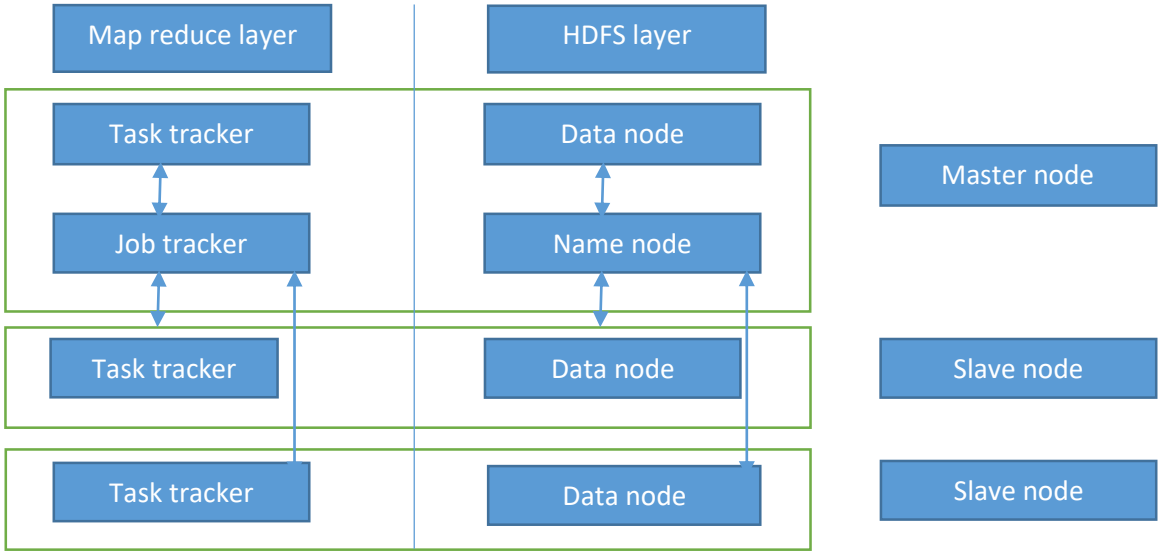
Big Data Analytics Notes

1. Explain Core component of Hadoop.
    - Hadoop distributed file system (HDFS) :
        a) HDFs is storage component of Hadoop.
        b) It is designed to store very large data sets across multiple machines.
        c) Features
            i. Distributed storage: Data is distributed across multiple nodes.
            ii. Fault tolerance: It is designed to handle h/w failures.
            iii. Scalability: it can scale horizontally by adding more machines.
    - Map reduce :
        a) Map reduce is the processing component of Hadoop.
        b) It is programming model and engine for processing and generating large datasets.
        c) The first operation is to read data from database and inserting operation of data into suitable format for analysis. It is known as mapping.
        d) The next operation is reduction, it usually reduces available map to set of proper values.
    - YARN :
        a) YARN stands for yet another resource negotiator.
        b) YARN is resource management layer of Hadoop.
        c) It allows multiple data processing engines like map Reduce, Apache spark, and Apache flink to share resources.
        d) YARN manages and allocates memory to applications running on Hadoop cluster.
    - Hadoop common :
        a) Hadoop common contain libraries, utilities, APIs that are shared across various Hadoop modules.
        b) Hadoop common is named as common as it provides required common functionalities.
        c) Hadoop common ensures compatibility among different Hadoop modules & components.

2. Explain Hadoop architecture.



Hadoop basically has master-slave architecture for storing data and distributed processing of data by using map reduce and HDFS methods.

a) Name Node :
   Name node represents all files and directory which is used in namespace.
b) Data node :
   Data node helps you to manage states of an HDFS node and allows you to cooperate with its blocks.
c) Master node :
   Master node allows you to conduct parallel processing of data by use Hadoop map reduce.
d) Slave node :
   Slave nodes are supplementary machines in the Hadoop cluster which permits you to store data to conduct complex calculations.
e) Features :
   i. Suitable for big data analysis
   ii. Scalability
   iii. Fault tolerance

3. Difference between traditional data and big data.

	Traditional data	Big data
1	Traditional data is generated in enterprise level.	Big data is generated in outside & enterprise level.
2	Its volume ranges from gigabytes to terabytes.	Its value ranges from petabytes to zetabytes.
3	Traditional database system deals with structured data.	Big data system deals with structured, semi structured and unstructured.
4	Traditional data is generated per hour or per day or more.	Big data is generated mainly per second.
5	Data integration is easy.	Data integration is difficult.
6	Traditional data is in manageable form.	Big data is in huge volume which becomes un manageable.

4. Explain types of big data.
   a) Unstructured :
      i. Any data with unknown form or the structure is classified as unstructured data.
      ii. Typical example of unstructured data is data source containing combination of text files, image, and videos like search in google engine.
      iii. Being huge in size unstructured data creates multiple challenges in terms of processing it.
      iv. Sources of unstructured :
         - Web pages
         - Images
         - Videos
         - Reports

   b) Structured :

i. Any data that can be stored, accessed and processed in form of fixed format is termed as structured data.
ii. Structured data is usually stored in well-defined schema such as databases. It is generally tabular with column & rows.
iii. SQL is used to manage structure data stored in databases.
iv. Sources :
- SQL databases
- Spreadsheets such as Excel
- Online forms
- Sensors such as RFID tags.

c) Semi structured :
i. Semi structured is third type of big data. Semi structured data can contain both structured & unstructured forms of data.
ii. Semi structured data is data that does not conform to data model but has some structure.
iii. Sources :
- Emails
- TCP/IP packets
- Zipped files
- Web pages.

5. Explain characteristics of big data.
a) Volume :
- Big data is used to refer to enormous amounts of information which contains terabytes petabytes of data.
- The volume of data is projected to change significantly in coming years.
- Volume (data at rest)
  - Terabytes, Petabytes
  - Records, Arch
  - Tables/files.

b) Veracity :
i. Veracity describes whether data can be trusted.
ii. Veracity arises due to high volume of data that brings incompleteness & inconsistency.
iii. Veracity (data in doubt)
* Trustworthiness
* Accurate
* Availability

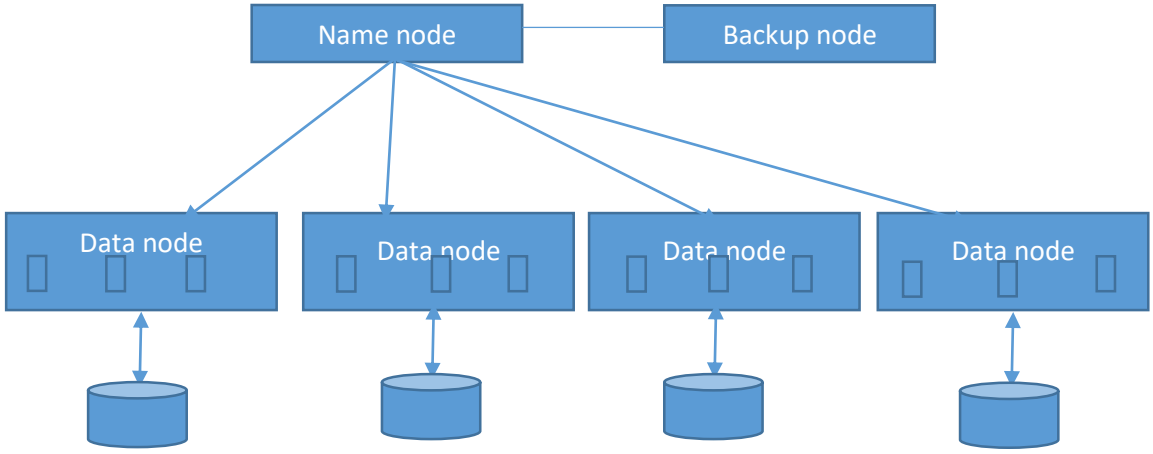
c) Velocity :
i. Velocity is speed in which data grows, process & becomes accessible.
ii. The flow of data is massive & continuous.
iii. Velocity (Data in motion)
* Streaming
* Processes
* Batch

d) Value :
i. It refers to turning data into value. By turning big data into value, business may generate revenue.

ii. After addressing volume, velocity, variety, veracity which takes lot of time, effort, resources you want to be sure of getting value from it.

iii. Value (Data into money)
* Statistical
* Events
* Correlations

6. Explain HDFS architecture.



- HDFS stands for Hadoop distributed file system.
- HDFS is primary storage system used by Hadoop applications.
- HDFS has become key tool for managing pools of big data & supporting big data analytics applications.
- The architecture used by HDFS is also known as master/slave architecture.

Advantages

i. High scalability
ii. Low cost

Disadvantages

i. Still rough : S/W under active development
ii. Cluster management is high

Features:

i. Data integrity
ii. Data locality

7. How Hadoop goals are covered by HDFS?
   1. Fault tolerance :
      a) HDFs achieves fault tolerance by replicating data across multiple nodes in cluster.
      b) If one node or replica fails, system can retrieve data from another replica.
   2. Scalability :
      a) HDFS is highly scalable and can handle large amount of data by distributing it across multiple nodes.

b) As data size increases more nodes can be added to Hadoop cluster.

3. Ease of use :
   a) HDFS is designed to be easy to use especially for applications that require batch processing.
   b) If provides simple file system interface, making is easy to developers.
4. Securing :
   a) HDFS provides basic security features such as access control list (ACLS) and file permissions.
5. High throughput :
   a) HDFS is optimized for high-throughput data access.
   b) It can efficiently handle large files and is well-suited for batch processing ---- such as map reduce jobs.

8. What is map reduce? Role of combiner with ex.
   a) Map reduce algorithm contains two important tasks: map & reduce.
   b) Map task takes set of data & converts it into another set of data.
   c) Reduce task takes output from map as input and combines those data tuples into smaller set of tuples.
   d) Phases :

i.	Input phase	:	Records input file & sends parsed data to mapper
ii.	Intermediate keys	:	Keys value pairs generated by mapper are known as IK
iii.	Map	:	takes series of key value pairs & processes each one of them to create more.
iv.	Combiner	:	Groups similar data from map phase
v.	Shuffle & sort	:	First step for reducing. It downloads key value pairs in local m/c, where reducing is present.
vi.	Reducer	:	Takes key value pairs as input & reduces each one of them.

Combiner:

a) Combiner always works in between mapper and reducer.
b) The output produced by mapper is intermediate output which is massive in size. If we directly feed this huge output to reducer, then it will result in network congestion.
c) So to minimize n/w congestion combiner is placed in between mapper & reducer.
d) It is not necessary to put combiner in map reduce program, it is optional.

9. Explain natural join & grouping & aggregation & relational algebra.

- Union :
  a) Map step :
     - Input : Key-value pairs (key, value) representing tuples from two relations.
     - Mapper fn : Emit key-value pairs with common key and tuple as value.

  b) Reduced step :
     - Input : Key-value pairs with key is common keys & values are tuples.
     - Reducer f^n : Emit tuples associated with common key.

- Selection :
  a) Map step :
     - Input : Key-value pairs (key, value) representing tuples.

- Mapper f^n : Apply selection condition to each tuple.
  Emit key-value pairs only for tuples satisfy condition.

b) Reduce step: Output from map step is result. No reduce step.

- Intersection :
  a) Map step :
     - Input : Key-value pairs (key, value) representing tuples from two relations.
     - Mapper f^n : Emit key-value pairs with common key and tuples as value.

  b) Reduce step :
     - Input : Key-value pairs where key is common & values are tuples.
     - Reducer f^n : If tuples are associated with common key from both relations, emit those tuples.

- Projection :
  a) Map step :
     - Input : Key-value pairs (key, value) representing tuples from relations.
     - Mapper f^n : Apply projection operation to each tuple.
       Emit key-value pairs with modified tuple that includes only selected attributes.

  b) Reduce step :
     - No reduce needed.
     - Output from map is result.

10. SQL vs NOSQL. How to overcome traditional system.

Sr no.	SQL	NOSQL
1	These databases have fixed or static or predefined schema.	These databases have a dynamic schema.
2	These databases are not suited for hierarchical data storage.	These databases are best suited for hierarchical data storage.
3	These databases are best suited for complex queries.	These databases are not so good for complex queries.
4	These are vertically scalable.	These are horizontally scalable.
5	Follows ACID property.	Follows CAP (consistency availability, partition tolerance).
6	Ex. MySQL, oracle	Ex. Mongo DB, HBase

11. No sql data architecture with key value store.
    a) Architecture pattern is a logical way of categorizing data that will be stored on database.
    b) Nosql is type of database which helps to perform operations on big data.

    Architecture patterns of Nosql :
    1. Key value store database
    2. Column store database
    3. Document database
    4. Graph database

1. Key value store database :
    - This model is one of the most basic models of Nosql database.
    - As per name, the data is stored in form of key value pairs.
    - The key is usually a sequence of strings, integers or characters but can also be more advanced data type.
    - The key value pairs storage databases generally store data as hash table where each key is unique.
    - The value can be of any type JSON, BLOB (binary large object), strings etc.,

    - Limitations :
        - Complex queries can delay performance.
        - Data can be involve in many to many relationship, which may collide.
    - Advantages :
        - Can handle large amount of data
        - Easy retrieval of data
    - Example :
        - Dynamo DB
        - Berkeley DB

2. Column store Database :
    - Column oriented databases work only on columns
    - They store large amount of data into columns together.
    - Every column is treated separately, but individual column may contain multiple other columns.
    - Columns are mode of storage in this type.
    - Example :
        i. HBase
        ii. Cassandra

3. Document Database :
    - Document database fetches and accumulates data in form of key value pairs but here values are called as documents.
    - Documents can be stated as complex data structure.
    - Document can be a form of text, array, string, JSON, XML or any.
    - Example :
        i. Mongo Database
        ii. Couch Database

4. Graph Database :
    - Architecture pattern deals with storage and management of data in graph.
    - The objects are called as nodes & they are joined together by relationship called edges.
    - This pattern is used in social networks where there are large number of entities.
    - Example :
        i. Neo4J
        ii. Flock Database

12. Issues & challenges in data stream query processing.
    - Limited resources :
        i. Challenge : Data streams are often high volume & continuous, which requires processing in real-time.

ii. Issue : Limited computing resources can lead to challenges in handling volume & velocity.

- High throughput :
  i. Challenge : Data streams may involve a large number of events per second.
  ii. Issue : Ensuring high throughput in processing is critical.

- Low latency :
  i. Challenge : Real-time processing demands low latency for timely action.
  ii. Issue : Delays in processing can result in outdated information.

- Scalability :
  i. Challenge : systems must be able to scale horizontally to handle increasing data.
  ii. Issue : Without scalable architectures, processing may decrease.

- Data quality & cleaning :
  i. Challenge : Incoming data can be noisy or contain error.
  ii. Issue : Ensuring data quality & implementing mechanisms for cleaning & filtering noisy data are crucial.

13. Data stream management system :
    - DSMS is an acronym for data stream management system.
    - It is nothing but software application just like DBMS but it involves processing and management of continuously slowing data steam rather than static data like excel or pdf.
    - Just like DBMS, DSMS also provides wide range of operations like storage, processing, analyzing, integration.
    - There are wide range of DSMS applications available in market like Apache flint, Apache kafka, Apache storm, etc.
    - DSMS process 2 type of quires.
      i. Standard
      ii. Adhoc
    - Layers :
      i. Data source layer
      ii. Data ingestion layer
      iii. Processing layer
      iv. Storage layer
      v. Querying layer
      vi. Visualization & reposting layer
      vii. Integration layer.

      i. Data source layer :
         a. First layer of DSMS is data source layer
         b. Contains data sources which includes sensors, social media feeds, financial market, stock market, etc.

      ii. Data ingestion layer :
         a. Bridge between data source & processing layer.
         b. Main purpose is to handle flow of data.

iii. Processing layer :
   a. Heart of DSMS
   b. If process the data stream in real time.
   c. To perform processing, it uses engines like Apache flink or apache storm, etc.

iv. Storage layer :
   a. After processing, storing of data is needed.
   b. Storage layer consist of storages like NOSQL databases, distributed databases, etc.

v. Querying layer :
   a. As above it supports 2 types of queries, i) Adhoc ii) Standard.
   b. This layer provides tools which can be used for querying & analyzing stored data stream.

vi. Visualization & reposting layer :
   a. This layer provides tools to perform visualization like chart, pie chart, histogram, etc.
   b. On the basis of visualization, it helps to generate report.

vii. Integration layer :
   a. This layer is responsible for integrating DSMS applications with data warehouses, ML application, NPL application ETC.
   b. Helps to improve already present applications.

14. Explain Collaborative filtering.

- Collaborative filtering is a recommendation technique that makes automatic prediction about the performance of user by collecting preferences from many users (collaborating).
- Idea is to recommend items to user based on preferences & opinions of other similar users.
- Two types : i) User based collaborative filtering
  ii) Item based collaborative filtering

i) User based :
   - Recommends items based on preferences of users who are similar to target user.
   - Assumes that user who agreed in past can agree in future as well.

ii) Item based :
   - Recommends item that are similar to those which user liked or interacted with in past.
   - Assumes that if user on item, can like items similar to it.

Example: (Amazon)

1) User based :
   - If user A and user B have similar purchasing or browsing history, and user A likes a product than collaborative filtering algo predicts that use B can like this product too.

- For example : If user A & user B have both purchased books by some author & user A rates book highly user B might get recommendation for that book.

2) Item based :
   - If user has previously purchased item X, Y & Z then collaborative filtering algo suggests other items that are similar to X, Y & Z.
   - For example : If user buys electronic gadgets, then system might recommend smart phone that is similar to post purchases.

Recommendation steps :

1. User data collection
2. User similarity calculation
3. Item similarity calculation
4. Recommendation generation
5. Feedback and iteration

Advantages :

1. Measuring similarity
2. Cosine similarity
3. Rounding the data.
4. Normalizing dating

15. How recommendation is done based on properties of product?

- Recommendation based on product properties involve suggesting item to user based on characteristics, features, attributes.
- This type of recommendation system if often used in e-commerce platforms for enhance user experience.

Steps:

a) Data collection :
   - Collect and maintain datasets that include detailed information about each product.
   - This information can contain category, brand, color, size, etc.

b) User interaction data :
   - Collects data on user interactions like clicks, views, purchases, ratings.
   - Crucial for understanding user preferences.

c) Feature extraction :
   - Identify key features of products in database.
   - Selecting attribute that can influence user preference.

d) User profiling :
   - Create user profiles based on their interactions & preferences based on products.

e) Similarity calculation :
   - Calculating similarity between products.

f) Recommendation generation :
   - Generate recommendation for user.
   - Two types i) Content based
     ii) Collaborative

16. Explain data structure in R.

- R programming :
  - R programming is a language and also software environment for statistical computing & data analysis.
  - R is an open source programming language & it is available on widely used platforms. Example windows, mac

- Data structures :
  - Vector is an ordered collection of basic data types.
  - Vectors are one dimensional data structures.
- List :
  - List are collection of objects & each element can be of different data type.
  - List are also one-dimensional data structures.
- Data frames :
  - Data frames are two-dimensional tables similar to matrices.
  - Data frames are generic data objects of R which are used to store tabular data.
- Matrices :
  - Matrices are two-dimensional array with rows & columns, containing elements of same data type.
  - To create a matrix in R you need to use function called "matrix".
- Arrays :
  - Arrays are multi-dimensional extension of matrices can store data in more than two dimensions.
  - To create array in R in need to use function called "array ()".

17. Explain Operators used to form data subsets in R.
- Using [ ] operator :
  - Using [ ] operator, elements of vectors and observations from data frames can be accessed.
  - To neglect some indexes '-' is used to access all other indexes of vector or data frame.
- Using [[ ]] operator :
  - [[ ]] operator is used for subsetting of list-objects.
  - This operator is same as [ ] operators but [[ ]] selects only one element where [ ] selects more than 1 element in single command.
- Using $ operator :
  - $ operator can be used for lists and data frames in R.
  - Unlike [ ] operator, it selects only single observation at a time.
  - $ is applicable for only recursive objects.
- Using subset ( ) function :
  - Subset ( ) function in R programming is used to create subset of vectors, matrices, data frames based on conditions provided in parameters.
  - Syntax : subset (X, subset, select)
    a) X indicates object
    b) Subset indicates logical expression on basis of which subsetting is done.
    c) Select indicates column to select.

18. Explain Bloom filter.
- Bloom filter is space efficient probabilistic data structure.
- If is used to test whether an element is member of set or not.

- It may return false positives, but never returns false negatives.
- The basic idea behind bloom filter involves using multiple hash functions to map elements to fixed- sized array of bits.
- When adding an element to filter, the corresponding bits at hash function positions are set to 1.
- When checking for presence of element, filter checks bit at hash function position.
- If all bits are set to 1, then element is considered to be in the set.
- If any of bits are 0, then element is definitely not in set.

Operators:

1. Insertion :
   When an element is inserted, it is hashed using multiple hash function and corresponding bits in array are set to 1.
2. Membership Test :
   To check if element in the set, hash it with same hash function used using insertion & also check if all corresponding bits are set to 1.
3. Deletion :
   Bloom filters do no support deletion of operations once bit it set to 1, it cannot be reliable reset without affecting other elements.

   - The base data structure of bloom filter is bit vector.
   - Bloom filter consist of :
     1. An array
     2. Hash function
     3. A set

19. Data node vs name node.

Sr. No.	Parameters	Name node	Data node
1	Rate	It is master node	It is slave node
2	Responsibility	Stores meta data	Stores actual data block
3	In case of failure	If fails, entire, file system becomes inoperable	If fails, data can be replicated & accessed via other node
4	Number	1 per cluster	Multiple per cluster
5	Function	Tracks file location	Stores data block