Movie Recommendation with Market Basket Analysis

Introduction

In this project, we applied a data mining algorithm, Apriori, to mine a relationship among films and build a movie recommendation engine. Apriori is a technique in Market Basket Analysis used to discover items that are frequently sold together. Frequently purchased itemset suggests marketing opportunity when customers displayed interest in the subset items. In this case, movies can be viewed as a set of items. We obtained our training data from MovieLens's website(http://grouplens.org/datasets/movielens/ (http://grouplens.org/datasets/movielens/)). We used `MovieLens 20M Dataset` dataset which consisted 20,000,263 user ratings, across 27,278 movies and 138,493 raters. We found that the mining technique can be utilized to uncover an underlying connection within the movies. It can also be used in a movie recommendation, but a number of suggested films can be quite limited and the quality of such suggestions can be vary.

Explore Movie Data

We extract the compressed file in `ml-20m` directory. We interest in 2 data files:

- movies.csv(contains information about movies)
- ratings.csv(user ratings)

We firstly investigate movies.csv file

```
#load all required package
library(arules)
library(dplyr)
library(reshape2)
library(Matrix)
library(stringr)
library(stringdist)
library(ggplot2)

setwd("/home/vitidn/mydata/repo_git/MovieMBA/")
```

```
movies = read.csv("ml-20m/movies.csv",
                  colClasses = c("integer","character","character"),
                  sep = ",",
                  stringsAsFactors = FALSE)

head(movies)
```

```
##   movieId                               title
## 1       1                    Toy Story (1995)
## 2       2                      Jumanji (1995)
## 3       3             Grumpier Old Men (1995)
## 4       4            Waiting to Exhale (1995)
## 5       5 Father of the Bride Part II (1995)
## 6       6                         Heat (1995)
##                                          genres
## 1 Adventure|Animation|Children|Comedy|Fantasy
## 2                  Adventure|Children|Fantasy
## 3                              Comedy|Romance
## 4                        Comedy|Drama|Romance
## 5                                      Comedy
## 6                       Action|Crime|Thriller
```

We separate released year from a title

```
movies$year = as.numeric(str_sub( str_trim(movies$title) ,start = -5,end = -2))
```

```
## Warning: NAs introduced by coercion
```

Some movies have no release year/invalid title format. We note their movieId and discard them for this analysis.

```
discard_movie_id = which(is.na(movies$year))
#display discarded movies
movies$title[discard_movie_id]
```

```
##  [1] "Babylon 5"
##  [2] "Millions Game, The (Das Millionenspiel)"
##  [3] "Bicycle, Spoon, Apple (Bicicleta, cullera, poma)"
##  [4] "Mona and the Time of Burning Love (Mona ja palavan rakkauden aika) (1983))"
##  [5] "Diplomatic Immunity (2009– )"
##  [6] "Big Bang Theory, The (2007-)"
##  [7] "Brazil: In the Shadow of the Stadiums"
##  [8] "Slaying the Badger"
##  [9] "Tatort: Im Schmerz geboren"
## [10] "National Theatre Live: Frankenstein"
## [11] "The Court-Martial of Jackie Robinson"
## [12] "In Our Garden"
## [13] "Stephen Fry In America - New World"
## [14] "Two: The Story of Roman & Nyro"
## [15] "Li'l Quinquin"
## [16] "A Year Along the Abandoned Road"
## [17] "Body/Cialo"
## [18] "Polskie gówno"
## [19] "The Third Reich: The Rise & Fall"
## [20] "My Own Man"
## [21] "Moving Alan"
## [22] "Michael Laudrup - en Fodboldspiller"
```

```
movies = movies[-discard_movie_id,]
```

Title is extracted

```
movies$title = str_sub( str_trim(movies$title) ,start = 1,end = -8)
```

Next, we would like to extract genres for each movie(noted that each film can belong to more than one genre). We look at total number of genres.

```
all_genres = unique(unlist(str_split(movies$genres,"\\|")))

all_genres
```

```
##  [1] "Adventure"          "Animation"          "Children"
##  [4] "Comedy"             "Fantasy"            "Romance"
##  [7] "Drama"              "Action"             "Crime"
## [10] "Thriller"           "Horror"             "Mystery"
## [13] "Sci-Fi"             "IMAX"               "Documentary"
## [16] "War"                "Musical"            "Western"
## [19] "Film-Noir"          "(no genres listed)"
```

We see 2 genres that are really not a genre definition. We investigate a number of movies without genre defined.

```
movies %>% filter(str_detect(genres,"(no genres listed)") ) %>% nrow()
```

```
## Warning: failed to assign NativeSymbolInfo for lhs since lhs is already
## defined in the 'lazyeval' namespace
```

```
## Warning: failed to assign NativeSymbolInfo for rhs since rhs is already
## defined in the 'lazyeval' namespace
```

```
## [1] 237
```

We create binary dummy variables for another 18 genres. We assign each film to the genre it belongs. We discard “IMAX” genre and assign every genres to movies without genre identified. We check the transformed result and drop `genres` column.

```
all_genres = all_genres[! all_genres %in% c("IMAX","(no genres listed)")]

for(genre in all_genres){
    movies[str_c("genre_",genre)] = ifelse(( str_detect(movies$genres,genre) | str_detec
t(movies$genres,"no genres") ) , 1 , 0)
}

#check the result
head(movies)
```

```
##   movieId                       title
## 1       1                   Toy Story
## 2       2                     Jumanji
## 3       3            Grumpier Old Men
## 4       4           Waiting to Exhale
## 5       5 Father of the Bride Part II
## 6       6                        Heat
##                                        genres year genre_Adventure
## 1 Adventure|Animation|Children|Comedy|Fantasy 1995               1
## 2                  Adventure|Children|Fantasy 1995               1
## 3                              Comedy|Romance 1995               0
## 4                        Comedy|Drama|Romance 1995               0
## 5                                      Comedy 1995               0
## 6                       Action|Crime|Thriller 1995               0
##   genre_Animation genre_Children genre_Comedy genre_Fantasy genre_Romance
## 1               1              1            1             1             0
## 2               0              1            0             1             0
## 3               0              0            1             0             1
## 4               0              0            1             0             1
## 5               0              0            1             0             0
## 6               0              0            0             0             0
##   genre_Drama genre_Action genre_Crime genre_Thriller genre_Horror
## 1           0            0           0              0            0
## 2           0            0           0              0            0
## 3           0            0           0              0            0
## 4           1            0           0              0            0
## 5           0            0           0              0            0
## 6           0            1           1              1            0
##   genre_Mystery genre_Sci-Fi genre_Documentary genre_War genre_Musical
## 1             0            0                 0         0             0
## 2             0            0                 0         0             0
## 3             0            0                 0         0             0
## 4             0            0                 0         0             0
## 5             0            0                 0         0             0
## 6             0            0                 0         0             0
##   genre_Western genre_Film-Noir
## 1             0               0
## 2             0               0
## 3             0               0
## 4             0               0
## 5             0               0
## 6             0               0
```

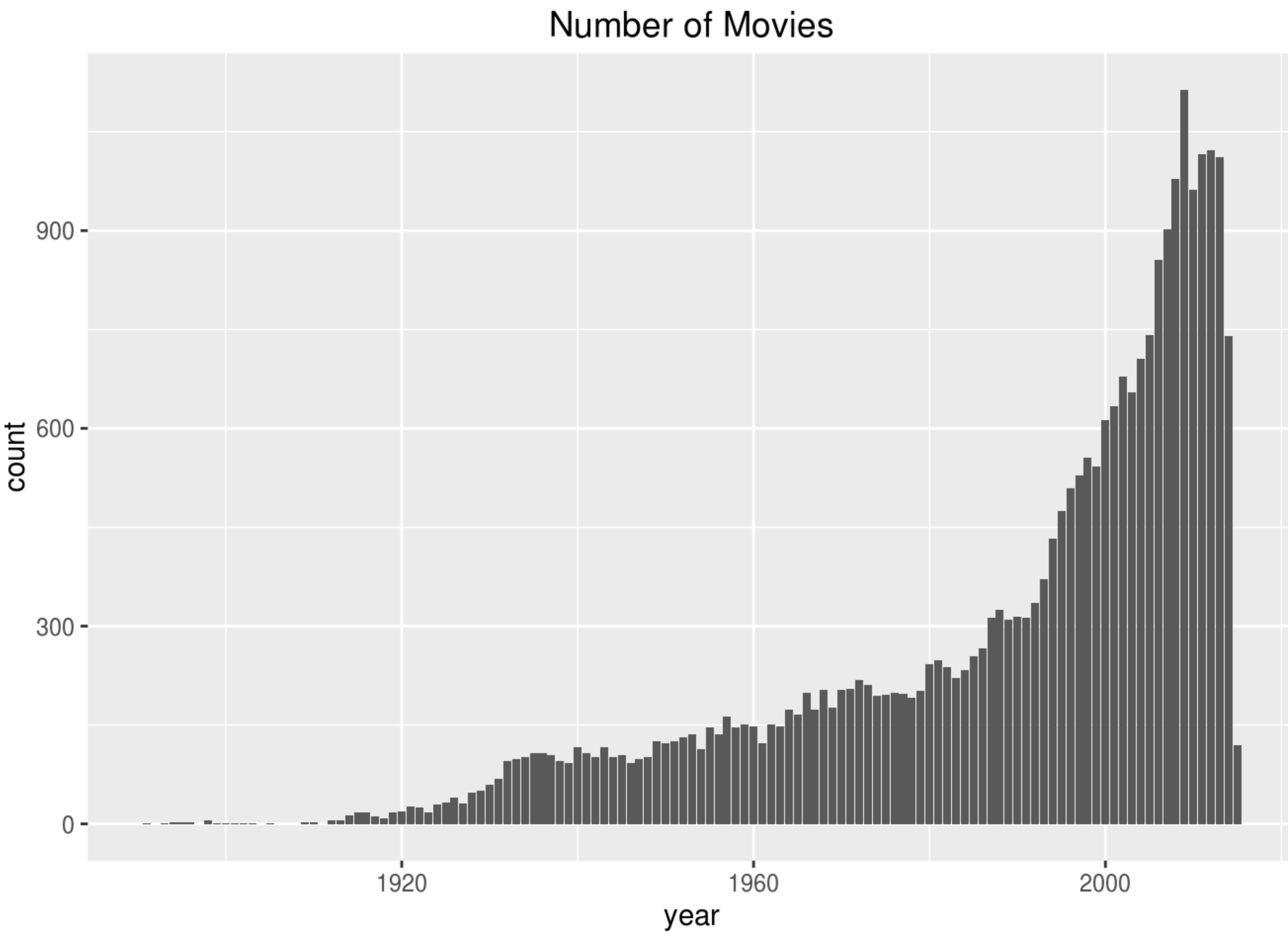
```
tail(movies)
```

```
##       movieId                                             title
## 27273  131252 Forklift Driver Klaus: The First Day on the Job
## 27274  131254                             Kein Bund für's Leben
## 27275  131256                            Feuer, Eis & Dosenbier
## 27276  131258                                       The Pirates
## 27277  131260                                      Rentun Ruusu
## 27278  131262                                         Innocence
##                           genres year genre_Adventure genre_Animation
## 27273              Comedy|Horror 2001               0               0
## 27274                     Comedy 2007               0               0
## 27275                     Comedy 2002               0               0
## 27276                  Adventure 2014               1               0
## 27277         (no genres listed) 2001               1               1
## 27278 Adventure|Fantasy|Horror 2014               1               0
##       genre_Children genre_Comedy genre_Fantasy genre_Romance genre_Drama
## 27273              0            1             0             0           0
## 27274              0            1             0             0           0
## 27275              0            1             0             0           0
## 27276              0            0             0             0           0
## 27277              1            1             1             1           1
## 27278              0            0             1             0           0
##       genre_Action genre_Crime genre_Thriller genre_Horror genre_Mystery
## 27273            0           0              0            1             0
## 27274            0           0              0            0             0
## 27275            0           0              0            0             0
## 27276            0           0              0            0             0
## 27277            1           1              1            1             1
## 27278            0           0              0            1             0
##       genre_Sci-Fi genre_Documentary genre_War genre_Musical genre_Western
## 27273            0                 0         0             0             0
## 27274            0                 0         0             0             0
## 27275            0                 0         0             0             0
## 27276            0                 0         0             0             0
## 27277            1                 1         1             1             1
## 27278            0                 0         0             0             0
##       genre_Film-Noir
## 27273               0
## 27274               0
## 27275               0
## 27276               0
## 27277               1
## 27278               0
```

```
movies$genres = NULL
```

We explore a number of movies for each year in the dataset that we have

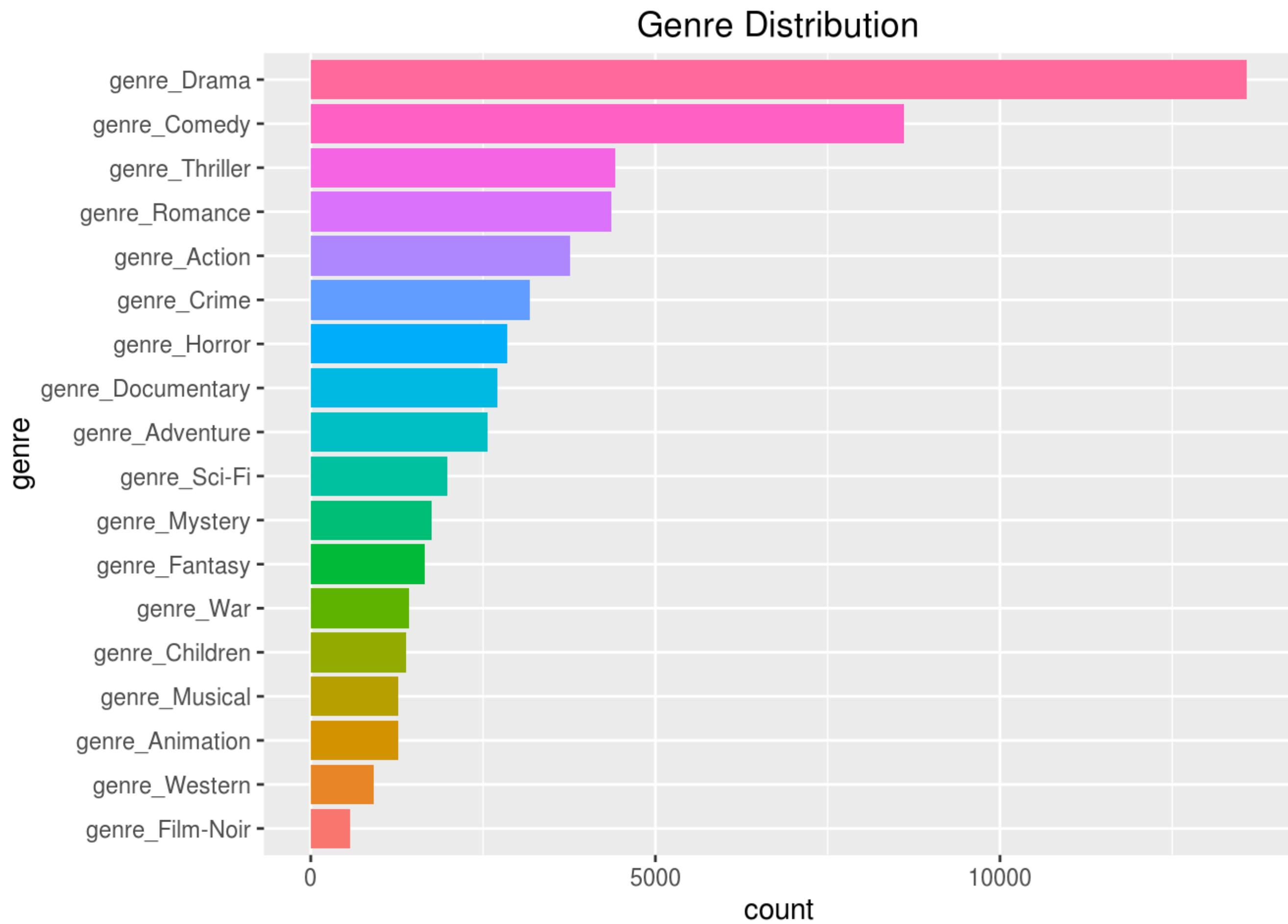
```
ggplot(movies,aes(x=year)) + geom_bar() + ggtitle("Number of Movies")
```



We also explore a distributon of each movie genres

```
genre_dist = colSums(movies[,4:21])
genre_dist_df = data.frame(genre = names(genre_dist),count = genre_dist)
genre_dist_df$genre = factor(genre_dist_df$genre,levels = names(sort(genre_dist,decreasi
ng = FALSE)))

ggplot(genre_dist_df,aes(x=genre,y=count,fill=genre)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    ggtitle("Genre Distribution") +
    theme(legend.position = "none")
```



Now, we get a basic understanding of our movie dataset. Genre and year that we extracted will served as a filter that users can use to narrow down their interest.

Construct Association Rules from Rating Data

We proceed to read ratings.csv and investigate the dataset. We skip reading `rating` and `timestamp` columns. Noted that we ignore the actual rating here as we put more focus on the fact that the scored movies hold some interesting quality that they at least led the viewers to view them.

```
ratings = read.csv("ml-20m/ratings.csv",
                   colClasses = c("integer","integer","NULL","NULL"),
                   sep=",",
                   stringsAsFactors = FALSE)

head(ratings)
```

```
##   userId movieId
## 1      1       2
## 2      1      29
## 3      1      32
## 4      1      47
## 5      1      50
## 6      1     112
```

We discard ratings that contain id in `discard_movie_id`

```
ratings = ratings %>% filter(! movieId %in% discard_movie_id )
```

```
## Warning: failed to assign NativeSymbolInfo for lhs since lhs is already
## defined in the 'lazyeval' namespace
```

```
## Warning: failed to assign NativeSymbolInfo for rhs since rhs is already
## defined in the 'lazyeval' namespace
```

We look at a total number of ratings left

```
dim(ratings)[1]
```

```
## [1] 19999575
```

We use `arules` package to perform the frequent itemset mining with Apriori algorithm. We construct User-Item matrix with binary values; 0 - a movie isn't seen by a user, and 1 - it is seen. The package use a sparse matrix object, `transactions` , to represent User-Item matrix. This prevents our computing machine from consuming all available RAM as most elements in the matrix will be zero.

```
#convert rating-per-row dataframe into sparse User-Item matrix
user_item_matrix <- as(split(ratings[,"movieId"], ratings[,"userId"]), "transactions")

#investigate the User-Item matrix
#transactions (rows) -> number of raters
#items (columns) -> number of movies
user_item_matrix
```

```
## transactions in sparse format with
##  138493 transactions (rows) and
##  26736 items (columns)
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  1638201  87.5   10693453  571.1  20885653 1115.5
## Vcells 29092943 222.0  132642697 1012.0 165348894 1261.6
```

Next, we mine for a frequent pair of movies that raters watched. We hypothesize that if movie A and B are frequently viewed together, there should be some underlying relationships between them that incite viewer's curiosity. We can use such finding to recommend movie B to a user if he/she already saw A(or vice versa).

We set the support threshold to 0.001(the pair is watched together by at least 139 raters) and the minimum confidence(the likelihood that if user watched movie A, he/she will also watch movie B) to 70%.

```
rule_param = list(
    supp = 0.001,
    conf = 0.7,
    maxlen = 2
)
```

We run Apriori based on the specified rule

```
assoc_rules = apriori(user_item_matrix,parameter = rule_param)
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.7    0.1    1 none FALSE            TRUE   0.001      1      2
##  target   ext
##   rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 138
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[26736 item(s), 138493 transaction(s)] done [3.71s].
## sorting and recoding items ... [7691 item(s)] done [0.60s].
## creating transaction tree ... done [0.13s].
## checking subsets of size 1 2 done [10.60s].
## writing ... [189611 rule(s)] done [0.19s].
## creating S4 object  ... done [0.06s].
```

We summarize the association rule

```
summary(assoc_rules)
```

```
## set of 189611 rules
##
## rule length distribution (lhs + rhs):sizes
##      2
## 189611
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2       2       2       2       2       2
##
## summary of quality measures:
##     support           confidence          lift
##  Min.   :0.001004   Min.   :0.7000   Min.   :  1.440
##  1st Qu.:0.001516   1st Qu.:0.7240   1st Qu.:  2.246
##  Median :0.002520   Median :0.7531   Median :  3.057
##  Mean   :0.006663   Mean   :0.7637   Mean   :  3.929
##  3rd Qu.:0.005719   3rd Qu.:0.7941   3rd Qu.:  4.323
##  Max.   :0.344516   Max.   :0.9700   Max.   :663.359
##
## mining info:
##              data ntransactions support confidence
##  user_item_matrix        138493   0.001        0.7
```

We constructed 189611 rules here. We also get summary statistics of “lift” for all rules. Lift is used to measure how the rule “if an user watched A then he will proceed to watch B” performs against chance. For example, if movie B is watched by every users, then the rule A => B will have 100% confidence but this rule will not be really interesting as there is no point to recommend it because everyone tend to watch it anyway. We can use lift to filter the “interestingness” of each rule. Lift equal 1 suggests that A and B are independent. The higher the number, the more they related.

With such huge number of rules, we filter only those that have lift exceed their 75% percentile(4.323).

```
assoc_rules = subset(assoc_rules, lift >= 4.323)

summary(assoc_rules)
```

```
## set of 47399 rules
##
## rule length distribution (lhs + rhs):sizes
##     2
## 47399
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2       2       2       2       2       2
##
## summary of quality measures:
##     support           confidence          lift
##  Min.   :0.001004   Min.   :0.7000   Min.   :  4.323
##  1st Qu.:0.001329   1st Qu.:0.7208   1st Qu.:  4.777
##  Median :0.001935   Median :0.7463   Median :  5.592
##  Mean   :0.003395   Mean   :0.7565   Mean   :  7.650
##  3rd Qu.:0.003488   3rd Qu.:0.7824   3rd Qu.:  7.586
##  Max.   :0.121017   Max.   :0.9700   Max.   :663.359
##
## mining info:
##              data ntransactions support confidence
##  user_item_matrix        138493   0.001        0.7
```

We cast `assoc_rules` to data.frame and look at some of the data

```
assoc_rules = as(assoc_rules,"data.frame")

head(assoc_rules)
```

```
##                  rules     support confidence     lift
## 1      {834} => {788} 0.001039764  0.7093596 5.225881
## 9       {732} => {95} 0.001249161  0.7393162 4.632199
## 30   {8485} => {4973} 0.001032543  0.8827160 5.020740
## 33 {73759} => {58559} 0.001090308  0.8531073 5.780869
## 37      {706} => {95} 0.001321366  0.7290837 4.568087
## 38     {706} => {788} 0.001379131  0.7609562 5.605995
```

The rules still contain `movieId` . We split movies in both sides to a new column

```
rules = sapply(assoc_rules$rules,function(x){
    x = gsub("[\\{\\}]", "", regmatches(x, gregexpr("\\{.*\\}", x))[[1]])
    x = gsub("=>",",",x)
    x = str_replace_all(x," ","")
    return( x )
})

rules = as.character(rules)
rules = str_split(rules,",")

assoc_rules$lhs_movie = sapply( rules, "[[", 1)
assoc_rules$rhs_movie = sapply( rules , "[[", 2)

assoc_rules$rules = NULL
rm(rules)
gc()
```

```
##            used  (Mb) gc trigger  (Mb)  max used   (Mb)
## Ncells  1652052  88.3    8554762 456.9  20885653 1115.5
## Vcells 29377707 224.2  106114157 809.6 165348894 1261.6
```

```
assoc_rules$lhs_movie = as.numeric(assoc_rules$lhs_movie)
assoc_rules$rhs_movie = as.numeric(assoc_rules$rhs_movie)
```

We join `assoc_rules` with `movies` to get titles on the left hand side and right hand side of the rule, and also their corresponding genres and released year.

```
assoc_rules = assoc_rules %>% left_join(movies,by=c("lhs_movie" = "movieId") )

assoc_rules$lhs_movie = NULL
col_name = colnames(assoc_rules)
col_name[5:24] = str_c("left.",col_name[5:24])
colnames(assoc_rules) = col_name

assoc_rules = assoc_rules %>% left_join(movies,by=c("rhs_movie" = "movieId"))
assoc_rules$rhs_movie = NULL
col_name = colnames(assoc_rules)
col_name[24:43] = str_c("right.",col_name[24:43])
colnames(assoc_rules) = col_name
```

```
##            used  (Mb) gc trigger  (Mb)  max used   (Mb)
## Ncells  1658466  88.6    6843809 365.5  20885653 1115.5
## Vcells 31174389 237.9   84891325 647.7 165348894 1261.6
```

Mining the Relationship and Recommending Movies

Now, we can look at the rules we mined. For example, we can look at top rules with highest lift.

```
assoc_rules %>% arrange(desc(lift)) %>% select(left.title,left.year,right.title,right.ye
ar,support,confidence,lift) %>% head()
```

```
##                            left.title left.year
## 1            Nymphomaniac: Volume I      2013
## 2           Nymphomaniac: Volume II      2013
## 3                  Faces of Death 3      1985
## 4                  Faces of Death 2      1981
## 5 Puppet Master 5: The Final Chapter      1994
## 6                   Puppet Master 4      1993
##                           right.title right.year     support confidence
## 1           Nymphomaniac: Volume II       2013 0.001061425  0.7424242
## 2            Nymphomaniac: Volume I       2013 0.001061425  0.9483871
## 3                  Faces of Death 2       1981 0.001068646  0.7668394
## 4                  Faces of Death 3       1985 0.001068646  0.7437186
## 5                   Puppet Master 4       1993 0.001263602  0.8215962
## 6 Puppet Master 5: The Final Chapter       1994 0.001263602  0.7882883
##       lift
## 1 663.3585
## 2 663.3585
## 3 533.6778
## 4 533.6778
## 5 512.5465
## 6 512.5465
```

For the top rules, we discover sequel/prequel relationship between the movies. We would like to find recommendations that have not-so-obvious relationship instead.

We can filter out results that have sequel-prequel relationship based on their similar titles. We do a naive filter here. Result with number on both sides or similar opening string is removed, we also exclude the “Thin man” serie.

```
assoc_rules = assoc_rules %>%
    filter( ! (grepl("[0-9]",left.title,perl = TRUE) &  grepl("[0-9]",right.title,perl =
 TRUE) ) ) %>%
    filter( ! (grepl("Thin Man",left.title,perl = TRUE) &  grepl("Thin
Man",right.title,perl = TRUE) ) ) %>%
    filter( substr( left.title,start = 1,stop =
min(5,str_length(left.title),str_length(right.title)) ) != substr( right.title,start =
1,stop = min(5,str_length(left.title),str_length(right.title)) ) ) %>%
    arrange(desc(lift))

head(assoc_rules %>% select(left.title,left.year,right.title,right.year,support,confiden
ce,lift),10)
```

```
##                          left.title left.year
## 1                     7 Plus Seven      1970
## 2                    In Like Flint      1967
## 3    Unvanquished, The (Aparajito)      1957
## 4    Unvanquished, The (Aparajito)      1957
## 5                        Seven Up!      1964
## 6  Frankenstein Meets the Wolf Man      1943
## 7                   Cocoanuts, The      1929
## 8                 House of Dracula      1945
## 9                          Tenebre      1982
## 10                    Pat and Mike      1952
##                                     right.title right.year     support
## 1                                     Seven Up!       1964 0.001350249
## 2                                 Our Man Flint       1965 0.001884572
## 3  Song of the Little Road (Pather Panchali)       1955 0.001927895
## 4             World of Apu, The (Apur Sansar)       1959 0.001769042
## 5                                         28 Up       1985 0.001834028
## 6                                 Wolf Man, The       1941 0.002130072
## 7                               Animal Crackers       1930 0.001249161
## 8                                 Wolf Man, The       1941 0.001046984
## 9                                      Suspiria       1977 0.001184175
## 10                                   Adam's Rib       1949 0.001487440
##    confidence     lift
## 1   0.7663934 295.6549
## 2   0.7331461 225.1344
## 3   0.7899408 143.7599
## 4   0.7248521 131.2248
## 5   0.7075209 129.6120
## 6   0.7195122 113.1072
## 7   0.7393162 111.0522
## 8   0.7004831 110.1158
## 9   0.7224670 109.9523
## 10  0.7803030 108.3917
```

Thre are many ideas that we can throw into the association rules. For example, we would like to look at modern movies that led users to view the older film.

```
assoc_rules %>%
    filter(left.year > 2000 & right.year < 1990) %>%
    arrange(desc(lift)) %>%
    select(left.title,left.year,right.title,right.year,support,confidence,lift) %>%
    head(20)
```

```
##                            left.title left.year
## 1  Cat Returns, The (Neko no ongaeshi)      2002
## 2    Tekkonkinkreet (Tekkon kinkurîto)      2006
## 3          Ponyo (Gake no ue no Ponyo)      2008
## 4                               Undead      2003
## 5                 Steamboy (Suchîmubôi)      2004
## 6                               Undead      2003
## 7                             Casshern      2004
## 8                        Inland Empire      2006
## 9                                Below      2002
## 10                              Undead      2003
## 11                   Home on the Range      2004
## 12                 Return to Never Land      2002
## 13                  Returner (Ritaanaa)      2002
## 14                           Dark Blue      2003
## 15                            Impostor      2002
## 16        Decade Under the Influence, A      2003
## 17                       Sunshine State      2002
## 18     I Am Trying to Break Your Heart      2002
## 19                     Hollywood Ending      2002
## 20                           Dark Blue      2003
##                             right.title right.year     support confidence
## 1  My Neighbor Totoro (Tonari no Totoro)       1988 0.005177157  0.8156997
## 2  My Neighbor Totoro (Tonari no Totoro)       1988 0.001270822  0.7242798
## 3  My Neighbor Totoro (Tonari no Totoro)       1988 0.007466081  0.7160665
## 4            Evil Dead II (Dead by Dawn)       1987 0.001083087  0.7109005
## 5                                  Akira       1988 0.003220379  0.7228525
## 6                             Thing, The       1982 0.001104749  0.7251185
## 7                                  Akira       1988 0.001234719  0.7037037
## 8                            Blue Velvet       1986 0.003682497  0.7254623
## 9                               Predator       1987 0.001631851  0.7361564
## 10                              Predator       1987 0.001119190  0.7345972
## 11                    Little Mermaid, The       1989 0.001472999  0.7208481
## 12                    Little Mermaid, The       1989 0.001819587  0.7179487
## 13                              Predator       1987 0.001552425  0.7026144
## 14                              Predator       1987 0.002180616  0.7006961
## 15                              Predator       1987 0.003155394  0.7003205
## 16                             Chinatown       1974 0.001364690  0.7325581
## 17                            Annie Hall       1977 0.002491101  0.7263158
## 18                     This Is Spinal Tap       1984 0.001552425  0.7570423
## 19                            Annie Hall       1977 0.002779924  0.7116451
## 20                      Untouchables, The       1987 0.002195057  0.7053364
##         lift
## 1  20.580924
## 2  18.274310
## 3  18.067079
## 4  12.641851
## 5  12.026671
## 6  11.735869
## 7  11.708077
## 8  11.320727
## 9   7.013311
## 10  6.998457
```

```
## 11  6.984218
## 12  6.956126
## 13  6.693759
## 14  6.675483
## 15  6.671905
## 16  6.626661
## 17  6.539014
## 18  6.530774
## 19  6.406934
## 20  6.325875
```

Many Ghibli's films and japanese animations appear here. It looks like modern japanese animations have enough power to draw viewers into their own world. In contrast, only few Disney animations top the chart, which can be because they are watched by nearly everyone, which resulted in lower lift scores. We are quite surprised to see Home on the Range led viewers back to The Little Mermaid. Critic reception for the film is quite low. May be that reminded viwers of Disney's renaissance era? Another notable exception is Inland Empire and Blue Velvet, which "Lynchian" structure in both films is discovered.

We can also incorporate movie's genres. We calculate the number of common genres among two films.

```
assoc_rules$common_genre = apply(assoc_rules,1,function(x){
                             sum(as.numeric(x[6:23]) & as.numeric(x[26:43]))
                             })
```

Then, we mine for a movie that led viewers to a totally different kind of movie(common_genre = 0). We prefer modern films which span across different years.

```
assoc_rules %>% filter(common_genre == 0) %>%
    filter( abs(left.year - right.year) >= 5 & left.year > 2000 & right.year > 2000) %>%
    select(left.title,left.year,right.title,right.year,support,confidence,lift) %>%
    head(20)
```

```
##                          left.title left.year           right.title
## 1                    Thirst (Bakjwi)      2009               Old Boy
## 2                         This Is 40      2012              Superbad
## 3                         This Is 40      2012      Wedding Crashers
## 4                               Noah      2014            District 9
## 5                       Imposter, The      2012       Children of Men
## 6                      Veronica Mars      2014                Avatar
## 7                The Raid 2: Berandal      2014                    Up
## 8                          Guard, The      2011       Children of Men
## 9                     Under the Skin      2013 No Country for Old Men
## 10 Stanley Kubrick: A Life in Pictures      2001       Children of Men
## 11                    Predestination      2014   Inglourious Basterds
## 12          Ricky Gervais Live: Animals      2003   Inglourious Basterds
## 13  Million Ways to Die in the West, A      2014                Avatar
## 14 Stanley Kubrick: A Life in Pictures      2001 No Country for Old Men
## 15            The Amazing Spider-Man 2      2014                    Up
## 16                               Chef      2014    Slumdog Millionaire
## 17                     Upstream Color      2013 No Country for Old Men
## 18         Evening with Kevin Smith, An      2002   Inglourious Basterds
## 19  Million Ways to Die in the West, A      2014   Inglourious Basterds
## 20                      Imposter, The      2012 No Country for Old Men
##    right.year     support confidence     lift
## 1        2003 0.001559646  0.8089888 18.05047
## 2        2007 0.001343028  0.7717842 16.02499
## 3        2005 0.001263602  0.7261411 14.17213
## 4        2009 0.001812366  0.7150997 12.32101
## 5        2006 0.001090308  0.7365854 11.37383
## 6        2009 0.001090308  0.7989418 11.34501
## 7        2009 0.001133631  0.7302326 10.91668
## 8        2006 0.002216719  0.7041284 10.87266
## 9        2007 0.001393572  0.8041667 10.86763
## 10       2006 0.001068646  0.7014218 10.83086
## 11       2009 0.001689616  0.7358491 10.78755
## 12       2009 0.001220278  0.7284483 10.67905
## 13       2009 0.001068646  0.7512690 10.66805
## 14       2007 0.001198617  0.7867299 10.63198
## 15       2009 0.002570527  0.7063492 10.55963
## 16       2008 0.001783484  0.7017045 10.55399
## 17       2007 0.001213058  0.7777778 10.51100
## 18       2009 0.001494660  0.7113402 10.42825
## 19       2009 0.001003661  0.7055838 10.34386
## 20       2007 0.001119190  0.7560976 10.21802
```

The top rule consisted of both Korean movies. Thirst, which led to our favorite film: Old Boy, is a film that we have never seen before but its synopsis does sound really interesting to us! This displays the case where we may need to consider a rule on both direction as well.

Lastly, we can use assocation rules to recommend a potential movie. Let the Right One In is our favorite film and we would like to explore further movies based on it.

```
assoc_rules %>%
    filter(str_detect(left.title,"Let the Right One In") | str_detect(right.title,"Let t
he Right One In")) %>%
    select(left.title,left.year,right.title,right.year,support,confidence,lift) %>%
    head(20)
```

```
##                                       left.title left.year
## 1                                Thirst (Bakjwi)      2009
## 2 Let the Right One In (Låt den rätte komma in)      2008
## 3 Let the Right One In (Låt den rätte komma in)      2008
## 4 Let the Right One In (Låt den rätte komma in)      2008
##                                      right.title right.year     support
## 1 Let the Right One In (Låt den rätte komma in)       2008 0.001509102
## 2                               Dark Knight, The       2008 0.017314955
## 3                                   Donnie Darko       2001 0.015697544
## 4         Eternal Sunshine of the Spotless Mind       2004 0.016210206
##   confidence      lift
## 1  0.7827715 35.660651
## 2  0.7888158  5.345213
## 3  0.7151316  5.287530
## 4  0.7384868  4.575665
```

Thirst(again) appeared here. It's interesting to note that both films contain vampirism element, have similar theme(as we guessed from the synopsis), and are not well known, which is reflected in the high lift score. The other three movies are significantly more popular. Their rules are not very interesting since many viewers also watch them anyway, regardless of the influence movie(reflected in the considerably lower lift scores). Donnie Darko, a surreal and mind-bending film, does make a bit surprise, as we didn't expect it to be heard of by so many viewers, but this perhap reflects an enthusiasm(and bias) in the film rating communities. Also noted that the number of movies that we can recommend depended on the cut off support value that we set. If we set this value to be too high, we will not be able to suggest anything.

Conclusion

We apply a traditional Market Basket Analysis technique to a film recommendation setting. The technique does not provide a recommendation in a fine-grained user level, as it can be typically done by Collaborative Filtering, but it does enable us to investigate an underlying relationship within the movies. We can utilized such finding to construct a new marketing campaign, research customer's behavior, or make a product suggestion. The mining technique can also be deployed in many problem contexts, provided that they can be formulated by Basket-Item scenario.