**Exercise 3 :** Application of Linear Regression

In this exercise you will learn how to apply scikit-learn Linear Regression model and Polynomial feature transformation model.

Using the built-in data set of housing prices of Boston metro area, you are required to build a model that will predict the housing prices in the Boston metro area.

1. Load the data set Boston housing prices from scikit learn
2. Normalize the data and target using Standard scaler.
3. Split the data and target to train and test sets such that 80% is train and 20% test
4. Initialize LinearRegression model and train the model
5. Calculate the score and note the performance of the model
6. Add mean square error calculation from scikit-learn metrics
7. Note the mean square error
8. Using scikit-learn PolynomialFeature API, add polynomial of degree 2 to generate additional polynomial features
9. Retrain the LinearRegression model using the transformed data set from 8
10. What is the score of the model with polynomial transformed data?
11. What is the mean square error of the model with polynomial transformed data?

Answer:

**from** sklearn.datasets **import** load\_boston  
**from** sklearn.preprocessing **import** StandardScaler  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.linear\_model **import** LinearRegression  
**from** sklearn.preprocessing **import** PolynomialFeatures  
**from** sklearn.metrics **import** mean\_squared\_error  
  
data=load\_boston()  
  
X = data.data  
y = data.target  
s\_data = StandardScaler()  
s\_target = StandardScaler()  
  
*#Notice that for regression we normalize both  
# features and target  
# and they are seperatly transformed and target also transformed*X = s\_data.fit\_transform(X)  
y = s\_target.fit\_transform(y.reshape(y.shape[0],1))  
  
pol = PolynomialFeatures(degree=2)  
X = pol.fit\_transform(X)  
  
X\_tr, X\_tst, y\_tr, y\_tst = train\_test\_split(X,y, shuffle=**True**, test\_size=0.2, random\_state=32)  
  
lr = LinearRegression()  
lr.fit(X\_tr, y\_tr)  
s=lr.score(X\_tst, y\_tst)  
mse = mean\_squared\_error(y\_tst, lr.predict(X\_tst))  
print(**'Score '**, s)  
print(**' MSE '**, mse)  
print(**'Hello'**)