USN | | | | | | | | | | |

Internal Assessment Test 3 – Dec 2024

Sub:	Big Data Analytics					Sub Code:	18CS71	Branch:	CSE
Date:	13/12/2024	Duration:	90 mins	Max Marks:	50	Sem / Sec:	7 –A/B/C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Use HiveQL for the following:	[10]	CO 4	L3

i) Create a table with a partition.-5 Marks

```
CREATE TABLE sales_data (
id INT,
product_name STRING,
amount DECIMAL(10, 2)
)
PARTITIONED BY (year INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

ii) Add, Rename and Drop a partition to a table-5 Marks

```
ALTER TABLE sales_data ADD PARTITION (year=2023)
LOCATION '/path/to/sales_data/2023';
```

Rename :

```
ALTER TABLE sales_data ADD PARTITION (year=2024)
LOCATION '/path/to/sales_data/2024';

ALTER TABLE sales_data DROP PARTITION (year=2023);
```

Drop:

```
ALTER TABLE sales_data DROP PARTITION (year=2023) PURGE;
```

		MARKS	CO	RBT
2 .a	Write a short note on Pig architecture design layers.	[05]	CO 6	L2

1.Pig Latin Layer (Language Layer)

Pig Latin is the language used in Pig for expressing data transformations. It is a simple, SQL-like scripting language that provides a high-level interface for working with data. Users write their data processing tasks using Pig Latin scripts, which are then converted into lower-level representations for execution.

Key features:

- It allows users to write complex data transformations using a more intuitive syntax than Java.
- Pig Latin scripts can include statements for loading, transforming, and storing data.

2.Parser Layer

- The parser is responsible for parsing Pig Latin scripts. It takes the Pig Latin statements as input and converts them into an internal logical plan. This plan is represented as a Directed Acyclic Graph (DAG), where each node represents a Pig operation (like loading, filtering, grouping, etc.). If the script is syntactically incorrect, the parser throws an error.

3.Optimizer Layer

Once the logical plan is generated, it goes through the optimizer layer. This layer performs various optimizations on the logical plan, such as removing unnecessary operations, reordering operations to improve performance, or combining operations. The aim is to make the plan more efficient before it's passed to the next stage.

4.Compiler Layer

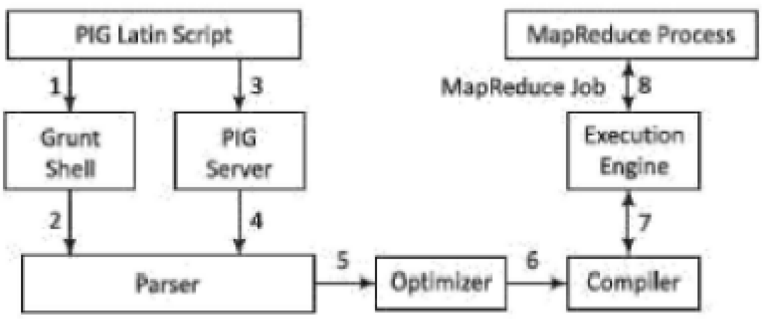
The compiler converts the optimized logical plan into an execution plan. It translates the logical operations into a series of physical operators, which can be executed on the Hadoop cluster. This plan is then passed to the execution engine.

5.Execution Layer

The execution layer interacts directly with the Hadoop infrastructure (MapReduce, YARN, or Tez). It executes the physical plan generated by the compiler. This layer divides the tasks into smaller jobs, which are then distributed across the Hadoop cluster for parallel execution. The execution engine handles the scheduling, resource allocation, and fault tolerance.

6.Hadoop Layer

The lowest layer is the actual Hadoop framework. Pig relies on Hadoop’s MapReduce (or YARN) for distributed data processing. It provides the infrastructure to store data in HDFS and perform computations across a large number of machines. Pig can run on top of Hadoop using MapReduce or even leverage newer execution engines like Apache Tez for performance improvements.

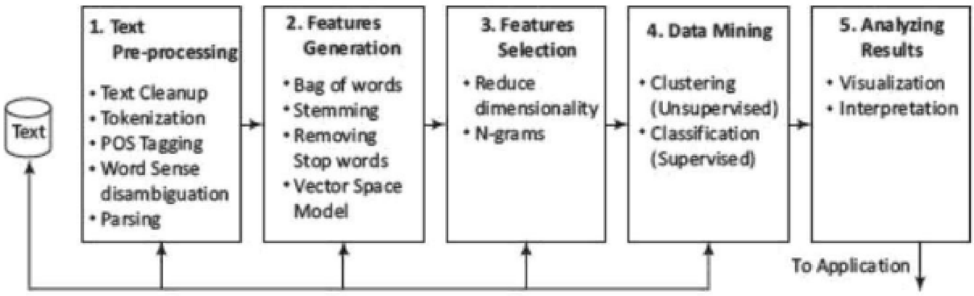


2.b	What is PIG in Big data? Explain the features of PIG.	[05]	CO 6	L2

Apache Pig is a high-level platform for processing and analyzing large datasets in the Hadoop ecosystem. It was developed by Yahoo! and is designed to simplify the process of writing complex data transformations and analytics using a higher-level scripting language called **Pig Latin**. Pig abstracts the complexities of writing low-level MapReduce code and provides a more efficient and flexible way to work with Hadoop.

Key Features of Apache Pig:

1. **High-Level Language (Pig Latin):**
   - **Pig Latin** is a simple, SQL-like language used for expressing data transformations. It is easy to learn and allows developers to write data processing tasks without needing to deal directly with the complexities of Hadoop's MapReduce framework.
   - Pig Latin supports common data processing operations like filtering, grouping, joining, and sorting, making it much more user-friendly than writing raw MapReduce code.
2. **Extensibility:**
   - Pig is extensible, meaning users can define their own functions to process data. This can be done by writing **User Defined Functions (UDFs)** in Java, Python, or other languages. These functions can be used in Pig Latin scripts to handle custom processing logic.
   - UDFs can be used to extend Pig's built-in functions or to implement domain-specific functionality.
3. **Optimization:**
   - Pig comes with an **optimizer** that optimizes queries automatically. It performs optimizations like filtering data early in the pipeline or eliminating redundant operations, which can lead to significant performance improvements.
   - The optimization layer works on the logical execution plan, allowing Pig to generate more efficient physical execution plans.
4. **Flexible Execution:**
   - Pig supports multiple execution engines. Initially, it used **MapReduce** for execution, but newer versions of Pig support **Tez** and **Apache Spark** for more efficient data processing.
   - It can run on a variety of cluster managers like **YARN** (Yet Another Resource Negotiator) or Hadoop's **MapReduce** framework, providing flexibility for different use cases.
5. **Data Flow and Procedural Model:**
   - Unlike SQL, which is declarative, Pig uses a **procedural** data flow model. This means the programmer specifies the steps required to process the data, as opposed to describing the final result.
   - This makes Pig more powerful for complex data transformations that require a series of steps, as compared to SQL's more static model.
6. **Interoperability with Hadoop:**
   - Pig is tightly integrated with **Hadoop** and can process data stored in **HDFS (Hadoop Distributed File System)**. It can also work with other Hadoop ecosystem tools, such as HBase, Hive, and more.
   - Pig can read and write data in multiple formats like **text files**, **Avro**, **Parquet**, **ORC**, and **JSON**.
7. **Schema Flexibility:**
   - Pig allows for **schema-less processing**, meaning you do not need to define a schema before processing the data. You can define the schema when reading the data or as part of the transformation.
   - This is useful when working with unstructured or semi-structured data, as it gives more flexibility in dealing with different data formats.
8. **Handles Complex Data Types:**

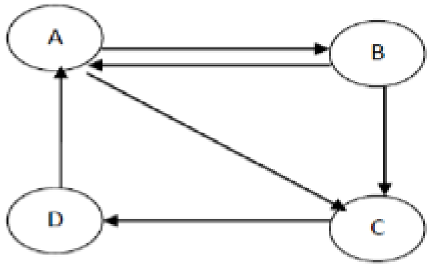
	○ Pig supports complex data types such as **tuples**, **bags**, and **maps**, which makes it more powerful when dealing with semi-structured or nested data formats. ○ This capability is useful in scenarios like processing JSON, XML, or other hierarchical data. 9. **Fault Tolerance:** ○ Like other components of the Hadoop ecosystem, Pig inherits Hadoop's fault-tolerant capabilities. Data is processed in parallel across multiple nodes in the cluster, and if a task fails, it can be retried on another node without data loss. 10. **Data Transformation & ETL:** ○ Pig is ideal for **ETL (Extract, Transform, Load)** tasks, as it can easily transform raw data into structured data suitable for analysis or loading into data warehouses. It supports operations like **join**, **group by**, **filter**, and **sort**, which are essential for ETL workflows.			
3 .a	Explain Five phases in a process pipeline in Text mining.  Text Mining is a rapidly evolving area of research. As the amount of social media and other text data grows, there is need for efficient abstraction and categorization of meaningful information from the text. The five phases for processing text are as follows: **Phase 1: Text pre-processing enables Syntactic/Semantic text-analysis and does the followings:** 1. Text cleanup is a process of removing unnecessary or unwanted information. Text cleanup converts the raw data by filling up the missing values, identifies and removes outliers, and resolves the inconsistencies. For example, removing comments, removing or escaping "%20" from URL for the web pages or cleanup the typing error, such as teh (the), do n't (do not) [%20 specifies space in a URL]. 2. Tokenization is a process of splitting the cleanup text into tokens (words) using white spaces and punctuation marks as delimiters. 3. Part of Speech (POS) tagging is a method that attempts labeling of each token (word) with an appropriate POS. Tagging helps in recognizing names of people, places, organizations and titles. English language set includes the noun, verb, adverb, adjective, prepositions and conjunctions. Part of Speech encoded in the annotation system of the Penn Treebank Project has 36 POS tags.4	[05]	CO 5	L2

	4. Word sense disambiguation is a method, which identifies the sense of a word used in a sentence; that gives meaning in case the word has multiple meanings. The methods, which resolve the ambiguity of words can be context or proximity based. Some examples of such words are bear, bank, cell and bass. 5. Parsing is a method, which generates a parse-tree for each sentence. Parsing attempts and infers the precise grammatical relationships between different words in a given sentence. **Phase 2: Features Generation is a process which first defines features (variables, predictors). Some of the ways of feature generations are:** 1. Bag of words-Order of words is not that important for certain applications. Text document is represented by the words it contains (and their occurrences). Document classification methods commonly use the bag-of-words model. The pre-processing of a document first provides a document with a bag of words. Document classification methods then use the occurrence (frequency) of each word as a feature for training a classifier. Algorithms do not directly apply on the bag of words, but use the frequencies. 2. Stemming-identifies a word by its root. (i) Normalizes or unifies variations of the same concept, such as speak for three variations, i.e., speaking, speaks, speakers denoted by [speaking, speaks, speaker-+ speak] (ii)Removes plurals, normalizes verb tenses and remove affixes. Stemming reduces the word to its most basic element. For example, impurification -+ pure. 3. Removing stop words from the feature space-they are the common words, unlikely to help text mining. The search program tries to ignore stop words. For example, ignores a, at, for, it, in and are. 4. Vector Space Model (VSM)-is an algebraic model for representing text documents as vector of identifiers, word frequencies or terms in the document index. VSM uses the method of term frequency-inverse document frequency (TF-IDF) and evaluates how important is a word in a document. When used in document classification, VSM also refers to the bag-of-words model. This bag of words is required to be converted into a term-vector in VSM. The term vector provides the numeric values corresponding to each term appearing in a document. The term vector is very helpful in feature generation and selection. Term frequency and inverse document frequency (IDF) are important metrics in text analysis. TF-IDF weighting is most common- Instead of the simple TF, IDF is used to weight the			

	importance of words in the document. **Phase 3: Features Selection is the process that selects a subset of features by rejecting** **irrelevant and/or redundant features (variables, predictors or dimension) according to defined** **criteria. Feature selection process does the following:** 1. Dimensionality reduction-Feature selection is one of the methods of division and therefore, dimension reduction. The basic objective is to eliminate irrelevant and redundant data. Redundant features are those, which provide no extra information. Irrelevant features provide no useful or relevant information in any context. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are dimension reduction methods. Discrimination ability of a feature measures relevancy of features. Correlation helps in finding the redundancy of the feature. Two features are redundant to each other if their values correlate with each other. 2. N-gram evaluation-finding the number of consecutive words of interest and extracting them. For example, 2-gram is a two word sequence, ["tasty food", "Good one"]. 3-gram is a three words sequence, ["Crime Investigation Department"]. 3. Noise detection and evaluation of outliers methods do the identification of unusual or suspicious items, events or observations from the data set. This step helps in cleaning the data. The feature selection algorithm reduces dimensionality that not only improves the performance of learning algorithms but also reduces the storage requirement for a dataset. The process enhances data understanding and its visualization. **Phase 4: Data mining techniques enable insights about the structured database that resulted** from the previous phases. Examples of techniques are: 1. Unsupervised learning (for example, clustering) (i) The class labels (categories) of training data are unknown (ii)Establish the existence of groups or clusters in the data Good clustering methods use high intra-cluster similarity and low inter-cluster similarity. Examples of uses - biogs, pattern and trends. 2. Supervised learning (for example, classification) (i) The training data is labeled indicating the class (ii)New data is classified based on the training set Classification is correct when the known label of test sample is identical with the resulting class computed from the classification model. Examples of uses are news filtering application, where it is required to automatically assign incoming documents to predefined categories; email spam filtering, where it is identified			

	whether incoming email messages are spam or not. Examples of text classification methods are Naive Bayes Classifier and SVMs. 3. Identifying evolutionary patterns in temporal text streams-the method is useful in a wide range of applications, such as summarizing of events in news articles and extracting the research trends in the scientific literature. Phase 5: Analysing results (i) Evaluate the outcome of the complete process. (ii) Interpretation of Result- If acceptable then results obtained can be used as an input for the next set of sequences. Else, the result can be discarded, and try to understand what and why the The process failed. (iii) Visualization - Prepare visuals from data, and build a prototype. (iv)Use the results for further improvement in activities at the enterprise, industry or institution.			
3.b	What are outliers? Discuss the reasons for having outliers in real time data. Outliers are data points that are numerically far distant from the rest of the points in a dataset, are termed as outliers. Outliers show significant variations from the rest of the points. Identification of outliers is important to improve data quality or to detect an anomaly There are several reasons for the presence of outliers in relationships. Some of these are: • Anomalous situation • Presence of a previously unknown fact • Human error (errors due to data entry or data collection) • Participants intentionally reporting incorrect data (This is common in self-reported measures and measures that involve sensitive data which participant doesn't want to disclose) • Sampling error (when an unfitted sample is collected from the population). Population means any group of data, which includes all the data of interest. For example, when analyzing 1000 students who gave an examination in a computer course, then the population is 1000. 100 games of chess will represent the population in analysis of 100 games of chess of a grandmaster.Sample means a subset of the population. Sample represents the population for uses, such as analysis and consists of randomly selected data. The reasons for having outliers in real time data. ● **Measurement or Data Entry Errors** ● **System or Process Changes** ● **External Shocks or Events** ● **Seasonality and Periodicity** ● **Anomalous Behavior or Rare Events** ● **Fraud or Malicious Activity** ● **Natural Variability** ● **Changes in Consumer Behavior** ● **Data Sampling Issues** ● **Statistical Noise**	[05]	CO 4	L2

4	Explain page ranking algorithm. Compute the Rank values for the nodes for the following network. Which is the highest rank node after computation?	[10]	CO 4	L3



PageRank is an algorithm developed by Google founders Larry Page and Sergey Brin that measures the relevance or importance of web pages on the Internet. Introduced in the late 1990s, it revolutionized web search by providing a method for ranking web pages based on their overall influence and popularity. The PageRank algorithm treats the web as a vast network of interconnected pages. Each page is represented on the web as a node with links between pages at the edges. The basic principle of PageRank is that a page is considered more important if other vital pages link it. The algorithm determines the initial PageRank value for each web page. This initial value can be uniform or based on certain factors, such as the number of incoming links to the page. The algorithm then repeatedly calculates the PageRank value of each page, taking into account the PageRank value of the pages that are related to the pages. During each iteration, the PageRank value of the page is updated based on the sum of the PageRank values of the incoming links. Pages with more inbound links have a more significant impact on the landing page's PageRank.

Nodes	Iteration 0	Iteration 1	Iteration 2
A	1/4	3/8	5/16 highest
B	1/4	1/8	3/16
C	1/4	2/8	4/16
D	1/4	2/8	4/16

→ assigning $\frac{1}{N} = \frac{1}{4}$ to all the nodes.

→ formula: $P_{t+1}(P_i) = \sum_{P_j} \frac{P_t(P_j)}{C(P_j)}$

→ Iteration I

For A we have B & D, so,

$$P_1(A) = \frac{P_0(B)}{C(B)} + \frac{P_0(D)}{C(D)} = \frac{\frac{1}{4}}{2} + \frac{1/4}{1} = \frac{3}{8}$$

For B, we have A

$$P_1(B) = \frac{P_0(A)}{C(A)} = \frac{1/4}{2} = \frac{1}{8}$$

for C, we have B & A

$$P_1(C) = \frac{P_0(A)}{C(A)} + \frac{P_0(B)}{C(B)} = \frac{1/4}{2} + \frac{1/4}{2} = \frac{2}{8}$$

for D, we have C.

$$P_1(D) = \frac{P_0(C)}{C(C)} = \frac{1/4}{1} = \frac{2}{8}$$

→ Iteration 2

for A we have B & D, so,

$$P_2(A) = \frac{P_1(B)}{C(B)} + \frac{P_2(D)}{C(D)} = \frac{1/8}{2} + \frac{2/8}{1} = \frac{5}{16}$$

for B we have A,

$$P_2(B) = \frac{P_1(A)}{C(A)} = \frac{3/8}{2} = \frac{3}{16}$$

for C we have B & A,

$$P_2(C) = P_1(A) + P_1(B) = \frac{3}{8} + \frac{1}{8} = 4$$

For C we have B & A,

$$P_2(C) = \frac{P_1(A)}{C(A)} + \frac{P_1(B)}{C(B)} = \frac{3/8}{2} + \frac{1/8}{2} = \frac{4}{16}$$

For D $P_2(D) = \frac{P_1(C)}{C(C)} = \frac{2/8}{1} = \frac{4}{16}$.

→ Therefore, we can say that A has the highest rank.

5 Explain Apriori algorithm. Solve Given problem on Apriori algorithm [10] CO 4 L4

Transaction ID	Rice	Pulse	Oil Milk	Apple
t1	1	1	1	0
t2	0	1	1	1
t3	0	0	0	1
t4	1	1	0	1
t5	1	1	1	0
t6	1	1	1	1

Define Parameters

- **Minimum Support Threshold**: Let's assume a minimum support of **50%** (i.e., an itemset must appear in at least 3 transactions).
- **Minimum Confidence Threshold**: Assume 70% for generating association rules.

1. Generate Frequent 1-itemsets

Count the occurrence of each item to find the itemsets that meet the minimum support threshold.

Item	Support Count	Support (%)
Rice	4	66.7%
Pulses	5	83.3%
Oil	4	66.7%
Milk	4	66.7%
Apple	4	66.7%

All items have support ≥ 50%, so they are **frequent 1-itemsets**.

Combine the frequent 1-itemsets and count their occurrences.

Itemset	Support Count	Support (%)
{Rice, Pulses}	4	66.7%
{Rice, Oil}	3	50%
{Rice, Milk}	3	50%
{Rice, Apple}	3	50%
{Pulses, Oil}	4	66.7%
{Pulses, Milk}	4	66.7%
{Pulses, Apple}	3	50%
{Oil, Milk}	4	66.7%
{Oil, Apple}	3	50%
{Milk, Apple}	3	50%

All itemsets have support ≥ 50%, so they are **frequent 2-itemsets**.

Generate Frequent 3-itemsets

Combine the frequent 2-itemsets and count their occurrences.

Itemset	Support Count	Support (%)
{Rice, Pulses, Oil}	3	50%
{Rice, Pulses, Milk}	3	50%
{Rice, Oil, Milk}	3	50%
{Pulses, Oil, Milk}	4	66.7%
{Pulses, Milk, Apple}	3	50%
{Oil, Milk, Apple}	3	50%

These are the **frequent 3-itemsets**.

These are the **frequent 3-itemsets**.

Generate Association Rules

Use the frequent itemsets to generate rules with confidence ≥ 70%.

Example:

- Rule: `{Pulses, Oil} → Milk`
  - Confidence = Support({Pulses, Oil, Milk}) / Support({Pulses, Oil})
  - Confidence = 4/4 = 100% (Rule is valid)

6.a	Write a short note on Collaborative filtering and Content based filtering for building recommendation systems.	[5]	CO 5	L2

Collaborative Filtering and **Content-Based Filtering** are two popular techniques used for building recommendation systems. Both aim to provide personalized recommendations to users, but they differ in how they make recommendations.

1. Collaborative Filtering

Collaborative filtering is based on the idea of leveraging the preferences or behaviors of similar users to recommend items. It assumes that if two users have agreed on one set of items, they are likely to agree on others as well.

- **User-Based Collaborative Filtering**: Recommends items by finding users similar to the target user and suggesting items they have liked.

<table>
<tr>
<td></td>
<td>

- **Item-Based Collaborative Filtering**: Recommends items similar to those the user has already liked, based on how other users have rated those items.

Advantages:

- Does not require item descriptions or content analysis.
- Can discover complex patterns based on user behavior.

Disadvantages:

- **Cold Start Problem**: Difficulty in recommending items for new users or new items that have no prior data.
- **Sparsity**: In large datasets, the number of interactions between users and items may be very sparse, making recommendations less accurate.

2. Content-Based Filtering

Content-based filtering recommends items based on the attributes of the items themselves and the preferences expressed by the user. This method uses the features of items (such as genre, author, keywords) and matches them with the user's past preferences.

- **Item Profiling**: Each item is represented by a set of features, and recommendations are made based on the similarity between these features and the user's preferences.
- **User Profiling**: The system builds a profile of user preferences based on the items they have interacted with in the past.

Advantages:

- Can recommend items even for new users or new items (no need for prior user interaction data).
- Works well when items have rich, descriptive content (e.g., movie genre, book author).

Disadvantages:

- Can lead to limited recommendations, as it only suggests items similar to those the user has already interacted with (lacks diversity).
- Requires detailed item metadata and feature extraction, which may not always be available.

</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.b</td>
<td>

Short note on Similarity measures like cosine similarity, Jaccard similarity etc.

Similarity measures are mathematical tools used to quantify the degree of similarity between two objects (e.g., documents, users, items) based on their features. These measures are commonly used in **recommendation systems**, **text mining**, and **data clustering**.

1. Cosine Similarity

Cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space. It is commonly used to measure document similarity in text mining.

</td>
<td>[5]</td>
<td>CO 5</td>
<td>L2</td>
</tr>
</table>

- **Formula**:
  cosine similarity=A · B // A // // B // \text{cosine similarity} = \frac{A \cdot B}{\|A\| \|B\|}
  where AA and BB are two vectors, · \cdot is the dot product, and // A // \|A\| and // B // \|B\| are the magnitudes of the vectors.

- **Range**: The value ranges from -1 (completely opposite) to 1 (completely similar), with 0 indicating no similarity.

Use Case: Commonly used in text mining to measure the similarity between two text documents based on the frequency of words.

2. Jaccard Similarity

Jaccard similarity measures the similarity between two sets by dividing the size of their intersection by the size of their union.

- **Formula**:
  Jaccard similarity=∣A∩B∣ ∣A∪B∣\text{Jaccard similarity} = \frac{|A \cap B|}{|A \cup B|}
  where AA and BB are two sets, and ∩\cap and ∪\cup represent the intersection and union of the sets, respectively.

- **Range**: The value ranges from 0 to 1, where 0 means no similarity and 1 means the sets are identical.

Use Case: Often used in situations where the data is binary or sparse, like comparing the presence or absence of items or features (e.g., user-item interactions).

3. Euclidean Distance

Euclidean distance measures the straight-line distance between two points in a multi-dimensional space.

- **Formula**:
  Euclidean distance=∑i=1n(xi−yi)2\text{Euclidean distance} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}
  where xix_i and yiy_i are the coordinates of the two points in n-dimensional space.

- **Range**: The value is always non-negative, with 0 indicating identical points and larger values indicating greater dissimilarity.

Use Case: Frequently used in clustering and classification tasks, such as K-means clustering.

CI **CCI** **HOD**

CO PO Mapping

Course Outcomes		Modules covered	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Investigate Hadoop framework and Hadoop Distributed File system.	**1**	2	0	2	2	3	0	0	0	0	0	0	0	0	0	0	3
CO2	Illustrate the concepts of NoSQL using MongoDB and Cassandra for Big Data.	**1,2**	2	3	2	3	3	0	0	0	0	0	0	0	0	2	0	3
CO3	Demonstrate the MapReduce programming model to process the big data along with Hadoop tools.	**3**	2	2	3	3	3	0	0	0	0	0	0	0	0	2	0	3
CO4	Use Machine Learning algorithms for real world big data.	**2,3,4**	2	3	3	2	3	0	0	0	0	0	0	0	0	2	0	3
CO5	Analyze web contents and Social Networks to provide analytics with relevant visualization tools.	**5**	2	3	3	3	3	0	0	0	0	0	0	0	0	2	0	3
CO6	Investigate Hadoop framework and Hadoop Distributed File system.	**5**	2	3	2	2	3	0	0	0	0	0	0	0	0	2	0	3

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				

PSO3	Apply software engineering methods to design, develop, test and manage software systems.
PSO4	Develop intelligent applications for business and industry