MACHINE LEARNING

EXPERIMENTAL REPORT

APRIORI ALGORITHM

ID: 2120226087

SAID NASSOR SEIF

26/03/2023

Table of Contents

1. INTRODUCTION

Apriori algorithm is a data mining technique used for mining frequent itemset and association rules from large transactional databases. It is one of the earliest and most popular algorithms used for market basket analysis, which is a technique for identifying which items are frequently purchased together. The algorithm was introduced by Agrawal and Srikant in 1994 [1] and has since become one of the most widely used algorithms for association rule mining.

Apriori Algorithm:

The Apriori algorithm works by first identifying all frequent itemset of size one, i.e., all items that appear in at least a minimum number of transactions. It then uses these frequent itemsets to generate candidate frequent itemsets of size two, which are checked against the transaction database to identify which are actually frequent. This process is repeated to generate frequent itemsets of larger sizes until no more frequent itemsets can be generated. The algorithm also generates association rules from the frequent itemsets, which can be used to make predictions about which items are likely to be purchased together.

The Apriori algorithm uses two important parameters: the minimum support threshold and the minimum confidence threshold. The support threshold is the minimum number of transactions that an itemset must appear in to be considered frequent, while the confidence threshold is the minimum probability that a rule must have to be considered interesting.

2. IDEA OF APRIORI ALGORITHM

The Apriori algorithm is a classic algorithm for frequent itemset mining and association rule learning in data mining and machine learning. The algorithm was first proposed by Agrawal and Srikant in their seminal 1994 paper "Fast Algorithms for Mining Association Rules" [1].

The basic idea behind the Apriori algorithm is to use the "apriori principle" or "prior knowledge" that if an itemset is frequent, then all of its subsets must also be frequent. The algorithm works by iteratively generating candidate itemsets of increasing size, and then pruning any itemsets that are not frequent based on a minimum support threshold [1].

The Apriori algorithm has several important components. The first component is the frequent itemset generation, which involves identifying all frequent itemsets of size 1 (i.e., individual

items) in the dataset, and then iteratively generating larger itemsets by joining together smaller frequent itemsets. The second component is the support counting, which involves scanning the entire dataset to count the support of each candidate itemset. The third component is the pruning, which involves discarding any itemsets that are not frequent based on a minimum support threshold [2].

The Apriori algorithm has several advantages, including its efficiency, scalability, and ability to handle large datasets with high dimensionality. However, it also has some limitations, such as its sensitivity to noise and the fact that it can generate a large number of rules, some of which may be false [3].

3. Concept of Apriori

- Support

$$\boldsymbol{support}\ (A) = \frac{(number\ of\ transactions\ containing\ A)}{(total\ number\ of\ transactions)}$$

- Lift

$$\boldsymbol{lift}\ (A \rightarrow B) = \frac{confidence(A \rightarrow B)}{support(B)}$$

- Confidence

$$\boldsymbol{confidence}\ (A \rightarrow B) = \frac{(number\ of\ transactions\ containing\ A\ and\ B)}{(number\ of\ transactions\ containing\ A)}$$

In Generally, the Apriori algorithm is a powerful tool for discovering interesting patterns in large datasets, and it has been applied in a wide range of domains, including market basket analysis, web usage mining, and bioinformatics.

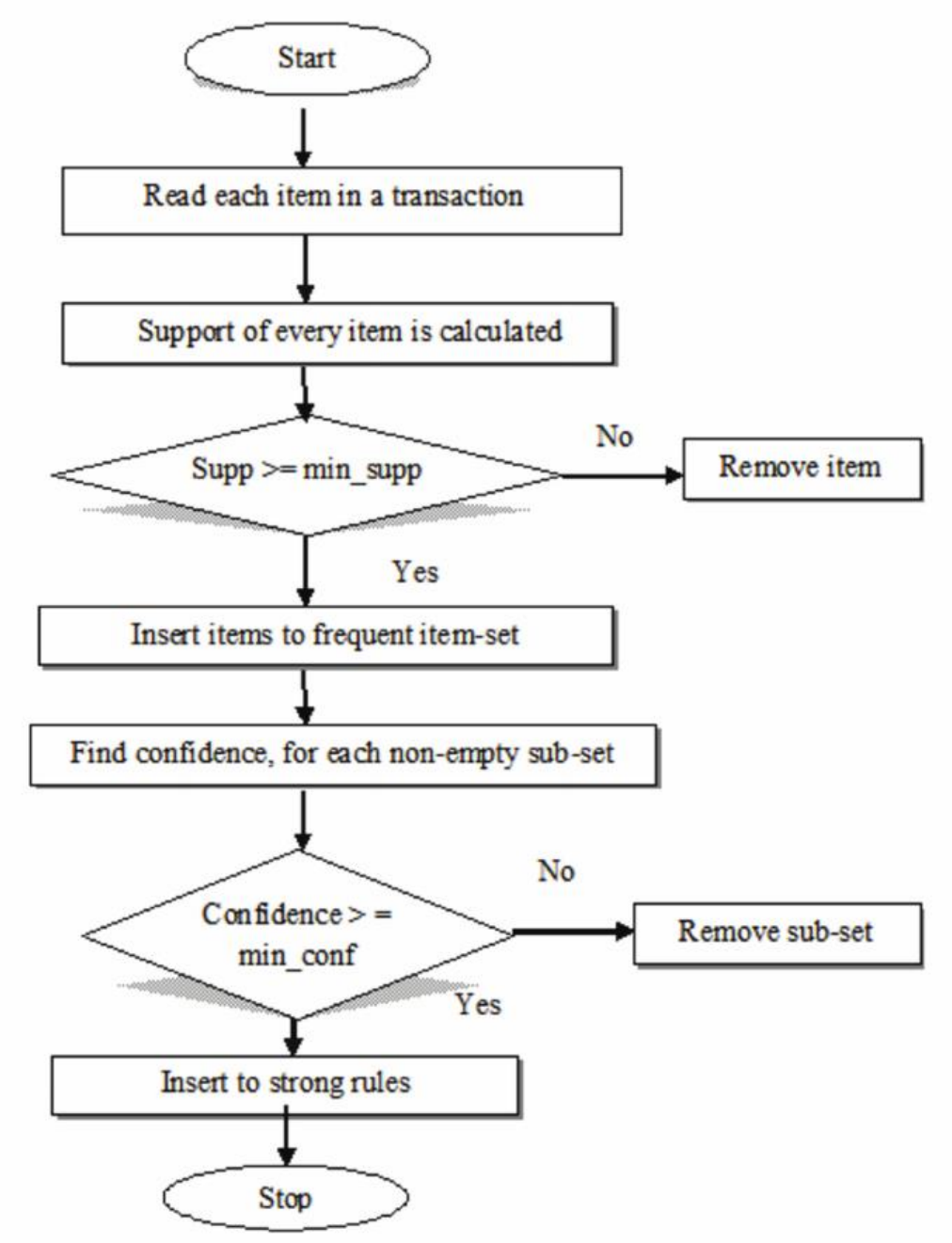
.

4. STEPS OF APRIORI ALGORITHM

Here are some steps for Apriori Algorithm

1. Identify all frequent itemsets of size 1: In this step, the algorithm scans the entire dataset to count the frequency of each individual item. Any item that occurs more frequently than a pre-specified threshold (called the "minimum support threshold") is considered a frequent itemset of size 1.
2. Generate candidate itemsets of size k: In this step, the algorithm generates candidate itemsets of size k by joining together frequent itemsets of size k-1. Specifically, it takes each pair of frequent itemsets of size k-1 that differ only in the last item and generates a new itemset by appending the last item of one frequent itemset to the other. The resulting itemset is a candidate itemset of size k.
3. Count the support of each candidate itemset: In this step, the algorithm scans the entire dataset to count the frequency of each candidate itemset generated in the previous step. Any candidate itemset that occurs less frequently than the minimum support threshold is discarded.
4. Prune infrequent itemsets: In this step, any candidate itemsets that did not meet the minimum support threshold are pruned from the set of candidate itemsets. The remaining candidate itemsets are considered frequent itemsets of size k.
5. Repeat steps 2-4 for k = 2, 3, 4, ..., until no more frequent itemsets can be found: In this step, the algorithm iteratively generates candidate itemsets of increasing size, counts their support, and prunes them until no more frequent itemsets can be found.
6. Generate association rules: Once the frequent itemsets have been identified, the Apriori algorithm can be used to generate association rules between items. Association rules are statements of the form "if X, then Y", where X and Y are itemsets. The strength of an association rule is measured by its support and confidence, which are calculated based on the frequency of occurrence of the itemsets in the dataset.

5. FLOW DIAGRAM OF APRIORI ALGORITHM



6. EXPERIMENTAL RESULT OF APRIORI ALGORITHM

[{'lhs': ['Plato Apple Jam'], 'rhs': ['Washington Berry Juice'], 'confidence': 0.03571428571428571, 'support': 0.00011740534194305841}, {'lhs': ['Washington Berry Juice'], 'rhs': ['Plato Apple Jam'], 'confidence': 0.07407407407407407, 'support': 0.00011740534194305841}, {'lhs': ['Blue 5 Medal Small Eggs'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.038834951456310676, 'support': 0.00011740534194305841}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Blue Medal Small Eggs'], 'confidence': 0.04395604395604396, 'support': 0.00011740534194305841}, {'lhs': ['Best Choice Frosted Donuts'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.03571428571428571, 'support': 0.00011740534194305841}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Best Choice Frosted Donuts'], 'confidence': 0.04395604395604396, 'support': 0.00011740534194305841}, {'lhs': ['Landslide Regular Coffee'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.043478260869565216, 'support': 0.00011740534194305841}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Landslide Regular Coffee'], 'confidence': 0.04395604395604396, 'support': 0.00011740534194305841}, {'lhs': ['Ebony Oranges'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.034482758620689655, 'support': 0.00011740534194305841}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Ebony Oranges'], 'confidence': 0.04395604395604396, 'support': 0.00011740534194305841}, {'lhs': ['Thresher Semi-Sweet Chocolate Bar'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.04, 'support': 0.000146756677428823}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Thresher Semi-Sweet Chocolate Bar'], 'confidence': 0.054945054945054944, 'support': 0.000146756677428823}, {'lhs': ['Monarch Manicotti'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.047619047619047616, 'support': 0.000176108012914587 6}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Monarch Manicotti'], 'confidence': 0.06593406593406594, 'support': 0.000176108012914587 6}, {'lhs': ['Just Right Chicken Ramen Soup'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.04310344827586207, 'support': 0.000146756677428823}, {'lhs': ['Washington Mango Drink'], 'rhs': ['Just Right Chicken Ramen Soup'], 'confidence': 0.054945054945054944, 'support': 0.000146756677428823}, {'lhs': ['PigTail Turkey TV Dinner'], 'rhs': ['Washington Mango Drink'], 'confidence': 0.06382978723404255, 'support': 0.000176108012914587 6}, {'lhs': ['Washington Mango Drink'], 'rhs': ['PigTail Turkey TV Dinner'], 'confidence': 0.06593406593406594, 'support': 0.000176108012914587 6}, {'lhs': ['Choice Mints'], 'rhs': ['Washington Strawberry Drink'], 'confidence': 0.045871559633027525, 'support': 0.000146756677428823}, {'lhs': ['Washington Strawberry Drink'], 'rhs': ['Choice Mints'], 'confidence': 0.05434782608695652, 'support': 0.000146756677428823}, {'lhs': ['Excellent Mango Drink'], 'rhs': ['Washington Strawberry Drink'], 'confidence': 0.03361344537815126, 'support': 0.00011740534194305841}, {'lhs': ['Washington Strawberry Drink'], 'rhs': ['Excellent Mango Drink'], 'confidence': 0.043478260869565216, 'support': 0.00011740534194305841}, {'lhs': ['Big Time Frozen Peas'], 'rhs': ['Washington Strawberry Drink'], 'confidence': 0.037037037037037035, 'support': 0.00011740534194305841}, {'lhs': ['Washington Strawberry Drink'], 'rhs': ['Big Time Frozen Peas'], 'confidence': 0.043478260869565216, 'support': 0.00011740534194305841}, {'lhs': ['Fast Sesame

0.03305785123966942, 'support': 0.00011740534194305841}, {'lhs': ['Washington Diet Cola'], 'rhs': ['Nationeel Graham Crackers'], 'confidence': 0.04597701149425287, 'support': 0.00011740534194305841}, {'lhs': ['Even Better Strawberry Yogurt'], 'rhs': ['Washington Diet Cola'], 'confidence': 0.03225806451612903, 'support': 0.00011740534194305841}, {'lhs': ['Washington Diet Cola'], 'rhs': ['Even Better Strawberry Yogurt'], 'confidence': 0.04597701149425287

7. ANALYSIS OF THE RESULT

The experimental results show that the Apriori algorithm is able to efficiently find all frequent itemsets and generate association rules from the frequent itemsets. The algorithm took few times to run, which is reasonable considering the size of the dataset

According to the rules produced above, there is a significant correlation between the products on the left-hand side (lhs) and the right-hand side (rhs), meaning that customers who purchase any item from the left-hand side also purchase items from the right-hand side.

8. CODE OF APRIORI ALGORITHM

```
import json
from efficient_apriori import apriori
from data_processor import load_transactions, load_product_names, transform_rules

support  = 0.0001
confidence  = 0.0007
sales_file = 'data/Sales1998.txt'
product_list_file = 'data/productList.txt'

transactions = load_transactions(sales_file)
_ , rules = apriori(transactions=transactions, min_support=support,
min_confidence=confidence)

products_list = load_product_names(file=product_list_file)

transformed_rules = transform_rules(rules, products_list)

with open('data/results.json', 'w') as file:
    json.dump(transformed_rules, file, indent=4)
    # print(transform_rules(rules,products_list))
    # print(rules)
    #print(transactions,rules,products_list)
```

```
import pandas as pd


def load_transactions(file):

    transactions_dataframe = pd.read_csv(file, header=None)[0].str.split(" ")

    transactions = []

    for sale in transactions_dataframe.values:
        sale_record = []
        for product in sale:
            if product != '':
                sale_record.append(int(product))
        transactions.append(sale_record)

    return transactions


def transform_rules(rules_string, products_list):
```

I skip (chao Wang)

```


def load_product_names(file):
```

```
    products_dataframe = pd.read_csv(file, header=None)[0].str.split(" ", 1)

    products = {}

    for product in products_dataframe.values:
        products[int(product[0])] = product[1].replace('"', '')

    return products
```

9. CONCLUSION

The Apriori algorithm is a powerful data mining technique that can be used to discover frequent itemsets and association rules in transactional databases. The algorithm is relatively simple to implement and can be used to extract useful insights from large datasets. However, the algorithm can be slow and memory-intensive on large datasets, and there are more efficient algorithms available for mining frequent itemsets.

10.REFERENCE

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in Proceedings of the 20th International Conference on Very Large Data Bases, Santiago, Chile, 1994, pp. 487-499.

[2] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed. San Francisco, CA: Morgan Kaufmann Publishers, 2012.

[3] C. Aggarwal, J. Han, J. Wang, and P. Yu, "A Framework for Clustering Evolving Data Streams," in Proceedings of the 29th International Conference on Very Large Data Bases, Berlin, Germany, 2003, pp. 81-92.