- Name: Krish Sukhani

  UID: 2018140059

  Batch: D

  Experiment No: 8

  DWM

```
!pip install apyori
```

```
    Requirement already satisfied: apyori in /usr/local/lib/python3.7/dist-packages (1.1.2)
```

import the required libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from apyori import apriori
```

```
from google.colab import drive
drive.mount("/content/gdrive")
```

```
    Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive
```

Importing the Dataset

If you carefully look at the data, we can see that the header is actually the first transaction. Each row corresponds to a transaction and each column corresponds to an item purchased in that specific transaction. The NaN tells us that the item represented by the column was not purchased in that specific transaction.

In this dataset there is no header row.

```
store_data = pd.read_csv('/content/gdrive/MyDrive/datasets/DataSet Association.csv',encoding= 'unicode_escape',h
```

```
store_data.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	
0	shrimp	almonds	avocado	vegetables mix	green grapes	whole weat flour	yams	cottage cheese	energy drink	tomato juice	low fat yogurt	green tea	honey	sa
1	burgers	meatballs	eggs	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2	chutney	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
3	turkey	avocado	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
4	mineral water	milk	energy bar	whole wheat rice	green tea	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N

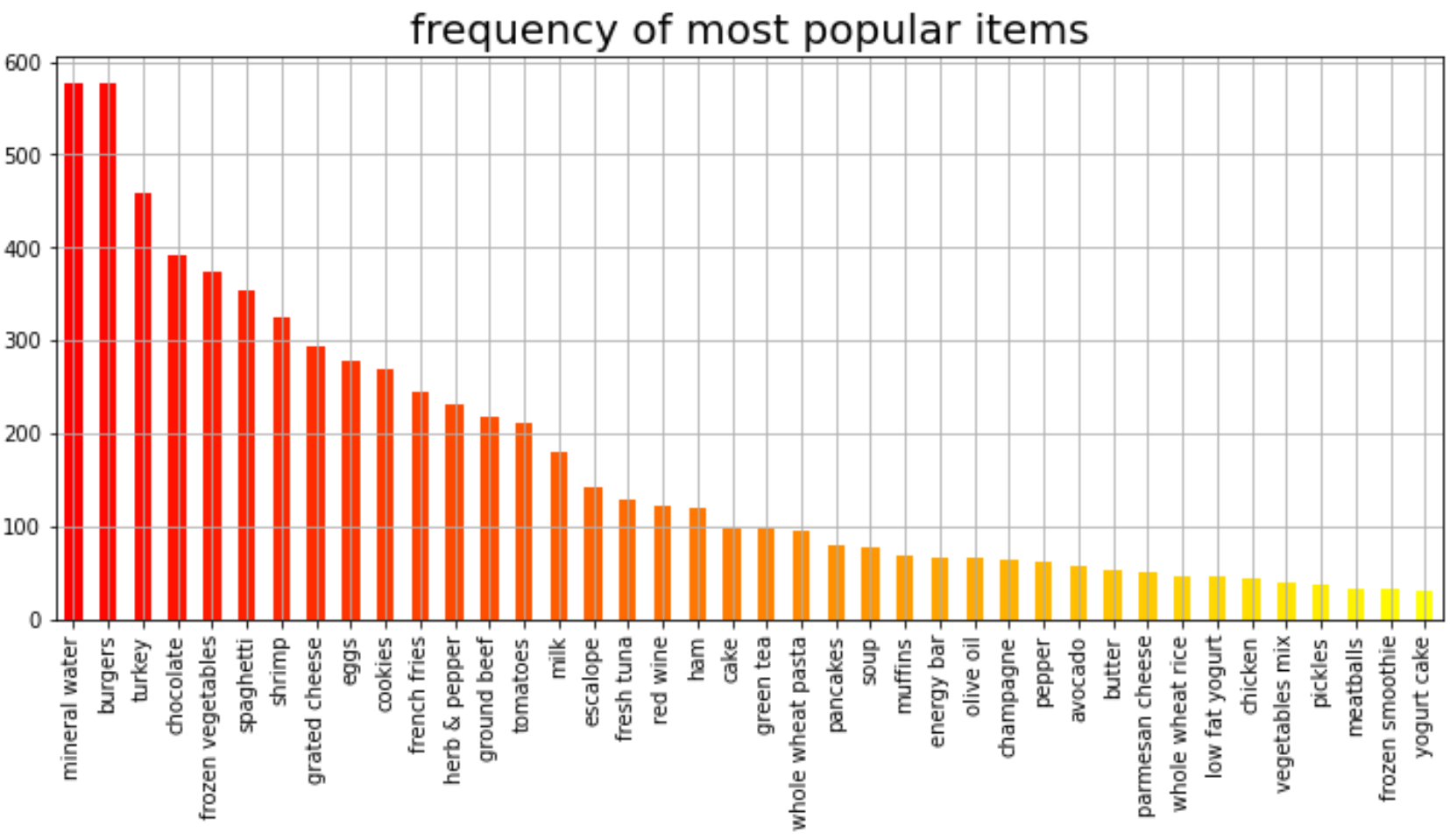
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import seaborn as sns

from wordcloud import WordCloud

plt.rcParams['figure.figsize'] = (5, 5)
wordcloud = WordCloud(background_color = 'white', width = 1200,  height = 1200, max_words = 121).generate(str(st
plt.imshow(wordcloud)
plt.axis('off')
plt.title('Most Popular Items',fontsize = 20)
plt.show()
```


```
plt.rcParams['figure.figsize'] = (12, 5)
color = plt.cm.autumn(np.linspace(0, 1, 40))
store_data[0].value_counts().head(40).plot.bar(color = color)
plt.title('frequency of most popular items', fontsize = 20)
plt.xticks(rotation = 90 )
plt.grid()
plt.show()
```




```
store_data.shape
```

```
(7501, 20)
```

```
store_data.isnull().count()
```

```
0     7501
1     7501
2     7501
3     7501
4     7501
5     7501
6     7501
7     7501
8     7501
9     7501
10    7501
11    7501
12    7501
13    7501
14    7501
15    7501
16    7501
17    7501
18    7501
19    7501
dtype: int64
```

Now we will use the Apriori algorithm to find out which items are commonly sold together, so that store owner can take action to place the related items together or advertise them together in order to have increased profit.

Data Proprocessing

```
#Converting this pandas dataframe into a list of lists
records = []
for i in range(0, 7501):
    records.append([str(store_data.values[i,j]) for j in range(0, 20)])
```

Applying Apriori

```
#5 on weekdays, 7 on weekends
association_rules = apriori(records, min_support=0.0052, min_confidence=0.2, min_lift=3, min_length=2)
association_results = list(association_rules)
print(len(association_results))
```

```
32
```

The first parameter is the list of list that you want to extract rules from.

min_support parameter is used to select the items with support values greater than the value specified by the parameter.

min_confidence parameter filters those rules that have confidence greater than the confidence threshold specified by the parameter

min_lift parameter specifies the minimum lift value for the short listed rules

min_length parameter specifies the minimum number of items that you want in your rules

Viewing the Results

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'mushroom cream sauce', 'escalope'}), support=0.005732568990801226, orderec
```

Observations:

The first item in the list is a list itself containing two items. The first item of the list shows the grocery items in the rule.

The support value for the first rule is 0.0057. This number is calculated by dividing the number of transactions containing mushroom cream sauce divided by total number of transactions. The confidence level for the rule is 0.3006 which shows that out of all the transactions that contain mushrooom cream sauce, 30.06% of the transactions also contain escalope. Finally, the lift of 3.79 tells us that escalope is 3.79 times more likely to be bought by the customers who buy mushrooom cream sauce compared to the default likelihood of the sale of escalope.

```
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("====================================")
```

```
Support: 0.007998933475536596
Confidence: 0.2714932126696833
Lift: 4.13077198425009
====================================
Rule: frozen vegetables -> nan
Support: 0.005332622317024397
Confidence: 0.23255813953488375
Lift: 3.260595522712454
====================================
Rule: frozen vegetables -> nan
Support: 0.008665511265164644
Confidence: 0.31100478468899523
Lift: 3.165328208890303
====================================
Rule: mineral water -> frozen vegetables
Support: 0.007199040127982935
Confidence: 0.30508474576271183
Lift: 3.200616332819722
====================================
Rule: olive oil -> frozen vegetables
Support: 0.005732568990801226
Confidence: 0.20574162679425836
Lift: 3.1303609383037156
====================================
Rule: frozen vegetables -> nan
Support: 0.005999200106652446
Confidence: 0.21531100478468898
Lift: 3.0187810222242093
====================================
Rule: tomatoes -> frozen vegetables
Support: 0.006665777896280496
Confidence: 0.23923444976076558
Lift: 3.4980460188216425
====================================
Rule: grated cheese -> nan
Support: 0.005332622317024397
Confidence: 0.3225806451612903
Lift: 3.283144395325426
====================================
```

```
Rule: herb & pepper -> mineral water
Support: 0.006665777896280496
Confidence: 0.39062500000000006
Lift: 3.975682666214383
=====================================
Rule: herb & pepper -> nan
Support: 0.0063991467804292 76
Confidence: 0.3934426229508197
Lift: 4.004359721511667
=====================================
Rule: nan -> spaghetti
Support: 0.005999200106652446
Confidence: 0.5232558139534884
Lift: 3.005315360233627
=====================================
Rule: olive oil -> nan
Support: 0.007199040127982935
Confidence: 0.20300751879699247
Lift: 3.088761457396025
=====================================
```

The second rule states that herb and cream sauce and ground beef are bought frequently. The support for herb and cream sauce is 0.015. The confidence for this rule is 0.3234 which means that out of all the transactions containing mushroom, 32.34% of the transactions are likely to contain ground beef as well. Finally, lift of 3.291 shows that the ground beef is 3.291 more likely to be bought by the customers that buy herb and cream sauce, compared to its default sale.

Conclusion

Association rule mining algorithms such as Apriori are very useful for finding simple associations between our data items. They are easy to implement and have high explain-ability. Hence, I have found the frequent patterns and association rules from the given dataset and also commented on how the support and confidence effect the results.