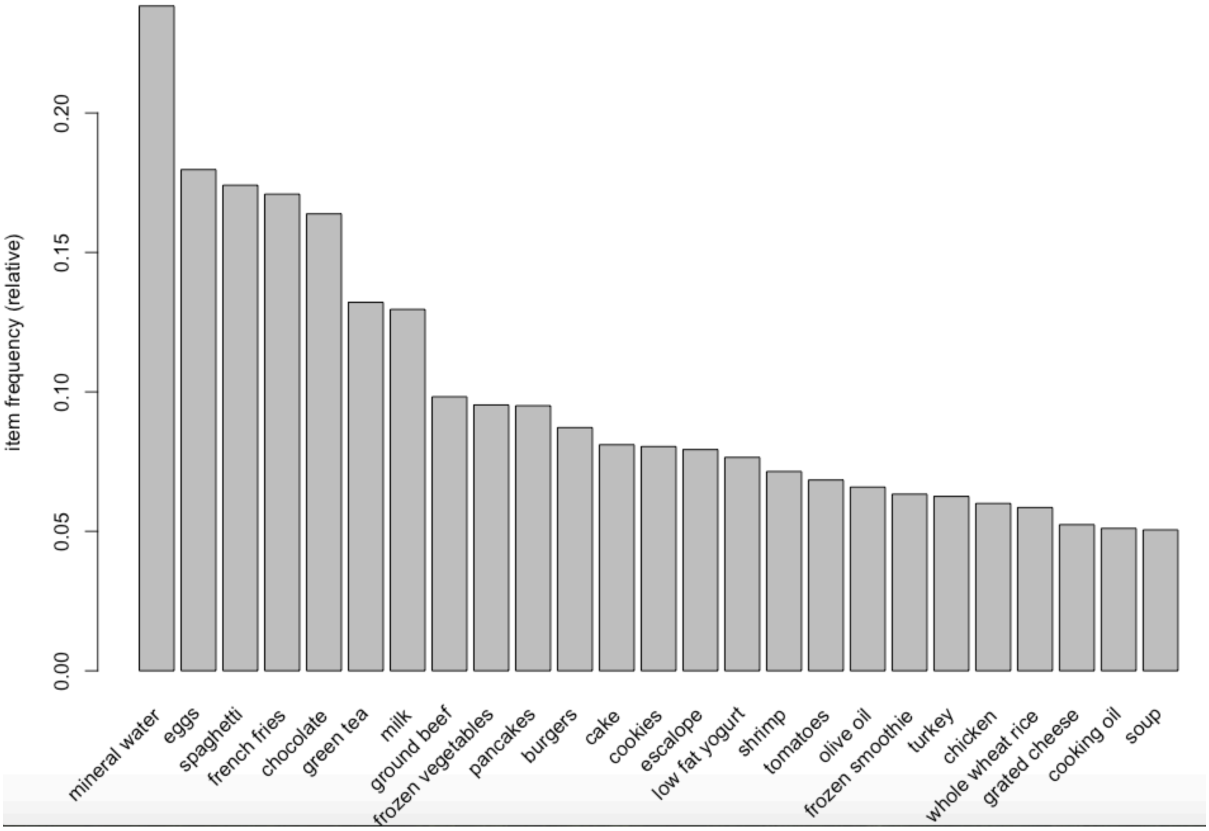
Market Basket Optimization

Goal: We have to find the association rules of different products so that the manager can optimize the sales of a supermarket by placing the products at the best place in the store.

About the dataset: The dataset consists of 7500 transactions that the customers bought from the supermarket during the whole week. This dataset will set as a base for our analysis since I am assuming that each customer goes to the supermarket once a week on average. The frequency plot of top 25 products purchased by the customers can be seen below. This will eventually help to find the best value of support.



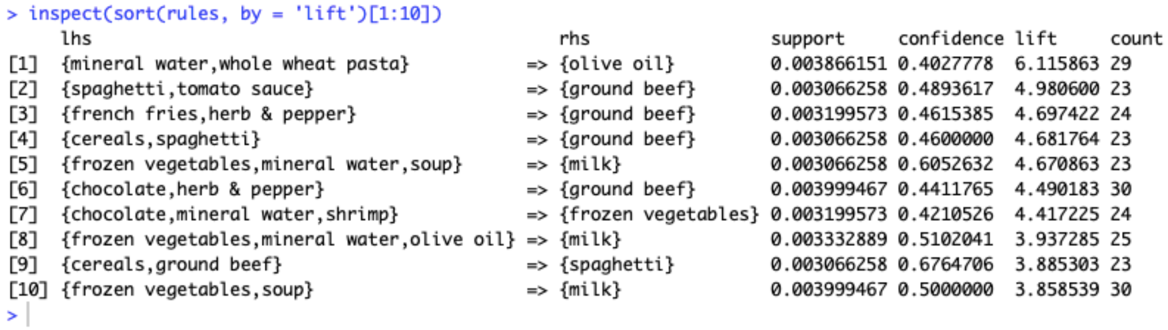
Data Preparation: For market basket optimization, dataset is handled in a different way. Currently, the dataset is in a .csv file. However, I am going to use "arules" package that takes a sparse matrix as an input. Therefore, I will transform the .csv file into a sparse matrix, which will assign 1 column to each of the products and there will be 7500 rows related to each of the transaction. Each cell will be 1 if that product was there in the consumer's basket, otherwise 0. I have also removed the duplicate values within the same transaction.

Training the model to build the rule: Assuming that the products that I am going to target are sold at least 3 times in a day, minimum support is equal to (3*5)/7500 = 0.003 and confidence = 0.4 that gave 281 strongest rules.

Visualizing the results:

1. **Support = 0.003, confidence = 0.4**

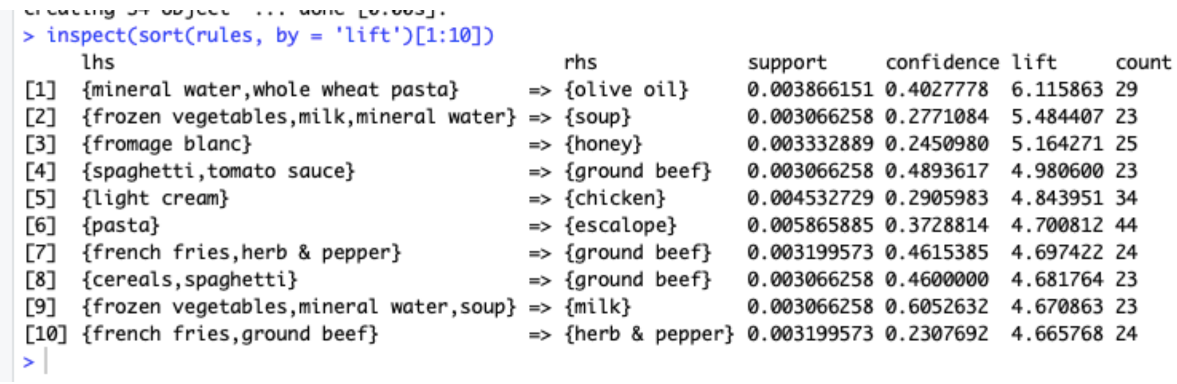
From the results, we can see the top 10 rules based on the decreasing values of lift value. For example, the top rule according to the model is that if a person buys mineral water and whole wheat pasta, then he will also but pasta. This rule has a support value of 0.0038 and confidence of 0.40, with a lift value of 6.11.



```
> inspect(sort(rules, by = 'lift')[1:10])
     lhs                                         rhs                  support     confidence lift     count
[1]  {mineral water,whole wheat pasta}        => {olive oil}          0.003866151 0.4027778  6.115863 29
[2]  {spaghetti,tomato sauce}                 => {ground beef}        0.003066258 0.4893617  4.980600 23
[3]  {french fries,herb & pepper}             => {ground beef}        0.003199573 0.4615385  4.697422 24
[4]  {cereals,spaghetti}                      => {ground beef}        0.003066258 0.4600000  4.681764 23
[5]  {frozen vegetables,mineral water,soup}   => {milk}               0.003066258 0.6052632  4.670863 23
[6]  {chocolate,herb & pepper}                => {ground beef}        0.003999467 0.4411765  4.490183 30
[7]  {chocolate,mineral water,shrimp}         => {frozen vegetables}  0.003199573 0.4210526  4.417225 24
[8]  {frozen vegetables,mineral water,olive oil} => {milk}            0.003332889 0.5102041  3.937285 25
[9]  {cereals,ground beef}                    => {spaghetti}          0.003066258 0.6764706  3.885303 23
[10] {frozen vegetables,soup}                 => {milk}               0.003999467 0.5000000  3.858539 30
>
```

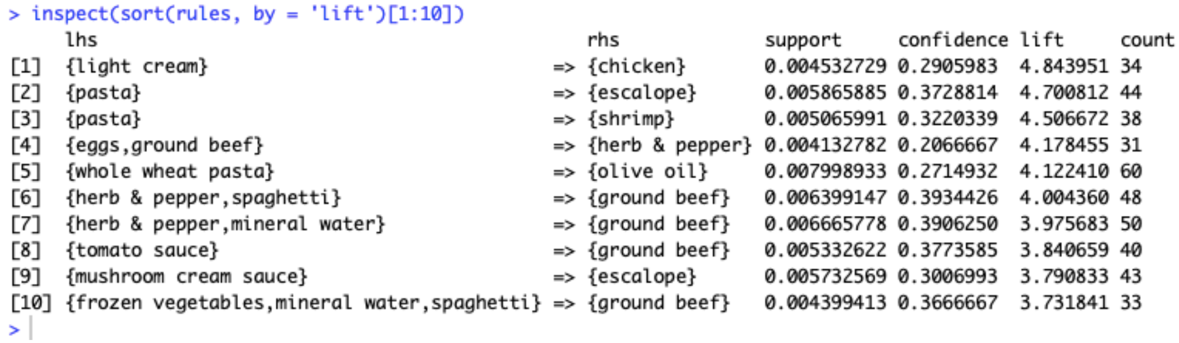
However, some of the rules looks irrelevant as they are not because of the Apriori rule, but because the support of those product was more and so they appeared on most of the transactions. For example, a person who bought chocolate and herb & pepper also bought ground beef looks irrelevant because the chocolate is bought by a lot of customers. Therefore, I am going to change the confidence to 0.2 to get better rules.

2. **Support = 0.003, confidence = 0.2**



```
creating S4 object  ... done [0.00s].
> inspect(sort(rules, by = 'lift')[1:10])
     lhs                                       rhs             support     confidence lift     count
[1]  {mineral water,whole wheat pasta}      => {olive oil}     0.003866151 0.4027778  6.115863 29
[2]  {frozen vegetables,milk,mineral water} => {soup}          0.003066258 0.2771084  5.484407 23
[3]  {fromage blanc}                        => {honey}         0.003332889 0.2450980  5.164271 25
[4]  {spaghetti,tomato sauce}               => {ground beef}   0.003066258 0.4893617  4.980600 23
[5]  {light cream}                          => {chicken}       0.004532729 0.2905983  4.843951 34
[6]  {pasta}                                => {escalope}      0.005865885 0.3728814  4.700812 44
[7]  {french fries,herb & pepper}           => {ground beef}   0.003199573 0.4615385  4.697422 24
[8]  {cereals,spaghetti}                    => {ground beef}   0.003066258 0.4600000  4.681764 23
[9]  {frozen vegetables,mineral water,soup} => {milk}          0.003066258 0.6052632  4.670863 23
[10] {french fries,ground beef}             => {herb & pepper} 0.003199573 0.2307692  4.665768 24
>
```

3. Support = 0.004, Confidence = 0.2 (Best rules)



```
> inspect(sort(rules, by = 'lift')[1:10])
     lhs                                       rhs             support     confidence lift     count
[1]  {light cream}                          => {chicken}       0.004532729 0.2905983  4.843951 34
[2]  {pasta}                                => {escalope}      0.005865885 0.3728814  4.700812 44
[3]  {pasta}                                => {shrimp}        0.005065991 0.3220339  4.506672 38
[4]  {eggs,ground beef}                     => {herb & pepper} 0.004132782 0.2066667  4.178455 31
[5]  {whole wheat pasta}                    => {olive oil}     0.007998933 0.2714932  4.122410 60
[6]  {herb & pepper,spaghetti}              => {ground beef}   0.006399147 0.3934426  4.004360 48
[7]  {herb & pepper,mineral water}          => {ground beef}   0.006665778 0.3906250  3.975683 50
[8]  {tomato sauce}                         => {ground beef}   0.005332622 0.3773585  3.840659 40
[9]  {mushroom cream sauce}                 => {escalope}      0.005732569 0.3006993  3.790833 43
[10] {frozen vegetables,mineral water,spaghetti} => {ground beef}   0.004399413 0.3666667  3.731841 33
>
```

Conclusion: With the help of the above visualization, we can use hit and trial method and take different values of support and confidence and place the products according to these rules that will help in optimizing the sales of the store.