Market Basket Insights

Phase 5 project submission

Project title: market Basket Insights

Phase 5: Submission document

**Topic: In this section we will document the complete project and prepare it for submission.**

**Introduction**

Market Basket Analysis is one of the fundamental techniques used by large retailers to uncover the association between items. In other words, it allows retailers to identify the relationship between items which are more frequently bought together.

Here is the list of tools and software commonly used in the process:

1. Programming language:

Python is the most popular language for machine learning due to its

Extensive libraries and frameworks. you can use libraries like numpy pandas scikit, learn and more.

1. Integrated development environment:

Choose an IDE for coding and running machine learning experiments. Some popular options include Jupiter notebook, google collab or traditional IDEs like PyCharm

1. Machine learning libraries: you will need various machine learning librares, including :

Scikit-learn for building and evaluating machine learning models.

Tensorflow or pyTourch for deep learning if needed:

XGBoost, lightGBM, CatBoost for gradient booster models.

1. Data visualization tools:

Tools like mac plot lib, c born or plotting are essential for data exploration and visulization

1. Data preprossing:

Libraries like pandas helps with data cleaning manipulation and preprossing

1. Data collection and storage:

Depending on your data source, you might need web scrapping tools or data base for data storage

1. Version control:

Version system controls like git are vulnerable for tracking changes in your code and collaborating with others.

1. Hyper parameter tunning:
2. Tools like Grip SearchCV or randomized search CV from scikt learn can help with hyper parameter tool

{

"cells": [

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"# Market Basket Analysis 101 with Real Example"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"### Association rules, Lift, Confidence, Support"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## Table of Content\n",

"1. Introduction\n",

"2. What is Market Basket Analysis?\n",

"3. Import Library / Data\n",

"4. Data pre-processing\n",

"5. Association Rules in Python\n",

"6. Deriving Association Rules\n",

"7. Findings / Conclusion"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## 1.0 Introduction"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Another exciting topic in marketing analytics is Market Basket Analysis. This is the topic of this publication. At the beginning of this post I will be introducing some key terms and metrics aimed at giving a sense of what â€œassociationâ€ in a rule means and some ways to quantify the strength of this association. Then I will show how to generate these rules from the dataset â€˜Online Retailâ€™ using the Apriori Algorithm.\n",

"\n",

"For this post the dataset Online Retail from the statistic platform â€œKaggleâ€ was used. You can download it from my â€œGitHub Repositoryâ€."

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"## 2.0 What is Market Basket Analysis?"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"Market Basket Analysis is a analysis technique which identifies the strength of association between pairs of products purchased together and identify patterns of co-occurrence.\n",

"\n",

"Market Basket Analysis creates If-Then scenario rules (association rules), for example, if item A is purchased then item B is likely to be purchased. The rules are probabilistic in nature or, in other words, they are derived from the frequencies of co-occurrence in the observations. Frequency is the proportion of baskets that contain the items of interest. The rules can be used in pricing strategies, product placement, and various types of cross-selling strategies."

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"How association rules work\n",

"\n",

"Association rule mining, at a basic level, involves the use of machine learning models to analyze data for patterns, or co-occurrences, in a database. It identifies frequent if-then associations, which themselves are the association rules.\n",

"\n",

"An association rule has two parts: an antecedent (if) and a consequent (then). An antecedent is an item found within the data. A consequent is an item found in combination with the antecedent.\n",

"\n",

"Association rules are created by searching data for frequent if-then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the data. Confidence indicates the number of times the if-then statements are found true. A third metric, called lift, can be used to compare confidence with expected confidence, or how many times an if-then statement is expected to be found true.\n",

"\n",

"Association rules are calculated from itemsets, which are made up of two or more items. If rules are built from analyzing all the possible itemsets, there could be so many rules that the rules hold little meaning. With that, association rules are typically created from rules well-represented in data.\n",

"\n",

"More about association rule can be found on https://michael-fuchs-python.netlify.app/2020/09/15/marketing-market-basket-analysis/"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"# 3.0 Dive into code - Import Libraries / Data"

]

},

{

"cell\_type": "code",

"execution\_count": 9,

"metadata": {},

"outputs": [],

"source": [

"import gc\n",

"import os\n",

"import time\n",

"import numpy as np\n",

"import pandas as pd\n",

"import matplotlib.pyplot as plt\n",

"import seaborn as sns\n",

"import datetime as dt\n",

"import mlxtend.preprocessing\n",

"import mlxtend.frequent\_patterns"

]

},

{

"cell\_type": "code",

"execution\_count": 14,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>InvoiceNo</th>\n",

" <th>StockCode</th>\n",

" <th>Description</th>\n",

" <th>Quantity</th>\n",

" <th>InvoiceDate</th>\n",

" <th>UnitPrice</th>\n",

" <th>CustomerID</th>\n",

" <th>Country</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>536365</td>\n",

" <td>85123A</td>\n",

" <td>WHITE HANGING HEART T-LIGHT HOLDER</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>2.55</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>536365</td>\n",

" <td>71053</td>\n",

" <td>WHITE METAL LANTERN</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>536365</td>\n",

" <td>84406B</td>\n",

" <td>CREAM CUPID HEARTS COAT HANGER</td>\n",

" <td>8</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>2.75</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>536365</td>\n",

" <td>84029G</td>\n",

" <td>KNITTED UNION FLAG HOT WATER BOTTLE</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>536365</td>\n",

" <td>84029E</td>\n",

" <td>RED WOOLLY HOTTIE WHITE HEART.</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" </tr>\n",

" <tr>\n",

" <th>...</th>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541904</th>\n",

" <td>581587</td>\n",

" <td>22613</td>\n",

" <td>PACK OF 20 SPACEBOY NAPKINS</td>\n",

" <td>12</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>0.85</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541905</th>\n",

" <td>581587</td>\n",

" <td>22899</td>\n",

" <td>CHILDREN'S APRON DOLLY GIRL</td>\n",

" <td>6</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>2.10</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541906</th>\n",

" <td>581587</td>\n",

" <td>23254</td>\n",

" <td>CHILDRENS CUTLERY DOLLY GIRL</td>\n",

" <td>4</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.15</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541907</th>\n",

" <td>581587</td>\n",

" <td>23255</td>\n",

" <td>CHILDRENS CUTLERY CIRCUS PARADE</td>\n",

" <td>4</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.15</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541908</th>\n",

" <td>581587</td>\n",

" <td>22138</td>\n",

" <td>BAKING SET 9 PIECE RETROSPOT</td>\n",

" <td>3</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.95</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>541909 rows Ã— 8 columns</p>\n",

"</div>"

],

"text/plain": [

" InvoiceNo StockCode Description Quantity \\\n",

"0 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6 \n",

"1 536365 71053 WHITE METAL LANTERN 6 \n",

"2 536365 84406B CREAM CUPID HEARTS COAT HANGER 8 \n",

"3 536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE 6 \n",

"4 536365 84029E RED WOOLLY HOTTIE WHITE HEART. 6 \n",

"... ... ... ... ... \n",

"541904 581587 22613 PACK OF 20 SPACEBOY NAPKINS 12 \n",

"541905 581587 22899 CHILDREN'S APRON DOLLY GIRL 6 \n",

"541906 581587 23254 CHILDRENS CUTLERY DOLLY GIRL 4 \n",

"541907 581587 23255 CHILDRENS CUTLERY CIRCUS PARADE 4 \n",

"541908 581587 22138 BAKING SET 9 PIECE RETROSPOT 3 \n",

"\n",

" InvoiceDate UnitPrice CustomerID Country \n",

"0 2010-12-01 08:26:00 2.55 17850.0 United Kingdom \n",

"1 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"2 2010-12-01 08:26:00 2.75 17850.0 United Kingdom \n",

"3 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"4 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"... ... ... ... ... \n",

"541904 2011-12-09 12:50:00 0.85 12680.0 France \n",

"541905 2011-12-09 12:50:00 2.10 12680.0 France \n",

"541906 2011-12-09 12:50:00 4.15 12680.0 France \n",

"541907 2011-12-09 12:50:00 4.15 12680.0 France \n",

"541908 2011-12-09 12:50:00 4.95 12680.0 France \n",

"\n",

"[541909 rows x 8 columns]"

]

},

"execution\_count": 14,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df = pd.read\_excel(\"./data/Retail.xlsx\", sheet\_name=\"Online Retail\")\n",

"df"

]

},

{

"cell\_type": "code",

"execution\_count": 15,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(541909, 8)"

]

},

"execution\_count": 15,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df.shape"

]

},

{

"cell\_type": "code",

"execution\_count": 16,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>Quantity</th>\n",

" <th>UnitPrice</th>\n",

" <th>CustomerID</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>count</th>\n",

" <td>541909.000000</td>\n",

" <td>541909.000000</td>\n",

" <td>406829.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>mean</th>\n",

" <td>9.552250</td>\n",

" <td>4.611114</td>\n",

" <td>15287.690570</td>\n",

" </tr>\n",

" <tr>\n",

" <th>std</th>\n",

" <td>218.081158</td>\n",

" <td>96.759853</td>\n",

" <td>1713.600303</td>\n",

" </tr>\n",

" <tr>\n",

" <th>min</th>\n",

" <td>-80995.000000</td>\n",

" <td>-11062.060000</td>\n",

" <td>12346.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25%</th>\n",

" <td>1.000000</td>\n",

" <td>1.250000</td>\n",

" <td>13953.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>50%</th>\n",

" <td>3.000000</td>\n",

" <td>2.080000</td>\n",

" <td>15152.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>75%</th>\n",

" <td>10.000000</td>\n",

" <td>4.130000</td>\n",

" <td>16791.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>max</th>\n",

" <td>80995.000000</td>\n",

" <td>38970.000000</td>\n",

" <td>18287.000000</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" Quantity UnitPrice CustomerID\n",

"count 541909.000000 541909.000000 406829.000000\n",

"mean 9.552250 4.611114 15287.690570\n",

"std 218.081158 96.759853 1713.600303\n",

"min -80995.000000 -11062.060000 12346.000000\n",

"25% 1.000000 1.250000 13953.000000\n",

"50% 3.000000 2.080000 15152.000000\n",

"75% 10.000000 4.130000 16791.000000\n",

"max 80995.000000 38970.000000 18287.000000"

]

},

"execution\_count": 16,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df.describe()"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"# 4.0 Preprocess the dataset"

]

},

{

"cell\_type": "code",

"execution\_count": 17,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>InvoiceNo</th>\n",

" <th>StockCode</th>\n",

" <th>Description</th>\n",

" <th>Quantity</th>\n",

" <th>InvoiceDate</th>\n",

" <th>UnitPrice</th>\n",

" <th>CustomerID</th>\n",

" <th>Country</th>\n",

" <th>Is\_C\_Present</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>536365</td>\n",

" <td>85123A</td>\n",

" <td>WHITE HANGING HEART T-LIGHT HOLDER</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>2.55</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>536365</td>\n",

" <td>71053</td>\n",

" <td>WHITE METAL LANTERN</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>536365</td>\n",

" <td>84406B</td>\n",

" <td>CREAM CUPID HEARTS COAT HANGER</td>\n",

" <td>8</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>2.75</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>536365</td>\n",

" <td>84029G</td>\n",

" <td>KNITTED UNION FLAG HOT WATER BOTTLE</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>536365</td>\n",

" <td>84029E</td>\n",

" <td>RED WOOLLY HOTTIE WHITE HEART.</td>\n",

" <td>6</td>\n",

" <td>2010-12-01 08:26:00</td>\n",

" <td>3.39</td>\n",

" <td>17850.0</td>\n",

" <td>United Kingdom</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>...</th>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541904</th>\n",

" <td>581587</td>\n",

" <td>22613</td>\n",

" <td>PACK OF 20 SPACEBOY NAPKINS</td>\n",

" <td>12</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>0.85</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541905</th>\n",

" <td>581587</td>\n",

" <td>22899</td>\n",

" <td>CHILDREN'S APRON DOLLY GIRL</td>\n",

" <td>6</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>2.10</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541906</th>\n",

" <td>581587</td>\n",

" <td>23254</td>\n",

" <td>CHILDRENS CUTLERY DOLLY GIRL</td>\n",

" <td>4</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.15</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541907</th>\n",

" <td>581587</td>\n",

" <td>23255</td>\n",

" <td>CHILDRENS CUTLERY CIRCUS PARADE</td>\n",

" <td>4</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.15</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541908</th>\n",

" <td>581587</td>\n",

" <td>22138</td>\n",

" <td>BAKING SET 9 PIECE RETROSPOT</td>\n",

" <td>3</td>\n",

" <td>2011-12-09 12:50:00</td>\n",

" <td>4.95</td>\n",

" <td>12680.0</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>541909 rows Ã— 9 columns</p>\n",

"</div>"

],

"text/plain": [

" InvoiceNo StockCode Description Quantity \\\n",

"0 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6 \n",

"1 536365 71053 WHITE METAL LANTERN 6 \n",

"2 536365 84406B CREAM CUPID HEARTS COAT HANGER 8 \n",

"3 536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE 6 \n",

"4 536365 84029E RED WOOLLY HOTTIE WHITE HEART. 6 \n",

"... ... ... ... ... \n",

"541904 581587 22613 PACK OF 20 SPACEBOY NAPKINS 12 \n",

"541905 581587 22899 CHILDREN'S APRON DOLLY GIRL 6 \n",

"541906 581587 23254 CHILDRENS CUTLERY DOLLY GIRL 4 \n",

"541907 581587 23255 CHILDRENS CUTLERY CIRCUS PARADE 4 \n",

"541908 581587 22138 BAKING SET 9 PIECE RETROSPOT 3 \n",

"\n",

" InvoiceDate UnitPrice CustomerID Country \\\n",

"0 2010-12-01 08:26:00 2.55 17850.0 United Kingdom \n",

"1 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"2 2010-12-01 08:26:00 2.75 17850.0 United Kingdom \n",

"3 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"4 2010-12-01 08:26:00 3.39 17850.0 United Kingdom \n",

"... ... ... ... ... \n",

"541904 2011-12-09 12:50:00 0.85 12680.0 France \n",

"541905 2011-12-09 12:50:00 2.10 12680.0 France \n",

"541906 2011-12-09 12:50:00 4.15 12680.0 France \n",

"541907 2011-12-09 12:50:00 4.15 12680.0 France \n",

"541908 2011-12-09 12:50:00 4.95 12680.0 France \n",

"\n",

" Is\_C\_Present \n",

"0 0 \n",

"1 0 \n",

"2 0 \n",

"3 0 \n",

"4 0 \n",

"... ... \n",

"541904 0 \n",

"541905 0 \n",

"541906 0 \n",

"541907 0 \n",

"541908 0 \n",

"\n",

"[541909 rows x 9 columns]"

]

},

"execution\_count": 17,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"'''\n",

"Create an indicator column stipulating whether the invoice number begins with 'C'\n",

"'''\n",

"\n",

"df['Is\_C\_Present'] = (\n",

" df['InvoiceNo']\n",

" .astype(str)\n",

" .apply(lambda x: 1 if x.find('C') != -1 else 0))\n",

"\n",

"df"

]

},

{

"cell\_type": "code",

"execution\_count": 10,

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<bound method IndexOpsMixin.value\_counts of 0 0\n",

"1 0\n",

"2 0\n",

"3 0\n",

"4 0\n",

" ..\n",

"541904 0\n",

"541905 0\n",

"541906 0\n",

"541907 0\n",

"541908 0\n",

"Name: Is\_C\_Present, Length: 541909, dtype: int64>"

]

},

"execution\_count": 10,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df.Is\_C\_Present.value\_counts"

]

},

{

"cell\_type": "code",

"execution\_count": 18,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>InvoiceNo</th>\n",

" <th>Description</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>536365</td>\n",

" <td>WHITE HANGING HEART T-LIGHT HOLDER</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>536365</td>\n",

" <td>WHITE METAL LANTERN</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>536365</td>\n",

" <td>CREAM CUPID HEARTS COAT HANGER</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>536365</td>\n",

" <td>KNITTED UNION FLAG HOT WATER BOTTLE</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>536365</td>\n",

" <td>RED WOOLLY HOTTIE WHITE HEART.</td>\n",

" </tr>\n",

" <tr>\n",

" <th>...</th>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541904</th>\n",

" <td>581587</td>\n",

" <td>PACK OF 20 SPACEBOY NAPKINS</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541905</th>\n",

" <td>581587</td>\n",

" <td>CHILDREN'S APRON DOLLY GIRL</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541906</th>\n",

" <td>581587</td>\n",

" <td>CHILDRENS CUTLERY DOLLY GIRL</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541907</th>\n",

" <td>581587</td>\n",

" <td>CHILDRENS CUTLERY CIRCUS PARADE</td>\n",

" </tr>\n",

" <tr>\n",

" <th>541908</th>\n",

" <td>581587</td>\n",

" <td>BAKING SET 9 PIECE RETROSPOT</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>530693 rows Ã— 2 columns</p>\n",

"</div>"

],

"text/plain": [

" InvoiceNo Description\n",

"0 536365 WHITE HANGING HEART T-LIGHT HOLDER\n",

"1 536365 WHITE METAL LANTERN\n",

"2 536365 CREAM CUPID HEARTS COAT HANGER\n",

"3 536365 KNITTED UNION FLAG HOT WATER BOTTLE\n",

"4 536365 RED WOOLLY HOTTIE WHITE HEART.\n",

"... ... ...\n",

"541904 581587 PACK OF 20 SPACEBOY NAPKINS\n",

"541905 581587 CHILDREN'S APRON DOLLY GIRL \n",

"541906 581587 CHILDRENS CUTLERY DOLLY GIRL \n",

"541907 581587 CHILDRENS CUTLERY CIRCUS PARADE\n",

"541908 581587 BAKING SET 9 PIECE RETROSPOT \n",

"\n",

"[530693 rows x 2 columns]"

]

},

"execution\_count": 18,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"'''\n",

"Filter out all transactions having either zero or a negative number of items.\n",

"Remove all invoice numbers starting with 'C' (using columns 'Is\_C\_Present').\n",

"Subset the dataframe down to 'InvoiceNo' and 'Descritpion'.\n",

"Drop all rows with at least one missing value.\n",

"'''\n",

"\n",

"\n",

"df\_clean = (\n",

" df\n",

" # filter out non-positive quantity values\n",

" .loc[df[\"Quantity\"] > 0]\n",

" # remove InvoiceNos starting with C\n",

" .loc[df['Is\_C\_Present'] != 1]\n",

" # column filtering\n",

" .loc[:, [\"InvoiceNo\", \"Description\"]]\n",

" # dropping all rows with at least one missing value\n",

" .dropna()\n",

")\n",

"\n",

"df\_clean"

]

},

{

"cell\_type": "code",

"execution\_count": 19,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[['HAND WARMER UNION JACK', 'HAND WARMER RED POLKA DOT'], ['ASSORTED COLOUR BIRD ORNAMENT', \"POPPY'S PLAYHOUSE BEDROOM \", \"POPPY'S PLAYHOUSE KITCHEN\", 'FELTCRAFT PRINCESS CHARLOTTE DOLL', 'IVORY KNITTED MUG COSY ', 'BOX OF 6 ASSORTED COLOUR TEASPOONS', 'BOX OF VINTAGE JIGSAW BLOCKS ', 'BOX OF VINTAGE ALPHABET BLOCKS', 'HOME BUILDING BLOCK WORD', 'LOVE BUILDING BLOCK WORD', 'RECIPE BOX WITH METAL HEART', 'DOORMAT NEW ENGLAND']]\n"

]

}

],

"source": [

"'''\n",

"Transform the data into a list of lists called invoice\_item\_list\n",

"\n",

"'''\n",

"\n",

"invoice\_item\_list = []\n",

"\n",

"for num in list(set(df\_clean.InvoiceNo.tolist())):\n",

" # filter data set down to one invoice number\n",

" tmp\_df = df\_clean.loc[df\_clean['InvoiceNo'] == num]\n",

" # extract item descriptions and convert to list\n",

" tmp\_items = tmp\_df.Description.tolist()\n",

" # append list invoice\_item\_list\n",

" invoice\_item\_list.append(tmp\_items)\n",

"\n",

"print(invoice\_item\_list[1:3])"

]

},

{

"cell\_type": "markdown",

"metadata": {},

"source": [

"To be able to run any models the data, currently in the list of lists form, needs to be encoded and recast as a dataframe. \n",

"\n",

"Outputted from the encoder is a multidimensional array, where each row is the length of the total number of unique items in the transaction dataset and the elements are Boolean variables, indicating whether that particular item is linked to the invoice number that row presents. \n",

"\n",

"With the data encoded, we can recast it as a dataframe where the rows are the invoice numbers and the columns are the unique items in the transaction dataset."

]

},

{

"cell\_type": "code",

"execution\_count": 20,

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>4 PURPLE FLOCK DINNER CANDLES</th>\n",

" <th>50'S CHRISTMAS GIFT BAG LARGE</th>\n",

" <th>DOLLY GIRL BEAKER</th>\n",

" <th>I LOVE LONDON MINI BACKPACK</th>\n",

" <th>I LOVE LONDON MINI RUCKSACK</th>\n",

" <th>NINE DRAWER OFFICE TIDY</th>\n",

" <th>OVAL WALL MIRROR DIAMANTE</th>\n",

" <th>RED SPOT GIFT BAG LARGE</th>\n",

" <th>SET 2 TEA TOWELS I LOVE LONDON</th>\n",

" <th>SPACEBOY BABY GIFT SET</th>\n",

" <th>...</th>\n",

" <th>returned</th>\n",

" <th>taig adjust</th>\n",

" <th>test</th>\n",

" <th>to push order througha s stock was</th>\n",

" <th>website fixed</th>\n",

" <th>wrongly coded 20713</th>\n",

" <th>wrongly coded 23343</th>\n",

" <th>wrongly marked</th>\n",

" <th>wrongly marked 23343</th>\n",

" <th>wrongly sold (22719) barcode</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>...</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>...</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>...</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

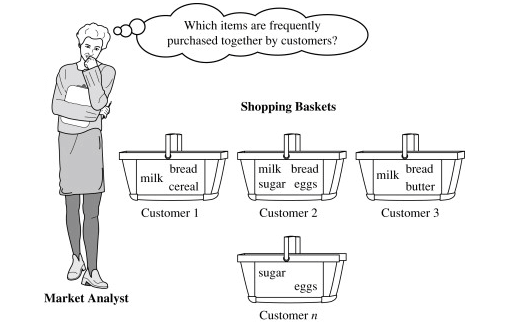
" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",

" <td>False</td>\n",



**Market Basket Insights**

**Phase 3 project submission**

**Project title: market basket insights**

**Phase 3: Development part 1**

## Table of contents

## What Is Association Rule for Market Basket Analysis?

## Algorithms Used in Market Basket Analysis

## Implementing Market Basket Analysis in Python

* **conclusion**

## What Is Association Rule for Market Basket Analysis?

Let I = {I1, I2,…, I m} be an item set. These item sets are called antecedents. Let D, the data, be a set of database transactions where each transaction T is a nonempty item set such that **T ⊆ I**. Each transaction is associated with an identifier called a TID(or T id). Let A be a set of items (item set). T is the Transaction that is said to contain A if **A ⊆ T**. An **Association Rule** is an implication of form **A ⇒ B**, where **A ⊂ I, B ⊂ I**, and **A ∩B = φ**.

The rule **A ⇒ B** holds in the data set(transactions) D with supports, where ‘s’ is the percentage of transactions in D that contain **A ∪ B** (i.e., the union of set A and set B, or both A and B). This is taken as the probability, **P(A ∪ B)**. Rule **A ⇒ B** has confidence **c** in the transaction set D, where c is the percentage of transactions in D containing **A** that also contains **B**. This is taken to be the conditional probability, like P (B|A). That is,

* ***support(A⇒ B) =P(A ∪  B)***
* ***confidence(A⇒ B) =P(B|A)***

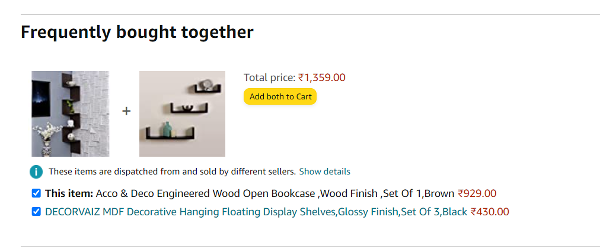
Rules that satisfy both a minimum support threshold (called min sup) and a minimum confidence threshold (called min conf) are called “***Strong”***.

* ***Confidence(A⇒ B) = P(B|A) =***
* ***support(A ∪ B) /support(A) =***
* ***support count(A ∪ B) / support count(A)***

Generally, Association Rule Mining can be viewed in a two-step process:

1. Find all frequent item sets: *By definition, each of these items e its will occur at least as  
   frequently as a pre-established minimum support count, min sup*.
2. Generate Association Rules from the frequent item sets: *By definition, these  
   rules must satisfy minimum support and minimum confidence.*

*Example:*



## Algorithms Used in Market Basket Analysis

There are multiple data mining techniques and algorithms used in Market Basket Analysis. One of the important objectives is “to predict the probability of items that are being bought together by customers.”

* **A priori Algorithm**
* **AIS**
* **SETM Algorithm**
* **FP Growth**

### 1. Apriority Algorithm

A priori Algorithm is a widely-used and well-known Association Rule algorithm and is a popular algorithm used in market basket analysis. It is also considered accurate and overtop AIS and SETM algorithms. It helps to find frequent item sets in transactions and identifies association rules between these items. The limitation of the A priori Algorithm is frequent item set generation. It needs to scan the database many times, leading to increased time and reduced performance as a computationally costly step because of a large dataset. It uses the concepts of Confidence and Support.

### 2. AIS Algorithm

The AIS algorithm creates multiple passes on the entire database or transactional data. During every pass, it scans all transactions. As you can see, in the first pass, it counts the support of separate items and determines then which of them are frequent in the database. Huge item sets of every pass are enlarged to generate candidate item sets. After each scanning of a transaction, the common item sets between the item sets of the previous pass and the items of this transaction are determined. This algorithm was the first published algorithm which is developed to generate all large item sets in a transactional database. It focused on the enhancement of databases with the necessary performance to process decision support. This technique is bounded to only one item in the consequent.

* **Advantage**: The AIS algorithm was used to find whether there was an association between items or not.
* **Disadvantage**: The main disadvantage of the AIS algorithm is that it generates too many candidates set that turn out to be small. As well as the data structure is to be maintained.

### 3. SETM Algorithm

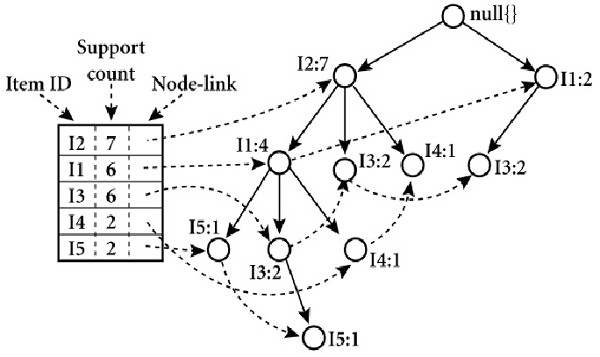
This Algorithm is quite similar to the AIS algorithm. The **SETM** algorithm creates collective passes over the database. As you can see, in the first pass, it counts the support of single items and then determines which of them are frequent in the database. Then, it also generates the candidate item sets by enlarging large item sets of the previous pass. In addition to this, the SETM algorithm recalls the TIDs (transaction ids) of the generating transactions with the candidate item sets.

* **Advantage**: While generating candidate, the SETM algorithm arranges candidate item sets together with the TID (transaction Id) in a sequential manner.
* **Disadvantage**: For every item set, there is an association with Tid; hence it requires more space to store a huge number of TIDs.

### 4. FP Growth

**FP Growth** is known as Frequent Pattern Growth Algorithm. FP growth algorithm is a concept of representing the data in the form of an FP tree or Frequent Pattern. Hence FP Growth is a method of Mining Frequent Item sets. This algorithm is advancement to the **Apriority Algorithm**. There is no need for candidate generation to generate a frequent pattern. This frequent pattern tree structure maintains the association between the item sets.

**A Frequent Pattern Tree** is a tree structure that is made with the earlier item sets of the data. The main purpose of the FP tree is to mine the most frequent patterns. Every node of the FP tree represents an item of that item set. The root node represents the null value, whereas the lower nodes represent the item sets of the data. The association of these nodes with the lower nodes, that is, between item sets, is maintained while creating the tree.



## Implementing Market Basket Analysis in Python

#### The Method

Here are the steps involved in using the a priori algorithm to implement MBA:

1. First, define the minimum support and confidence for the association rule.
2. Find out all the subsets in the transactions with higher support(sup) than the minimum support.
3. Find all the rules for these subsets with higher confidence than minimum confidence.
4. Sort these association rules in decreasing order.
5. Analyze the rules along with their confidence and support.

#### The Dataset

In this implementation, we have to use the Store Data dataset that is publicly available on Kaggle. This dataset contains a total of 7501 transaction records, where every record consists of a list of items sold in just one transaction.

#### Implementing Market Basket Analysis Using the A priori Method

The A priori algorithm is frequently used by data scientists. We are required to import the necessary libraries. Python provides the **pyori** as an API that is required to be imported to run the A priori Algorithm.

from IPython.core.display import HTML

HTML("""

<style>

.output\_png {

display: table-cell;

text-align: center;

vertical-align: middle;

horizontal-align: middle;

}

h1,h2 {

text-align: center;

background-color: pink;

padding: 20px;

margin: 0;

color: black;

font-family: ariel;

border-radius: 80px

}

h3 {

text-align: center;

border-style: solid;

border-width: 3px;

padding: 12px;

margin: 0;

color: black;

font-family: ariel;

border-radius: 80px;

border-color: gold;

}

body, p {

font-family: ariel;

font-size: 15px;

color: charcoal;

}

div {

font-size: 14px;

margin: 0;

}

h4 {

padding: 0px;

margin: 0;

font-family: ariel;

color: purple;

}

</style>

""")

## **Flow of Execution:**

1. Loading Necessary Packages
2. Loading dataset
3. Data Pre-Processing
4. Performing EDA
5. Apriori Implementation
6. Result Customization

## **Step - 1 :** Loading Necessary Packages

!pip install apyori ## Installing apriori library

Collecting apyori

Downloading apyori-1.1.2.tar.gz (8.6 kB)

Preparing metadata (setup.py) ... - done

Building wheels for collected packages: apyori

Building wheel for apyori (setup.py) ... - \ done

Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5974 sha256=c03b4c07b988bef21b16adf3c29d4091a48e0ba3cc7a41e46b557ef451395440

Stored in directory: /root/.cache/pip/wheels/cb/f6/e1/57973c631d27efd1a2f375bd6a83b2a616c4021f24aab84080

Successfully built apyori

Installing collected packages: apyori

Successfully installed apyori-1.1.2

WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv

import numpy as np # linear algebra

import pandas as pd # Data pre-processing

import seaborn as sns # Required for plotting

import matplotlib.pyplot as plt # Required for plotting

## **Step - 2 :** Loading dataset

df = pd.read\_csv("../input/groceries-dataset/Groceries\_dataset.csv") ## Loading dataset

df.head()

|  | **Member\_number** | **Date** | **itemDescription** |
| --- | --- | --- | --- |
| **0** | 1808 | 21-07-2015 | tropical fruit |
| **1** | 2552 | 05-01-2015 | whole milk |
| **2** | 2300 | 19-09-2015 | pip fruit |
| **3** | 1187 | 12-12-2015 | other vegetables |
| **4** | 3037 | 01-02-2015 | whole milk |

df.info() # Checking data type information for validation purposes

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 38765 entries, 0 to 38764

Data columns (total 3 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Member\_number 38765 non-null int64

1 Date 38765 non-null object

2 itemDescription 38765 non-null object

dtypes: int64(1), object(2)

memory usage: 908.7+ KB

Interpretation: - No Null values should be present

df.isnull().sum().sort\_values(ascending=False) ## Checking availability of NULL values

Member\_number 0

Date 0

itemDescription 0

dtype: int64

Note - No NULLs present

## **Step - 3 :** Data Pre-Processing

df['Date'] = pd.to\_datetime(df['Date']) ## Type-Conversion from Object to Dateime

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 38765 entries, 0 to 38764

Data columns (total 3 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Member\_number 38765 non-null int64

1 Date 38765 non-null datetime64[ns]

2 itemDescription 38765 non-null object

dtypes: datetime64[ns](1), int64(1), object(1)

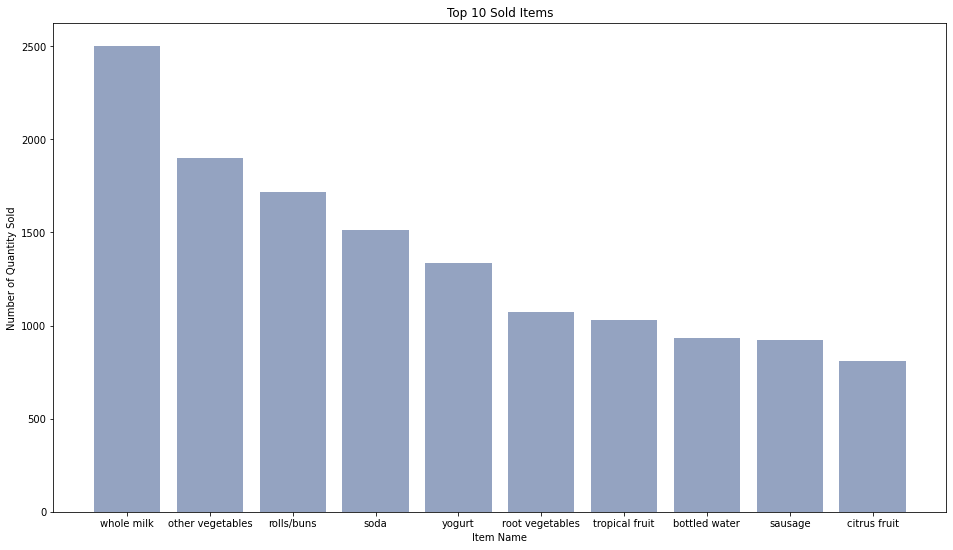
memory usage: 908.7+ KB

df.head() ## Schema check

|  | **Member\_number** | **Date** | **itemDescription** |
| --- | --- | --- | --- |
| **0** | 1808 | 2015-07-21 | tropical fruit |
| **1** | 2552 | 2015-05-01 | whole milk |
| **2** | 2300 | 2015-09-19 | pip fruit |
| **3** | 1187 | 2015-12-12 | other vegetables |
| **4** | 3037 | 2015-01-02 | whole milk |

## **Step - 4 :** Performing EDA

### **Step - 4.1 :** Top 10 Sold Items



## Creating distribution of Item Sold

Item\_distr = df.groupby(by = "itemDescription").size().reset\_index(name='Frequency').sort\_values(by = 'Frequency',ascending=False).head(10)

## Declaring variables

bars = Item\_distr["itemDescription"]

height = Item\_distr["Frequency"]

x\_pos = np.arange(len(bars))

## Defining Figure Size

plt.figure(figsize=(16,9))

# Create bars

plt.bar(x\_pos, height, color=(0.3, 0.4, 0.6, 0.6))

# Add title and axis names

plt.title("Top 10 Sold Items")

plt.xlabel("Item Name")

plt.ylabel("Number of Quantity Sold")

# Create names on the x-axis

plt.xticks(x\_pos, bars)

# Show graph

plt.show()

### **Step - 4.2 :** Month-Year Sales

df\_date=df.set\_index(['Date']) ## Setting date as index for plotting purpose

df\_date

|  | **Member\_number** | **item Description** |
| --- | --- | --- |
| **Date** |  |  |
| **2015-07-21** | 1808 | tropical fruit |
| **2015-05-01** | 2552 | whole milk |
| **2015-09-19** | 2300 | pip fruit |
| **2015-12-12** | 1187 | other vegetables |
| **2015-01-02** | 3037 | whole milk |
| **...** | ... | ... |
| **2014-08-10** | 4471 | sliced cheese |
| **2014-02-23** | 2022 | candy |
| **2014-04-16** | 1097 | cake bar |
| **2014-03-12** | 1510 | fruit/vegetable juice |
| **2014-12-26** | 1521 | cat food |

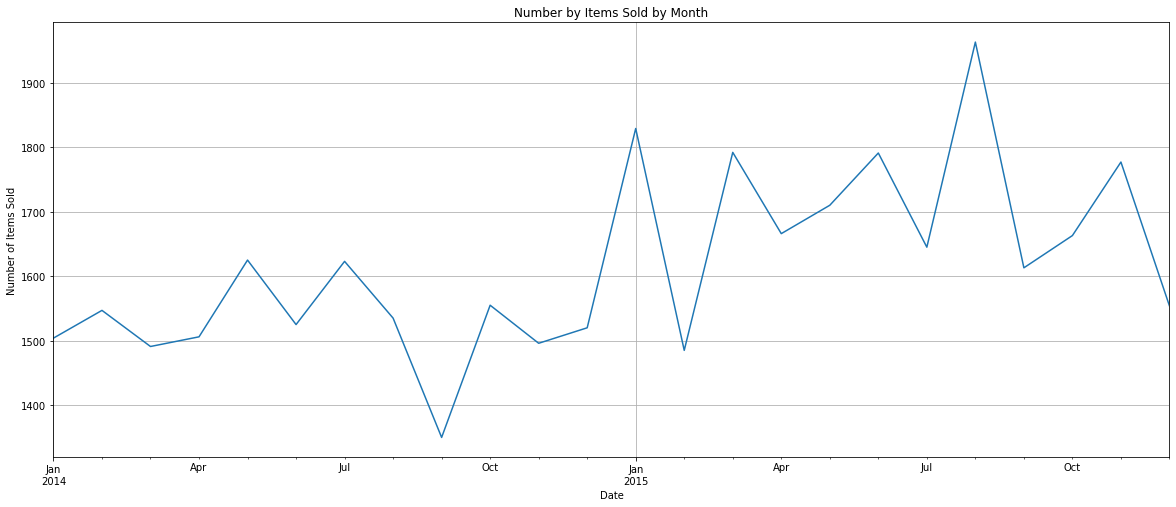
38765 rows × 2 columns

df\_date.resample("M")['itemDescription'].count().plot(figsize = (20,8), grid = True, title = "Number by Items Sold by Month").set(xlabel = "Date", ylabel = "Number of Items Sold")

[Text(0.5, 0, 'Date'), Text(0, 0.5, 'Number of Items Sold')]

## **Step - 5 :** Apriori Implementation

Apriori is an algorithm for frequent itemset mining and association rule learning over relational databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent itemsets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.



### **Step - 5.1 :** Data Preparation

cust\_level = df[["Member\_number", "itemDescription"]].sort\_values(by = "Member\_number", ascending = False) ## Selecting only required variables for modelling

cust\_level['itemDescription'] = cust\_level['itemDescription'].str.strip() # Removing white spaces if any

cust\_level

|  | **Member number** | **Item Description** |
| --- | --- | --- |
| **3578** | 5000 | soda |
| **34885** | 5000 | semi-finished bread |
| **11728** | 5000 | fruit/vegetable juice |
| **9340** | 5000 | bottled beer |
| **19727** | 5000 | root vegetables |
| **...** | ... | ... |
| **13331** | 1000 | whole milk |
| **17778** | 1000 | pickled vegetables |
| **6388** | 1000 | sausage |
| **20992** | 1000 | semi-finished bread |
| **8395** | 1000 | whole milk |

38765 rows × 2 columns

### **Step - 5.2 :** Create Transaction list

transactions = [a[1]['itemDescription'].tolist() for a in list(cust\_level.groupby(['Member\_number']))] ## Combing all the items in list format for each cutomer

### **Step - 5.3 :** Train Model

from apyori import apriori ## Importing apriori package

rules = apriori(transactions = transactions, min\_support = 0.002, min\_confidence = 0.05, min\_lift = 3, min\_length = 2, max\_length = 2) ## Model Creation

results = list(rules) ## Storing results in list format for better visualisation

results

[RelationRecord(items=frozenset({'UHT-milk', 'kitchen towels'}), support=0.002308876346844536, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'kitchen towels'}), items\_add=frozenset({'UHT-milk'}), confidence=0.30000000000000004, lift=3.821568627450981)]),

RelationRecord(items=frozenset({'potato products', 'beef'}), support=0.002565418163160595, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'potato products'}), items\_add=frozenset({'beef'}), confidence=0.4545454545454546, lift=3.8021849395239955)]),

RelationRecord(items=frozenset({'canned fruit', 'coffee'}), support=0.002308876346844536, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'canned fruit'}), items\_add=frozenset({'coffee'}), confidence=0.4285714285714286, lift=3.7289540816326534)]),

RelationRecord(items=frozenset({'domestic eggs', 'meat spreads'}), support=0.0035915854284248334, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'meat spreads'}), items\_add=frozenset({'domestic eggs'}), confidence=0.4, lift=3.0042389210019267)]),

RelationRecord(items=frozenset({'flour', 'mayonnaise'}), support=0.002308876346844536, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'flour'}), items\_add=frozenset({'mayonnaise'}), confidence=0.06338028169014086, lift=3.3385991625428253), OrderedStatistic(items\_base=frozenset({'mayonnaise'}), items\_add=frozenset({'flour'}), confidence=0.12162162162162163, lift=3.338599162542825)]),

RelationRecord(items=frozenset({'napkins', 'rice'}), support=0.0030785017957927143, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'rice'}), items\_add=frozenset({'napkins'}), confidence=0.2448979591836735, lift=3.011395094315329)]),

RelationRecord(items=frozenset({'sparkling wine', 'waffles'}), support=0.002565418163160595, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'sparkling wine'}), items\_add=frozenset({'waffles'}), confidence=0.21739130434782608, lift=3.1501535477614353)])]

## **Step - 6 :** Result Customization

## Creating user-defined function for arranging the results obtained from model into readable format

def inspect(results):

lhs = [tuple(result[2][0][0])[0] for result in results]

rhs = [tuple(result[2][0][1])[0] for result in results]

supports = [result[1] for result in results]

confidences = [result[2][0][2] for result in results]

lifts = [result[2][0][3] for result in results]

return list(zip(lhs, rhs, supports, confidences, lifts))

resultsinDataFrame = pd.DataFrame(inspect(results), columns = ['Left Hand Side', 'Right Hand Side', 'Support', 'Confidence', 'Lift'])

resultsinDataFrame.nlargest(n=10, columns="Lift") ## Showing best possible scenarios

|  | **Left Hand Side** | **Right Hand Side** | **Support** | **Confidence** | **Lift** |
| --- | --- | --- | --- | --- | --- |
| **0** | kitchen towels | UHT-milk | 0.002309 | 0.300000 | 3.821569 |
| **1** | potato products | beef | 0.002565 | 0.454545 | 3.802185 |
| **2** | canned fruit | coffee | 0.002309 | 0.428571 | 3.728954 |
| **4** | flour | mayonnaise | 0.002309 | 0.063380 | 3.338599 |
| **6** | sparkling wine | waffles | 0.002565 | 0.217391 | 3.150154 |
| **5** | rice | napkins | 0.003079 | 0.244898 | 3.011395 |
| **3** | meat spreads | domestic eggs | 0.003592 | 0.400000 | 3.004239 |

## Conclusion

In this tutorial, we discussed  Market Basket Analysis and learned the steps to implement it from scratch using Python. We then implemented Market Basket Analysis using Apriori Algorithm. We also looked into the various uses and advantages of this algorithm and learned that we could also use FP Growth and AIS algorithms to implement Market Basket Analysis.

**Key Takeaways**

* Market Basket Analysis is a business strategy used to design store layouts based on customers’ shopping behavior and purchase histories.
* This idea is also applicable to machine learning algorithms to teach machines to help businesses, especially in the e-commerce sector.
* In this article, we have gone through a step-by-step guide to implementing the apriori algorithm in Python and also looked into the math behind the association rules.

**Association Rules :**



**An example of Association Rules**

* Assume there are 100 customers
* 10 of them bought milk, 8 bought butter and 6 bought both of them.
* bought milk => bought butter
* support = P(Milk & Butter) = 6/100 = 0.06
* confidence = support/P(Butter) = 0.06/0.08 = 0.75
* lift = confidence/P(Milk) = 0.75/0.10 = 7.5

**Usage**:

1. How likely is one to buy bread if (s)he bought milk & eggs?
2. Product placement optimization
3. Product recomendations

# Advantages:

1. Fast
2. Works with relatively small amounts of data
3. Few if any feature engineering

# Feature engineering:

Association rule is the process of engineering data into a predictive feature that fits the requirements (and / or improves the performance) of a machine learning model.

# Apriory:

Apriory is the algorithm implementing association rule mining over structured data.

# How Does the Apriori Algorithm Work?

The key concept in the Apriori algorithm is that it assumes all subsets of a frequent itemset to be frequent. Similarly, for any infrequent itemset, all its supersets must also be infrequent.

Let us try and understand the working of an Apriori algorithm with the help of a very famous business scenario, market basket analysis.

Here is a dataset consisting of six transactions in an hour. Each transaction is a combination of 0s and 1s, where 0 represents the absence of an item and 1 represents the presence of it.

We can find multiple rules from this scenario. For example, in a transaction of wine, chips, and bread, if wine and chips are bought, then customers also buy bread.

{wine, chips} => {bread}

In order to select the interesting rules out of multiple possible rules from this small business scenario, we will be using the following measures:

* **Support**
* **Confidence**
* **Lift**
* **Conviction**

Remember I told y’all that we’ll get back to the three most popular criteria evaluating the quality or the strength of an association rule. There are **support, confidence**and**lift**:  
1. Support is the percentage of transactions containing a particular combination of items relative to the total number of transactions in the database. The support for the combination A and B would be,

P(AB) or P(A) for Individual A

2. Confidence measures how much the consequent (item) is dependent on the  
antecedent (item). In other words, confidence is the conditional probability of the consequent given the antecedent,

P(B|A)

where P(B|A) = P(AB)/P(A)

3. Lift (also called improvement or impact) is a measure to overcome the  
problems with support and confidence. Lift is said to measure the difference — measured in ratio — between the confidence of a rule and the expected confidence. Consider an association rule “if A then B.” The lift for the rule is defined as

P(B|A)/P(B) or P(AB)/[P(A)P(B)].

As shown in the formula, lift is symmetric in that the lift for “if A then B” is the same as the lift for “if B then A.”  
4. Each criterion has its advantages and disadvantages but in general we would like association rules that have high confidence, high support, and high lift.

As a summary,

Confidence = P(B|A)

Support = P(AB)

Lift = P(B|A)/P(B)

**Input Dataset :**

<https://github.com/stedy/Machine-Learning-with-R-datasets/blob/master/groceries.csv>

# **Let’s look at the code of market basket analysis using Python:**

**Import the Library**

#import packages  
#mlxtend for calculation of support,confidence and lift  
import sys  
import pandas as pd  
from mlxtend.preprocessing import TransactionEncoder  
from mlxtend.frequent\_patterns import apriori

**Read data and Display**

dataframe = pd.read\_csv("groceries.csv", sep='delimiter', header=None, engine='python')  
display(dataframe.head(20))  
print(dataframe.shape)

# Preprocessing on Data

* Here we need a data in form of list for Apriori Algorithm.

#converting the dataframe to a list  
data = dataset.values.tolist()  
data#convert the single string in each list to multiple strings separated by commas  
table = []for x in data:  
 new\_list = []  
 for y in x:  
 for z in y.split(','):  
 new\_list.append(z)  
 table.append(new\_list)  
   
table#encode the datasette = TransactionEncoder()  
te\_ary = te.fit(table).transform(table)  
df = pd.DataFrame(te\_ary, columns=te.columns\_)  
df

Using Apriori algorithm:

#generate frequent itemsets using Apriori algorithmfrequent\_itemsets = apriori(df,min\_support=0.2,use\_colnames=True)  
frequent\_itemsets#generating association rules  
from mlxtend.frequent\_patterns import association\_rules  
rules =  
association\_rules(frequent\_itemsets,metric='support',min\_threshold=0.1)ruleses\_specific = rules[['antecedents', 'consequents', 'support']]  
res\_specific.head(20)

Code:

**CODE :**import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd# Data Preprocessing  
dataset = pd.read\_csv(‘groceries.csv’)transactions = []for i in range(0, 9835):  
 transactions.append([str(dataset.values[i,j]) for j in range(0, 32)])# Training Apriori on the dataset  
from apyori import apriori  
rules = apriori(transactions, min\_support = 0.007, min\_confidence = 0.5, min\_lift = 3, min\_length = 2)# Visualising the results  
results = list(rules)dataset.head()  
dataset.shape  
**OUTPUT**:  
(9835, 32)  
**CODE :**for a in results:  
 print("------------------------------------------------------")  
 print(a)  
**OUTPUT :** RelationRecord(items=frozenset({'citrus fruit', 'other vegetables', 'root vegetables'}), support=0.010371123538383325, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'citrus fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5862068965517241, lift=3.0296084222733612)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'other vegetables', 'root vegetables'}), support=0.012302999491611592, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5845410628019324, lift=3.020999134344196)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'nan', 'citrus fruit', 'other vegetables', 'root vegetables'}), support=0.010269445856634469, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'nan', 'citrus fruit', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5838150289017341, lift=3.0172468782178425)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'nan', 'other vegetables', 'root vegetables'}), support=0.012201321809862735, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'nan', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5825242718446603, lift=3.010576044977527)])  
-------------------------------------------------------------------------------------------------------------  
RelationRecord(items=frozenset({'tropical fruit', 'whole milk', 'other vegetables', 'root vegetables'}), support=0.007015760040671073, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'tropical fruit', 'whole milk', 'root vegetables'}), items\_add=frozenset({'other vegetables'}), confidence=0.5847457627118644, lift=3.0220570553185424)])

# Conclusion

* More algorithms
* More parameter tuning
* More data complexities

This article is a walkthrough for a basic example of implementation of association rule learning for market basket analysis. We focused on theory and application of the most common algorithms.

From the output above, we see that the top associations are not surprising, with one flavor of an item being purchased with another flavor from the same item family . As mentioned, one common application of association rules mining is in the domain of recommending systems. Once item pairs have been identified as having positive relationship, recommendations can be made to customers in order to increase sales. And hopefully, along the way, also introduce customers to items they never would have tried before or even imagined existed!