**Project #1: California Housing Price Prediction**

**Question1: Load the data**:

* Read the “**housing.csv**” file from the folder into the program.
* Print first few rows of this data.
* Extract input (X) and output (Y) data from the dataset.

*Q#1 Code:*

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

*#1.1*

housing\_data = pd.read\_excel('/home/labsuser/Datasets/housing.xlsx')

*#1.2*

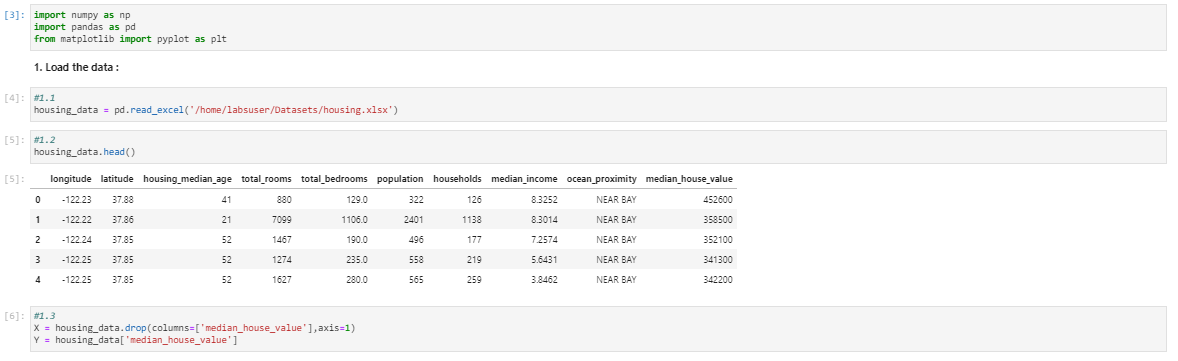
housing\_data.head()

*#1.3*

X = housing\_data.drop(columns=['median\_house\_value'],axis=1)

Y = housing\_data['median\_house\_value']

*Screenshot:*



**Question2: Handle missing values** :

* Fill the missing values with the mean of the respective column.

*Q#2 Code:*

for i in X.columns:

if(X[i].hasnans):

print("Column", i, "-","has NANs",X[i].hasnans)

X[i].fillna(value=X[i].mean(), inplace=True)

print("NANs replaced with mean value in column", i)

if(Y.hasnans):

Y.fillna(value=Y.mean(), inplace=True)

*Screenshot:*



**Question3: Encode categorical data** :

* Convert categorical column in the dataset to numerical data.

*Q#3 Code:*

X['ocean\_proximity'].unique()

X['ocean\_proximity'].replace(to\_replace=["NEAR BAY", "<1H OCEAN", "INLAND","NEAR OCEAN","ISLAND"], value=[1,2,3,4,5], inplace=True)

X['ocean\_proximity'].unique()

*Screenshot:*



**Question4: Split the dataset** :

* Split the data into 80% training dataset and 20% test dataset.

*Q#4 Code:*

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(X,Y, test\_size=0.20)

print(x\_train.shape)

print(y\_train.shape)

print(x\_test.shape)

print(y\_test.shape)

*Screenshot:*



**Question5: Standardize data** :

* Standardize training and test datasets.

*Q#5 Code:*

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(x\_train)

scaler.fit(x\_test)

scaler.fit(pd.DataFrame(y\_train))

scaler.fit(pd.DataFrame(y\_test))

trans\_x\_train = scaler.transform(x\_train)

trans\_x\_test = scaler.transform(x\_test)

trans\_y\_train = scaler.transform(pd.DataFrame(y\_train))

trans\_y\_test = scaler.transform(pd.DataFrame(y\_test))

#housing\_data.columns

trns\_train\_df = pd.concat([pd.DataFrame(trans\_x\_train, columns=['longitude', 'latitude', 'housing\_median\_age', 'total\_rooms',

'total\_bedrooms', 'population', 'households', 'median\_income',

'ocean\_proximity']), pd.DataFrame(trans\_y\_train, columns=['median\_house\_value'])], axis=1)

#trns\_train\_df.shape

trns\_test\_df = pd.concat([pd.DataFrame(trans\_x\_test, columns=['longitude', 'latitude', 'housing\_median\_age', 'total\_rooms',

'total\_bedrooms', 'population', 'households', 'median\_income',

'ocean\_proximity']), pd.DataFrame(trans\_y\_test, columns=['median\_house\_value'])], axis=1)

#trns\_test\_df.shape

*Screenshot:*



**Question6: Perform Linear Regression**:

* Perform Linear Regression on training data.
* Predict output for test dataset using the fitted model.
* Print root mean squared error (RMSE) from Linear Regression.

            [ HINT: Import **mean\_squared\_error** from **sklearn.metrics** ]

*Q#6 Code:*

*#6.1*

from sklearn.linear\_model import LinearRegression

model = LinearRegression()

model.fit(trans\_x\_train,trans\_y\_train)

*#6.2*

pred\_val = model.predict(trans\_x\_test)

new\_df = pd.concat([trns\_test\_df, pd.DataFrame(pred\_val, columns=['Predicted\_Value'])],axis=1)

new\_df['err\_pctg'] = abs(new\_df['median\_house\_value'] - new\_df['Predicted\_Value'])/new\_df['median\_house\_value']

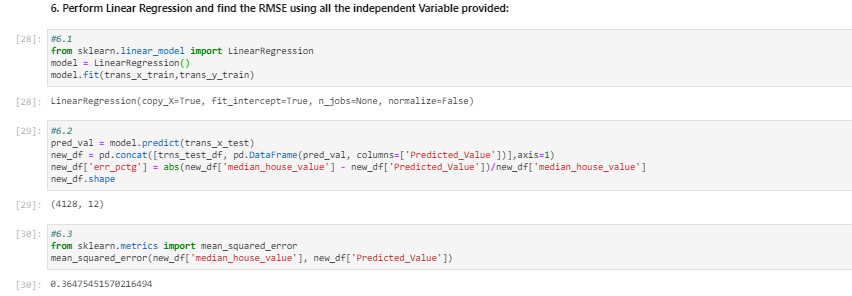
new\_df.shape

*#6.3*

from sklearn.metrics import mean\_squared\_error

mean\_squared\_error(new\_df['median\_house\_value'], new\_df['Predicted\_Value'])

*Screenshot:*



**Question7: Bonus exercise: Perform Linear Regression with one independent variable** :

* Extract just the median\_income column from the independent variables (from **X\_train** and **X\_test**).
* Perform Linear Regression to predict housing values based on **median\_income**.
* Predict output for test dataset using the fitted model.
* Plot the fitted model for training data as well as for test data to check if the fitted model satisfies the test data.

*Q#7 Code:*

*#7.1*

X1= x\_train['median\_income']

Y1= y\_train

X2= x\_test['median\_income']

Y2= y\_test

*#7.2*

new\_model=LinearRegression()

new\_model.fit(pd.DataFrame(X1),Y1)

*#7.3*

pred\_val=new\_model.predict(pd.DataFrame(X2))

new\_df1 = pd.concat([X2.reset\_index(),Y2.reset\_index(),pd.DataFrame(pred\_val, columns=['Pred\_Value'])],axis=1)

new\_df1.head()

new\_df1['err\_pctg'] = abs(new\_df1['median\_house\_value'] - new\_df1['Pred\_Value'])/new\_df1['median\_house\_value']

1-new\_df1.err\_pctg.mean()

*#7.4*

from matplotlib import pyplot as plt

plt.figure(figsize=(30,30))

plt.scatter(new\_df1['median\_house\_value'], new\_df1['Pred\_Value'])

plt.xlabel('median\_house\_value')

plt.ylabel('Pred\_Value')

plt.show()

*Screenshot:*

