

Lab 03 visualize

Bảng phân công

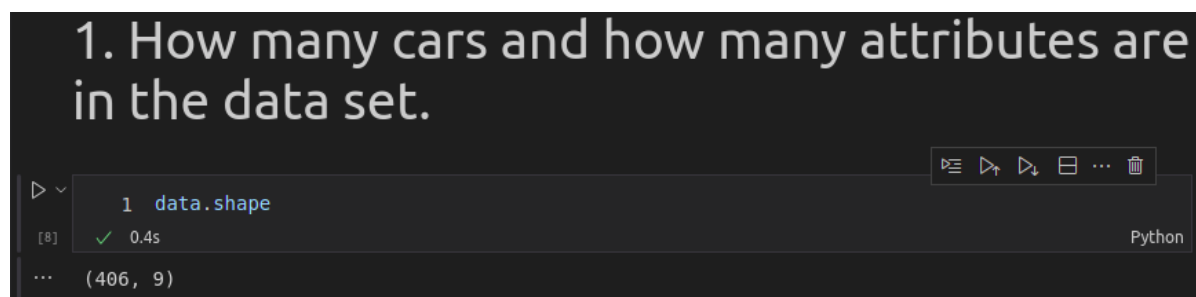
MSSV	Tên	Mức độ hoàn thành	Công việc	Đóng góp
19120212	Vũ Công Duy	100%	Section 3 : code 1->9, nhận xét 1->5	33%
19120202	Võ Tiến Dũng	100%	Section 4 : code và debug giúp Minh, viết báo cáo nhận xét	33%
191096	Lưu Gia Minh	100%	Section 3 : nhận xét 5->9 dựa vào biểu đồ Section 4 : code	33%

Các thư viện trực quan hoá dùng trong bài lab

- `matplotlib` : thư viện phổ biến trong visualize. Tương thích với các thư viện `numpy`, `sklearn`, `pandas`
- `seaborn` : kế thừa từ `matplotlib`, vẽ những biểu đồ phức tạp tốt hơn và có tính thẩm mỹ hơn `matplotlib`.

Section 3 : Exploratory analysis of Car MPG data

1. How many cars and how many attributes are in the data set.



```
1 data.shape
[8] ✓ 0.4s
... (406, 9)
```

Có tổng cộng 406 chiếc xe được khảo sát và 9 attributes trong data set này.

2.

How many distinct car companies are represented in the data set?

Trả lời : 312

```
1 #distinct car
2 len(data['car_name'].unique())

[11] ✓ 0.4s

... 312
```

What is the name of the car with the best MPG?

Trả lời : mazda glc

```
1 # best MPG
2 data.sort_values(by='mpg', ascending=False).head(1).car_name

[22] ✓ 0.6s

... 329 mazda glc
Name: car_name, dtype: object
```

What car company produced the most 8-cylinder cars?

Trả lời : ford

```
1 data['company'] = data['car_name'].str.split(' ').str.get(0)
2 data[data['cylinders']==8][['company', 'car_name']].drop_duplicates().groupby(
    'company').count()

[54] ✓ 0.1s
```

company	car_name
amc	7
buick	6
cadillac	2
chevrolet	12
chevy	2
chrysler	4
dodge	12
ford	15
hi	1
mercury	5
oldsmobile	6
plymouth	9
pontiac	5

What are the names of 3-cylinder cars?

Trả lời : danh sách tên xe ở dưới hình

```
1 ## 3-cylinder
2 data[data['cylinders']==3.0]['car_name'].drop_duplicates()

[58] ✓ 0.8s

... 78      mazda rx2 coupe
    118      maxda rx3
    250      mazda rx-4
    341      mazda rx-7 gs
    Name: car_name, dtype: object
```

Do some internet search that can tell you about the history and popularity of those 3-cylinder cars.

Ô tô sử dụng động cơ xăng 3 xi-lanh tại thị trường Việt Nam được coi là kinh tế. Mức độ bảo trì thấp và cung cấp hiệu suất tốt cũng như những chiếc xe sử dụng động cơ 4 xi-lanh. Công nghệ phát triển và sản xuất động cơ sử dụng 3 xi-lanh không có nghĩa là sức mạnh sẽ suy giảm theo bất kỳ cách nào. Bởi vì dung tích xi lanh kết hợp cũng có thể làm gần với động cơ 4 xi-lanh

Source: <https://www.xeoto.com.vn/news/dong-co-3-xy-lanh-nho-tiet-kiem-co-nen-su-dung-hay-khong-a-110>

3. What is the range, mean, and standard deviation of each attribute? Pay attention to potential missing values

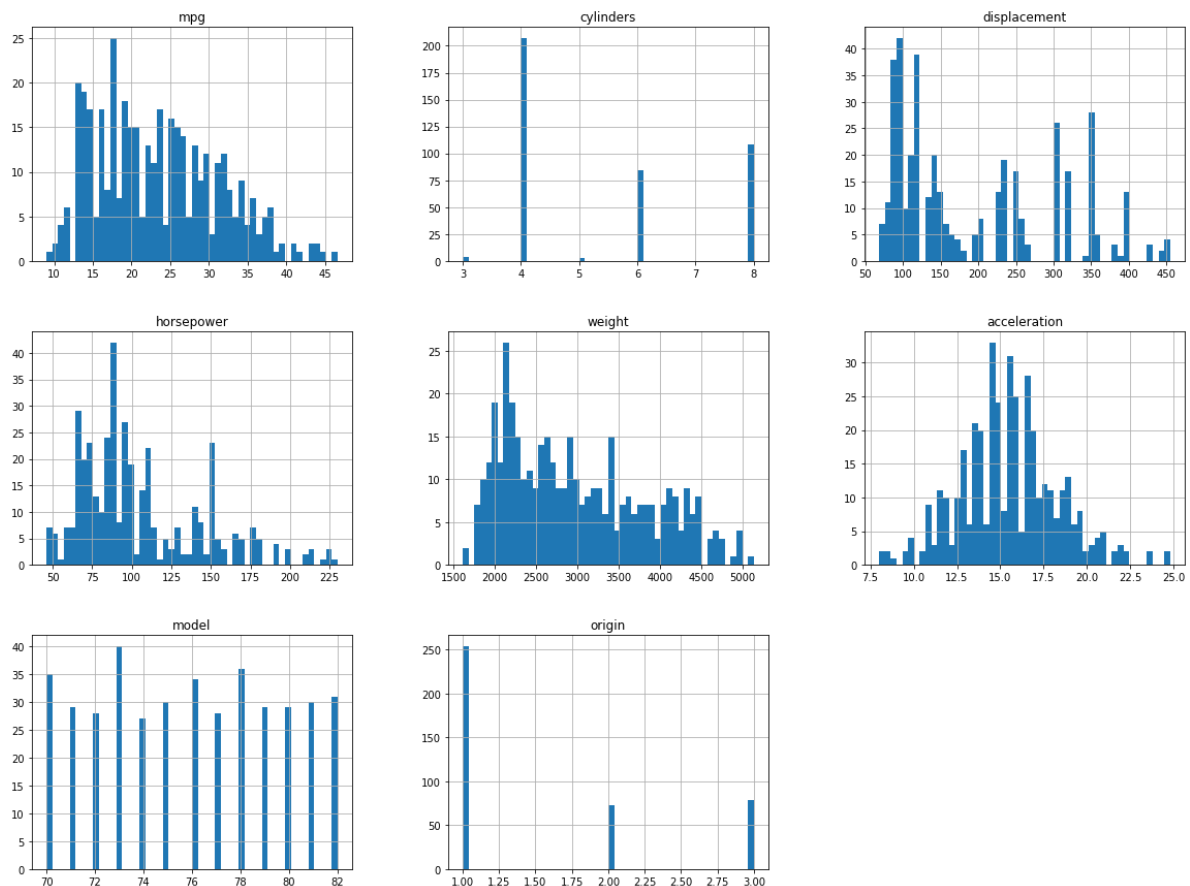
Kết quả như hình dưới

```
1 #agg with all columns
2 data.select_dtypes(include=['int','float']).agg(['count','mean','max','min','std'],
3 missing='drop')

[17] ✓ 0.8s Python
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model	origin
count	398.000000	406.000000	406.000000	400.000000	406.000000	406.000000	406.000000	406.000000
mean	23.514573	5.475369	194.779557	105.082500	2979.413793	15.519704	75.921182	1.568966
max	46.600000	8.000000	455.000000	230.000000	5140.000000	24.800000	82.000000	3.000000
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.000000	1.000000
std	7.815984	1.712160	104.922458	38.768779	847.004328	2.803359	3.748737	0.797479

4. Plot histograms for each attribute. Pay attention to the appropriate choice of number of bins. Write 2-3 sentences summarizing some interesting aspects of the data by looking at the histograms.



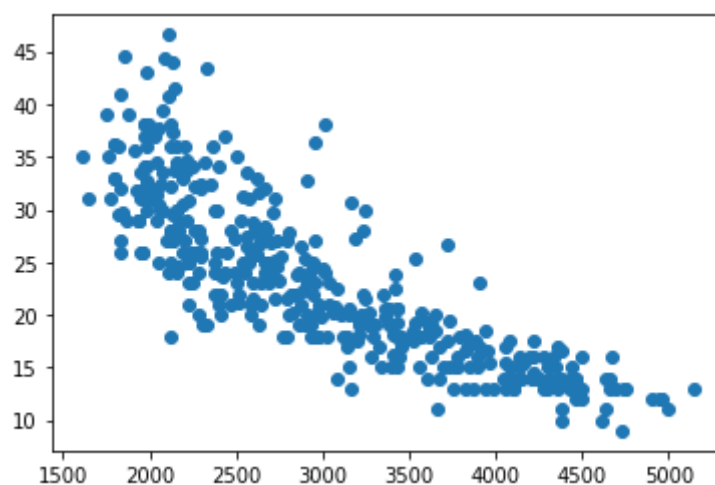
Nhận xét (chỉ mang tính giả thuyết, ý tưởng)

`cylinders`: xi lanh của xe thường là số chẵn (4-6-8).

`weight`, `horsepower`, `mpg`: lệch trái, có xu hướng giảm dần về bên phải (long-tail)

`acceleration`: phân phối đều, mean trong khoảng 15-17. Chứng tỏ nhu cầu về tốc độ xe không bị bias quá nhiều.

5. Plot a scatterplot of weight vs. MPG attributes. What do you conclude about the relationship between the attributes? What is the correlation coefficient between the 2 attributes?



Hai thuộc tính này nghịch biến, có mối quan hệ là một hàm phi tuyến lồi với variance giảm dần khi `weight` tăng và `mpg` giảm.

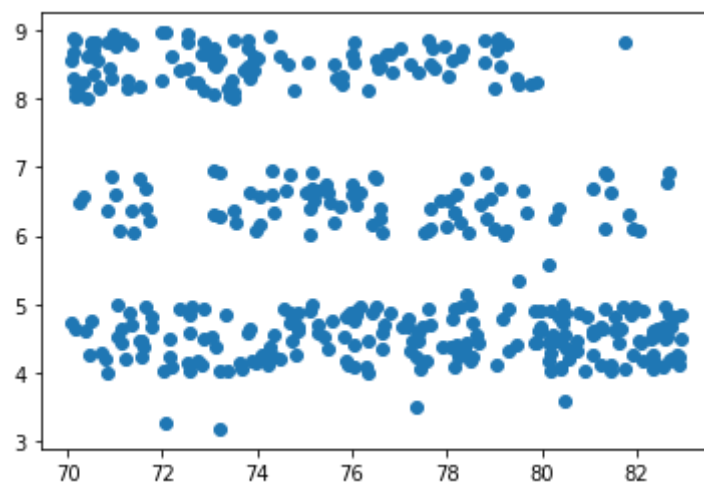
```
1 #correlation between weight and mpg
2 data.corr()['mpg']['weight']

[35] ✓ 0.6s

... -0.8317409332443344
```

Như đã đề cập ở trên, hai thuộc tính có mối quan hệ nghịch biến nên coef có giá trị tiến gần tới -1.

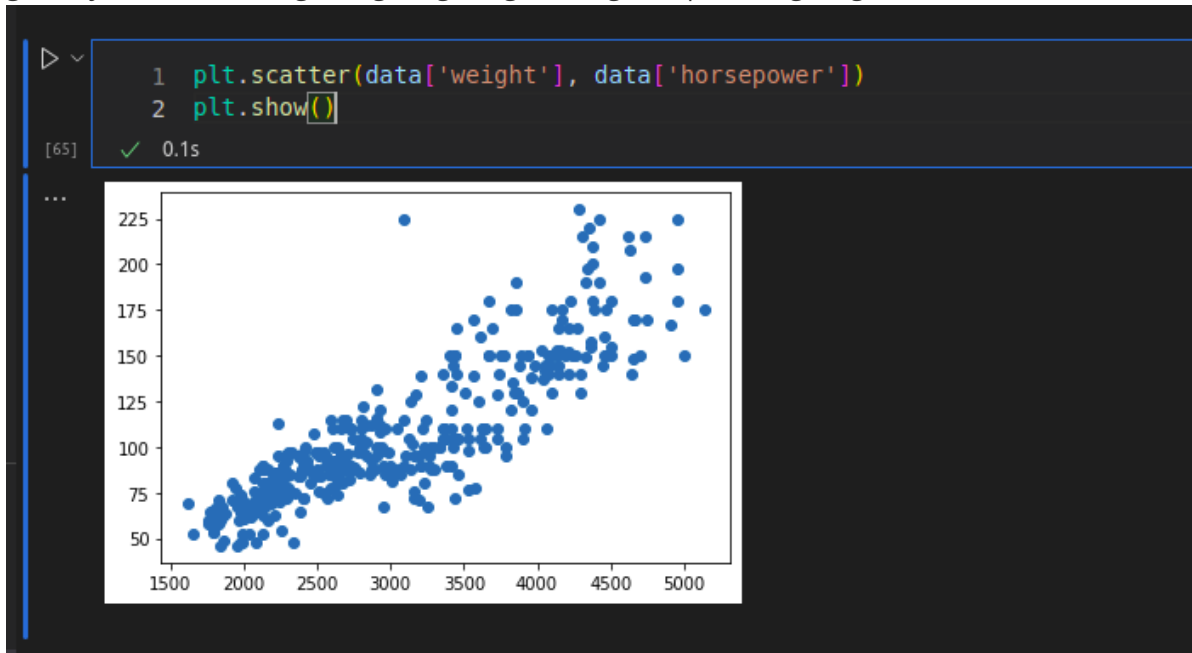
6. Plot a scatterplot of year vs. cylinders attributes. Add a small random noise to the values to make the scatterplot look nicer. What can you conclude? Do some internet search about the history of car industry during 70's that might explain the results



- Năm 72 chứng kiến sự tăng lên đáng kể về số lượng xe có xi lanh ở tầm trung (4 xi lanh). Tuy nhiên, sự phân bố lại quay trở về như những năm 70,71.
- Kể từ năm 80 đổ đi, số lượng xe có xi lanh ở mức cao (6 xi lanh) giảm mạnh, đồng thời số lượng xe có 2 xi lanh lại tăng mạnh.

7. Show 2 more scatterplots that are interesting do you. Discuss what you see.

Bạn em tiến hành trực quan giữa hai thuộc tính là `weight` và `horsepower`. Bạn em đưa ra một giả thuyết là : xe có trọng lượng càng nặng thì công suất phải càng tăng.



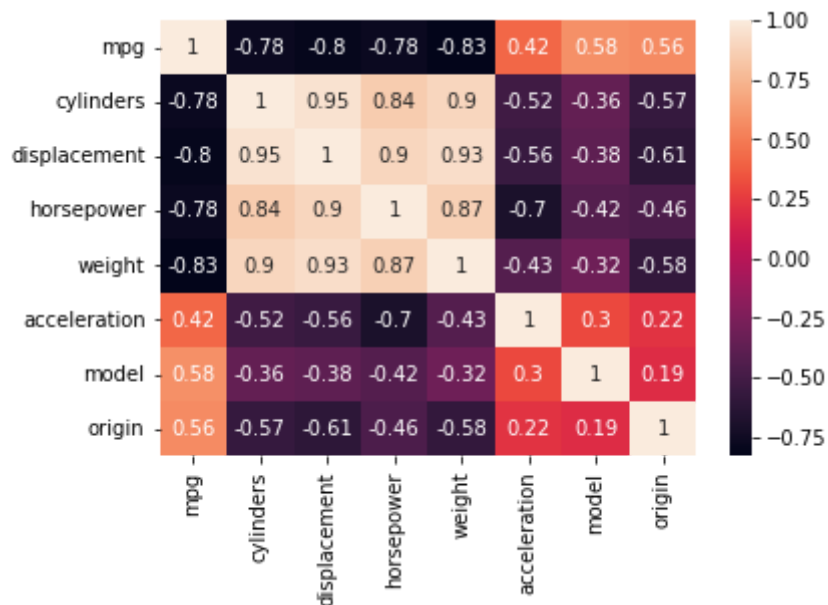
Đúng như những gì bạn em đã dự đoán. Tuy nhiên, bạn em thấy một hiện tượng là khi horsepower, weight càng tăng thì variance của đường hồi quy cũng sẽ càng tăng.

8. Plot a time series for all the companies that show how many new cars they introduces during each year. Do you see some interesting trends?



Em không thấy trend thú vị nào cả.(Đã cố gắng hết sức)

Calculate the pairwise correlation, and draw the heatmap with Matplotlib. Do you see some interesting correlation?



Nhận xét :

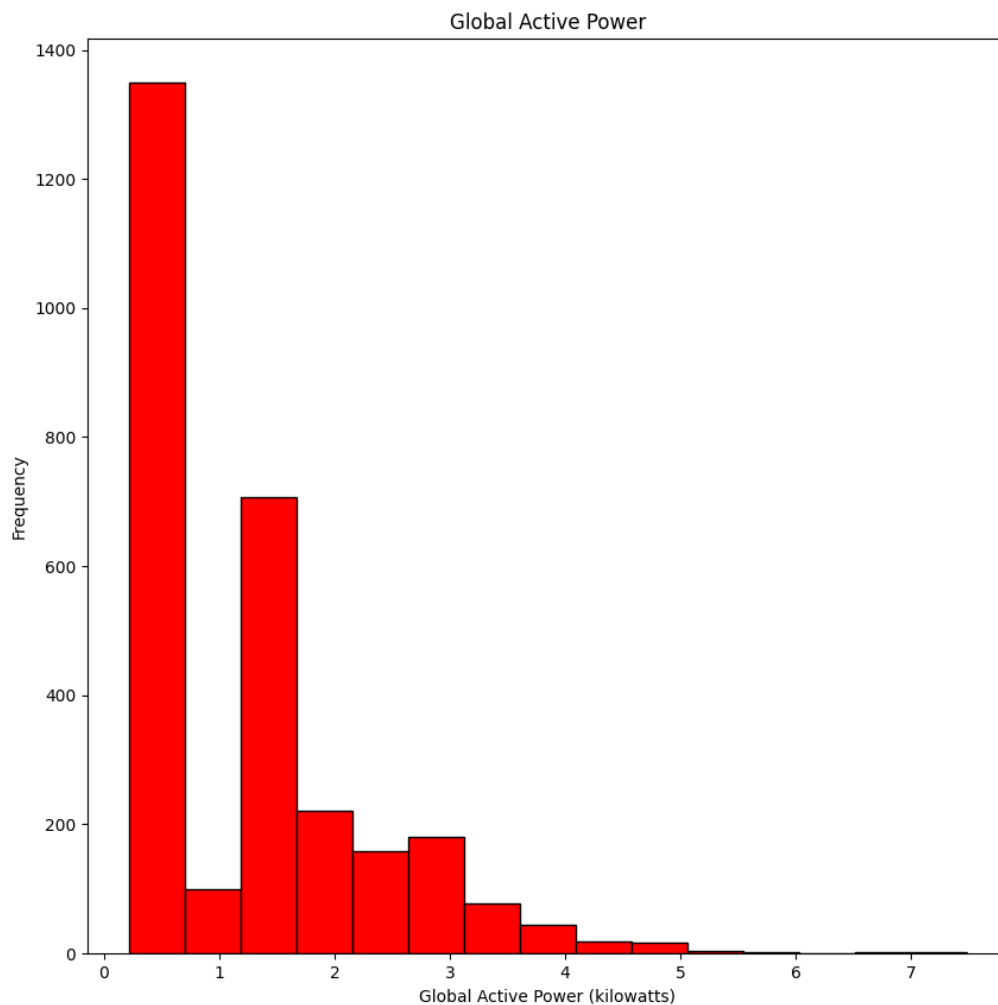
Các thuộc tính `cylinders`, `displacement`, `horsepower` :

- rất tương quan với nhau
- rất nghịch biến so với `mpg`
- tương quan nhẹ với các biến `acceleration`, `model`, `origin`.

Các thuộc tính `acceleration`, `model`, `origin` :

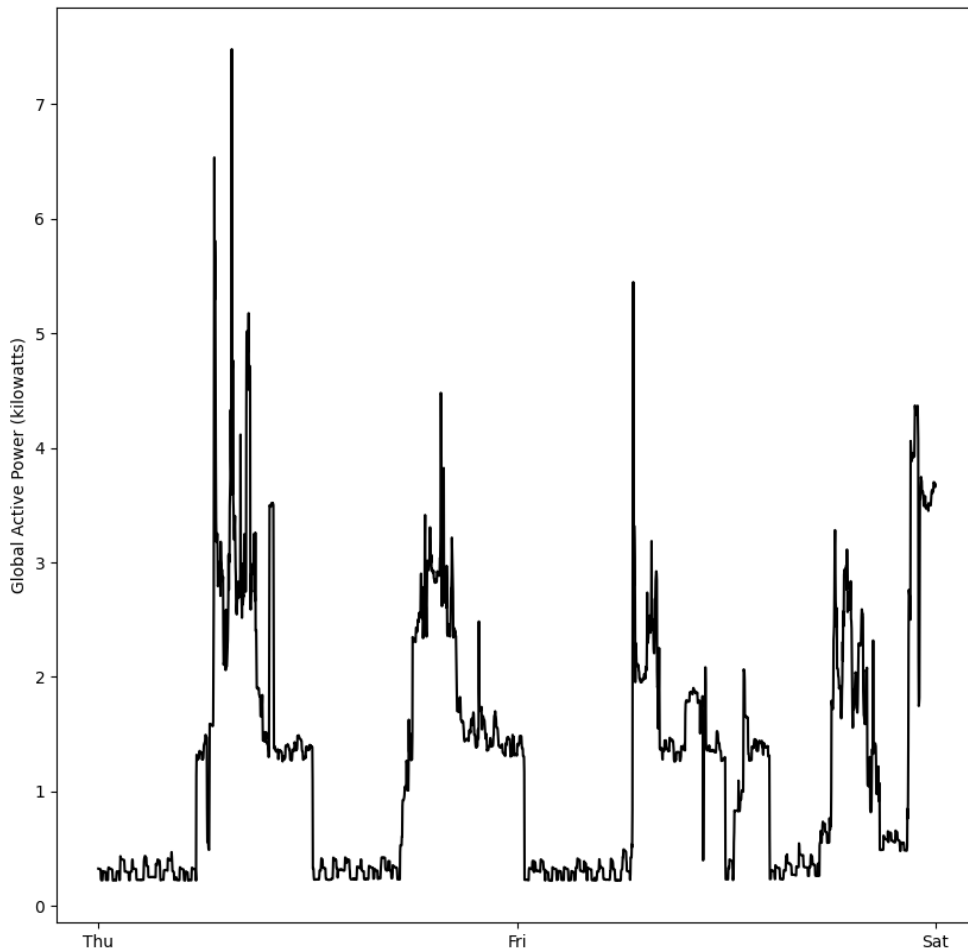
- tương quan nghịch biến ở mức độ vừa với nhau.
- tương quan đồng biến vừa phải với `mpg`

Section 4



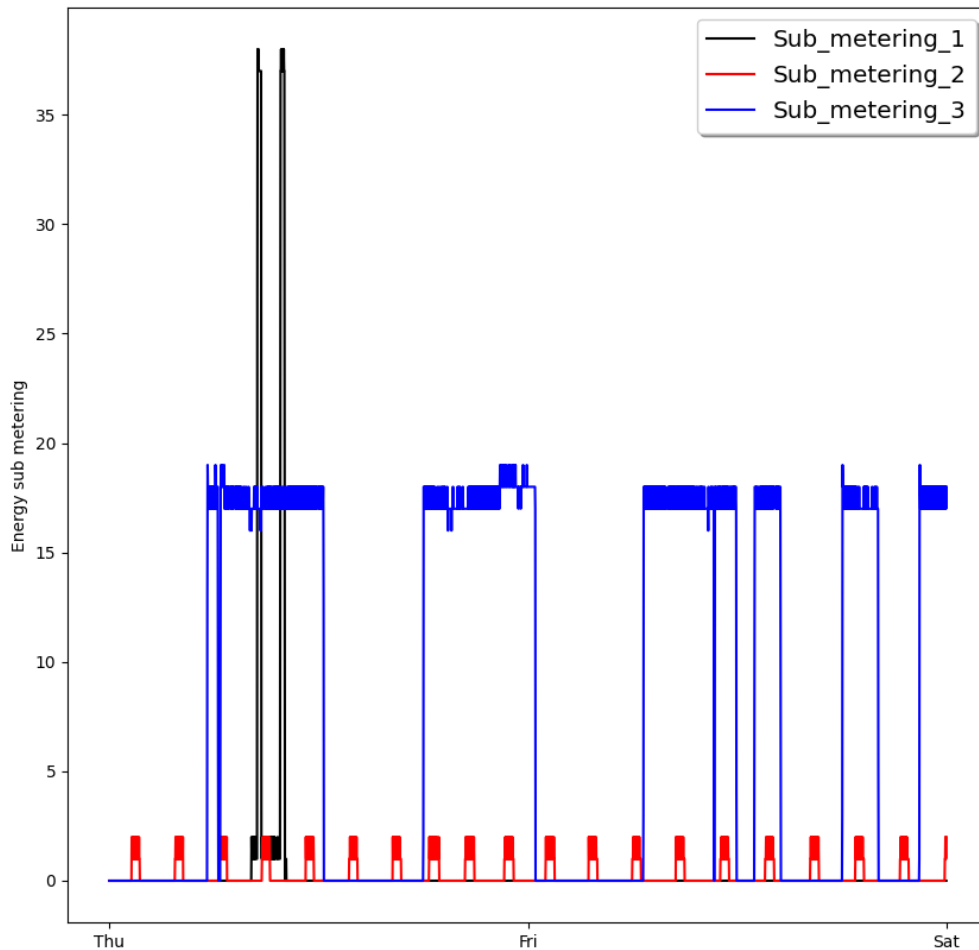
Hình một miêu tả công suất tiêu thụ điện theo đơn vị Kilo-Watts (kW) của các hộ gia đình trong vòng 1 phút.

Nhìn rõ thì ta thấy lượng điện của các hộ gia đình chủ yếu rơi vào mức từ 2kW trở xuống. Còn lại thì có một số nhỏ các trường hợp các gia đình xài nhiều hơn nhưng con số này là rất thấp.



Xem tiếp tới hình 2, hình này miêu tả tổng nhu cầu sử dụng điện của các thiết bị điện (trục tung) tại một thời điểm (trục hoành). Nếu giá trị tại một thời điểm cao tức là tổng nhu cầu sử dụng các thiết bị tăng lên trong khoảng thời gian đó.

Em cũng đã thử plot một số figures về các ngày khác thì thấy rằng trong một ngày thì có chung một cái khung, một ngày sẽ chia ra làm các buổi tạm gọi là sáng-trưa (6h - 12h), trưa-chiều (12h-17h), tối (17h-24h), khuya-sáng (0h-6h). Thì nhu cầu sử dụng điện lớn nhất của các hộ gia đình nằm ở 2 buổi sáng-trưa và tối. Cũng dễ hiểu vì hai khoảng thời gian này là khoảng thời gian mà mọi người sinh hoạt, hoạt động nhiều nhất, như là nấu ăn, thiết bị làm mát, sử dụng thiết bị điện tử, vân vân. Còn khoảng thời gian khuya và trưa-chiều, thông thường mọi người sẽ không có xu hướng hoạt động nhiều, đa phần đây là thời gian nghỉ ngơi của các hộ gia đình.

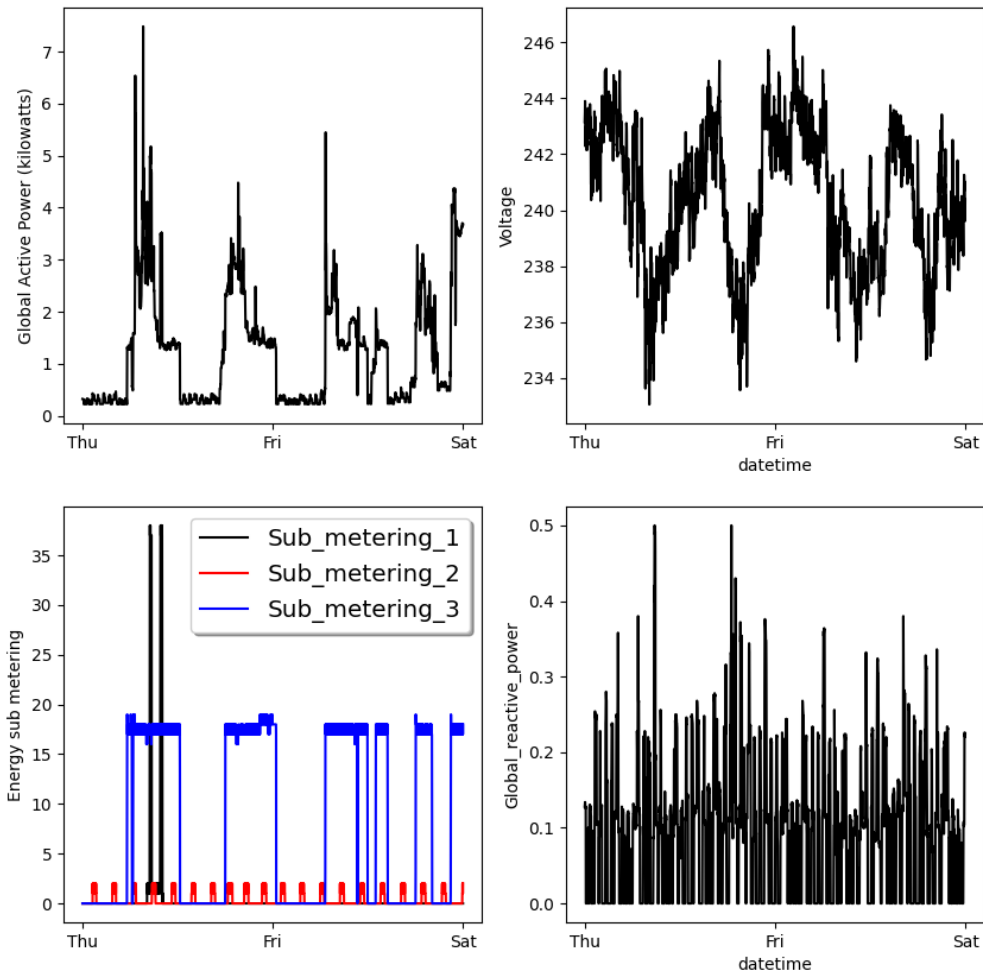


Với hình 3 ta sẽ tiến hành trực quan 3 khía cạnh nhỏ hơn của việc sử dụng điện năng, đầu tiên là nấu nướng (đường màu đen), đường giặt ủi (màu đỏ), đường cuối cùng là điện năng dành cho việc cân bằng nhiệt độ, cụ thể ở đây là điều khiển nhiệt độ nước và máy điều hòa (đường màu xanh nước biển).

Đầu tiên thì ta thấy được đường màu đen khá là ổn định chỉ có 2 điểm thì đường màu đen lên cao thất thường là vào tầm giữa trưa ngày thứ 5. Theo chúng em nghĩ thì nhà này không nấu nướng bằng bếp điện quá nhiều nên mức sử dụng ổn định, việc có 2 điểm dị thường thì có thể là do 2 lần rửa chén cho việc ăn cả tuần hoặc đơn giản là bữa trưa hôm đó có một món cần lò vi sóng (hoặc cả 2). Điện năng tiêu thụ cho 2 điểm này là tầm 38Wh.

Tiếp theo phân tích kỹ đường màu xanh thì thấy cái tần suất xài máy điều hòa khá là dễ hiểu và khá là giống với tổng điện năng. Những giờ hoạt động nhiều thì điều hòa cần hoạt động nhiều, còn ban đêm thì nhiệt độ ổn định hơn. Đây cũng là đường chính khiến dữ liệu chúng ta có phân bố giống với ở hình đầu tiên. Điện năng trung bình tiêu thụ cho mỗi khung giờ mà xài nhiều vào tầm 18Wh

Cuối cùng là đường màu đỏ là đường cho nhu cầu giặt sấy, chiếu sáng và tủ lạnh. Thực ra ở đây tụi em cũng thấy hơi kỳ vì nó có phân bố lúc tắt lúc mở như thế. Vì ở nhà em chưa bao giờ rút cái tủ lạnh ra cả. Điện năng khá đồng đều và có quy luật nằm trong khoảng từ 0Wh đến 2Wh.



Hình 4 thì chỉ là gộp 3 hình trên lại cộng với một đồ thể hiện Voltage (Hiệu điện thế - Trục tung) theo thời gian (Trục hoành). Hiệu điện thế khá là ổn định. Với trung bình tầm 240V và độ lệch chuẩn tầm 1%.

Về các bước thực hiện

Phần code phía dưới là dùng để tải data từ trên mạng, ở đây em dùng 4 thư viện thêm là `os`, `zipfile`, `urllib` và `io` đây là 2 gói có sẵn của Python với mục đích như sau:

- `os` dùng để tìm file bằng các hàm xây dựng sẵn.
- `zipfile` dùng để giải nén cái file có đuôi là `*.zip`
- `urllib` là thư viện để em thực hiện gửi nhận các HTTP Request.
- `io` thì dùng để đọc và chuyển các dữ liệu thành `bytes`

```
import os
import urllib request

DATA_PATH = "../household_power_consumption.txt"

# download the data if it is not already downloaded

if not os.path.exists(DATA_PATH):
    URL = "https://nymph332088.github.io/CIS4340/labassignments/Lab3/household_power_consumption.zip"
    print(f>Data has not downloaded yet, download data at: {URL}")

    from io import BytesIO
    from zipfile import ZipFile
    print("Retrieving the information")
    res = urllib request.urlopen(URL)
    zipfile = ZipFile(BytesIO(res.read()))
    zipfile.extractall(path=".")
else:
    print(f>Data was downloaded")
```

Phân đọc dữ liệu. Em có sử dụng dòng số 23 để thêm 1 cột `Time` vào dữ liệu, cột này gồm thông tin cả ngày và giờ của điểm dữ liệu trên chuỗi thời gian. Em có cột này tại vì xúu nữa sẽ dễ để dùng hàm có sẵn của `matplotlib` hơn.

```
df = pd.read_csv(DATA_PATH, sep=";", na_values="?")

# As suggested, we convert the Date columns following the format DD/MM/YYYY.
df.Date = pd.to_datetime(df.Date, format="%d/%m/%Y")

# Settings the filter for choosing the date that we care about, here is 01.02.2007 and 02.02.2007
start_date = '2007-02-01'
end_date = '2007-02-02'
filt = (df.Date >= start_date) & (df.Date <= end_date)
selected_df = df.loc[filt]

selected_df.Time = pd.to_datetime(selected_df.Date.astype(str)+' ' + selected_df.Time)
```

Ở đây em sẽ giải thích sơ qua về một trong 4 hàm plot. Đó là hàm `plot2`

Thực ra em thấy ở đây chỉ có 2 dòng hơi khó hiểu cho người đọc. Đó là 2 dòng

```
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%a'))
plt.gca().xaxis.set_major_locator(mdates.DayLocator())
```

Đây là 2 dòng dùng để hiện tên viết tắt của mấy cái thứ `Thu Fri Sat`.

Lúc đầu thì em định rằng sẽ scale hết ngày giờ về từ \$0\$ đến \$1\$ rồi em sẽ plot ra. Sau đó dùng `set_xticklabel` nhưng mà em làm nó khá là lỗi, và kiểu 3 cái `Thu Fri Sat` nó bị dồn lại làm một. Em cũng thử debug, nhưng em có tra thử Google thì thấy được đường link [này](#). cộng với đọc thêm document ở [đây](#) và cuối cùng là xem thêm chuẩn format thời gian quốc tế ở [W3School](#) và tụi em đã sửa được nó thành như phía trên.

```
def plot2():
    plt.figure(figsize=(10,10))
    # config the axis following the format in https://www.w3schools.com/python/python_datetime.asp
    # and the web we got https://www.earthdatascience.org/courses/use-data-open-source-python/use-time-series-data-in-python/date-time-types-in-pand
    # The docs can be found on the main page of matplotlib
    plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%a'))
    plt.gca().xaxis.set_major_locator(mdates.DayLocator())
    plt.plot(selected_df.Time, selected_df.Global_active_power, color="#000")
    plt.ylabel("Global Active Power (kilowatts)")
    plt.savefig("../figures/plot2.png")
    plt.show()
```

