```
In [1]:
    import pandas
as pd import numpy
as np import random
as rnd
import seaborn as sns
    import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train test split
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import mean squared error, mean absolute error, explained variance sco
from sklearn.metrics import classification report, confusion matrix
In [2]:
df = pd.read csv('/content/House Price India.csv')
In [3]:
print(df.columns.values)
['id' 'Date' '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'
 'Area of the house (excluding basement)' 'Area of the basement'
 'Built Year' 'Renovation Year' 'Postal Code' 'Lattitude'
 'Longitude' 'living_area_renov' 'lot_area_renov' 'Number of schools
 nearby' 'Distance from the airport' 'Price']
In [4]:
df.head()
```

b	Date	number of bedrooms	of	living area	ea lot area		number of floors	waterfront present	number of views	condition of the house		Built Year	Renovatio n	
			bathroom s										Ро	Yea r
														С
	0 6	762810145 42	2491	5	2.50	3650	9050	2.0	0	4	5 1921		0 122	
	1 6	762810635 42	2491	4	2.50	2920	4000	1.5	0	0	5 1909		0 122	
	2 6	762810998 42	2491	5	2.75	2910	9480	1.5	0	0	3 1939		0 122	
	3 6	762812605 42	2491	4	2.50	3310	42998	2.0	0	0	3 2001		0 122	
	4 6	762812919 42	2491	3	2.00	2710	4500	1.5	0	0	4 1929		0 122	

5 rows × 23 columns

Team ID: NM2023TMID06755 Team Member: P.Subbulakshmi

Out[4]:

id

	df.tail()										
	Out[5	number of	number of	living area	lot area	number	waterfront		condition	Built Year	Renov
id	Date	bedrooms	bathrooms			of	present	of	of the		ation Year
	1										
											F

14615	676283025 ^{id} 0	4D27a3te4 bedrooms ² s	bat	hroom ¹ ·	11 v 111 q 1 a 5 re 5 a 6	20 a 0 r e ⁰ ea ⁰	number 10.0f floors	p resen	numberco o Of views	of the4 Y1Y9e5a7r	Yea 0
14616	6/62830339	42/34	3	2.0	1680	a /UUU	1.5	U	U	4 1968	U
14617	6762830618	42734	2	1.0	1070	6120	1.0	0	0	3 1962	0
14618	6762830709	42734	4	1.0	1030	6621	1.0	0	0	4 1955	0
14619	6762831463	42734	3	1.0	900	4770	1.0	0	0	3 1969	2009

floors

views

house

5 rows × 23 columns

```
Out[6]:
df.isnull().sum()
id
Date
                                            0
number of bedrooms
                                            0
number of bathrooms
                                            0
                                            0
living area
                                            0
lot area
number of floors
                                            0
waterfront present
                                            0
number of views
condition of the house
grade of the house
Area of the house (excluding basement)
Area of the basement
                                            0
                                            0
Built Year
Renovation Year
                                            0
Postal Code
                                            0
Lattitude
                                            0
Longitude
                                            0
living area renov
                                            0
lot area renov
                                            0
Number of schools nearby
                                            0
                                            0
Distance from the airport
Price
dtype: int64
```

In [7]:

In [9]:

Y=dataset[:,22]

```
Qut[10]:
array([2380000., 1400000., 1200000., ..., 209000., 205000., 146000.])
In [11]:
min max. scaler=MinMaxScaler()
X scale=min max scaler fit transform(X)
Out[12:
X_scale:
                           , 0.125
                                                              , 0.26666667,
                                           , ..., 0.5
array([[0.00578811, 0.
        0.30202047],
                               , 0.09375 , ..., 0.5 , 0.03333333,
       [0.0284775 , 0.
        0.17344529],
       [0.04528616, 0.
                               , 0.125 , ..., 0.
                                                              , 0.1
        0.14720546],
       . . . ,
                                                              , 0.46666667,
       [0.95378774, 1.
                               , 0.03125 , ..., 0.5
        0.01718709],
       [0.95800148, 1.
                              , 0.09375 , ..., 1. , 0.13333333,
       0.01666229],
       [0.99291535, 1.
                               , 0.0625 , ..., 0.5 , 0.16666667,
        0.00892154]])
In [13]:
Xntrain, X_val_and_test, Y_train, Y_val_and_test=train_test_split(X_scale, Y,
test_size=0.3)
X_nval_5; X_test, Y_test=train_test_split(X_test and test, Y_test and test,
test size=0 5)
(10234, 23) (2193, 23) (2193, 23) (10234,) (2193,) (2193,)
print(X_train.shape,X_val.shape,X_test.shape,Y_train.shape,Y_val.shape,Y_test.shape
In [16]:
made175equential([
    Danco (32 activation=!rolu! innut chano=(23 )) Danco (32 a
mpdelscompile(optimizer='sgd',
              logg-thingry areasontropy! motrice-[loggraph]]
hist=model.fit(X train,Y train,batch size=32,ep
          ochs=100, validation data=(X val, Y val
```

In [10]:

```
Epoch 1/100
320/320 [=============== ] - 2s 4ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 2/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 3/100
                =======] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [======
- val loss: nan - val accuracy:
0.0000e+00 Epoch 4/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 5/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 6/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 7/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 8/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 9/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 10/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 11/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 12/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 13/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 14/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 15/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 16/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 17/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 18/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 19/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 20/100
320/320 [=============== ] - 1s 4ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 21/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 22/100
                =======] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [=======
- val loss: nan - val accuracy:
0.0000e+00 Epoch 23/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 24/100
- val loss: nan - val accuracy: 0.0000e+00
```

```
Epoch 25/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 26/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 27/100
                  =======] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [========
- val loss: nan - val accuracy:
0.0000e+00 Epoch 28/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 29/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 30/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 31/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 32/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 33/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 34/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 35/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 36/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 37/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 38/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 39/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 40/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 41/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 42/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 43/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 44/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 45/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 46/100
                 =======] - 1s 4ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [========
- val loss: nan - val accuracy:
0.0000e+00 Epoch 47/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 48/100
- val loss: nan - val accuracy: 0.0000e+00
```

```
Epoch 49/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 50/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 51/100
                 =======] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [=======
- val loss: nan - val accuracy:
0.0000e+00 Epoch 52/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 53/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 54/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 55/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 56/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 57/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 58/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 59/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 60/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 61/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 62/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 63/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 64/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 65/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 66/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 67/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 68/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 69/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 70/100
            320/320 [========
- val loss: nan - val accuracy:
0.0000e+00 Epoch 71/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 72/100
- val loss: nan - val accuracy: 0.0000e+00
```

```
Epoch 73/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 74/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 75/100
                 ======] - 1s 4ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [=======
- val loss: nan - val accuracy:
0.0000e+00 Epoch 76/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 77/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 78/100
320/320 [============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 79/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 80/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 81/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 82/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 83/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 84/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 85/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 86/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 87/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 88/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 89/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 90/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 91/100
320/320 [=============== ] - 1s 3ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 92/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 93/100
320/320 [=============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val accuracy:
0.0000e+00 Epoch 94/100
                 =======] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [=======
- val loss: nan - val accuracy:
0.0000e+00 Epoch 95/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 96/100
- val loss: nan - val accuracy: 0.0000e+00
```

```
Epoch 97/100
320/320 [============== ] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
- val loss: nan - val_accuracy:
0.0000e+00 Epoch 98/100
- val loss: nan - val accuracy:
0.0000e+00 Epoch 99/100
                       ======] - 1s 2ms/step - loss: nan - accuracy: 0.0000e+00
320/320 [========
- val_loss: nan - val_accuracy:
0.0000e+00 Epoch 100/100
- val loss: nan - val accuracy: 0.0000e+00
In [19]:
model.evaluate(X test, Y test)[1]
69/69 [============= ] - Os 2ms/step - loss: nan - accuracy: 0.0000e+00
Out[19]:
0.0
In [20]:
a = model.predict(X)
457/457 [==========] - 1s 1ms/step
In [21]:
a[:10]
Out[21]:
array([[nan],
     [nan],
     [nan],
     [nan],
     [nan],
     [nan],
     [nan],
     [nan],
     [nan],
     [nan]], dtype=float32)
```