```
import pandas as pd
import numpy as np
import random as rnd

# visualization
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# scaling and train test split
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler

# creating a model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.optimizers import Adam

# evaluation on test data
from sklearn.metrics import mean_squared_error,mean_absolute_error,explained_variance
from sklearn.metrics import classification_report,confusion_matrix


df = pd.read_csv('/content/drive/MyDrive/House Price India.csv')



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']
```

```
# preview the data
df.head()
```

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

5 rows × 23 columns

```
# preview the data
df.tail()
```

	id	Date	number of bedrooms	number of bathrooms	living area	lot area	number of floors	waterfron presen
14615	6762830250	42734	2	1.5	1556	20000	1.0	
14616	6762830339	42734	3	2.0	1680	7000	1.5	
14617	6762830618	42734	2	1.0	1070	6120	1.0	
14618	6762830709	42734	4	1.0	1030	6621	1.0	
14619	6762831463	42734	3	1.0	900	4770	1.0	

5 rows × 23 columns

```
df.isnull().sum()
```

```
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
```

```
df.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
```

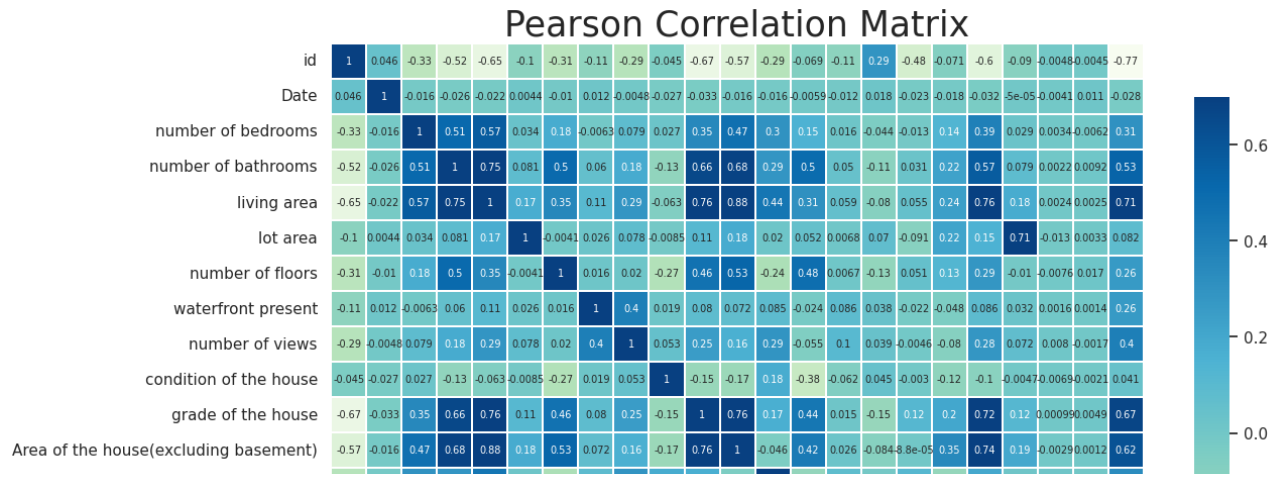
```
df.describe().transpose()
```

	count	mean	std	min	25%	
id	14620.0	6.762821e+09	6237.574799	6.762810e+09	6.762815e+09	6
Date	14620.0	4.260454e+04	67.347991	4.249100e+04	4.254600e+04	4
number of bedrooms	14620.0	3.379343e+00	0.938719	1.000000e+00	3.000000e+00	3
number of bathrooms	14620.0	2.129583e+00	0.769934	5.000000e-01	1.750000e+00	2
living area	14620.0	2.098263e+03	928.275721	3.700000e+02	1.440000e+03	1
lot area	14620.0	1.509328e+04	37919.621304	5.200000e+02	5.010750e+03	7
number of floors	14620.0	1.502360e+00	0.540239	1.000000e+00	1.000000e+00	1
waterfront	14620.0	7.660739e-03	0.087193	0.000000e+00	0.000000e+00	0

```
sns.set(style="whitegrid", font_scale=1)

plt.figure(figsize=(13,13))
plt.title('Pearson Correlation Matrix',fontsize=25)
sns.heatmap(df.corr(),linewidths=0.25,vmax=0.7,square=True,cmap="GnBu",linecolor='w',
            annot=True, annot_kws={"size":7}, cbar_kws={"shrink": .7})
```

```
<Axes: title={'center': 'Pearson Correlation Matrix'}>
```



```
price_corr = df.corr()
print(price_corr)
```

```
                                                id      Date  number of bedrooms  \
id                                        1.000000  0.045966           -0.329034
Date                                      0.045966  1.000000           -0.015663
number of bedrooms                       -0.329034 -0.015663            1.000000
number of bathrooms                      -0.516909 -0.026485            0.509784
living area                              -0.648127 -0.021958            0.570526
lot area                                 -0.100269  0.004392            0.034416
number of floors                         -0.312305 -0.010335            0.177294
waterfront present                       -0.112937  0.012006           -0.006257
number of views                          -0.293004 -0.004782            0.078665
condition of the house                   -0.045061 -0.027402            0.026597
grade of the house                       -0.673448 -0.033097            0.352945
Area of the house(excluding basement)    -0.565116 -0.015994            0.473599
Area of the basement                     -0.290806 -0.015711            0.300332
Built Year                               -0.068645 -0.005869            0.152954
Renovation Year                          -0.109155 -0.011636            0.016132
Postal Code                               0.294709  0.018243           -0.044156
Lattitude                                -0.479334 -0.023327           -0.013163
Longitude                                -0.070841 -0.018231            0.135712
living_area_renov                        -0.599900 -0.032495            0.389855
lot_area_renov                           -0.089604 -0.000050            0.029400
Number of schools nearby                 -0.004821 -0.004071            0.003397
Distance from the airport                -0.004542  0.011457           -0.006157
Price                                    -0.773114 -0.027919            0.308460

                                         number of bathrooms  living area  \
id                                                 -0.516909    -0.648127
Date                                               -0.026485    -0.021958
number of bedrooms                                  0.509784     0.570526
number of bathrooms                                 1.000000     0.753517
living area                                         0.753517     1.000000
lot area                                            0.080806     0.174420
number of floors                                    0.502924     0.354743
waterfront present                                  0.060104     0.105837
number of views                                     0.183789     0.287728
condition of the house                             -0.128232    -0.063358
grade of the house                                  0.663054     0.761835
```

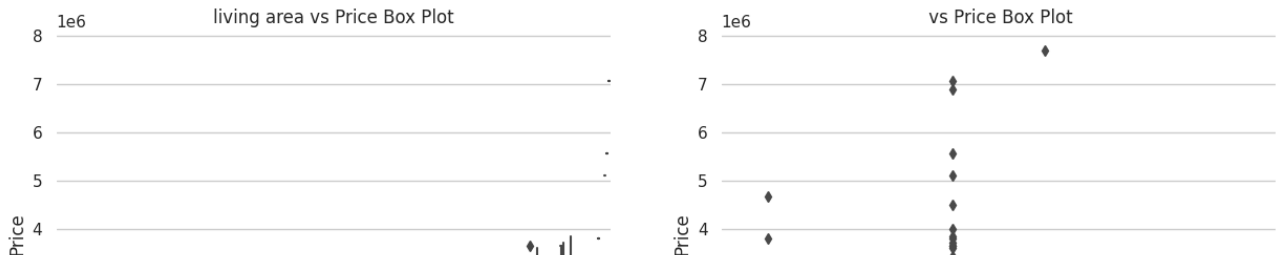
```
Area of the house(excluding basement)          0.684391     0.875793
Area of the basement                           0.287190     0.441491
Built Year                                     0.498127     0.309602
Renovation Year                                0.049669     0.059400
Postal Code                                   -0.105546    -0.080303
Lattitude                                      0.031156     0.054518
Longitude                                      0.223904     0.240208
living_area_renov                              0.570530     0.757571
lot_area_renov                                 0.078627     0.180312
Number of schools nearby                       0.002180     0.002370
Distance from the airport                      0.009206     0.002511
Price                                          0.531735     0.712169

                                            lot area  number of floors  \
id                                         -0.100269         -0.312305
Date                                        0.004392         -0.010335
number of bedrooms                          0.034416          0.177294
number of bathrooms                         0.080806          0.502924
living area                                 0.174420          0.354743
lot area                                    1.000000         -0.004138
number of floors                           -0.004138          1.000000
waterfront present                          0.026282          0.016316
```

```
sns.set(style="whitegrid", font_scale=1)

f, axes = plt.subplots(1, 2,figsize=(15,5))
sns.boxplot(x=df['living area'],y=df['Price'], ax=axes[0])
sns.boxplot(x=df['number of floors'],y=df['Price'], ax=axes[1])
sns.despine(bottom=True, left=True)
axes[0].set(xlabel='living area', ylabel='Price', title='living area vs Price Box Plo
axes[1].set(xlabel='number of floors', ylabel='Price', title=' vs Price Box Plot')
```

```
[Text(0.5, 0, 'number of floors'),
 Text(0, 0.5, 'Price'),
 Text(0.5, 1.0, ' vs Price Box Plot')]
```

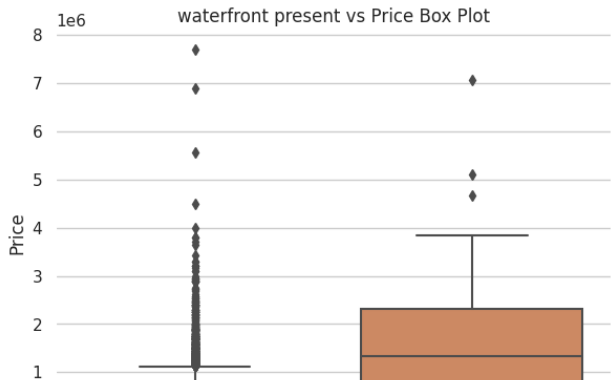


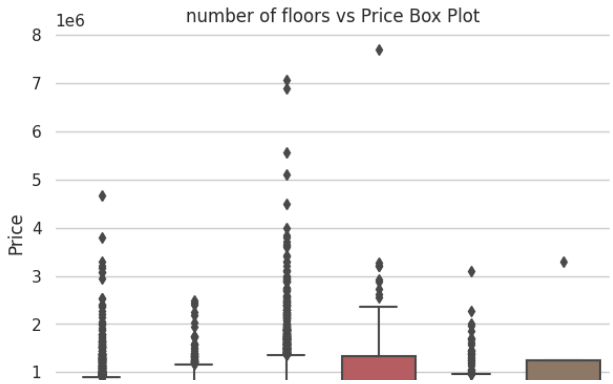

```
f, axes = plt.subplots(1, 2,figsize=(15,5))
sns.boxplot(x=df['waterfront present'],y=df['Price'], ax=axes[0])
sns.boxplot(x=df['number of floors'],y=df['Price'], ax=axes[1])
```

```
sns.despine(left=True, bottom=True)
axes[0].set(xlabel='waterfront present', ylabel='Price', title='waterfront present vs
axes[1].set(xlabel='number of floors', ylabel='Price', title='number of floors vs Pri

f, axe = plt.subplots(1, 1,figsize=(15,5))
sns.boxplot(x=df['grade of the house'],y=df['Price'], ax=axe)
sns.despine(left=True, bottom=True)
axe.set(xlabel='Grade', ylabel='Price', title='Grade vs Price Box Plot')
```

```
[Text(0.5, 0, 'Grade'),
 Text(0, 0.5, 'Price'),
 Text(0.5, 1.0, 'Grade vs Price Box Plot')]
```






```
df = df.drop('Postal Code', axis=1)
df = df.drop('lot area',axis=1)


df['Date'] = pd.to_datetime(df['Date'])
```

```
df['month'] = df['Date'].apply(lambda date:date.month)
df['year'] = df['Date'].apply(lambda date:date.year)

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

# Check the new columns
print(df.columns.values)
```

```
['number of bedrooms' 'number of bathrooms' 'living 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' 'Lattitude' 'Longitude'
 'living_area_renov' 'lot_area_renov' 'Number of schools nearby'
 'Distance from the airport' 'Price' 'month' 'year']
```

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

# Label
y = df['Price']

# Split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,random_state=10
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(10234, 20)
(4386, 20)
(10234,)
(4386,)
```

```
scaler = MinMaxScaler()

# fit and transfrom
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# everything has been scaled between 1 and 0
print('Max: ',X_train.max())
print('Min: ', X_train.min())
```

```
Max:  1.0000000000000002
Min:  0.0
```

```
model = Sequential()

# input layer
model.add(Dense(19,activation='relu'))

# hidden layers
model.add(Dense(19,activation='relu'))
model.add(Dense(19,activation='relu'))
model.add(Dense(19,activation='relu'))

# output layer
```

```
model.add(Dense(1))

model.compile(optimizer='adam',loss='mse')



model.fit(x=X_train,y=y_train.values,
          validation_data=(X_test,y_test.values),
          batch_size=128,
          epochs=400,
          verbose=2)
```

```
Epoch 1/400
80/80 - 2s - loss: 425401450496.0000 - val_loss: 425759834112.0000 - 2s/epoch - 
Epoch 2/400
80/80 - 0s - loss: 425223225344.0000 - val_loss: 425189736448.0000 - 333ms/epoch
Epoch 3/400
80/80 - 0s - loss: 423075872768.0000 - val_loss: 420195893248.0000 - 322ms/epoch
Epoch 4/400
80/80 - 0s - loss: 411137409024.0000 - val_loss: 398059700224.0000 - 316ms/epoch
Epoch 5/400
80/80 - 0s - loss: 371945504768.0000 - val_loss: 337670799360.0000 - 206ms/epoch
Epoch 6/400
80/80 - 0s - loss: 288151928832.0000 - val_loss: 231803879424.0000 - 207ms/epoch
Epoch 7/400
80/80 - 0s - loss: 178511429632.0000 - val_loss: 133073543168.0000 - 226ms/epoch
Epoch 8/400
80/80 - 0s - loss: 112337772544.0000 - val_loss: 102041051136.0000 - 209ms/epoch
Epoch 9/400
80/80 - 0s - loss: 99842580480.0000 - val_loss: 99232309248.0000 - 221ms/epoch -
Epoch 10/400
80/80 - 0s - loss: 98124103680.0000 - val_loss: 97883815936.0000 - 223ms/epoch -
Epoch 11/400
80/80 - 0s - loss: 96622043136.0000 - val_loss: 96390930432.0000 - 225ms/epoch -
Epoch 12/400
80/80 - 0s - loss: 95040888832.0000 - val_loss: 94858723328.0000 - 201ms/epoch -
Epoch 13/400
80/80 - 0s - loss: 93416529920.0000 - val_loss: 93233651712.0000 - 224ms/epoch -
Epoch 14/400
80/80 - 0s - loss: 91684757504.0000 - val_loss: 91578900480.0000 - 206ms/epoch -
Epoch 15/400
80/80 - 0s - loss: 89940025344.0000 - val_loss: 89774481408.0000 - 218ms/epoch -
Epoch 16/400
80/80 - 0s - loss: 88102903808.0000 - val_loss: 88037498880.0000 - 227ms/epoch -
Epoch 17/400
80/80 - 0s - loss: 86223413248.0000 - val_loss: 86146310144.0000 - 208ms/epoch -
Epoch 18/400
80/80 - 0s - loss: 84279771136.0000 - val_loss: 84245405696.0000 - 223ms/epoch -
Epoch 19/400
80/80 - 0s - loss: 82268717056.0000 - val_loss: 82269331456.0000 - 226ms/epoch -
Epoch 20/400
80/80 - 0s - loss: 80172343296.0000 - val_loss: 80242098176.0000 - 207ms/epoch -
Epoch 21/400
80/80 - 0s - loss: 78057365504.0000 - val_loss: 78123130880.0000 - 217ms/epoch -
Epoch 22/400
80/80 - 0s - loss: 75879964672.0000 - val_loss: 75995299840.0000 - 257ms/epoch -
Epoch 23/400
80/80 - 0s - loss: 73651929088.0000 - val_loss: 73840271360.0000 - 246ms/epoch -
Epoch 24/400
80/80 - 0s - loss: 71414980608.0000 - val_loss: 71657275392.0000 - 222ms/epoch -
Epoch 25/400
80/80 - 0s - loss: 69177393152.0000 - val_loss: 69455020032.0000 - 217ms/epoch -
```

```
Epoch 26/400
80/80 - 0s - loss: 66930868224.0000 - val_loss: 67291369472.0000 - 233ms/epoch -
Epoch 27/400
80/80 - 0s - loss: 64734511104.0000 - val_loss: 65313955840.0000 - 201ms/epoch -
Epoch 28/400
80/80 - 0s - loss: 62659198976.0000 - val_loss: 63283535872.0000 - 196ms/epoch -
Epoch 29/400
80/80 - 0s - loss: 60650418176.0000 - val_loss: 61439680512.0000 - 221ms/epoch -
Epoch 30/400
```

```
losses = pd.DataFrame(model.history.history)

plt.figure(figsize=(15,5))
sns.lineplot(data=losses,lw=3)
plt.xlabel('Epochs')
plt.ylabel('')
plt.title('Training Loss per Epoch')
sns.despine()
```


```
predictions = model.predict(X_test)

print('MAE: ',mean_absolute_error(y_test,predictions))
print('MSE: ',mean_squared_error(y_test,predictions))
print('RMSE: ',np.sqrt(mean_squared_error(y_test,predictions)))
print('Variance Regression Score: ',explained_variance_score(y_test,predictions))

print('\n\nDescriptive Statistics:\n',df['Price'].describe())
```

```
138/138 [==============================] - 0s 1ms/step
MAE:  103940.57364697332
MSE:  28979052391.70136
RMSE:  170232.34825291391
Variance Regression Score:  0.7849347519748286


Descriptive Statistics:
 count    1.462000e+04
```

```
mean     5.389322e+05
std      3.675324e+05
min      7.800000e+04
25%      3.200000e+05
50%      4.500000e+05
75%      6.450000e+05
max      7.700000e+06
Name: Price, dtype: float64
```

```
single_house = df.drop('Price',axis=1).iloc[0]
print(f'Features of new house:\n{single_house}')

# reshape the numpy array and scale the features
single_house = scaler.transform('grade of the house'.values.reshape(-1, 19))

# run the model and get the price prediction
print('\nPrediction Price:',model.predict(single_house)[0,0])

# original price
print('\nOriginal Price:',df.iloc[0]['Price'])
```

```
Features of new house:
number of bedrooms                      5.0000
number of bathrooms                     2.5000
living area                          3650.0000
number of floors                        2.0000
waterfront present                      0.0000
number of views                         4.0000
```

Colab paid products - Cancel contracts here