Market Basket Analysis

What is Market Basket Analysis?

- A technique used to analyze customer purchasing behavior.
- Identifies patterns in customer transactions.
- Helps in understanding which products are frequently bought together

What is the Apriori Algorithm?

- A classic algorithm used in data mining for learning association rules.
- Uses a bottom-up approach(approche ascendante), where frequent subsets are extended one item at a time.
- Ensures that any subset of a frequent itemset is also frequent.

Steps in the Apriori algorithm:

- **Identify frequent items:**

  Generate itemsets whose support is higher than a minimum threshold.

- **Generate association rules:**

  Create rules from frequent itemsets.

  Calculate the confidence and strength of the rules.

- **Filter strong rules:**

  Selects rules whose confidence and strength are higher than the specified thresholds.

What is Data Mining?

- **Definition:** The process of discovering patterns, correlations, and insights from large amounts of data by using statistical and computational techniques.
- **Goal:** To extract useful information and transform it into an understandable structure for further use.

Practical Application

1.Feature Engineering :

```
df.shape
```

```
(541909, 8)
```

load data into a data frame, to get a quick overview of the structure and content of the dataset

```
[5]: df.head()
```

[5]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

```
[6]: df.tail()
```

[6]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0	France

The number of missing values :

```
InvoiceNo          0
StockCode          0
Description     1454
Quantity           0
InvoiceDate        0
UnitPrice          0
CustomerID    135080
Country            0
dtype: int64
```

- "The Description column has 1454 missing entries, which might affect any analysis that relies on product descriptions. It's important to understand why these descriptions are missing and consider if this affects any specific subset of data."
- "The CustomerID column has a significant number of missing values (135080), indicating that a large portion of transactions are not linked to specific customers. This might be due to guest checkouts or incomplete data capture."

Display information about the DataFrame :

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  datetime64[ns]
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(1), object(4)
memory usage: 33.1+ MB
```

```
#get the summary statistics
df.describe()
```

	Quantity	InvoiceDate	UnitPrice	CustomerID
count	541909.000000	541909	541909.000000	406829.000000
mean	9.552250	2011-07-04 13:34:57.156386048	4.611114	15287.690570
min	-80995.000000	2010-12-01 08:26:00	-11062.060000	12346.000000
25%	1.000000	2011-03-28 11:34:00	1.250000	13953.000000
50%	3.000000	2011-07-19 17:17:00	2.080000	15152.000000
75%	10.000000	2011-10-19 11:27:00	4.130000	16791.000000
max	80995.000000	2011-12-09 12:50:00	38970.000000	18287.000000
std	218.081158	NaN	96.759853	1713.600303

TYPE OF FEATURES :

```
df.dtypes
```

```
[13]:

InvoiceNo              object
StockCode              object
Description            object
Quantity                int64
InvoiceDate    datetime64[ns]
UnitPrice             float64
CustomerID            float64
Country                object
dtype: object
```

2.Data Cleaning:

First, some of the descriptions have spaces that need to be removed

Description	Q
WHITE HANGING HEART T-LIGHT HOLDER	
WHITE METAL LANTERN	
CREAM CUPID HEARTS COAT HANGER	
KNITTED UNION FLAG HOT WATER BOTTLE	
RED WOOLLY HOTTIE WHITE HEART.	

drop the rows that don't have invoice numbers

```
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True) #drop rows that dont have
```

converting InvoiceNo to string

```
df['InvoiceNo']=df['InvoiceNo'].astype('str')#converting InvoiceNo to string
```

Delete credit transactions(those with invoice numbers containing C):
'Credit transactions' refer to transactions where products are returned or credits are issued to customers. These transactions are recorded in the accounting system with special invoice numbers to indicate that they are returns or refunds rather than regular sales.

```
df.shape
```

```
[43]:

(532621, 8)
```

After cleaning up the data, we need to combine the items into a single transaction per line, with each product encoded as a single variable. For simplicity's sake, we're only going to look at sales in France.

1. Filter data for France:

```
#get data for samples which have Country as France
basket = df[df['Country'] == "France"]
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
26	536370	22728	ALARM CLOCK BAKELIKE PINK	24	2010-12-01 08:45:00	3.75	12583.0	France
27	536370	22727	ALARM CLOCK BAKELIKE RED	24	2010-12-01 08:45:00	3.75	12583.0	France
28	536370	22726	ALARM CLOCK BAKELIKE GREEN	12	2010-12-01 08:45:00	3.75	12583.0	France
29	536370	21724	PANDA AND BUNNIES STICKER SHEET	12	2010-12-01 08:45:00	0.85	12583.0	France
30	536370	21883	STARS GIFT TAPE	24	2010-12-01 08:45:00	0.65	12583.0	France

2. Grouping by invoice number and description:

```
.groupby(['InvoiceNo', 'Description'])['Quantity']
```

3. Calculate sum of quantities and reorganize data:

```
.sum().unstack().reset_index().fillna(0)
```

Sum quantities: sum() adds up the quantities of each product in each invoice.
Reorganize with unstack(): This method transforms the Description column into separate columns, one for each product, creating a table where each row represents a single transaction with the products as columns.
Reset indexes: reset_index() transforms the hierarchical index into ordinary columns.
Replace NaN with 0: fillna(0) replaces missing values (which appear when a product has not been purchased in a transaction) with 0

4. Define index as invoice number:

```
.set_index('InvoiceNo')
```

Description	COLOUR SPACEBOY PEN	COLOURED PARTY BALLOONS	HOUSE PAINTED WOOD	MESSAGE CARDS WITH ENVELOPES	12 PENCIL SMALL TUBE WOODLAND	SMALL TUBE RED RETROSPOT	PENCILS SMALL TUBE SKULL	PENCILS TALL TUBE POSY	TALL TUBE RED RETROSPOT	12 PENCILS TALL TUBE WOODLAND	CHRISTMAS GLASS BALL 20 LIGHTS	CUTLERY SET PANTRY DESIGN	WO CUT DISPOS
InvoiceNo													
536370	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
536852	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
536974	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537065	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537463	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537468	24.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537693	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537897	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
537967	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

```
print(basket.shape)
```

```
(392, 1563)
```

There are a lot of zeros in the data but we also need to make sure any positive values are converted to a 1 and anything less than 0 is set to 0.

```
Description  10 COLOUR SPACEBOY PEN  12 COLOURED PARTY BALLOONS  \
InvoiceNo
536370                            0                          0
536852                            0                          0
536974                            0                          0
537065                            0                          0
537463                            0                          0

Description  12 EGG HOUSE PAINTED WOOD  12 MESSAGE CARDS WITH ENVELOPES  \
InvoiceNo
536370                               0                                0
536852                               0                                0
536974                               0                                0
537065                               0                                0
537463                               0                                0

Description  12 PENCIL SMALL TUBE WOODLAND  \
InvoiceNo
```

Now that the data is structured properly, we can generate frequent item sets that have a support of at least 7%
the Apriori algorithm works best with DataFrames containing Boolean values (True/False) rather than integer values (0/1). To avoid this warning and improve performance, you can convert the DataFrame to a Boolean value before applying the Apriori algorithm.

```
Description  10 COLOUR SPACEBOY PEN  12 COLOURED PARTY BALLOONS  \
InvoiceNo
536370                       False                      False
536852                       False                      False
536974                       False                      False
537065                       False                      False
537463                       False                      False

Description  12 EGG HOUSE PAINTED WOOD  12 MESSAGE CARDS WITH ENVELOPES  \
InvoiceNo
536370                          False                           False
536852                          False                           False
536974                          False                           False
537065                          False                           False
537463                          False                           False

Description  12 PENCIL SMALL TUBE WOODLAND  \
InvoiceNo
```

The final step is to generate the rules with their corresponding support, confidence and lift.

```
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules.head()
```

metric :
This parameter specifies the metric to be used to evaluate rule quality.
Here, we use “lift” to select rules based on their lift.
min_threshold :
This parameter specifies the minimum threshold for the chosen metric.
Here, we define a threshold of 1 for the lift, which means that only rules with a lift of at least 1 will be retained.

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
0	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE PINK)	0.096939	0.102041	0.073980	0.763158	7.478947	0.064088	3.791383	0.959283
1	(ALARM CLOCK BAKELIKE PINK)	(ALARM CLOCK BAKELIKE GREEN)	0.102041	0.096939	0.073980	0.725000	7.478947	0.064088	3.283859	0.964734
2	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878	0.976465
3	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181	0.979224
4	(ALARM CLOCK BAKELIKE GREEN)	(POSTAGE)	0.096939	0.765306	0.084184	0.868421	1.134737	0.009996	1.783673	0.131484

Metrics that can be used:

Support:

- Le support d'un itemset est la proportion de transactions dans la base de données qui contient cet itemset.
- C'est une mesure de la fréquence d'apparition d'un ensemble d'items.
- Formellement, pour un itemset X, le support est défini comme :

$$\text{Support}(X) = \frac{\text{Nombre de transactions contenant } X}{\text{Nombre total de transactions}}$$

Confiance (Confidence):

- La confiance d'une règle d'association $A \rightarrow B$ est une mesure de la probabilité que B soit acheté lorsqu'un client achète A.
- Formellement, pour une règle $A \rightarrow B$, la confiance est définie comme :

$$\text{Confiance}(A \rightarrow B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)}$$

Lift:

- Le lift mesure l'importance d'une règle d'association par rapport à l'indépendance statistique des items A et B.
- Il compare la probabilité de B apparaissant lorsque A est présent à la probabilité de B apparaissant indépendamment de A.
- Formellement, pour une règle $A \rightarrow B$, le lift est défini comme :

$$\text{Lift}(A \rightarrow B) = \frac{\text{Confiance}(A \rightarrow B)}{\text{Support}(B)}$$

- Un lift supérieur à 1 indique une relation positive entre A et B, tandis qu'un lift inférieur à 1 indique une relation négative.

1. Leverage

Définition :

- Le leverage mesure la différence entre le support observé d'une règle $A \rightarrow B$ et le support attendu si A et B étaient indépendants.
- Le leverage est défini comme :

$$\text{Leverage}(A \rightarrow B) = \text{Support}(A \cup B) - \text{Support}(A) \times \text{Support}(B)$$

Interprétation :

- Un leverage de 0 signifie qu'il n'y a pas de relation entre A et B.
- Un leverage positif indique une corrélation positive entre A et B.
- Un leverage négatif indique une corrélation négative entre A et B.

2. Conviction

Définition :

- La conviction mesure la dépendance entre A et B en comparant la probabilité que A apparaisse sans B (en l'absence de B).
- La conviction est définie comme :

$$\text{Conviction}(A \rightarrow B) = \frac{1 - \text{Support}(B)}{1 - \text{Confiance}(A \rightarrow B)}$$

Interprétation :

- Une conviction de 1 signifie que A et B sont indépendants.
- Une conviction supérieure à 1 indique une relation positive entre A et B.
- Une conviction infinie (approximativement) indique une confiance de 1 (i.e., chaque fois que A apparaît, B apparaît aussi).

3. Zhang's Metric

Définition :

- Zhang's Metric (ou l'indice de Zhang) est une métrique qui mesure la force d'une règle d'association en tenant compte de la proportion de transactions contenant A, B, et $A \cup B$.
- Zhang's Metric est défini comme :

$$\text{Zhang's Metric}(A \rightarrow B) = \frac{\text{Support}(A \cup B) - \text{Support}(A)}{\max(\text{Support}(A \cup B) \times (1 - \text{Support}(A)), \text{Support}(A)}$$

Interprétation :

- Zhang's Metric varie entre -1 et 1.
- Une valeur proche de 1 indique une forte corrélation positive entre A et B.
- Une valeur proche de -1 indique une forte corrélation négative entre A et B.
- Une valeur de 0 indique aucune corrélatic

select the most significant and potentially interesting rules for deeper analysis

```
rules[(rules['lift'] >= 6) &
      (rules['confidence'] >= 0.8)]
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
2	(ALARM CLOCK BAKELIKE RED)	(ALARM CLOCK BAKELIKE GREEN)	0.094388	0.096939	0.079082	0.837838	8.642959	0.069932	5.568878	0.976465
3	(ALARM CLOCK BAKELIKE GREEN)	(ALARM CLOCK BAKELIKE RED)	0.096939	0.094388	0.079082	0.815789	8.642959	0.069932	4.916181	0.979224
75	(SET/6 RED SPOTTY PAPER PLATES)	(SET/20 RED RETROSPOT PAPER NAPKINS)	0.127551	0.132653	0.102041	0.800000	6.030769	0.085121	4.336735	0.956140
76	(SET/6 RED SPOTTY PAPER CUPS)	(SET/6 RED SPOTTY PAPER PLATES)	0.137755	0.127551	0.122449	0.888889	6.968889	0.104878	7.852041	0.993343
77	(SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS)	0.127551	0.137755	0.122449	0.960000	6.968889	0.104878	21.556122	0.981725
79	(ALARM CLOCK BAKELIKE RED, POSTAGE)	(ALARM CLOCK BAKELIKE GREEN)	0.086735	0.096939	0.071429	0.823529	8.495356	0.063021	5.117347	0.966081
80	(ALARM CLOCK BAKELIKE GREEN, POSTAGE)	(ALARM CLOCK BAKELIKE RED)	0.084184	0.094388	0.071429	0.848485	8.989353	0.063483	5.977041	0.970454
114	(SET/6 RED SPOTTY PAPER CUPS, POSTAGE)	(SET/6 RED SPOTTY PAPER PLATES)	0.117347	0.127551	0.102041	0.869565	6.817391	0.087073	6.688776	0.966763
116	(POSTAGE, SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS)	0.107143	0.137755	0.102041	0.952381	6.913580	0.087281	18.107143	0.958000
119	(SET/6 RED SPOTTY PAPER PLATES)	(SET/6 RED SPOTTY PAPER CUPS, POSTAGE)	0.127551	0.117347	0.102041	0.800000	6.817391	0.087073	4.413265	0.978070
120	(SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO	(SET/6 RED SPOTTY PAPER PLATES)	0.102041	0.127551	0.099490	0.975000	7.644000	0.086474	34.897959	0.967949

⇒In looking at the rules, it seems that the green and red alarm clocks are purchased together and the red paper cups, napkins and plates are purchased together