Hashir khalid

08

BS (IT)

Apriori & Association Rules

Write a paragraph stating whether apriorpi algorithm and association rules is the most suitable tool for giving the given data. The suitability of the Apriori algorithm and association rules as a tool for analyzing the given data depends on the nature and characteristics of the data itself. Apriori algorithm is a widely used method for mining frequent itemsets in large datasets, making it suitable for identifying patterns and relationships between items. Association rules, derived from the Apriori algorithm, help in uncovering meaningful associations and dependencies among variables. If the given data consists of transactions or categorical variables where the focus is on identifying frequent itemsets and association rules, then the Apriori algorithm and association rules would be a suitable choice. However, if the data contains numerical variables or requires complex analysis beyond itemset mining, other techniques or algorithms might be more appropriate. Ultimately, the selection of the most suitable tool depends on the specific characteristics and objectives of the given data analysis task.

Import pandas as pd

From mlxtend.frequent\_patterns import apriori, association\_rules

Data = pd.read\_csv(‘data.csv’)

Data = data.replace(‘A’, 0)

Grading\_scale = [

(85, ‘A+’),

(80, ‘A’),

(75, ‘B+’),

(65, ‘B’),

(58, ‘C+’),

(55, ‘C-‘),

(50, ‘D’),

(0, ‘F’)

]

Def convert\_marks\_to\_grade(marks):

Marks = int(marks)

For threshold, grade in grading\_scale:

If marks >= threshold:

Return grade

Data[‘FOP’] = data[‘FOP’].apply(convert\_marks\_to\_grade)

Data[‘OOPS’] = data[‘OOPS’].apply(convert\_marks\_to\_grade)

Data\_encoded = pd.get\_dummies(data[[‘FOP’, ‘OOPS’]])

Frequent\_itemsets = apriori(data\_encoded, min\_support=0.1, use\_colnames=True)

Rules = association\_rules(frequent\_itemsets, metric=’confidence’, min\_threshold=0.6)

Rules = rules.sort\_values(by=’confidence’, ascending=False)

Most\_frequent\_pair = frequent\_itemsets.sort\_values(by=’support’, ascending=False).iloc[0]

Print(“Most Frequent Pair of grades:”)

Print(most\_frequent\_pair)

Print(“\nStrong Association Rules:”)

For index, rule in rules.iterrows():

Antecedents = ‘, ‘.join(rule[‘antecedents’])

Consequents = ‘, ‘.join(rule[‘consequents’])

Confidence = rule[‘confidence’]

Print(f”{antecedents} → {consequents} (confidence: {confidence})”)