ml-assignmet1

March 4, 2024

[5]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
```

[4]:
```
# a) Read the data with pandas and describe the data
data = pd.read_csv('/content/housing.csv')
print("Description of the data:")
print(data.describe())
```

```
Description of the data:
          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]:
```
# b) Find data type and shape of each column
print("\nData types of each column:")
print(data.dtypes)
print("\nShape of the data:")
print(data.shape)
```

```
Data types of each column:
longitude             float64
latitude              float64
housing_median_age    float64
total_rooms           float64
total_bedrooms        float64
population            float64
households            float64
median_income         float64
median_house_value    float64
ocean_proximity        object
dtype: object

Shape of the data:
(20640, 10)
```

[7]:
```
# c) Find the null values (if yes fill the null values with '0' or mean of that␣
  ↪column)
null_values = data.isnull().sum()
print("\nNull values in the data:")
print(null_values)

# Fill null values with mean of the column
data.fillna(data.mean(), inplace=True)
```

```
Null values in the data:
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

```
<ipython-input-7-a491f8faab6e>:7: FutureWarning: The default value of
numeric_only in DataFrame.mean is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  data.fillna(data.mean(), inplace=True)
```

[16]:
```
# d) find features and target variables
# Assuming the target variable is in the last column
features = data.iloc[:, :-1]
target = data.iloc[:, -1]
print(features)
print(target)
```

```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
0        -122.23     37.88                41.0        880.0           129.0
1        -122.22     37.86                21.0       7099.0          1106.0
2        -122.24     37.85                52.0       1467.0           190.0
3        -122.25     37.85                52.0       1274.0           235.0
4        -122.25     37.85                52.0       1627.0           280.0
…            …        …                    …          …               …
20635    -121.09     39.48                25.0       1665.0           374.0
20636    -121.21     39.49                18.0        697.0           150.0
20637    -121.22     39.43                17.0       2254.0           485.0
20638    -121.32     39.43                18.0       1860.0           409.0
20639    -121.24     39.37                16.0       2785.0           616.0

       population  households  median_income  median_house_value
0           322.0       126.0         8.3252            452600.0
1          2401.0      1138.0         8.3014            358500.0
2           496.0       177.0         7.2574            352100.0
3           558.0       219.0         5.6431            341300.0
4           565.0       259.0         3.8462            342200.0
…             …           …              …                 …
20635       845.0       330.0         1.5603             78100.0
20636       356.0       114.0         2.5568             77100.0
20637      1007.0       433.0         1.7000             92300.0
20638       741.0       349.0         1.8672             84700.0
20639      1387.0       530.0         2.3886             89400.0

[20640 rows x 9 columns]
0        NEAR BAY
1        NEAR BAY
2        NEAR BAY
3        NEAR BAY
4        NEAR BAY
          …
20635      INLAND
```

```
20636      INLAND
20637      INLAND
20638      INLAND
20639      INLAND
Name: ocean_proximity, Length: 20640, dtype: object
```

[17]:
```
# e) Split the data into train and test
X_train, X_test, y_train, y_test = train_test_split(features, target,␣
 ↪test_size=0.2, random_state=42)
print(X_train,y_train)
print(X_test,y_test)
```

```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
14196    -117.03     32.71                33.0       3126.0           627.0
8267     -118.16     33.77                49.0       3382.0           787.0
17445    -120.48     34.66                 4.0       1897.0           331.0
14265    -117.11     32.69                36.0       1421.0           367.0
2271     -119.80     36.78                43.0       2382.0           431.0
…            …         …                   …            …              …
11284    -117.96     33.78                35.0       1330.0           201.0
11964    -117.43     34.02                33.0       3084.0           570.0
5390     -118.38     34.03                36.0       2101.0           569.0
860      -121.96     37.58                15.0       3575.0           597.0
15795    -122.42     37.77                52.0       4226.0          1315.0

       population  households  median_income  median_house_value
14196      2300.0       623.0         3.2596            103000.0
8267       1314.0       756.0         3.8125            382100.0
17445       915.0       336.0         4.1563            172600.0
14265      1418.0       355.0         1.9425             93400.0
2271        874.0       380.0         3.5542             96500.0
…             …           …              …                  …
11284       658.0       217.0         6.3700            229200.0
11964      1753.0       449.0         3.0500             97800.0
5390       1756.0       527.0         2.9344            222100.0
860        1777.0       559.0         5.7192            283500.0
15795      2619.0      1242.0         2.5755            325000.0

[16512 rows x 9 columns] 14196    NEAR OCEAN
8267     NEAR OCEAN
17445    NEAR OCEAN
14265    NEAR OCEAN
2271         INLAND
            …
11284     <1H OCEAN
11964        INLAND
5390      <1H OCEAN
860       <1H OCEAN
```

```
15795      NEAR BAY
Name: ocean_proximity, Length: 16512, dtype: object
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
20046    -119.01     36.06                25.0       1505.0      537.870553
3024     -119.46     35.14                30.0       2943.0      537.870553
15663    -122.44     37.80                52.0       3830.0      537.870553
20484    -118.72     34.28                17.0       3051.0      537.870553
9814     -121.93     36.62                34.0       2351.0      537.870553
…           …        …                    …          …            …
15362    -117.22     33.36                16.0       3165.0      482.000000
16623    -120.83     35.36                28.0       4323.0      886.000000
18086    -122.05     37.31                25.0       4111.0      538.000000
2144     -119.76     36.77                36.0       2507.0      466.000000
3665     -118.37     34.22                17.0       1787.0      463.000000

       population  households  median_income  median_house_value
20046      1392.0       359.0         1.6812             47700.0
3024       1565.0       584.0         2.5313             45800.0
15663      1310.0       963.0         3.4801            500001.0
20484      1705.0       495.0         5.7376            218600.0
9814       1063.0       428.0         3.7250            278000.0
…           …           …             …                 …
15362      1351.0       452.0         4.6050            263300.0
16623      1650.0       705.0         2.7266            266800.0
18086      1585.0       568.0         9.2298            500001.0
2144       1227.0       474.0         2.7850             72300.0
3665       1671.0       448.0         3.5521            151500.0

[4128 rows x 9 columns] 20046        INLAND
3024         INLAND
15663      NEAR BAY
20484     <1H OCEAN
9814     NEAR OCEAN
           …
15362     <1H OCEAN
16623    NEAR OCEAN
18086     <1H OCEAN
2144         INLAND
3665      <1H OCEAN
Name: ocean_proximity, Length: 4128, dtype: object
```

[18]:
```python
# f) Normalize the data with min-max scaling
scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
print(X_train_scaled)
print(X_test_scaled)
```

```
[[0.72908367 0.01702128 0.62745098 … 0.10228581 0.19032151 0.18144461]
 [0.61653386 0.12978723 0.94117647 … 0.12415721 0.22845202 0.75690616]
 [0.38545817 0.22446809 0.05882353 … 0.05508962 0.25216204 0.32494918]
 …
 [0.59462151 0.15744681 0.68627451 … 0.08649893 0.16789424 0.42701061]
 [0.23804781 0.53510638 0.2745098  … 0.09176122 0.35994676 0.55360803]
 [0.19223108 0.55531915 1.         … 0.20407828 0.14314285 0.63917468]]
[[0.53187251 0.37340426 0.47058824 … 0.0588719  0.08146784 0.06742446]
 [0.48705179 0.27553191 0.56862745 … 0.09587239 0.14009462 0.06350695]
 [0.19023904 0.55851064 1.         … 0.15819766 0.2055282  1.        ]
 …
 [0.22908367 0.50638298 0.47058824 … 0.09324124 0.60205376 1.        ]
 [0.45717131 0.44893617 0.68627451 … 0.07778326 0.15759093 0.1181459 ]
 [0.59561753 0.17765957 0.31372549 … 0.07350765 0.21049365 0.2814442 ]]
```