

California Housing Price Prediction

December 27, 2022

1 Import required libraries

```
[27]: import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from math import sqrt
import matplotlib.pyplot as plt
```

2 Load The Data

```
[2]: housing = pd.read_excel("housing.xlsx")
```

3 Print first few rows of this data

```
[3]: housing.head(5)
```

```
[3]:
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
0	-122.23	37.88	41	880	129.0	
1	-122.22	37.86	21	7099	1106.0	
2	-122.24	37.85	52	1467	190.0	
3	-122.25	37.85	52	1274	235.0	
4	-122.25	37.85	52	1627	280.0	

	population	households	median_income	ocean_proximity	median_house_value
0	322	126	8.3252	NEAR BAY	452600
1	2401	1138	8.3014	NEAR BAY	358500
2	496	177	7.2574	NEAR BAY	352100
3	558	219	5.6431	NEAR BAY	341300
4	565	259	3.8462	NEAR BAY	342200

4 Fill the missing values with the mean of the respective column

```
[4]: housing.isnull().sum()
```

```
[4]: longitude          0
      latitude          0
      housing_median_age  0
      total_rooms        0
      total_bedrooms    207
      population        0
      households        0
      median_income     0
      ocean_proximity    0
      median_house_value  0
      dtype: int64
```

```
[5]: housing.total_bedrooms = housing.total_bedrooms.fillna(housing.total_bedrooms.
      ↪mean())
      housing.isnull().sum()
```

```
[5]: longitude          0
      latitude          0
      housing_median_age  0
      total_rooms        0
      total_bedrooms    0
      population        0
      households        0
      median_income     0
      ocean_proximity    0
      median_house_value  0
      dtype: int64
```

5 Convert categorical column in the dataset to numerical data

```
[6]: le = LabelEncoder()
```

```
[7]: housing['ocean_proximity']=le.fit_transform(housing['ocean_proximity'])
```

6 Standardize training and test datasets

```
[8]: names = housing.columns
     scaler = StandardScaler()
```

```
[9]: scaled_df = scaler.fit_transform(housing)
     scaled_df = pd.DataFrame(scaled_df, columns=names)
     scaled_df.head()
```

```
[9]:
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
0	-1.327835	1.052548	0.982143	-0.804819	-0.975228	
1	-1.322844	1.043185	-0.607019	2.045890	1.355088	
2	-1.332827	1.038503	1.856182	-0.535746	-0.829732	
3	-1.337818	1.038503	1.856182	-0.624215	-0.722399	
4	-1.337818	1.038503	1.856182	-0.462404	-0.615066	

	population	households	median_income	ocean_proximity	median_house_value
0	-0.974429	-0.977033	2.344766	1.291089	2.129631
1	0.861439	1.669961	2.332238	1.291089	1.314156
2	-0.820777	-0.843637	1.782699	1.291089	1.258693
3	-0.766028	-0.733781	0.932968	1.291089	1.165100
4	-0.759847	-0.629157	-0.012881	1.291089	1.172900

7 Extract input (X) and output (Y) data from the dataset

```
[10]: X_features = [
     ↪ ['longitude', 'latitude', 'housing_median_age', 'total_rooms', 'total_bedrooms', 'population', 'h
     ↪ 'median_income', 'ocean_proximity', 'median_house_value']
     X = scaled_df[X_features]
     Y = scaled_df['median_house_value']
```

```
[11]: print(type(X))
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
[12]: print(type(Y))
```

```
<class 'pandas.core.series.Series'>
```

```
[13]: print(X.shape)
```

```
(20640, 10)
```

```
[14]: print(Y.shape)
```

(20640,)

8 Split the data into 80% training dataset and 20% test dataset

```
[15]: from sklearn.model_selection import train_test_split
      x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=1)
```

```
[16]: print (x_train.shape, y_train.shape)
```

(16512, 10) (16512,)

```
[17]: print (x_test.shape, y_test.shape)
```

(4128, 10) (4128,)

9 Perform Linear Regression

```
[18]: linreg=LinearRegression()
      linreg.fit(x_train,y_train)
```

```
[18]: LinearRegression()
```

```
[19]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
[19]: LinearRegression(normalize=False)
```

```
[20]: y_predict = linreg.predict(x_test)
```

```
[21]: print(sqrt(mean_squared_error(y_test,y_predict)))
      print((r2_score(y_test,y_predict)))
```

2.451652034014511e-15

1.0

10 Extract just the median_income column from the independent variables

```
[22]: x_train_Income=x_train[['median_income']]
      x_test_Income=x_test[['median_income']]
```

```
[23]: print(x_train_Income.shape)
      print(y_train.shape)
```

```
(16512, 1)
(16512,)
```

11 Perform Linear Regression to predict housing values based on median_income

```
[24]: linreg=LinearRegression()
linreg.fit(x_train_Income,y_train)
y_predict = linreg.predict(x_test_Income)
```

12 Predict output for test dataset using the fitted model

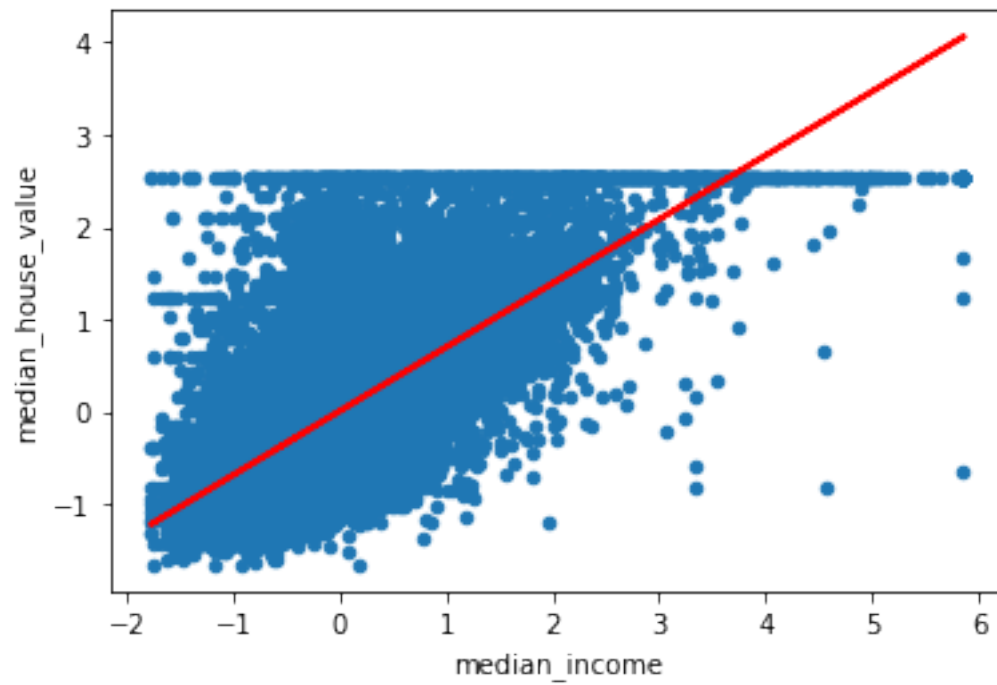
```
[25]: print(linreg.intercept_, linreg.coef_)
print(sqrt(mean_squared_error(y_test,y_predict)))
print((r2_score(y_test,y_predict)))
```

```
0.005623019866893164 [0.69238221]
0.7212595914243148
0.47190835934467734
```

13 Plot the fitted model for training data as well as for test data to check if the fitted model satisfies the test data

```
[28]: scaled_df.plot(kind='scatter',x='median_income',y='median_house_value')
plt.plot(x_test_Income,y_predict,c='red',linewidth=2)
```

```
[28]: [<matplotlib.lines.Line2D at 0x7f63ae7ab190>]
```



[]: