Chapter 1
Big Data Analytics

1.1 Introduction

The latest disruptive trends and developments in digital age comprise social networking, mobility, analytics and cloud, popularly known as SMAC. The year 2016 saw Big Data Technologies being leveraged to power business intelligence applications. What holds in store for 2020 and beyond?

Big Data for governance and for competitive advantage is going to get the big push in 2020 and beyond. The tug of war between governance and data value will be there to balance in 2020 and beyond. Enterprises will put to use the enormous data or Big Data they already have about their customers, employees, partners and other stakeholders by deploying it for both regulatory use cases and non-regulatory use cases of value to business management and business development. Regulatory use cases require governance, data quality and lineage so that a regulatory body can analyze and track the data to its source all through its various transformations. On the other hand, the non-regulatory use of data can be like 360° customer monitoring or offering customer services where high cardinality, real time and mix of structured, semi-structured and unstructured data will produce more effective results.

It is expected that in 2020 businesses will shift to a data-driven approach. All businesses today require analytical and operational capabilities to address customers, process claims, use interfaces to IOT devices such as sensors in real time, at a personalized level, for each individual customer. For example, an e-commerce site can provide individual recommendations after checking prices in real time. Similarly, health monitoring for providing medical advice through telemedicine can be made operational using IOT devices for monitoring all individual vital health parameters. Health insurance companies can process valid claims and stop paying fraudulent claims by combining analytics techniques with their operational systems. Media companies can deliver personalized content through set-top boxes. The list of such use cases is endless. For achieving the delivery of such use cases, an agile platform is essentially required which can provide both analytical results and also operational efficiency so as to make the office operations more relevant and accurate, backed

C. S. R. Prabhu et al., *Big Data Analytics: Systems, Algorithms, Applications*,
https://doi.org/10.1007/978-981-15-0094-7_1

by analytical reasoning. In fact, in 2020 and beyond the business organizations will go beyond just asking questions to taking great strides to achieve both initial and long-term business values.

Agility, both in data and in software, will become the differentiator in business in 2020 and beyond. Instead of just maintaining large data lakes, repositories, databases or data warehouses, enterprises will leverage on data agility or the ability to understand data in contexts and take intelligent decisions on business actions based on data analytics and forecasting.

The agile processing models will enable the same instance of data to support batch analytics, interactive analytics, global messaging, database models and all other manifestations of data, all in full synchronization. More agile data analytics models will be required to be deployed when a single instance of data can support a broader set of tools. The end outcome will be agile development and application platform that supports a very broad spectrum of processing and analytical models.

Block chain is the big thrust area in 2020 in financial services, as it provides a disruptive way to store and process transactions. Block chain runs on a global network of distributive computer systems which any one can view and examine. Transactions are stored in blocks such that each block refers to previous block, all of them being time-stamped and stored in a form unchangeable by hackers, as the world has a complete view of all transactions in a block chain. Block chain will speed up financial transactions significantly, at the same time providing security and transparency to individual customers. For enterprises, block chain will result in savings and efficiency. Block chain can be implemented in Big Data environment.

In 2020, microservices will be offered in a big way, leveraging on Big Data Analytics and machine learning by utilizing huge amount of historical data to better understand the context of the newly arriving streaming data. Smart devices from IOT will collaborate and analyze each other, using machine learning algorithms to adjudicate peer-to-peer decisions in real time.

There will also be a shift from post-event and real-time analytics to pre-event and action (based on real-time data from immediate past).

Ubiquity of connected data applications will be the order of the day. In 2020, modern data applications will be highly portable, containerized and connected quickly replacing vertically integrated monolithic software technologies.

Productization of data will be the order of the day in 2020 and beyond. Data will be a product, a commodity, to buy or to sell, resulting in new business models for monetization of data.

1.2 What Is Big Data?

Supercomputing at Internet scale is popularly known as Big Data. Technologies such as distributed computing, parallel processing, cluster computing and distributed file system have been integrated to take the new avatar of Big Data and data science. Commercial supercomputing, now known as Big Data, originated at companies such as Google, Facebook, Yahoo and others, operates at Internet scale that needed to process the ever-increasing numbers of users and their data which was of very large volume, with large variety, high veracity and changing with high velocity which had a great value. The traditional techniques of handling data and processing it were found to be completely deficient to rise up to the occasion. Therefore, new approaches and a new paradigm were required. Using the old technologies, the new framework of Big Data Architecture was evolved by the very same companies who needed it. Thence came the birth of Internet-scale commercial supercomputing paradigm or Big Data.

1.3 Disruptive Change and Paradigm Shift in the Business Meaning of Big Data

This paradigm shift brought disruptive changes to organizations and vendors across the globe and also large social networks so as to encompass the whole planet, in all walks of life, in light of Internet of things (IOT) contributing in a big way to Big Data. Big Data is not the trendy new fashion of computing, but it is sure to transform the way computing is performed and it is so disruptive that its impact will sustain for many generations to come.

Big Data is the commercial equivalent of HPC or supercomputing (for scientific computing) with a difference: Scientific supercomputing or HPC is computation intensive with scientific calculations as the main focus of computing, whereas Big Data is only processing very large data for mostly finding out the patterns of behavior in data which were previously unknown.

Today, Internet-scale commercial companies such as Amazon, eBay and Filpkart use commercial supercomputing to solve their Internet-scale business problems, even though commercial supercomputing can be harnessed for many more tasks than simple commercial transactions as fraud detection, analyzing bounced checks or tracking Facebook friends! While the scientific supercomputing activity came downward and commercial supercomputing activity went upward, they both are reaching a state of equilibrium. Big data will play an important role in ‘decarbonizing’ the global economy and will also help work toward Sustainable Development Goals.

Industry 4.0, Agriculture or Farming 4.0, Services 4.0, Finance 4.0 and beyond are the expected outcomes of the application IOT and Big Data Analytics techniques together to the existing versions of the same sectors of industry, agriculture or farming, services, finance, by weaving together of many sectors of the economy to the one new order of the World 4.0. Beyond this, the World 5.0 is aimed to be achieved by the governments of China and Japan by deploying IOT and Big Data in a big way, a situation which may become ‘big brothers,’ becoming too powerful in tracking everything aiming to control everything! That is where we need to find a scenario of Humans 8.0 who have human values or Dharma, so as to be independent and yet have a sustainable way of life. We shall now see how the Big Data technologies based on Hadoop and Spark can handle practically the massive amounts of data that is pouring in modern times.

1.4 Hadoop

Hadoop was the first commercial supercomputing software platform that works at scale and also is affordable at scale. Hadoop is based on exploiting parallelism and was originally developed in Yahoo to solve specific problems. Soon it was realized to have large-scale applicability to problems faced across the Internet-scale companies such as Facebook or Google. Originally, Yahoo utilized Hadoop for tracking all user navigation clicks in web search process for harnessing it for advertisers. This meant millions of clickstream data to be processed on tens of thousands of servers across the globe on an Internet-scale database that was economical enough to build and operate. No existing solutions were found capable to handle this problem. Hence, Yahoo built, from scratch, the entire ecosystem for effectively handling this requirement. Thus was born Hadoops [1]. Like Linux, Hadoop was also in open source. Just as Linux spans over clusters of servers, clusters of HPC servers or Clouds, so also Hadoop has created the Big Data Ecosystem of new products, vendors, new startups and disruptive possibilities. Even though in open-source domain originally, today even Microsoft Operating System supports Hadoop.

1.5 Silos

Traditionally, IT organizations partition expertise and responsibilities which constrains collaboration between and among groups so created. This may result in small errors in supercomputing scale which may result in huge losses of time and money. A 1% error, say for 300 terabytes, is 3 million megabytes. Fixing such bugs will be an extremely expensive exercise.

In scientific supercomputing area, small teams managed well the entire environment. Therefore, it is concluded that a small team with a working knowledge of the entire platform works the best. Silos become impediments in all circumstances, both in scientific and in commercial supercomputing environments. Internet-scale computing can and will work only when it is taken as a single platform (not silos of different functions). A small team with complete working knowledge of the entire platform is essential. However, historically since the 1980s, the customers and user community were forced to look at computing as silos with different vendors for hardware, operating system, database and development platform. This leads to a silo-based computing. In Big Data and Hadoop, this is replaced with a single platform or a single system image and single ecosystem of the entire commercial supercomputing activities.

Supercomputers are Single Platforms

Originally, mainframes were single platforms. Subsequently, silos of products from a variety of vendors came in. Now again in Big Data, we are arriving at a single platform approach.

1.5.1 Big Bang of Big Data

Big Data will bring about the following changes:

1. Silo mentality and silo approach will be closed and will give rise to platform approach.
2. All the pre-Hadoop products will be phased out gradually since they will be ridiculously slow, small and expensive, compared to the Hadoop class of platforms.
3. Traditional platform vendors will therefore give way to Hadoop class of frameworks, either by upgrading or bringing out new platforms so as to meet the requirements.

1.5.2 Possibilities

The possibilities Big Data opens up are endless. Answers to questions hitherto never asked can be and will be answerable in the Big Data environment.

In the context of Internet of things (IOT), the data that can flow will be really big, in real time. In addition to the transactional data, the big time, big variety of data includes text, sensor data, audio and video data also. It expects processing and response in real time, which can be really delivered in Big Data Analytics. This means, while the data is still being collected, it can be analyzed in real time and plans or decisions can be made accordingly. This can enable the significant edge over competitors in terms of knowing in advance the trends, opportunities or impending dangers of problems much earlier than the competitors. Usage scenarios and use cases can be as follows.

Farmers get sensor data from smart farms to take decisions on crop management; automotive manufactures get real-time sensor data from cars sold and also monitor car health continuously through real-time data received from car-based sensor network. Global outbreaks of infectious diseases can be monitored in real time, so as to take preemptive steps to arrest their spread.

Previously, data was captured from different sources and accumulated in a supercomputer for being processed slowly, not in real time. The Big Data Ecosystem enables real-time processing of data in Hadoop clusters. Organizations are facing so massive volumes of data that if they do not know how to manage it, they will be overwhelmed by it. Whether the wall of data rises as a fog or as a tsunami, it can be collected with a common pool of data reservoir in Hadoop cluster, in real time, and processed in real time. This will be the superset of all individual sources of data in all organizations. Organizations can integrate their traditional internal data infrastructure as databases or data warehouses with a new Big Data infrastructure with multiple new channels of data. This integration is essential, along with the appropriate governance structure for the same.

1.5.3 Future

Big Data will change the course of history—the disruptive technology is thrusting computer science into a new vista away from the good old Von Neumann sequential computer model into the new Hadoop cluster model of parallel computing with real huge data being processed in real time.

1.5.4 Parallel Processing for Problem Solving

Conventionally, when large data is required to be processed adequately fast to meet the requirements of the application, parallel processing was identified to be the correct approach.

Parallel processing was achieved by multiple CPUs sharing the same storage in a network. Thus, we had the approaches of storage area network (SAN) or network access storage (NAS). Alternatively, ‘shared nothing’ architectures with each of the parallel CPUs having its own storage with stored data are also possible.

Due to rapid technology development, the processor speed shot up from 44 mips (million instructions per second) at 40 MHz in 1990 to 147,600 MIPS at 3.3 GHZ and beyond after 2010. RAM capacities went up from 640 KB in 1990 to 32 GB (8 such modules) and beyond after 2010. Storage disk capacities went up from 1 GB in 1990 to 1 TB and beyond after 2010 [2].

But, importantly, the disk latency speeds had not grown much beyond their 1990 ratings of about 80 MB/s.

While PC computing power grew 200,000% and storage disk capacity 50,000%, read/seek latency of the disk storage has not grown anywhere near that. Therefore, if we require to read 1 TB at 80 Mb/s, one disk takes 3.4 h, 10 disks take 20 min, 100 disks take 2 min, and 1000 disks take 12 s. This means that parallel reading of data from disks and processing them parallelly are the only answers.

Parallel data processing is really the answer. This was addressed earlier in grid computing where a large number of CPUs and disks are connected in a network for parallel processing purpose. The same was achieved in cluster computing with all CPUs being connected through a high-speed interconnection network (ICN).

While parallel processing, as a concept, may be simple, it becomes extremely challenging and difficult to write and implement parallel applications. Serious problems of data distribution for parallel computing followed by integration or summation of the results so generated also become very important. Since each node or CPU of the parallel CPU network computes only one small piece or part of the data, it becomes essential to keep track of the initial fragmentation of the data to be able to make sense during the integration of the data after the completion of computations. This means we will spend a lot of time and effort in management and housekeeping of the data much more than for computing itself.

Hardware failures in network need to be handled by switching over to standby machines. Disk failures also need to be considered. To process large data in parallel, we need to handle partial hardware failures without causing a total processing failure. If a CPU fails, we need to shift the job to a backup CPU.

1.5.5 Why Hadoop?

When data is stored in multiple locations, the synchronization of the changed data due to any update becomes a problem. If the same data is replicated (not for backup recovery but for processing purposes), then each replication location requires to be concerned with the backup of the data and the recovery of the data—this leads to greater complexity. In theory, if we can, we should keep only one single version of the data (as it happens in RDBMS). But in Hadoop environment, large volumes of data are stored in parallel and do not have an update capability.

What is the Answer?

Appropriate software that can handle all these issues effectively is the answer. That functionality is made available in Hadoop Distributed File System (HDFS).

1.5.6 Hadoop and HDFS

Hadoop and HDFS were initiated in Apache (under Notch project) developed at Yahoo by Doug Cutting for being able to process Internet-scale data. Since high-powered systems were expensive, commodity work stations were deployed. Large volumes of data were distributed across all these systems and processed in parallel. Failures of CPU and disk were common. Therefore, replication was done. In case of failure, the replicated backup node or disk will be utilized. Hadoop is a batch processing environment. No random access or update is possible. Throughput is given more importance.

Hadoop is an open-source project of Apache Foundation, and it is basically a framework written in Java [3]. Hadoop uses Google's MapReduce programming model and Google File System for data storage, as its basic foundations. Today, Hadoop is a core computing infrastructure for Yahoo, Facebook, LinkedIn, Twitter, etc.

Hadoop handles massive amounts of structured, semi-structured and unstructured data, using inexpensive commodity servers.

Hadoop is a 'shared nothing' parallel processing architecture.

Hadoop replicates its data across multiple computers (in a cluster), so that if one such computer server (node) goes down, the data it contained can still be processed by retrieving it from its replica stored in another server (node).

Hadoop is for high throughput, rather than low latency—therefore, Hadoop performs only batch operations, handling enormous quantity of data—response time in real time is not being considered.

Hadoop is not online transaction processing (OLTP) and also not online analytical processing (OLAP), but it complements both OLTP and OLAP. Hadoop is not the equivalent or replacement of a DBMS or RDBMS (other supporting environments over Hadoop as extensions such as Hive and other tools provide the database (SQL or similar) functionality over the data stored in Hadoop, as we shall see later in this chapter). Hadoop is good only when the work is parallelized [4]. It is not good to use Hadoop if the work cannot be parallelized (parallel data processing in large data environments). Hadoop is not good for processing small files. Hadoop is good for processing huge data files and datasets, in parallel.

What are the advantages of Hadoop and what is its storage structure?

(a) **Native Format Storage**: Hadoop's data storage framework called Hadoop Distributed File System (HDFS) can store data in its raw, native format. There is no structure that is imposed while keeping in data or storing data. HDFS is a schema-less storage structure. It is only later, when data needs to be processed, that a structure is imposed on the raw data.

(b) **Scalability**: Hadoop can store and distribute very large datasets (involving thousands of terabytes (or petabytes) of data).

(c) **Cost-Effectiveness**: The cost per terabyte of storage of data is the lowest in Hadoop.

(d) **Fault Tolerance and Failure Resistance**: Hadoop ensures replicated storage of data on duplicate server nodes in the cluster which ensures nonstop availability of data for processing, even upon the occurrence of a failure.

(e) **Flexibility**: Hadoop can work with all kinds of data: structured, semi-structured and unstructured data. It can help derive meaningful business insights from unstructured data, such as email conversations, social media data and postings and clickstream data.

(f) **Application**: Meaningful purposes such as log analysis, data mining, recommendation systems and market campaign analysis are all possible with Hadoop infrastructure.

(g) **High Speed and Fast Processing**: Hadoop processing is extremely fast, compared to the conventional systems, owing to 'move code to data' paradigm.

1.5.7 Hadoop Versions 1.0 and 2.0

Hadoop 1.0 and Hadoop 2.0 are the two versions. In Hadoop 1.0, there are two parts: (a) data storage framework which is the Hadoop Distributed File System (HDFS) which is schema-less storage mechanism; it simply stores the data files, and it stores in any format, whatsoever; the idea is to store data in its most original form possible; this enables the organization to be flexible and agile, without constraint on how to implement; and (b) data processing framework. This provides the functional programming model known as MapReduce. It has two functions: Map and Reduce functions to process data. The Mappers take in a set of key–value pairs and generate intermediate data (which is another set of key–value pairs). The Reduce function then acts on the input to process and produce the output data. The two functions, Map and Reduce, seemingly work in isolation from one another, so as to enable the processing to be highly distributed in a highly parallel, fault-tolerant and reliable manner.

1.5.7.1 Limitations of Hadoop 1.0

1. The requirement of proficiency in MapReduce programming along with proficiency in Java.
2. Only batch processing is supported, which can be useful only for typical batch applications such as log analysis and large-scale data mining and not useful for other applications.
3. Hadoop 1.0 is largely computationally coupled with MapReduce. Thus, DBMS has no option but to either deploy MapReduce programming in processing data or pull out data from Hadoop 1.0 and then process in DBMS. Both of these options are not attractive.

Therefore, Hadoop 2.0 attempted to overcome these constraints.

1.5.8 Hadoop 2.0

In Hadoop 2.0, the HDFS continues to be the data storage framework. However, a new and separate resource management framework called Yet Another Resource Negotiator or YARN has been added. Any application which is capable of dividing itself into parallel tasks is supported by YARN. YARN coordinates the allocation of subtasks of the submitted application, thereby enhancing the scalability, flexibility and efficiency of the application. It performs by deploying ‘Application Master’ in place of the old ‘Job Tracker,’ running application on resources governed by a new Node Manager (in place of old ‘Task Tracker’). Application Master is able to run any application and not just MapReduce.

Therefore, MapReduce programming is not essential. Further, real-time processing is also supported in addition to the old batch processing. In addition to MapReduce model of programming, other data processing functions such as data standardization and master data management also can now be performed naturally in HDFS.

1.6 HDFS Overview

If large volumes of data are going to be processed very fast, then we essentially require: (i) Parallelism: Data needs to be divided into parts and processed in parts simultaneously or parallelly in different nodes. (ii) Fault tolerance through data replication: Data needs to be replicated in three or more simultaneously present storage devices, so that even if some of these storage devices fail at the same time, the others will be available (the number of replication as three or more are decided by the replication factor given by the administrator or developer concerned). (iii) Fault tolerance through node (server) replication: In case of failure of the processing nodes, the alternate node takes over the processing function. We process the data on the node where the data resides, thereby limiting transferring of the data between all the nodes (programs to process the data are also accordingly replicated in different nodes).

Hadoop utilizes Hadoop Distributed File System (HDFS) and executes the programs on each of the nodes in parallel [5]. These programs are MapReduce jobs that split the data into chunks which are processed by the 'Map' task in parallel. The 'framework' sorts the output of the 'Map' task and directs all the output records with the same key values to the same nodes. This directed output hence then becomes the input into the 'Reduce' task (summing up or integration) which also gets processed in parallel.

- HDFS operates on the top of an existing file system (of the underlying OS in the node) in such a way that HDFS blocks consist of multiple file system blocks (thus, the two file systems simultaneously exist).
- No updates are permitted.
- No random access is permitted (streaming reads alone are permitted).
- No caching is permitted.
- Each file is broken into blocks and stored in three or more nodes in HDFS to achieve reliability through redundancy by replication.
- Master node (also known as name node) carries a catalogue or directory of all the slave nodes (or data nodes).
- Slave nodes (or data nodes) contain the data.
- Limited file security.

Data read by the local OS file system gets cached (as it may be called up for reading again any time, as HDFS cannot perform the caching of the data).

HDFS performs only batch processing using sequential reads. There is no random reading capability, nor there is any capability to update the data in place.

The master node includes name node, Job Tracker and secondary name node for backup.

The slave node consists of data nodes and Task Tracker. Data nodes are replicated for fault tolerance.

HDFS uses simple file permissions (similar to Linux) for read/write purposes.

1.6.1 MapReduce Framework

HDFS described above works on MapReduce framework.

What is MapReduce? It is a simple methodology to process large-sized data by distributing across a large number of servers or nodes. The master node will first partition the input into smaller subproblems which are then distributed to the slave nodes which process the portion of the problem which they receive. (In principle, this decomposition process can continue to many levels as required). This step is known as Map step.

In the Reduce step, a master node takes the answers from all the subproblems and combines them in such a way as to get the output that solves the given application problem.

Clearly, such parallel processing requires that there are no dependencies in the data. For example, if daily temperature data in different locations in different months is required to be processed to find out the maximum temperature among all of them, the data for each location for each month can be processed parallelly and finally the maximum temperature for all the given locations can be combined together to find out the global maximum temperature. The first phase of sending different locations of data to different nodes is called Map Phase, and the final step of integrating all the results received from different nodes into the final answer is called Reduce Phase.

MapReduce framework also takes care of other tasks such as scheduling, monitoring and re-executing failed tasks. HDFS and MapReduce framework run in the same set of nodes. Configuration allows effective scheduling of tasks on the nodes where data is present (data locality). This results in very high throughput. Two daemons (master) Job Tracker and (slow) Task Tracker for cluster nodes are deployed as follows.

1.6.2 Job Tracker and Task Tracker

Job Tracker performs

(1) Management of cluster and
(2) Application management.

In managing the cluster, it keeps free and busy notes and assigns the tasks accordingly.

In application management, it receives the application problem from the client (by the user) and replicates the same into all the nodes. It will split the input data into blocks which will be sent to the Task Trackers in data nodes (Fig. 1.1).

The Task Tracker is responsible for executing the individual tasks assigned by the Job Tracker. A single Task Tracker exists per slave node and spawns multiple MapReduce tasks in parallel. Task Tracker sends continuous heartbeat messages to Job Tracker. If heartbeat message is not received indicating failure of node, then the task will be assigned to another node by Job Tracker.

(a) MapReduce Framework:

Phases	Daemons
Map Reduce	Job Tracker Task Tracker

(b) Client

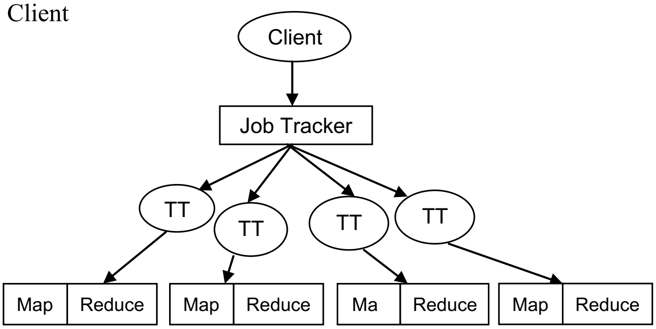

Fig. 1.1 MapReduce framework and Job Tracker

1.6.3 YARN

YARN, which is the latest version of MapReduce or MapReduce 2, has two tasks (1) resource manager and (2) application manager.

Resource manager is fixed and static. It performs node management for free and busy nodes for allocating the resources for Map and Reduce phases.

For every application, there is a separate **application manager** dynamically generated (on any data node). Application manager communicates with the **resource manager,** and depending on the availability of data nodes (or node managers in them) it will assign the Map Phase and Reduce Phase to them.

1.7 Hadoop Ecosystem

1. Hadoop Distributed File System (**HDFS**) simply stores data files as close to the original format as possible.
2. **HBase** is a Hadoop database management system and compares well with RDBMS. It supports structured data storage for large tables.
3. **Hive** enables analysis of large data with a language similar to SQL, thus enabling SQL type of processing of data in a Hadoop cluster.
4. **Pig** is an easy-to-understand data flow language, helpful in analyzing Hadoop-based data. Pig scripts are automatically converted to MapReduce jobs by the

Pig Interpreter, thus enabling SQL-type processing of Hadoop data [6]. By using Pig, we overcome the need of MapReduce-level programming.

5. **ZooKeeper** is a coordinator service for distributed applications.
6. **Oozie** is a workflow schedule system to manage Apache Hadoop Jobs.
7. **Mahout** is a scalable machine learning and data mining library.
8. **Chukwa** is a data collection system for managing large distributed systems.
9. **Sqoop** is used to transfer bulk data between Hadoop and as structured data management systems such as relational databases.
10. **Ambari** is a web-based tool for provisioning, managing and monitoring Apache Hadoop clusters.
11. **Ganglia** is the monitoring tool.
12. **Kafka** is the stream processing platform.

We will be covering all the above later in Chap. 4 (Fig. 1.2).

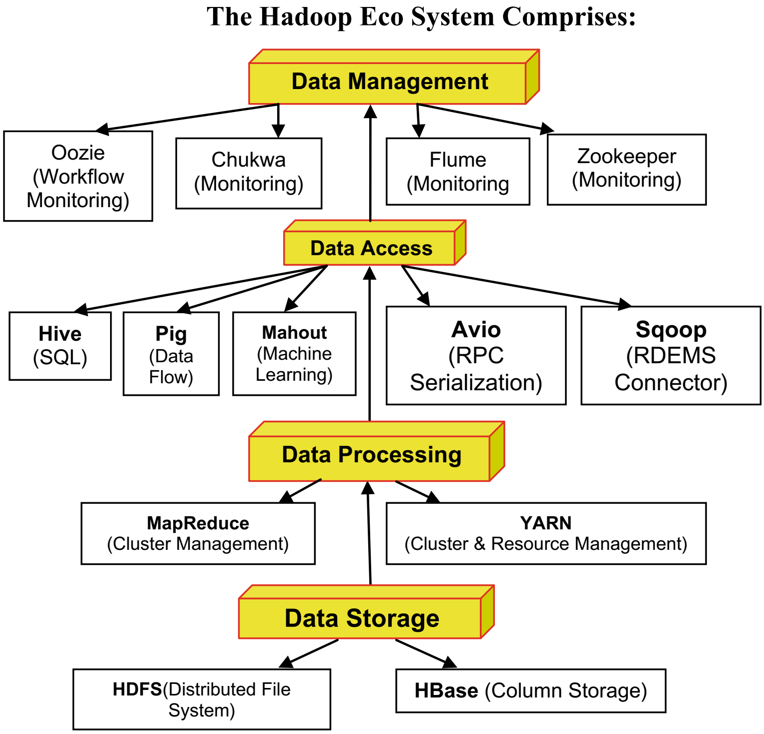

Fig. 1.2 Hadoop ecosystem elements—various stages of data processing

1.7.1 Cloud-Based Hadoop Solutions

a. Amazon Web Services (AWS) offers Big Data services on cloud for very low cost.
b. Google BigQuery or Google Cloud Storage connector for Hadoop empowers performing MapReduce jobs on data in Google Cloud Storage.

1.7.2 Spark and Data Stream Processing

Batch processing of ready-to-use historical data was one of the first use cases for big data processing using Hadoop environment. However, such batch processing systems have high latency, from a few minutes to few hours to process a batch. Thus, there is a long wait before we see the results of such large volume batch processing applications.

Sometimes, data is required to be processed as it is being collected. For example, to detect fraud in an e-commerce system, we need real-time and instantaneous processing speeds. Similarly, network intrusion or security breach detection must be in real time. Such situations are identified as applications of data stream processing domain. Such data stream processing applications require handling of high velocity of data in real time or near real time.

A data stream processing application executing in a single machine will not be able to handle high-velocity data. A distributed stream processing framework on a Hadoop cluster can effectively address this issue.

Spark Streaming Processing

Spark streaming is a distributed data stream processing framework. It enables easy development of distributed applications for processing live data streams in near real time. Spark has a simple programming model in Spark Core. On top of Spark Core, Spark streaming is a library. It provides scalable, fault-tolerant and high-throughput distributed stream processing platform by inheriting all features of Spark Core. Spark libraries include: Spark SQL, Spark MLlib, Spark ML and GraphX. We can analyze a data stream using Spark SQL. We can also apply machine learning algorithms on a data stream using Spark ML. We can apply graph processing algorithms on a data stream using GraphX.

Architecture

A data stream is divided into microbatches of very small fixed time intervals. Data in each microbatch is stored as a RDD which is processed by Spark Core. Any RDD operations can be applied to an RDD and can be created by Spark streaming. The final results of RDD operations are streamed out in batches.

Sources of Data Streams

Sources of data streams can be any basic source as TCP sockets and actors, files or advanced streams such as KIKA, Flume, MQTT, ZeroMQ and Twitter. For advanced sources, we need to acquire libraries from external sources for deployment, while for basic sources we can have library support within spark itself.

API

Even though Spark library is written in Scala, APIs are provided for multiple languages such as Java and Python, in addition to native Scala itself.

We cover Spark in greater detail in Chap. 4.

1.8 Decision Making and Data Analysis in the Context of Big Data Environment

Big Data is characterized by large volume, high speed or velocity, and large accuracy or veracity and variety. Current trend is that of data flowing in from a variety of unstructured sources such as sensors, mobile phones and emails, in addition to the conventional enterprise data in structured forms such as databases and data warehouses. There exists a need for correct decision making while taking an integrated, unified and comprehensive view of the data coming from all these different and divergent sources. Even the regular data analysis techniques such as data mining algorithms are required to be redefined, extended, modified or adapted for Big Data scenarios. In comparison with the conventional statistical analysis techniques, the Big Data Analytics differ in the sense of scale and comprehensiveness of the data availability. In traditional statistical analysis approach, the data processed was only a sample. This was so due to the fact that data was scarce and comprehensive data was not available. But in today’s context, the situation is exactly opposite. There is a ‘data deluge.’ Data, both structured or semi-structured and unstructured, flows in nonstop, either as structured data through various information systems and databases and data warehouses or as unstructured data in social networks, emails, sensors in Internet of things (IOT). All this data needs to be processed and sensible conclusions to be drawn from that deluge of data. Data analytics techniques which have been around for processing data need to be extended or adapted for the current Big Data scenario.

1.8.1 Present-Day Data Analytics Techniques

Knowledge Discovery in Database (KDD) or data mining techniques are aimed at automatically discovering previously unknown, interesting and significant patterns in given data and thereby build predictive models. Data mining process enables us to find out gems of information by analyzing data using various techniques such as decision trees, clustering and classification, association rule mining and also advanced techniques such as neural networks, support vector machines and genetic algorithms. The respective algorithms for this purpose include apriori and dynamic item set counting. While the algorithms or the processes involved in building useful models can be well understood, the implementation of the same calls for large efforts. Luckily, open-source tool kits and libraries such as Weka are available based on Java Community Processes JSR73 and JSR247 which provide a standard API for data mining. This API is known as Java Data Mining (JDM).

The data mining processes and algorithms have strong theoretical foundations, drawing from many fields such as mathematics, statistics and machine learning. Machine learning is a branch of artificial intelligence (AI) which deals with developing algorithms that machines can use to automatically learn the patterns in data. Thus, the goal and functionality of data mining and machine learning coincide. Sometimes, advanced data mining techniques with the machine being able to learn are also called machine learning techniques. Data mining differs from the conventional data analysis. Conventional data analysis aims only at fitting the given data to already existing models. In contrast, data mining finds out new or previously unknown patterns in the given data. Online analytical processing (OLAP) aims at analyzing the data for summaries and trends, as a part of data warehousing technologies and its applications. In contrast, data mining aims at discovering previously unknown, non-trivial and significant patterns and models present within the data. Therefore, data mining provides a new insight into the data being analyzed. However, all these related topics of data warehousing, OLAP and data mining are broadly identified as business intelligence (BI).

Present-day data analytics techniques are well identified as: (1) clustering, (2) classification, (3) regression, etc. The core concepts involved in data mining are as follows:

(1) Attributes: numerical, ordinal and nominal,
(2) Supervised and unsupervised learning and
(3) The practical process of data mining.

Attributes

A learning algorithm learns from the patterns in the input data and then can perform prediction. Input data is in the form of examples where each example consists of a set of attributes or dimensions. Each attribute should be independent of all other attributes, and therefore, its value cannot be computed on the basis of the values of other attributes. Attributes can be numeric, as real numbers which are continuous numbers (as age) or discrete values (as number of people). Closeness of two numbers is given by their difference.

Attributes can be ordinal values which are discrete, but in an order within them (as small, medium, large) with no specific or exact measurement involved.

Attributes can be nominal values which are discrete values with no particular order such as categorical values (color of eyes as black, brown or blue) with each category being independent and with no distance between any two categories being identifiable.

Algorithms that discover relationship between different attributes in a dataset are known as 'association rule algorithms.' Algorithms which are capable of predicting the value of an attribute based on the value of another attribute, based on its importance in clustering, are called 'attribute importance algorithms.'

Classification of learning is supervised learning and unsupervised learning.

In supervised learning, we have and use training datasets with a set of instances as example, for which the predicted value in known. Each such example consists of a set of input attributes and one predicted attribute. The objective of the algorithms is to build a mathematical model that can predict the output attribute value given a set of input attribute values.

Simple predictive models can be **decision trees** or **Bayesian belief networks** or **rule induction**.

Advanced predictive models can be **neural networks** and **regression**. **Support vector machine (SVM)** also provides advanced mathematical models for prediction.

The accuracy of prediction on new data is based on the prediction accuracy of the training data. When the target or predicted attribute is a categorical value, then the prediction model is known as classifier and the problem being tackled is called classification. On the other hand, when the attribute is a continuous variable it is called regression and the problem being tackled is also known as regression.

In the **unsupervised learning,** there is no predicted value to be learned. The algorithm for **unsupervised learning** simply analyzes the given input data and distributes them into clusters. A two-dimensional cluster can be identified such that the elements in each cluster are more closely related each other than to the elements in another cluster. Algorithms such as K-means clustering, hierarchical clustering and density-based clustering are various clustering algorithms.

1.9 Machine Learning Algorithms

We shall now examine the machine learning algorithms as a very brief overview. Later in Chap. 6, a detailed treatment will be provided.

Decision trees are simplest of learning algorithms which can perform classification, dealing with only nominal attributes. As an example, let us examine a very simple problem for a classification of gender-wise customers to be identified for purchasing a lottery ticket based on their income and age as follows.

Figure 1.3 shows an example of a decision tree to decide whether a customer will purchase a lottery ticket or not. Nodes are testing points, and branches are outcomes of the testing points. Usually, '>' (greater than) sign is placed to the right-hand side and '<' (less than) sign is placed to the left-hand side as indicated below:

It is easy to convert a decision tree into a rule as follows:

If gender is male and income is $\geq$100,000, then 'Yes', else 'No'.

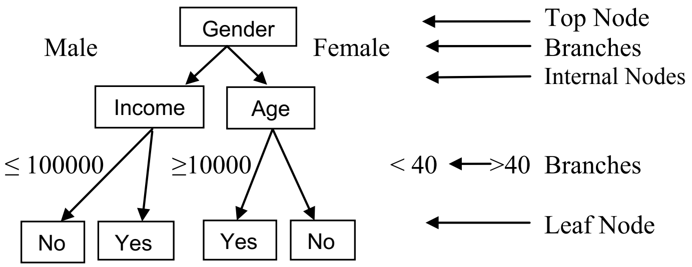

Fig. 1.3 Decision tree example

If gender is female and age is ≤40, then 'Yes', else 'No'.

(a) **Clustering algorithms** are aiming at creating a cluster of data elements which are most closely related, more than the elements in another cluster. K-means is the most popular clustering algorithms, where K-randomly selected clusters are seeded where K is the number of predefined clusters. Each example element is then associated with Kth cluster whose center is the closest to that element. At the end of the iteration, the means of K-clusters are recomputed by looking at all the points associated with the cluster. This process is repeated until the elements do not move any more between the clusters.

(b) **Hierarchical clustering algorithm** has data points, and each data point starts as its own cluster. Next two points that are most similar are combined together into the parent's node. This process is continued and repeated until we have no points left to continue.

(c) **Density-based clustering algorithms** find out high-density areas that are separated from the low-density areas. The number of high-density clusters alone is counted as the total number of clusters, ignoring low-density areas as just noise.

(d) **Regression** If we have two points in two-dimensional space to be joined together as a line, it is represented by the linear equation $y = ax + b$. Same approach can be extended for higher-dimensional function that best fit multiple points in multidimensional space. Regression-based algorithms represent data in a matrix form and transform the matrix to compute the required parameters. They require numerical attributes to create predictive models. The squared error between the predicted and the actual values is minimized for all cases in the training datasets. Regression can be linear or polynomial in multiple dimensions. Logistic regression is the statistical technique utilized in the context of prediction.

(e) **Neural Networks** Neural networks can function as classifiers and also as predictive models. Multilayered perceptron (MLP) and radial basis function (RBF) are the most common neural networks. A multilayered perceptron (MLP) consists of many layers beginning an input layer as shown in Fig. 1.4. The number of inputs is the same as the number of attributes. The input values may be varying between -1 and 1 depending on nature of transformation function used by the node. Links in the network correspond to a weight by which the output from the

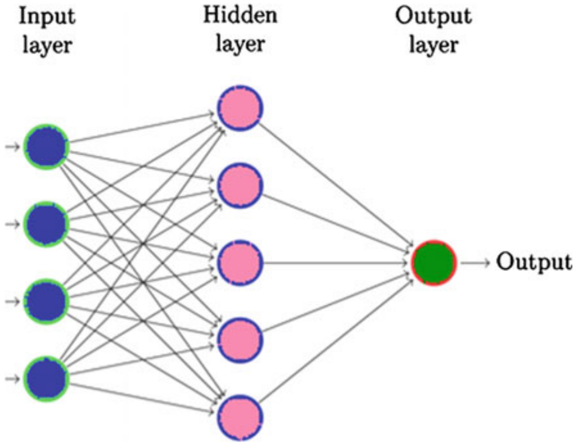

Fig. 1.4 Multi layered perceptron (three layers)

node is multiplied. The second layer is known as hidden layer in a three-layered network.

The input to a given node is the sum of the outputs from two nodes multiplied by the weight associated with the link. The third layer is output layer, and it predicts the attribute of interest. Building a predictive model for MLP comprises of estimating the weights associated with each of the links. The weights can be decided by a learning procedure known as back-propagation. Since there is no guarantee that global minimum will be found by this procedure, the learning process may be enhanced to run in conjunction with some optimization methodologies such as genetic algorithms.

In contrast, in radial basic function or RBF, the data is clustered into *K*-clusters using *K*-means clustering algorithm. Each cluster then corresponds to a node in the network, the output from which is dependent on the nearness or proximity of input to the center of the node the output from this layer is finally transformed into the final output using Weights. Learning the weights associated with such links is a problem of linear regression.

Modular Neural Networks

Modular constructive neural networks are more adaptive and more generalized in comparison with the conventional approaches.

In applications such as moving robots, it becomes necessary to define and execute a trajectory to a predefined goal while avoiding obstacles in an unknown environment. This may require solutions to handle certain crucial issues as overflow of sensorial information with conflicting objectives. Modularity may help in circumventing such problems. A modular approach, instead of a monolithic approach, is helpful since it combines all the information available and navigates the robot through an unknown environment toward a specific goal position.

(f) Support vector machine (SVM) and **relevance vector machine (RVM)** algorithms for learning are becoming popular. They come under kernel methods, and they combine principles from statistics, optimization and learning in a sound mathematical framework capable of solving large and complex problems in the areas of intelligent control and industry, information management, information security, finance, business, bioinformatics and medicine. If we consider a two-dimensional space with a large number of points, there exist a large number of lines that can be used to divide the points into two segments; let us call these lines as separating lines. Now we define ‘margin’ as the distance between a separating line and a parallel line that passes through the closest point to the lines. SVM selects the line that has the maximum margin associated with it. Points that this second parallel line passed through are known as **support vector points**. This model can be extended to be generalized for multiple dimensions, and its performance was observed to be very good, with output variables either as continuous or as discrete containing two values.

(g) Bayesian algorithms are algorithms based on probability theory. Naïve Bayesian classifier is one such simple but good classifier. In this algorithm, it is assumed that input attributes are independent among themselves and the prediction is made based on estimating the probability for the training data. Bayesian belief networks (BBNs) are directed acyclic graphs where a link denotes this conditional distributed function between the parent node and the child node.

Applications of SVMs

SVMs have a large variety of applications as follows:

(a) **Intelligent Control and Industrial Applications**: Intelligent signal processing aiming at quality monitoring, fault detection and control in an industrial process is performed by SVMs as part of quality monitoring tools is analyzing complex data patterns. Results showed that SVMs performed better than neural networks in these contexts. SVMs can handle feature spaces of high dimension. In regression problems, also SVMs have been deployed for control of a robot in a single plane.

(b) **Information Management**: In areas such as text classification for search process, SVMs have been deployed for statistical pattern analysis and inductive learning based on a large number of pattern instances. In the context of digital data classification and learning and ensemble kernel-based learning, the techniques using SVMs are also deployable.

(c) **Information Security**: In steganography, the JPEG images are utilized to Camouflage the secret data for sending secret messages. SVMs have been deployed to address this problem by constructing models by extracting correctly a set of features based on image characteristics for reliable validations of a set of JPEG images.

1.10 Evolutionary Computing (EC)

Evolutionary computing (EC) and evolutionary algorithm (EA) are denoting a kind of optimization methodology which is only inspired by (and not exactly the same as) nature-based phenomena such as biological evolution and also behavior of living organisms. These techniques which attempt to mimic or simulate evolution in nature for applying to real-world problems include genetic algorithm (GA), evolutionary programming (EP), evolutionary strategies (ES), genetic programming (GP), learning classifier system (LCS), differential evolution (DE), estimation of distributed algorithm (BDA), swarm intelligence (SI) algorithms like ant colony optimization (ACO) and practical swarm optimization (PSO). All these different algorithms have similar framework in algorithmic characteristics and also in their implementation details as three fundamental essential operations and two optional operations:

The first step is initialization followed by
the second step 'fitness evaluation and selection' followed by
the third step of 'population reproduction and variation.'

The new population is evaluated again, and iterations continued until a termination criterion is satisfied. In addition, some of these algorithms may have local search (LS) procedure and such algorithms are called mimetic algorithms (MAs). Further, techniques from machine learning have also been applied to EC to enhance their functionality and also vice versa; i.e., EC algorithmic technique is applied to machine learning (ML) algorithms. Thus, machine learning (ML) has become an effective and powerful area with the applications in wide-ranging fields of life [7].

1.11 Conclusion

In this chapter, we have presented the background and motivation for data science and Big Data Analytics. We have also presented a very brief introductory overview of all the major algorithms prevalent today for data mining and machine learning including neural networks, SVMs and evolutionary computing algorithms. The detailed treatment of the same can be found elsewhere. Luckily, we have ready-to-use open-source data mining and machine learning platforms such as Weka even though many other proprietary tools are also available (see http://www.donoz.org//computers/software/databases/Data_mining/Tool_Vendors/).

1.12 Review Questions

1. What are the current trends in IT? Explain in detail.
2. Where and how Big Data Analytics stands in current trends in IT?

3. How Big Data Analytics has business value? What are the possible sectors of IT in which Big Data Analytics can be deployed?
4. Explain a file processing model and development.
5. What is block chain technology and how it can be used?
6. Define the paradigm of Big Data and the role of Big Data Analytics in current business scenarios.
7. What is Hadoop? What is its importance? What are its benefits?
8. Which are the various scenarios and when Hadoop can be deployed?
9. Explain HDFS architecture and functionality.
10. What is Hadoop 2.0? What are its benefits over Hadoop 1.0?
11. Explain MapReduce framework and paradigm.
12. What is YARN? Why it is required?
13. Explain the various modules in Hadoop ecosystem.
14. Explain Spark and its architecture. What are its benefits?
15. What is machine learning? Compare data mining with machine learning?
16. Describe decision trees.
17. Describe clustering and its algorithm.
18. Describe regression and its application.
19. Explain neural networks and their categories.
20. Explain relevance vector machines.
21. Explain support vector machines.
22. What is evolutionary computing and what are its algorithms? Explain with examples.

References and Bibliography

1. C.B.B.D Manyika, *Big Data: The Next Frontier for Innovation, Competition and Productivity* (McKinsey Global Institute, 2011)
2. IBM, *Big Data and Netezza Channel Development* (2012)
3. http://hadoop.apache.org [online]
4. D.R John Gantz, *The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far in the East* (IDC, 2013)
5. http://ercoppa.github.io/HadoopInternals [online]
6. S. Acharya, S. Chellappan, *Big Data and Analytics* (2015)
7. A. Ghoting, *SystemML: Declarative Machine Learning on Map Reduce* (IBW Watson research center, 2011)
8. K.D. Strang, Z. Sun, Big data paradigm: what is the status of privacy and security? Ann. Data Sci. Heidelb. **4**(1), 1–17 (2017)
9. L. Cui, F.R. Yu, Q. Yan, When big data meets software-defined networking: SDN for big data and big data for SDN. IEEE Netw. **30**(1), 58 (New York, Jan–Feb 2016)
10. R.M. Alguliyev, R.T. Gasimova, R.N. Abbasli, The obstacles in big data process. Int. J. Modern Edu. Comput. Sci. **9**(3), (Hong Kong, Mar 2017)
11. F. Pourkamali-Anaraki, S. Becker, Preconditioned Data Scarification for Big Data with Applications to PCA and K-Means. IEEE Trans. Inf. Theory **63**(5), 2954–2974 (New York, 2017)
12. K. Vishwa, *Trends 2016: Big Data, IOT take the plunge* (Voice & Data, New Delhi, Apr 5, 2016

13. J. Kremer, K. Stensbo-Smidt, F. Gieseke, K.S. Pedersen, C. Igel, Big Universe, big data: machine learning and image analysis for Astronomy. IEEE Intell. Syst. **32**(2),16–22 (Los Alamitos, 2017)
14. R. Varshney, *Why Enterprises will En-route India for Big Data Analytics* (Express Computer, Mumbai, Jul 15, 2016)
15. G. Guruswamy, *How to Avoid the Common Big Data Follies in 2016* (Express Computer, Mumbai, Apr 22, 2016)
16. Big Data, Big Science: Students Share 'Big Data' Research at Poster Session US Fed News Service, Including US State News; Washington, D.C. (Washington, D.C, 01 May 2017)
17. Electronics and Telecommunications Research Institute; Researchers Submit Patent Application, *Big Data Distribution Brokerage System Using Data Verification and Method Thereof, for Approval (USPTO 20170140351) Information Technology* (Newsweekly, Atlanta, Jun 6, 2017), p. 910
18. P. Lade, R. Ghosh, S. Srinivasan, Manufacturing analytics and industrial internet of things. IEEE Intell. Syst. **32**(3), 74–79 (Los Alamitos, 2017)
19. *Research and Markets; Securing Big Data Infrastructure: An Evolving Market Ecosystem-Research and Markets Information Technology* (Newsweekly, Atlanta, Feb 23, 2016), p. 453
20. N.A. Shozi, J. Mtsweni, Big data privacy and security: a systematic analysis of current and future challenges, in *International Conference on Cyber Warfare and Security*; Reading: 296-XI. (Academic Conferences International Limited, Reading, 2016)
21. *Big Data in Leading Industry Verticals: Retail, Insurance, Healthcare, Government, and Manufacturing 2015–2020—Research and Markets* (Business Wire, New York, 27 Jan 2016)
22. *Securing Big Data Infrastructure: An Evolving Market Ecosystem* (PR Newswire, New York, 08 Feb 2016)
23. *Big data report 2016—Global Strategic Business Report 2014–2022: The Need to Turn Big Data' Into Big Advantage Drives Focus on Big Data Technologies & Services NASDAQ OMX's News Release Distribution Channel* (New York, 19 Dec 2016)
24. K. Yang, H. Qi, H. Li, K. Zheng, S. Zhou, et al., An efficient and fine-grained big data access control scheme with privacy-preserving policy. IEEE Internet Th. J. **4**(2), 563–571 (Piscataway, 2017)
25. C.P. Chullipparambil, *Big Data Analytics Using Hadoop Tools* (San Diego State University, ProQuest Dissertations Publishing, 2016). 10106013
26. M. Pascalev, Privacy exchanges: restoring consent in privacy self-management. Eth. Inf. Technol. **19**(1), 39–48 (Dordrecht, 2017)
27. W. Feng, E.A. Mack, R. Maciewjewski, Analyzing entrepreneurial social networks with big data wang. Ann. Am. Assoc. Geogr. **107**(1), 130–150 (Washington, 2017)