California Housing Price Prediction

August 2, 2022

[1]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder,StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,r2_score ,mean_absolute_error
from math import sqrt
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

0.0.1 Load the data

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

[3]:
```
data
```

[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	
…	…	…	…	…	…	
20635	-121.09	39.48	25	1665	374.0	
20636	-121.21	39.49	18	697	150.0	
20637	-121.22	39.43	17	2254	485.0	
20638	-121.32	39.43	18	1860	409.0	
20639	-121.24	39.37	16	2785	616.0	

	population	households	median_income	ocean_proximity	\
0	322	126	8.3252	NEAR BAY	
1	2401	1138	8.3014	NEAR BAY	
2	496	177	7.2574	NEAR BAY	
3	558	219	5.6431	NEAR BAY	
4	565	259	3.8462	NEAR BAY	

```
…            …          …              …               …
20635         845         330         1.5603          INLAND
20636         356         114         2.5568          INLAND
20637        1007         433         1.7000          INLAND
20638         741         349         1.8672          INLAND
20639        1387         530         2.3886          INLAND

       median_house_value
0                  452600
1                  358500
2                  352100
3                  341300
4                  342200
…                     …
20635               78100
20636               77100
20637               92300
20638               84700
20639               89400

[20640 rows x 10 columns]
```

[4]:
```
data.head()
```

[4]:
```
   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
```

[5]:
```
data.columns
```

[5]:
```
Index(['longitude', 'latitude', 'housing_median_age', 'total_rooms',
       'total_bedrooms', 'population', 'households', 'median_income',
       'ocean_proximity', 'median_house_value'],
      dtype='object')
```

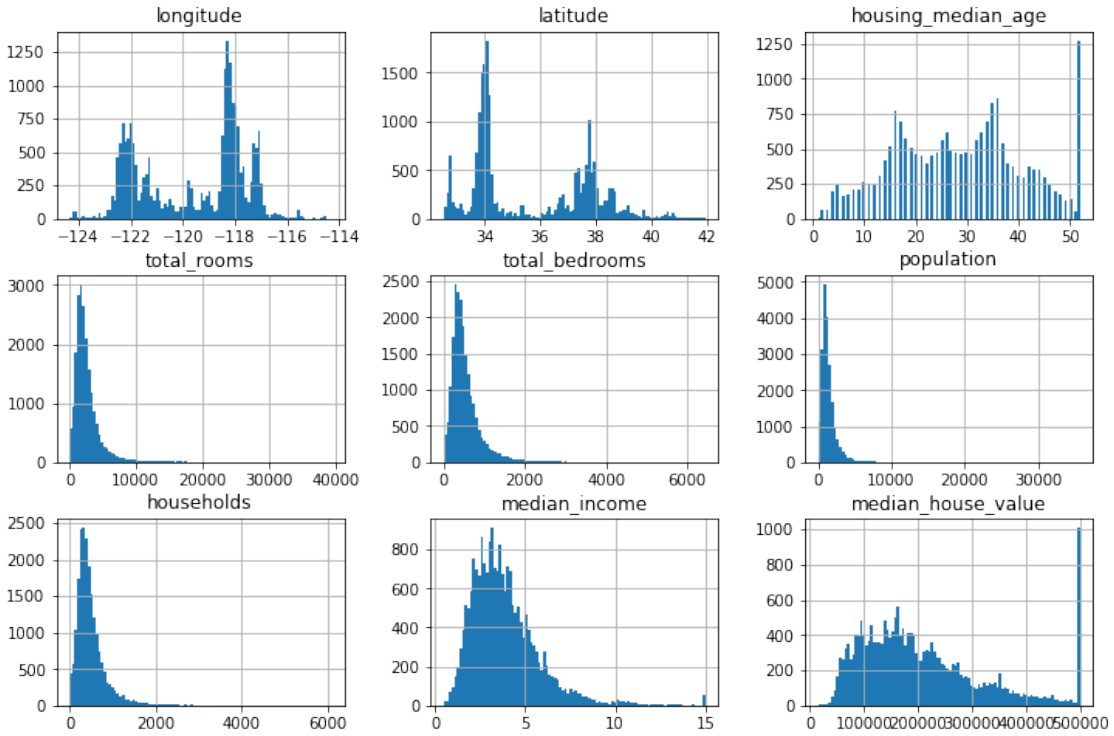
[6]:
```
X_Features=['longitude', 'latitude', 'housing_median_age', 'total_rooms',
       'total_bedrooms', 'population', 'households', 'median_income',
```

```
       'ocean_proximity']
X=data[X_Features]
Y=data['median_house_value']

print(type(X))
print(type(Y))
print(data.shape)
print(X.shape)
print(Y.shape)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.series.Series'>
(20640, 10)
(20640, 9)
(20640,)
```

[7]:
```
data.hist(bins=100, figsize=(12, 8))
plt.show()
```



0.0.2 Handle missing values :

[8]:
```
data.isnull().sum()
```

[8]:
```
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
```

[9]:
```
data.total_bedrooms=data.total_bedrooms.fillna(data.total_bedrooms.mean())
data.isnull().sum()
```

[9]:
```
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
```

[10]:
```
data.isnull().sum().any()
```

[10]:
```
False
```

0.0.3 Encode categorical data

[11]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
```

```
 2   housing_median_age  20640 non-null  int64
 3   total_rooms         20640 non-null  int64
 4   total_bedrooms      20640 non-null  float64
 5   population          20640 non-null  int64
 6   households          20640 non-null  int64
 7   median_income       20640 non-null  float64
 8   ocean_proximity     20640 non-null  object
 9   median_house_value  20640 non-null  int64
dtypes: float64(4), int64(5), object(1)
memory usage: 1.6+ MB
```

[12]:
```
data['ocean_proximity'].unique()
```

[12]:
```
array(['NEAR BAY', '<1H OCEAN', 'INLAND', 'NEAR OCEAN', 'ISLAND'],
      dtype=object)
```

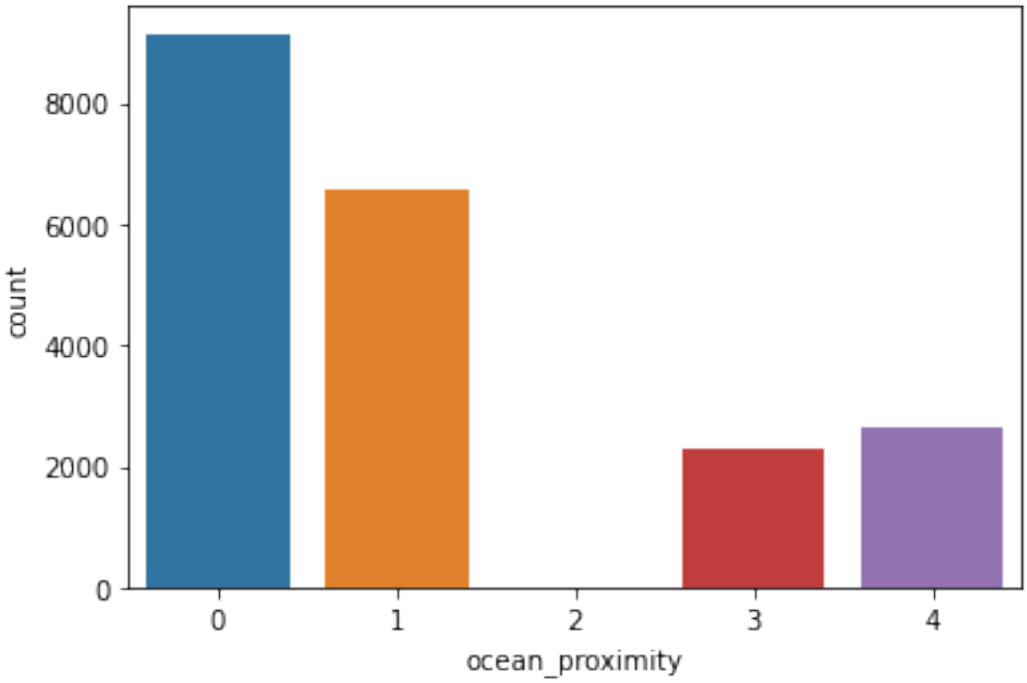
[13]:
```
data_le = LabelEncoder()
data['ocean_proximity']=data_le.fit_transform(data['ocean_proximity'])
```

[14]:
```
data['ocean_proximity'].unique()
```

[14]:
```
array([3, 0, 1, 4, 2])
```

[15]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.countplot(x=data['ocean_proximity'])
```

[15]:
```
<AxesSubplot:xlabel='ocean_proximity', ylabel='count'>
```



[16]:
```
data['ocean_proximity'].head()
```

[16]:
```
0    3
1    3
2    3
3    3
4    3
Name: ocean_proximity, dtype: int64
```

0.0.4 Standardize data :

[17]:
```
names = data.columns
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)
scaled_data = pd.DataFrame(scaled_data, columns=names)
scaled_data.head()
```

[17]:

	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
```

[18]:
```
data.drop(['population','total_bedrooms','latitude'],axis=1,inplace=True)
data.drop(['households','longitude'],axis=1,inplace=True)
X=data.drop('median_house_value',axis=1)
y=data.median_house_value
```

[19]:
```
data
```

[19]:
```
       housing_median_age  total_rooms  median_income  ocean_proximity  \
0                      41          880         8.3252                3
1                      21         7099         8.3014                3
2                      52         1467         7.2574                3
3                      52         1274         5.6431                3
4                      52         1627         3.8462                3
…                     …           …              …                …
20635                  25         1665         1.5603                1
20636                  18          697         2.5568                1
20637                  17         2254         1.7000                1
20638                  18         1860         1.8672                1
20639                  16         2785         2.3886                1

       median_house_value
0                  452600
1                  358500
2                  352100
3                  341300
4                  342200
…                   …
20635               78100
20636               77100
20637               92300
20638               84700
20639               89400

[20640 rows x 5 columns]
```

0.0.5 Split the dataset :

[20]:
```
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=100)

print (x_train.shape, y_train.shape)
print (x_test.shape, y_test.shape)
```

```
(16512, 4) (16512,)
(4128, 4) (4128,)
```

0.0.6 Perform Linear Regression

[21]:
```
linreg=LinearRegression()
linreg.fit(x_train,y_train)
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
y_predict = linreg.predict(x_test)

print('MAE     :', mean_absolute_error(y_test,y_predict))
print('MSE     :', mean_squared_error(y_test,y_predict))
print('RMSE    :', np.sqrt(mean_squared_error(y_test,y_predict)))
print('R2 Score:', (r2_score(y_test,y_predict)))
```

```
MAE     : 59031.51147586589
MSE     : 6242317582.243216
RMSE    : 79008.3386880348
R2 Score: 0.5367129067790649
```

0.0.7 Perform Linear Regression with one independent variable

[22]:
```
x_train_median_income=x_train[['median_income']]
x_test_median_income=x_test[['median_income']]
```

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

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

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

print('MAE     :', mean_absolute_error(y_test,y_predict))
```

```
print('MSE     :', mean_squared_error(y_test,y_predict))
print('RMSE    :', np.sqrt(mean_squared_error(y_test,y_predict)))
print('R2 Score:', (r2_score(y_test,y_predict)))
```

```
MAE     : 62151.532889553906
MSE     : 6836850218.862475
RMSE    : 82685.24789140125
R2 Score: 0.49258838196670596
```

[25]:
```
plt.plot(y_test,y_predict, 'o', color='blue')
m, b = np.polyfit(y_test,y_predict, 1)
plt.plot(y_test, m*y_test+b, color='red')
```

[25]: [<matplotlib.lines.Line2D at 0x7f6701549a90>]