Market Basket Insights

MARKET BASKET INSIGHTS PYTHON PROGRAM WITH RECOMENTATION SYSTEM AND VISUALIZATION:

```
import pandas as pd
import matplotlib.pyplot as plt
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

def find(data, ):
    data = list(data.apply(lambda x: x.split(" ")))
    te = TransactionEncoder()
    te_data = te.fit(data).transform(data)
    df = pd.DataFrame(te_data, columns=te.columns_)
    df1 = apriori(df, min_support=0.01, use_colnames=True)
    df2 = df1.sort_values(by="support", ascending=False)
    print(df2)
    df_ar = association_rules(df1, metric="confidence", min_threshold=0.5)
    print(df_ar)
    return df1,df_ar

def rec(list1,target_item=None):
    if target_item:
        recommendations = list1[list1['antecedents'] == frozenset({target_item})]
        if not recommendations.empty:
            print(f"\nRecommendations for {target_item}:")
            print(recommendations[['consequents', 'confidence']])
        else:
            print(f"No recommendations found for {target_item}.")
    return None

def vis(df1,df_ar,category_name):
    df1.sort_values(by="support", ascending=False).plot(kind='bar',
                                            x='itemsets', y='support', legend=False)
    plt.title(f'Support for {category_name}')
    plt.xlabel('Itemsets')
    plt.ylabel('Support')
    plt.show()
```

```
    plt.scatter(df_ar['support'], df_ar['confidence'], alpha=0.5)
    plt.xlabel('Support')
    plt.ylabel('Confidence')
    plt.title(f'Association Rules for {category_name}')
    plt.show()

#pd.set_option("display.max_columns", None)
#pd.set_option("display.max_rows", None)

data = pd.read_csv("Book1.csv")

cc,bb=find(data["Itemname"])
vis(cc,bb,"Itemname")

while True:
    target_item = input("enter the item name to get the
                                                    recomendation:")
    rec(bb,target_item.upper())
    stop = input("you want to stop the recomandation to press Q,
                                            else press any input key:")
    if "Q" ==stop.upper():
        break

cc,bb=find(data["Country"])
vis(cc,bb, "Country")

while True:
    target_item = input("enter the item name to get the
                                                    recomendation:")
    rec(bb,target_item.upper())
    stop = input("you want to stop the recomandation to press Q,
                                            else press any input key:")
    if "Q" ==stop.upper():
        break
```

OUTPUT:

FOR ITEM SETS:

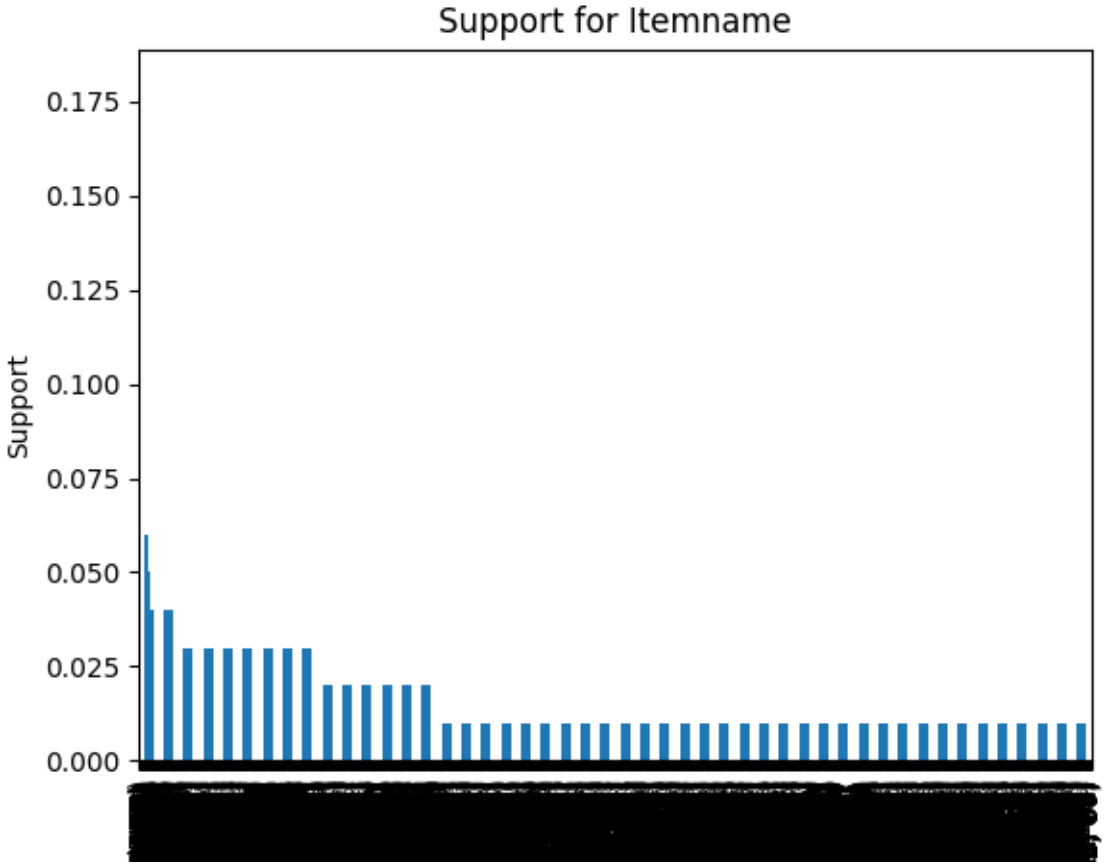
APRIORI ALGORITHM AND ACCOSIATION RULES:

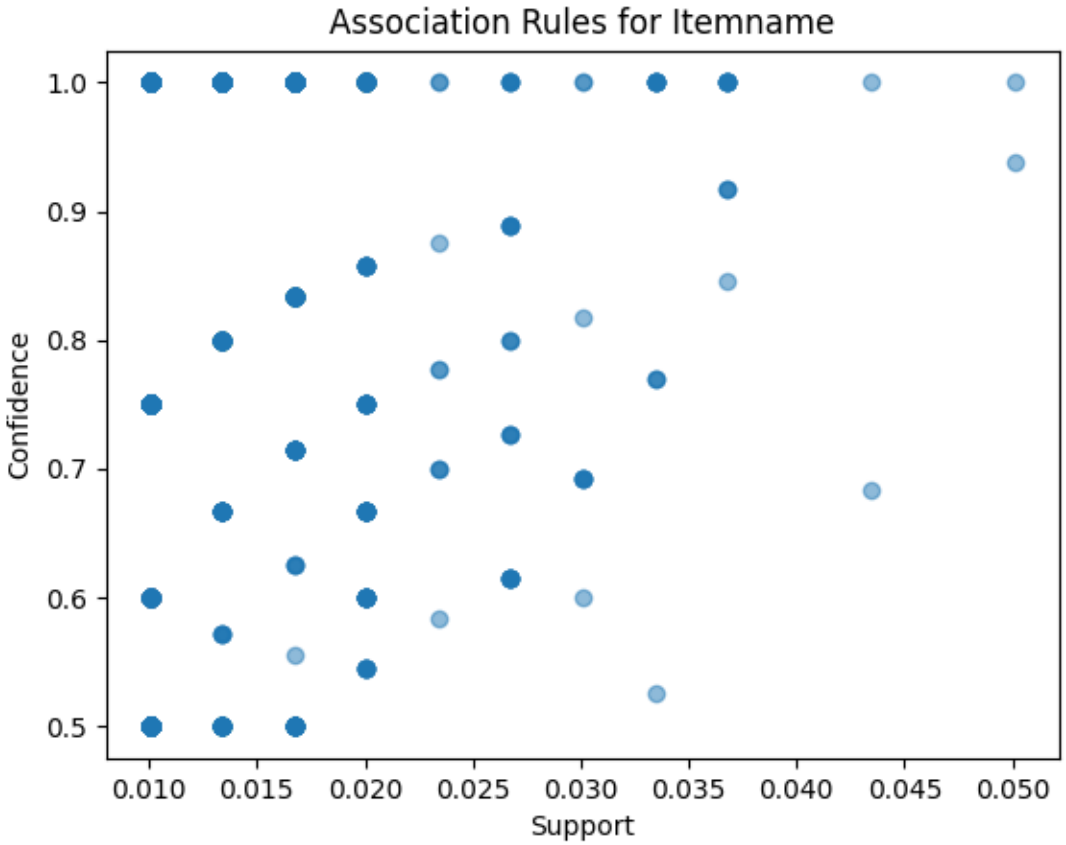
```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18362.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

E:\others\i am chief minister project\project>market_basket_insights_2.py
     support               itemsets
119  0.133779                  (RED)
143  0.113712                (WHITE)
126  0.090301                  (SET)
121  0.083612            (RETROSPOT)
69   0.083612                (HEART)
..        ...                   ...
467  0.010033  (WHITE, ANTIQUE, FRAME)
465  0.010033        (S/3, ANT, WOOD)
464  0.010033       (ANT, S/3, WHITE)
461  0.010033      (FINISH, S/3, ANT)
411  0.010033            (S/3, WOOD)

[822 rows x 2 columns]
           antecedents                       consequents  antecedent support  consequent support  ...      lift  leverage  conviction  zhangs_metric
0                 (10)                         (LIGHTS)            0.010033            0.013378  ...  74.750000  0.009899         inf       0.996622
1              (LIGHTS)                            (10)            0.013378            0.010033  ...  74.750000  0.009899    3.959866       1.000000
2                 (2)                         (CABINET)            0.013378            0.020067  ...  37.375000  0.009765    3.919732       0.986441
3              (CABINET)                            (2)            0.020067            0.013378  ...  37.375000  0.009765    1.973244       0.993174
4                 (2)                         (DRAWER)            0.013378            0.023411  ...  32.035714  0.009720    3.906355       0.981921
...               ...                              ...                 ...                 ...  ...        ...       ...         ...            ...
3733    (KNITTED, HOT)          (BOTTLE, FLAG, UNION, WATER)            0.016722            0.016722  ...  59.800000  0.016443         inf       1.000000
3734  (KNITTED, BOTTLE)            (HOT, FLAG, UNION, WATER)            0.016722            0.016722  ...  59.800000  0.016443         inf       1.000000
3735            (FLAG)  (UNION, KNITTED, WATER, HOT, BOTTLE)            0.016722            0.016722  ...  59.800000  0.016443         inf       1.000000
3736           (UNION)   (FLAG, KNITTED, WATER, HOT, BOTTLE)            0.033445            0.016722  ...  29.900000  0.016163    1.966555       1.000000
3737         (KNITTED)     (FLAG, UNION, WATER, HOT, BOTTLE)            0.020067            0.016722  ...  49.833333  0.016387    5.899666       1.000000

[3738 rows x 10 columns]
```

RECOMMANDATION SYSTEM:

```
C:\Windows\System32\cmd.exe
enter the item name to get the recomendation:KNITTED

Recommendations for KNITTED:
                          consequents  confidence
79                           (BOTTLE)    0.833333
183                            (FLAG)    0.833333
229                             (HOT)    0.833333
254                           (UNION)    0.833333
256                           (WATER)    0.833333
653                   (FLAG, BOTTLE)    0.833333
665                    (HOT, BOTTLE)    0.833333
678                  (BOTTLE, UNION)    0.833333
682                  (BOTTLE, WATER)    0.833333
998                      (HOT, FLAG)    0.833333
1011                    (FLAG, UNION)    0.833333
1017                    (FLAG, WATER)    0.833333
1181                     (HOT, UNION)    0.833333
1185                     (HOT, WATER)    0.833333
1262                   (UNION, WATER)    0.833333
1905              (FLAG, HOT, BOTTLE)    0.833333
1935            (FLAG, BOTTLE, UNION)    0.833333
1947            (FLAG, BOTTLE, WATER)    0.833333
1969             (HOT, BOTTLE, UNION)    0.833333
1977             (HOT, BOTTLE, WATER)    0.833333
1994           (BOTTLE, UNION, WATER)    0.833333
2365              (HOT, FLAG, UNION)    0.833333
2377              (HOT, FLAG, WATER)    0.833333
2400            (FLAG, UNION, WATER)    0.833333
2603             (HOT, UNION, WATER)    0.833333
3163        (BOTTLE, HOT, FLAG, UNION)    0.833333
3186        (BOTTLE, HOT, FLAG, WATER)    0.833333
3236      (BOTTLE, FLAG, UNION, WATER)    0.833333
3259       (HOT, BOTTLE, UNION, WATER)    0.833333
3437        (HOT, FLAG, UNION, WATER)    0.833333
3737  (FLAG, UNION, WATER, HOT, BOTTLE)    0.833333
you want to stop the recomandation to press Q, else press any input key:q
```

VISUALIZATION:





FOR COUNTRY:

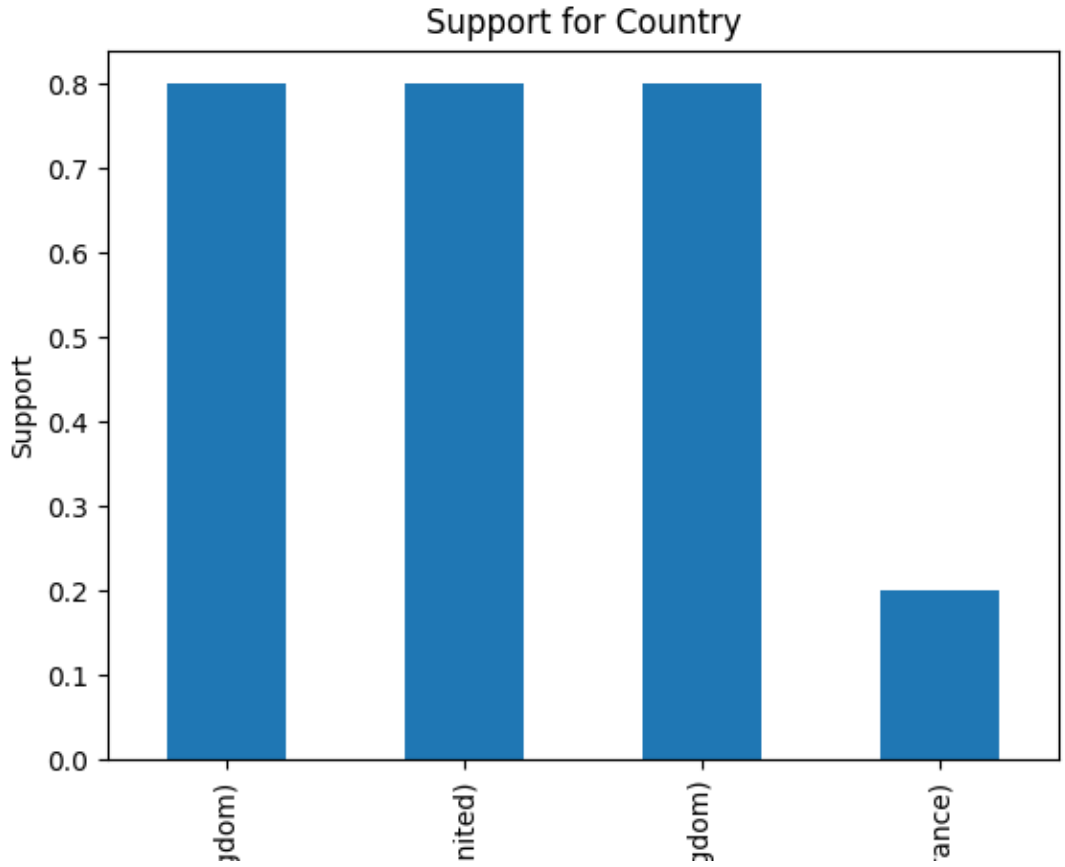
APRIORI ALGORITHM AND ACCOSIATION RULES:

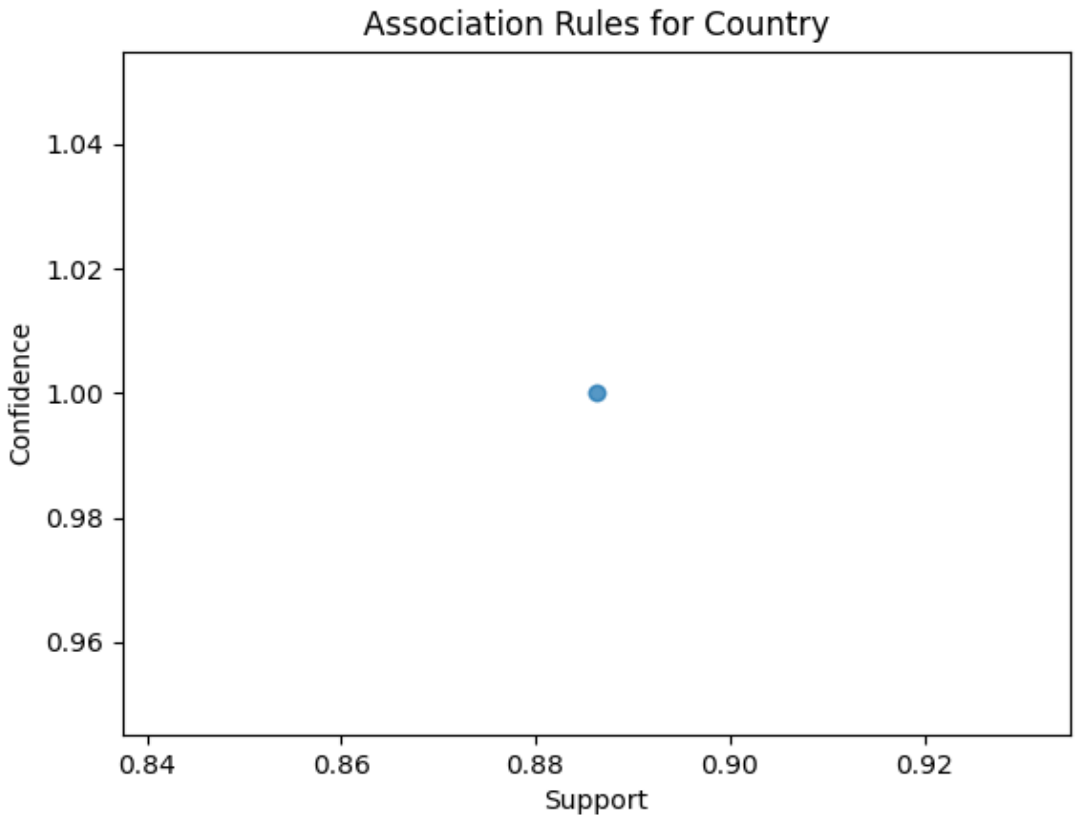
```
C:\Windows\System32\cmd.exe
    support           itemsets
2  0.886288          (Kingdom)
3  0.886288           (United)
4  0.886288  (Kingdom, United)
1  0.066890           (France)
0  0.046823        (Australia)
  antecedents consequents  antecedent support  consequent support   support  confidence      lift  leverage  conviction  zhangs_metric
0   (Kingdom)    (United)           0.886288           0.886288  0.886288         1.0  1.128302  0.100782         inf            1.0
1    (United)   (Kingdom)           0.886288           0.886288  0.886288         1.0  1.128302  0.100782         inf            1.0
```

RECOMMANDATION SYSTEM:

```
C:\Windows\System32\cmd.exe
enter the item name to get the recomendation:Kingdom
No recommendations found for KINGDOM.
you want to stop the recomandation to press Q, else press any input key:(United
enter the item name to get the recomendation:United
No recommendations found for UNITED.
you want to stop the recomandation to press Q, else press any input key:d
enter the item name to get the recomendation:Australia
No recommendations found for AUSTRALIA.
you want to stop the recomandation to press Q, else press any input key:France
enter the item name to get the recomendation:France
No recommendations found for FRANCE.
you want to stop the recomandation to press Q, else press any input key:Kingdom, United
enter the item name to get the recomendation:Kingdom, United
No recommendations found for KINGDOM, UNITED.
you want to stop the recomandation to press Q, else press any input key:q
```

VISUALIZATION:

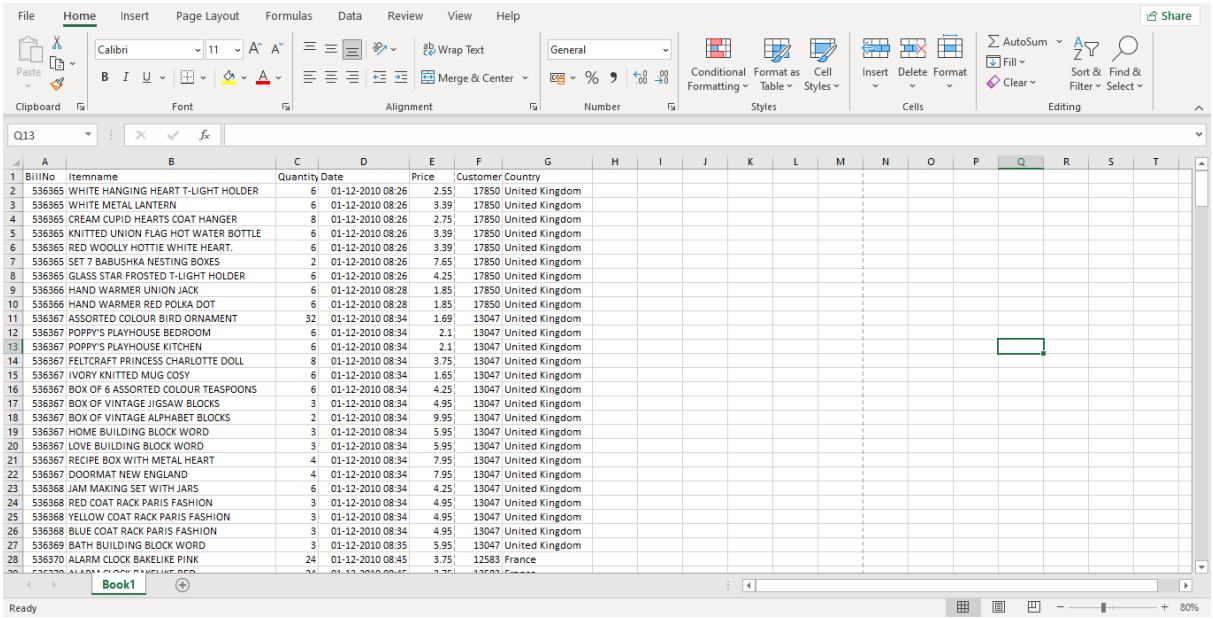




EXPLANATION:

- This code is performing market basket analysis on a dataset.
- Market basket analysis is the process of finding associations or relationships between items that are frequently bought together.
- The code imports necessary libraries such as pandas, matplotlib, and mlxtend.
- Pandas library is used to work with tabular data and perform operations such as slicing, filtering, and merging.
- Matplotlib library is used to create plots and charts.
- Mlxtend library provides functions for frequent pattern mining and association rule learning.
- The function "find" takes in a dataset as input (presumably in the form of a pandas DataFrame).
- It splits the entries in each row by space into a list using the lambda function and apply method from pandas.
- This is done to make the data suitable for market basket analysis, where transactions are typically represented by lists of items.
- A TransactionEncoder object called "te" is created.
- This encoder converts transaction datasets into one-hot encoded format, which can then be used with different algorithms.
- One-hot encoding represents each item in a transaction as a binary variable.

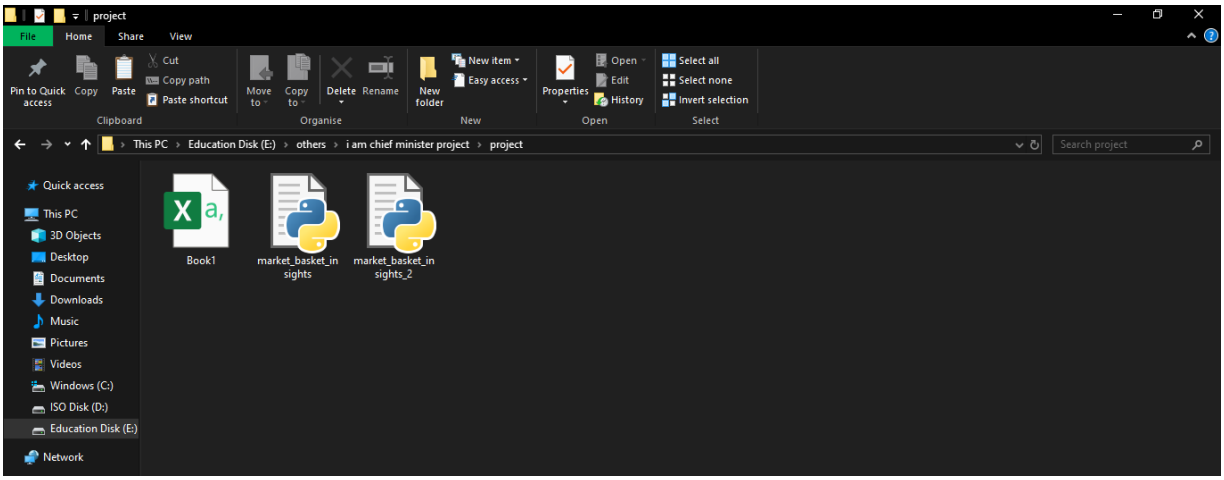
- The apriori algorithm from mlxtend library is used to find frequent itemsets from the dataset.
- Apriori uses an iterative approach to mine frequent patterns by gradually increasing the length of combinations it generates until no more valid extensions can be found.
- Once the frequent itemsets are obtained, association rules are generated using the association_rules function from mlxtend library.
- Association rules specify relationships between sets of items based on their frequency in the dataset.
- These rules can then be used to uncover insights like if certain items are commonly purchased together or if purchase of one item affects likelihood of purchasing another.
- Overall, this code shows how to use market basket analysis techniques on a given dataset using mlxtend library functions.
- The code first imports the pandas library, followed by the matplotlib.pyplot module.
- This will allow us to create graphs and charts.
- Next, we import the TransactionEncoder module.
- This will allow us to encode transactions into a format that can be processed by the MLxtend platform.
- Finally, we use the find() function to search for transactions in our data set.
- The find() function takes two parameters: data and a filter object.
- The filter object allows us to specify which transactions we want to include in our search.
- From the code above, it seems that we are performing association rule mining using the Apriori algorithm on a dataset.



BillNo	Itemname	Quantity	Date	Price	Customer	Country
536365	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850	United Kingdom
536365	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850	United Kingdom
536365	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850	United Kingdom
536365	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850	United Kingdom
536365	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850	United Kingdom
536365	SET 7 BABUSHKA NESTING BOXES	2	01-12-2010 08:26	7.65	17850	United Kingdom
536365	GLASS STAR FROSTED T-LIGHT HOLDER	6	01-12-2010 08:26	4.25	17850	United Kingdom
536366	HAND WARMER UNION JACK	6	01-12-2010 08:28	1.85	17850	United Kingdom
536366	HAND WARMER RED POLKA DOT	6	01-12-2010 08:28	1.85	17850	United Kingdom
536367	ASSORTED COLOUR BIRD ORNAMENT	32	01-12-2010 08:34	1.69	13047	United Kingdom
536367	POPPY'S PLAYHOUSE BEDROOM	6	01-12-2010 08:34	2.1	13047	United Kingdom
536367	POPPY'S PLAYHOUSE KITCHEN	6	01-12-2010 08:34	2.1	13047	United Kingdom
536367	FELTCRAFT PRINCESS CHARLOTTE DOLL	8	01-12-2010 08:34	3.75	13047	United Kingdom
536367	IVORY KNITTED MUG COSY	6	01-12-2010 08:34	1.65	13047	United Kingdom
536367	BOX OF 6 ASSORTED COLOUR TEASPOONS	6	01-12-2010 08:34	4.25	13047	United Kingdom
536367	BOX OF VINTAGE JIGSAW BLOCKS	3	01-12-2010 08:34	4.95	13047	United Kingdom
536367	BOX OF VINTAGE ALPHABET BLOCKS	2	01-12-2010 08:34	9.95	13047	United Kingdom
536367	HOME BUILDING BLOCK WORD	3	01-12-2010 08:34	5.95	13047	United Kingdom
536367	LOVE BUILDING BLOCK WORD	3	01-12-2010 08:34	5.95	13047	United Kingdom
536367	RECIPE BOX WITH METAL HEART	4	01-12-2010 08:34	7.95	13047	United Kingdom
536367	DOORMAT NEW ENGLAND	4	01-12-2010 08:34	7.95	13047	United Kingdom
536368	JAM MAKING SET WITH JARS	6	01-12-2010 08:34	4.25	13047	United Kingdom
536368	RED COAT RACK PARIS FASHION	3	01-12-2010 08:34	4.95	13047	United Kingdom
536368	YELLOW COAT RACK PARIS FASHION	3	01-12-2010 08:34	4.95	13047	United Kingdom
536368	BLUE COAT RACK PARIS FASHION	3	01-12-2010 08:34	4.95	13047	United Kingdom
536369	BATH BUILDING BLOCK WORD	3	01-12-2010 08:35	5.95	13047	United Kingdom
536370	ALARM CLOCK BAKELIKE PINK	24	01-12-2010 08:45	3.75	12583	France

- First, we use the `fit()` method of the `te` (TransactionEncoder) object to encode our dataset into a binary matrix where each column represents a unique item and each row represents a transaction.
- The resulting binary matrix is stored in `te_data`.
- Next, we create a DataFrame called `df` from the encoded data using the column names obtained from `te.columns_`.
- This allows us to work with a more human-readable representation of our data.
- We then apply Apriori algorithm on this dataframe by calling `apriori()` function with min_support value set as 0.01 which means the minimum support needed for an itemset to be considered frequent is 1%.
- The result of the Apriori algorithm is stored in `df1`, where each row represents a frequent itemset and contains two columns: "support" (indicating how frequently the itemset occurs) and "itemsets" (which lists the items that make up that frequent itemset).
- We then sort this dataframe (`df2`) based on support values in descending order, so that we can see the most significant associations first.
- We display both `df2` and `df_ar` on the console using print statements.
- Finally, we apply association rules to these frequent itemsets by calling `association_rules()` function with metric set to "confidence" and min_threshold set to 0.5.
- This means only rules with confidence greater than or equal to 50% will be generated.
- The resulting association rules are stored in `df_ar`, which consists of several columns including antecedent(s), consequent(s), support, confidence
- The code will first fit a model to the training data and then will transform the dataset into a DataFrame.
- The DataFrame will be sorted by support and the highest confidence association rules will be displayed in a separate column.
- This code appears to be a Python function for generating recommendations based on a given target item.
- The function takes in two parameters: `list1` and `target_item`, and returns two variables `df1` and `df_ar`.
- However, these return variables are not used or referenced anywhere else in the code, so their purpose is unclear.
- The core functionality of this code is the `rec()` function.
- Within this function, it first checks if a `target_item` is provided.
- If there is a target item, it filters the `list1` DataFrame to find rows where the antecedents column contains the target item.
- If there are any matching rows found (`recommendations` DataFrame is not empty), it prints a message stating that recommendations are found for the target item, followed by printing the `consequents` and `confidence` columns from the recommendations DataFrame.

- On the other hand, if no matching rows are found, it prints a message stating that no recommendations were found for the target item.
- Overall, this code seems to be part of a larger recommender system where association rules are used to generate recommendations based on antecedent-consequent relationships between items.
- The specific implementation details or dependencies required for this function cannot be determined without additional context.
- The code creates a list of recommendations for an item, which can be either a single item or a set of items.
- If the target_item is not None, then the code will print out the recommendations for that item.
- Otherwise, it will return None.
- This code seems to define a function called "vis" that takes three parameters: df1, df_ar, and category_name.
- In the first line of the function, it looks like df1 DataFrame is sorted in descending order based on the "support" column and then a bar plot is created with x-axis representing the itemsets and y-axis representing the support values.
- The next few lines set the title, x-label, and y-label for the first plot.
- Then another scatter plot is created using df_ar DataFrame where support values are mapped to x-axis and confidence values are mapped to y-axis.
- The alpha parameter specifies the transparency level for each data point in the scatter plot.
- Finally, there are labels added for x-axis and y-axis of second plot.
- From what I can infer, this code might be used to visualize support and confidence values of itemsets based on a given category.
- The code will create a bar graph displaying the support and confidence for each category.
- The x-axis will show the number of itemsets in that category, while the y-axis will show the respective support and confidence values.



CONCLUSION:

Market basket analysis is a powerful technique for uncovering hidden patterns and associations between products.

By understanding customer purchasing behaviour and identifying potential cross-selling opportunities, retailers can make more informed decisions about product placement, marketing campaigns, and inventory management.