AI ASSIGNMENT - 1

#preprocessing the dataset

import pandas as pd

# Load the dataset into a pandas dataframe

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

#check basic information

print(df.head())

print(df.info())

id Date number of bedrooms number of bathrooms living area \

0 6762810145 42491 5 2.50 3650

1 6762810635 42491 4 2.50 2920

2 6762810998 42491 5 2.75 2910

3 6762812605 42491 4 2.50 3310

4 6762812919 42491 3 2.00 2710

lot area number of floors waterfront present number of views \

0 9050 2.0 0 4

1 4000 1.5 0 0

2 9480 1.5 0 0

3 42998 2.0 0 0

4 4500 1.5 0 0

condition of the house ... Built Year Renovation Year Postal Code \

0 5 ... 1921 0 122003

1 5 ... 1909 0 122004

2 3 ... 1939 0 122004

3 3 ... 2001 0 122005

4 4 ... 1929 0 122006

Lattitude Longitude living\_area\_renov lot\_area\_renov \

0 52.8645 -114.557 2880 5400

1 52.8878 -114.470 2470 4000

2 52.8852 -114.468 2940 6600

3 52.9532 -114.321 3350 42847

4 52.9047 -114.485 2060 4500

Number of schools nearby Distance from the airport Price

0 2 58 2380000

1 2 51 1400000

2 1 53 1200000

3 3 76 838000

4 1 51 805000

[5 rows x 23 columns]

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 14620 entries, 0 to 14619

Data columns (total 23 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 id 14620 non-null int64

1 Date 14620 non-null int64

2 number of bedrooms 14620 non-null int64

3 number of bathrooms 14620 non-null float64

4 living area 14620 non-null int64

5 lot area 14620 non-null int64

6 number of floors 14620 non-null float64

7 waterfront present 14620 non-null int64

8 number of views 14620 non-null int64

9 condition of the house 14620 non-null int64

10 grade of the house 14620 non-null int64

11 Area of the house(excluding basement) 14620 non-null int64

12 Area of the basement 14620 non-null int64

13 Built Year 14620 non-null int64

14 Renovation Year 14620 non-null int64

15 Postal Code 14620 non-null int64

16 Lattitude 14620 non-null float64

17 Longitude 14620 non-null float64

18 living\_area\_renov 14620 non-null int64

19 lot\_area\_renov 14620 non-null int64

20 Number of schools nearby 14620 non-null int64

21 Distance from the airport 14620 non-null int64

22 Price 14620 non-null int64

dtypes: float64(4), int64(19)

memory usage: 2.6 MB

None

#drop the 'date' column

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

df.drop('Date', axis=1, inplace=True)

#convert the 'number of bedrooms' column to integer data type

df['number of bedrooms'] = df['number of bedrooms'].astype(int)

display(df)

id number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house grade of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 5 2.50 3650 9050 2.0 0 4 5 10 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 4 2.50 2920 4000 1.5 0 0 5 8 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 5 2.75 2910 9480 1.5 0 0 3 8 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 4 2.50 3310 42998 2.0 0 0 3 9 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 3 2.00 2710 4500 1.5 0 0 4 8 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 2 1.50 1556 20000 1.0 0 0 4 7 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 3 2.00 1680 7000 1.5 0 0 4 7 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 2 1.00 1070 6120 1.0 0 0 3 6 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 4 1.00 1030 6621 1.0 0 0 4 6 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 3 1.00 900 4770 1.0 0 0 3 6 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 22 columns

#check for missing values

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

print(df.isna().sum())

print(df)

id 0

Date 0

number of bedrooms 0

number of bathrooms 0

living area 0

lot area 0

number of floors 0

waterfront present 0

number of views 0

condition of the house 0

grade of the house 0

Area of the house(excluding basement) 0

Area of the basement 0

Built Year 0

Renovation Year 0

Postal Code 0

Lattitude 0

Longitude 0

living\_area\_renov 0

lot\_area\_renov 0

Number of schools nearby 0

Distance from the airport 0

Price 0

dtype: int64

id Date number of bedrooms number of bathrooms \

0 6762810145 42491 5 2.50

1 6762810635 42491 4 2.50

2 6762810998 42491 5 2.75

3 6762812605 42491 4 2.50

4 6762812919 42491 3 2.00

... ... ... ... ...

14615 6762830250 42734 2 1.50

14616 6762830339 42734 3 2.00

14617 6762830618 42734 2 1.00

14618 6762830709 42734 4 1.00

14619 6762831463 42734 3 1.00

living area lot area number of floors waterfront present \

0 3650 9050 2.0 0

1 2920 4000 1.5 0

2 2910 9480 1.5 0

3 3310 42998 2.0 0

4 2710 4500 1.5 0

... ... ... ... ...

14615 1556 20000 1.0 0

14616 1680 7000 1.5 0

14617 1070 6120 1.0 0

14618 1030 6621 1.0 0

14619 900 4770 1.0 0

number of views condition of the house ... Built Year \

0 4 5 ... 1921

1 0 5 ... 1909

2 0 3 ... 1939

3 0 3 ... 2001

4 0 4 ... 1929

... ... ... ... ...

14615 0 4 ... 1957

14616 0 4 ... 1968

14617 0 3 ... 1962

14618 0 4 ... 1955

14619 0 3 ... 1969

Renovation Year Postal Code Lattitude Longitude living\_area\_renov \

0 0 122003 52.8645 -114.557 2880

1 0 122004 52.8878 -114.470 2470

2 0 122004 52.8852 -114.468 2940

3 0 122005 52.9532 -114.321 3350

4 0 122006 52.9047 -114.485 2060

... ... ... ... ... ...

14615 0 122066 52.6191 -114.472 2250

14616 0 122072 52.5075 -114.393 1540

14617 0 122056 52.7289 -114.507 1130

14618 0 122042 52.7157 -114.411 1420

14619 2009 122018 52.5338 -114.552 900

lot\_area\_renov Number of schools nearby Distance from the airport \

0 5400 2 58

1 4000 2 51

2 6600 1 53

3 42847 3 76

4 4500 1 51

... ... ... ...

14615 17286 3 76

14616 7480 3 59

14617 6120 2 64

14618 6631 3 54

14619 3480 2 55

Price

0 2380000

1 1400000

2 1200000

3 838000

4 805000

... ...

14615 221700

14616 219200

14617 209000

14618 205000

14619 146000

[14620 rows x 23 columns]

#drop the row with missing values

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

df.dropna(inplace=True)

display(df)

id Date number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 42491 5 2.50 3650 9050 2.0 0 4 5 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 42491 4 2.50 2920 4000 1.5 0 0 5 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 42491 5 2.75 2910 9480 1.5 0 0 3 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 42491 4 2.50 3310 42998 2.0 0 0 3 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 42491 3 2.00 2710 4500 1.5 0 0 4 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 0 4 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 0 4 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 0 3 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 0 4 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 42734 3 1.00 900 4770 1.0 0 0 3 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 23 columns

#impute missing values with mean

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

df.fillna(df.mean(), inplace=True)

#impute missing values with the median

df.fillna(df.median(), inplace=True)

display(df)

id Date number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 42491 5 2.50 3650 9050 2.0 0 4 5 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 42491 4 2.50 2920 4000 1.5 0 0 5 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 42491 5 2.75 2910 9480 1.5 0 0 3 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 42491 4 2.50 3310 42998 2.0 0 0 3 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 42491 3 2.00 2710 4500 1.5 0 0 4 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 0 4 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 0 4 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 0 3 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 0 4 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 42734 3 1.00 900 4770 1.0 0 0 3 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 23 columns

#convert categorial variable into numerical variables using one-hot encoding

df = pd.get\_dummies(df, columns=['waterfront present', 'Built Year', 'Postal Code'])

display(df)

id Date number of bedrooms number of bathrooms living area lot area number of floors number of views condition of the house grade of the house ... Postal Code\_122063 Postal Code\_122064 Postal Code\_122065 Postal Code\_122066 Postal Code\_122067 Postal Code\_122068 Postal Code\_122069 Postal Code\_122070 Postal Code\_122071 Postal Code\_122072

0 6762810145 42491 5 2.50 3650 9050 2.0 4 5 10 ... 0 0 0 0 0 0 0 0 0 0

1 6762810635 42491 4 2.50 2920 4000 1.5 0 5 8 ... 0 0 0 0 0 0 0 0 0 0

2 6762810998 42491 5 2.75 2910 9480 1.5 0 3 8 ... 0 0 0 0 0 0 0 0 0 0

3 6762812605 42491 4 2.50 3310 42998 2.0 0 3 9 ... 0 0 0 0 0 0 0 0 0 0

4 6762812919 42491 3 2.00 2710 4500 1.5 0 4 8 ... 0 0 0 0 0 0 0 0 0 0

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 4 7 ... 0 0 0 1 0 0 0 0 0 0

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 4 7 ... 0 0 0 0 0 0 0 0 0 1

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 3 6 ... 0 0 0 0 0 0 0 0 0 0

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 4 6 ... 0 0 0 0 0 0 0 0 0 0

14619 6762831463 42734 3 1.00 900 4770 1.0 0 3 6 ... 0 0 0 0 0 0 0 0 0 0

14620 rows × 208 columns

#split the dataset into training and testing sets

from sklearn.model\_selection import train\_test\_split

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

X = df.drop('Price', axis=1)

y = df['Price']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

display(df)

id Date number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 42491 5 2.50 3650 9050 2.0 0 4 5 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 42491 4 2.50 2920 4000 1.5 0 0 5 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 42491 5 2.75 2910 9480 1.5 0 0 3 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 42491 4 2.50 3310 42998 2.0 0 0 3 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 42491 3 2.00 2710 4500 1.5 0 0 4 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 0 4 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 0 4 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 0 3 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 0 4 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 42734 3 1.00 900 4770 1.0 0 0 3 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 23 columns

#building the ANN model

from keras.models import Sequential

from keras.layers import Dense

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

#define the model architecture

model = Sequential()

model.add(Dense(64, input\_dim=X\_train.shape[1], activation='relu'))

model.add(Dense(32, activation='relu'))

model.add(Dense(1, activation='linear'))

#compile the model

model.compile(loss='mean\_squared\_error', optimizer='adam', metrics=['mean\_squared\_error'])

#train the model

model.fit(X\_train, y\_train, epochs=50, batch\_size=32)

display(df)

Epoch 1/50

366/366 [==============================] - 2s 3ms/step - loss: 200632132698112.0000 - mean\_squared\_error: 200632132698112.0000

Epoch 2/50

366/366 [==============================] - 1s 3ms/step - loss: 192855572480.0000 - mean\_squared\_error: 192855572480.0000

Epoch 3/50

366/366 [==============================] - 1s 2ms/step - loss: 337801805824.0000 - mean\_squared\_error: 337801805824.0000

Epoch 4/50

366/366 [==============================] - 1s 2ms/step - loss: 2269407608832.0000 - mean\_squared\_error: 2269407608832.0000

Epoch 5/50

366/366 [==============================] - 1s 2ms/step - loss: 30485159870464.0000 - mean\_squared\_error: 30485159870464.0000

Epoch 6/50

366/366 [==============================] - 1s 2ms/step - loss: 13260391710720.0000 - mean\_squared\_error: 13260391710720.0000

Epoch 7/50

366/366 [==============================] - 1s 2ms/step - loss: 20270043627520.0000 - mean\_squared\_error: 20270043627520.0000

Epoch 8/50

366/366 [==============================] - 1s 2ms/step - loss: 20007740243968.0000 - mean\_squared\_error: 20007740243968.0000

Epoch 9/50

366/366 [==============================] - 1s 2ms/step - loss: 22758824083456.0000 - mean\_squared\_error: 22758824083456.0000

Epoch 10/50

366/366 [==============================] - 1s 2ms/step - loss: 18359703830528.0000 - mean\_squared\_error: 18359703830528.0000

Epoch 11/50

366/366 [==============================] - 1s 2ms/step - loss: 19714839412736.0000 - mean\_squared\_error: 19714839412736.0000

Epoch 12/50

366/366 [==============================] - 1s 2ms/step - loss: 19308413452288.0000 - mean\_squared\_error: 19308413452288.0000

Epoch 13/50

366/366 [==============================] - 1s 2ms/step - loss: 18749774102528.0000 - mean\_squared\_error: 18749774102528.0000

Epoch 14/50

366/366 [==============================] - 1s 2ms/step - loss: 18597451661312.0000 - mean\_squared\_error: 18597451661312.0000

Epoch 15/50

366/366 [==============================] - 1s 2ms/step - loss: 19730008113152.0000 - mean\_squared\_error: 19730008113152.0000

Epoch 16/50

366/366 [==============================] - 1s 2ms/step - loss: 17491349733376.0000 - mean\_squared\_error: 17491349733376.0000

Epoch 17/50

366/366 [==============================] - 1s 3ms/step - loss: 17773098958848.0000 - mean\_squared\_error: 17773098958848.0000

Epoch 18/50

366/366 [==============================] - 1s 3ms/step - loss: 16122558545920.0000 - mean\_squared\_error: 16122558545920.0000

Epoch 19/50

366/366 [==============================] - 1s 3ms/step - loss: 19060121141248.0000 - mean\_squared\_error: 19060121141248.0000

Epoch 20/50

366/366 [==============================] - 1s 3ms/step - loss: 235399439974400.0000 - mean\_squared\_error: 235399439974400.0000

Epoch 21/50

366/366 [==============================] - 1s 3ms/step - loss: 214273982464.0000 - mean\_squared\_error: 214273982464.0000

Epoch 22/50

366/366 [==============================] - 1s 2ms/step - loss: 278393257984.0000 - mean\_squared\_error: 278393257984.0000

Epoch 23/50

366/366 [==============================] - 1s 2ms/step - loss: 414115856384.0000 - mean\_squared\_error: 414115856384.0000

Epoch 24/50

366/366 [==============================] - 1s 2ms/step - loss: 931880304640.0000 - mean\_squared\_error: 931880304640.0000

Epoch 25/50

366/366 [==============================] - 1s 2ms/step - loss: 15196713123840.0000 - mean\_squared\_error: 15196713123840.0000

Epoch 26/50

366/366 [==============================] - 1s 2ms/step - loss: 17709668499456.0000 - mean\_squared\_error: 17709668499456.0000

Epoch 27/50

366/366 [==============================] - 1s 2ms/step - loss: 16694032465920.0000 - mean\_squared\_error: 16694032465920.0000

Epoch 28/50

366/366 [==============================] - 1s 2ms/step - loss: 15147645009920.0000 - mean\_squared\_error: 15147645009920.0000

Epoch 29/50

366/366 [==============================] - 1s 2ms/step - loss: 15231653773312.0000 - mean\_squared\_error: 15231653773312.0000

Epoch 30/50

366/366 [==============================] - 1s 2ms/step - loss: 15058609373184.0000 - mean\_squared\_error: 15058609373184.0000

Epoch 31/50

366/366 [==============================] - 1s 2ms/step - loss: 14874152271872.0000 - mean\_squared\_error: 14874152271872.0000

Epoch 32/50

366/366 [==============================] - 1s 2ms/step - loss: 15957810479104.0000 - mean\_squared\_error: 15957810479104.0000

Epoch 33/50

366/366 [==============================] - 1s 2ms/step - loss: 14542455177216.0000 - mean\_squared\_error: 14542455177216.0000

Epoch 34/50

366/366 [==============================] - 1s 2ms/step - loss: 14233113722880.0000 - mean\_squared\_error: 14233113722880.0000

Epoch 35/50

366/366 [==============================] - 1s 3ms/step - loss: 14346975444992.0000 - mean\_squared\_error: 14346975444992.0000

Epoch 36/50

366/366 [==============================] - 1s 3ms/step - loss: 14532047011840.0000 - mean\_squared\_error: 14532047011840.0000

Epoch 37/50

366/366 [==============================] - 1s 3ms/step - loss: 14912918126592.0000 - mean\_squared\_error: 14912918126592.0000

Epoch 38/50

366/366 [==============================] - 1s 3ms/step - loss: 12931882287104.0000 - mean\_squared\_error: 12931882287104.0000

Epoch 39/50

366/366 [==============================] - 1s 2ms/step - loss: 14798681014272.0000 - mean\_squared\_error: 14798681014272.0000

Epoch 40/50

366/366 [==============================] - 1s 2ms/step - loss: 11656067284992.0000 - mean\_squared\_error: 11656067284992.0000

Epoch 41/50

366/366 [==============================] - 1s 2ms/step - loss: 13240746639360.0000 - mean\_squared\_error: 13240746639360.0000

Epoch 42/50

366/366 [==============================] - 1s 2ms/step - loss: 17319711473664.0000 - mean\_squared\_error: 17319711473664.0000

Epoch 43/50

366/366 [==============================] - 1s 2ms/step - loss: 9401361498112.0000 - mean\_squared\_error: 9401361498112.0000

Epoch 44/50

366/366 [==============================] - 1s 2ms/step - loss: 12246226829312.0000 - mean\_squared\_error: 12246226829312.0000

Epoch 45/50

366/366 [==============================] - 1s 2ms/step - loss: 12621315047424.0000 - mean\_squared\_error: 12621315047424.0000

Epoch 46/50

366/366 [==============================] - 1s 2ms/step - loss: 11537283547136.0000 - mean\_squared\_error: 11537283547136.0000

Epoch 47/50

366/366 [==============================] - 1s 2ms/step - loss: 13200380657664.0000 - mean\_squared\_error: 13200380657664.0000

Epoch 48/50

366/366 [==============================] - 1s 2ms/step - loss: 11707709652992.0000 - mean\_squared\_error: 11707709652992.0000

Epoch 49/50

366/366 [==============================] - 1s 2ms/step - loss: 13745899175936.0000 - mean\_squared\_error: 13745899175936.0000

Epoch 50/50

366/366 [==============================] - 1s 2ms/step - loss: 8161715027968.0000 - mean\_squared\_error: 8161715027968.0000

id Date number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 42491 5 2.50 3650 9050 2.0 0 4 5 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 42491 4 2.50 2920 4000 1.5 0 0 5 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 42491 5 2.75 2910 9480 1.5 0 0 3 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 42491 4 2.50 3310 42998 2.0 0 0 3 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 42491 3 2.00 2710 4500 1.5 0 0 4 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 0 4 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 0 4 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 0 3 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 0 4 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 42734 3 1.00 900 4770 1.0 0 0 3 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 23 columns

#testing the model

df = pd.read\_csv('/content/drive/MyDrive/House Price India.csv')

#evaluate the model on the test set

test\_loss, test\_acc = model.evaluate(X\_test, y\_test)

#print the test accuracy

print('Test Accuracy:', test\_acc)

display(df)

92/92 [==============================] - 0s 3ms/step - loss: 7946286661632.0000 - mean\_squared\_error: 7946286661632.0000

Test Accuracy: 7946286661632.0

id Date number of bedrooms number of bathrooms living area lot area number of floors waterfront present number of views condition of the house ... Built Year Renovation Year Postal Code Lattitude Longitude living\_area\_renov lot\_area\_renov Number of schools nearby Distance from the airport Price

0 6762810145 42491 5 2.50 3650 9050 2.0 0 4 5 ... 1921 0 122003 52.8645 -114.557 2880 5400 2 58 2380000

1 6762810635 42491 4 2.50 2920 4000 1.5 0 0 5 ... 1909 0 122004 52.8878 -114.470 2470 4000 2 51 1400000

2 6762810998 42491 5 2.75 2910 9480 1.5 0 0 3 ... 1939 0 122004 52.8852 -114.468 2940 6600 1 53 1200000

3 6762812605 42491 4 2.50 3310 42998 2.0 0 0 3 ... 2001 0 122005 52.9532 -114.321 3350 42847 3 76 838000

4 6762812919 42491 3 2.00 2710 4500 1.5 0 0 4 ... 1929 0 122006 52.9047 -114.485 2060 4500 1 51 805000

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

14615 6762830250 42734 2 1.50 1556 20000 1.0 0 0 4 ... 1957 0 122066 52.6191 -114.472 2250 17286 3 76 221700

14616 6762830339 42734 3 2.00 1680 7000 1.5 0 0 4 ... 1968 0 122072 52.5075 -114.393 1540 7480 3 59 219200

14617 6762830618 42734 2 1.00 1070 6120 1.0 0 0 3 ... 1962 0 122056 52.7289 -114.507 1130 6120 2 64 209000

14618 6762830709 42734 4 1.00 1030 6621 1.0 0 0 4 ... 1955 0 122042 52.7157 -114.411 1420 6631 3 54 205000

14619 6762831463 42734 3 1.00 900 4770 1.0 0 0 3 ... 1969 2009 122018 52.5338 -114.552 900 3480 2 55 146000

14620 rows × 23 columns