Project Report on

Analysing an E-commerce Online Retail dataset to perform Market Basket and Item-based Collaborative Filtering Algorithms

Submitted by

Anushka Pandhare
Gowri Dath
Mandeep Singh Narang
Yasmin Sultana
Zhenyuan Tian

BUAN 6356: Business Analytics with R
University of Texas at Dallas

TABLE OF CONTENTS

Sr. No.	Title	Page No.
1.	Abstract	
2.	Introduction • Background of the study • Dataset Information • Problem Statement • Objectives	1
3.	Data Exploration • Variables Exploration • Dataset cleaning • Dataset visualisation	3
4.	Algorithms • Market Basket Analysis • Item-based Collaborative Filtering	14
5.	Results and Interpretation	20
6.	Suggestions and Further Improvements	22
7.	Conclusion	23
8.	Appendix 1: Data Exploration in R Studio – Part 1 Appendix 2: Data Exploration in R Studio – Part 2 Appendix 2: Implementation of Market Basket Analysis in R Studio Appendix 3: Implementation of Item Based Collaborative Filtering in R Studio	i-xxxiv
9.	References	

ABSTRACT

The explosive growth of the world-wide-web and the emergence of e-commerce has led to the development of recommender systems-a personalized information filtering technology used to identify a set of N items that will be of interest to a certain user. E-commerce systems use recommenders as the most important analytic tools to be deployed. It's a known fact that ecommerce platforms like Amazon and Netflix have increased their revenues by 10% to 25% after using recommenders. Recommenders are not all the same, while their end goal is to optimise sales, different recommenders are applied to different datasets.

Market Basket Analysis is used for data that focuses exclusively on products. It's a form of statistical analysis that finds items frequently bought together by consumers and recommends them in the future. Now a days Market Basket Analysis is the most commonly implemented recommender system with tags of "Customers who bought product A also bought product B". MBA works by creating a set of rules called Association rules that showcase the association of multiple products between each other.

User-based Collaborative filtering is the most successful technology for building recommender systems to date, and is extensively used in many commercial recommender systems. Unfortunately, the computational complexity of these methods grows linearly with the number of customers that in typical commercial applications can grow to be several millions. To address these scalability concerns item-based recommendation techniques have been developed that analyse the user-item matrix to identify relations between the different items, and use these relations to compute the list of recommendations.

In this project, we present two cases – one, market basket analysis was performed on the e-commerce dataset by initially generating association rules using Apriori from the arules library. Key steps here were converting the dataset to transaction form and then implementing Apriori algorithm. The key parameters in determining association rules - support and confidence – were assumed. Second, we implemented item-based collaboration filtering which first determines the similarities between the products being purchased and uses them to recommend it to a similar user. The key steps in this class of algorithms are (i) the method used to compute the similarity between the items, and (ii) the method used to combine these similarities in order to compute the similarity between a basket of items and a candidate recommender item.

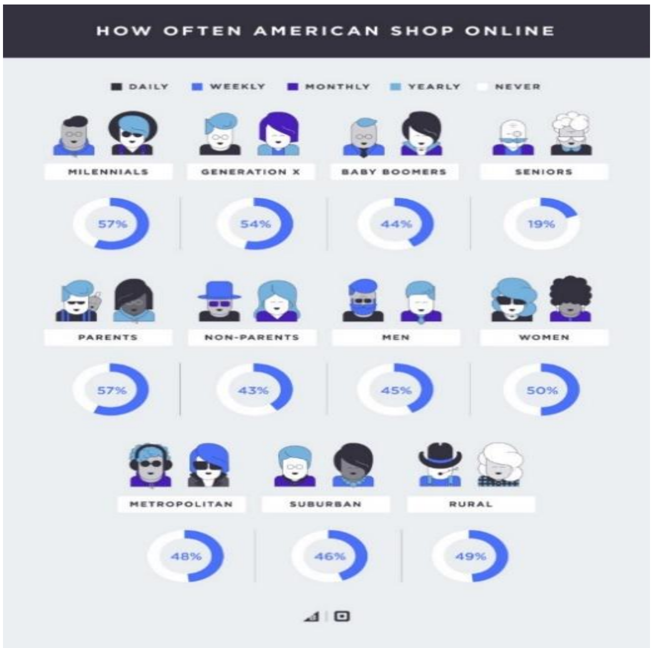
INTRODUCTION

I. Background of the study

"Ecommerce (or electronic commerce) refers to the buying and selling of goods (or services) on the internet."

Ecommerce was initially introduced 40 years ago in its earliest form. Since then, electronic commerce has helped countless businesses grow with the help of new technologies, improvements in internet connectivity, and widespread consumer and business adoption.

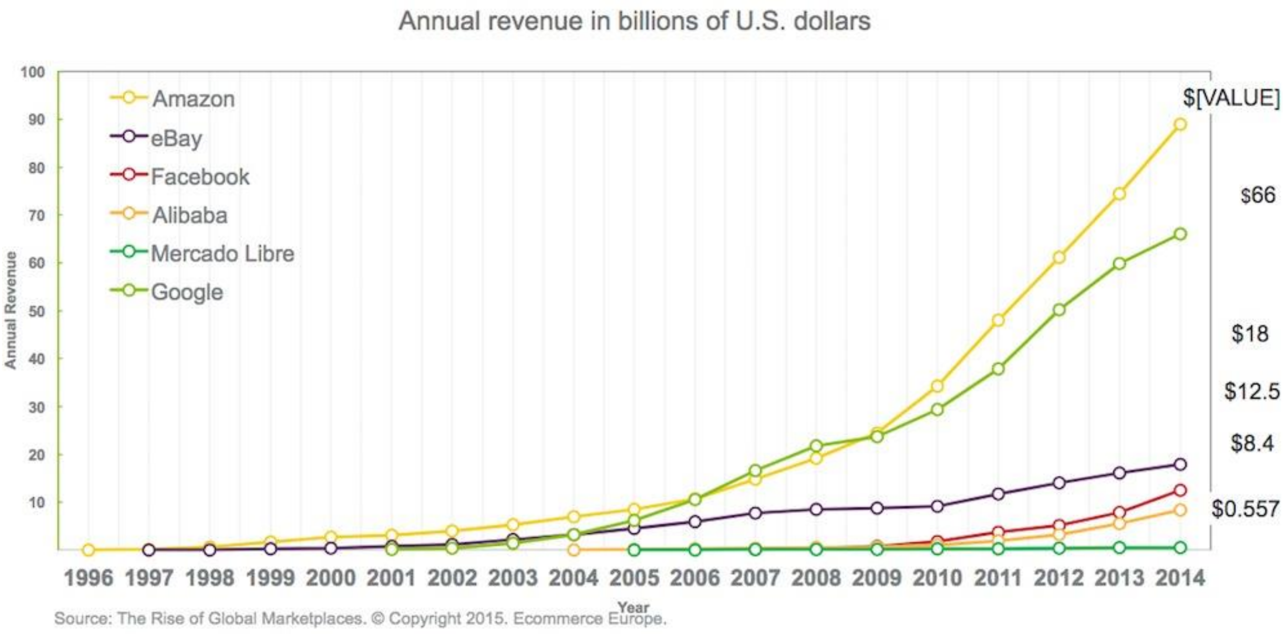
One of the first ecommerce transactions was made back in 1982, and today, it is growing by as much as 23% year over year. Forbes reported in 2016 that 57% of people surveyed in 24 countries across six continents had made an online purchase in the past six months.



Source: https://www.bigcommerce.com/blog/ecommerce/#ecommerce-timeline

From mobile shopping to online payment encryption and beyond, ecommerce encompasses a wide variety of data, systems, and tools for both online buyers and sellers. Most businesses with an ecommerce presence use an ecommerce store and/or an ecommerce platform to conduct both online marketing and sales activities and to oversee logistics and fulfilment.

When it comes to ecommerce, a word that first comes to mind is growth. Ecommerce expert Gary Hoover's research shows that just in the last 14 years, the growth of ecommerce companies has skyrocketed across the board. And some merchandise lines (like clothing and beauty products in particular) have achieved a remarkable 25% average CGR between 2000-2014. U.S. Department of Commerce data shows that ecommerce sales currently average about 9.1% of total retail sales. That means there is still endless opportunity for brands to launch an ecommerce website and to expand their reach.



- There may be as many as 2.14 billion digital buyers worldwide by 2021 (eMarketer)
- U.S. ecommerce sales of apparel, footwear, and accessories projected to exceed $123M by 2022 (Statista)
- Shoppers spend 36% of their budget online on average (BigCommerce)

II. Dataset Information

The dataset procured from Kaggle has been made available from the UCI Machine Learning Repository. This is a transnational data set which contains all the actual transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers.

Attribute information

Invoice No: Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.

StockCode: Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.

Description: Product (item) name. Nominal.

Quantity: The quantities of each product (item) per transaction. Numeric.

InvoiceDate: Invoice Date and time. Numeric, the day and time when each transaction was generated.

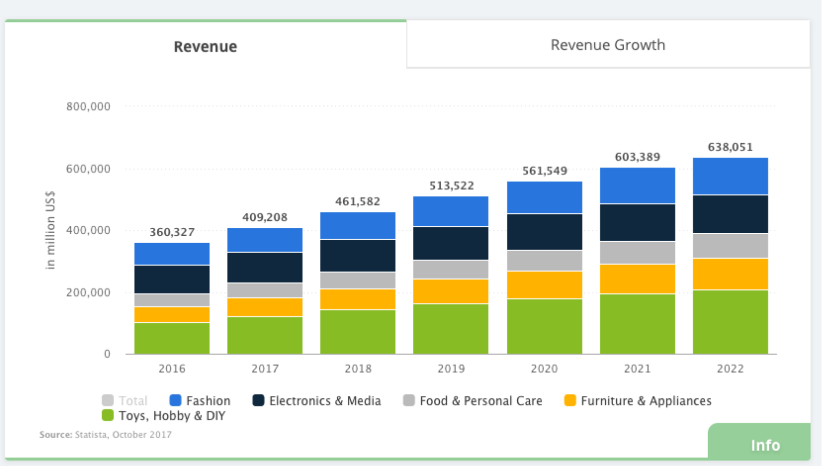
UnitPrice: Unit price. Numeric, Product price per unit in sterling.

CustomerID: Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.

Country: Country name. Nominal, the name of the country where the customer resides.

III. Problem statement

The future of ecommerce is a bright one. Ecommerce is showing a growing trend and the ecommerce revenue in the U.S. alone is expected to reach $638 million by 2022.



Thus, the **key problem we are addressing** is to recommend products to customers based on their past likings and opinions of other like-minded users and determine the products and market of focus.

IV. Objectives of the study

- To improve the sales of online e-commerce website using historical customer data to derive association rules and finding products to group together
- To generate a list of top items strongly associated with a particular basket
- To suggest new items to a customer using item-based collaborative filtering based on his transactional history
- To draw conclusions and make necessary suggestions

DATA EXPLORATION

I. *Variables exploration*

The dataset used is an Online Retail dataset obtained from the UCI Machine Learning Repository. This is a transnational dataset that contains all transactions occurring between Dec 1st 2010 to Dec 9th 2011 for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts and many customers of the company are wholesalers.

The dataset contains 541909 rows and 8 columns i.e. variables. We explore each variable individually.

```
> summary(Online_Retail)
  InvoiceNo           StockCode          Description          Quantity          InvoiceDate
 Length:541909      Length:541909      Length:541909      Min.   :-80995.00   Min.   :2010-12-01 08:26:00
 Class :character   Class :character   Class :character   1st Qu.:     1.00   1st Qu.:2011-03-28 11:34:00
 Mode  :character   Mode  :character   Mode  :character   Median :     3.00   Median :2011-07-19 17:17:00
                                                          Mean   :     9.55   Mean   :2011-07-04 13:34:57
                                                          3rd Qu.:    10.00   3rd Qu.:2011-10-19 11:27:00
                                                          Max.   : 80995.00   Max.   :2011-12-09 12:50:00

   UnitPrice          CustomerID       Country
 Min.   :-11062.06   Min.   :12346   Length:541909
 1st Qu.:     1.25   1st Qu.:13953   Class :character
 Median :     2.08   Median :15152   Mode  :character
 Mean   :     4.61   Mean   :15288
 3rd Qu.:     4.13   3rd Qu.:16791
 Max.   : 38970.00   Max.   :18287
                     NA's   :135080
> dim(Online_Retail)
[1] 541909      8
> names(Online_Retail)
[1] "InvoiceNo"   "StockCode"   "Description" "Quantity"    "InvoiceDate" "UnitPrice"   "CustomerID"
[8] "Country"
>
```

1. **InvoiceNo**:

This variable is a code which is a unique number assigned to each transaction. A transaction contains one invoice number that includes the stock codes and descriptions of items purchased, quantity of the purchased items, the date and time stamp, unit price, customer ID of the customer purchasing these items and their Country.

We find that over a period of 13 months there have been 25,900 unique transactions.

For example – Invoice number 536366 is of Customer 17850 purchasing items 22633 and 22632 on the 1st of Dec 2010 at 8:28 am.

```
> Online_Retail[Online_Retail$InvoiceNo=="536366" ,]
# A tibble: 2 x 8
  InvoiceNo StockCode Description         Quantity InvoiceDate         UnitPrice CustomerID Country
  <chr>     <chr>     <chr>                  <dbl> <dttm>                  <dbl>      <dbl> <chr>
1 536366    22633     HAND WARMER UNION JACK     6 2010-12-01 08:28:00      1.85      17850 United King~
2 536366    22632     HAND WARMER RED POLKA~     6 2010-12-01 08:28:00      1.85      17850 United King~
>
```

2. **StockCode :**

This is a unique code assigned to each product sold by the retail store for identification purposes. Our dataset has 4070 unique stock codes which means that there are 4070 unique products sold by them.

For example – the stock code 21911 is for Garden Metal Sign.

3. **Description**

It is a short description describing the product. There are 4212 unique descriptions in our dataset. From this we interpret that unique products are 4070 but descriptions are more because one product can have multiple descriptions.

For example –
22632 is a hand warmer having red polka dot design whereas 22633 is also a hand warmer but having union jack design on it.

4. **Quantity**

This tells us the number of products bought by each customer. By checking the summary of Quantity variable, we see that minimum value is shown as -80995 which is a garbage value as quantity cannot be negative. Hence, we have to clean this column.

5. **InvoiceDate**

InvoiceDate variable is a POSIXct variable that shows the Date and Time of a transaction together. There are 23260 unique InvoiceDate values. It is inconvenient to do analysis so we need to separate this variable in a date-only variable.

6. **Unit Price**

This variable tells us the price per unit of a product. Combining unit price with the quantity variable we can calculate revenue of each transaction.

7. **CustomerID**

This is a unique number assigned to each customer to track their purchases. We have 4373 unique customers.
By summarizing the data, we found that "CustomerID" has a lot of NAs in it.

```
> summary(Retail_data)
   InvoiceNo        StockCode                                 Description        Quantity
 573585 :  1114   85123A :  2313   WHITE HANGING HEART T-LIGHT HOLDER:  2369   Min.   :-80995.00
 581219 :   749   22423  :  2203   REGENCY CAKESTAND 3 TIER          :  2200   1st Qu.:     1.00
 581492 :   731   85099B :  2159   JUMBO BAG RED RETROSPOT           :  2159   Median :     3.00
 580729 :   721   47566  :  1727   PARTY BUNTING                     :  1727   Mean   :     9.55
 558475 :   705   20725  :  1639   LUNCH BAG RED RETROSPOT           :  1638   3rd Qu.:    10.00
 579777 :   687   84879  :  1502   ASSORTED COLOUR BIRD ORNAMENT     :  1501   Max.   : 80995.00
 (Other):537202   (Other):530366   (Other)                           :530315
         InvoiceDate        UnitPrice            CustomerID            Country
 10/31/11 14:41:  1114   Min.   :-11062.06   Min.   :12346    United Kingdom:495478
 12/8/11 9:28  :   749   1st Qu.:     1.25   1st Qu.:13953    Germany       :  9495
 12/9/11 10:03 :   731   Median :     2.08   Median :15152    France        :  8557
 12/5/11 17:24 :   721   Mean   :     4.61   Mean   :15288    EIRE          :  8196
 6/29/11 15:58 :   705   3rd Qu.:     4.13   3rd Qu.:16791    Spain         :  2533
 11/30/11 15:13:   687   Max.   : 38970.00   Max.   :18287    Netherlands   :  2371
 (Other)       :537202                       NA's   :135080   (Other)       : 15279
```

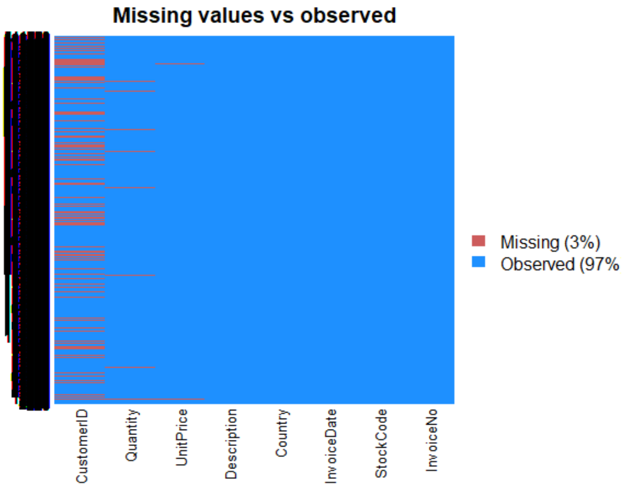
8. **Country**

This variable tells us the country from which a transaction is made. As this dataset is transnational, even though it is of a UK based retail store, we have data from 38 countries.

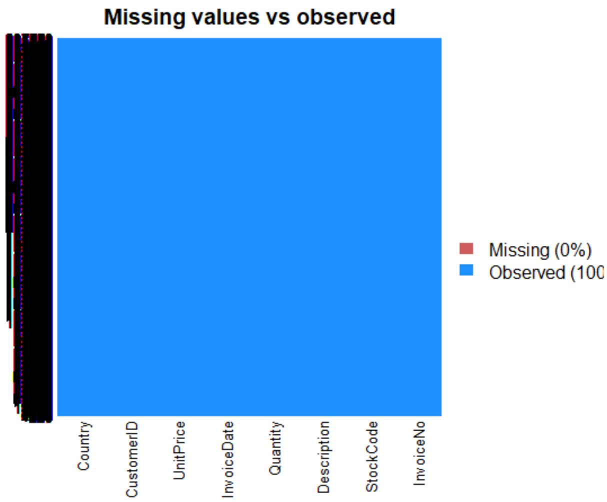
```
> length(unique(Online_Retail$Country))                     #We have data from 38 countries
[1] 38
> unique(Online_Retail$Country)
 [1] "United Kingdom"       "France"               "Australia"            "Netherlands"
 [5] "Germany"              "Norway"               "EIRE"                 "Switzerland"
 [9] "Spain"                "Poland"               "Portugal"             "Italy"
[13] "Belgium"              "Lithuania"            "Japan"                "Iceland"
[17] "Channel Islands"      "Denmark"              "Cyprus"               "Sweden"
[21] "Austria"              "Israel"               "Finland"              "Bahrain"
[25] "Greece"               "Hong Kong"            "Singapore"            "Lebanon"
[29] "United Arab Emirates" "Saudi Arabia"         "Czech Republic"       "Canada"
[33] "Unspecified"          "Brazil"               "USA"                  "European Community"
[37] "Malta"                "RSA"
>
```

II. ***Data cleaning***

After exploring our variables, we find that there are a lot of negative values and NA values. So, we recode the negative values as NA and omit the rows containing NA values altogether.



After omitting the rows containing NA values we see this –



Adding new columns:

1. We create a revenue variable which is a product of unit price and quantity.
2. We create a date only variable from InvoiceDate.
3. We further separate it into day/month/year columns.
4. We also create a time variable from InvoiceDate.
5. We derive the hour from this newly created time variable.

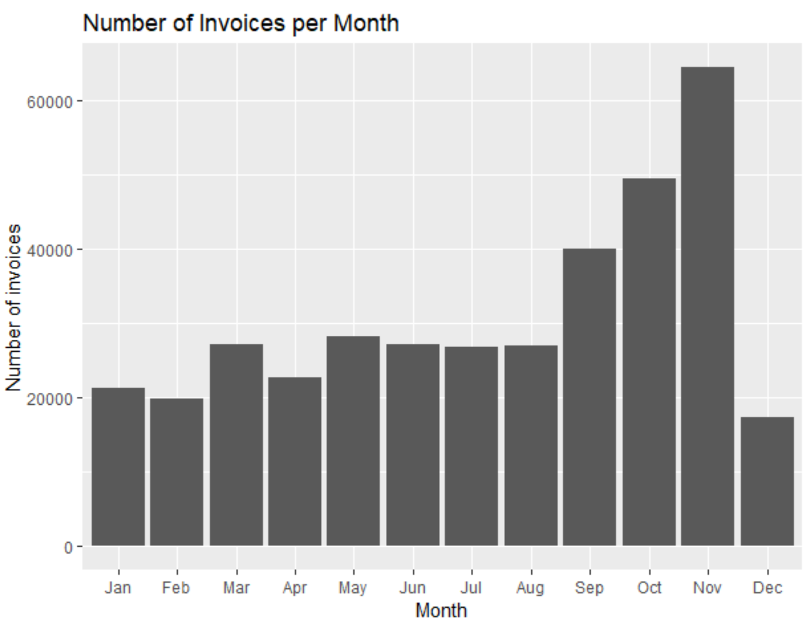
```
> head(Online_Retail_Data, n =10)
    InvoiceNo StockCode                         Description Quantity    InvoiceDate UnitPrice CustomerID
 1:    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6 12/1/2010 8:26      2.55      17850
 2:    536365     71053                 WHITE METAL LANTERN        6 12/1/2010 8:26      3.39      17850
 3:    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8 12/1/2010 8:26      2.75      17850
 4:    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6 12/1/2010 8:26      3.39      17850
 5:    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6 12/1/2010 8:26      3.39      17850
 6:    536365     22752        SET 7 BABUSHKA NESTING BOXES        2 12/1/2010 8:26      7.65      17850
 7:    536365     21730   GLASS STAR FROSTED T-LIGHT HOLDER        6 12/1/2010 8:26      4.25      17850
 8:    536366     22633              HAND WARMER UNION JACK        6 12/1/2010 8:28      1.85      17850
 9:    536366     22632           HAND WARMER RED POLKA DOT        6 12/1/2010 8:28      1.85      17850
10:    536367     84879       ASSORTED COLOUR BIRD ORNAMENT       32 12/1/2010 8:34      1.69      13047
           Country       date time year month day hours Revenue dayname
 1: United Kingdom 2010-12-01 8:26 2010    12   1     8   15.30     Wed
 2: United Kingdom 2010-12-01 8:26 2010    12   1     8   20.34     Wed
 3: United Kingdom 2010-12-01 8:26 2010    12   1     8   22.00     Wed
 4: United Kingdom 2010-12-01 8:26 2010    12   1     8   20.34     Wed
 5: United Kingdom 2010-12-01 8:26 2010    12   1     8   20.34     Wed
 6: United Kingdom 2010-12-01 8:26 2010    12   1     8   15.30     Wed
 7: United Kingdom 2010-12-01 8:26 2010    12   1     8   25.50     Wed
 8: United Kingdom 2010-12-01 8:28 2010    12   1     8   11.10     Wed
 9: United Kingdom 2010-12-01 8:28 2010    12   1     8   11.10     Wed
10: United Kingdom 2010-12-01 8:34 2010    12   1     8   54.08     Wed
```

III. Data Visualization

While exploring data we find interesting patterns:

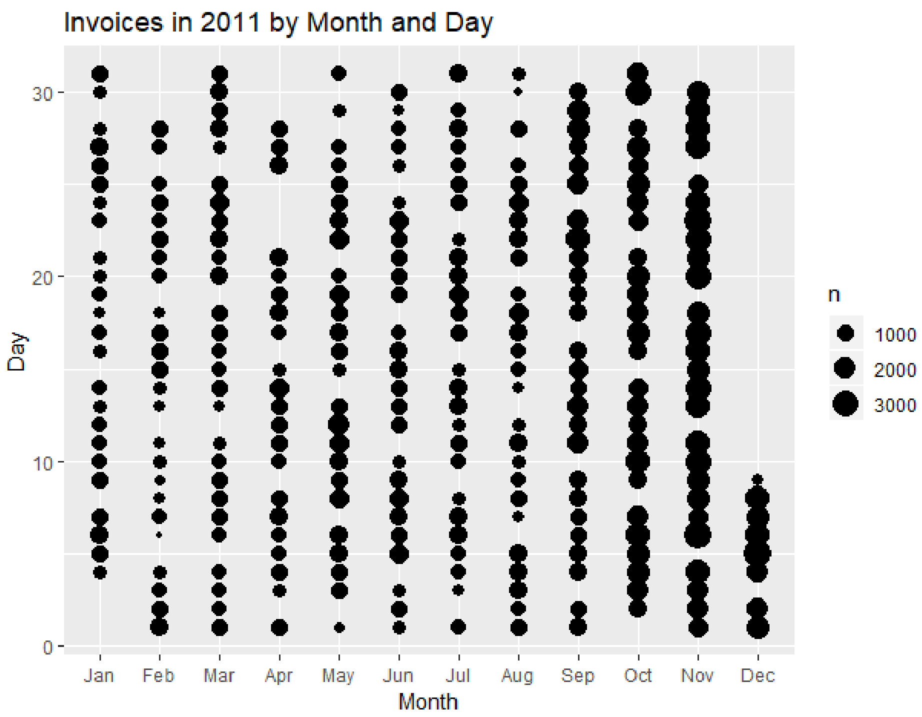
1. *Transactions made in each month*

When we plot our InvoiceNo (unique transactions) against Month, we see that highest number of orders were placed in the month of November. (We have considered the months of the year 2011 and ignored Dec 2010 as that was the only month in that year).



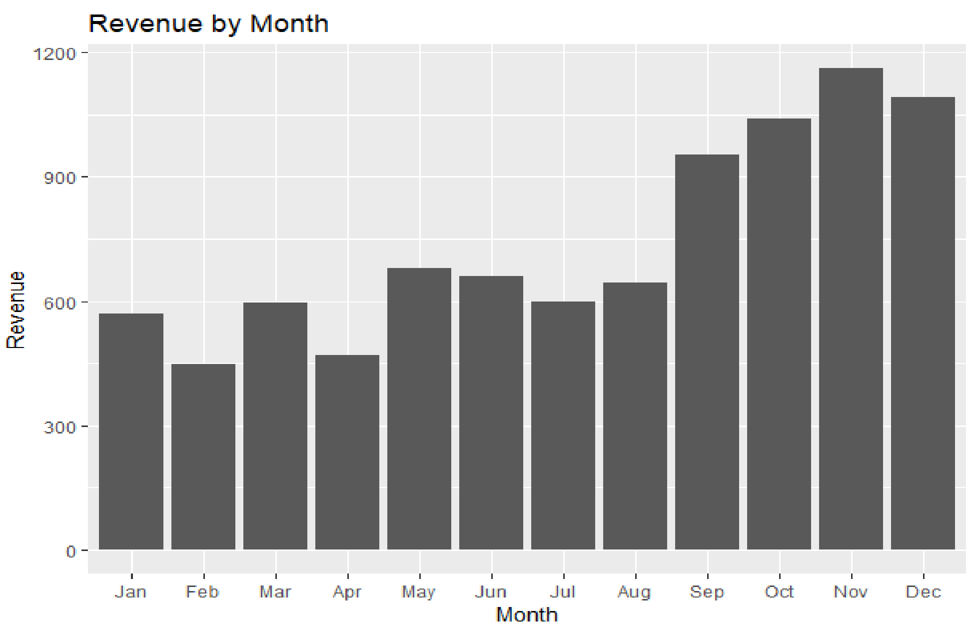
2. Transactions made by day and month

We noticed that highest orders were placed in November. But we notice that orders placed in December are almost equal to the ones placed in February even though our data has just 9 days of December.



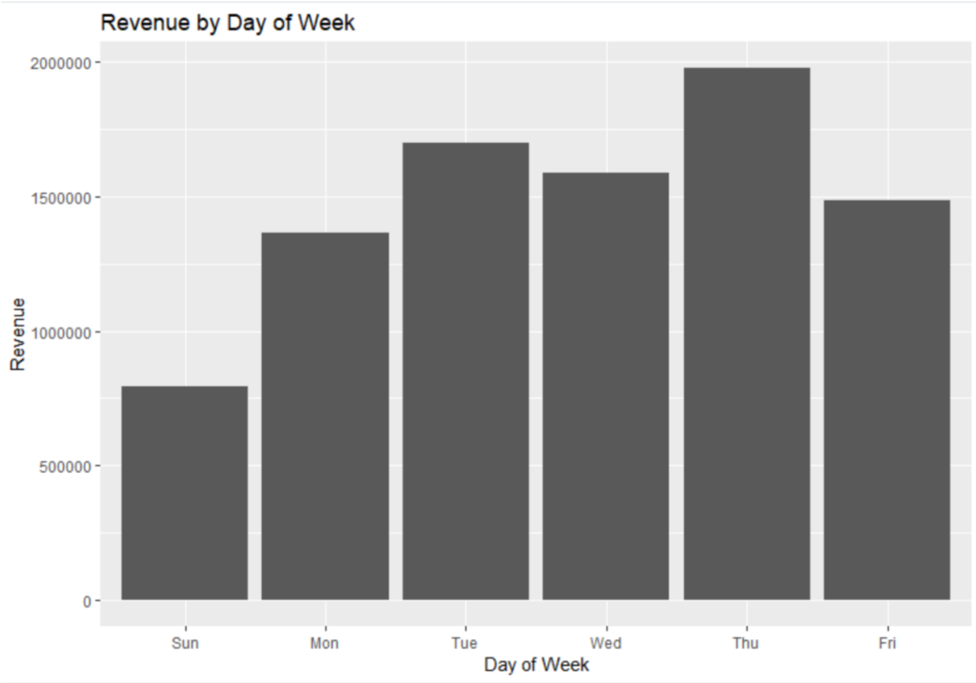
3. Revenue generated in each month

As the highest number of orders were places in November, we know the revenue of November is going to be highest. But although lowest transactions were made in the month of December, the revenue of December is the second highest, even though the data of December 2011 is only for 9 days. This tells us that although less items were purchased in December, either expensive products were sold, or bulk quantity was purchased.



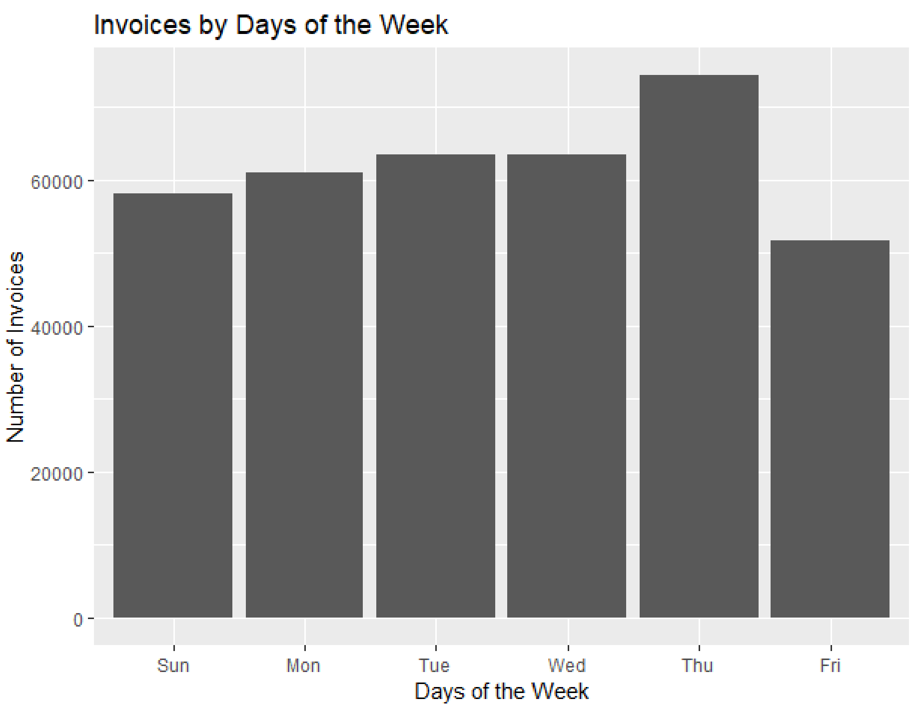
4. *Revenue generated on each day of the week*

We notice that even though almost equal number of transactions were made on the first 4 days of the week, their revenues are quite different from each other. This is because different quantities of items were sold. Highest revenue was generated on a Thursday (amounting to 1,976,859) and lowest revenue was generated on a Sunday (amounting to 792,514).



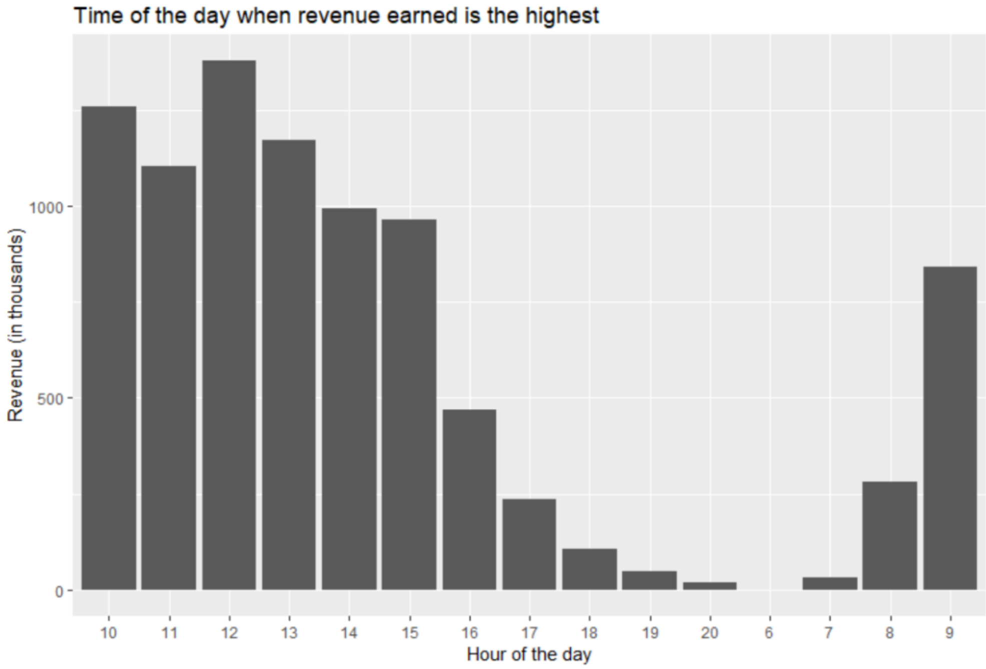
5. *Transactions made on various days of the week*

We notice that highest orders were placed on Thursday. You notice that no orders were placed on Saturday which means that Saturday is an off day for the store.



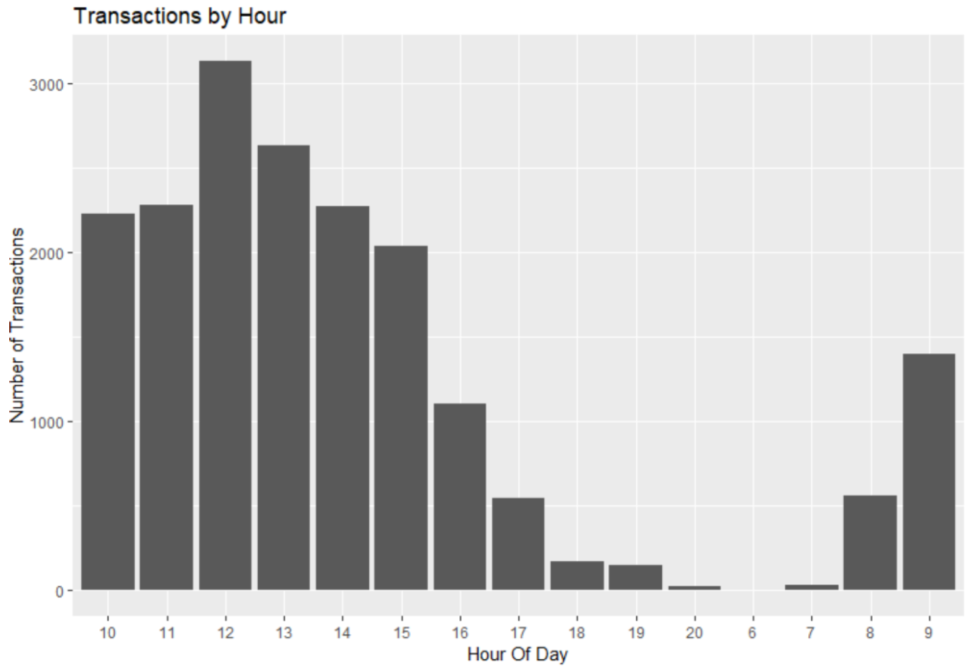
6. Time of the day which recorded the highest revenue

On plotting the hour of the day against the revenue, it can be observed that the online retail store has generated most of its revenues between hour 10 and hour 17 of the day. The highest revenue was generated during the 12th hour (amounting to 1,378,571), while negligible revenue was generated during the 6th hour (amounting to 4.25).



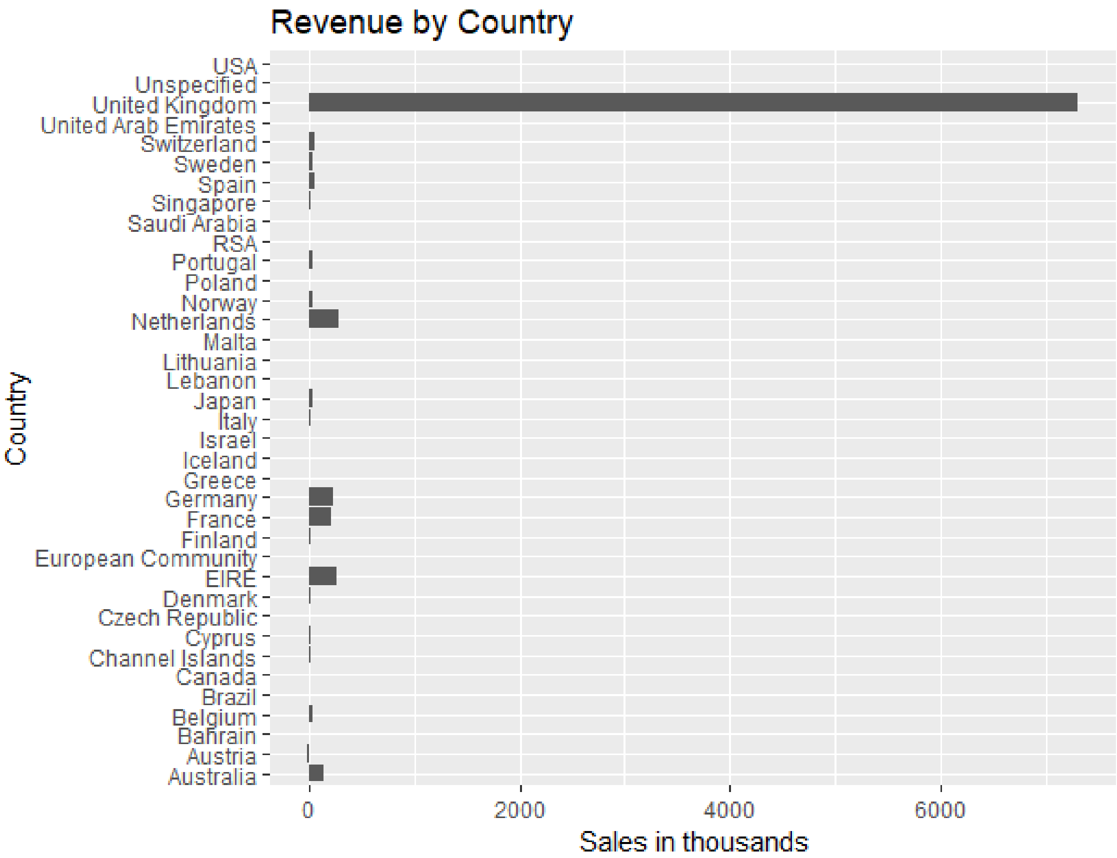
7. Time of the day that recorded the highest number of transactions

On plotting the hour of the day against the InvoiceNo, we observe that the maximum transactions occurred during the 12th hour while with a majority of the transactions occurring between the 10th and 17th hour. We can infer that most of the customers visit the website between these hours and shop.



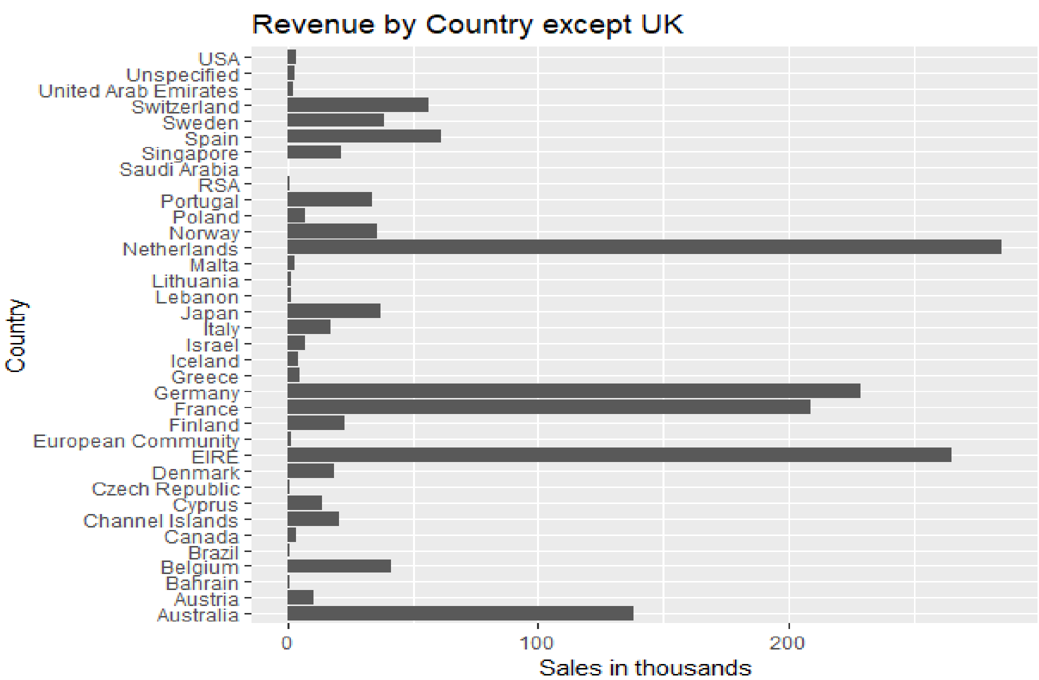
8. Revenue generated by each country

We see that highest revenue is generated by UK which is obvious since our retail store is UK based.



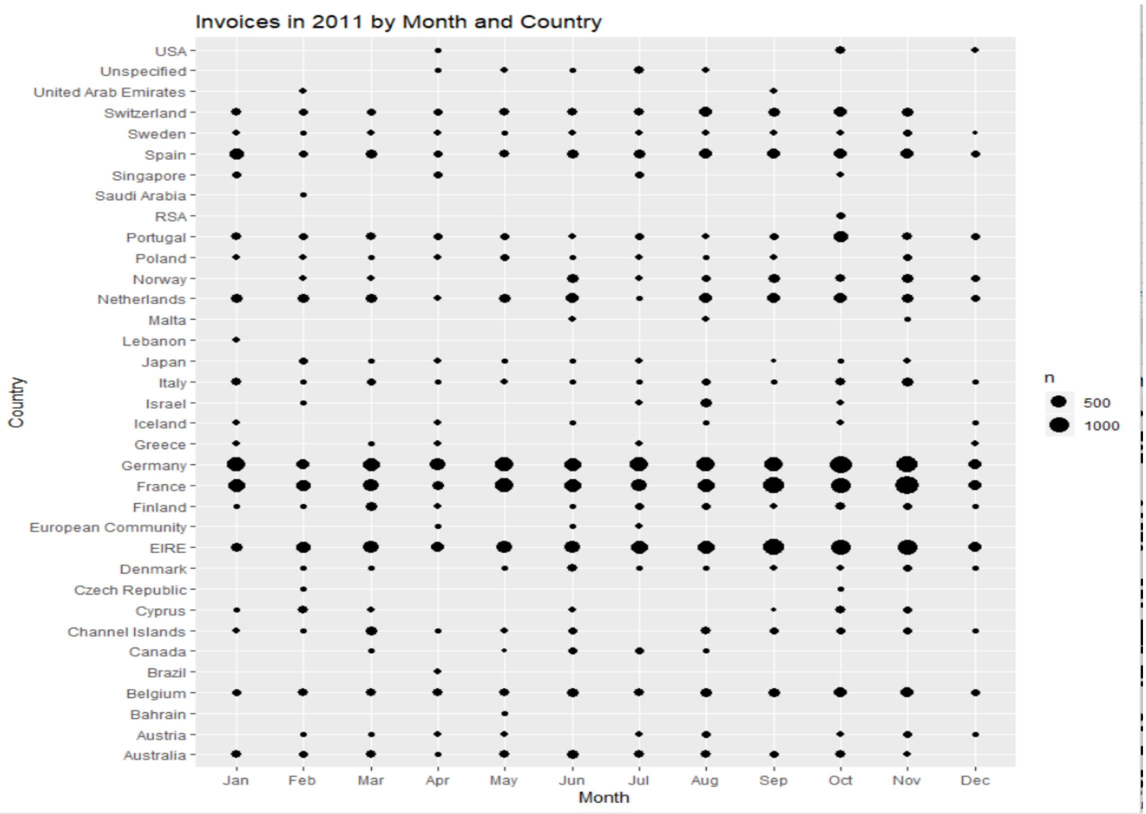
9. Revenue generated by countries except UK

Once we filter out the country UK, we notice that maximum revenue is generated by Netherlands, EIRE, Germany, France and Australia.



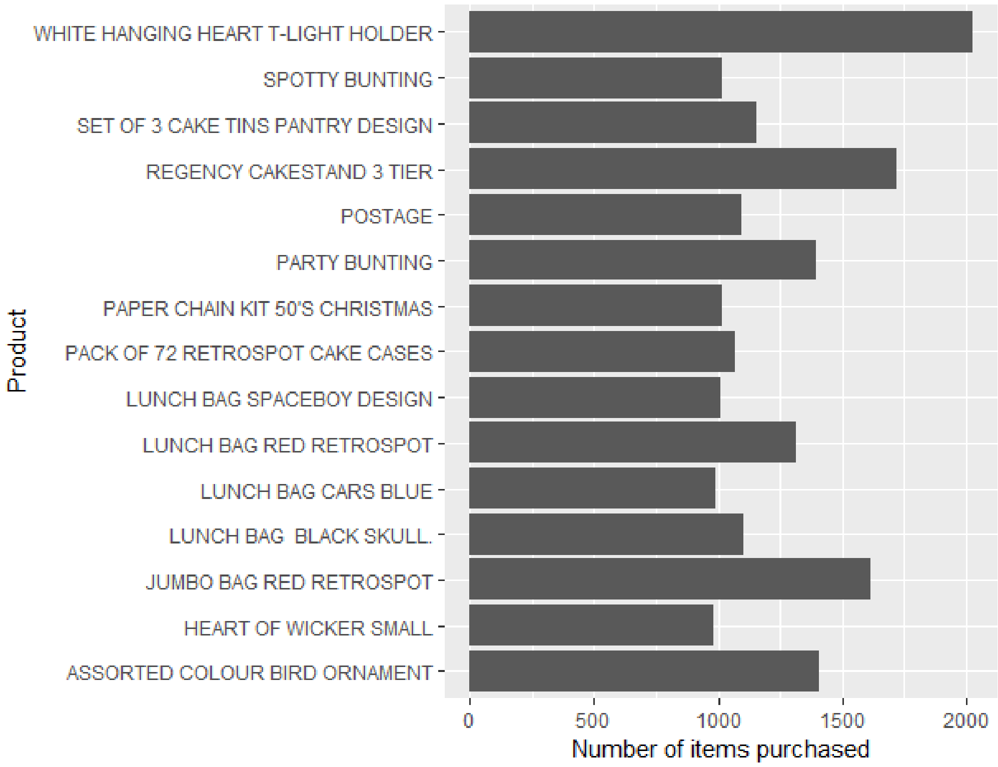
10. Revenue generated by each country (except UK) monthly

We see that Germany and France generate a lot of revenue almost throughout the year whereas countries like Lebanon, Israel, Iceland, Greece and a few others don't have transactions every month and hence don't generate revenue each month.



11. Most popular products

This retail store sells over 4000 products, but we compile a list of their top 15 products



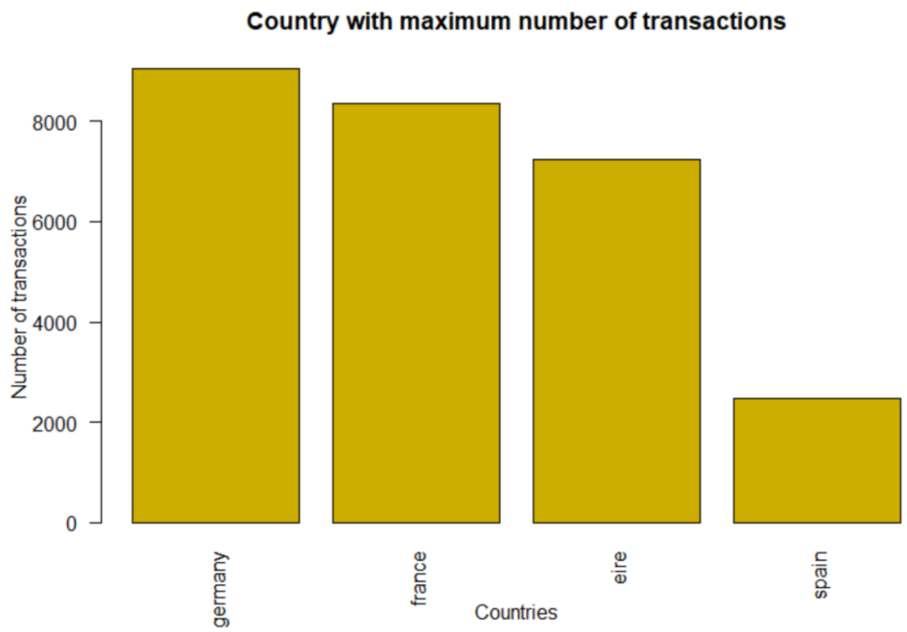
12. Country with the highest number of transactions

On generating a word cloud of the countries based on the total number of transactions in each country, we conclude that United Kingdom had the highest number of transactions (i.e. InvoiceNo) with the number of transactions being 35,4321.

13. Top 5 countries with the highest number of transactions

We know that United States had the highest number of transactions. Upon filtering UK from the list, we find that Germany, France, Eire and Spain recorded the highest number of transactions following United Kingdom.

Thus, the top 5 countries according to the number of transactions is United Kingdom, Germany, France, Eire and Spain.



ALGORITHM 1: MARKET BASKET ANALYSIS

Market Basket Analysis (MBA) also known as association rule learning or affinity analysis, is a data mining technique that can be used in various fields, such as marketing, bioinformatics, education field, nuclear science etc. The main aim of MBA in marketing is to provide the information to the retailer for an understanding of the purchase behaviors of buyers, which can help the retailer in making correct decisions.
There are various algorithms available for performing the MBA. One of the famous and very powerful algorithm is the *apriori* algorithm. Market Basket Analysis is one of the major techniques used by big retailing stores to uncover relationships between items. It works by looking on the basket of the customer to find the patterns of the items that occur together frequently in transactions. In simple words, we can say that it helps the retailers to figure out what two/more product are frequently bought together by the customers.
These Association Rules are mainly for analyzing retail basket or transaction data and are intended to identify strong rules discovered in transaction data using measures of interestingness, based on the concept of strong rules.

The three major parameters in the Market Basket analysis are:

1. *Support*:

It is the probability of buying two product(X,Y) together in the sample of size N.
Support = Freq (X,Y)/N

2. *Confidence*:

It is the relative probability of buying two product(X,Y) together given that the customer is buying product X.
Confidence=Freq(X,Y)/Freq(X)

3. *Lift*:

Freq(X,Y)/Freq(X)*(Freq(Y)/N)It is a parameter to judge the result of the analysis. If Lift>1 then it is statistically a good predictor to rely upon.

Note:
Sometimes we see different types of results in an analysis. So, we divide the results into three categories.

- Actionable Rule: The rules those make sense and we can work to implement them in the market.
  (This is the type of results that we are interested in)
- Trivial Rule: Very obvious results (Unworthy information).
- Inexplicable Rule: Very complex rules that we cannot explain them practically, they might have occurred due to some impurities in data or some small-time random behaviour of customers.

After having close look on the data set, we started cleaning the dataset and transformed into the required format, the transaction form, to perform the ***apriori algorithm***.

***Raw data to transactional data*:**

We have seen the data structure and summary and we know that for transactions we only need "InvoiceNo" and "Description", and these two fields do not have any missing or NA entry. So, the data is cleaned already.

Using following code we converted the data to Transactional data:

```
write.csv(Retail_data_wip2, file = "Retail_trans.csv")
trans = read.transactions("Retail_trans.csv", format = "single", sep = ",", cols = c("InvoiceNo", "Description"))
inspect(trans[1:5])
```

```
> inspect(trans[1:5])
    items                                   transactionID
[1] {CREAM CUPID HEARTS COAT HANGER,
     GLASS STAR FROSTED T-LIGHT HOLDER,
     KNITTED UNION FLAG HOT WATER BOTTLE,
     RED WOOLLY HOTTIE WHITE HEART.,
     SET 7 BABUSHKA NESTING BOXES,
     WHITE HANGING HEART T-LIGHT HOLDER,
     WHITE METAL LANTERN}                          536365
[2] {HAND WARMER RED POLKA DOT,
     HAND WARMER UNION JACK}                       536366
[3] {ASSORTED COLOUR BIRD ORNAMENT,
     BOX OF 6 ASSORTED COLOUR TEASPOONS,
     BOX OF VINTAGE ALPHABET BLOCKS,
     BOX OF VINTAGE JIGSAW BLOCKS ,
     DOORMAT NEW ENGLAND,
     FELTCRAFT PRINCESS CHARLOTTE DOLL,
     HOME BUILDING BLOCK WORD,
     IVORY KNITTED MUG COSY ,
     LOVE BUILDING BLOCK WORD,
     POPPY'S PLAYHOUSE BEDROOM ,
     POPPY'S PLAYHOUSE KITCHEN,
     RECIPE BOX WITH METAL HEART}                  536367
[4] {BLUE COAT RACK PARIS FASHION,
     JAM MAKING SET WITH JARS,
     RED COAT RACK PARIS FASHION,
     YELLOW COAT RACK PARIS FASHION}               536368
[5] {BATH BUILDING BLOCK WORD}                     536369
```

We can see that data has transformed to Transactions. In above image we can see the first 5 Transactions in the data.

***Running the algorithm*:**

Now we have transformed the data to its desired format, we can perform the apriori algorithm.

With support = 0.001 (we are only interested in the frequently purchased products) and confidence = 0.8

```r
## running the model
library(arules)
library(arulesViz)

rules = apriori(trans,parameter = list(supp = 0.001,conf=0.8))
rules<- sort(rules,by= "confidence",decreasing = "T")
inspect(rules[1:10])
```

```
> summary(rules)
set of 30598898 rules

rule length distribution (lhs + rhs):sizes
       2        3        4 
     160   509093 30089645 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   4.000   4.000   3.983   4.000   4.000 

summary of quality measures:
    support           confidence          lift             count       
 Min.   :0.001004   Min.   :0.8000   Min.   :  9.001   Min.   : 26.00  
 1st Qu.:0.001042   1st Qu.:0.8387   1st Qu.: 25.233   1st Qu.: 27.00  
 Median :0.001120   Median :0.8824   Median : 39.645   Median : 29.00  
 Mean   :0.001168   Mean   :0.8905   Mean   : 55.175   Mean   : 30.24  
 3rd Qu.:0.001236   3rd Qu.:0.9355   3rd Qu.: 69.987   3rd Qu.: 32.00  
 Max.   :0.024865   Max.   :1.0000   Max.   :647.500   Max.   :644.00  

mining info:
  data ntransactions support confidence
 trans         25900   0.001        0.8
```

Here, we ran the apriori model and we found 160 rules with length of 2 (two products involved), 509093 rules for length 3 and 30089645 rules for length 4.

ALGORITHM 2: ITEM-BASED COLLABORATIVE FILTERING

Recommender systems apply statistical and knowledge discovery techniques to the problem of making product recommendations based on previously recorded data. Such recommendations can help to improve the conversion rate by helping the customer to find products she/he wants to buy faster and promote cross-selling by suggesting additional products. Recommender systems are typically divided into 2 categories:

1) ***Content Based***
2) ***Collaborative Filtering***

Content-based approaches are based on the idea that if we can elicit the preference structure of a customer (user) concerning product (item) attributes then we can recommend items which rank high for the user's most desirable attributes.

Collaborative Filtering focuses on the idea that given rating data by many users for many items (e.g., 1 to 5 stars for movies elicited directly from the users), one can predict a user's rating for an item not known to her or him **(*User Based*)** or create for a user a so called top-N lists of recommended item **(*Item Based*).**

Our project focuses on Item based Collaborative filtering, which recommends Top 5 products to a customer.

Item Based Collaborative Filtering (IBCF)

A model-based approach which produces recommendations based on the relationship between items inferred from the rating matrix. The assumption behind this approach is that users will prefer items that are similar to other items they like.

We would require *Invoice No*, *StockCode* and *Quantity* to build an IBCF model. By analyzing our dataset, we observe that there are few values of the variable *Quantity* and *Unit price* that are negative. Using the negative values wouldn't give us an accurate model. Hence, we replace the negative values with NA values and omit these NA values. This data preprocessing step has been explained in the Data Exploration section of the project.

Steps in ICBF Algorithm

- ***Creation of Buying Matrix:*** Once the data is free from negative values, we create a matrix that determines the total quantity of every item bought by every customer. This step is implemented by converting the data in long format to wide format. Wide format data is much easier to interpret and read.

- ***Conversion of wide format to sparse matrix:*** This sparse matrix includes 1's and 0's, where 1 means that we inferred that the customer has a preference for an item and 0 means that either the user does not like the item or does not know about it. A matrix with real valued ratings can be transformed into a 0-1 matrix with *binarize()* and a user specified threshold *(min_ratings*) on the raw or normalized ratings.

- ***Spitting the Dataset:*** This step involves splitting the data into training and testing. We randomly split the data into 80% training and 20% testing.

- ***Estimating the parameters for the model***: The model-building step consists of calculating a similarity matrix containing all item-to item similarities using a given similarity measure. The most popular are the *Pearson correlation* and *Cosine similarity*. All pairwise similarities are stored in a $n \times n$ similarity matrix S. To reduce the model size to $n \times k$ with $k < n$, for each item only a list of the k most similar items and their similarity values are stored. However, if we assume all zeroes are missing values, then this lead to the problem that we cannot compute similarities using Pearson correlation or Cosine similarity since the not missing parts of the vectors only contains ones. A similarity measure which only focuses on matching ones and thus prevents the problem with zeroes is the *Jaccard index*. The Jaccard index can be used between users for user-based filtering and between items for item-based filtering.

- ***Building the Model:*** The actual implementations for the recommendation algorithms are managed using the registry mechanism provided by package registry. The registry is called *recommenderRegistry* and it stores recommendation method names and a short description. The registry mechanism is hidden from the user and the creator function *Recommender()* uses it in the background to map a recommender method name to its implementation. However, the registry can be directly queried by *recommenderRegistry$get_entries()*. Let's have a look at the default parameters for the method IBCF:

```
> recommender_models$IBCF_binaryRatingMatrix$parameters
$`k`
[1] 30

$method
[1] "Jaccard"

$normalize_sim_matrix
[1] FALSE

$alpha
[1] 0.5
```

Class Recommender implements the data structure to store recommendation models. The creator method (shown below) takes data as a *ratingMatrix*, a method name and some optional parameters for the method and returns a Recommender object.

Recommender(data, method, parameter = NULL)

- ***Prediction using the testing data:*** As described, we randomly split the data set and use only 20% for testing. Return value of prediction is top-N-List of recommendation item for each user in test dataset. We can predict top-N recommendations for active users using:

predict(object, newdata, n=10, type=c("topNList","ratings","ratingMatrix"), ...)

Predict can return either top-N lists (default setting) or predicted ratings. *Object* is the recommender object, *newdata* is the data for the active users (Testing data). For top-N lists *n* is the maximal number of recommended items in each list and *predict()* will return an objects of class *topNList* which contains one top-N list for each active user.

RESULTS AND INTERPRETATION

1. Market Basket Analysis

The summary of the rules gives us some very interesting information:

The number of rules: 30598898

The distribution of rules by length: a length of 4 items has the most rules.

The summary of quality measures: ranges of support, confidence, and lift.

The information on data mining: total data mined, and the minimum parameters were set earlier.

Here are the top 10 rules sorted by lift:

>Inspect (rules [1:10])

```
> inspect(rules[1:10])
     lhs                                   rhs                                   support confidence     lift count
[1]  {BEADED CRYSTAL HEART PINK ON STICK,
      FRYING PAN BLUE POLKADOT,
      HANGING WOOD AND FELT HEART}      => {GREETING CARD, TWO SISTERS.}       0.001003861          1 647.5000    26
[2]  {BOTTLE BAG RETROSPOT ,
      GREEN ENAMEL+GLASS HAIR COMB}     => {FOLK FELT HANGING MULTICOL GARLAND} 0.001003861          1 424.5902    26
[3]  {GREETING CARD, OVERCROWDED POOL.,
      PINK FAIRY CAKE CUSHION COVER,
      SKULL SHOULDER BAG}               => {FOLK FELT HANGING MULTICOL GARLAND} 0.001042471          1 424.5902    27
[4]  {FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.,
      POCKET BAG PINK PAISELY BROWN SPOT} => {FOLK FELT HANGING MULTICOL GARLAND} 0.001042471          1 424.5902    27
[5]  {BISCUITS SMALL BOWL LIGHT BLUE,
      FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.}  => {FOLK FELT HANGING MULTICOL GARLAND} 0.001003861          1 424.5902    26
[6]  {DECORATION  BUTTERFLY  MAGIC GARDEN,
      FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.}  => {FOLK FELT HANGING MULTICOL GARLAND} 0.001042471          1 424.5902    27
[7]  {BIRD DECORATION GREEN POLKADOT,
      FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.}  => {FOLK FELT HANGING MULTICOL GARLAND} 0.001003861          1 424.5902    26
[8]  {BOTTLE BAG RETROSPOT ,
      FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.}  => {FOLK FELT HANGING MULTICOL GARLAND} 0.001042471          1 424.5902    27
[9]  {FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.,
      POCKET BAG BLUE PAISLEY RED SPOT}  => {FOLK FELT HANGING MULTICOL GARLAND} 0.001003861          1 424.5902    26
[10] {FOLDING UMBRELLA RED/WHITE POLKADOT,
      GREETING CARD, OVERCROWDED POOL.,
      JUMBO BAG CHARLIE AND LOLA TOYS}   => {FOLK FELT HANGING MULTICOL GARLAND} 0.001003861          1 424.5902    26
```

The interpretations are quite intuitive but also very much worth notifying:

Rule No.1: 100% customers bought {BEADED CRYSTAL HEART PINK ON STICK, FRYING PAN BLUE POLKADOT, HANGING WOOD AND FELT HEART} and {GREETING CARD, TWO SISTERS.} together. A good explanation for this is that people purchase these decoration products, greeting cards and cooking wares together to get prepared for the holidays. Notice how most transactions are made in November, this rule probably implies the fact that people buy these type of "Christmas" products together to get ready for it.

Rule No.2: 100% customers bought

{BOTTLE BAG RETROSPOT ,

GREEN ENAMEL+GLASS HAIR COMB} and {FOLK FELT HANGING MULTICOL GARLAND} together. A good explanation for this is females who love beauty care products also have a tendency to buy some decoration products together.

Similar patterns are observed for rules 3 to rules 10, people like to purchase different types of decoration products, toys and holiday products together.

The reason why we sort the rules by lift is because that the lift indicates how many times more our rule is likely to exist than a random purchase of the RHS products with everything else.

2. *Collaborative Filtering:*

Let's look at an example executed by our recommender model. We predicted the Top 5 Products for the customer with the customer id = 12361.

The top 5 products are:

```
> itemCode[vvv]
   StockCode             Description
1:     20728      LUNCH BAG CARS BLUE
2:     20727 LUNCH BAG  BLACK SKULL.
3:     22384 LUNCH BAG PINK POLKADOT
4:     23206  LUNCH BAG APPLE DESIGN
5:     22383  LUNCH BAG SUKI  DESIGN
```

The overall purchase made by the customer is:

```
   StockCode                         Description V1
1      20725              LUNCH BAG RED SPOTTY 10
2      20725            LUNCH BAG RED RETROSPOT 10
3      20726                 LUNCH BAG WOODLAND 10
4      22326 ROUND SNACK BOXES SET OF4 WOODLAND  6
5      22328  ROUND SNACK BOXES SET OF 4 FRUITS  6
6      22382           LUNCH BAG SPACEBOY DESIGN 10
7      22555           PLASTERS IN TIN STRONGMAN 12
8      22629                 SPACEBOY LUNCH BOX 12
9      22630                 DOLLY GIRL LUNCH BOX 12
10     22631              CIRCUS PARADE LUNCH BOX 12
11      POST                             POSTAGE  1
```

By observing the above purchases made, the recommended items closely resemble the purchases made by the customer.

SUGGESTIONS AND FURTHER IMPROVEMENTS

In market basket analysis, once the rules are generated, for the products that are included in each single rule have to be inspected and then give customers recommendation links saying that these products are commonly bought together for customer's convenience. Market Basket analysis also helps in boosting not only sales but also advertisement.

We believe that Top N recommender algorithms implemented in this project can be improved by combining elements from both the user and item-based approaches. User based approach for dynamically computing a neighbourhood of similar user are better suited to provide truly personalized information. On the other hand, the item-based approaches by directly computing the similarity between items appear to compute more accurate recommendations. One potential limitation to item-based approach is that the item to item similarity may not be able to provide sufficient degree of personalization. In this case, an approach that first identifies a reasonably large neighbourhood of similar items and then uses this subset to derive the item-based recommendation model may be able to combine the best of both the approaches and perform even better recommendations.

CONCLUSION

In this project, we analysed an Online Retail dataset by performing data cleaning, data exploration, implementation of recommender systems and interpretation of the results. In data exploration, we explored the various variables of the dataset and plotted graphs to get interesting observations. In market basket analysis, we first converted the dataset into the transaction form which lets us perform the algorithm on it. Then we used the Apriori algorithm to find out the association rules setting support to 0.001 and confidence to 0.8. The results showed worthy patterns that cannot be easily speculated if not using the algorithm and are also explainable with theories. And this allowed us to give our recommendations. In collaborative filtering, we implemented a Top N recommender algorithm that uses item to item similarity to compute the recommendations. Our results demonstrated how we could predict the Top 5 items to customer based on the transactional history of the customer. We incorporated the Jaccard Index method and avoided the use of Pearson and Cosine to avoid the treatment of zeroes as missing values. The item-based CF method on an average provides more accurate recommendations than the traditional user-based CF. The proposed algorithms are substantially faster allowing real time recommendations independent of the size of the user item matrix. Based on our findings we offered suggestions that can be used by any ecommerce business.

Appendix 1:

Data Exploration in R Studio- Part 1

```
library(readxl)

# Loading the Dataset

Online_Retail <- read_excel("C:/Users/anush/Desktop/BA with R/Project/Online R
etail.xlsx")

View(Online_Retail)

str(Online_Retail)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    541909 obs. of  8 variables:

##  $ InvoiceNo  : chr  "536365" "536365" "536365" "536365" ...

##  $ StockCode  : chr  "85123A" "71053" "84406B" "84029G" ...

##  $ Description: chr  "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANT
ERN" "CREAM CUPID HEARTS COAT HANGER" "KNITTED UNION FLAG HOT WATER BOTTLE" ..
.

##  $ Quantity   : num  6 6 8 6 6 2 6 6 6 32 ...

##  $ InvoiceDate: POSIXct, format: "2010-12-01 08:26:00" "2010-12-01 08:26:00
" ...

##  $ UnitPrice  : num  2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...

##  $ CustomerID : num  17850 17850 17850 17850 17850 ...

##  $ Country    : chr  "United Kingdom" "United Kingdom" "United Kingdom" "Un
ited Kingdom" ...
```

```
summary(Online_Retail)
```

```
##   InvoiceNo           StockCode          Description       

##  Length:541909       Length:541909      Length:541909     

##  Class :character    Class :character   Class :character  

##  Mode  :character    Mode  :character   Mode  :character  

##   Quantity              InvoiceDate                    UnitPrice       

##  Min.   :-80995.00   Min.   :2010-12-01 08:26:00   Min.   :-11062.06  

##  1st Qu.:     1.00   1st Qu.:2011-03-28 11:34:00   1st Qu.:     1.25  

##  Median :     3.00   Median :2011-07-19 17:17:00   Median :     2.08  

##  Mean   :     9.55   Mean   :2011-07-04 13:34:57   Mean   :     4.61  

##  3rd Qu.:    10.00   3rd Qu.:2011-10-19 11:27:00   3rd Qu.:     4.13  

##  Max.   : 80995.00   Max.   :2011-12-09 12:50:00   Max.   : 38970.00  

##                                                                       

##    CustomerID       Country         

##  Min.   :12346    Length:541909     

##  1st Qu.:13953    Class :character  

##  Median :15152    Mode  :character  

##  Mean   :15288
```

```
##   3rd Qu.:16791
##   Max.   :18287
##   NA's   :135080
dim(Online_Retail)
## [1] 541909      8
names(Online_Retail)
## [1] "InvoiceNo"   "StockCode"   "Description" "Quantity"    "InvoiceDate"
## [6] "UnitPrice"   "CustomerID"  "Country"
#Explore individual variables
length(unique(Online_Retail$InvoiceNo)) #we see there are 25900 unique invoice
s i.e transactions made
## [1] 25900
Online_Retail[Online_Retail$InvoiceNo=="536366" ,] #Invoice number 536366 show
s customer 17850 bought 2 items
## # A tibble: 2 x 8
##   InvoiceNo StockCode Description Quantity InvoiceDate         UnitPrice
##   <chr>     <chr>     <chr>          <dbl> <dttm>                  <dbl>
## 1 536366    22633     HAND WARME~        6 2010-12-01 08:28:00      1.85
## 2 536366    22632     HAND WARME~        6 2010-12-01 08:28:00      1.85
## # ... with 2 more variables: CustomerID <dbl>, Country <chr>
length(unique(Online_Retail$StockCode)) #ProductID. There are 4070 products so
ld by this company
## [1] 4070
length(unique(Online_Retail$Description)) #Description of the product sold. de
scriptions are more than products as they can have multiple colors or sizes
## [1] 4212
summary(Online_Retail$Quantity)
##       Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -80995.00      1.00      3.00      9.55     10.00  80995.00
length(unique(Online_Retail$InvoiceDate)) #Invoice Date is the time and date o
f the transaction
## [1] 23260
length(unique(Online_Retail$CustomerID)) #CustomerID we have 4272 unique custo
mers
## [1] 4373
length(unique(Online_Retail$Country)) #We have data from 38 countries
## [1] 38
unique(Online_Retail$Country)
##  [1] "United Kingdom"       "France"               "Australia"
##  [4] "Netherlands"          "Germany"              "Norway"
##  [7] "EIRE"                 "Switzerland"          "Spain"
## [10] "Poland"               "Portugal"             "Italy"
```

```
## [13] "Belgium"              "Lithuania"            "Japan"
## [16] "Iceland"              "Channel Islands"      "Denmark"
## [19] "Cyprus"               "Sweden"               "Austria"
## [22] "Israel"               "Finland"              "Bahrain"
## [25] "Greece"               "Hong Kong"            "Singapore"
## [28] "Lebanon"              "United Arab Emirates" "Saudi Arabia"
## [31] "Czech Republic"       "Canada"               "Unspecified"
## [34] "Brazil"               "USA"                  "European Community"
## [37] "Malta"                "RSA"
```

```r
#We check our data for NA's and negative values
sum(is.na(Online_Retail))  #We have 1,36,534 NA values in our dataset
```

```
## [1] 136534
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
Online_Retail <- Online_Retail %>% mutate(Quantity = replace(Quantity, Quantit
y<=0, NA), UnitPrice = replace(UnitPrice, UnitPrice<=0, NA)) #Replace -ve valu
es with NA too

sum(is.na(Online_Retail))

#Now NA values have increased to 149675
```

```
## [1] 149675
```

```r
colSums(is.na(Online_Retail))  #We see which columns have NA values
```

```
##   InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice
##           0           0        1454       10624           0        2517
##  CustomerID     Country
##      135080           0
```

```r
library(Amelia)
```

```
## Loading required package: Rcpp
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.5, built: 2018-05-07)
## ## Copyright (C) 2005-2018 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
missmap(Online_Retail, main = "Missing values vs observed")
## Warning in if (class(obj) == "amelia") {: the condition has length > 1 and
## only the first element will be used
## Warning: Unknown or uninitialised column: 'arguments'.
## Warning: Unknown or uninitialised column: 'arguments'.
## Warning: Unknown or uninitialised column: 'imputations'.
Online_Retail1 <- na.omit(Online_Retail) #We create Online_Retail1 dataset aft
er cleanign our data
View(Online_Retail1)
colSums(is.na(Online_Retail1))    #We see our dataset has been cleared of NA v
alues
##   InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice
##           0           0           0           0           0           0
##  CustomerID     Country
##           0           0
dim(Online_Retail1)     #We are left with 397,884 rows after cleaning
## [1] 397884      8
#As part of our cleaning we add a few columns
library(lubridate)
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##     date
class(Online_Retail1$InvoiceDate)
#POSIXcT stores both date and time of a timezone
## [1] "POSIXct" "POSIXt"
head(Online_Retail1$InvoiceDate)
## [1] "2010-12-01 08:26:00 UTC" "2010-12-01 08:26:00 UTC"
## [3] "2010-12-01 08:26:00 UTC" "2010-12-01 08:26:00 UTC"
## [5] "2010-12-01 08:26:00 UTC" "2010-12-01 08:26:00 UTC"
Date_Only <- as.Date(Online_Retail1$InvoiceDate)  #We separate date from time
Online_Retail1$Date_Only <- as.Date(Online_Retail1$InvoiceDate) #create new da
te only column
head(Date_Only)
## [1] "2010-12-01" "2010-12-01" "2010-12-01" "2010-12-01" "2010-12-01"
## [6] "2010-12-01"
Online_Retail1$Year <- year(Online_Retail1$Date_Only)
Online_Retail1$Month <- month(Online_Retail1$Date_Only, label=T)
Online_Retail1$Day <- day(Online_Retail1$Date_Only)
```

```
Online_Retail1$Revenue <- Online_Retail1$Quantity * Online_Retail1$UnitPrice
    #Create column Revenue = Quantity*Unit Price

head(Online_Retail1)
## # A tibble: 6 x 13
##   InvoiceNo StockCode Description Quantity InvoiceDate         UnitPrice
##   <chr>     <chr>     <chr>          <dbl> <dttm>                  <dbl>
## 1 536365    85123A    WHITE HANG~        6 2010-12-01 08:26:00      2.55
## 2 536365    71053     WHITE META~        6 2010-12-01 08:26:00      3.39
## 3 536365    84406B    CREAM CUPI~        8 2010-12-01 08:26:00      2.75
## 4 536365    84029G    KNITTED UN~        6 2010-12-01 08:26:00      3.39
## 5 536365    84029E    RED WOOLLY~        6 2010-12-01 08:26:00      3.39
## 6 536365    22752     SET 7 BABU~        2 2010-12-01 08:26:00      7.65
## # ... with 7 more variables: CustomerID <dbl>, Country <chr>,
## #   Date_Only <date>, Year <dbl>, Month <ord>, Day <int>, Revenue <dbl

View(Online_Retail1)
#After cleaning the data we plot it


##Plot 1 - Month vs Invoices
library(ggplot2)
library(dplyr)
filter <- Online_Retail1 %>% filter(Year==2011) %>% count(Month)
View(filter) ##We see number of invoices is highest for Nov = 64,531
ggplot(filter, aes(Month, n)) + ggtitle("Number of Invoices per Month")    +

     geom_col() +
     labs(x="Month", y="Number of invoices" #We confirm that transactions
    were highest in the month of November



##Plot 2 - Month vs Day of Invoices
filter_1 <- Online_Retail1 %>% filter(Year==2011) %>% group_by(Month, Day) %>%
 count(Month)
View(filter_1)
ggplot(filter_1, aes(Month, Day, size=n)) + geom_point() + ggtitle("Invoices i
n 2011 by Month and Day")


##Plot 3 - Revenue raked in by country
filter_2 <- Online_Retail1 %>% group_by(Country) %>% summarise(Revenue = sum(R
evenue))
```

```
ggplot(filter_2, aes(y=Revenue/1000, x=Country)) + geom_bar(stat = "identity")
 + ggtitle("Revenue by Country") +

  labs(x="Country", y="Sales in thousands") + coord_flip()


#Plot 4- As the data is of UK we remove UK and check again for highest contrib
utors

filter_3 <- Online_Retail1 %>% group_by(Country) %>% summarise(Revenue = sum(R
evenue)) %>% filter(Country!="United Kingdom")

ggplot(filter_3, aes(y=Revenue/1000, x=Country)) + geom_bar(stat = "identity")
 + ggtitle("Revenue by Country except UK") +

    labs(x="Country", y="Sales in thousands") + coord_flip()



#Plot 5 - Revenue by Month

filter_4 <- Online_Retail1 %>% group_by(Month) %>% summarise(Revenue = sum(Rev
enue))

ggplot(filter_4, aes(y=Revenue/1000, x=Month)) + geom_bar(stat = "identity") +
 ggtitle("Revenue by Month") +

      labs(x="Month", y="Revenue")



#plot 6 - Split transactions by weekday

Online_Retail1$Week_day <- wday(Online_Retail1$Date_Only, label = T)

filter_5 <- Online_Retail1 %>% filter(Year==2011) %>% count(Week_day)

ggplot(filter_5, aes(Week_day, n)) + ggtitle("Invoices by Days of the Week") +
 geom_col() + labs(x = "Days of the Week", y = "Number of Invoices")


#Plot 7 - Split revenue by weekday

filter_6 <- Online_Retail1 %>% group_by(Week_day) %>% summarise(Revenue = sum(
Revenue))

ggplot(filter_6, aes(y=Revenue/1000, x=Week_day)) + geom_bar(stat = "identity"
) + ggtitle("Revenue by Week") +

      labs(x="Days of the week", y="Revenue")



#PLot 8 - Revenue raked by Country in each month

filter_7 <- Online_Retail1 %>% filter(Year==2011) %>% filter(Country!="United
Kingdom") %>% group_by(Month, Country) %>% count(Month)

ggplot(filter_7, aes(Month, Country, size=n)) + geom_point() + ggtitle("Invoic
es in 2011 by Month and Country")



#Plot 9 - Most popular product
```

```
filter_8 <- Online_Retail1 %>% group_by(StockCode, Description) %>% summarise(
count= n()) %>% arrange(desc(count)) %>% head(n=15)

ggplot(filter_8, aes(x=Description, y=count)) + geom_bar(stat= "identity") + c
oord_flip() +

  labs(y="Number of items purchased", x="Product")
```

Appendix 2:

Data Exploration in R Studio- Part 2

```
#Yasmin Sultana: Data Exploration of Online Retail Data
#loading the required R packages
library(data.table)
library(readr)
library(dplyr)
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:data.table':
##
##     between, first, last
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library(ggplot2)
library(DataExplorer)
library(lubridate)
##
## Attaching package: 'lubridate'
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday,
##     week, yday, year
## The following object is masked from 'package:base':
##
##     date
library(datetime)
library(tidyr)
library(tm)
## Loading required package: NLP
##
## Attaching package: 'NLP'
## The following object is masked from 'package:ggplot2':
##
```

```
##     annotate
library(SnowballC)
library(wordcloud)
## Loading required package: RColorBrewer
library(RColorBrewer)

#loading the csv file into R using function fread
Online_Retail_Data <-fread("BA Project data Yasmin Sultana.csv")

#displaying the class of the dataset
class(Online_Retail_Data)
## [1] "data.table" "data.frame"
#finding the number of rows in the dataset
dim(Online_Retail_Data)
## [1] 541909      8
#listing the variables in the data
names(Online_Retail_Data)
## [1] "InvoiceNo"   "StockCode"   "Description" "Quantity"    "InvoiceDate"
## [6] "UnitPrice"   "CustomerID"  "Country"
# 8 columns namely: Invoice No, Stock Code, Description, InvoiceDate, UnitPric
e, CustomerID, Country

#printing the first 10 rows of the dataset
head(Online_Retail_Data,n=5)
##    InvoiceNo StockCode                         Description Quantity
## 1:    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6
## 2:    536365     71053                 WHITE METAL LANTERN        6
## 3:    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8
## 4:    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6
## 5:    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6
##       InvoiceDate UnitPrice CustomerID        Country
## 1: 12/1/2010 8:26      2.55      17850 United Kingdom
## 2: 12/1/2010 8:26      3.39      17850 United Kingdom
## 3: 12/1/2010 8:26      2.75      17850 United Kingdom
## 4: 12/1/2010 8:26      3.39      17850 United Kingdom
## 5: 12/1/2010 8:26      3.39      17850 United Kingdom
#printing the last 10 rows of the dataset toyotacorolla
tail(Online_Retail_Data,n=10)
##     InvoiceNo StockCode                         Description Quantity
##  1:    581587     22726       ALARM CLOCK BAKELIKE GREEN        4
```

```
##  2:    581587     22730      ALARM CLOCK BAKELIKE IVORY        4
##  3:    581587     22367 CHILDRENS APRON SPACEBOY DESIGN        8
##  4:    581587     22629             SPACEBOY LUNCH BOX       12
##  5:    581587     23256      CHILDRENS CUTLERY SPACEBOY        4
##  6:    581587     22613     PACK OF 20 SPACEBOY NAPKINS       12
##  7:    581587     22899     CHILDREN'S APRON DOLLY GIRL        6
##  8:    581587     23254    CHILDRENS CUTLERY DOLLY GIRL        4
##  9:    581587     23255 CHILDRENS CUTLERY CIRCUS PARADE        4
## 10:    581587     22138    BAKING SET 9 PIECE RETROSPOT        3
##          InvoiceDate UnitPrice CustomerID Country
##  1: 12/9/2011 12:50      3.75      12680  France
##  2: 12/9/2011 12:50      3.75      12680  France
##  3: 12/9/2011 12:50      1.95      12680  France
##  4: 12/9/2011 12:50      1.95      12680  France
##  5: 12/9/2011 12:50      4.15      12680  France
##  6: 12/9/2011 12:50      0.85      12680  France
##  7: 12/9/2011 12:50      2.10      12680  France
##  8: 12/9/2011 12:50      4.15      12680  France
##  9: 12/9/2011 12:50      4.15      12680  France
## 10: 12/9/2011 12:50      4.95      12680  France
```

```
#showing the summary of the dataset
summary(Online_Retail_Data)
```

```
##   InvoiceNo           StockCode          Description
##  Length:541909      Length:541909      Length:541909
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     Quantity          InvoiceDate          UnitPrice
##  Min.   :-80995.00   Length:541909      Min.   :-11062.06
##  1st Qu.:     1.00   Class :character   1st Qu.:     1.25
##  Median :     3.00   Mode  :character   Median :     2.08
##  Mean   :     9.55                      Mean   :     4.61
##  3rd Qu.:    10.00                      3rd Qu.:     4.13
##  Max.   : 80995.00                      Max.   : 38970.00
##
##    CustomerID       Country
```

```
##  Min.   :12346    Length:541909
##  1st Qu.:13953    Class :character
##  Median :15152    Mode  :character
##  Mean   :15288
##  3rd Qu.:16791
##  Max.   :18287
##  NA's   :135080
#We can see that variables: Quantity and Unit Price has negative values. This
is not possible. So, we remove the negative values from our anaysis.


#finding the number of different products sold
n_distinct(Online_Retail_Data$StockCode)
## [1] 4070
#We can see that the organisation sold 4070 products


#finding which columns have missing values
colSums(is.na(Online_Retail_Data))
##   InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice
##           0           0           0           0           0           0
##  CustomerID     Country
##      135080           0
#We can see that column CustomerID has 135080 missing values. Thus, the number
 of customers annot be accuratey identified.


#finding the nuber of countries that the organisation sells its products in
n_distinct(Online_Retail_Data$Country)
## [1] 38
#We can see that the organisatino sells its products in 38 countries


#naming the 38 countries that the organisation sells its products in
unique(Online_Retail_Data$Country)
##  [1] "United Kingdom"       "France"               "Australia"
##  [4] "Netherlands"          "Germany"              "Norway"
##  [7] "EIRE"                 "Switzerland"          "Spain"
## [10] "Poland"               "Portugal"             "Italy"
## [13] "Belgium"              "Lithuania"            "Japan"
## [16] "Iceland"              "Channel Islands"      "Denmark"
## [19] "Cyprus"               "Sweden"               "Austria"
## [22] "Israel"               "Finland"              "Bahrain"
## [25] "Greece"               "Hong Kong"            "Singapore"
```

```
## [28] "Lebanon"               "United Arab Emirates" "Saudi Arabia"
## [31] "Czech Republic"        "Canada"               "Unspecified"
## [34] "Brazil"                "USA"                  "European Community"
## [37] "Malta"                 "RSA"
```

```
sum(is.na(Online_Retail_Data))
```

```
## [1] 135080
```

```
#The number of missing values before removing the negative values is 135080.


#omitting all rows from the dataset that contain negative values
Online_Retail_Data<-Online_Retail_Data[Online_Retail_Data$Quantity>0,]
Online_Retail_Data<-Online_Retail_Data[Online_Retail_Data$UnitPrice>0,]



#finding the number of missing vales in the dataset before eliminating the mis
sing values
sum(is.na(Online_Retail_Data))
```

```
## [1] 132220
```

```
#finding the total number of rows and columns before eliminating missing value
s
dim(Online_Retail_Data)
```

```
## [1] 530104      8
```

```
#finding which columns have missing values
colSums(is.na(Online_Retail_Data))
```

```
##   InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice
##           0           0           0           0           0           0
##  CustomerID     Country
##      132220           0
```

```
#We can see that column CustomerID has missing values


#omitting missing values from the dataset
Online_Retail_Data<-na.omit(Online_Retail_Data)
#finding the total number of rows and columns after eliminating the missing va
lues i.e. 397884 rows and 8 columns
dim(Online_Retail_Data)
```

```
## [1] 397884      8
```

```
#checking the summary of the dataset to verify if the missing values are remov
ed from the dataset
summary(Online_Retail_Data)
```

```
##   InvoiceNo           StockCode          Description
##  Length:397884      Length:397884      Length:397884
##  Class :character   Class :character   Class :character
```

```
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##     Quantity         InvoiceDate          UnitPrice          CustomerID
##  Min.    :    1.00   Length:397884      Min.    :   0.001   Min.   :12346
##  1st Qu.:    2.00   Class :character   1st Qu.:   1.250   1st Qu.:13969
##  Median :    6.00   Mode  :character   Median :   1.950   Median :15159
##  Mean   :   12.99                      Mean   :   3.116   Mean   :15294
##  3rd Qu.:   12.00                      3rd Qu.:   3.750   3rd Qu.:16795
##  Max.   :80995.00                      Max.   :8142.750   Max.   :18287
##    Country
##  Length:397884
##  Class :character
##  Mode  :character
##
##
##
#InvoiveDate represents both the date and time of the transactions
#Thus, we separate the date and time components of the transactions for furthe
r analysis
datetimesplit<-data.frame(do.call(rbind,strsplit(Online_Retail_Data$InvoiceDat
e," ")))
names(datetimesplit)[1:2]<- c("date","time")


Online_Retail_Data$date<-datetimesplit$date
Online_Retail_Data$time<-Online_Retail_Data$time <- sapply(Online_Retail_Data$
InvoiceDate, FUN = function(x) {strsplit(x, split = '[ ]')[[1]][2]})


is.Date(as.Date(Online_Retail_Data$date)) #TRUE
## [1] TRUE
#converting bar$date to date class
Online_Retail_Data$date <- as.Date(Online_Retail_Data$date, "%m/%d/%Y")


#creating month, year,day and hours variables
Online_Retail_Data$year = lubridate::year(Online_Retail_Data$date)
Online_Retail_Data$month = lubridate::month(Online_Retail_Data$date)
Online_Retail_Data$day = lubridate::day(Online_Retail_Data$date)
Online_Retail_Data$hours <- sapply(Online_Retail_Data$time, FUN = function(x)
{strsplit(x, split = '[:]')[[1]][1]})
```

```r
#creating a new variable Revenue
Online_Retail_Data$Revenue <- Online_Retail_Data$Quantity * Online_Retail_Data
$UnitPrice


#Viewing the data after adding new columns
View(Online_Retail_Data)


##finding day of the week
Online_Retail_Data$dayname <- wday(Online_Retail_Data$date, label=TRUE)
head(Online_Retail_Data, n =10)
```

```
##     InvoiceNo StockCode                         Description Quantity
##  1:    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6
##  2:    536365     71053                 WHITE METAL LANTERN        6
##  3:    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8
##  4:    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6
##  5:    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6
##  6:    536365     22752        SET 7 BABUSHKA NESTING BOXES        2
##  7:    536365     21730   GLASS STAR FROSTED T-LIGHT HOLDER        6
##  8:    536366     22633              HAND WARMER UNION JACK        6
##  9:    536366     22632           HAND WARMER RED POLKA DOT        6
## 10:    536367     84879       ASSORTED COLOUR BIRD ORNAMENT       32
##         InvoiceDate UnitPrice CustomerID        Country       date time
##  1: 12/1/2010 8:26      2.55      17850 United Kingdom 2010-12-01 8:26
##  2: 12/1/2010 8:26      3.39      17850 United Kingdom 2010-12-01 8:26
##  3: 12/1/2010 8:26      2.75      17850 United Kingdom 2010-12-01 8:26
##  4: 12/1/2010 8:26      3.39      17850 United Kingdom 2010-12-01 8:26
##  5: 12/1/2010 8:26      3.39      17850 United Kingdom 2010-12-01 8:26
##  6: 12/1/2010 8:26      7.65      17850 United Kingdom 2010-12-01 8:26
##  7: 12/1/2010 8:26      4.25      17850 United Kingdom 2010-12-01 8:26
##  8: 12/1/2010 8:28      1.85      17850 United Kingdom 2010-12-01 8:28
##  9: 12/1/2010 8:28      1.85      17850 United Kingdom 2010-12-01 8:28
## 10: 12/1/2010 8:34      1.69      13047 United Kingdom 2010-12-01 8:34
##     year month day hours Revenue dayname
##  1: 2010    12   1     8   15.30     Wed
##  2: 2010    12   1     8   20.34     Wed
##  3: 2010    12   1     8   22.00     Wed
##  4: 2010    12   1     8   20.34     Wed
##  5: 2010    12   1     8   20.34     Wed
##  6: 2010    12   1     8   15.30     Wed
##  7: 2010    12   1     8   25.50     Wed
```

```
## 8: 2010 12 1 8 11.10 Wed
## 9: 2010 12 1 8 11.10 Wed
## 10: 2010 12 1 8 54.08 Wed
Online_Retail_Data$Country <- as.factor(Online_Retail_Data$Country)
Online_Retail_Data$month <- as.factor(Online_Retail_Data$month)
Online_Retail_Data$day <- as.factor(Online_Retail_Data$day)
Online_Retail_Data$year <- as.factor(Online_Retail_Data$year)
Online_Retail_Data$hours <- as.factor(Online_Retail_Data$hours)
Online_Retail_Data$dayname <- as.factor(Online_Retail_Data$dayname)

##finding the year whose revenue is the highest
RevenuebyYear<-Online_Retail_Data %>% group_by(year) %>% summarise(Revenue=sum
(Revenue))
ggplot(RevenuebyYear,aes(x=year, y=Revenue/1000)) + geom_col() + labs(x = 'Ye
ar', y = 'Revenue (in thousands)', title = 'Revenue by Year (in thousands)')
```

```
#We can see that the revenue earned during 2011 is greater than the revenue ea
rned in 2010
RevenuebyYear
## # A tibble: 2 x 2
##   year   Revenue
##   <fct>    <dbl>
## 1 2010   572714.
## 2 2011  8338694.
#Revenue earned in 2011 is 8,338,694 and the revenue earned in 2010 is 572,714
.

##finding the month in which the revenue was the highest and the month in whic
h the revenue was the lowest
RevenuebyMonth<-Online_Retail_Data %>% group_by(month) %>% summarise(Revenue=s
um(Revenue))
ggplot(RevenuebyMonth,aes(x=month, y=Revenue/1000)) + geom_col() + labs(x = '
Month', y = 'Revenue (in thosands)', title = 'Revenue by Month')
```

```
RevenuebyMonth
## # A tibble: 12 x 2
##    month  Revenue
##    <fct>    <dbl>
##  1 1      569445.
##  2 2      447137.
```

```
##   3 3       595501.
##   4 4       469200.
##   5 5       678595.
##   6 6       661214.
##   7 7       600091.
##   8 8       645344.
##   9 9       952838.
## 10 10     1039319.
## 11 11     1161817.
## 12 12     1090907.
```

```
#We can see that the highest revenue is earned during November i.e. month 11 (amounting to 1,161,817.), and the lowest revenue is earned during February i.e. month 2 (amounting to 447,137.)


##determining which day of the week has the highest revenue
RevenuebyDayofWeek<-Online_Retail_Data %>% group_by(dayname)%>% summarise(Revenue=sum(Revenue))
ggplot(RevenuebyDayofWeek,aes(x=dayname, y=Revenue)) +  geom_col() + labs(x = 'Day of Week', y = 'Revenue', title = 'Revenue by Day of Week')
```

```
#We can see that highest revenue was earned on Thursday, while the lowest revenue was earned on a Sunday.
RevenuebyDayofWeek
```

```
## # A tibble: 6 x 2
##   dayname  Revenue
##   <ord>      <dbl>
## 1 Sun       792514.
## 2 Mon      1367146.
## 3 Tue      1700635.
## 4 Wed      1588336.
## 5 Thu      1976859.
## 6 Fri      1485917.
```

```
#Revenue on Thursday=1,976,859 (highest). Revenue was recorded as 792,514 on Sunday (lowest)


##determining the time of the day when the revenue earned is the maximum and the time of the day when the revenue earned was minimum
RevenuebyHourofDay<-Online_Retail_Data %>% group_by(hours) %>% summarise(Revenue=sum(Revenue))
ggplot(RevenuebyHourofDay,aes(x=hours, y=Revenue/1000)) +  geom_col() + labs(x = 'Hour of the day', y = 'Revenue (in thousands)', title = 'Revenue by Hour of the day')
```

```
#The highest revenue is earned at 12 pm. Almost no revenue is earned at 6 am.
RevenuebyHourofDay
## # A tibble: 15 x 2
##    hours    Revenue
##    <fct>      <dbl>
##  1 10    1261193.
##  2 11    1104559.
##  3 12    1378571.
##  4 13    1173265.
##  5 14     995629.
##  6 15     966192.
##  7 16     468886.
##  8 17     234414.
##  9 18     104954.
## 10 19      49028.
## 11 20      18933.
## 12 6           4.25
## 13 7       31059.
## 14 8      282116.
## 15 9      842605.
#Revene earned at 12pm=1,378,571. (highest). Revenue earned at 6am=amounting t
o 4.25 (lowest)


##determining the transactions by hour of the day
TransactionsbyHour<-Online_Retail_Data %>% group_by(hours) %>% summarise(trans
actions=n_distinct(InvoiceNo))
ggplot(TransactionsbyHour,aes(x = hours, y = transactions)) + geom_col() + lab
s(x = 'Hour Of Day', y = 'Number of Transactions', title = 'Transactions by Ho
ur of the day')
```

```
#The peak sales hour is 12 pm. The organisation has made negligible sales at 6
am.
TransactionsbyHour
## # A tibble: 15 x 2
##    hours transactions
##    <fct>        <int>
##  1 10            2226
##  2 11            2277
##  3 12            3130
```

```
##   4 13              2636
##   5 14              2274
##   6 15              2037
##   7 16              1100
##   8 17               544
##   9 18               169
## 10 19               144
## 11 20                18
## 12 6                  1
## 13 7                 29
## 14 8                555
## 15 9               1393
```

```
#The sales quantity during the 12th hour=3130. Sales quantity during the 6th h
our=1 (negligible)


##determining the country that earned the highest revenue
ggplot(Online_Retail_Data,aes(x=Country, y=Revenue/1000)) +  geom_col() + labs
(x = 'Country', y = 'Revenue (in thousands)', title = 'Revenue by Country') +
coord_flip()
```

```
#The country that earned that earned the highest revenue is United Kingdom


##Using tag clouds to determine the country with the highest number of transac
tions
#eliminating spaces between words in a string to prevent the words in the stri
ng from being considered as two separate strings
Online_Retail_Data$Country<-gsub("[[:space:]]", "", Online_Retail_Data$Country
)
write.table(Online_Retail_Data$Country, file = "Country.txt", sep = "\t",row.n
ames = FALSE, col.names = "COUNTRY")


#importing text file interactively
text <- readLines(file.choose())


#loading the data as corpus
docs <- Corpus(VectorSource(text))


tdm <- TermDocumentMatrix(docs) #creating a term document matrix
m <- as.matrix(tdm)
sorting <- sort(rowSums(m),decreasing=TRUE)
tagcloud_CountrybyTrans <- data.frame(word=names(sorting),freq=sorting)
```

```
#viewing the top 5 countries based on the number of transactions

CountrybyTransactions<-data.frame(freq=sorting)

head(CountrybyTransactions,5)
```

```
##                 freq
## unitedkingdom 354321
## germany         9040
## france          8341
## eire            7236
## spain           2484
```

```
#We can see that UK has more number of transactions with word frequency being the highest


#creating the word cloud to display the country with the highest number of transactions

set.seed(111994)

wordcloud(words = tagcloud_CountrybyTrans$word, freq = tagcloud_CountrybyTrans$freq, min.freq = 1,

 max.words=200, random.order=FALSE, rot.per=0.35,

 colors=brewer.pal(8, "Accent"))
```

```
#As we have already found United Kingdom to be the country with the highest number of transactions, we are now finding the 2nd, 3rd, 4th and 5th country that is leading in terms of the number of transactions

barplot(tagcloud_CountrybyTrans[2:5,]$freq, las = 2, names.arg = tagcloud_CountrybyTrans[2:5,]$word,

  col ="gold3", main ="Country with maximum number of transactions",

  ylab = "Number of transactions", xlab="Countries")
```

Appendix 3:

Implementation of Market Basket Analysis in R Studio

```r
Retail_data<- read.csv("Online Retail.csv")##Reading the Raw dataset and namin
g as Retail_data

## Installing Required packages

##we need to perform some data manipulation so we have attached the concern li
braries

library(dplyr)

##

## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':

##

##     filter, lag

## The following objects are masked from 'package:base':

##

##     intersect, setdiff, setequal, union

library(arules)

## Loading required package: Matrix

##

## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':

##

##     recode

## The following objects are masked from 'package:base':

##

##     abbreviate, write

library(arulesViz)

## Loading required package: grid

library(tidyverse)

## ── Attaching packages ──────────────────────────────────────────────────────
──────────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.0.0     ✔ readr   1.1.1

## ✔ tibble  1.4.2     ✔ purrr   0.2.5

## ✔ tidyr   0.8.1     ✔ stringr 1.3.1

## ✔ ggplot2 3.0.0     ✔ forcats 0.3.0

## ── Conflicts ───────────────────────────────────────────────────────────────
────────────────────────────────────────────────────── tidyverse_conflicts() ──

## ✖ tidyr::expand()  masks Matrix::expand()

## ✖ dplyr::filter()  masks stats::filter()
```

```
## ✖ dplyr::lag()     masks stats::lag()
## ✖ arules::recode() masks dplyr::recode()
library(stringr)
library(lubridate)
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##     date
library(ggplot2)
class(Retail_data) ## Viewing the Raw data file
## [1] "data.frame"
##taking first look on the data by checking class, dimension, structure, summa
ry and some of the records from top and bottom.
dim(Retail_data)
## [1] 541909      8
names(Retail_data)
## [1] "InvoiceNo"   "StockCode"   "Description" "Quantity"    "InvoiceDate"
## [6] "UnitPrice"   "CustomerID"  "Country"
glimpse(Retail_data)
## Observations: 541,909
## Variables: 8
## $ InvoiceNo   <fct> 536365, 536365, 536365, 536365, 536365, 536365, 53...
## $ StockCode   <fct> 85123A, 71053, 84406B, 84029G, 84029E, 22752, 2173...
## $ Description <fct> WHITE HANGING HEART T-LIGHT HOLDER, WHITE METAL LA...
## $ Quantity    <int> 6, 6, 8, 6, 6, 2, 6, 6, 6, 32, 6, 6, 8, 6, 6, 3, 2...
## $ InvoiceDate <fct> 12/1/10 8:26, 12/1/10 8:26, 12/1/10 8:26, 12/1/10 ...
## $ UnitPrice   <dbl> 2.55, 3.39, 2.75, 3.39, 3.39, 7.65, 4.25, 1.85, 1....
## $ CustomerID  <int> 17850, 17850, 17850, 17850, 17850, 17850, 17850, 1...
## $ Country     <fct> United Kingdom, United Kingdom, United Kingdom, Un...
summary(Retail_data)
##    InvoiceNo        StockCode
##   573585 :  1114   85123A :  2313
##   581219 :   749   22423  :  2203
##   581492 :   731   85099B :  2159
##   580729 :   721   47566  :  1727
##   558475 :   705   20725  :  1639
##   579777 :   687   84879  :  1502
##   (Other):537202   (Other):530366
```

```
##                               Description        Quantity
##  WHITE HANGING HEART T-LIGHT HOLDER:  2369   Min.   :-80995.00
##  REGENCY CAKESTAND 3 TIER          :  2200   1st Qu.:     1.00
##  JUMBO BAG RED RETROSPOT           :  2159   Median :     3.00
##  PARTY BUNTING                     :  1727   Mean   :     9.55
##  LUNCH BAG RED RETROSPOT           :  1638   3rd Qu.:    10.00
##  ASSORTED COLOUR BIRD ORNAMENT     :  1501   Max.   : 80995.00
##  (Other)                           :530315
##          InvoiceDate       UnitPrice          CustomerID
##  10/31/11 14:41:  1114   Min.   :-11062.06   Min.   :12346
##  12/8/11 9:28  :   749   1st Qu.:     1.25   1st Qu.:13953
##  12/9/11 10:03 :   731   Median :     2.08   Median :15152
##  12/5/11 17:24 :   721   Mean   :     4.61   Mean   :15288
##  6/29/11 15:58 :   705   3rd Qu.:     4.13   3rd Qu.:16791
##  11/30/11 15:13:   687   Max.   : 38970.00   Max.   :18287
##  (Other)       :537202                       NA's   :135080
##            Country
##  United Kingdom:495478
##  Germany       :  9495
##  France        :  8557
##  EIRE          :  8196
##  Spain         :  2533
##  Netherlands   :  2371
##  (Other)       : 15279
```

```
head(Retail_data)
```

```
##   InvoiceNo StockCode                         Description Quantity
## 1    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6
## 2    536365     71053                 WHITE METAL LANTERN        6
## 3    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8
## 4    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6
## 5    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6
## 6    536365     22752        SET 7 BABUSHKA NESTING BOXES        2
##    InvoiceDate UnitPrice CustomerID        Country
## 1 12/1/10 8:26      2.55      17850 United Kingdom
## 2 12/1/10 8:26      3.39      17850 United Kingdom
## 3 12/1/10 8:26      2.75      17850 United Kingdom
## 4 12/1/10 8:26      3.39      17850 United Kingdom
## 5 12/1/10 8:26      3.39      17850 United Kingdom
## 6 12/1/10 8:26      7.65      17850 United Kingdom
```

```
tail(Retail_data)
```

```
##        InvoiceNo StockCode                     Description Quantity
## 541904    581587     23256     CHILDRENS CUTLERY SPACEBOY         4
## 541905    581587     22613     PACK OF 20 SPACEBOY NAPKINS       12
## 541906    581587     22899    CHILDREN'S APRON DOLLY GIRL         6
## 541907    581587     23254   CHILDRENS CUTLERY DOLLY GIRL         4
## 541908    581587     23255 CHILDRENS CUTLERY CIRCUS PARADE        4
## 541909    581587     22138   BAKING SET 9 PIECE RETROSPOT         3
##           InvoiceDate UnitPrice CustomerID Country
## 541904 12/9/11 12:50      4.15      12680  France
## 541905 12/9/11 12:50      0.85      12680  France
## 541906 12/9/11 12:50      2.10      12680  France
## 541907 12/9/11 12:50      4.15      12680  France
## 541908 12/9/11 12:50      4.15      12680  France
## 541909 12/9/11 12:50      4.95      12680  France
```

```
## Manipulating the Date column, making the dates in Date format and looking f
or the changes those have been made
## creating newcolumns for Date Month and year and then consolidating all in s
ingle data frame
## looking at the sumary of the new data


Retail_data$InvoiceDate <- mdy_hm(Retail_data$InvoiceDate)
Time<- hour(Retail_data$InvoiceDate)
new_date<- as.Date(format(Retail_data$InvoiceDate,"%Y-%m-%d"))
new_month<- month(Retail_data$InvoiceDate)
new_year<- year(Retail_data$InvoiceDate)
Retail_data<-data.frame(Retail_data,new_date,new_month,new_year,Time)


summary(Retail_data)
```

```
##     InvoiceNo         StockCode
##   573585 :  1114   85123A :  2313
##   581219 :   749   22423  :  2203
##   581492 :   731   85099B :  2159
##   580729 :   721   47566  :  1727
##   558475 :   705   20725  :  1639
##   579777 :   687   84879  :  1502
##   (Other):537202   (Other):530366
##                                Description        Quantity
##   WHITE HANGING HEART T-LIGHT HOLDER:  2369   Min.   :-80995.00
##   REGENCY CAKESTAND 3 TIER          :  2200   1st Qu.:     1.00
```

```
##  JUMBO BAG RED RETROSPOT           :  2159   Median :     3.00
##  PARTY BUNTING                     :  1727   Mean   :     9.55
##  LUNCH BAG RED RETROSPOT           :  1638   3rd Qu.:    10.00
##  ASSORTED COLOUR BIRD ORNAMENT     :  1501   Max.   : 80995.00
##  (Other)                           :530315
##   InvoiceDate                     UnitPrice           CustomerID
##  Min.   :2010-12-01 08:26:00   Min.   :-11062.06   Min.   :12346
##  1st Qu.:2011-03-28 11:34:00   1st Qu.:     1.25   1st Qu.:13953
##  Median :2011-07-19 17:17:00   Median :     2.08   Median :15152
##  Mean   :2011-07-04 13:34:57   Mean   :     4.61   Mean   :15288
##  3rd Qu.:2011-10-19 11:27:00   3rd Qu.:     4.13   3rd Qu.:16791
##  Max.   :2011-12-09 12:50:00   Max.   : 38970.00   Max.   :18287
##                                                    NA's   :135080
##            Country           new_date             new_month
##  United Kingdom:495478   Min.   :2010-12-01   Min.   : 1.000
##  Germany       :  9495   1st Qu.:2011-03-28   1st Qu.: 5.000
##  France        :  8557   Median :2011-07-19   Median : 8.000
##  EIRE          :  8196   Mean   :2011-07-04   Mean   : 7.553
##  Spain         :  2533   3rd Qu.:2011-10-19   3rd Qu.:11.000
##  Netherlands   :  2371   Max.   :2011-12-09   Max.   :12.000
##  (Other)       : 15279
##     new_year          Time
##  Min.   :2010   Min.   : 6.00
##  1st Qu.:2011   1st Qu.:11.00
##  Median :2011   Median :13.00
##  Mean   :2011   Mean   :13.08
##  3rd Qu.:2011   3rd Qu.:15.00
##  Max.   :2011   Max.   :20.00
##
head(Retail_data)
##   InvoiceNo StockCode                        Description Quantity
## 1    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6
## 2    536365     71053                 WHITE METAL LANTERN        6
## 3    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8
## 4    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6
## 5    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6
## 6    536365     22752        SET 7 BABUSHKA NESTING BOXES        2
##           InvoiceDate UnitPrice CustomerID        Country   new_date
## 1 2010-12-01 08:26:00      2.55      17850 United Kingdom 2010-12-01
```

```
## 2 2010-12-01 08:26:00      3.39      17850 United Kingdom 2010-12-01
## 3 2010-12-01 08:26:00      2.75      17850 United Kingdom 2010-12-01
## 4 2010-12-01 08:26:00      3.39      17850 United Kingdom 2010-12-01
## 5 2010-12-01 08:26:00      3.39      17850 United Kingdom 2010-12-01
## 6 2010-12-01 08:26:00      7.65      17850 United Kingdom 2010-12-01
##   new_month new_year Time
## 1        12     2010    8
## 2        12     2010    8
## 3        12     2010    8
## 4        12     2010    8
## 5        12     2010    8
## 6        12     2010    8
```

```
## Creating final data set for the model
## Removing the missing entries in the item field
flag1<-Retail_data$Description !=""
Retail_data_wip2<- Retail_data[flag1,c(1,3)]
##converting Data frame to transactions


write.csv(Retail_data_wip2, file = "Retail_trans.csv")
trans = read.transactions("Retail_trans.csv", format = "single", sep = ",", co
ls = c("InvoiceNo", "Description"))
inspect(trans[1:5])
```

```
##     items                                      transactionID
## [1] {CREAM CUPID HEARTS COAT HANGER,
##      GLASS STAR FROSTED T-LIGHT HOLDER,
##      KNITTED UNION FLAG HOT WATER BOTTLE,
##      RED WOOLLY HOTTIE WHITE HEART.,
##      SET 7 BABUSHKA NESTING BOXES,
##      WHITE HANGING HEART T-LIGHT HOLDER,
##      WHITE METAL LANTERN}                             536365
## [2] {HAND WARMER RED POLKA DOT,
##      HAND WARMER UNION JACK}                          536366
## [3] {ASSORTED COLOUR BIRD ORNAMENT,
##      BOX OF 6 ASSORTED COLOUR TEASPOONS,
##      BOX OF VINTAGE ALPHABET BLOCKS,
##      BOX OF VINTAGE JIGSAW BLOCKS ,
##      DOORMAT NEW ENGLAND,
##      FELTCRAFT PRINCESS CHARLOTTE DOLL,
##      HOME BUILDING BLOCK WORD,
##      IVORY KNITTED MUG COSY ,
```

```
##       LOVE BUILDING BLOCK WORD,
##       POPPY'S PLAYHOUSE BEDROOM ,
##       POPPY'S PLAYHOUSE KITCHEN,
##       RECIPE BOX WITH METAL HEART}                    536367
## [4] {BLUE COAT RACK PARIS FASHION,
##       JAM MAKING SET WITH JARS,
##       RED COAT RACK PARIS FASHION,
##       YELLOW COAT RACK PARIS FASHION}                 536368
## [5] {BATH BUILDING BLOCK WORD}                        536369
```

```
## running the model and sorting the results by confidence
## inspecting the rules

rules = apriori(trans,parameter = list(supp = 0.001,conf=0.8))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 24
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[4223 item(s), 24446 transaction(s)] done [0.20s].
## sorting and recoding items ... [2751 item(s)] done [0.01s].
## creating transaction tree ... done [0.01s].
## checking subsets of size 1 2 3 4
## Warning in apriori(trans, parameter = list(supp = 0.001, conf = 0.8)):
## Mining stopped (time limit reached). Only patterns up to a length of 4
## returned!
##  done [16.53s].
## writing ... [38785484 rule(s)] done [8.08s].
## creating S4 object  ... done [20.90s].
```

```
rules<- sort(rules,by= "confidence",decreasing = "T")
```

```
inspect(rules[1:10])
```

```
##      lhs                                   rhs
     support confidence     lift count
## [1]  {SILVER MINI TAPE MEASURE }        => {JUMBO BAG PINK VINTAGE PAISLEY}
 0.001063569          1 27.87457    26
## [2]  {SILVER MINI TAPE MEASURE }        => {STRAWBERRY CHARLOTTE BAG}
 0.001063569          1 33.30518    26
## [3]  {SILVER MINI TAPE MEASURE }        => {LUNCH BAG CARS BLUE}
 0.001063569          1 20.84058    26
## [4]  {SILVER MINI TAPE MEASURE }        => {WOODLAND CHARLOTTE BAG}
 0.001063569          1 28.99881    26
## [5]  {SILVER MINI TAPE MEASURE }        => {RED RETROSPOT CHARLOTTE BAG}
 0.001063569          1 23.28190    26
## [6]  {OLD ROSE COMBO BEAD NECKLACE}     => {DOTCOM POSTAGE}
 0.001840792          1 34.47955    45
## [7]  {PINK BUTTERFLY HANDBAG W BOBBLES} => {DOTCOM POSTAGE}
 0.004008836          1 34.47955    98
## [8]  {SILVER MINI TAPE MEASURE ,

##       TRAVEL SEWING KIT}                => {JUMBO BAG PINK VINTAGE PAISLEY}
 0.001022662          1 27.87457    25
## [9]  {SILVER MINI TAPE MEASURE ,

##       TRAVEL SEWING KIT}                => {STRAWBERRY CHARLOTTE BAG}
 0.001022662          1 33.30518    25
## [10] {SILVER MINI TAPE MEASURE ,

##       TRAVEL SEWING KIT}                => {LUNCH BAG CARS BLUE}
 0.001022662          1 20.84058    25
```

Appendix 4:

Implementation of the Item Based Collaborative Filtering in R Studio

The below implementation uses the default parameters of the recommender system to recommend top 5 products to the customer with id = 12349. The Top 5 products suggested by the model closely resemble the 85 purchases made by the customer.

```
library(readr)
library(dplyr)
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library(ggplot2)
library(DataExplorer)
library(methods)
library(recommenderlab)
## Loading required package: Matrix
## Loading required package: arules
##
## Attaching package: 'arules'
## The following object is masked from 'package:dplyr':
##
##     recode
## The following objects are masked from 'package:base':
##
##     abbreviate, write
## Loading required package: proxy
##
## Attaching package: 'proxy'
## The following object is masked from 'package:Matrix':
##
##     as.matrix
## The following objects are masked from 'package:stats':
##
##     as.dist, dist
```

```
## The following object is masked from 'package:base':
##
##     as.matrix
## Loading required package: registry
library(data.table)
##
## Attaching package: 'data.table'
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
library(ggplot2)
library(knitr)
#Loading the Data set
EData_df <- fread("C:/Users/gowri/Desktop/Business Analytics with R/RProject/O
nline Retail.csv")
class(EData_df)
## [1] "data.table" "data.frame"
head(EData_df,1)
##    InvoiceNo StockCode                        Description Quantity
## 1:    536365    85123A WHITE HANGING HEART T-LIGHT HOLDER        6
##       InvoiceDate UnitPrice CustomerID        Country
## 1: 12/1/2010 8:26      2.55      17850 United Kingdom
#Data Preprocessing step, removing the negative values and preparing the data
for the recommender model
EData_df[Quantity<=0,Quantity:=NA]
EData_df[UnitPrice<=0,UnitPrice:=NA]
EData_df <- na.omit(EData_df)


#Sorting the data by stockcode
setkeyv(EData_df, c('StockCode', 'Description'))
head(EData_df,3)
##    InvoiceNo StockCode                Description Quantity     InvoiceDate
## 1:    536370     10002 INFLATABLE POLITICAL GLOBE       48  12/1/2010 8:45
## 2:    536382     10002 INFLATABLE POLITICAL GLOBE       12  12/1/2010 9:45
## 3:    536863     10002 INFLATABLE POLITICAL GLOBE        1 12/3/2010 11:19
##    UnitPrice CustomerID        Country
## 1:      0.85      12583         France
## 2:      0.85      16098 United Kingdom
## 3:      0.85      17967 United Kingdom
itemCode <- unique(EData_df[, c('StockCode', 'Description')])
```

```
head(itemCode,1)
##    StockCode                Description
## 1:     10002 INFLATABLE POLITICAL GLOBE
setkeyv(EData_df, NULL)


#Creation of a buying matrix
cast_df <- dcast(EData_df, CustomerID ~ StockCode, value.var = 'Quantity',fun.
aggregate = sum, fill=0)
head(cast_df[,3504:3508])
##    90133 90135 90136 90138 90141A
## 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
CustId <- cast_df[,1] # Storing the Customer ID's in one table
cast_df <- cast_df[,-c(1,3504:3508)]   #Dropping the columns where customers h
ave not bought an item


for (i in names(cast_df))
cast_df[is.na(get(i)), (i):= 0]


#Conversion to sparse matrix
df_train <- as.matrix(cast_df)
df_train <- df_train[rowSums(df_train) > 5,colSums(df_train) > 5]
df_train <- binarize(as(df_train, "realRatingMatrix"), minRatin = 1)
head(df_train,3)
## 1 x 3461 rating matrix of class 'binaryRatingMatrix' with 1 ratings.
#Splitting the data set into validation and testing set
split_train <- sample(x = c(TRUE, FALSE), size = nrow(df_train),replace = TRUE
, prob = c(0.8, 0.2))
Valid <- df_train[!split_train]
Train <- df_train[split_train]


#Checking the default parameters of the recommender system
recommender_models <- recommenderRegistry$get_entries(dataType ="binaryRatingM
atrix")
recommender_models$IBCF_binaryRatingMatrix$parameters
## $k
## [1] 30
```

```
##
## $method
## [1] "Jaccard"
##
## $normalize_sim_matrix
## [1] FALSE
##
## $alpha
## [1] 0.5
#Setting the parameters
method <- 'IBCF'
parameter <- list(method = 'Jaccard')
n_recommended <- 5
n_training <- 1000

#Building the model
IBCF_model <- Recommender(data = Train, method = method, parameter = parameter
)
model_details <- getModel(IBCF_model)



#Prediction using the validation dataset
IBCF_predicted <-predict(object = IBCF_model, newdata=Valid,n = n_recommended,
 type="topNList")
as(IBCF_predicted,"list")[1:5]
## $`4`
## [1] "21094" "23274" "23512" "23498" "22633"
##
## $`9`
## [1] "23170" "23173" "23171" "23172" "23175"
##
## $`13`
## [1] "23127" "22921" "22447" "22386" "23203"
##
## $`22`
## [1] "17012F" "22384"  "22383"  "17012B" "20728"
##
## $`25`
## [1] "22142" "23552" "23498" "22646" "22699"
#Storing the Customer ID results found through the model
```

```
user1 <- CustId[as.integer(names(IBCF_predicted@items[1]))]
user1
```

```
##     CustomerID
## 1:       12349
```

```
# Items recommeded for User1
vvv <- IBCF_predicted@items[[1]]
vvv
```

```
## [1]  333 2062 2284 2270 1453
```

```
vvv <- rownames(model_details$sim)[vvv]
vvv
```

```
## [1] "21094" "23274" "23512" "23498" "22633"
```

```
itemCode[vvv]
```

```
##     StockCode                       Description
## 1:      21094      SET/6 RED SPOTTY PAPER PLATES
## 2:      23274 STAR T-LIGHT HOLDER WILLIE WINKIE
## 3:      23512      EMBROIDERED RIBBON REEL ROSIE
## 4:      23498              CLASSIC BICYCLE CLIPS
## 5:      22633             HAND WARMER UNION JACK
```

```
user1_buy <- EData_df[CustomerID==12349, sum(Quantity), by=StockCode]
merge(itemCode,user1_buy, by='StockCode')
```

```
##      StockCode                         Description V1
##   1:      20685               DOORMAT RED RETROSPOT  6
##   2:      20914 SET/5 RED RETROSPOT LID GLASS BOWLS  6
##   3:      20914    SET/5 RED SPOTTY LID GLASS BOWLS  6
##   4:      21086          SET/6 RED SPOTTY PAPER CUPS 12
##   5:      21136         PAINTED METAL PEARS ASSORTED 16
##   6:      21231       SWEETHEART CERAMIC TRINKET BOX 36
##   7:      21232       STRAWBERRY CERAMIC TRINKET BOX 36
##   8:      21232       STRAWBERRY CERAMIC TRINKET POT 36
##   9:      21411          GINGHAM HEART  DOORSTOP RED  3
##  10:      21531         RED RETROSPOT SUGAR JAM BOWL  6
##  11:      21533             RETROSPOT LARGE MILK JUG  3
##  12:      21535         RED RETROSPOT SMALL MILK JUG  6
##  13:      21563          RED HEART SHAPE LOVE BUCKET  6
##  14:      21564        PINK  HEART SHAPE LOVE BUCKET  6
##  15:      21787              RAIN PONCHO RETROSPOT 24
##  16:      22059        CERAMIC STRAWBERRY DESIGN MUG 12
##  17:      22064           PINK DOUGHNUT TRINKET POT 12
##  18:      22070       SMALL RED RETROSPOT MUG IN BOX  6
```

```
## 19:     22071    SMALL WHITE RETROSPOT MUG IN BOX  6
## 20:     22131     FOOD CONTAINER SET 3 LOVE HEART  6
## 21:     22195        LARGE HEART MEASURING SPOONS 12
## 22:     22326  ROUND SNACK BOXES SET OF4 WOODLAND  6
## 23:     22333   RETROSPOT PARTY BAG + STICKER SET  8
## 24:     22423            REGENCY CAKESTAND 3 TIER  1
## 25:     22430           ENAMEL WATERING CAN CREAM  4
## 26:     22441   GROW YOUR OWN BASIL IN ENAMEL MUG  8
## 27:     22553              PLASTERS IN TIN SKULLS 12
## 28:     22554    PLASTERS IN TIN WOODLAND ANIMALS 12
## 29:     22555           PLASTERS IN TIN STRONGMAN 12
## 30:     22556       PLASTERS IN TIN CIRCUS PARADE 12
## 31:     22557     PLASTERS IN TIN VINTAGE PAISLEY 12
## 32:     22567             20 DOLLY PEGS RETROSPOT 12
## 33:     22601      CHRISTMAS RETROSPOT ANGEL WOOD 12
## 34:     22666     RECIPE BOX PANTRY YELLOW DESIGN  6
## 35:     22692         DOORMAT WELCOME TO OUR HOME  4
## 36:     22704                     WRAP RED APPLES 25
## 37:     22720    SET OF 3 CAKE TINS PANTRY DESIGN  3
## 38:     22722   SET OF 6 SPICE TINS PANTRY DESIGN  4
## 39:     22832           BROCANTE SHELF WITH HOOKS  2
## 40:     22960            JAM MAKING SET WITH JARS  6
## 41:     23020         GLASS  SONGBIRD STORAGE JAR  1
## 42:     23020          GLASS SONGBIRD STORAGE JAR  1
## 43:     23108          SET OF 10 LED DOLLY LIGHTS  2
## 44:     23112            PARISIENNE CURIO CABINET  2
## 45:     23113               PANTRY CHOPPING BOARD  3
## 46:     23198      PANTRY MAGNETIC  SHOPPING LIST 12
## 47:     23236                  DOILEY BISCUIT TIN  6
## 48:     23236                  DOILEY STORAGE TIN  6
## 49:     23236          STORAGE TIN VINTAGE DOILEY  6
## 50:     23236           STORAGE TIN VINTAGE DOILY  6
## 51:     23240   SET OF 4 KNICK KNACK TINS  DOILEY  6
## 52:     23240    SET OF 4 KNICK KNACK TINS DOILEY  6
## 53:     23240     SET OF 4 KNICK KNACK TINS DOILY  6
## 54:     23253     16 PC CUTLERY SET PANTRY DESIGN  4
## 55:     23253  16 PIECE CUTLERY SET PANTRY DESIGN  4
## 56:     23263   SET OF 3 WOODEN HEART DECORATIONS 12
## 57:     23273  HEART T-LIGHT HOLDER WILLIE WINKIE 12
```

```
## 58:    23283                 DOORMAT VINTAGE LEAF  2
## 59:    23283         DOORMAT VINTAGE LEAVES DESIGN  2
## 60:    23293   SET OF 12 FAIRY CAKE BAKING CASES  8
## 61:    23294    SET OF 6 SNACK LOAF BAKING CASES  8
## 62:    23295    SET OF 12 MINI LOAF BAKING CASES  8
## 63:    23296      SET OF 6 TEA TIME BAKING CASES  8
## 64:    23439          HAND WARMER RED LOVE HEART 12
## 65:    23460                 SWEETHEART WALL TIDY  2
## 66:    23493     VINTAGE DOILY TRAVEL SEWING KIT 10
## 67:    23494     VINTAGE DOILY DELUXE SEWING KIT  3
## 68:    23497        CLASSIC CHROME BICYCLE BELL 12
## 69:    23497         CLASSIC CROME BICYCLE BELL 12
## 70:    23514       EMBROIDERED RIBBON REEL SALLY  6
## 71:    23545                     WRAP RED DOILEY 25
## 72:    23545              WRAP RED VINTAGE DOILY 25
## 73:    35970           ZINC FOLKART SLEIGH BELLS 12
## 74:    37448     CERAMIC CAKE DESIGN SPOTTED MUG 12
## 75:    37500        TEA TIME TEAPOT IN GIFT BOX 12
## 76:   47504H          ENGLISH ROSE SPIRIT LEVEL 12
## 77:    48184                DOORMAT ENGLISH ROSE  6
## 78:    48185                  DOORMAT FAIRY CAKE  4
## 79:    48194                      DOORMAT HEARTS  2
## 80:   84078A     SET/4 WHITE RETRO STORAGE CUBES  1
## 81:    84978    HANGING HEART JAR T-LIGHT HOLDER 12
## 82:   85014A       BLACK/BLUE POLKADOT UMBRELLA  3
## 83:   85014B            RED RETROSPOT UMBRELLA  3
## 84:    85053          FRENCH ENAMEL CANDLEHOLDER  6
## 85:     POST                             POSTAGE  1
##     StockCode                        Description V1
```

REFERENCES

Online Websites

[1] https://www.smartcat.io/blog/2017/improved-r-implementation-of-collaborative-filtering

[2] https://dzone.com/articles/improved-r-implementation-of-collaborative-filtering

[3] https://www.bigcommerce.com/blog/ecommerce/#ecommerce-timeline

Journal Articles

[1] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, Item-Based Collaborative Filtering Recommendation Algorithms, GroupLens Research Group/Army HPC Research Center Department of Computer Science and Engineering University of Minnesota, Minneapolis, MN 5545

[2] George Karypis , Evaluation of Item-Based Top-N Recommendation Algorithms, University of Minnesota, Department of Computer Science and Army HPC Research Center, Minneapolis, MN 55455