Instructions: Read the comments in code cells as you complete the problems. Submit your completed notebook and PDF files to Blackboard.

In [13]:
```
#0 Run this cell
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
```

In [14]:
```
#0 Run this cell and enter your first name when prompted.
# Do not modify the code here.
name = input("Enter your first name: ")
url = "https://raw.githubusercontent.com/babdelfa/ML/main/california_housing.c
name = pd.read_csv(url)
print("\nSample of the data: \n")
print(name.head())
print('The data has been loaded. Refer to the dataframe using your name.')
print('\nBegin the analysis below.')
```

```
Enter your first name: jij

Sample of the data:

   MedInc  HouseAge  AveRooms  AveBedrms  Population  AveOccup  Latitude  \
0  3.2083      34.0  5.471264   1.063218      1627.0  3.116858     32.78
1  5.7049       9.0  6.699571   1.085837       731.0  3.137339     34.66
2  1.6125      52.0  3.135135   1.364865       286.0  1.932432     37.78
3  5.2586      17.0  6.945035   1.131206      1809.0  3.207447     38.69
4  5.0380       9.0  5.428415   0.967213      2581.0  2.820765     38.42

   Longitude  MedHouseValue
0    -115.56          0.762
1    -118.17          1.732
2    -122.40          1.125
3    -121.26          1.370
4    -122.79          1.856
The data has been loaded. Refer to the dataframe using your name.

Begin the analysis below.
```

In [15]:
```
#1 Provide summary statistics about the data using the describe method.
name.describe()
```

Out[15]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Long
count	4128.000000	4128.000000	4128.000000	4128.000000	4128.000000	4128.000000	4128.000000	4128.00
mean	3.928064	28.267684	5.436901	1.091814	1421.930717	2.988522	35.644944	-119.58
std	1.904240	12.485409	1.903414	0.351354	1051.047314	3.698413	2.150855	1.99
min	0.499900	1.000000	1.885057	0.444444	8.000000	0.970588	32.560000	-124.23
25%	2.612150	18.000000	4.488618	1.004777	795.000000	2.432214	33.930000	-121.79
50%	3.595100	28.000000	5.300644	1.048780	1173.000000	2.815526	34.270000	-118.51
75%	4.813000	37.000000	6.083468	1.101280	1752.000000	3.265220	37.700000	-118.00
max	15.000100	52.000000	47.515152	11.181818	11935.000000	230.172414	41.860000	-114.56

In [16]:
```
#2 Provide a correlation matrix on the dataframe using the corr method.
name.corr()
```

Out[16]:

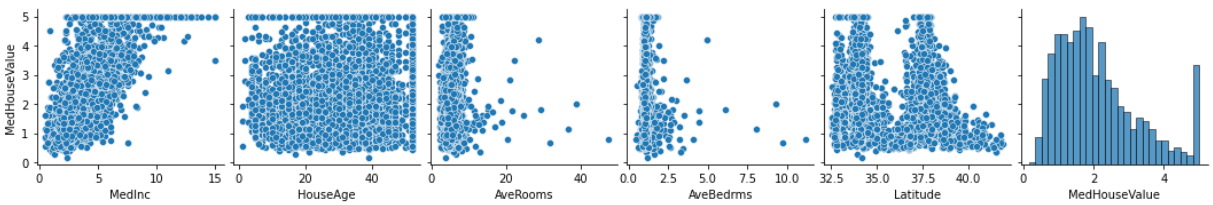
	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude
MedInc	1.000000	-0.098871	0.430025	-0.079185	-0.009787	0.016813	-0.054324	-0.044666
HouseAge	-0.098871	1.000000	-0.179765	-0.097652	-0.311671	0.031231	0.004006	-0.111073
AveRooms	0.430025	-0.179765	1.000000	0.742803	-0.094080	0.021448	0.137633	-0.059797
AveBedrms	-0.079185	-0.097652	0.742803	1.000000	-0.077782	0.005210	0.078649	0.010896
Population	-0.009787	-0.311671	-0.094080	-0.077782	1.000000	0.109704	-0.131691	0.132512
AveOccup	0.016813	0.031231	0.021448	0.005210	0.109704	1.000000	-0.015511	0.027616
Latitude	-0.054324	0.004006	0.137633	0.078649	-0.131691	-0.015511	1.000000	-0.921209
Longitude	-0.044666	-0.111073	-0.059797	0.010896	0.132512	0.027616	-0.921209	1.000000
MedHouseValue	0.683157	0.133851	0.186472	-0.066514	-0.034604	-0.049769	-0.148020	-0.048220

In [17]:
```
#3 Review median house value correlaton coefficients with the other variables
# Using the drop method, remove the three variables with the weakest correlati


drop = ["Population","AveOccup","Longitude"]
name.drop(columns=drop, inplace=True)
print(name)
```

```
      MedInc  HouseAge  AveRooms  AveBedrms  Latitude  MedHouseValue
0     3.2083      34.0  5.471264   1.063218     32.78          0.762
1     5.7049       9.0  6.699571   1.085837     34.66          1.732
2     1.6125      52.0  3.135135   1.364865     37.78          1.125
3     5.2586      17.0  6.945035   1.131206     38.69          1.370
4     5.0380       9.0  5.428415   0.967213     38.42          1.856
...      ...       ...       ...        ...       ...            ...
4123  5.7480      28.0  6.002427   1.029126     37.63          3.021
4124  2.0625      52.0  4.323353   1.026946     34.01          1.344
4125  2.1944      33.0  4.131313   0.984848     34.04          2.033
4126  5.6360      49.0  5.117647   0.894118     34.16          2.679
4127  3.3438      25.0  4.297674   1.108140     33.84          1.533

[4128 rows x 6 columns]
```

In [21]:
```
#4 Use seaborn's pairplot function on the data showing median house value on t
sns.pairplot(name, y_vars = "MedHouseValue")
```

Out[21]:
```
<seaborn.axisgrid.PairGrid at 0x7fb3078a6fd0>
```



In [27]:
```
#5 Use a for statement to run a simple learning regression model for each of t
#  features predicting the target (median house value)

for x in name.columns[:-1]:
    X =   name[[x]]   #feature variable
    y =   name.MedHouseValue   #target variable
    reg1 =  LinearRegression()        #instantiate the model
    reg1.fit(X, y)          #Train the model using X and y
    print(x, reg1.score(X, y))
```

```
MedInc 0.46670303605951935
HouseAge 0.017916166265603772
AveRooms 0.03477164709990799
AveBedrms 0.004424072299616277
Latitude 0.021909951040249176
```

In [44]:
```
rn1(sklearn) to do a train_test_split on the data
 ra2ndom_state to 433.
 mo3del using the LinearRegression estimator.
sin4g the train data.
re5on train data.
re6on test data.
  7
y_t8rain, y_test = train_test_split(name[["MedInc","HouseAge","AveRooms","AveBedrms"
es9ion()         # instantiate the model uing the estimator
 y10train)      # train the model
g211score(X_train, y_train))
g212score(X_test, y_test))
```

```
Train: 0.5572196613842659
Test: 0.5229219735139651
```

In [45]:
```
X_test[:1]
```

Out[45]:

	MedInc	HouseAge	AveRooms	AveBedrms	Latitude
403	3.75	29.0	5.318841	1.123188	34.04

In [47]:
```
#7 Use the predict method to find the target value (i.e., median house value)
# the features' values in the first row of X_test
reg2.predict([[3.75, 29.0,5.318841, 1.123188, 34.04]])
```

Out[47]:
```
array([2.14247342])
```

In [39]:
```
y_test[:1]
```

Out[39]:
```
403    1.906
Name: MedHouseValue, dtype: float64
```

In [41]:
```
#8 What is the amount difference between the predicted median house value from
# and the observed median house value? (found in y_test)
difference = 1.906 - 2.14247342
print(difference)
```

```
-0.23647342000000005
```

In [48]:
```
sume a model where the target variable is the MedHouseValue and the remaining colu
a 20-Fold cross-validation showing the ten scores. Also show the average of the te
= LinearRegression()
sults = cross_val_score(reg3, name[["MedInc","HouseAge","AveRooms","AveBedrms","Lat
(cv_results)
sults.mean()
```

```
[0.57357707 0.56383348 0.54648759 0.55034607 0.59756961 0.54713964
 0.51005965 0.54130444 0.52998809 0.49594595]
```

Out[48]:
```
0.5456251600969492
```

In [50]:
```
#10 What is the amount difference between the model score using the train-test
# the cross-validation approach?
difference1 = 0.57357707 - 0.5456251600969492
print(difference1)
```

```
0.027951909903050853
```

```
##### 11 Between reg2 and reg3, which model seems to be better in predicting median house values? Justify your answer.

reg2>reg3 is better because it is closer to 1
```

```
#### 12 Based on your analysis from # 5, which feature seems to have less weight in predict the target? Justify your answer.

Avebedroom. because it is closet to 0
```