Evaluating Property Prices in South Africa using Machine Learning

Mr Ashlin Darius Govindasamy
University of South Africa

September 5, 2022

Contents

Chapter 1

Methodology to evaluating value of properties

1.0.1 Introduction

This paper is a real life study and application of how I evaluated/modeled/predicted a home's sale price to be sold in Johannesburg. Methodologies and techniques will be discussed so you can also evaluate your home this way using techniques of Machine Learning. How do we do this?

Consider the following:

We have a dataset of homes that have been sold in Johannesburg or some suburb in that dataset we have the following features:

- Number of bedrooms
- Number of bathrooms
- Number of garages
- Size of Land (ERF)
- Floor Size (Size of House)

We also have the following features that are not numerical but categorical:

- Type of House (Townhouse, House, Flat)
- Type of Garage (Carport, Garage, No Garage)
- Type of Kitchen (Open Plan, Separate, No Kitchen)
- Type of Bathroom (En-Suite, Separate, No Bathroom)
- Type of Bedroom (En-Suite, Separate, No Bedroom)
- Garden
- Patio
- Pool

that list could go on and on but we will keep it simple for now.

We can also have data about the suburb where the house is located, Security Features, the year it was built, the year it was sold, the price it was sold for and even a photo of the house could be used to predict the price.

In this paper we use a dataset obtained from a Real Estate Company in Johannesburg. It consists of recent records of homes that have been sold in the suburb Bezuidenhout Valley.

The dataset came from a .pdf format which i used some python script to scrape records from that .pdf real estate sales report. For you to obtain that real estate sales report you can purchase one from Property24 or write some API to scrape it.
I converted the data into a pandas dataframe and saved it as a .csv file.

The dataset $\mathcal{D}$ consists of n records of homes that have been sold in Bezuidenhout Valley.
Each record $\mathcal{D}_i$ consists of m features.
Which features we use to predict the price of the house is up to us.

For this paper we will use the following features:
$\mathcal{D}_i = \{'HomeMeters','ErfSize','NumberBedrooms','NumberofGarage','NumberofBathrooms'\}$

1.0.2 Cons with this approach and notes on model evaluation

Using the features which i used in this paper is not the best approach to predict the price of a home.
As i am left with the unknown features that i did not use to predict the price of the home.

Example:
I dont know how the home looks, what maintainance is required? Is the home in a good area? Is the home in a bad area? How is the security there?

For this paper i never train a model to look at the photos of the home and predict the price of the home.
This model is biased towards the features that i used to train it.

My intentions when buying a property i look for ERF Size and Floor Size.
I love space,land and big houses so i used those features to train my model because over t i intent to improve/construct the home better and it will increase in value over t
I am sure many other people look for different features when buying a home.
This is why i say this model is biased towards the features that i used to train it and i am sure some of your guys might also use my methodology and thinking. When reading this paper.

But accounting for some missing features is better than not accounting for any features at all.
By using home data of the same neighbourhood we can predict the price of a home in that neighbourhood using the m features set. That will account for the missing features.

Note:
The model resonably predicts the price of a home in the same neighbourhood.
But judging by the features i used to predict the price of the home, the model is not accurate enough to predict the price of a home in a different neighbourhood.
This is because the model is not trained on data of different neighbourhoods. The model is trained on data of the same neighbourhood.
The model i created might cross the boundary of the sales price or under estimate the sales price.
You should use this model as a guide to predict the price of a home in the same neighbourhood.
Think of your home that you are going to sell as a home you want to buy and how much am i willing to pay

for it. You know the condition of your home, you know the area, you know the security features, you know the maintainance required.
Use your discretion if you want to overfit or understate the model. It also depends on your mood on how fast you want to move out or get rid of your property.

1.0.3 Methodology

Step 1: Obtain the Dataset
First you need to obtain the data of your sales of the neighbourhood over t.
You can obtain the data from a real estate company or scrape it from the internet if you got a site containing the information.

Step 2: Clean the Dataset
The dataset might contain missing values, outliers, incorrect data types, incorrect values, incorrect feature names.
You need to clean the dataset to make it usable for your model.
Pick the features you want to use to predict the price of the home.

Step 3: Split the Dataset
Split the dataset into a training set and a testing set.
The training set is used to train the model.
The testing set is used to test the model.

Step 4: Train the Model
Train the model on the training set.
The model learns from the training set.
The model learns the relationship between the features and the price of the home.

Step 5: Test the Model
Test the model on the testing set.
The model predicts the price of the home using the features.
The model compares the predicted price of the home with the actual price of the home.
The model calculates the error between the predicted price and the actual price.
The model calculates the accuracy of the model.

Step 6: Evaluate the Model
Evaluate the model.
The model is evaluated by the accuracy of the model.
The model is evaluated by the error of the model.

In this paper my models is as follows:
I built two models.

Model 1:
I used the features:
$\mathcal{D}_i = \{'HomeMeters',' ErfSize',' NumberBedrooms',' NumberofGarage',' NumberofBathrooms'\}$

I took the $\mathcal{D}$ and trained the dataset using the Sklearn Linear Regression model.
$f_{reg}(\mathcal{D}_i) = \beta_0 + \beta_1 \cdot \mathcal{D}_i['HomeMeters'] + \beta_2 \cdot \mathcal{D}_i['ErfSize'] + \beta_3 \cdot \mathcal{D}_i['NumberBedrooms'] + \beta_4 \cdot \mathcal{D}_i['NumberofGarage'] + \beta_5 \cdot \mathcal{D}_i['NumberofBathrooms']$

Model 2:
I also used the same features:
$\mathcal{D}_i = \{'HomeMeters','ErfSize','NumberBedrooms','NumberofGarage','NumberofBathrooms'\}$
but i used Tensorflow with Keras to build a neural network model.

For this model i used the following architecture:

- Input Layer: n of training data
- Hidden Layer 1: 64 neurons
- Hidden Layer 2: 64 neurons
- Output Layer: 1 neuron

I converted the pandas csv data into a numpy array / tensor.

A tensor is a generalization of vectors and matrices to potentially higher dimensions.
I used the Adam optimizer to train the model.

Okay we can now proceed to viewing my results in the next following pages. Hope your enjoy!

Chapter 2

Real Life Example and Explaination

2.1 Creating Machine Learning models to evaluate price of properties in Bezuidenhout Valley

2.1.1 Overview and Objectives

Overview

This notebook is used to find an accurate machine learning model to predict/evaluate Sale Price of properties using given specifications of homes.

Objectives

- Clean our dataset to isolate variables suitable for running a model on.
- Check which variables influences the house price.
- Build a `Linear Regression Model` using `Sklearn` and a `Neural Network Model` using `Tensorflow with Keras`
- Visualise our data and project insights on our dataset

Data Engineering Dataset

Importing our dataset for property data in Bezuidenhout Valley

[66]:
```
import pandas as pd
df = pd.read_csv('data.csv')

# drop cash column
df = df.drop(columns=['Cash'])

#viewing our dataset
df
```

[66]:
```
                   Street Address              Township  \
0    214 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
1     77 9TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
2    212 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
3    225 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
4    193 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
5    276 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
6    122 9TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
```

7	17 ORLANDO STREET KENSINGTON	KENSINGTON
8	224 8TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
9	66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
10	40 10TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
11	64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
12	2 7TH STREET BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
13	177 7TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
14	35 8TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
15	258 7TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
16	83 10TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
17	68 9TH IVANUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
18	16 11TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY
19	221 8TH AVENUE BEZUIDENHOUT VALLEY	BEZUIDENHOUT VALLEY

	Erf I Portion	Sales Date	Reg Date	Sales Price	Size	R/m^2	\
0	594 0	20211018	20220128.000	R 1200000	495	R 2 424	
1	987 0	20220110	20220215.000	R 950 000	495	R 919	
2	592 0	20210716	20220309.000	R 1 200 000	495	R 2 424	
3	605 0	20210803	20211025.000	R 1 075 000	495	R2 172	
4	573 0	20211214	NaN	R 1 500 000	495	R3030	
5	942 0	20220520	NaN	R 1 350 000	495	R 2727	
6	1123 0	20211218	20220316.000	R 1 225 000	495	R 2475	
7	2515 0	20220121	20220328.000	R 1 280 000	495	R 2 586	
8	890 0	20201224	20210407.000	R 1 250 000	495	R 2525	
9	977 0	20211021	20211210.000	R 1 420 000	495	R 2 869	
10	1148 0	20210803	20211112.000	R 1 100 000	495	R 2 222	
11	976 0	20210714	20211102.000	R1 250 000	495	R 2525	
12	1131 0	20210806	20211115.000	R 1 300 000	495	R 2626	
13	285 0	20220422	NaN	R 900 000	495	R 1 818	
14	587 0	20210226	20210624.000	R 1 400 000	495	R 2828	
15	638	20210618	20210906.000	R1 250 000	495	R 2525	
16	1104 0	20210919	20220309.000	R 950000	495	R 919	
17	1069 0	20210218	20210419.000	R 1000000	495	R 020	
18	1221 0	20210610	20211007.000	R 1325 000	543	R 440	
19	601 10	20200701	20201013.000	R 1 270 000	495	R 2566	

	Distance	Bedroom	Bath	Garage	HomeM
0	94	7.000	7.000	2	300
1	123	NaN	NaN	NaN	-
2	103	3.000	2.000	1	-
3	69	3.000	2.000	-	-
4	248	3.000	2.500	NaN	200
5	478	1.000	3.000	1	221
6	407	3.000	3.000	1	-
7	442	3.000	3.000	1	237
8	32	4.000	2.000	1	-
9	290	3.000	0.000	NaN	264
10	231	3.000	2.000	1	180
11	304	6.000	6.000	NaN	218
12	322	4.000	1.000	1	269
13	391	3.000	2.000	1	-
14	124	3.000	2.000	2	-
15	328	2.000	1.000	NaN	97

```
16        297    3.000 3.000       1   105
17        158    4.000 4.000       2     -
18        303    3.000 3.000       1   356
19         53    3.000 3.000       -     -
```

Cleaning up dataset only using data where we got HomeM values

[67]:
```python
# drop df where HomeM is -
df = df[df['HomeM'] != '-']
df
# drop index
df = df.reset_index(drop=True)
df
```

[67]:
```
                                 Street Address             Township  \
0             214 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
1             193 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
2             276 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
3                   17 ORLANDO STREET KENSINGTON           KENSINGTON
4   66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
5              40 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
6   64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
7                2 7TH STREET BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
8             258 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
9              83 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
10             16 11TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY

   Erf I Portion  Sales Date     Reg Date  Sales Price  Size     R/m^2  \
0         594 0    20211018 20220128.000    R 1200000   495   R 2 424
1         573 0    20211214          NaN  R 1 500 000   495     R3030
2         942 0    20220520          NaN  R 1 350 000   495    R 2727
3        2515 0    20220121 20220328.000  R 1 280 000   495   R 2 586
4         977 0    20211021 20211210.000  R 1 420 000   495   R 2 869
5        1148 0    20210803 20211112.000  R 1 100 000   495   R 2 222
6         976 0    20210714 20211102.000   R1 250 000   495    R 2525
7        1131 0    20210806 20211115.000  R 1 300 000   495    R 2626
8          638     20210618 20210906.000   R1 250 000   495    R 2525
9        1104 0    20210919 20220309.000     R 950000   495     R 919
10       1221 0    20210610 20211007.000   R 1325 000   543     R 440

    Distance  Bedroom  Bath Garage HomeM
0         94    7.000 7.000      2   300
1        248    3.000 2.500    NaN   200
2        478    1.000 3.000      1   221
3        442    3.000 3.000      1   237
4        290    3.000 0.000    NaN   264
5        231    3.000 2.000      1   180
6        304    6.000 6.000    NaN   218
7        322    4.000 1.000      1   269
8        328    2.000 1.000    NaN    97
9        297    3.000 3.000      1   105
10       303    3.000 3.000      1   356
```

Calculating R/HomeM Column

[68]:
```
# calc Sales Price / HomeM
# convert Sales Price to float
df['Sales Price'] = df['Sales Price'].str.replace('R','')
# remove spaces from Sales Price
df['Sales Price'] = df['Sales Price'].str.replace(' ','')
df['R/HomeM'] = df['Sales Price'].astype(float) / df['HomeM'].astype(float)
df
```

[68]:
```
                               Street Address            Township  \
0            214 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
1            193 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
2            276 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
3                  17 ORLANDO STREET KENSINGTON           KENSINGTON
4   66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
5             40 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
6   64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
7               2 7TH STREET BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
8            258 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
9             83 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
10            16 11TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY

   Erf I Portion  Sales Date     Reg Date Sales Price  Size    R/m^2  \
0          594 0    20211018 20220128.000     1200000   495   R 2 424
1          573 0    20211214          NaN     1500000   495     R3030
2          942 0    20220520          NaN     1350000   495    R 2727
3         2515 0    20220121 20220328.000     1280000   495   R 2 586
4          977 0    20211021 20211210.000     1420000   495   R 2 869
5         1148 0    20210803 20211112.000     1100000   495   R 2 222
6          976 0    20210714 20211102.000     1250000   495    R 2525
7         1131 0    20210806 20211115.000     1300000   495    R 2626
8            638    20210618 20210906.000     1250000   495    R 2525
9         1104 0    20210919 20220309.000      950000   495     R 919
10        1221 0    20210610 20211007.000     1325000   543     R 440

    Distance  Bedroom  Bath Garage HomeM   R/HomeM
0         94    7.000 7.000      2   300  4000.000
1        248    3.000 2.500    NaN   200  7500.000
2        478    1.000 3.000      1   221  6108.597
3        442    3.000 3.000      1   237  5400.844
4        290    3.000 0.000    NaN   264  5378.788
5        231    3.000 2.000      1   180  6111.111
6        304    6.000 6.000    NaN   218  5733.945
7        322    4.000 1.000      1   269  4832.714
8        328    2.000 1.000    NaN    97 12886.598
9        297    3.000 3.000      1   105  9047.619
10       303    3.000 3.000      1   356  3721.910
```

Average R/HomeM

[69]:
```
print('R' + str(df['R/HomeM'].mean()))
```

```
R6429.284178520146
```

Seperating Erf and Portion into seperate columns

[70]:
```
df['Erf'] = df['Erf I Portion'].str.split(' ').str[0]
df['Portion'] = df['Erf I Portion'].str.split(' ').str[1]
df
```

[70]:
```
                              Street Address              Township  \
0             214 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
1             193 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
2             276 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
3                   17 ORLANDO STREET KENSINGTON           KENSINGTON
4   66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
5             40 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
6   64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
7               2 7TH STREET BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
8             258 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
9             83 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
10            16 11TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY

   Erf I Portion  Sales Date     Reg Date Sales Price  Size    R/m^2  \
0          594 0    20211018 20220128.000     1200000   495  R 2 424
1          573 0    20211214          NaN     1500000   495    R3030
2          942 0    20220520          NaN     1350000   495   R 2727
3         2515 0    20220121 20220328.000     1280000   495  R 2 586
4          977 0    20211021 20211210.000     1420000   495  R 2 869
5         1148 0    20210803 20211112.000     1100000   495  R 2 222
6          976 0    20210714 20211102.000     1250000   495   R 2525
7         1131 0    20210806 20211115.000     1300000   495   R 2626
8            638    20210618 20210906.000     1250000   495   R 2525
9         1104 0    20210919 20220309.000      950000   495    R 919
10        1221 0    20210610 20211007.000     1325000   543    R 440

    Distance  Bedroom  Bath Garage HomeM   R/HomeM   Erf Portion
0         94    7.000 7.000      2   300  4000.000   594       0
1        248    3.000 2.500    NaN   200  7500.000   573       0
2        478    1.000 3.000      1   221  6108.597   942       0
3        442    3.000 3.000      1   237  5400.844  2515       0
4        290    3.000 0.000    NaN   264  5378.788   977       0
5        231    3.000 2.000      1   180  6111.111  1148       0
6        304    6.000 6.000    NaN   218  5733.945   976       0
7        322    4.000 1.000      1   269  4832.714  1131       0
8        328    2.000 1.000    NaN    97 12886.598   638     NaN
9        297    3.000 3.000      1   105  9047.619  1104       0
10       303    3.000 3.000      1   356  3721.910  1221       0
```

Replace all NaNs to 0

[71]:
```
df = df.fillna(0)
```

2.1.2 Building machine learning model to predict price of property using HomeM,Erf,Bedroom,Garage,Bathroom

Predicting price of property using Linear Regression from Sklearn

[72]:
```
# use polynomial linear regression to predict home sales price
from sklearn.linear_model import LinearRegression
X = df[['HomeM','Erf','Bedroom','Garage','Bath']]
y = df['Sales Price']
model = LinearRegression()
model.fit(X,y)
# predict home sales price
prediction = model.predict([[248,495,5,1,4]])[0]
prediction = round(prediction,2)

print('R',prediction)
print("Model Accuracy is :",model.score(X,y))
```

```
R 1251315.92
Model Accuracy is : 0.7149720907585728
```

Running our model on the actual sales price

[73]:
```
# iterate through all the data and predict home sales price
# create a new dataframe to store the predicted values

# suppressing warnings for pandas/sci-learn
import warnings
warnings.filterwarnings('ignore')

df_predicted = pd.DataFrame(columns=['Street␣
↪Address','HomeM','Erf','Bedroom','Garage','Bath','Actual Sale Price','Predicted Sale␣
↪Price'])
for i in range(len(df)):
    # convert all values to numeric
    df['HomeM'][i] = df['HomeM'][i].replace('R','')
    df['HomeM'][i] = df['HomeM'][i].replace(' ','')
    df['HomeM'][i] = float(df['HomeM'][i])
    df['Erf'][i] = df['Erf'][i].replace(' ','')
    df['Erf'][i] = float(df['Erf'][i])
    df['Bedroom'][i] = float(df['Bedroom'][i])
    df['Garage'][i] = float(df['Garage'][i])
    df['Bath'][i] = float(df['Bath'][i])
    df['Sales Price'][i] = df['Sales Price'][i].replace('R','')
    df['Sales Price'][i] = df['Sales Price'][i].replace(' ','')
    df['Sales Price'][i] = float(df['Sales Price'][i])
    # predict home sales price
    prediction = model.
↪predict([[df['HomeM'][i],df['Erf'][i],df['Bedroom'][i],df['Garage'][i],df['Bath'][i]]])[0]
    prediction = round(prediction,2)
    # append to dataframe
    df_predicted = df_predicted.append({'Street Address':df['Street␣
↪Address'][i],'HomeM':df['HomeM'][i],'Erf':df['Erf'][i],'Bedroom':
↪df['Bedroom'][i],'Garage':df['Garage'][i],'Bath':df['Bath'][i],'Actual Sale Price':
↪df['Sales Price'][i],'Predicted Sale Price':prediction},ignore_index=True)
```

```
df_predicted
```

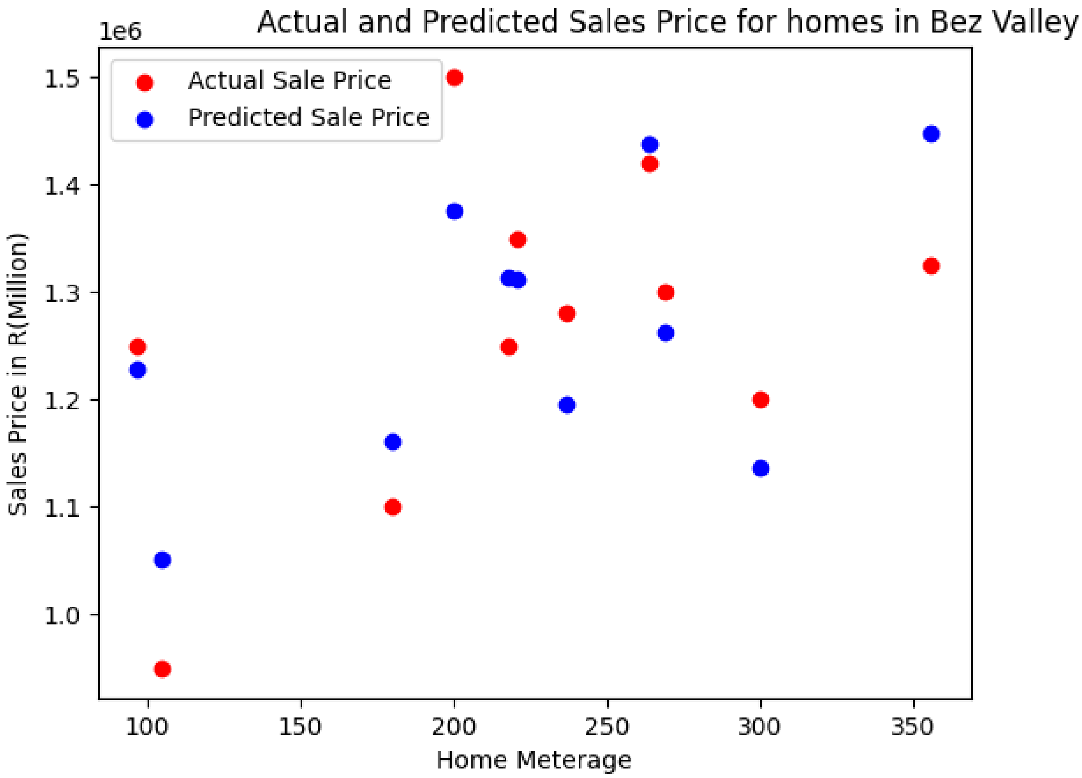
[73]:

```
                                      Street Address  HomeM      Erf  Bedroom  \
0              214 7TH AVENUE BEZUIDENHOUT VALLEY 300.000  594.000    7.000
1              193 8TH AVENUE BEZUIDENHOUT VALLEY 200.000  573.000    3.000
2              276 8TH AVENUE BEZUIDENHOUT VALLEY 221.000  942.000    1.000
3                    17 ORLANDO STREET KENSINGTON 237.000 2515.000    3.000
4    66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY 264.000  977.000    3.000
5              40 10TH AVENUE BEZUIDENHOUT VALLEY 180.000 1148.000    3.000
6    64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY 218.000  976.000    6.000
7                2 7TH STREET BEZUIDENHOUT VALLEY 269.000 1131.000    4.000
8              258 7TH AVENUE BEZUIDENHOUT VALLEY  97.000  638.000    2.000
9              83 10TH AVENUE BEZUIDENHOUT VALLEY 105.000 1104.000    3.000
10             16 11TH AVENUE BEZUIDENHOUT VALLEY 356.000 1221.000    3.000

    Garage  Bath  Actual Sale Price  Predicted Sale Price
0    2.000 7.000        1200000.000           1136949.610
1    0.000 2.500        1500000.000           1375792.990
2    1.000 3.000        1350000.000           1311757.930
3    1.000 3.000        1280000.000           1195158.110
4    0.000 0.000        1420000.000           1438484.610
5    1.000 2.000        1100000.000           1161445.670
6    0.000 6.000        1250000.000           1313778.490
7    1.000 1.000        1300000.000           1263275.450
8    0.000 1.000        1250000.000           1228893.520
9    1.000 3.000         950000.000           1051638.010
10   1.000 3.000        1325000.000           1447825.610
```

Plotting Graph of Actual and Predicted Sales Price for homes in Bez Valley

[74]:

```
import matplotlib.pyplot as plt
plt.scatter(df_predicted['HomeM'],df_predicted['Actual Sale Price'],color='red')
plt.scatter(df_predicted['HomeM'],df_predicted['Predicted Sale Price'],color='blue')
plt.xlabel('Home Meterage')
plt.ylabel('Sales Price in R(Million)')
plt.title('                              Actual and Predicted Sales Price for homes in␣
 ↪Bez Valley')
# add a legend
plt.legend(['Actual Sale Price','Predicted Sale Price'])
plt.show()
```



For fun how much our Somerset West Home Specs would sell for in Bez Valley

[75]:
```
#HomeM,Erf,Bedroom,Garage,Bathroom"
prediction = model.predict([[1028,414,5,3,4]])[0]
prediction = round(prediction,2)
print('R',prediction)
```

R 2202585.72

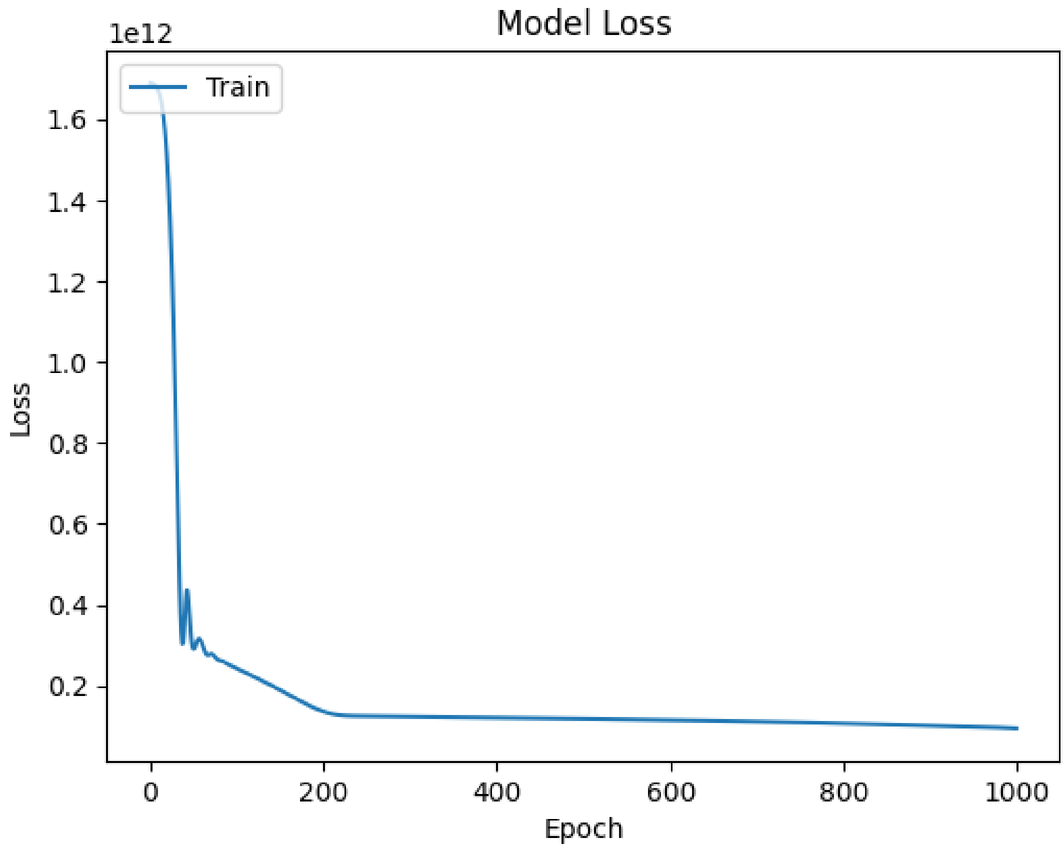
Using tensorflow to build a deep learning neural network model

[76]:
```
from unicodedata import name
import tensorflow as tf
import pydot
import graphviz
import seaborn as sns
import numpy as np

train_dataset = df.sample(frac=0.8, random_state=0)
test_dataset = df.drop(train_dataset.index)

X_train = train_dataset[['HomeM','Erf','Bedroom','Garage','Bath']].astype(float).values
y_train = train_dataset['Sales Price'].astype(float).values

# build deep learning model
model = tf.keras.models.Sequential([
```

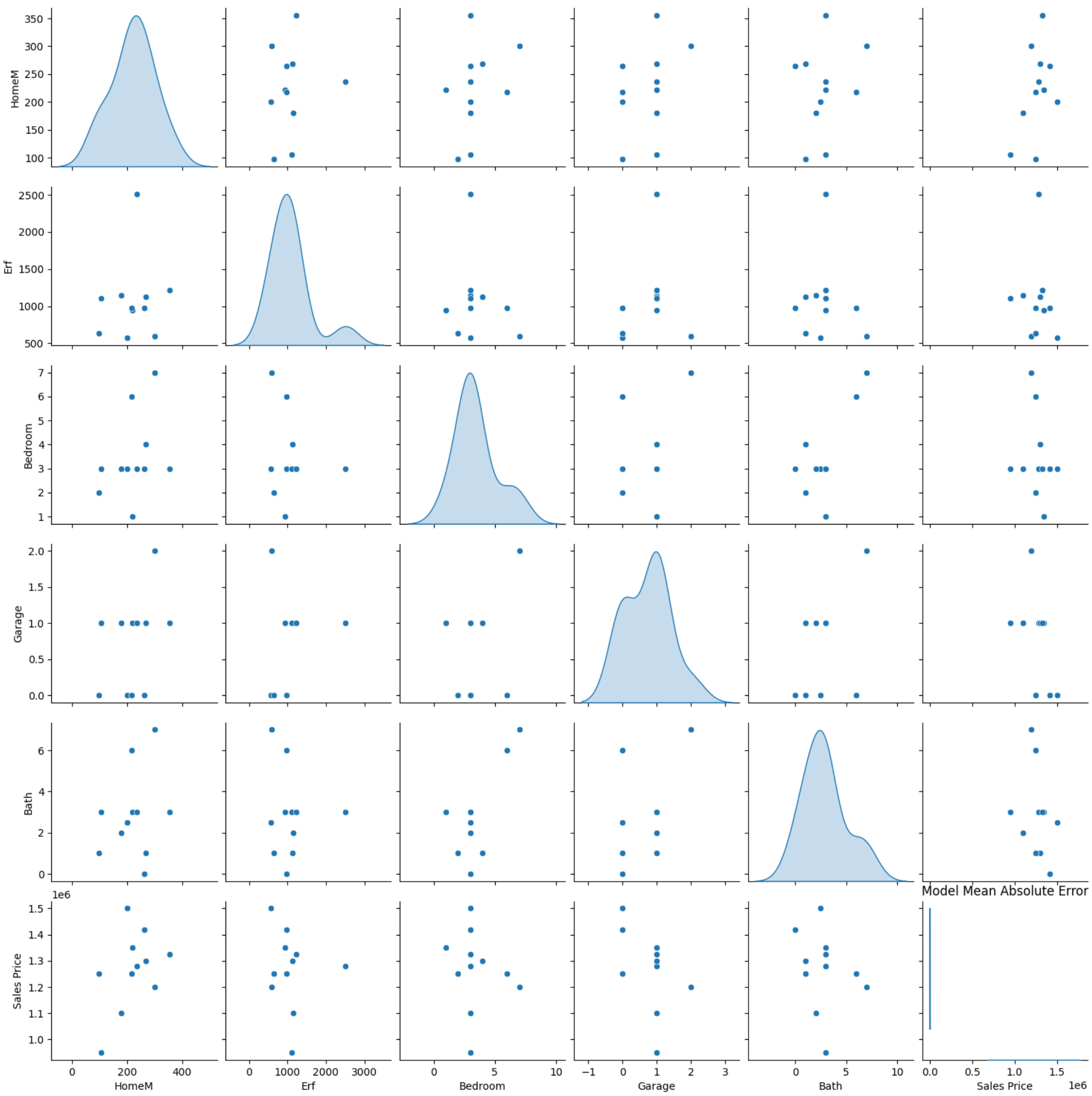
```
    tf.keras.layers.Dense(64, activation='relu',name='Input_Layer',␣
↪input_shape=[len(X_train[0])]),
    tf.keras.layers.Dense(64, activation='relu',name='Hidden_Layer_1'),
    tf.keras.layers.Dense(64, activation='relu',name='Hidden_Layer_2'),
    tf.keras.layers.Dense(1,name='Output_layer')
    ])

model.compile(loss='mean_squared_error',
                optimizer=tf.keras.optimizers.Adam(0.01),
                metrics=['mean_absolute_error', 'mean_squared_error'])

model.summary()

# train model
history = model.fit(X_train, y_train, epochs=1000, verbose=0)

# plot loss
plt.plot(history.history['loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train'], loc='upper left')
plt.show()


#inspecting data
sns.pairplot(df[['HomeM','Erf','Bedroom','Garage','Bath','Sales Price']],␣
↪diag_kind="kde").savefig('pairplot.png')

# plot mean absolute error
plt.plot(history.history['mean_absolute_error'])
plt.title('Model Mean Absolute Error')
plt.ylabel('Mean Absolute Error')

# predict house sales price
X_test = test_dataset[['HomeM','Erf','Bedroom','Garage','Bath']].astype(float).values
y_test = test_dataset['Sales Price'].astype(float).values
y_pred = model.predict(X_test).flatten()

pd.set_option('display.float_format', lambda x: '%.3f' % x)

# for all the data in a pandas dataframe predict house sales price
X = df[['HomeM','Erf','Bedroom','Garage','Bath']].astype(float).values
y = df['Sales Price'].astype(float).values
y_pred = model.predict(X).flatten().astype(float)
```

```
Model: "sequential_4"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Input_Layer (Dense)         (None, 64)                384


 Hidden_Layer_1 (Dense)      (None, 64)                4160
```

```
 Hidden_Layer_2 (Dense)      (None, 64)                4160

 Output_layer (Dense)        (None, 1)                 65

=================================================================
Total params: 8,769
Trainable params: 8,769
Non-trainable params: 0
_________________________________________________________________
```




```
1/1 [==============================] - 0s 35ms/step
1/1 [==============================] - 0s 14ms/step
```



Making a new prediction of a house in Bez Valley using a Neural Network

[77]:
```python
#"HomeM,Erf,Bedroom,Garage,Bathroom
X_new = [[248,495,5,1,4]]
y_new = model.predict(X_new).flatten().astype(float)

# put actual vs predicted house sales price in a table
df_predicted = pd.DataFrame({'HomeM':X_new[0][0],'Erf':X_new[0][1],'Bedroom':
↪X_new[0][2],'Garage':X_new[0][3],'Bathroom':X_new[0][4],'Predicted Sale Price':
↪y_new})
df_predicted
```

```
1/1 [==============================] - 0s 55ms/step
```

[77]:

	HomeM	Erf	Bedroom	Garage	Bathroom	Predicted Sale Price
0	248	495	5	1	4	1410527.375

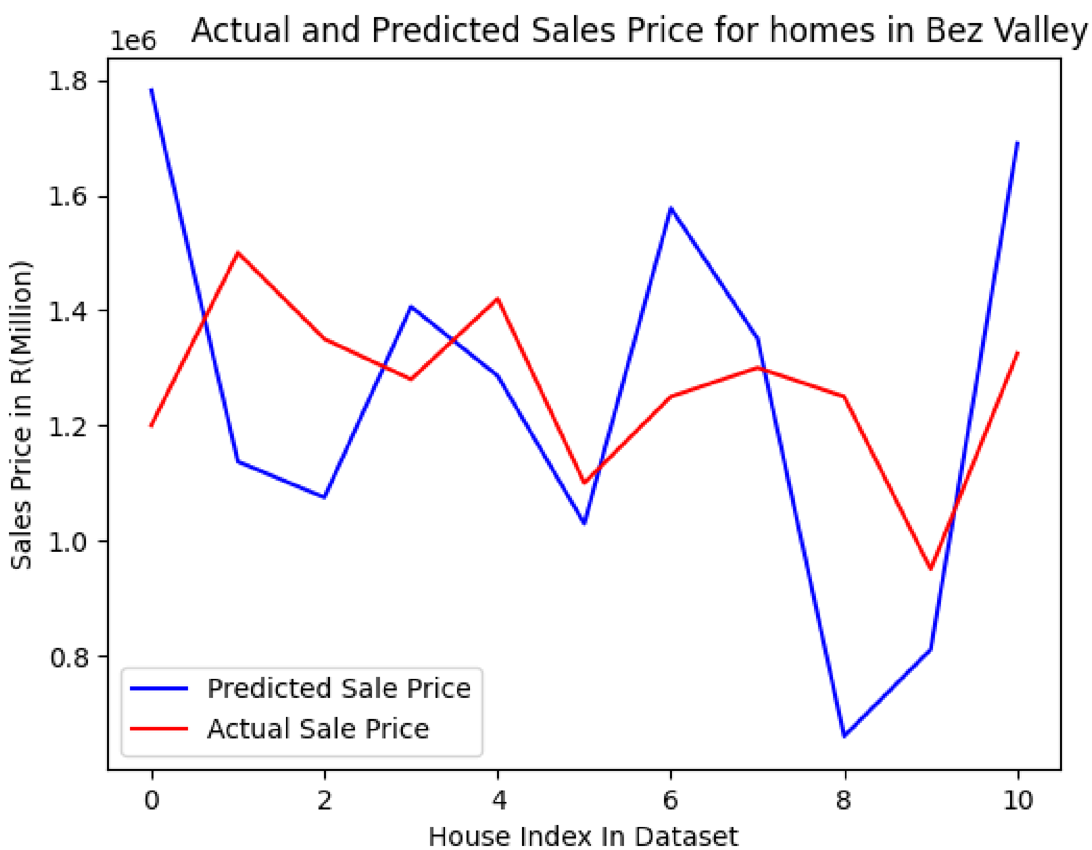
Our Dataframe using Neural Networks to predict sales price

[78]:
```
df['Predicted Sales Price'] = y_pred
df['Actual Sales Price'] = y
df
```

[78]:
```
                             Street Address           Township  \
0           214 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
1           193 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
2           276 8TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
3                 17 ORLANDO STREET KENSINGTON           KENSINGTON
4    66 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
5            40 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
6    64 ALBERTINA SISULU ROAD BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
7              2 7TH STREET BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
8           258 7TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
9            83 10TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY
10           16 11TH AVENUE BEZUIDENHOUT VALLEY  BEZUIDENHOUT VALLEY

   Erf I Portion  Sales Date     Reg Date Sales Price  Size    R/m^2  \
0          594 0    20211018 20220128.000 1200000.000   495  R 2 424
1          573 0    20211214        0.000 1500000.000   495    R3030
2          942 0    20220520        0.000 1350000.000   495   R 2727
3         2515 0    20220121 20220328.000 1280000.000   495  R 2 586
4          977 0    20211021 20211210.000 1420000.000   495  R 2 869
5         1148 0    20210803 20211112.000 1100000.000   495  R 2 222
6          976 0    20210714 20211102.000 1250000.000   495   R 2525
7         1131 0    20210806 20211115.000 1300000.000   495   R 2626
8            638    20210618 20210906.000 1250000.000   495   R 2525
9         1104 0    20210919 20220309.000  950000.000   495    R 919
10        1221 0    20210610 20211007.000 1325000.000   543    R 440

    Distance  Bedroom  Bath Garage   HomeM   R/HomeM      Erf Portion  \
0         94    7.000 7.000  2.000 300.000  4000.000  594.000        0
1        248    3.000 2.500  0.000 200.000  7500.000  573.000        0
2        478    1.000 3.000  1.000 221.000  6108.597  942.000        0
3        442    3.000 3.000  1.000 237.000  5400.844 2515.000        0
4        290    3.000 0.000  0.000 264.000  5378.788  977.000        0
5        231    3.000 2.000  1.000 180.000  6111.111 1148.000        0
6        304    6.000 6.000  0.000 218.000  5733.945  976.000        0
7        322    4.000 1.000  1.000 269.000  4832.714 1131.000        0
8        328    2.000 1.000  0.000  97.000 12886.598  638.000        0
9        297    3.000 3.000  1.000 105.000  9047.619 1104.000        0
10       303    3.000 3.000  1.000 356.000  3721.910 1221.000        0

    Predicted Sales Price  Actual Sales Price
0             1782373.375         1200000.000
1             1137158.250         1500000.000
2             1074770.250         1350000.000
3             1406023.250         1280000.000
4             1285983.125         1420000.000
5             1029428.188         1100000.000
6             1577281.875         1250000.000
7             1350767.375         1300000.000
```

```
8              660075.375        1250000.000
9              810844.938         950000.000
10            1689470.750        1325000.000
```

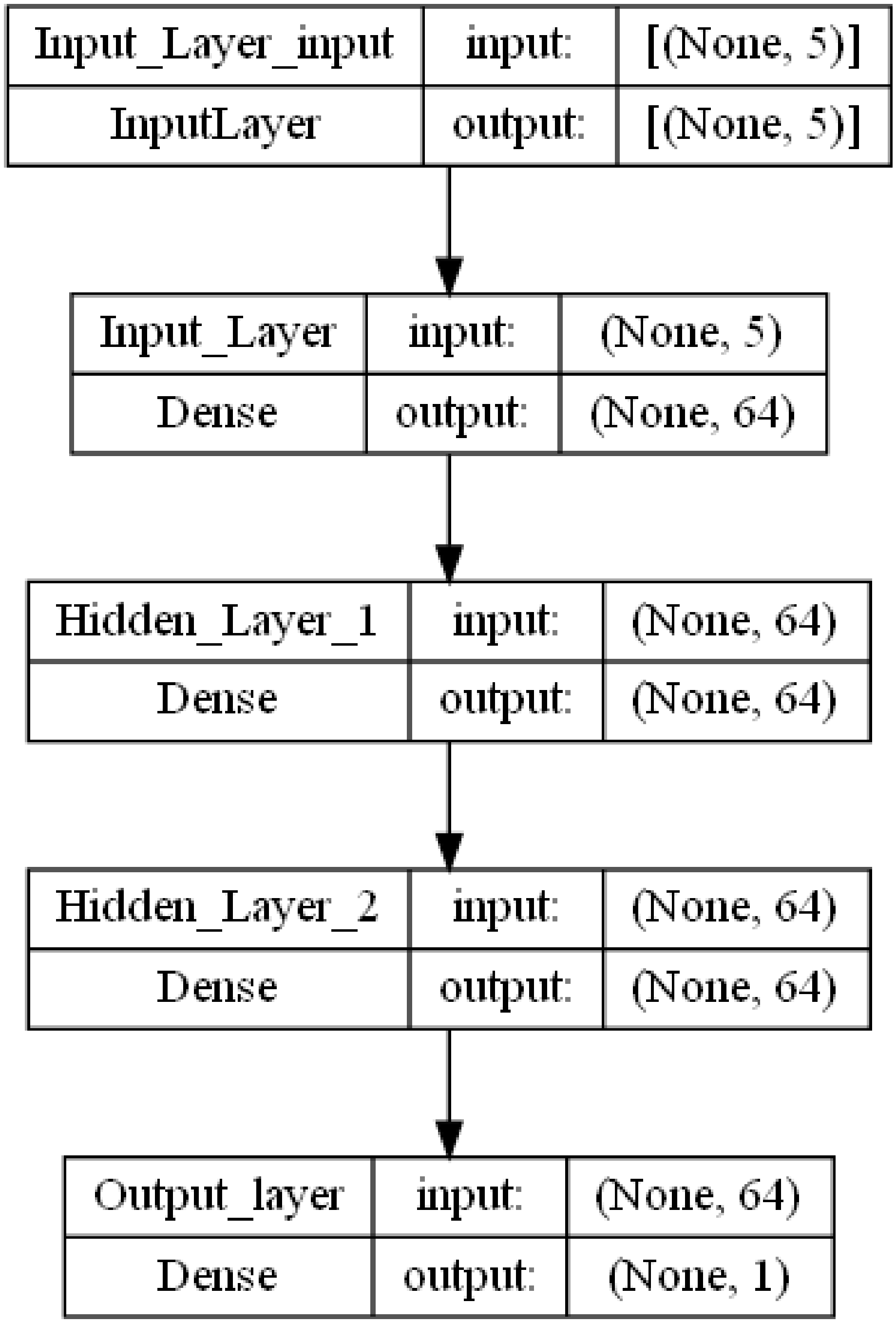
[79]:
```
# drawing line plot of actual vs predicted house sales price and saving fig
plt.plot(df['Predicted Sales Price'],color='blue')
plt.plot(df['Actual Sales Price'],color='red')
plt.xlabel('House Index In Dataset')
plt.ylabel('Sales Price in R(Million)')
plt.title('           Actual and Predicted Sales Price for homes in Bez Valley')
# add a legend
plt.legend(['Predicted Sale Price','Actual Sale Price'])

#export fig
plt.savefig('Actual_vs_Predicted_Sales_Price.png')
plt.show()
```



Displaying model as a neural network

[80]:
```
tf.keras.utils.plot_model(model, to_file='model.png', show_shapes=True,␣
 ↪show_layer_names=True)
```

[80]:


Input_Layer_input
input:
[(None, 5)]
InputLayer
output:
[(None, 5)]
Input_Layer
input:
(None, 5)
Dense
output:
(None, 64)
Hidden_Layer_1
input:
(None, 64)
Dense
output:
(None, 64)
Hidden_Layer_2
input:
(None, 64)
Dense
output:
(None, 64)
Output_layer
input:
(None, 64)
Dense
output:
(None, 1)