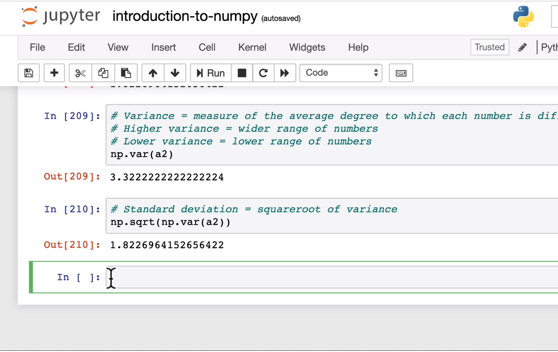
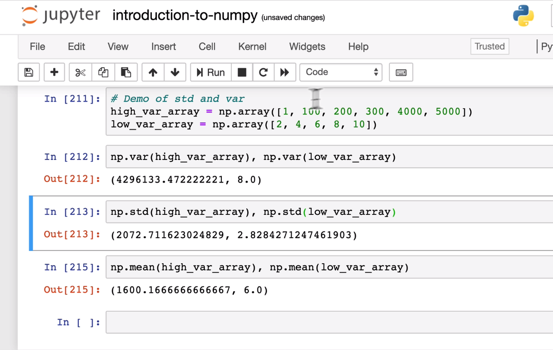
**Np mean,variance,std deviation**



Standard deviation = square root of variance.



**Variance and standard deviation are measuring the spread of data.**



**histogram**

%matplotlib inline

Import matplotlib.pyplot as plt

Plt.hist(high\_var\_array)

Plt.show()

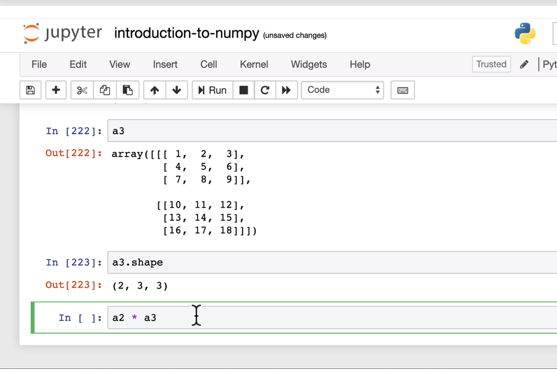
**Reshape and Transpose:**

a2

a2.shape

a3

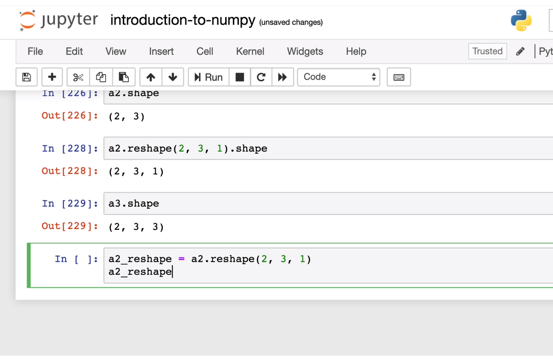
a2 \*& a3



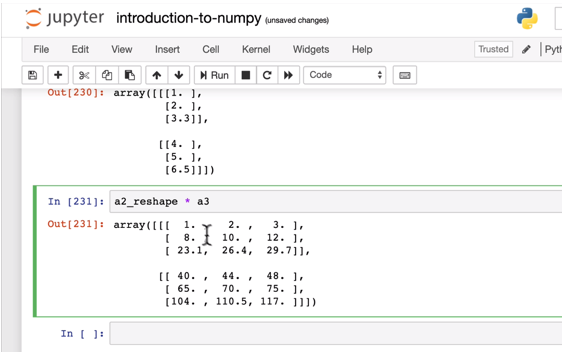
**Reshape**

**We need to reshape based on broadcasting rules:**

**Two elements are equal or one of them has to be 1**



Now the shape matched .

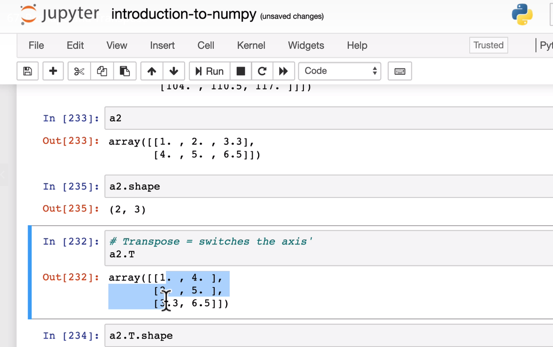


**Transpose:**

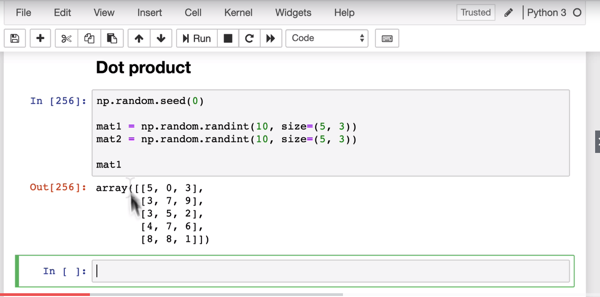
**Transpose switches/swaps the axis**

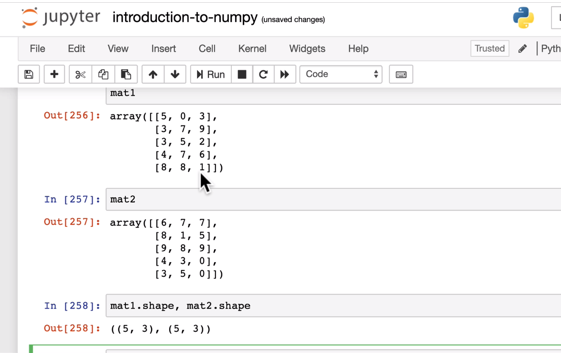
A2.T

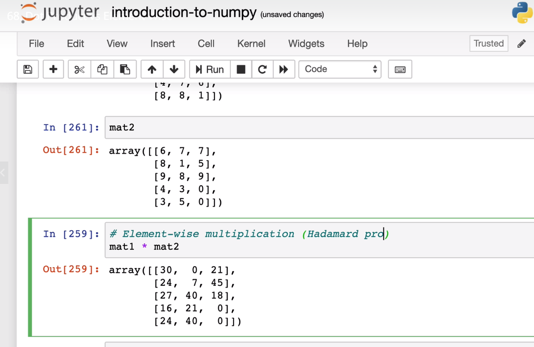
A2.T.shape



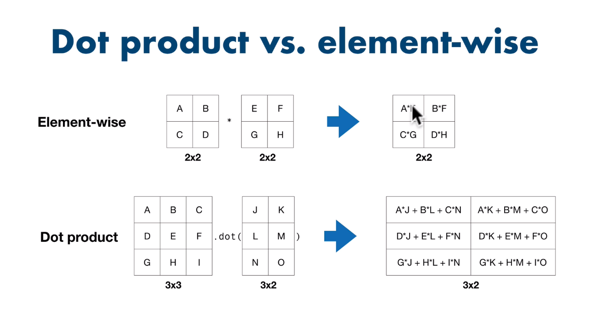
**Dot Product**

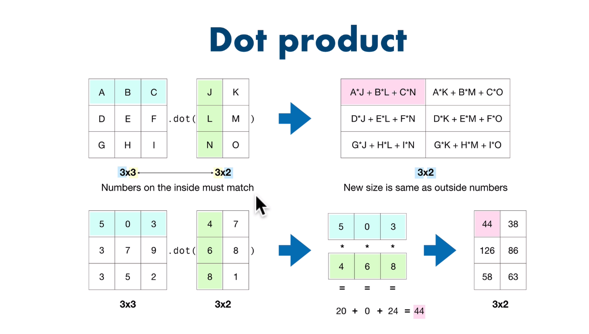


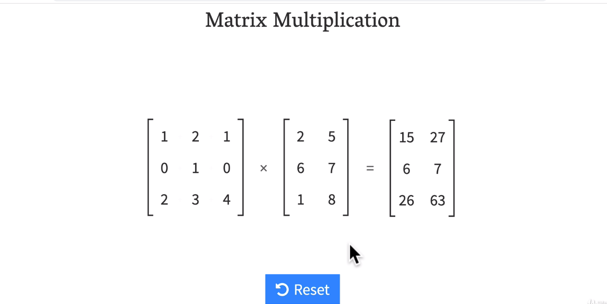




Np.dot(mat1,mat2)

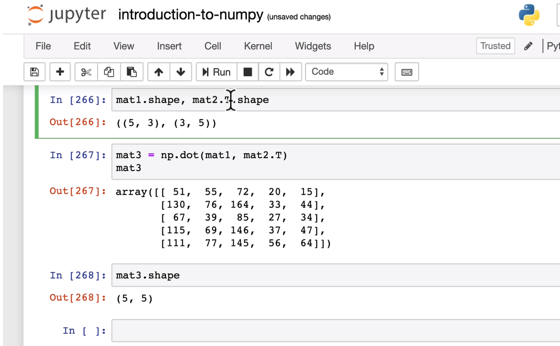




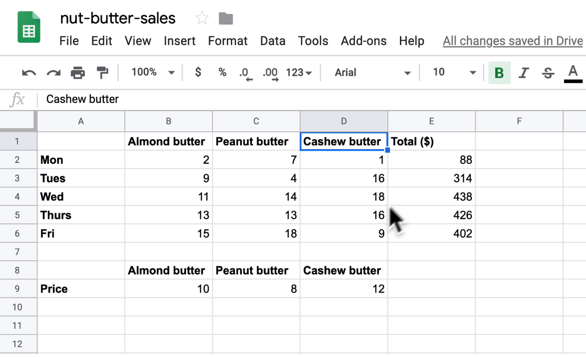




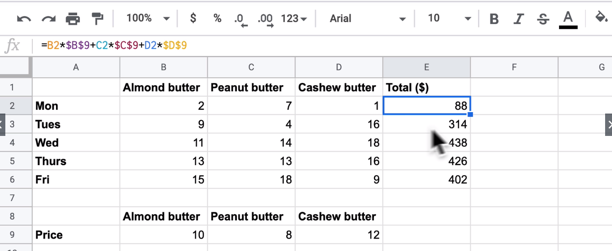
**Dot product works after Transposing:**



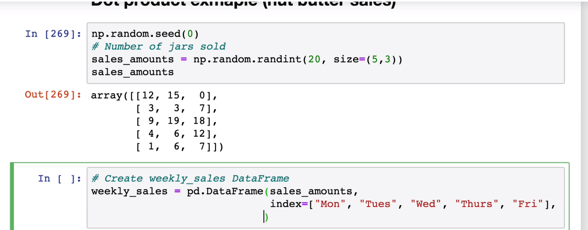
**Nut Butter sales exercise:**

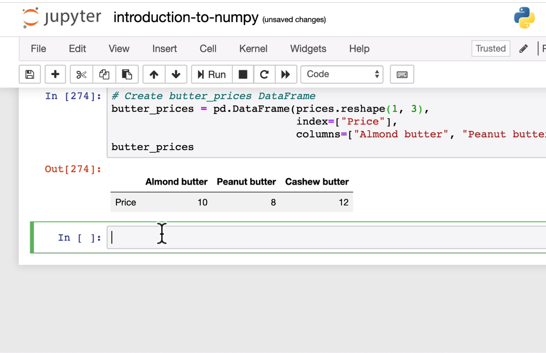


**Dot product sales formula:**

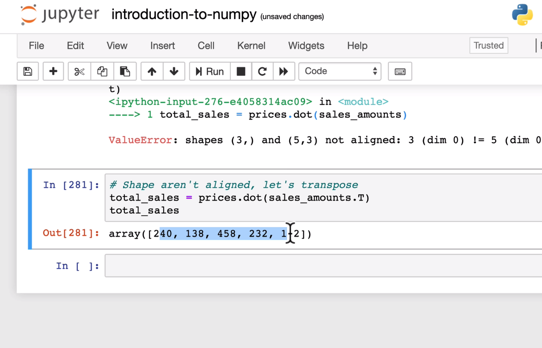


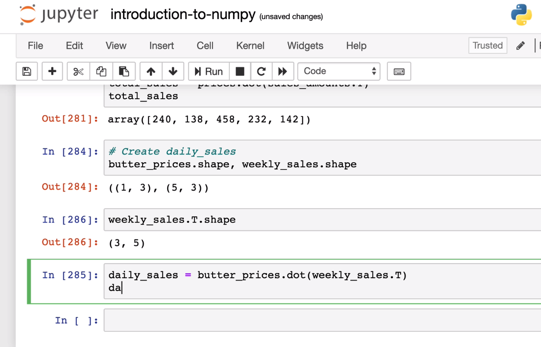
**Lets do the same using numpy dot product**

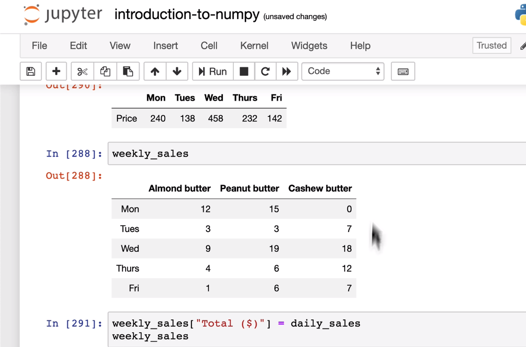


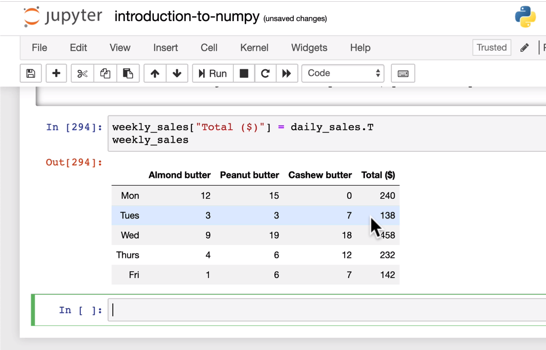


**Total \_Sales:**



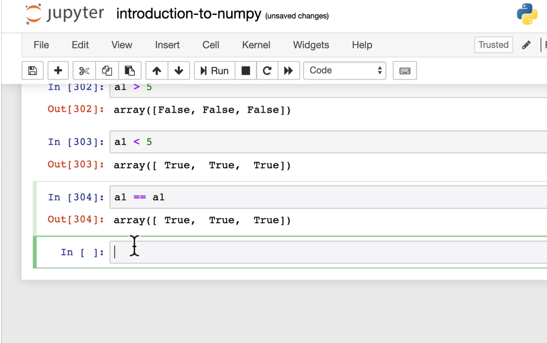




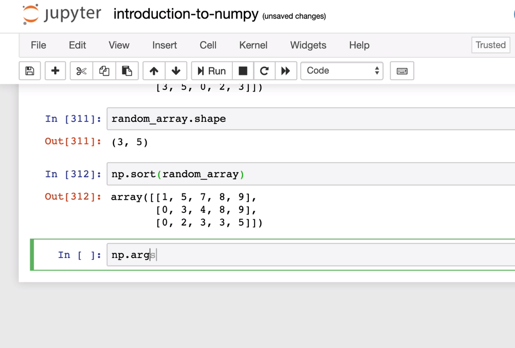


**Comparison Operators:**

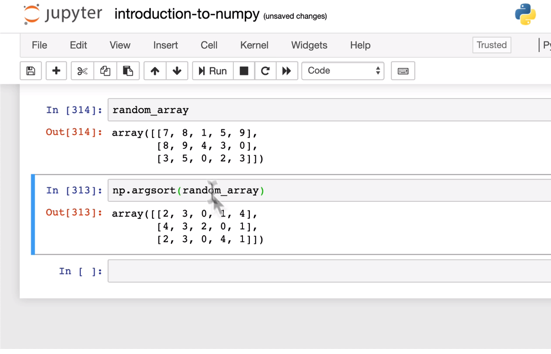
A1 > a2



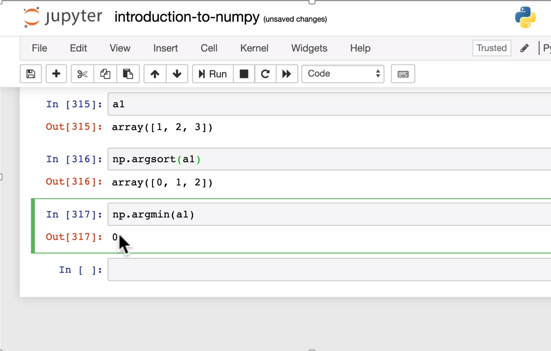
**Numpy sorting:**

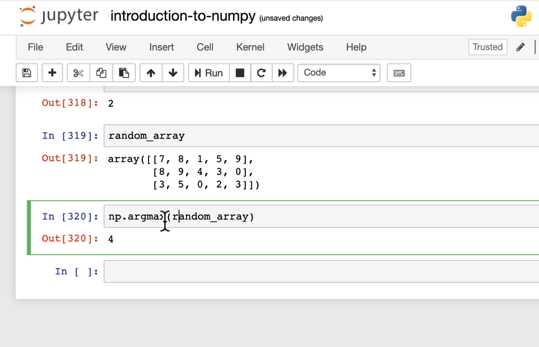


**Argsort array ( sorting by indexes)**



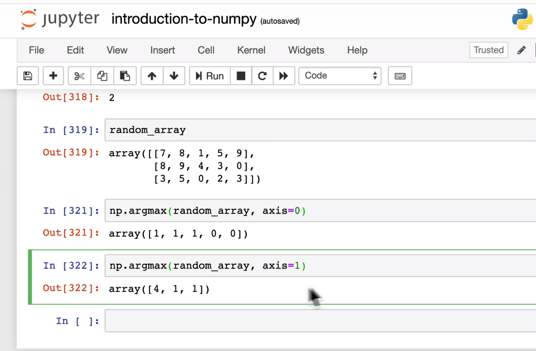
**Argmin and argmax**





**Argrmax for columns =axis 0**

**Argmax for rows = axis1**

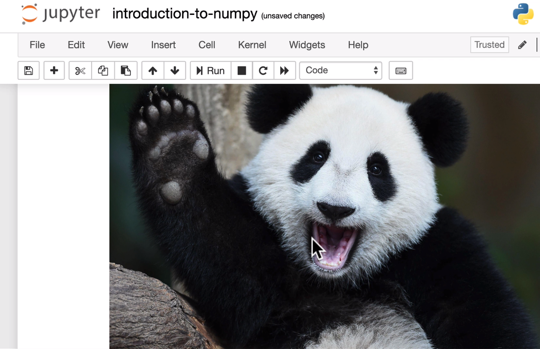


**Example of convert of photo to numbers**

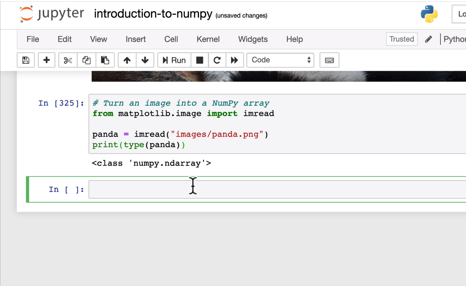
**Through numpy:**

**To import an image:**

<img src=”images/panda.png”/>



**Now lets turn this image into numpy array:**

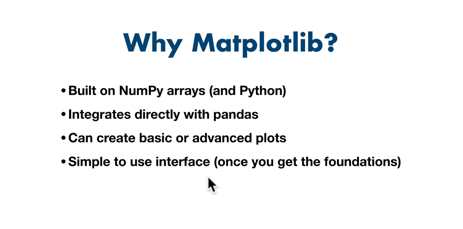


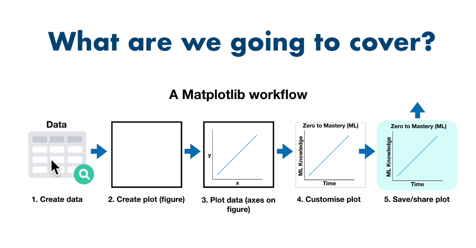
**To get first 5 arays from the picture:**

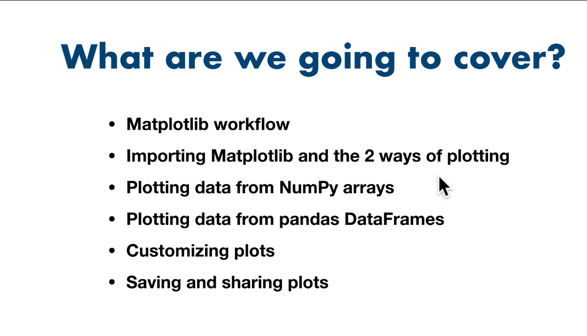


Imread function can read a photo and turn into a numpy ndarray and that is what we use in machine learning.

**Matplotlib:**

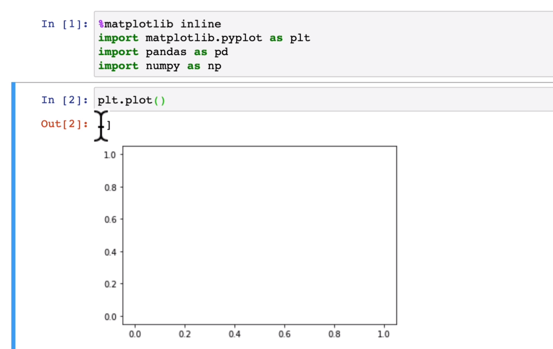


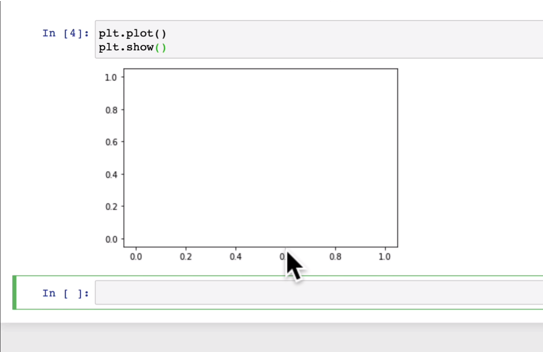


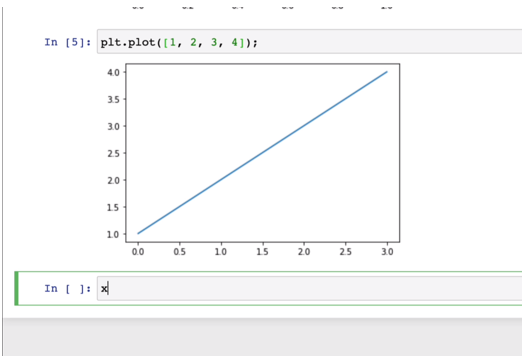


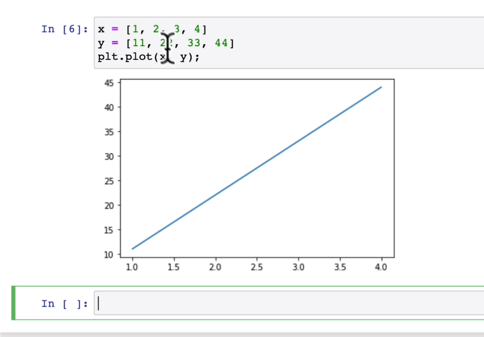
**Importing and using matplotlib**

%matplotlib inline **🡪 This will say display the visuals with in jupyter notebook**

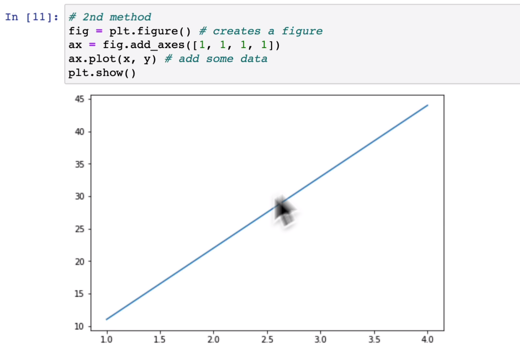




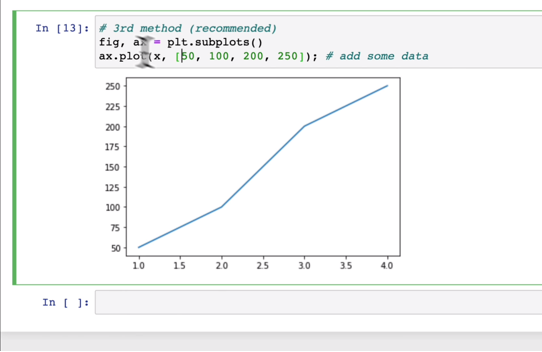




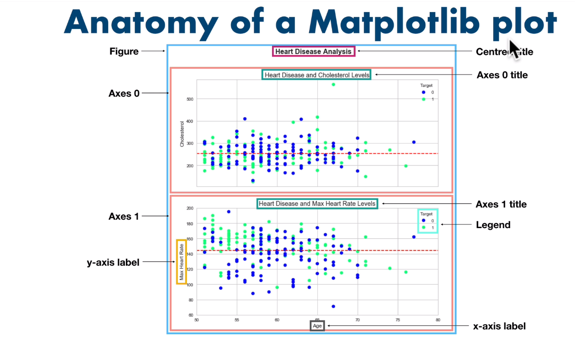
**2nd method:**



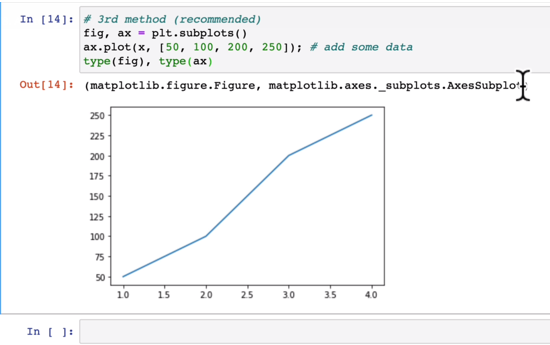
**3rd method: (Recommended)**



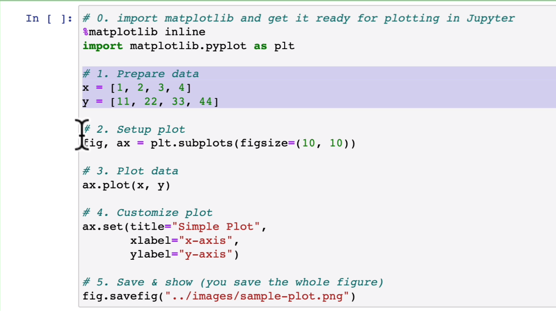
**Matplotlib anatomy:**



**Object oriented API:**



**Creating a plot:**



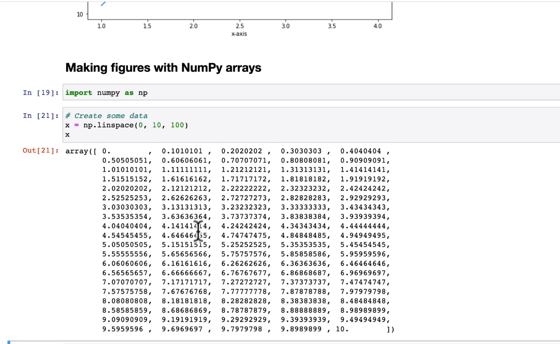


**Scatter plot and bar plot:**

**Making figures with numpy arrays:**

Import numpy as np

X = np.linspace(0,10,100)



**#plot the data**

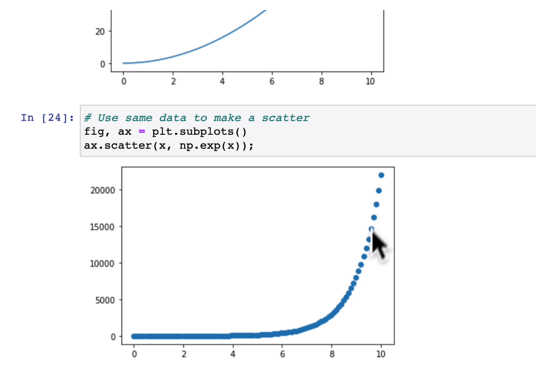
Fig, ax = plt.subplots()

Ax.plot(x,x\*\*2);

**Scatter plots:**

Fig, ax = plt.subplots()

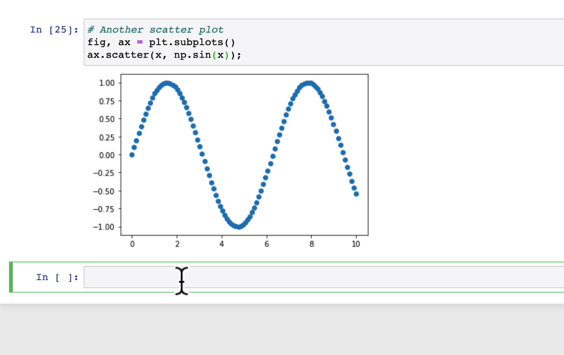
Ax.scatter(x,np.exp(x));



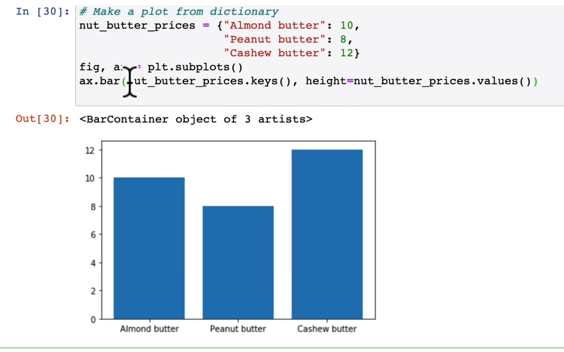
#Another scatter plot:

Fig, ax = plt.subplots()

Ax.scatter(x,np.sin(x)); ( Like SineTheta in trigonometry)



**#Make a plot from Dictionary:**

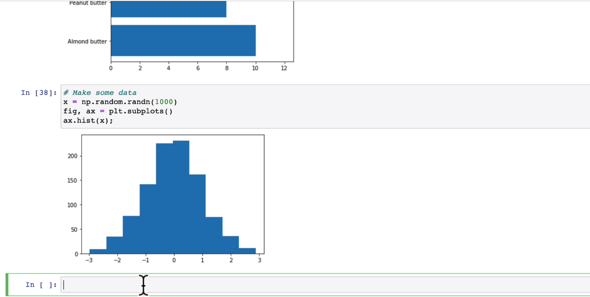




**Horizontal bar:**

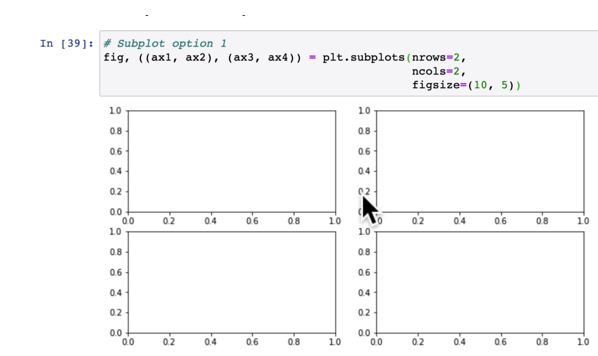


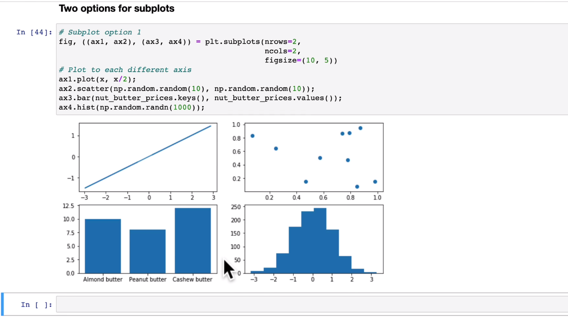
**Histograms:**



**Two options for subplots:**

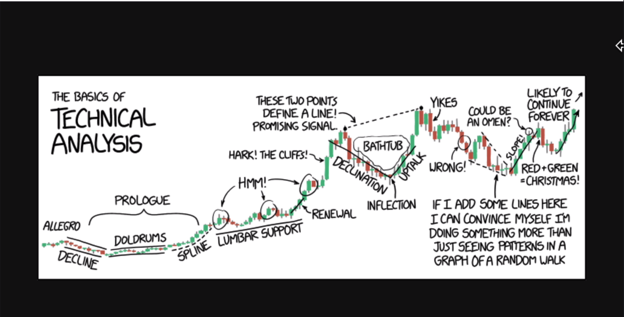
**#option1**



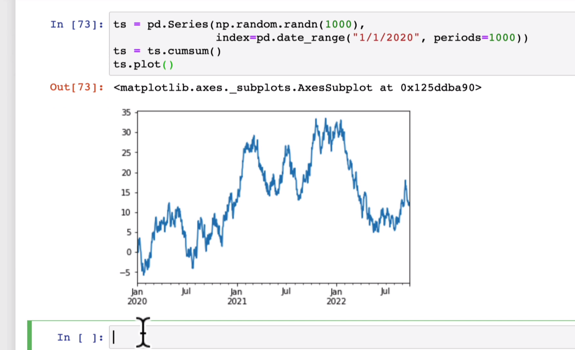


**Subplots option2:**

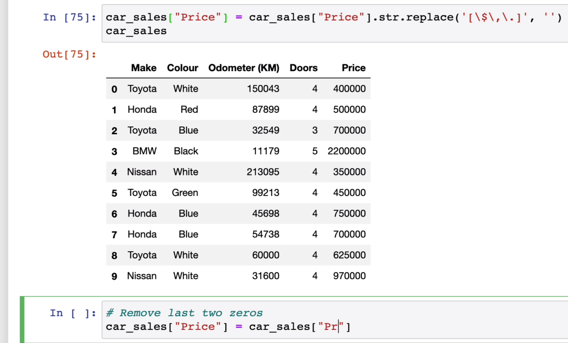




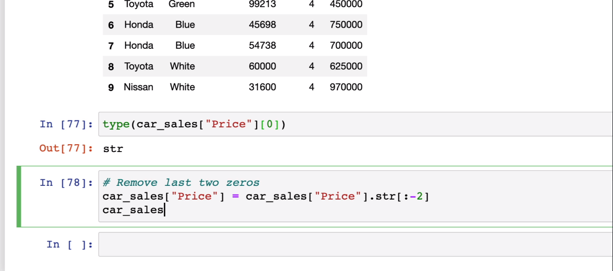
**Plotting from pandas dataframes:**

**Plotting from series:** 

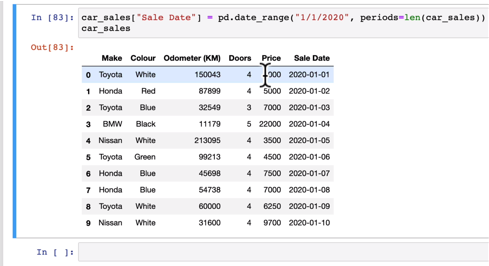
**Plotting from pandas dataframe:**

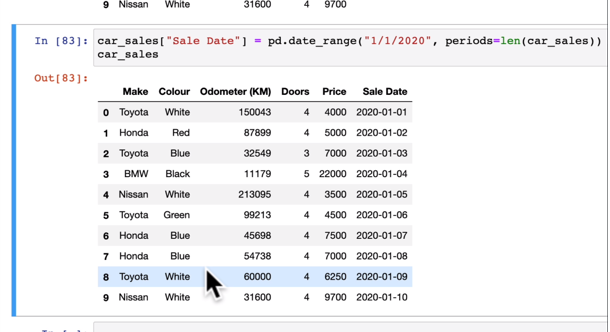


Add regex and change dollar to string



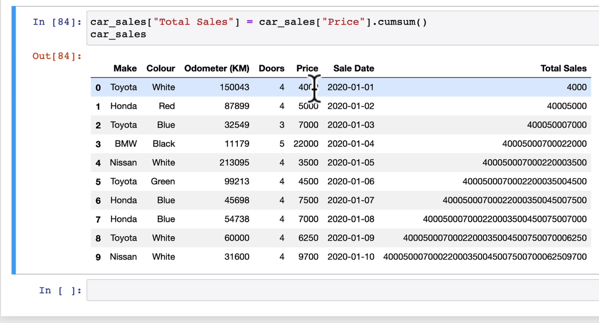
**Add a column:**

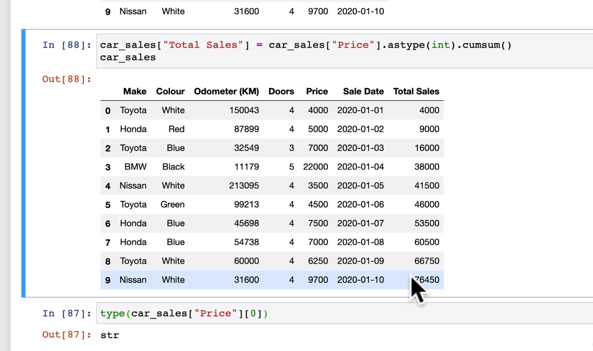




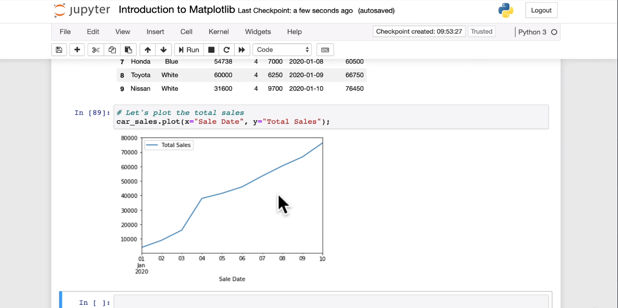
**Cumsum() : To have car sales total sales:**

**Since it is still string it did not work properly so we need to change it to integer and then do the cumsum**

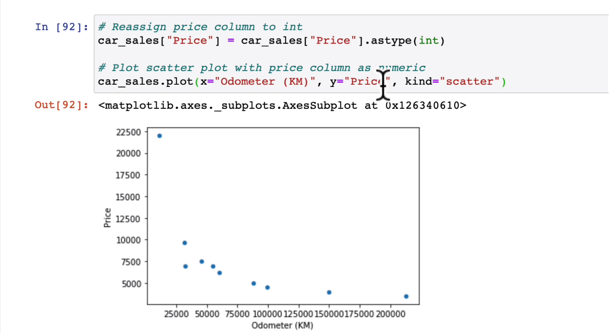




**Let’s plot the total\_sales:**



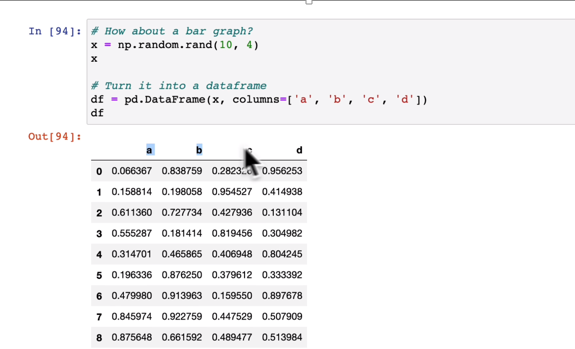
**Scatter plot:**



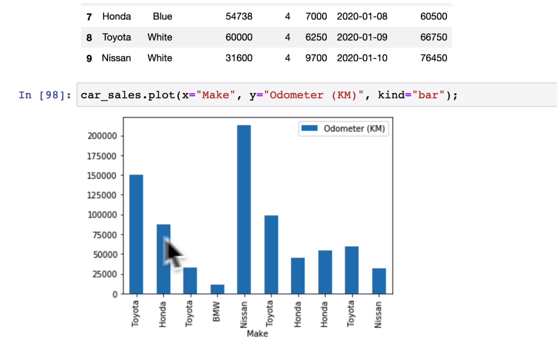
**Bar Graph using pandas datafrme:**

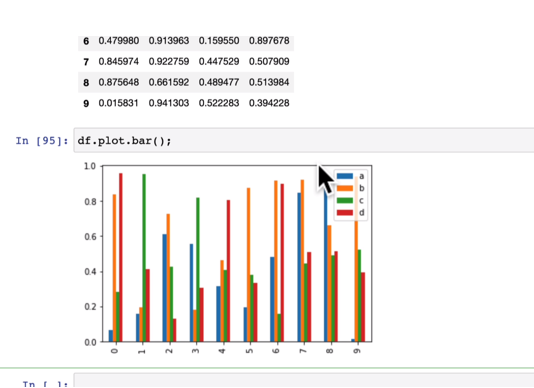
**One more method is**

**Df.plot(kind=”bar”);**

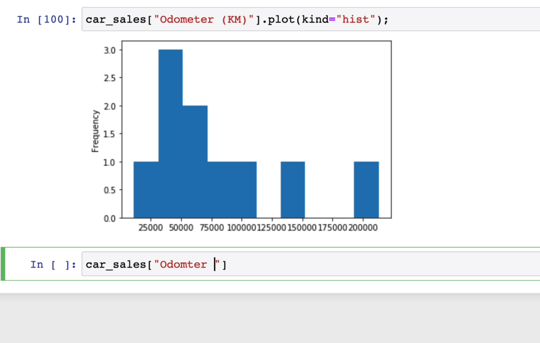


**Bar chart for a particular columns:**

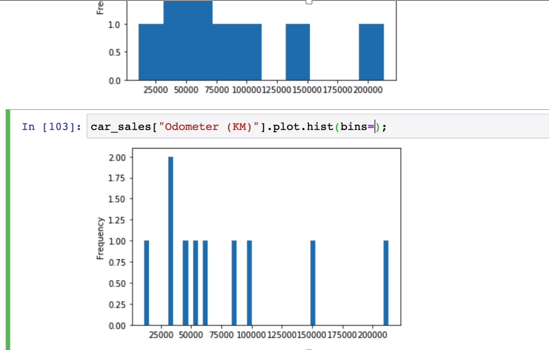




**Histograms are great for dpread of data visualization:**

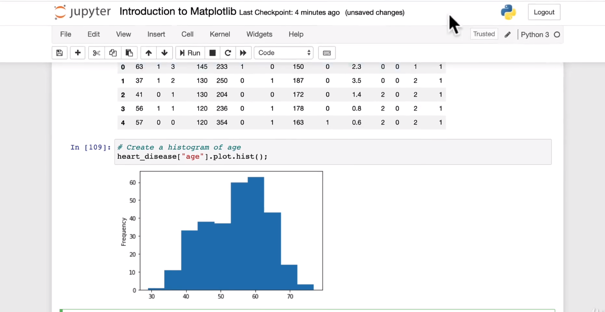


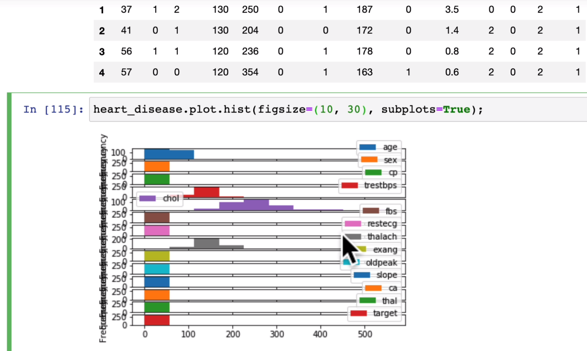
**Bins give group separation in visualization and choosing idealistic bins for our graph:**





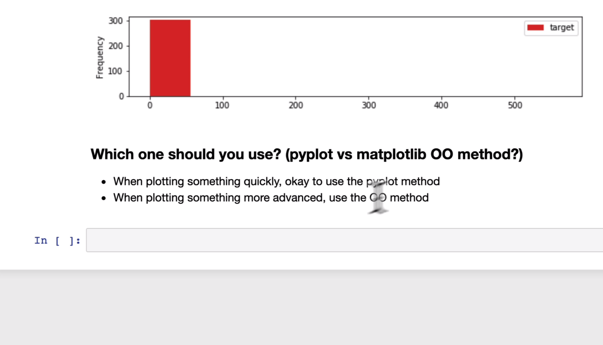
Lets do histogram on this based on distribution of age column:

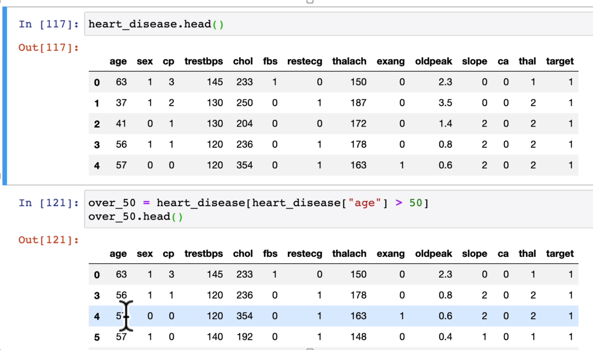




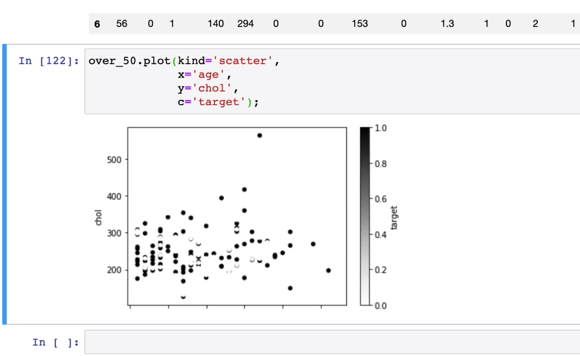
**Until now we were using direct numpy method of pyplot**

**Plotting using Object oriented API**



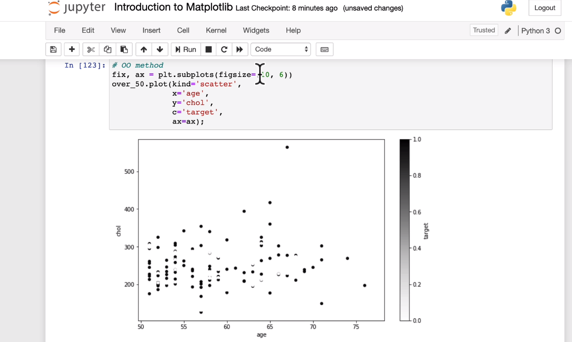


**Pyplot method:**



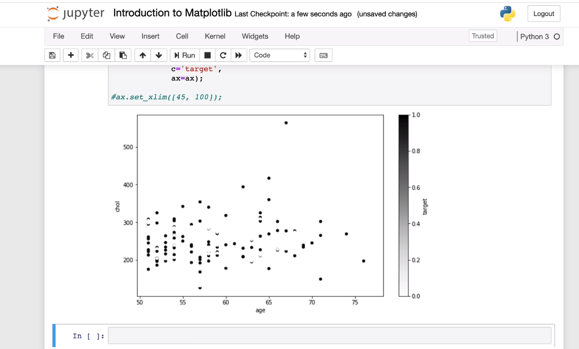
**# OO method:**

**Ax method:**

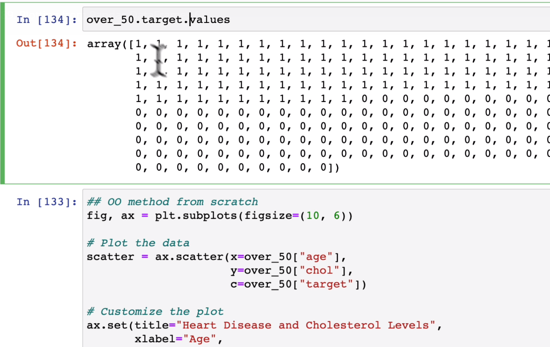


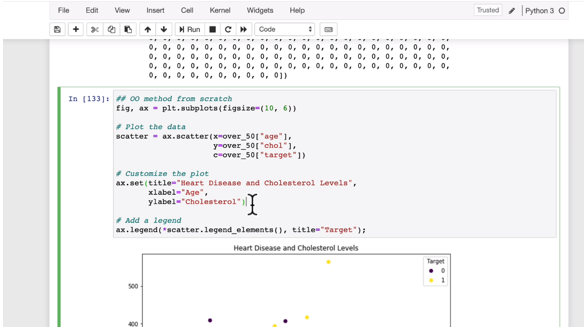
**We can set axis limits:**

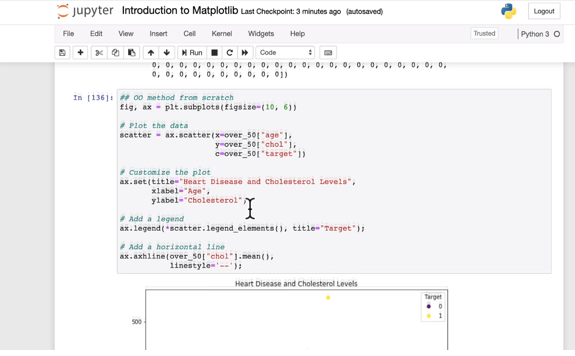
**Ax.set\_xlim([45,100]);**

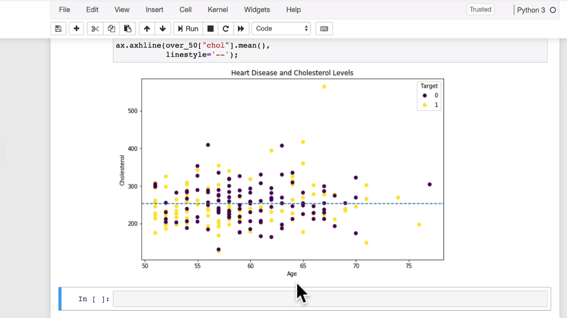


**Object Oriented method from scratch:**



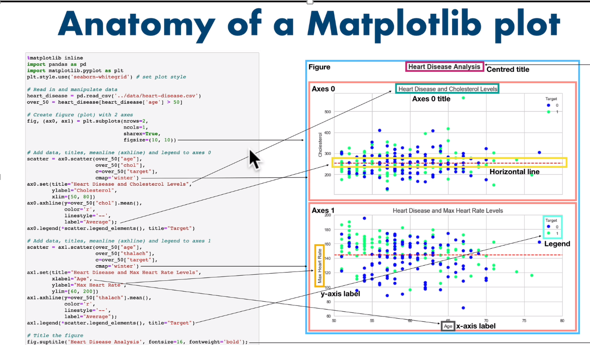


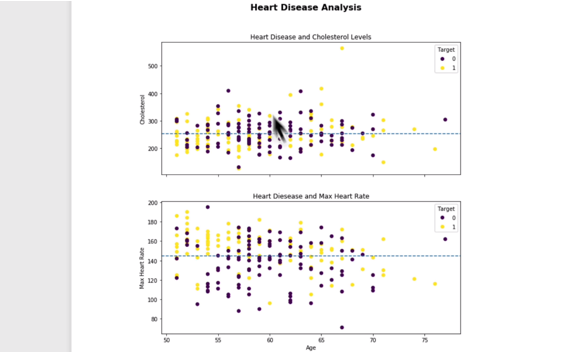




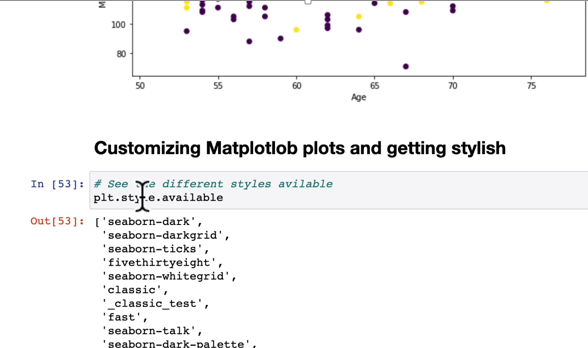
**Now let’s create a subplot:**

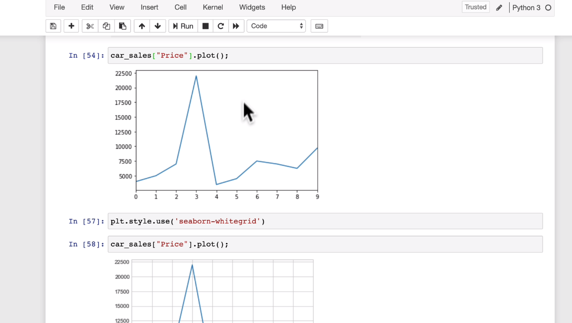
Below is the entore code to bring out the object oriented plot using common x axis and using mean value:



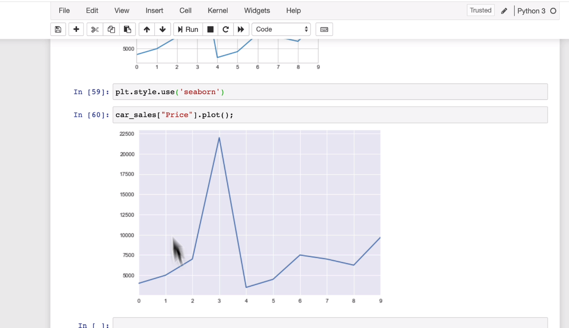


**Let’s customize the plots:**

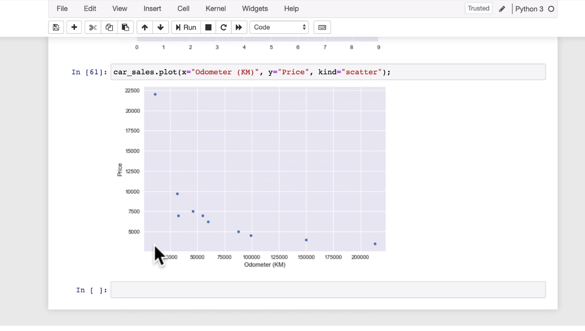




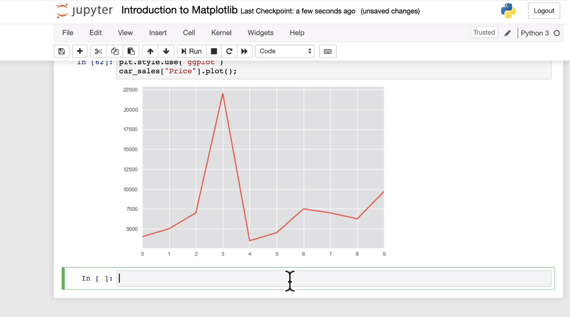
Another appearance style:



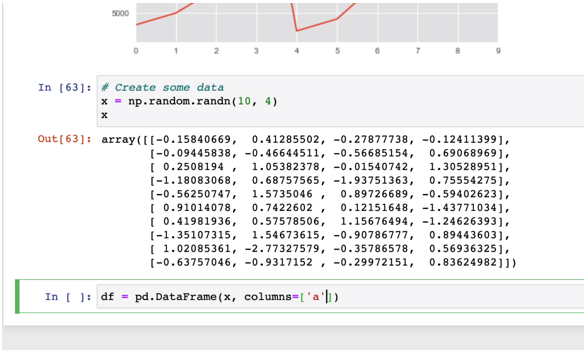
**Using plot like below:**

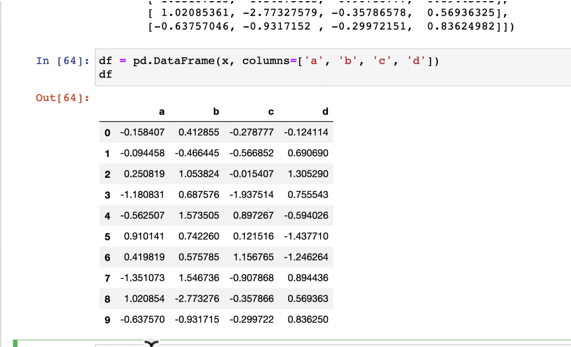


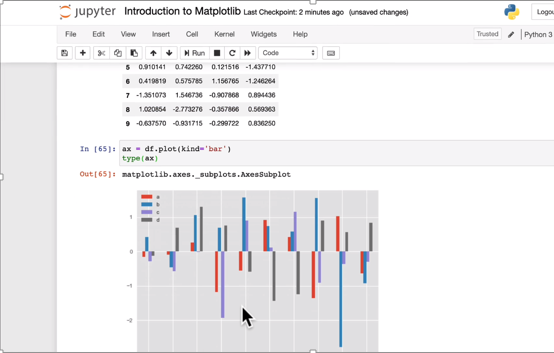
**Ggplot()**



**Let’s create some data set:**



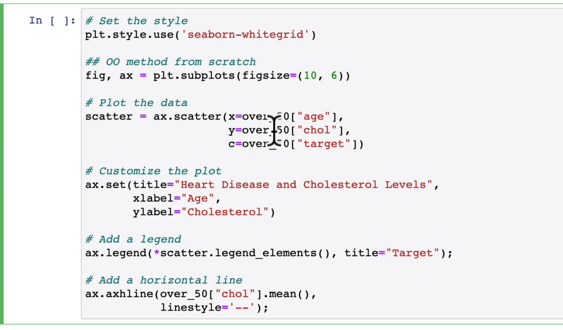


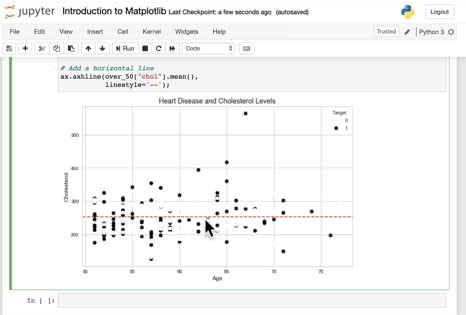


**Add legend,title etc:**



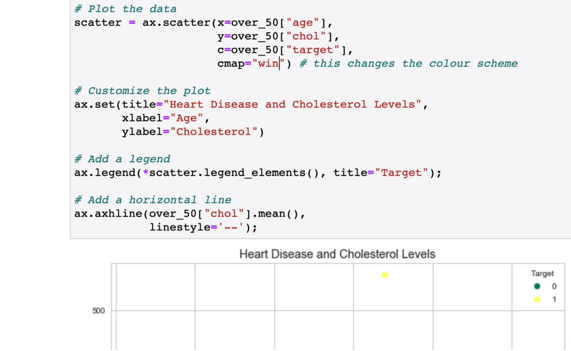
**Plotting subplot using OO method with different style:**



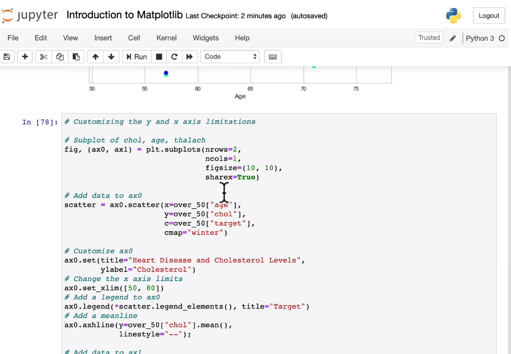


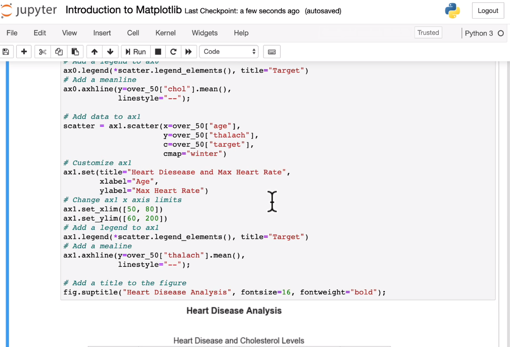
**Use of cmap:**

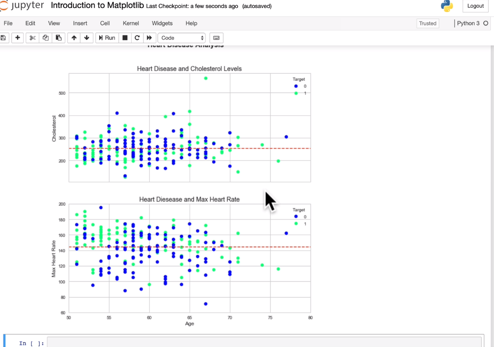
**Google Matplotlib colors tutorial:**



**Setting xlim and ylim and also cmap**



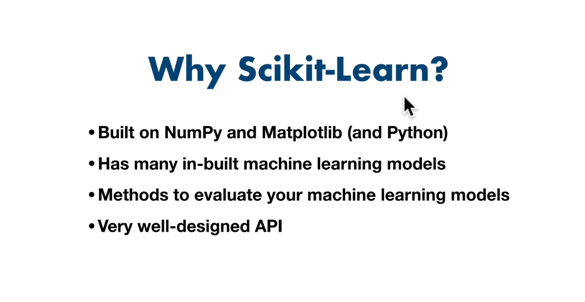




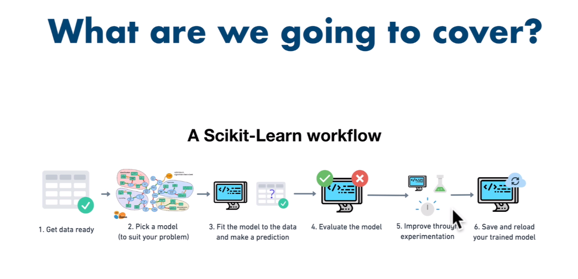
**Now saving the figure**

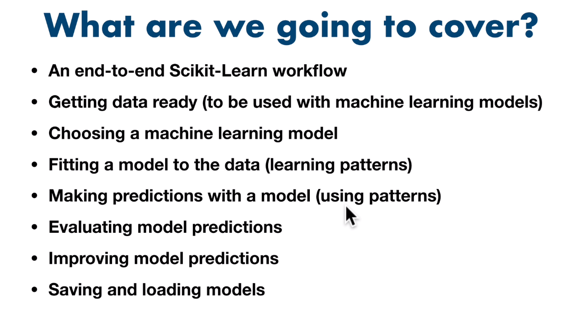
Fig.savefig(“heart-disease.png”)

**Scikit-Learn:**

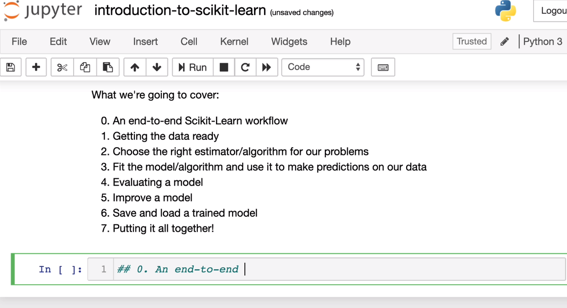


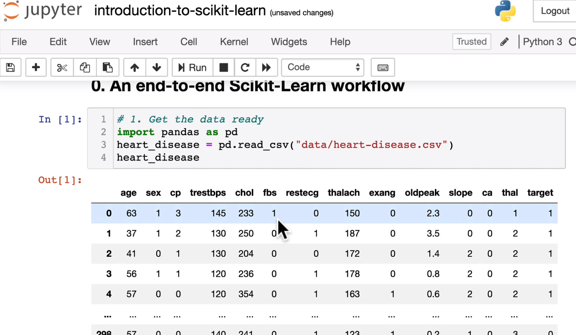
**Scikit\_Learn workflow:**



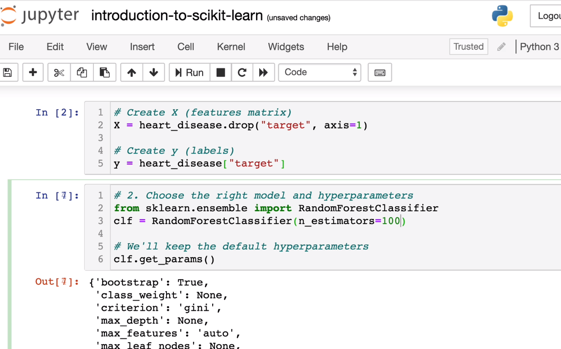


**Let’s do this through heart-disease project data:**



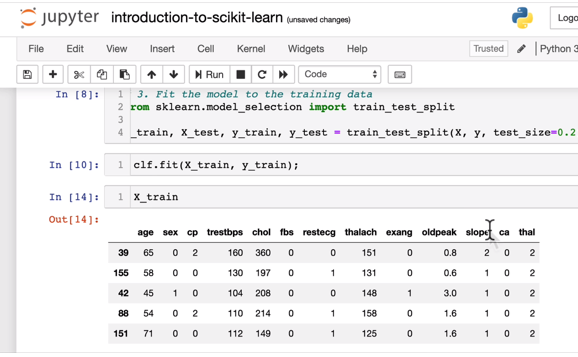


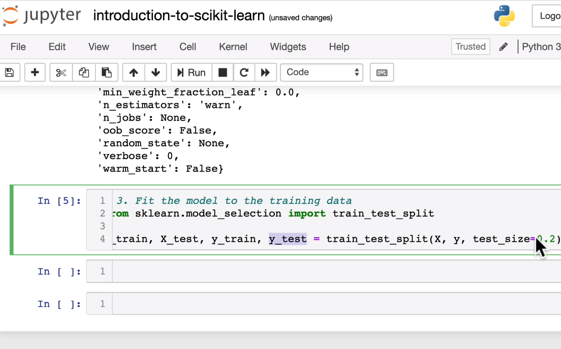
from sklearn.ensemble import RandomForestClassifier



**Now split the data and make a prediction:**

**Now fit the data and make sure the shapes are equal.**





**Let’s predict the model**

