# Topic: Association Rules

**Instructions**

Please share your answers filled inline in the word document. Submit Python code and R code files wherever applicable.

Please ensure you update all the details:

**Name:Anandakrishnan k v**

**Batch Id:**  19042021

**Topic: - Association Rules.**

**Q1)**

1. **Problem Statement: -**

Kitabi Duniya , a famous book store in India, which was established before Independence, the growth of the company was incremental year by year, but due to online selling of books and wide spread Internet access its annual growth started to collapse, seeing sharp downfalls, you as a Data Scientist help this heritage book store gain its popularity back and increase footfall of customers and provide ways the business can improve exponentially, apply Association Rule Algorithm, explain the rules, and visualize the graphs for clear understanding of solution.

* 1. Objective

help heritage book store gain its popularity back and increase footfall of customers and provide ways the business can improve exponentially

* 1. Constraints (if any)

Maximize: the profit

1.  Work on each feature of the dataset to create a data dictionary as displayed in the below image**:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name of feature | Description | Type | Relevance |
| ChildBks | Book category | binary | relevant |
| |  |  |  |  | | --- | --- | --- | --- | | YouthBks |  |  |  | | Book category | binary | relevant |
| |  | | --- | | CookBks | | Book category | binary | relevant |
| |  |  | | --- | --- | | DoItYBks |  | | Book category | binary | relevant |
| RefBks | Book category | binary | relevant |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ArtBks |  |  |  |  |  | | Book category | binary | relevant |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | GeogBks |  |  |  |  | | Book category | binary | relevant |
| |  |  |  |  | | --- | --- | --- | --- | | ItalCook |  |  |  | | Book category | binary | relevant |
| |  |  |  | | --- | --- | --- | | ItalAtlas |  |  | | Book category | binary | relevant |
| |  |  | | --- | --- | | ItalArt |  | | Book category | binary | relevant |
| Florence | Book category | binary | relevant |

**Using R and Python codes perform:**

**3.Data Pre-processing**

* 1. **Data Cleaning, Feature Engineering, etc.**

**R code:-**

######### book #########

# Load the dataset

library(readr)

input <- read\_csv(file.choose())

mydata <- input

## 3.1) DATA CLEANING & PREPROCESSING

# missing data checking

sum(is.na(mydata)) ## no null values

## outlier treatment

# since data is binary standardized form wouldn't gonna have no outliers

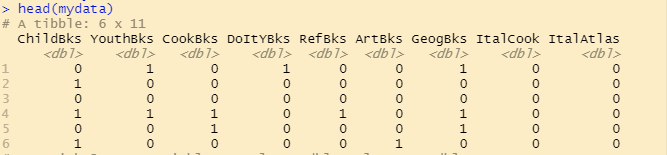
# install arules pakages for building association rules

install.packages("arules")

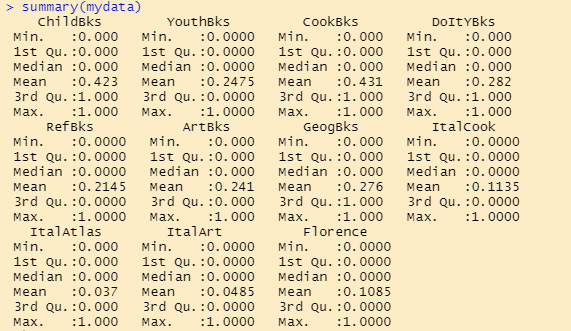
library("arules") # Used for building association rules i.e. apriori algorithm

# summary

Head(mydata)



summary(mydata)



# converting to matrix format for rules formation

qq <- as.matrix(mydata)

# the matrix data set is converting to transaction form by eliminating '0' showing no purchasing of perticular product

qq <- as(qq,"transactions")

# making rules using apriori algorithm

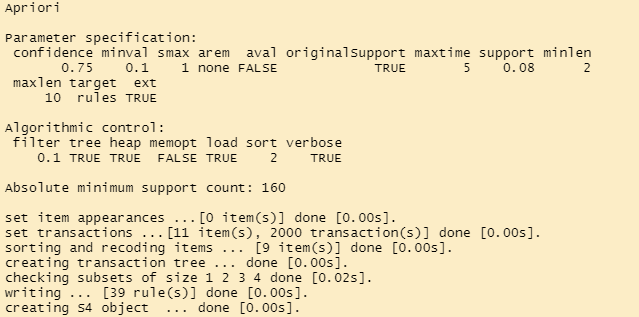
# Keep changing support and confidence values to obtain different rules

**4.Model Building**

**4.1 Application of Apriori Algorithm.**

# 4.1) APPLICATION OF APRIORI ALGORITHM

arules <- apriori(qq, parameter = list(support = 0.08, confidence = 0.75, minlen = 2))



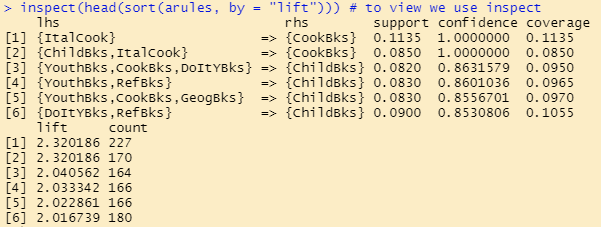
arules # 39 rules

**4.Model Building**

**4.2 Build most frequent item sets and plot the rules.**

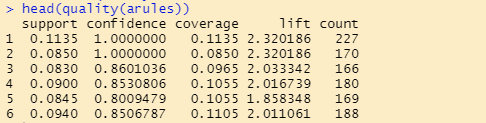
# 4.2) BUILD MOST FREQUENT ITEM SET AND PLOT THE RULE

inspect(head(sort(arules, by = "lift"))) # to view we use inspect



# Overal quality

head(quality(arules))

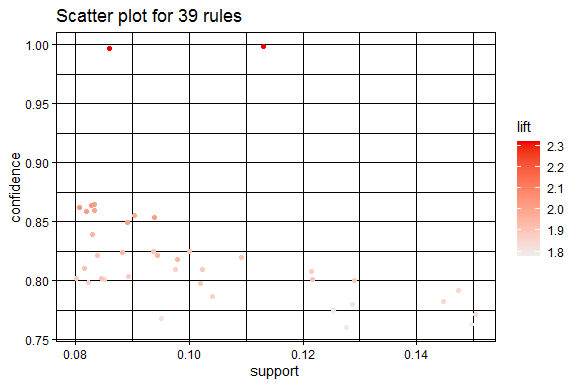


# install.packages("arueslViz")

library("arulesViz") # for visualizing rules

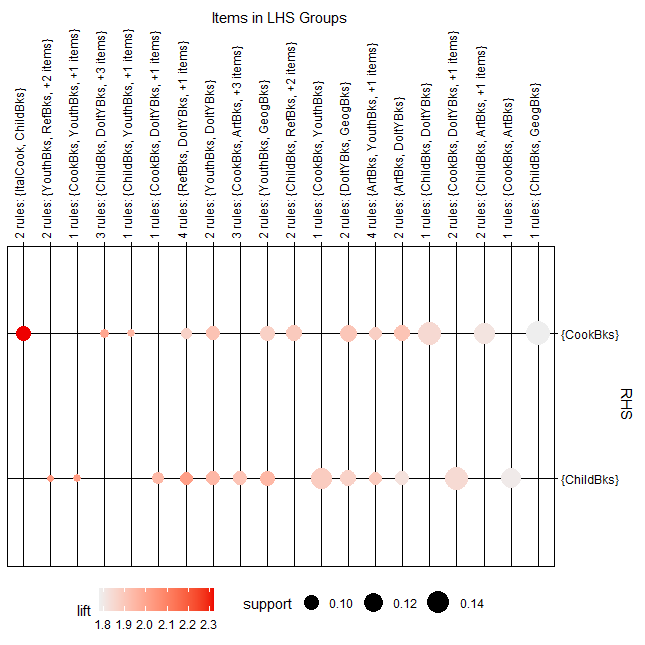
# Different Ways of Visualizing Rules

plot(arules)

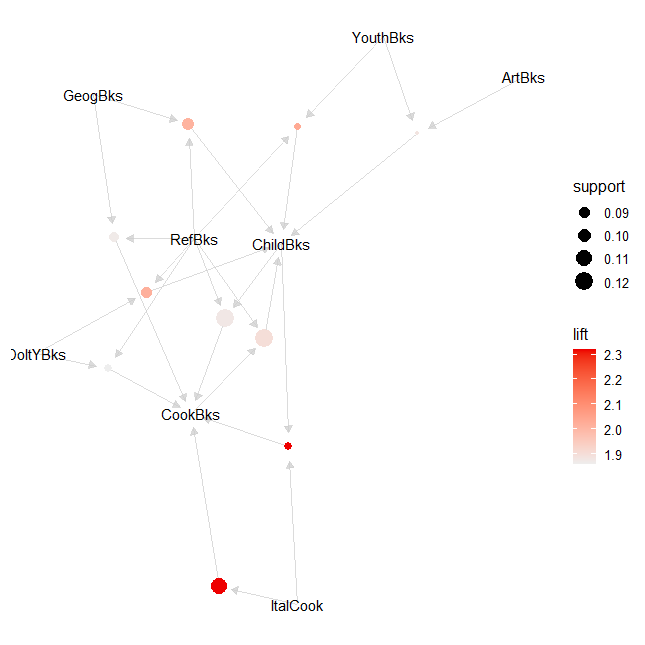


windows()

plot(arules, method = "grouped")



plot(arules[1:10], method = "graph") # for good visualization try plotting only few rules



write(arules, file = "a\_rules.csv", sep = ",")

getwd()

**Python code:-**

########### book ##############

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats.mstats import winsorize

# load the data set

book = pd.DataFrame(bookcsv)

book.shape

book1 = book.copy(deep=True)

###### Null value Treatment ########

book1.isna().sum() ## no null values

###### Summary of the data set ####

book1.columns

book1.describe()

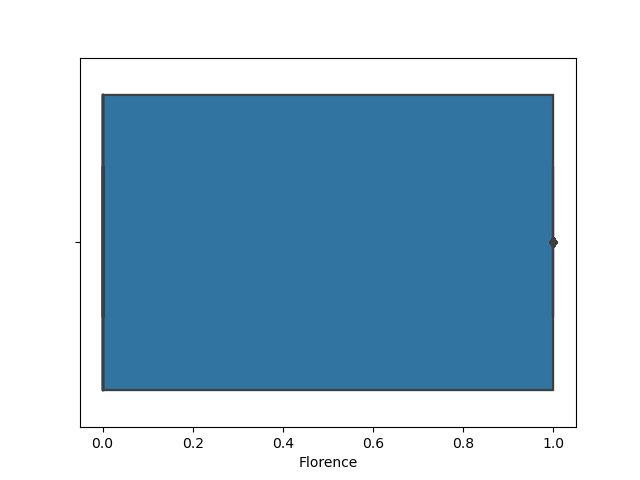
####### Outlier Treatment ########

# Boxplots

for i in book1.columns:

sns.boxplot(book1[i].dropna())

plt.show() ## no outliers because data set is in binary normalized format; & combined boxplot not showig any outliers



####### checking data is normalisedformat or not #######

for i in book1.columns:

for x in book1[i]:

if(x!=0 | x!=1):

standared = "false"

row=x

break

else:

standared = "true"

print(standared) ## data is already in normalized form since every datas have either 1 or 0 values, so standarded = true

# since data is in normalised format directly going for further analysis

############ Zero variance analysis #############

book1.shape

## importing ###

from sklearn.feature\_selection import VarianceThreshold

# Feature selector that removes all low-variance features that meets the variance threshold limit

var\_thres = VarianceThreshold(threshold=0) # Threshold is subjective.

var\_thres.fit(book1) ### fit the var\_thres to data set book1

# Generally we remove the columns with zero variance, but i took thresold value 0 (Near Zero Variance)

var\_thres.get\_support() ### it giving an array out, where zero variant column treat as False value. we already fit var\_thres to book1. so it gives corresponding information on book1

book1.columns[var\_thres.get\_support()] ## non-zero variant column names

constant\_columns = [column for column in book1.columns if column not in book1.columns[var\_thres.get\_support()]]

print(len(constant\_columns)) ### number of zero variant variables

# since number of zero variant columns = 0 ==> none of the variables or column having zero variant property ; so directy going for further analysis without doing any zero variant treatment

######################## forming association rules ######################

# Implementing Apriori algorithm from mlxtend

# conda install mlxtend

# or

# pip install mlxtend

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

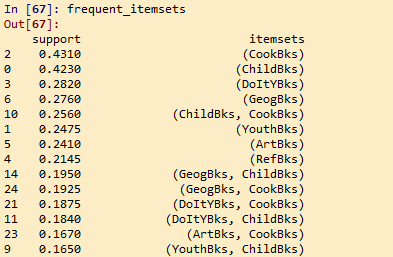
# getting support of each combination using apriori

frequent\_itemsets = apriori(book1, min\_support = 0.08, max\_len = 3, use\_colnames = True)

# Most Frequent item sets based on support

frequent\_itemsets.sort\_values('support', ascending = False, inplace = True)

frequent\_itemsets



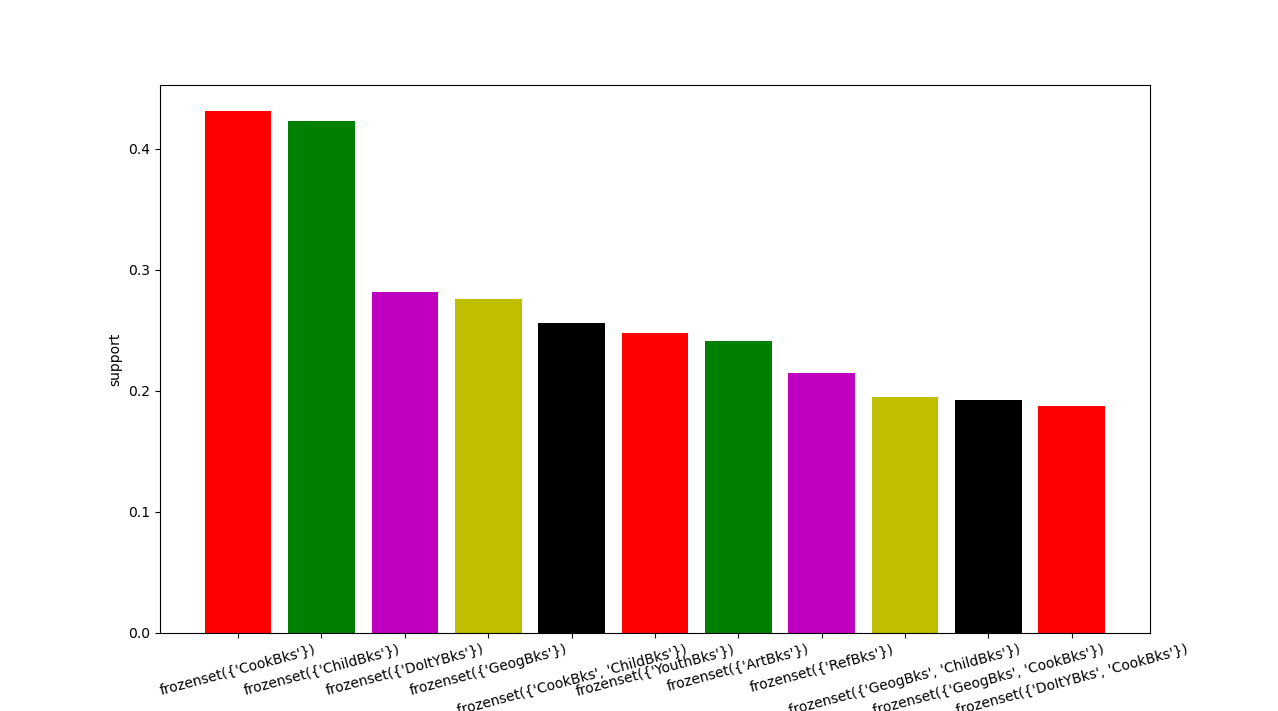
plt.bar(x = list(range(0, 11)), height = frequent\_itemsets.support[0:11], color ='rgmyk')

plt.xticks(list(range(0, 11)), frequent\_itemsets.itemsets[0:11], rotation=20)

plt.xlabel('item-sets')

plt.ylabel('support')

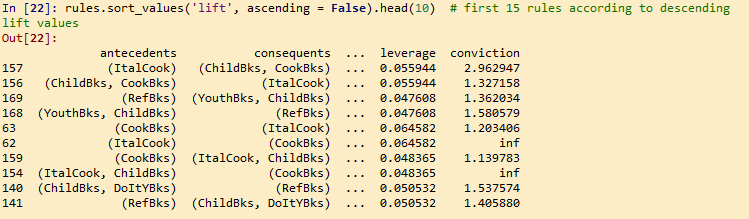
plt.show()



rules = association\_rules(frequent\_itemsets, metric = "lift", min\_threshold = 1)

rules.head(20) # first 15 rules

rules.sort\_values('lift', ascending = False).head(10) # first 15 rules according to descending lift values



**Summary:-**

* **The book having highest supports (most frequently sell) are cookBks, childBks,DoityBks and GeogBks in order**
* **The combination of Books having lift ratio with considerable support are “ItalCook , childBks & CookBks”, “RefBks, YouthBks, ChildBks”**

**6. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.**

**Ans:-**

* R and Python code results the best books having high support and highly lifted combinations of books in the market in previous days.
* There are 2 ways we can improves the sales of the books in th store. Either by make availability of popular books (books having high demand(support))& providing direct offer for them or by providing better offer & better nearby placement for combination of books having high lift ratio with considerable support value.
* CookBks, childBks,DoityBks and GeogBks these are the books having Highest demand or suport in the market. So ensure the availability of these books in store at any time
* Its showing “ItalCook , childBks & CookBks” & “RefBks, YouthBks, ChildBks” are highly lifted combinations of Books. The book store can improve their sails by either providing better offers for this products Or arrange those combination of Books in nearby shelves one after the other(like ItalBks followed by ChildBks & cookBks)

**Q2)**

**Problem Statement:**

**The Departmental Store, has gathered the data of the products it sells on a**

**Daily basis. Using Association Rules concepts, provide the insights on the rules and the plots.**

* 1. **Objective:-**

Using Association Rules concepts, provide the insights on the rules and the plots

* 1. **Constraints (if any)**

Maximize: sales

Maximize: profit

**R and Python codes perform:**

R code:-

########## groceries #########

install.packages("arules")

data()

library("arules") # Used for building association rules i.e. apriori algorithm

data("Groceries") # loading the Groceries Data

?Groceries

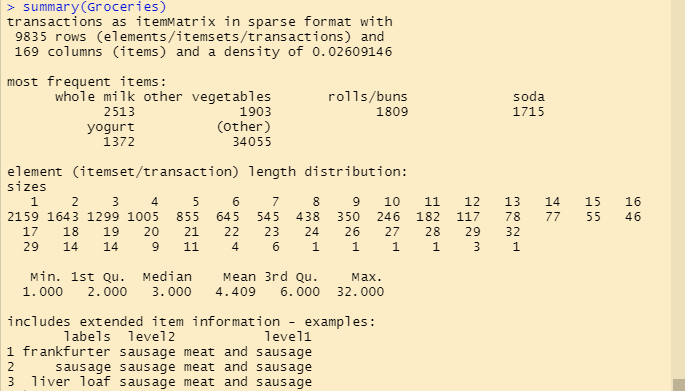
inspect(Groceries[1:5]) # showing only top 10 transactions

class(Groceries) # Groceries is in transactions format

# summary

head(Groceries)

summary(Groceries)



# making rules using apriori algorithm

# Keep changing support and confidence values to obtain different rules

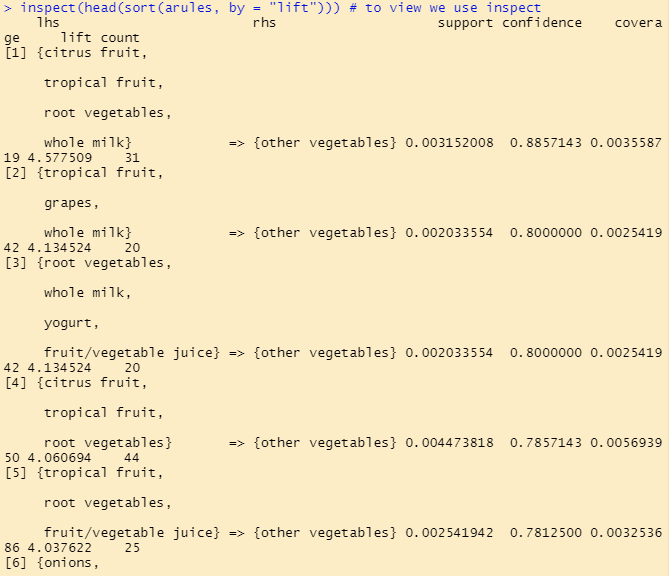
# Building rules using apriori algorithm

arules <- apriori(Groceries, parameter = list(support = 0.002, confidence = 0.75, minlen = 2))

arules # 39 rules

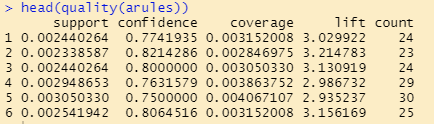
# Viewing rules based on lift value

inspect(head(sort(arules, by = "lift"))) # to view we use inspect



# Overal quality

head(quality(arules))

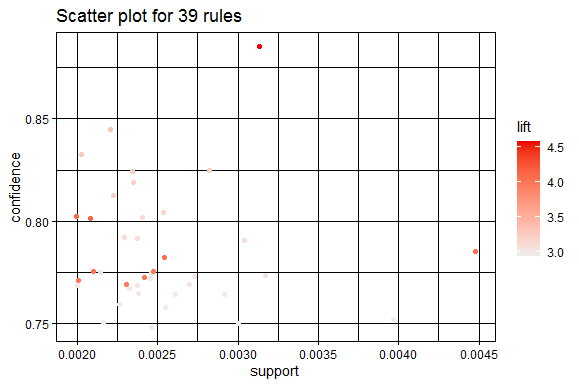


# install.packages("arueslViz")

library("arulesViz") # for visualizing rules

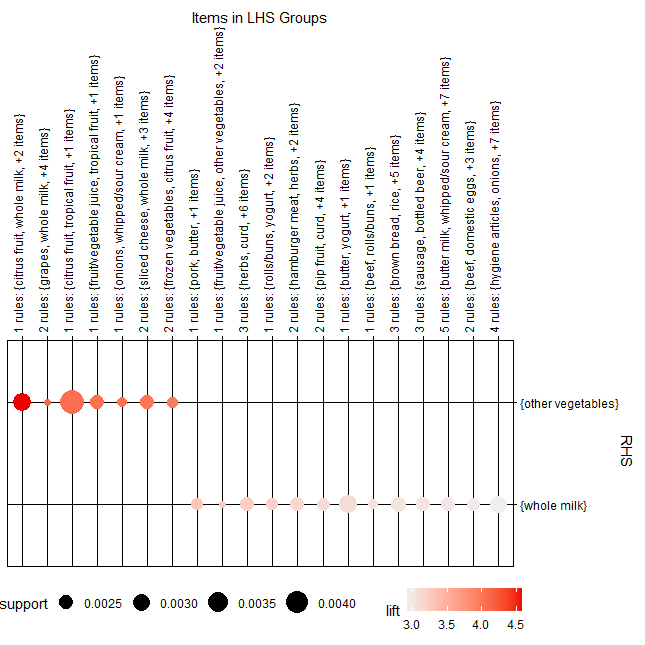
# Different Ways of Visualizing Rules

plot(arules)

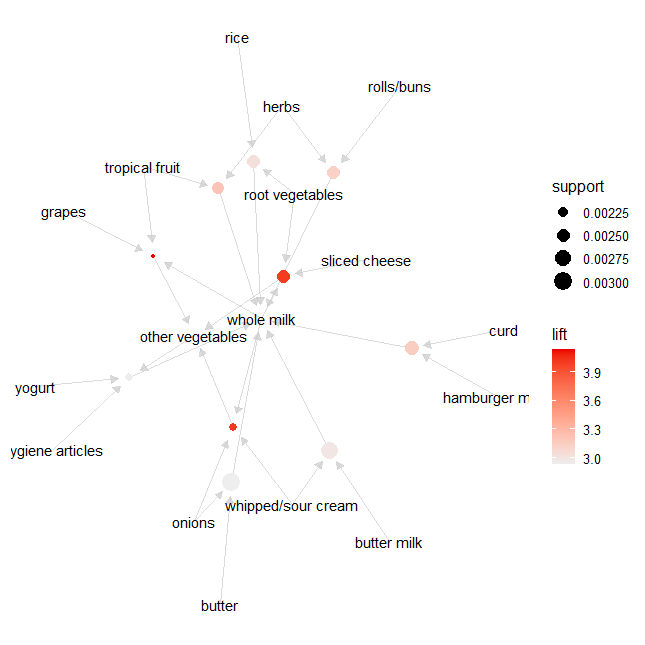


windows()

plot(arules, method = "grouped")



plot(arules[1:10], method = "graph") # for good visualization try plotting only few rules



write(arules, file = "a\_rules.csv", sep = ",")

getwd()

**Python code:-**

################# groceries ##################

## importing packages

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats.mstats import winsorize

####### creating groceries dataframe ######

# Implementing Apriori algorithm from mlxtend

conda install mlxtend

# or

# pip install mlxtend

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

groceries = []

with open("G:\\association rule\\groceries.csv") as f:

groceries = f.read()

# splitting the data into separate transactions using separator as "\n"

groceries = groceries.split("\n")

groceries\_list = []

for i in groceries:

groceries\_list.append(i.split(","))

all\_groceries\_list = [i for item in groceries\_list for i in item]

from collections import Counter # ,OrderedDict

item\_frequencies = Counter(all\_groceries\_list)

# after sorting

item\_frequencies = sorted(item\_frequencies.items(), key = lambda x:x[1])

# Storing frequencies and items in separate variables

frequencies = list(reversed([i[1] for i in item\_frequencies]))

items = list(reversed([i[0] for i in item\_frequencies]))

# barplot of top 10

import matplotlib.pyplot as plt

plt.bar(height = frequencies[0:11], x = list(range(0, 11)), color = 'rgbkymc')

plt.xticks(list(range(0, 11), ), items[0:11])

plt.xlabel("items")

plt.ylabel("Count")

plt.show()

# Creating Data Frame for the transactions data

groceries\_series = pd.DataFrame(pd.Series(groceries\_list))

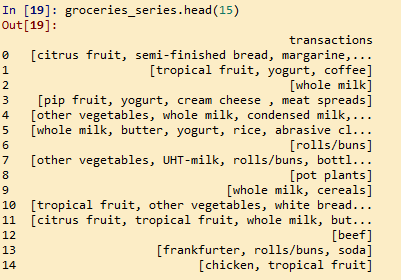
###### Null value Treatment ########

groceries\_series.isna().sum() ## no null values

groceries\_series = groceries\_series.iloc[:200, :] # selecting only first 200 transactions

groceries\_series.columns = ["transactions"]

groceries\_series.head(15)



# creating a dummy columns for the each item in each transactions ... Using column names as item name

X1 = groceries\_series['transactions'].str.join(sep = '\*').str.get\_dummies(sep = '\*')

############### Data cleaning and Preprocessing begins ###############

# load the data set

X1.shape

X = X1.copy(deep=True)

###### Null value Treatment ########

X.isna().sum() ## no null values

##### summary of the data set ######

X.columns

X.describe()

####### Outlier Treatment ########

# Boxplots

for i in X.columns:

sns.boxplot(X[i].dropna())

plt.show() ## no outliers because data set is in binary normalized format; & combined boxplot not showig any outliers

####### standardization//normalization ####

#data set is already in normalized format so need to do standardization/normalization here

############ Zero variance analysis #############

X.shape

## importing ###

from sklearn.feature\_selection import VarianceThreshold

# Feature selector that removes all low-variance features that meets the variance threshold limit

var\_thres = VarianceThreshold(threshold=0) # Threshold is subjective.

var\_thres.fit(X) ### fit the var\_thres to data set X

# Generally we remove the columns with zero variance, but i took thresold value 0 (Near Zero Variance)

var\_thres.get\_support() ### it giving an array out, where zero variant column treat as False value. we already fit var\_thres to X. so it gives corresponding information on X

X.columns[var\_thres.get\_support()] ## non-zero variant column names

constant\_columns = [column for column in X.columns if column not in X.columns[var\_thres.get\_support()]]

print(len(constant\_columns)) ### number of zero variant variables

# since number of zero variant columns = 0 ==> none of the variables or column having zero variant property ; so directy going for further analysis without doing any zero variant treatment

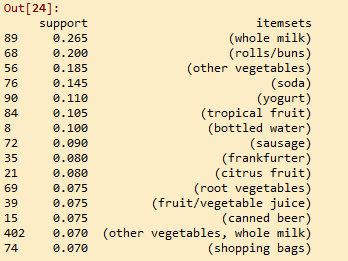
######################## forming association rules ######################

frequent\_itemsets = apriori(X, min\_support = 0.0075, max\_len = 3, use\_colnames = True)

# Most Frequent item sets based on support

frequent\_itemsets.sort\_values('support', ascending = False, inplace = True)

frequent\_itemsets.head(15)



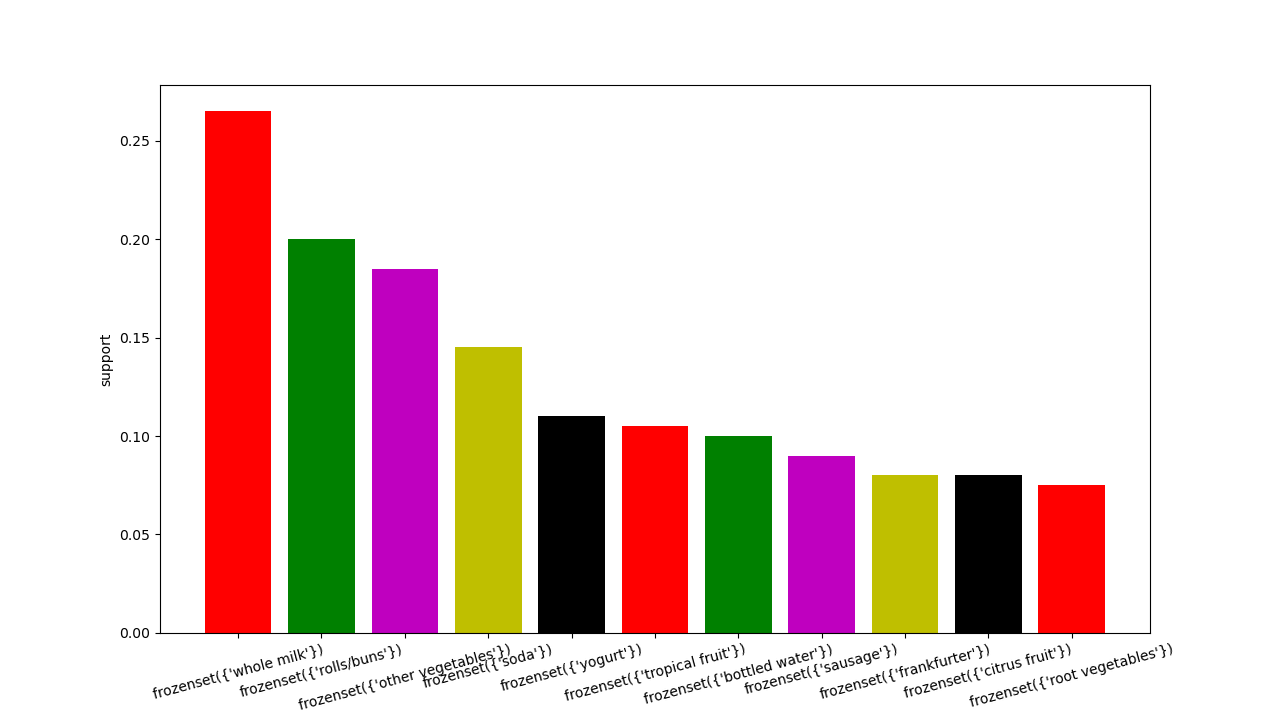
plt.bar(x = list(range(0, 11)), height = frequent\_itemsets.support[0:11], color ='rgmyk')

plt.xticks(list(range(0, 11)), frequent\_itemsets.itemsets[0:11], rotation=20)

plt.xlabel('item-sets')

plt.ylabel('support')

plt.show()



rules = association\_rules(frequent\_itemsets, metric = "lift", min\_threshold = 1)

rules.head(20)

rules.sort\_values('lift', ascending = False).head(10)

rules[["antecedents","consequents","support","lift"]].head(10)

**Summary:-**

**( here R code & Python code have different results since considering only first 200 transaction in ython due to loading issues while doing dummy variable creation for products. So R code showing more accurate product. So we do considering that only)**

* **The products having highest supports (most frequently sell) are whole milk, citrus fruit, other vegetables, root vegetables, tropical fruits, grapes, yogurt, rolls/buns etc**
* **The combination of products having high lift ratios with considerable support are “citrus fruit, tropical fruits, root vegetables, whole milk & other vegetables”, “tropical fruits,grapes, whole milk & other vegetables”**

**6. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.**

**Ans:-**

* R and Python code results the best products having high support and highly lifted combinations of products from the store in previous days.
* There are 2 ways we can improves the sales of the products in the store. Either by ensuring enough availabilty and better offer for popular products those having high demand(support) in the market or by providing better offer and placement for combination of products having high lift ratio & considerable support.
* The products & product combinaton having high support and lift ratios are already shown above in summary

**Q3)**

**Problem Statement:**

A film distribution company wants to target audience based on their likes and dislikes, you as a Chief Data Scientist Analyze the data and come up with different rules of movie list so that the business objective is achieved.

* 1. **Objective:-**

based on audience likes and dislikes come up with different rules of movie list so that the business objective is achieved.

* 1. **Constraints (if any)**

Maximimize: customer satisfaction

Maximize: the customers number

Maximize: profit

**R and Python codes perform:**

**R code:-**

######### my movies #########

# Load the dataset

library(readr)

input <- read\_csv(file.choose())

mydata1 <- input

# eliminate non\_numeric & less informative datas

mydata <- mydata1[,c(6:15)]

## DATA CLEANING AND EDA BEGINS

# missing data checking

sum(is.na(mydata)) ## no null values

## outlier treatment

# since data is binary standardized form wouldn't gonna have no outliers

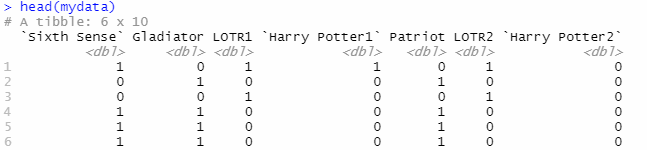
# install arules pakages for building association rules

# install.packages("arules")

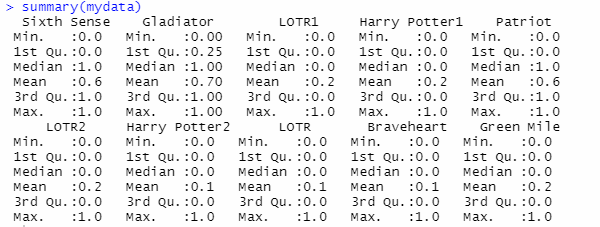
library("arules") # Used for building association rules i.e. apriori algorithm

# summary

head(mydata)



summary(mydata)



# converting to matrix format for rules formation

qq <- as.matrix(mydata)

# the matrix data set is converting to transaction form by eliminating '0' showing no purchasing of perticular product

qq <- as(qq,"transactions")

# making rules using apriori algorithm

# Keep changing support and confidence values to obtain different rules

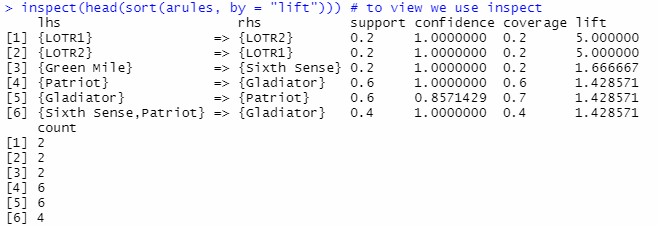
# Building rules using apriori algorithm

arules <- apriori(qq, parameter = list(support = 0.11, confidence = .6, minlen = 2))

arules # 12 rules

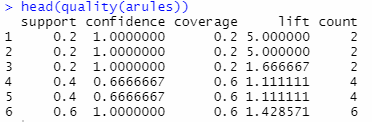
# Viewing rules based on lift value

inspect(head(sort(arules, by = "lift"))) # to view we use inspect



# Overal quality

head(quality(arules))

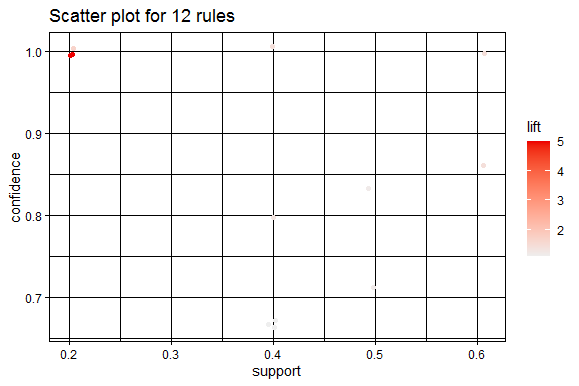


# install.packages("arueslViz")

library("arulesViz") # for visualizing rules

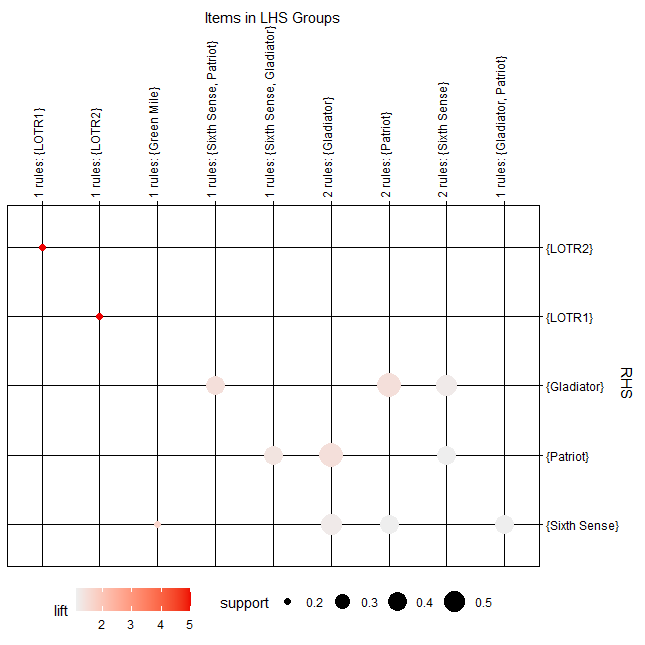
# Different Ways of Visualizing Rules

plot(arules)

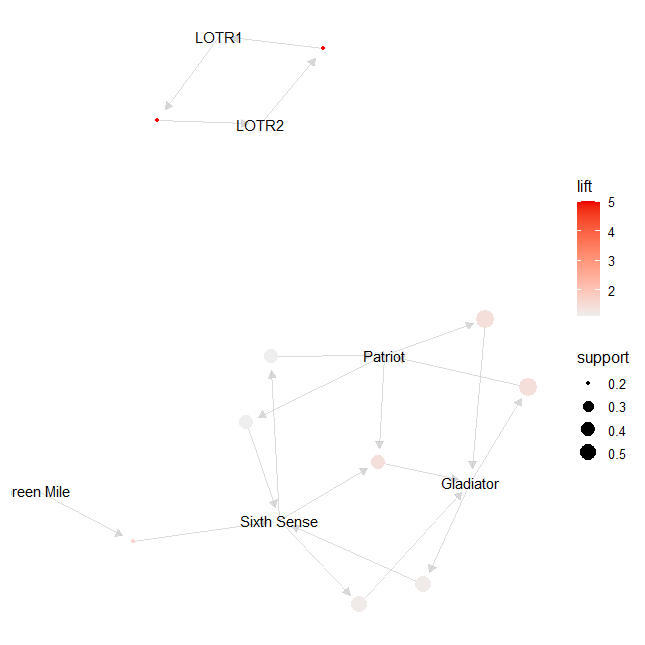


windows()

plot(arules, method = "grouped")



plot(arules[1:10], method = "graph") # for good visualization try plotting only few rules



write(arules, file = "a\_rules.csv", sep = ",")

getwd()

**Python code:-**

########### my movies ##############

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats.mstats import winsorize

# load the data set

movie = pd.DataFrame(my\_moviescsv)

movie.shape

movie1 = movie.iloc[:,5:] ## numeric datas only

###### Null value Treatment ########

movie1.isna().sum() ## no null values

###### Summary of the data set ####

movie1.columns

movie1.describe()

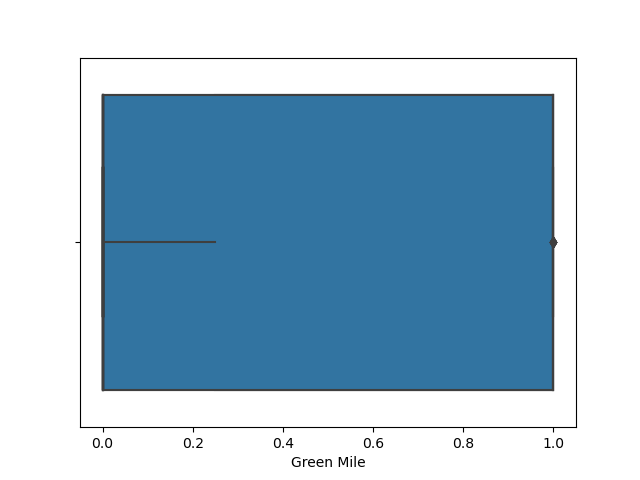
####### Outlier Treatment ########

# Boxplots

for i in movie1.columns:

sns.boxplot(movie1[i].dropna())

plt.show() ## no outliers because data set is in binary normalized format; & combined boxplot not showig any outliers



####### checking data is normalised format or not #######

for i in movie1.columns:

for x in movie1[i]:

if(x!=0 | x!=1):

standared = "false"

row=x

break

else:

standared = "true"

print(standared) ## data is already in normalized form since every datas have either 1 or 0 values, so standarded = true

# since data is in normalised format directly going for further analysis

############ Zero variance analysis #############

movie1.shape

## importing ###

from sklearn.feature\_selection import VarianceThreshold

# Feature selector that removes all low-variance features that meets the variance threshold limit

var\_thres = VarianceThreshold(threshold=0) # Threshold is subjective.

var\_thres.fit(movie1) ### fit the var\_thres to data set movie1

# Generally we remove the columns with zero variance, but i took thresold value 0 (Near Zero Variance)

var\_thres.get\_support() ### it giving an array out, where zero variant column treat as False value. we already fit var\_thres to movie1. so it gives corresponding information on movie1

movie1.columns[var\_thres.get\_support()] ## non-zero variant column names

constant\_columns = [column for column in movie1.columns if column not in movie1.columns[var\_thres.get\_support()]]

print(len(constant\_columns)) ### number of zero variant variables

# since number of zero variant columns = 0 ==> none of the variables or column having zero variant property ; so directy going for further analysis without doing any zero variant treatment

######################## forming association rules ######################

# Implementing Apriori algorithm from mlxtend

# conda install mlxtend

# or

# pip install mlxtend

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

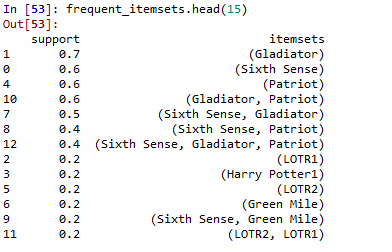
# getting support of each combination using apriori

frequent\_itemsets = apriori(movie1, min\_support = 0.11, max\_len = 3, use\_colnames = True)

# Most Frequent item sets based on support

frequent\_itemsets.sort\_values('support', ascending = False, inplace = True)

frequent\_itemsets.head(15)



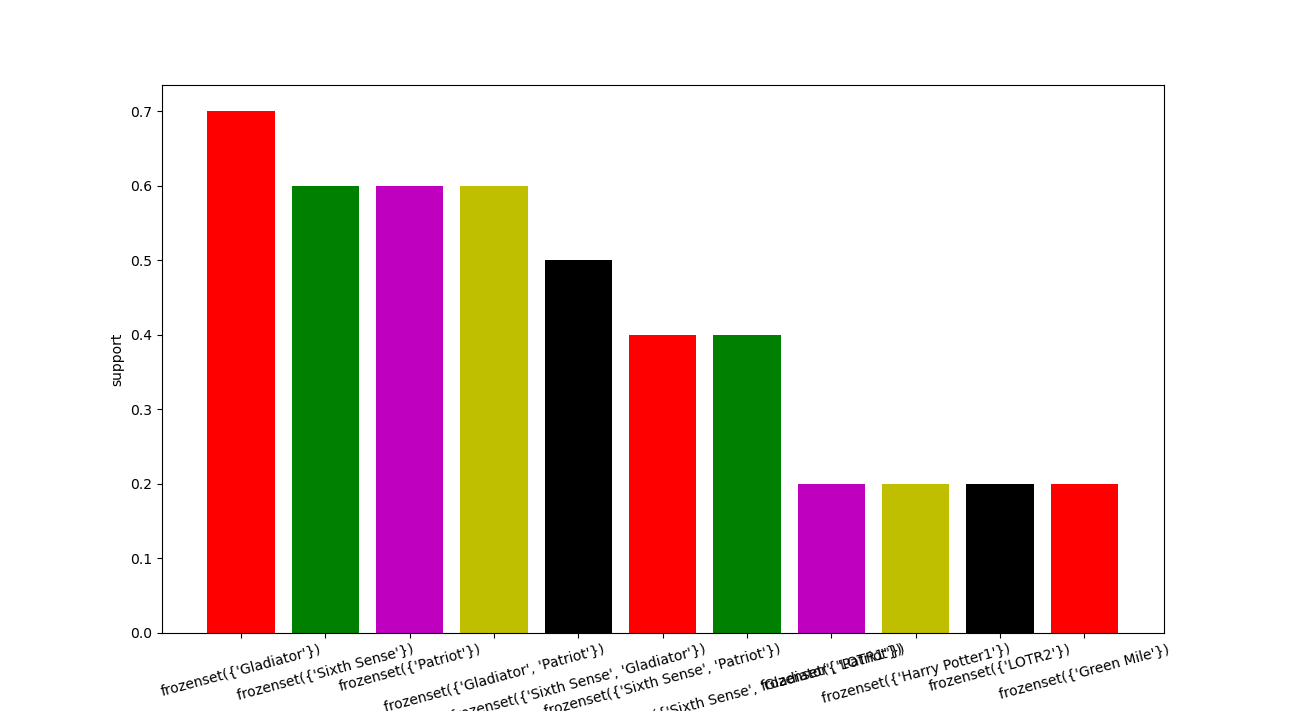
plt.bar(x = list(range(0, 11)), height = frequent\_itemsets.support[0:11], color ='rgmyk')

plt.xticks(list(range(0, 11)), frequent\_itemsets.itemsets[0:11], rotation=15)

plt.xlabel('item-sets')

plt.ylabel('support')

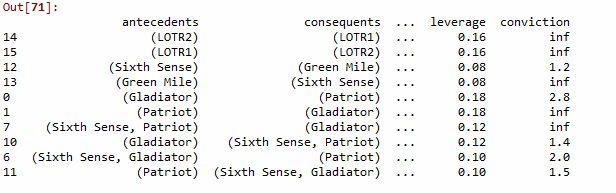
plt.show()



rules = association\_rules(frequent\_itemsets, metric = "lift", min\_threshold = 1)

rules.head(20)

rules.sort\_values('lift', ascending = False).head(10)



**Summary:-**

* **The films having highest supports (most frequently watched) are Gladiator, Sixth sense, Patriot, LOTR 1, Harry potter, LOTR 2, Green mile etc in order.**
* **The combination of products having high lift ratios with considerable support are “LOTR 1 & LOTR 2”, “Sixth sense & Green mile”, “Gladiator & patriot”, “Sixth sense , Gladiator & patriot”**

**6. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.**

**Ans:-**

* R and Python code results the best films having high support and combinations of films having high lift ratios in previous days.
* The films Gladiator & sixth sense having highest support. So films having similar gentre will win the market
* Highest lift ratio with considerable support is for LOTR 1 & LOTR 2 film combination. This is series type movie, indicate that those people watch series type films first part would watch its futher parts too. So if there have any series type films if its first part have considerable support (means the first part of the film need to have better theme & content that will catch the audience. it can understand from film preview through a film reviewer or any other person have enough knowledge about the film) claim its 2nd part distribution rights also at the first place itself at cheap rate.

**Q4)**

1. **Problem Statement: -**

**Problem Statement: -**

**A Mobile Phone manufacturing company wants to launch its three brand new phone into the market, but before going with its traditional marketing approach this time it want to analyze the data of its previous model sales in different regions and you have been hired as an Data Scientist to help them out, use the Association rules concept and provide your insights to the company’s marketing team to improve its sales.**

* 1. Objective

Analyze the data of its previous model sales in different regions and use the Association rules concept to improve its sales.

* 1. Constraints (if any)

Maximize: the sales

Maximixe: customer satisfaction

Maximize: profit

**R and Python codes perform:**

**R code:-**

######### my phone #########

# Load the dataset

library(readr)

input <- read\_csv(file.choose())

mydata1 <- input

# eliminate non\_numeric & less informative datas

mydata <- mydata1[,c(4:9)]

## DATA CLEANING AND EDA BEGINS

# missing data checking

sum(is.na(mydata)) ## no null values

## outlier treatment

# since data is binary standardized form wouldn't gonna have no outliers

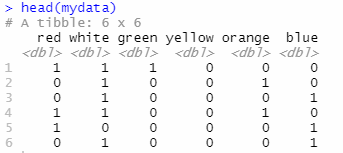
# install arules pakages for building association rules

# install.packages("arules")

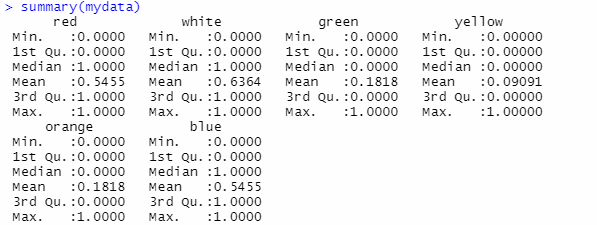
library("arules") # Used for building association rules i.e. apriori algorithm

# summary

head(mydata)



summary(mydata)



# converting to matrix format for rules formation

qq <- as.matrix(mydata)

# the matrix data set is converting to transaction form by eliminating '0' showing no purchasing of perticular product

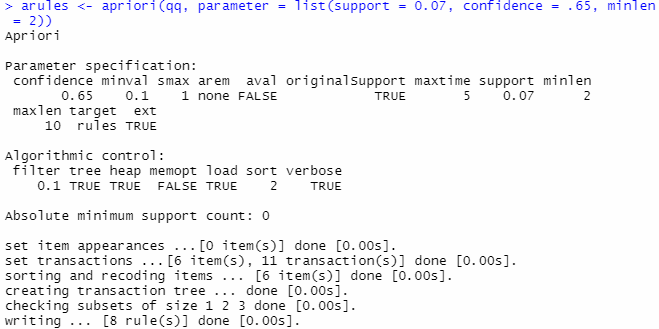
qq <- as(qq,"transactions")

# making rules using apriori algorithm

# Keep changing support and confidence values to obtain different rules

# Building rules using apriori algorithm

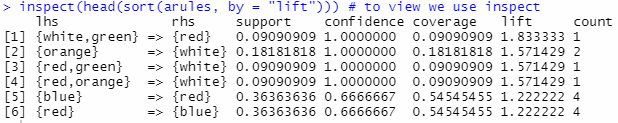
arules <- apriori(qq, parameter = list(support = 0.07, confidence = .65, minlen = 2))



arules # 8 rules

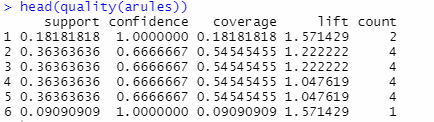
# Viewing rules based on lift value

inspect(head(sort(arules, by = "lift"))) # to view we use inspect



# Overal quality

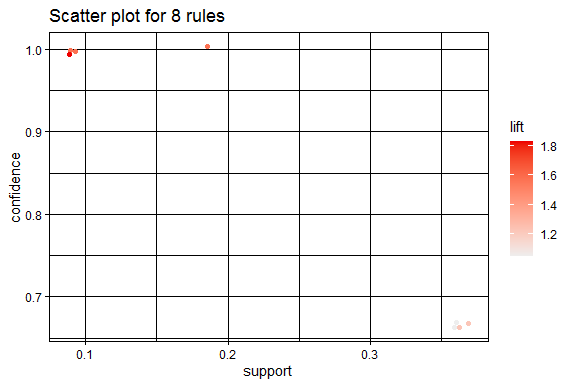
head(quality(arules))



# install.packages("arueslViz")

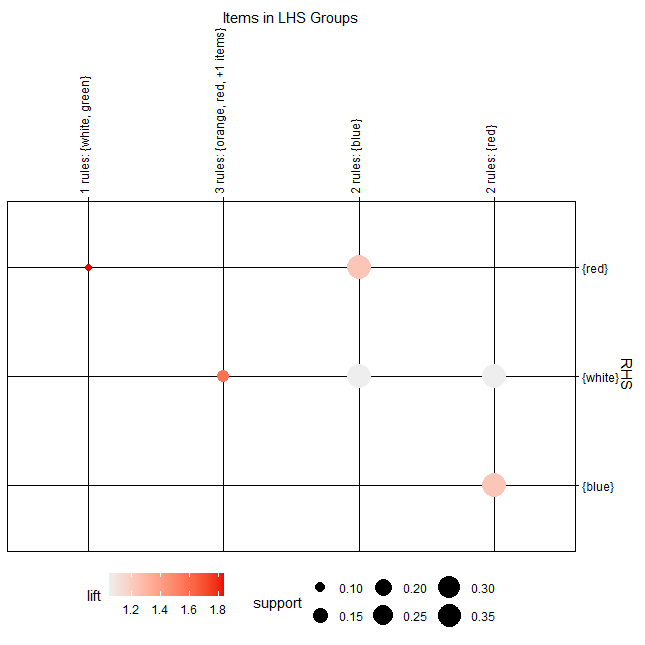
library("arulesViz") # for visualizing rules

# Different Ways of Visualizing Rules

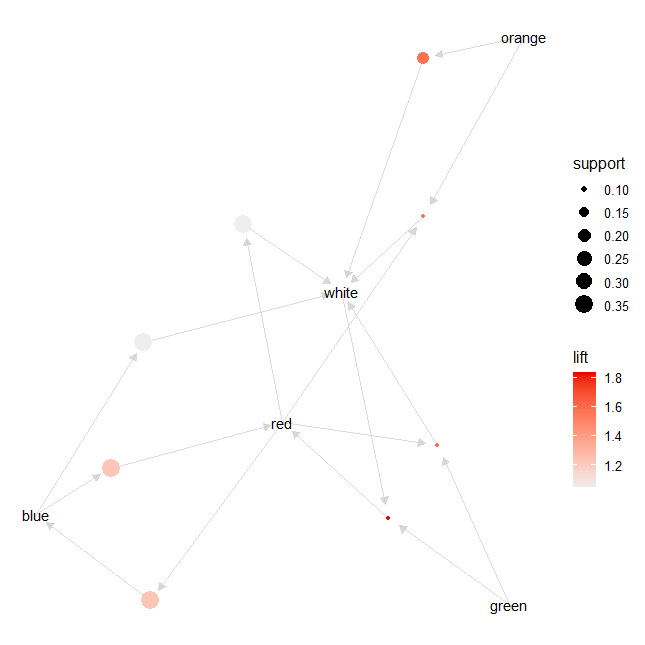


windows()

plot(arules, method = "grouped")



plot(arules[1:10], method = "graph") # for good visualization try plotting only few rules



write(arules, file = "a\_rules.csv", sep = ",")

getwd()

**Python Code:-**

############### myphonedata.csv ##############

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from scipy.stats.mstats import winsorize

# load the data set

df = pd.DataFrame(myphonedatacsv)

df.shape

df1 = df.iloc[:,3:] ## numeric datas only

###### Null value Treatment ########

df1.isna().sum() ## no null values

###### Summary of the data set ####

df1.columns

df1.describe()

####### Outlier Treatment ########

# Boxplots

for i in df1.columns:

sns.boxplot(df1[i].dropna())

plt.show() ## no outliers because data set is in binary normalized format; & combined boxplot not showig any outliers

####### checking data is normalised format or not #######

for i in df1.columns:

for x in df1[i]:

if(x!=0 | x!=1):

standared = "false"

row=x

break

else:

standared = "true"

print(standared) ## data is already in normalized form since every datas have either 1 or 0 values, so standarded = true

# since data is in normalised format directly going for further analysis

############ Zero variance analysis #############

df1.shape

## importing ###

from sklearn.feature\_selection import VarianceThreshold

# Feature selector that removes all low-variance features that meets the variance threshold limit

var\_thres = VarianceThreshold(threshold=0) # Threshold is subjective.

var\_thres.fit(df1) ### fit the var\_thres to data set df1

# Generally we remove the columns with zero variance, but i took thresold value 0 (Near Zero Variance)

var\_thres.get\_support() ### it giving an array out, where zero variant column treat as False value. we already fit var\_thres to df1. so it gives corresponding information on df1

df1.columns[var\_thres.get\_support()] ## non-zero variant column names

constant\_columns = [column for column in df1.columns if column not in df1.columns[var\_thres.get\_support()]]

print(len(constant\_columns)) ### number of zero variant variables

# since number of zero variant columns = 0 ==> none of the variables or column having zero variant property ; so directy going for further analysis without doing any zero variant treatment

######################## forming association rules ######################

# Implementing Apriori algorithm from mlxtend

# conda install mlxtend

# or

# pip install mlxtend

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

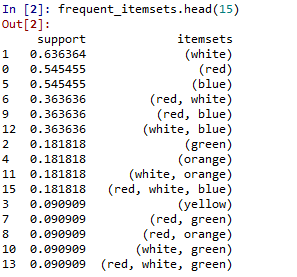
# getting support of each combination using apriori

frequent\_itemsets = apriori(df1, min\_support = 0.075, max\_len = 3, use\_colnames = True)

# Most Frequent item sets based on support

frequent\_itemsets.sort\_values('support', ascending = False, inplace = True)

frequent\_itemsets.head(15)

****

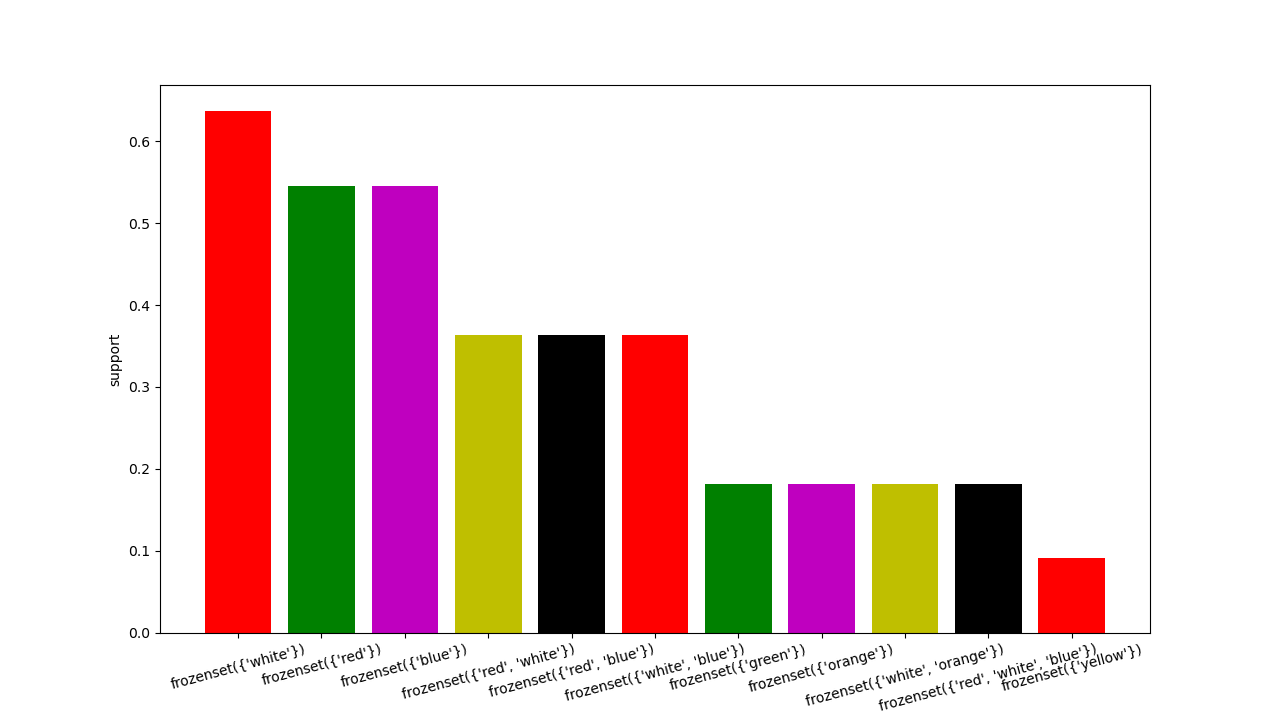
plt.bar(x = list(range(0, 11)), height = frequent\_itemsets.support[0:11], color ='rgmyk')

plt.xticks(list(range(0, 11)), frequent\_itemsets.itemsets[0:11], rotation=15)

plt.xlabel('item-sets')

plt.ylabel('support')

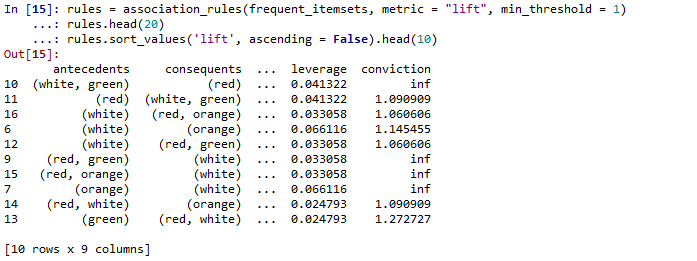
plt.show()



rules = association\_rules(frequent\_itemsets, metric = "lift", min\_threshold = 1)

rules.head(20)

rules.sort\_values('lift', ascending = False).head(10)



**Summary:-**

* **The mobiles having highest supports (most frequently sell) are of colour white, red & blue etc in order.**
* **The combination of mobile colour having high lift ratios with considerable support are “white , green, red”, “whit, red & orange”.**

**6. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.**

**Ans:-**

* R and Python code results the best mobiles in colour having high support and lift ratio in market for previous days.
* White & red mobile colour having highest support means highest demand in market. So make availablity of perticular colour mobile in more.
* White, green & red colour mobile colour has the highest lift ratio, means those who bought white colour mobile also bought green & red colour in same transaction oe sequential transaction. So if we can provide mobile covers having better appearance & corresponding colour as a free gift along with mobile purchae or in a dicount rate that will attract customers thereby can increase the sails.

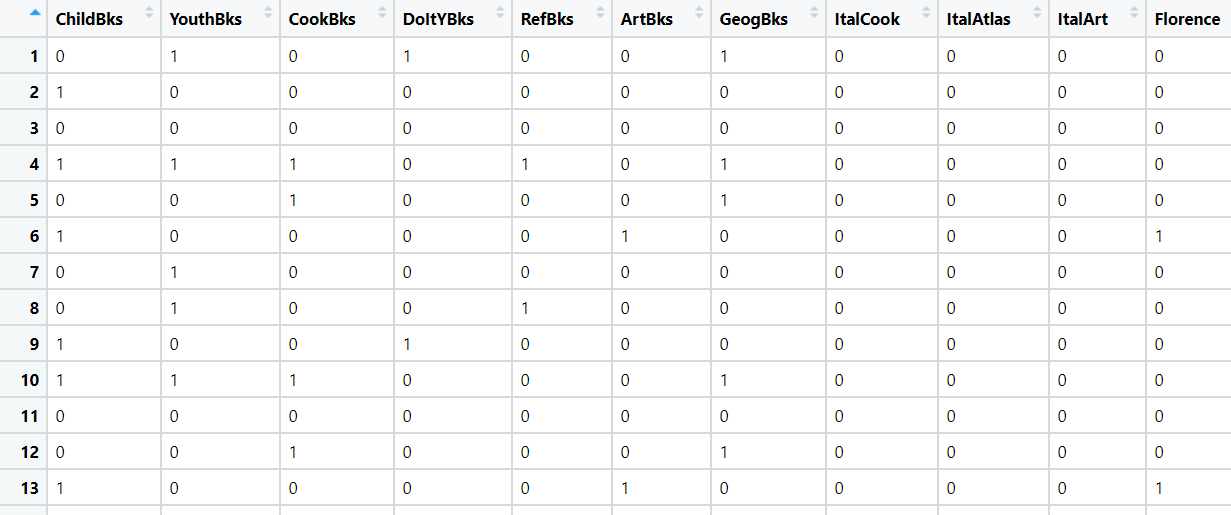
**Note:**

1. For each assignment, the solution should be submitted in the above format
2. Research and Perform all possible steps for improving the rules and also check if you can take out sub rules from main rules.
3. All the codes (executable programs) are running without errors
4. Documentation of the module should be submitted along with R & Python codes, elaborating on every step mentioned here that is commenting is necessary in the codes.
5. Please send all files at once whilst submitting assignments.

**Problem Statement: -**

Kitabi Duniya , a famous book store in India, which was established before Independence, the growth of the company was incremental year by year, but due to online selling of books and wide spread Internet access its annual growth started to collapse, seeing sharp downfalls, you as a Data Scientist help this heritage book store gain its popularity back and increase footfall of customers and provide ways the business can improve exponentially, apply Association Rule Algorithm, explain the rules, and visualize the graphs for clear understanding of solution.

**1.) Books.csv**

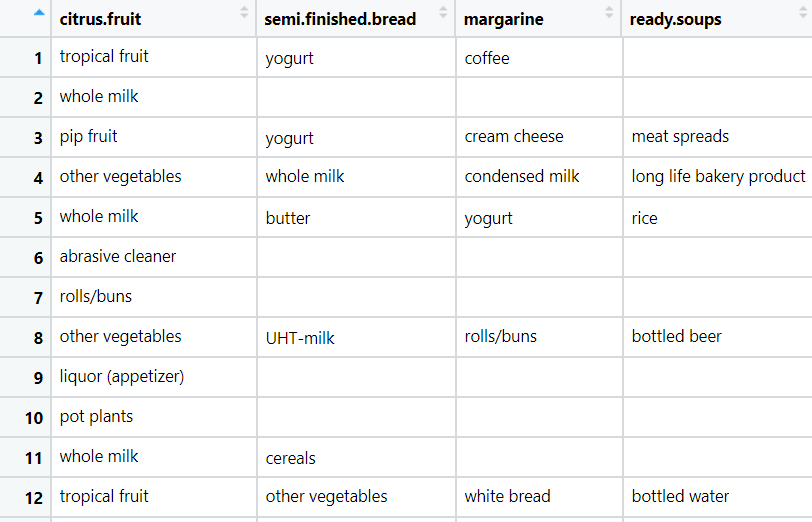


**Problem Statement:**

**The Departmental Store, has gathered the data of the products it sells on a**

**Daily basis. Using Association Rules concepts, provide the insights on the rules and the plots.**

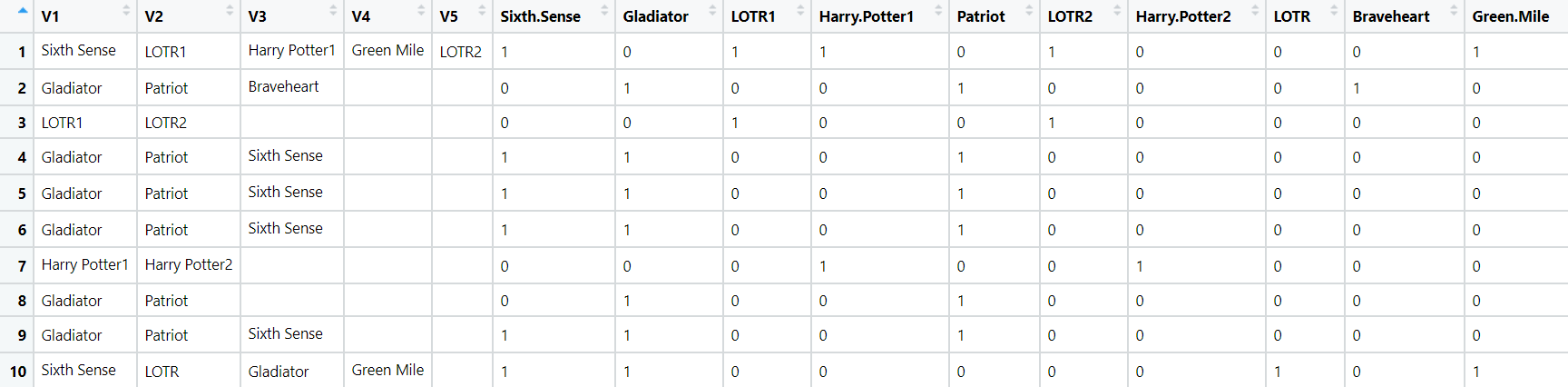
**2.) Groceries.csv**



**Problem Statement:**

**A film distribution company wants to target audience based on their likes and dislikes, you as a Chief Data Scientist Analyze the data and come up with different rules of movie list so that the business objective is achieved.**

**3.) my\_movies.csv**



**Problem Statement: -**

**A Mobile Phone manufacturing company wants to launch its three brand new phone into the market, but before going with its traditional marketing approach this time it want to analyze the data of its previous model sales in different regions and you have been hired as an Data Scientist to help them out, use the Association rules concept and provide your insights to the company’s marketing team to improve its sales.**

**4.) myphonedata.csv**



**Problem Statement: -**

**A retail store in India, has its transaction data, and it would like to know the buying pattern of the consumers in its locality, you have been assigned this task to provide the manager with rules on how the placement of products needs to be there in shelves so that it can improve the buying patterns of consumes and increase customer footfall.**

**5.) transaction\_retail.csv**

