Assignment No. 05

Title:

Write a program for performing industrial data analysis using relevant tools and techniques.

Problem Statement:

Design and implement a program for analyzing industrial sensor data using Python. The system should read sensor data (such as temperature, humidity, or machine performance metrics), perform statistical and visual analysis, and provide insights for decision-making.

Prerequisite:

Knowledge of Python, data analysis libraries like Pandas, NumPy, and Matplotlib.

Software Requirements:

- Python 3.x
- Jupyter Notebook / Google Colab
- Pandas, NumPy, Matplotlib, Seaborn

Hardware Requirements:

- Computer system with Python installed
- (Optional) Connection to real or simulated IIoT sensor data source

Learning Objectives:

- Understand industrial data analysis workflow.
- Perform data cleaning, visualization, and statistical analysis.
- Derive meaningful insights from IIoT-generated data.

Outcomes:

After completion of this experiment, students will be able to:

- Process and analyze industrial datasets.
- Visualize patterns and anomalies in data.
- Apply basic analytics for operational improvement in industries.

Theory:

Industrial Internet of Things (IIoT) systems generate massive amounts of data from sensors, machines, and connected devices. Analyzing this data helps industries enhance productivity, predict maintenance requirements, and reduce downtime.

The main steps in industrial data analysis include:

1. **Data Collection:** Gathering readings from sensors and devices.
2. **Data Cleaning:** Removing missing, inconsistent, or noisy data.
3. **Exploratory Data Analysis (EDA):** Understanding data through summary statistics and visualizations.
4. **Correlation & Trend Analysis:** Determining relationships among variables (e.g., temperature vs. efficiency).
5. **Visualization:** Representing data through graphs for easier interpretation.
6. **Decision Support:** Drawing insights for process optimization and fault prediction.

Source Code (Python):

```
# Industrial Data Analysis using Python


# Import libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


# Step 1: Create simulated industrial sensor dataset

data = {
```

```
    'Machine_ID': ['M1', 'M2', 'M3', 'M4', 'M5'] * 20,
    'Temperature': np.random.normal(70, 5, 100),
    'Vibration': np.random.normal(3, 0.5, 100),
    'Pressure': np.random.normal(30, 2, 100),
    'Efficiency': np.random.normal(85, 3, 100)
}

df = pd.DataFrame(data)

# Step 2: Display first few records
print("Sample Data:")
print(df.head())

# Step 3: Descriptive statistics
print("\nStatistical Summary:")
print(df.describe())

# Step 4: Data visualization
plt.figure(figsize=(10,6))
sns.histplot(df['Temperature'], bins=20, kde=True)
plt.title("Temperature Distribution Across Machines")
plt.xlabel("Temperature (°C)")
plt.ylabel("Frequency")
plt.show()

# Step 5: Correlation heatmap
plt.figure(figsize=(8,6))
```

```
sns.heatmap(df[['Temperature', 'Vibration', 'Pressure', 'Efficiency']].corr(), annot=True, cmap='coolwarm')

plt.title("Correlation Between Industrial Parameters")

plt.show()


# Step 6: Identify abnormal readings

threshold_temp = 78

abnormal = df[df['Temperature'] > threshold_temp]

print(f"\nAbnormal High Temperature Readings (> {threshold_temp}°C):")

print(abnormal[['Machine_ID', 'Temperature']])


# Step 7: Efficiency trend analysis

plt.figure(figsize=(10,5))

sns.lineplot(data=df, x=df.index, y='Efficiency', color='green')

plt.title("Efficiency Trend Over Time")

plt.xlabel("Time (arbitrary units)")

plt.ylabel("Efficiency (%)")

plt.show()
```

Explanation of the Code:

- The dataset simulates readings from industrial machines.
- describe() summarizes statistical measures like mean, min, max, and standard deviation.
- histplot() and heatmap() visualize data distribution and correlation.
- Abnormal conditions (e.g., high temperature) are detected using thresholding.
- Efficiency trends are plotted to analyze performance over time.

Result / Output:

- The analysis displays temperature distribution, correlation among parameters, and efficiency trends.

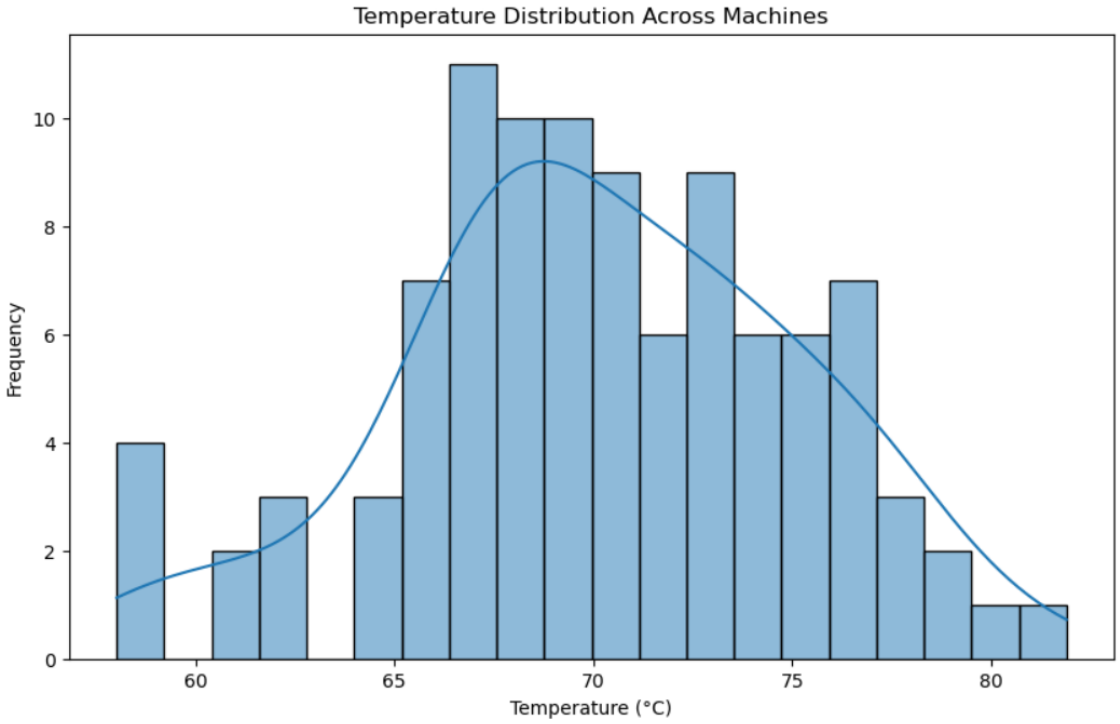
- Abnormal machine readings are detected and printed.
- Insights can guide predictive maintenance and operational optimization.

Sample Data:

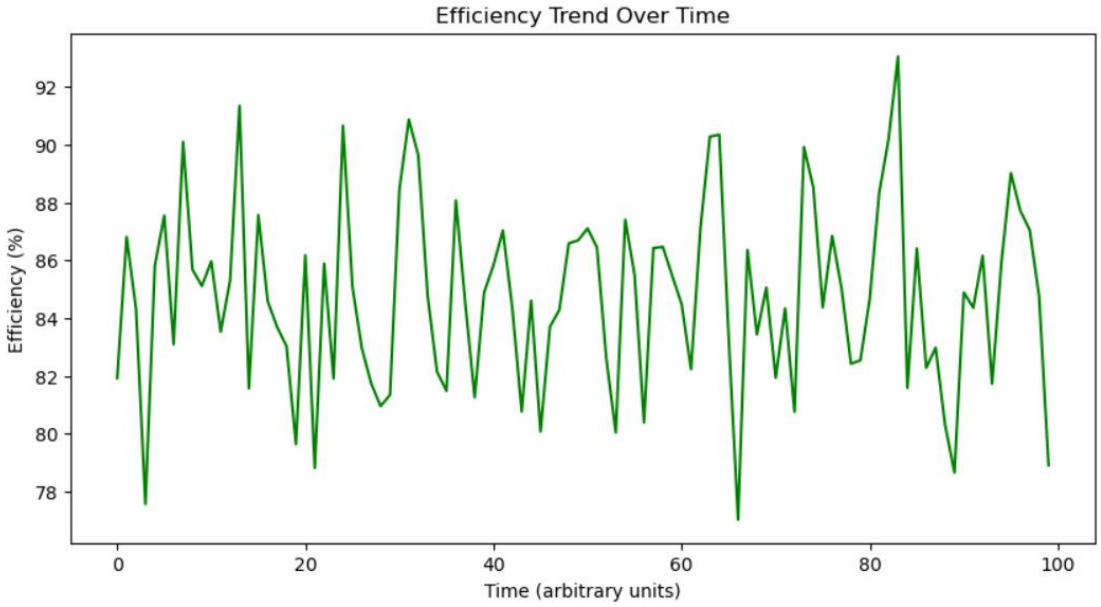
	Machine_ID	Temperature	Vibration	Pressure	Efficiency
0	M1	69.844598	2.804138	29.706520	81.923687
1	M2	75.564488	3.879467	28.981728	86.806765
2	M3	68.339894	2.764770	34.440403	84.319406
3	M4	71.953025	3.173078	32.945258	77.571288
4	M5	72.512830	2.493327	28.429737	85.808393

Statistical Summary:

	Temperature	Vibration	Pressure	Efficiency
count	100.000000	100.000000	100.000000	100.000000
mean	70.163300	3.102082	30.022860	84.728217
std	5.010760	0.533414	1.990343	3.248419
min	58.022585	1.774381	22.625052	77.032286
25%	67.167995	2.794035	28.819608	82.268880
50%	69.944998	3.141189	30.045688	84.762344
75%	73.636499	3.421802	31.232456	86.718462
max	81.893445	4.193522	34.992418	93.046394



```
Abnormal High Temperature Readings (> 78°C):
   Machine_ID  Temperature
6          M2    79.116532
47         M3    79.965213
74         M5    81.893445
94         M5    78.307006
99         M5    78.811996
```



Conclusion:

This experiment demonstrates the use of Python for industrial data analysis in IIoT systems. Through statistical and visual exploration of simulated sensor data, it enables understanding of operational trends, anomaly detection, and process efficiency improvement.