Retail Sales Analytics Using Apache Spark

Abhijith.A.Thampi
Roll No: AM.EN.U4AIE20102
Computer Science(AI)
Amrita School of Engineering, Amritapuri, Kollam, Kerala

Ajay G Nair
Roll No: AM.EN.U4AIE20108
Computer Science(AI)
Amrita School of Engineering, Amritapuri, Kollam, Kerala

Govind Nandakumar
Roll No: AM.EN.U4AIE20129
Computer Science(AI)
Amrita School of Engineering, Amritapuri, Kollam, Kerala

Vivin
Roll No: AM.EN.U4AIE20173
Computer Science(AI)
Amrita School of Engineering, Amritapuri, Kollam, Kerala

***Abstract*—In this project we are analysing Retail-Dataset using Apache Spark. Apache Spark enables large and big data analyses. It does this by parallel processing using different threads and cores optimally. It can therefore improve performance on a cluster but also on a single machine.**
The retail market being a highly competitive space and has companies innovating in the field of consumer behaviour analysis to find new buying trends. The retail industry traditionally has been generating detailed data on consumer behaviour and purchase history through various transactional and customer relationship systems. Exploiting these data to increase basket value and optimize margin remains a challenge with traditional technologies of business intelligence tools and data warehouse architectures. As consumers are empowered by data and mobility, with access to competitive prices and information while visiting stores, retailers' capacity to integrate data from online (browsing pattern, clickstreams, social media) and in-store customer experience (POS, surveys, CRM etc) and be able to react in real time is key. Data analytics is used for improving marketing efficiency and raising profits is not a new fad in the retail industry.
Here we are trying to analyze and train the chosen dataset. From the given dataset of retails we find and analyze the product-wise sales distribution of the massive dataset. At last we find the total sales completed by a customer from the dataset.

Keywords—Big Data Analytics, Spark, PySpark, Architecture

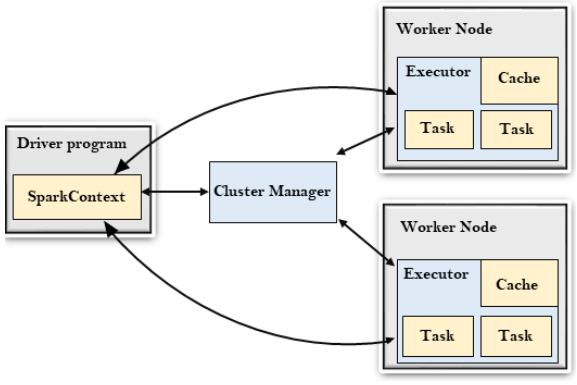
I. INTRODUCTION

Sales and customer data provide important insight for companies. Analytics has today an important role in providing data about sales trends, customer profiles, warnings about (un)profitable products, sales volume forecasts and stock supply as well as personalizing customer relationships and giving product recommendations. All this information needs to be delivered in time, preferably in a (semi)automated fashion. These data can become large to the point that analytical tools and software have performance issues. For example, many businesses embrace spreadsheet software like Excel for understandable reasons, but it would break down with large data. Many data scientists work with Python/R, but modules like Pandas would become slow and run out of memory with large data as well.

Apache Spark enables large and big data analyses. It does this by parallel processing using different threads and cores optimally. It can therefore improve performance on a cluster but also on a single machine. It can do this for (i) unstructured data such as text or for (ii) structured data such as Data Frames arranged in columns. Spark does not require loading all data into memory before processing and it is faster than for example Hadoop. Spark is a multi-language tool. Interfaces exist for Scala, Java, Python and R users, and it can be used in the cloud. These and other features make it a suitable platform for large scale data analyses.

II. APACHE SPARK

Apache Spark enables large and big data analyses. It does this by parallel processing using different threads and cores optimally. It can therefore improve performance on a cluster but also on a single machine. It can do this for (i) unstructured data such as text or for (ii) structured data such as Data Frames arranged in columns. Spark does not require loading all data into memory before processing and it is faster than for example Hadoop.
Spark is a multi-language tool. Interfaces exist for Scala, Java, Python and R users, and it can be used in the cloud. These and other features make it a suitable platform for large scale data analyses. Google trends suggest that PySpark - Spark with a Python interface - enjoys increasing popularity.



A. *Features*

Apache Spark has many distinguishable features. Following is a description of some of these features:

- *Speed:* Apache Spark is a tool that can be used for running Spark applications in Apache Hadoop cluster. Apache Spark is hundred times faster than Apache Hadoop and ten times faster than accessing data from the disk. Spark utilizes the idea of a Resilient Distributed Dataset (RDD), and enables it to distinctly store data inside the memory.
- *Usability:* Spark enables users to swiftly write applications in various programming languages such as Java, Scala, R and Python. This helps programmers to develop and run their applications on languages familiar to them which makes it easy to develop parallel applications.
- *Advanced analytics:* Besides the simple map and reduce operations, Sparks favors SQL queries, data streaming, and other complicated analytics such as machine learning, and graph algorithms.
- *Runs everywhere:* Apache Spark can be run on various platforms such as Apache Hadoop YARN, Mesos, EC2, Kubernetes or in the cloud using the Apache Spark standalone cluster mode. It can retrieve several data information that are HDFS, Cassandra, HBase etc.
- *In-memory computing:* In-memory cluster computation allows Spark to run iterative machine learning algorithms and aids bilateral querying and data streaming analysis at super-fast speeds. Spark keeps data in the RAM of the servers so that the stored data can be accessed quickly
- *Real-time stream processing:* Spark streaming grasps real-time stream processing along with other configurations concluding that spark streaming is simple, fault tolerant and unsegregated.

B. *Ecosystem*

The Apache Spark ecosystem consists of the following main components:

- *Spark SQL:* Formerly known as Shark. Spark SQL is a distributed framework that works with structured and semi-structured data. It facilitates analytical and interactive application for both streaming and historical data which can be accessed from various sources such as JSON, Parquet and Hive table

- *Spark* Streaming: It enables users to process streaming of data in real time. In order to perform streaming analysis, Spark streaming enhances the fast scheduling capability of Apache Spark by inserting data into mini batches. An operation known as transformation is then applied to those mini batches that can be easily obtained from live streams and data sources such as Twitter, Apache Kafka, IoT sensors, and Amazon Kinesis.

- *MLlib:* It delivers high-quality algorithms with high speed and makes machine learning easy to use and scale. Several machine learning algorithms such as regression, classification, clustering, and linear algebra are present. It also provides a library for lower level machine learning primitives like the generic gradient descent optimization algorithm. It also provides other functions such as model evaluation and data import. It can be used in Java, Scala, and Python.

- *GraphX:* It is a graph computation engine that enables the building, manipulation, transformation and execution of graph-structured data at a large scale. It consists of various Spark RDD API that facilitates the creation of directed graphs.

- *Spark Core:* Various functionalities of Apache Spark are built on top of the Spark core. It provides a vast range of APIs as well as applications for programming languages such as Scala, Java, and Python APIs to facilitate the ease of development. In-memory computation is implemented in Spark core in order to deliver speed and to solve the issue of MapReduce.

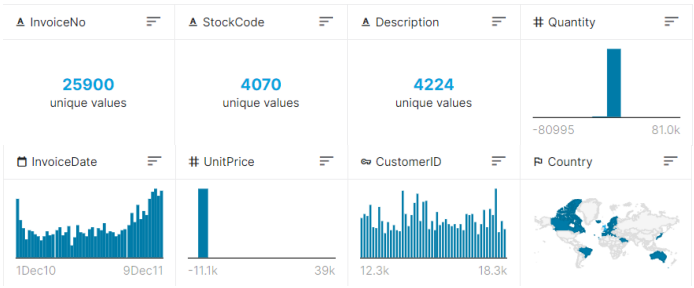
- *SparkR:* It is a package for R that enables data scientists to leverage the power of Spark from R shell. Since DataFrame is the basic data structure for data processing in R, similarly SparkR DataFrame is the fundamental unit of SparkR. It can perform various functions on large datasets such as selection, filtering and aggregation.

III. PySpark

PySpark has been released in order to support the collaboration of Apache Spark and Python, it actually is a Python API for Spark. In addition, PySpark, helps you interface with Resilient Distributed Datasets (RDDs) in Apache Spark and Python programming language. Along with writing Spark applications using Python APIs, it also provides the PySpark shell for interactively analyzing your data in a distributed environment. PySpark supports most of Spark's features such as Spark SQL, DataFrame, Streaming, MLlib (Machine Learning) and Spark Core

IV. Dataset

The dataset used for analysis has been taken from kaggle. This dataset contains all purchases made for an online retail company based in the UK during an eight month period. The data set contains 8 columns and 5,41,910 rows. It specifies details such as Invoice Number, Stock Code, Description of product, Quantity bought, Invoice date, Price per unit, Customer ID and Country. Using all this information, we try to find the overall product-wise sales distribution.



V. Implementation

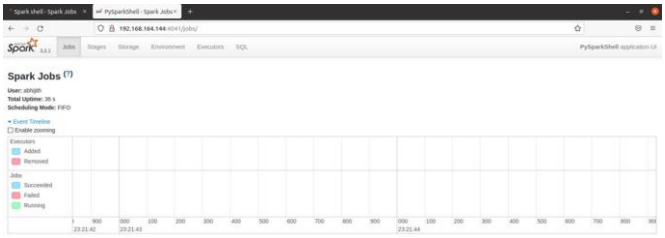
For implementing and carrying out this project, we use Apache Spark. First, we set up a spark session after importing all the necessary libraries. Then we load the massive dataset with around 5 Lakh entries. We see that the time required to load such a large dataset is relatively small when using Spark. Due to the large size of the dataset, we limit the number of entries that are displayed while sample code execution.

Once the dataset is loaded to Spark, it can be easily modified or manipulated according to our convenience. Instead of displaying the traditional Spark DataFrame output, we use a spark setting to get a Pandas-like output. A number of descriptive statistics are obtained, like count, standard deviation, mean, minimum and maximum. We make sure to remove canceled transactions in the dataset.

Using the inbuilt functionalities in Python and Spark, information about the column type can be obtained. Since we now know the column types, we could now cast column quantity from integer to float. Multiple columns can be used in a computation. The total amount a customer spends can be computed by product of the price of a single product with quantity purchased. We can select rows with specific products, or search our data using keywords. We find big buyers, probably organizational customers, by filtering the quantity above a certain level. The number of data records per country is calculated, allowing us to know which country interacts or does the most transactions. The analysis also includes which hours are preferred by customers while making purchases.

VI. Results

The below screenshots show the PySpark WebUI and list the details of the jobs executed by PySpark. It also displays the time and space required to execute each specific jobs.



VII. Conclusion

Thus, we were able to analyze the huge dataset in a matter of minutes with the help of Python API for Spark, i:e PySpark. Because of this, both the analytics and the programming section has been simplified to a great extent. If we did this analysis using any other software facilities, computational time might've increased exponentially. We can use this to deal with large datasets required in deep learning and other such implementations. We can infer from the analysis that purchase/sales in the UK exceeds that in other countries from the given dataset.

References

[1] J. Ellingwood, "Hadoop, storm, samza, spark, and flink: Big data frameworks compared," Website, 10 2016. [Online]. Available: https://www.digitalocean.com/community/tutorials/hadoopstorm-samza-spark-and-flink-big-data-frameworks-compared

[2] Ummadisetty, Vijaykumar. "Online Retail Data Set." *Kaggle*, 26 Oct. 2017, https://www.kaggle.com/datasets/vijayuv/onlineretail.

[3] Weber, Ben. "A Brief Introduction to PySpark." *Medium*, Towards Data Science, 17 Dec. 2018, https://towardsdatascience.com/a-brief-introduction-to-pyspark-ff4284701873.

[4] Ostrowski, Radek. "Introduction to Apache Spark with Examples and Use Cases." *Toptal Engineering Blog*, Toptal, 23 Mar. 2015, https://www.toptal.com/spark/introduction-to-apache-spark.