IMPORT LIBRARIES AND DATASETS

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!pip install jupyterthemes
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-w
Collecting jupyterthemes
  Downloading jupyterthemes-0.20.0-py2.py3-none-any.whl (7.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.0/7.0 MB 64.8 MB/s eta 0:00:0
Requirement already satisfied: jupyter-core in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: ipython>=5.4.1 in /usr/local/lib/python3.8/dis
Requirement already satisfied: matplotlib>=1.4.3 in /usr/local/lib/python3.8/
Requirement already satisfied: notebook>=5.6.0 in /usr/local/lib/python3.8/di
Collecting lesscpy>=0.11.2
  Downloading lesscpy-0.15.1-py2.py3-none-any.whl (46 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 46.7/46.7 KB 5.0 MB/s eta 0:00:
Collecting jedi>=0.10
  Downloading jedi-0.18.2-py2.py3-none-any.whl (1.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 82.0 MB/s eta 0:00:0
Requirement already satisfied: backcall in /usr/local/lib/python3.8/dist-pack
Requirement already satisfied: decorator in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.8/dis
Requirement already satisfied: pygments in /usr/local/lib/python3.8/dist-pack
Requirement already satisfied: pexpect in /usr/local/lib/python3.8/dist-packa
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.8/d
Requirement already satisfied: prompt-toolkit<2.1.0,>=2.0.0 in /usr/local/lib
Requirement already satisfied: pickleshare in /usr/local/lib/python3.8/dist-p
Collecting ply
  Downloading ply-3.11-py2.py3-none-any.whl (49 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.6/49.6 KB 5.7 MB/s eta 0:00:
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.8/
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /u
Requirement already satisfied: tornado<7,>=4.1 in /usr/local/lib/python3.8/di
Requirement already satisfied: jupyter-client<7.0.0,>=5.2.0 in /usr/local/lib
Requirement already satisfied: nbformat in /usr/local/lib/python3.8/dist-pack
Requirement already satisfied: pyzmq>=17 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.8/d
Requirement already satisfied: terminado>=0.8.1 in /usr/local/lib/python3.8/d
Requirement already satisfied: nbconvert<6.0 in /usr/local/lib/python3.8/dist-
```

```
Requirement already satisfied: jinja2<=3.0.0 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: prometheus-client in /usr/local/lib/python3.8/d
Requirement already satisfied: ipykernel in /usr/local/lib/python3.8/dist-pack
Requirement already satisfied: Send2Trash in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.8/d
Requirement already satisfied: parso<0.9.0,>=0.8.0 in /usr/local/lib/python3.8
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.8/di
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.8/d
Requirement already satisfied: bleach in /usr/local/lib/python3.8/dist-package
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.
Requirement already satisfied: defusedxml in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.8/
Requirement already satisfied: testpath in /usr/local/lib/python3.8/dist-packa
Requirement already satisfied: fastjsonschema in /usr/local/lib/python3.8/dist
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.8/dis
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: wcwidth in /usr/local/lib/python3.8/dist-packag
Requirement already satisfied: ptyprocess in /usr/local/lib/python3.8/dist-pac
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 i
Requirement already satisfied: importlib-resources>=1.4.0 in /usr/local/lib/py
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.8/dist-
Requirement already satisfied: webencodings in /usr/local/lib/python3.8/dist-p
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.8/dist-pa
```

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


from jupyterthemes import jtplot
jtplot.style(theme='monokai', context='notebook', ticks=True, grid=False)
# setting the style of the notebook to be monokai theme
# this line of code is important to ensure that we are able to see the x and y a
# If you don't run this code line, you will notice that the xlabel and ylabel or

```

```python
house_df = pd.read_csv('/content/drive/MyDrive/real estate /realestate_prices.cs
```

```
house_df
```

	id	date	price	bedrooms	bathrooms	sqft_living	sq:
0	7129300520	20141013T000000	221900.0	3	1.00	1180	
1	6414100192	20141209T000000	538000.0	3	2.25	2570	
2	5631500400	20150225T000000	180000.0	2	1.00	770	
3	2487200875	20141209T000000	604000.0	4	3.00	1960	
4	1954400510	20150218T000000	510000.0	3	2.00	1680	
...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	
21611	291310100	20150116T000000	400000.0	3	2.50	1600	
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	

21613 rows × 21 columns

```
house_df.head(5)
```

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_l
0	7129300520	20141013T000000	221900.0	3	1.00	1180	56
1	6414100192	20141209T000000	538000.0	3	2.25	2570	72
2	5631500400	20150225T000000	180000.0	2	1.00	770	100
3	2487200875	20141209T000000	604000.0	4	3.00	1960	50
4	1954400510	20150218T000000	510000.0	3	2.00	1680	80

5 rows × 21 columns

```
house_df.tail(10)
```

	id	date	price	bedrooms	bathrooms	sqft_living	s
21603	7852140040	20140825T000000	507250.0	3	2.50	2270	
21604	9834201367	20150126T000000	429000.0	3	2.00	1490	
21605	3448900210	20141014T000000	610685.0	4	2.50	2520	
21606	7936000429	20150326T000000	1007500.0	4	3.50	3510	
21607	2997800021	20150219T000000	475000.0	3	2.50	1310	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	
21611	291310100	20150116T000000	400000.0	3	2.50	1600	
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	

10 rows × 21 columns

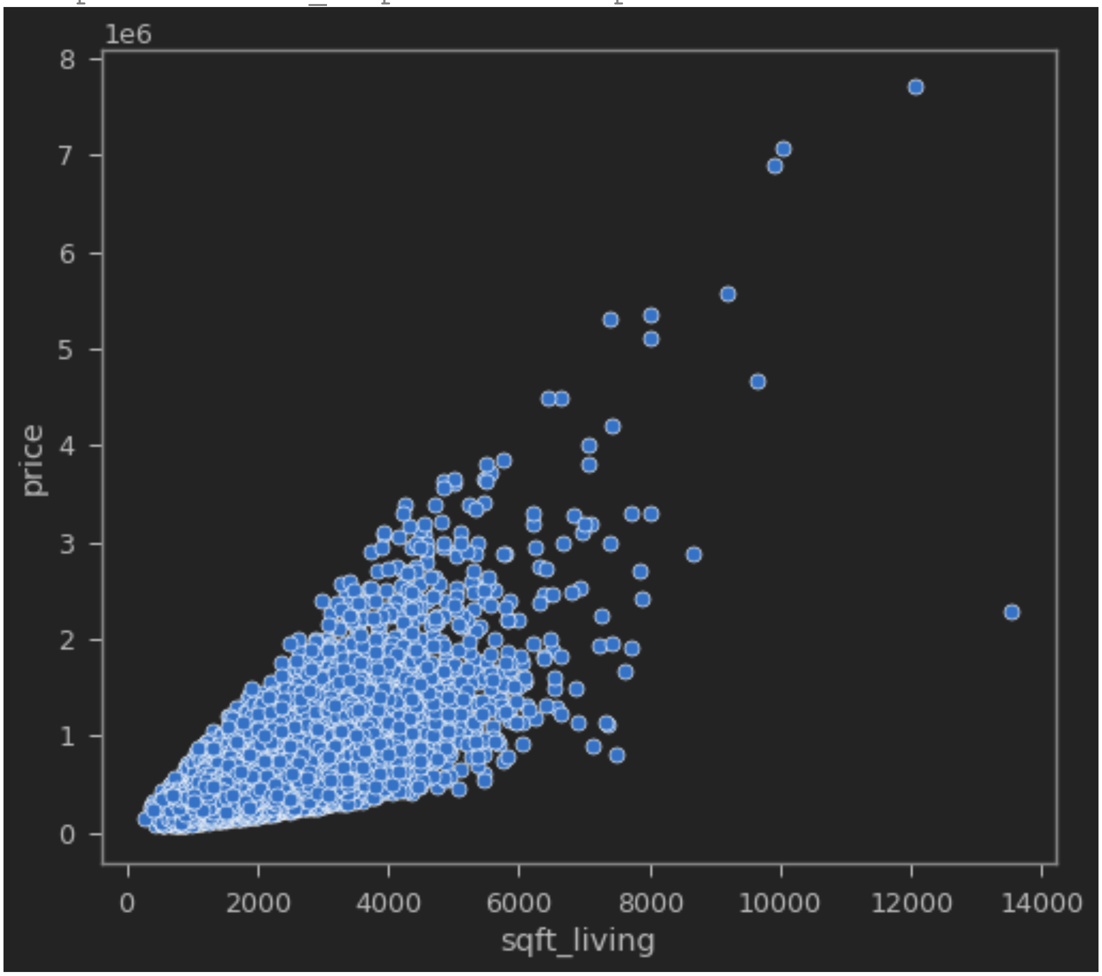
```
house_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   id             21613 non-null  int64  
 1   date           21613 non-null  object 
 2   price          21613 non-null  float64
 3   bedrooms       21613 non-null  int64  
 4   bathrooms      21613 non-null  float64
 5   sqft_living    21613 non-null  int64  
 6   sqft_lot       21613 non-null  int64  
 7   floors         21613 non-null  float64
 8   waterfront     21613 non-null  int64  
 9   view           21613 non-null  int64  
 10  condition      21613 non-null  int64  
 11  grade          21613 non-null  int64  
 12  sqft_above     21613 non-null  int64  
 13  sqft_basement  21613 non-null  int64  
 14  yr_built       21613 non-null  int64  
 15  yr_renovated   21613 non-null  int64  
 16  zipcode        21613 non-null  int64  
 17  lat            21613 non-null  float64
 18  long           21613 non-null  float64
 19  sqft_living15  21613 non-null  int64  
 20  sqft_lot15     21613 non-null  int64  
dtypes: float64(5), int64(15), object(1)
memory usage: 3.5+ MB
```

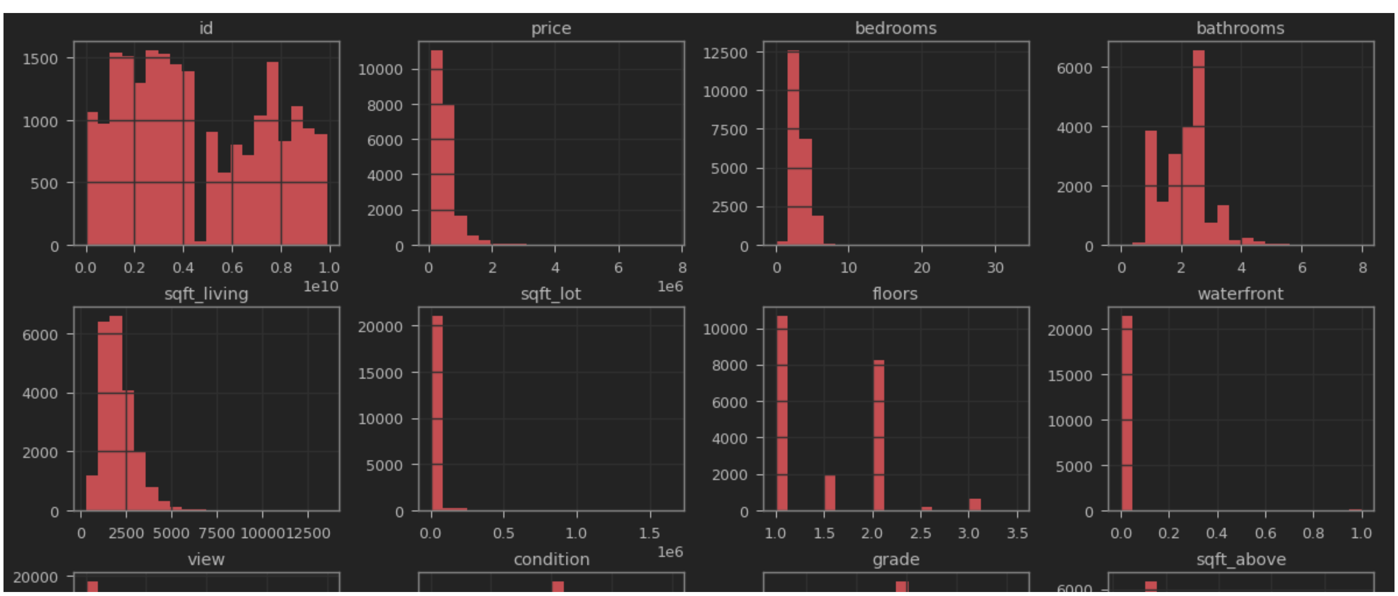
PERFORM DATA VISUALIZATION

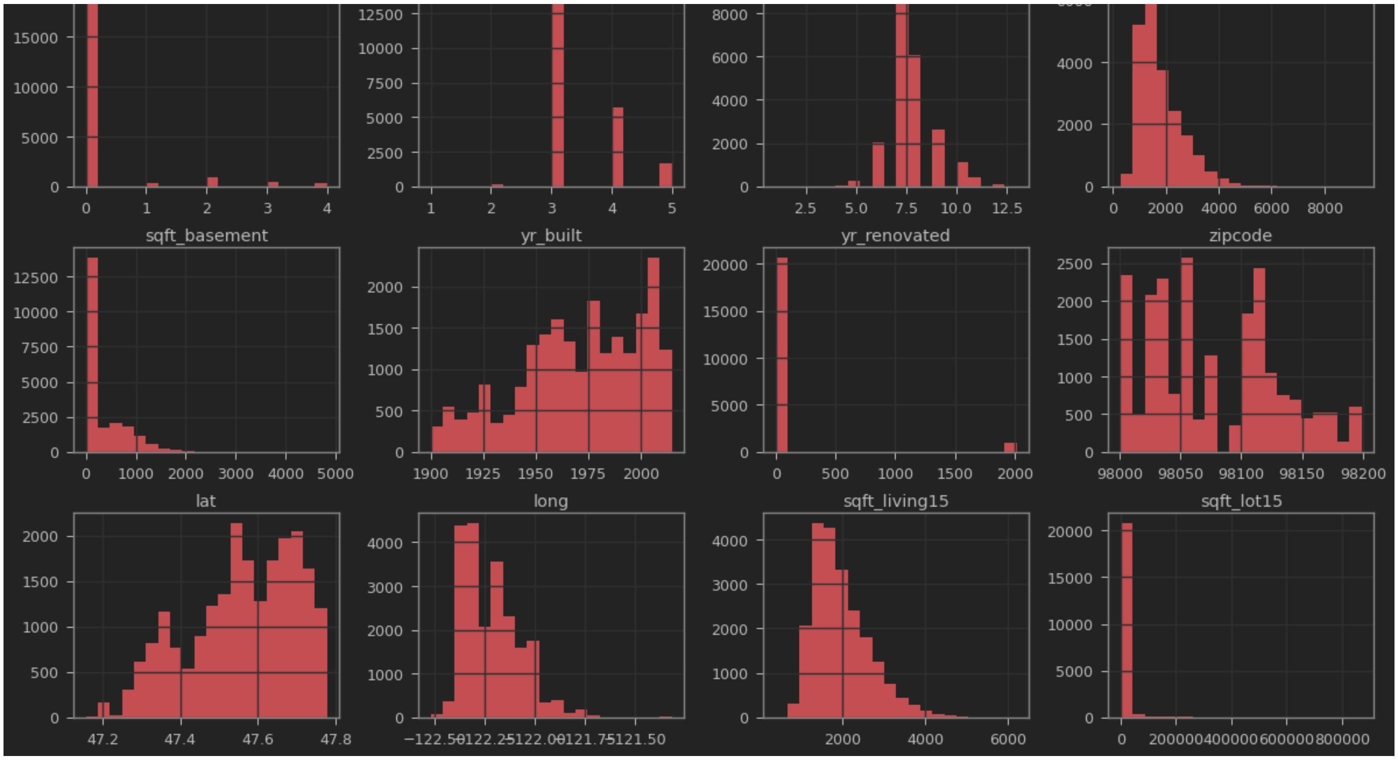
```
sns.scatterplot(x = 'sqft_living', y = 'price', data = house_df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9a9a51c8b0>
```



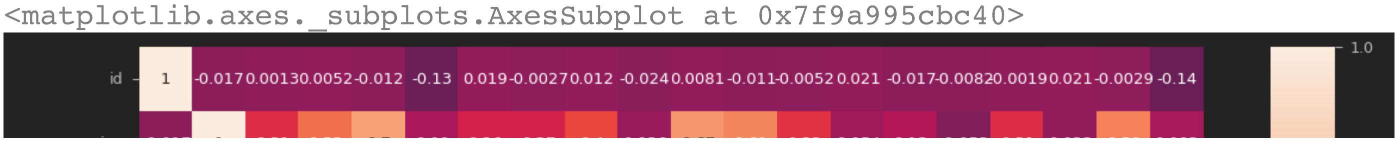

```
house_df.hist(bins = 20, figsize = (20,20), color = 'r');
```

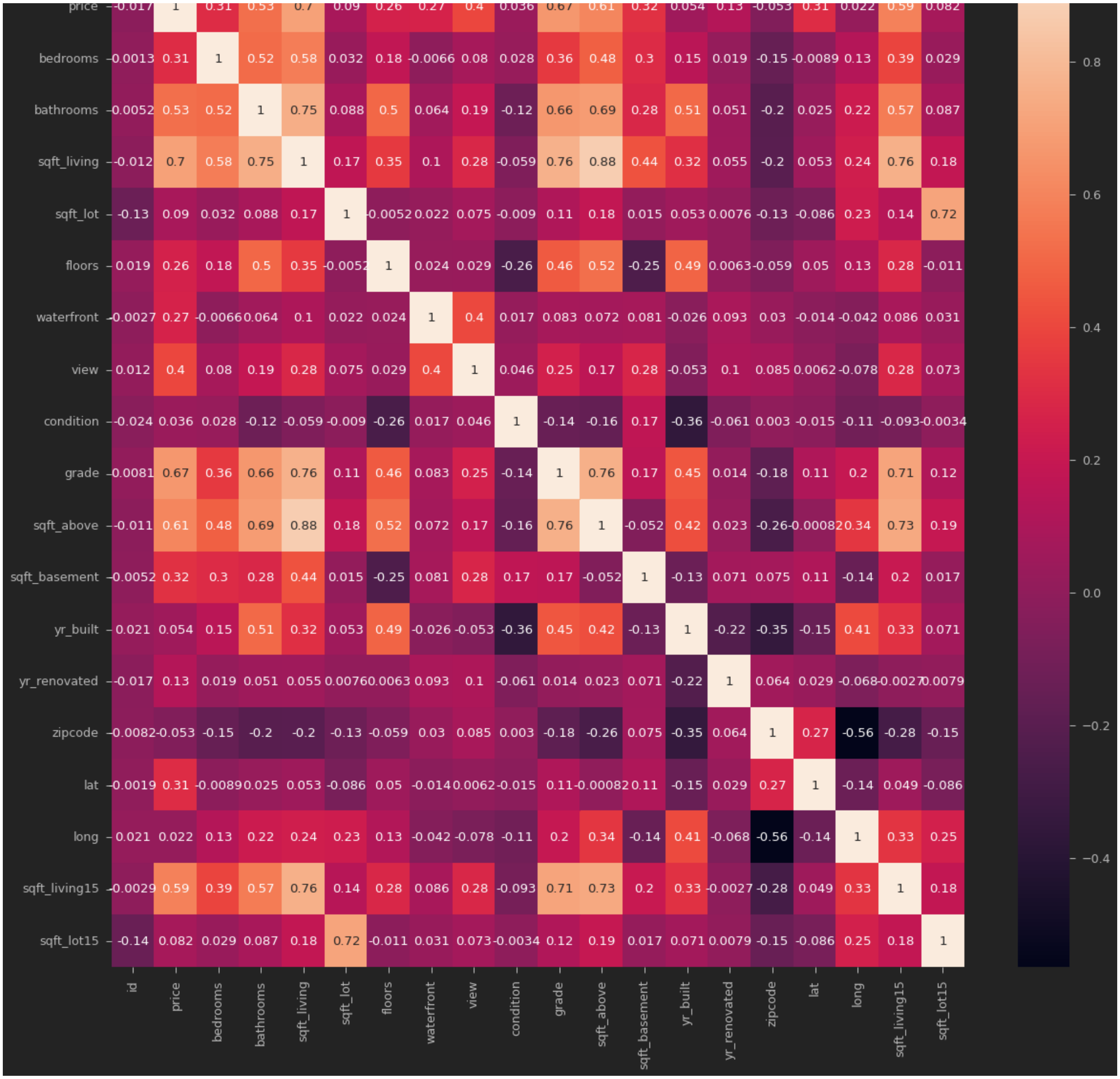





```
f, ax = plt.subplots(figsize = (20, 20))
sns.heatmap(house_df.corr(), annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9a995cbc40>
```




price
bedrooms
bathrooms
sqft_living
sqft_lot
floors
waterfront
view
condition
grade
sqft_above
sqft_basement
yr_built
yr_renovated
zipcode
lat
long
sqft_living15
sqft_lot15
id

```
house_df_sample = house_df[ ['price', 'bedrooms', 'bathrooms', 'sqft_living', 's
```

```
house_df_sample
```

	price	bedrooms	bathrooms	sqft_living	sqft_lot	sqft_above	sqft_bas
0	221900.0	3	1.00	1180	5650	1180	
1	538000.0	3	2.25	2570	7242	2170	
2	180000.0	2	1.00	770	10000	770	
3	604000.0	4	3.00	1960	5000	1050	
4	510000.0	3	2.00	1680	8080	1680	
...	...	...	...	...	...	...	
21608	360000.0	3	2.50	1530	1131	1530	
21609	400000.0	4	2.50	2310	5813	2310	
21610	402101.0	2	0.75	1020	1350	1020	
21611	400000.0	3	2.50	1600	2388	1600	
21612	325000.0	2	0.75	1020	1076	1020	

21613 rows × 8 columns

PERFORM DATA CLEANING AND FEATURE ENGINEERING

```
selected_features = ['bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot', 'floors
```

```
X = house_df[selected_features]
```

```
X
```

	bedrooms	bathrooms	sqft_living	sqft_lot	floors	sqft_above	sqft_base
0	3	1.00	1180	5650	1.0	1180	
1	3	2.25	2570	7242	2.0	2170	
2	2	1.00	770	10000	1.0	770	
3	4	3.00	1960	5000	1.0	1050	
4	3	2.00	1680	8080	1.0	1680	
...	...	...	...	...	...	...	
21608	3	2.50	1530	1131	3.0	1530	
21609	4	2.50	2310	5813	2.0	2310	
21610	2	0.75	1020	1350	2.0	1020	
21611	3	2.50	1600	2388	2.0	1600	
21612	2	0.75	1020	1076	2.0	1020	

21613 rows × 7 columns

```
y = house_df['price']
```

```
y
```

```
0        221900.0
1        538000.0
2        180000.0
3        604000.0
4        510000.0
           ...
21608    360000.0
21609    400000.0
21610    402101.0
21611    400000.0
21612    325000.0
Name: price, Length: 21613, dtype: float64
```

```
X.shape
```

```
(21613, 7)
```

```
y.shape
```

```
(21613,)
```

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

```
X_scaled
```

```
array([[0.09090909, 0.125     , 0.06716981, ..., 0.        , 0.09758772,
        0.        ],
       [0.09090909, 0.28125   , 0.17207547, ..., 0.4       , 0.20614035,
        0.08298755],
       [0.06060606, 0.125     , 0.03622642, ..., 0.        , 0.05263158,
        0.        ],
       ...,
       [0.06060606, 0.09375   , 0.05509434, ..., 0.4       , 0.08004386,
        0.        ],
       [0.09090909, 0.3125    , 0.09886792, ..., 0.4       , 0.14364035,
        0.        ],
       [0.06060606, 0.09375   , 0.05509434, ..., 0.4       , 0.08004386,
        0.        ]])
```

```
X_scaled.shape
```

```
(21613, 7)
```

```
scaler.data_max_
```

```
array([3.300000e+01, 8.000000e+00, 1.354000e+04, 1.651359e+06,
       3.500000e+00, 9.410000e+03, 4.820000e+03])
```

```
scaler.data_min_
```

```
array([  0.,   0., 290., 520.,   1., 290.,   0.])
```

```
y = y.values.reshape(-1,1)
```

```
y_scaled = scaler.fit_transform(y)
```

```
y_scaled
```

```
array([[0.01926557],
       [0.06072131],
       [0.01377049],
       ...,
       [0.04289849],
       [0.04262295],
       [0.03278689]])
```

TRAIN A DEEP LEARNING MODEL WITH LIMITED NUMBER OF FEATURES

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_siz
```

```
X_train.shape
```

```
(16209, 7)
```

```
X_test.shape
```

```
(5404, 7)
```

```
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(100, input_dim = 7, activation = 'relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(1, activation = 'linear'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 100)               800

 dense_1 (Dense)             (None, 100)               10100

 dense_2 (Dense)             (None, 100)               10100

 dense_3 (Dense)             (None, 1)                 101

=================================================================
Total params: 21,101
Trainable params: 21,101
Non-trainable params: 0
_________________________________________________________________
```

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

```
epochs_hist = model.fit(X_train, y_train, epochs = 100, batch_size = 50, validat
```

```
Epoch 1/100
260/260 [==============================] - 1s 3ms/step - loss: 0.0011 - val_l
Epoch 2/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0011 - val_l
Epoch 3/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0011 - val_l
Epoch 4/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 5/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 6/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 7/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 8/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 9/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 10/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 11/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
Epoch 12/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_l
```

```
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_lo
Epoch 13/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_lo
Epoch 14/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_lo
Epoch 15/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_lo
Epoch 16/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0010 - val_lo
Epoch 17/100
260/260 [==============================] - 1s 2ms/step - loss: 9.9406e-04 - va
Epoch 18/100
260/260 [==============================] - 1s 2ms/step - loss: 9.9036e-04 - va
Epoch 19/100
260/260 [==============================] - 1s 2ms/step - loss: 9.9066e-04 - va
Epoch 20/100
260/260 [==============================] - 1s 2ms/step - loss: 9.9623e-04 - va
Epoch 21/100
260/260 [==============================] - 1s 2ms/step - loss: 9.8653e-04 - va
Epoch 22/100
260/260 [==============================] - 1s 2ms/step - loss: 9.8726e-04 - va
Epoch 23/100
260/260 [==============================] - 1s 2ms/step - loss: 9.8762e-04 - va
Epoch 24/100
260/260 [==============================] - 1s 2ms/step - loss: 9.7829e-04 - va
Epoch 25/100
260/260 [==============================] - 1s 2ms/step - loss: 9.7300e-04 - va
Epoch 26/100
260/260 [==============================] - 1s 2ms/step - loss: 9.8843e-04 - va
Epoch 27/100
260/260 [==============================] - 0s 2ms/step - loss: 9.7504e-04 - va
Epoch 28/100
260/260 [==============================] - 1s 2ms/step - loss: 9.8565e-04 - va
Epoch 29/100
260/260 [==============================] - 1s 2ms/step - loss: 9.7771e-04 - va
Epoch 30/100
260/260 [==============================] - 1s 2ms/step - loss: 9.7868e-04 - va
```

EVALUATE TRAINED DEEP LEARNING MODEL PERFORMANCE

```
epochs_hist.history.keys()
```

```
dict_keys(['loss', 'val_loss'])
```

```
plt.plot(epochs_hist.history['loss'])
plt.plot(epochs_hist.history['val_loss'])
plt.title('Model Loss Progress During Training')
plt.xlabel('Epoch')
plt.ylabel('Training and Validation Loss')
plt.legend(['Training Loss', 'Validation Loss'])
```

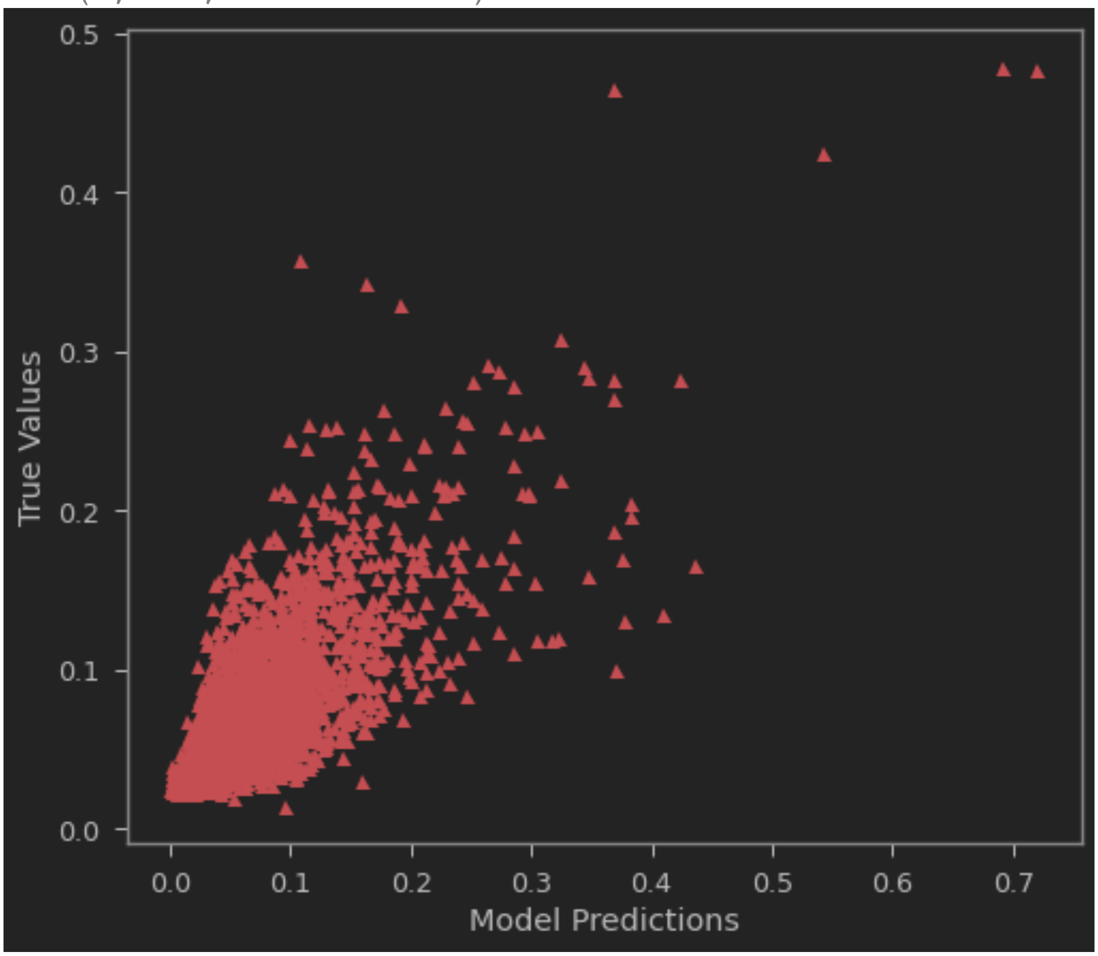
```
<matplotlib.legend.Legend at 0x7f9a38c520a0>
```

```
# 'bedrooms','bathrooms','sqft_living','sqft_lot','floors', 'sqft_above', 'sqft_
X_test_1 = np.array([[ 4, 3, 1960, 5000, 1, 2000, 3000 ]])

scaler_1 = MinMaxScaler()
X_test_scaled_1 = scaler_1.fit_transform(X_test_1)

y_predict_1 = model.predict(X_test_scaled_1)

y_predict_1 = scaler.inverse_transform(y_predict_1)
y_predict_1
```

```
1/1 [==============================] - 0s 75ms/step
array([[288391.1]], dtype=float32)
```

```
y_predict = model.predict(X_test)
plt.plot(y_test, y_predict, "^", color = 'r')
plt.xlabel('Model Predictions')
plt.ylabel('True Values')
```

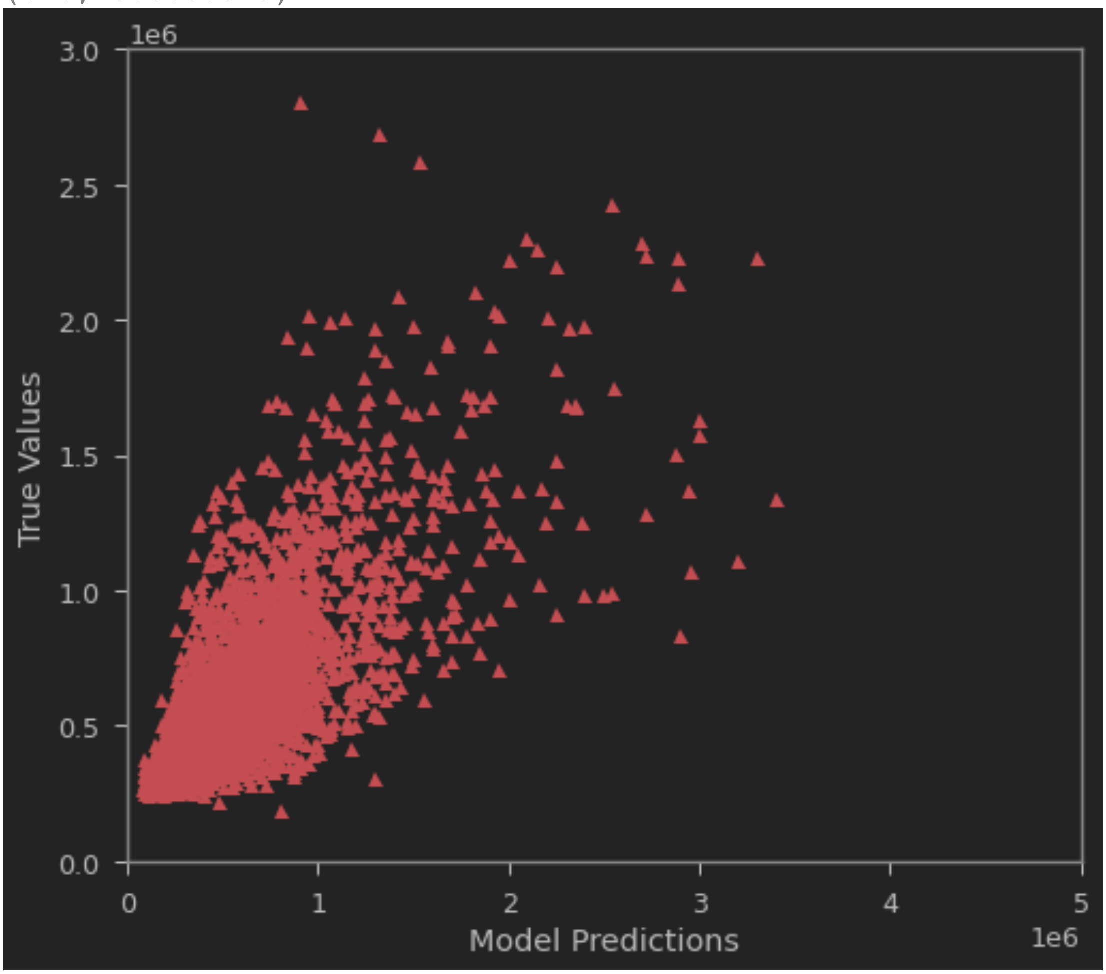
```
169/169 [==============================] - 0s 1ms/step
Text(0, 0.5, 'True Values')
```




```
y_predict_orig = scaler.inverse_transform(y_predict)
y_test_orig = scaler.inverse_transform(y_test)
```

```
plt.plot(y_test_orig, y_predict_orig, "^", color = 'r')
plt.xlabel('Model Predictions')
plt.ylabel('True Values')
plt.xlim(0, 5000000)
plt.ylim(0, 3000000)
```

```
(0.0, 3000000.0)
```




```
k = X_test.shape[1]
n = len(X_test)
n
```

```
5404
```

```
k
```

```
7
```

```

from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from math import sqrt

RMSE = float(format(np.sqrt(mean_squared_error(y_test_orig, y_predict_orig)),'.3
MSE = mean_squared_error(y_test_orig, y_predict_orig)
MAE = mean_absolute_error(y_test_orig, y_predict_orig)
r2 = r2_score(y_test_orig, y_predict_orig)
adj_r2 = 1-(1-r2)*(n-1)/(n-k-1)

print('RMSE =',RMSE, '\nMSE =',MSE, '\nMAE =',MAE, '\nR2 =', r2, '\nAdjusted R2

```

```
RMSE = 227049.828
MSE = 51551624455.33176
MAE = 152877.96062812267
R2 = 0.5760681183789966
Adjusted R2 = 0.5755181696815639
```

TRAIN AND EVALUATE A DEEP LEARNING MODEL WITH INCREASED NUMBER OF FEATURES (INDEPENDANT VARIABLES)

```
selected_features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors', '
'yr_renovated', 'zipcode', 'lat', 'long', 'sqft_living15', 'sqft_lot15']

X = house_df[selected_features]
```

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

```
y = house_df['price']
```

```
y = y.values.reshape(-1,1)
y_scaled = scaler.fit_transform(y)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_siz
```

```
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(10, input_dim = 19, activation = 'relu'))
model.add(Dense(10, activation = 'relu'))
model.add(Dense(1, activation = 'linear'))
```

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

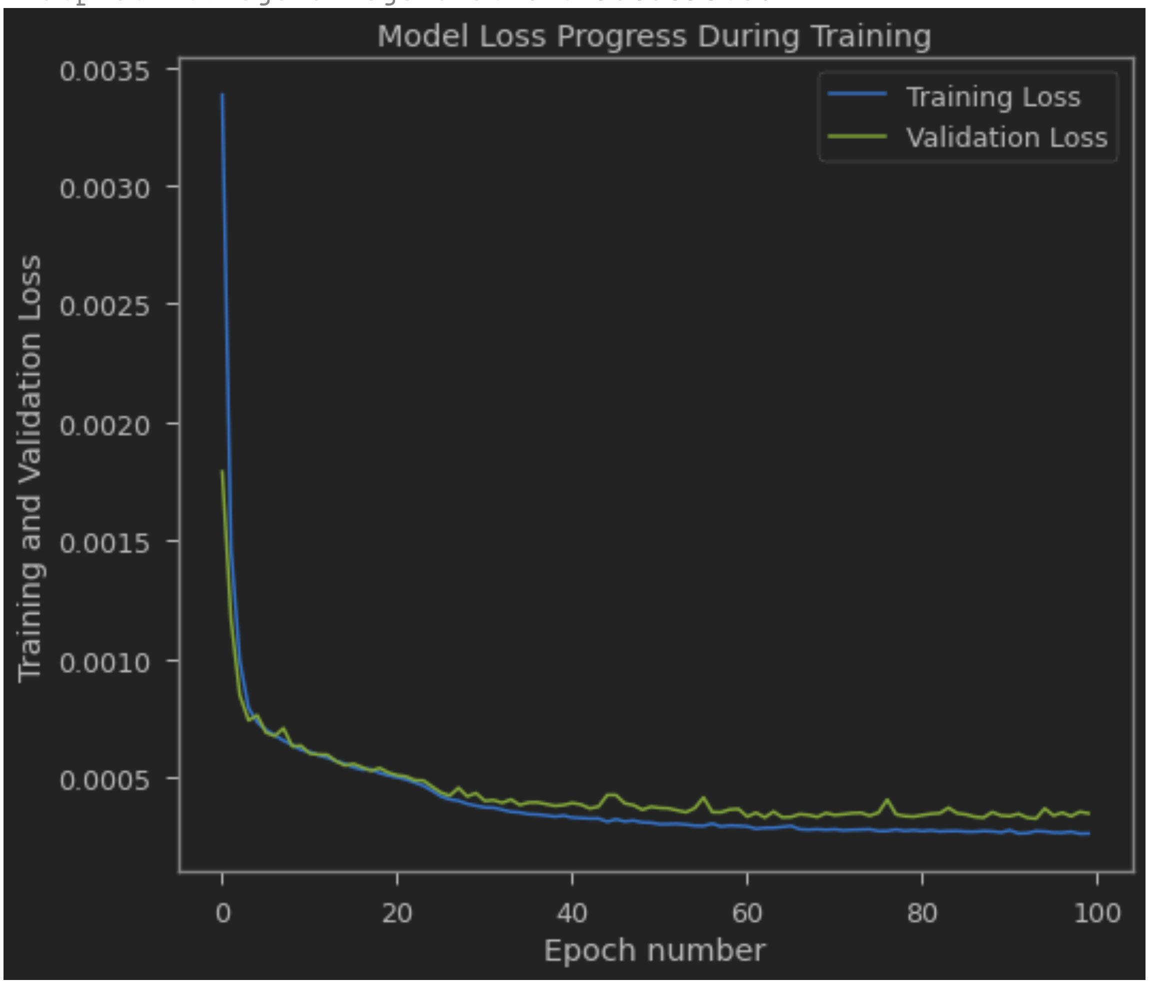
```
epochs_hist = model.fit(X_train, y_train, epochs = 100, batch_size = 50, verbose
```

```
Epoch 1/100
260/260 [==============================] - 1s 2ms/step - loss: 0.0034 - val_l
Epoch 2/100
260/260 [==============================] - 0s 2ms/step - loss: 0.0015 - val_l
Epoch 3/100
260/260 [==============================] - 0s 2ms/step - loss: 9.9270e-04 - va
Epoch 4/100
260/260 [==============================] - 1s 2ms/step - loss: 7.9283e-04 - va
Epoch 5/100
260/260 [==============================] - 0s 2ms/step - loss: 7.3193e-04 - va
Epoch 6/100
260/260 [==============================] - 0s 2ms/step - loss: 6.9962e-04 - va
Epoch 7/100
260/260 [==============================] - 0s 2ms/step - loss: 6.7743e-04 - va
Epoch 8/100
260/260 [==============================] - 0s 2ms/step - loss: 6.5490e-04 - va
Epoch 9/100
260/260 [==============================] - 0s 2ms/step - loss: 6.3647e-04 - va
Epoch 10/100
260/260 [==============================] - 0s 2ms/step - loss: 6.1570e-04 - va
Epoch 11/100
260/260 [==============================] - 0s 2ms/step - loss: 6.0698e-04 - va
Epoch 12/100
260/260 [==============================] - 0s 2ms/step - loss: 5.9310e-04 - va
Epoch 13/100
260/260 [==============================] - 0s 2ms/step - loss: 5.8291e-04 - va
Epoch 14/100
260/260 [==============================] - 0s 2ms/step - loss: 5.6829e-04 - va
Epoch 15/100
260/260 [==============================] - 0s 2ms/step - loss: 5.5721e-04 - va
Epoch 16/100
260/260 [==============================] - 0s 2ms/step - loss: 5.4173e-04 - va
Epoch 17/100
260/260 [==============================] - 1s 2ms/step - loss: 5.3188e-04 - va
```

```
Epoch 18/100
260/260 [==============================] - 0s 2ms/step - loss: 5.3390e-04 - va
Epoch 19/100
260/260 [==============================] - 0s 2ms/step - loss: 5.1736e-04 - va
Epoch 20/100
260/260 [==============================] - 0s 2ms/step - loss: 5.0727e-04 - va
Epoch 21/100
260/260 [==============================] - 1s 2ms/step - loss: 5.0021e-04 - va
Epoch 22/100
260/260 [==============================] - 0s 2ms/step - loss: 4.9025e-04 - va
Epoch 23/100
260/260 [==============================] - 0s 2ms/step - loss: 4.7798e-04 - va
Epoch 24/100
260/260 [==============================] - 0s 2ms/step - loss: 4.6377e-04 - va
Epoch 25/100
260/260 [==============================] - 0s 2ms/step - loss: 4.4256e-04 - va
Epoch 26/100
260/260 [==============================] - 0s 2ms/step - loss: 4.1919e-04 - va
Epoch 27/100
260/260 [==============================] - 0s 2ms/step - loss: 4.0625e-04 - va
Epoch 28/100
260/260 [==============================] - 0s 2ms/step - loss: 4.0042e-04 - va
Epoch 29/100
260/260 [==============================] - 0s 1ms/step - loss: 3.8744e-04 - va
Epoch 30/100
260/260 [==============================] - 0s 2ms/step - loss: 3.7969e-04 - va
```

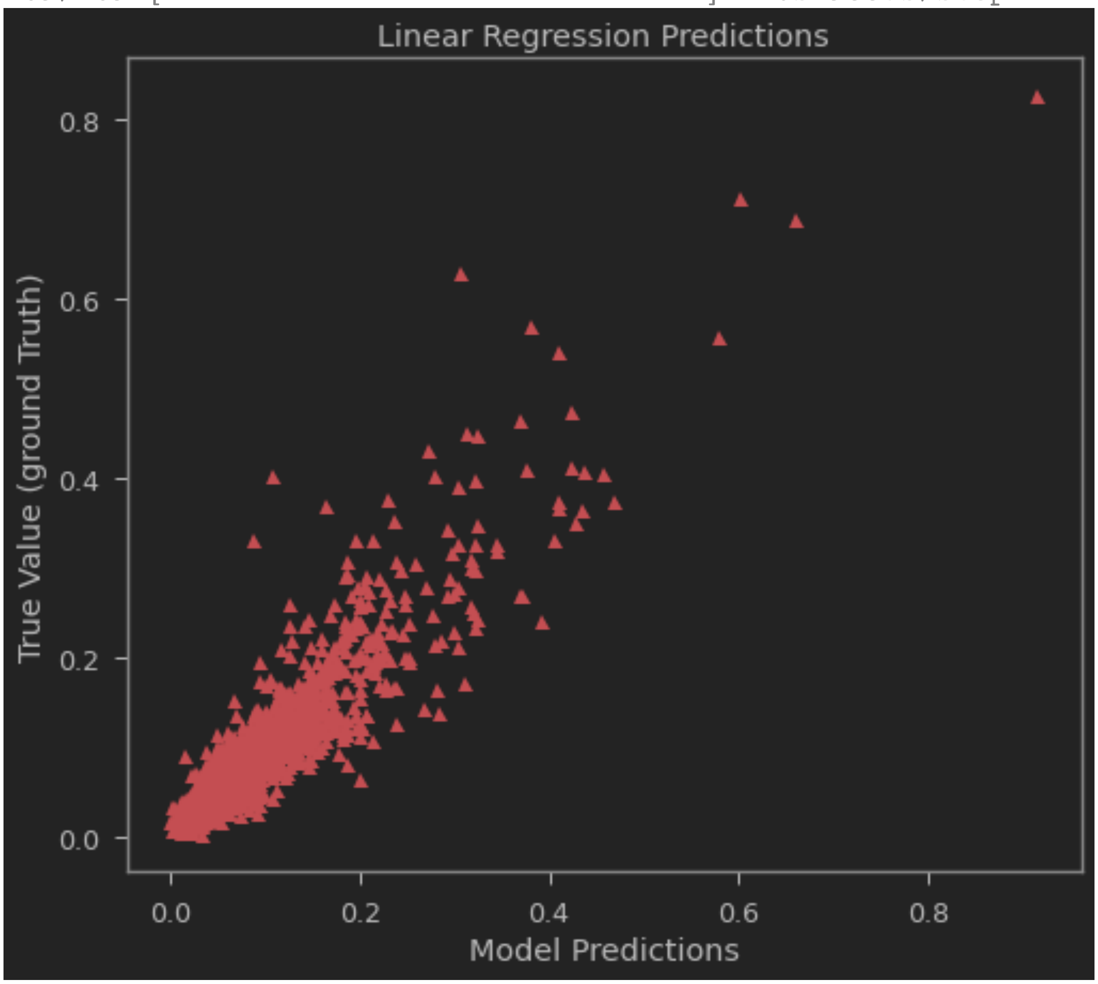
```
plt.plot(epochs_hist.history['loss'])
plt.plot(epochs_hist.history['val_loss'])
plt.title('Model Loss Progress During Training')
plt.ylabel('Training and Validation Loss')
plt.xlabel('Epoch number')
plt.legend(['Training Loss', 'Validation Loss'])
```

```
<matplotlib.legend.Legend at 0x7f9a3a595be0>
```



```
y_predict = model.predict(X_test)
plt.plot(y_test, y_predict, "^", color = 'r')
plt.xlabel("Model Predictions")
plt.ylabel("True Value (ground Truth)")
plt.title('Linear Regression Predictions')
plt.show()
```

```
169/169 [==============================] - 0s 933us/step
```




```
y_predict_orig = scaler.inverse_transform(y_predict)
y_test_orig = scaler.inverse_transform(y_test)

```

```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from math import sqrt

RMSE = float(format(np.sqrt(mean_squared_error(y_test_orig, y_predict_orig)),'.3
MSE = mean_squared_error(y_test_orig, y_predict_orig)
MAE = mean_absolute_error(y_test_orig, y_predict_orig)
r2 = r2_score(y_test_orig, y_predict_orig)
adj_r2 = 1-(1-r2)*(n-1)/(n-k-1)

print('RMSE =',RMSE, '\nMSE =',MSE, '\nMAE =',MAE, '\nR2 =', r2, '\nAdjusted R2

```

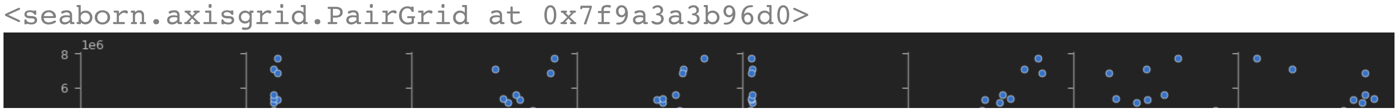
```
RMSE = 148187.376
MSE = 21959498322.88438
MAE = 84156.12517059123
R2 = 0.849369982608905
Adjusted R2 = 0.8491745767301544
```

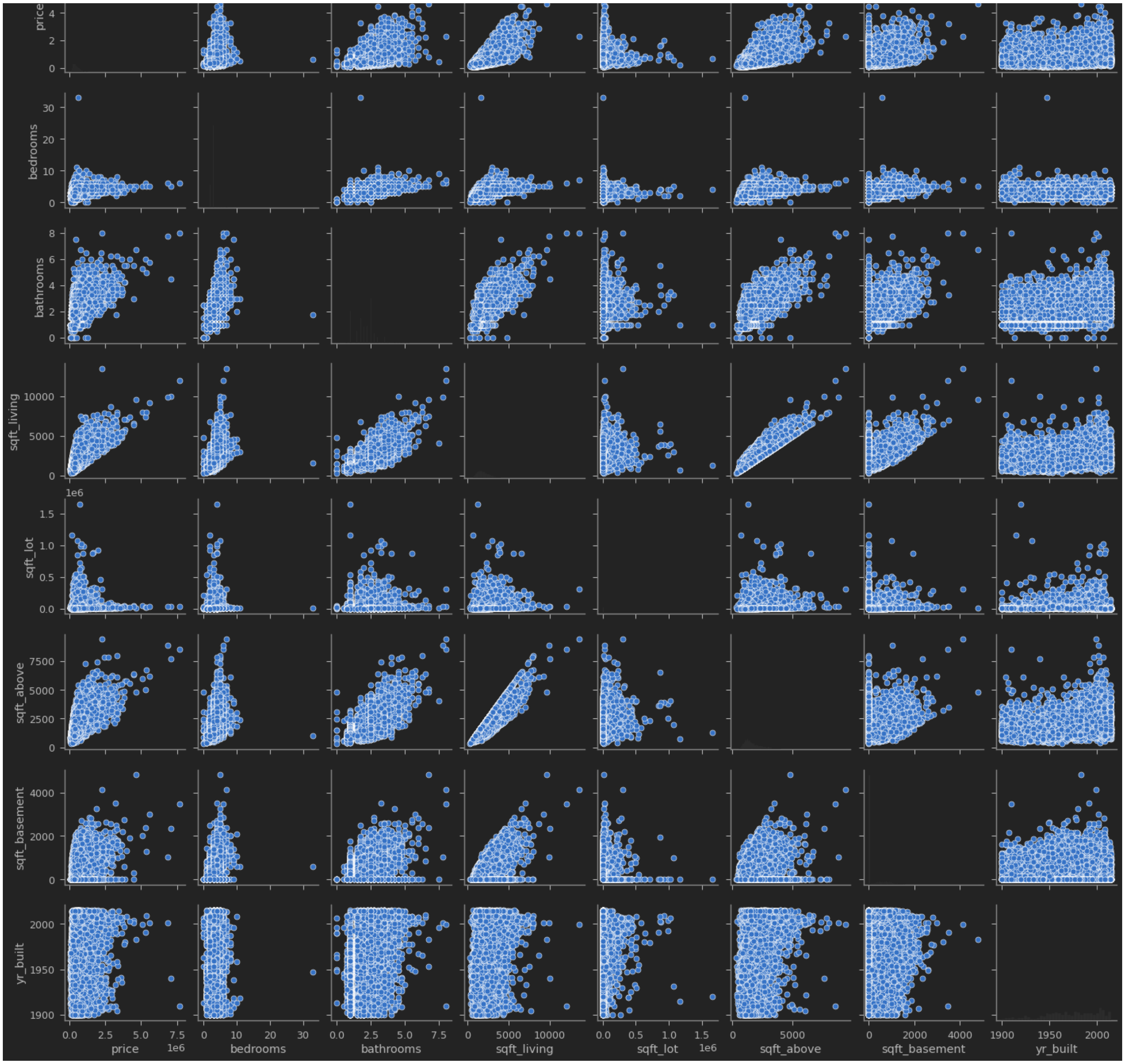
```
house_df.describe()
```

	id	price	bedrooms	bathrooms	sqft_living	sqft_l
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+

```
sns.pairplot(house_df_sample)
```

<seaborn.axisgrid.PairGrid at 0x7f9a3a3b96d0>




price
bedrooms
bathrooms
sqft_living
sqft_lot
sqft_above
sqft_basement
yr_built
1e6
1e6
1e6

```
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(100, input_dim = 7, activation = 'relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(100, activation='relu'))
model.add(Dense(200, activation='relu'))
model.add(Dense(1, activation = 'linear'))

model.summary()

```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_7 (Dense)             (None, 100)               800

 dense_8 (Dense)             (None, 100)               10100

 dense_9 (Dense)             (None, 100)               10100

 dense_10 (Dense)            (None, 200)               20200

 dense_11 (Dense)            (None, 1)                 201

=================================================================
Total params: 41,401
Trainable params: 41,401
Non-trainable params: 0
_________________________________________________________________
```

```
import tensorflow.keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model = Sequential()
model.add(Dense(10, input_dim = 19, activation = 'relu'))
model.add(Dense(10, activation = 'relu'))
model.add(Dense(200, activation = 'relu'))
model.add(Dense(200, activation = 'relu'))

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

```

```