MATPLOTLIB

הספרייה matplotlib היא אחת הספריות השימושיות ביותר לצורך הצגת נתונים בגרפים דו-ממדים ותלת-ממדים ובתמונות.
הספרייה היא קוד פתוח ונכתבה ברובה בשפת פייתון למעט חלקים מסוימים בשפה C וב-javascript.
במסמך הבא נציג כמה מהתכונות הבולטות של הספרייה, כמובן שרוחב היריעה קצרה מלהכיל את כל תוכן הספרייה, לכן נמליץ על האתר הרשמי של הספרייה לעוד אינפורמציה אודותיה.

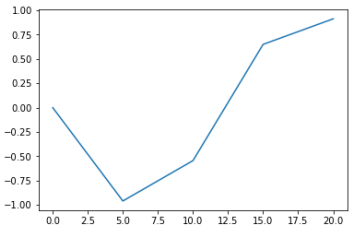
התקנה:

```
python -m pip install -U pip
python -m pip install -U matplotlib
```

- line chart

ברוב הפעמים כשנצטרך תבנית קווית למשל הצגה של פונקציה דו ממדית, נשתמש במרחב השם pyplot וכמוסכמה מייבאים אותו בשם plt.
בדיוק כמו גרף דו ממדי, שמצויין בציר x וציר y ,יש לציין את ערכי ה-x וה-y של הגרף, לרוב x ייצג את המקור ו-היא פונקציית התמונה. לאחר שהגדרנו אותם נשתמש בפונקציה plot(x,y) כדי להגדיר אותם.
בחלק מהide יש צורך גם בפונקציה show() כדי לראות את התמונה על המסך.
בדוגמא הבאה נשתמש בפונקציה linspace(start , end ,how_many) שמחזירה מערך עם כמות מספרים (how_many) ובטווח שבין start ל-end וכל המספרים במרחק שווה אחד מהשני:

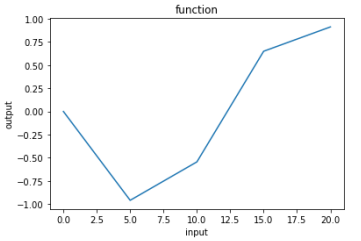
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0,20,5)
y= np.sin(x)

plt.plot(x,y)
plt.show()
```



אם נרצה להוסיף גם תוויות לצירים וכותרת נוכל להשתמש בפונקציות
xlabel() ylabel() ו-title() שמקבלות מחרוזת ומדפיסות אותן למסך.
יכול להיות בהדפסה יוסף לנו איזשהו כיתוב שלא קשור לתמונה, זה משום שג'ופיטר
מוסיף את הפעולות שקוראות בקונסול ביחד עם הפעולה שנעשת עם הפונקציה plot(),
במקרה כזה אפשר לכתוב ; בסוף השורה, או להשתמש בפונקציה show() כדי לא לראות את ההדפסה:

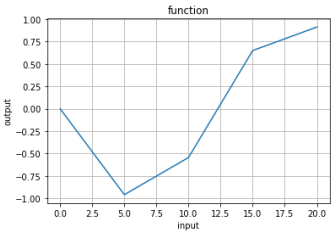
```
plt.xlabel("input")
plt.ylabel("output")
plt.title("function")
```

```
plt.plot(x,y);
```



אפשר גם להוסיף רשת מאחורה אם נשתמש בפונקציה grid() לפני ההדפסה:

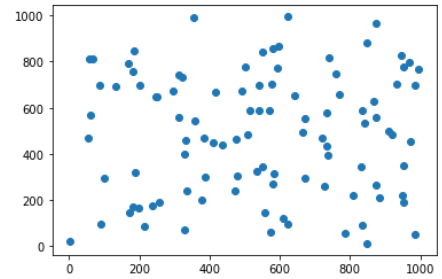
```
plt.xlabel("input")
plt.ylabel("output")
plt.title("function")
plt.plot(x,y)
plt.grid()
plt.show()
```



- Scaterpoint

אוסף של נקודות מציינים עם הפונקציה scatter(X,Y), הפונקציה מקבלת אוסף של נקודות עם ערכי x ו-y ומציירת אותם על הלוח. נראה דוגמא: נשתמש בפונקציה random.randint(row,col , size) כדי לקבל מערך דו ממדי עם 1000 שורות ושני עמודות (x ו-y) של ערכים שלמים רנדומליים, ונצייר אותם על הלוח:

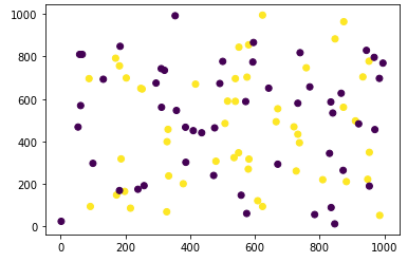
```
data = np.random.randint(1,1000 , size = (100,2))
plt.scatter(data[:,0] , data[:,1])
plt.show()
```



יכול להיות שנרצה לצבוע את הנקודות. בשביל לצבוע אותן נצטרך מערך חד ממדי בגודל של מספר הנקודות בגרף, אח"כ נצטרך להגדיר ערך עבור כל נקודה בגרף ולתת לה מספר, כל מספר ייוצג בצבע שונה.
למשל ניקח את המערך שלנו ונקבע שכל 50 הנקודות הראשונות בגרף יהיו בצבע אחר מהאחרות.
ניצור מערך color שמלא באפסים ונגדיר לו שמהנקודה אפס ועד 50 יקבלו את הערך 1 וכל השאר ישארו 0:

```
colors = np.zeros(100)
```

```
colors[:50] = 1
plt.scatter(data[:,0] , data[:,1] , c = colors)
plt.show()
```



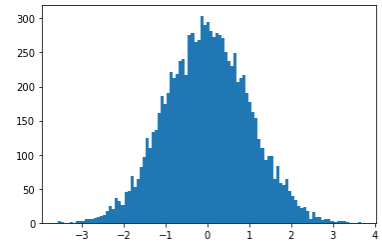
– Histogram

היסטוגרמה הוא גרף עמודות. בדר"כ משתמשים בו כדי להציג התפלגויות של ערכים.
בשביל להגדיר גרף בצורה של היסטוגרמה נשתמש בפונקציה hist() שכל מה שהיא מקבלת זה מערך דו ממדי של ערכים.
אם נרצה להוסיף עמודות לגרף, כלומר להציג אותו יותר "מעוגל" נוכל להשתמש בפרמטר bins ולהגדיר לו מספר עמודות.

בדוגמא הבאה נראה את ההבדל בין ההתפלגויות השונות, נבנה מערכים של עשרת אלפים נקודות בצורה רנדומלית לפי הפונקציה הבסיסית random, התפלגות נורמלית, התפלגות פואסונית וכו'.

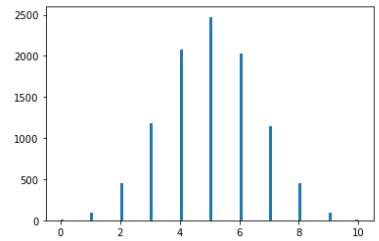
```
print('~~~ normal distribution: ~~~')
normal_dist = np.random.normal(size=10000)
plt.hist(normal_dist, bins = 100)
plt.show()
```

```
~~~ normal distribution: ~~~
```



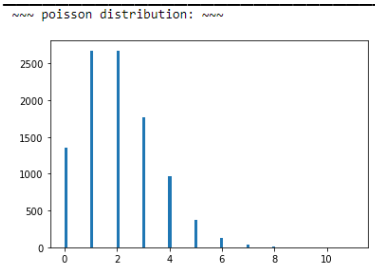

```
binomial_dis = np.random.binomial(n=10, p=0.5, size=10000)
print('~~~ binomial distribution: ~~~')
plt.hist(binomial_dis, bins = 100)
plt.show()
```

```
~~~ binomial distribution: ~~~
```

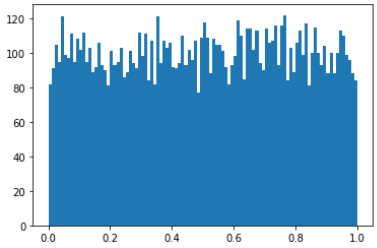



```
poisson_dis = np.random.poisson(lam=2, size=10000)
print('~~~ poisson distribution: ~~~')
plt.hist(poisson_dis, bins = 100)
```

```
plt.show()
```




```
random_array = np.random.random(10000)
plt.hist(random_array, bins = 100)
plt.show()
```



תמונות-

טעינה של תמונה הוא דבר שנפוץ בעיקר בתחום של ראייה ממוחשבת, אבל לא רק.
בשביל לטעון תמונה נצטרך כמה דברים: דבר ראשון נצטרך להשתמש במרחב שם שונה של הספרייה שהוא image כדי לא להתבלבל ניתן לו את השם mpimg .
שלב שני יהיה לייבא תמונה, האפשרות הפשוטה היא לייבא תמונה שיושבת כבר במחשב, אבל לפעמים נרצה דווקא תמונה מהרשת, במקרה כזה נוכל להשתמש בספרייה urllib שכבר מותקנת בכל פייתון, ומאפשרת עם הפונקציה request.urlretrieve(url ,name) לעשות wget על כתובת מסוימת ולהוריד את התוכן של האתר. הפונקציה מקבלת url וניתן להוסיף פרמטר עם שם שנרצה לקרוא לקובץ שהורד.
אם נרצה להוריד תמונה נצטרך פשוט לשלוח את הurl של התמונה והפונקציה תוריד לנו אותה לתיקייה שציינו ב-name, לאחר שעשינו את זה נוכל להטעין את התמונה עם הפונקציה imread(address) של matplotlib.image, שמקבלת כארגומנט את הכתובת של התמונה ששמורה לנו לוקלית:

```
import matplotlib.image as mpimg
import urllib

url_str = "https://i.pinimg.com/736x/9d/f1/f8/9df1f82852b0d020ccf6430c17b8ce36.jpg"
urllib.request.urlretrieve(url_str, "python.jpg")

img = mpimg.imread('python.jpg')
```

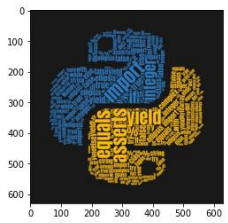
המשתנה img שקיבלנו הוא מערך, אם נדפיס אותו נגלה שכל הערכים שלו הם מספרים שלמים, שערכם לא עובר את 255 וזה כי הטיפוס של הערכים בו הם uint8 כלומר מספרים טבעיים שערכם המקסימלי הוא 2^8 (8 ביטים).
ואם נבדוק את המבנה של המערך עם הפונקציה shape נגלה דבר מעניין:

```
img.shape
```

```
(630, 630, 3)
```

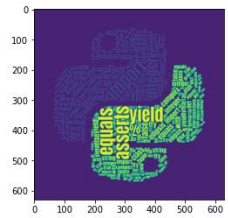
זהו בעצם מערך תלת ממדי שהערכים שלו הם גובה אורך (בפיקסלים) ועוד משהו עם שלושה ערכים
הממד השלישי הזה הוא בעצם כמות הצבע של כל פיקסל בתמונה לפי rgb -כל צבע בנוי משלושה צבעי יסוד אדום, ירוק וכחול. הממד השלישי הזה אומר לנו כמה מרוכז כל צבע בתוך הפיקסל.
בשביל לראות את התמונה נצטרך להשתמש בפונקציה נצטרך להשתמש בפונקציה imshow() של מרחב השם pyplot :

```
plt.imshow(img)
plt.show()
```



לפעמים לא נרצה בכלל צבעים בתמונה כי הם תופסים הרבה מקום שלא בטוח שיש לנו.
במקרה כזה נוכל להוריד את הממד שלישי מהתמונה פשוט:

```
lum_img = img[:, :, 0]

plt.imshow(lum_img)
plt.show()
```



בתמונה החדשה שקיבלנו הצבעים הם מוגדרים לפי הספרייה matplotlib ואינם איזשהו סטאנדרט.
הצבעיים מיצגים את ה"תאורה" בתמונה, כלומר כמה הצבעים באותה נקודה חמים(אדמדמים) או קרים(כחולים).
אפשר גם להציג את אותה התמונה בצבע שחור לבן (גווני אפור), אם נשתמש בארגומנט cmap של הפונקציה ונגדיר לו את הגוון שאותו נרצה:

```
plt.imshow(lum_img, cmap = 'gray')
plt.show()
```