International Journal of Distributed and Cloud Computing
7 (1) June 2019, 01-04
http://www.publishingindia.com/ijdcc

Value Proposition and ETL Process in Big Data Environment

Prateek Kumar[1] and Veena Gaded[2*]
[1]RVCE, Bangalore, Karnataka, India.
[2]RVCE, Bangalore, Karnataka, India. Email: veena.gadad@gmail.com
*Corresponding Author

Abstract: For any retail company, managing inventory is of prime importance. Every store should have enough items so that it can fulfill the demand. To achieve this, the stores must be restocked before those items become out of stock. For restocking, the items must arrive from a fulfillment center which distributes the items to various stores, also called distribution centers. Since, distribution center and fulfillment centers are generally far from each other, there is a delay between request for restock and the time it takes for the item to reach from fulfillment centers to distribution centers. To prevent out of stock conditions, the request should be made by considering the time it takes for an item to arrive from fulfillment center. The quantity of item also determines the request time as only few quantities of large items can be sent at once and need multiple transits to restock to the required numbers. Along with these, there are other conditions like general traffic, seasonal climate variations, etc. that can affect the transit time of items. All of these conditions must be taken care while deciding when the item is requested. The proposed system decides the request time and quantity of items along with different variations by training from years of data. This allows the system to work more efficiently and prevent the out of stock conditions to increase sales of the company.

Keywords: Big data, ETL process, HDFS, SparkML, SparkSQL, Value proposition.

I. Introduction

Value of a project is determined by how well it can serve the organization, having the knowledge about what the project is supposed to do and how it is going to achieve the goals help determine how the final result is going to be. This allows determining the business value of the project and also enables people responsible to make better decisions throughout the development of the project. Various graphs are used to display the data which displays the on-going problems which can be understood by non-technical people [1]. Since the data is huge, the selection of visualization software must be done on the basis of fast data ingestion and quick indexing so that the visualization is done at a faster speed. A Lucene based text search engine is preferred for the operation as it stores indices as text which makes it faster to search hence the elastic stack is chosen for the project [7]. Elastic stack contains Elasticsearch, a Lucene based text search engine, Logstash, for data ingestion, and Kibana, to create visualizations [8]. These visualizations are then studied to determine the importance of the project.

ETL, Extract Transform and Load, is a process in databases. Data extraction is the process where data is extracted from various data sources, both homogenous and heterogenous [9]. Data transformation is the process in which the data is transformed in a format that can be used for the purpose of analysis. Data loading is the process of loading the data on a suitable storage like other databases or datamart. To increase the performance, the process is parallelized. This parallelization is done using Spark in this project. Spark also provides other tools such as SparkSQL and SparkML which makes it one in all tool for this project. SparkSQL allows writing SQL queries to extract data from databases and use its built-in machine learning framework to use it on trained data generated by filtering the available data for required attributes. Spark is also compatible with Hadoop so it allows seamless integration with scalable storage. It can load data on and from HDFS parallely which speeds up all the storage operations and queries. Since Spark provide such seamless integration, it is an ideal tool for big data operations and development of all the modules of this project.

Many approaches have been used till date for value proposition analysis. The techniques used had their own advantages and disadvantages. This section describes some of the works done in past, along with their summarized conclusion.

The study by Bowman and Forman [1, 6] tries to increase ETL efficiency and reduce the time of processing in distributed system using Hadoop Distributed File System (HDFS) and Apache Spark. Usually, there is an increase in the time taken by ETL for processing, with the increase in number of records in the data source. ETL needs to process equal number of records as are in the data sources. The work uses an ETL method using Spark. It is observed that it is an efficient model and the amount of data to be processed is reduced using this model.

Armbrust [2] elaborates the benefits of Apache Spark as against Hadoop MapReduce. For the analysis and storage of very large datasets, Hadoop MapReduce is used. The process of reading data from disk and writing result in Hadoop Distributed File System, is a time-consuming process. It also consumes a significant amount of disk space. Spark removes these flaws in the system. Spark is proven to be more efficient than MapReduce in this study, on the basis of processing real time data and pattern analysis.

Kimball [3] explains the importance of distributed data processing, in cloud computing, for analyzing the data at large scale. Apache Hadoop MapReduce, even due to its low-level programming interface and long implementation process, has standardized itself in this area from a long time. This has created a shift to more advanced platforms for data flow, for example, Apache Spark or Apache Flink. These platforms not only improve performance by the use of in-memory processing, but also provide filtering and join operators for built-in high-level data processing functionality. This makes analysis of data easier than how it used to be with Hadoop MapReduce. This paper gives a comparative study of three distributed data processing platforms, i.e., Apache Hadoop MapReduce, Apache Spark and Apache Flink. It was observed from the results that Spark and Flink are a better choice when compared to MapReduce. For this reason, they are more preferred over MapReduce in application.

The study by Ross [4, 5] shows the exponential increase in rate of data in past few decades. There is a need to obtain meaningful results from this data. This leads to the development of data processing platforms like Hadoop and Spark. A distributed nodes method is used in Hadoop as underlying architecture for data analysis. Spark works effectively as it can cache the results obtained from the past queries. Although Spark is memory bound. In this paper the underlying architecture of Spark and Hadoop are compared.

II. Methodology

The existing system manually decides the number of items to be shipped to a particular distribution center on a particular day so that they do not run out of stuck. The item and store data are collected and stored in a spreadsheet file which is analyzed by people responsible manually and they come up with a number. This process is slow and not accurate. The proposed system solves this problem by moving the entire operation on a big data environment so that all the data is available at one place and required attributes are filtered out. This filtered data is used as a training set for a machine learning algorithm which will predict the final item number for a particular distribution center. This makes the process much faster and accurate.

A. Dataset Source

The data for value proposition and ETL is obtained from various databases such as Oracle, Teradata and Cassandra. The data consists of item and store data which is then used to filter the required attributes.

B. Proposed Framework

As in Fig. 1 the data is extracted from Oracle DB and Teradata for Value Proposition. The extracted data is loaded on the logstash and inserted into Elasticsearch and then viewed through the visualization tool called Kibana.

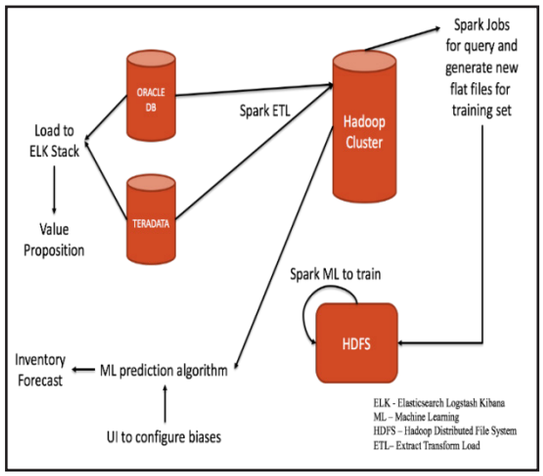

Fig. 1: System Architecture

After Value Proposition, the data is extracted from the databases by writing spark programs and then stored in hadoop clusters. The stored data is then processed and filtered according to need and transformed in a way which is in the form that can be used by SparkML to run machine learning algorithms. The transformed data is then stored on HDFS which would be read by SparkML. After training the machine learning algorithm based on one or two years of data, the current data is used as test data and the machine learning algorithm is used to predict the final forecast for the number of items needed at particular DC on a particular day.

A UI is also developed for users to configure biases for the items if some items require prioritization or if there is specific condition on the item such as seasonal sales.

III. Results and Analysis

The evaluation metric is the availability of data across the modules. The data being extracted from the databases must be fully available. Lost data causes problems in analysis. Therefore, the data should be extracted completely from the databases.

In Spark, the execution of dataframes are automatically optimized. This optimization is done by a query optimizer. This optimizer understands the logic of operation and how the data is structured and then makes intelligent decisions to increase the computation speed. Since JVM bytecode is generated for the execution, Scala, Java and Python, all languages have same

high performance. Fig. 2 compares the performance during execution of group-by-aggregation done on 10 million pairs of integers. It is done on single machine. Since the compilation is done into JVM bytecode for both Python and Scala, there is almost no difference between the languages. From the Fig. 3, it can be seen that RDD on Python is worse than dataframes by a factor of 5 and from RDD with Scala by a factor of 2.

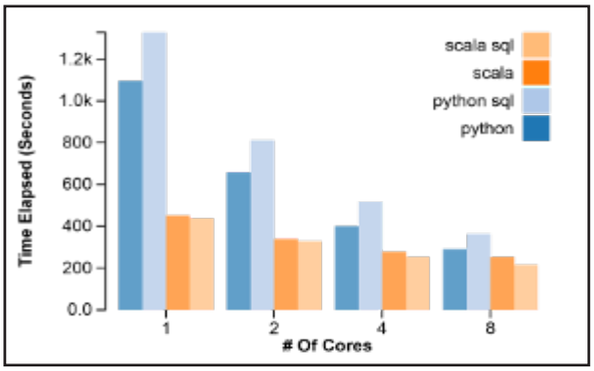

Fig. 2: Performance of Aggregating 10 million int pairs (sec)

To test the performance of Spark with Scala and Python, dataset consisting of two text file tables, of sizes 297M and 229M were selected. Every script was run with a collect statement at the end to ensure that each step was executed. The time was recorded between the start of the script and the end of the script. It was run for 1, 2, 4 and 8 cores. In the Fig. 4, it can be observed that the fastest performance was achieved when SparkSQL was used with Scala and the slowest was with Python. When the number of cores increases the results even out likely because of parallelization.

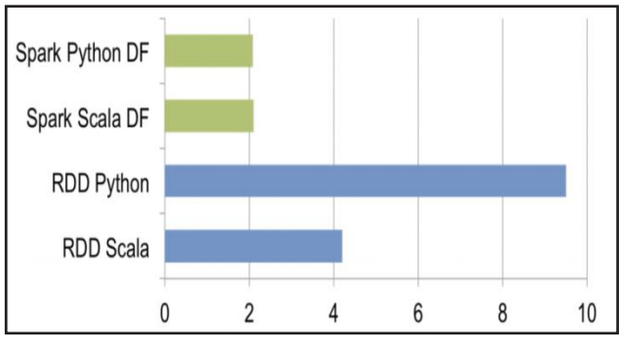

Fig. 3: Spark Query Run Time Using Python and Scala

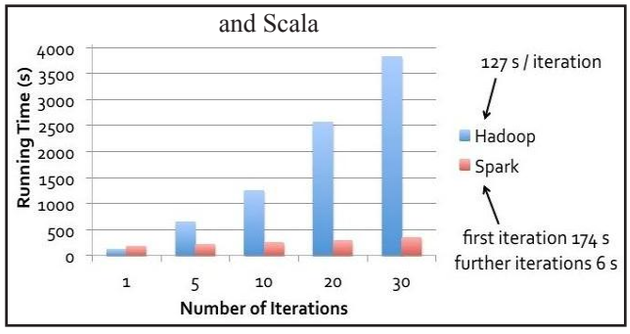

Fig. 4: Logistic Regression Performance

In Fig. 4, it can be observed that Spark has much faster execution than Hadoop. Machine learning algorithms runs iteratively. Algorithms that require multiple iterations often involve bottleneck of input and output when using implementations via MapReduce. MapReduce uses task-based parallelization that are too computational expensive for these iterative algorithms. On Spark, intermediate datasets are cached after each iteration many iterations are run on dataset that reduces the input / output and allows the algorithm to be fast and run in a fault tolerant manner.

IV. Constraints and Dependencies

There are many challenges in choosing the correct system to process the data. The system must be able to handle large amount of data but at the same time must be able to process queries faster. To achieve this, the system must have support for parallelization. For value proposition, it must be made sure that Elasticsearch and Kibana are running prior to using Logstash to input data. There are several dependency issues that can occur while integrating the components. The java version along with the Scala and spark version along with the connection files, are all dependent on each other. Specific version of one software is dependent on the specific version of another.

V. Limitations

All the information gained from data is derived information. No new data is created. Project is limited by parameters and feature set can only have the parameters available and not the derived parameters. The processing speed is limited by the resources available by the systems used.

VI. Future Enhancements

The data extracted and filtered from the databases will be processed in SparkML by choosing a suitable machine learning algorithm according to the requirement. A UI will be developed to enable user to input biases for various items and include that when running the algorithm.

VII. Conclusion

The aim of the paper is to solve the problem of inventory management across the stores. The amount of profits the company makes depends on the number of items each of their store sells. But to sell the items, the items must be present at the store and must be restocked at proper intervals. This restocking is done in advance so that the item never goes out of stock. The challenge is to decide the day at which the restocking is to be requested, as it takes certain time for the items to reach the store from the fulfillment center, and the number of items that are needed at particular store. The current system predicts the number of items required at each store manually by managing a spreadsheet of past year data and deciding the quantity based on some formula.

This project tries to speed up the process and make the process more accurate by using the concepts of Big Data and cluster computing frameworks. The data from the databases are huge and hence, are stored on a scalable storage such as HDFS. Spark is used for extracting the data from databases and storing on scalable storage. Spark parallelizes the operation so the process is fast. Spark is then used for filtering the data obtained from various databases so that only the required attributes are available to act as training set. The project then aims to predict the inventory quantity by using SparkML, a machine learning framework for Spark, to train using the training set and then predict the final number of items based on current data. This provides better accuracy as training set is much bigger than the spreadsheets and hence provide a better information about item sale pattern.

References

[1] M. Bowman, S. K. Debray, and L. L. Peterson. "Reasoning about naming systems," ACM *Transactions on Programming Languages and Systems (TOPLAS),* vol. 15, no. 5, pp. 795-825, 1993.

[2] M. Armbrust, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, and M. Zaharia, "Spark SQL: Relational data processing in spark," *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, SIGMOD'15*, pp. 1383-1394, 2015.

[3] R. Kimball, and J. Caserta, *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting Cleaning Conforming and Delivering Data*, Wiley Publishing, Inc., 2017.

[4] R. Kimball, and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, 3rd ed., Jonh Wiley & Sons, Inc., 2017.

[5] D. M. Tank, A. Ganatra, Y. P. Kosta, and C. K. Bhensdadia, "Speeding ETL processing in data warehouses using high-performance joins for Changed Data Capture (CDC)," pp. 365-368, October 2017.

[6] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research,* vol. 3, pp. 1289-1305, March 2003.

[7] I. Mekterovic, and L. Brkic, "Delta view generation for incremental loading of large dimensions in a data warehouse," *2016 38th International Convention on Information and Communication Technology Electronics and Microelectronics (MIPRO),* pp. 1417-1422, May 2016.

[8] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," *2017 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST),* pp. 1-10, May 2017.

[9] T. Hey, S. Tansley, and K. Tolle, *The Fourth Paradigm: Data Incentive Scientific Discovery*, Microsoft Corporation, 2009.