

NAME: M. Siri Chandana

HT.NO:2203A51492

BATCH:12

AIML ASSIGNMENT – 5

Question 1:

<https://www.kaggle.com/datasets/camnugent/california-housing-prices>

Download the dataset from the above link.

a) Read the data with pandas and describe the data

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import MinMaxScaler
```

```
data = pd.read_csv('housing.csv')
```

```
data_description = data.describe()
```

```
print("Data description:\n", data_description)
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

# a) Read the data with pandas and describe the data
data = pd.read_csv('housing.csv')
data_description = data.describe()
print("Data description:\n", data_description)
```

Data description:

	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

completed at 8:13 PM

b) Find data type and shape of each column

```
import pandas as pd

from sklearn.model_selection import train_test_split

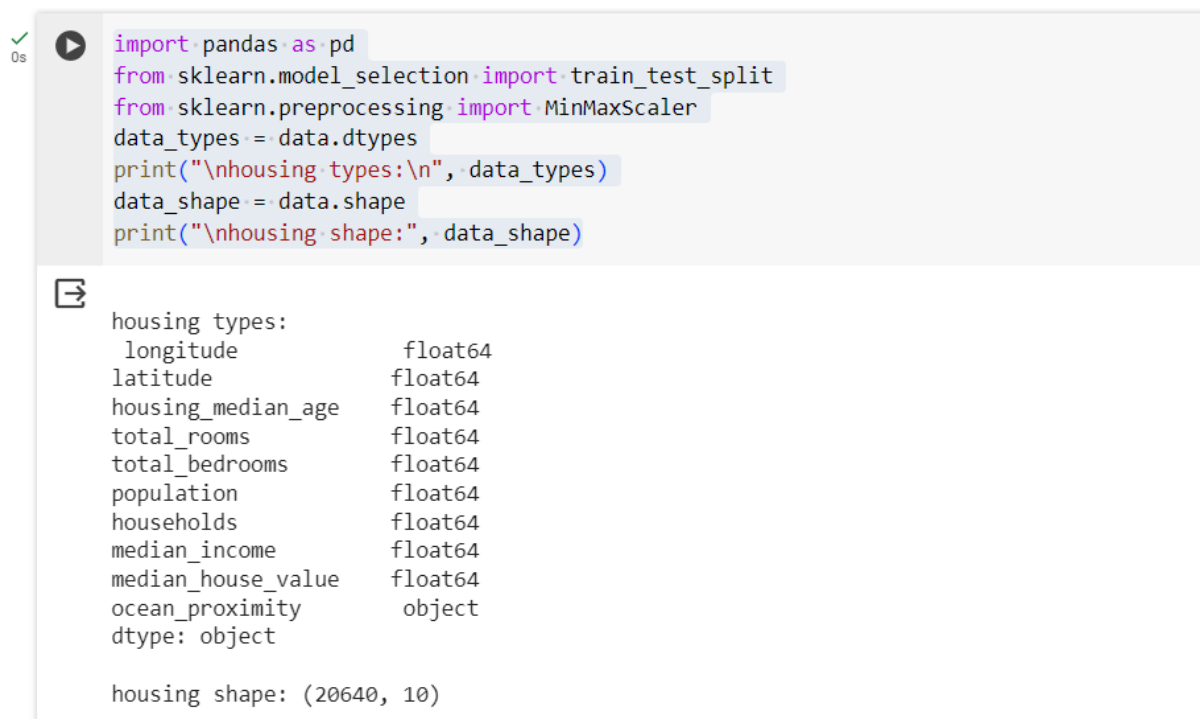
from sklearn.preprocessing import MinMaxScaler

data_types = data.dtypes

print("\nhousing types:\n", data_types)

data_shape = data.shape

print("\nhousing shape:", data_shape)
```

A screenshot of a Jupyter Notebook cell. The top part shows the code being executed, which imports pandas as pd, imports train_test_split from sklearn.model_selection, imports MinMaxScaler from sklearn.preprocessing, and then calculates data_types = data.dtypes, data_shape = data.shape, and prints them. The bottom part shows the output of the code, which is a table of data types for each column and the shape of the data. The output is as follows:

```
housing types:
  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

housing shape: (20640, 10)
```

c) Find the null values (if yes fill the null values with '0' or mean of that column)

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler

null_values = data.isnull().sum()

print("\nNull values:\n", null_values)
```

```
null_values = data.isnull().sum()
print("\nNull values:\n", null_values)
data_filled = data.fillna(data.mean())
```



A screenshot of a Jupyter Notebook interface. The top part shows a code cell with the following Python code: `import pandas as pd`, `from sklearn.model_selection import train_test_split`, `from sklearn.preprocessing import MinMaxScaler`, `null_values = data.isnull().sum()`, `print("\nNull values:\n", null_values)`, `null_values = data.isnull().sum()`, `print("\nNull values:\n", null_values)`, and `data_filled = data.fillna(data.mean())`. The code is highlighted with a light blue background. Below the code cell is an output cell showing the result of the `print` statements. It displays two identical text blocks, each showing the null values for various features: `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), and `ocean_proximity` (0). The output is formatted as a table. At the bottom right of the notebook interface, there is a status bar indicating that the code was completed at 8:13 PM.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
null_values = data.isnull().sum()
print("\nNull values:\n", null_values)
null_values = data.isnull().sum()
print("\nNull values:\n", null_values)
data_filled = data.fillna(data.mean())
```

```
Null values:
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

Null values:
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
```

✓ 0s completed at 8:13 PM

d) find features and target variables

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler

# Features are all columns except the target variable

features = data.drop(columns=["median_house_value"])
```

Target variable is "median_house_value"

target = data["median_house_value"]

✓
0s

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
# Features are all columns except the target variable
features = data.drop(columns=["median_house_value"])

# Target variable is "median_house_value"
target = data["median_house_value"]
```

e) Split the data into train and test.

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler

from sklearn.model_selection import train_test_split

Split the data into 80% train and 20% test

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2,
random_state=42)

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

      # Split the data into 80% train and 20% test
      X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)
```