Hadoop Mapreduce Working

Ahmed Ali Abid (2021065)

Sarim Ahmed (2021572)

Zaid Dandia (2021719)

1. Introduction

In the modern era of data-driven decision-making, data science has become a cornerstone of innovation and efficiency. This report outlines the comprehensive process undertaken to build and evaluate a data science pipeline for processing large-scale datasets. Using a distributed system (Hadoop) and advanced Python-based libraries, the project focuses on key aspects such as data ingestion, preprocessing, exploratory data analysis (EDA), visualization, and machine learning model execution.

The objective is to demonstrate the potential of combining distributed systems with machine learning to handle complex data workflows effectively. This report delves into the tools used, techniques implemented, and results obtained.

---

2. Tools Specification

Tools and Frameworks

1. **Hadoop Distributed File System (HDFS):** Used for distributed storage and processing of large datasets.
2. **Python Libraries:**
    - **Pandas:** For data manipulation and analysis.
    - **Matplotlib:** For generating visualizations.
    - **Scikit-learn:** For machine learning model development and evaluation.
    - **Subprocess Module:** For interaction with the HDFS file system.

Architecture

The architecture of the data processing pipeline is as follows:

1. **Data Ingestion:** Data files are stored in HDFS for scalable processing.
2. **Preprocessing:** Raw data is cleaned and validated using Python scripts.
3. **EDA and Visualization:** Key patterns and distributions are identified using statistical summaries and visualizations.
4. **Machine Learning:** A Decision Tree Classifier is trained and evaluated on preprocessed data.

---

3. Description of Techniques

3.1. Data Ingestion

Process:

- The raw dataset is stored in HDFS using `hdfs dfs -put` commands.
- Python scripts access the data via `hdfs dfs -cat` commands for processing.

Purpose:

- Efficient handling of large datasets through distributed storage.
- Enabling parallel processing for faster execution.

3.2. Data Preprocessing

Steps:

1. **Data Validation:** Ensures numerical features are valid and skips malformed rows.
2. **Feature Extraction:** Extracts relevant columns such as `sepal_length`, `sepal_width`, `petal_length`, and `petal_width`.
3. **Aggregation:** Computes mean, min, max, and count statistics for each feature.

Purpose:

- To ensure data quality and readiness for analysis.

Key Concepts:

- Feature engineering: Transforming raw data into usable features.
- Data cleaning: Removing invalid entries.

3.3. Exploratory Data Analysis (EDA)

Steps:

1. **Summary Statistics:** Descriptive statistics are computed for all features.
2. **Visualization:**
    - Histograms for feature distributions.
    - Scatter plots to identify correlations between variables.

Purpose:

- To understand dataset characteristics and relationships between features.

Key Concepts:

- Descriptive statistics and data visualization.
- Identifying patterns and outliers.

3.4. Machine Learning Model Execution

Steps:

1. **Data Splitting:** The dataset is split into training (80%) and testing (20%) subsets.
2. **Model Training:** A Decision Tree Classifier is trained using Scikit-learn.
3. **Evaluation:**
   - Accuracy score computation.
   - Confusion matrix generation.
   - Classification report (precision, recall, F1-score).

Purpose:

- To develop and validate a predictive model for classifying iris species.

Key Concepts:

- Decision Trees: A supervised learning algorithm for classification.
- Evaluation Metrics: Precision, recall, and F1-score.

---

4. Results and Outputs

4.1. EDA Outputs

Summary Statistics:

- petal_length   count:150,mean:3.76,min:1.00,max:6.90
- petal_width   count:150,mean:1.20,min:0.10,max:2.50
- sepal_length   count:150,mean:5.84,min:4.30,max:7.90
- sepal_width   count:150,mean:3.05,min:2.00,max:4.40

4.2. Machine Learning Outputs

Confusion Matrix:

```
[11  0  0]
 [ 0  8  1]
 [ 0  1  8]]
```

Metric	Iris-setosa	Iris-versicolor	Iris-virginica	Accuracy	Macro Avg	Weighted Avg
Precision	**1.00**	**0.89**	**0.89**	**0.93**	**0.93**	**0.93**
Recall	**1.00**	**0.89**	**0.89**	**0.93**	**0.93**	**0.93**
F1-Score	**1.00**	**0.89**	**0.89**	**0.93**	**0.93**	**0.93**
Support (Count)	**11**	**9**	**9**	**29**	**-**	**-**

5. References

1. Hadoop Documentation: https://hadoop.apache.org
2. Scikit-learn Documentation: https://scikit-learn.org
3. Matplotlib Documentation: https://matplotlib.org
4. Pandas Documentation: https://pandas.pydata.org

Conclusion

This project demonstrates the integration of distributed systems and machine learning to process and analyze data efficiently. Using Hadoop for scalable storage and Python for advanced analytics ensures a robust and efficient pipeline for data science workflows. Future improvements could include using advanced algorithms like Random Forests and employing tools like Apache Spark for greater scalability.