fornia-housing-price-prediction-2

June 15, 2023

1 California-Housing-Price-Prediction

[ ]:
```
#Import Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
#Loading the Dataset
df = pd.read_csv('1553768847_housing.csv')
```

[3]:
```
df.head() #Analysising top 5 datasets
```

[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]:
```
df.tail() #Analyzing the bottom 5 datasets
```

[4]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
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  \
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
20635               78100
20636               77100
20637               92300
20638               84700
20639               89400
```

[5]:
```
df.describe() #Analyzing the statistical measures of the dataset
```

[5]:
```
          longitude      latitude  housing_median_age   total_rooms  \
count  20640.000000  20640.000000        20640.000000  20640.000000
mean    -119.569704     35.631861           28.639486   2635.763081
std        2.003532      2.135952           12.585558   2181.615252
min     -124.350000     32.540000            1.000000      2.000000
25%     -121.800000     33.930000           18.000000   1447.750000
50%     -118.490000     34.260000           29.000000   2127.000000
75%     -118.010000     37.710000           37.000000   3148.000000
max     -114.310000     41.950000           52.000000  39320.000000

       total_bedrooms    population    households  median_income  \
count    20433.000000  20640.000000  20640.000000   20640.000000
mean       537.870553   1425.476744    499.539680       3.870671
std        421.385070   1132.462122    382.329753       1.899822
min          1.000000      3.000000      1.000000       0.499900
25%        296.000000    787.000000    280.000000       2.563400
50%        435.000000   1166.000000    409.000000       3.534800
75%        647.000000   1725.000000    605.000000       4.743250
max       6445.000000  35682.000000   6082.000000      15.000100

       median_house_value
count        20640.000000
mean        206855.816909
std         115395.615874
min          14999.000000
25%         119600.000000
50%         179700.000000
75%         264725.000000
max         500001.000000
```

[6]:
```
df.info() #Overall info on the dataset
```

```
<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      20433 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
```

[7]:
```
df['ocean_proximity'].value_counts() #Analyzing the values for column␣
↪"ocean_proximity"
```

[7]:
```
<1H OCEAN     9136
INLAND        6551
NEAR OCEAN    2658
NEAR BAY      2290
ISLAND           5
Name: ocean_proximity, dtype: int64
```

[8]:
```
df.shape #For total count of rows and column
```

[8]:
```
(20640, 10)
```

[9]:
```
df_shuffled = df.sample(n=len(df), random_state=1)
df_shuffled
```

[9]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
4712	-118.36	34.06	39	2810	670.0	
2151	-119.78	36.78	37	2185	455.0	
15927	-122.42	37.73	46	1819	411.0	
82	-122.28	37.81	52	340	97.0	
8161	-118.13	33.82	37	1530	290.0	
…	…	…	…	…	…	
10955	-117.88	33.76	17	1768	474.0	
17289	-119.63	34.42	42	1765	263.0	
5192	-118.26	33.93	42	1433	295.0	

```
12172    -117.16     33.73                  10         2381           454.0
235      -122.20     37.79                  35         1802           459.0

       population  households  median_income ocean_proximity  \
4712         1109         624         3.2500       <1H OCEAN
2151         1143         438         1.9784          INLAND
15927        1534         406         4.0132        NEAR BAY
82            200          87         1.5208        NEAR BAY
8161          711         283         5.1795       <1H OCEAN
…             …           …              …               …
10955        1079         436         1.7823       <1H OCEAN
17289         753         260         8.5608       <1H OCEAN
5192          775         293         1.1326       <1H OCEAN
12172        1323         477         2.6322          INLAND
235          1009         390         2.3036        NEAR BAY

       median_house_value
4712               355000
2151                70700
15927              229400
82                 112500
8161               225400
…                   …
10955              205300
17289              500001
5192               104800
12172              140700
235                126000

[20640 rows x 10 columns]
```

[10]:
```
pd.get_dummies(df_shuffled['ocean_proximity']).head()
```

[10]:
```
       <1H OCEAN  INLAND  ISLAND  NEAR BAY  NEAR OCEAN
4712           1       0       0         0           0
2151           0       1       0         0           0
15927          0       0       0         1           0
82             0       0       0         1           0
8161           1       0       0         0           0
```

[11]:
```
df_shuffled.drop('ocean_proximity',axis =1).head()
```

[11]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
4712     -118.36     34.06                  39         2810           670.0
2151     -119.78     36.78                  37         2185           455.0
15927    -122.42     37.73                  46         1819           411.0
82       -122.28     37.81                  52          340            97.0
```

```
8161     -118.13     33.82                 37         1530           290.0

       population  households  median_income  median_house_value
4712         1109         624         3.2500              355000
2151         1143         438         1.9784               70700
15927        1534         406         4.0132              229400
82            200          87         1.5208              112500
8161          711         283         5.1795              225400
```

[12]:
```
df_final = pd.concat([df_shuffled.drop('ocean_proximity',axis =1), pd.
 ↪get_dummies(df_shuffled['ocean_proximity'])], axis=1)
df_final
```

[12]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
4712     -118.36     34.06                  39         2810           670.0
2151     -119.78     36.78                  37         2185           455.0
15927    -122.42     37.73                  46         1819           411.0
82       -122.28     37.81                  52          340            97.0
8161     -118.13     33.82                  37         1530           290.0
…            …         …                   …            …              …
10955    -117.88     33.76                  17         1768           474.0
17289    -119.63     34.42                  42         1765           263.0
5192     -118.26     33.93                  42         1433           295.0
12172    -117.16     33.73                  10         2381           454.0
235      -122.20     37.79                  35         1802           459.0

       population  households  median_income  median_house_value  <1H OCEAN  \
4712         1109         624         3.2500              355000          1
2151         1143         438         1.9784               70700          0
15927        1534         406         4.0132              229400          0
82            200          87         1.5208              112500          0
8161          711         283         5.1795              225400          1
…             …           …              …                   …          …
10955        1079         436         1.7823              205300          1
17289         753         260         8.5608              500001          1
5192          775         293         1.1326              104800          1
12172        1323         477         2.6322              140700          0
235          1009         390         2.3036              126000          0

       INLAND  ISLAND  NEAR BAY  NEAR OCEAN
4712        0       0         0           0
2151        1       0         0           0
15927       0       0         1           0
82          0       0         1           0
8161        0       0         0           0
…       …       …         …           …
10955       0       0         0           0
```

```
17289       0       0         0           0
5192        0       0         0           0
12172       1       0         0           0
235         0       0         1           0

[20640 rows x 14 columns]
```

[13]:
```
df_final = df_final[['longitude', 'latitude',
                     'housing_median_age', 'total_rooms',
                     'total_bedrooms', 'population',
                     'households', 'median_income',
                     '<1H OCEAN', 'INLAND',
                     'ISLAND', 'NEAR BAY', 'NEAR OCEAN', 'median_house_value']]
df_final
```

[13]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
4712     -118.36     34.06                  39         2810           670.0
2151     -119.78     36.78                  37         2185           455.0
15927    -122.42     37.73                  46         1819           411.0
82       -122.28     37.81                  52          340            97.0
8161     -118.13     33.82                  37         1530           290.0
…           …        …                     …           …              …
10955    -117.88     33.76                  17         1768           474.0
17289    -119.63     34.42                  42         1765           263.0
5192     -118.26     33.93                  42         1433           295.0
12172    -117.16     33.73                  10         2381           454.0
235      -122.20     37.79                  35         1802           459.0

       population  households  median_income  <1H OCEAN  INLAND  ISLAND  \
4712         1109         624         3.2500          1       0       0
2151         1143         438         1.9784          0       1       0
15927        1534         406         4.0132          0       0       0
82            200          87         1.5208          0       0       0
8161          711         283         5.1795          1       0       0
…             …           …              …          …       …       …
10955        1079         436         1.7823          1       0       0
17289         753         260         8.5608          1       0       0
5192          775         293         1.1326          1       0       0
12172        1323         477         2.6322          0       1       0
235          1009         390         2.3036          0       0       0

       NEAR BAY  NEAR OCEAN  median_house_value
4712          0           0              355000
2151          0           0               70700
15927         1           0              229400
82            1           0              112500
8161          0           0              225400
```

```
…          …          …                  …
10955         0          0              205300
17289         0          0              500001
5192          0          0              104800
12172         0          0              140700
235           1          0              126000

[20640 rows x 14 columns]
```

[14]:
```
df_final.isnull().sum()
```

[14]:
```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
<1H OCEAN               0
INLAND                  0
ISLAND                  0
NEAR BAY                0
NEAR OCEAN              0
median_house_value      0
dtype: int64
```

[15]:
```
df_final = df_final.dropna()
len(df_final)
```

[15]:
```
20433
```

[16]:
```
train_pd, test_pd, val_pd = df_final[:8500], df_final[8500:16000],␣
 ↪df_final[16000:]
len(train_pd), len(test_pd), len(val_pd)
```

[16]:
```
(8500, 7500, 4433)
```

[17]:
```
X_train, y_train = train_pd.to_numpy()[:,:-1], train_pd.to_numpy()[:,-1]
X_train
```

[17]:
```
array([[-118.36,   34.06,   39.  , …,    0.  ,    0.  ,    0.  ],
       [-119.78,   36.78,   37.  , …,    0.  ,    0.  ,    0.  ],
       [-122.42,   37.73,   46.  , …,    0.  ,    1.  ,    0.  ],
       …,
       [-119.81,   36.78,   36.  , …,    0.  ,    0.  ,    0.  ],
       [-118.29,   33.75,   37.  , …,    0.  ,    0.  ,    1.  ],
```

```
       [-118.23,   33.89,   35.  , …,    0.  ,    0.  ,    0.  ]])
```

[18]:
```
X_train.shape
```

[18]:
```
(8500, 13)
```

[19]:
```
y_train
```

[19]:
```
array([355000.,  70700., 229400., …,  79700., 218900.,  95100.])
```

[20]:
```
y_train.shape
```

[20]:
```
(8500,)
```

[21]:
```
X_val, y_val = val_pd.to_numpy()[:,:-1], val_pd.to_numpy()[:,-1]
X_test, y_test = test_pd.to_numpy()[:,:-1], test_pd.to_numpy()[:,-1]
X_train.shape, y_train.shape, X_val.shape, y_val.shape, X_test.shape, y_test.
 ↪shape
```

[21]:
```
((8500, 13), (8500,), (4433, 13), (4433,), (7500, 13), (7500,))
```

[22]:
```
#Scaling and transforming the data
from sklearn.preprocessing import StandardScaler
```

[29]:
```
scaler = StandardScaler().fit((X_train[:, :8]))

def preprocessor(X):
    A = np.copy(X)
    A[:,:8] = scaler.transform(X[:, :8])
    return A
X_train, X_val, X_test = preprocessor(X_train), preprocessor(X_val),␣
 ↪preprocessor(X_test)
X_train_preprocessed
```

[29]:
```
array([[ 0.59490582, -0.7288167 ,  0.81049402, …,  0.        ,
         0.        ,  0.        ],
       [-0.11518538,  0.54448242,  0.65148559, …,  0.        ,
         0.        ,  0.        ],
       [-1.43535495,  0.98920087,  1.36702352, …,  0.        ,
         1.        ,  0.        ],
       …,
       [-0.13018731,  0.54448242,  0.57198138, …,  0.        ,
         0.        ,  0.        ],
       [ 0.62991032, -0.87393535,  0.65148559, …,  0.        ,
         0.        ,  1.        ],
       [ 0.65991417, -0.8083979 ,  0.49247716, …,  0.        ,
         0.        ,  0.        ]])
```

[30]:
```
X_train.shape, X_val.shape, X_test.shape
```

[30]:
```
((8500, 13), (4433, 13), (7500, 13))
```

[26]:
```
pd.DataFrame(X_train_preprocessed)
```

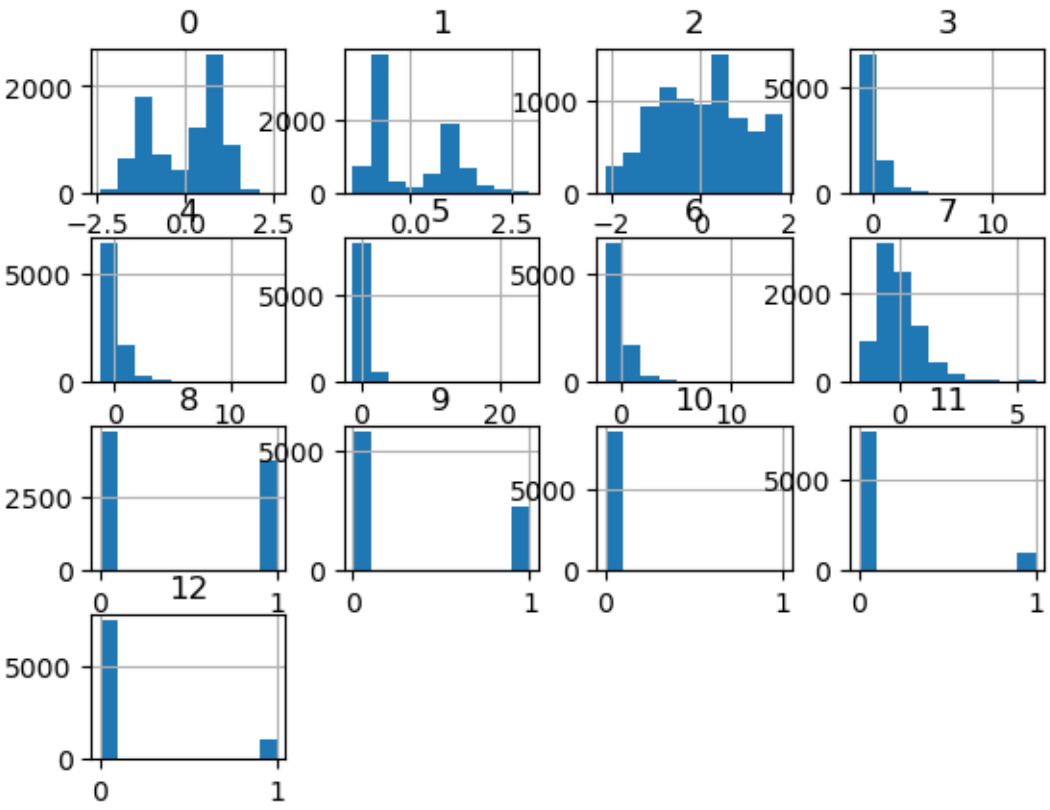
[26]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.594906	-0.728817	0.810494	0.076059	0.309903	-0.280417	0.323273	-0.321302	1.0	0.0	0.0	0.0	0.0
1	-0.115185	0.544482	0.651486	-0.207360	-0.199496	-0.250111	-0.163096	-0.982794	0.0	1.0	0.0	0.0	0.0
2	-1.435355	0.989201	1.367024	-0.373329	-0.303746	0.098405	-0.246772	0.075718	0.0	0.0	0.0	1.0	0.0
3	-1.365346	1.026651	1.844049	-1.044011	-1.047706	-1.090649	-1.080921	-1.220839	0.0	0.0	0.0	1.0	0.0
4	0.709921	-0.841167	0.651486	-0.504382	-0.590431	-0.635172	-0.568403	0.682432	1.0	0.0	0.0	0.0	0.0
…	…	…	…	…	…	…	…	…	…	…	…	…	…
8495	0.629910	-0.714773	1.844049	-0.621377	-0.514613	-0.391835	-0.385361	-1.020301	1.0	0.0	0.0	0.0	0.0
8496	0.919948	-0.930110	-1.018103	-0.454047	-0.438796	-0.426597	-0.476882	-0.229694	1.0	0.0	0.0	0.0	0.0
8497	-0.130187	0.544482	0.571981	-0.449966	-0.535937	-0.680630	-0.529180	-0.451353	0.0	1.0	0.0	0.0	0.0
8498	0.629910	-0.873935	0.651486	-0.600064	-0.585692	-0.586148	-0.563174	-0.605802	0.0	0.0	0.0	0.0	1.0
8499	0.659914	-0.808398	0.492477	-0.629086	-0.462489	0.319458	-0.411510	-0.870170	1.0	0.0	0.0	0.0	0.0

[8500 rows x 13 columns]

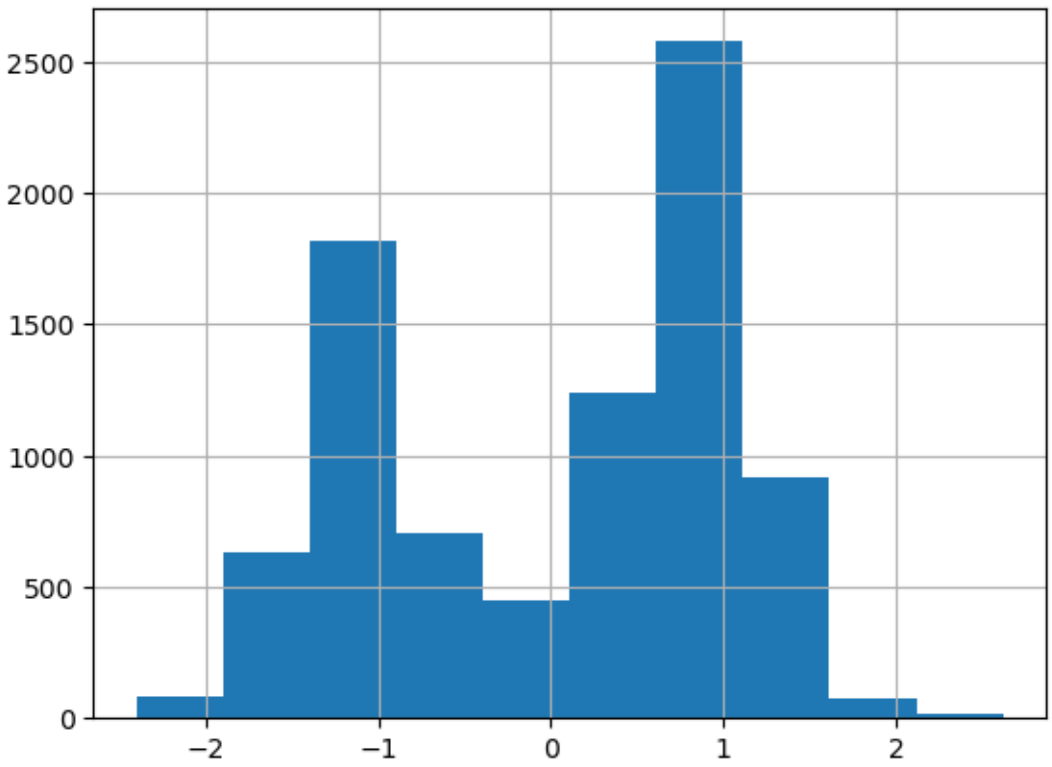
[62]:
```
pd.DataFrame(X_train_preprocessed).hist()
```

[62]:
```
array([[<Axes: title={'center': '0'}>, <Axes: title={'center': '1'}>,
        <Axes: title={'center': '2'}>, <Axes: title={'center': '3'}>],
       [<Axes: title={'center': '4'}>, <Axes: title={'center': '5'}>,
        <Axes: title={'center': '6'}>, <Axes: title={'center': '7'}>],
       [<Axes: title={'center': '8'}>, <Axes: title={'center': '9'}>,
        <Axes: title={'center': '10'}>, <Axes: title={'center': '11'}>],
       [<Axes: title={'center': '12'}>, <Axes: >, <Axes: >, <Axes: >]],
      dtype=object)
```



[28]:
```
pd.DataFrame(X_train_preprocessed)[0].hist()
```

[28]: <Axes: >



[31]:
```
#MSE = sum(y^(X)-y)^2/n
```

[35]:
```
from sklearn.metrics import mean_squared_error as mse
from sklearn.linear_model import LinearRegression
lm = LinearRegression().fit(X_train, y_train)
mse(lm.predict(X_train), y_train, squared = False), mse(lm.predict(X_val),␣
 ↪y_val, squared = False)
```

[35]:
```
(68248.5885194117, 69053.16237243109)
```

[45]:
```
from sklearn.neighbors import KNeighborsRegressor
km = KNeighborsRegressor(n_neighbors=10).fit(X_train, y_train)
mse(km.predict(X_train), y_train, squared = False), mse(km.predict(X_val),␣
 ↪y_val, squared = False)
```

[45]:
```
(55308.22961265735, 62312.41464220683)
```

[52]:
```
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor(max_depth=10).fit(X_train, y_train)
mse(rfr.predict(X_train), y_train, squared = False), mse(rfr.predict(X_val),␣
 ↪y_val, squared = False)
```

[52]: (40459.25531412211, 55541.50181835979)

[61]:
```
from sklearn.ensemble import GradientBoostingRegressor
gbr = GradientBoostingRegressor(n_estimators=250).fit(X_train, y_train)
mse(gbr.predict(X_train), y_train, squared = False), mse(gbr.predict(X_val),␣
 ↪y_val, squared = False)
```

[61]: (45177.214998102114, 51832.62812961522)

[ ]: