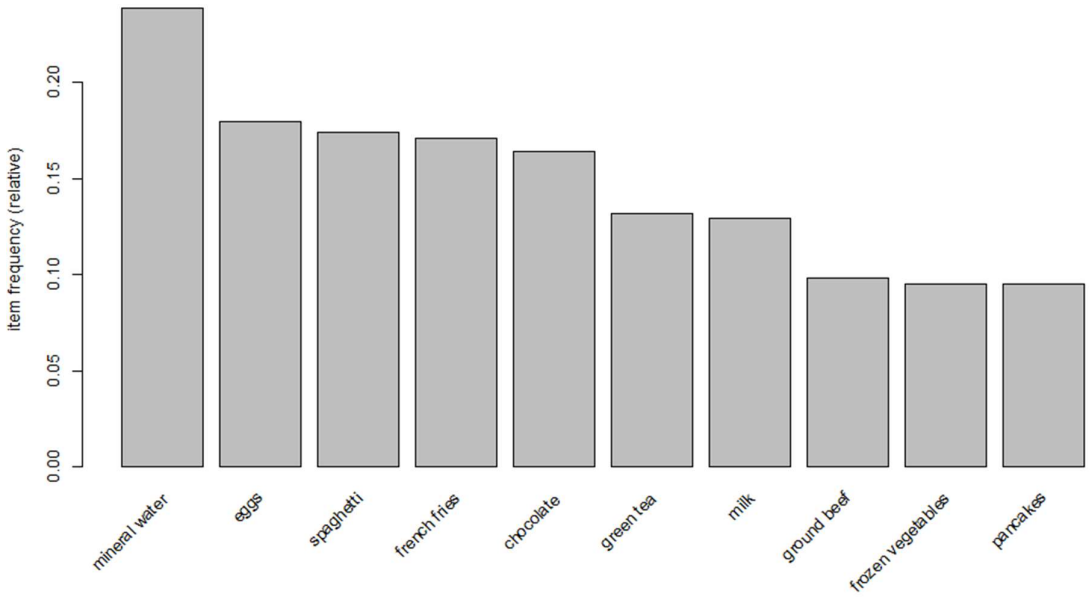
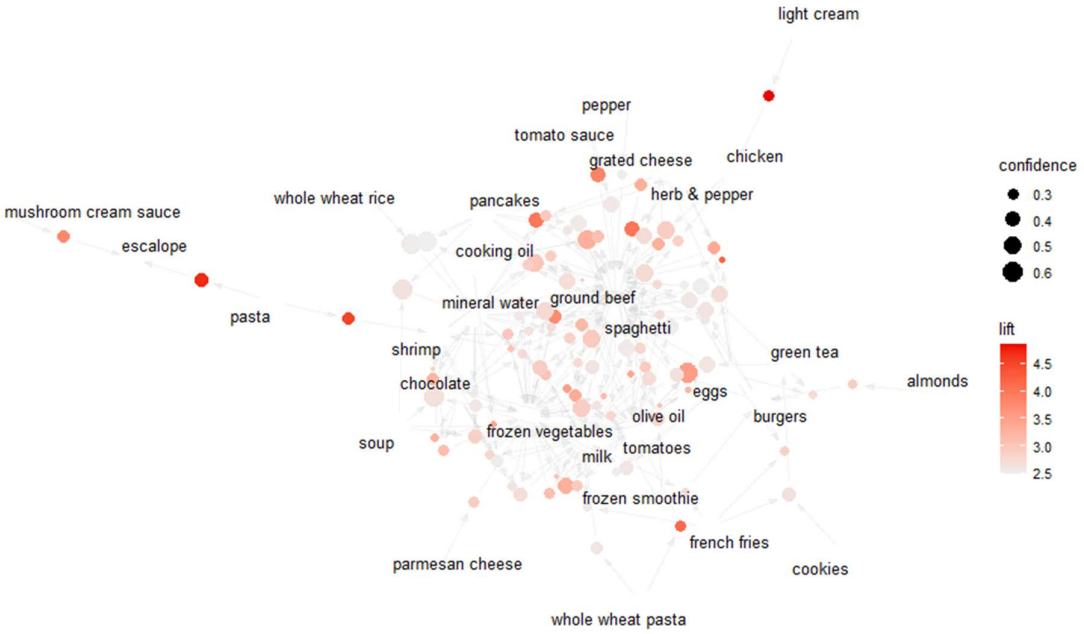
```
> # Loading data
> dataset = read.transactions('Market_Basket_Optimisation.csv',
+                             sep = ',', rm.duplicates = TRUE)
distribution of transactions with duplicates:
1
5
> # Structure
> str(dataset)
Formal class 'transactions' [package "arules"] with 3 slots
  ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
  .. .. ..@ i       : int [1:29358] 0 1 3 32 38 47 52 53 59 64 ...
  .. .. ..@ p       : int [1:7502] 0 20 23 24 26 31 32 34 37 40 ...
  .. .. ..@ Dim     : int [1:2] 119 7501
  .. .. ..@ Dimnames:List of 2
  .. .. .. ..$ : NULL
  .. .. .. ..$ : NULL
  .. .. ..@ factors : list()
  ..@ itemInfo   :'data.frame': 119 obs. of  1 variable:
  .. ..$ labels: chr [1:119] "almonds" "antioxydant juice" "asparagus" "avocado" ...
  ..@ itemsetInfo:'data.frame': 0 obs. of  0 variables
> # Fitting model
> # Training Apriori on the dataset
> set.seed = 220 # Setting seed
> associa_rules = apriori(data = dataset,
+                         parameter = list(support = 0.004,
+                                          confidence = 0.2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.2    0.1    1 none FALSE            TRUE       5   0.004      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: 30

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[119 item(s), 7501 transaction(s)] done [0.00s].
sorting and recoding items ... [114 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [811 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
> # Plot
> itemFrequencyPlot(dataset, topN = 10)
> # Visualising the results
> inspect(sort(associa_rules, by = 'lift')[1:10])
     lhs                         rhs                support confidence   coverage     lift count
[1]  {light cream}            => {chicken}       0.004532729  0.2905983 0.01559792 4.843951    34
[2]  {pasta}                  => {escalope}      0.005865885  0.3728814 0.01573124 4.700812    44
[3]  {pasta}                  => {shrimp}        0.005065991  0.3220339 0.01573124 4.506672    38
[4]  {eggs,
      ground beef}            => {herb & pepper} 0.004132782  0.2066667 0.01999733 4.178455    31
[5]  {whole wheat pasta}      => {olive oil}     0.007998933  0.2714932 0.02946274 4.122410    60
[6]  {herb & pepper,
      spaghetti}              => {ground beef}   0.006399147  0.3934426 0.01626450 4.004360    48
[7]  {herb & pepper,
      mineral water}          => {ground beef}   0.006665778  0.3906250 0.01706439 3.975683    50
[8]  {tomato sauce}           => {ground beef}   0.005332622  0.3773585 0.01413145 3.840659    40
[9]  {mushroom cream sauce} => {escalope}      0.005732569  0.3006993 0.01906412 3.790833    43
[10] {frozen vegetables,
      mineral water,
      spaghetti}              => {ground beef}   0.004399413  0.3666667 0.01199840 3.731841    33
> plot(associa_rules, method = "graph",
+      measure = "confidence", shading = "lift")
Warning message:
Too many rules supplied. Only plotting the best 100 using 'lift' (change control parameter max if needed).
>
```


Item frequency (relative)
0.00
0.05
0.10
0.15
0.20
mineral water
eggs
spaghetti
french fries
chocolate
green tea
milk
ground beef
frozen vegetables
pancakes


light cream
pepper
tomato sauce
grated cheese
chicken
herb & pepper
whole wheat rice
pancakes
mushroom cream sauce
escalope
cooking oil
mineral water
ground beef
pasta
spaghetti
shrimp
green tea
chocolate
eggs
almonds
olive oil
burgers
frozen vegetables
soup
milk
tomatoes
frozen smoothie
french fries
parmesan cheese
cookies
whole wheat pasta
confidence
0.3
0.4
0.5
0.6
lift
4.5
4.0
3.5
3.0
2.5