PROJECT REPORT

Supermarket Selles Transction Data Mining and Data Analysis Using Association Rule

Reporter:Eyasu Taye, Kalkidan Tesfaye, Chala Bahiru, Bilisie Melese

Department:Computer Science

CONTENTS:

Part Ⅰ

Introduction

Introduction

Supermarket data, Association rule mining, Frequent items

Supermarkets have large number of customers checking into the items and to know customers need its better to identify which products bought frequently, which products are bought together,and association between items.

Association rule mining is one of the principal problems treated in KDD and can be defined as extracting the interesting correlation and relation among huge number of transactions.

Frequent itemset is generally adopted to generate association rules. As the amount of data stored supermarket database grows twice as fast as the speed of the fastest processor available to analyse it. Main purpose of analysing frequent itemset is to find the association relationship among the large number of database items. It is used to describe the patterns of customers' purchase in the supermarket.

Part Ⅱ

Motivation and Description

Motivation

Our motivation behined mining the supermarket data?

Finding inherent regularities in data is the motivation behind this supermarket data analysis project. Association rule mining is one of the technique to identify underlying relations between different items, it helps to identify which items of supermarket mostly bought together and their correlations.

For instance, if item A and B are bought together more frequently then several steps can be taken to increase the profit.

1.Do no go far away

A and B can be placed together so that when a customer buys one of the product he/she doesn't have to go far away to buy the other product.

2.Adverisement Campaign.

People who buy one of the products can be targeted through an advertisement campaign to buy the other one.

3.Collective Discount

Collective discounts can be offered on those products, if the customer buys both.

4. What to buy together

Both A and B can be packaged together.

Part Ⅲ

Objective

Objective

What do we want to achieve?

The objective of analysing supermarket transaction data is for identification of items that frequently occurred togehter in the traniction found in the databases so that the out come will increase effectiveness in supermarket sell.

Part Ⅳ

Methodology

Methodology

Data Preprocessing	Step 1	Step 2	Step 3

The dataset has the following limitations

- Incomplete and Null values
- No header row is necessary
- Unused column

Data Cleaning

1. Make complete the data by inserting default value for null values.
2. Remove header option from the dataset since it is not necessary so the first row of the dataset can not be treated as header.

Data Reduction

Remove unused column from the dataset.

Data Trnsformation

Transform the csv dataset to list data type.

Methodology

So the data is become understandable by the machine to be processed.

That is

Each row corresponds to a transaction and,

Each column corresponds to an item purchased in that specific transaction.

Detail On Data Preprocessing

The dataset dimension

In [49]:
```
#check its dimension
store_data.shape
```

Out[49]: (7500, 27)

7500 rows

27 columns

Detail On Data Preprocessing

Columns which are no more important in the dataset

In [89]:
```
#count null values and convert tehm to NaN default value
store_data.isnull().sum()
```

Out[89]:
```
shrimp                 0
almonds             1754
avocado             3112
vegetables mix      4156
green grapes        4972
whole weat flour    5637
yams                6132
cottage cheese      6520
energy drink        6847
tomato juice        7106
low fat yogurt      7245
green tea           7347
honey               7414
salad               7454
mineral water       7476
salmon              7493
antioxydant juice   7497
frozen smoothie     7497
spinach             7498
olive oil           7500
meat                7500
onion               7500
garlic              7500
dairy               7500
apples              7500
seafood             7500
bananas             7500
dtype: int64
```


```
olive oil           7500
meat                7500
onion               7500
garlic              7500
dairy               7500
apples              7500
seafood             7500
bananas             7500
dtype: int64
```


Are not important for our objectives

Must be removed.

Detail On Data Preprocessing

The first row as a header

The first row is treated as a header, must be transformed to normal row

True = The value is null
False = The value is not null

Out[50]:

	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	...	frozen smoothie	spinach	olive oil	meat	pasta	soups	dairy	eg
0	False	False	False	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
1	False	True	True	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
2	False	False	True	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
3	False	False	False	False	False	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
4	False	True	True	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
7495	False	False	False	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
7496	False	False	False	False	False	False	True	True	True	True	...	True	True	True	True	True	True	True	Tr
7497	False	True	True	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
7498	False	False	True	True	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr
7499	False	False	False	False	True	True	True	True	True	True	...	True	True	True	True	True	True	True	Tr

7500 rows × 27 columns

Detail On Data Preprocessing

Data reduction by removing irrelevant columns

```
In [57]: #as we can see from the result the 20th to 27th column has all null values so we can elimintate it.
         store_data.drop(store_data.columns[[19,20,21,22,23,24,25,26]], axis=1, inplace=True)

In [58]: store_data.isnull().sum()

Out[58]: 0        0
         1     1754
         2     3112
         3     4156
         4     4972
         5     5637
         6     6132
         7     6520
         8     6847
         9     7106
         10    7245
         11    7347
         12    7414
         13    7454
         14    7476
         15    7493
         16    7497
         17    7497
         18    7498
         dtype: int64
```

```
olive oil        7500
meat             7500
onion            7500
garlic           7500
dairy            7500
apples           7500
seafood          7500
bananas          7500
dtype: int64
```

8 Columns are Removed

Detail On Data Preprocessing

A header row is removed
Null is transformed to NaN value

Out[21]:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	salad	mineral water	salmon	antioxydant juice	frozen smoothie	spinach
urgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
hutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

A header row is removed
Null is transformed to NaN value

* Now the dataset is clean enough

Data Analysis

The apriori class requires some parameter values to work.

- The first parameter must be list of list that to extract rules from.
- The second parameter is the min_support parameter.
- The min_confidence parameter.
- he min_lift parameter specifies the minimum lift value for the short listed rules.
- Finally, the min_length parameter specifies the minimum number of items that you want in your rules.

We want to create rules for items purchased five times a day in one week

That means

7*5 = 35 times per week,

So the support = 35/7500 = 0.0045

Let

- Minimum confidence = 20% or 0.2
- Lift = 3
- Min length = 2

by this parameters
48 rules are dicovered

Part V

Evaluation

Evalution

First Rule

```
Rule: chicken -> light cream
Support: 0.004532728969470737
Confidence: 0.2905982905982905 7
Lift: 4.843950617283950 95
=====================================
```

- The support value is 0.0045. This number is calculated by dividing the number of transactions containing light cream divided by total number of transactions.
- .The support value is 0.0045. This number is calculated by dividing the number of transactions containing light cream divided by total number of transactions.
- The confidence level for the rule is 0.2905 which shows that out of all the transactions that contain light cream, 29.05% of the transactions also contain chicken.
- The lift of 4.84 tells us that chicken is 4.84 times more likely to be bought by the customers who buy light cream compared to the default likelihood of the sale of chicken.

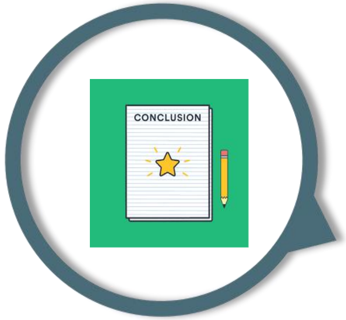
Evalution

Second Rule

```
Rule: mushroom cream sauce -> escalope
Support: 0.005732568990801226
Confidence: 0.3006993006993007
Lift: 3.790832696715049
=====================================
```

- The second rule states that **mushroom cream sauce** and **escalope** are bought frequently
- The support for mushroom cream sauce is 0.0057.
- The confidence for this rule is 0.3006 which means that out of all the transactions containing mushroom, 30.06% of the transactions are likely to contain escalope as well.
- Lift of 3.79 shows that the escalope is 3.79 more likely to be bought by the customers that buy mushroom cream sauce, compared to its default sale.

Conclusion



Association rule mining for supermarket dataset has been presented, Mining has been applied to sales data of dataset. In proposed project, the apriori algorithm has been used on super market dataset which gives associations of two products which has maximum support, It reduces the size of the itemsets in the database considerably providing a good performance. Thus, data mining helps consumers and industries better in the decision-making process.

Thank you for watching

Click here to add the text, the text is the extraction of your thought, in order to finally present the good effect of the release, please try to be concise and concise; if necessary, add or subtract the text.

Reporter:XXXX

Department:XXXXX

In [1]:
```
# Importing libraries
```

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from apyori import apriori
```

In [3]:
```
# Loading the dataset
```

In [4]:
```
store_data = pd.read_csv('I:\\Datasets\\store_data.csv')
```

In [5]:
```
#checking the dataset
```

In [6]:
```
    #check its dimension
store_data.shape
```

Out[6]: (7500, 27)

In [7]:
```
    #check null values
store_data.isnull()
```

Out[7]:

	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	...
0	False	False	False	True	True	True	True	True	True	True	...
1	False	True	True	True	True	True	True	True	True	True	...
2	False	False	True	True	True	True	True	True	True	True	...
3	False	False	False	False	False	True	True	True	True	True	...
4	False	True	True	True	True	True	True	True	True	True	...
...	...	...	...	...	...	...	...	...	...	...	...
7495	False	False	False	True	True	True	True	True	True	True	...
7496	False	False	False	False	False	False	True	True	True	True	...
7497	False	True	True	True	True	True	True	True	True	True	...
7498	False	False	True	True	True	True	True	True	True	True	...
7499	False	False	False	False	True	True	True	True	True	True	...

7500 rows × 27 columns

In [8]:
```
#count null values and convert tehm to NaN default value
store_data.isnull().sum()
```

Out[8]:
```
shrimp                  0
almonds              1754
avocado              3112
vegetables mix       4156
green grapes         4972
whole weat flour     5637
yams                 6132
cottage cheese       6520
energy drink         6847
tomato juice         7106
low fat yogurt       7245
green tea            7347
honey                7414
salad                7454
mineral water        7476
salmon               7493
antioxydant juice    7497
frozen smoothie      7497
spinach              7498
olive oil            7500
meat                 7500
onion                7500
garlic               7500
dairy                7500
apples               7500
seafood              7500
bananas              7500
dtype: int64
```

In [9]:
```
#for our processing we do not need a header row
```

In [10]:
```
store_data = pd.read_csv('I:\\Datasets\\store_data.csv', header=None)
```

In [11]:
```
store_data.head()
```

Out[11]:

	0	1	2	3	4	5	6	7	8	9	...	
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	...	f smo
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	...	

5 rows × 27 columns

In [119]:
```
#as we can see from the result the 20th to 27th column has all null values :
store_data.drop(store_data.columns[[19,20,21,22,23,24,25,26]], axis=1, inpla
```

In [120]:
```
store_data.isnull().sum()
```

Out[120]:
```
0        0
1     1754
2     3112
3     4156
4     4972
5     5637
6     6132
7     6520
8     6847
9     7106
10    7245
11    7347
12    7414
13    7454
14    7476
15    7493
16    7497
17    7497
18    7498
dtype: int64
```

In [121]:
```
store_data.head()
```

Out[121]:

	0	1	2	3	4	5	6	7	8	9	10
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN

In [122]:
```
#Data Proprocessing
#The Apriori library we are going to use requires our dataset to be in the f
#where the whole dataset is a big list and each transaction in the dataset i
#Currently we have data in the form of a pandas dataframe.
#To convert our pandas dataframe into a list of lists, we execute the script
```

In [123]:
```
records = []
for i in range(0, 7501):
    records.append([str(store_data.values[i,j]) for j in range(0, 18)])
```

In [124]:
```
#Applying Apriori
# The first parameter is the list of list that you want to extract rules fr
# The second parameter is the min_support parameter.
# This parameter is used to select the items with support values greater th
# Next, the min_confidence parameter filters those rules that have confiden
# Similarly, the min_lift parameter specifies the minimum lift value for th
# Finally, the min_length parameter specifies the minimum number of items t
```

In [125]:
```
# Let's suppose that we want rules for only those items that are purchased 
# since our dataset is for a one-week time period.
# The support for those items can be calculated as 35/7500 = 0.0045.
# The minimum confidence for the rules is 20% or 0.2.
# Similarly, we specify the value for lift as 3 and finally min_length is 2
```

In [126]:
```
association_rules = apriori(records, min_support=0.0045, min_confidence=0.2
```

In [127]:
```
# Viewing the Results
```

In [128]:
```
association_results = list(association_rules)
```

In [129]:
```
print(association_results)
```

```
[RelationRecord(items=frozenset({'light cream', 'chicken'}), support=0.0
04532728969470737, ordered_statistics=[OrderedStatistic(items_base=froze
nset({'light cream'}), items_add=frozenset({'chicken'}), confidence=0.29
059829059829057, lift=4.84395061728395)]), RelationRecord(items=frozense
t({'escalope', 'mushroom cream sauce'}), support=0.005732568990801226, o
rdered_statistics=[OrderedStatistic(items_base=frozenset({'mushroom crea
m sauce'}), items_add=frozenset({'escalope'}), confidence=0.300699300699
3007, lift=3.790832696715049)]), RelationRecord(items=frozenset({'past
a', 'escalope'}), support=0.005865884548726837, ordered_statistics=[Orde
redStatistic(items_base=frozenset({'pasta'}), items_add=frozenset({'esca
lope'}), confidence=0.3728813559322034, lift=4.700811850163794)]), Relat
ionRecord(items=frozenset({'ground beef', 'herb & pepper'}), support=0.0
15997866951073192, ordered_statistics=[OrderedStatistic(items_base=froze
nset({'herb & pepper'}), items_add=frozenset({'ground beef'}), confidenc
e=0.3234501347708895, lift=3.2919938411349285)]), RelationRecord(items=f
rozenset({'ground beef', 'tomato sauce'}), support=0.005332622317024397,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'tomato sauc
e'}), items_add=frozenset({'ground beef'}), confidence=0.377358490566037
7, lift=3.840659481324083)]), RelationRecord(items=frozenset({'whole whe
```

In [130]:
```
len(association_results)
```

Out[130]:
```
45
```

In [131]:
```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'light cream', 'chicken'}), support=0.0045
32728969470737, ordered_statistics=[OrderedStatistic(items_base=frozenset
({'light cream'}), items_add=frozenset({'chicken'}), confidence=0.29059829
059829057, lift=4.84395061728395)])
```

In [132]:
```python
for item in association_results:

    # first index of the inner list
    # Contains base item and add item
    pair = item[0]
    items = [x for x in pair]
    print("Rule: " + items[0] + " -> " + items[1])

    #second index of the inner list
    print("Support: " + str(item[1]))

    #third index of the list located at 0th
    #of the third index of the inner list

    print("Confidence: " + str(item[2][0][2]))
    print("Lift: " + str(item[2][0][3]))
    print("=====================================")
```

```
Rule: light cream -> chicken
Support: 0.004532728969470737
Confidence: 0.29059829059829057
Lift: 4.84395061728395
=====================================
Rule: escalope -> mushroom cream sauce
Support: 0.005732568990801226
Confidence: 0.3006993006993007
Lift: 3.790832696715049
=====================================
Rule: pasta -> escalope
Support: 0.005865884548726837
Confidence: 0.3728813559322034
Lift: 4.700811850163794
=====================================
Rule: ground beef -> herb & pepper
Support: 0.015997866951073192
Confidence: 0.3234501347708895
Lift: 3.2919938411349285
```

In [ ]: