جزوه شماره ۱۱

MATPLOTLIB LIBRARY

ArcGIS Pro

محمدطاهر طاهرپور

دانشجوی ارشد مدیریت شهری

دانشگاه تهران

mttaherpoor
@mtaherpoor
09336144947

ناهید نعمتی کوتنائی (تیسا)

دکتری جغرافیا و برنامه‌ریزی شهری

مدرس دانشگاه

Dr.nemati.K
@Nemati_k
09112230798

پاییز ۱۴۰۲

فهرست مطالب:

Matplotlib Library ۴
Matplotlib چیه؟ ۴
چطوری از Matplotlib استفاده کنیم؟ ۴
کدهای این بخش در یک نگاه ۹
ماژول pylab ۱۱
Figure, Subplot and Axes ۱۲
Figure چیه؟ ۱۲
subplot چیه؟ ۱۴
Axes چیه؟ ۱۶
کدهای این بخش در یک نگاه ۱۷
سفارسی سازی شکل‌ها یا Figure Customization ۱۹
کدهای این بخش در یک نگاه ۲۳
سفارشی سازی گرافها در matplotlib ۲۵
کدهای این بخش در یک نگاه ۲۶
Grid, Spines, Ticks ۲۷
Ticksچیه؟ ۲۷
Spines چیه؟ ۲۸
Grid چیه؟ ۲۸
کدهای این بخش در یک نگاه ۲۹
خلاصه دستورهای Matplotlib.pyplot ۳۰
حل چندتا تمرین ۳۳
جواب تمرینها ۳۴
نمونه نمودار Line plot ۳۴
نمونه نمودار bar ۳۴
نمونه نمودار scatter ۳۵
نمونه نمودار stack ۳۶
نمونه نمودار pie ۳۷

نمونه نمودار Polar ۳۸
نمونه نمودار hist یا هیستوگرام ۳۹
نمونه نمودار box ۴۰
نمونه نمودار violin ترکیب هیستوگرام و box ۴۱
Heat map ۴۲
کدهای این بخش در یک نگاه ۴۳

Matplotlib Library

تو این جزوه یکی دیگه از مهمترین کتابخونه‌های پایتون به اسم matplotlib رو با هم یاد میگیریم. این کتابخونه هم مثل کتابخونه numpy و pandas واسه علم داده یا Data Science ضروری هست. مطالب این جزوه خلاصه مطالب آموزشی Udemy - Data Visualization with Python Masterclass - Python A-Z بخش مربوط به Fundamentals of Data Science و سایت https://matplotlib.org هست.

Matplotlib چیه؟

کتابخونه matplotlib یکی دیگه از کتابخونه‌های پایتون هست و ازش برای تصویرسازی داده یا data vitalization استفاده میشه. در واقع بهمون کمک میکنه که داده‌های پیچیده و پراکنده رو منظم کنیم. یه نمودار دو بعدی از داده‌ها بهمون میده که میتونیم بهش رنگ بدیم که تصویرسازی داده‌ها رو ساده‌تر کنیم.

یه زبان برنامه نویسی داریم به اسم MATLAB که میشه ازش واسه محاسبات آماری و ریاضی و گرافیکی استفاده کرد و توابع زیادی هم داره. برای اینکه بتونیم تو پایتون خروجیهای مشابه MATLAB داشته باشیم از pyplot استفاده میکنیم که کلی تابع داره و شبیه MATLAB عمل میکنه. در واقع باید با یه نقطه کلمه pyplot رو به matplotlib وصل کنیم و به عنوان plt بیاریمش تو نوت بوک ArcGIS Pro که بتونیم باهاش کار کنیم.

import matplotlib.pyplot as plt

جدول زیر انواع نمودارها، توابع تصویری، توابع محوری و توابع شکلی که میشه با این کتابخونه تولید کرد رو برات لیست کرده.

انواع نمودار Types of Plots	توابع شکلی Image Functions	توابع محوری Axis Function	توابع شکلی Figure Functions
bar boxplot hist pie plot scatter step	imread imsave imshow	axes text title xlabel ylabel xlim yscale	figtext figure show savefig close

چطوری از Matplotlib استفاده کنیم؟

باز هم میتونید تو نوت بوک ArcGIS Pro تمرینش کنید ولی یه جاهایی ممکنه نوت بوک ArcGIS Pro واسه نشون دادن خروجیها محدودیت داشته باشه و بهتون پیام خطا بده. برای همین طبق مطالبی که تو جزوه قبلی (جزوه ۱۰) گفتیم کدها رو تو Jupyter Notebook مینویسیم و بعد به پروژه‌مون اضافه میکنیم.

numPy و matplotlib رو با هم وارد نوت بوک میکنیم. یه عبارت جادویی دیگه هم باید بنویسیم: **%matplotlib inline** که یه تابع هست که باعث میشه نمودارهایی که تو نوت بوکمون ترسیم میکنیم همینجا بتونیم ببینیمشون و تو همین نوت بوک هم ذخیره شن و واسه دیدنشون نیازی به استفاده از **plt.show()** نداشته باشیم.

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

ولی این دســتور تو نوت بوک ArcGIS Pro واســه نشــون دادن نمودار فعال نیســت و فقط اطلاعات رو برامون ذخیره میکنه. یعنی وقتی کد رو اجرا می‌کنیم به جای نمودار بهمون متن نوشته میده و نمودار رو زیر کد بهمون نشون نمیده. واسه همین واسه دیدن نمودارها بعد از ترسیمشون باید **plt.show()** رو بنویسیم.

تمرین اول:

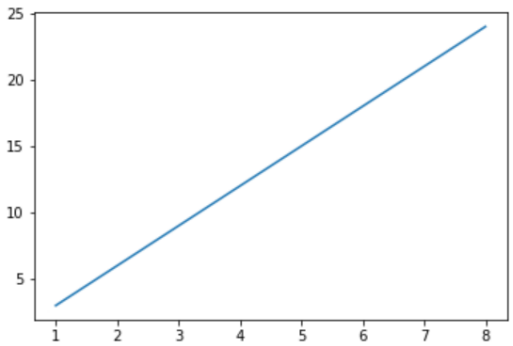
بعد از وارد کردن کتابخونه‌های مورد نیاز، یه آرایه به اسم N_A_X با نامپی می‌سازیم که از عدد ۱ تا ۸ رو بهمون بده و یه آرایه دیگه به اسم N_A_Y می‌ســـازیم که شـــامل اعداد ۳ تا ۲۶ با گام ۳ باشـــه. با np.arange() میسازیمشون.

```
N_A_X = np.arange(1,9)

N_A_X

array([1, 2, 3, 4, 5, 6, 7, 8])

N_A_Y = np.arange(3,27,3)

N_A_Y

array([ 3,  6,  9, 12, 15, 18, 21, 24])
```

حالا با تابع **plt.plot()** تبدیلشـون میکنیم به نمودار. آرایه اول و دوم با کاما از هم جدا میشن و میرن تو پرانتز.

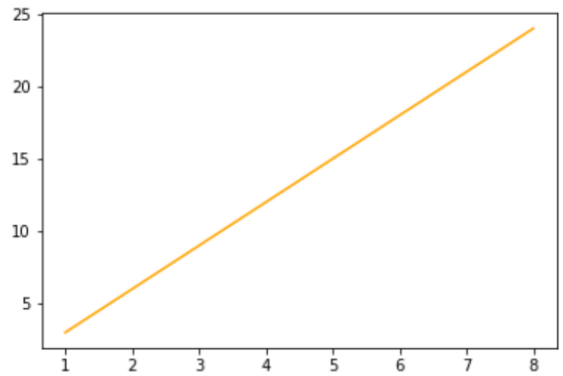
از plt.show() هم واسه دیدن نمودار زیر کد استفاده میکنیم.

```
plt.plot(N_A_X,N_A_Y)
plt.show()
```

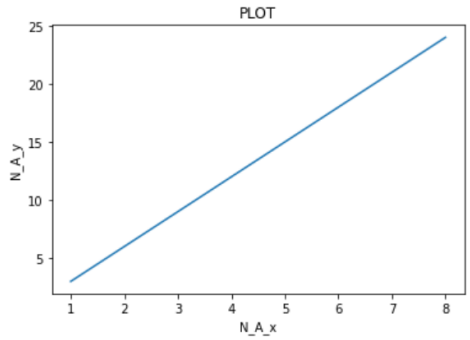


میتونیم با color رنگ گراف رو تغییر بدیم. رنگ رو باید داخل کوتیشـــن جلوی **color = " "** تو پرانتز plt.plot() بنویسیم. با **plt.xlabel()** و **plt.ylabel()** میشه برچسبهای محور افقی و عمودی رو مشخص کرد. با **plt.title()** میتونیم بهش عنوان بدیم.

```
plt.plot(N_A_X,N_A_Y, color = "orange")
plt.show()
```




```
plt.xlabel("N_A_x")
plt.ylabel("N_A_y")
plt.title("PLOT")
plt.plot(N_A_x,N_A_y)
plt.show()
```

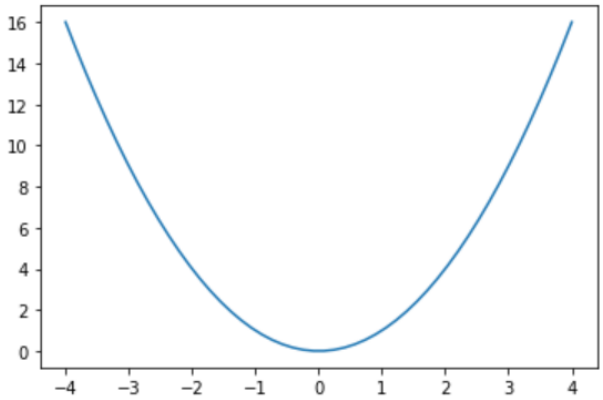


تمرین دوم:

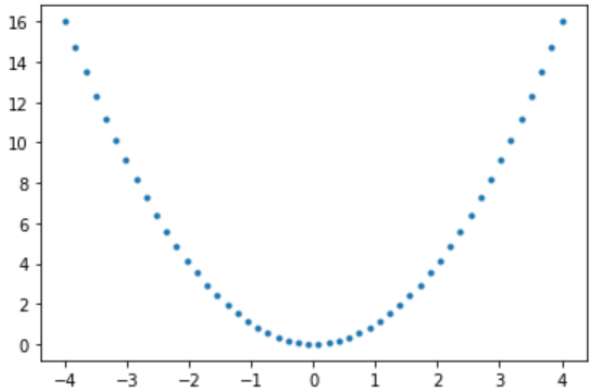
با تابع **np.linspace()** به همراه عدد اول و آخر آرایه به اســـم LS_X تولید می‌کنیم. اگه یادت باشـــه این تابع برعکس تابع **np.arange()** **عدد آخر رو هم نشون میده.** گام هم میتونه داشته باشه. یه آرایه دیگه هم به اسم LS_Y درست میکنیم که تابع اول رو به توان ۲ برسونه و بعد تبدیلشون میکنیم به نمودار و نمایشش میدیم.

اگه بخوایم خط محور رو به صورت نقطه چین ببینیم باید بعد از نوشتن آرایه‌ها داخل پرانتز plt.plot() عبارت "." رو وارد کنیم.

```
LS_X = np.linspace(-4,4,50)

LS_Y = LS_X ** 2

plt.plot(LS_X,LS_Y, ".")
plt.show()
```

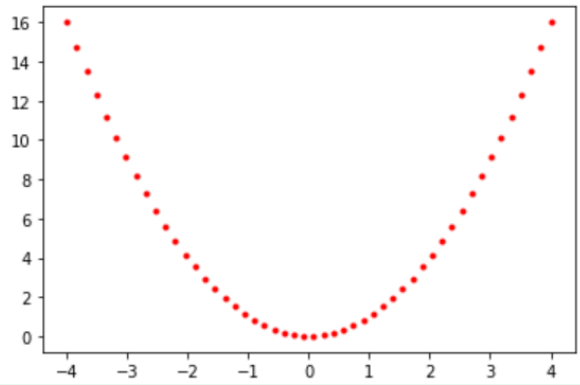


```
LS_X = np.linspace(-4,4,50)

LS_Y = LS_X ** 2

plt.plot(LS_X,LS_Y)
plt.show()
```



واسه اینکه نقطه‌چینها قرمز شن باید "r." رو وارد کنیم.

```
LS_X = np.linspace(-4,4,50)

LS_Y = LS_X ** 2

plt.plot(LS_X,LS_Y, "r.")
plt.show()
```



از این لینک میتونید ترسیمات مختلف برای **نقطه و خط و رنگ** روی نمودار رو از سایت اصلی matplotlib ببینید:

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html

character	description
'.'	point marker
','	pixel marker
'o'	circle marker
'v'	triangle_down marker
'^'	triangle_up marker
'<'	triangle_left marker
'>'	triangle_right marker
'1'	tri_down marker
'2'	tri_up marker
'3'	tri_left marker
'4'	tri_right marker
'8'	octagon marker
's'	square marker
'p'	pentagon marker
'P'	plus (filled) marker
'*'	star marker
'h'	hexagon1 marker
'H'	hexagon2 marker
'+'	plus marker
'x'	x marker
'X'	x (filled) marker
'D'	diamond marker
'd'	thin_diamond marker
'|'	vline marker
'_'	hline marker

Line Styles

character	description
'-'	solid line style
'--'	dashed line style
'-.'	dash-dot line style
':'	dotted line style

Example format strings:

```
'b'    # blue markers with default shape
'or'   # red circles
'-g'   # green solid line
'--'   # dashed line with default color
'^k:'  # black triangle_up markers connected by a dotted line
```

Colors

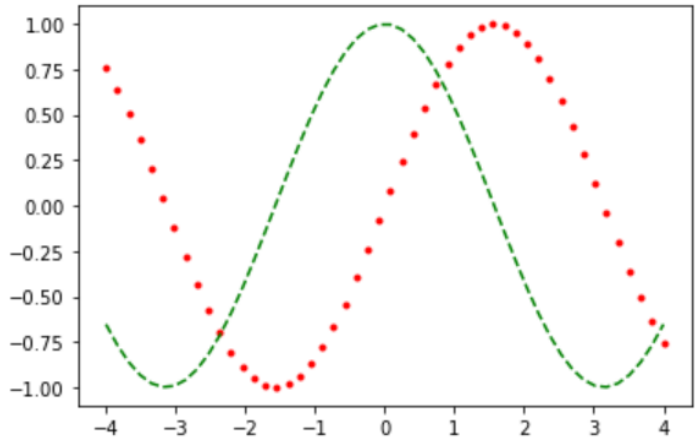
The supported color abbreviations are the single letter codes

character	color
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

تمرین سوم:

برای ترسیم چندتا نمودار با هم برای مثال برای ترسیم سینوس و کسینوس نمودارها باید از متد sin() و cos() استفاده کنیم. تو پرانتز plt.plot() دستور np.sin() رو برای ترسیم کسینوس آرایه LS_X با نقطه چین قرمز یعنی "r." و همینطور دستور np.cos() رو برای ترسیم کسینوس برای آرایه LS_X با دستور np.cos() به صورت خط چین سبز با "g-" بهش بده. در واقع np.sin() و np.cos() به جای محور y تو پرانتز نوشته میشن.

```
plt.plot(LS_X,np.sin(LS_X), "r.")
plt.plot(LS_X,np.cos(LS_X), "g--")
plt.show()
```



تمرین چهارم:

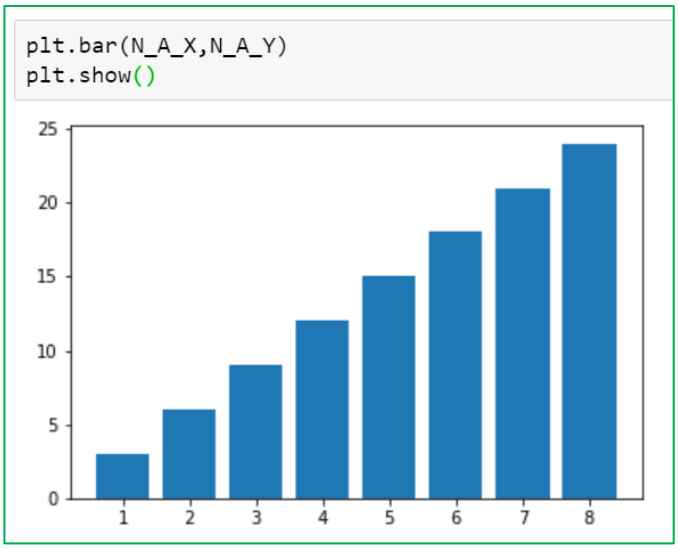
یه آرایه تک بعدی با **np.arange()** از عدد ۱ تا ۸ میســـازیم.

یه آرایه دیگه هم از اعداد ۳ تا ۲۶ با گام ۳ میسازیم.

```
N_A_X = np.arange(1,9)

N_A_X

array([1, 2, 3, 4, 5, 6, 7, 8])

N_A_Y = np.arange(3,27,3)

N_A_Y

array([ 3,  6,  9, 12, 15, 18, 21, 24])
```

بعد میبریمشـــون رو نمودار bar اینکار با **plt.bar()** انجام میشـــه. نمودار bar یه نمودار ســـتونی هســـت که ستونها با فاصله یکنواختی از هم قرار میگیرن. ازش واسه مقایسه مقادیر گروه‌های مختلف استفاده میشه.

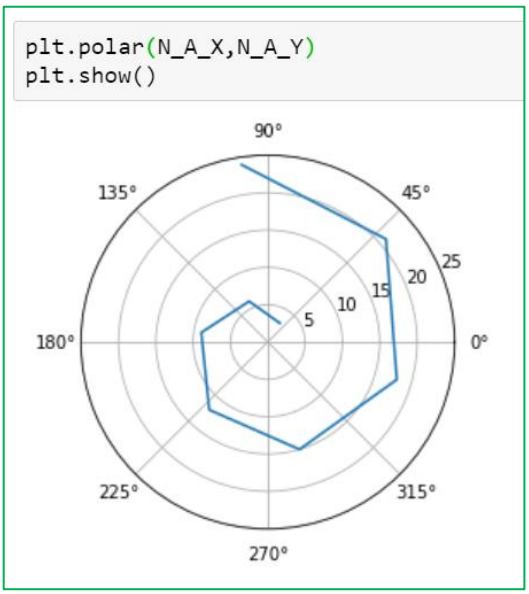
```
plt.bar(N_A_X,N_A_Y)
plt.show()
```



تمرین پنجم:

واسه ترسیم نمودار Polar باید بنویسیم: **plt.polar()** . نمودار قطبی واسه نشون دادن رابطه بین دو یا چند متغیر استفاده میشه.

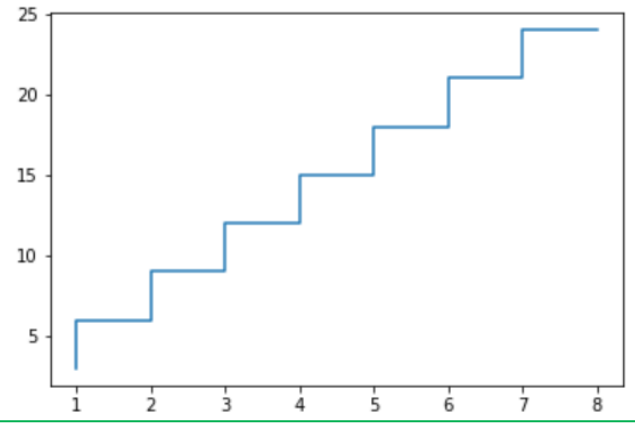
```
plt.polar(N_A_X,N_A_Y)
plt.show()
```



تمرین ششم:

واســـه ترســـیم نمودار پله‌ای بـایـد بنویســـیم:
plt.step() برای داده‌هایی با ویژگی های گســـســـته یا گام مانند استفاده میشه

```
plt.step(N_A_X,N_A_Y)
plt.show()
```

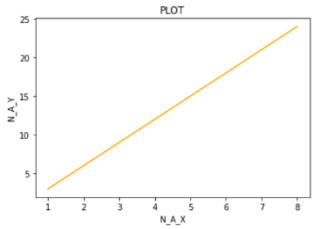


کدهای این بخش در یک نگاه

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

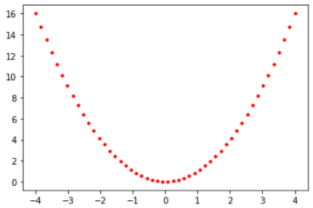
تمرین #۱

```
N_A_X = np.arange(1,9)
N_A_Y = np.arange(3,27,3)
plt.xlabel("N_A_X")
plt.ylabel("N_A_Y")
plt.title("PLOT")
plt.plot(N_A_X, N_A_Y, color = "orange")
plt.show()
```



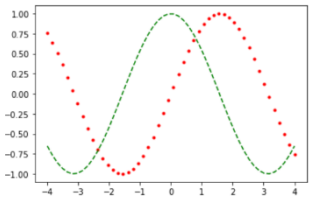
تمرین#۲

```
LS_X = np.linspace(-4,4,50)
LS_Y = LS_X ** 2
plt.plot(LS_X, LS_Y, "r.")
plt.show()
```



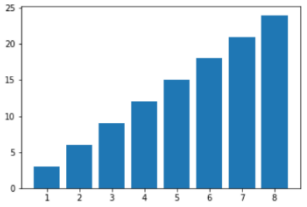
تمرین#۳

```
plt.plot(LS_X, np.sin(LS_X), "r.")
plt.plot(LS_X, np.cos(LS_X), "g--")
plt.show()
```



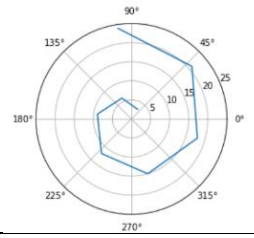
تمرین#۴

```
N_A_X = np.arange(1,9)
N_A_Y = np.arange(3,27,3)
plt.bar(N_A_X, N_A_Y)
plt.show()
```



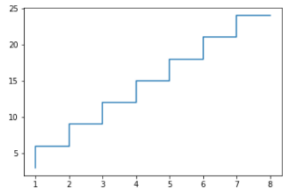
تمرین#۵

```
plt.polar(N_A_X, N_A_Y)
plt.show()
```



تمرین#۶

```
plt.step(N_A_X, N_A_Y)
plt.show()
```



ماژول pylab

ماژول pylab یه واسط رویه‌ای یا procedural interface برای matplotlib هست و کنارش نصب میشه. در واقع یه ماژول راحت هست که myplotlib.pyplot (**واسه ترسیم نمودار استفاده میشه**) رو با numpy (**برای محاسبات ریاضی و کار با ارایه‌ها استفاده میشه**) با یه نام واحد وارد میکنه.

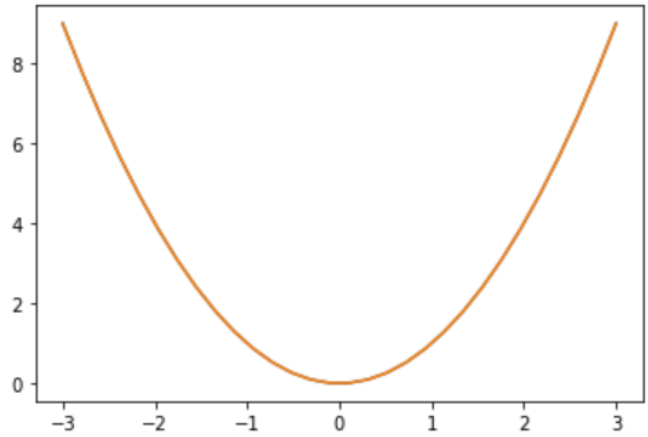
یادت هست که ماژولها رو چطوری وارد میکنیم؟ از دستور from به همراه اسم ماژول و عبارت import استفاده میکنیم.

```
from pylab import *
```

```
X = linspace(-3,3,30)
```

```
Y = X ** 2
```

```
plot(X,Y)
plt.show()
```



کار با pylab تو پایتون خیلی توصیه نمیشه.

Figure, Subplot and Axes

Figure چیه؟

figure یه نگهدارنده ماژول تو کلاس figure در matplotlib هست. همه آبجکتهایی که معرف متن و برچسب هستن رو نگه میداره. واسه صدا زدنش باید از **plt.figure()** استفاده کنیم.

تمرین اول:

یه فیلگور با plt.figure() به اسم Figure تعریف میکینم و یه آرایه با np.arange() به اسم X پایینش میسازیم که از عدد ۱- تا ۵ رو با گام ۰/۱ بهمون خروجی بده.

```
Fiqure = plt.figure()
```

```
X = np.arange(-1,5,0.1)
```

```
X
```

```
array([-1.00000000e+00, -9.00000000e-01, -8.00000000e-01, -7.00000000e-01,
       -6.00000000e-01, -5.00000000e-01, -4.00000000e-01, -3.00000000e-01,
       -2.00000000e-01, -1.00000000e-01, -2.22044605e-16,  1.00000000e-01,
        2.00000000e-01,  3.00000000e-01,  4.00000000e-01,  5.00000000e-01,
        6.00000000e-01,  7.00000000e-01,  8.00000000e-01,  9.00000000e-01,
        1.00000000e+00,  1.10000000e+00,  1.20000000e+00,  1.30000000e+00,
        1.40000000e+00,  1.50000000e+00,  1.60000000e+00,  1.70000000e+00,
        1.80000000e+00,  1.90000000e+00,  2.00000000e+00,  2.10000000e+00,
        2.20000000e+00,  2.30000000e+00,  2.40000000e+00,  2.50000000e+00,
        2.60000000e+00,  2.70000000e+00,  2.80000000e+00,  2.90000000e+00,
        3.00000000e+00,  3.10000000e+00,  3.20000000e+00,  3.30000000e+00,
        3.40000000e+00,  3.50000000e+00,  3.60000000e+00,  3.70000000e+00,
        3.80000000e+00,  3.90000000e+00,  4.00000000e+00,  4.10000000e+00,
        4.20000000e+00,  4.30000000e+00,  4.40000000e+00,  4.50000000e+00,
        4.60000000e+00,  4.70000000e+00,  4.80000000e+00,  4.90000000e+00])
```

حالا یه متغیر به اسم Y میسازیم. میخوایم یه تابع نمایی باشه. **exponential یا مقدار نمایی رو با np.exp() مینویسیم که یه آرگومان واحد میگیره و مقدارش رو به صورت عدد اعشاری برمیگردونه.**

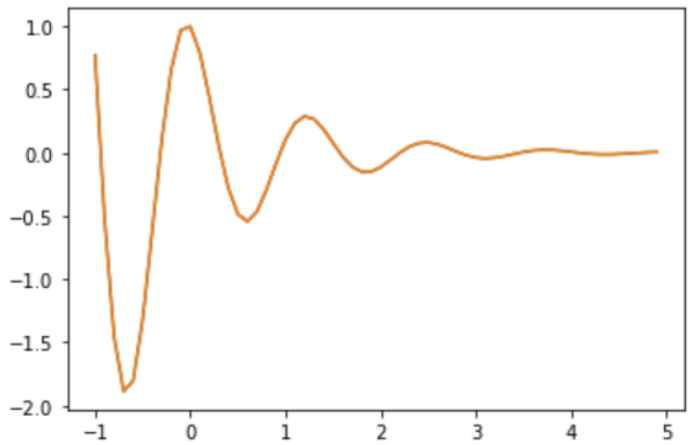
از آرایه X به عنوان آرگومان استفاده میکنیم ولی اعدادش در منفی باید ضرب بشن.

بعد ضربش میکنیم در کسینوس آرایه X که اعدادش در ۵ ضرب شدن.

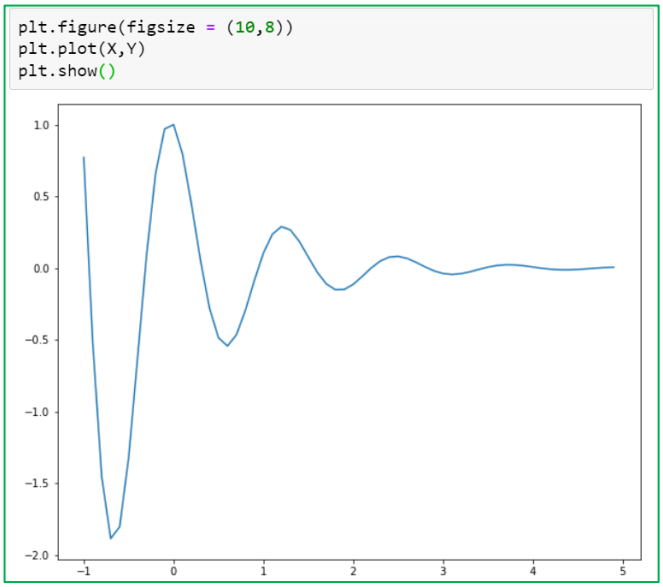
حالا با plt.plot(X,Y) تبدیش میکنیم به نمودار و با plt.show() نمایشش میدیم.

```
Y = np.exp(-X)*np.cos(5*X)
```

```
plt.plot(X,Y)
plt.show()
```

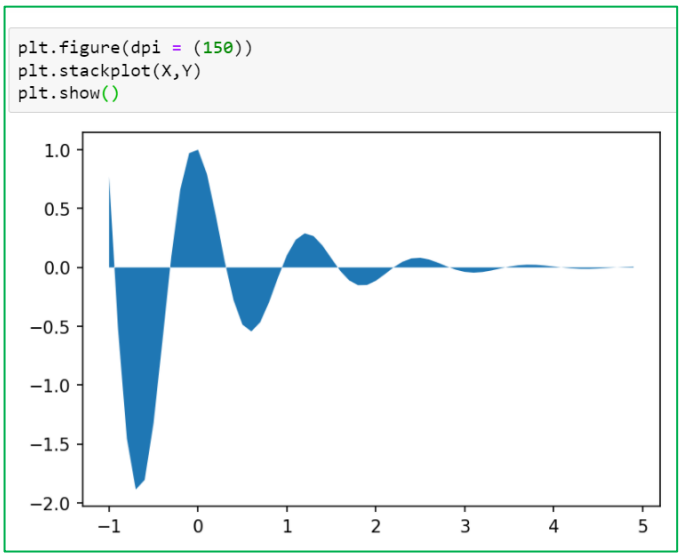


```
plt.figure(figsize = (10,8))
plt.plot(X,Y)
plt.show()
```



حالا میشـــه از figure اســـتفاده کرد. اندازه فیگور رو هم با **figsize = (,)** تغییر میدیم که نوع داده‌اش tuple هست،

```
plt.figure(dpi = (150))
plt.stackplot(X,Y)
plt.show()
```



تمرین دوم:

اندازه تصویر بزرگ هست برای تغییر اندازه باید **dpi = ()** بهش بدیم. میتونیم همزمان یه نوع دیگه نمودار هم براش تعریف کنیکم. مثـل **plt.stackplot()**. بعـد ازش میخوایم که نمایشش بده.

```
plt.figure(facecolor = 'red')
plt.plot(X,Y)
plt.show()
```

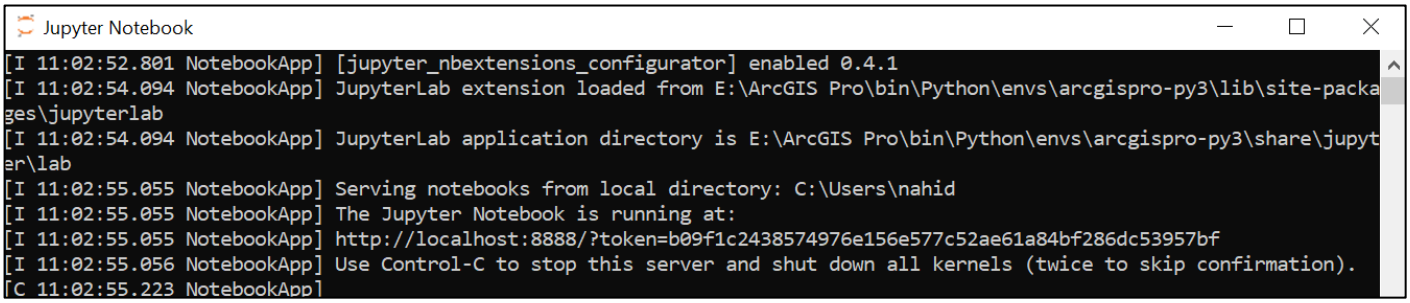
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
In [56]:
Line 3:     plt.show()

File E:\ArcGIS Pro\bin\Python\envs\gisenv\Lib\site-packages\matplotlib\pyplot.py, in show:
Line 378:   return _backend_mod.show(*args, **kwargs)

File E:\ArcGIS Pro\bin\Python\envs\gisenv\Lib\site-packages\ipykernel\pylab\backend_inline.py, in show:
Line 41:    display(

TypeError: 'NoneType' object is not iterable
---------------------------------------------------------------------------
```

تمرین سوم:

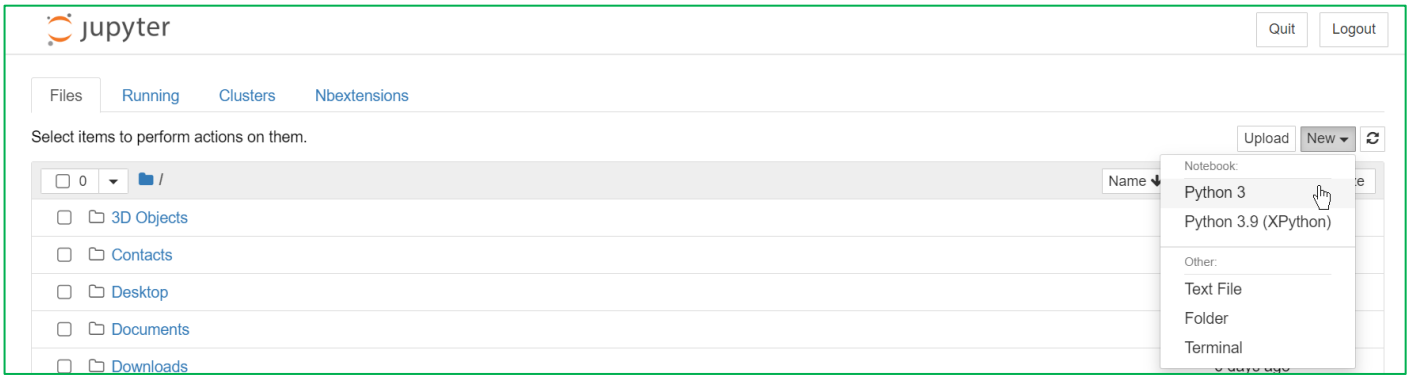
واســـه رنـگ دادن بـه بـک گرانـد نمودار از **facecolor = " "** استفاده میکنیم. *ولی این دستور تو نوت بوک ArcGIS Pro ورژن 3.0.0 کـار نمیکنـه و بهمون خطا برمیگردونه.*

بریم تو Jupyter Notebook بنویسـیمش ببینم درسـت میشـه. طبق جزوه ۱۰ حالت local ژوپیتر نوت بوک رو باز کن. و کد مقابل رو به همراه کتابخونه‌ها و آرایه‌های مورد نیازش تو نوت بوک بنویس و اجرا کن.

یادآوری: Jupyter Notebook رو توبخش Start Program ســـرچ کن و پنجره‌اش رو باز کن ولی هیچ کاری انجام نده. صبر کن تا صفحه Jupyter تو بروزر برات باز شه.



تو پنجره باز شده برو روی New و Python3 رو انتخاب کن که یه نوت بوک تو صفحه جدید برات باز کنه.

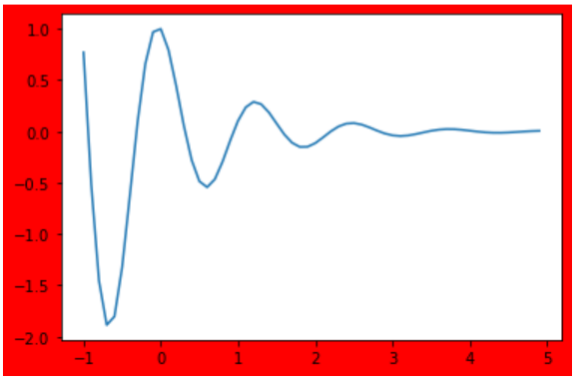


بعد کدت رو بنویس و اجرا کن.

میبینی که بدون پیام خطا کد رو برات اجرا میکنه و پشت نمودار رنگ قرمز میندازه. میتونی این نوت بوک رو ذخیره کن و بیاریش تو پروژه کریت تو ArcGIS Pro. (جزوه ۱۰ رو ببین)

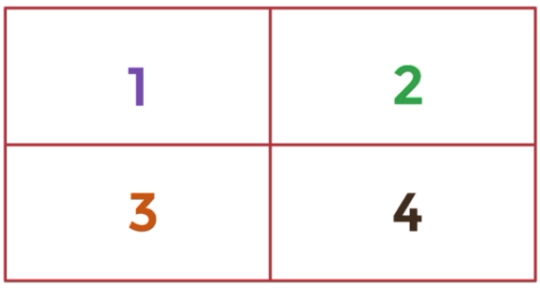
```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
Figure = plt.figure(facecolor = 'red')
X = np.arange(-1,5,0.1)
Y = np.exp(-X)*np.cos(5*X)
plt.plot(X,Y)
plt.show()
```



subplot چیه؟

وقتی با matplotlib کار میکنیم ممکنه گاهی وقتها بخوایم نمودارهای مشابه همزمان بسازیم. اینکار با تابع **plt.subplot()** انجام میشه و ترتیب قرارگیری نمودارها به شکل زیر می‌آد.



تمرین چهارم:

دو تا آرایه X و Y تعریف می‌کنیم. ازش یه پلات میگیریم.

```
X1 = np.arange(1,9)
```

```
X1
```

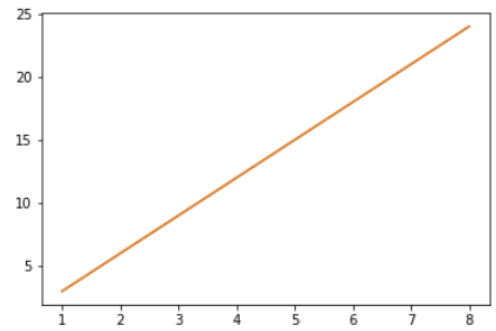
```
array([1, 2, 3, 4, 5, 6, 7, 8])
```

```
Y1 = np.arange(3,27,3)
```

```
Y1
```

```
array([ 3,  6,  9, 12, 15, 18, 21, 24])
```

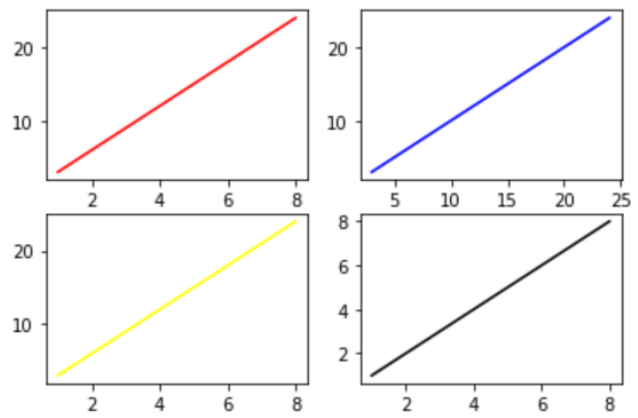
```
plt.plot(X1,Y1)
plt.show()
```



حالا چندتا subplot براش تعریف میکنیم. دو تا عدد اول داخل پرانتز به **سطر و ستون** اشاره داره و عدد سوم **شماره نمودار** رو مشخص میکنه. نمودار اول با ۱ مشخص میشه دومی با ۲ و به همین ترتیب.

رنگهای متفاوت هم میتونیم بهشون بدیم. میشه جای X و Y رو هم عوض کرد که اعداد سطر رو ستون متفاوت شن.

```
plt.subplot(2,2,1)
plt.plot(X1,Y1,"red")
plt.subplot(2,2,2)
plt.plot(Y1,Y1,"blue")
plt.subplot(2,2,3)
plt.plot(X1,Y1,"yellow")
plt.subplot(2,2,4)
plt.plot(X1,X1,"black")
plt.show()
```

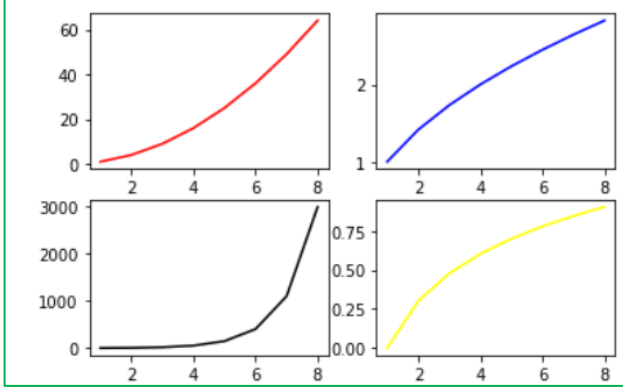


تمرین پنجم:

به جای نوشتن Y1 برای X1 یه سری عملیات ریاضی مثل تصویر مقابل مینویسیم که نمودارهای متفاوتی داشته باشیم.

- تو اولی X1 رو هم خودش ضرب کردیم.
- برای دومی ریشه دوم ازش گرفتیم.
- برای سومی تابع نمایی ساختیم
- برای چهارمی لگاریتم گرفتیم.

```
plt.subplot(2,2,1)
plt.plot(X1,X1*X1,"red")
plt.subplot(2,2,2)
plt.plot(X1,np.sqrt(X1),"blue")
plt.subplot(2,2,3)
plt.plot(X1,np.exp(X1),"black")
plt.subplot(2,2,4)
plt.plot(X1,np.log10(X1),"yellow")
plt.show()
```

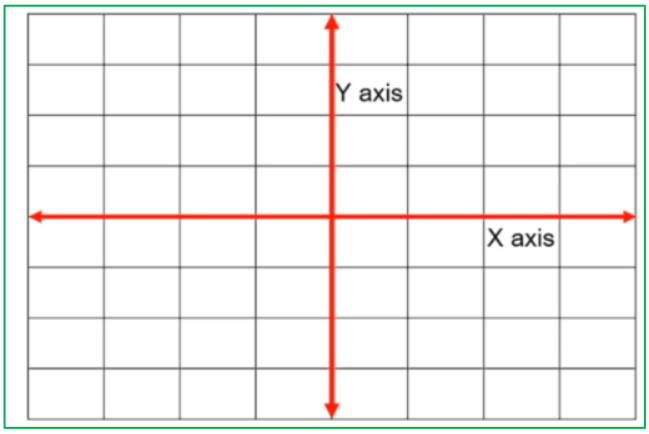


Axes چیه؟

به محورها X و Y مثل شکل مقابل میگن axes. کلاس axes تو matplotlib شامل اغلب عناصر شکلی یا figure و sets هســـت. object مربوط به axes در واقع منطقه یا موقعیت تصویر هست.

هر figure میتونه شـامل کلی محور یا axes باشـه ولی هر آبجکت axis فقط یه figure داره.

یه متد داریم که میشـــه باهاش محور رو به تصـــویر اضافه کرد و شامل **add_axes()**. میشه.



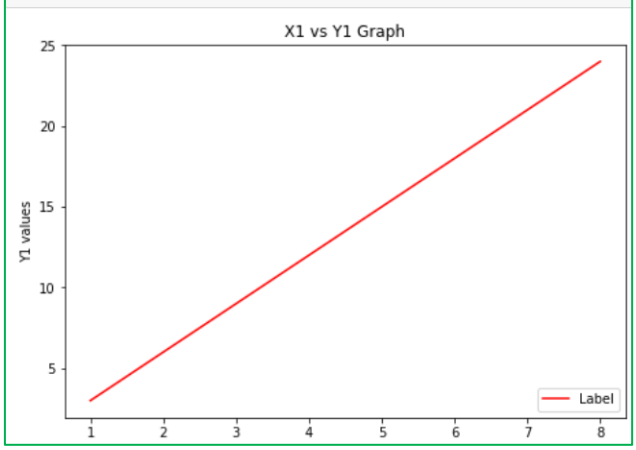
تمرین ششم:

اول بایـد آبجکت figure رو تعریف کنیم. بعـد محور رو به فیگور اضافه میکنیم. بهش اعداد شروع و پایان هر نمودار رو میدیم. اول دو تا شـروع یعنی ۰ و ۰ و بعد دو تا پایان یعنی ۱ و ۱.

برای محورهای x و y برچســب میگذاریم با set_xlable. و se_ylable. و بهش عنوان با set.title() و راهنما با legend().

میدیم. loc = "" مشـــخص میکنه که راهنما کجا گذاشـــته نوشته شه.

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.set_xlabel("X1 values")
Axes.set_ylabel("Y1 values")
Axes.set_title("X1 vs Y1 Graph")
Axes.plot(X1,Y1,"red")
Axes.legend(labels = ["Label"], loc = "lower right")
plt.show()
```



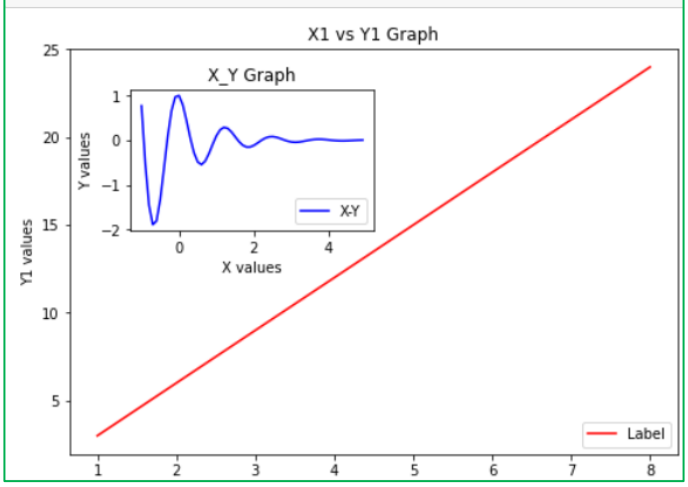
تمرین هفتم:

میخوایم یه پلات چندگانه یا توی هم درست کنیم.

یه محور دیگه میســازیم که به Figure قبلی اضـافه شــه. میتونیم بهش برچســب و راهنما و ... هم مثل اولی اضافه کنیم.

ازش پلات میسازیم و نمایشش میدیم.

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.set_xlabel("X1 values")
Axes.set_ylabel("Y1 values")
Axes.set_title("X1 vs Y1 Graph")
Axes.plot(X1,Y1,"red")
Axes.legend(labels = ["Label"], loc = "lower right")

Axes_2 = Figure.add_axes([0.1,0.55,0.4,0.35])
Axes_2.set_xlabel("X values")
Axes_2.set_ylabel("Y values")
Axes_2.set_title("X_Y Graph")
Axes_2.plot(X,Y,"blue")
Axes_2.legend(labels = ["X-Y"], loc = "lower right")
plt.show()
```

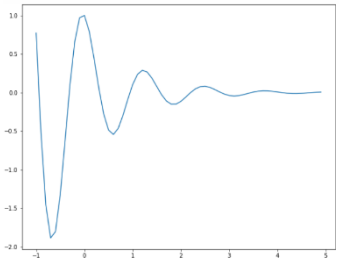


کدهای این بخش در یک نگاه

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

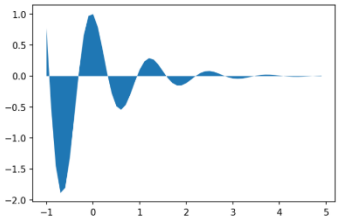
تمرین #۱

```
Figure = plt.figure(figsize = (10,8))
X = np.arange(-1,5,0.1)
Y = np.exp(-X)*np.cos(5*X)
plt.plot(X,Y)
plt.show()
```



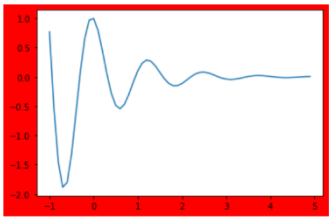
تمرین #۲

```
plt.figure(dpi = (150))
plt.stackplot(X,Y)
plt.show()
```



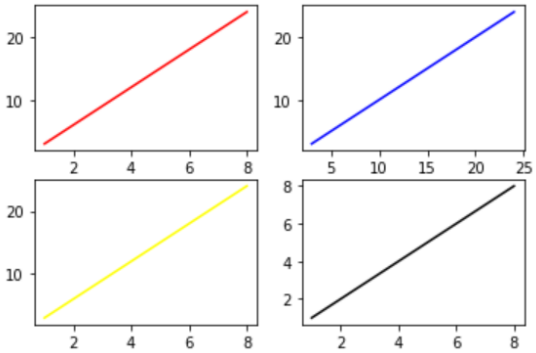
تمرین #۳: اجرا شده در نوت بوک ژوپیتر

```
Figure = plt.figure(facecolor = 'red')
X = np.arange(-1,5,0.1)
Y = np.exp(-X)*np.cos(5*X)
plt.plot(X,Y)
plt.show()
```



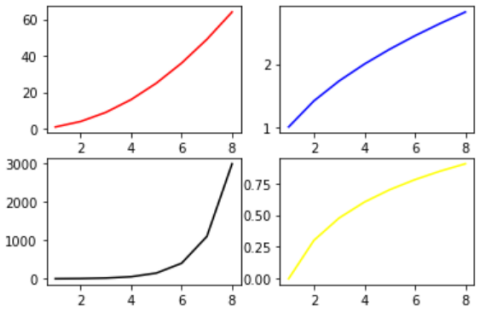
تمرین #۴

```
X1 = np.arange(1,9)
Y1 = np.arange(3,27,3)
plt.subplot(2,2,1)
plt.plot(X1,Y1,"red")
plt.subplot(2,2,2)
plt.plot(Y1,Y1,"blue")
plt.subplot(2,2,3)
plt.plot(X1,Y1,"yellow")
plt.subplot(2,2,4)
plt.plot(X1,X1,"black")
plt.show()
```



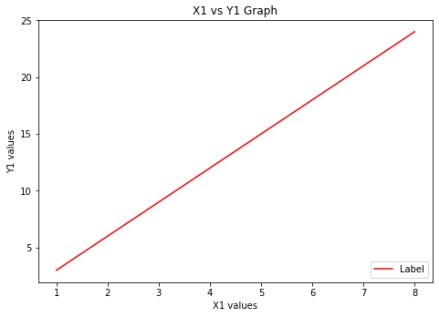
تمرین #۵

```
plt.subplot(2,2,1)
plt.plot(X1,X1*X1,"red")
plt.subplot(2,2,2)
plt.plot(X1,np.sqrt(X1),"blue")
plt.subplot(2,2,3)
plt.plot(X1,np.exp(X1),"black")
plt.subplot(2,2,4)
plt.plot(X1,np.log10(X1),"yellow")
plt.show()
```



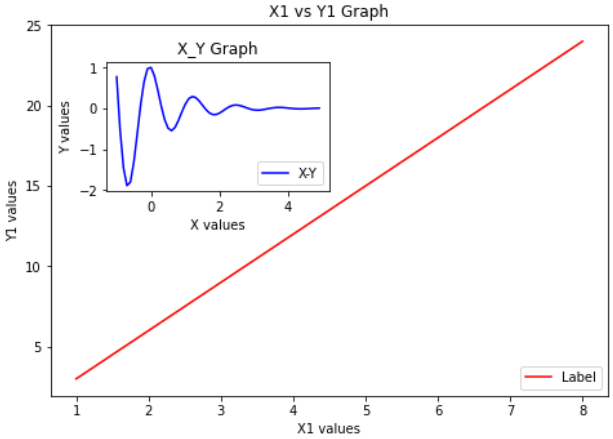
تمرین #۶

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.set_xlabel("X1 values")
Axes.set_ylabel("Y1 values")
Axes.set_title("X1 vs Y1 Graph")
Axes.plot(X1,Y1,"red")
Axes.legend(labels = ["Label"], loc = "lower right")
plt.show()
```



تمرین #۷

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.set_xlabel("X1 values")
Axes.set_ylabel("Y1 values")
Axes.set_title("X1 vs Y1 Graph")
Axes.plot(X1,Y1,"red")
Axes.legend(labels = ["Label"], loc = "lower right")
Axes_2 = Figure.add_axes([0.1,0.55,0.4,0.35])
Axes_2.set_xlabel("X values")
Axes_2.set_ylabel("Y values")
Axes_2.set_title("X_Y Graph")
Axes_2.plot(X,Y,"blue")
Axes_2.legend(labels = ["X-Y"], loc = "lower right")
plt.show()
```



سفارسی سازی شکل‌ها یا Figure Customization

میخوایم فیگورها یا شکلهای سفارشی شده بسازیم.

تمرین اول:

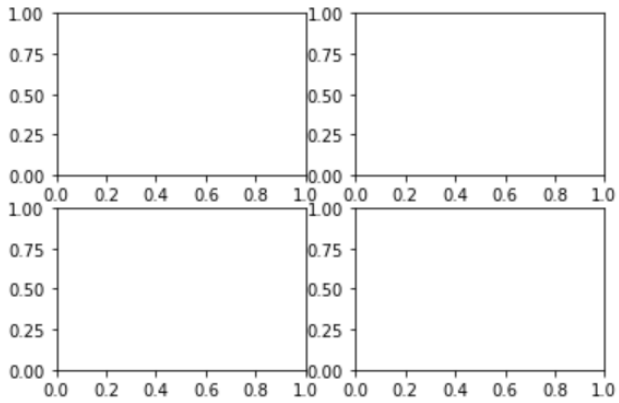
نامپی، مت پلات لیب و عبارت جادویی رو تو خط اول وارد میکنیم.

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

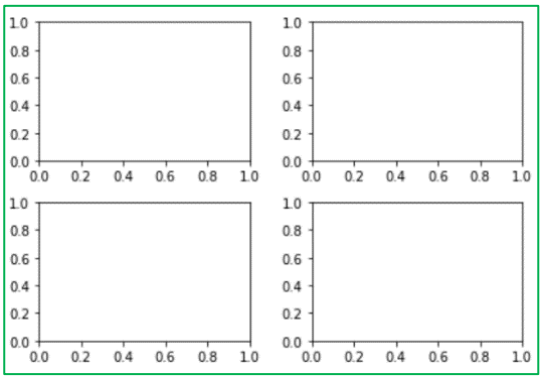
یه آرایه بین ۱ تا ۹ و یکی دیگه بین ۳ و ۲۷ با گام ۳ با arange() درســت میکنیم. واســه ســاخت نمودارهای مشــابه همزمان از subplot() اســتفاده میکنیم که اینجا باهاش یه Figure و ۴ تا Axes رو میســازیم. پس دو تا سطر و دو تا ستون بهش میدیم.

```
X = np.arange(1,9)
Y = np.arange(3,27,3)
```

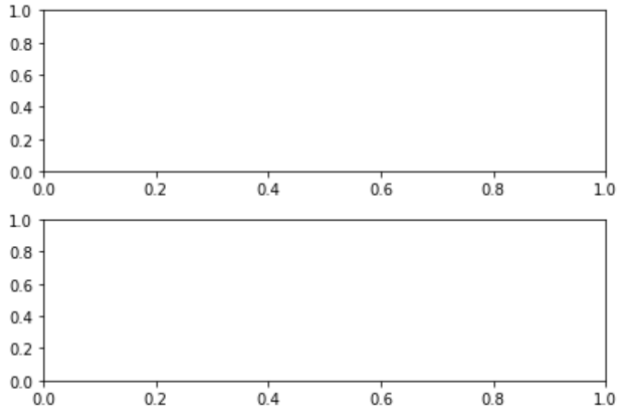
```
Figure, Axes = plt.subplots(nrows=2,ncols=2)
plt.show()
```



این محورها خیلی به هم نزدیک هســتن و به هم ریخته شدن. با تابع **plt.tight_layout()** میشه مرتبشون کرد.



```
Figure, Axes = plt.subplots(nrows=2,ncols=1)
plt.tight_layout()
plt.show()
```



اگه تعداد ستونها رو ۱ بدیم به شکل زیر درمی‌آن.

نوع داده محور رو هم میشه با دستور **type()** دید. میتونیم اطلاعات اولین و دومین محور رو پرینت کنیم.

```
print(type(Axes))
```

```
<class 'numpy.ndarray'>
```

```
print(Axes[0])
```

```
AxesSubplot(0.0929977,0.577778;0.863484x0.370833)
```

```
print(Axes[1])
```

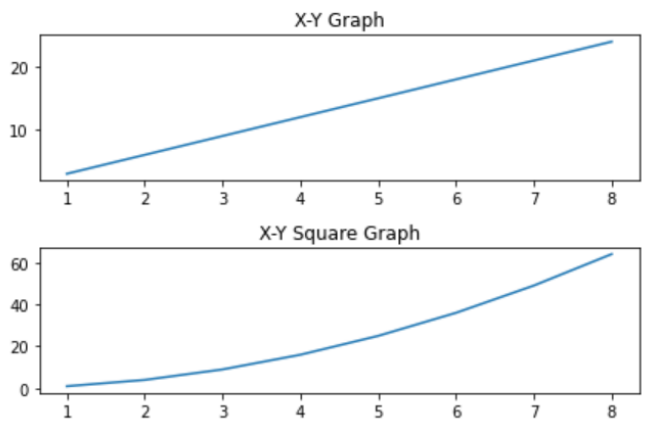
```
AxesSubplot(0.0929977,0.0965278;0.863484x0.370833)
```

```
for ax in Axes:
    print(ax.plot(X,Y))
```

```
[<matplotlib.lines.Line2D object at 0x000001F44FE91490>]
[<matplotlib.lines.Line2D object at 0x000001F45037D9D0>]
```

کدهای بالا به شکل یکپارچه به صورت زیر در می‌آد:

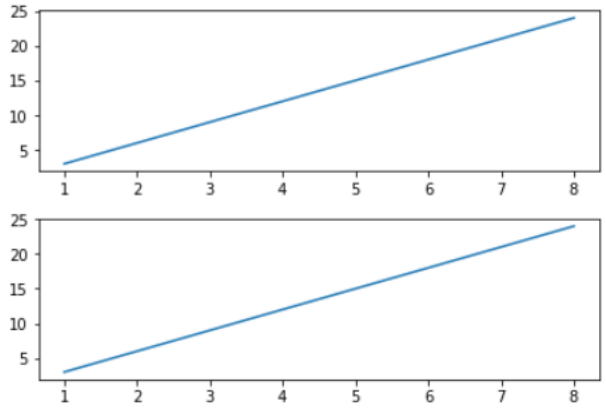
```
Figure, Axes = plt.subplots(nrows=2,ncols=1)
Axes[0].plot(X,Y)
Axes[0].set_title("X-Y Graph")
Axes[1].plot(X,X**2)
Axes[1].set_title("X-Y Square Graph")
plt.tight_layout()
plt.show()
```



میتونیم محور رو با حلقه for تکرارش کنیم.

```
X = np.arange(1,9)
Y = np.arange(3,27,3)
Figure, Axes = plt.subplots(nrows=2, ncols=1)
plt.tight_layout()
print(type(Axes))
print(Axes[0])
print(Axes[1])
for ax in Axes:
    print(ax.plot(X,Y))
plt.show()
```

```
<class 'numpy.ndarray'>
AxesSubplot(0.0929977,0.577778;0.863484x0.370833)
AxesSubplot(0.0929977,0.0965278;0.863484x0.370833)
[<matplotlib.lines.Line2D object at 0x000001E0C7E94820>]
[<matplotlib.lines.Line2D object at 0x000001E0C7E79640>]
```



تمرین دوم:

میتونیم پلاتهای متفاوتی رو از هر محور بســـازیم.

برای اینکار ایندکس محور رو باید با [] مشخص کنیم.

به جای محور y هم میشـــه یه ســـری عملیات روی محور x انجام داد.

میتونیم براشـــون عنوان هم با **.set_title()** تعریف کنیم.

تمرین سوم:

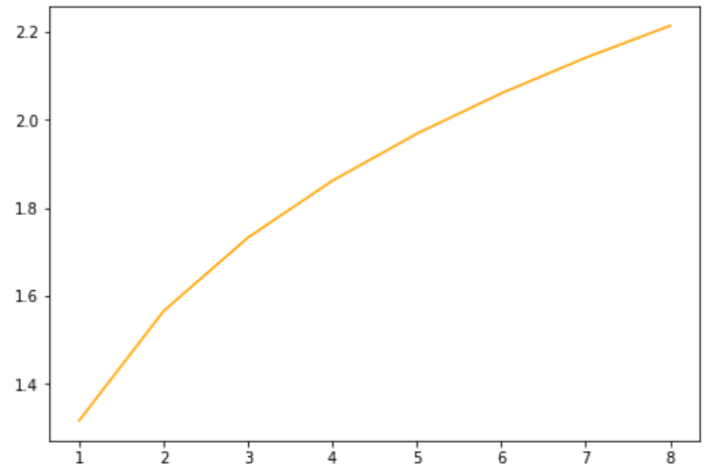
گاهی نیاز داریم که Figureهایی با اندازه‌های مختلف بســـازیم. یه Figure و یه Axes جدید تعریف میکنیم. با add_axes(). محور رو به فیگور اضافه میکنیم. بعد ازش پلات میگیریم.

رنگ هم میتونیم بهش بدیم.

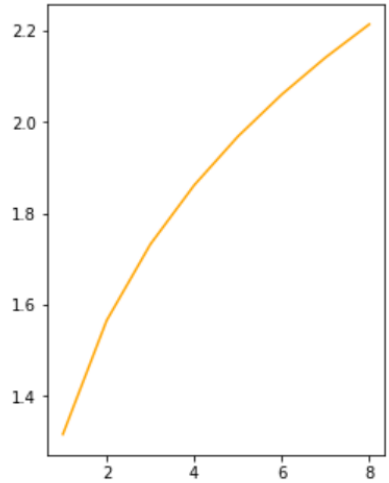
سایزش رو هم میتونیم با figsize() عوض کنیم. میتونیم ابعاد ۶ و ۴ بهش بدیم و ازش پلات بگیریم.

میتونیم یکبار دیگه اعداد ۳و ۴ بهش بدیم و نتیجه رو مقایسه کنیم.

```
Figure_S = plt.figure(figsize=(6,4))
Axes_S = Figure_S.add_axes([0,0,1,1])
Axes_S.plot(X,Y**0.25, color="orange")
plt.show()
```




```
Figure_S = plt.figure(figsize=(3,4))
Axes_S = Figure_S.add_axes([0,0,1,1])
Axes_S.plot(X,Y**0.25, color="orange")
plt.show()
```




```
Figure_S = plt.figure(figsize=(3,4),facecolor="red")
Axes_S = Figure_S.add_axes([0,0,1,1])
Axes_S.plot(X,Y**0.25, color="orange")
plt.show()
```

```
---------------------------------------------------------------
----------
TypeError                                 Traceback (most recent
call last)
In  [32]:
Line 4:     plt.show()

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-package
s\matplotlib\pyplot.py, in show:
Line 378:   return _backend_mod.show(*args, **kwargs)

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-package
s\ipykernel\pylab\backend_inline.py, in show:
Line 41:    display(

TypeError: 'NoneType' object is not iterable
---------------------------------------------------------------
```

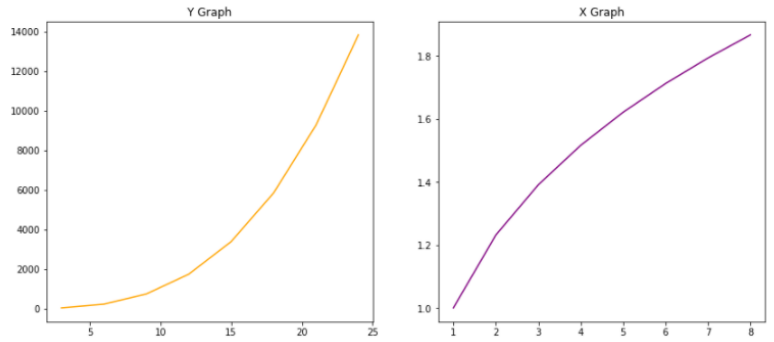
میشه یا facecolor() رنگ پشت زمینه هم به نمودار داد که تو نوت بوک ArcGIS Pro بهمون خطا میده.

اگه دوست داری نتیجه رو ببینی این کد رو تو نوت بوک ژوپیتر بنویس.

تمرین چهارم:

یه Figure و Axes دیگه با یه سطر و دو تا ستون تعریف میکنیم. بهش اندازه میدیم و اطلاعات نمودارهای X و Y رو به همراه عنوان براش وارد میکنیم و ازشون پلات میسازیم.

```
Figure_2, Axes_2 = plt.subplots(nrows=1,ncols=2,figsize=(14,6))
Axes_2[0].plot(Y,Y**3, color="orange")
Axes_2[0].set_title("Y Graph")
Axes_2[1].plot(X,X**0.3, color="purple")
Axes_2[1].set_title("X Graph")
plt.show()
```



تمرین پنجم:

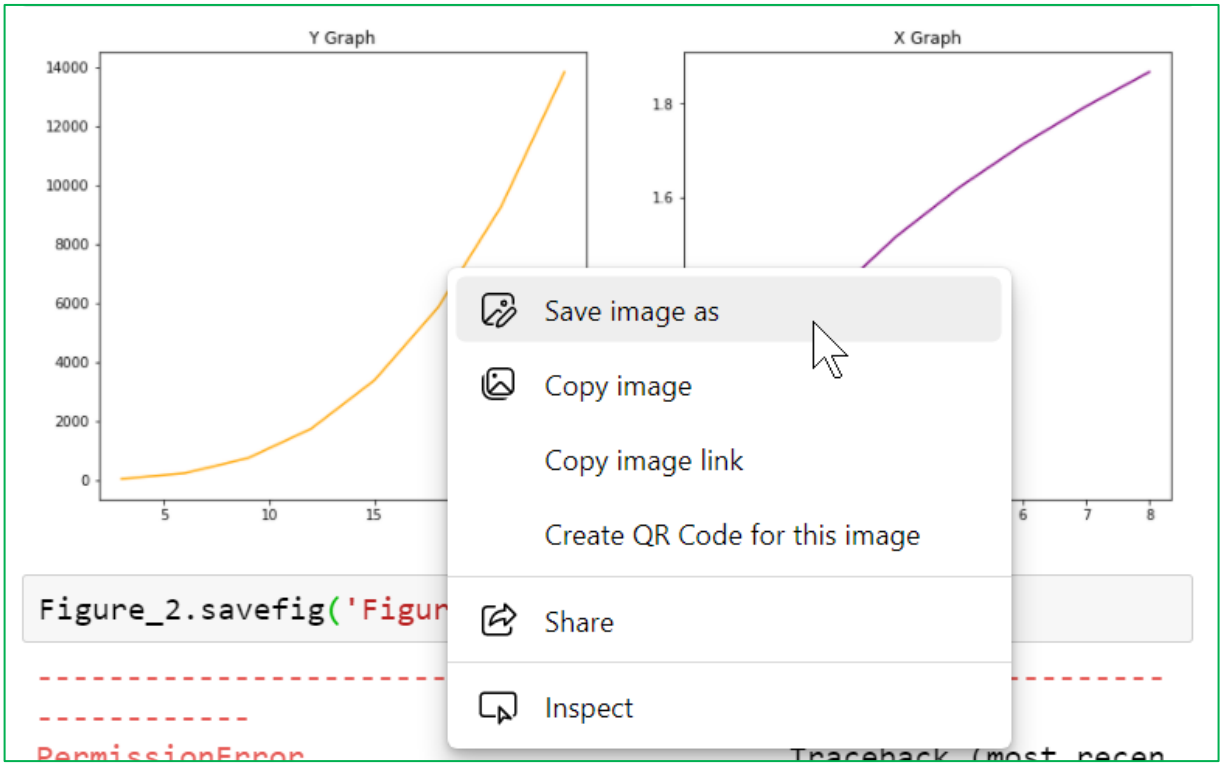
میتونیم فیگورها رو با تابع **savefig()**. ذخیره کنیم. که تو مســـیری که نوت بوک توش ذخیره شـــده فایلش با فرمت png درست میشه ولی تو ArcGIS Pro باز هم Error میبینیم. به فرمت pdf هم میشه خروجی گرفت ولی ما تو این نوت بوک خطا دریافت میکنیم.

```
Figure_2.savefig("Figure_2.png")

---------------------------------------------------------------------------
PermissionError                           Traceback (most recent call last)
In  [41]:
Line 1:     Figure_2.savefig("Figure_2.png")

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\matplotlib\figure.py, in savefig:
Line 3015:  self.canvas.print_figure(fname, **kwargs)

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\matplotlib\backend_bases.py, in print_figure:
Line 2255:  result = print_method(

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\matplotlib\backend_bases.py, in wrapper:
Line 1669:  return func(*args, **kwargs)

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\matplotlib\backends\backend_agg.py, in print_png:
Line 509:   mpl.image.imsave(

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\matplotlib\image.py, in imsave:
Line 1616:  image.save(fname, **pil_kwargs)

File E:\ArcGIS Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\PIL\Image.py, in save:
Line 2297:  fp = builtins.open(filename, "w+b")

PermissionError: [Errno 13] Permission denied: 'Figure_2.png'
---------------------------------------------------------------------------
```

میشه واسه رفع مشکل دو تا کار کرد:

۱- کد رو ببری تو نوت بوک ژوپیتر بنویســی که تصــویر رو تو مســیر C:\Users\nahid ذخیره میکنه و بعد بیاریش تو مسیر فایلهای پروژه‌ات.

۲- میشه روی نمودارها کلیک راست کرد و ازشون save as گرفت که بشه تو گزارشها ازشون استفاده کرد.

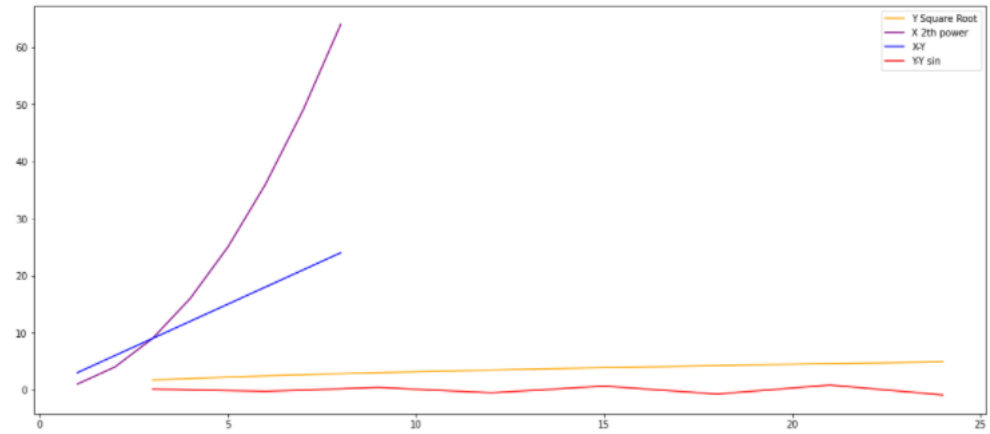


تمرین ششم:

میتونیم تو یه فیگور هر چند تا محوری که دوست داشتیم ایجاد کنیم.

یه فیگور و محور دیگه تعریف میکنیم. اطلاعات ۴ تا محور رو به صـــورت مجزا به همراه برچســـب و رنگ بهش میدیم. در انتها هم میگیم که برای کل نمودار یه راهنما یا legend بزنه.

```
Figure_3,Axes_3 = plt.subplots(figsize=(14,6))
Axes_3 = Figure_3.add_axes([0,0,1,1])

Axes_3.plot(Y,Y**0.5, label = "Y Square Root", color="orange")
Axes_3.plot(X,X**2, label = "X 2th power", color = "purple")
Axes_3.plot(X,Y, label = "X-Y", color = "blue")
Axes_3.plot(Y,np.sin(Y), label = "Y-Y sin", color="red")
Axes_3.legend()
plt.show()
```



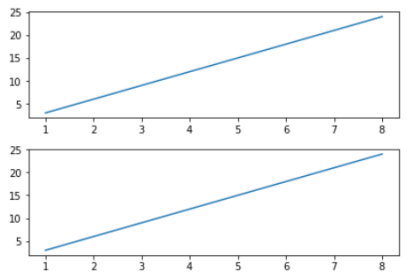
کدهای این بخش در یک نگاه

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

#تمرین۱

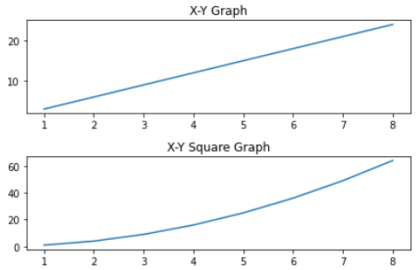
```
X = np.arange(1,9)
Y = np.arange(3,27,3)
Figure, Axes = plt.subplots(nrows=2, ncols=1)
plt.tight_layout()
print(type(Axes))
print(Axes[0])
print(Axes[1])
for ax in Axes:
    print(ax.plot(X,Y))
plt.show()
```

```
<class 'numpy.ndarray'>
AxesSubplot(0.0929977,0.577778;0.863484x0.370833)
AxesSubplot(0.0929977,0.0965278;0.863484x0.370833)
[<matplotlib.lines.Line2D object at 0x000001E0C7E94820>]
[<matplotlib.lines.Line2D object at 0x000001E0C7E79640>]
```



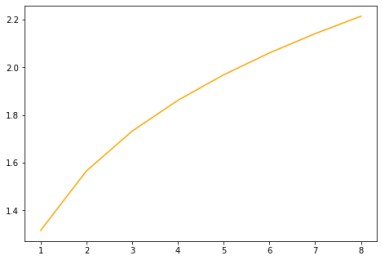
#تمرین۲

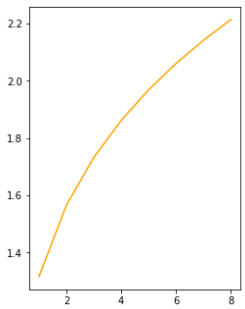
```
Figure, Axes = plt.subplots(nrows=2, ncols=1)
Axes[0].plot(X,Y)
Axes[0].set_title("X-Y Graph")
Axes[1].plot(X,X**2)
Axes[1].set_title("X-Y Square Graph")
plt.tight_layout()
plt.show()
```



#تمرین۳

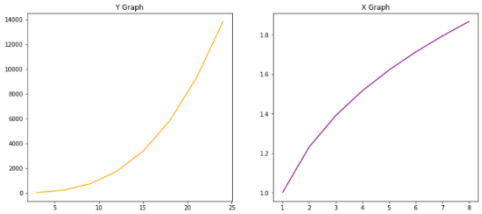
```
Figure_S = plt.figure(figsize=(6,4))
Axes_S = Figure_S.add_axes([0,0,1,1])
Axes_S.plot(X,Y**0.25, color="orange")
plt.show()
Figure_S = plt.figure(figsize=(3,4))
Axes_S = Figure_S.add_axes([0,0,1,1])
Axes_S.plot(X,Y**0.25, color="orange")
plt.show()
```





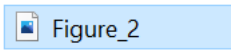
#تمرین۴

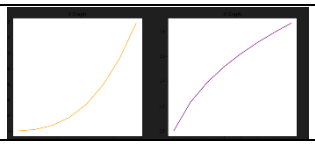
```
Figure_2, Axes_2 = plt.subplots(nrows=1, ncols=2, figsize=(14,6))
Axes_2[0].plot(Y,Y**3, color="orange")
Axes_2[0].set_title("Y Graph")
Axes_2[1].plot(X,X**0.3, color="purple")
Axes_2[1].set_title("X Graph")
plt.show()
```



#تمرین۵: اجرا شده در نوت بوک ژوپیتر

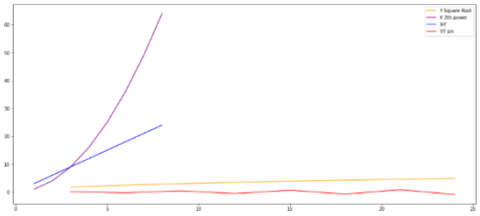
```
Figure_2.savefig("Figure_2.png")
```





#تمرین۶

```
Figure_3, Axes_3 = plt.subplots(figsize =(14,6))
Axes_3 = Figure_3.add_axes([0,0,1,1])
Axes_3.plot(Y,Y**0.5, label = "Y Square Root", color="orange")
Axes_3.plot(X,X**2, label = "X 2th power", color="purple")
Axes_3.plot(X,Y, label = "X-Y", color="blue")
Axes_3.plot(Y,np.sin(Y), label = "Y-Y sin", color="red")
Axes_3.legend()
plt.show()
```



سفارشی سازی گرافها در matplotlib

تو بخش قبلی یاد گرفتیم گه چطوری فیگورها رو سفارشی سازی کنیم. تو این بخش میخوایم سفارشی سازی گرافها رو باهم یاد بگیریم.

تمرین اول:

باز هم نامپی و مت پلاتلیب و عبارت جادویی رو وارد میکنیم.

آبجکت آرایه‌ها رو تعریف میکنیم و ازشون خروجی میگیریم.

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
X = np.random.randint(1,12,5)
Y = np.random.randint(1,10,5)
```

```
X
```

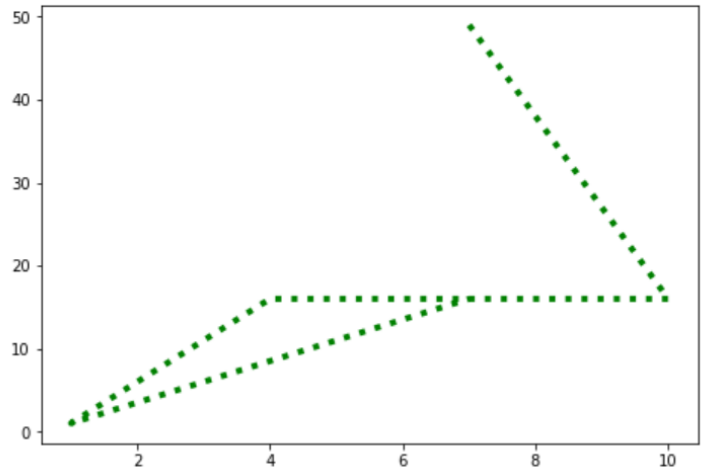
```
array([ 7, 10,  4,  1,  7])
```

```
Y
```

```
array([7, 4, 4, 1, 4])
```

بعدش باید Figure و Axes رو تعریف کنیم.

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color ="green", lw = 4, ls = ":")
plt.show()
```



میتونیم رنگبهش بدیم.

ضـــخامت خط رو با **linewidth =** میدیم که میشه به جاش از **lw** هم استفاده کرد.

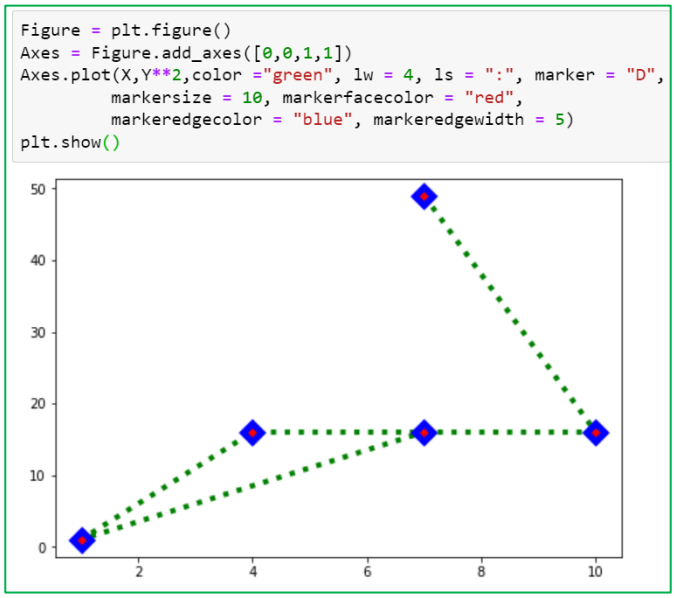
با **Linestyle =** هم میشه شکل خط رو تغییر داد. به جای linestyle میشه **ls** هم نوشت.

اگه شکل نمودارتون با من فرق داره واسه این هست که از random.ranint() استفاده کردیم و اعدادمون با هم متفاوت هست.

تمرین دوم:

با **marker =** هم میشه تغییراتی روی نمودار ایجاد کرد. میشـــه **markersize =** و **markerfacecolor =** هم بـهـش داد. بـهـش مـیـشـــه **markeredgecolor =** و **markeredgewidth =** هم داد.

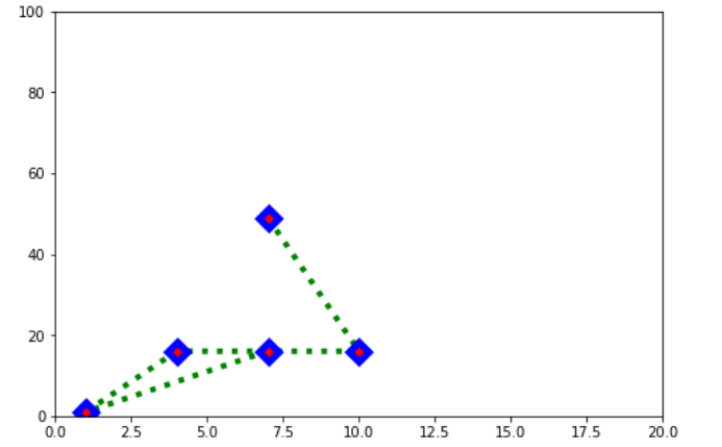
```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color ="green", lw = 4, ls = ":", marker = "D",
          markersize = 10, markerfacecolor = "red",
          markeredgecolor = "blue", markeredgewidth = 5)
plt.show()
```



تمرین سوم:

میتونیم محدوده یا Limit محورها رو هم با **set_xlim()** و **set_ylim()**. تغییر بدیم. اولین عدد شروع و دومین عدد پایان محور هست.

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color ="green", lw = 4, ls = ":", marker = "D",
          markersize = 10, markerfacecolor = "red",
          markeredgecolor = "blue", markeredgewidth = 5)
Axes.set_xlim(0,20)
Axes.set_ylim(0,100)
plt.show()
```

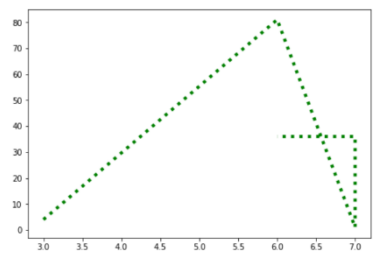


کدهای این بخش در یک نگاه

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

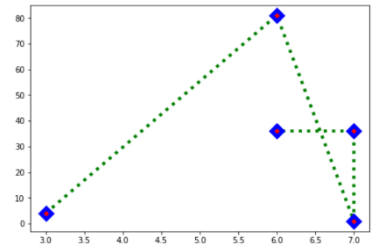
#تمرین۱

```
X = np.random.randint(1,12,5)
Y = np.random.randint(1,10,5)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color="green", lw=4, ls=":")
plt.show()
```



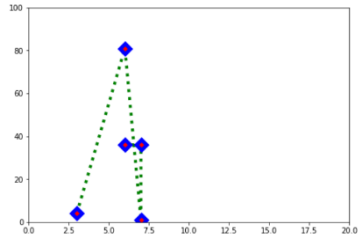
#تمرین۲

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color="green", lw=4, ls=":", marker= "D",
        markersize = 10, markerfacecolor = "red",
        markeredgecolor = "blue", markeredgewidth = 5)
plt.show()
```



#تمرین۳

```
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Axes.plot(X,Y**2,color="green", lw=4, ls=":", marker= "D",
        markersize = 10, markerfacecolor = "red",
        markeredgecolor = "blue", markeredgewidth = 5)
Axes.set_xlim(0,20)
Axes.set_ylim(0,100)
plt.show()
```



Grid, Spines, Ticks

Ticks چیه؟

از **ticks** واسه مشخص کردن مقادیر خاص روی محورهای نمودار استفاده میشه. یه سری ticks به صورت پیش فرض وجود داره ولی میتونیم خودمون هم سفارشیش کنیم که نمودار رو بهینه‌تر کنه. واسه اینکار از xticks() و yticks() استفاده میشه.

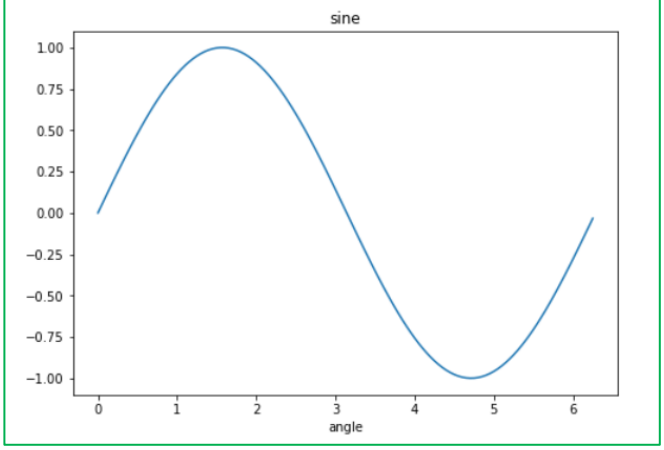
Math رو هم واسه انجام محاسبات همراه نامپی و بقیه موارد وارد میکنیم.

```
import numpy as np
import matplotlib.pyplot as plt
import math
%matplotlib inline
```

تمرین اول:

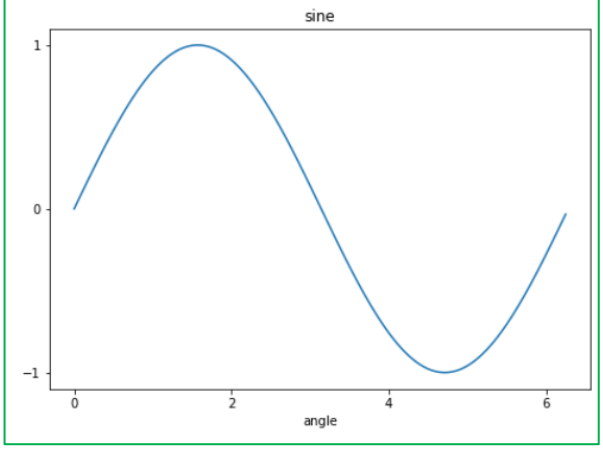
X و Y و همینطور Figure و Axes رو مثل مراحل قبل تعریف میکنیم.

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
plt.show()
```



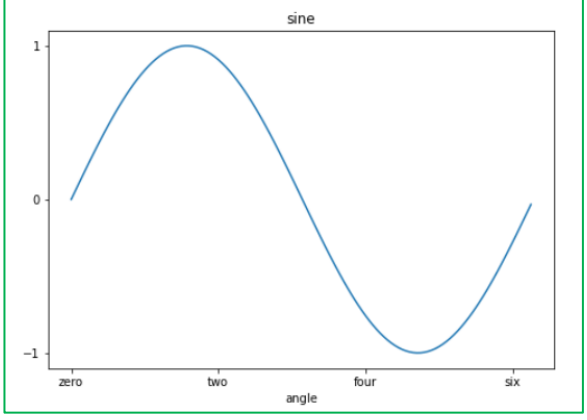
واسه محور x برچسب میزنیم و به محور عنوان میدیم.

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_yticks([-1,0,1])
plt.show()
```



حالا میتونیم بهش ticks اضافه کنیم. با نوشتن set_xticks() و st_yticks(). تغییراتی روی نوشته‌های محور X و Y ایجاد میشه.

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
plt.show()
```



میتونیم مقادیر محور x رو با set_xticklabels() به صورت نوشتاری دربیاریم.

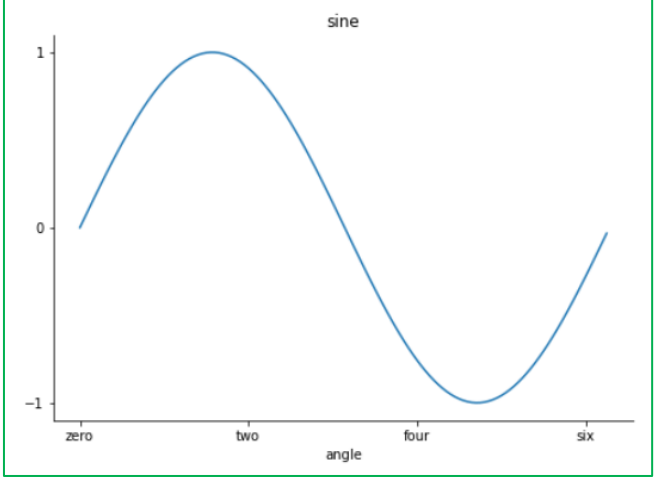
Spines چیه؟

Spineها خطوطی هســـتن که tickهای محور رو به هم متصــل میکنن. میتونیم برای مثال بگیم که ســمت راســت و بالای نمودار spine نداشته باشه یعنی خطی دیده نشه. واسه اینکـار بـایـد این محـدوده‌هـا رو False بـدیم. میتونیم بـا set_visible() اینکار رو انجام بدیم.

تمرین دوم:

کـد بـالا رو مجـدد کپی کن و بـا spines[].set_visible().
محورهایی که نمیخوای نشـون داده بشـه رو تو [] با کوتیشـن بنویس و تو پرانتز False رو وارد کن.

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
Axes.spines["right"].set_visible(False)
Axes.spines["top"].set_visible(False)
plt.show()
```



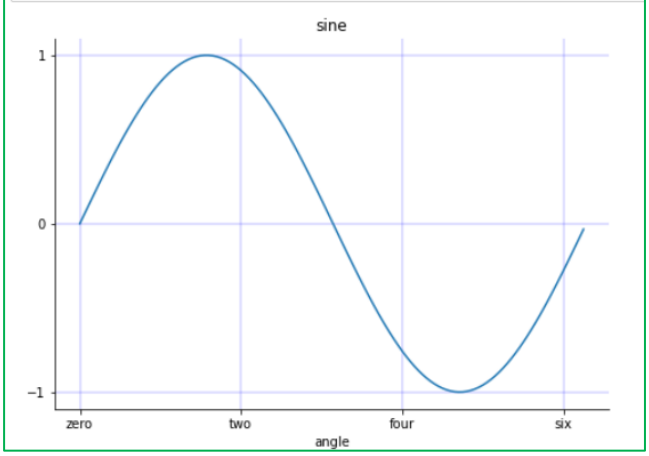
Grid چیه؟

واســه شــبکه‌بندی کردن نمودار اســتفاده میشــه. از grid(). برای نمایش دادن شــبکه گرید روی نمودار اســتفاده میشه که باید داخل پرانتزش True وارد کنیم.

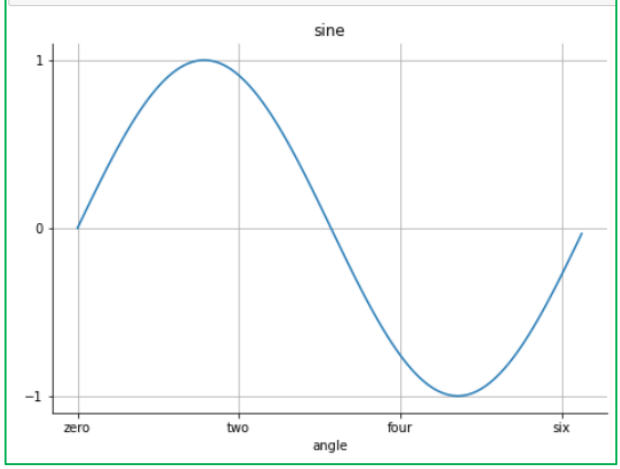
تمرین سوم:

کد بالا رو کپی کن و grid(). رو بهش اضــافه کن و تو پرانتزش بنویس True. میتونیم به جای True برای شــبکه گرید **رنگ و شکل و ضخامت شبکه** رو با color = ""، ls ="" و lw= مشخص کنیم.

```
X = np.arange(0, math.pi*2, 0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
Axes.spines["right"].set_visible(False)
Axes.spines["top"].set_visible(False)
Axes.grid(color="b",ls="-", lw="0.25")
plt.show()
```




```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
Axes.spines["right"].set_visible(False)
Axes.spines["top"].set_visible(False)
Axes.grid(True)
plt.show()
```

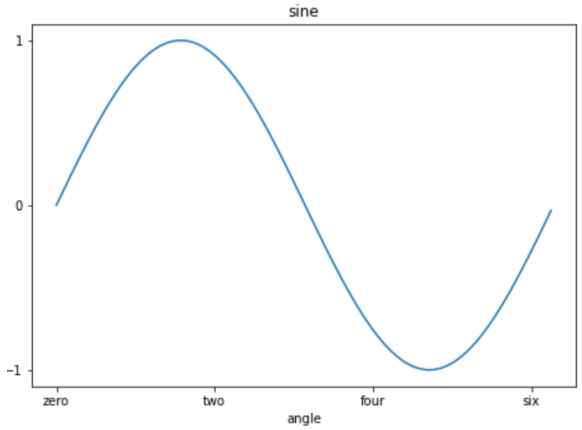


کدهای این بخش در یک نگاه

```
import numpy as np
import matplotlib.pyplot as plt
import math
%matplotlib inline
```

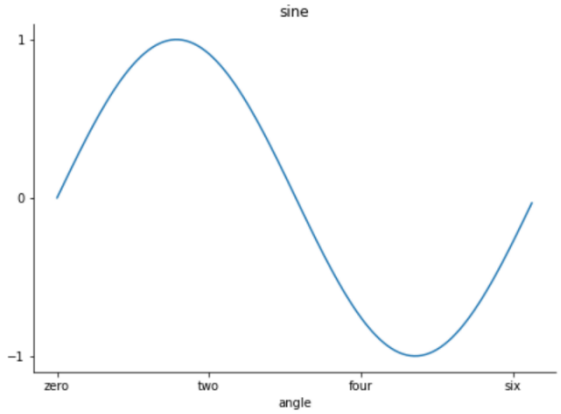
#تمرین۱

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
plt.show()
```



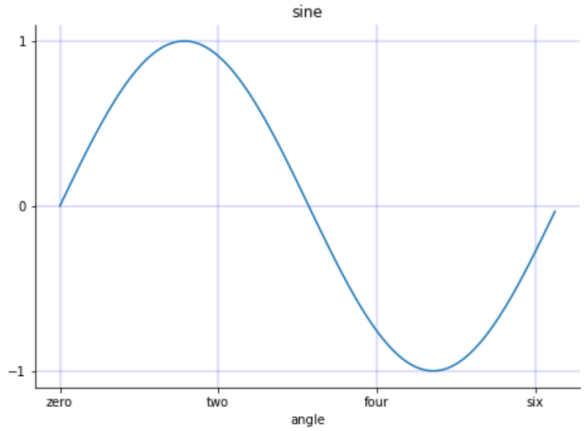
#تمرین۲

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
Axes.spines["right"].set_visible(False)
Axes.spines["top"].set_visible(False)
plt.show()
```



#تمرین۳

```
X = np.arange(0,math.pi*2,0.05)
Figure = plt.figure()
Axes = Figure.add_axes([0,0,1,1])
Y = np.sin(X)
Axes.plot(X,Y)
Axes.set_xlabel("angle")
Axes.set_title("sine")
Axes.set_xticks([0,2,4,6])
Axes.set_xticklabels(["zero","two","four","six"])
Axes.set_yticks([-1,0,1])
Axes.spines["right"].set_visible(False)
Axes.spines["top"].set_visible(False)
Axes.grid(color="b", ls= "-", lw = 0.25)
plt.show()
```



خلاصه دستورهای Matplotlib.pyplot

توابع و متدها	توضیحات
%matplotlib inline	یه تابع هست که باعث میشه نمودارهایی که تو نوت بوکمون ترسیم میکنیم همینجا بتونیم ببینیمشـــون و تو همین نوت بوک هم ذخیره شن
plt.show()	واسه نمایش نمودارهای ساخته شده استفاده میشه.
plt.plot() color = linewidth = / lw = Linestyle = / ls = marker = markersize = markerfacecolor = markeredgecolor = markeredgewidth =	واسه ساخت نمودار خطی با دو مقدار x و y استفاده میشه. تو پرانتزش میشه با color = به نمودار رنگ داد میشه. ضخامت و نوع خط هم میشه براش تعریف کرد. میشه رنگ و طرح رو با هم ترکیب کرد مثلا: 'r.' یا '–g'. بـا marker هم میشـــه روی خط علائم نشـــونـد و بـه خودش و حاشیه‌هاش اندازه و رنگ داد.
plt.xlabel()/plt.ylabel ()	برای لیبل زدن به محورهای x و y استفاده میشه.
plt.title()	برای عنوان دادن به نمودار استفاده میشه.
.sin() و .cos()	برای ترسیم نمودارهای سینوسی و کسینوسی استفاده میشه.
plt.bar() .get_height()/ .get_width	ترسیم نمودار میله ای استفاده میشه. ارتفاع و عرض میله‌ها رو میشه بهش داد.
plt.scatter()	نمودار پراکندگی داده‌ها با دو تا متغیر x و y هست.
plt.polar() .fill()	واسه ترسیم نمودار قطبی استفاده میشه که واسه نشون دادن رابطه بین دو یا چند متغیر هست. با ()fill. میشه داخل نمودار رو رنگی کرد و بهش شفافیت هم داد.
plt.step()	واسه ترسیم نمودار پله‌ای استفاده میشه.
plt.figure() figsize = (,)/ dpi = () facecolor = " "	واسه ساخت فیگور یا شکل استفاده میشه. تعیین اندازه شکل رنگ پشت زمینه به نموار
plt.hist() bins, n, patches .axvline()	هیستوگرام میده. در این نمودار، محور افقی به مقادیر مختلف متغیر پیوسـته اختصـاص داده میشـه و محور عمودی نشـان دهنده فراوانی هر بازه هست. n یه لیســت هســت که تعداد آیتمها رو تو هر bin مشــخص میکنه. bins نقطه شــروع bin یا میله رو مشــخص میکنه. patches هم یه لیست آبجکت هست برای هر bin. در واقع مستطیلهایی روی نمودار هســتن که رنگ پیش فرض آبی دارن. ()axvline. یه خط عمودی در سرتاسر میله‌ها اضافه میکنه.

plt.bar()	نمودار جعبه‌ای با استفاده از ۵ تا عدد زیر وضعیت گروهی از داده‌ها رو به تصویر میکشه. - **Min یا حداقل:** کمترین مقدار در دسته داده‌ها (بدون در نظر گرفتن داده‌های پرت). - **چارک اول:** ۲۵ درصد داده‌ها کمتر از این مقدار هستن. - **چارک دوم یا میانگین:** مقدار وسط دسته داده‌ها هست. نصف مقادیر کمتر و نصف مقادیر بیشتر از اون مقدار قرار دارن. - **چارک سوم:** ۷۵ درصد داده‌ها کمتر از اون مقدار هستن. - **Max یا حداکثر:** بزرگترین مقدار در دسته داده‌ها (بدون در نظر گرفتن داده‌های پرت).
plt.violinplot() showmeans = True	نمودار ویولنی از نمودار جعبه‌ای ساده اطلاعات بیشتری رو منتقل میکنه. برای مقایسهٔ داده‌های آماری به صورت خلاصه (مثل بازه‌ها و چارکها) کاربرد داره ولی امکان مشاهدهٔ تغییرات و اختلافات در داده رو نمیده.
.cor() plt.imshow() plt.colornbar()	هیت مپ Heat map یک روش دیداری برای نمایش داده‌های دو بعدی هست. در این روش، هر مقدار داده با یک رنگ متفاوت نمایش داده میشه. - از corr() واسه ماتریس همبستگی استفاده میکنیم. اعددش بین ۱- تا ۱+ هستن. یعنی از منفی‌ترین تا مثبت‌ترین. از قرمز تیره میده تا قرمز روشن. - از imshow() برای نمایش تصویر استفاده میکنیم. بهمون تصاویر مربعی میده. - colorbar() تعریف میکنیم که کنار تصویر بهمون یه میله رنگی هم بده.
plt.stackplot()	واسه نمایش توالی زمانی استفاده میشه که به صورت پشته‌ای از منحنی‌ها رسم میشه.
plt.pie() plt.axis()	نمودار pie برامون میسازه. یه autopct داره که نشون دهنده میزان عدد اعشار هر نقطه داده هست. باید تو پرانتز plt.axis() کلمه 'equal' وارد شه که x و y برابر بگیره و دایره خروجی بده.
plt.subplot()	ترسیم نمودارهای مشابه به صورت همزمان. سه تا آرگومان داره اولی عدد **سطر**، دومی عدد **ستون** و سومی **شماره نمودار** هست. رنگ هم میشه بهشون داد.
.add_axes()	واسه اضافه کردن محور یا axes به فیگور استفاده میشه. ۴ تا عدد میگیره دو تا اول نقاط شروع و دو تا عدد دوم نقاط پایان محورها هستن.

واسه راهنما زدن استفاده میشه. با loc = مشخص میکنیم که راهنما کجا بیاد.	.legend()
وقتی چند تا نمودار کنار هم میذاریم و میخوایم اعداد محورهاشون تو هم نرن استفاده میشه.	plt.tight_layout()
نوع داده محور رو مشخص میکنه. واســـه عنوان دادن به هر نمودار موقع ترســـیم نمودارهای چندتایی استفاده میشه.	type() .set_title()
واسه ذخیره کردن نمودارها استفاده میشه.	.savefig()
محدوده محورها رو میشـه تغییر داد. اولین عدد داخل پرانتز شـروع و دومین عدد پایانش هست.	.set_xlim()/.set_ylim()
واسـه تغییر اعداد و عبارات روی محور x و y اسـتفاده میشـه. با label میشه اعداد رو به صورت نوشتاری روی محور نشون داد.	set.xticks()/ set.yticks() set_xticklabels()
خطوط ۴ طرف محور هستن که میشه برشون داشت. پیش فرضشون True هست.	.spins[].set_visible()
واســـه شـــبکه بندی نمودار اســـتفاده میشـــه. تو پرانتزش باید True بنویسیم. میشه بهش ضخامت و نوع خط هم داد.	.grid() ls = / lw =

حل چندتا تمرین

نمونه ای از نمودارهای زیر روبا داده‌های دلخواه ترسیم کن:

۱- Line plot
۲- Bar plot
۳- Scatter plot
۴- Stack plot
۵- Pie plot
۶- Polar plot
۷- Hist plot
۸- Box plot
۹- Violin plot
۱۰- Heat map

جواب تمرینها

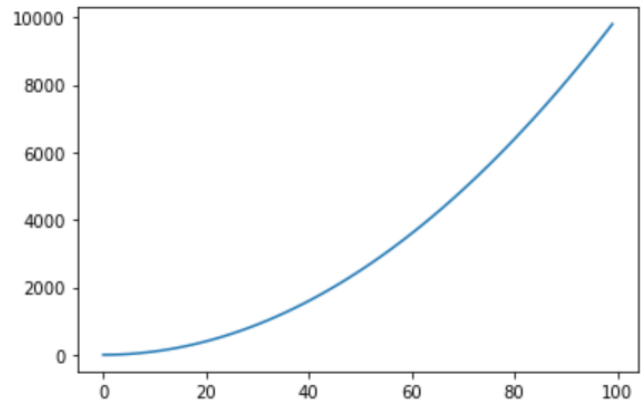
نمودارهای پایه در matplotlib

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import calendar
%matplotlib inline
```

Numpy، matplotlib.pyplot رو به همراه math و calendar وارد کن. عبارت جادویی %matplotlib inline رو هم بنویس. شیفت اینتر بزن که بری تو سلول بعدی. حالا میخوایم با هم ترسیم ۱۰ تا نمودار رو یاد بگیریم.

نمونه نمودار Line plot

```
X = range(100)
Y = [value**2 for value in X]
plt.plot(X,Y)
plt.show()
```



یه نمودار خطی هست که ازش واسه نشون دادن تغییرات یه متغیر در طول زمان استفاده میشه. مثل GDP، نرخ تورم، تاریخچه شاخصهای عمده سهام و ...

دو تا آرایه X و Y درست کن. واسه تعریف X از range(100) استفاده کن که اعدادی بین ۰ تا ۹۹ تولید شه. واسه تعریف Y یه لیست درست کن (باید بذاری تو براکت []) یه حلقه روی X بنویس که هر کدوم از مقادیر رو از X به توان ۲ برسونه و تو آرایه Y خروجی بده. در واقع اعداد Y شامل 0, 1, 4, 9, 16, 25 , ... میشن.

بعد با plt.plot(X ,Y) ازشون پلات درست کن. با plt.show() نمایشش بده.

نمونه نمودار bar

یه نمودار ستونی هست که ستونها با فاصله یکنواختی از هم قرار میگیرن. واسه مقایسه مقادیر گروه‌های مختلف استفاده میشه. مثلا برای مقایسه میزان فروش بستنی در هر ماه از یک سال. برای ساختش مراحل زیر رو میریم:

- پکیج تقویم پایتون رو با import calendar وارد میکنیم که بتونیم ماه‌ها رو وارد کنیم.
- یه آرایه درست میکنیم به اسم month_num برای شماره ماه‌ها که باید فرمتش عددی باشه یعنی از ۱ تا ۱۲.
- بعد مقدار فروش (مثلا بستنی) رو برای هر ماه با اسم آرایه units_sold وارد میکنیم.
- با sublpots() فیگور و محور رو تعریف میکنیم.
- tickها رو برای مشخص کردن اطلاعات مشخص رو محور بهش میدیم. تو پرانتز xticks متغیر month_num رو وارد میکنیم و با calendar میخوایم که اسامی ماه‌ها رو روی محور x بندازه. باید بنویسیم calendar.month_name[1:13] که اسامی ماه‌ها رو از January تا December بهمون روی محور X نشون بده. ۲۰ درجه هم میچرخونیمش که اسامی تو هم نره. واسه چرخش از rotation = استفاده میکنیم.
- یه نمودار bar میسازیم و آرایه month_num رو به عنوان محور x و آرایه units_sold رو به عنوان محور y بهش معرفی میکنیم.

- حالا میخوایم مقادیر داده‌ها رو بالای نمودار بیاریم. یه حلقه for مینویسیم. با get_height() ارتفاع رو براش مشخص میکنیم. واسه محور text مینویسم که هر نمودار مقدارش که برای unit_sold مشخص کردیم رو بهمون نشون بده. ha وضعیت تراز افقی رو مشخص میکنه و va وضعیت تراز عمودی رو.

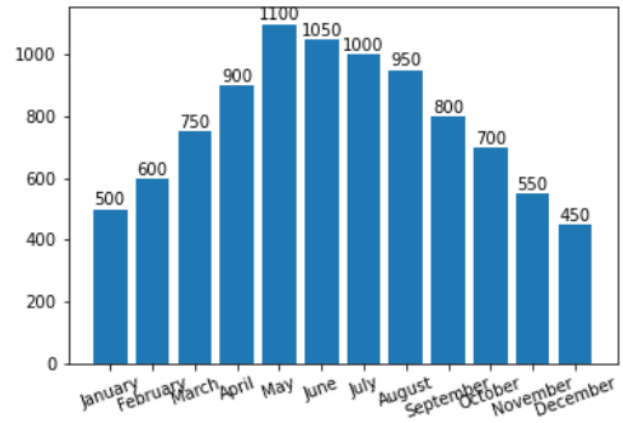
واسه ویژگیهای بیشتر این لینک رو ببین:

matplotlib.pyplot.text — Matplotlib 3.8.0 documentation

- با plt.show() ازش خروجی میگیریم.

```
import calendar
```

```
month_num = [1,2,3,4,5,6,7,8,9,10,11,12]
units_sold = [500,600,750,900,1100,1050,1000,950,800,700,550,450]
fig, ax = plt.subplots()
plt.xticks(month_num, calendar.month_name[1:13], rotation=20)
plot = ax.bar(month_num, units_sold)
for rect in plot:
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2., 1.002*height, '%d' % int(height), ha='center', va='bottom')
plt.show()
```



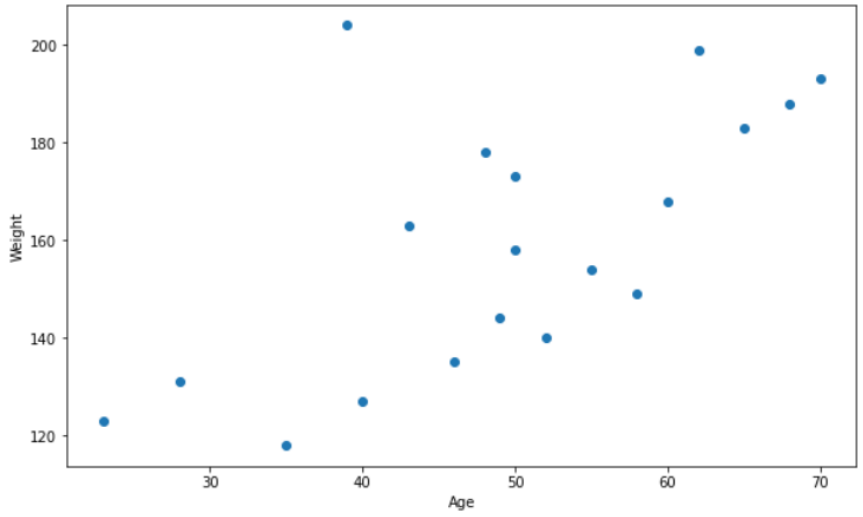
نمونه نمودار scatter

واسه مقایسه پراکندگی دو تا متغیر استفاده میشه. واسه نمایش داده‌های عددی و متغیرهای کمی استفاده میشه. ارتباط بین متغیرها رو بهمون میده. میتونیم باهاش الگوی حاکم بر داده‌ها رو پیدا کنیم و میزان همبستگی داده‌ها رو بررسی کنیم.

- اینبار میخوایم از pandas استفاده کنیم. پس باید واردش کنیم. import pandas as pd
- یه فیگور می سازیم و اندازه‌اش رو به ۱۰ و ۶ تغییر میدیم. از figsize = استفاده میکنیم.
- یه فایل اکسل به اسم scatter_ex.xlsx اضافه میکنیم (*این فایل رو برات تو مسیر ذخیره پروژه matplotlib گذاشتم. اگه میخوای از فایل دیگه‌ای استفاده کنی یادت باشه که باید بذاریش تو مسیری که نوت بوکهات ذخیره شدن یا آدرس دقیقشو بگی مثل: r"D:/ scatter_ex.xlsx" البته این شیوه آدرس‌دهی رو تو Arcpy با هم یاد میگیریم*). این فایل اطلاعات سن و وزن افراد رو داره. اسم متغیرش رو age_weight میذاریم. چون این اطلاعات تو شیت age_weight ذخیره شده باید تو پرانتز بعد از اسم فایل بیاریمش.
- محورهای x و y رو تعریف میکنیم. برای x از اطلاعات ستون age استفاده کردیم و برای y از اطلاعات ستون weight که باید بذاریمشون داخل کوتیشن و براکت ["]

- بعد میگیم که از محورهای X و Y نمودار scatter برامون بسازه.
- برای محورهای x و y برچسب هم تعریف میکنیم. برای اینکار از plt.xlabel() و plt.ylabel() استفاده کردیم.
- با plt.show() هم نمایشش میدیم.

```
plt.figure(figsize=(10,6))
age_weight = pd.read_excel('scatter_ex.xlsx','age_weight')
X = age_weight['age']
Y = age_weight['weight']
plt.scatter(X,Y)
plt.xlabel('Age')
plt.ylabel('Weight')
plt.show()
```

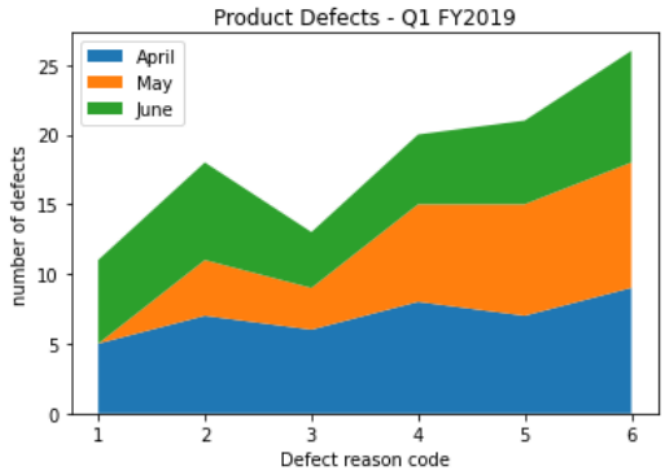


نمونه نمودار stack

از این نمودار واسه نمایش توالی‌های زمانی استفاده میشه و به صورت پشته‌ای از منحنی‌ها رسم میشه. واسه ساختش باید مراحل زیر رو بریم:

- یه آرایه با نامپی تعریف میکنیم. نوع داده‌اش رو هم با dtype = عدد صحیح میذاریم.
- برای هر کدوم از ماه‌های April, may, June یه لیست مجزا با ۶ تا عدد درست میکنیم.
- برچسبها رو تعریف میکنیم.
- با subplots() فیگور و محور رو بهش میدیم.
- نمودار stockplot() رو تعریف میکنیم.
- بهش راهنما یا legend(loc =). اختصاص میدیم. پیش فرض موقعیت راهنما سمت راست و بالا هست که موقعیت ۱ هست. موقعیت ۲ میشه چپ بالا یعنی پادساعتگرد میچرخه.
- برچسبهای محور x و y رو مشخص میکنیم.
- عنوان بهش میدیم.
- ازش خروجی میگیریم.

```
x = np.array([1, 2, 3, 4, 5, 6], dtype = np.int32)
Apr = [5, 7, 6, 8, 7, 9]
May = [0, 4, 3, 7, 8, 9]
June = [6, 7, 4, 5, 6, 8]
labels = ["April", "May", "June"]
fig, ax = plt.subplots()
ax.stackplot(x, Apr,May,June, labels = labels)
ax.legend(loc = 2)
plt.xlabel('Defect reason code')
plt.ylabel('number of defects')
plt.title('Product Defects - Q1 FY2019')
plt.show()
```



نمونه نمودار pie

این نمودار معمولا برای نمایش سهم‌ها، بصورت عددی و درصدی استفاده میشه. مثلا سهم هر ایالت در میزان GDP، نمرات دانش‌آموزان به تفکیک A, B, C, D برای هر کلاس و .... واسه ترسیمش مراحل زیر رو میریم:

- یه لیست با مقادیر رشته‌ای یه اسم labels درست می‌کنیم. انواع ژانر فیلمها رو که در یک سال ساخته شده بهش میدیم.
- یه لیست دیگه با مقادیر عددی به اسم sizes درست میکنیم. تعدادش باید با تعداد برچسبها یکی باشه. این اعداد به صورت دایره ساعتگرد مشخص میشن.
- یه تاپل به اسم explode هم می‌سازیم که هر اسلایس یا برش نمودار رو مشخص میکنه.
- بعد نمودار pie رو با plt.pie() درست میکنیم. autopct نشون دهنده میزان عدد اعشار هر نقطه داده هست. اینجا مینویسیم '%1.1f&&'[1]. shadow مشخص میکنه که اسلایسها با سایه دیده شن یا نه. شروع نمودار از چه زاویه‌ای باشه رو هم با startangel = بهش میدیم.
- plt.axis() هم معادل equal هست که کمک میکنه نمودار به صورت دایره‌ای دیده شه. یعنی x , y برابر هستن.

واسه اطلاعات بیشتر این لینک رو ببین:

matplotlib.pyplot.axis — Matplotlib 3.8.0 documentation

این کد تو نوت بوک ArcGIS Pro خطا میده و باید تو نوت بوک ژوپیتر نوشته شه.

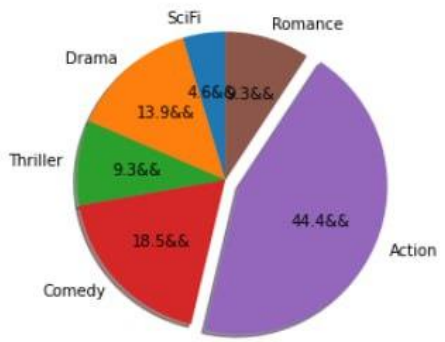
```
labels = ["SciFi","Drama","Thriller","Comedy","Action","Romance"
sizes = [5,15,10,20,48,10]
explode = (0,0,0,0,0.1,0)
plt.pie(sizes, labels=labels, explode=explode, autopct='%1.1f&&'
        shadow=True, startangle=90)
plt.axis('equal')
plt.show()
```

```
---------------------------------------------------------------
------------
TypeError                                 Traceback (most recen
t call last)
In  [2]:
Line 7:     plt.show()

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\mat
plotlib\pyplot.py, in show:
Line 378:   return _backend_mod.show(*args, **kwargs)

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\ipy
kernel\pylab\backend_inline.py, in show:
Line 41:    display(

TypeError: 'NoneType' object is not iterable
---------------------------------------------------------------
```

[1] عبارت %1.1f معمولا تو زبانهای برنامه‌نویسی بخصوص زبانهایی مثل Python و C استفاده میشه که میشه که واسه قالب‌بندی و نمایش اعداد اعشاری بکار میره. % مقداری رو نگه میداره که باید فرمت یا قالب‌بندی بشه. عدد ۱ که قبل نقطه می‌آد حداقل عرض فیلد یا ستون رو مشخص میکنه یعنی تعداد حرفی که میشه تو اون فیلد نوشت حداقل یه حرف هست. نقطه بعد از ۱ واسه جدا کردن حداقل عرض از دقت یا precision استفاده میشه. ۱ بعد از نقطه تعداد رقم اعشار رو نشون میده و f هم یعنی عددی که داره قالب‌بندی میشه یه عدد اعشاری هست. به عبارت دیگه %1.1f یعنی یه عدد اعشاری با حداقل ۱ عدد صحیح و یه عدد اعشاری نمایش داده میشه. عبارت && هم نشونه عملگر AND هست.

بعد از اجرای کد تو نوت‌بوک ژوپیتر میتونی نوت بوکش رو بیاری کنار بقیه نوت بوکهات و جزو اسناد پروژه داشته باشیش.

```
lables=["SciFi","Drama","Thriller","Comedy","Action","Romance"]
sizes=[5,15,10,20,48,10]
explode =(0,0,0,0,0.1,0)
plt.pie(sizes,labels=lables,explode=explode,autopct="%1.1f&&",shadow=True,startangle=90)
plt.axis("equal")
plt.show()
```



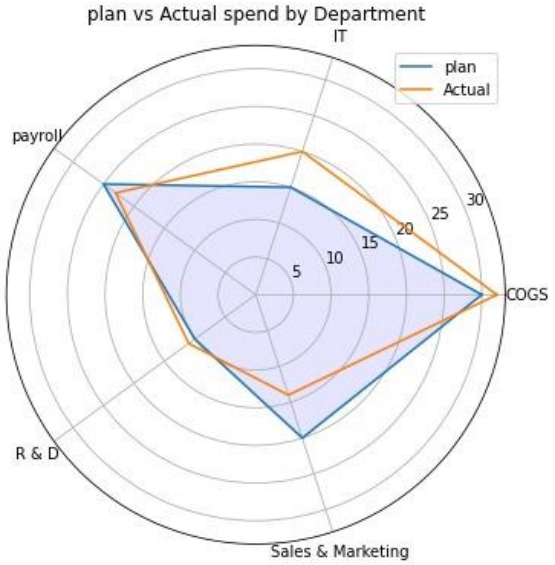
نمونه نمودار Polar

نمودار قطبی یا نمودارهای دایره‌ای، از محورهای قطبی به جای محورهای مستطیلی استفاده می‌کنه. از این نمودارها واسه نشون دادن روابط بین دو یا چند متغیر و همینطور نشون دادن تغییرات زمانی در یه سری زمانی یا روابط بین شعاع و زوایا یا تغییرات در داده‌ها استفاده میکنن.

واسه ترسیمش باید مراحل زیر رو بریم:

- میخوایم رابطه بین میزان هزینه برنامه‌ریزی شده رو با هزینه واقعی تو بخشهای مختلف یه سازمان بسنجیم. اسم دپارتمانها رو به صورت لیست وارد میکنیم. [,]
- میزان هزینه برنامه‌ریزی شده یا rp و هزینه واقعی یا ra رو هم به صورت لیست مینویسیم.
- با ()np.linspace اولین و آخرین آرگومان رو بهش میدیم. در واقع با اینکار دایره ما به تعداد داده‌ها (دپارتمانهای مختلف) که ۵ تا هست تقسیم میشه یا برش میخوره. از ۰ تا 2pi.
- یه فیگور به همراه اندازه اش میسازیم.
- توی پرانتز subplot مقدار Polar رو True میدیم.
- خط هفتم یا (lines, labels) برامون گرید یا شبکه روی نمودار ایجاد میکنه. lines و labels رو به صورت tuples تو پرانتز مینویسیم (,).
- تبدیلش میکنیم به نمودار که اینجا با rp یا هزینه برنامه‌ریزی شده می‌سازیمش.
- ()fill. داخل نمودار رو رنگ میده که b به معنی آبی رو بهش دادیم. با alpha هم بهش شفافیت یا transparency میدیم.
- مجدد تبدیلش میکنیم به پلات ولی اینبار با ra یعنی هزینه واقعی. میتونیم براش ()fill. هم تعریف کنیم ولی رنگها تو هم میره.
- بهش راهنما و عنوان هم میدیم و با ()plt.show ازش خروجی میگیریم.

```
Depts = ["COGS", "IT", "payroll", "R & D", "Sales & Marketing"]
rp = [30,15,25,10,20,30]
ra = [32,20,23,11,14,32]
theta = np.linspace(0, 2*np.pi, len(rp))
plt.figure(figsize = (10,6))
plt.subplot(polar =True)
(lines, labels) = plt.thetagrids(range(0,360, int(360/len(Depts))), (Depts))
plt.plot(theta, rp)
plt.fill(theta, rp, 'b', alpha = 0.1)
plt.plot(theta, ra)
plt.legend(labels=('plan','Actual'), loc=1)
plt.title("plan vs Actual spend by Department")
plt.show()
```



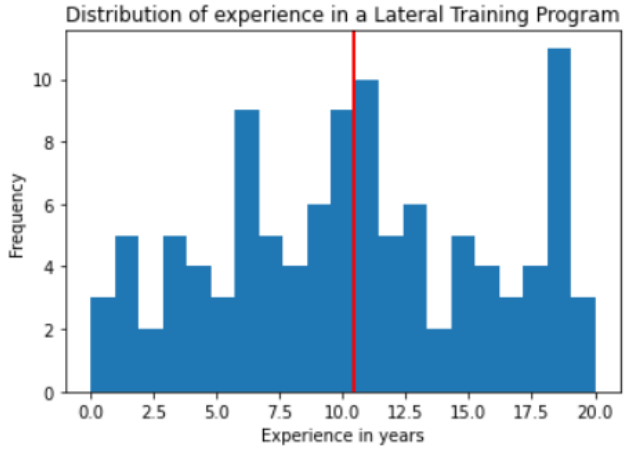
نمونه نمودار hist یا هیستوگرام

نمودار hist plot یکی از نمودارهای محبوب در تجزیه و تحلیل داده‌ها هســـت. از این نمودار برای نشـــون دادن توزیع فراوانی یک متغیر پیوسته استفاده میشه. در این نمودار، محور افقی به مقادیر مختلف متغیر پیوسته اختصاص داده میشه و محور عمودی نشون دهنده فراوانی هر بازه هست.

برای ترسیمش مراحل زیر رو میریم.

- یه آرایه نامپی درســـت میکنیم که شـــامل داده‌های تجربه کاری در Lateral Training Program یا برنامه آموزش جانبی در سالهای مختلف هست.
- nbins رو به ۲۱ دادیم. Bin به میله‌های هیســـتوگرام گفته میشـــه. در واقع میخوایم داده‌ها تو ۲۱ میله نمایش داده بشن.
- با plt.hist() میتونیم یه هیستورگرام بسازیم. سه تا پارامتر داره. bins، n و patches. n یه لیست هست که تعداد آیتمها رو تو هر bin مشخص میکنه. bins نقطه شروع bin یا میله رو مشخص میکنه. patches هم یه لیست آبجکت برای هر bin هست. در واقع همون مستطیلهایی روی نمودار هستن که رنگ پیش فرض آبی دارن.
- برچسب و عنوان براش مشخص میکنم.
- با axvline(). مشـــخص میکنیم که داده‌ها چطوری پخش بشـــن. در واقع axvline(). یه خط عمودی در سرتاسر میله‌ها اضافه میکنه. اینجا خواستیم که یه خط عمودی قرمز با ضخامت ۲ در میانگین یا وسط نمودار برامون ترسیم کنه.

```
grp_exp = np.array([12, 15, 13, 20,19, 20, 11, 19, 11, 12, 19, 13, 12,
                    10, 6, 19, 3, 1, 1, 0, 4, 4, 6, 5, 3, 7, 12, 7, 9,
                    8, 12, 11, 11, 18, 19, 18, 19, 3, 6, 5, 6, 9, 11,
                    10, 14, 14, 16, 17, 17, 19, 0, 2, 0, 3, 1, 4, 6,
                    6, 8, 7, 7, 6, 7, 11, 11, 10, 11, 10, 13, 13, 15,
                    18, 20, 19, 1, 10, 8, 16, 19, 19, 17, 16, 11, 1,
                    10, 13, 15, 3, 8, 6, 9, 10, 15, 19, 2, 4, 5, 6, 9,
                    11, 10, 9, 10, 9, 15, 16, 18, 13])
nbins = 21
n, bins, patches = plt.hist(grp_exp, bins = nbins)
plt.xlabel("Experience in years")
plt.ylabel("Frequency")
plt.title("Distribution of experience in a Lateral Training Program")
plt.axvline(x = grp_exp.mean(), linewidth = 2, color = 'r')
plt. show()
```



نمونه نمودار box

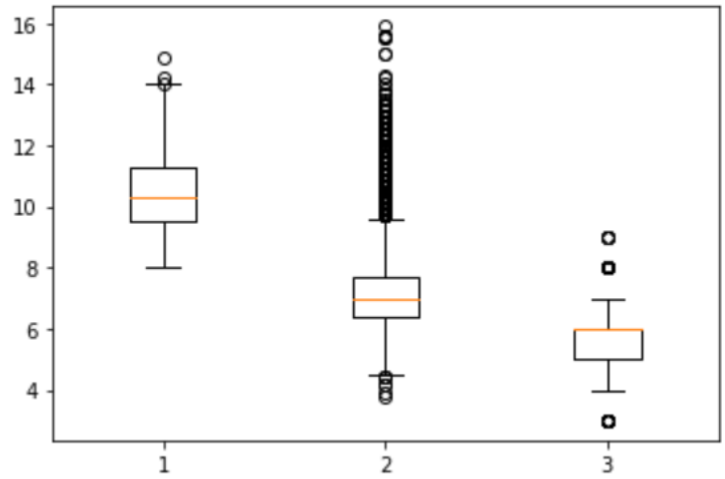
نمودار جعبه‌ای با استفاده از ۵ تا عدد زیر وضعیت گروهی از داده‌ها رو به تصویر میکشه.

- ✓ **Min یا حداقل:** کمترین مقدار در دسته داده‌ها (بدون در نظر گرفتن داده‌های پرت).
- ✓ **چارک اول:** ۲۵ درصد داده‌ها کمتر از این مقدار هستن.
- ✓ **چارک دوم یا میانگین:** مقدار وسط دسته داده‌ها هست. نصف مقادیر کمتر و نصف مقادیر بیشتر از اون مقدار قرار دارن.
- ✓ **چارک سوم:** ۷۵ درصد داده‌ها کمتر از اون مقدار هستن.
- ✓ **Max یا حداکثر:** بزرگترین مقدار در دسته داده‌ها (بدون در نظر گرفتن داده‌های پرت).

برای ترسیمش باید مراحل زیر رو بریم:

- یه داده csv میخوایم بیاریم پس یادت نره که pandas رو هم import کنی. فایل winequality.csv رو داخل فولدری که نوت بوک رو داره میخونه ذخیره کنیم.
- سه تا ستون از ویژگیهایی که تو این جدل هست رو باید تو یه لیست تعریف کنیم.
- با boxplot() تبدیلشون میکنیم به نمودار جعبه‌ای
- با plt.show() هم نمایشش میدیم.

```
wine_quality = pd.read_csv('winequality.csv')
data = [wine_quality['alcohol'], wine_quality['fixed acidity'], wine_quality['quality']]
plt.boxplot(data)
plt.show()
```

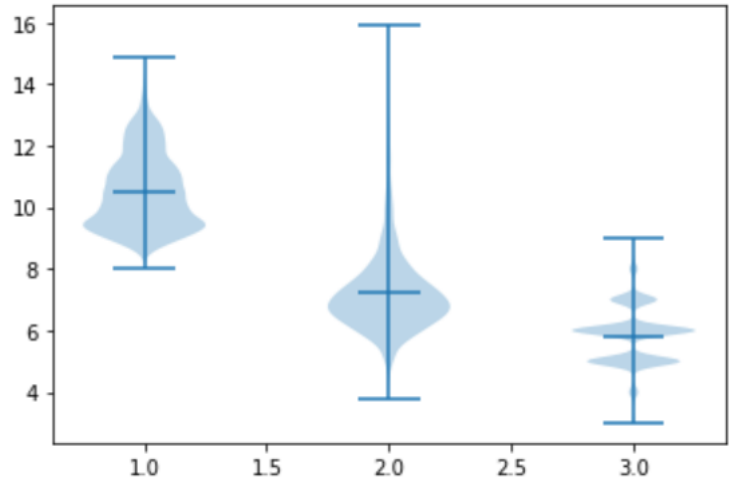


نمونه نمودار violin ترکیب هیستوگرام و box

نمودار ویولنی نسبت به نمودار جعبه‌ای ساده، اطلاعات بیشتری رو منتقل میکنه. برای مقایسهٔ داده‌های آماری به صورت خلاصه (مانند بازه‌ها و چارکها) کاربرد داره ولی امکان مشاهدهٔ تغییرات و اختلافات در داده رو نمیده. برای ترسیمش باید مراحل زیر رو بریم:

- دو خط اول کد بالا رو کپی میکنیم.
- با violinplot() نمودارش رو میســازیم. گزینه showmwans رو هم روی True میذاریم که میانگین رو بهمون با خط نشون بده.
- با plt.show() نمودار رو نمایش میدیم.

```
wine_quality = pd.read_csv('winequality.csv')
data = [wine_quality['alcohol'], wine_quality['fixed acidity'], wine_quality['quality']]
plt.violinplot(data, showmeans = True)
plt.show()
```



Heat map

Heat map یک روش دیداری برای نمایش داده‌های دو بعدی هســـت. در این روش، هر مقدار داده با یک رنگ متفاوت نمایش داده میشه. واسه ترسیمش باید مراحل زیر رو بریم:

- از مجموعه داده بالا استفاده میکنیم. خط اول کد بالا رو کپی کن.
- از ()corr واســـه ماتریس همبســـتگی اســـتفاده میکنیم. اعددش بین ۱- تا ۱+ هســـتن. یعنی از منفی‌ترین تا مثبت‌ترین یا از قرمز تیره میده تا قرمز روشن.
- فیگور تعریف میکنیم و بهش اندازه میدیم.
- از ()imshow برای نمایش تصویر استفاده میکنیم. بهمون تصاویر مربعی میده.
- ()colorbar تعریف میکنیم که کنار تصویر بهمون یه میله رنگی هم بده.
- Tickها یا اطلاعات روی محور x و y رو بهش میدیم. نوشـــته‌های محور x رو ۲۰ درجه میچرخونیم که تو هم نرن.
- با ()plt.show نمایشش میدیم.

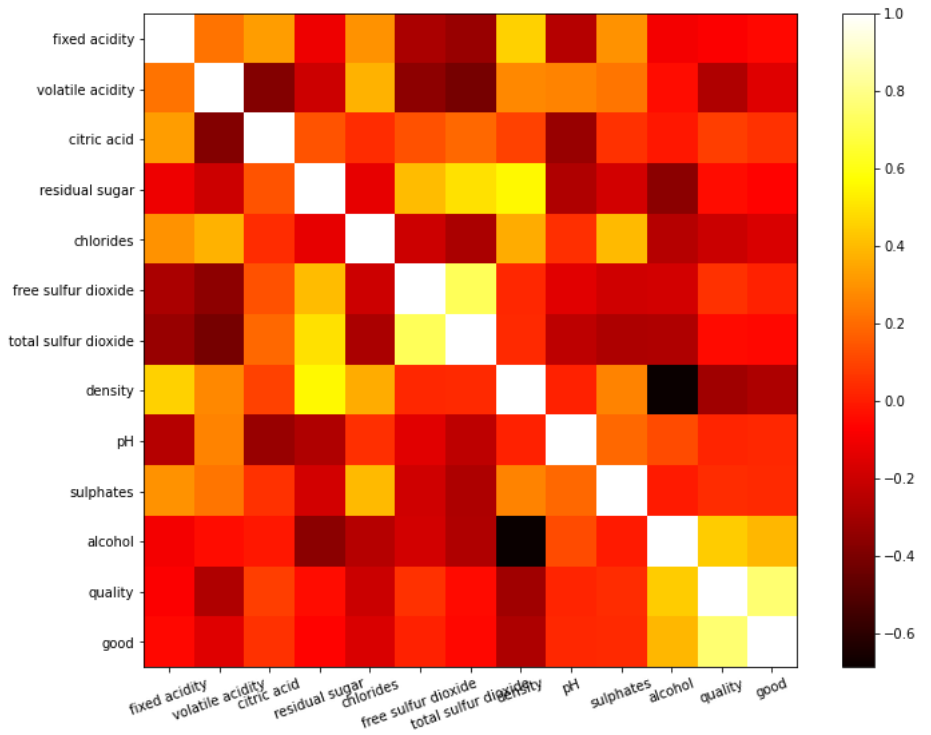
این کد تو ArcGIS Pro خطا میده واسـه همین میبریمش تو نوت بوک ژوپیتر. فقط یادت باشـه که فایل csv رو هم ببری تو مسیر c:\user\nahid کپی کنی.

```
#Hit map
wine_quality = pd.read_csv('winequality.csv')
corr = wine_quality.corr()
plt.figure(figsize=(12,9))
plt.imshow(corr, cmap = 'hot')
plt.colorbar()
plt.xticks(range(len(corr)), corr.columns, rotation = 20)
plt.yticks(range(len(corr)), corr.columns)
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
In  [2]:
Line 3:     corr = wine_quality.corr()

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\pandas\core\frame.py, in corr:
Line 10707: mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\pandas\core\frame.py, in to_numpy:
Line 1892:  result = self._mgr.as_array(dtype=dtype, copy=copy, na_value=na_value)

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\pandas\core\internals\managers.py, in as_array:
Line 1656:  arr = self._interleave(dtype=dtype, na_value=na_value)

File E:\ArcGIS Pro\bin\Python\envs\gisenv\lib\site-packages\pandas\core\internals\managers.py, in _interleave:
Line 1715:  result[rl.indexer] = arr

ValueError: could not convert string to float: 'red'
---------------------------------------------------------------------------
```

در صورت تمایل میتونی نوت بوک تولید شده رو ببری تو مسیر فایلهای پروژه.

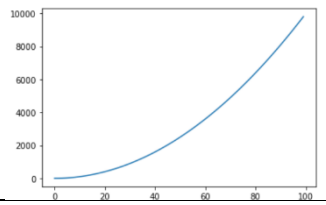
```
wine_quality = pd.read_csv('winequality.csv')
corr = wine_quality.corr()
plt.figure(figsize=(12,9))
plt.imshow(corr, cmap = 'hot')
plt.colorbar()
plt.xticks(range(len(corr)), corr.columns, rotation = 20)
plt.yticks(range(len(corr)), corr.columns)
plt.show()
```



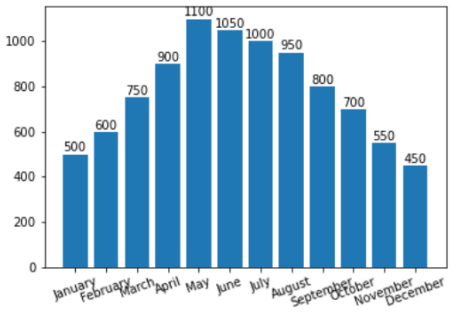
کدهای این بخش در یک نگاه

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
import calendar
%matplotlib inline
```

```
#Line plot
X = range(100)
Y = [value ** 2 for value in X]
plt.plot(X,Y)
plt.show()
```

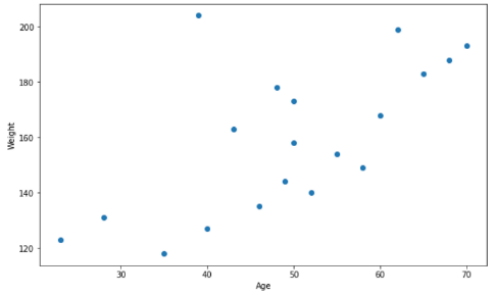


```
#bar plot
month_name = [1,2,3,4,5,6,7,8,9,10,11,12]
units_sold = [500,600,750,900,1100,1050,1000,950,800,700,550,450]
fig, ax = plt.subplots()
plt.xticks(month_name, calendar.month_name[1:13], rotation=20)
plot = ax.bar(month_name,units_sold)
for rect in plot:
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2., 1.002*height, '%d' % int(height), ha = 'center', va = 'bottom')
plt.show()
```



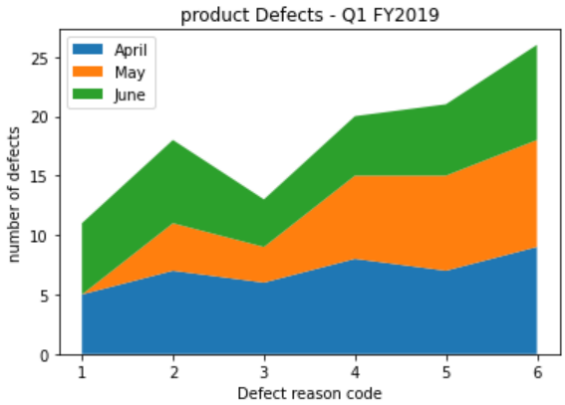
#scatter bar

```
plt.figure(figsize=(10,6))
age_weight = pd.read_excel('scatter_ex.xlsx', 'age_weight')
X = age_weight['age']
Y = age_weight['weight']
plt.scatter(X,Y)
plt.xlabel('Age')
plt.ylabel('Weight')
plt.show()
```



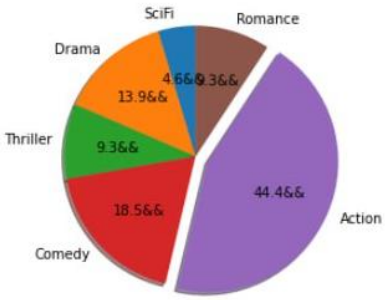
#Stack plot

```
x = np.array([1,2,3,4,5,6], dtype=np.int32)
Apr = [5,7,6,8,7,9]
May = [0,4,3,7,8,9]
June = [6,7,4,5,6,8]
labels = ["April","May","June"]
fig, ax = plt.subplots()
ax.stackplot(x,Apr,May,June, labels=labels)
ax.legend(loc=2)
plt.xlabel('Defect reason code')
plt.ylabel('number of defects')
plt.title('product Defects - Q1 FY2019')
plt.show()
```



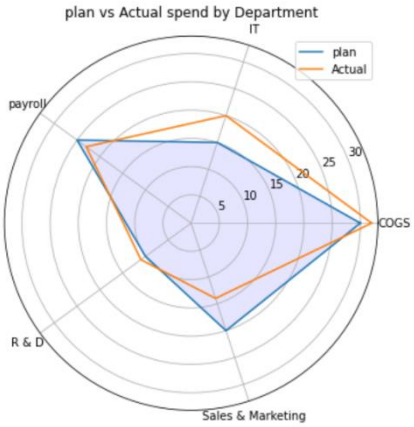
#Pie plot: with Jupyter Notebook

```
labels = ["SciFi","Drama","Thriller","Comedy","Action","Romance"]
sizes = [5,15,10,20,48,10]
explode = (0,0,0,0,0.1,0)
plt.pie(sizes, labels=labels, explode=explode, autopct='%1.1f&&',
      shadow=True, startangle=90)
plt.axis('equal')
plt.show()
```



#polar plot

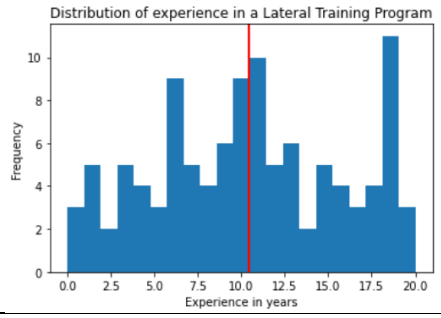
```
Depts= ["COGS","IT","payroll","R & D","Sales & Marketing"]
rp = [30,15,25,10,20,30]
ra = [32,20,23,11,14,32]
theta = np.linspace(0, 2*np.pi, len(rp))
plt.figure(figsize = (10,6))
plt.subplot(polar = True)
(lines, labels) = plt.thetagrids(range(0,360, int(360/len(Depts))),
(Depts))
plt.plot(theta, rp)
plt.fill(theta, rp, 'b', alpha = 0.1)
plt.plot(theta, ra)
plt.legend(labels = ('plan',"Actual"), loc = 1)
plt.title("plan vs Actual spend by Department")
plt.show()
```



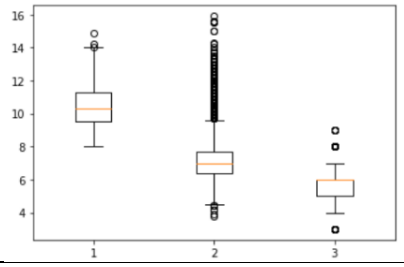
#Hist plot

```
gre_exp = np.array([12,15,13,20,19,20,11,19,11,12,19,13,12,10,6,19,3,1,1,0,4,4,6,5,3,7,12,7,9,8,12,11,11,
18,19,18,19,3,6,5,6,9,11,10,14,14,16,17,17,19,0,2,0,3,1,4,6,6,8,7,7,6,7,11,11,10,11,10,13,13,15,18,20,19,
1,0,8,16,19,19,17,16,11,1,10,13,15,3,8,6,9,10,15,19,2,4,5,6,9,11,10,9,10,9,15,16,18,13])
```

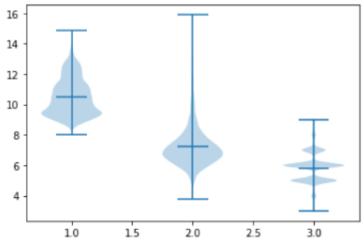
```
nbins = 21
n,bins,parches = plt.hist(gre_exp, bins = nbins)
plt.xlabel("Experience in years")
plt.ylabel("Frequency")
plt.title("Distribution of experience in a Lateral Training Program")
plt.axvline(x=gre_exp.mean(), linewidth = 2, color = 'r')
plt.show()
```



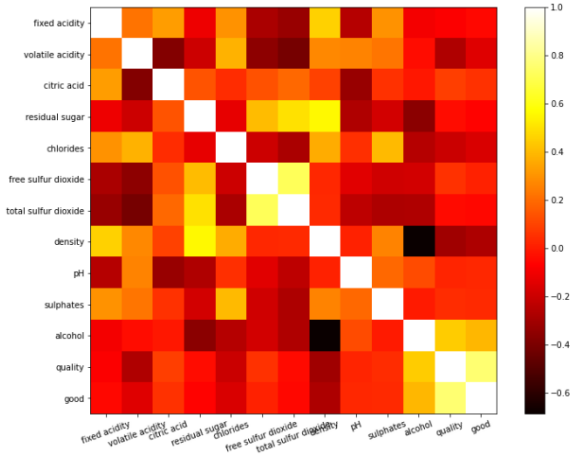

```
#box plot
wine_quality = pd.read_csv('winequality.csv')
data = [wine_quality['alcohol'], wine_quality['fixed acidity'],
        wine_quality['quality']]
plt.boxplot(data)
plt.show()
```



```
#violin plot
wine_quality = pd.read_csv('winequality.csv')
data = [wine_quality['alcohol'], wine_quality['fixed acidity'],
        wine_quality['quality']]
plt.violinplot(data, showmeans = True)
plt.show()
```



```
#Heat map with Jupyter Notebook
wine_quality = pd.read_csv('winequality.csv')
corr = wine_quality.corr()
plt.figure(figsize=(12,9))
plt.imshow(corr, cmap = 'hot')
plt.colorbar()
plt.xticks(range(len(corr)), corr.columns, rotation = 20)
plt.yticks(range(len(corr)), corr.columns)
plt.show()
```



تو جزوه بعدی قرار هست در مورد کتابخونه Seaborn و Squarify با هم یاد بگیریم که باز هم دونستنش واسه علم داده ضرروی هست و ازشون واسه زیبایی بیشتر نمودارها استفاده میشه. یه کتابخونه دیگه هم کنارشون یاد میگیری به اسم missingno که واسه چک کردن داده‌های موجود در یک جدول ازش استفاده میشه. این جزوه رو حسابی تمرین کن که درک جزوه بعدی برات راحت‌تر شه.