Assignment 2

Title: Household Power Consumption Analysis
➢ Measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years.

Objective:
The objective of this lab assignment is to explore and analyze a dataset containing measurements of electric power consumption in a household over a period of almost 4 years. You will perform various data visualization tasks to gain insights into electrical quantities, sub#metering values, and overall trends.

Code:

1. Load the dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# 1. Load the data
url = "/kaggle/input/ml-household-power-consumption/household_power_consumption.csv"
data = pd.read_csv(url, sep=';', na_values='?')
# Combine Date and Time
data['Datetime'] = pd.to_datetime(data['Date'] + ' ' +
data['Time'], dayfirst=True)
data.drop(columns=['Date', 'Time'], inplace=True)
data.set_index('Datetime', inplace=True)
# Convert numeric columns
cols_to_numeric = ['Global_active_power',
'Global_reactive_power', 'Voltage',
'Global_intensity', 'Sub_metering_1',
'Sub_metering_2', 'Sub_metering_3']
data[cols_to_numeric].apply(pd.to_numeric, errors='coerce')
```

Datetime	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
2006-12-16 17:24:00	4.216	0.418	234.84	18.4	0.0	1.0	17.0
2006-12-16 17:25:00	5.360	0.436	233.63	23.0	0.0	1.0	16.0
2006-12-16 17:26:00	5.374	0.498	233.29	23.0	0.0	2.0	17.0
2006-12-16 17:27:00	5.388	0.502	233.74	23.0	0.0	1.0	17.0
2006-12-16 17:28:00	3.666	0.528	235.68	15.8	0.0	1.0	17.0
...	...	...	...	...	...	...	...
2010-11-26 20:58:00	0.946	0.000	240.43	4.0	0.0	0.0	0.0
2010-11-26 20:59:00	0.944	0.000	240.00	4.0	0.0	0.0	0.0
2010-11-26 21:00:00	0.938	0.000	239.82	3.8	0.0	0.0	0.0
2010-11-26 21:01:00	0.934	0.000	239.70	3.8	0.0	0.0	0.0
2010-11-26 21:02:00	0.932	0.000	239.55	3.8	0.0	0.0	0.0

2075259 rows × 7 columns

Assignment 2

2. Detect outlier, missing value from the data.

```
# 2. Detect Missing Values and Outliers
print("Missing Values:\n", data.isnull().sum())
# Drop rows with missing values
data.dropna(inplace=True)
# Detect outliers in Global_active_power
Q1 = data['Global_active_power'].quantile(0.25)
Q3 = data['Global_active_power'].quantile(0.75)
IQR = Q3 - Q1
outliers = data[(data['Global_active_power'] < (Q1 - 1.5 *
IQR)) |
(data['Global_active_power'] > (Q3 + 1.5 *
IQR))]
print("Outliers detected:", len(outliers))
```

```
Missing Values:
 Global_active_power      25979
Global_reactive_power    25979
Voltage                  25979
Global_intensity         25979
Sub_metering_1           25979
Sub_metering_2           25979
Sub_metering_3           25979
dtype: int64
Outliers detected: 94907
```

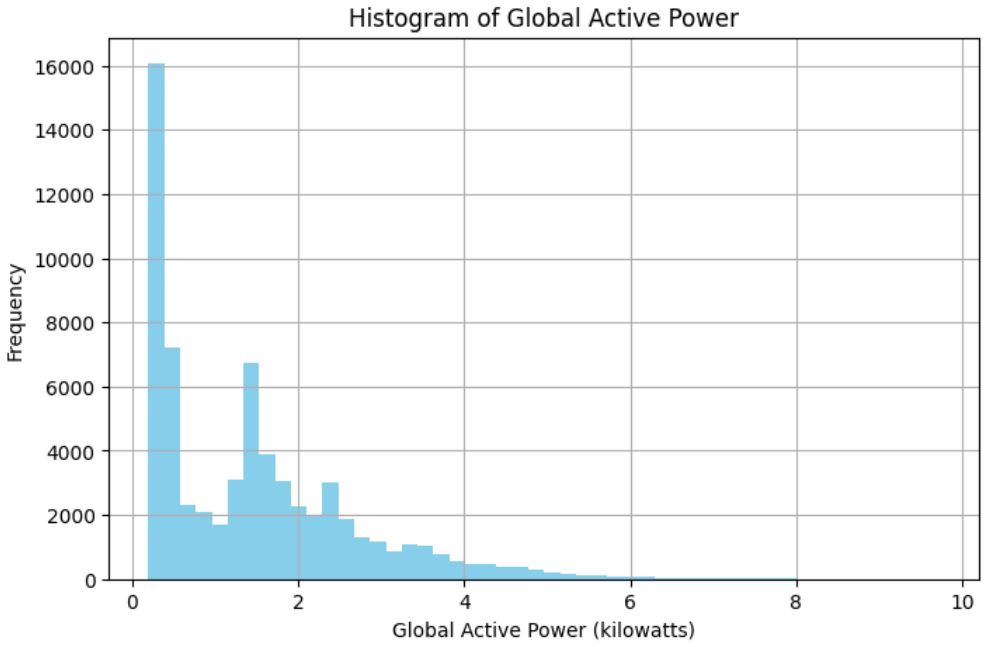
3. Subset the data from the given dates (December 2006 and November 2009)

```
# 3. Subset Data: December 2006 and November 2009
dec_2006 = data.loc['2006-12']
nov_2009 = data.loc['2009-11']
subset_data = pd.concat([dec_2006, nov_2009])
```

4. Create a histogram

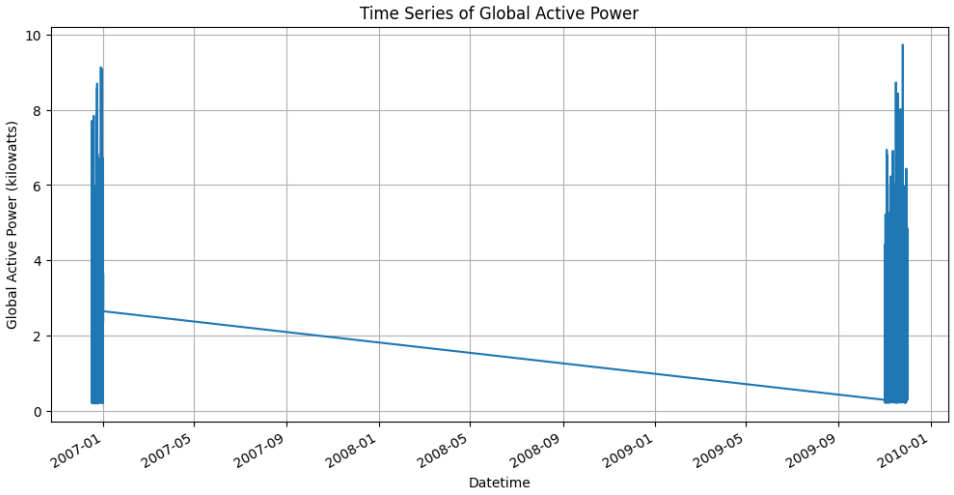
```
# 4. Histogram
plt.figure(figsize=(8,5))
subset_data['Global_active_power'].hist(bins=50,
color='skyblue')
plt.title('Histogram of Global Active Power')
```

```
plt.xlabel('Global Active Power (kilowatts)')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



5. Create a Time series

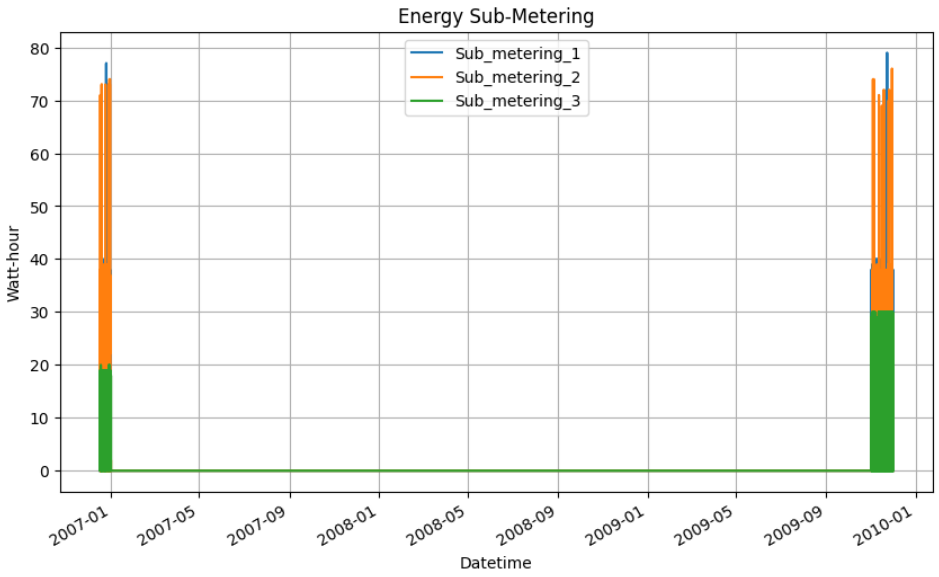
```
# 5. Time Series Plot
plt.figure(figsize=(12,6))
subset_data['Global_active_power'].plot()
plt.title('Time Series of Global Active Power')
plt.ylabel('Global Active Power (kilowatts)')
plt.xlabel('Datetime')
plt.grid(True)
plt.show()
```



Assignment 2

6. Create a plot for sub metering

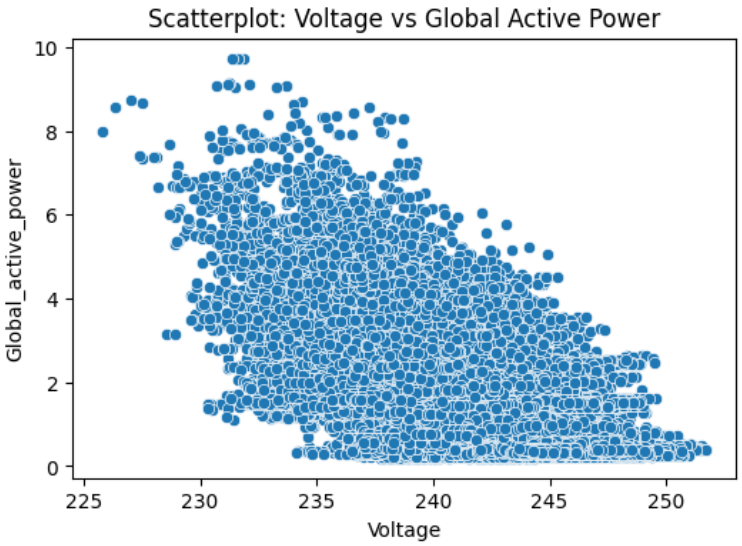
```
# 6. Sub-Metering Plot
plt.figure(figsize=(10,6))
subset_data['Sub_metering_1'].plot(label='Sub_metering_1')
subset_data['Sub_metering_2'].plot(label='Sub_metering_2')
subset_data['Sub_metering_3'].plot(label='Sub_metering_3')
plt.legend()
plt.title('Energy Sub-Metering')
plt.ylabel('Watt-hour')
plt.grid(True)
plt.show()
```



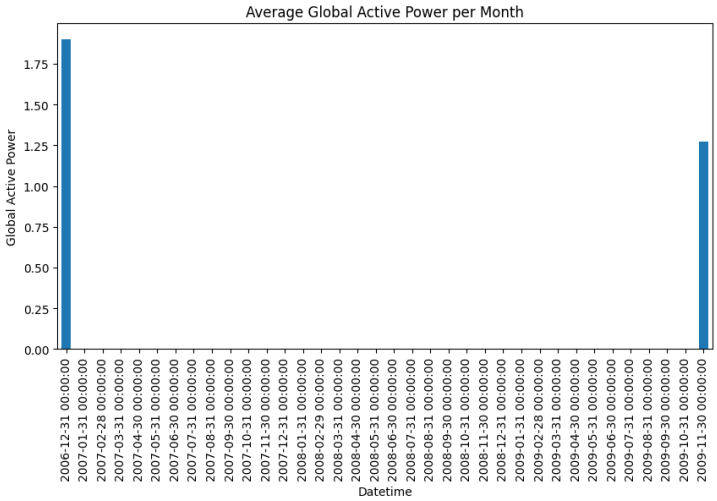
Assignment 2

7. Create multiple plots, such as, Scatterplot, Histogram, Bar Chart, Pie Chart, Count plot, Boxplot, Heatmap, Distplot, Jointplot

```
# Scatterplot
plt.figure(figsize=(6,4))
sns.scatterplot(x='Voltage', y='Global_active_power',
data=subset_data)
plt.title("Scatterplot: Voltage vs Global Active Power")
plt.show()
```

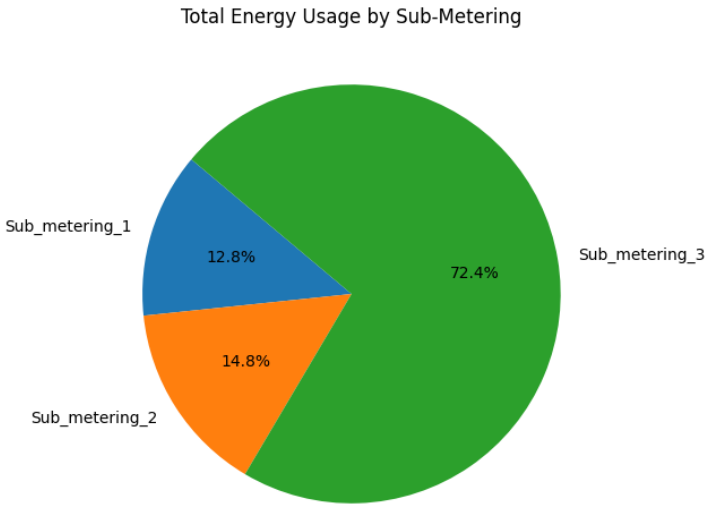



```
# Bar Chart (Average Power per Month)
monthly_avg = subset_data.resample('ME').mean()
monthly_avg['Global_active_power'].plot(kind='bar',
figsize=(10,5))
plt.title("Average Global Active Power per Month")
plt.ylabel("Global Active Power")
plt.show()
```

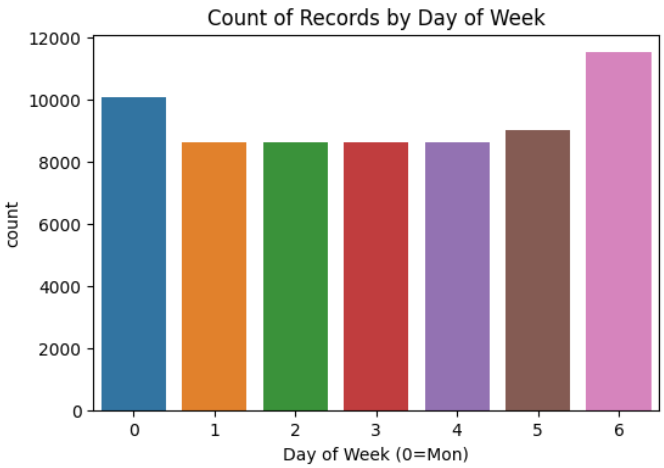


Assignment 2

```
# Pie Chart (Total Energy by Sub Metering)
sub_totals = subset_data[['Sub_metering_1',
'Sub_metering_2', 'Sub_metering_3']].sum()
plt.figure(figsize=(6,6))
plt.pie(sub_totals, labels=sub_totals.index,
autopct='%1.1f%%', startangle=140)
plt.title("Total Energy Usage by Sub-Metering")
plt.show()
```

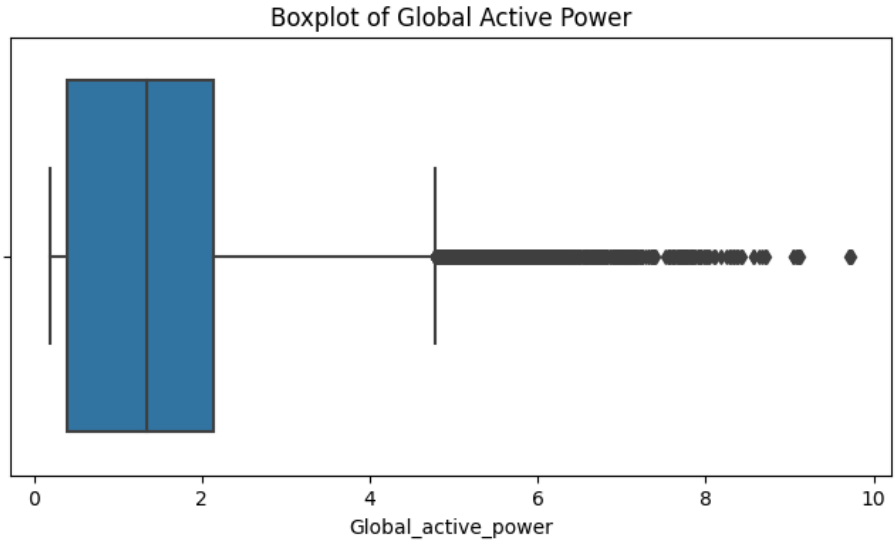



```
# Countplot (Days of Week)
plt.figure(figsize=(6,4))
sns.countplot(x=subset_data.index.dayofweek)
plt.title("Count of Records by Day of Week")
plt.xlabel("Day of Week (0=Mon)")
plt.show()
```

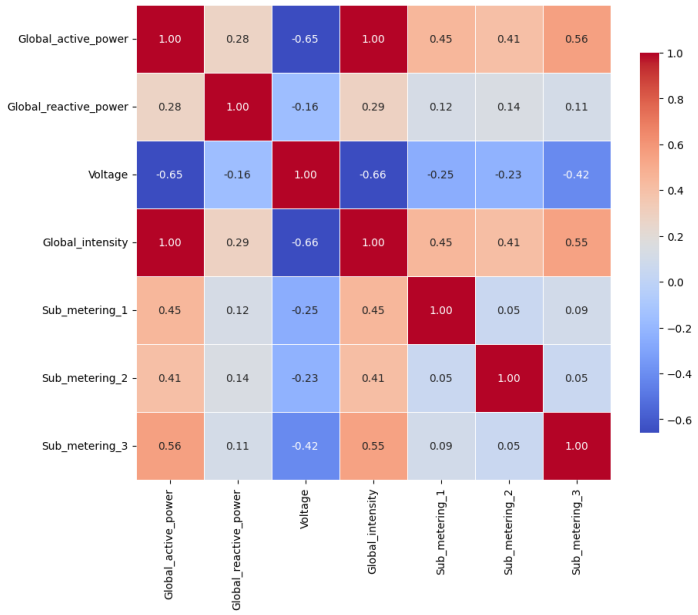


Assignment 2

```
# Boxplot
plt.figure(figsize=(8,4))
sns.boxplot(x=subset_data['Global_active_power'])
plt.title("Boxplot of Global Active Power")
plt.show()
```

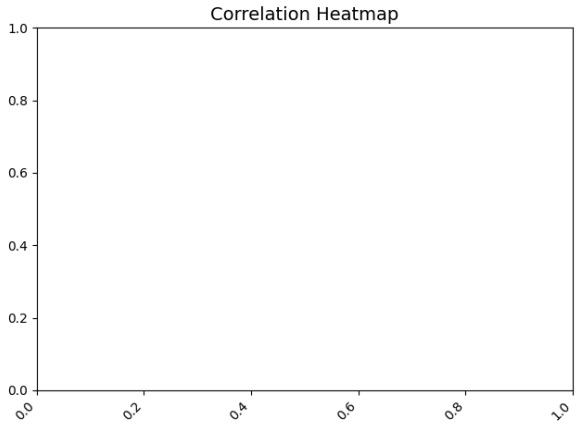



```
# Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(subset_data.corr(), annot=True,
cmap='coolwarm', fmt=".2f", linewidths=0.5, square=True,
cbar_kws={"shrink": 0.8}, annot_kws={"size": 10})
```

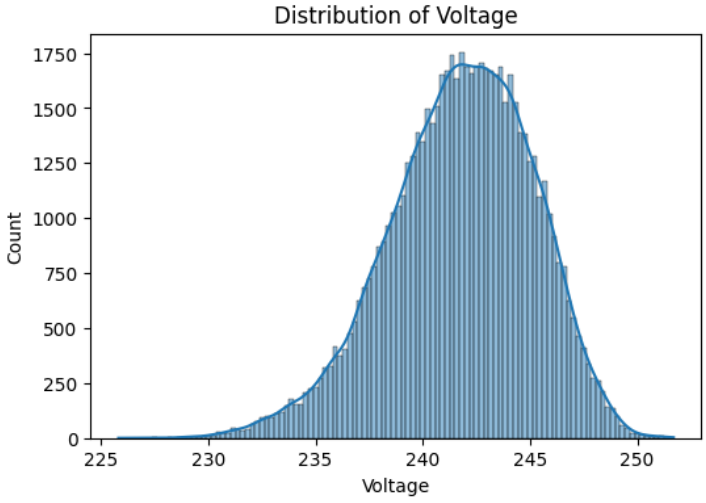


Assignment 2

```
# Correlation Heatmap
plt.title("Correlation Heatmap", fontsize=14)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(rotation=0, fontsize=10)
plt.tight_layout()
plt.show()
```



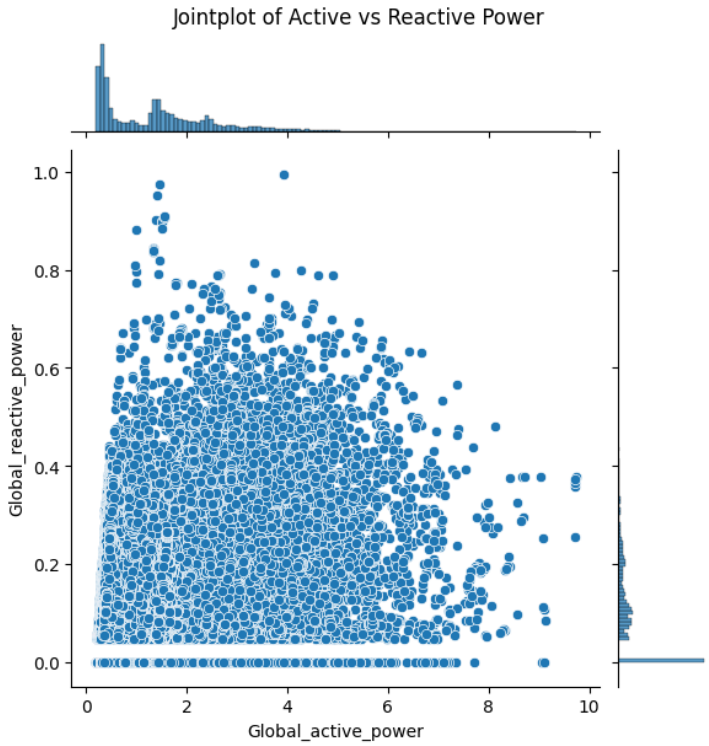

```
# Distplot (using histplot with KDE)
plt.figure(figsize=(6,4))
sns.histplot(subset_data['Voltage'], kde=True)
plt.title("Distribution of Voltage")
plt.show()
```




```
# Jointplot
sns.jointplot(x='Global_active_power',
y='Global_reactive_power', data=subset_data,
```

Assignment 2

```
kind='scatter')
plt.suptitle("Jointplot of Active vs Reactive Power",
y=1.02)
plt.show()
```



Assignment 2

- **Which visualization techniques shows optimum visualization.**

The visualization techniques that show optimum visualization—based on clarity, insight, and usefulness for the given dataset—are:

Time Series Plot

Why: It effectively shows how power consumption changes over time, highlighting trends, peaks, and fluctuations.

Sub-Metering Plot (Line Plot)

Why: It clearly compares energy usage across different appliances or zones over time, making it easy to identify consumption patterns.

Correlation Heatmap

Why: It quickly reveals relationships (positive or negative) between variables, helping identify which variables influence each other.

These three techniques are considered most optimum for visualizing the dataset because they provide the most actionable insights in an interpretable manner.