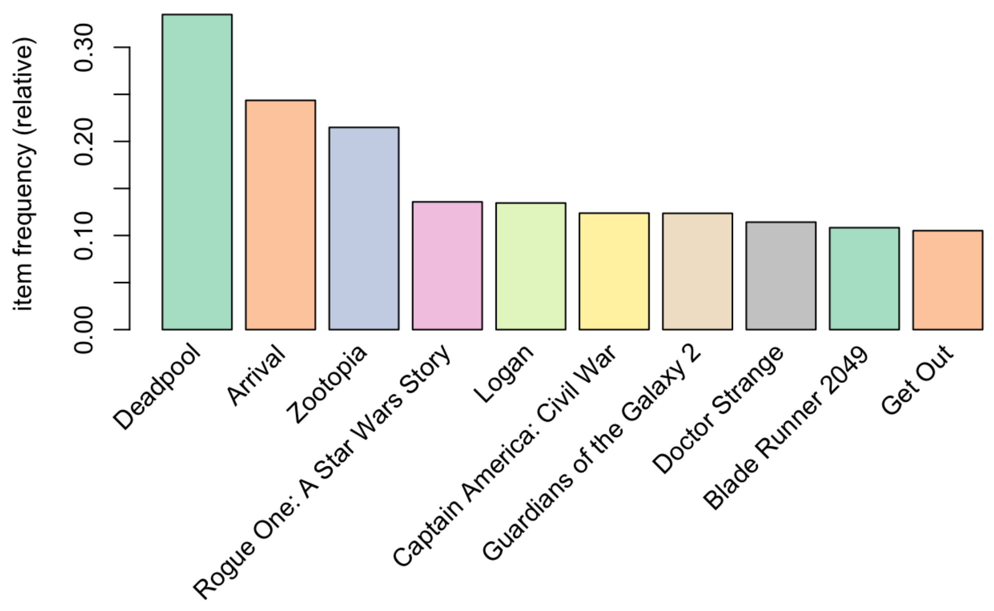
```
## [1] 3222
```

Below, I run the following commands to read the data in transactional format:

```
# Read the data in transactional format
tr_data <- read.transactions("data_movielens_hw1.csv", format = "single",
sep = ",", cols = c("userId", "title"), header = TRUE)
```

Before mining the rules, we will look at the 10 most frequent items from the transactions.

```
# Create plot for 10 most frequent items
itemFrequencyPlot(tr_data, topN = 10, col=brewer.pal(8,'Pastel2'))
```



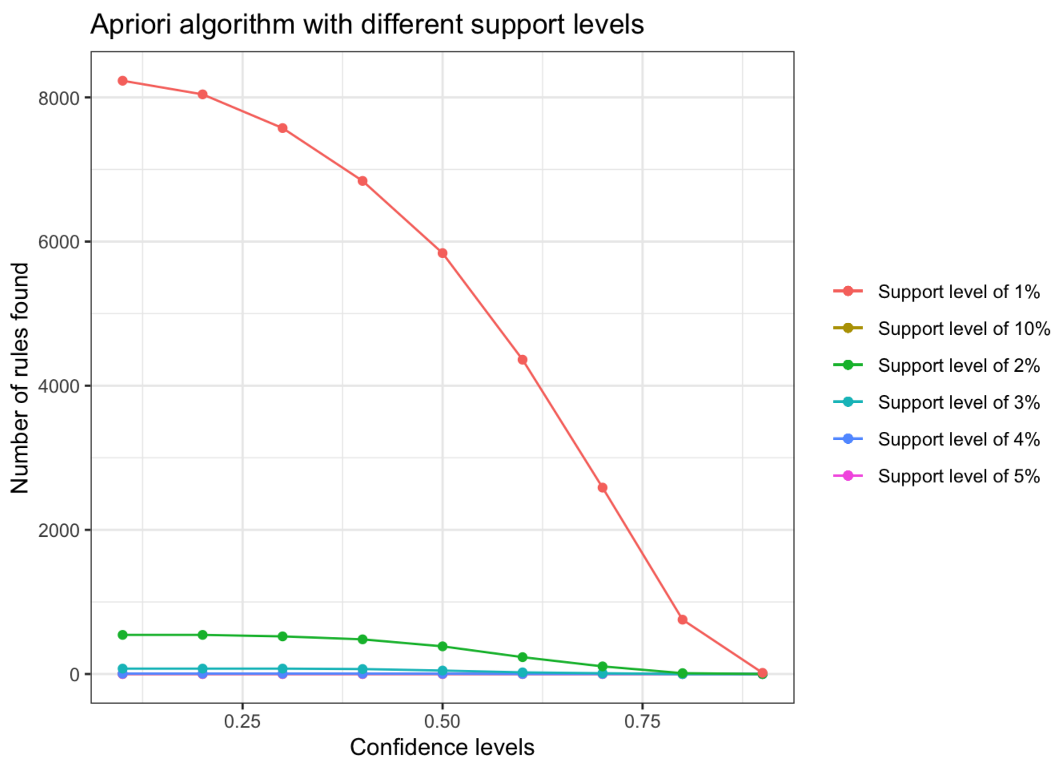
The movie Deadpool appears to be the most frequent item in all transactions followed by Arrival and Zootopia. Let's see if this superiority is reflected in the association rules.

Choosing the hyperparameters for apriori function

Here I try different values of support and confidence and see graphically how many rules are generated for each combination to determine the support and confidence value for manageable size rules. However, I set the minimum size of the rule to 3, since we want to develop movie recommendation according to its association with at least two other movies.

```r
# Support and confidence values
supportLevels <- c(0.1, 0.05, 0.04, 0.03, 0.02, 0.01) # create different support levels in a vector
confidenceLevels <- c(0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1) # create different confidence levels in a vector

# Empty integers to store number of rules in each confidence level
rules_sup10 <- integer(length=9)
rules_sup5 <- integer(length=9)
rules_sup4 <- integer(length=9)
rules_sup3 <- integer(length=9)
rules_sup2 <- integer(length=9)
rules_sup1 <- integer(length=9)

# Create for loop to try different values of support and confidence
# Apriori algorithm with a support level of 10%
for (i in 1:length(confidenceLevels)) {

  rules_sup10[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[1],
                                   conf=confidenceLevels[i], minlen = 3)))

}

# Apriori algorithm with a support level of 5%
for (i in 1:length(confidenceLevels)){

  rules_sup5[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[2],
                                  conf=confidenceLevels[i], minlen = 3)))

}

# Apriori algorithm with a support level of 4%
for (i in 1:length(confidenceLevels)){

  rules_sup4[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[3],
                                  conf=confidenceLevels[i], minlen = 3)))

}

# Apriori algorithm with a support level of 3%
for (i in 1:length(confidenceLevels)){

  rules_sup3[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[4],
                                  conf=confidenceLevels[i], minlen = 3)))

}

# Apriori algorithm with a support level of 2%
for (i in 1:length(confidenceLevels)){

  rules_sup2[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[5],
                                  conf=confidenceLevels[i], minlen = 3)))

}

# Apriori algorithm with a support level of 1%
for (i in 1:length(confidenceLevels)){

  rules_sup1[i] <- length(apriori(tr_data, parameter=list(sup=supportLevels[6],
                                  conf=confidenceLevels[i], minlen = 3)))

}
```

```r
# Create data frame to store the number of rules created above and the confidence levels
num_rules <- data.frame(rules_sup10, rules_sup5, rules_sup4, rules_sup3, rules_sup2, rules_sup1, confidenceL
evels)

# Create plot for number of rules found with a support level of 10%, 5%, 4%, 3%, 2%, and 1% as the y axis
# and confidence level as the x axis
ggplot(data=num_rules, aes(x=confidenceLevels)) +

  # Plot line and points (support level of 10%)
  geom_line(aes(y=rules_sup10, colour="Support level of 10%")) +
  geom_point(aes(y=rules_sup10, colour="Support level of 10%")) +

  # Plot line and points (support level of 5%)
  geom_line(aes(y=rules_sup5, colour="Support level of 5%")) +
  geom_point(aes(y=rules_sup5, colour="Support level of 5%")) +

  # Plot line and points (support level of 4%)
  geom_line(aes(y=rules_sup4, colour="Support level of 4%")) +
  geom_point(aes(y=rules_sup4, colour="Support level of 4%")) +

  # Plot line and points (support level of 3%)
  geom_line(aes(y=rules_sup3, colour="Support level of 3%")) +
  geom_point(aes(y=rules_sup3, colour="Support level of 3%")) +

  # Plot line and points (support level of 2%)
  geom_line(aes(y=rules_sup2, colour="Support level of 2%")) +
  geom_point(aes(y=rules_sup2, colour="Support level of 2%")) +

  # Plot line and points (support level of 1%)
  geom_line(aes(y=rules_sup1, colour="Support level of 1%")) +
  geom_point(aes(y=rules_sup1, colour="Support level of 1%")) +


  # Labs and theme
  labs(x="Confidence levels", y="Number of rules found",
       title="Apriori algorithm with different support levels") +
  theme_bw() +
  theme(legend.title=element_blank())
```



Here, I analyze the result:
- Support level 5% and 10%, we do not identify any rules.
- Support level of 3-4%, we only identify a few rules with very low confidence levels.

- Support level of 2%, we start to get manageable size rules with confidence level of 50%.
- Support level of 1%, we still get dozens of rules here.

To sum up, I choose support level of 2%, meaning that the rule must occur in at least 2% of the transactions in order to be considered for inclusion in the final set of rules and a confidence level of 60%, meaning that the antecedent of the rule (the left-hand side) must be associated with the consequent (the right-hand side) at least 60% of the time. Furthermore, I also set the minimum size of the rule to 3, to develop movie recommendation according to its association with at least two other movies.

```
# Run the apriori algorithm on the transaction data with the selected hyperparameters
rules <- apriori(tr_data,
                parameter = list(support = 0.02,
                                 confidence = 0.6,
                                 minlen = 3))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5    0.02      3
##  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: 497
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[3222 item(s), 24857 transaction(s)] done [0.04s].
## 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 ... [234 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

From the summary we can infer the following:

- 234 rules were formed
- Minimum and maximum values for support, confidence and lift are displayed
- Out of the 234 rules formed the counts of rule length is also displayed

```
# Print the summary rules
summary(rules)
```

```
## set of 234 rules
##
## rule length distribution (lhs + rhs):sizes
##   3   4
## 153  81
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     3.0     3.0     3.0     3.3     4.0     4.0
##
## summary of quality measures:
##     support        confidence      coverage          lift          count
##  Min.   :0.020   Min.   :0.60   Min.   :0.025   Min.   : 1.8   Min.   : 498
##  1st Qu.:0.021   1st Qu.:0.65   1st Qu.:0.030   1st Qu.: 2.3   1st Qu.: 526
##  Median :0.023   Median :0.69   Median :0.034   Median : 4.9   Median : 565
##  Mean   :0.024   Mean   :0.69   Mean   :0.035   Mean   : 4.5   Mean   : 600
##  3rd Qu.:0.025   3rd Qu.:0.74   3rd Qu.:0.038   3rd Qu.: 5.8   3rd Qu.: 630
##  Max.   :0.041   Max.   :0.83   Max.   :0.056   Max.   :10.6   Max.   :1015
##
## mining info:
##     data ntransactions support confidence
##  tr_data         24857    0.02        0.6
##                                                                                 call
##  apriori(data = tr_data, parameter = list(support = 0.02, confidence = 0.6, minlen = 3))
```

Let's sort the rules based on confidence levels.

```
# Print the first 6 rows of rules and sort by the highest confidence level
print(inspect(head(sort(rules, by="confidence", decreasing = "TRUE"))))
```

```
##     lhs                                 rhs                         support confidence coverage lift count
## [1] {Doctor Strange,
##      Guardians of the Galaxy 2,
##      Logan}                          => {Deadpool}                    0.026       0.83    0.032  2.5   656
## [2] {Captain America: Civil War,
##      Doctor Strange,
##      Logan}                          => {Deadpool}                    0.025       0.83    0.030  2.5   611
## [3] {Deadpool,
##      Doctor Strange,
##      Untitled Spider-Man Reboot}   => {Guardians of the Galaxy 2}   0.020       0.83    0.025  6.7   506
## [4] {Captain America: Civil War,
##      Doctor Strange,
##      Thor: Ragnarok}               => {Guardians of the Galaxy 2}   0.020       0.82    0.025  6.6   498
## [5] {Captain America: Civil War,
##      Guardians of the Galaxy 2,
##      Logan}                          => {Deadpool}                    0.023       0.82    0.029  2.4   579
## [6] {Captain America: Civil War,
##      Logan,
##      Rogue One: A Star Wars Story} => {Deadpool}                    0.020       0.81    0.025  2.4   498
## NULL
```

We can see that the movie Deadpool is being mentioned many times, where we can conclude that it is in good agreement with the most frequency that we obtain in support rates part.

Moreover, it interesting to see that the rules are mostly combinations of Marvel Cinematic Universe movies. This is probably because most of the person who watched in the platform are Marvel fans.

With having highest confidence, we can infer that 83% that a person already watch Doctor Strange, Logan, Guardians of the Galaxy 2/Captain America will watch Deadpool.

Here, I will sorted the rules by lift levels.

```
# Print the first 6 rows of rules and sort by the highest lift level
print(inspect(head(sort(rules, by="lift", decreasing = "TRUE"))))
```

```
##     lhs                                     rhs                             support confidence coverage lift c
ount
## [1] {Captain America: Civil War,
##      Thor: Ragnarok}                     => {Untitled Spider-Man Reboot}   0.024       0.65    0.037 10.6
594
## [2] {Doctor Strange,
##      Thor: Ragnarok}                     => {Untitled Spider-Man Reboot}   0.022       0.61    0.037  9.9
559
## [3] {Deadpool,
##      Guardians of the Galaxy 2,
##      Thor: Ragnarok}                     => {Untitled Spider-Man Reboot}   0.020       0.61    0.033  9.9
505
## [4] {Avengers: Infinity War - Part I,
##      Untitled Spider-Man Reboot}         => {Thor: Ragnarok}               0.021       0.80    0.026  9.3
514
## [5] {Avengers: Infinity War - Part I,
##      Captain America: Civil War}         => {Thor: Ragnarok}               0.022       0.76    0.029  8.9
539
## [6] {Avengers: Infinity War - Part I,
##      Guardians of the Galaxy 2}          => {Thor: Ragnarok}               0.026       0.75    0.035  8.8
653
## NULL
```

A lift value greater than one indicates that items in RHS are more likely to be watched and liked with items on LHS. Similarly a lift value lesser than one implies that items in RHS are unlikely to be watched and liked with items in LHS.

In the above first transaction, we can conclude that Untitled Spider-Man Reboot are being watched and liked ten times more with Captain America: Civil War and Thor: Ragnarok than it being watched and liked alone.

Moreover, I would like to inspect the data by standardized lift because lift itself is susceptible to noise in small databases. Rare itemsets with low counts (low probability), which by chance occur a few times (or only once) together, can produce enormous lift values.

Here, I compute the standardized lift formula and form a new data frame that merge the standardized lift value with other association rules metrics *using the formula from the Lab*.

```
qual<-quality(rules) # extract quality measures
pA <- qual$coverage
pB <- qual$confidence/qual$lift
# compute lift upper and lower bounds
U <- apply(cbind(1/pA, 1/pB), 1, min)
L <- apply(cbind(1/pA + 1/pB - 1/(pA*pB), 0.02/(pA*pB), 0.6/pB, 0), 1, max)
std_lift <- (qual$lift - L)/(U - L) # standardized lift
df2 <- data.frame(rule = labels(rules),
                  lift=qual$lift,L,U,std_lift) # print rules and associated metrics and store it as a datafra
me "df2"
```

```
df3 <- df2[order(df2$std_lift, decreasing=TRUE),] # create df3 with sorted standardized lift by the highest
number
print(head(df3,10)) # print the first 6 rows of df3
```

```
##                                                                                      rule
## 214                       {Doctor Strange,Guardians of the Galaxy 2,Logan} => {Deadpool}
## 185 {Captain America: Civil War,Doctor Strange,Guardians of the Galaxy 2} => {Deadpool}
## 189                     {Captain America: Civil War,Doctor Strange,Logan} => {Deadpool}
## 141                                                {Doctor Strange,Logan} => {Deadpool}
## 106                       {Doctor Strange,Thor: Ragnarok} => {Guardians of the Galaxy 2}
## 50             {Doctor Strange,Untitled Spider-Man Reboot} => {Guardians of the Galaxy 2}
## 134                                   {Captain America: Civil War,Logan} => {Deadpool}
## 129                          {Captain America: Civil War,Doctor Strange} => {Deadpool}
## 36             {Thor: Ragnarok,Untitled Spider-Man Reboot} => {Guardians of the Galaxy 2}
## 64                                    {Logan,Untitled Spider-Man Reboot} => {Deadpool}
##     lift   L   U std_lift
## 214  2.5 1.9 3.0     0.55
## 185  2.4 1.8 3.0     0.48
## 189  2.5 2.0 3.0     0.48
## 141  2.3 1.8 3.0     0.44
## 106  6.2 4.9 8.1     0.42
## 50   6.3 5.0 8.1     0.42
## 134  2.3 1.8 3.0     0.42
## 129  2.3 1.8 3.0     0.41
## 36   6.1 4.9 8.1     0.40
## 64   2.3 1.8 3.0     0.39
```

By sorting the standardized lift, we can see in the first transaction, Deadpool are being watched and liked more with Doctor Strange, Guardians of the Galaxy 2, Logan than it being watched and liked alone (with standardized lift of 0.55). From this, we can see that our standardized lift is in good agreement with our finding by using the confidence metric.

Summary

- There are many Marvel superhero movie lovers in the platform as being seen in the interesting rules that we obtained above are all Marvel superhero movies.
- This is make sense because Marvel movies are always connected/related with each other. For example: before watching Thor Ragnarok, people will watch Guardians of Galaxy 2 (released before Thor Ragnarok) first because in Thor Ragnarok there will be a part that is connected with Guardians of Galaxy 2. Thus, people will mostly watch all superhero films to keeping in track with the storyline before watching new released Marvel movies.

References

Market Basket Analysis. Available at: https://rpubs.com/CFernandez/686167 (https://rpubs.com/CFernandez/686167) (Accessed: February 25, 2023).