Pattern Discovery with Web usage Mining using Apriori and FP-Growth Algorithms

Kondi Srujan Kumar#1, M Ashish Naidu#2, K Radha#3
#1, 2 III-B.TECH-CSE, Rudraram, GITAM University, Hyderabad, Telangana, India

Abstract

In Data Mining, Association Rule Mining is a standard and well researched technique for finding out the relations between variables in large datasets. Association rule is used as a precursor to different Data Mining techniques like classification, clustering and prediction. The aim of the paper is to compare the performance of the Apriori algorithm and Frequent Pattern growth algorithm by comparing their capabilities and Pros and cons of Apriori and FP-Growth Algorithms. The evaluation study shows that the FP-growth algorithm is efficient than the Apriori algorithm. This Paper Presents about the Pattern discovery from weblog data using web usage mining. Top-down approach in mining frequent item sets.

Keywords - Apriori, FP Growth, Classification, Prediction

I. INTRODUCTION

Association Mining aims to extract correlations, frequent patterns, and association structures which are attention-grabbing among set of things or objects in transaction data based relational databases or different data repositories. Two statistical measures that govern Association Rule Mining are Support and Confidence. Support should be measured as to how often it should occur in the database. Confidence may well be gauged to seek out the strength of the rule. The Association rules are interesting if they satisfy each a minimum Support threshold and a minimum Confidence threshold [2].

This paper aims to present a performance evaluation of Apriori and FP-growth algorithms. The distinction between the two algorithms is that the Apriori algorithm generates candidate frequent itemsets and also the FP-growth algorithm avoids candidate generation and it develops a tree by economical and efficient 'divide and conquer' strategy.

A. Data for Association Rules

Association models are designed to use transactional data. Nulls in transactional data are assumed to represent values that are known but not present in the transaction. For example, three items out of hundreds of possible items might be purchased in a single transaction. The items that were not purchased are known but not present in the transaction. Transactional data, by its nature, is sparse. Only a small fraction of the attributes are non-zero or non-null in any given row. Apriori interprets all null values as indications of sparsity.

B. Association Rule Mining

An Association rule is an expression of the form X → Y means that whenever X seems, Y also tends to appear. X and Y are itemsets. An itemsets is nothing but a collection of database items. X is usually stated as the rule’s antecedent and Y as the consequent of the rule. Association rules are stated as Boolean rules encompassing with Support and Confidence. Support is the proportion of transactions in an exceedingly information that satisfy the rule. Confidence denotes the chance of Y being a true subject to X or P (Y|X).

Association Rule Mining is usually split up into two separate steps as stipulated below.

1. Find all frequent itemset: An itemset that happens, a minimum as often as a planned minimum Support count.
2. Generate strong Association rules from the frequent Itemset: The rules should satisfy minimum Support and minimum Confidence.

Advantages of FP-Growth Algorithm

These are the pros of Fp-growth:
- There are only 2 passes over data-set
- This algorithm Compresses data-set
- There is no candidate generation.
- It is much faster than apriori

Advantages of Apriori Algorithm

These are the pros of apriori algorithm:
- It is easy to understand and implement.
- These are the cons of Apriori algorithm.
- When you need a large number of candidate rules then this algorithm is computationally expensive.
- It is also an expensive method to calculate support because the calculation has to go through the entire database.

II. APRIORI ALGORITHM

Apriori algorithm, a classic algorithm, is useful in mining frequent itemsets and relevant association rules. Usually, you operate this algorithm on a
database containing a large number of transactions. One such example is the items customers buy at a supermarket. It helps the customers buy their items with ease, and enhances the sales performance of the departmental store. This algorithm has utility in the field of healthcare as it can help in detecting adverse drug reactions (ADR) by producing association rules to indicate the combination of medications and patient characteristics that could lead to ADRs [3]. Three significant components comprise the apriori algorithm. They are as follows.

- Support
- Confidence
- Lift

**SUPPORT:** Support is the ratio of transactions that include all the items in the antecedent and consequent to the number of total transactions. Support can be expressed in probability notation as follows.

\[
\text{Support} = \frac{n_s}{n_t}
\]

**CONFIDENCE:** Confidence is the ratio of the rule support to the number of transactions that include the antecedent. Confidence can be expressed in probability notation as follows.

\[
\text{Confidence} = \frac{\text{Support}(A \implies B)}{\text{Support}(A)}
\]

**LIFT:** Lift indicates the strength of a rule over the random co-occurrence of the antecedent and the consequent, given their individual support. It provides information about the improvement, the increase in probability of the consequent given the antecedent. Lift is defined as follows.

\[
\text{Lift} = \frac{\text{Confidence}}{P(B)}
\]

A. **Steps For Apriori Algorithm**

- **Step 1:** Scan the transaction database to get the support S of each 1-itemset, compare S with min_sup, and get a set of frequent 1-itemsets, L1
- **Step 2:** Use L k-1 join L k-1 to generate a set of candidate k-itemsets. And use Apriori property to prune the unfrequented k-itemsets from this set
- **Step 3:** Scan the transaction database to get the support S of each candidate k-itemset in the final set, compare S with min_sup, and get a set of frequent k-itemsets, L k
- **Step 4:** The candidate set = Null N
  - If candidate set = Null then repeat step 2
- **Step 5:** For each frequent itemset I, generate all nonempty subsets of I
- **Step 6:** For every nonempty subset s of I, output the rule “s => (I-s)” if confidence C of the rule “s => (I-s)” (=support S of I/support S of s) ³ min_conf

---

**The following methods can be used to improve the efficiency of apriori algorithm**

- **Transaction reduction** – A transaction not containing any frequent k-itemset becomes useless in subsequent scans.
- **Hash-based Itemset Counting** – Exclude the k-itemset whose corresponding hashing bucket count is less than the threshold is an infrequent itemset.

**B. Fp Growth Algorithm**

Fp growth algorithm (frequent pattern growth). Fp growth algorithm is an improvement of apriori algorithm. Fp growth algorithm used for finding frequent itemset in a transaction database without candidate generation. Fp growth represents frequent items in frequent pattern trees or FP-tree. FP-Tree is constructed using 2 passes over the dataset:

- **Pass 1**:
  - Scan data and find support for each item
  - Discard infrequent items
  - Sort frequent items in decreasing order based on their support

Use this order when building the FP-Tree, so common prefixes can be shared.

- **Pass 2**:
  - Nodes correspond to items and have a counter
  - FP-Growth reads 1 transaction at a time and maps it to a path
  - Fixed order is used, so paths can overlap when transactions share items (when they have the same prefix)
  - In this case, counters are incremented

Pointers are maintained between nodes containing the same item, creating singly linked lists (dotted lines)

- The more paths that overlap, the higher the compression. FP-tree may fit in the memory.

Generating FP-Trees Pseudocode

The algorithmic program works as follows:

1. Scan the transaction database once, as among the Apriori algorithmic program, to seek out all the frequent items and their Support.
2. Sort the frequent items in descending order of their Support.
3. Initially, begin making the FP-tree with a root “null”.
4. Get the primary transaction from the transaction database. Takeaway all non-frequent items and list the remaining items in line with the order among the sorted frequent items.
5. Use the transaction to construct the primary branch of the tree with each node corresponding to a frequent
item and showing that item’s frequency that’s one for the primary transaction.
6. Get the next transaction from the transaction database. Takeaway all non-frequent items and list the remaining items in line with the order among the sorted frequent items.
7. Insert the transaction within the tree using any common prefix that may appear. Increase the item counts.
8. Continue with Step 6 until all transactions among the database are processed.

C. Discrimination Of Apriori V/S Fp-Growth

Various Comparisons are explained below with the help of different parameters for Apriori and FP-Growth Algorithms [1], [2].

<table>
<thead>
<tr>
<th>SNO</th>
<th>Parameter</th>
<th>Apriori</th>
<th>FP-Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Technique</td>
<td>Generates singletons, pairs, triplets, etc.</td>
<td>Insert sorted items by frequency into a pattern tree.</td>
</tr>
<tr>
<td>2</td>
<td>Runtime</td>
<td>Candidate generation is extremely slow. Runtime increases exponentially depending on the number of different items</td>
<td>Runtime increases linearly depending on the number of transactions and items</td>
</tr>
<tr>
<td>3</td>
<td>Memory Usage</td>
<td>Saves singletons, pairs, triplets, etc.</td>
<td>Stores a compact version of the database</td>
</tr>
<tr>
<td>4</td>
<td>Parallelizability</td>
<td>Candidate generation is very parallelizable</td>
<td>Data are very interdependent, each node needs the root</td>
</tr>
<tr>
<td>5</td>
<td>Search type</td>
<td>Breadth first search</td>
<td>Divide and conquer</td>
</tr>
<tr>
<td>6</td>
<td>Database</td>
<td>Sparse/dense datasets</td>
<td>Large and medium datasets</td>
</tr>
</tbody>
</table>

Table-1: Comparisons Of Apriori And Fp-Growth Algorithms

III. PATTERN DISCOVERY FROM WEBLOG

Web usage mining, is the method of mining which is used for user browsing and access patterns [4]. At the side of website, to identify the web users to capture the data along with their browsing behaviors, weblog mining is used. This paper mainly aims to describe about user behavior in classifying the patterns of the browsing and navigation data of web users and also measure the performance of the Frequent Pattern Growth algorithm and Apriori algorithm by comparing their performances.

The Apriori algorithm and FP Growth algorithm are compared by applying the rapid miner tool to discover frequent user patterns along with user behavior in the web log. Both the algorithms help to analyze the patterns of web site usage and the features of user behavior knowledge obtained from web usage. For more effective browsing, introduce personalization and to enhance the web design fp growth can be used. The results through this experiment mainly focus on the instances and time taken for execution calculated on the two algorithms. In terms of time complexity FP-growth gives better performance.

The weblog mining is used on huge weblog files sources to discover automatically and analyze wealth of useful emerging and user’s behavioral patterns. It is a tremendous task to identify the task of website users.

A. Top-down approach in mining frequent item sets

The improved FP-growth algorithm is based on top down approach i.e. TD-IFP-Growth is introduced [6]. Item name, count, node link and flag are four attributes of improved FP-growth algorithm. Node link is used to link the nodes with identical items which help to look for a specific node rapidly. Thus TD-IFP-Growth algorithm quickly traverses the tree. In the existing FP-Growth algorithm, the FP-tree adopts top down approach in which conditional pattern base and sub-FP-trees are generated. Whereas the proposed TD-IFP-Growth overcomes the problem of existing FP-Growth algorithm by searching the FP-tree which is quite opposite to the FP-Growth algorithm. The TD-IFP-Growth algorithm consumes less amount of time and memory because it will not generate conditional pattern.

B. Different approaches for frequent itemset mining

Data mining means to retrieve hidden analytical information from huge databases, it is helping the organizations as they focus only on essential information in their data warehouses. Data mining tools are used for future development and performances, allowing the organizations to create proactive, idea for decision making systems. The traditional data mining problem is Frequent Itemset Mining as it requires huge computations and input and output traffic capacity. One of the approaches which runs on Hadoop cluster is one of the recent popular distributed frameworks which focus on parallel processing. The proposed framework extends the characteristics of Apriori algorithm which is related to the frequent item set invention. The performance and scalability are highly improved when compared to the existing approaches. The algorithm which is proposed is tested on large datasets distributed system on heterogeneous cluster [5].

IV. CONCLUSION

Data mining means to retrieve hidden analytical information from huge databases, it is helping the organizations as they focus only on essential information in their data warehouses. Data mining tools are used for future development and performances, allowing the organizations to create proactive, idea for decision making systems. Association Mining aims to extract correlations, frequent patterns, and association structures which are attention-grabbing among set of things or objects in transaction data based relational databases or different
data repositories. Two statistical measures that govern Association Rule Mining are Support and Confidence. Support should be measured as to how often it should occur in the database. Confidence may well be gauged to seek out the strength of the rule. The Association rules are interesting if they satisfy each a minimum Support threshold and a minimum Confidence threshold.

Web usage mining is the method of mining which is used for user browsing and access patterns. At the side of website, to identify the web users to capture the data along with their browsing behaviors, web log mining is used. This paper mainly aims to describe about user behavior in classifying the patterns of the browsing and navigation data of web users and also measure the performance of the Frequent Pattern Growth algorithm and Apriori algorithm by comparing their performances.

REFERENCES

[4] Different Approaches for Frequent Itemset Mining P.V. Nikam1*, D.S. Deshpande
[5] Improved FP-Growth Algorithm Based on Top Down Approach in Mining Frequent Itemsets