Association Rules

Taylor Shrode

November 22, 2020

Introduction

Association rules help uncover relationships between items from huge databases (Market basket analysis using r, 2018). Applications of association rules are found in marketing, market basket analysis in retailing, clustering, and classification. (Market basket analysis using r, 2018). These rules can tell us what items customers frequently buy together and then can be used for numerous marketing strategies such as (Market basket analysis using r, 2018):

1. Changing the store layout according to trends
2. Customer behavior analysis
3. Identifying trending items customers buy

Association rules give an output as rules in the form **if this then that** (Market basket analysis using r, 2018). When mining for association rules, the unit of "togetherness" is called a *transaction* (sometimes referred to as *baskets*) (Zumel, 2019). The transaction could be a single shopping basket, or a single user session on a website. A transaction depends on the problem. The objects that comprise a transaction are called *items* in an *itemset* (Zumel, 2019). These can be the products in a shopping basket, or the pages visited during a website session.

To create the association rules, we will be using a publicly available *Book-Crossing Dataset* as our data (WinVector, n.d.). This dataset consists of the following three logs (Book-crossing dataset, n.d.):

1. **BX-Users**: Contains the users. Note that user IDs (**User-ID**) have been anonymized and mapped to integers. Demographic data is provided (**Location**, **Age**) if available. Otherwise, these fields contain NULL-values.
2. **BX-Books**: Books are identified by their respective ISBN. Invalid ISBNs have already been removed from the dataset. Moreover, some content-based information is given (**Book-Title**, **Book-Author**, **Year-Of-Publication**, **Publisher**), obtained from Amazon Web Services. Note that in case of several authors, only the first is provided. URLs linking to cover images are also given, appearing in three different flavors (**Image-URL-S**, **Image-URL-M**, **Image-URL-L**), i.e., small, medium, large. These URLs point to the Amazon web site.

3. **BX-Book-Ratings**: Contains the book rating information. Ratings (**Book-Rating**) are either explicit, expressed on a scale from 1-10 (higher values denoting higher appreciation), or implicit, expressed by 0.

We will need to transform our data into a "transaction" format, so that the website visited is the transaction and the title of book that has been rated are the itemset that comprise the transaction. In other words, we want our data to consist of **User-ID** and **Book-Title**.

Before we proceed, we need to define rules, and various metrics that help us understand the strength of the rule. First, a *rule* ("if X then Y") means that every time we see the itemset X in a transaction, we expect to see Y, with a given confidence. When we use the *apriori algorithm*, Y is always an itemset with one item (Zumel, 2019). Next, suppose we have a database of transactions T and the itemset X. Then the *support* of X is the number of transactions that contain X divided by the total number of transactions in T (Zumel, 2019). The *confidence* of the rule "if X, then Y" gives the fraction or percentage of the time that a rule is true, relative to how often you see X (Zumel, 2019). In other words, if $support(X)$ is how often the itemset X occurs in a transaction, and $support(X, Y)$ is how often both itemsets X and Y occur in a transaction, then the confidence of the rule is

$$Confidence = \frac{support(\{X, Y\})}{support(X)}.$$

Finally, *lift* compares the frequency of an observed pattern with how often you would expect to see that pattern by chance (Zumel, 2019). The lift of a rule "if X then Y" is given by

$$Lift = \frac{support(\{X, Y\})}{support(X) * support(Y)}$$

If the lift is close to 1, then there is a good chance that the observed pattern occurred by chance. The larger the lift is, the more likely that the pattern is "real" (Zumel, 2019).

The goal of association rule mining is to find all the interesting rules in the database, with at least a given minimum support and a minimum given confidence. For this assignment, we will begin with a minimum support of 0.005, and a minimum confidence of 0.70.

Load Libraries

Before we begin, we need to load the necessary libraries.

```r
library(sqldf)
library(DataExplorer)
library(dplyr)
library(ggplot2)
library(arules)
library(RColorBrewer)
library(arulesViz)
```

The **sqldf** allows us to manipulate our dataframe using SQL statements. The **DataExplorer** package allows us to view how complete our data is. The **dplyr** package is loaded to perform various data wrangling techniques. The **ggplot2** is loaded to create various plots and the **RColorBrewer** plot is loaded to manipulate the colors of the plots. Finally, the **arules** package allows us to generate the association rules and the **arulesViz** package allows us to display the rules.

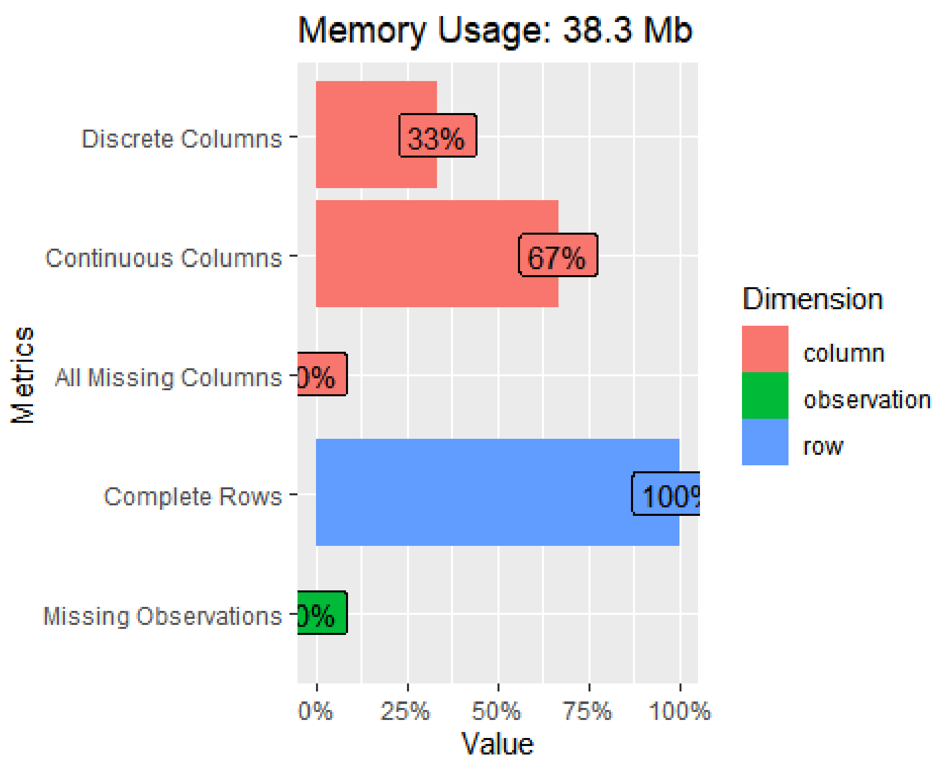
Load Data Set

Now, we can load our dataset by specifying the path to our dataset and then loading the dataset.

```
load(
  "C:/Users/07hoc/OneDrive/Documents/Data Science Documents/bxBooks.RData"
  )
```

First, we will view our **bxBookRatings** dataset by viewing the first 5 entries, viewing the structure of our data, and determing the “complete-ness” of the data.

```
head(bxBookRatings) #view bxBookRatings: contains UserID (int), ISBN (chr),
and BookRating (int)

##   User.ID       ISBN Book.Rating
## 1  276725 034545104X           0
## 2  276726 0155061224           5
## 3  276727 0446520802           0
## 4  276729 052165615X           3
## 5  276729 0521795028           6
## 6  276733 2080674722           0

str(bxBookRatings)

## 'data.frame':    1149780 obs. of  3 variables:
##  $ User.ID    : int  276725 276726 276727 276729 276729 276733 276736
276737 276744 276745 ...
##  $ ISBN       : chr  "034545104X" "0155061224" "0446520802" "052165615X"
...
##  $ Book.Rating: int  0 5 0 3 6 0 8 6 7 10 ...

plot_intro(bxBookRatings)
```



The output above indicates that our dataframe contains 1,149,870 observations, 3 variables, and does not contain any NA values. Now, we apply the same functions to our **bxBooks** dataset.

```
head(bxBooks) #view bxBooks: contains ISBN (chr), BookTitle (chr), BookAuthor
(chr), YearOfPublication (int), Publisher (chr), ImagesURLsizeS (chr),
ImagesURLsizeM (chr), ImagesURLsizeL  (chr)

##         ISBN
## 1 0195153448
## 2 0002005018
## 3 0060973129
## 4 0374157065
## 5 0393045218
## 6 0399135782
##
Book.Title
## 1
Classical Mythology
## 2
Clara Callan
## 3
Decision in Normandy
## 4 Flu: The Story of the Great Influenza Pandemic of 1918 and the Search
for the Virus That Caused It
## 5
The Mummies of Urumchi
```

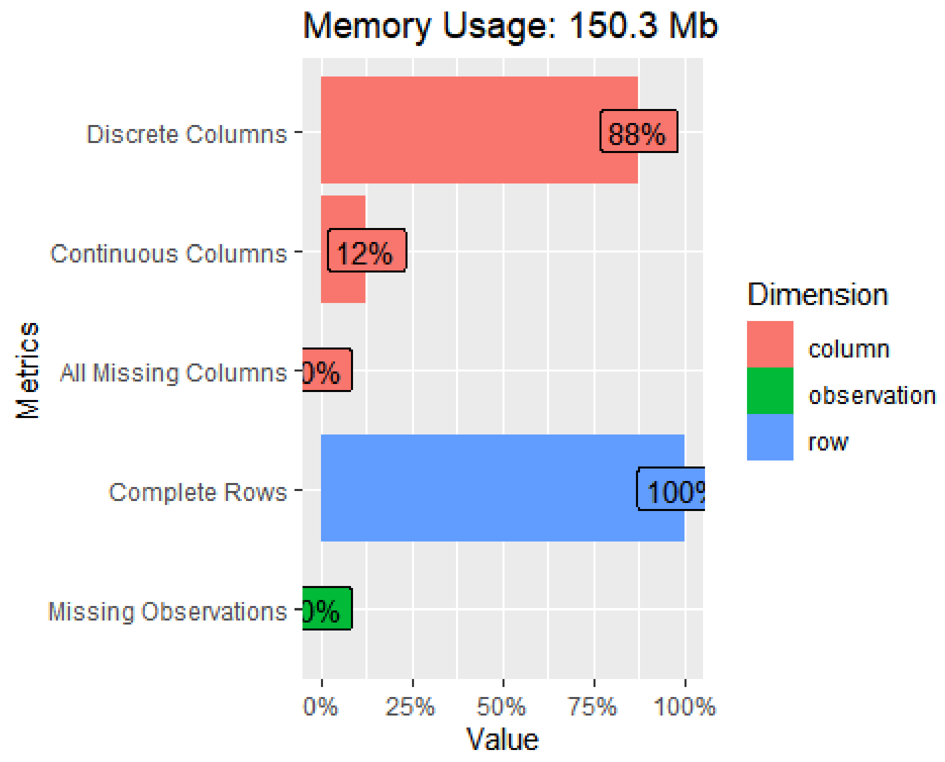
```
## 6
The Kitchen God's Wife
##            Book.Author Year.Of.Publication                   Publisher
## 1   Mark P. O. Morford                2002    Oxford University Press
## 2 Richard Bruce Wright                2001      HarperFlamingo Canada
## 3          Carlo D'Este                1991            HarperPerennial
## 4     Gina Bari Kolata                1999       Farrar Straus Giroux
## 5      E. J. W. Barber                1999 W. W. Norton &amp; Company
## 6              Amy Tan                1991           Putnam Pub Group
##                                                   Image.URL.S
## 1 http://images.amazon.com/images/P/0195153448.01.THUMBZZZ.jpg
## 2 http://images.amazon.com/images/P/0002005018.01.THUMBZZZ.jpg
## 3 http://images.amazon.com/images/P/0060973129.01.THUMBZZZ.jpg
## 4 http://images.amazon.com/images/P/0374157065.01.THUMBZZZ.jpg
## 5 http://images.amazon.com/images/P/0393045218.01.THUMBZZZ.jpg
## 6 http://images.amazon.com/images/P/0399135782.01.THUMBZZZ.jpg
##                                                   Image.URL.M
## 1 http://images.amazon.com/images/P/0195153448.01.MZZZZZZZ.jpg
## 2 http://images.amazon.com/images/P/0002005018.01.MZZZZZZZ.jpg
## 3 http://images.amazon.com/images/P/0060973129.01.MZZZZZZZ.jpg
## 4 http://images.amazon.com/images/P/0374157065.01.MZZZZZZZ.jpg
## 5 http://images.amazon.com/images/P/0393045218.01.MZZZZZZZ.jpg
## 6 http://images.amazon.com/images/P/0399135782.01.MZZZZZZZ.jpg
##                                                   Image.URL.L
## 1 http://images.amazon.com/images/P/0195153448.01.LZZZZZZZ.jpg
## 2 http://images.amazon.com/images/P/0002005018.01.LZZZZZZZ.jpg
## 3 http://images.amazon.com/images/P/0060973129.01.LZZZZZZZ.jpg
## 4 http://images.amazon.com/images/P/0374157065.01.LZZZZZZZ.jpg
## 5 http://images.amazon.com/images/P/0393045218.01.LZZZZZZZ.jpg
## 6 http://images.amazon.com/images/P/0399135782.01.LZZZZZZZ.jpg
```

```
str(bxBooks) #can remove ImagesURLS, M, L
```

```
## 'data.frame':    271379 obs. of  8 variables:
##  $ ISBN               : chr  "0195153448" "0002005018" "0060973129"
"0374157065" ...
##  $ Book.Title         : chr  "Classical Mythology" "Clara Callan"
"Decision in Normandy" "Flu: The Story of the Great Influenza Pandemic of
1918 and the Search for the Virus That Caused It" ...
##  $ Book.Author        : chr  "Mark P. O. Morford" "Richard Bruce Wright"
"Carlo D'Este" "Gina Bari Kolata" ...
##  $ Year.Of.Publication: int  2002 2001 1991 1999 1999 1991 2000 1993 1996
2002 ...
##  $ Publisher          : chr  "Oxford University Press" "HarperFlamingo
Canada" "HarperPerennial" "Farrar Straus Giroux" ...
##  $ Image.URL.S        : chr
"http://images.amazon.com/images/P/0195153448.01.THUMBZZZ.jpg"
"http://images.amazon.com/images/P/0002005018.01.THUMBZZZ.jpg"
"http://images.amazon.com/images/P/0060973129.01.THUMBZZZ.jpg"
"http://images.amazon.com/images/P/0374157065.01.THUMBZZZ.jpg" ...
```

```
##  $ Image.URL.M        : chr
"http://images.amazon.com/images/P/0195153448.01.MZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0002005018.01.MZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0060973129.01.MZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0374157065.01.MZZZZZZZ.jpg" ...
##  $ Image.URL.L        : chr
"http://images.amazon.com/images/P/0195153448.01.LZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0002005018.01.LZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0060973129.01.LZZZZZZZ.jpg"
"http://images.amazon.com/images/P/0374157065.01.LZZZZZZZ.jpg" ...
```

```
plot_intro(bxBooks)
```

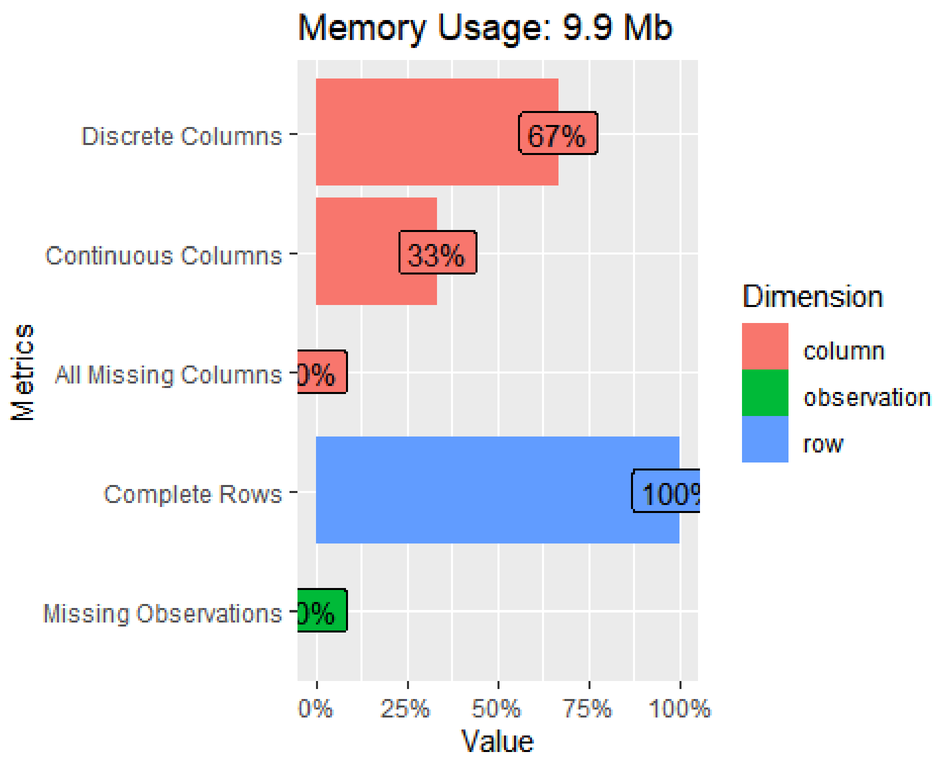


The output above indicates that our dataframe contains 271,379 observations, 8 variables, and does not contain any NA values. Finally, we view our **bxUsers** dataset.

```
head(bxUsers) #view bxUsers: contains UserID (int), Location (chr), Age (chr)

##   User.ID                             Location  Age
## 1       1                   nyc, new york, usa NULL
## 2       2            stockton, california, usa   18
## 3       3      moscow, yukon territory, russia NULL
## 4       4            porto, v.n.gaia, portugal   17
## 5       5 farnborough, hants, united kingdom NULL
## 6       6       santa monica, california, usa   61

str(bxUsers)
```

```
## 'data.frame':	278858 obs. of  3 variables:
##  $ User.ID : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Location: chr  "nyc, new york, usa" "stockton, california, usa"
"moscow, yukon territory, russia" "porto, v.n.gaia, portugal" ...
##  $ Age     : chr  "NULL" "18" "NULL" "17" ...
```

```
plot_intro(bxUsers)
```



Finally, the output above indicates that our dataframe contains 278,858 observations, 3 variables, and does not contain any NA values.

Clean Data

Now, we can continue with cleaning our data, which consists of removing periods from the column names, removing special characters, and changing our book titles to lowercase. First, we will remove the periods from our column names.

```
#remove period from colnames
colnames(bxBookRatings) <- gsub(".", "_", colnames(bxBookRatings), fixed=T)
colnames(bxBooks) <- gsub(".", "_", colnames(bxBooks), fixed=T)
colnames(bxUsers) <- gsub(".", "_", colnames(bxUsers), fixed=T)
```

Next, we need to remove the **Image-URL-S**, **Image-URL-M**, and **Image-URL-L** columns.

```
bxBooks_subset <- bxBooks[,-c(6:8)]
tail(bxBooks_subset$Book_Title, 20) #5: "Petite histoire de la
dÃ?Â©sinformation"
```

```
##  [1] "Amok."
##  [2] "Introducing Nietzsche (Foundations in Children's Ministry)"
##  [3] "Core Web Programming (2nd Edition)"
##  [4] "The Unified Modeling Language Reference Manual (Addison-Wesley
Object Technology Series)"
##  [5] "Petite histoire de la dÃ?Â©sinformation"
##  [6] "Driving to Detroit: Memoirs of a Fast Woman"
##  [7] "Dreamsnake"
##  [8] "Der Mossad."
##  [9] "Slow Food(The Case For Taste)"
## [10] "Strong Democracy : Participatory Politics for a New Age"
## [11] "Burpee Gardening Cyclopedia: A Concise, Up to Date Reference for
Gardeners at All Levels"
## [12] "Tropical Rainforests: 230 Species in Full Color (Golden Guide)"
## [13] "Cocktail Classics"
## [14] "Anti Death League"
## [15] "Flashpoints: Promise and Peril in a New World"
## [16] "There's a Bat in Bunk Five"
## [17] "From One to One Hundred"
## [18] "Lily Dale : The True Story of the Town that Talks to the Dead"
## [19] "Republic (World's Classics)"
## [20] "A Guided Tour of Rene Descartes' Meditations on First Philosophy
with Complete Translations of the Meditations by Ronald Rubin"
```

Notice, that fifth book title in the output above, *Petite histoire de la dÃ?Â©sinformation*, contains non-US characters. To handle these, we can use the code below (WinVector, n.d.). "The current engine locale determines which characters are considered printable or alphabetic, the format of numerals and dates, and the collating sequence of characters (Set or get locale-specific information, n.d.)." The **Sys.setlocale()** returns a string describing the current locale after setting it to what you asked for. In this case, we set our locale to "C" means the locale used by the C language (Set or get locale-specific information, n.d.).

```
Sys.setlocale('LC_ALL','C')

## [1] "C"

tail(bxBooks_subset$Book_Title, 20) #5: "Petite histoire de la
dC?B)sinformation"

##  [1] "Amok."
##  [2] "Introducing Nietzsche (Foundations in Children's Ministry)"
##  [3] "Core Web Programming (2nd Edition)"
##  [4] "The Unified Modeling Language Reference Manual (Addison-Wesley
Object Technology Series)"
##  [5] "Petite histoire de la d\303?\302\251sinformation"
##  [6] "Driving to Detroit: Memoirs of a Fast Woman"
##  [7] "Dreamsnake"
##  [8] "Der Mossad."
##  [9] "Slow Food(The Case For Taste)"
## [10] "Strong Democracy : Participatory Politics for a New Age"
## [11] "Burpee Gardening Cyclopedia: A Concise, Up to Date Reference for
Gardeners at All Levels"
## [12] "Tropical Rainforests: 230 Species in Full Color (Golden Guide)"
## [13] "Cocktail Classics"
## [14] "Anti Death League"
## [15] "Flashpoints: Promise and Peril in a New World"
## [16] "There's a Bat in Bunk Five"
## [17] "From One to One Hundred"
## [18] "Lily Dale : The True Story of the Town that Talks to the Dead"
## [19] "Republic (World's Classics)"
## [20] "A Guided Tour of Rene Descartes' Meditations on First Philosophy
with Complete Translations of the Meditations by Ronald Rubin"
```

Notice the book title above has been converted to *Petite histoire de la dC?B)sinformation*. Now, we can remove punctuation from our book titles (WinVector, n.d.). While there are numerous ways of accomplishing this task, we will use the **gsub()** function, which allows us to replace a pattern (Regular expressions in r, n.d.). In this case, our pattern is punctuation characters, so we can use the **[:punct:]** character class.

```
booktokens <- gsub("[[:punct:]]", "", bxBooks_subset$Book_Title)
tail(booktokens, 20)
```

```
##  [1] "Amok"
##  [2] "Introducing Nietzsche Foundations in Childrens Ministry"
##  [3] "Core Web Programming 2nd Edition"
##  [4] "The Unified Modeling Language Reference Manual AddisonWesley Object
Technology Series"
##  [5] "Petite histoire de la d\303\302sinformation"
##  [6] "Driving to Detroit Memoirs of a Fast Woman"
##  [7] "Dreamsnake"
##  [8] "Der Mossad"
##  [9] "Slow FoodThe Case For Taste"
## [10] "Strong Democracy  Participatory Politics for a New Age"
## [11] "Burpee Gardening Cyclopedia A Concise Up to Date Reference for
Gardeners at All Levels"
## [12] "Tropical Rainforests 230 Species in Full Color Golden Guide"
## [13] "Cocktail Classics"
## [14] "Anti Death League"
## [15] "Flashpoints Promise and Peril in a New World"
## [16] "Theres a Bat in Bunk Five"
## [17] "From One to One Hundred"
## [18] "Lily Dale  The True Story of the Town that Talks to the Dead"
## [19] "Republic Worlds Classics"
## [20] "A Guided Tour of Rene Descartes Meditations on First Philosophy with
Complete Translations of the Meditations by Ronald Rubin"
```

Consider our original book title, *Petite histoire de la dÃ?Â©sinformation*. It has now been converted to *Petite histoire de la dCBsinformation*. Now, we will remove any unwanted space characters, which include tab, newline, vertical tab, form feed, carriage return, or space (Regular expressions in r, n.d.). We can do this by specifying our pattern to the **[:space:]** character class, followed by the "$" character.

```
sum(grepl("[[:space:]]+$", booktokens))

## [1] 162

cleantitle <- sub("[[:space:]]+$","",booktokens)
tail(cleantitle, 20)

##  [1] "Amok"
##  [2] "Introducing Nietzsche Foundations in Childrens Ministry"
##  [3] "Core Web Programming 2nd Edition"
##  [4] "The Unified Modeling Language Reference Manual AddisonWesley Object
Technology Series"
##  [5] "Petite histoire de la d\303\302sinformation"
##  [6] "Driving to Detroit Memoirs of a Fast Woman"
##  [7] "Dreamsnake"
##  [8] "Der Mossad"
##  [9] "Slow FoodThe Case For Taste"
## [10] "Strong Democracy  Participatory Politics for a New Age"
## [11] "Burpee Gardening Cyclopedia A Concise Up to Date Reference for
Gardeners at All Levels"
## [12] "Tropical Rainforests 230 Species in Full Color Golden Guide"
## [13] "Cocktail Classics"
## [14] "Anti Death League"
## [15] "Flashpoints Promise and Peril in a New World"
## [16] "Theres a Bat in Bunk Five"
## [17] "From One to One Hundred"
## [18] "Lily Dale  The True Story of the Town that Talks to the Dead"
## [19] "Republic Worlds Classics"
## [20] "A Guided Tour of Rene Descartes Meditations on First Philosophy with
Complete Translations of the Meditations by Ronald Rubin"
```

Finally, we can convert our book titles to all lowercase words.

```
booktokens <- tolower(cleantitle)
tail(booktokens, 20)

##  [1] "amok"
##  [2] "introducing nietzsche foundations in childrens ministry"
##  [3] "core web programming 2nd edition"
##  [4] "the unified modeling language reference manual addisonwesley object
technology series"
##  [5] "petite histoire de la d\343\342sinformation"
##  [6] "driving to detroit memoirs of a fast woman"
##  [7] "dreamsnake"
##  [8] "der mossad"
##  [9] "slow foodthe case for taste"
## [10] "strong democracy  participatory politics for a new age"
## [11] "burpee gardening cyclopedia a concise up to date reference for
```

```
gardeners at all levels"
## [12] "tropical rainforests 230 species in full color golden guide"
## [13] "cocktail classics"
## [14] "anti death league"
## [15] "flashpoints promise and peril in a new world"
## [16] "theres a bat in bunk five"
## [17] "from one to one hundred"
## [18] "lily dale  the true story of the town that talks to the dead"
## [19] "republic worlds classics"
## [20] "a guided tour of rene descartes meditations on first philosophy with
complete translations of the meditations by ronald rubin"
```

Using our clean data, we can create a new dataframe that contains ISBN numbers, our clean book titles, and the cleaned titles (WinVector, n.d.).

```
Books <- data.frame(ISBN=bxBooks_subset$ISBN, token=booktokens,
title=cleantitle)

head(Books)
```

```
##         ISBN
## 1 0195153448
## 2 0002005018
## 3 0060973129
## 4 0374157065
## 5 0393045218
## 6 0399135782
##
token
## 1
classical mythology
## 2
clara callan
## 3
decision in normandy
## 4 flu the story of the great influenza pandemic of 1918 and the search for
the virus that caused it
## 5
the mummies of urumchi
## 6
the kitchen gods wife
##
title
## 1
Classical Mythology
## 2
Clara Callan
## 3
Decision in Normandy
## 4 Flu The Story of the Great Influenza Pandemic of 1918 and the Search for
```

```
the Virus That Caused It
## 5
The Mummies of Urumchi
## 6
The Kitchen Gods Wife
```

```
length(Books$token)
```

```
## [1] 271379
```

```
length(unique((Books$token)))
```

```
## [1] 237550
```

Notice, the length of our **token** column exceeds the number of unique book titles, which means our data contains duplicates. To handle this, we can manipulate our dataframe using the **sqldf** package, which will allow us to perform SQL queries in R.

Manipulate Data Frame and Merge Data

Our final dataframe needs to consist of unique ISBN numbers, ratings, userID's, book titles, author of the book, year of publication, and the publisher. First, we will select the minimum ISBN number for each book and save this ISBN number for each book title and save this as the **unique_isbn**. We will then group this data by the book titles. This allows us to pick a unique ISBN number for every book title (WinVector, n.d.).

```
bookmap <- sqldf('SELECT min(ISBN) as unique_isbn,
                        token
                 FROM Books
                 GROUP BY token')
head(bookmap)
```

```
##   unique_isbn
## 1  1568580010
## 2  0590567330
## 3  1565920317
## 4  0563208422
## 5  0964147726
## 6  0743415183
##
token
## 1
a novel
## 2  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 3
a nutshell handbook
## 4                                                                 allo allo
the war diaries of rene artois
## 5
always have popsicles
```

```
## 6
and for starters

dim(bookmap)

## [1] 237549      2
```

Now that we have our unique book titles and unique ISBN numbers, we need to get the publication information for each book title and ISBN number above. This includes the author, year of publication, and publisher. We will

```
publication <- sqldf("SELECT Books.token as token,
bxBooks_subset.Book_Author, bxBooks_subset.Year_Of_Publication,
bxBooks_subset.Publisher
                      FROM Books, bxBooks_subset
                     WHERE bxBooks_Subset.ISBN=Books.ISBN")

dim(publication)

## [1] 271379      4

head(publication)

##
token
## 1
classical mythology
## 2
clara callan
## 3
decision in normandy
## 4 flu the story of the great influenza pandemic of 1918 and the search for
the virus that caused it
## 5
the mummies of urumchi
## 6
the kitchen gods wife
##            Book_Author Year_Of_Publication                  Publisher
## 1   Mark P. O. Morford                2002    Oxford University Press
## 2 Richard Bruce Wright                2001      HarperFlamingo Canada
## 3          Carlo D'Este               1991            HarperPerennial
## 4      Gina Bari Kolata               1999       Farrar Straus Giroux
## 5       E. J. W. Barber               1999 W. W. Norton &amp; Company
## 6               Amy Tan               1991           Putnam Pub Group
```

Similarly, we need to gather user ID’s and ratings.

```
ratings <- sqldf('SELECT bxBookRatings.User_ID as user_id, Books.token as
token, bxBookRatings.Book_Rating as rating
                    FROM Books, bxBookRatings
                    WHERE bxBookRatings.ISBN = Books.ISBN')
```

```
head(ratings)

##   user_id              token rating
## 1       2 classical mythology      0
## 2       8        clara callan      5
## 3   11400        clara callan      0
## 4   11676        clara callan      8
## 5   41385        clara callan      0
## 6   67544        clara callan      8

dim(ratings)

## [1] 1031176       3
```

Finally, using the data queried above, we can merge our data together to create our final dataframe (WinVector, n.d.). We can do this by using the **merge()** function and merge by the column **token**.

```
bookdata <- merge(ratings, bookmap, by="token")
bookdata <- merge(bookdata, publication, by = "token")
head(bookdata)

##
token
## 1
a novel
## 2  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 3  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 4  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 5  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 6
a nutshell handbook
##   user_id rating unique_isbn     Book_Author Year_Of_Publication
## 1  102967      0  1568580010 Michael Brodsky                1994
## 2   96448      9  0590567330     Karen Hesse                1999
## 3   35859      0  0590567330     Karen Hesse                1999
## 4   18995      0  0590567330     Karen Hesse                1999
## 5   55927      0  0590567330     Karen Hesse                1999
## 6  271245      0  1565920317   Donnalyn Frey                1993
##                    Publisher
## 1             Pub Group West
## 2 Hyperion Books for Children
## 3 Hyperion Books for Children
## 4 Hyperion Books for Children
## 5 Hyperion Books for Children
## 6                   O'Reilly
```

Now, we will want to check for any duplicate rows. To check for duplicate rows, we can use the **duplicated()** function (Identify and remove duplicate data in r, n.d.).

```
sum(duplicated(bookdata))

## [1] 81358
```

To remove the duplicated rows, we can **!duplicated()** where **!** is a logical negation and would mean that we do not want duplicate rows (Identify and remove duplicate data in r, n.d.). We will use the **distinct()** function from the **dplyr** package and can be used to keep distinct rows in the dataframe (Identify and remove duplicate data in r, n.d.).

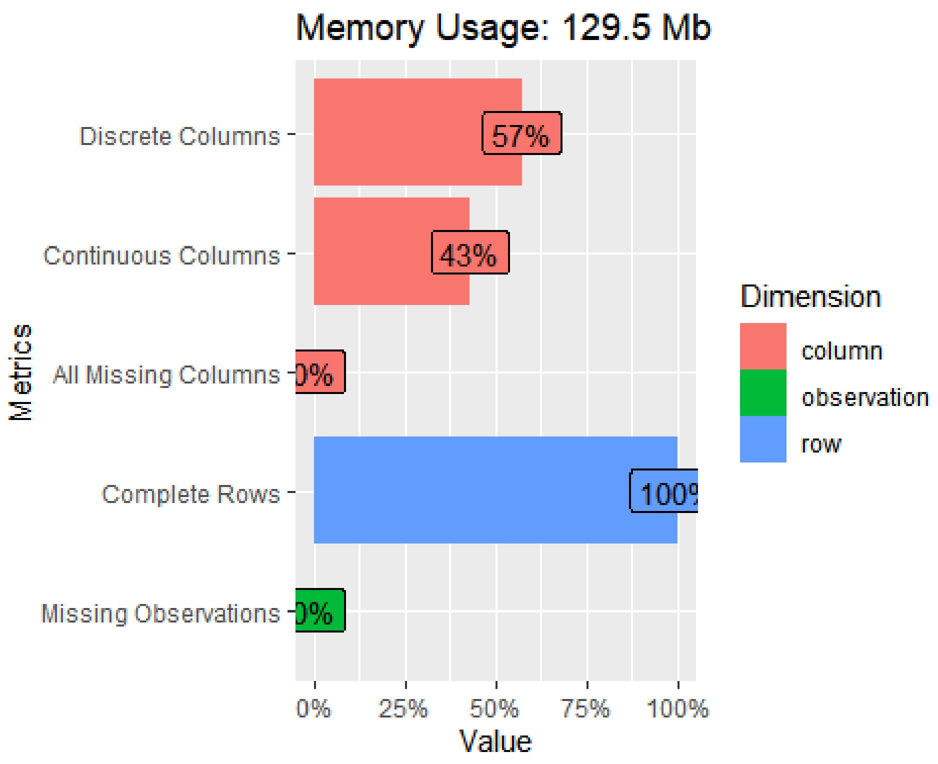
```
final_book <- bookdata %>% distinct()
#str(final_book)
```

Using our **final_book** dataframe, we can proceed with exploring our data.

Explore Data

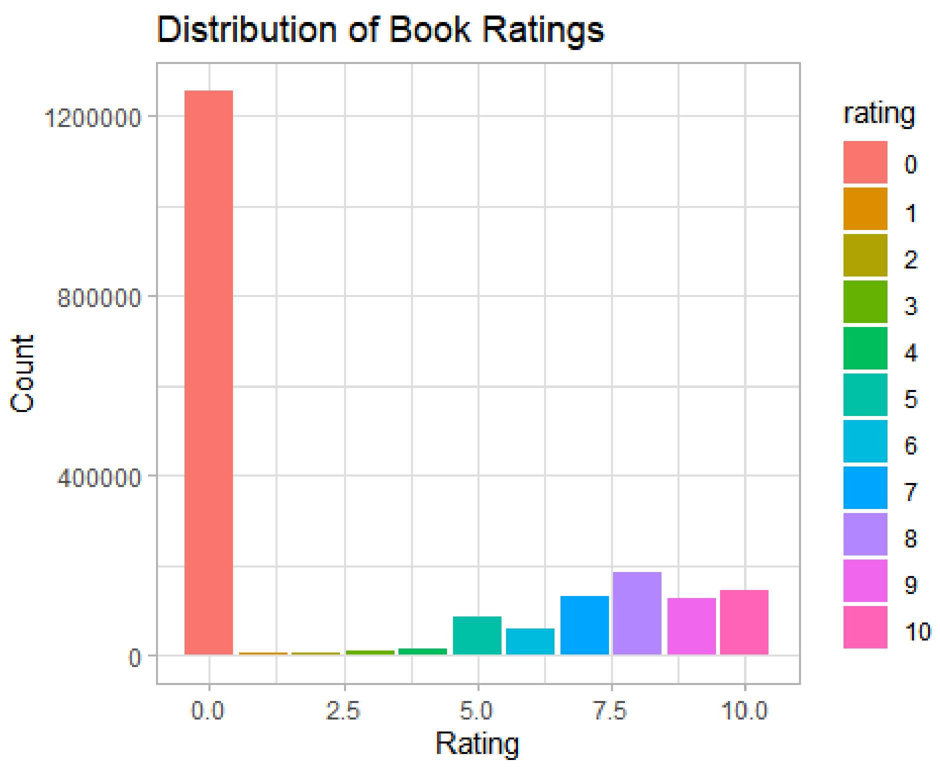
First, we need to determine if our dataframe contains any missing data. It shouldn't since none of the original dataframes contained any missing values.

```
plot_intro(final_book)
```



The plot above confirms our assumption about no missing data. The first thing we will look at is a distribution of ratings in our dataframe.

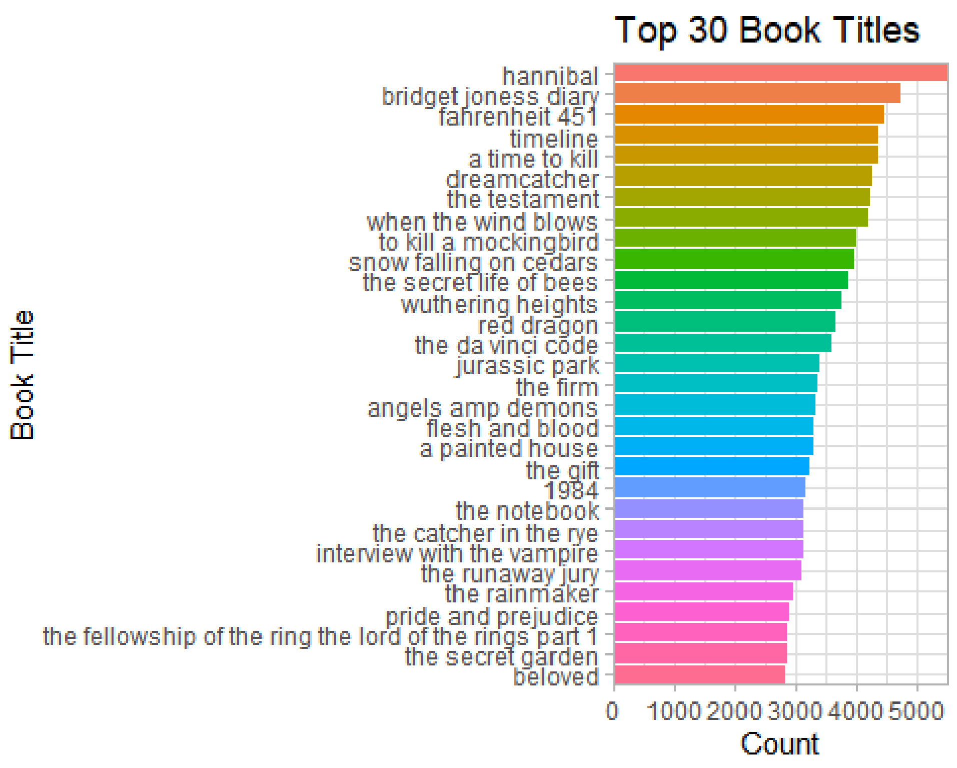
```
ggplot(data = final_book) + aes(x = rating, fill = factor(rating)) +
  geom_bar() +
  labs(title = "Distribution of Book Ratings", y = "Count", x = "Rating",
fill = "rating") +
  theme_light()
```



Our plot above indicates that none of our users gave a rating of 1 or 2, and most of the ratings are 0. Recall from the description of the **bx-Book-Rating** dataset that 0 represented implicit values (implied values that were not plainly expressed). In other words, most of the users in our dataset did not give any ratings to their books.

Now, we will determine the top 30 book titles in our final data frame. We can do this by creating a table of our book titles and sort them by decreasing frequency. To plot these values, we then convert our sorted table to a dataframe and only keep the top 30 books (R - plot table objects with ggplot, n.d.).

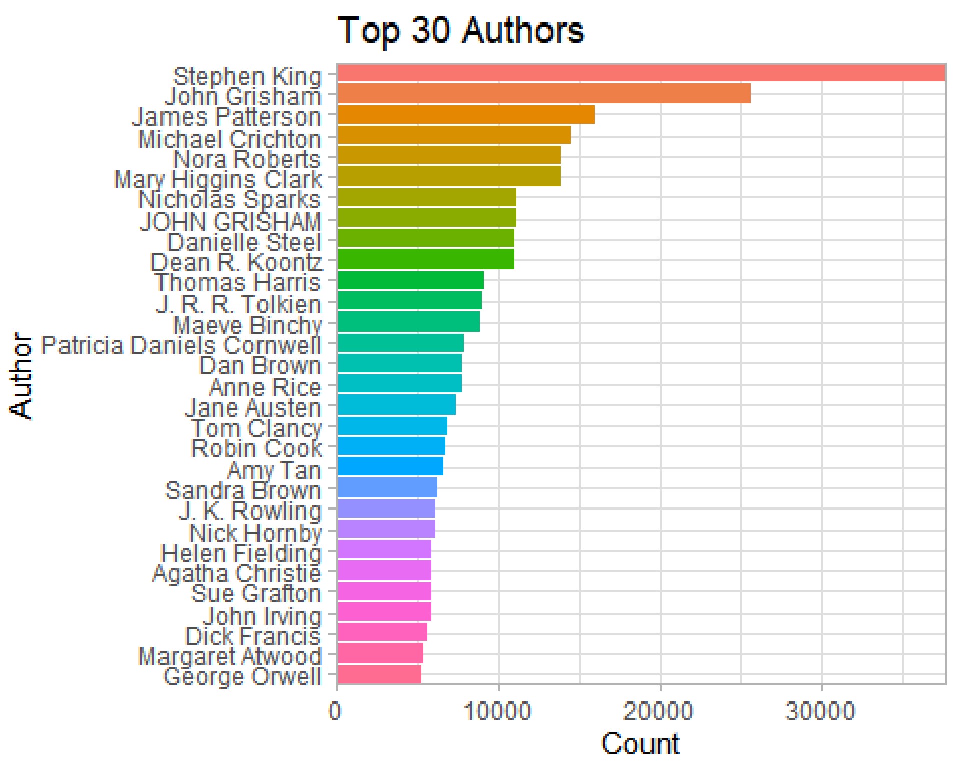
```
sorted_titles <- sort(table(final_book$token), decreasing = TRUE)
top_titles <- as.data.frame(sorted_titles[1:30])


ggplot(data = top_titles) + aes(x = reorder(Var1, Freq), y= Freq, fill =
Var1) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Top 30 Book Titles", y = "Count", x = "Book Title") +
  theme_light() +
  coord_flip(xlim = NULL, ylim = NULL, expand = FALSE, clip = "on")
```



The top read book in our data includes *Hannibal* and *Bridget Jones's Diary*. Both books appear in our dataframe over 4,500 times.

Now, we will view the top 30 authors in our dataframe, using similar steps above.

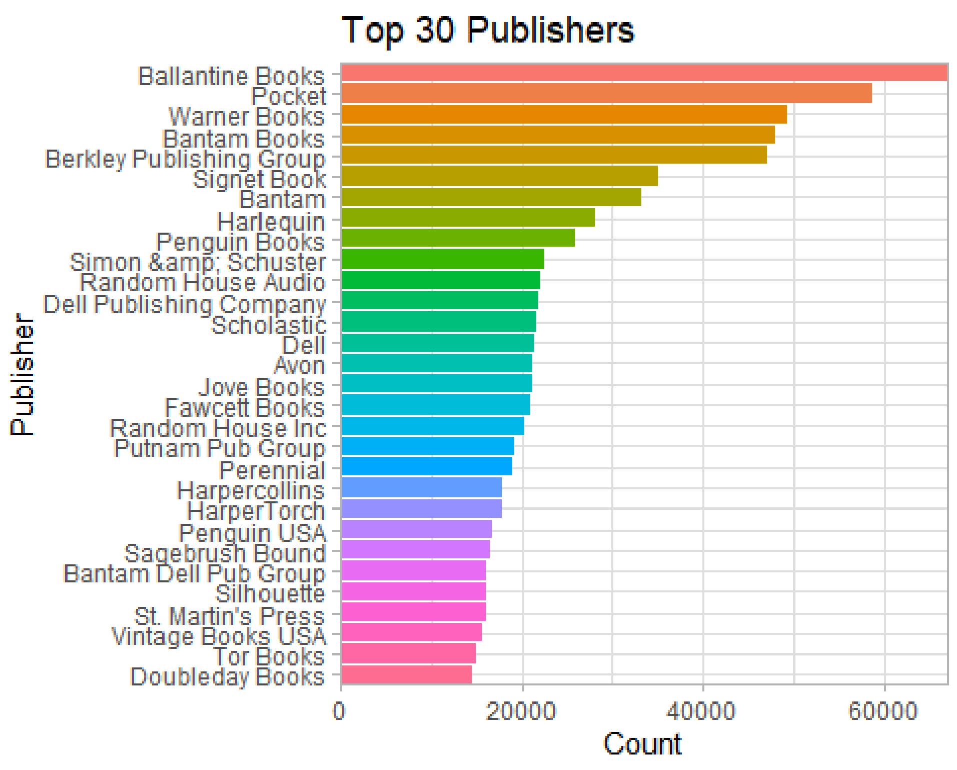
```
#top authors
sorted_author <- sort(table(final_book$Book_Author), decreasing = TRUE)
top_author <- as.data.frame(sorted_author[1:30])


ggplot(data = top_author) + aes(x = reorder(Var1, Freq), y= Freq, fill =
Var1) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Top 30 Authors", y = "Count", x = "Author") +
  theme_light() +
  coord_flip(xlim = NULL, ylim = NULL, expand = FALSE, clip = "on")
```



A few of the top authors in our dataframe include *Stephen King*, *James Patterson*, *Nora Roberts*, and *Nicholas Sparks*.

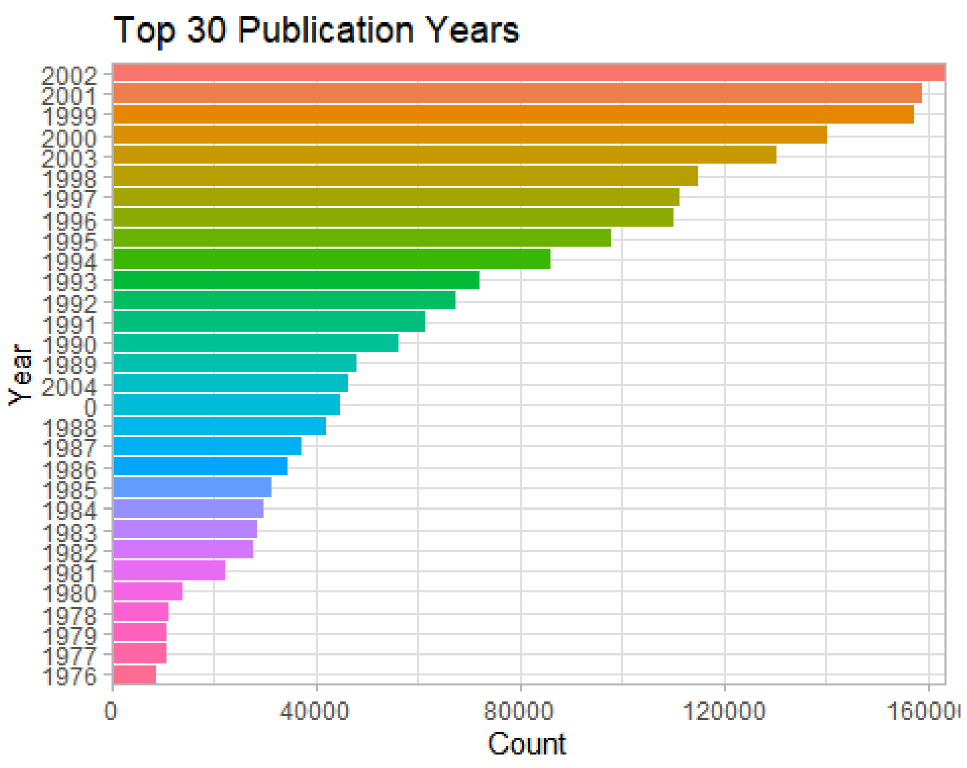
Now, we can view the top publishers in our dataframe.

```
sorted_publisher <- sort(table(final_book$Publisher), decreasing = TRUE)
top_publisher <- as.data.frame(sorted_publisher[1:30])


ggplot(data = top_publisher) + aes(x = reorder(Var1, Freq), y= Freq, fill =
Var1) +
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Top 30 Publishers", y = "Count", x = "Publisher") +
  theme_light() +
  coord_flip(xlim = NULL, ylim = NULL, expand = FALSE, clip = "on")
```



Finally, we can view the top years of publication for our books.

```
#top year_of_publication
sorted_year <- sort(table(final_book$Year_Of_Publication), decreasing = TRUE)
top_year <- as.data.frame(sorted_year[1:30])


ggplot(data = top_year) + aes(x = reorder(Var1, Freq), y= Freq, fill = Var1)
+
  geom_bar(stat = "identity", show.legend = FALSE) +
  labs(title = "Top 30 Publication Years", y = "Count", x = "Year") +
  theme_light() +
  coord_flip(xlim = NULL, ylim = NULL, expand = FALSE, clip = "on")
```



From the output above, many of the books in our data were published in the late 1990's and the early 2000's.

Write Data TSV File and Convert Data to Transactions Using (arules) Package

As mentioned above, to generate association rules, our data needs to be converted to transactions. To do this, we first need to convert our variables to factors. To do this, we can use the **mutate()** function in the **dplyr** package (R - changing factor levels with dplyr mutate, n.d.).

```
final_book <- final_book %>% mutate(token=as.factor(token))
final_book <- final_book %>% mutate(user_id=as.factor(user_id))
final_book <- final_book %>% mutate(rating=as.factor(rating))
final_book <- final_book %>% mutate(unique_isbn=as.factor(unique_isbn))
final_book <- final_book %>% mutate(Book_Author=as.factor(Book_Author))
final_book <- final_book %>%
mutate(Year_Of_Publication=as.factor(Year_Of_Publication))
final_book <- final_book %>% mutate(Publisher=as.factor(Publisher))
```

Now, we need to save our data in a tab- or column-separated format. We can save our data by using the **write.table()** function, which is used to export a dataframe to a file (Writing data from r to txt|csv files, n.d.). The arguments used when writing our file below include (Writing data from r to txt|csv files, n.d.):

- x: Dataframe to be written

- *file*: The name of the result file
- *sep*: Field separator string (" for tab-separated)
- *row.names*: FALSE to indicate that the rows names of *x* are to not be written.
- *col.names*: Can be TRUE or FALSE, but we include a character vector of the column names to be written.

```
write.table(x = final_book,
            file="books_merged.tsv",
            sep="\t",
            row.names = FALSE,
            col.names = c("Book_Title", "User_ID", "Rating", "ISBN",
"Book_Author", "Year_Of_Publication", "Publisher"))
```

Now, we need to convert the format above to a "transaction" class. This is a sparse matrix, where each row is a transaction, and each column is an item (book). The binary numbers (0,1) are used to indicate the no-interest/interest on a specific item. We can do this by using the **read.transactions()** function in the **arules** package. The arguments we will use include (Read transaction data in arules, n.d.):

- *file*: File name
- *format*: Format of dataset
  - For "single" format, each line corresponds to a single item, containing at least ids for the transaction and the item
- *sep*: Field separator string (" for tab-separated)
- *cols*: For the "single" format, *cols* is a numeric or character vector of length two giving the numbers or names of the columns (fields) with the transaction and item ids, respectively
- *header*: TRUE to include column names
- *rm.duplicates*: TRUE to remove duplicate items

```
transaction_book <- read.transactions(file ='books_merged.tsv', format =
"single", sep="\t", cols=c("User_ID", "Book_Title"), header = TRUE,
rm.duplicates = TRUE)
```

Using our **transaction_book**, we can explore the transactions and item frequency plots.

Explore Transaction Data

To view the object class, we can use the command below.

```
class(transaction_book)
```

```
## [1] "transactions"
## attr(,"package")
## [1] "arules"
```

The object is of class transactions (Zumel, 2019). To view the **transaction_book** object dimensions, we can use the command below (Zumel, 2019).

```
transaction_book

## transactions in sparse format with
##  92108 transactions (rows) and
##  236501 items (columns)
```

To view the how the columns are labeled, we can use the **colnames()** function (Zumel, 2019).

```
colnames(transaction_book)[1:5]

## [1] "   a novel"
## [2] " a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america"
## [3] " a nutshell handbook"
## [4] " allo allo the war diaries of rene artois"
## [5] " always have popsicles"
```

We can see our columns are labeled by book title. Similarly, to view how the row names are labeled, we can use the **rownames()** function (Zumel, 2019).

```
rownames(transaction_book)[1:5]

## [1] "10"     "1000"   "100001" "100002" "100004"
```

Our row names are labeled by customer. To get more information about our transaction object, we can use the **summary()** function (Market basket analysis using r, 2018). The partial output is below.

```
summary(transaction_book)

## transactions as itemMatrix in sparse format with
##  92108 rows (elements/itemsets/transactions) and
##  236501 columns (items) and a density of 4.709269e-05
##
## most frequent items:
##               wild animus  the lovely bones a novel         the da vinci
code
##                      2502                      1295
897
## the nanny diaries a novel           a painted house
(Other)
##                       821                       818
1019517
##
```

```
## element (itemset/transaction) length distribution:
## sizes
##     1     2     3     4     5     6     7     8     9    10    11    12
13
## 51282 10805  5758  3853  2699  2043  1609  1239  1078   899   757   643
555
##    14    15    16    17    18    19    20    21    22    23    24    25
26
##   459   460   396   342   333   267   260   236   220   197   181   180
173
##    27    28    29    30    31    32    33    34    35    36    37    38
39


++++++++++++++++++++++

##  1491  1499  1516  1520  1527  1554  1590  1599  1624  1628  1646  1674
1699
##     1     1     1     1     2     1     1     1     1     1     1     1
1
##  1723  1741  1806  1815  1872  1886  1973  2048  2070  2151  2204  2208
2223
##     1     1     1     1     1     1     1     1     1     1     1     1
1
##  2241  2285  2286  2304  2322  2336  2370  2397  2423  2556  2811  2874
2903
##     1     1     1     1     1     1     1     1     1     1     1     1
1
##  2939  3258  3969  4229  5525  5705  5736  6345 10509
##     1     1     1     1     1     1     1     1     1
##
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     1.00     1.00     1.00    11.14     4.00 10509.00
##
## includes extended item information - examples:
##
labels
## 1
a novel
## 2  a light in the storm the civil war diary of amelia martin fenwick
island delaware 1861 dear america
## 3
a nutshell handbook
##
## includes extended transaction information - examples:
##   transactionID
## 1            10
## 2          1000
## 3        100001
```

The output above gives us the following information (Market basket analysis using r, 2018):

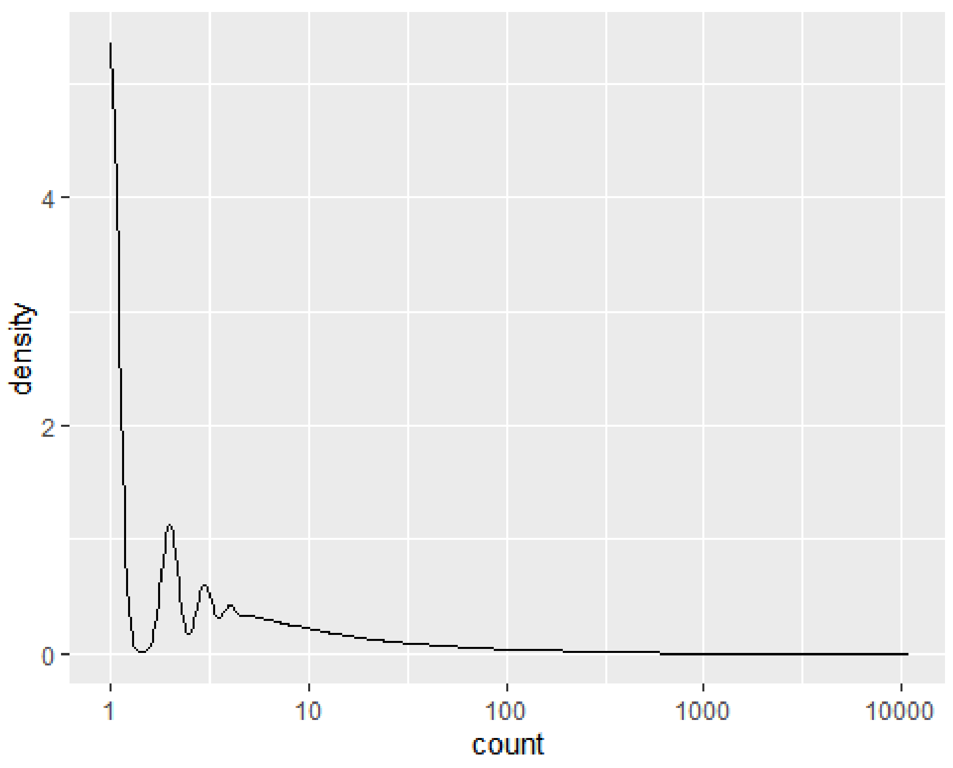
- There are 92,108 transactions (rows) and 236,502 items (columns)
- Density tells the percentage of non-zero cells in a sparse matrix. You can say it as the total number of items that are purchased divided by a possible number of items in that matrix.
    - We can calculate how many items were purchased by using density: $92{,}108 * 236{,}502 * 4.709249e^{-05} = 1{,}025{,}850$
- Most frequent items
- Element (itemset/transaction) length distribution: This is telling you how many transactions are there for 1-itemset, for 2-itemset and so on. The first row is telling you a number of items and the second row is telling you the number of transactions.
    - For example, there is only 51,282 for one item, 10,805 for two items, and there are 10,509 in one transaction which is the longest.

Now, to examine the distribution of transaction sizes, we can use the **size()** function (Zumel, 2019).

```
booksizes <- size(transaction_book)
summary(booksizes)

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     1.00     1.00     1.00    11.14     4.00 10509.00
```

The output above suggests that most customers (at least half of them, in fact) only expressed interest in one book. But someone has expressed interest in more than 10,000 (Zumel, 2019). To look more closely at the size distribution, we can use the code below.

```
quantile(booksizes, probs = seq(0,1,0.1))

##    0%   10%   20%   30%   40%   50%   60%   70%   80%   90%  100%
##     1     1     1     1     1     1     2     3     5    13 10509

ggplot(data.frame(count = booksizes)) +
  geom_density(aes(x = count)) +
  scale_x_log10()
```



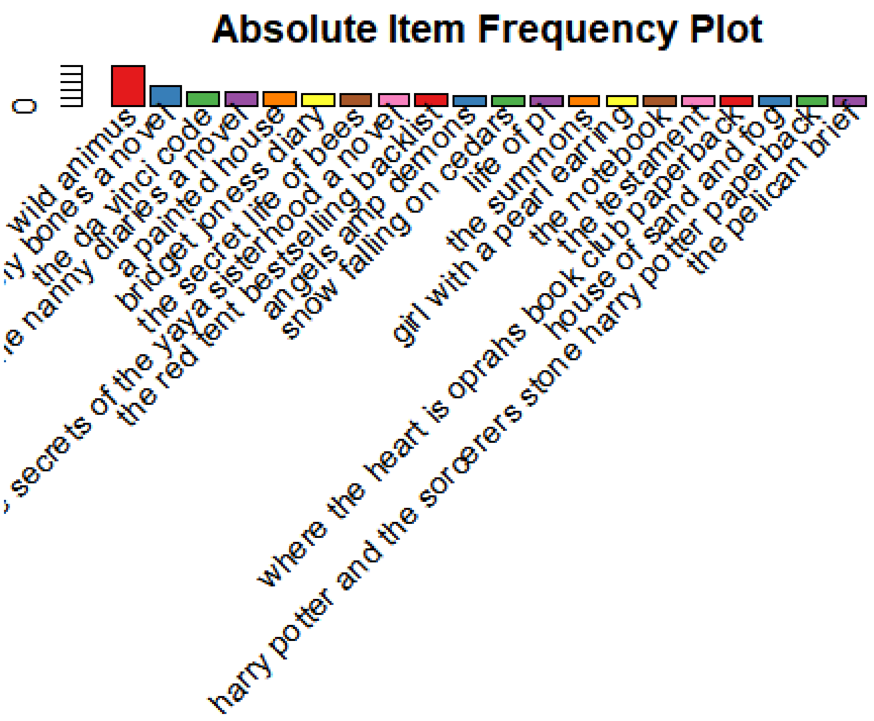
By looking at the transaction distribution in 10% increments and plotting the distribution, we can see that 90% of customers expressed their interest in fewer than 15 books. The remaining 10% of customers expressed their interest in up to 100 books. To view the portion of the customers, we can use the **quantile()** function again, but view size distribution for the 91% – 99% quantiles (Zumel, 2019).

```
quantile(booksizes, probs = seq(0.91, 1, 0.01))

##   91%   92%   93%   94%   95%   96%   97%   98%   99%  100% 
##    15    17    20    25    31    40    56    89   179 10509
```

We can see that 99% of customers expressed their interest in less than 180 books.

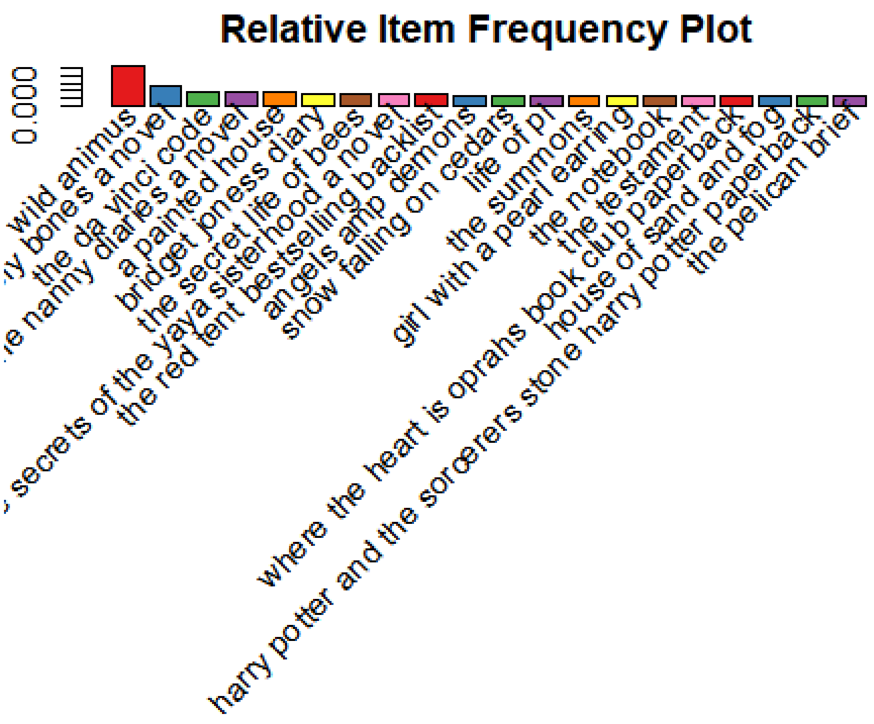
To view the distribution of objects, we can create an item frequency bar plot (Market basket analysis using r, 2018).

```
par(mar = rep(2, 4))
arules::itemFrequencyPlot(transaction_book, topN = 20, type = "absolute", col
= brewer.pal(8,"Set1"), main = "Absolute Item Frequency Plot")
```



The plot above allows us to plot the top 20 highest frequency items. By specifying **type = “absolute”**, we plot the numeric frequencies of each item independently. To plot them by how many times each item has appeared compared to others, we can use **type = "relative**.

```
par(mar = rep(2, 4))
arules::itemFrequencyPlot(transaction_book, topN = 20, type = "relative", col
= brewer.pal(8,"Set1"), main = "Relative Item Frequency Plot")
```



In both cases, *Wild Animus* and *The Lovely Bones: A Novel* are the most occurring items.

Create Association Rules

Using our transaction object, we can begin to mine association rules using the **apriori()** function. The **apriori algorithm** mines frequent itemsets and relevant association rules (Zumel, 2019). There are three significant components that comprise the apriori algorithm: support, confidence, and lift. To use the **apriori()** function, we need to decide on a minimum support level and confidence level.

As determined above, half of the customers in our data only expressed interest in a single book. We want to find books that occur together in people's interest lists, so we need to filter out the customers who only had interest in a single book (Zumel, 2019).

```
transaction_book_2 <- transaction_book[booksizes > 1]
dim(transaction_book_2)

## [1]  40826 236501
```

Now, we will use a support level of 0.005 and a confidence level of 0.70.

```
association_rules1 <- apriori(transaction_book_2, parameter = list(support =
0.005, conf = 0.70))

## Apriori
##
## Parameter specification:
```

```
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.7    0.1    1 none FALSE            TRUE       5   0.005      1
##  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: 204
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[231844 item(s), 40826 transaction(s)] done [0.73s].
## sorting and recoding items ... [252 item(s)] done [0.02s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [3 rule(s)] done [0.00s].
## creating S4 object  ... done [0.02s].
```

```
summary(association_rules1)
```

```
## set of 3 rules
##
## rule length distribution (lhs + rhs):sizes
## 3
## 3
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       3       3       3       3       3       3
##
## summary of quality measures:
##     support           confidence        coverage              lift
##  Min.   :0.005021   Min.   :0.7678   Min.   :0.005683   Min.   :68.71
##  1st Qu.:0.005021   1st Qu.:0.8238   1st Qu.:0.005695   1st Qu.:78.01
##  Median :0.005021   Median :0.8798   Median :0.005707   Median :87.31
##  Mean   :0.005021   Mean   :0.8437   Mean   :0.005977   Mean   :81.79
##  3rd Qu.:0.005021   3rd Qu.:0.8817   3rd Qu.:0.006124   3rd Qu.:88.33
##  Max.   :0.005021   Max.   :0.8836   Max.   :0.006540   Max.   :89.35
##      count
##  Min.   :205
##  1st Qu.:205
##  Median :205
##  Mean   :205
##  3rd Qu.:205
##  Max.   :205
##
## mining info:
##                data ntransactions support confidence
##  transaction_book_2         40826   0.005        0.7
```

We have found 3 rules. We can print out the rules by using the **inspect()** function.

```
inspect(sort(association_rules1, by = "confidence", decreasing = TRUE))

##     lhs                                                 rhs
support confidence    coverage     lift count
## [1] {harry potter and the goblet of fire book 4,
##      harry potter and the prisoner of azkaban book 3} => {harry potter and
the chamber of secrets book 2}  0.00502131  0.8836207 0.005682653 68.71371
205
## [2] {harry potter and the chamber of secrets book 2,
##      harry potter and the goblet of fire book 4}      => {harry potter and
the prisoner of azkaban book 3} 0.00502131  0.8798283 0.005707147 89.35291
205
## [3] {harry potter and the chamber of secrets book 2,
##      harry potter and the prisoner of azkaban book 3} => {harry potter and
the goblet of fire book 4}      0.00502131  0.7677903 0.006539950 87.31422
205
```

The rules we have found involve the novel series, Harry Potter. Approximately 89% of customers who bought the third and fourth Harry Potter books, also bought the second book. Further, 88% of customers who bought the second and fourth Harry Potter books, also bought the third book. Finally, only 77% of customers who bought the second and third Harry Potter books, also bought the fourth book.

To get more rules, we will have to adjust the support and confidence levels. Below, we will use a support level of 0.001 and a confidence level of 0.60.

```
association_rules2 <- apriori(transaction_book, parameter = list(support =
0.001, conf = 0.60))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.6    0.1    1 none FALSE            TRUE       5   0.001      1
##  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: 92
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[236501 item(s), 92108 transaction(s)] done [0.80s].
## sorting and recoding items ... [1027 item(s)] done [0.02s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4 done [0.02s].
## writing ... [77 rule(s)] done [0.00s].
## creating S4 object  ... done [0.02s].
```

```
summary(association_rules2)
```

```
## set of 77 rules
##
## rule length distribution (lhs + rhs):sizes
##  2  3  4
## 12 56  9
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.000   3.000   3.000   2.961   3.000   4.000
##
## summary of quality measures:
##     support           confidence        coverage              lift
##  Min.   :0.001010   Min.   :0.6000   Min.   :0.001064   Min.   :100.5
##  1st Qu.:0.001064   1st Qu.:0.6939   1st Qu.:0.001303   1st Qu.:149.3
##  Median :0.001107   Median :0.7881   Median :0.001487   Median :192.0
##  Mean   :0.001270   Mean   :0.7825   Mean   :0.001674   Mean   :219.2
##  3rd Qu.:0.001303   3rd Qu.:0.8733   3rd Qu.:0.001629   3rd Qu.:309.1
##  Max.   :0.002899   Max.   :1.0000   Max.   :0.004582   Max.   :549.9
##      count
##  Min.   : 93
##  1st Qu.: 98
##  Median :102
##  Mean   :117
##  3rd Qu.:120
##  Max.   :267
##
## mining info:
##              data ntransactions support confidence
##  transaction_book         92108   0.001        0.6
```

We have found 77 rules. We can then view the top 5 rules.

```
inspect(sort(association_rules2[1:5], by = "confidence", decreasing = TRUE))
#print and sort rules by confidence
```

```
##     lhs                                                    rhs
support confidence    coverage     lift count
## [1] {the amber spyglass his dark materials book 3}        => {the subtle
knife his dark materials book 2}
0.001172537  0.7714286 0.001519955 318.6311   108
## [2] {nicolae the rise of antichrist left behind no 3}     => {tribulation
force the continuing drama of those left behind left behind no 2} 0.001042255
0.7619048 0.001367959 385.5908    96
## [3] {the return of the king the lord of the rings part 3} => {the two
towers the lord of the rings part 2}
0.001726234  0.6737288 0.002562210 244.3142   159
## [4] {the two towers the lord of the rings part 2}         => {the return
of the king the lord of the rings part 3}
0.001726234  0.6259843 0.002757632 244.3142   159
## [5] {the two towers the lord of the rings part 2}         => {the
fellowship of the ring the lord of the rings part 1}
0.001682807  0.6102362 0.002757632 157.8866   155
```

Now, we see books that are a part of the Lord of the Rings series. To evaluate these rules, we can use the *coverage* and *Fisher's exact test* (Zumel, 2019). Coverage is the support on the left side of the rule (X) and it tells us how often the rule would be applied in the dataset. Fisher's exact test is a significance test for whether the observed patter is real or not, which is similar to lift (Zumel, 2019). We want the p-value that the test returns to be small because this indicates that the probability that we see the observed pattern by chance is also small. To use these measures, we can use the **interestMeasure()** function with the discovered rules, list of measures, and the transaction dataset as arguments (Zumel, 2019).

```
measures <- interestMeasure(association_rules2,
                            measure = c("coverage", "fishersExactTest"),
                            transactions = transaction_book_2)
summary(measures)
```

```
##     coverage         fishersExactTest
##  Min.   :0.001064   Min.   : 0.000e+00
##  1st Qu.:0.001303   1st Qu.: 0.000e+00
##  Median :0.001487   Median : 0.000e+00
##  Mean   :0.001674   Mean   :1.871e-171
##  3rd Qu.:0.001629   3rd Qu.: 0.000e+00
##  Max.   :0.004582   Max.   :1.441e-169
```

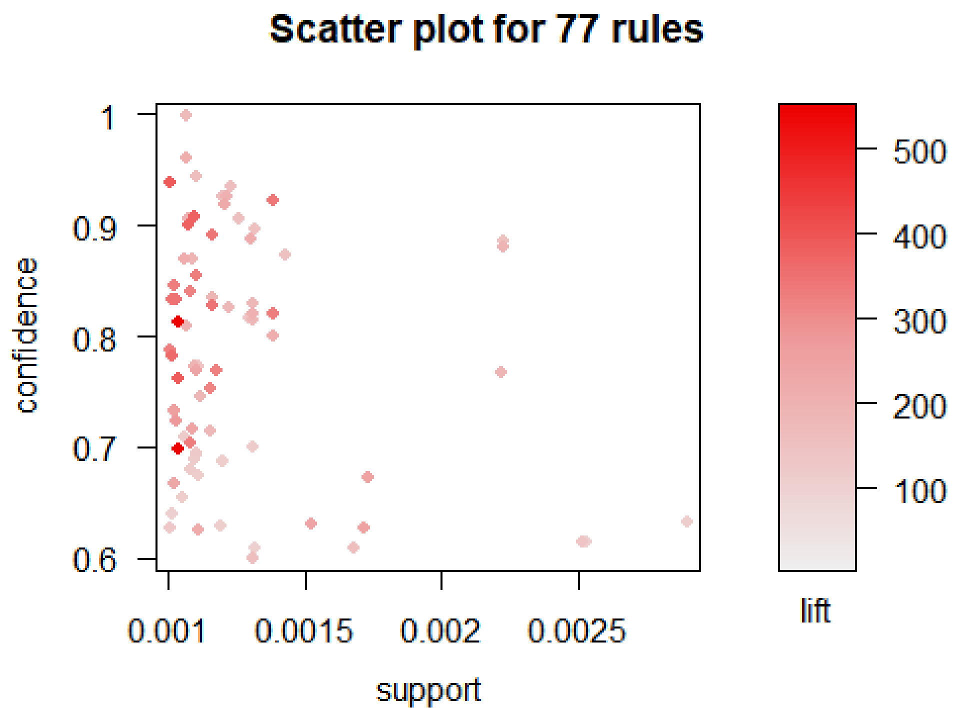
The coverage values range from 0.001 and 0.004, which is equivalent to a range of about $40 - 163$ people ($number\ of\ transactions * coverage\ value$). All the p-values for Fisher's test are small, so this suggests that the rules reflect actual customer behavior patterns (Zumel, 2019).

Visualize Association Rules

Using the **arulesViz** package, we can visualize our association rules. First, we can create a scatterplot using the **plot()** function, which returns the support on the x-axis and plots confidence on the y-axis (Market basket analysis using r, 2018). It also color-codes the plot by lift values.

```
subrules <- association_rules2[quality(association_rules2)$confidence >0.50]
plot(subrules, method = "scatterplot", measure = c("support", "confidence"),
shading = "lift")
```

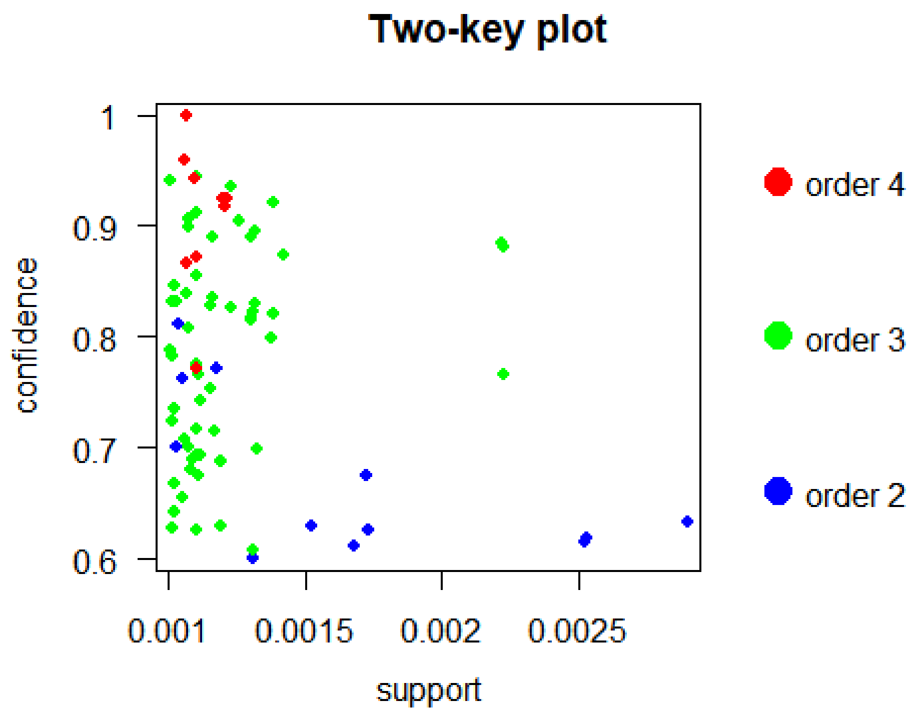
```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```



The plot above suggests that rules with high lift have low support. To plot using order for coloring, we can create a *two-key* plot (Market basket analysis using r, 2018). Order refers to the number of items in the rule.

```r
plot(subrules, method = "two-key plot", measure = c("support", "confidence"),
shading = "order")
```

```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```



Most of our rules have a total of 3 items in them (order 3).

Conclusion

For this assignment, we experimented with association rules and transaction objects. Association rules look to find relationships and patterns in data. In particular, the apriori algorithm is used in association rule mining and is one of the most frequently used algorithims (Market basket analysis using r, 2018). We began our assignment by loading our data and applying various data cleaning techniques. From here, we used the **sqldf** package to manipulate our dataframe using SQL and merege our desired attributes into a single dataframe. Once the final dataframe was created, we created barplots to view the distributions of ratings, and to view the top 30 books, authors, publishers, and publication years. Next, we saved our data and converted it into a transaction format. This converts our data into a sparse matrix, which is where most of the elements are zero (Market basket analysis using r, 2018). Using our transaction object, we used the apriori algorithm to generate association rules. In both sets of mined rules, we found that the books with the highest confidence were books found in a series: *Harry Potter* and *Lord of the Rings*. Using coverage and Fisher's exact test, we found that the rules we found are rules that reflect actual customer behavior patterns. Further inspection also found that rules with a lower lift had higher support and rules with three items in them occurred the most.

Resources

Book-crossing dataset. (n.d.). Retrieved November 22, 2020, from http://www2.informatik.uni-freiburg.de/~cziegler/BX/

Identify and remove duplicate data in r. (n.d.). Datanovia. Retrieved November 22, 2020, from https://www.datanovia.com/en/lessons/identify-and-remove-duplicate-data-in-r/

Market basket analysis using r. (2018, August 21). DataCamp Community. https://www.datacamp.com/community/tutorials/market-basket-analysis-r

R - changing factor levels with dplyr mutate. (n.d.). Stack Overflow. Retrieved November 22, 2020, from https://stackoverflow.com/questions/28190435/changing-factor-levels-with-dplyr-mutate

R - plot table objects with ggplot. (n.d.). Stack Overflow. Retrieved November 22, 2020, from https://stackoverflow.com/questions/26788049/plot-table-objects-with-ggplot

Read transaction data in arules. (n.d.). Retrieved November 22, 2020, from https://rdrr.io/cran/arules/man/read.transactions.html

Regular expressions in r. (n.d.). Retrieved November 22, 2020, from https://rstudio-pubs-static.s3.amazonaws.com/74603_76cd14d5983f47408fdf0b323550b846.html

Set or get locale-specific information. (n.d.). Retrieved November 22, 2020, from https://docs.tibco.com/pub/enterprise-runtime-for-R/5.0.0/doc/html/Language_Reference/base/Sys.setlocale.html

WinVector. (n.d.). GitHub. Retrieved November 22, 2020, from https://github.com/WinVector/zmPDSwR

Writing data from r to txt|csv files. (n.d.). Retrieved November 22, 2020, from http://www.sthda.com/english/wiki/writing-data-from-r-to-txt-csv-files-r-base-functions

Zumel (2019) - Chapter 9 from Practical Data Science with R, 2nd.