Report on the Exploratory Data Analysis (EDA) Python Code

Introduction

This report summarizes the Python code developed for Exploratory Data Analysis (EDA) on a dataset related to solar radiation, temperature, and other meteorological variables. The code employs various data analysis techniques to assess data quality, visualize trends, and explore relationships between different variables.

Objectives

The primary objectives of the EDA are:

1. **Data Overview**: Understand the structure and contents of the dataset.
2. **Data Cleaning**: Identify and handle missing values, outliers, and anomalies.
3. **Statistical Analysis**: Calculate summary statistics to understand data distributions.
4. **Visual Analysis**: Create visual representations to identify patterns and trends.
5. **Correlation Analysis**: Explore relationships between solar radiation and temperature measures.

1. Data Loading

The code begins by importing ne

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore

"""
This script performs exploratory data analysis on the given dataset.
It prints out the column names, data overview, first few rows of the dataset,
summary statistics, missing values, negative values in GHI, DNI, DHI, and outliers
in the sensor columns.
"""
data = pd.read_csv(r'C:\Users\tesfaneshyisaiase\Documents\front\week-0\data\benin-malanville.csv')

print("Column Names:")
print(data.columns)
print("Data Overview:")
print(data.info())
print("\nFirst few rows of the dataset:")
print(data.head())
```

cessary libraries and loading a CSV dataset using Pandas:

2. Data Overview

Initial checks provide insights into the dataset:

- **Data Structure**: The dimensions and types of columns are printed.
- **First Few Rows**: A preview of the dataset is displayed to understand its contents.

3. Summary Statistics

Summary statistics are calculated to provide an overview of key metrics:

```
summary_statistics = data.describe()
print("\nSummary Statistics:")
print(summary_statistics)
```

This includes mean, median, standard deviation, min, max, and quartiles for each numeric column.

4. Missing Values and Negative Entries

The code checks for missing values and identifies any negative entries in critical columns:

```
missing_values = data.isnull().sum()
print("\nMissing Values:")
print(missing_values[missing_values > 0])
```

5. Outlier Detection and Time Series Analysis

Outliers are detected using the Interquartile Range (IQR) method, specifically for sensor readings and wind speed: and The Timestamp column is converted to a datetime format, and various time series plots are generated to visualize trends in solar radiation and temperature over time:

```python
def detect_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] < lower_bound) | (df[column] > upper_bound)]

sensor_columns = ['ModA', 'ModB', 'WS', 'WSgust']
outliers = {}
for column in sensor_columns:
    outliers[column] = detect_outliers_iqr(data, column)

print("\nOutliers Detected:")
for column, outlier_data in outliers.items():
    if not outlier_data.empty:
        print(f"{column}:\n{outlier_data}\n")


data['Timestamp'] = pd.to_datetime(data['Timestamp'])
# Set the 'Timestamp' column as the index to enable time series analysis
data.set_index('Timestamp', inplace=True)
```

6. Visualizing Relationships

Visualizations such as scatter plots and histograms are used to analyze relationships between various variables:

- **Scatter Plots**: Used to assess the relationship between wind speed and direction.
- **Histograms**: Display the frequency distribution of key variables.

```
plt.figure(figsize=(14, 7))
sns.lineplot(data=data[['GHI', 'DNI', 'DHI', 'Tamb']])
plt.title('Time Series of Solar Radiation and Temperature')
plt.xlabel('Timestamp')
plt.ylabel('Values')
plt.legend(['GHI', 'DNI', 'DHI', 'Tamb'])
plt.show()


cleaned_data = data[data['Cleaning'] == 1]
plt.figure(figsize=(14, 7))
sns.lineplot(data=cleaned_data[['ModA', 'ModB']])
plt.title('Sensor Readings After Cleaning')
plt.xlabel('Timestamp')
plt.ylabel('Sensor Readings')
plt.legend(['ModA', 'ModB'])
plt.show()


correlation_matrix = data[['GHI', 'DNI', 'DHI', 'TModA', 'TModB']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix for Solar Radiation and Temperature')
plt.show()


plt.figure(figsize=(14, 7))
sns.scatterplot(data=data, x='WS', y='WD', hue='WSgust', size='WSgust', sizes=(20, 200))
plt.title('Wind Speed vs Wind Direction')
plt.xlabel('Wind Speed (WS)')
plt.ylabel('Wind Direction (WD)')
plt.show()


plt.figure(figsize=(14, 7))
sns.scatterplot(data=data, x='RH', y='Tamb', hue='GHI')
plt.title('Temperature vs Relative Humidity')
plt.xlabel('Relative Humidity (RH)')
```

Name: Tesfanesh yisaiase