Problem Statement

Build the linear regression model using scikit learn in boston data to predict 'Price' based on other dependent variable.

Here is the code to load the data

import numpy as np

import pandas as pd

import scipy.stats as stats

import matplotlib.pyplot as plt

import sklearn from sklearn.datasets

import load\_boston boston = load\_boston()

bos = pd.DataFrame(boston.data)

**##Boston Dictionary Keys**

boston.keys()

op1.PNG

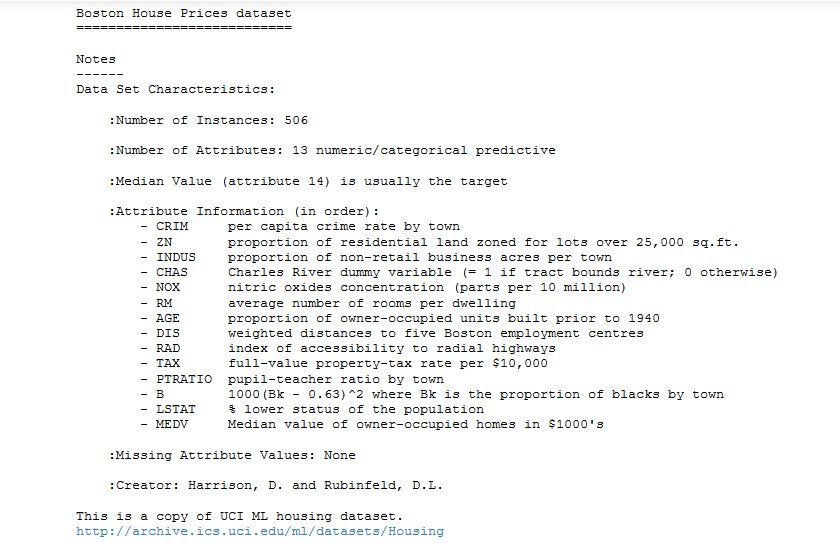
boston.data.shape

op2.PNG

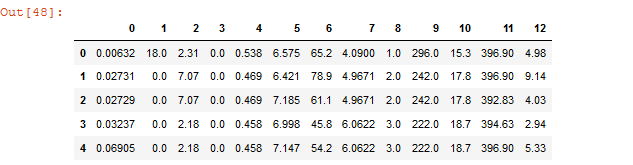
print(boston.feature\_names)

op3.PNG

print (boston.DESCR)

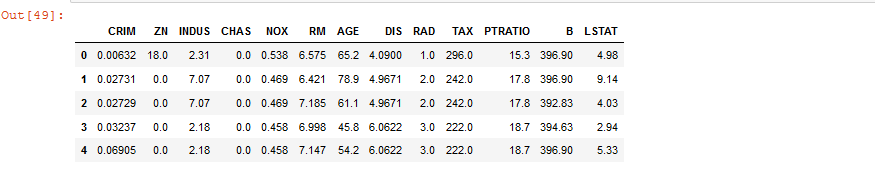


bos.head()



bos.columns = boston.feature\_names

bos.head()

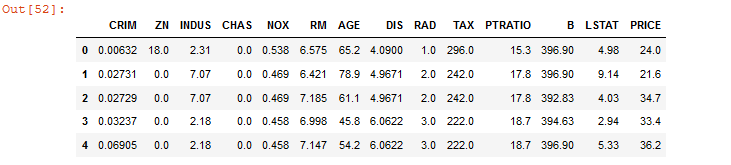


boston.target[:5]

op7.PNG

bos["PRICE"] = boston.target

bos.head()



#Skikit learning

from sklearn.linear\_model import LinearRegression

X = bos.drop('PRICE', axis = 1)

#linear object

lm = LinearRegression()

lm.fit(X,bos.PRICE)

op9.PNG

#Intercept and coefficients

print("Estimated intercept coefficient:",lm.intercept\_)

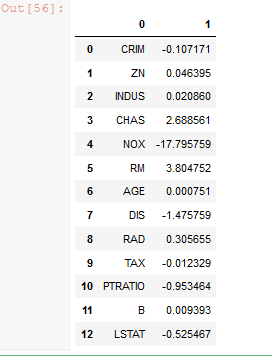
op10.PNG

print("number of coefficients:",len(lm.coef\_))

op11.PNG

#column 0 is 'features' and 1 is 'estimated coefficients'

pd.DataFrame(list(zip(X.columns, lm.coef\_)))



#plot between true housing prices and true RM

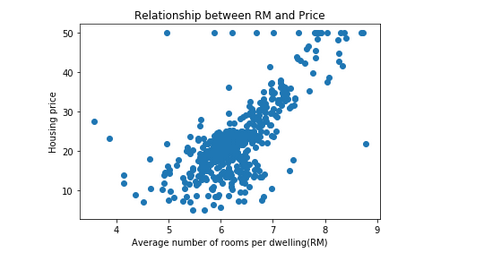
plt.scatter(bos.RM, bos.PRICE)

plt.xlabel("Average number of rooms per dwelling(RM)")

plt.ylabel("Housing price")

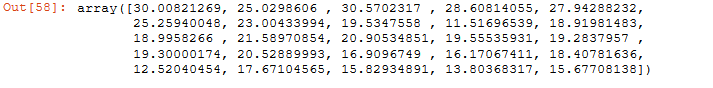
plt.title("Relationship between RM and Price")

plt.show()



#predicting prices for first 25 houses

lm.predict(X)[0:25]



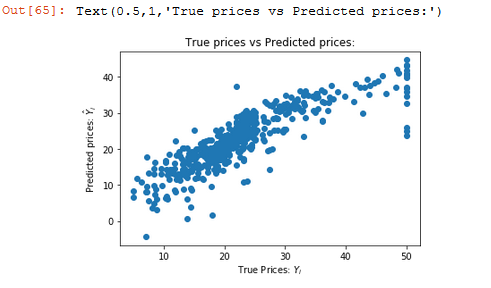
#plot between true prices and predicted prices

plt.scatter(bos.PRICE, lm.predict(X))

plt.xlabel("True Prices: $Y\_i$")

plt.ylabel("Predicted prices: $\hat{Y}\_i$")

plt.title("True prices vs Predicted prices:")



#Mean Squared Error

mseFull = np.mean((bos.PRICE - lm.predict(X))\*\*2)

print(mseFull)

op16.PNG

#Train-test split

from sklearn.cross\_validation import cross\_val\_score

X\_train, X\_test, Y\_train, Y\_test = sklearn.cross\_validation.train\_test\_split(X, bos.PRICE, test\_size=0.33, random\_state= 5)

print("Fit a model X\_train, and calculate MSE with Y\_train:", np.mean((Y\_train-lm.predict(X\_train))\*\*2))

print("Fit a model X\_train, and calculate MSE with X\_test, Y\_test:",np.mean(Y\_test - lm.predict(X\_test))\*\*2)

op17.PNG

#Residuals vs Residual plot (blue) Training and (green) tezt data

plt.scatter(lm.predict(X\_train),lm.predict(X\_train) - Y\_train,c='b',s=40,alpha=0.5)

plt.scatter(lm.predict(X\_test),lm.predict(X\_test) - Y\_test,c='g',s=40)

plt.hlines(y = 0, xmin = 0, xmax = 50)

plt.title("Residual plot using training(blue) and test(green) data")

plt.ylabel("Residuals")

