```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from keras.models import Sequential
from keras.layers import Dense


from google.colab import files
uploaded = files.upload()
```

- **House Price India.csv**(text/csv) - 1524561 bytes, last modified: 5/15/2023 - 100% done

```
Saving House Price India.csv to House Price India.csv
```

```
print(uploaded)
```

```
{'House Price India.csv': b'id,Date,number of bedrooms,number of bathrooms,living area,lot area,number of floors,waterfront present,numb
```

```
dt=pd.read_csv("House Price India.csv")
dt
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4
1	6762810635	42491	4	2.50	2920	4000	1.5	0	0
2	6762810998	42491	5	2.75	2910	9480	1.5	0	0
3	6762812605	42491	4	2.50	3310	42998	2.0	0	0
								0	0
...	...	...	...	...	...	...	...	...	...
14615	6762830250	42734	2	1.50	1556	20000	1.0	0	0
14616	6762830339	42734	3	2.00	1680	7000	1.5	0	0
14617	6762830618	42734	2	1.00	1070	6120	1.0	0	0
14618	6762830709	42734	4	1.00	1030	6621	1.0	0	0
14619	6762831463	42734	3	1.00	900	4770	1.0	0	0

Automatic saving failed. This file was updated remotely or in another tab. Show diff

14620 rows × 23 columns

```
dt.head()
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfront present	number of views	co
0	6762810145	42491	5	2.50	3650	9050	2.0	0	4	
1	6762810635	42491	4	2.50	2920	4000	1.5	0	0	
2	6762810998	42491	5	2.75	2910	9480	1.5	0	0	
3	6762812605	42491	4	2.50	3310	42998	2.0	0	0	
4	6762812919	42491	3	2.00	2710	4500	1.5	0	0	

5 rows × 23 columns

```
dt.info()
```

```
<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
```

```
dt.describe()
```

aterfront present	number of views	condition of the house	...	Built Year	Renovation Year	Postal Code	La
20.000000	14620.000000	14620.000000	...	14620.000000	14620.000000	14620.000000	1462
0.087193	0.766259	0.664151	...	29.493625	416.216661	19.082418	(
0.000000	0.000000	1.000000	...	1900.000000	0.000000	122003.000000	5
0.000000	0.000000	3.000000	...	1951.000000	0.000000	122017.000000	5
0.000000	0.000000	3.000000	...	1975.000000	0.000000	122032.000000	5
0.000000	0.000000	4.000000	...	1997.000000	0.000000	122048.000000	5
1.000000	4.000000	5.000000	...	2015.000000	2015.000000	122072.000000	5

Automatic saving failed. This file was updated remotely or in another tab. Show diff

2033.062244

```
feat=dt.columns[2:22]
X=dt[feat]
y=dt["Price"]


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)


model = Sequential()

model.add(Dense(units=64, activation='relu', input_dim=X_train.shape[1]))


model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=128, activation='relu'))
model.add(Dense(units=1, activation='linear'))


model.compile(optimizer='adam', loss='mean_squared_error')
```

```
model.fit(X_train_scaled, y_train, batch_size=32, epochs=100, verbose=1)
```

```
Epoch 25/100
366/366 [==============================] - 1s 2ms/step - loss: 29752340480.0000
Epoch 26/100
366/366 [==============================] - 1s 2ms/step - loss: 29548716032.0000
Epoch 27/100
366/366 [==============================] - 1s 2ms/step - loss: 29519638528.0000
Epoch 28/100
366/366 [==============================] - 1s 2ms/step - loss: 29353168896.0000
Epoch 29/100
366/366 [==============================] - 1s 2ms/step - loss: 29261303808.0000
Epoch 30/100
366/366 [==============================] - 1s 3ms/step - loss: 29109972992.0000
Epoch 31/100
366/366 [==============================] - 1s 3ms/step - loss: 28984788992.0000
Epoch 32/100
366/366 [==============================] - 1s 3ms/step - loss: 28903006208.0000
Epoch 33/100
366/366 [==============================] - 1s 2ms/step - loss: 28809162752.0000
Epoch 34/100
366/366 [==============================] - 1s 2ms/step - loss: 28762785792.0000
Epoch 35/100
366/366 [==============================] - 1s 2ms/step - loss: 28634490880.0000
Epoch 36/100
366/366 [==============================] - 1s 2ms/step - loss: 28726990848.0000
Epoch 37/100
366/366 [==============================] - 1s 2ms/step - loss: 28456185856.0000
Epoch 38/100
366/366 [==============================] - 1s 2ms/step - loss: 28257912832.0000
Epoch 39/100
366/366 [==============================] - 1s 2ms/step - loss: 28263297024.0000
Epoch 40/100
366/366 [==============================] - 1s 2ms/step - loss: 28216397824.0000
Epoch 41/100
366/366 [==============================] - 1s 2ms/step - loss: 28163231744.0000
Epoch 42/100
366/366 [==============================] - 1s 2ms/step - loss: 27989772288.0000
```



```
912.0000
Epoch 44/100
366/366 [==============================] - 1s 2ms/step - loss: 27839526912.0000
Epoch 45/100
366/366 [==============================] - 1s 2ms/step - loss: 27849678848.0000
Epoch 46/100
366/366 [==============================] - 1s 2ms/step - loss: 27872190464.0000
Epoch 47/100
366/366 [==============================] - 1s 3ms/step - loss: 27677267968.0000
Epoch 48/100
366/366 [==============================] - 1s 3ms/step - loss: 27618424832.0000
Epoch 49/100
366/366 [==============================] - 1s 3ms/step - loss: 27717744640.0000
Epoch 50/100
366/366 [==============================] - 1s 2ms/step - loss: 27468634112.0000
Epoch 51/100
366/366 [==============================] - 1s 2ms/step - loss: 27435974656.0000
Epoch 52/100
366/366 [==============================] - 1s 2ms/step - loss: 27261542400.0000
Epoch 53/100
366/366 [==============================] - 1s 2ms/step - loss: 27299522560.0000
Epoch 54/100
```

```
loss = model.evaluate(X_test_scaled, y_test, verbose=0)
print("Mean Squared Error on Test Set:", loss)
```

```
Mean Squared Error on Test Set: 25925464064.0
```

```
predictions = model.predict(X_test_scaled)
print(predictions)
```

```
92/92 [==============================] - 0s 1ms/step
[[396410.66]
 [587008.6 ]
 [481035.53]
 ...
 [998544.9 ]
 [288535.88]
 [496524.84]]
```

✓ 1s completed at 08:25

Automatic saving failed. This file was updated remotely or in another tab. Show diff