Capstone Project Report

Name: Sumit Patel

Course: AI and ML (Batch – AUG 2020)

Duration: 10 months

Association Rule Mining: Market Basket Analysis

Problem Statement:

Apriori is a statistical algorithm for implementing associate rule mining, that primarily relies on three components: Life, Support and Confidence. Using this algorithm try to find the rules that describe the relation between each of the products that were brought by the customers.

Prerequisites

What things you need to install the software and how to install them:

Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-as-an-internal-or-externalcommand/. Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic. Second and easier option is to download anaconda and use its anaconda prompt to run the commands.

To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6 then run below commands in command prompt/terminal to install these packages:

pip install numpy

pip install pandas

pip install matplotlib

pip install mlxtend

If you have chosen to install anaconda then run below commands in anaconda prompt to install these packages:

conda install -c anaconda numpy

conda install -c anaconda pandas

conda install -c anaconda matplotlib

conda install -c anaconda mlxtend

Dataset used:

Dataset Link: Store Data
https://drive.google.com/file/d/1y5DYn0dGoSbC22xowBq2d4po6h1JxcTQ/view?usp=sharing

Method used for detection

Apriori, Association_rules

Importing the libraries and loading dataset.

```
# Import suitable libraries
import numpy as np
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
import matplotlib.pyplot as plt
```

```
data = pd.read_csv('store_data.csv', header = None)
data.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
0	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	spina
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

Training Data:

```
items = set()
for col in data:
    items.update(data[col].unique())
print(items)
```

```
{'parmesan cheese', 'mashed potato', nan, 'whole weat flour', 'grated cheese', 'green tea', 'whole wheat rice', 'nonfat milk',
'tomato sauce', 'ketchup', 'rice', 'fresh bread', 'turkey', 'barbecue sauce', 'antioxydant juice', 'cooking oil', 'flax seed',
'spinach', 'shallot', 'oil', 'mayonnaise', 'salt', 'brownies', 'chocolate', 'butter', 'carrots', 'mushroom cream sauce', 'cooki
es', 'herb & pepper', 'tomatoes', 'eggplant', 'french wine', 'hand protein bar', 'chocolate bread', 'eggs', 'toothpaste', 'spag
hetti', 'mint green tea', 'chutney', 'fresh tuna', 'bug spray', 'mint', 'white wine', 'water spray', 'ham', 'clothes accessorie
s', 'soup', 'honey', 'yogurt cake', 'shrimp', 'sparkling water', 'melons', 'gluten free bar', 'sandwich', 'light cream', 'crea
m', 'blueberries', 'frozen smoothie', 'vegetables mix', 'shampoo', 'napkins', 'extra dark chocolate', 'low fat yogurt', 'sala
d', 'cake', ' asparagus', 'pancakes', 'magazines', 'protein bar', 'almonds', 'milk', 'french fries', 'pepper', 'babies food',
'green grapes', 'energy bar', 'frozen vegetables', 'muffins', 'tea', 'corn', 'soda', 'bramble', 'avocado', 'burger sauce', 'pic
kles', 'red wine', 'candy bars', 'pet food', 'burgers', 'strawberries', 'strong cheese', 'cottage cheese', 'pasta', 'ground bee
f', 'fromage blanc', 'mineral water', 'gums', 'black tea', 'tomato juice', 'oatmeal', 'zucchini', 'escalope', 'green beans', 'd
essert wine', 'energy drink', 'cider', 'asparagus', 'chicken', 'body spray', 'hot dogs', 'champagne', 'yams', 'light mayo', 'ba
con', 'meatballs', 'olive oil', 'salmon', 'cauliflower', 'whole wheat pasta', 'cereals', 'chili'}
```

```
itemset = set(items)
encoded_vals = []
for index, row in data.iterrows():
    rowset = set(row)
    labels = {}
    uncommons = list(itemset - rowset)
    commons = list(itemset.intersection(rowset))
    for uc in uncommons:
        labels[uc] = 0
    for com in commons:
        labels[com] = 1
    encoded_vals.append(labels)
encoded_vals[0]
ohe_df = pd.DataFrame(encoded_vals)
```

```
ohe_df = ohe_df.drop(ohe_df.columns[[2]], axis = 1)
ohe_df.head()
```

	parmesan cheese	mashed potato	milk	grated cheese	whole wheat rice	nonfat milk	tomato sauce	ketchup	french fries	rice	...	frozen smoothie	vegetables mix	yams	low fat yogurt	salad	olive oil	cottage cheese	salmon	min w
0	0	0	0	0	0	0	0	0	0	0	...	1	1	1	1	1	1	1	1	
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
4	0	0	1	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	

Association Rules and Final pairs

```
freq_items = apriori(ohe_df, min_support=0.02, use_colnames=True, verbose=1)
freq_items.head(7)
```

```
Processing 897 combinations | Sampling itemset size 32
```

	support	itemsets
0	0.129583	(milk)
1	0.052393	(grated cheese)
2	0.058526	(whole wheat rice)
3	0.170911	(french fries)
4	0.026530	(pepper)
5	0.043061	(fresh bread)
6	0.062525	(turkey)

```
rules = association_rules(freq_items, metric="confidence", min_threshold=0.01)
rules.head()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(french fries)	(milk)	0.170911	0.129583	0.023730	0.138846	1.071482	0.001583	1.010756
1	(milk)	(french fries)	0.129583	0.170911	0.023730	0.183128	1.071482	0.001583	1.014956
2	(milk)	(frozen vegetables)	0.129583	0.095321	0.023597	0.182099	1.910382	0.011245	1.106099
3	(frozen vegetables)	(milk)	0.095321	0.129583	0.023597	0.247552	1.910382	0.011245	1.156781
4	(milk)	(chocolate)	0.129583	0.163845	0.032129	0.247942	1.513276	0.010898	1.111823