matplotlib-introduction

December 26, 2025

1 Introduction to Matplotlib

[ ]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

[ ]:
```
plt.plot() # create a empty box
```

[ ]:
```
plt.plot(); # -> semicolon for the array
```

[ ]:
```
plt.plot()
plt.show() # -> same action like a semicolon
```

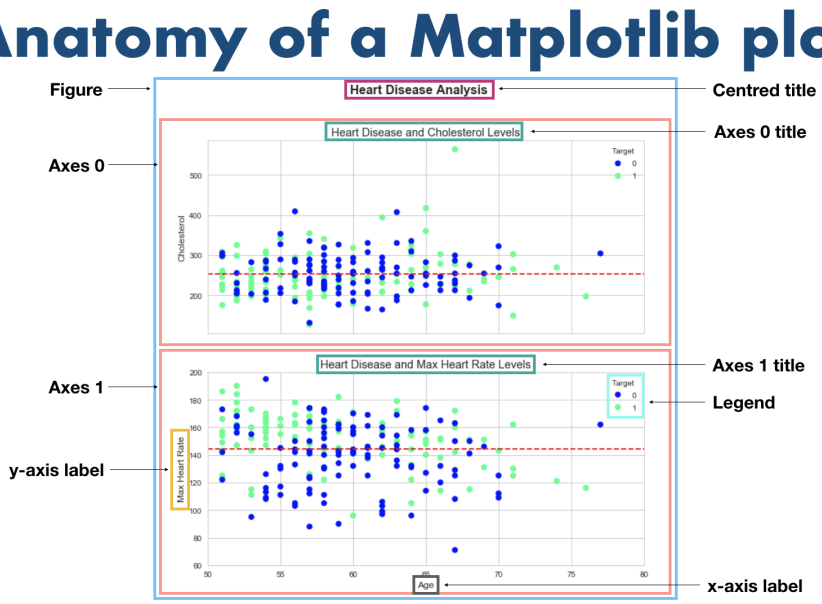
[ ]:
```
plt.plot([1,2,3]);
```

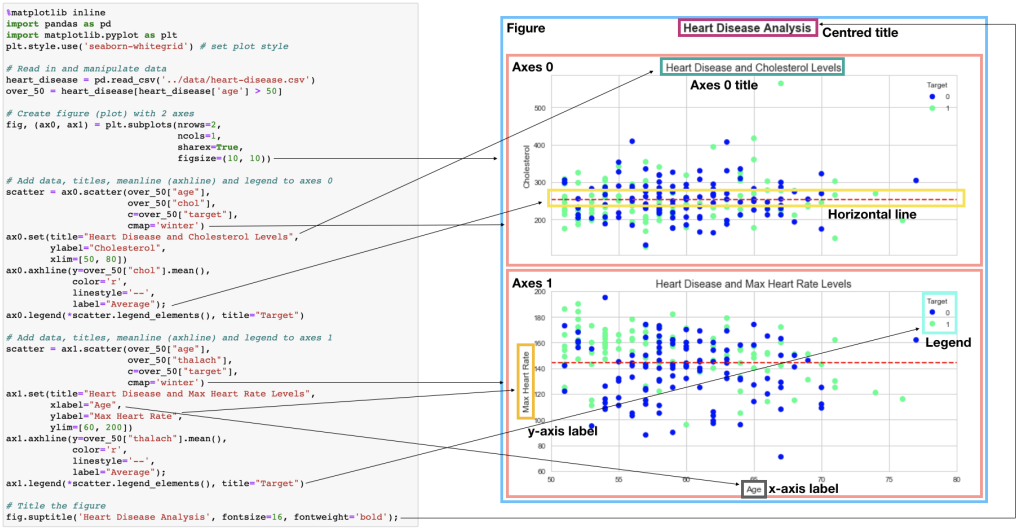
[ ]:
```
x = [1,2,3,4]
y = [11,22,33,44]
plt.plot(x,y);
```

[ ]:
```
# 1st method
fig = plt.figure(); # creates a figure
ax = fig.add_subplot() # adds some axex
plt.show();
```

[ ]:
```
#second method
fig = plt.figure()
ax = fig.add_axes([1,1,1,1])
ax.plot(x,y) # add some data
plt.show();
```

[ ]:
```
# third method (recommended)
fig, ax = plt.subplots()
ax.plot(x,y);
```





2 Matplotlib example Workflow

```
[ ]: # 0 import matplotlib
     import matplotlib.pyplot as plt

     # 1 prepare date
     x=[1,2,3,4]
     y=[11,22,33,44]

     # 2 setup plot
```

```
fig, ax = plt.subplots(figsize =(5,10)) # width and height

# 3 plot data
ax.plot(x,y)

# 4 Customize plot
ax.set(title="Simple Plot",
       xlabel="x-axis",
       ylabel="y-axis")

# 5 Save & show

fig.savefig("./export_plot.png")
```

3 Making figures with NumPy arrays

We want: * Line Plot * Scatter Plot * Bar Plot * Histogram * Subplots

[ ]:
```
#Create some date
x = np.linspace(0,10,100) #Return evenly spaced numbers over a specified␣
↪interval.
x
```

[ ]:
```
#plot the data
fig, ax = plt.subplots()
ax.plot(x,x**2); # default line plot
```

[ ]:
```
# use the same data for a scatter plot

fig, ax = plt.subplots()
ax.scatter(x, np.exp(x));
```

[ ]:
```
# Another scatter
fig, ax = plt.subplots()
ax.scatter(x, np.sin(x));
```

[ ]:
```
# Make a plot from dictionary
nut_butter_prices = {"almond butter":10,
                     "Peanut butter": 20,
                     "Cashew butter":22}
fig, ax = plt.subplots()
ax.bar(nut_butter_prices.keys(),nut_butter_prices.values()); # Bar diagram
ax.set(title="Romik's Butter Store",
       ylabel="Price €",
       xlabel="Name");
```

[ ]:
```
fig, ax = plt.subplots()
ax.barh(nut_butter_prices.keys(),nut_butter_prices.values()); #horizontal bar
```

[ ]:
```
# Histogram
x = np.random.randn(1000)
fig, ax = plt.subplots();
ax.hist(x);
```

4 two options for subplots

[ ]:
```
#option 1
fig,  ((ax1,ax2),(ax3,ax4)) = plt.subplots(nrows=2,
                                           ncols=2,
                                           figsize=(10,5))
ax1.plot(x,x/2)
ax2.scatter(np.random.random(10),np.random.random(10))
ax3.bar(nut_butter_prices.keys(),nut_butter_prices.values());
ax4.hist(x);
```

[ ]:
```
# option 2
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(10,5))

# Plot to each diffrent index
ax [0,0].plot(x,x/2)
ax[0,1].scatter(x,x/2)
ax[1,0].bar(x,x/2)
ax[1,1].hist(x)
```

5 Plotting from DataFrame Pandas

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

[ ]:
```
# make a df
car_sales = pd.read_csv("./car-sales.csv")
car_sales
```

[ ]:
```
ts = pd.Series (np.random.randn(1000), index = pd.date_range("1/1/2026",␣
↪periods=1000))
ts
```

[ ]:
```
ts = ts.cumsum() # sum all values
ts.plot()
```

[ ]:
```
car_sales
```

```
[ ]: car_sales["Price"] = car_sales["Price"].str.replace('[\$\,\.]','', regex = True).
     ↪astype(int) / 100
```

```
[ ]: car_sales
```

```
[ ]: car_sales["Sale Date"]= pd.date_range("1/1/2026", periods = len (car_sales))
     car_sales
```

```
[ ]: car_sales["Total Sales"] = car_sales["Price"].cumsum()
```

```
[ ]: car_sales
```

```
[ ]: # Let's plot the total sales
     car_sales.plot(x = "Sale Date", y = "Total Sales");
```

```
[ ]: car_sales.plot(x ="Odometer (KM)", y ="Price", kind = "scatter")
```

```
[ ]: # How about a bar graph?
     x = np.random.rand(10,4)
     x

     # turn into a df
     df = pd.DataFrame(x, columns =['a','b','c','d'])
     df
```

```
[ ]: df.plot.bar()
```

```
[ ]: df.plot(kind = "bar");
```

```
[ ]: car_sales
```

```
[ ]: car_sales.plot(x="Make", y = "Odometer (KM)", kind = "bar");
```

```
[ ]: car_sales.groupby("Make")["Odometer (KM)"].mean().plot(kind="bar", figsize=(10,␣
     ↪5))
```

```
[ ]: # How about histograms?
     car_sales["Odometer (KM)"].plot.hist();
```

```
[ ]: car_sales["Odometer (KM)"].plot(kind = "hist")
```

```
[ ]: car_sales["Odometer (KM)"].plot.hist(bins = 20)
```

```
[ ]: # lets try on a another dataset
     heart_disease = pd.read_csv("heart-disease.csv")
     heart_disease[:10]
```

[ ]:
```
# Create a histogram of age
heart_disease["age"].plot.hist(bins = 50)
```

[ ]:
```
heart_disease.head()
```

[ ]:
```
heart_disease.plot.hist(figsize=(10,50),subplots = True);
```

5.0.1 Which one you should use (pyplot or OO method)

- when plotting something quickly, okay to use the pyplot method
- when plotting something more advanced, use the OO method

[ ]:
```
heart_disease[:10]
```

[ ]:
```
over_50 = heart_disease[heart_disease["age"] > 50]
over_50
```

[ ]:
```
len(over_50)
```

[ ]:
```
#pyplot method
over_50.plot(kind = "scatter",
             x = "age",
             y= "chol",
             c = "target")
```

[ ]:
```
#oo method mixed with pyplot method
fig, ax = plt.subplots (figsize=(10,6))
over_50.plot(kind = "scatter",
             x = "age",
             y= "chol",
             c = "target",
            ax=ax);
ax.set_xlim([45,100])
```

[ ]:
```
# OO method from scratch
fig, ax = plt.subplots (figsize=(10,6))

# Plot the Data
scatter = ax.scatter ( x = over_50["age"],
                       y =  over_50["chol"],
                       c =  over_50["target"]);

# Customize
ax.set(title = "Heart Disease and Cholestrol Levels",
       xlabel ="Age",
       ylabel = "Cholesterol");

# add a legend
```

```
ax.legend(*scatter.legend_elements(), title = "Target");

#Add a h-line
ax.axhline(over_50["chol"].mean(),
           linestyle = "--");
```

[ ]:
```
over_50.head()
```

[ ]:
```
# subplot of chol ,age, thalach
fig , (ax0,ax1) = plt.subplots (ncols=1,
                                nrows =2,
                                figsize = (10,10))
# Add data to ax0
scatter = ax0.scatter( x = over_50["age"],
                       y = over_50["chol"]
                       , c = over_50["target"],
                      cmap ="winter") ## changes the colour sheme
# Customize
ax0.set(title = "Heart Disease and Cholestrol Levels",
        xlabel ="Age",
        ylabel = "Cholesterol");

# add a legend
ax0.legend(*scatter.legend_elements(), title = "Target");

#Add a h-line
ax0.axhline(over_50["chol"].mean(),
           linestyle = "--");



# Add data to ax1

scatter = ax1.scatter( x = over_50["age"],
                       y = over_50["thalach"]
                       , c = over_50["target"],
                      cmap ="winter")
# Customize
ax1.set(title = "Heart Disease and Max Heart Rate",
        xlabel ="Age",
        ylabel = "Max Heart Rate");

ax1.legend(*scatter.legend_elements(),title ="Target")

#Add a h-line
ax1.axhline(over_50["thalach"].mean(),
           linestyle = "--");
```

```
# add a title to the figure
fig.suptitle("Heart Disease Analysis", fontsize = 16, fontweight = "bold")
```

5.1 Customizing Matplotlib

[ ]:
```
# diffrent styles
plt.style.available
```

[ ]:
```
car_sales["Price"].plot();
```

[ ]:
```
plt.style.use("seaborn-v0_8-whitegrid")
```

[ ]:
```
car_sales["Price"].plot();
```

[ ]:
```
plt.style.use("seaborn-v0_8")
```

[ ]:
```
car_sales["Price"].plot();
```

[ ]:
```
# Create some data
x = np.random.randn(10,4)
```

[ ]:
```
df = pd.DataFrame(x, columns=["a","b","c","d"])
df
```

[ ]:
```
ax = df.plot(kind = "bar")
ax
```

[ ]:
```
# Customize our plot with the set method

ax = df.plot(kind = "bar")
# Add some labels and title
ax.set(title="Random Number Bar Graph from DataFrame",
       xlabel  = "Row Number",
       ylabel = "Random Number")
# Make the legend visible
ax.legend().set_visible(True)
```

[ ]:
```
# subplot of chol ,age, thalach
fig , (ax0,ax1) = plt.subplots (ncols=1,
                                nrows =2,
                                figsize = (10,10))
# Add data to ax0
scatter = ax0.scatter( x = over_50["age"],
                       y = over_50["chol"]
                       , c = over_50["target"],
                      cmap ="winter") ## changes the colour sheme
```

```
# Customize
ax0.set(title = "Heart Disease and Cholestrol Levels",
       xlabel ="Age",
       ylabel = "Cholesterol");

# add a legend
ax0.legend(*scatter.legend_elements(), title = "Target");

#Add a h-line
ax0.axhline(over_50["chol"].mean(),
          linestyle = "--");



# Add data to ax1

scatter = ax1.scatter( x = over_50["age"],
                      y = over_50["thalach"]
                      , c = over_50["target"],
                     cmap ="winter")
# Customize
ax1.set(title = "Heart Disease and Max Heart Rate",
       xlabel ="Age",
       ylabel = "Max Heart Rate");

ax1.legend(*scatter.legend_elements(),title ="Target")

#Add a h-line
ax1.axhline(over_50["thalach"].mean(),
          linestyle = "--");

# add a title to the figure
fig.suptitle("Heart Disease Analysis", fontsize = 16, fontweight = "bold");
```

[ ]:
```
# Saving and sharing your Plot
fig.savefig("heart-disease-analysis-plot-saved-with-code.png")
```

[ ]:
```

```