**DATA MINING TECHNIQUES USING ASSOCIATION RULE MINING, K-MEANS CLUSTERING, CLASSIFICATION AND SHINY DASHBOARD USING R AND SAS MINER WORKSTATION**

**BY**

**TAIWO OLORUNTELE**

**STUDENT ID: @00609735**

**ADVANCED STATISTICS AND DATA MINING MODULE**

**School of Computing, Science & Engineering**

**University of Salford, Manchester.**

**January 17, 2022**

**ASSOCIATION RULE MINING**

Abstract:

Association rule mining, also known as Market Basket analysis is one of the most important machine learning models available as its main function is to uncover relationships between a wide variety of items in a dataset. Everyday, retailers and industry experts generate large amounts of data and definitely, human beings can not sit to analyze millions of data row by row, which is why Association Rule Mining is a concept. In this report, I have executed Association Rule Mining perfectly, making use of the apriori algorithm on items bought in United Kingdom.

Introduction:

You may walk into Sainsbury’s and discover Bread and Butter or Jam being placed directly beside one another or Milk being placed beside Cereals. This wasn’t done based on just intuition alone. Association Rule mining was employed in this case as items in one various baskets were examined and found Bread and Butter in same, or Milk and Cereals in one. This in turn drives more profit, as well as

It is not only applied in the retail industry but also for recommendation systems, what are people watching on Netflix after they’re done with one movie? Association Rule Mining is also a reason why Spotify has a faster growing database of users than Apple Music, as reviews from people claim it has a better recommendation system.

Aim and Objectives:

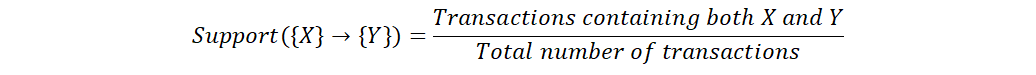
With the market basket analysis, my main aim was to find:-

1. Most frequently bought items
2. Find the best rules using the apriori algorithm by their Lift

Literature review:

The concept of Association Rule Mining is quite a vast one with terminologies one has to understand, in order to get the best results.

Support: This has to do with the frequency of items/transactions in baskets. It is simply calculated by:



Let’s take a look at an example:

Basket 1 = Fanta, Lolli Pop, Chocolates

Basket 2 = Waffles, Fanta, Chocolates , Cutlery

Basket 3 = Lolli pop Corn Flakes, Fanta, Napkins

Basket 4 =Waffles, Fanta, Chocolates, Cutlery

We shall look at the Fanta and Chocolates transactions, this appeared in Basket 1, Basket 2 and Basket 3. So frequency is 3 of a total of 4 baskets.

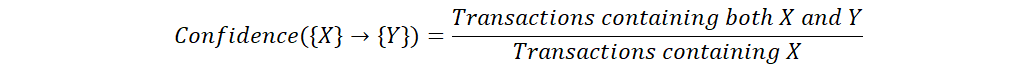
Support = ¾ = 0.75

Taking a look at Waffles and Cutlery, it appears in Basket 2 and 4 of a total of 4 baskets.

Support = 2/4 = 0.5

Technically, you would want to go for items with the highest Support as they can explain the relationships better than ones with lower.

Confidence: This can be explained as the likelihood of occurrence of the consequent on the antecedent. The antecedent here being the first item and the consequent being what was bought after. As above, we can use Fanta as our antecedent and Chocolates as our consequent. Below is how Confidence is calculated:



We find how many times they both occur then divide by the amount of times Fanta occurs. Both occur 3 times and Fanta occurs 4 times, that makes a confidence of 0.75.

Lift: This basically can be explained as the rise in probability of having the consequent with the knowledge of antecedent being present over the probability of having the consequent on the cart without any knowledge about presence of antecedent. In simpler terms as well, Confidence divided by the consequent. It is said to be the most important parameter because when you have your results, you are looking for the ones with the highest Lift. It is mathematically represented below:

Letter

Description automatically generated with low confidence

Back to our example, we have the confidence as 0.75, which we can use to calculate our Lift. Chocolates as our consequent, can be seen to have been bought 3 times out of 4 transactions.

So, 0.75/0.75 =1. Our Lift in this scenario, is 1.

Data: Below is an excerpt of the data to be used for implementation of Association Rule Mining gotten from: <https://archive.ics.uci.edu/ml/datasets/online+retail>

A screenshot of a computer

Description automatically generated with medium confidence

R implementation

Packages used:

For the successful implementation in R, I had to make use of some very important packages namely:

Arules: This is the core foundation and was used to run the apriori algorithm as well as importing in transaction format and data analysis.

Arulesviz: This is an extended package of arules, which is used to visualize association rules and frequent item datasets

Sqldf: This provides the option to include SQL statements to manipulate an R dataframe

Dplyr: This is a set of tools that extensively helps with data manipulation

Readxl: This is used to import files in excel format

VIM: This is used for data cleansing to visualize missing rows in data frame

Data pre-processing:

Upon importing, the data was seen to have an initial 541909 and 8 columns. Inspecting for empty rows using ‘aggr’ from the ‘VIM’ package, I got the below results of 1454 empty rows in the Description column and 135080 in the CustomerID column.

Graphical user interface

Description automatically generated with low confidence

After this, using the ‘filter’ function from the dplyr package, I was able to filter my data to just items bought in just United Kingdom. As it is with Association Rule Mining, you need only two columns and I decided to go with just the InvoiceID and Description columns which contains my list of products.

Graphical user interface, application, Word

Description automatically generated

Now onto cleaning the dataset to be free of empty rows, I employed the use of the sqldf to extract the rows where the Description IS NOT NULL. Once again, I ran the ‘aggr’ function to confirm I was left with no rows.

Analysis:

Now my data has been cleaned, I can run a perfect analysis. Using the pipe argument and ‘count’ function which I used to group each unique item by the amount of times it appeared and it was seen that the WHITE HANGING HEART T-LIGHT HOLDER was purchased 2271 times, followed by the JUMBO BAG RED RETROSPOT which appeared 2001 times.

Text

Description automatically generated

After this, I had to save my clean dataset into a new file which was imported back into R with the ‘read.transactions’ function so it can be read in transactional format. The format for my dataset was in the ‘single’ format, rather than the ‘basket’ format then used the ‘inspect’ function(as it’s in transaction format) to confirm it was imported currently.

Text

Description automatically generated

Onto running my default apriori algorithm, I saw there were no rules as I aanticipated and upon seeing this, I observed the minimum support it executed was 0.1 then I knew I had to make some few changes.

Text, letter

Description automatically generated

Running the apriori algorithm once more, I set my parameters as follows:

Confidence = 0.7

Support = 0.01

Minimum Length = 2

Maximum Length =3

The rationale behind this was that there were very many different items, so therefore I would have a low amount of support and also a minimum of 70% confidence was high enough to give me good results. Below is my result from setting new parameters:

Text, letter

Description automatically generated

As seen above, a total of 104 rules was generated, which was good and okay to work with. After this, I sorted the top 100 rules by Lift and for a clearer view, I exported to a .csv format.

A screenshot of a computer

Description automatically generated with medium confidence

Looking at the results sorted by Lift, it can be seen that the first and second are about the same rules, with same Lift but different Confidence levels so I would go for the one with the higher Confidence level which says people that bought REGENCY TEA PLATE PINK and REGENCY TEA PLATE ROSES also bought REGENCY TEA PLATE GREEN. The next result as well shows the same but with a chain count of 2. My advise to the store after this would be to load up on more REGENCY products.

Earlier, I made a count of the most bought items which happened to be WHITE HANGING HEART T-LIGHT HOLDER, now let’s take a look at results with it as our target variable to know what people are buying alongside it.

Text

Description automatically generated

|  |
| --- |
| 1. CHARLIE + LOLA RED HOT WATER BOTTLE, LARGE WHITE HONEYCOMB PAPER BELL |
| 1. RED WOOLLY HOTTIE WHITE HEART.,VINTAGE BILLBOARD DRINK ME MUG |
| 1. LOCAL CAFE MUG,RED WOOLLY HOTTIE   It can be seen that the three had highest Lift as well as a maximum confidence of 100% meaning that a purchase of any of the three transactions would results into a purchase of the WHITE HANGING HEART T-LIGHT HOLDER. |

SAS Implementation

The SAS Miner Workstation was used as well to run the analysis on same dataset which I saved as a csv from R after cleaning.

On importing, I’ve had to set variables as thus:

Table

Description automatically generated

The Description being my target column as it contains the list of products, InvoiceNo as my ID and a third column listing the rows, which is irrelevant as to why it was selected as Rejected.

As my parameters on R, I set my max chain count to 3 and a confidence level of 70%.

Table

Description automatically generated with medium confidence

Below is the Rules table generated after running:

Diagram

Description automatically generated with medium confidence

1. REGENCY TEA PLATE ROSES & REGENCY TEA PLATE GREEN ==> REGENCY TEA PLATE PINK rule has a Lift of 64 and confidence of 79
2. REGENCY TEA PLATE PINK ==> REGENCY TEA PLATE ROSES & REGENCY TEA PLATE GREEN rule has a lift of 64 and confidence of 82
3. REGENCY TEA PLATE ROSES & REGENCY TEA PLATE PINK ==> REGENCY TEA PLATE GREEN rule has a Lift of 61 and Confidence of 94

COMPARISON

Now let’s compare both results and look for differences or similarities. Both programs produced similar number of transaction counts as it should be.

1. REGENCY TEA PLATE ROSES & REGENCY TEA PLATE GREEN ==> REGENCY TEA PLATE PINK has both confidence as 79 and a Lift of 64 which makes SAS and R results both equal
2. REGENCY TEA PLATE ROSES & REGENCY TEA PLATE PINK ==> REGENCY TEA PLATE GREEN has both confidence as 94 and a Lift of 61 as well, making another very equal comparison.
3. REGENCY TEA PLATE PINK => REGENCY TEA PLATE GREEN has both confidence at 89 and a Lift of 58 as well.
4. REGENCY TEA PLATE GREEN => REGENCY TEA PLATE PINK has both Confidence as 71 and a Lift of 58 once more

Going by this, we can see both R and SAS Miner Workstation produced exact same results for the first 4 rules with same Confidence and Lift.

CLASSIFICATION(DECISION TREE)

Abstract:

Decision Tree is a supervised learning method used for classification models. Here, a target variable is involved to be predicted using features from same dataset. It is one of the simplest classification methods to interpret as it follows simple decision branches till it gets to the final branch. As simple as it is, its main drawback is that it can create over-complex trees(overfitting). Then, you have to employ pruning techniques. By setting the minimum number of samples for the leaf nodes or setting the maximum depth of the tree, you can avoid this. (scikit-learn 2021)

INTRODUCTION

In reality, let us make a prime example of how a decision tree works. For example, you want to purchase a house, there are certain factors you consider.

1. What region is would you like it?
2. Midlands or North England?
3. If Midlands, would you like Birmingham?
4. If Birmingham, would you like a duplex or a flat?
5. If a flat, how many rooms?
6. If 2 rooms, then you buy.

These factors are called ‘Nodes’ in decision trees, so let’s visualize this:

Diagram

Description automatically generated

As shown above, the decisions are broken down into one or more decisions to further influence a final decision.

Decision node: This indicates a point of conditional progression: if a condition is met, then processing continues one way; if not, then another. (Sparxsystems.com 2022)

Leaf node: These are the nodes of a decision tree which have no additional nodes coming off them. They don't split the data any further and this is the final node.

Aim and Objectives:

With the decision tree, my main aim here was to classify salaries below or above 50k based on several factors such as Sex, race, occupation, number of hours worked etc then test the accuracy level.

Literature Review:

Decision Trees had its first application sometime in the 1960’s and has grown to one of the most effective Classification models. With application in various industries, it keeps gaining a widespread use as it is easy to be used, free of ambiguity and works well even with missing values. Target variables could either be discrete or continuous and you’d get a great result. (Ying-Lu 2015)

To get a great model, the following needs to be done:

1. Selection of important variables: When choosing your input variables, you need to be very exact with its important. You definitely wouldn’t want to have a lot of unnecessary branches as the more input variables, the more the branches.
2. Class variables: You would need to have each class accordingly, set the categorical as factors so it could easily be identified
3. Splitting: Here, you would need to split your data into two and have a good ratio for your training and testing datasets. Because you would need to test the accuracy of your model, you would have to split into two. For best practice, anything above a 70-30 split is fine. 70% going to the training dataset and 30% to the testing dataset.
4. Stopping: As mentioned earlier concerning overfitting, this needs to be avoided and can be done by doing the following:

* Specifying the minimum number of records in a leaf
* Specifying the minimum number of records in a node prior to splitting;
* Specifying the depth of any leaf from the root node

Data:

Gotten from <https://archive.ics.uci.edu/ml/datasets/adult>, this is an extraction done by Barry Becker from the 1994 Census database to predict whether an individual makes 50k a year. The dataset was downloaded in the .data format, which is a text format and was imported into Excel after which I added the respective column names. Inspecting the dataset on Excel, there were no missing value and an excerpt of the data is shown below with an explanation of the columns to follow.

A screenshot of a computer

Description automatically generated with medium confidence

Age: Represents the age

Workclass: Represents the sector of work

Fnlgwt: This is Final Weight, the number of people each row in the data represents

Education: Represents their educational achievement

Marital status: Represents if they’re married or not

Occupation: Represents what they do for a living

Race: Represents what racial background they’re from

Sex: Represents their gender

Hours: Represents how many hours they work weekly

Country: Represents what country they belong to

Salary: Represents what they earn here, which is our target variable

R IMPLEMENTATION

Packages used:

Party: With ctree() being the main tool here, the party package is used to implement decision trees

Dplyr: This is a set of tools for data manipulation

Importing the dataset into R with the read.csv this time and using the ‘dim’ function, it was seen to have 32561 rows and 15 columns

A picture containing text

Description automatically generated

With the ‘select’ function from dplyr, I extracted the columns which I needed to use i.e Sex, Race, Hours and Salary. My decision to use this was based on racial and gender discrimination and is supposedly a factor to how much one is paid and Hours, as you tend to earn higher or lower depending on how much hours of work you put in.

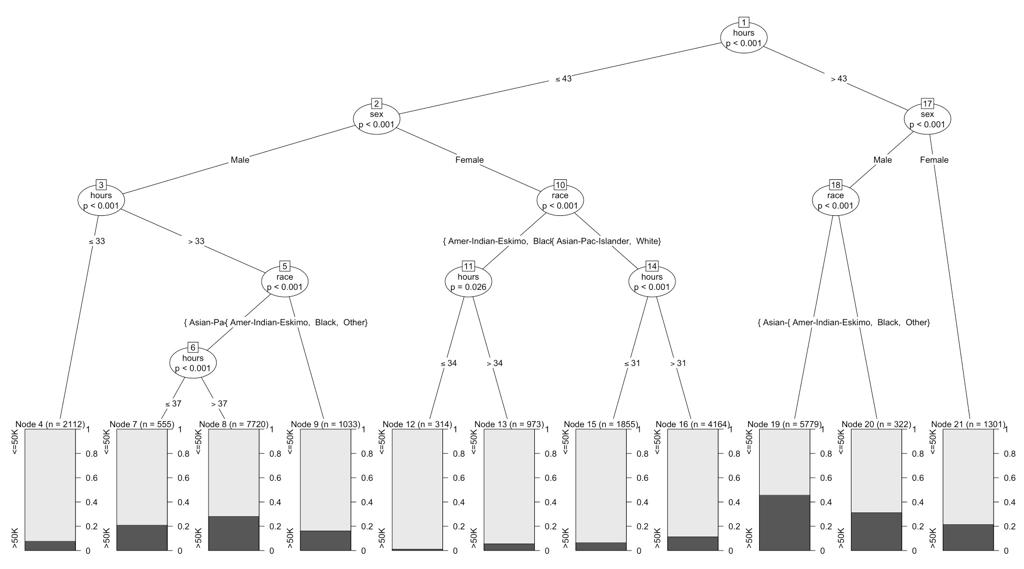
Checking the structure of the data, I noticed the classes were in character format which wouldn’t give the preferred results, so I used the as.factor function to switch the Sex, Race and Salary columns to factors, leaving the Hours as integer.

Text, letter

Description automatically generated

After this, I split the data into my train and test with an 80-20 ration using the ‘sample’ function then extracted my rows for the partition.

Using the ctree function and selecting Sex, Hours and Race to train the tree, I got the below tree:



As the picture above, it can be seen that Node 1, Node 2 and Node 17 are our decision nodes and p-values being below 0.05 represents high significant level.

Taking a look at Node 19 with 5779 observations, it is seen that it contains close to 0.5% of data for Salaries above 50k. Tracing the tree nodes from the top, we can see it falls in the split above 43 hours then a further split into the Male category then the Asian. Suffice to say, if you work over 43 hours and is a Male with Asian background, you’re more likely to earn above 50k.

Taking a look at Node 12 with 314 observations, falls below 43 hours of work then Split into the female category and a further split into the American-Indian race. After this, it goes back to the Hours to classify below or 34 hours of work. Suffice to say, if you work below 43 hours and is a female from the American Indian race and happen to work below 34 hours, you’re more likely to earn below 50k.

Going further to the accuracy calculation, after creating a confusion matrix, I was able to get the training accuracy at 76%

Text

Description automatically generated with low confidence

Because this is not permissible to use as it hasn’t been tested on the tested dataset, I went further to test on the test dataset to get a 75.4% accuracy.

Text

Description automatically generated with medium confidence

This means the model got 75.4% of predictions correct with

an error of 14.6%.

SAS IMPLEMENTATION

Once again, the Decision Tree model was ran on SAS to make for comparison. For SAS, I have two trees i.e the regular Decision Tree and the HP Decision Tree then ran a model comparison for both. I imported my dataset, imported then set the Input and and Target variables as follows as same as R:

A picture containing text, crossword puzzle

Description automatically generated

Thereafter, I used the Data Partition tool to do a partition into Train, Validate and Test with 50%, 30% and 20% respectively.

Table

Description automatically generated

Below is how the data was partitioned:

Table

Description automatically generated

Next, I used the Stat Explore Node to get the highest Variable Worth which happened to be Hours once more, same as R.

Chart

Description automatically generated with medium confidence

Below is my initial Decision Tree :

A picture containing diagram

Description automatically generated

And then my HP Tree(High Precision Decision Tree)

Graphical user interface, application, Word

Description automatically generated

The HP is seen to have more nodes than the regular Decision Tree.

SAS Model Comparison

Table

Description automatically generated

The HPTree has a lower misclassification rate, pointing to a higher accuracy level and also a lower Average Squared error. This makes it a better fit for classification.

SAS and R COMPARISON

From R, I was able to get an accuracy of 75.4% and compared with the HPTree from SAS which stands at 76.2%. It is fairly equal, but with SAS having a slight advantage of about 1%.

**K-MEANS CLUSTERING**

Abstract:

The concept of k-means clustering is based on a set of n points in dimensional space and an integer k and to determine a set of k points in the dimension space known as centers, to achieve minimizinzing the mean squared distance between data points to their nearest center. (A.Y.Wu 2002)

**Introduction:**

K-means clustering is the most known clustering method and argued to be unsupervised or supervised, with not much clear view as at now but most opinions tend to it being unsupervised. It is a very great data mining methodology to group given data with great similarity within the same cluster and the great dissimilarity between different clusters. It can be applied on either discrete or continuous data with ease and utilizes normalization in some cases. To get the number of clusters, one could specify a range of cluster numbers or go with the elbow method.

I have used the k-means clustering on the credit dataset, which was gotten from a list of profiles about how users behaved with their credit cards.

Below are the columns with their meanings:

**CUST**ID : Identification of Credit Card holder (Categorical)  
**BALANCE** : Balance amount left in their account to make purchases (  
**BALANCEFREQUENCY** : How frequently the Balance is updated, score between 0 and 1 (1 = frequently updated, 0 = not frequently updated)  
**PURCHASES** : Amount of purchases made from account  
**ONEOFF**PURCHASES : Maximum purchase amount done in one-go  
**INSTALLMENTSPURCHASES** : Amount of purchase done in installment  
**CASH**ADVANCE : Cash in advance given by the user  
**PURCHASESFREQUENCY** : How frequently the Purchases are being made, score between 0 and 1 (1 = frequently purchased, 0 = not frequently purchased)  
**ONEOFFPURCHASESFREQUENCY** : How frequently Purchases are happening in one-go (1 = frequently purchased, 0 = not frequently purchased)  
**PURCHASESINSTALLMENTSFREQUENCY** : How frequently purchases in installments are being done (1 = frequently done, 0 = not frequently done)  
**CASHADVANCEFREQUENCY** : How frequently the cash in advance being paid  
**CASHADVANCETRX** : Number of Transactions made with "Cash in Advanced"  
**PURCHASES**TRX : Numbe of purchase transactions made  
**CREDITLIMIT** : Limit of Credit Card for user  
**PAYMENTS** : Amount of Payment done by user  
**MINIMUM\_PAYMENTS** : Minimum amount of payments made by user  
**PRCFULLPAYMENT** : Percent of full payment paid by user  
**TENURE** : Tenure of credit card service for user

Literature Review:

In the earlier days, Hierarchical clustering was the method used by biologists and social scientists and cluster analysis became a branch of statistical multivariate analysis.

Aim and Objectives :

* Use K-Means clustering to group entities into clusters based on salary

R IMPLEMENTATION

Packages used :

Dplyr : This is used for data manipulation

Factoextra: This is used to visualize the output of clusters

Cluster: This is the package used for creating clusters

I imported the dataset into R Studio with the read.csv function then selected the following columns for use: BALANCE, PURCHASES, ONEOFF\_PURCHASES, INSTALLMENTS\_PURCHASES, PURCHASES\_FREQUENCY, TENURE.

After which I created Pairs for the new dataset:

Diagram

Description automatically generated

Next, I created a plot to show the relationship between INSTALLMENTS\_PURCHASES and ONEOFF\_PURCHASES

Chart, scatter chart

Description automatically generated

Based on the data being uneven with numbers, I had to normalise the data then I was able to:

* Create a distance matrix and visualize it
* Run hierarchial clustering

Text

Description automatically generated with medium confidence

Now onto the k-means clustering, I used the elbow method to get the number of clusters and from the figure below, the optimal number was 2

Chart, line chart

Description automatically generated

Seeing 2 as my cluster number, I used it to plot my cluster as below:

Chart, scatter chart

Description automatically generated

SAS IMPLEMENTATION

As I have done with the rest, I ran the kmeans clustering on R SAS Miner as well to give the below images:

Graphical user interface, application

Description automatically generated

Two clusters can be seen above with most falling into cluster 1.

After attaching the Segment Profile, I found PURCHASES as my most important variable as shown below:

Graphical user interface

Description automatically generated

Graphical user interface, application

Description automatically generated

**SENTIMENTAL ANALYSIS**

ABSTRACT:

Sentiment analysis in data mining involves the mining of text to extract certain information. This helps businesses to understand the people’s sentiments towards their product or service either from site review, facebook or tweets etc (Gupta 2018)

INTRODUCTION:

Businesses over time need to know what the general public feel about the services or products they offer. They might keep selling and gaining profit but wouldn’t it be nicer to know what people feel about their product to know areas of improvement? This is why Text Mining became popular. It is a very strong tool that can be utilized in almost every aspect, not only businesses but for the purpose of this project, I have used it to analyze the sentiment felt towards various restaurants based on the reviews their customers left behind.

Literature Review:

There are very many processes which could be used in sentimental analysis, you could just want to get a wordcloud of common words, you could also it by list and visualize with a bar chart. Sentiment Analysis

provides information is positive, negative or neutral about a specific topic or product. This is widely termed as Opinion Mining. (Hemamalini 2009)

R IMPLEMENTATION

Packages used:

dplyr: This is mainly used for data manipulation

wordcloud: This generates a wordcloud of words within the dataset

tm: This is the basis of the sentimental analysis, that contains most of the functions used in the process.

Data Preparation and Pre-Preprocessing:

After importation and installing the packages, certain processes needed to have been taken before before being able to get the sentiment analysis and wordcloud.

First, I made a count of all the hotels to find the top 30 hotels:

Table

Description automatically generated

After this, I extracted the top 30 into vector format with which I ran a filter function to extract them from the initial dataset to have a new dataset with only the 30 hotels I needed.

Onto the main processing and data cleaning itself, these were the steps taken:

* Extracting the review column for each hotel into vectors
* Converted all characters to lower case using ‘tolower’ function
* Removed all punctuations from the lower cased character using the ‘gsub’ function
* Removed digits as well with the ‘gsub’function
* Converted every hotel review into corpus, with an example of Aussie Pub Kamala below

Graphical user interface

Description automatically generated

* Removed stopwords from the corpus as they were unnecessary
* Removed white spaces using ‘stripWhitespace’ as I noticed there was a lot of white spaces after removing various characters

ANALYSIS

After the pre-processing and preparation, our data is set to go. I then converted all corpus to a stem document using ‘stemDocument’ function which will be used for the analysis.

To go ahead with the analysis, I made use of the positive and negative lexicon dictionary, which contains a list of positive and negative words which the analysis shall be based upon and graded by percentage. The top words in the dictionary, shown below:

Graphical user interface, text, application

Description automatically generated

Going further, I created a function used to generate a wordcloud calculate total positive and negative words.

Below is a wordcloud from the Da Mario Restaurant then the percentage of negative and positive words generated:

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

From the screenshots above, it can be seen that the Amalfi has the highest positive percentage, at 90% with Alibaba Restaurant being the lowest at 65%. From this, it can be recommended that Alibaba is the best restaurant to visit if looking to eat out.

REFERENCES

A.Y.Wu (2002). "An efficient k-means clustering algorithm: analysis and implementation." 2022, from <http://ieeexplore.ieee.org/document/1017616/#:~:text=Abstract%3A%20In%20k%2Dmeans%20clustering,point%20to%20its%20nearest%20center>.

Gupta, S. (2018). "Sentiment Analysis: Concept, Analysis and Applications." from <https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17#:~:text=Sentiment%20analysis%20is%20contextual%20mining,service%20while%20monitoring%20online%20conversations>.

Hemamalini, S. (2009). "LITERATURE REVIEW ON SENTIMENT ANALYSIS." from <http://www.ijstr.org/final-print/apr2020/Literature-Review-On-Sentiment-Analysis.pdf>.

scikit-learn (2021). "1.10. Decision Trees." Retrieved 2022, from <https://scikit-learn.org/stable/modules/tree.html>.

Sparxsystems.com (2022). "Decision | Enterprise Architect User Guide." from <https://sparxsystems.com/enterprise_architect_user_guide/14.0/model_domains/decision.html>

‌.

Ying-Lu (2015). "Decision tree methods: applications for classification and prediction." 2022, from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4466856/>.

APPENDIX

ASSOCIATION RULE MINING

# set working directory

setwd("C:\\Users\\Oloruntele\\Desktop\\R\\Coursework\\Association Rule")

# confirm working directory

getwd()

# install and load the arules and arulesviz packages

install.packages("arules")

library(arules)

installed.packages("arulesviz")

library(arulesViz)

#install read excel package

install.packages("readxl")

library(readxl)

#read the xlsx file

onlineretail <- read\_excel("Online Retail.xlsx")

#view the file

View(onlineretail)

#installing VIM package

install.packages("VIM")

library(VIM)

#see which columns are empty

aggr(onlineretail, plot = FALSE)

#install dplyr package

install.packages("dplyr")

library(dplyr)

#filter the country to united kingdom

onlineretail <- filter(onlineretail, Country == "United Kingdom")

#select columns to use

onlineretail <- select(onlineretail, InvoiceNo, Description)

View(onlineretail)

#install sqldf

install.packages("sqldf")

library(sqldf)

#filter out empty Description rows

onlineretail <- sqldf("SELECT InvoiceNo, Description FROM onlineretail WHERE Description IS NOT NULL")

#check empty rows again

aggr(onlineretail, plot = FALSE)

#View the data

View(onlineretail)

#see the most frequent products bought

onlineretail %>% count(Description, sort = TRUE)

summary(onlineretail)

str(onlineretail)

#write into a new csv

write.csv(onlineretail, "onlineretailUK.csv")

# read the dataset in transaction format

onlineretailUK <- read.transactions("onlineretailUK.csv", format = "single", sep = "," , cols = 2:3)

#inspect top 10 elements

inspect(head(onlineretailUK,10))

#run first apriori algorithm

rules <- apriori(onlineretailUK)

#specify 70% confidence and support of 0.01 and minlen of 2 with maxlen of 3

rules <- apriori(onlineretailUK, parameter = list(conf = 0.7, supp = 0.01, minlen = 2, maxlen = 3))

inspect(rules)

#check rules for most bought item

rules1 <- apriori(onlineretailUK, parameter = list(conf = 0.7, supp = 0.001, minlen = 2, maxlen = 3), appearance = list(rhs = c("WHITE HANGING HEART T-LIGHT HOLDER"), default = "lhs"))

inspect(rules1)

#sort rules1 by lift

rules1bylift <- head(rules1, n = 100, by = "lift")

inspect(rules1bylift)

#sort rules by lift

rulesbylift <- head(rules, n = 100, by = "lift")

inspect(rulesbylift)

write(rulesbylift, "rulesbylift.csv", sep = ",")

write(rules1bylift, "rules1bylift.csv", sep = ",")

DECISION TREE

#Set working directory

setwd("C:\\Users\\Oloruntele\\Desktop\\R\\Coursework\\Decision Tree")

#confirm working directory

getwd()

#install packages needed

install.packages("party")

library(party)

install.packages("dplyr")

library(dplyr)

#importing data

adult <- read.csv("adult.csv", header = TRUE)

#number of rows and columns

dim(adult)

#View imported data

View(adult)

#Extract needed columns

adult <- select(adult, sex, race, hours, salary)

#View new data

View(adult)

#structure of data

str(adult)

#change class to factor

adult$sex <- as.factor(adult$sex)

adult$race <- as.factor(adult$race)

adult$salary <- as.factor(adult$salary)

#split dataset into two

set.seed(65)

split <- sample(2, nrow(adult), replace = TRUE, prob = c(0.8,0.2))

#extract rows for training and testing dataset

train <- adult[split==1,]

test <- adult[split==2,]

#tree for training dataset

train\_tree <- ctree(salary ~ sex + race + hours, data = train)

print(train\_tree)

plot(train\_tree, type = "simple")

#make predictions for tree

predict(train\_tree)

#generate frequency table

freq\_train <- table(predict(train\_tree), train$salary)

#print table

print(freq\_train)

#calculate accuracy of train

sum(diag(freq\_train))/sum(freq\_train)

#tabulating prediction on test set

freq\_test <- table(predict(train\_tree, newdata = test), test$salary)

print(freq\_test)

#calculate accuracy on test set

sum(diag(freq\_test))/sum(freq\_test)

CLUSTERING

creditcard <- read.csv("CreditCard.csv", header = T)

View(creditcard)

#selecting columns

creditcard <- select(creditcard, BALANCE, PURCHASES, ONEOFF\_PURCHASES,

INSTALLMENTS\_PURCHASES, PURCHASES\_FREQUENCY, TENURE)

#save as csv

write.csv(creditcard, "credit.csv")

#install packages

install.packages("dplyr")

library(dplyr)

install.packages("cluster")

library(cluster)

install.packages("factoextra")

library(factoextra)

#create pairs

pairs(creditcard)

#see relationship between installment purchases and one off purchases

plot(INSTALLMENTS\_PURCHASES ~ ONEOFF\_PURCHASES, data = creditcard)

#normalise function

normalise <- function(df)

{

return(((df- min(df)) /(max(df)-min(df))\*(1-0))+0)

}

#normalise data

creditcard\_norm<-as.data.frame(lapply(creditcard,normalise))

View(creditcard\_norm)

#create distance matrix

creditcard\_dist <- dist(creditcard\_norm,method = "euclidean",)

print(creditcard\_dist)

#visualize distance matrix

fviz\_dist(creditcard\_dist)

#hiearchal clustering

creditcard.hclust <- hclust(creditcard\_dist)

creditcard.hclust

#get number of clusters with elbow

fviz\_nbclust(creditcard\_norm, kmeans, method = "wss")

#perform clustering

set.seed(55)

creditcard\_kmeans <- kmeans(creditcard\_norm, 2, nstart = 30)

creditcard\_kmeans$cluster

creditcard\_kmeans$size

#cluster plot

fviz\_cluster(creditcard\_kmeans,creditcard\_norm)

SENTIMENT ANALYSIS

setwd("C:\\Users\\Oloruntele\\Desktop\\R\\Test Coursework\\hotel reviews\\")

getwd()

#Reading csv file

hotelreviews <- read.csv("tourist\_accommodation\_reviews.csv", header = T)

View(hotelreviews)

#install packages

install.packages("dplyr")

library(dplyr)

install.packages("wordcloud")

library(wordcloud)

install.packages("tm")

library(tm)

#Counting number of hotels

Hotel\_Count <- hotelreviews %>% count(hotelreviews$Hotel.Restaurant.name, sort = TRUE)

View(Hotel\_Count)

#extracting top 30 hotels into vector format

ThirtyHotels <- Hotel\_Count[1:30,1]

ThirtyHotels

#extracting list of hotels from initial dataset

ThirtyHotels <- filter(hotelreviews, hotelreviews$Hotel.Restaurant.name %in% ThirtyHotels)

View(ThirtyHotels)

#filtering each hotel out separately

Da\_Mario <- subset(hotelreviews, Hotel.Restaurant.name == "Da Mario")

No\_6\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "No.6 Restaurant" )

Sabai\_Sabai <- subset(hotelreviews, Hotel.Restaurant.name == "Sabai Sabai" )

The\_Pizza\_Company <- subset(hotelreviews, Hotel.Restaurant.name == "The Pizza Company" )

The\_Family\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "The Family Restaurant" )

Outdoor\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Outdoor Restaurant" )

gether\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "2gether Restaurant" )

Spices\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "3 Spices Restaurant" )

Bar360 <- subset(hotelreviews, Hotel.Restaurant.name == "360 ° Bar" )

Sea\_Breeze <- subset(hotelreviews, Hotel.Restaurant.name == "9' Sea Breeze" )

Seafood <- subset(hotelreviews, Hotel.Restaurant.name == "99 Seafood" )

A\_Spoonful\_of\_Sugar <- subset(hotelreviews, Hotel.Restaurant.name == "A Spoonful of Sugar" )

Acqua\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Acqua Restaurant" )

After\_Beach\_Bar <- subset(hotelreviews, Hotel.Restaurant.name == "After Beach Bar" )

Albatross\_Cafe <- subset(hotelreviews, Hotel.Restaurant.name == "Albatross Cafe" )

Ali\_Baba\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Ali Baba Restaurant")

Amalfi <- subset(hotelreviews, Hotel.Restaurant.name == "Amalfi" )

Anchor\_Inn <- subset(hotelreviews, Hotel.Restaurant.name == "Anchor Inn" )

Andaman\_Grill <- subset(hotelreviews, Hotel.Restaurant.name == "Andaman Grill" )

Angus\_OTools\_Irish\_Pub <- subset(hotelreviews, Hotel.Restaurant.name == "Angus O'Tool's Irish Pub" )

Anns\_Kitchen\_Bar\_and\_Grill <- subset(hotelreviews, Hotel.Restaurant.name == "Ann's Kitchen Bar and Grill" )

Ann\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Ann Restaurant" )

Ao\_Chalong\_Yacht\_Club\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Ao Chalong Yacht Club Restaurant" )

Arabia\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Arabia Restaurant" )

Atsumi\_Raw\_Cafe <- subset(hotelreviews, Hotel.Restaurant.name == "Atsumi Raw Cafe" )

Audy\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "Audy Restaurant" )

Aussie\_Pub\_Kamala <- subset(hotelreviews, Hotel.Restaurant.name == "Aussie Pub Kamala" )

Autogrill\_Risto\_Bar\_Pizza <- subset(hotelreviews, Hotel.Restaurant.name == "Autogrill Risto Bar Pizza" )

BHive\_Gallery\_Bar\_and\_Restaurant <- subset(hotelreviews, Hotel.Restaurant.name == "B-Hive Gallery, Bar and Restaurant" )

#See first reviews of Aussie Pub Kama

head(Aussie\_Pub\_Kamala$Review)

#Extracting hotel reviews into text vectors

Da\_Mario\_review <- Da\_Mario$Review

No\_6\_Restaurant\_review <- No\_6\_Restaurant$Review

Sabai\_Sabai\_review <- Sabai\_Sabai$Review

The\_Pizza\_Company\_review <- The\_Pizza\_Company$Review

The\_Family\_Restaurant\_review <- The\_Family\_Restaurant$Review

Outdoor\_Restaurant\_review <- Outdoor\_Restaurant$Review

gether\_Restaurant\_review <- gether\_Restaurant$Review

Spices\_Restaurant\_review <- Spices\_Restaurant$Review

Bar360\_review <- Bar360$Review

Sea\_Breeze\_review <- Sea\_Breeze$Review

Seafood\_reiew <- Seafood$Review

A\_Spoonful\_of\_Sugar\_review <- A\_Spoonful\_of\_Sugar$Review

Acqua\_Restaurant\_review <- Acqua\_Restaurant$Review

After\_Beach\_Bar\_review <- After\_Beach\_Bar$Review

Albatross\_Cafe\_review <- Albatross\_Cafe$Review

Ali\_Baba\_Restaurant\_review <- Ali\_Baba\_Restaurant$Review

Amalfi\_review <- Amalfi$Review

Anchor\_Inn\_review <- Anchor\_Inn$Review

Andaman\_Grill\_review <- Andaman\_Grill$Review

Angus\_OTools\_Irish\_Pub\_review <- Angus\_OTools\_Irish\_Pub$Review

Anns\_Kitchen\_Bar\_and\_Grill\_review <- Anns\_Kitchen\_Bar\_and\_Grill$Review

Ann\_Restaurant\_review <- Ann\_Restaurant$Review

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review <- Ao\_Chalong\_Yacht\_Club\_Restaurant$Review

Arabia\_Restaurant\_review <- Arabia\_Restaurant$Review

Atsumi\_Raw\_Cafe\_review <- Atsumi\_Raw\_Cafe$Review

Audy\_Restaurant\_review <- Audy\_Restaurant$Review

Aussie\_Pub\_Kamala\_review <- Aussie\_Pub\_Kamala$Review

Autogrill\_Risto\_Bar\_Pizza\_review <- Autogrill\_Risto\_Bar\_Pizza$Review

BHive\_Gallery\_Bar\_and\_Restaurant\_review <- BHive\_Gallery\_Bar\_and\_Restaurant$Review

#converting characters to lower case

Da\_Mario\_review <- tolower(Da\_Mario\_review)

No\_6\_Restaurant\_review <- tolower(No\_6\_Restaurant\_review)

Sabai\_Sabai\_review <- tolower(Sabai\_Sabai\_review)

The\_Pizza\_Company\_review <- tolower(Sabai\_Sabai\_review)

The\_Family\_Restaurant\_review <- tolower(The\_Family\_Restaurant\_review)

Outdoor\_Restaurant\_review <- tolower(Outdoor\_Restaurant\_review)

gether\_Restaurant\_review <- tolower(gether\_Restaurant\_review)

Spices\_Restaurant\_review <- tolower(Spices\_Restaurant\_review)

Bar360\_review <- tolower(Bar360\_review)

Sea\_Breeze\_review <- tolower(Sea\_Breeze\_review)

Seafood\_reiew <- tolower(Seafood\_reiew)

A\_Spoonful\_of\_Sugar\_review <- tolower(A\_Spoonful\_of\_Sugar\_review)

Acqua\_Restaurant\_review <- tolower(Acqua\_Restaurant\_review)

After\_Beach\_Bar\_review <- tolower(After\_Beach\_Bar\_review)

Albatross\_Cafe\_review <- tolower(Albatross\_Cafe\_review)

Ali\_Baba\_Restaurant\_review <- tolower(Ali\_Baba\_Restaurant\_review)

Amalfi\_review <- tolower(Amalfi\_review)

Anchor\_Inn\_review <- tolower(Anchor\_Inn\_review)

Andaman\_Grill\_review <- tolower(Andaman\_Grill\_review)

Angus\_OTools\_Irish\_Pub\_review <- tolower(Angus\_OTools\_Irish\_Pub\_review)

Anns\_Kitchen\_Bar\_and\_Grill\_review <- tolower(Anns\_Kitchen\_Bar\_and\_Grill\_review)

Ann\_Restaurant\_review <- tolower(Ann\_Restaurant\_review)

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review <- tolower(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review)

Arabia\_Restaurant\_review <- tolower(Arabia\_Restaurant\_review)

Atsumi\_Raw\_Cafe\_review <- tolower(Atsumi\_Raw\_Cafe\_review)

Audy\_Restaurant\_review <- tolower(Audy\_Restaurant\_review)

Aussie\_Pub\_Kamala\_review <- tolower(Aussie\_Pub\_Kamala\_review)

Autogrill\_Risto\_Bar\_Pizza\_review <- tolower(Autogrill\_Risto\_Bar\_Pizza\_review)

BHive\_Gallery\_Bar\_and\_Restaurant\_review <- tolower(BHive\_Gallery\_Bar\_and\_Restaurant\_review)

#confirm

head(BHive\_Gallery\_Bar\_and\_Restaurant\_review)

#Remove punctuations from the reviews

Da\_Mario\_review <- gsub("[[:punct:]]", "", Da\_Mario\_review)

No\_6\_Restaurant\_review <- gsub("[[:punct:]]", "", No\_6\_Restaurant\_review)

Sabai\_Sabai\_review <- gsub("[[:punct:]]", "", Sabai\_Sabai\_review)

The\_Pizza\_Company\_review <- gsub("[[:punct:]]", "", The\_Pizza\_Company\_review)

The\_Family\_Restaurant\_review <- gsub("[[:punct:]]", "", The\_Family\_Restaurant\_review)

Outdoor\_Restaurant\_review <- gsub("[[:punct:]]", "", Outdoor\_Restaurant\_review)

gether\_Restaurant\_review <- gsub("[[:punct:]]", "", gether\_Restaurant\_review)

Spices\_Restaurant\_review <- gsub("[[:punct:]]", "", Spices\_Restaurant\_review)

Bar360\_review <- gsub("[[:punct:]]", "", Bar360\_review)

Sea\_Breeze\_review <- gsub("[[:punct:]]", "", Sea\_Breeze\_review)

Seafood\_reiew <-gsub("[[:punct:]]", "", Seafood\_reiew)

A\_Spoonful\_of\_Sugar\_review <- gsub("[[:punct:]]", "", A\_Spoonful\_of\_Sugar\_review)

Acqua\_Restaurant\_review <- gsub("[[:punct:]]", "", Acqua\_Restaurant\_review)

After\_Beach\_Bar\_review <- gsub("[[:punct:]]", "", After\_Beach\_Bar\_review)

Albatross\_Cafe\_review <- gsub("[[:punct:]]", "", Albatross\_Cafe\_review)

Ali\_Baba\_Restaurant\_review <- gsub("[[:punct:]]", "", Ali\_Baba\_Restaurant\_review)

Amalfi\_review <- gsub("[[:punct:]]", "", Amalfi\_review)

Anchor\_Inn\_review <- gsub("[[:punct:]]", "", Anchor\_Inn\_review)

Andaman\_Grill\_review <- gsub("[[:punct:]]", "", Andaman\_Grill\_review)

Angus\_OTools\_Irish\_Pub\_review <- gsub("[[:punct:]]", "", Angus\_OTools\_Irish\_Pub\_review)

Anns\_Kitchen\_Bar\_and\_Grill\_review <- gsub("[[:punct:]]", "", Anns\_Kitchen\_Bar\_and\_Grill\_review)

Ann\_Restaurant\_review <- gsub("[[:punct:]]", "", Ann\_Restaurant\_review)

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review <- gsub("[[:punct:]]", "", Ao\_Chalong\_Yacht\_Club\_Restaurant\_review)

Arabia\_Restaurant\_review <- gsub("[[:punct:]]", "", Arabia\_Restaurant\_review)

Atsumi\_Raw\_Cafe\_review <- gsub("[[:punct:]]", "", Atsumi\_Raw\_Cafe\_review)

Audy\_Restaurant\_review <- gsub("[[:punct:]]", "", Audy\_Restaurant\_review)

Aussie\_Pub\_Kamala\_review <- gsub("[[:punct:]]", "", Aussie\_Pub\_Kamala\_review)

Autogrill\_Risto\_Bar\_Pizza\_review <- gsub("[[:punct:]]", "", Autogrill\_Risto\_Bar\_Pizza\_review)

BHive\_Gallery\_Bar\_and\_Restaurant\_review <- gsub("[[:punct:]]", "", BHive\_Gallery\_Bar\_and\_Restaurant\_review)

#Remove digits from the reviews

Da\_Mario\_review <- gsub("[[:digit:]]", "", Da\_Mario\_review)

No\_6\_Restaurant\_review <- gsub("[[:digit:]]", "", No\_6\_Restaurant\_review)

Sabai\_Sabai\_review <- gsub("[[:digit:]]", "", Sabai\_Sabai\_review)

The\_Pizza\_Company\_review <- gsub("[[:digit:]]", "", The\_Pizza\_Company\_review)

The\_Family\_Restaurant\_review <- gsub("[[:digit:]]", "", The\_Family\_Restaurant\_review)

Outdoor\_Restaurant\_review <- gsub("[[:digit:]]", "", Outdoor\_Restaurant\_review)

gether\_Restaurant\_review <- gsub("[[:digit:]]", "", gether\_Restaurant\_review)

Spices\_Restaurant\_review <- gsub("[[:digit:]]", "", Spices\_Restaurant\_review)

Bar360\_review <- gsub("[[:digit:]]", "", Bar360\_review)

Sea\_Breeze\_review <- gsub("[[:digit:]]", "", Sea\_Breeze\_review)

Seafood\_reiew <-gsub("[[:digit:]]", "", Seafood\_reiew)

A\_Spoonful\_of\_Sugar\_review <- gsub("[[:digit:]]", "", A\_Spoonful\_of\_Sugar\_review)

Acqua\_Restaurant\_review <- gsub("[[:digit:]]", "", Acqua\_Restaurant\_review)

After\_Beach\_Bar\_review <- gsub("[[:digit:]]", "", After\_Beach\_Bar\_review)

Albatross\_Cafe\_review <- gsub("[[:digit:]]", "", Albatross\_Cafe\_review)

Ali\_Baba\_Restaurant\_review <- gsub("[[:digit:]]", "", Ali\_Baba\_Restaurant\_review)

Amalfi\_review <- gsub("[[:digit:]]", "", Amalfi\_review)

Anchor\_Inn\_review <- gsub("[[:digit:]]", "", Anchor\_Inn\_review)

Andaman\_Grill\_review <- gsub("[[:digit:]]", "", Andaman\_Grill\_review)

Angus\_OTools\_Irish\_Pub\_review <- gsub("[[:digit:]]", "", Angus\_OTools\_Irish\_Pub\_review)

Anns\_Kitchen\_Bar\_and\_Grill\_review <- gsub("[[:digit:]]", "", Anns\_Kitchen\_Bar\_and\_Grill\_review)

Ann\_Restaurant\_review <- gsub("[[:digit:]]", "", Ann\_Restaurant\_review)

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review <- gsub("[[:digit:]]", "", Ao\_Chalong\_Yacht\_Club\_Restaurant\_review)

Arabia\_Restaurant\_review <- gsub("[[:digit:]]", "", Arabia\_Restaurant\_review)

Atsumi\_Raw\_Cafe\_review <- gsub("[[:digit:]]", "", Atsumi\_Raw\_Cafe\_review)

Audy\_Restaurant\_review <- gsub("[[:digit:]]", "", Audy\_Restaurant\_review)

Aussie\_Pub\_Kamala\_review <- gsub("[[:digit:]]", "", Aussie\_Pub\_Kamala\_review)

Autogrill\_Risto\_Bar\_Pizza\_review <- gsub("[[:digit:]]", "", Autogrill\_Risto\_Bar\_Pizza\_review)

BHive\_Gallery\_Bar\_and\_Restaurant\_review <- gsub("[[:digit:]]", "", BHive\_Gallery\_Bar\_and\_Restaurant\_review)

#confirm

head(Autogrill\_Risto\_Bar\_Pizza\_review)

#convert to corpus

Da\_Mario\_review\_corpus <- Corpus(VectorSource(Da\_Mario\_review))

No\_6\_Restaurant\_review\_corpus <- Corpus(VectorSource(No\_6\_Restaurant\_review))

Sabai\_Sabai\_review\_corpus <- Corpus(VectorSource(Sabai\_Sabai\_review))

The\_Pizza\_Company\_review\_corpus <- Corpus(VectorSource(The\_Pizza\_Company\_review))

The\_Family\_Restaurant\_review\_corpus <- Corpus(VectorSource(The\_Family\_Restaurant\_review))

Outdoor\_Restaurant\_review\_corpus <- Corpus(VectorSource(Outdoor\_Restaurant\_review))

gether\_Restaurant\_review\_corpus <- Corpus(VectorSource(gether\_Restaurant\_review))

Spices\_Restaurant\_review\_corpus <- Corpus(VectorSource(Spices\_Restaurant\_review))

Bar360\_review\_corpus <- Corpus(VectorSource(Bar360\_review))

Sea\_Breeze\_review\_corpus <- Corpus(VectorSource(Sea\_Breeze\_review))

Seafood\_reiew\_corpus <- Corpus(VectorSource(Seafood\_reiew))

A\_Spoonful\_of\_Sugar\_review\_corpus <- Corpus(VectorSource(A\_Spoonful\_of\_Sugar\_review))

Acqua\_Restaurant\_review\_corpus <- Corpus(VectorSource(Acqua\_Restaurant\_review))

After\_Beach\_Bar\_review\_corpus <- Corpus(VectorSource(After\_Beach\_Bar\_review))

Albatross\_Cafe\_review\_corpus <- Corpus(VectorSource(Albatross\_Cafe\_review))

Ali\_Baba\_Restaurant\_review\_corpus <- Corpus(VectorSource(Ali\_Baba\_Restaurant\_review))

Amalfi\_review\_corpus <- Corpus(VectorSource(Amalfi\_review))

Anchor\_Inn\_review\_corpus <- Corpus(VectorSource(Anchor\_Inn\_review))

Andaman\_Grill\_review\_corpus <- Corpus(VectorSource(Andaman\_Grill\_review))

Angus\_OTools\_Irish\_Pub\_review\_corpus <- Corpus(VectorSource(Angus\_OTools\_Irish\_Pub\_review))

Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus <- Corpus(VectorSource(Anns\_Kitchen\_Bar\_and\_Grill\_review))

Ann\_Restaurant\_review\_corpus <- Corpus(VectorSource(Ann\_Restaurant\_review))

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus <- Corpus(VectorSource(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review))

Arabia\_Restaurant\_review\_corpus <- Corpus(VectorSource(Arabia\_Restaurant\_review))

Atsumi\_Raw\_Cafe\_review\_corpus <- Corpus(VectorSource(Atsumi\_Raw\_Cafe\_review))

Audy\_Restaurant\_review\_corpus <- Corpus(VectorSource(Audy\_Restaurant\_review))

Aussie\_Pub\_Kamala\_review\_corpus <- Corpus(VectorSource(Aussie\_Pub\_Kamala\_review))

Autogrill\_Risto\_Bar\_Pizza\_review\_corpus <- Corpus(VectorSource(Autogrill\_Risto\_Bar\_Pizza\_review))

BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus <- Corpus(VectorSource(BHive\_Gallery\_Bar\_and\_Restaurant\_review))

#confirm

Aussie\_Pub\_Kamala\_review\_corpus

#remove stopwords from corpus

Da\_Mario\_review\_corpus <- tm\_map(Da\_Mario\_review\_corpus, removeWords,stopwords("english"))

No\_6\_Restaurant\_review\_corpus <- tm\_map(No\_6\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Sabai\_Sabai\_review\_corpus <- tm\_map(Sabai\_Sabai\_review\_corpus, removeWords,stopwords("english"))

The\_Pizza\_Company\_review\_corpus <- tm\_map(The\_Pizza\_Company\_review\_corpus, removeWords,stopwords("english"))

The\_Family\_Restaurant\_review\_corpus <- tm\_map(The\_Family\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Outdoor\_Restaurant\_review\_corpus <- tm\_map(Outdoor\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

gether\_Restaurant\_review\_corpus <- tm\_map(gether\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Spices\_Restaurant\_review\_corpus <- tm\_map(Spices\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Bar360\_review\_corpus <- tm\_map(Bar360\_review\_corpus, removeWords,stopwords("english"))

Sea\_Breeze\_review\_corpus <- tm\_map(Sea\_Breeze\_review\_corpus, removeWords,stopwords("english"))

Seafood\_reiew\_corpus <- tm\_map(Seafood\_reiew\_corpus, removeWords,stopwords("english"))

A\_Spoonful\_of\_Sugar\_review\_corpus <- tm\_map(A\_Spoonful\_of\_Sugar\_review\_corpus, removeWords,stopwords("english"))

Acqua\_Restaurant\_review\_corpus <- tm\_map(Acqua\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

After\_Beach\_Bar\_review\_corpus <- tm\_map(After\_Beach\_Bar\_review\_corpus, removeWords,stopwords("english"))

Albatross\_Cafe\_review\_corpus <- tm\_map(Albatross\_Cafe\_review\_corpus, removeWords,stopwords("english"))

Ali\_Baba\_Restaurant\_review\_corpus <- tm\_map(Ali\_Baba\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Amalfi\_review\_corpus <- tm\_map(Amalfi\_review\_corpus, removeWords,stopwords("english"))

Anchor\_Inn\_review\_corpus <- tm\_map(Anchor\_Inn\_review\_corpus, removeWords,stopwords("english"))

Andaman\_Grill\_review\_corpus <- tm\_map(Andaman\_Grill\_review\_corpus, removeWords,stopwords("english"))

Angus\_OTools\_Irish\_Pub\_review\_corpus <- tm\_map(Angus\_OTools\_Irish\_Pub\_review\_corpus, removeWords,stopwords("english"))

Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus <- tm\_map(Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus, removeWords,stopwords("english"))

Ann\_Restaurant\_review\_corpus <- tm\_map(Ann\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus <- tm\_map(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Arabia\_Restaurant\_review\_corpus <- tm\_map(Arabia\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Atsumi\_Raw\_Cafe\_review\_corpus <- tm\_map(Atsumi\_Raw\_Cafe\_review\_corpus, removeWords,stopwords("english"))

Audy\_Restaurant\_review\_corpus <- tm\_map(Audy\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

Aussie\_Pub\_Kamala\_review\_corpus <- tm\_map(Aussie\_Pub\_Kamala\_review\_corpus, removeWords,stopwords("english"))

Autogrill\_Risto\_Bar\_Pizza\_review\_corpus <- tm\_map(Autogrill\_Risto\_Bar\_Pizza\_review\_corpus, removeWords,stopwords("english"))

BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus <- tm\_map(BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus, removeWords,stopwords("english"))

#inspect for confirmation

inspect(Atsumi\_Raw\_Cafe\_review\_corpus)

#strip white spaces

Da\_Mario\_review\_corpus <- tm\_map(Da\_Mario\_review\_corpus, stripWhitespace)

No\_6\_Restaurant\_review\_corpus <- tm\_map(No\_6\_Restaurant\_review\_corpus, stripWhitespace)

Sabai\_Sabai\_review\_corpus <- tm\_map(Sabai\_Sabai\_review\_corpus, stripWhitespace)

The\_Pizza\_Company\_review\_corpus <- tm\_map(The\_Pizza\_Company\_review\_corpus, stripWhitespace)

The\_Family\_Restaurant\_review\_corpus <- tm\_map(The\_Family\_Restaurant\_review\_corpus, stripWhitespace)

Outdoor\_Restaurant\_review\_corpus <- tm\_map(Outdoor\_Restaurant\_review\_corpus, stripWhitespace)

gether\_Restaurant\_review\_corpus <- tm\_map(gether\_Restaurant\_review\_corpus, stripWhitespace)

Spices\_Restaurant\_review\_corpus <- tm\_map(Spices\_Restaurant\_review\_corpus, stripWhitespace)

Bar360\_review\_corpus <- tm\_map(Bar360\_review\_corpus, stripWhitespace)

Sea\_Breeze\_review\_corpus <- tm\_map(Sea\_Breeze\_review\_corpus, stripWhitespace)

Seafood\_reiew\_corpus <- tm\_map(Seafood\_reiew\_corpus, stripWhitespace)

A\_Spoonful\_of\_Sugar\_review\_corpus <- tm\_map(A\_Spoonful\_of\_Sugar\_review\_corpus, stripWhitespace)

Acqua\_Restaurant\_review\_corpus <- tm\_map(Acqua\_Restaurant\_review\_corpus, stripWhitespace)

After\_Beach\_Bar\_review\_corpus <- tm\_map(After\_Beach\_Bar\_review\_corpus, stripWhitespace)

Albatross\_Cafe\_review\_corpus <- tm\_map(Albatross\_Cafe\_review\_corpus, stripWhitespace)

Ali\_Baba\_Restaurant\_review\_corpus <- tm\_map(Ali\_Baba\_Restaurant\_review\_corpus, stripWhitespace)

Amalfi\_review\_corpus <- tm\_map(Amalfi\_review\_corpus, stripWhitespace)

Anchor\_Inn\_review\_corpus <- tm\_map(Anchor\_Inn\_review\_corpus, stripWhitespace)

Andaman\_Grill\_review\_corpus <- tm\_map(Andaman\_Grill\_review\_corpus, stripWhitespace)

Angus\_OTools\_Irish\_Pub\_review\_corpus <- tm\_map(Angus\_OTools\_Irish\_Pub\_review\_corpus, stripWhitespace)

Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus <- tm\_map(Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus, stripWhitespace)

Ann\_Restaurant\_review\_corpus <- tm\_map(Ann\_Restaurant\_review\_corpus, stripWhitespace)

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus <- tm\_map(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus, stripWhitespace)

Arabia\_Restaurant\_review\_corpus <- tm\_map(Arabia\_Restaurant\_review\_corpus, stripWhitespace)

Atsumi\_Raw\_Cafe\_review\_corpus <- tm\_map(Atsumi\_Raw\_Cafe\_review\_corpus, stripWhitespace)

Audy\_Restaurant\_review\_corpus <- tm\_map(Audy\_Restaurant\_review\_corpus, stripWhitespace)

Aussie\_Pub\_Kamala\_review\_corpus <- tm\_map(Aussie\_Pub\_Kamala\_review\_corpus, stripWhitespace)

Autogrill\_Risto\_Bar\_Pizza\_review\_corpus <- tm\_map(Autogrill\_Risto\_Bar\_Pizza\_review\_corpus, stripWhitespace)

BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus <- tm\_map(BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus, stripWhitespace)

#inspect

inspect(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus)

#create stem document

Da\_Mario\_review\_corpus\_stem <- tm\_map(Da\_Mario\_review\_corpus, stemDocument)

No\_6\_Restaurant\_review\_corpus\_stem <- tm\_map(No\_6\_Restaurant\_review\_corpus, stemDocument)

Sabai\_Sabai\_review\_corpus\_stem <- tm\_map(Sabai\_Sabai\_review\_corpus, stemDocument)

The\_Pizza\_Company\_review\_corpus\_stem <- tm\_map(The\_Pizza\_Company\_review\_corpus, stemDocument)

The\_Family\_Restaurant\_review\_corpus\_stem <- tm\_map(The\_Family\_Restaurant\_review\_corpus, stemDocument)

Outdoor\_Restaurant\_review\_corpus\_stem <- tm\_map(Outdoor\_Restaurant\_review\_corpus, stemDocument)

gether\_Restaurant\_review\_corpus\_stem <- tm\_map(gether\_Restaurant\_review\_corpus, stemDocument)

Spices\_Restaurant\_review\_corpus\_stem <- tm\_map(Spices\_Restaurant\_review\_corpus, stemDocument)

Bar360\_review\_corpus\_stem <- tm\_map(Bar360\_review\_corpus, stemDocument)

Sea\_Breeze\_review\_corpus\_stem <- tm\_map(Sea\_Breeze\_review\_corpus, stemDocument)

Seafood\_reiew\_corpus\_stem <- tm\_map(Seafood\_reiew\_corpus, stemDocument)

A\_Spoonful\_of\_Sugar\_review\_corpus\_stem <- tm\_map(A\_Spoonful\_of\_Sugar\_review\_corpus, stemDocument)

Acqua\_Restaurant\_review\_corpus\_stem <- tm\_map(Acqua\_Restaurant\_review\_corpus, stemDocument)

After\_Beach\_Bar\_review\_corpus\_stem <- tm\_map(After\_Beach\_Bar\_review\_corpus, stemDocument)

Albatross\_Cafe\_review\_corpus\_stem <- tm\_map(Albatross\_Cafe\_review\_corpus, stemDocument)

Ali\_Baba\_Restaurant\_review\_corpus\_stem <- tm\_map(Ali\_Baba\_Restaurant\_review\_corpus, stemDocument)

Amalfi\_review\_corpus\_stem <- tm\_map(Amalfi\_review\_corpus, stemDocument)

Anchor\_Inn\_review\_corpus\_stem <- tm\_map(Anchor\_Inn\_review\_corpus, stemDocument)

Andaman\_Grill\_review\_corpus\_stem <- tm\_map(Andaman\_Grill\_review\_corpus, stemDocument)

Angus\_OTools\_Irish\_Pub\_review\_corpus\_stem <- tm\_map(Angus\_OTools\_Irish\_Pub\_review\_corpus, stemDocument)

Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus\_stem <- tm\_map(Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus, stemDocument)

Ann\_Restaurant\_review\_corpus\_stem <- tm\_map(Ann\_Restaurant\_review\_corpus, stemDocument)

Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus\_stem <- tm\_map(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus, stemDocument)

Arabia\_Restaurant\_review\_corpus\_stem <- tm\_map(Arabia\_Restaurant\_review\_corpus, stemDocument)

Atsumi\_Raw\_Cafe\_review\_corpus\_stem <- tm\_map(Atsumi\_Raw\_Cafe\_review\_corpus, stemDocument)

Audy\_Restaurant\_review\_corpus\_stem <- tm\_map(Audy\_Restaurant\_review\_corpus, stemDocument)

Aussie\_Pub\_Kamala\_review\_corpus\_stem <- tm\_map(Aussie\_Pub\_Kamala\_review\_corpus, stemDocument)

Autogrill\_Risto\_Bar\_Pizza\_review\_corpus\_stem <- tm\_map(Autogrill\_Risto\_Bar\_Pizza\_review\_corpus, stemDocument)

BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus\_stem <- tm\_map(BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus, stemDocument)

#import lexicons

positive\_lexicon <- read.csv("positive-lexicon.txt")

negative\_lexicon <- read.csv("negative-lexicon.txt")

#check list of lexicons

head(positive\_lexicon)

head(negative\_lexicon)

#create sentiment and wordcloud function

sentiment <- function(stem\_corpus)

{

#generate wordclouds

wordcloud(stem\_corpus,

min.freq = 3,

colors=brewer.pal(8, "Dark2"),

random.color = TRUE,

max.words = 100)

#Calculating the count of total positive and negative words in each review

#Create variables and vectors

total\_pos\_count <- 0

total\_neg\_count <- 0

pos\_count\_vector <- c()

neg\_count\_vector <- c()

#Calculate the size of the corpus

size <- length(stem\_corpus)

for(i in 1:size)

{

#All the words in current review

corpus\_words<- list(strsplit(stem\_corpus[[i]]$content, split = " "))

#positive words in current review

pos\_count <-length(intersect(unlist(corpus\_words), unlist(positive\_lexicon)))

#negative words in current review

neg\_count <- length(intersect(unlist(corpus\_words), unlist(negative\_lexicon)))

total\_pos\_count <- total\_pos\_count + pos\_count ## overall positive count

total\_neg\_count <- total\_neg\_count + neg\_count ## overall negative count

}

#Calculating overall percentage of positive and negative words of all the reviews

total\_pos\_count ## overall positive count

total\_neg\_count ## overall negative count

total\_count <- total\_pos\_count + total\_neg\_count

overall\_positive\_percentage <- (total\_pos\_count\*100)/total\_count

overall\_negative\_percentage <- (total\_neg\_count\*100)/total\_count

overall\_positive\_percentage ## overall positive percentage

#Create a dataframe with all the positive and negative reviews

df<-data.frame(Review\_Type=c("Postive","Negitive"),

Count=c(total\_pos\_count ,total\_neg\_count ))

print(df) #Print

overall\_positive\_percentage<-paste("Percentage of Positive Reviews:",

round(overall\_positive\_percentage,2),"%")

return(overall\_positive\_percentage)

}

#run sentiment functions

sentiment(Da\_Mario\_review\_corpus\_stem)

sentiment(No\_6\_Restaurant\_review\_corpus\_stem)

sentiment(Sabai\_Sabai\_review\_corpus\_stem)

sentiment(The\_Family\_Restaurant\_review\_corpus\_stem)

sentiment(The\_Pizza\_Company\_review\_corpus\_stem)

sentiment(Outdoor\_Restaurant\_review\_corpus\_stem)

sentiment(gether\_Restaurant\_review\_corpus\_stem)

sentiment(Spices\_Restaurant\_review\_corpus\_stem)

sentiment(Bar360\_review\_corpus\_stem)

sentiment(Seafood\_reiew\_corpus\_stem)

sentiment(A\_Spoonful\_of\_Sugar\_review\_corpus\_stem)

sentiment(After\_Beach\_Bar\_review\_corpus\_stem)

sentiment(Acqua\_Restaurant\_review\_corpus\_stem)

sentiment(Albatross\_Cafe\_review\_corpus\_stem)

sentiment(Ali\_Baba\_Restaurant\_review\_corpus\_stem)

sentiment(Amalfi\_review\_corpus\_stem)

sentiment(Anchor\_Inn\_review\_corpus\_stem)

sentiment(Andaman\_Grill\_review\_corpus\_stem)

sentiment(Angus\_OTools\_Irish\_Pub\_review\_corpus\_stem)

sentiment(Ann\_Restaurant\_review\_corpus\_stem)

sentiment(Anns\_Kitchen\_Bar\_and\_Grill\_review\_corpus\_stem)

sentiment(Ao\_Chalong\_Yacht\_Club\_Restaurant\_review\_corpus\_stem)

sentiment(Arabia\_Restaurant\_review\_corpus\_stem)

sentiment(Atsumi\_Raw\_Cafe\_review\_corpus\_stem)

sentiment(Audy\_Restaurant\_review\_corpus\_stem)

sentiment(Aussie\_Pub\_Kamala\_review\_corpus\_stem)

sentiment(Autogrill\_Risto\_Bar\_Pizza\_review\_corpus\_stem)

sentiment(BHive\_Gallery\_Bar\_and\_Restaurant\_review\_corpus\_stem)