Matplotlib

```
import pandas as pd
import matplotlib.pyplot as plt


from google.colab import drive
import os

drive.mount('/content/drive')
os.chdir('/content/drive/MyDrive/')
for item in os.listdir():
  print(item)
print("--------------")
os.chdir('/content/drive/MyDrive/cloud/GitHub/AdvDataViz/Notebooks/')
for item in os.listdir():
  print(item)
print("--------------")
notebooks = "/content/drive/MyDrive/cloud/GitHub/AdvDataViz/Notebooks"
print(os.listdir(notebooks))
print("--------------")

file = "heart-disease.csv"
file_path = os.path.join(notebooks, file)
with open(file_path, "r") as f:
  contents = f.read()
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/con
learningStore
healthyCar
startup
Artificial Intelligence
cloud
--------------
student_performance.csv
churn.csv
heart-disease.csv
employee_attrition_.csv
Top 50 US Tech Companies.csv
01 Python_Pandas.ipynb
02 Matplotlib.ipynb
03 Matplotlib  - Exercise.ipynb
03 Matplotlib - Exercise Solutions.ipynb
04 Continuous Variables - Histogram .ipynb
05 Continuous Variables - Histogram - Exercise Solutions.ipynb
05 Continuous Variables - Histogram - Exercise .ipynb
06 Continuous Variables - Boxplot.ipynb
07 Continuous Variables - Boxplot - Exercise .ipynb
07 Continuous Variables - Boxplot - Exercise Solutions.ipynb
08 Continuous Variables - Scatterplot.ipynb
09 Continuous Variables - Scatterplot - Exercise .ipynb
09 Continuous Variables - Scatterplot - Exercise Solutions.ipynb
11 Pandas Data Visualization.ipynb
12 Seaborn.ipynb
13 Seaborn - Exercise .ipynb
13 Seaborn - Exercise Solution.ipynb
myplotlib.py
10 Categorical Variables - Bar_Pie.ipynb
14 Functions.ipynb
```

```
15 Custom Modules.ipynb
--------------
['student_performance.csv', 'churn.csv', 'heart-disease.csv', 'employee_attrition_.csv', 'Top 5
--------------
```

Dataset: Heart Disease

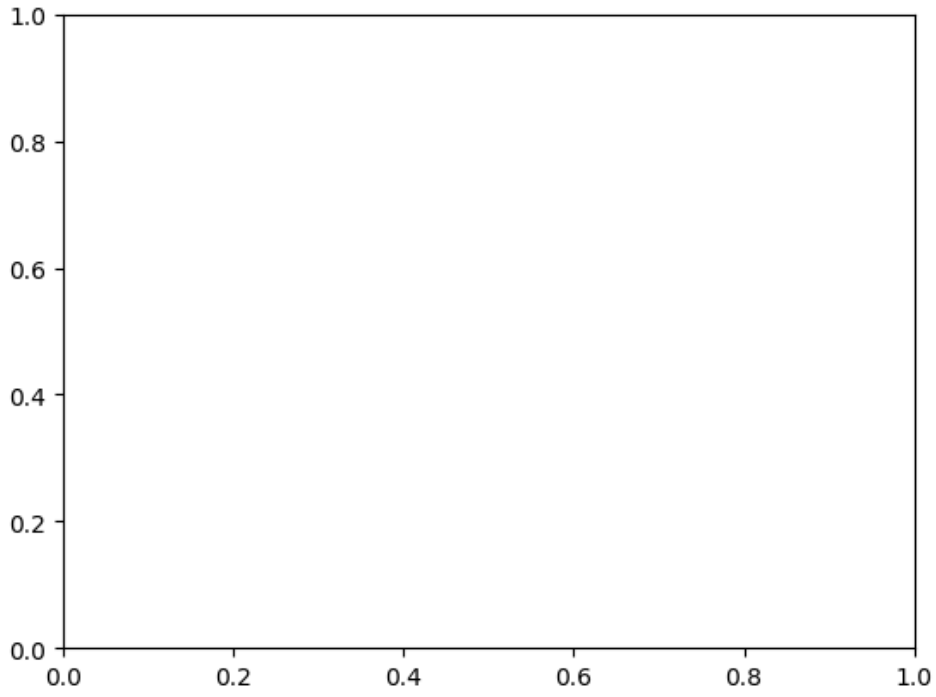
```
#df = pd.read_csv("heart-disease.csv")

df = pd.read_csv(file_path)

df.head()
```

	age	sex	chest_pain	rest_bp	chol	max_hr	st_depr	heart_disease
0	63	female	3	145	233	150	2.3	1
1	37	female	2	130	250	187	3.5	1
2	41	male	1	130	204	172	1.4	1
3	56	female	1	120	236	178	0.8	1
4	57	male	0	120	354	163	0.6	1

Next steps: Generate code with df | View recommended plots | New interactive sheet

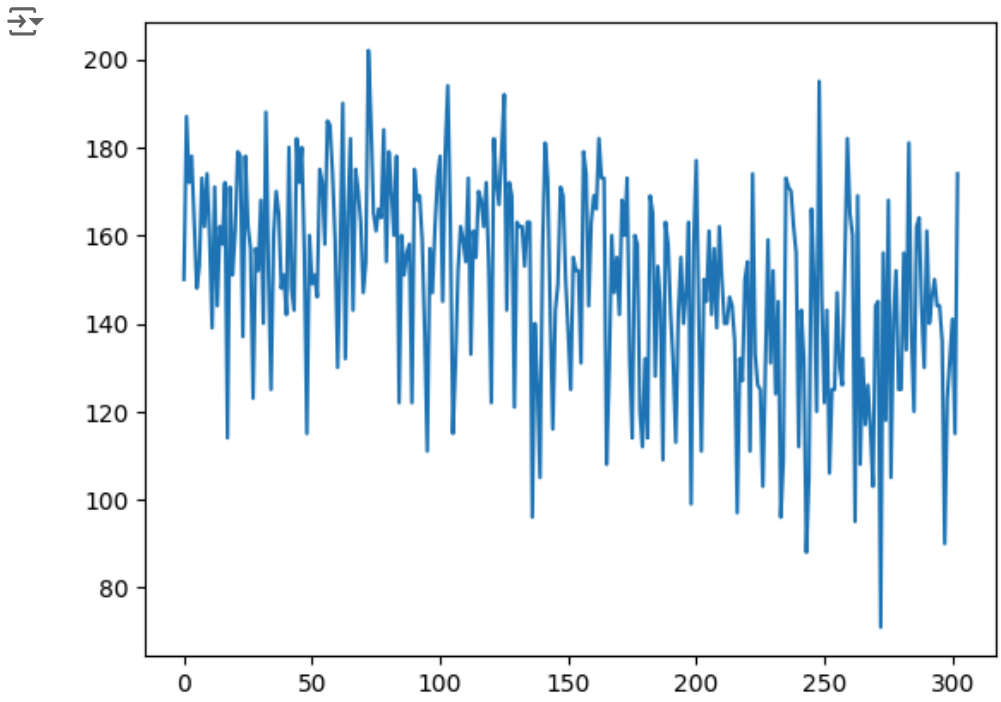
Figures and Axes

```
# subplots() returns a figure and axes object (unpacked)

fig, ax = plt.subplots()
```



Line Plot

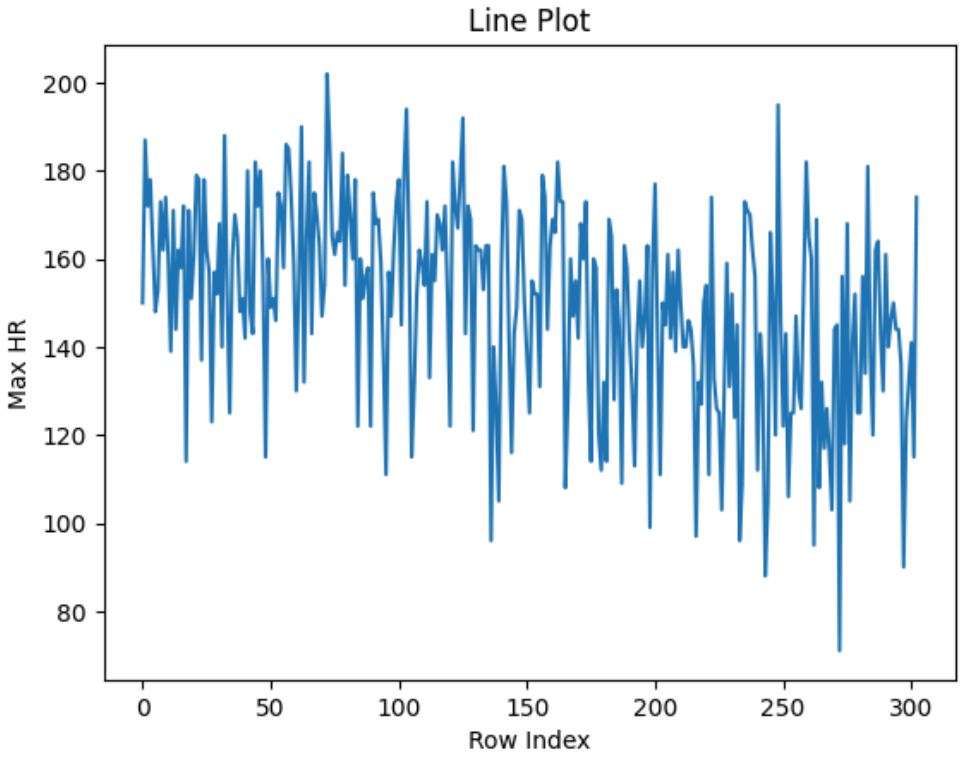
```
# Defaults to display the index of the row on the x-axis

fig, ax = plt.subplots()

ax.plot(df["max_hr"]);
```



Set properties

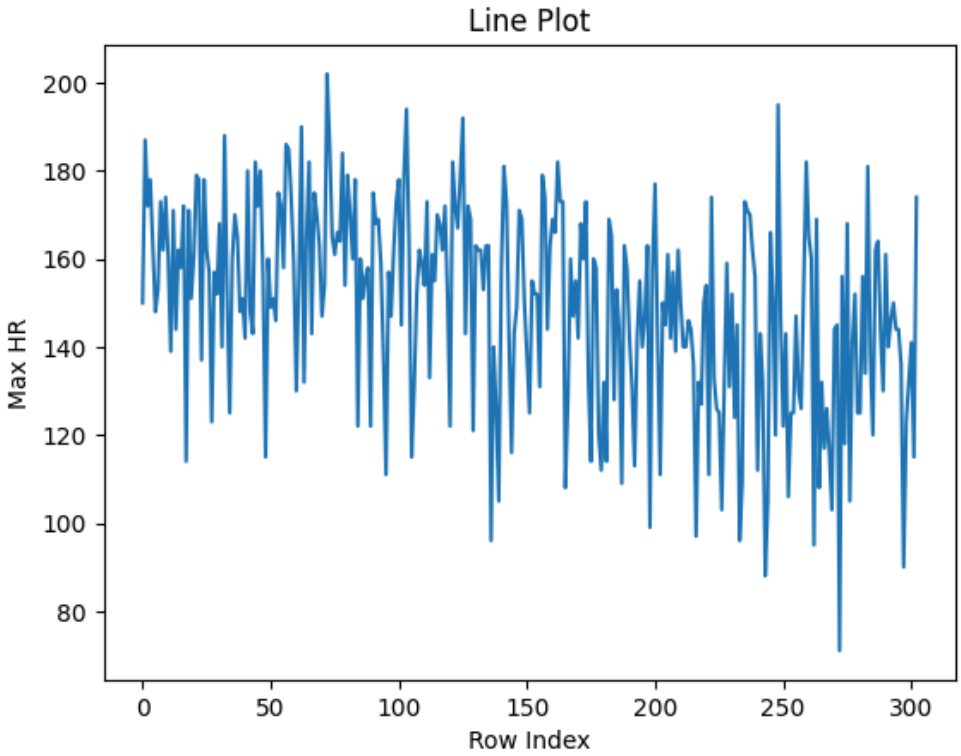
A more flexible way of setting properties

```
fig, ax = plt.subplots()

ax.plot(df["max_hr"])

# A more flexible way of setting properties
ax.set_title("Line Plot")
ax.set_xlabel("Row Index")
ax.set_ylabel("Max HR");
```



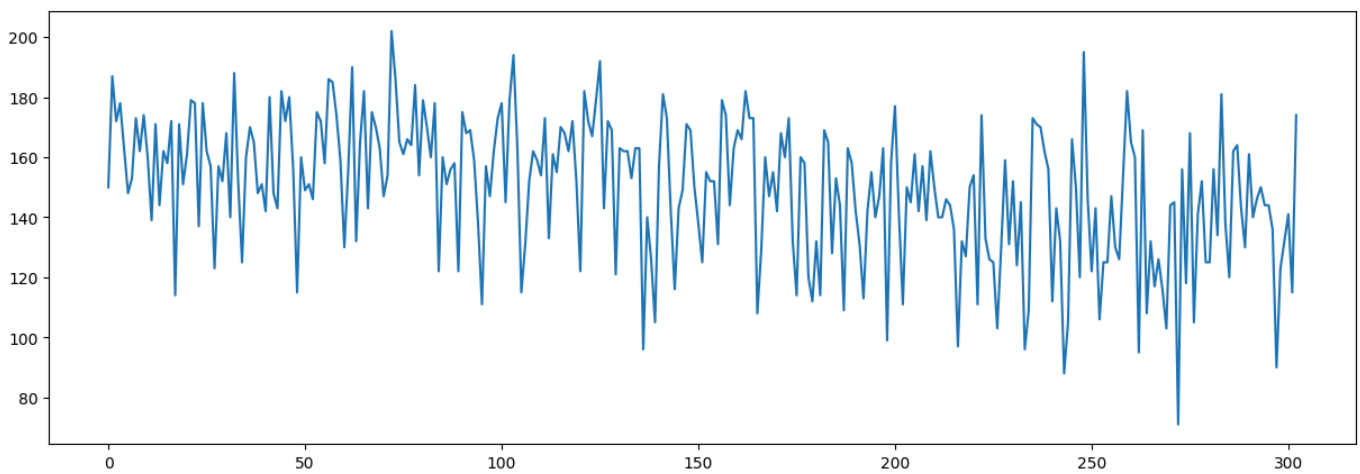
A more convenient way of setting properties

```
fig, ax = plt.subplots()

ax.plot(df["max_hr"]);

# A more convenient way of setting properties
ax.set(title="Line Plot", xlabel="Row Index", ylabel="Max HR");
```



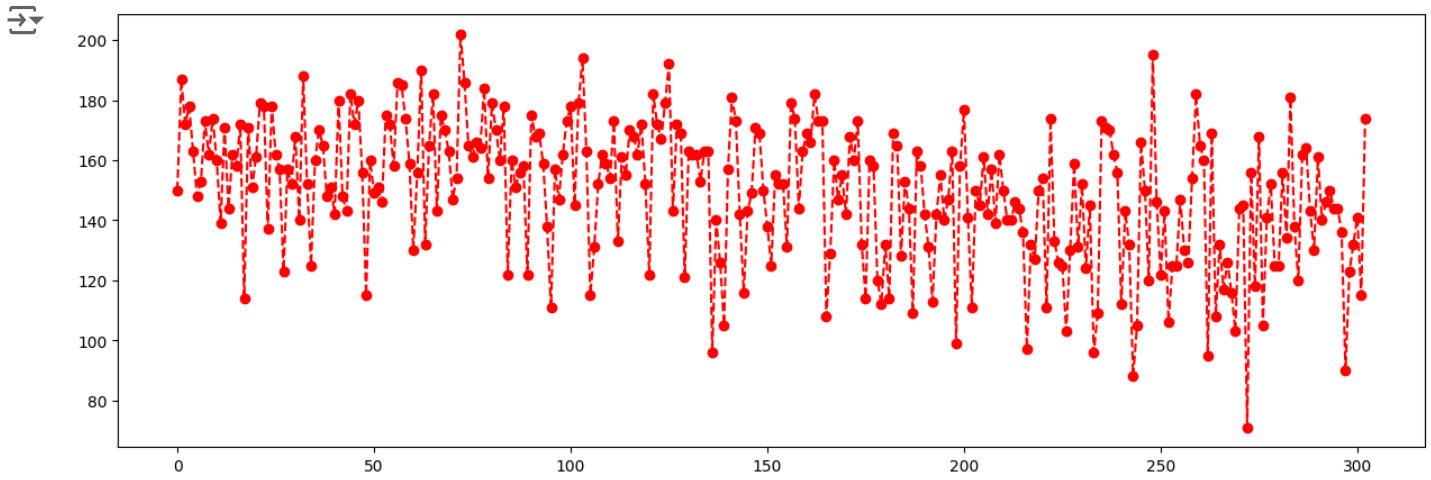
Set the figsize

```
# Set the size of the figure (figsize=(w, h))

fig, ax = plt.subplots(figsize=(15,5))

ax.plot(df["max_hr"]);
```



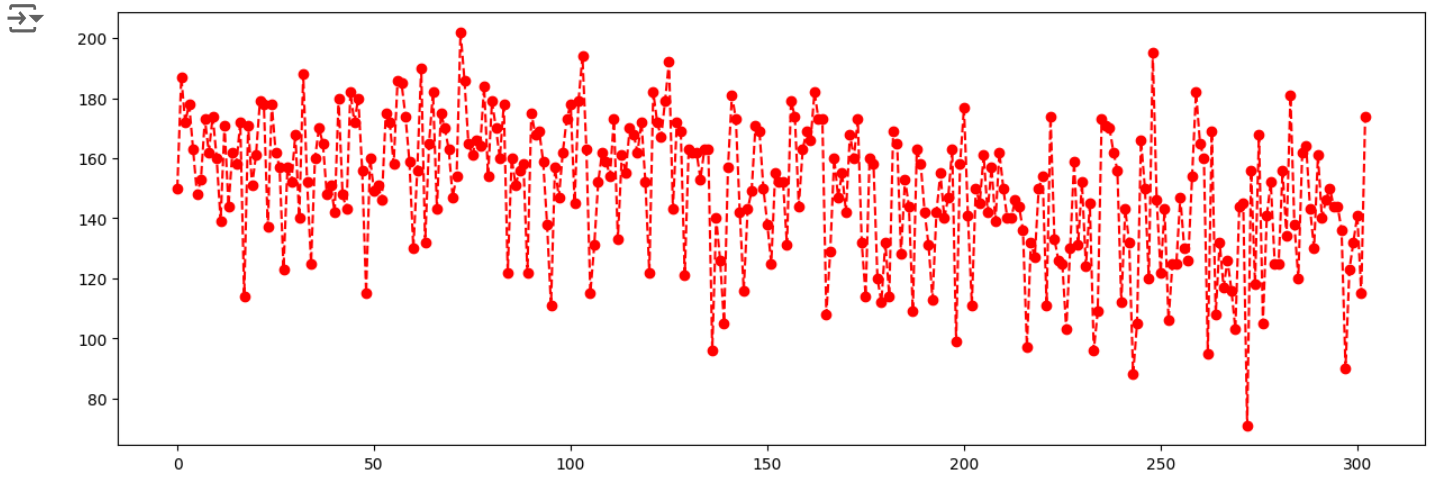
Modify line style

```
fig, ax = plt.subplots(figsize=(15,5))

                                # linestyles: (dashdot, dotted, solid), markers: ("o", "s", v", "x'
ax.plot(df["max_hr"], color="red", linestyle="dashed", marker="o");
```



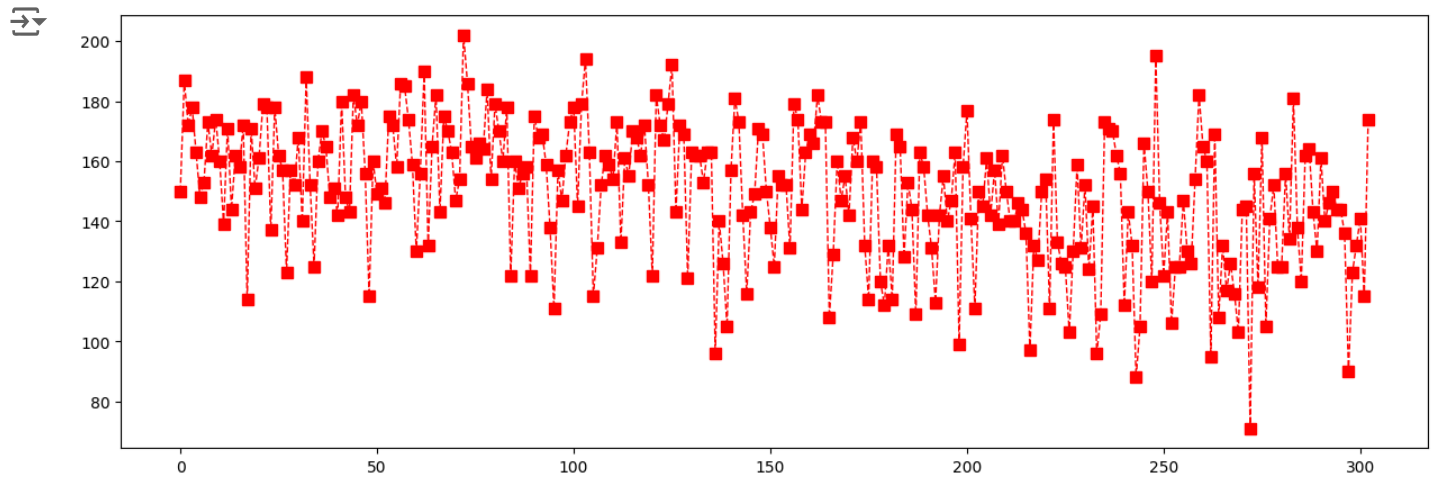
Modify line style with format string

```
fig, ax = plt.subplots(figsize=(15,5))

                    # format string: color, linestyle, marker
                                # red,  dashed,  circle
ax.plot(df["max_hr"], "r--o");
```



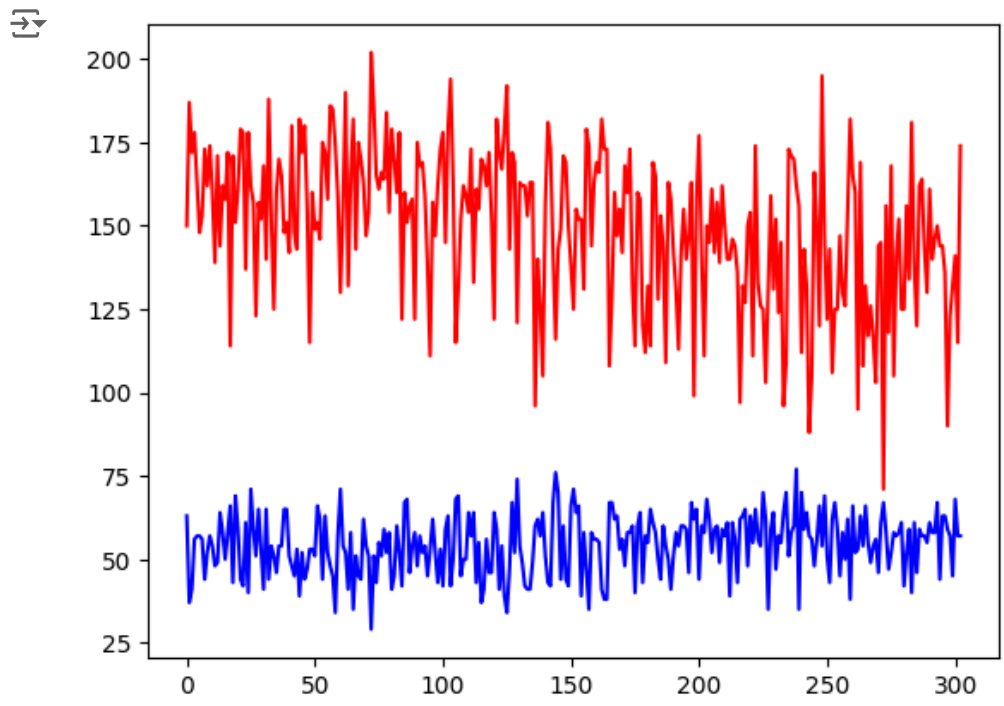
Customize markersize and linewidth

```
fig, ax = plt.subplots(figsize=(15,5))
                                                    # square
ax.plot(df["max_hr"], color="red", linestyle="dashed", marker="s", markersize=7, linewidth=1);
```



Create multiple plots in same axis

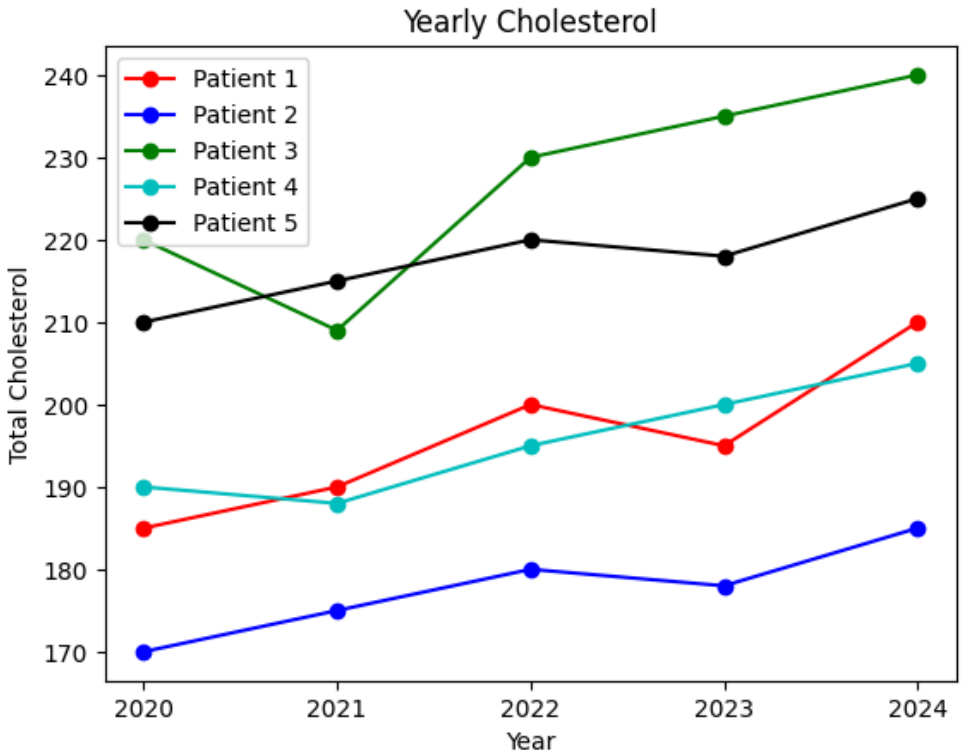
```
fig, ax = plt.subplots()

ax.plot(df["max_hr"], color="red")
ax.plot(df["age"], color="blue");
```



Provide the values to be plotted

Yearly Total Cholesterol Values for Various Patients

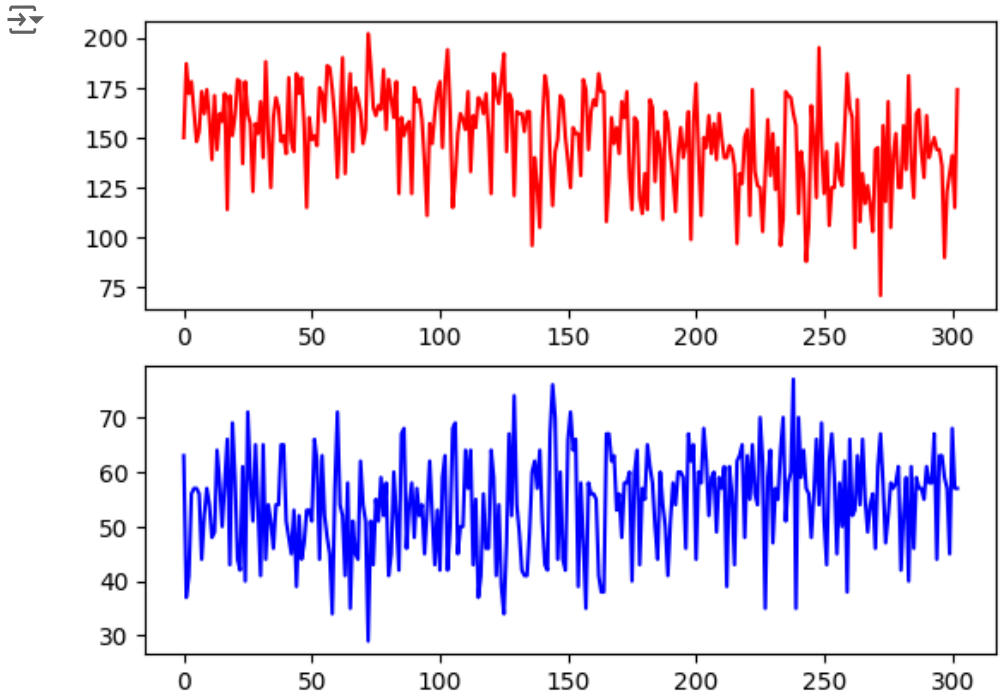
```
fig, ax = plt.subplots()

# x-axis
years = ["2020", "2021", "2022", "2023", "2024"]

# y-axis
p1 = [185, 190, 200, 195, 210]
p2 = [170, 175, 180, 178, 185]
p3 = [220, 209, 230, 235, 240]
p4 = [190, 188, 195, 200, 205]
p5 = [210, 215, 220, 218, 225]

                        # label is used for the legend
ax.plot(years, p1, "r-o", label="Patient 1")
ax.plot(years, p2, "b-o", label="Patient 2")
ax.plot(years, p3, "g-o", label="Patient 3")
ax.plot(years, p4, "c-o", label="Patient 4")
ax.plot(years, p5, "k-o", label="Patient 5")

ax.set_title("Yearly Cholesterol")
ax.set_xlabel("Year")
ax.set_ylabel("Total Cholesterol")

# Display a legend
ax.legend();

# Display a legend (just outside of the plot; up 1, and over to the right 1)
#ax.legend(bbox_to_anchor=(1, 1));
```



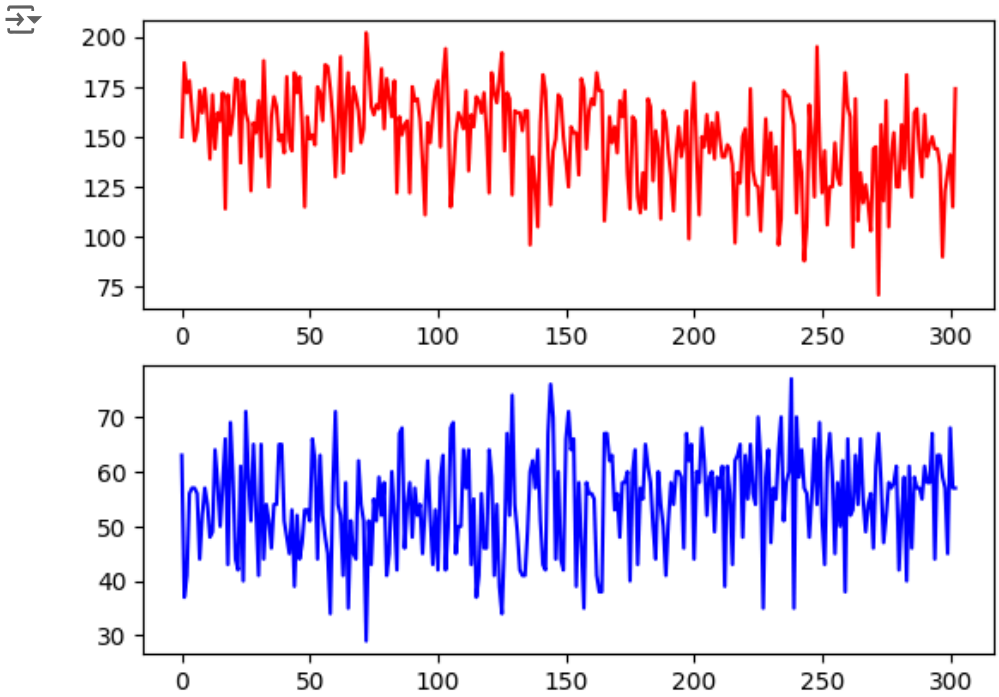
Create multiple plots in separate axes

Separate rows (unpacking axes)

```
# Render plots in separate axes;   subplots(n_rows, n_cols)

#      tuple                # rows
fig, (ax1, ax2) = plt.subplots(2)

ax1.plot(df["max_hr"], color="red")
ax2.plot(df["age"], color="blue");
```

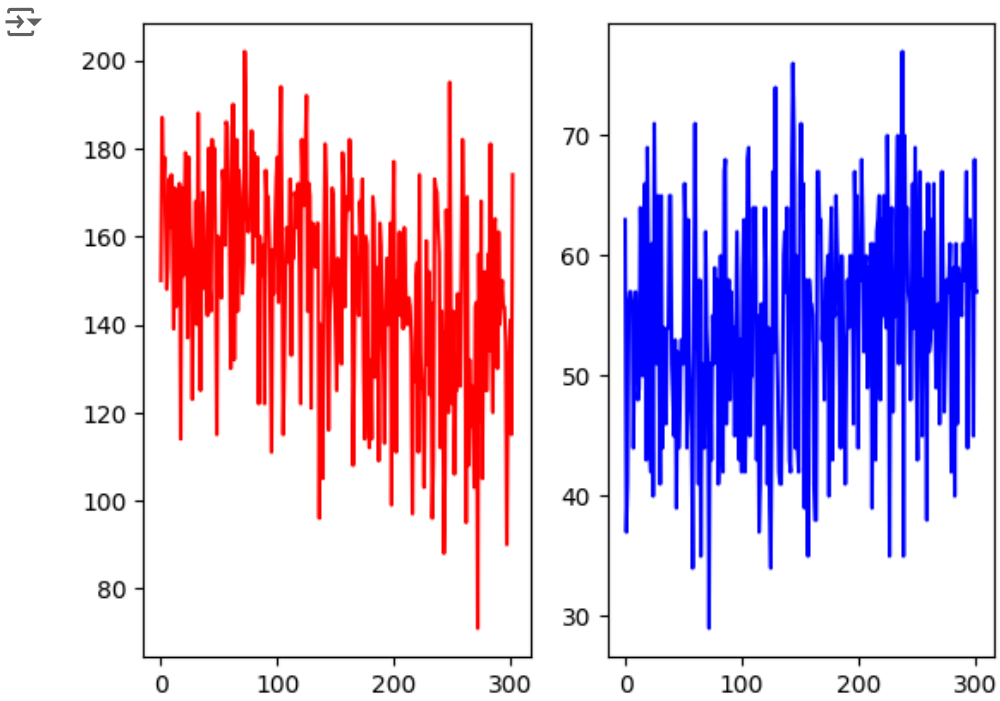



```
# Render plots in separate axes;   subplots(n_rows, n_cols)

#      tuple                # rows
fig, (top, bot) = plt.subplots(2)

top.plot(df["max_hr"], color="red")
bot.plot(df["age"], color="blue");
```



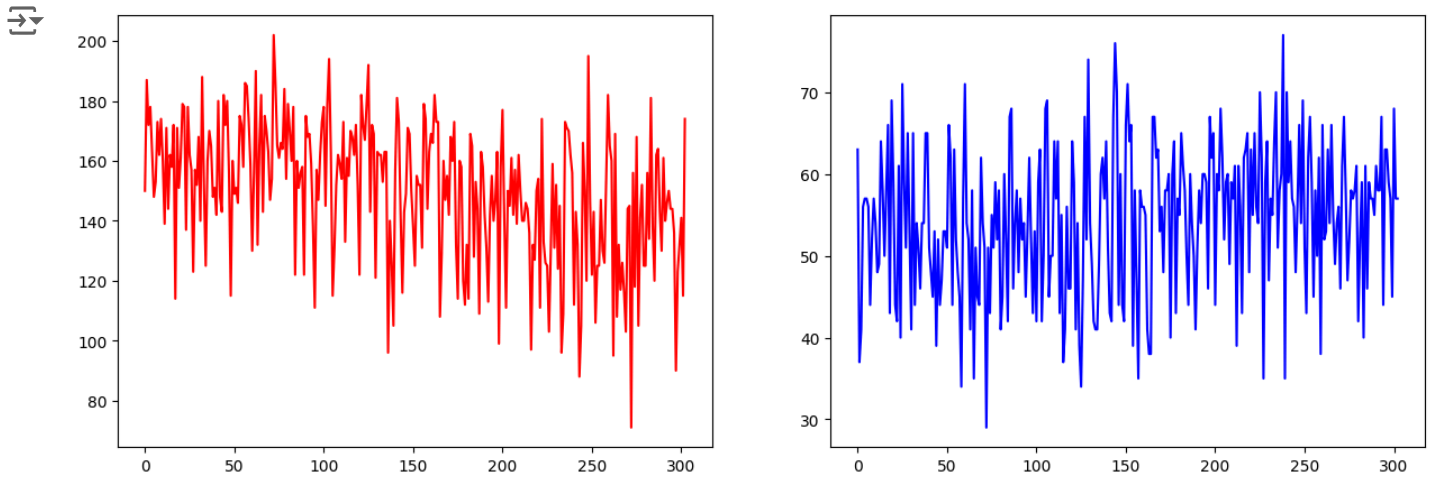
Separate columns

```
# Render plots in separate axes;   subplots(n_rows, n_cols)

                          # 1 row, 2 columns
fig, (left, right) = plt.subplots(1,2)


left.plot(df["max_hr"], color="red")
right.plot(df["age"], color="blue");
```



Set the figsize

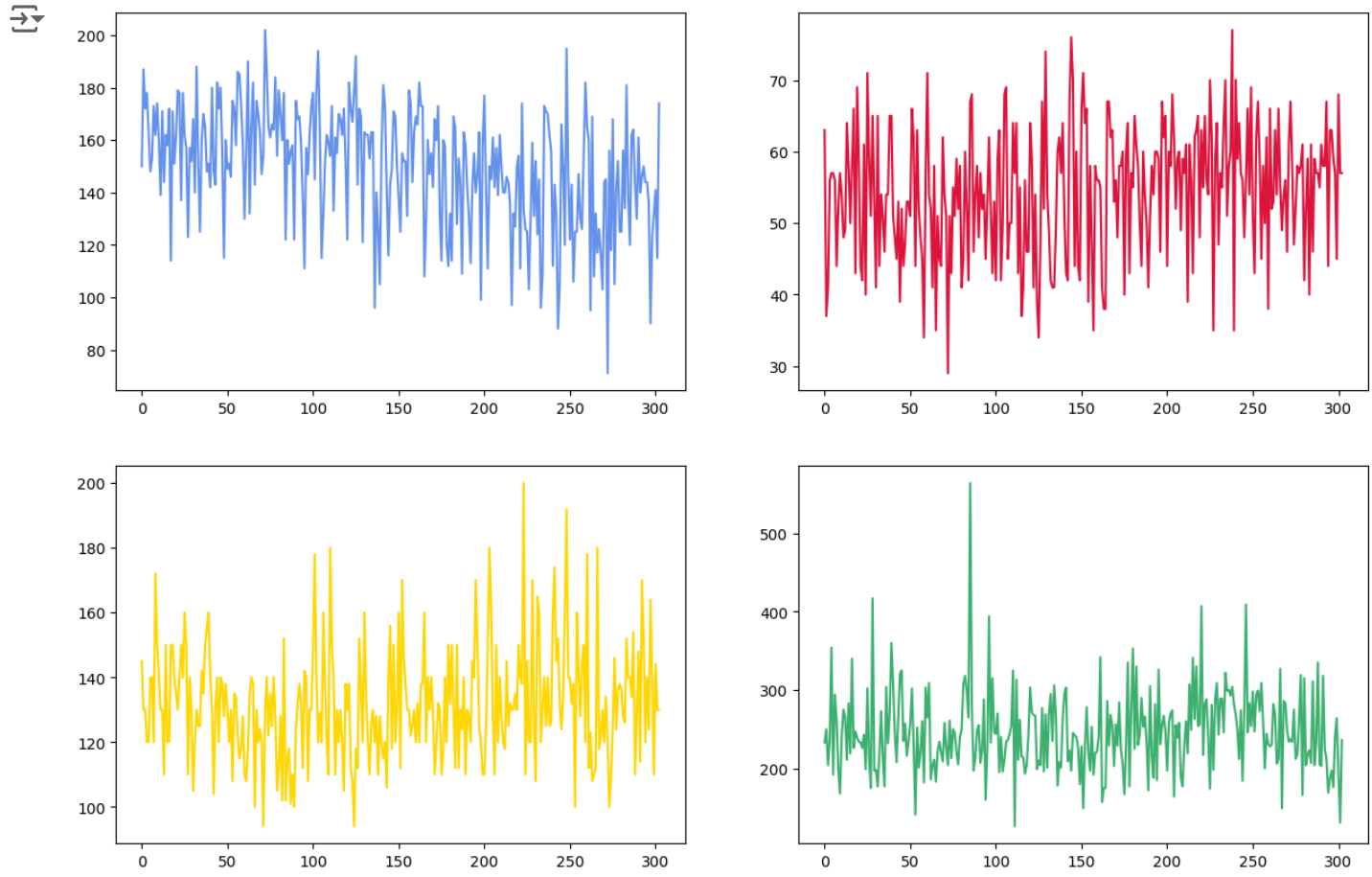
```
# Set the size of the figure           (figsize=(w, h))
fig, (left, right) = plt.subplots(1,2, figsize=(15, 5))


left.plot(df["max_hr"], color="red")
right.plot(df["age"], color="blue");
```



Multiple panel plots

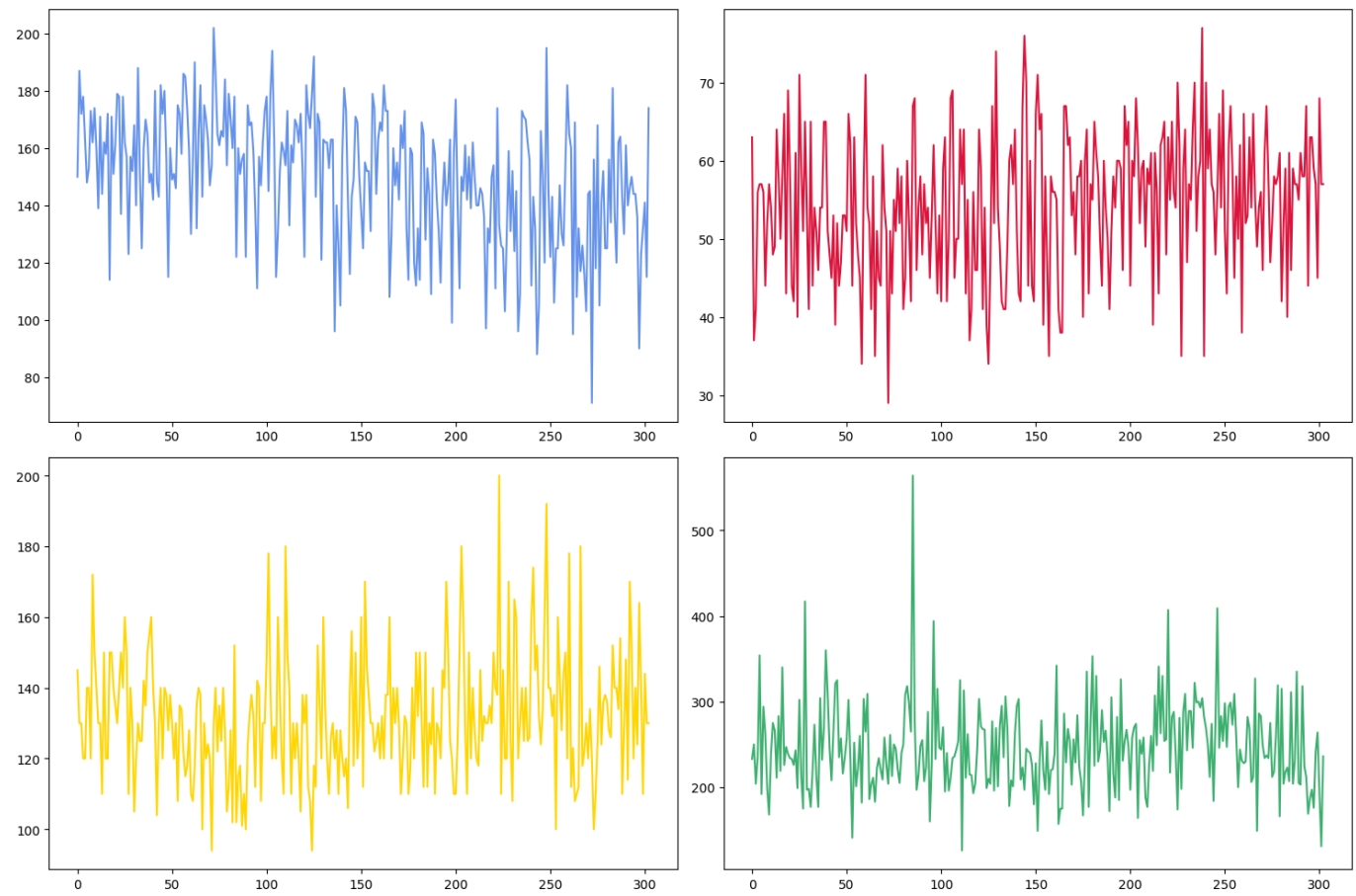
[Named Colors]

```
# Unpack the axes
fig, ((top_L,top_R),(bot_L,bot_R)) = plt.subplots(2,2, figsize=(15, 10))


top_L.plot(df["max_hr"], color="cornflowerblue")
top_R.plot(df["age"], color="crimson")
bot_L.plot(df["rest_bp"], color="gold")
bot_R.plot(df["chol"], color = "mediumseagreen");
```



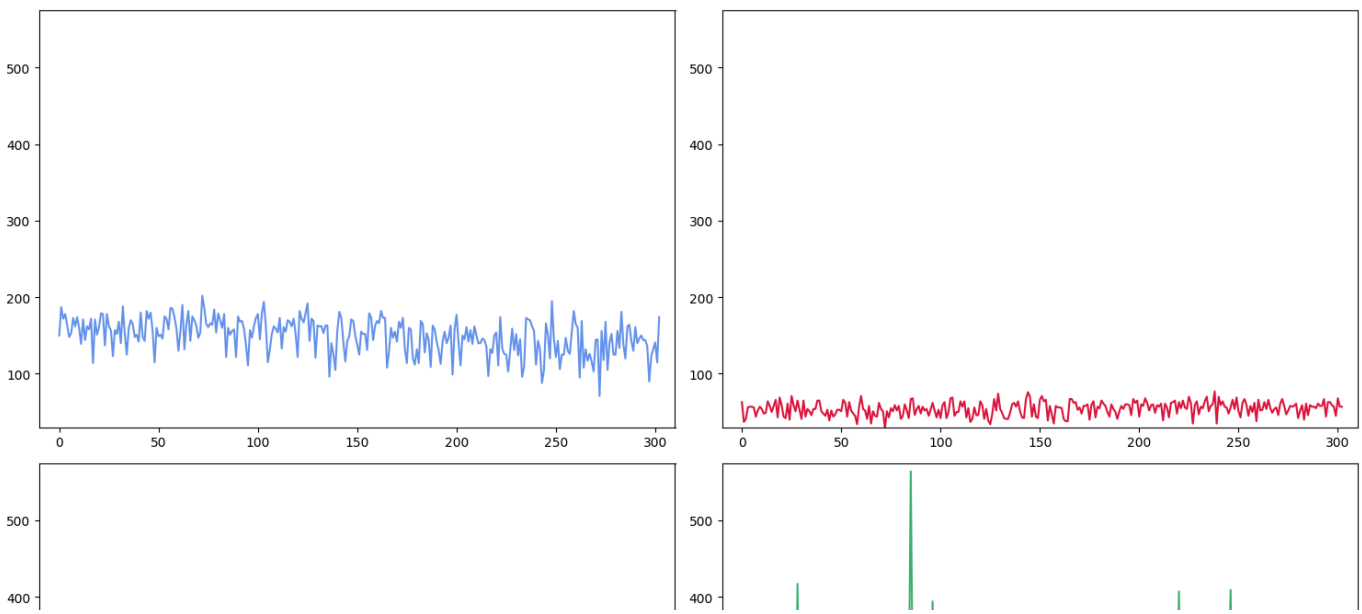
Improve (tighten) the layout

```
fig, ((top_L,top_R),(bot_L,bot_R)) = plt.subplots(2,2, figsize=(15, 10))


top_L.plot(df["max_hr"], color="cornflowerblue")
top_R.plot(df["age"], color="crimson")
bot_L.plot(df["rest_bp"], color="gold")
bot_R.plot(df["chol"], color = "mediumseagreen")


# Adjusts (tighten) the layout for better appearance
plt.tight_layout();
```



Scale all axes to the same range of values

```
fig, ((top_L,top_R),(bot_L,bot_R)) = plt.subplots(2,2, figsize=(15, 10))


top_L.plot(df["max_hr"], color="cornflowerblue")
top_R.plot(df["age"], color="crimson")
bot_L.plot(df["rest_bp"], color="gold")
bot_R.plot(df["chol"], color = "mediumseagreen")


#Scale the plots to have the same x-axis and y-axis dimensions
# x-axis
top_L.set_xlim((-10, 310))
top_R.set_xlim((-10, 310))
bot_L.set_xlim((-10, 310))
bot_R.set_xlim((-10, 310))

# y-axis
top_L.set_ylim((30, 575))
top_R.set_ylim((30, 575))
bot_L.set_ylim((30, 575))
bot_R.set_ylim((30, 575))


plt.tight_layout();
```


500
400
300
200
100
0
50
100
150
200
250
300
500
400
300
200
100
0
50
100
150
200
250
300
500
400
500
400