import matplotlib.pyplot as plt  
import numpy as np  
from sklearn import datasets, linear\_model, metrics  
  
# load the boston dataset  
boston = datasets.load\_boston(return\_X\_y=False)  
  
# defining feature matrix(X) and response vector(y)  
X = boston.data  
y = boston.target  
  
# splitting X and y into training and testing sets  
from sklearn.model\_selection import train\_test\_split  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.4,  
 random\_state=1)  
  
# create linear regression object  
reg = linear\_model.LinearRegression()  
  
# train the model using the training sets  
reg.fit(X\_train, y\_train)  
  
# regression coefficients  
print('Coefficients: ', reg.coef\_)  
  
# variance score: 1 means perfect prediction  
print('Variance score: {}'.format(reg.score(X\_test, y\_test)))  
  
# plot for residual error  
  
## setting plot style  
plt.style.use('fivethirtyeight')  
  
## plotting residual errors in training data  
plt.scatter(reg.predict(X\_train), reg.predict(X\_train) - y\_train,  
 color="green", s=10, label='Train data')  
  
## plotting residual errors in test data  
plt.scatter(reg.predict(X\_test), reg.predict(X\_test) - y\_test,  
 color="blue", s=10, label='Test data')  
  
## plotting line for zero residual error  
plt.hlines(y=0, xmin=0, xmax=50, linewidth=2)  
  
## plotting legend  
plt.legend(loc='upper right')  
  
## plot title  
plt.title("Residual errors")  
  
## method call for showing the plot  
plt.show()