Data Science

Project_Report_10

Under supervision of Dr/ Magda Madbouly

This report is to state the problem which will be solved in advance and to describe the R-code which is written to solve that problem

First:

The members of the group which worked on the solution of this problem:

No.	Name	ID	Group
1	احمد سمير عبد الفتاح امين	20221450304	G2
2	احمد صلاح عبد القادر مرسي	20221445695	G2
3	نور الدين محمد محمود	20221150099	G2
4	مروان أشرف محمد عبد الباقى	20221442040	G2

Second:

The problem description:

The program works on three main topics

1. Data visualization
2. Data Classification (Clustering)
3. Generating association rules between items of the data

And takes four main inputs from the user in order to perform its function properly which are

1. The path of the dataset which needs performing the main Three topics mentioned earlier
2. The number of clusters (groups) which are desired that the data be in
3. The minimum support of the apriori algorithm
4. The minimum confidence of the apriori algorithm

The output of the program:

1. Plots, graphs, scatter plots, box plots and pie graphs (Data visualization) in order to be easy to the user to understand the complex and the huge amount of data
2. Clusters (groups) of the data to provide relations between items of the data (if exist)
3. Association rules between the items of the data to provide better understanding of those items

Third:

Dataset description:

The dataset is about a grocery shop's transactions in a time interval which consists of:

1. Items: which the customers have bought
2. Count: which is the number of items that each customer has bought
3. Customer: which are the names of the customers
4. Age: the ages of the customers
5. City: the cities which the customers came from
6. Payment Type: the type of payment that the customer has used to pay for his\her items (Cash and Credit)

- The number of transactions is 9835
- divided in cash and credit
- The number of transactions with cash payment type is 4957
- The number of transactions with credit payment type is 4878
- The age of the customers varies between 22 and 60 years old
- The cities which the customers came from are (Alexandria - Aswan – Cairo – Dakahlia - Fayoum - Gharbia Giza – Hurghada – Port Said - Sohag
- The maximum city of customers is Alexandria with 1954 customers
- The minimum city of customers is Giza with 623 customers
- The maximum number of items bought is 32
- The minimum number of items bought is 1
- The total number of items bought in this dataset during this interval of time is 43367 items

Fourth:

The problem explanation:

The dataset which is provided is huge and can't be useful as is it must be cleaned and shown in a way that gives us relations and information in order to enhance the sales of the grocery shop.

Group members' roles:

احمد سمير عبد الفتاح امين: Data visualization

احمد صلاح عبد القادر مرسي: clustering (K-Means)

نور الدين محمد محمود: association rules (apriori algorithm)

مروان أشرف محمد عبد الباقي: report writing and rechecking functionality of code

Fifth:

Libraries used in the code:

library("dplyr")→ To manipulate the data and to use functions like (select , mutate , arrange , filter) to clean data

library("stats")→ To be able to use dplyr package

library("arules")→ To be able to use association rules (apriori algorithm)

library("gtools")→ To calculate permutation

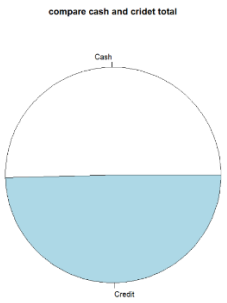
1. The visualization of the data:

- Compare cash and credit totals:

Code:

```
table(dataset$paymentType)

pie(

x= table(dataset$paymentType),

main="compare cash and cridet total",

)
```

Output:



- Compare each age and sum of total spending:

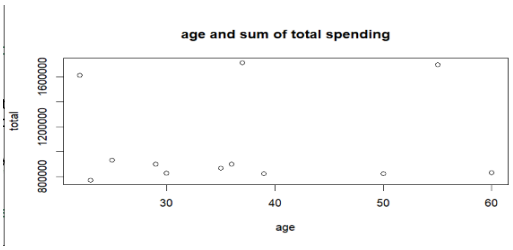
Code:

```
total_age<-group_by(dataset,age)

total_age<-summarize(total_age,totalage=sum(total))

total_age
```

plot(x=total_age$age, y=total_age$totalage,main = "age and sum of total spending",

xlab ="age",ylab="total")

output:

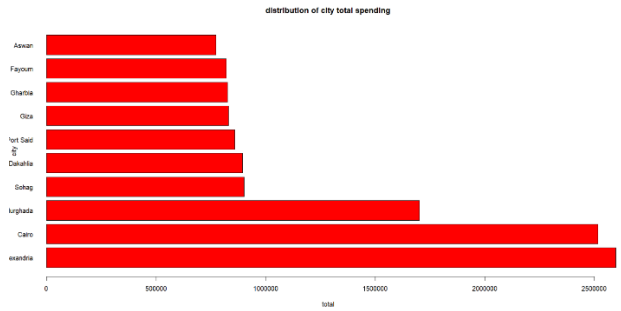


- Show each city total spending and arrange it by total descending:

Code:

```
total_spending<-group_by(dataset,city)
total_spending<-summarize(total_spending,totalsp=sum(total))
total_spending<-arrange(total_spending,desc(total_spending$totalsp))
total_spending
barplot(
  height =total_spending$totalsp,
  name=total_spending$city,
  col="red",
  main = "distribution of city total spending",
  xlab="total",
  ylab = "city",
  horiz=TRUE,
  las=1
)
```

Output:

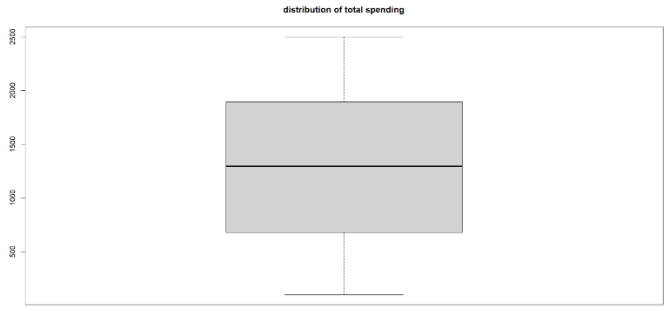


- Display the distribution of total spending:

Code:

```
boxplot(
  x=dataset$total,
  main="distribution of total spending")
```

Output:



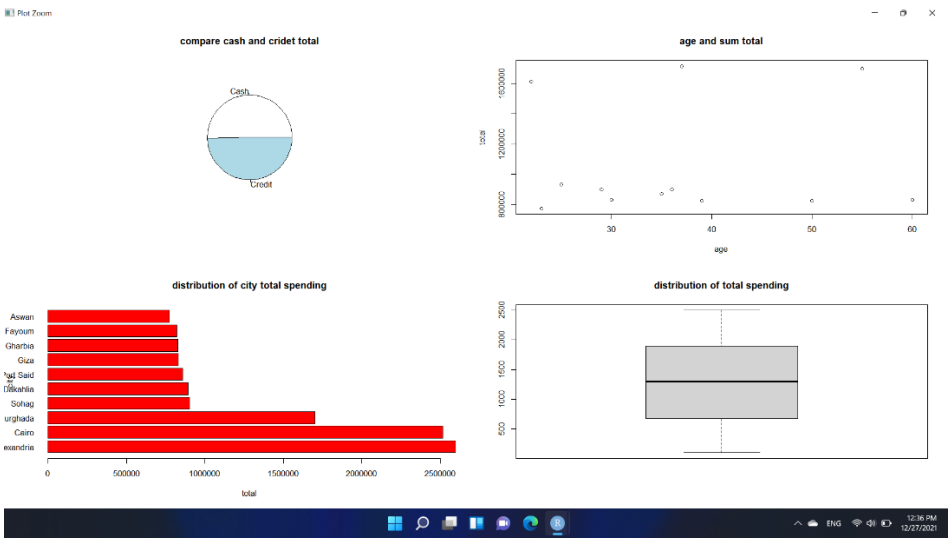
- Put all previous plots in one dashboard:

Code:

```
par(mfrow=c(2,2))


pie(
  x= table(dataset$paymentType),
  main="compare cash and cridet total",
)


plot(x=total_age$age, y=total_age$totalage,main = "age and sum total",
```

```
    xlab ="age",ylab="total" )


barplot(
  height =total_spending$totalsp,
  name=total_spending$city,
  col="red",
  main = "distribution of city total spending",
  xlab="total",
  ylab = "city",
  horiz=TRUE,
  las=1
)
boxplot(
  x=dataset$total,
  main="distribution of total spending")
```

Output:



2. The clustering of the Data (by K-Means):

- Cleaning the data:

Code:

datasetPath <- readline ("Enter Path (ex: c:/xxx/xxx):")→ to make the use enter the file path of the dataset

datapoints <- select (datapoints, customer, age, total)→ to select the columns from the dataset with names (customer – age - total) and neglect the rest of the dataset

datapoints <- group_by(datapoints, customer ,age)

datapoints <- as.data.frame (summarise (datapoints, SumTotal = sum(total)))

datapoints

Output:

```
> datapoints
   customer age  Total
1      Adel  50 824064
2     Ahmed  30 829587
3      Eman  23 772871
4    Farida  22 794570
5     Hanan  22 819231
6      Huda  39 825147
7     Magdy  36 901010
8     Maged  60 831272
9   Mohamed  25 932250
10    Rania  37 893789
11    Sameh  35 869668
12     Samy  55 841167
13    Sayed  37 820900
14   Shimaa  55 857901
15    Walaa  29 900797
>
```

- Kmeans

x <- readline (prompt = "How many clusters? ") →to take the number of clusters (Groups) from the user

KM <- kmeans (datapoints [, c (FALSE,TRUE,TRUE)], centers = x)

cluster <- KM$cluster

result <- cbind (datapoints, cluster)

result <- arrange (result, cluster, customer)

result

Output:

	customer	age	SumTotal	cluster
1	Adel	50	824064	1
2	Ahmed	30	829587	1
3	Eman	23	772871	1
4	Farida	22	794570	1
5	Hanan	22	819231	1
6	Huda	39	825147	1
7	Maged	60	831272	1
8	Samy	55	841167	1
9	Sayed	37	820900	1
10	Magdy	36	901010	2
11	Mohamed	25	932250	2
12	Rania	37	893789	2
13	Sameh	35	869668	2
14	Shimaa	55	857901	2
15	Walaa	29	900797	2

The method which is used for clustering is K-Means

3. The generation of association rule (by apriori algorithm):

Code:

itemsPath <- readline ("Enter Path (ex: c:/xxx/xxx): ")→ to take the dataset path from User as input

data_items<-read.transactions(itemsPath , sep =",")

inspect(data_items)

min_support<-as.numeric(readline("what is the minimum support? "))

min_confidince<-as.numeric(readline("what is the minimum confidince? "))

if((min_support<=1 & min_support>=0.001) & (min_confidince<=1 & min_confidince>=0.001)){ → to make the minimum confidence between 1 and 0.001 and to make the minimum support between 1 and 0.001

apriori_rules <- apriori(data_items,parameter = list(supp = min_support, conf = min_confidince,minlen=2))

inspect(apriori_rules)

}else{

print("please enter numbers between 0.001 and 1")

}

Output:

```
> library("gtools")
> data_items<-read.transactions("C:/Users/1Marwan Ashraf/.1A MyStuff/الكلية/Data Science/data project/items.txt",sep =",")
> apriori_rules <- apriori(data_items,parameter = list(supp = 0.01, conf = 0.01,minlen=2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
       0.01    0.1    1 none FALSE            TRUE       5    0.01      2     10
 target  ext
  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 98

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
sorting and recoding items ... [88 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [522 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Console Terminal Jobs

R 4.1.2 · C:/Users/jacko/

```
> inspect(apriori_rules)
      lhs                    rhs                    support    confidence coverage   lift      count
[1]   {hard cheese}       => {whole milk}           0.01006609 0.41250000 0.02440264 1.6143802  99
[2]   {whole milk}        => {hard cheese}          0.01006609 0.03939515 0.25551601 1.6143802  99
[3]   {butter milk}       => {other vegetables}     0.01037112 0.37090909 0.02796136 1.9189326 102
[4]   {other vegetables}  => {butter milk}          0.01037112 0.05365597 0.19328927 1.9189326 102
[5]   {butter milk}       => {whole milk}           0.01159126 0.41454545 0.02796136 1.6223854 114
[6]   {whole milk}        => {butter milk}          0.01159126 0.04536411 0.25551601 1.6223854 114
[7]   {ham}               => {whole milk}           0.01148958 0.44140625 0.02602949 1.7275091 113
[8]   {whole milk}        => {ham}                  0.01148958 0.04496618 0.25551601 1.7275091 113
[9]   {sliced cheese}     => {whole milk}           0.01077783 0.43983402 0.02450432 1.7213560 106
[10]  {whole milk}        => {sliced cheese}        0.01077783 0.04218066 0.25551601 1.7213560 106
[11]  {oil}               => {whole milk}           0.01128622 0.40217391 0.02806304 1.5739675 111
[12]  {whole milk}        => {oil}                  0.01128622 0.04417031 0.25551601 1.5739675 111
[13]  {berries}           => {yogurt}               0.01057448 0.31901840 0.03314692 2.2835124 104
[14]  {yogurt}            => {berries}              0.01057448 0.07569141 0.13970513 2.2835124 104
[15]  {berries}           => {other vegetables}     0.01016777 0.30674847 0.03314692 1.5869917 100
[16]  {other vegetables}  => {berries}              0.01016777 0.05260389 0.19328927 1.5869917 100
[17]  {berries}           => {whole milk}           0.01169293 0.35276074 0.03314692 1.3805817 115
[18]  {whole milk}        => {berries}              0.01169293 0.04576204 0.25551601 1.3805817 115
[19]  {onions}            => {other vegetables}     0.01423488 0.45901639 0.03101169 2.3747639 140
[20]  {other vegetables}  => {onions}               0.01423488 0.07364545 0.19328927 2.3747639 140
[21]  {onions}            => {whole milk}           0.01209964 0.39016393 0.03101169 1.5269647 119
[22]  {whole milk}        => {onions}               0.01209964 0.04735376 0.25551601 1.5269647 119
[23]  {hamburger meat}    => {other vegetables}     0.01382816 0.41590214 0.03324860 2.1517083 136
[24]  {other vegetables}  => {hamburger meat}       0.01382816 0.07154129 0.19328927 2.1517083 136
[25]  {hamburger meat}    => {whole milk}           0.01474326 0.44342508 0.03324860 1.7354101 145
[26]  {whole milk}        => {hamburger meat}       0.01474326 0.05769996 0.25551601 1.7354101 145
[27]  {hygiene articles}  => {whole milk}           0.01281139 0.38888889 0.03294357 1.5219746 126
[28]  {whole milk}        => {hygiene articles}     0.01281139 0.05013928 0.25551601 1.5219746 126
[29]  {salty snack}       => {other vegetables}     0.01077783 0.28494624 0.03782410 1.4741958 106
[30]  {other vegetables}  => {salty snack}          0.01077783 0.05576013 0.19328927 1.4741958 106
[31]  {salty snack}       => {whole milk}           0.01118454 0.29569892 0.03782410 1.1572618 110
```

The algorithm which is used for association rules is Apriori Algorithm