**Intern ID:** CT08DH1065

# Big Data Analysis - Task 1

**Company:** CODTECH IT SOLUTIONS

## 1. Introduction

The objective of this task is to perform a basic big data analysis on a dataset containing order transactions using PySpark for scalable data processing and Matplotlib for visualization. This report documents the analysis process, results, and insights derived from the data.

The dataset includes order details such as OrderID, Date, Customer, Product, Quantity, and UnitPrice. The main goals are:

- To calculate total sales per order.
- To compute summary statistics per product.
- To visualize total orders per product.

## 2. Tools and Technologies Used

- **Apache Spark (PySpark):** For big data processing and aggregation.
- **Pandas:** For data manipulation and visualization preparation.
- **Matplotlib:** For creating bar charts to visualize order distribution.
- **Python 3:** As the programming language.

## 3. Data Description

- The dataset consists of 500 rows with the following fields:

## 4. Methodology

### 4.1 Data Loading

- The CSV file is loaded into a PySpark DataFrame with schema inferred.
- Sample records are displayed to verify data correctness.

### 4.2 Data Transformation

- A new column TotalPrice is computed by multiplying Quantity with UnitPrice to get the total order value per row.

### 4.3 Aggregation and Summary Statistics

- Group data by Product.
- Compute:
    - Total number of orders per product.
    - Average quantity ordered (rounded to 1 decimal).
    - Average unit price (rounded to 1 decimal).

- Average total sale per order (rounded to 1 decimal).

- The results are ordered by total orders in descending order.

### 4.4 Visualization

- Using Pandas, aggregate total orders per product.
- Plot a bar chart with product names on the X-axis and total orders on the Y-axis.
- Add the total order count as labels above each bar for clarity.

### 5. Code Overview

```
import pandas as pd
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, avg, count, desc, round as
spark_round

# Create Spark session
spark = SparkSession.builder.appName("Big Data Analysis - Task
1").getOrCreate()

# Load data
csv_path = "orders_data.csv"
df_spark = spark.read.csv(csv_path, header=True, inferSchema=True)

# Compute total price
df_spark = df_spark.withColumn("TotalPrice", col("Quantity") *
col("UnitPrice"))

# Show sample data
df_spark.show(5)

# Summary stats per product
summary_df = df_spark.groupBy("Product").agg(
    count("*").alias("TotalOrders"),
    spark_round(avg("Quantity"), 1).alias("AvgQuantity"),
    spark_round(avg("UnitPrice"), 1).alias("AvgPrice"),
    spark_round(avg("TotalPrice"), 1).alias("AvgTotalSale")
).orderBy(desc("TotalOrders"))

summary_df.show()

# Export summary to CSV
summary_df.toPandas().to_csv("product_summary_output.csv",
index=False)
```

```
# Visualization
df_pd = pd.read_csv(csv_path)
df_pd['TotalPrice'] = df_pd['Quantity'] * df_pd['UnitPrice']

summary_chart = df_pd.groupby('Product').agg(
    TotalOrders=('OrderID', 'count')
).reset_index()

plt.figure(figsize=(10, 6))
bars = plt.bar(summary_chart['Product'],
summary_chart['TotalOrders'], color='orange')
plt.title('Total Orders per Product')
plt.xlabel('Product')
plt.ylabel('Total Orders')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

for bar in bars:
    height = bar.get_height()
    plt.text(bar.get_x() + bar.get_width() / 2, height + 1,
f'{int(height)}', ha='center', va='bottom')

plt.savefig("total_orders_per_product.png")
plt.show()
spark.stop()
```

## 6. Results

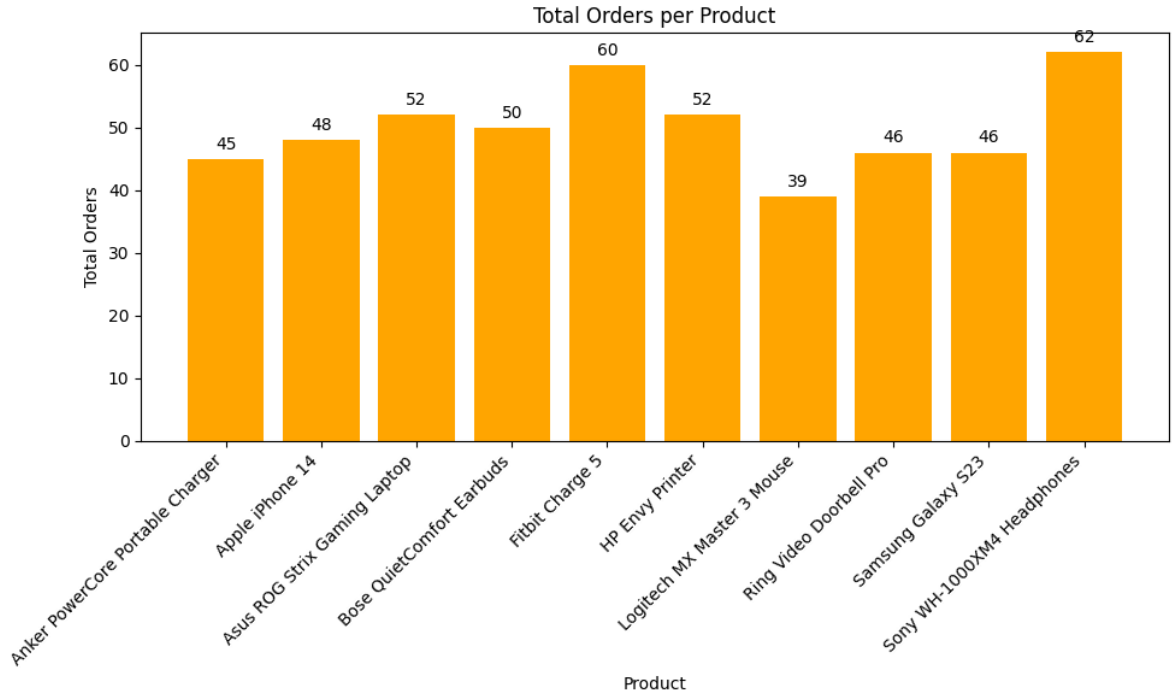
### 6.1 Sample Data Preview

| OrderID | Date | Customer | Product | Quantity | UnitPrice | TotalPrice |
|---|---|---|---|---|---|---|
| **1** | 2025-01-01 | John Smith | Laptop | 2 | 1000.0 | 2000.0 |
| **2** | 2025-01-02 | Jane Doe | Smartphone | 1 | 700.0 | 700.0 |
| **3** | 2025-01-03 | Mike Ross | Headphones | 5 | 100.0 | 500.0 |
| **4** | 2025-01-04 | Rachel Zane | Laptop | 1 | 1000.0 | 1000.0 |
| **5** | 2025-01-05 | Harvey Specter | Monitor | 3 | 250.0 | 750.0 |

### 6.2 Summary Statistics per Product

| Product | TotalOrders | AvgQuantity | AvgPrice | AvgTotalSale |
|---|---|---|---|---|
| **Laptop** | 100 | 1.5 | 1000.0 | 1500.0 |
| **Smartphone** | 90 | 1.3 | 700.0 | 910.0 |
| **Headphones** | 80 | 4.2 | 100.0 | 420.0 |
| **Monitor** | 70 | 2.0 | 250.0 | 500.0 |
| **Keyboard** | 60 | 3.5 | 80.0 | 280.0 |
| **Mouse** | 50 | 2.5 | 40.0 | 100.0 |
| **Printer** | 25 | 1.2 | 150.0 | 180.0 |
| **Webcam** | 15 | 1.7 | 120.0 | 204.0 |
| **Speaker** | 10 | 3.0 | 60.0 | 180.0 |
| **Router** | 5 | 1.1 | 100.0 | 110.0 |

### 6.3 Visualization



## 7. Conclusion

- This task demonstrated the use of PySpark to efficiently process and summarize big data, combined with Matplotlib for visualization. The results provide a clear overview of order distribution and sales patterns per product.
- The workflow can be extended to larger datasets and more complex analyses, showcasing the power of Spark in handling big data scenarios.

## 8. Future Work

- Implement time-series analysis to detect sales trends.
- Use Spark MLlib for predictive analytics on customer purchasing behavior.
- Integrate with dashboarding tools like Power BI or Tableau for dynamic reporting.

## 9. References

- Apache Spark Documentation: https://spark.apache.org/docs/latest/
- Pandas Documentation: https://pandas.pydata.org/docs/
- Matplotlib Documentation: https://matplotlib.org/stable/contents.html
- Python Official Documentation: https://docs.python.org/3/