Experiment - 9

Aim: To perform Exploratory data analysis using Apache Spark and Pandas

Theory:

1. **What is Apache Spark and it works?**
   Apache Spark is an open-source platform that enables fast and scalable big data processing. It excels at handling large datasets across a distributed environment and is widely used for tasks like machine learning, real-time analytics, and SQL-based operations.

It supports multiple languages including Scala (its native language), Java, Python (via PySpark), R, and SQL.

Spark generally works in these stages:

1. **Driver Program Initialization**
   The user writes the code using Spark APIs. The Driver Program manages the execution, builds a Directed Acyclic Graph (DAG) from the code, and controls job flow.
2. **Resource Allocation by Cluster Manager**
   A Cluster Manager such as YARN, Mesos, or Spark's own manager assigns resources and launches worker nodes called Executors across the system.
3. **Task Distribution**
   Spark's DAG Scheduler breaks the job into individual tasks, which are then sent to the Executors for execution. These tasks may involve data loading, transformation, or saving.
4. **Task Execution and Result Collection**
   Executors process data primarily in memory, which speeds up performance. They cache intermediate data and return final results either to the Driver or storage.

Use Cases:

- Processing streaming or log data in real time
- Running machine learning algorithms on massive datasets
- Managing ETL (Extract, Transform, Load) workflows
- Analyzing structured and unstructured big data

2. **How data exploration done in Apache spark? Explain steps.**
   Data exploration using Apache Spark, especially with PySpark, helps to understand data before performing in-depth analysis or machine learning. It gives insight into data patterns, structure, and quality.:

1. **Importing Data**
   The process begins by loading data into Spark from files (like CSV, JSON, or Parquet), databases, or cloud sources using Spark's built-in connectors.
2. **Inspecting the Schema**
   After loading, the structure of the dataset is reviewed to check column names, data types, and nullability. This helps in validating and interpreting the data accurately.
3. **Previewing the Data**
   A few rows are displayed to get a general idea of the dataset. This helps in spotting incorrect entries, formatting issues, or missing values.
4. **Summary Statistics**
   Statistical details such as count, mean, min/max values, and standard deviation are generated to assess the distribution and detect anomalies like outliers.
5. **Detecting Missing Data**
   Identifying null or missing values is crucial for deciding how to clean the data—whether to fill, drop, or replace those entries.
6. **Analyzing Distribution**
   The frequency and distribution of values in categorical or numerical fields are examined. This helps detect class imbalance or commonly occurring values.
7. **Studying Correlations**
   Relationships between numeric features are analyzed using correlation to understand how closely they are related. This can support feature selection later.
8. **Filtering and Conditions**
   To dive deeper, data is filtered using logical conditions to focus on specific records, like those with a particular attribute or range.
9. **Sampling the Data**
   When dealing with massive datasets, a small portion is sampled for quicker exploration and visualization, saving computational effort.
10. **Preparing for Further Analysis**
    After exploration, data is cleaned and prepped for modeling or visualization using external tools, as Spark isn't primarily used for plots. The knowledge from this stage informs the next steps.

Conclusions:

Apache Spark is a robust and efficient distributed data processing system built for high-performance analysis at scale. Using a driver-executor model, it processes large volumes of data in memory and supports versatile APIs like RDDs, DataFrames, and Datasets. This makes it suitable for large-scale data manipulation across different programming environments.