MARKET BASKET INSIGHTS

Phase 1:

Problem definition and design thinking

In this part you will need to understand the problem statement and create a document on what have you understood and how will you proceed ahead with solving the problem. Please think on a design and present in form of a document.

Problem Definition:

The problem is to perform market basket analysis on a provided dataset to unveil hidden patterns and associations between products. The goal is to understand customer purchasing behavior and identify potential cross-selling opportunities for a retail business. This project involves using association analysis techniques, such as Apriori algorithm, to find frequently co-occurring products and generate insights for business optimization.

Design Thinking:

1.Data Source: Choose a dataset containing transaction data, including lists of purchased products.

2.Data Preprocessing: Prepare the transaction data by transforming it into a suitable format for association analysis.

3.Association Analysis: Utilize the Apriori algorithm to identify frequent itemsets and generate association rules.

4.Insights Generation: Interpret the association rules to understand customer behavior and crossselling opportunities.

5.Visualization: Create visualizations to present the discovered associations and insights..

6.Business Recommendations: Provide actionable recommendations for the retail business based on the insights

Phase 2:

Innovation

1.Data Collection: Gather transaction data that includes information on items purchased, transaction IDs, and timestamps

This data can come from point-of-sale systems, e-commerce platforms, or any other relevant sources.

2. Data Preprocessing: Clean and preprocess the data to ensure accuracy. Remove duplicates, handle missing values, and format the data for analysis.

3. Basket Creation: Group transactions by unique transaction IDs to create "baskets" containing the items purchased together during each transaction.

4. Support and Confidence Calculation:

Support: Calculate the support for each itemset (combination of items) in the dataset. Support measures the frequency of occurrence of an itemset in the baskets.

Confidence: Calculate the confidence for association rules. Confidence measures the likelihood that if item A is purchased, item B will also be purchased./

5. Association Rule Mining:

- Use algorithms like Apriori or FP-Growth to discover association rules.

- Association rules consist of antecedents (items in the "if" part) and consequents (items in the "then" part). For example: {A} => {B}.

6.Filtering and interpretation:

- Set thresholds for support and confidence to filter out relevant rules. This helps focus on meaningful insights.

- Interpret the generated association rules to understand which products are frequently bought together. For example, you might find that customers who purchase milk are likely to buy bread as well

7. Visualization and Reporting:

- Create visualizations, such as scatter plots or network graphs, to represent the relationships between products.

- Generate reports that highlight actionable insights for merchandising, marketing, and inventory management teams.

8. Implementation:

- Implement the insights gained from market basket analysis into business strategies. This could involve optimizing store layouts, creating bundled promotions, or improving recommendation systems for e-commerce platforms.

9. Iterative Analysis:

- Continuously monitor and analyze market basket data to identify evolving trends and adapt strategies accordingly

Phase 3 :

Development part 1

Consider the following dataset and we will find frequent itemsets and generate association rules for them.

TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2

minimum confidence is 60%

Step-1: K=1

(I) Create a table containing support count of each item present in dataset – Called C1(candidate set)

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

(II) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.

Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

Step-2: K=2

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

Generate candidate set C2 using L1 (this is called join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common.

Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent.Check for each itemset)

Now find support count of these itemsets by searching in dataset.

(III) compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.

Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

Step-3:
Generate candidate set C3 using L2 (join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common. So here, for L2, first element should match.
So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}

Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2},{I2, I3},{I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)

find support count of these remaining itemset by searching in dataset.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

(II) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.

Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

Generate candidate set C4 using L3 (join step). Condition of joining Lk-1 and Lk-1 (K=4) is that, they should have (K-2) elements in common. So here, for L3, first 2 elements (items) should match.

Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent). So no itemset in C4

We stop here because no frequent itemsets are found further

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –

A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.

Confidence(A->B)=Support_count(A∪B)/Support_count(A)

So here, by taking an example of any frequent itemset, we will show the rule generation.
Itemset {I1, I2, I3} //from L3
so rules can be

[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%
[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%
[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%
[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%
[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%
[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

Phase 4:

Development Part 2

DATA COLLECTION

Gather transaction data that includes information on what items were purchased together. This can be obtained from point-of-sales system or e-commerce platforms

```
import pandas as pd

# Sample transaction data (replace with your dataset)
data = {
    'TransactionID': [1, 2, 3, 4, 5],
    'Items': ['A, B, D', 'B, C', 'A, B, C', 'A, D', 'B, C, D']
}

# Create a DataFrame from the data
df = pd.DataFrame(data)

# Split the 'Items' column into a list of items
df['Items'] = df['Items'].str.split(', ')

# Transform the data into a binary format (one-hot encoding)
basket = pd.get_dummies(df['Items'].apply(pd.Series).stack()).sum(level=0)

# Print the resulting dataset
print(basket)
```

DATA PREPROCESSING

It is a crucial step in market basket analysis. Below is a python code snippet that covers the some common data preprocessing tasks such as removing duplicate, handling missing values and encoding categorical data for market basket analysis.

```
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
data = {

    'TransactionID': [1, 2, 3, 4, 5],
    'Items': ['A, B, D', 'B, C', 'A, B, C', 'A, D', 'B, C, D']
}

# Create a DataFrame from the data
df = pd.DataFrame(data)
df['Items'] = df['Items'].str.split(', ')
basket = pd.get_dummies(df['Items'].apply(pd.Series).stack()).sum(level=0)
basket = basket.drop_duplicates()
frequent_itemsets = apriori(basket, min_support=0.1, use_colnames=True)
# Generate association rules
rules = association_rules(frequent_itemsets, metri="lift", min_threshold=1.0)
# Display frequent itemsets and association rules
print("Frequent Itemsets:")
print(frequent_itemsets)
print("\nAssociation Rules:")
print(rules)
```

FEATURE ENGINEERING

Feature engineering typically involve creating new features or transforming

Existing once to improve the performance of a machine learning model this specific code for feature engineering can vary widely depending on data set

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder

# Load your dataset into a Pandas DataFrame
data = pd.read_csv('your_data.csv')

# Example 1: Standardizing numeric features
numeric_features = ['bill no','date', 'customer id']
scaler = StandardScaler()
data[numeric_features] = scaler.fit_transform(data[numeric_features])

# Example 2: Encoding categorical features
categorical_features = ['']
encoder = OneHotEncoder()
encoded_features = encoder.fit_transform(data[categorical_features]).toarray()
encoded_feature_names = encoder.get_feature_names(categorical_features)
data = pd.concat([data, pd.DataFrame(encoded_features, columns=encoded_feature_names)], axis=1)
data.drop(categorical_features, axis=1, inplace=True)

# Example 3: Creating new features
data['age_squared'] = data['age'] ** 2
data['log_income'] = np.log(data['income'])
```

This is very basic example and feature for market basket analysis insights.

VISUALIZATION

Visualization data is essential part of data analysis and model interpretation. Here's an example of how to create basic visualization library Matplotlib .you'll need to have Matplotlip installed

```python
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [10, 15, 13, 18, 20]
plt.scatter(x, y)
plt.xlabel('X-axis label')
plt.ylabel('Y-axis label')
plt.title('Scatter Plot')
plt.show()
categories = ['Category A', 'Category B', 'Category C']
values = [25, 40, 30]
plt.bar(categories, values)
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Bar Chart')
plt.show()
x = [1, 2, 3, 4, 5]
y = [10, 15, 13, 18, 20]
plt.plot(x, y, marker='o', linestyle='-')
plt.xlabel('X-axis label')
plt.ylabel('Y-axis label')
plt.title('Line Plot')
plt.show()
```

This is the very basic example for visualization in market basket analysis insights.

EVALUATION

It is the performance of machine learning models is crucial for understanding how well they're doing .Here's is the basic example of how to evaluate a classification model using python and scikit-learn

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.ensemble import RandomForestClassifier
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
# Print the results
print(f'Accuracy: {accuracy:.2f}')
print('Confusion Matrix:')
print(conf_matrix)
print('Classification Report:')
print(class_report)
```