Lab-Assignment - 1

Measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years. Different electrical quantities and some sub-metering values are available. Dataset: https://d396qusza40orc.cloudfront.net/exdata%2Fdata%2Fhousehold_power_consumption.zip (https://d396qusza40orc.cloudfront.net/exdata%2Fdata%2Fhousehold_power_consumption.zip) Perform the following:

Exercise 1:

1. Load the data
2. Read first 5 rows to get headers
3. Read 2900 rows that contain information on 2007-02-01 and 2007-02-02
4. Converting Date and Time variables to Date/Time format

Exercise 2: Subset the loaded data for 2007-02-01 and 2007-02-02

Exercise 3:

1. Histogram of global active power consumption
2. Global active consumption over time
3. Energy sub metering

Exercise 1:

1. Load the data: First, download the dataset from the provided link and extract it. We'll find a file named "household_power_consumption.txt."

In [1]:
```
import pandas as pd

# Load the data into a DataFrame
data = pd.read_csv(r"C:\Users\raval\jupyter_notebook\pdeu_data_science\machine_learning_lab
```

2. Read first 5 rows to get headers

In [2]:
```
# Display the first few rows of the DataFrame
data.head()
```

Out[2]:

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1
0	16/12/2006	17:24:00	4.216	0.418	234.84	18.4	0.0
1	16/12/2006	17:25:00	5.360	0.436	233.63	23.0	0.0
2	16/12/2006	17:26:00	5.374	0.498	233.29	23.0	0.0
3	16/12/2006	17:27:00	5.388	0.502	233.74	23.0	0.0
4	16/12/2006	17:28:00	3.666	0.528	235.68	15.8	0.0

3. Read 2900 rows for 2007-02-01 and 2007-02-02
4. Converting Date and Time variables to Date/Time format

In [3]:
```python
# Read 2900 rows for 2007-02-01 and 2007-02-02

# Convert the 'Date' column to datetime
data['Date'] = pd.to_datetime(data['Date'])
data.head()
```

```
C:\Users\raval\AppData\Local\Temp\ipykernel_9732\1974775207.py:4: UserWarning: Parsing dat
es in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to
inconsistently parsed dates! Specify a format to ensure consistent parsing.
  data['Date'] = pd.to_datetime(data['Date'])
```

Out[3]:

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_
0	2006-12-16	17:24:00	4.216	0.418	234.84	18.4	0.0	
1	2006-12-16	17:25:00	5.360	0.436	233.63	23.0	0.0	
2	2006-12-16	17:26:00	5.374	0.498	233.29	23.0	0.0	
3	2006-12-16	17:27:00	5.388	0.502	233.74	23.0	0.0	
4	2006-12-16	17:28:00	3.666	0.528	235.68	15.8	0.0	

In [4]:
```python
data1=data[(data["Date"]=="2007-02-01") | (data["Date"]=="2007-02-02")]
data1
```

Out[4]:

	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_
23436	2007-02-01	00:00:00	0.442	0.122	241.06	1.8	0.
23437	2007-02-01	00:01:00	0.370	0.000	241.22	1.6	0.
23438	2007-02-01	00:02:00	0.368	0.000	241.03	1.6	0.
23439	2007-02-01	00:03:00	0.370	0.000	241.41	1.6	0.
23440	2007-02-01	00:04:00	0.370	0.000	241.22	1.6	0.
...	...	...	...	...	...	...	.
69511	2007-02-02	23:55:00	3.696	0.226	240.90	15.2	0.

Exercise 2:

Subset the loaded data for 2007-02-01 and 2007-02-02

In [5]:
```
# Subset the data based on the given dates
subset_data = data[(data["Date"]=="2007-02-01") | (data["Date"]=="2007-02-02")]

# Display the subsetted data
subset_data.head()
```

Out[5]:

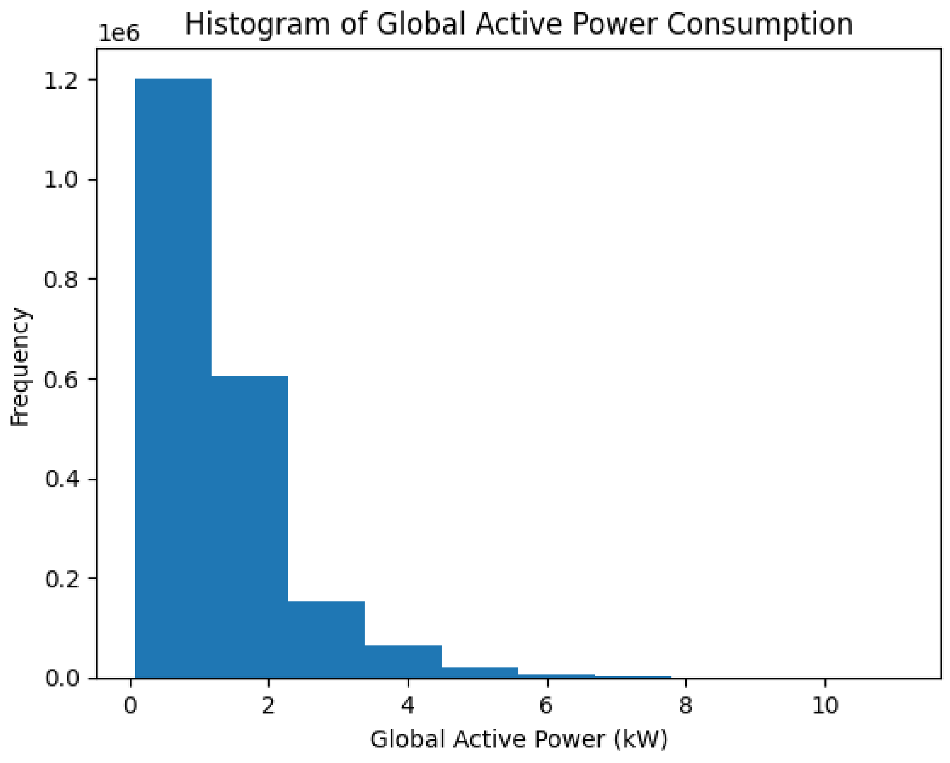
	Date	Time	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	S
23436	2007-02-01	00:00:00	0.442	0.122	241.06	1.8	0.0	
23437	2007-02-01	00:01:00	0.370	0.000	241.22	1.6	0.0	
23438	2007-02-01	00:02:00	0.368	0.000	241.03	1.6	0.0	
23439	2007-02-01	00:03:00	0.370	0.000	241.41	1.6	0.0	
23440	2007-02-01	00:04:00	0.370	0.000	241.22	1.6	0.0	

Exercise 3:

1. Histogram of global active power consumption

Create a histogram: To create a histogram of the electric power consumption, we can plot the "Global_active_power" column using matplotlib or any other plotting library:

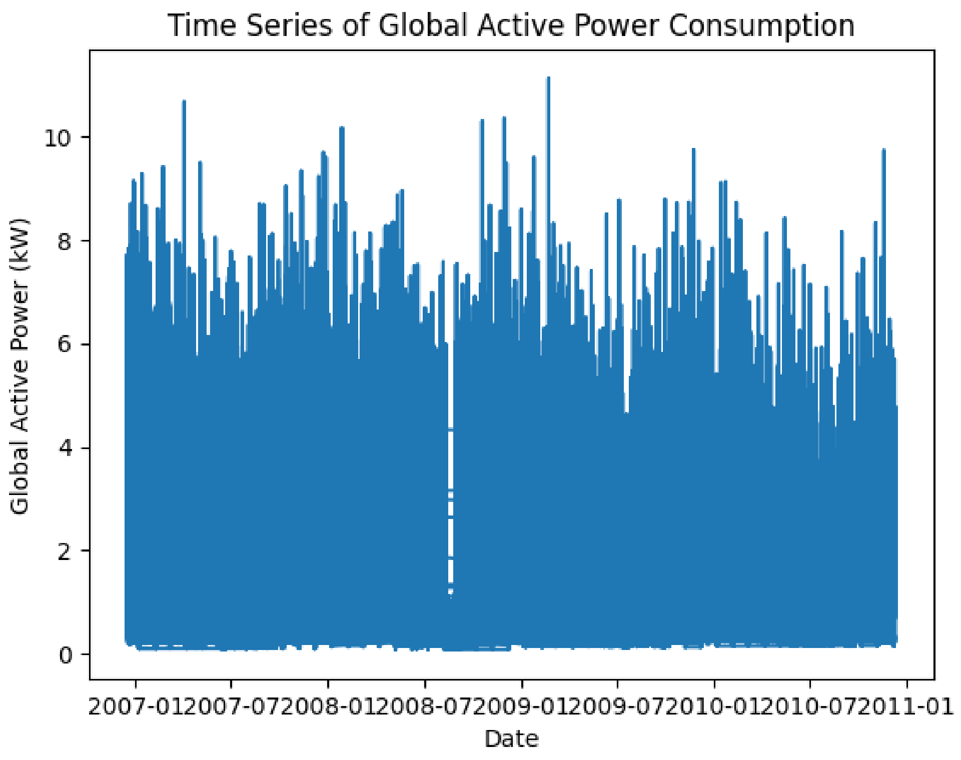
In [6]:
```
import matplotlib.pyplot as plt
# Plot a histogram of global active power consumption
plt.hist(data['Global_active_power'])
plt.xlabel('Global Active Power (kW)')
plt.ylabel('Frequency')
plt.title('Histogram of Global Active Power Consumption')
plt.show()
```



2. Global active consumption over time

Create a time series: To create a time series plot of the electric power consumption over time, we can plot the "Global_active_power" column against the "Date" column:

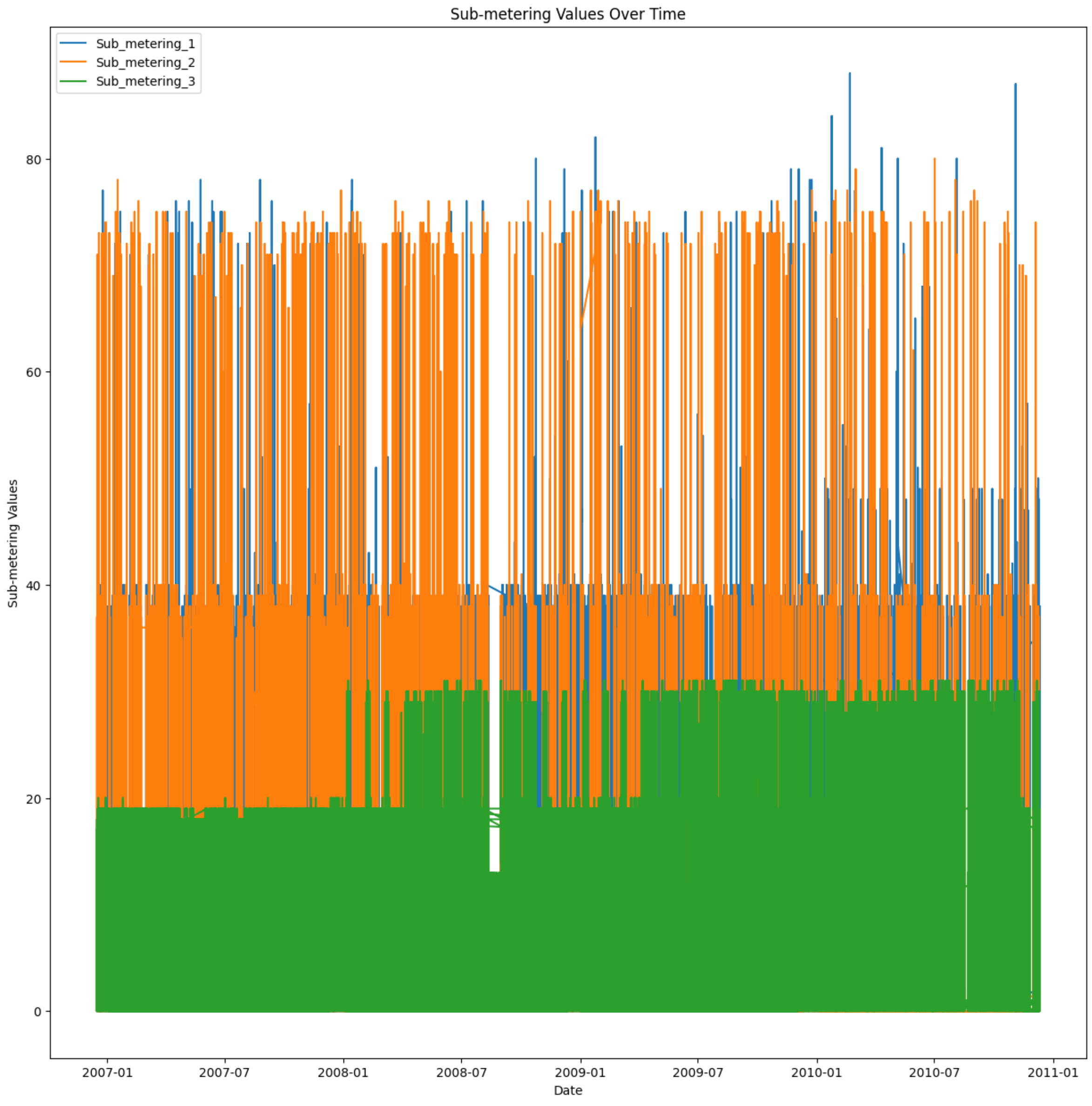
In [7]:
```
# Create a time series plot of global active power consumption
plt.plot(data['Date'], data['Global_active_power'])
plt.xlabel('Date')
plt.ylabel('Global Active Power (kW)')
plt.title('Time Series of Global Active Power Consumption')
plt.rcParams['figure.figsize'] = [15,15]
plt.show()
```



3. Energy sub metering

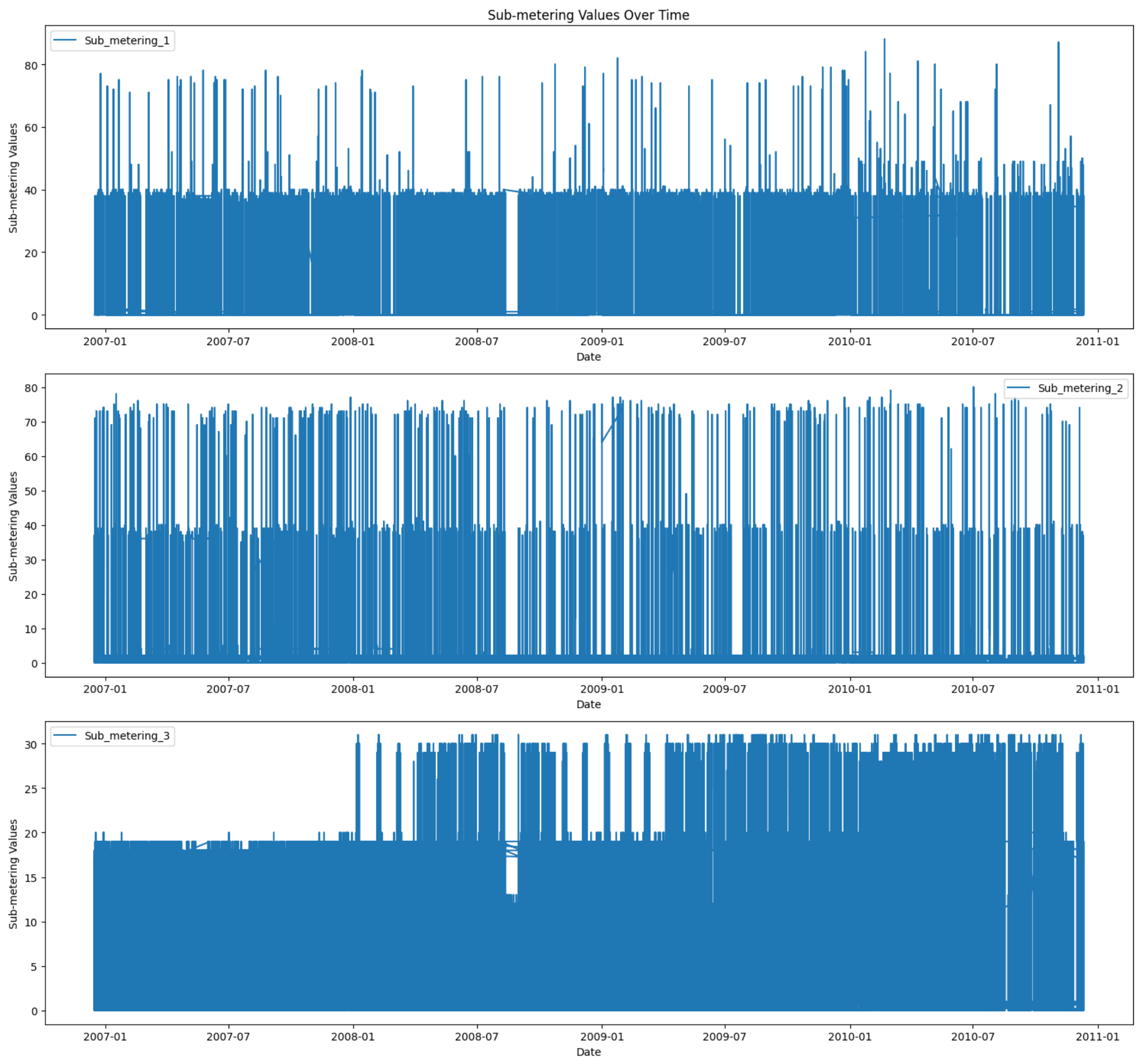
Create a plot for sub-metering: To create a plot for sub-metering values, we can plot the relevant columns from the dataset.

```
In [8]: # Create a plot for sub-metering
        plt.plot(data['Date'], data['Sub_metering_1'], label='Sub_metering_1')
        plt.plot(data['Date'], data['Sub_metering_2'], label='Sub_metering_2')
        plt.plot(data['Date'], data['Sub_metering_3'], label='Sub_metering_3')
        plt.xlabel('Date')
        plt.ylabel('Sub-metering Values')
        plt.title('Sub-metering Values Over Time')
        plt.legend()
        plt.rcParams['figure.figsize'] = [15, 14]
        plt.show()
```



In [9]:
```python
# 2nd way of subploting

plt.subplot(3, 1, 1)
plt.plot(data['Date'], data['Sub_metering_1'], label='Sub_metering_1')
plt.title('Sub-metering Values Over Time')
plt.xlabel('Date')
plt.ylabel('Sub-metering Values')
plt.legend()

plt.subplot(3, 1, 2)
plt.plot(data['Date'], data['Sub_metering_2'], label='Sub_metering_2')
plt.xlabel('Date')
plt.ylabel('Sub-metering Values')
plt.legend()

plt.subplot(3, 1, 3)
plt.plot(data['Date'], data['Sub_metering_3'], label='Sub_metering_3')
plt.xlabel('Date')
plt.ylabel('Sub-metering Values')
plt.legend()

# plt.rcParams['figure.figsize'] = [12, 11]
plt.tight_layout()
plt.show()
```

```
C:\Users\raval\AppData\Local\Temp\ipykernel_9732\237813748.py:23: UserWarning: Creating le
gend with loc="best" can be slow with large amounts of data.
  plt.tight_layout()
C:\python311\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Creating legen
d with loc="best" can be slow with large amounts of data.
  fig.canvas.print_figure(bytes_io, **kw)
```


Sub-metering Values Over Time
Sub_metering_1
Sub_metering_2
Sub_metering_3
Sub-metering Values
Date