Matplotlib

From matplotlib import pyplot as plt

If I want to plot a 2-d plot with x-axis and y-axis so we should use the plot function:

Plt.plot(x-axis,y-axis)

And to show this plot we should use:

Plt.show()

Plt.tilte(‘the title’) >> to give a title to your plot

Plt.xlable(‘x-axis name’) >> to give a name to x-axis

Plt.ylable(‘y-axis name’) >> to give a name to y-axis

Till now that will lead to a 2-d plot with a single line that represents our data.

So if we want to add more lines we could easily do that:

Plt.plot(x-axis,new y-axis)

To know information about each line we should use:

Plt.legend([‘fist line name’ , ‘second line name’]) ## note: as a list

Or we can do the same thing like this:

Plt.plot(x-axis,y-axis, label = ‘fist line name’)

Plt.plot(x-axis,new y-axis,lable=’ second line name’’)

We can also specify the color of the line and its style like:

Plt.plot(x-axis,y-axis,color='k' ,linestyle='_b_' , label = 'fist line name')

Plt.tight_layout() >> if I want to set the padding of lines.

Plt.grid(True) >> if I want a grid in the background.

Print(plt.style.available)>> to know the built_in styles in matplotlib

Now if I want to use one of those built_in styles:

Plt.style.use('pass any style I want')

So now I don't need to use any formatting like: ,color='k' ,linestyle='_b_'

Plt.xkcd() >> if I want my plot to appear in a weird style.

Now if I want to save that:

Plt.savefig('plot.png') >> it will save the figure in the current directory. Note : we use this method before plt.show().

Plt.bar is the same like plt.plot but it show bars and we can mix between them in the same figure (the bars and lines)

But if I want to show more than a bar in the same figure:

We have to import numpy as np and:

X_indexes = np.arange(len(x-axis))

Then we have to set the width of bars so they should not stick together.

Width = 0.25

```
ages_x = [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
dev_y = [38496, 42000, 46752, 49320, 53200,
        56000, 62316, 64928, 67317, 68748, 73752]
plt.plot(X_indexes-width, dev_y,width = width ,color="#444444", label="All Devs")
py_dev_y = [45372, 48876, 53850, 57287, 63016,
            65998, 70003, 70000, 71496, 75370, 83640]
 plt.plot(X_indexes, py_dev_y, ,width = width,color="#008fd5", label="Python")
 js_dev_y = [37810, 43515, 46823, 49293, 53437,
            56373, 62375, 66674, 68745, 68746, 74583]
 plt.plot(X_indexes+width, js_dev_y, ,width = width,color="#e5ae38", label="JavaScript")
```

know we will add a name to the new x-axis:

plt.xticks(ticks= X_indexes,labels=ages_x)

```
plt.style.use("fivethirtyeight")
```

to read a csv file:

```
data = pd.read_csv('data.csv')
ids = data['Responder_id'] >> the first column of the data
lang_responses = data['LanguagesWorkedWith'] >> the second one
language_counter = Counter() >> that’s basically a dictonary
for response in lang_responses:
    language_counter.update(response.split(';'))
languages = []
popularity = []
for item in language_counter.most_common(15):
    languages.append(item[0])
    popularity.append(item[1])

plt.barh(languages, popularity) >> to draw a horizontal bar
```

```
slices = [59219, 55466, 47544, 36443, 35917]
labels = ['JavaScript', 'HTML/CSS', 'SQL', 'Python', 'Java']
explode = [0, 0, 0, 0.1, 0]
```
>> **to explode a specific slice of my pie chart**

```
plt.pie(slices, labels=labels, explode=explode, shadow=True,
    startangle=90, autopct='%1.1f%%',
    wedgeprops={'edgecolor': 'black'})
```

slices >> the data I want to show

labels >> the labes of those data

explode >> to explode the slices I want

shadow >> to give a shadow to each slice

startangle >> to rotate the chart

autopct='%1.1f%%', >> to show the percentage of each slice

wedgeprops >> to give the edges a specific color

```
plt.title("My Awesome Pie Chart")
plt.tight_layout()
plt.show()
```