Big Data in Cloud Computing with HDFS and Spark- The future of analytics

Dr. R. Hemalatha #1, K.Keerthana *2,
Associate Professor and Head, PG & Research, Department of Computer Science,
Tiruppur Kumaran College for Women, Tirupur, TamilNadu, India
1 hemdaksin@gmail.com
* Research Scholar (M.Phil,), PG & Research, Department of Computer Science,
Tiruppur Kumaran College for Women, Tirupur, TamilNadu, India
2 keerthi1644@gmail.com

Abstract— **In this modern era, usage of Cloud computing has skyrocketed since the birth of Big Data. Big Data is an evolving term that describes any voluminous amount of structured, semi structured and unstructured data. Big Data by deploying Big Data software like Hadoop on Cloud background which gives users an innovative pattern of service called as Analytics as a Service. Hadoop MapReduce is written in Java and is infamous for being very difficult to program. Apache Spark is setting the world of Big Data on fire. Spark was intended to read and write data from and to Hadoop Distributed File System, as well as other storage systems, such as HBase and Amazon's S3. Spark is easier to program and includes an interactive mode.**

Keywords—Hadoop,HDFSMapReduce,Spark,Cloud computing,Fog computing, Petabytes

I. INTRODUCTION

With the emergence of digital age, the amount of data being generated, stored and shared has been on the rise. From data warehouses, web pages and blogs to audio, video streams, all of these are sources of massive amounts of data. The order of Petabyte, Exabyte, and Zetabyte and further rising year by year are generally referred as Big Data. Cloud computing has a great structural design to execute Big Data. Cloud computing refers to on-demand computer resources and systems available across the network that can provide a number of integrated computing services without local resources to facilitate user access. Cloud computing offers compact infrastructure maintenance cost, user access, and proficient management along with less charges and automatic control by individual or enterprises. When making an attempt to understand the concept of Big Data, the words and Hadoop cannot be avoided. Hive, Sqoop, Zookeeper, Spark and Storm are some other essential tools for crunching Big Data.

II. CLOUD COMPUTING

Cloud computing is a method of delivering technology to the consumer by using Internet servers for processing and data storage, while the client system uses the data. While in the past, software had to be shipped on a CD-ROM, and updates had to be downloaded afterwards to keep the software secure and bug-free, Cloud computing allows vendors to deliver software and services over the Internet without the need for traditional media or installation. A popular benefit of Cloud supporting Big Data can be noted at both Google and Amazon. Clearly, numerous combinations of deployment and delivery models exist for Big Data in the Cloud. Scalability, Elasticity, Resource pooling, Self-service, Fault-tolerance, Pay as you go are some Cloud characteristics which make it an important part of Big Data ecosystem.

Cloud computing has become more popular as the Internet is prolific, available in remote locations and on devices as small as your smart phone. The use of Cloud computing for business data has become increasingly popular in recent years, as more and more businessmen and administrations recognize the value of protecting data to the greatest possible extent. Some foremost widespread Cloud applications globally are Amazon web Services (AWS), Google compute Engine, Rackspace, Salesforce.com, IBM Cloud Managed Services, among others. Cloud services have created it potential for little and medium businesses (SMBs) to get on par with giant firms.

III. BIG DATA

Big Data could be an assortment of enormous datasets that cannot be adequately processed using traditional processing techniques. Massive data datasets are so large and sophisticated that traditional data processing application software are deficient to cope with them. This offers rise to the idea referred to as Big Data.Big Data needs novel techniques and technologies to form new patterns of combinations to find out the largely hidden values from the huge collections of data sets that are massive, dissimilar and complex. Big Data uses Apache Hadoop, Hive, Apache Spark, and Sqoop which are open source frameworks to store and analyze data.

IV. BIG DATA IN CLOUD ENVIRONMENT

Cloud computing is a paradigm for allowing ever-present, well-situated, and on-demand network approach to a number of configured computing resources that can be promptly provisioned and unconfined with minimal management effort or service provider interaction. It provides the contact to applications as utilities above the internet. Big Data requires the bunch of servers to support the tools and technologies that process on enormous volume, the variety of formats and high velocity. Clouds already have the pool of servers, networking

resources, and storage that can scale up and down when desired. It offers the cost-effective way to hold Big Data technologies.

In Cloud computing, we have Cloud server. With the Cloud server, many of the Cloud service providers are connected and they all are works parallel to each other and manage/balance the load.

The Cloud computing enables miniature to intermediate sized business to implement Big Data technology with a reduced commitment of company resources. The processing capability of the Big Data model could provide new insights to the business pertaining to performance improvement, decision making support, and modernization in business models, products, and services. Benefits of implementing Big Data technology through Cloud computing are cost savings in hardware and processing, as well as the capability to experiment with Big Data technology ahead of making a large commitment of company resources. Several models of Cloud computing services are offered to the businesses to judge, with each model having trade-offs between the advantage of cost savings and the concerns data security and loss of control.

V. BENEFITS OF BIG DATA

There are some benefits when the organization applied the Big data analytics-

- Client-based analysis
- Identification of market and sales chances
- Computerized decisions for real-time processes
- Risk Quantification
- Better knowledge of business changes
- Better scheduling and forecasting
- Fraud detection
- Detection of consumer behavior from click streams etc.

VI. APPROACHES FOR SECURITY OF BIG DATA IN CLOUD COMPUTING ENVIRONMENT:

Here we state few security measures that can be used to enhance the Cloud computing environment.

1) *Encryption*:

Since the data in any system will be present in a cluster, a hacker can easily steal the data from the system. This issue is the gaining popularity in the industry, especially when it comes to public and externally hosted Cloud implementations. To avoid this, we may go for encrypting the data. Different encryption mechanisms can be used on different systems and the keys generated should be stored secretly behind firewalls. By selecting this technique the data of the user may be kept securely.

2) *Nodes authentication*:

The node must be authenticated whenever it joins the cluster. If the node turns out to be a malicious cluster then such nodes must not be authenticated.

3) *Honeypot nodes*:

The honeypot nodes appear to be like a regular node but are a trap. It automatically traps the hackers and will not allow any damage to happen to the data.

4) *Access control*:

The differential privacy and access control in the distributed environment will be a good measure of security. To prevent the information from leaking we use a SELinux. Several users should be granted diverse access privileges with regard to different data pieces. The access authorization must be controlled only by the owner in dubious Cloud environments.

VII. BIG DATA ANALYTICS TOOL HADOOP FRAMEWORK

The Apache Hadoop is an open-source project administrated by the Apache software foundation. The software was originally developed by the world's largest Internet companies to capture and analyze the data that they generate. Hadoop is capable of storing any kind of data in its local format and to perform a broad variety of analyses and transformations on that data. Hadoop can store even petabytes of data inexpensively. Hadoop is reliable, robust and handles hardware and system failures automatically, without losing data.

Modules of Hadoop includes

Hadoop Common: This includes the general utilities that maintain the other Hadoop modules.

HDFS: The Hadoop Distributed File System provides unrestricted, high-speed access to the application data.

Hadoop YARN: This technology accomplishes scheduling of job and efficient management of the cluster resource.

MapReduce: Highly efficient methodology for parallel processing of huge volumes of data.

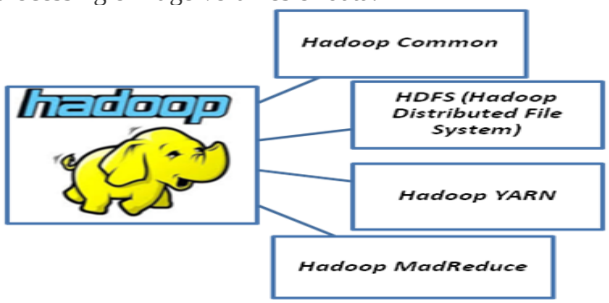

Fig 1.Modules of spark

A. *HDFS and MapReduce*

HDFS is a distributed file system envisioned to store large files unfold across numerous physical systems and hard drives. HDFS are split apart and spread out over multiple physical machines and hard drives. As a user, these details are transparent; you don't need to know how your files are broken apart or where they are stored.

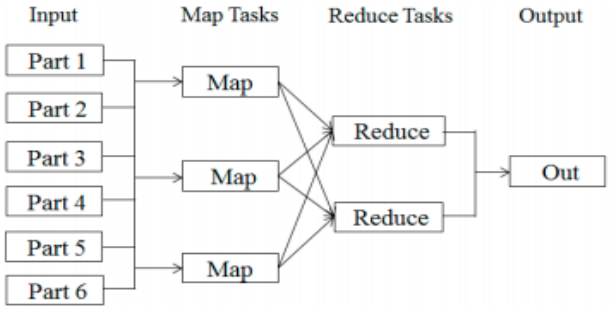

Fig 2. The process of MapReduce framework.

MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map allows different points of the distributed cluster to distribute their work. Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples.

B. *Limitations of Hadoop for Big Data analysis*

In Hadoop, data is highly available, highly scalable and reliable. Hadoop is fault tolerant, easy to use and not very expensive. Despite of all this features it has some limitations as well.

- Hadoop is not suitable for small data.

- In MapReduce data can be distributed in parallel and MapReduce reports the results back after processing, which increases the time and reduce processing speed.
- Hadoop supports batch processing only, it does not process streamed data, and hence overall performance is lower.
- Hadoop is not suitable for Real-time data processing.
- It is not so efficient for iterative processing, as Hadoop does not support cyclic data flow.
- It is vulnerable by nature and not efficient for caching.
- Hadoop only ensures that data job is complete, but it is unable to guarantee when the job will complete.

C. *Dawn of Spark*

Because of the limitations of Hadoop, the need of Spark emerged. This made the system friendlier to play with a huge amount of data. Spark provides in-memory processing of data thus improve the processing speed.

VIII.SPARK

Spark was developed in University of California, Berkeley. Apache Spark is a high-performance processing framework for in-memory analytic computational processing with well-designed and significant development APIs to permit data workers to efficiently execute streaming, machine learning workloads that demand fast iterative access to datasets. However, it was built to be installed on top of Hadoop cluster; however, its ability to parallel processing allows it runs independently as well. Fast processing, Flexibility, In-memory computing, Better analytics, Compatible with Hadoop are some main features of Apache Spark.

Spark runs on Hadoop, Mesos, standalone, or in the Cloud. It can access various data sources including HDFS, Cassandra, HBase, MongoDB, Redis, Kafta and S3 etc.

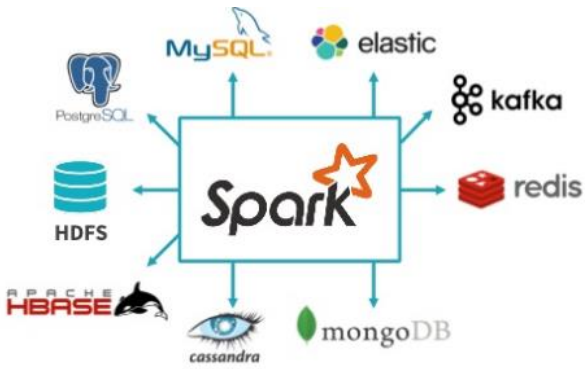

Fig 3.Spark

A. *Spark's Architecture*

Apache Spark can run three ways:

Standalone – The Hadoop cluster can be equipped with all the resources statically and Spark can run with MapReduce in parallel. This is the simplest deployment.

On Hadoop YARN – Spark can be executed on top of YARN without any pre-installation. This deployment utilizes the maximum strength of Spark and other components.

Spark In MapReduce (SIMR) – It enables running Spark jobs as well as Spark shell, on Hadoop MapReduce without installing Spark.

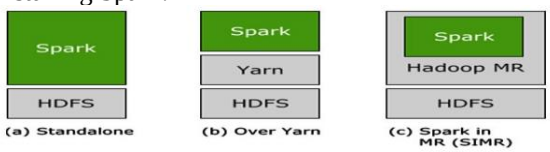

Fig 4.Spark's Architecture

B. *Spark Platform*

Spark uses Scala language for development, which is concise and efficient. The fundamental programming abstraction of Spark is called Resilient Distributed Datasets (RDD), a logical collection of data partitioned across machines. By allowing user programs to load data into a cluster's memory and to query it repeatedly, Spark is well suited to iterative algorithms, such as ant colony algorithm. Spark divides a computer cluster into a master node and multiple worker nodes. The master node is responsible for task scheduling, resource allocation and error management. Worker nodes process Map tasks and Reduce tasks in parallel. User program distribution and input data splitting are automatically completed by the platform.

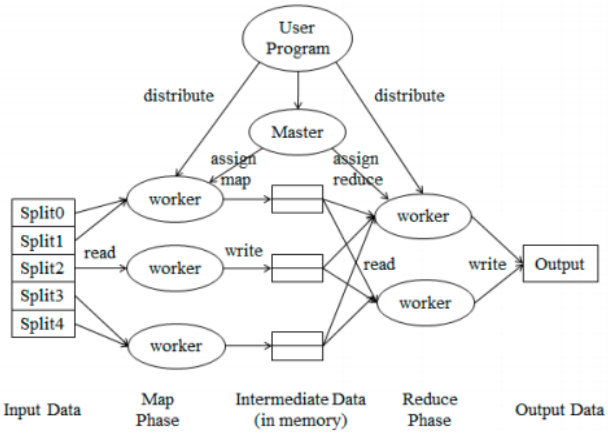

Fig 5.The working principle of Spark.

C. *Resilient, Distributed Datasets, DataFrames and Dataset APIs*

Spark stores the data for the job in *Resilient, Distributed Datasets* (RDDs), where a logical data set is partitioned over the cluster. The user can specify that data in an RDD should be cached in memory for subsequent reuse. In contrast, MapReduce has no such mechanism, so a complex job requiring a sequence of MapReduce jobs will be penalized by a complete flush to disk of intermediate data, followed by a subsequent reloading into memory by the next job.

A *DataFrame* is a distributed collection of data organized into named columns. It is theoretically equivalent to a table in a relational database or an R/Python Dataframe. Along with Dataframe, Spark also introduced catalyst optimizer, which leverages advanced programming features to build an extensible query optimizer.

Dataset API is an extension to DataFrames that has a type-safe, object-oriented programming interface. Each Dataset also has an untyped view called a DataFrame, which is a Dataset of Row. Dataset represents a logical plan that describes the computation required to produce the data.

D. *Components of Spark*

Spark core: Spark Core is responsible for basic input and output operations and provide some functionalities like task dispatching, job scheduling, examining the jobs on Spark clusters, networking with various storage systems, fault recovery and memory management.

There are two operations performed on RDDs:

- **Transformation:** It is a function that produces new RDD from the existing RDDs.
- **Action:** In Transformation, RDDs are created from each other. But when we want to work with the actual dataset, then, at that point we use Action.

Spark SQL: Leverage the power of declarative queries and optimized storage by running SQL like queries on Spark data

that is present Resilient Distributed Datasets (RDDs) and other external resources.

- Mid-query fault-tolerance
- Fully compatible with existing Hive data.
- Data Frames and SQL afford a widespread method to access a variety of data sources. It includes Hive, Avro, Parquet, ORC, JSON, ElasticSearch, Cloudant, Redshift, Mongo and JDBC.

Spark Streaming: It allows the developers to perform batch processing and streaming of data with ease in the same application.

Fig 6.Spark Streaming

Spark Streaming makes it easy to construct scalable fault-tolerant streaming applications. *Discretized streams or* DStreams can be produced from sources for instance Kafka, MySQL, Flume, and Cassandra, or by applying high-level operations on other DStreams.

MLlib: It eases the deployment and development of scalable machine learning pipelines.MLlib includes algorithms for working with Spark features. Moreover, it divided into these groups:

- **Extraction:** Extracting features from "raw" data.
- **Transformation**: Scaling, converting, or modifying features.
- **Selection**: Selecting a division of a larger set of features.
- **Locality Sensitive Hashing (LSH)**: This class of algorithms combines aspects of feature transformation with other algorithms.

MLlib library has implementations for various common machine-learning algorithms –

i. Clustering- K-means
ii. Classification – naïve Bayes, logistic regression, SVM
iii. Decomposition- Principal Component Analysis (PCA) and Singular Value Decomposition (SVD)
iv. Regression –Linear Regression
v. Collaborative Filtering-Alternating Least Squares for Recommendations

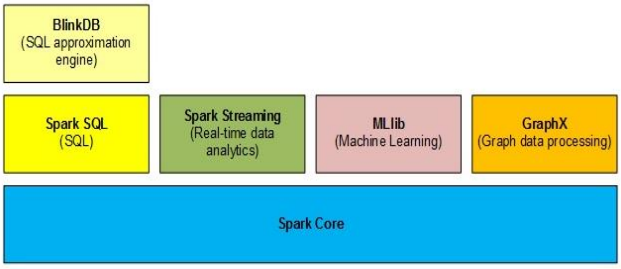

Fig 7.Spark Ecosystem

GraphX: Data scientist can work with graph and non-graph sources to achieve flexibility and resilience in graph construction and transformation.

SparkR: SparkR is an R package which is a light-weight frontend distributed with Apache Spark. SparkR supports operating on a variety of data sources through the Spark *DataFrame* interface. SparkR also supports distributed machine learning using MLlib.

BlindDB: BlindDB or Blind Database is also known as an Approximate SQL database. If there is a huge amount of data barraging and you are not really interested in exactitude, or in exact results, but just want to have a rough or an approximate picture, BlindDB gets you the same. Firing a query, doing some sort of sampling, and giving out some output is called Approximate SQL.

E. Spark Implementation using MDS Algorithm

The programming model of Spark does not allow tasks to communicate with one another, therefore all-to-all collectives are not supported in Spark. As a result, at each iteration the data needs to be collected at the master process and then a new set of tasks has to be created with the new data for the next iteration. This adds two additional overheads to the algorithm. The first is task creation and distribution at each iteration. The other is caused by the additional I/O that needs to be done at each step to perform reduce and broadcast instead of an AllReduce operation. Because of these limitations the main loops of the MDS(Multi-Dimensional Scaling) algorithm are executed in the driver program and large tasks such as calculate BC, calculate MM and calculate Stress are performed as distributed map-reduce tasks. These results in a numerous MapReduce phases. The resulting values are then broadcast from the driver program to the cluster before the next iteration is executed. Several RDD's that contain distance data and weight data are cached to improve performance. Because Spark does not allow in-place changes on RDDs, the algorithm generates intermediate data sets that are not required in the MPI (Message passing interface) implementation. These intermediate data sets increase the memory usage of the algorithm, which is problematic because memory is a limiting factor when running MDS on very large matrices.

F. Spark Implementation using K-Means Algorithm

K-Means is an efficient and practical classical machine learning algorithm for data clustering. The algorithm maintains K cluster points called centroids. At each iteration, a new set of centroids are calculated and fed back to the beginning of the iteration. The algorithm partitions the points into multiple map tasks and uses the full set of centroids in every one. Each map task assigns its points to their nearest centroid. The average of points is reduced for each centroid to get the new set of centroids, which are broadcast to the next iteration. Spark MLlib provides an implementation of K-Means, which is used for evaluations.

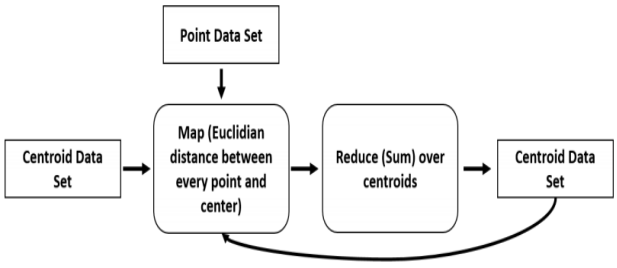

Fig 8.K-Means data flow graph for Spark

In MPI after the AllReduce operation, the centroids are updated within the program in-place. On the other hand, for Spark these centroids need to be broadcast back to all the tasks that do the nearest neighbor calculation in the next iteration. We can argue that all reduce operation in MPI is equivalent to reduce + broadcast, which is the mechanism used in Spark. But it is worth noting that AllReduce can be optimized for greater efficiency than running reduce and broadcast separately by using algorithms such as recursive doubling.

G. Spark clusters

Each program we will write is a Driver Program. It uses a Spark Context to communicate with the Cluster Manager,

which is an abstraction over Hadoop YARN, Mesos, standalone mode, EC2, and local mode. The Cluster Manager allocates resources. An Executor JVM process is created on each worker node per client application. It manages local resources, such as the cache and it runs tasks, which are provided by your program in the form of Java jar files or Python scripts. Because each application has its own executor process per node, applications cannot share data through the Spark Context. External storage has to be used.

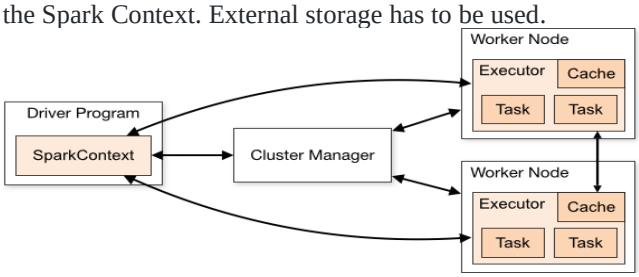

Fig 9.Anatomy of Spark cluster

H. Cluster Management in Apache Spark

Apache Spark applications can run in 3 different cluster managers

Standalone Cluster – If only Spark is running, then this is one of the easiest to setup cluster manager that can be used for novel deployments. In standalone mode - Spark manages its own cluster. In standalone mode, each application runs an executor on every node within the cluster.

Apache Mesos – While Spark and Mesos emerged together from the AMPLab at Berkeley, Mesos is now one of several clustering options for Spark, along with Hadoop YARN, which is growing in popularity, and Spark's "standalone" mode.

YARN- YARN comes with most of the Hadoop distributions and is the only cluster manager in Spark that supports security. YARN cluster manager allows dynamic sharing and central configuration of the same pool of cluster resources between various frameworks that run on YARN. The number of executors to use can be chosen by the user unlike the Standalone mode. YARN is a better choice when big Hadoop cluster is already in use at production.

I. Applications of Spark

Spark is a widely used technology adopted by most of the industries.

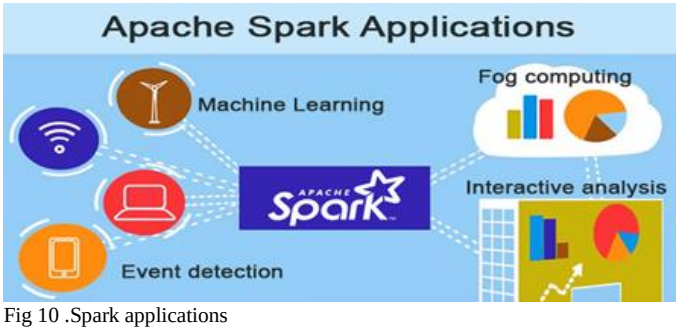

Fig 10 .Spark applications

Some of the leading applications of Apache Spark are Machine learning, Fog Computing, Event detection and Interactive analysis.

Fog Computing is a decentralized computing infrastructure in which data, compute, storage and applications are distributed in the most logical, efficient place between the data source and the Cloud.

Interactive analysis: With the Spark shell, we can easily learn on how to use the API, and a powerful tool for an interactive analysis of data will be provided. The shell for Spark is available in Python or Scala.

Machine learning: In general, machine learning may be broken down into two classes of algorithms: supervised and unsupervised.

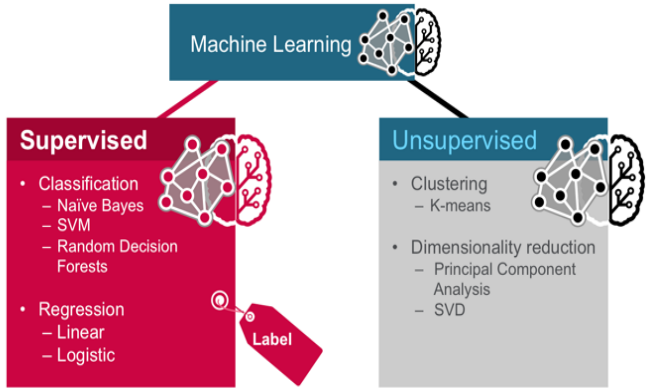

Fig 11.Machine learning algorithms

Three common categories of machine learning techniques are Classification, Clustering and Collaborative Filtering.

ML Model	Problems	Algorithms
Supervised Learning	Classification, Regression, Anomaly Detection	Logistic Regression, Back Propagation Neural Network
Unsupervised Learning	Clustering, Dimensionality reduction	k-Means , Apriori algorithm
Semi-Supervised Learning	Classification, Regression	Self training, Semi-supervised Support Vector Machines (S3VMs)

Table 1.Machine learning algorithms

Classification: Gmail uses a machine learning technique called classification to designate if an email is spam or not, based on the data of an email: the sender, recipients, subject, and message body. Classification is a function that assigns items in a collection to targeted classes.

Clustering: Google News uses a technique called clustering to group news articles into different categories, based on title and content. Clustering algorithms discover clusters that occur in collections of data.

Collaborative Filtering: Amazon uses a machine learning technique called collaborative filtering to decide which products users will like based on their history and similarity to other users.

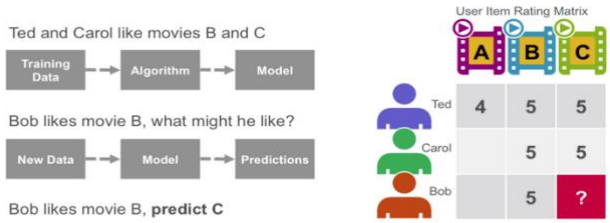

Fig 12.Collaborative filtering

J. Apache Spark use cases

- Banks are using the Spark to access and analyze the parameters which can help them, make right business decisions for credit risk assessment, targeted advertising and customer segmentation.
- Companies Using Spark in e-commerce Industry
- Spark is employed in genomic sequencing to reduce the time needed to process genome data.
- Apache Spark is employed within the video gaming to progress their business opportunities like targeted advertising, auto adjustment of gaming levels supported complexness, player custody and far additional.

- Yahoo uses Apache Spark for personalizing its news WebPages and for targeted advertising.
- Trip Advisor is a foremost travel website that uses Apache Spark to provide advice to millions of travelers by comparing hundreds of websites to find the best hotel prices for its customers.
- Spark is compatible with Apache Hive data warehousing system and it can run 100 xs faster when compared to Hive.
- Stream processing: It is used in Log processing and Fraud detection in live streams for alerts, aggregates and analysis
- Sensor data processing: In sensor data processing data is fetched and joined from numerous sources, in-memory dataset is in fact advantageous as they are easy and fast to process.

K. Some major difference between Hadoop and Spark

S.NO	FEATURES	HADOOP MAPREDUCE	SPARK
1.	Execution	Hadoop MapReduce reads and writes data from disk, so it slows down the processing speed.	Spark reduces number of operations by storing the intermediate data in memory, so that it becomes a lightning fast cluster computing tool.
2.	Accessibility	Hadoop MapReduce is written in java and is disreputable for being very difficult to program.	Spark has comfortable APIs for Java, Scala and Python and also includes Spark SQL.
3.	Difficulty	In MapReduce, developers need to hand code each and every operation which makes it very difficult to work.	Spark is easy to program as it has a lot of high-level operators with Resilient Distributed Dataset.
4.	Easy to manage	MapReduce can perform only batch processing.	Spark is a complete data analytics engine. It can perform Streaming ,batch processing, iterative processing etc.
5.	Real-time analysis	MapReduce is not suitable for real-time processing as it is designed to perform only batch processing.	Spark can process real-time data from live streams such as Facebook and Twitter.
6.	Fault-Tolerance	MapReduce is more fault tolerant than Spark	Spark also has good failure tolerance.
7.	Security	Hadoop MapReduce is more secure because of Kerberos.	Spark supports authentication through shared secret password and it becomes little less secure when compared to MapReduce.
8.	Cost	MapReduce is cheaper when compared to Spark.	Spark is more cost-effective as it requires lot of RAM to run.
9.	Data processing	MapReduce is great for batch processing.	Spark can process even existing machine learning libraries and graphs.

IX. FUTURE OF BIG DATA IN CLOUD

Lowering prices and handling complexness are going to be the first motivating factors for Cloud-based Hadoop deployments in 2018.In upcoming years Spark will continue to be the processing engine of choice for Hadoop systems. Technologies and Platforms with demonstrated capability in the area of unstructured data processing and analysis will be in great demand.

X. CONCLUSION

The differences between Apache Spark vs. Hadoop MapReduce displays that Apache Spark is much advance cluster computing engine than MapReduce. Spark will handle any variety of necessities (batch, interactive, iterative, streaming, graph) whereas MapReduce limits to batch processing. But however, Hadoop provides options that Spark doesn't possess, like a distributed file system. Spark is gaining considerable momentum and is a promising alternative to support ad-hoc queries. The perfect Big Data circumstances are exactly as the designers proposed for Hadoop and Spark to work together on the same team.

References

1. Judith Hurwitz, Alan Nugent, Dr.Fern Halper and Marcia Kaufman, *"Big Data for Dummies"*.
2. Anshu Dhabhai1, Yogesh Kumar Gupta,Banasthali *Vidypith "A Study of Big Data in Cloud Environment with their Related Challenges"*.
3. Smaranika Mohapatra, Jharana Paikaray, and Neelamani Samal, "Future Trends in Cloud Computing and Big Data." *Journal of Computer Sciences and Applications*, vol. 3, no. 6 (2015): 137-142. doi: 10.12691/jcsa-3-6-6.
4. Venkatesh Shrivatsa D Perur, Nivedita Jalihal, *"A Study on Use of Big Data in Cloud Computing Environment"*.
5. Shivaram Venkataraman , Zongheng Yang, Davies Liu , Eric Liang, Hossein Falaki, Xiangrui Meng , Reynold Xin, Ali Ghodsi , Michael Franklin , Ion Stoica , Matei Zaharia,AMPLab UC Berkeley, Databricks Inc., MIT CSAIL . *"Scaling R Programs withSpark*