Research Question

You are a Data analyst at Carrefour Kenya and are currently undertaking a project that will inform the marketing department on the most relevant marketing strategies that will result in the highest no. of sales (total price including tax)

Association Analysis

```
# We first we install the required arules library
#
install.packages("arules")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

```
# Loading the arules library
#
library(arules)
```

```
Loading required package: Matrix


Attaching package: 'arules'


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

    abbreviate, write
```

```
# Loading our transactions dataset from our csv file
path <-"http://bit.ly/SupermarketDatasetII"

trans<-read.transactions(path, sep = ",")
trans
```

```
Warning message in asMethod(object):
"removing duplicated items in transactions"
transactions in sparse format with
 7501 transactions (rows) and
 119 items (columns)
```

```
# Verifying the object's class
class(trans)
```

'transactions'

```
# Previewing our first 5 transactions
inspect(trans[1:5])
```

```
    items
[1] {almonds,
     antioxydant juice,
     avocado,
     cottage cheese,
     energy drink,
     frozen smoothie,
     green grapes,
     green tea,
     honey,
     low fat yogurt,
     mineral water,
     olive oil,
     salad,
     salmon,
     shrimp,
     spinach,
     tomato juice,
     vegetables mix,
     whole weat flour,
     yams}
[2] {burgers,
     eggs,
     meatballs}
[3] {chutney}
[4] {avocado,
     turkey}
[5] {energy bar,
     green tea,
     milk,
     mineral water,
     whole wheat rice}
```

```
#Preview the items that make up our dataset,
items<-as.data.frame(itemLabels(trans))
colnames(items) <- "Item"
head(items, 10)
```

A data.frame: 10 × 1

	Item
	<chr>
1	almonds
2	antioxydant juice
3	asparagus
4	avocado
5	babies food
6	bacon

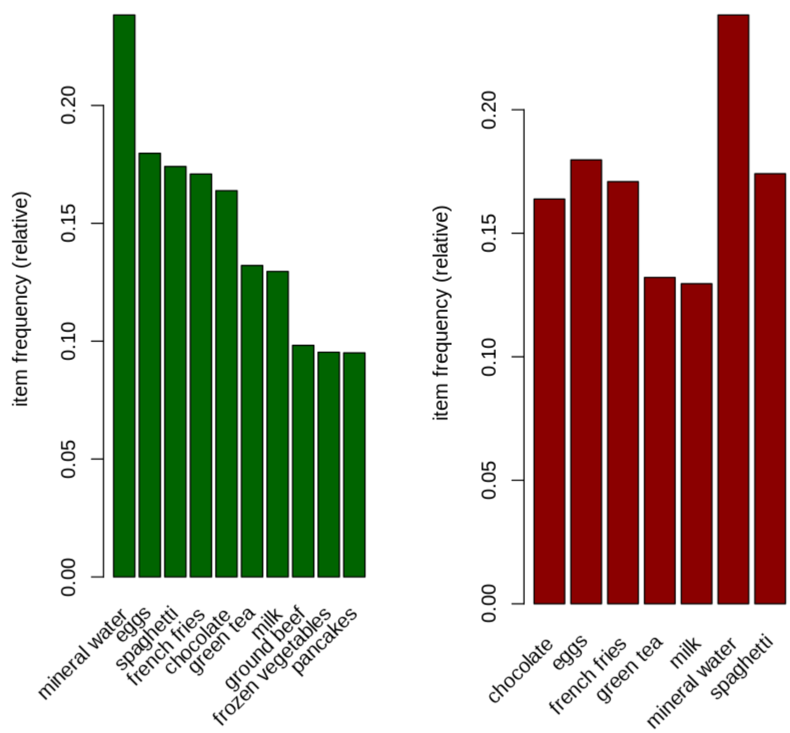
```
# Generating a summary of the transaction dataset
# This would give us some information such as the most purchased items,
# distribution of the item sets (no. of items purchased in each transaction), etc.
summary(trans)
```

```
transactions as itemMatrix in sparse format with
 7501 rows (elements/itemsets/transactions) and
 119 columns (items) and a density of 0.03288973

most frequent items:
mineral water          eggs     spaghetti  french fries     chocolate
         1788          1348          1306          1282          1229
      (Other)
        22405

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
1754 1358 1044  816  667  493  391  324  259  139  102   67   40   22   17    4
  18   19   20
   1    2    1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   3.914   5.000  20.000

includes extended item information - examples:
             labels
1           almonds
2 antioxydant juice
3         asparagus
```

```
# Exploring the frequency of some articles
# i.e. transacations ranging from 6 to 10 and performing
# some operation in percentage terms of the total transactions
#
itemFrequency(trans[, 6:10],type = "absolute")
round(itemFrequency(trans[, 6:10],type = "relative")*100,2)
```

bacon: 65 barbecue sauce: 81 black tea: 107 blueberries: 69 body spray: 86

```
# Producing a chart of frequencies and fitering
# to consider only items with a minimum percentage
# of support/ considering a top x of items
# ---
# Displaying top 10 most common items in the transactions dataset
# and the items whose relative importance is at least 10%
#
par(mfrow = c(1, 2))
# plot the frequency of items
itemFrequencyPlot(trans, topN = 10,col="darkgreen")
itemFrequencyPlot(trans, support = 0.1,col="darkred")
```



Items that are more common include mineral water, eggs, spaghetti, french fries and chocolate.

```
# Building a model based on association rules
# We use Min Support as 0.001 and confidence as 0.8
rules <- apriori (trans, parameter = list(supp = 0.001, conf = 0.8))
rules
```

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.8    0.1    1 none FALSE            TRUE       5   0.001      1
 maxlen target  ext
     10  rules TRUE

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

Absolute minimum support count: 7

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[119 item(s), 7501 transaction(s)] done [0.00s].
sorting and recoding items ... [116 item(s)] done [0.00s].
```

```
# Building a apriori model with Min Support as 0.002 and confidence as 0.8.
rules2 <- apriori (trans,parameter = list(supp = 0.002, conf = 0.8))
rules2
```

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.8    0.1    1 none FALSE            TRUE       5   0.002      1
 maxlen target  ext
     10  rules TRUE

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

Absolute minimum support count: 15

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[119 item(s), 7501 transaction(s)] done [0.00s].
sorting and recoding items ... [115 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 done [0.00s].
writing ... [2 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
set of 2 rules
```

```
# Building apriori model with Min Support as 0.002 and confidence as 0.6.
rules3 <- apriori (trans, parameter = list(supp = 0.001, conf = 0.6))
rules3
```

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.6    0.1    1 none FALSE            TRUE       5   0.001      1
 maxlen target  ext
     10  rules TRUE

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

Absolute minimum support count: 7

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[119 item(s), 7501 transaction(s)] done [0.00s].
```

```
#Performing an exploration of the model
summary(rules)
```

```
set of 74 rules

rule length distribution (lhs + rhs):sizes
 3  4  5  6
15 42 16  1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.000   4.000   4.000   4.041   4.000   6.000

summary of quality measures:
    support           confidence        coverage             lift
 Min.   :0.001067   Min.   :0.8000   Min.   :0.001067   Min.   : 3.356
 1st Qu.:0.001067   1st Qu.:0.8000   1st Qu.:0.001333   1st Qu.: 3.432
 Median :0.001133   Median :0.8333   Median :0.001333   Median : 3.795
 Mean   :0.001256   Mean   :0.8504   Mean   :0.001479   Mean   : 4.823
 3rd Qu.:0.001333   3rd Qu.:0.8889   3rd Qu.:0.001600   3rd Qu.: 4.877
 Max.   :0.002533   Max.   :1.0000   Max.   :0.002666   Max.   :12.722
     count
 Min.   : 8.000
 1st Qu.: 8.000
 Median : 8.500
 Mean   : 9.419
 3rd Qu.:10.000
 Max.   :19.000

mining info:
  data ntransactions support confidence
 trans          7501   0.001        0.8
```

```
# Observing rules built in our model i.e. first 5 model rules
# ---
#
inspect(rules[1:5])
```

```
    lhs                              rhs              support     confidence
```

```
    [1] {frozen smoothie,spinach}    => {mineral water} 0.001066524 0.8888889
    [2] {bacon,pancakes}             => {spaghetti}     0.001733102 0.8125000
    [3] {nonfat milk,turkey}         => {mineral water} 0.001199840 0.8181818
    [4] {ground beef,nonfat milk}    => {mineral water} 0.001599787 0.8571429
    [5] {mushroom cream sauce,pasta} => {escalope}      0.002532996 0.9500000
        coverage    lift      count
    [1] 0.001199840  3.729058  8
    [2] 0.002133049  4.666587 13
    [3] 0.001466471  3.432428  9
    [4] 0.001866418  3.595877 12
    [5] 0.002666311 11.976387 19
```

```
# Ordering these rules by a criteria such as the level of confidence
# then looking at the first five rules.
# We can also use different criteria such as: (by = "lift" or by = "support")
#
rules<-sort(rules, by="confidence", decreasing=TRUE)
inspect(rules[1:5])
```

```
        lhs                                            rhs             support
    [1] {french fries,mushroom cream sauce,pasta} => {escalope}      0.001066524
    [2] {ground beef,light cream,olive oil}       => {mineral water} 0.001199840
    [3] {cake,meatballs,mineral water}            => {milk}          0.001066524
    [4] {cake,olive oil,shrimp}                   => {mineral water} 0.001199840
    [5] {mushroom cream sauce,pasta}              => {escalope}      0.002532996
        confidence coverage    lift      count
    [1] 1.00       0.001066524 12.606723  8
    [2] 1.00       0.001199840  4.195190  9
    [3] 1.00       0.001066524  7.717078  8
    [4] 1.00       0.001199840  4.195190  9
    [5] 0.95       0.002666311 11.976387 19
```

```
# If we're interested in making a promotion relating to the sale of  milk,
# Let's create a subset of rules concerning these products
# This would tell us the items that the customers bought before purchasing yogurt

milk <- subset(rules, subset = rhs %pin% "milk")

# Then order by confidence
milk<-sort(milk, by="confidence", decreasing=TRUE)
inspect(milk[1:5])
```

```
        lhs                                rhs    support     confidence
    [1] {cake,meatballs,mineral water}    => {milk} 0.001066524 1.0000000
    [2] {escalope,hot dogs,mineral water} => {milk} 0.001066524 0.8888889
    [3] {meatballs,whole wheat pasta}     => {milk} 0.001333156 0.8333333
    [4] {black tea,frozen smoothie}       => {milk} 0.001199840 0.8181818
    [5] {burgers,ground beef,olive oil}   => {milk} 0.001066524 0.8000000
        coverage    lift     count
    [1] 0.001066524 7.717078  8
    [2] 0.001199840 6.859625  8
    [3] 0.001599787 6.430898 10
```

```
[4] 0.001466471 6.313973  9
[5] 0.001333156 6.173663  8
```

Before a customer buys milk there is a chance that they bought cake,meatballs and mineral water.

```
# What if we wanted to determine items that customers might buy
# who have previously bought milk?
# ---
#
# Subset the rules
milk <- subset(rules, subset = lhs %pin% "milk")

# Order by confidence
milk<-sort(milk, by="confidence", decreasing=TRUE)

# inspect top 5
inspect(milk[1:5])
```

```
    lhs                      rhs                     support confidence    coverage     l
[1] {frozen vegetables,
     milk,
     spaghetti,
     turkey}              => {mineral water}     0.001199840  0.9000000 0.001333156 3.775
[2] {cake,
     meatballs,
     milk}                => {mineral water}     0.001066524  0.8888889 0.001199840 3.729
[3] {burgers,
     milk,
     salmon}              => {spaghetti}         0.001066524  0.8888889 0.001199840 5.105
[4] {chocolate,
     ground beef,
     milk,
     mineral water,
     spaghetti}           => {frozen vegetables} 0.001066524  0.8888889 0.001199840 9.325
[5] {ground beef,
     nonfat milk}         => {mineral water}     0.001599787  0.8571429 0.001866418 3.595
```

A customer might buy mineral water, spaghetti and frozen vegetables after buying milk.

Conclusion and recommendation

From the above analysis we are able to know what product or products are likely to be purchased after another product is purchased. We are also able to tell which product or products are likely to have been purchased before a certain product is purchased. In our case, if the marketing department was to improve sales of milk, they can consider placing the products that can be purchased before or after milk(for example fresh vegetables and mineral water) is purchased on the same line so that the customer can access them easily and hence encourage the purchase of milk.