COMPARATIVE ANALYSIS OF APRIORI AND FP-GROWTH ALGORITHMS FOR FREQUENT PATTERN MINING USING APACHE SPARK

[1]Shan Ahmed Usmani, [2]Syed Khurram Kamran, [3]Muhammad Zeeshan, [4]Noman Islam, [5]Zamin Ali Khan

[1,2,3]Department of Computer Science and Information Technology, NEDUET, Karachi, Pakistan.

[4,5]Iqra University, Karachi, Pakistan

Abstract

Looking for frequent patterns from the large database in Data Mining (DM) is a difficult task and researchers are working extensively to develop novel algorithms. In recent years, a number of scalable and effective algorithms for frequent itemsets mining (for big data analytics) have been proposed by many researchers. The two most efficient algorithms used for this purpose are FP-Growth and Apriori. Apriori algorithm candidate itemsets and tests if they are frequent. FP-Growth technique generates a tree dynamically and uses pattern fragment growth to unearth the frequent itemsets from the dataset. The purpose of this work is to discover the best among these two algorithms, while using the widely used big data platform "Apache Spark". The paper has employed Istacart open-source data available on the Kaggle website, having approximately 3 millions transactions. The performance has been evaluated on the basis of execution time, cost, and number of scans. It is found that FP-Growth is many times faster and less resource consuming than apriori algorithm.

Keywords: association rules mining, FP-Growth, apriori, apache spark

Introduction

Association rules mining algorithm has been extensively used by online websites to recommend potential products to customers. It is based on market basket analysis and works on the assumption that shopping habits of persons are common. Itemsets can be used to recommend products to customers. There are many algorithms available for market basket analysis, in which FP growth and Apriori are the most common and efficient ones. The objective of this study is the comparative analysis between them by using the Apache spark. Apache spark a data processing framework that can quickly perform processing tasks on very large data sets and can also distribute data processing tasks across multiple computers, either on its own or in order with other distributed computing tools [13]. The Resilient Distributed Datasets (RDDs) are used for outcomes storage in memory. The fragment growth pattern is utilized in FP-Growth in order to extract frequent patterns from a large dataset. On the other hand, frequent itemsets are generated in the Apriori algorithm and all of its subgroups must also be frequent. Apriori algorithm generates candidate itemsets and tests if they are frequent. This paper analyzes the performance of both FP-Growth and Apriori algorithms on the basis of execution time, accuracy, number of scans, and the cost of computation.

Literature review

A number of studies have employed association rules mining in various studies. In [1], FP-Growth algorithm was developed for big data platforms. In another work, Guided FP-Growth algorithm was proposed for big data platform [2]. An improvement of FP-Growth algorithm was proposed in [3]. In several studies comparative studies have been provided. For instance, a comparison of FP-Growth and apriori was performed for big data platforms in [4]. A comparative analysis of FP-Growth and apriori for big data platforms have been performed in [6]. A comparison of free python libraries for data mining and big data analytics was provided in [7].

Besides, several implementation for FP-Growth and apriori algorithms have been provided. An improved FP-Growth algorithm for apache spark has been proposed in [5]. A map-reduce based FP-Growth mining algorithm was presented in [8]. In [9], a parallel version of FP-Growth called S-FPG was proposed for spark. A caching based FP-Growth implementation for spark as proposed in [10].

FP-Growth and apriori have been used in several studies for miscellaneous case studies. In one study, FP-Growth was employed for cache replacement in ad hoc networks [11]. In another study [12], FP-Growth algorithm was employed for service discovery in mobile ad hoc network.

The objective of this study is to perform a comparative analysis of Apriori and FP growth algorithm using PySpark on python programing. It has been seen that there are studies performed on the evaluation of both algorithm using many platforms like python, C++, R programing and hadoop (HDFS) etc., but the authors couldn't find the analysis on big data analytics platform apache spark. Therefore, this study performed comparison of both algorithms using PySpark.

It has been observed that most pattern mining algorithms are usually implemented on hadoop or undistributed systems. Considering the limitations in map reduce, the authors were encouraged to use spark for comparison of pattern mining algorithms. Rest of the sections of this paper is organized as follows. The methodology is presented next which is followed by results. The paper concludes with summarizing the research and avenues for future research.

Methodology

This section presents the methodology of the proposed work. It comprises a number of phases such as data collection, preprocessing and implementation.

Data Collection

The data for this study was collected from kaggle.com, as no market or departmental store would allow to use proprietary data. The data is about Instacart market (of size 197mb) having 32,434,489 rows. The data was available in the form of excel files as shown below.

order_id	product_id	add_to_cart_order	reordered
2	33120	1	1
2	28985	2	1
2	9327	3	0
2	45918	4	1
2	30035	5	0
2	17794	6	1
2	40141	7	1
2	1819	8	1
2	43668	9	0
3	33754	1	1
3	24838	2	1
3	17704	3	1
3	21903	4	1
3	17668	5	1
3	46667		
3	17461		

product_id	product_name	aisle_id	department_id
1	Chocolate Sandwich Cookies	61	19
2	All-Seasons Salt	104	13
3	Robust Golden Unsweetened Oolong Tea	94	7
4	Smart Ones Classic Favorites Mini Rigatoni Wi	38	1
5	Green Chile Anytime Sauce	5	13
6	Dry Nose Oil	11	11
7	Pure Coconut Water With Orange	98	7
8	Cut Russet Potatoes Steam N' Mash	116	1
9	Light Strawberry Blueberry Yogurt	120	16
10	Sparkling Orange Juice & Prickly Pear Beverage	115	7
11	Peach Mango Juice	31	7
12	Chocolate Fudge Layer Cake	119	1
13	Saline Nasal Mist	11	11
	Cleaner	74	17
		56	18

Table1: Raw Data

Preprocessing

There were some pre-processing required before using the data for association rules mining. There were two main files. One file contains the order id and product id. In this file there were multiple rows having same order id but different product ids. While, the second file contains product names along with their ids. There are 49689 unique products while there are 11720 different orders. To apply the algorithms, one needs the data in a text file and with all products of a distinct order arranged in a single row separated by ",". So,this work first preprocess the dataand then merged both the files on the basis of product id. Table 2 shows the result.

	order_id	product_id	add_to_cart_order	reordered	product_name
0	2	33120	1	1	Organic Egg Whites
1	2	28985	2	1	Michigan Organic Kale
2	2	9327	3	0	Garlic Powder
3	2	45918	4	1	Coconut Butter
4	2	30035	5	0	Natural Sweetener

Table2: Processed Data

Then, python pandas library was used to apply "group by" function to group the data by distinct orders and then appended the product_name column. Table 3 shows the pre-processed data.

	order_id	product_name
0	2	Organic Egg Whites,Michigan Organic Kale,Garli...
1	3	Total 2% with Strawberry Lowfat Greek Strained...
2	4	Plain Pre-Sliced Bagels,Honey/Lemon Cough Drop...
3	5	Bag of Organic Bananas,Just Crisp, Parmesan,Fr...
4	6	Cleanse,Dryer Sheets Geranium Scent,Clean Day ...

Table3: Data after goupring

The data was exported with the product_name column into a text file to have in the desired format. i.e. all the products in each order appended in a single row.

```
Organic Hass Avocado,Chocolate Protein Soy & Dairy Protein Shake,Strawberry C Monster Smoothie
Organic Roma Tomato,Organic Cilantro,Organic Red Onion
Organic Tomato Cluster,All Natural Chicken Drumsticks
```

Fig1: Data Alignment

Multiple sheets were there, but one has to preprocess the data to make it useable for desired task. We take two tables (order_product and product). In order_product table, we had the order_id, and product_id. But there were number of item rows for single order. And in product table we had the product_id and product_name columns. We first took join between these two tables with respect to product id, so we get the product name with order id, then we used group by to take single order in singe row, having multiple item names (which the customer bought) in front of order_id. Then the data is written in the text file having single order in a row and all the fields were separated by ",".

e.g: Order_Id, Product1, Product2, Product3

Implementation

We performed the testing using pySpark and a windows server. The spark version is 3.0.2. the machine has 6 cores and 16Gb of ram. Pyspark has a built-in library function to evaluate FP-Growth on an algorithm, so we used that and applied it on various subsets of our dataset. For Apriori Algorithm, there is no built-in function in the Pyspark library, so we created the algorithm ourselves and tested it on the same dataset as FP-Growth.

Results

In our experiments we found that Frequent Pattern Growth is exponentially faster than Apriori method when deployed on a windows machine using Pyspark. On a subset of complete dataset, the time taken by both algorithms is illustrated in Fig 2 and Fig 3.

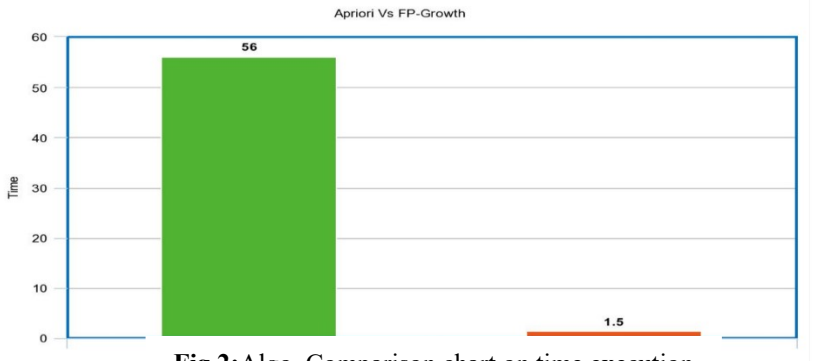

Fig 2: Algo. Comparison chart on time execution

We can clearly see that FP-Growth generates the frequent itemsets and association rules much faster than the Apriori which scans the dataset multiple times to find the rules while FP-Growth just scans once i.e. saving the execution time. Below is the illustration of time taken by Apriori Algorithm to generate association rules in data of different itemsets. We can see that the time taken increases more with increase in no. of rows of data.

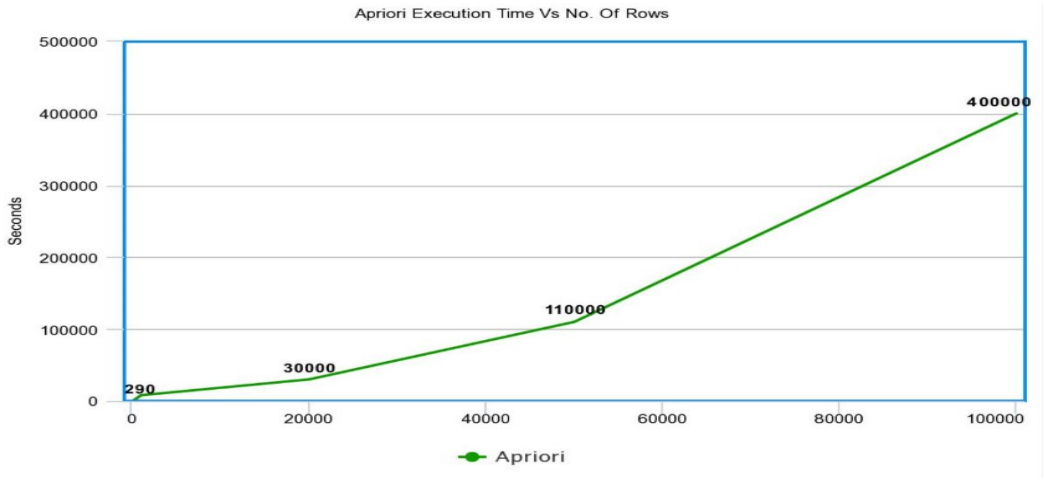

Fig 3: Apriori Execution time Vs. # of rows

Below is the illustration of how FP-Growth works with datasets of various sizes (Fig 4). We can see that the time taken to process a greater number of rows increase but in comparison to Apriori the time taken and increase in time both are almost negligible.

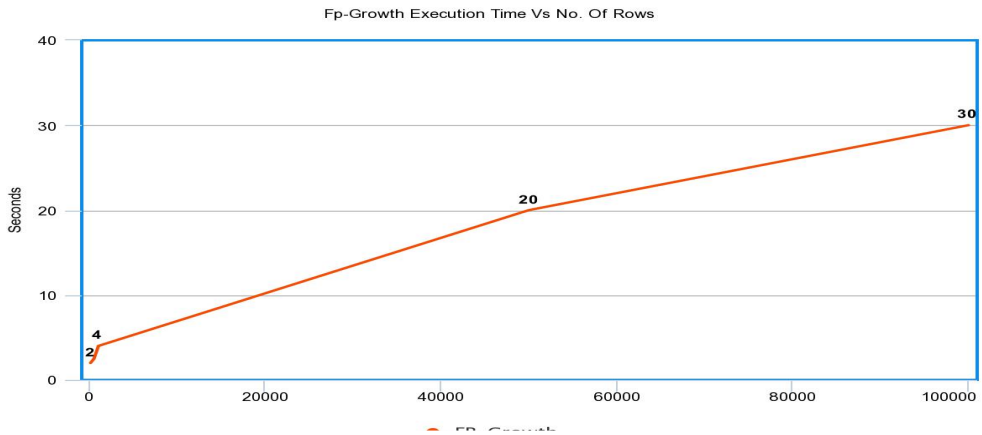

Fig 4: FP-Growth Execution time Vs. # of rows

Conclusion

This study was performed to compare the two most used methods for market basket analysis i.e. Apriori Algorithm and Frequent Pattern Growth Algorithm using one of the most popular data processing Framework called 'Spark'. We used the Python variant of spark known as 'Pyspark'. The Performance of FP-Growth was found to be significantly higher than the Apriori algorithm. The execution time for FP-Growth was very low while Apriori took a lot of time to generate results. Also, the computational cost using Apriori is quite higher as we can we it takes enough time to deliver. The main reason high time consumption was Apriori scans the data multiple times to generate candidate itemsets.so if the data is very large and complex, the execution time increases exponentially. While FP-Growth works on the concept of 'Divide & Conquer'. And it scans the whole database only twice. Which decreases the execution time required in the 'Apriori Algorithm'. So, we can conclude by this study that FP-Growth outperforms Apriori when using 'Spark'.

References

[1] Kanigiri Bharathi, Birru devender,"Frequent itemset mining from big data using FP-growth algorithm", Complexity International Journal, vol. 24(3), pp. 582-591, 2020

[2]Lior Shabtay, Rami Yaari and Itai Dattner, "A Guided FP-growth algorithm for multitude-targetedmining of big data", arxiv.org, 2018

[3] Ming Yin, Wenjie Wang,, Yang Liu,, and Dan Jiang, "An improvement of FP-Growth association rulemining algorithm based on adjacency table" in proceedings of MATEC Web of Conference, 2018

[4] Ravi Ranjan, Aditi Sharma, "Evaluation of Frequent Itemset Mining Platforms using Apriori and FPGrowth Algorithm", in proceedings of 4th International Conference on Computers and Management 2018

[5] Yuhang Miao, Jinxing Lin, Nuo Xu, "An improved parallel FP-growth algorithm based on Spark and its application", in proceedings of the 38th Chinese Control Conference, China, 2019

[6] DanieleApiletti, ElenaBaralis, TaniaCerquitelli, PaoloGarza, FabioPulvirenti, LucaVenturini, "Frequent Itemsets Mining for Big Data: A Comparative Analysis", Big Data Research, vol. 9, pp. 67-83, 2017

[7] I. Stančin and A. Jović, "An overview and comparison of free Pythonlibraries for data mining and big data analysis", MIPRO, China 2019

[8] Dawen Xia , Xiaonan Lu, Huaqing Li , Wendong Wang, Yantao Li and Zili Zhang, "A MapReduce-Based Parallel Frequent Pattern Growth Algorithm for Spatiotemporal Association Analysis of Mobile Trajectory Big Data", Complexity, 2018

[9]Aissatou Diaby dite Gassama, Fode Camara, Samba Ndiaye, "S-FPG: A Parallel Version of FP-Growth Algorithm under Apache Spark™", in proceedings of e 2nd IEEE International Conference on Cloud Computing and Big Data Analysis, 2017

[10] Zhicheng Cai, Xingyu, ZhuYuehui, ZhengDuan, LiuLei Xu, "A Caching-Based Parallel FP-Growth in Apache Spark", in proceedings of International Conference on Algorithms and Architectures for Parallel Processing 2018

[11] Noman Islam, Zubair A. Shaikh, "Exploiting correlation among data items for cache replacement in Ad-hoc Networks", in proceedings of 2nd IEEE International Conference on Information Management and Engineering, China, 2010

[12] Noman Islam, Zubair A. Shaikh, "Service discovery in Mobile Ad hoc networks using Association Rules Mining", in proceedings of IEEE 13th International Multitopic Conference, China 2009

[13] Meng, Xiangrui, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman et al. "Mllib: Machine learning in apache spark." The Journal of Machine Learning Research vol. 17, no. 1,pp. 1235-1241, 2016,