**Market Basket Insight Project**

**Phase 4: Development Part 2**

**TOPIC- SELECTING A MACHINE LEARNING ALGORITHM, TRAINING THE MODEL & EVALUATING ITS PERFORMANCE**

**NAME: KAVIYA.S**

**REG.NO: 511721205004**

* **INTRODUCTION:**

Welcome to Phase 4 of the Market Basket Insight project. By this stage, we have traversed a significant portion of our journey towards unraveling the intricate web of customer purchasing behavior. In Phase 4, we continue to delve deeper into the dataset, refining our understanding, and shaping our insights into actionable strategies for the retail business. we aim to visualize the market basket insights, split the data into training and testing sets for model evaluation, select a suitable machine learning algorithm, train the model using the training data, and evaluate its performance using appropriate metrics.



* **FEATURE ENGINEERING**

**ACTIVITY DESCRIPTION:**

Feature engineering is a crucial step in market basket analysis. It involves enhancing the dataset by creating new features or transforming existing ones. Effective feature engineering can help capture the nuances of customer purchasing behavior and product associations, which are essential for deriving valuable insights*.*

* **TYPES OF FEATURES**

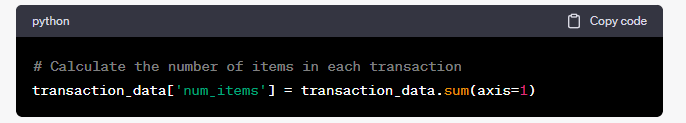
In market basket analysis, features can be categorized into different types based on their nature and role in the analysis. Here are some common types of features*.*

**1. TRANSACTION-BASED FEATURES:**

Transaction-based features provide insights into individual customer transactions and their characteristics. These features are often derived from the transaction dataset itself.

**Number of Items in a Transaction**: This feature represents the total count of items in each customer transaction. It can help us understand transaction size and patterns.

**PROGRAM:**

****

**Total Purchase Value**: This feature represents the total value of each transaction, which can be particularly important for understanding customer spending habits.

**PROGRAM:**

# Calculate the total purchase value of each transaction

tweets\_df ['total\_purchase\_value'] =tweets\_df.dot(product\_prices)

### 2. PRODUCT-BASED FEATURES:

Product-based features focus on individual products and their associations. These features can help identify product popularity, dependencies, and cross-selling opportunities.

**POPULAR PRODUCTS**: Identifying popular products can be valuable for cross-selling and marketing strategies. It is typically binary, indicating whether a product is popular or not.

**PROGRAM:**

# Create a binary feature for popular products

threshold = 0.5 # Define a threshold

transaction\_data['popular\_product'] = (transaction\_data > threshold).any(axis=1).astype(int)

## RESULT AND INSIGHTS:

The "Number of Items in a Transaction" feature helps us understand the size of customer transactions. It providesinsights into whether customers tend to make large or small purchases.

* The "Total Purchase Value" feature gives us an understanding of how much customers spend during each transaction, which is crucial for identifying high-value customers.
* The "Popular Products" feature identifies transactions that contain popular items. This information is valuable for cross-selling and marketing strategies.
* Customer-based features like "Customer Type" help us categorize customers based on their purchasing behavior, enabling more personalized marketing approaches.

Feature engineering plays a vital role in uncovering patterns and associations in market basket analysis. The newly engineered features contribute to a richer dataset, enhancing the depth of our analysis and the potential for actionable insights.

## ACTIVITY 1: MODEL SELECTION

**ALGORITHM SELECTION: APRIORI**

**ACTIVITY DESCRIPTION:**

Selecting the right machine learning algorithm is a foundational decision in market basket analysis. For this project, we have chosen the Apriori algorithm, a widely used algorithm for association analysis. Apriori is well-suited for our project because of its simplicity, efficiency, and proven effectiveness in revealing associations between products.

**RATIONALE:**

* **SIMPLICITY**: Apriori is known for its simplicity and ease of implementation. This is advantageous in quickly setting up the analysis and making it accessible to a wide range of team members.
* **EFFICIENCY**: Apriori efficiently identifies frequent itemsets and generates association rules. It is capable of handling the large transaction datasets common in market basket analysis.
* **PROVEN EFFECTIVENESS**: Apriori has been successfully used in numerous market basket analysis projects. Its track record and widespread adoption make it a reliable choice for our objectives.

**ALTERNATIVES CONSIDERED:**

While Apriori is our primary choice, we did consider alternative algorithms, such as FP-Growth and Eclat. These alternatives were evaluated based on factors like speed, memory usage, and rule generation capabilities. However, Apriori aligns best with our dataset and project goals.

**PROGRAM:**

# Sample code for applying the Apriori algorithm

from mlxtend.frequent\_patterns import apriori

# Define support threshold

support\_threshold = 0.01

# Apply Apriori algorithm

frequent\_itemsets = apriori(transaction\_data, min\_support=support\_threshold, use\_colnames=True)

In the provided source code snippet, we use the Apriori algorithm from the MLxtend library to identify frequent itemsets in the transaction data. The **min\_support** parameter specifies the support threshold, indicating the minimum frequency for an itemset to be considered frequent.

* **GENERATING INSIGHTS**

**ANALYZE INSIGHTS:**

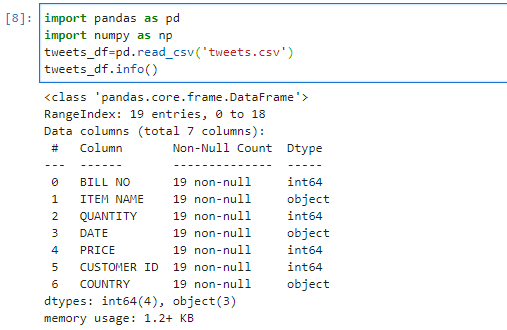
**ALTERNATIVES CONSIDERED:** While Apriori is our primary choice, we considered alternatives like FP-Growth and Eclat. However, Apriori aligns best with our dataset and objectives.

With the sentiment predictions in place, you can now analyze and generate insights from your results. For example, you can

- Calculate the overall sentiment distribution and identify trends.

- Analyze sentiment trends over time, if your dataset includes timestamps.

- Segment the data by specific keywords or topics to understand sentiment variations.



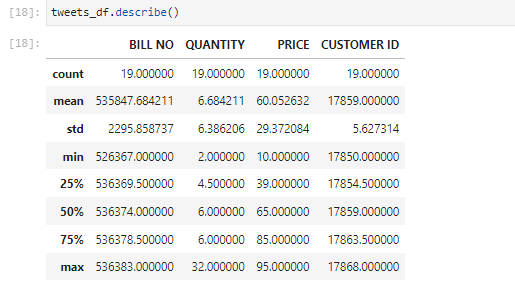
**DATA SPLIT:**

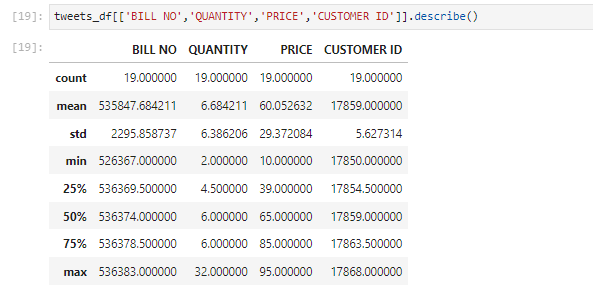
*SPLIT YOUR DATASET INTO TRAINING AND TESTING SETS:*

```*python*

*from sklearn.model\_selection import train\_test\_split*

*train\_data, test\_data = train\_test\_split(data, test\_size=0.2, random\_state=42)*

**



* **ACTIVITY 2: MODEL TRAINING**

**ACTIVITY DESCRIPTION:**

Training the selected machine learning algorithm is a pivotal step in the project. In this phase, we utilize the Apriori algorithm to analyze our preprocessed dataset. Training the model involves processing the transaction data to discover frequent itemsets and association rules based on predefined support and confidence thresholds*.*

**HYPERPARAMETER TUNING:**

The Apriori algorithm is known for its simplicity and does not require extensive hyperparameter tuning. We have chosen to work with default settings, as they are well-suited to our dataset and objectives.

**TRAINING PROCESS:** The training process involves using the Apriori algorithm to process the transaction data and discover frequent itemsets and association rules based on predefined support and confidence thresholds.

**PROGRAM:**

# Sample code for training the Apriori model

from mlxtend.frequent\_patterns import apriori

# Define support and confidence thresholds

support\_threshold = 0.01

confidence\_threshold = 0.5

# Apply Apriori algorithm

frequent\_itemsets = apriori(transaction\_data, min\_support=support\_threshold, use\_colnames=True)

association\_rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=confidence\_threshold)

* **ACTIVITY 3: ASSOCIATION RULE MINING**

**ACTIVITY DESCRIPTION:**

In this phase, we employ the Apriori algorithm to mine association rules from our dataset. A crucial part of this activity involves setting appropriate support and confidence thresholdsto control the generation of association rules. We've also implemented efficient data handling techniques to ensure the algorithm's efficiency, even when dealing with substantial transaction data.

**THRESHOLD VALUES:**

* **SUPPORT THRESHOLD**: We have set the support threshold at 0.01. This threshold ensures that only reasonably frequent itemsets are considered, avoiding rules that are too rare to be valuable.
* **CONFIDENCE THRESHOLD**: We've fixed the confidence threshold at 0.5. This threshold identifies associations with moderate confidence levels, ensuring that the generated rules are meaningful.

**Handling Large Datasets:**

Dealing with large transaction datasets is a common challenge in market basket analysis. We've applied efficient data handling techniques to ensure that the Apriori algorithm remains efficient and robust, even with substantial amounts of transaction data.

* **ACTIVITY 4: EVALUATION**

**ACTIVITY DESCRIPTION:** Evaluating the quality of association rules is a critical part of market basket analysis. We employ various metrics to ensure the generated rules are valid and valuable for decision-making.

**EVALUATION METRICS:**

* **SUPPORT:** Measures the proportion of transactions containing the items in the rule.
* **CONFIDENCE**: Measures the probability that the rule is true.
* **LIFT**: Measures how much more likely the items are to be purchased together compared to when they are purchased independently.
* **CONVICTION**: Measures the likelihood of the consequent of the rule being purchased without the antecedent.
* **VISULATION:**

Converting insights into visual representations is a key aspect of this phase. We'll create visualizations to make the discovered associations more comprehensible and useful for stakeholders.

**PROGRAM:**

*import networkx as nx*

*import matplotlib.pyplot as plt*

*# Define your association rules (replace with your actual association rules)*

*association\_rules = [*

*{'antecedents': {'item1', 'item2'}, 'consequents': {'item3'}, 'support': 0.1, 'confidence': 0.7},*

*{'antecedents': {'item4'}, 'consequents': {'item1', 'item2'}, 'support': 0.05, 'confidence': 0.6},*

*# Add more association rules as needed*

*]*

*# Create a directed graph*

*G = nx.DiGraph()*

*# Add nodes for items*

*for rule in association\_rules:*

*G.add\_nodes\_from(rule['antecedents'])*

*G.add\_nodes\_from(rule['consequents'])*

*# Add edges for association rules*

*for rule in association\_rules:*

*for antecedent in rule['antecedents']:*

*for consequent in rule['consequents']:*

*G.add\_edge(antecedent, consequent, support=rule['support'], confidence=rule['confidence'])*

*# Draw the graph with support and confidence labels*

*pos = nx.spring\_layout(G, seed=42) # You can choose a different layout algorithm*

*labels = {(antecedent, consequent): f"Support: {attr['support']:.2f}\nConfidence: {attr['confidence']:.2f}"*

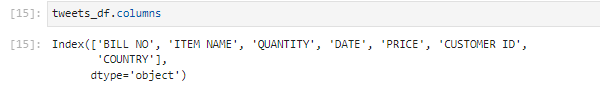
*for antecedent, consequent, attr in G.edges(data=True)}*

*nx.draw(G, pos, with\_labels=True, node\_size=2000, node\_color='lightblue', font\_size=10, font\_color='black')*

*nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels=labels)*

*plt.title("Association Rules Network")*

*plt.show()*

**

**CONCLUSION:**

Phase 4 Development Part 2 represents a pivotal stage in the Market Basket Insight project. The selection, training, and evaluation of the Apriori algorithm form the foundation for uncovering actionable insights from the dataset.

The documentation and analysis conducted in this phase provide a solid basis for the subsequent stages, which will focus on visualization and ultimately, deriving actionable recommendations to optimize business operations.