Q5 - Apriori on NYRESTAURANTS Dataset - Detailed Answers

Problem statement

Run Apriori algorithm to find frequent itemsets and association rules on NYRESTAURANTS dataset and use appropriate evaluation measures to compute correctness of obtained patterns.

Preprocessing transactions

1. Load dataset of ordered items per transaction. 2. Convert to a one-hot encoded dataframe with each column as an item and rows as transactions (True/False or 1/0). 3. Remove rare items or noise if necessary.

Apriori algorithm & metrics

Use mlxtend.frequent_patterns.apriori to find frequent itemsets with a minimum support. Then use mlxtend.frequent_patterns.association_rules to generate rules with metrics: support, confidence, lift, leverage, conviction. Evaluate by inspecting top rules by lift and support and by checking domain relevance (do rules make business sense?). Use holdout sample or temporal split to test rule stability.

Sample Python code

```
# Apriori on NYRESTAURANTS dataset (example)
import pandas as pd
from mlxtend.frequent_patterns import apriori, association_rules
from sklearn.model_selection import train_test_split

# assume we have a one-hot encoded transactions dataframe 'trans' (columns are menu items)
# trans = pd.read_csv('ny_restaurant_transactions_onehot.csv')

frequent_itemsets = apriori(trans, min_support=0.01, use_colnames=True)
rules = association_rules(frequent_itemsets, metric='confidence', min_threshold=0.5)

# sort and inspect
rules_sorted = rules.sort_values(['lift','confidence'], ascending=[False, False])
print(rules_sorted[['antecedents','consequents','support','confidence','lift']].head(20))

# Evaluation: use support/confidence/lift thresholds, and if possible compute rule stability on
train, test = train_test_split(trans, test_size=0.3, random_state=42)
fi_train = apriori(train, min_support=0.01, use_colnames=True)
rules_train = association_rules(fi_train, metric='confidence', min_threshold=0.5)
```

Evaluation guidance

Interpretation: high lift (>1) indicates positive association, lift >>1 strong. Use domain knowledge: some co-occurrences are obvious (e.g., 'burger' with 'fries'). To measure correctness quantitatively, compute precision-like measure by verifying rules on held-out transactions (fraction of antecedent occurrences where consequent also occurs).