In [230…

```
#Name: Aiden O'Hara
#ID: 800956781
# Github Link: https://github.com/ajohara812/Aiden
```

In [273…

```
import numpy as np
import pandas as pd

#Data Visualization
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.datasets import load_iris
```

In [274…

```
housing = pd.DataFrame(pd.read_csv("Housing.csv"))
housing2 = pd.DataFrame(pd.read_csv("Housing.csv"))
housing.head()
```

Out[274…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
0	13300000	7420	4	2	3	yes	no	no	no
1	12250000	8960	4	4	4	yes	no	no	no
2	12250000	9960	3	2	2	yes	no	yes	no
3	12215000	7500	4	2	2	yes	no	yes	no
4	11410000	7420	4	1	2	yes	yes	yes	no

In [275…

```
m = len(housing)
```

In [276…

```
varlist = ['mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', '

def binary_map(x):
    return x.map({'yes': 1, 'no': 0})

housing[varlist] = housing[varlist].apply(binary_map)
housing2[varlist] = housing2[varlist].apply(binary_map)

housing.head()
```

Out[276…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
0	13300000	7420	4	2	3	1	0	0	0
1	12250000	8960	4	4	4	1	0	0	0
2	12250000	9960	3	2	2	1	0	1	0
3	12215000	7500	4	2	2	1	0	1	0
4	11410000	7420	4	1	2	1	1	1	0

In [277…

```
from sklearn.model_selection import train_test_split
df_train, df_test = train_test_split(housing, train_size=0.7, test_size = 0.3, random_s

from sklearn.model_selection import train_test_split
S_Train, S_Test = train_test_split(housing2, train_size=0.7, test_size = 0.3, random_st
```

In [278…

```
num_vars = ['area', 'bedrooms', 'stories', 'parking', 'price']
df_Newtrain = df_train[num_vars]
df_Newtest = df_test[num_vars]
df_Newtrain.head()
df_Newtest.head()
```

Out[278…

```
     area  bedrooms  stories  parking    price
316  5900         4        2        1  4060000
 77  6500         3        3        0  6650000
360  4040         2        1        0  3710000
 90  5000         3        2        0  6440000
493  3960         3        1        0  2800000
```

In [279…

```
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import MinMaxScaler, StandardScaler

#define standard scaler
scaler = MinMaxScaler()
df_Newtrain[num_vars] = scaler.fit_transform(df_Newtrain[num_vars])
df_Newtest[num_vars] = scaler.fit_transform(df_Newtest[num_vars])
df_Newtrain.head(20)
df_Newtest.head(20)
```

Out[279…

```
         area  bedrooms   stories   parking     price
316  0.365217      0.50  0.333333  0.333333  0.200000
 77  0.417391      0.25  0.666667  0.000000  0.424242
360  0.203478      0.00  0.000000  0.000000  0.169697
 90  0.286957      0.25  0.333333  0.000000  0.406061
493  0.196522      0.25  0.000000  0.000000  0.090909
209  0.436522      0.25  0.000000  0.000000  0.272727
176  0.593043      0.25  0.000000  0.666667  0.303030
249  0.286087      0.50  0.333333  0.000000  0.241818
516  0.133913      0.00  0.000000  0.333333  0.060606
426  0.086957      0.25  0.000000  0.000000  0.138788
  6  0.598261      0.50  1.000000  0.666667  0.727273
```

	area	bedrooms	stories	parking	price
497	0.194261	0.00	0.000000	0.000000	0.078788
422	0.175652	0.00	0.000000	0.000000	0.139394
424	0.121739	0.25	0.333333	0.000000	0.139394
529	0.197391	0.25	0.333333	0.000000	0.045455
499	0.167826	0.25	0.333333	0.000000	0.078788
498	0.026087	0.00	0.333333	0.000000	0.078788
55	0.373913	0.25	0.333333	0.333333	0.484848
476	0.360870	0.25	0.333333	0.333333	0.103030
486	0.373913	0.00	0.000000	0.000000	0.096970

In [280…

```
y_Newtrain = df_Newtrain.pop('price')
y2_Newtest = df_Newtest.pop('price')
X_Newtrain = df_Newtrain
X2_Newtest = df_Newtest
```

In [281…

```
X_Newtrain.head()
```

Out[281…

	area	bedrooms	stories	parking
126	0.378694	0.4	0.0	0.666667
363	0.132921	0.2	0.0	0.000000
370	0.180756	0.2	0.0	0.666667
31	0.367698	0.4	1.0	0.666667
113	0.547766	0.4	0.0	0.666667

In [282…

```
X2_Newtest.head()
```

Out[282…

	area	bedrooms	stories	parking
316	0.365217	0.50	0.333333	0.333333
77	0.417391	0.25	0.666667	0.000000
360	0.203478	0.00	0.000000	0.000000
90	0.286957	0.25	0.333333	0.000000
493	0.196522	0.25	0.000000	0.000000

In [283…

```
y_Newtrain.head()
y2_Newtest.head()
```

Out[283…

```
316    0.200000
```

```
77     0.424242
360    0.169697
90     0.406061
493    0.090909
Name: price, dtype: float64
```

In [284…

```
Y = y_Newtrain.values
Y2 = y2_Newtest.values
```

In [285…

```
X0 = df_Newtrain.values[:,0]
```

In [286…

```
price = df_train.values[:, 0]
area = df_train.values[:, 1]
bedrooms = df_train.values[:, 2]
bathrooms = df_train.values[:, 3]
stories = df_train.values[:, 4]
mainroad = df_train.values[:, 5]
guestroom = df_train.values[:, 6]
basement = df_train.values[:, 7]
hotwater = df_train.values[:, 8]
ac = df_train.values[:, 9]
parking = df_train.values[:, 10]
prefarea = df_train.values[:, 11]
furnished = df_train.values[:, 12]


price1 = df_test.values[:, 0]
area1 = df_test.values[:, 1]
bedroom1 = df_test.values[:, 2]
bathrooms1 = df_test.values[:, 3]
stories1 = df_test.values[:, 4]
mainroad1 = df_test.values[:, 5]
guestroom1 = df_test.values[:, 6]
basement1 = df_test.values[:, 7]
hotwater1 = df_test.values[:, 8]
ac1 = df_test.values[:, 9]
parking1 = df_test.values[:, 10]
prefarea1 = df_test.values[:, 11]
furnished1 = df_test.values[:, 12]


X_f = df_test.values[:,(1,2,3,4,10)]
m = len(df_test)
X_00 = np.ones((m,1))
X_f = np.hstack((X_00, X_f))
Y = df_train.values[:,0]
Y2 = df_test.values[:,0]
Y.astype('int64')
Y2.astype('int64')


M = len(df_train)
m = len(S_Train)
N = len(df_test)
n = len(S_Test)
```

In [287…

```
X_0 = np.ones((m, 1))
X_1 = area.reshape(m,1)
X_2 = bedrooms.reshape(m,1)
X_3 = bathrooms.reshape(m,1)
X_4 = stories.reshape(m,1)
X_5 = mainroad.reshape(m,1)
X_6 = guestroom.reshape(m,1)
X_7 = basement.reshape(m,1)
X_8 = hotwater.reshape(m,1)
X_9 = ac.reshape(m,1)
X_10 = parking.reshape(m,1)
X_11 = prefarea.reshape(m,1)
X_12 = furnished.reshape(m,1)
```

In [288…

```
X0 = np.ones((n, 1))
X1 = area2.reshape(n,1)
X2 = bedroom2.reshape(n,1)
X3 = bathrooms2.reshape(n,1)
X4 = stories2.reshape(n,1)
X5 = mainroad2.reshape(n,1)
X6 = guestroom2.reshape(n,1)
X7 = basement2.reshape(n,1)
X8 = hotwater2.reshape(n,1)
X9 = ac2.reshape(n,1)
X10 = parking2.reshape(n,1)
X11 = prefarea2.reshape(n,1)
X12 = furnished2.reshape(n,1)
```

In [289…

```
area = np.hstack((X0, X1))
bedrooms = np.hstack((X0, X2))
stories = np.hstack((X0, X4))
mainroad = np.hstack((X0, X5))
guestroom = np.hstack((X0, X6))
basement = np.hstack((X0, X7))
hotwater = np.hstack((X0, X8))
ac = np.hstack((X0, X9))
parking = np.hstack((X0, X10))
prefarea = np.hstack((X0, X11))
furnished = np.hstack((X0, X12))

Q1 = np.hstack((X0, X1,X2,X3,X4,X10))
Q2 = np.hstack((X0, X1,X2,X3,X4,X10))
```

In [290…

```
theta = np.zeros(6)
theta2 = np.zeros(6)
iterations = 1500;
alpha = 0.000001;
```

In [ ]:

```

```

In [300…

```
def compute_cost(X, Y, theta, Reg):

    predictions = X.dot(theta)
    errors = np.subtract(predictions - Y)
    sqrErrors = np.square(errors)
    if (Reg == 0):
        J = 1 / (2 * m) * np.sum(sqrErrors)
    else:
        J = 1/(2*m) * (np.sum(sqrErrors) + lam * (np.sum(theta) - theta2[0]))
        return J
```

In [303…

```
def gradient_descent(X,X2,Y, Y2, lamba, Reg, theta, alpha, iterations):

    cost_history = np.zeros(iterations)
    cost_history = np.zeros(iterations)

    for i in range(iterations):
        predictions = X.dot(theta)
        errors = np.subtract(predictions/10000, Y/10000)
        sum_delta = (alpha / m) * X.transpose().dot(errors);
        if (Reg == 0):
            theta = theta - sum_delta
        else:
            theta = theta * (1-alpha *(lamba/M))- sum_delta

        cost_history[i] = (compute_cost(X, Y,lamba, theta, Reg))
        cost_history2[i] = (compute_cost(X2,Y2, theta))

    return theta, cost_history ,cost_history2
```

In [304…

```
theta, cost_history, cost_history2 = gradient_descent(Q1, Q2, price, price2, 0.5, 0, th
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-304-4d919418409d> in <module>
----> 1 theta, cost_history, cost_history2 = gradient_descent(Q1, Q2, price, price2, 0.
5, 0, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-303-7b40cbdda848> in gradient_descent(X, X2, Y, Y2, lamba, Reg, theta, al
pha, iterations)
      6     for i in range(iterations):
      7         predictions = X.dot(theta)
----> 8         errors = np.subtract(predictions/10000, Y/10000)
      9         sum_delta = (alpha / m) * X.transpose().dot(errors);
     10         if (Reg == 0):

ValueError: operands could not be broadcast together with shapes (164,) (381,)
```

In [305…

```
plt.plot(range(1,iterations+1),cost_history,color ='blue')
plt.plot(range(1,iterations+1),cost_history2,color ='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
```

```
<ipython-input-305-28483e46f19c> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color ='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color ='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [295…

```
# Question 1b
```

In [306…

```
Qb = np.stack((X0,X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11))
Q1b =  np.hstack((X_0,X_1,X_2,X_3,X_4,X_5,X_6,X_7,X_8,X_9,X_10,X_11))
```

In [307…

```
theta = np.zeros(12)
iterations = 1500;
alpha = 0.000001;
```

In [308…

```
theta, cost_history, cost_history2 = gradient_descent(Qb, Q1b, price, price2, 0.5, 0, t
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-308-ccdfe15ad6fc> in <module>
----> 1 theta, cost_history, cost_history2 = gradient_descent(Qb, Q1b, price, price2,
0.5, 0, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-303-7b40cbdda848> in gradient_descent(X, X2, Y, Y2, lamba, Reg, theta, al
pha, iterations)
      5 
      6     for i in range(iterations):
----> 7         predictions = X.dot(theta)
      8         errors = np.subtract(predictions/10000, Y/10000)
      9         sum_delta = (alpha / m) * X.transpose().dot(errors);

ValueError: shapes (12,164,1) and (12,) not aligned: 1 (dim 2) != 12 (dim 0)
```

In [152…

```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Convergence of gradient descent')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-152-01cd831329a9> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')
```

```
NameError: name 'cost_history' is not defined
```

In [ ]:

In [ ]:

In [ ]:

```
# Problem 2a
```

In [251…

```
num_vars=['price','area','bedrooms','bathrooms','stories','mainroad','guestroom','basem
df_Newtrain=df_train[num_vars]
df_Newtest=df_test[num_vars]
df_Newtrain.head()
```

Out[251…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating
126	5880000	7160	3	1	1	1	0	1	0
363	3710000	3584	2	1	1	1	0	0	1
370	3640000	4280	2	1	1	1	0	0	0
31	8400000	7000	3	1	4	1	0	0	0
113	6083000	9620	3	1	1	1	0	1	0

In [252…

```
scaler=MinMaxScaler()
df_Newtrain[num_vars]=scaler.fit_transform(df_Newtrain[num_vars])
df_Newtrain.head(20)
```

Out[252…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
126	0.393333	0.378694	0.4	0.000000	0.000000	1.0	0.0	1.0	
363	0.186667	0.132921	0.2	0.000000	0.000000	1.0	0.0	0.0	
370	0.180000	0.180756	0.2	0.000000	0.000000	1.0	0.0	0.0	
31	0.633333	0.367698	0.4	0.000000	1.000000	1.0	0.0	0.0	
113	0.412667	0.547766	0.4	0.000000	0.000000	1.0	0.0	1.0	
222	0.286667	0.516564	0.2	0.000000	0.000000	1.0	0.0	1.0	
462	0.126667	0.035052	0.4	0.000000	0.333333	0.0	0.0	1.0	
177	0.332667	0.302405	0.4	0.000000	0.000000	1.0	0.0	1.0	
57	0.523333	0.505155	0.6	0.333333	1.000000	1.0	1.0	0.0	
244	0.266667	0.252234	0.4	0.000000	0.333333	1.0	1.0	1.0	
24	0.650000	0.491409	0.4	0.333333	0.333333	1.0	0.0	0.0	
17	0.686667	0.470790	0.4	0.333333	1.000000	1.0	0.0	0.0	

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
402	0.166667	0.290722	0.2	0.000000	0.000000	1.0	0.0	0.0	
66	0.493333	0.793814	0.2	0.000000	0.000000	1.0	0.0	1.0	
238	0.272667	0.196564	0.6	0.333333	0.333333	1.0	0.0	1.0	
272	0.246667	0.166667	0.4	0.000000	0.000000	1.0	1.0	1.0	
261	0.259333	0.127835	0.4	0.000000	0.333333	1.0	0.0	0.0	
353	0.193333	0.082887	0.4	0.000000	0.666667	1.0	0.0	0.0	
94	0.433333	0.298969	0.6	0.333333	1.000000	1.0	0.0	0.0	
180	0.330000	0.195876	0.6	0.333333	0.000000	0.0	0.0	1.0	

In [253…

```
scaler=MinMaxScaler()
df_Newtest[num_vars]=scaler.fit_transform(df_Newtest[num_vars])
df_Newtest.head(20)
```

Out[253…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
316	0.200000	0.365217	0.50	0.5	0.333333	0.0	0.0	1.0	
77	0.424242	0.417391	0.25	0.5	0.666667	1.0	0.0	0.0	
360	0.169697	0.203478	0.00	0.0	0.000000	1.0	0.0	0.0	
90	0.406061	0.286957	0.25	0.0	0.333333	1.0	0.0	0.0	
493	0.090909	0.196522	0.25	0.0	0.000000	1.0	0.0	0.0	
209	0.272727	0.436522	0.25	0.0	0.000000	1.0	0.0	0.0	
176	0.303030	0.593043	0.25	0.0	0.000000	1.0	0.0	0.0	
249	0.241818	0.286087	0.50	0.5	0.333333	1.0	1.0	1.0	
516	0.060606	0.133913	0.00	0.0	0.000000	0.0	1.0	0.0	
426	0.138788	0.086957	0.25	0.0	0.000000	0.0	0.0	0.0	
6	0.727273	0.598261	0.50	1.0	1.000000	1.0	0.0	0.0	
497	0.078788	0.194261	0.00	0.0	0.000000	1.0	0.0	0.0	
422	0.139394	0.175652	0.00	0.0	0.000000	0.0	0.0	0.0	
424	0.139394	0.121739	0.25	0.0	0.333333	0.0	0.0	1.0	
529	0.045455	0.197391	0.25	0.0	0.333333	1.0	0.0	1.0	
499	0.078788	0.167826	0.25	1.0	0.333333	0.0	1.0	0.0	
498	0.078788	0.026087	0.00	0.0	0.333333	1.0	0.0	0.0	
55	0.484848	0.373913	0.25	0.0	0.333333	1.0	0.0	0.0	
476	0.103030	0.360870	0.25	0.0	0.333333	1.0	0.0	1.0	
486	0.096970	0.373913	0.00	0.0	0.000000	1.0	0.0	0.0	

In [254…
```
y_Normtrain=df_Newtrain
df_Normtrain=df_Newtrain
```

In [255…
```
y_Normtest=df_Newtest
df_Normtest=df_Newtest
```

In [256…
```
df_Normtrain.head()
```

Out[256…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheat
126	0.393333	0.378694	0.4	0.0	0.0	1.0	0.0	1.0	
363	0.186667	0.132921	0.2	0.0	0.0	1.0	0.0	0.0	
370	0.180000	0.180756	0.2	0.0	0.0	1.0	0.0	0.0	
31	0.633333	0.367698	0.4	0.0	1.0	1.0	0.0	0.0	
113	0.412667	0.547766	0.4	0.0	0.0	1.0	0.0	1.0	

In [257…
```
df_Normtest.head()
```

Out[257…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
316	0.200000	0.365217	0.50	0.5	0.333333	0.0	0.0	1.0	
77	0.424242	0.417391	0.25	0.5	0.666667	1.0	0.0	0.0	
360	0.169697	0.203478	0.00	0.0	0.000000	1.0	0.0	0.0	
90	0.406061	0.286957	0.25	0.0	0.333333	1.0	0.0	0.0	
493	0.090909	0.196522	0.25	0.0	0.000000	1.0	0.0	0.0	

In [258…
```
y_Normtrain.head()
```

Out[258…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheat
126	0.393333	0.378694	0.4	0.0	0.0	1.0	0.0	1.0	
363	0.186667	0.132921	0.2	0.0	0.0	1.0	0.0	0.0	
370	0.180000	0.180756	0.2	0.0	0.0	1.0	0.0	0.0	
31	0.633333	0.367698	0.4	0.0	1.0	1.0	0.0	0.0	
113	0.412667	0.547766	0.4	0.0	0.0	1.0	0.0	1.0	

In [259…

```
y_Normtest.head()
```

Out[259…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
316	0.200000	0.365217	0.50	0.5	0.333333	0.0	0.0	1.0	
77	0.424242	0.417391	0.25	0.5	0.666667	1.0	0.0	0.0	
360	0.169697	0.203478	0.00	0.0	0.000000	1.0	0.0	0.0	
90	0.406061	0.286957	0.25	0.0	0.333333	1.0	0.0	0.0	
493	0.090909	0.196522	0.25	0.0	0.000000	1.0	0.0	0.0	

In [260…

```
Y = y_Normtrain.values
```

In [261…

```
Y2 = y_Normtest.values
```

In [262…

```
price = df_Normtrain.values[:, 0]
area = df_Normtrain.values[:, 1]
bedrooms = df_Normtrain.values[:, 2]
bathrooms = df_Normtrain.values[:, 3]
stories = df_Normtrain.values[:, 4]
mainroad = df_Normtrain.values[:, 5]
guestroom = df_Normtrain.values[:, 6]
basement = df_Normtrain.values[:, 7]
hotwater = df_Normtrain.values[:, 8]
ac = df_Normtrain.values[:, 9]
parking = df_Normtrain.values[:, 10]
prefarea = df_Normtrain.values[:, 11]


price_1 = df_Normtest.values[:, 0]
area_1 = df_Normtest.values[:, 1]
bedroom_1 = df_Normtest.values[:, 2]
bathrooms_1 = df_Normtest.values[:, 3]
stories_1 = df_Normtest.values[:, 4]
mainroad_1 = df_Normtest.values[:, 5]
guestroom_1 = df_Normtest.values[:, 6]
basement_1 = df_Normtest.values[:, 7]
hotwater_1 = df_Normtest.values[:, 8]
ac_1 = df_Normtest.values[:, 9]
parking_1 = df_Normtest.values[:, 10]
prefarea_1 = df_Normtest.values[:, 11]
```

In [263…

```
X0 = np.ones((m, 1))
X1 = area.reshape(m,1)
X2 = bedrooms.reshape(m,1)
X3 = bathrooms.reshape(m,1)
X4 = stories.reshape(m,1)
X5 = mainroad.reshape(m,1)
X6 = guestroom.reshape(m,1)
X7 = basement.reshape(m,1)
X8 = hotwater.reshape(m,1)
```

```
X9 = ac.reshape(m,1)
X10 = parking.reshape(m,1)
X11 = prefarea.reshape(m,1)
```

In [264…

```
X12 = np.ones((n, 1))
X13 = area2.reshape(n,1)
X14 = bedroom2.reshape(n,1)
X15 = bathrooms2.reshape(n,1)
X16 = stories2.reshape(n,1)
X17 = mainroad2.reshape(n,1)
X18 = guestroom2.reshape(n,1)
X19 = basement2.reshape(n,1)
X20 = hotwater2.reshape(n,1)
X21 = ac2.reshape(n,1)
X22 = parking2.reshape(n,1)
X23 = prefarea2.reshape(n,1)
```

In [265…

```
Q2 = np.hstack((X0,X1,X2,X3,X4,X10))
Q2a = np.hstack((X12,X13,X14,X15,X16,X22))
```

In [266…

```
theta = np.zeros(6)
theta2 = np.zeros(6)
iterations = 150000;
alpha = 0.1;
```

In [267…

```
theta, cost_history,cost_history2 = gradient_descent(Q2,Q2a, price, price2, 0.5,0, thet
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-267-6c60c6d55556> in <module>
----> 1 theta, cost_history,cost_history2 = gradient_descent(Q2,Q2a, price, price2, 0.5
,0, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-249-e3ac8dee640c> in gradient_descent(Q1, Q2, Y, Y2, lamba, Reg, theta, a
lpha, iterations)
      7         predictions = Q1.dot(theta)
      8         errors = np.subtract(predictions/10000, Y/10000)
----> 9         sum_delta = (alpha / m) * Q1a.transpose().dot(errors);
     10         if (Reg == 0):
     11             theta = theta - sum_delta

NameError: name 'Q1a' is not defined
```

In [268…

```
plt.plot(range(1,iterations+1),cost_history,color='green')
plt.plot(range(1,iterations+1),cost_history2,color='yellow')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Normalized')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-268-45b661304f21> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='green')
      2 plt.plot(range(1,iterations+1),cost_history2,color='yellow')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [269…

```
scaler=StandardScaler()
housing2[num_vars]= scaler.fit_transform(housing2[num_vars])
df_Newtest.head(20)
```

Out[269…

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe
316	0.200000	0.365217	0.50	0.5	0.333333	0.0	0.0	1.0	
77	0.424242	0.417391	0.25	0.5	0.666667	1.0	0.0	0.0	
360	0.169697	0.203478	0.00	0.0	0.000000	1.0	0.0	0.0	
90	0.406061	0.286957	0.25	0.0	0.333333	1.0	0.0	0.0	
493	0.090909	0.196522	0.25	0.0	0.000000	1.0	0.0	0.0	
209	0.272727	0.436522	0.25	0.0	0.000000	1.0	0.0	0.0	
176	0.303030	0.593043	0.25	0.0	0.000000	1.0	0.0	0.0	
249	0.241818	0.286087	0.50	0.5	0.333333	1.0	1.0	1.0	
516	0.060606	0.133913	0.00	0.0	0.000000	0.0	1.0	0.0	
426	0.138788	0.086957	0.25	0.0	0.000000	0.0	0.0	0.0	
6	0.727273	0.598261	0.50	1.0	1.000000	1.0	0.0	0.0	
497	0.078788	0.194261	0.00	0.0	0.000000	1.0	0.0	0.0	
422	0.139394	0.175652	0.00	0.0	0.000000	0.0	0.0	0.0	
424	0.139394	0.121739	0.25	0.0	0.333333	0.0	0.0	1.0	
529	0.045455	0.197391	0.25	0.0	0.333333	1.0	0.0	1.0	
499	0.078788	0.167826	0.25	1.0	0.333333	0.0	1.0	0.0	
498	0.078788	0.026087	0.00	0.0	0.333333	1.0	0.0	0.0	
55	0.484848	0.373913	0.25	0.0	0.333333	1.0	0.0	0.0	
476	0.103030	0.360870	0.25	0.0	0.333333	1.0	0.0	1.0	
486	0.096970	0.373913	0.00	0.0	0.000000	1.0	0.0	0.0	

In [270…

```
price3 = Test.values[:, 0]
area3 = Test.values[:, 1]
bedroom3 = Test.values[:, 2]
bathrooms3 = Test.values[:, 3]
stories3 = Test.values[:, 4]
```

```
mainroad3 = Test.values[:, 5]
guestroom3 = Test.values[:, 6]
basement3 = Test.values[:, 7]
hotwater3 = Test.values[:, 8]
ac3 = Test.values[:, 9]
parking3 = Test.values[:, 10]
prefarea3 = Test.values[:, 11]


price_3 = Train.values[:, 0]
area_3 = Train.values[:, 1]
bedrooms_3 = Train.values[:, 2]
bathrooms_3 = Train.values[:, 3]
stories_3 = Train.values[:, 4]
mainroad_3 = Train.values[:, 5]
guestroom_3 = Train.values[:, 6]
basement_3 = Train.values[:, 7]
hotwater_3 = Train.values[:, 8]
ac_3 = Train.values[:, 9]
parking_3 = Train.values[:, 10]
prefarea_3 = Train.values[:, 11]
```

In [271…

```
S0 = np.ones((N,1))
S1 = price2.reshape(N,1)
S2 = area2.reshape(N,1)
S3 = bedroom2.reshape(N,1)
S4 = bathrooms2.reshape(N,1)
S5 = stories2.reshape(N,1)
S6 = mainroad2.reshape(N,1)
S7 = basement2.reshape(N,1)
S8 = hotwater2.reshape(N,1)
S9 = ac2.reshape(N,1)
S10 = parking2.reshape(N,1)
S11 = prefarea2.reshape(N,1)


S12 = np.ones((M,1))
S13 = price.reshape(M,1)
S14 = area.reshape(M,1)
S15 = bedrooms.reshape(M,1)
S16 = bathrooms.reshape(M,1)
S17 = stories.reshape(M,1)
S18 = mainroad.reshape(M,1)
S19 = basement.reshape(M,1)
S20 = hotwater.reshape(M,1)
S21 = ac.reshape(M,1)
S22 = parking.reshape(M,1)
S23 = prefarea.reshape(M,1)
```

In [272…

```
Q2 = np.hstack((S12,S13,S,S15,S16,S22))
Q_2b = np.hstack((S0,S1,S2,S3,S4,S10))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-272-e5343c6714df> in <module>
----> 1 Q2 = np.hstack((S12,S13,S,S15,S16,S_22))
      2 Q_2SS = np.hstack((S0,S1,S2,S3,S4,S10))
```

```
NameError: name 'S' is not defined
```

In [ ]:
```
theta = np.zeros(6)
theta2 = np.zeros(6)
iterations = 1500;
alpha = 0.000001;
```

In [ ]:
```
theta, cost_history,cost_history2 = gradient_descent(Q2,Q_2b, price, price2, 0.5, 0, th
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

In [ ]:
```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Standardized Graph')
```

In [ ]:
```
Q2 = np.hstack((X0,X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11))
Q_2b = np.hstack((X_0,X_1,X_2,X_3,X_4,X_5,X_6,X_7,X_8,X_9,X_10,X_11))
```

In [ ]:
```
theta = np.zeros(12)
theta2 = np.zeros(12)
iterations = 1500;
alpha = 0.000001;
```

In [ ]:
```
theta, cost_history, cost_history2 = gradient_descent(Q2, Q_2b, price, price2, 0.5, 0,
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

In [ ]:
```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Normalized Graph')
```

In [311…
```
Q2 = np.hstack((S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11))
Q_2b = np.hstack((S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7,S_8,S_9,S_10,S_11))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-311-01bbf7f6348b> in <module>
      1 Q2 = np.hstack((S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11))
----> 2 Q_2b = np.hstack((S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7,S_8,S_9,S_10,S_11))

NameError: name 'S_0' is not defined
```

In [312…
```
theta = np.zeros(12)
theta2 = np.zeros(12)
iterations = 1500;
alpha = 0.000001;
```

In [313…
```
theta1, cost_history, cost_history2 = gradient_descent(Q2, Q_2b, price, price2, 0.5,0,
print('Final value of theta =', theta1)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-313-6972e3b14997> in <module>
----> 1 theta1, cost_history, cost_history2 = gradient_descent(Q2, Q_2b, price, price2,
0.5,0, theta1, alpha, iterations)
      2 print('Final value of theta =', theta1)
      3 print('cost_history =', cost_history)

NameError: name 'Q_2b' is not defined
```

In [314…
```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Standardized Graph 2')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-314-aee976662354> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [315…
```
# 3A
```

In [316…
```
Q3 = np.hstack((X_0, X_1,X_2,X_3,X_4,X_10))
Q_3 = np.hstack((X_0T, X_1T,X_2T,X_3T,X_4T,X_10T))
```

In [317…
```
Q3S = np.hstack((S_0, S_1,S_2,S_3,S_4,S_10))
Q_3 = np.hstack((S_0T, S_1T,S_2T,S_3T,S_4T,S_10T))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-317-82c2b956f398> in <module>
----> 1 Q3S = np.hstack((S_0, S_1,S_2,S_3,S_4,S_10))
      2 Q_3 = np.hstack((S_0T, S_1T,S_2T,S_3T,S_4T,S_10T))

NameError: name 'S_0' is not defined
```

In [318…
```
theta = np.zeros(6)
```

```
theta2 = np.zeros(6)
iterations = 1500;
alpha = 0.000001;
```

In [319…

```
theta, cost_history, cost_history2 = gradient_descent(Q3, Q_3, price, price2, 0.5, 1, t
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-319-85e6285f0ed6> in <module>
----> 1 theta, cost_history, cost_history2 = gradient_descent(Q3, Q_3, price, price2,
0.5, 1, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-303-7b40cbdda848> in gradient_descent(X, X2, Y, Y2, lamba, Reg, theta, al
pha, iterations)
     13             theta = theta * (1-alpha *(lamba/M))- sum_delta
     14
---> 15         cost_history[i] = (compute_cost(X, Y,lamba, theta, Reg))
     16         cost_history2[i] = (compute_cost(X2,Y2, theta))
     17

TypeError: compute_cost() takes 4 positional arguments but 5 were given
```

In [320…

```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Penelty of Normalization Graph')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-320-c4f2d0bda2c3> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [321…

```
theta, cost_history, cost_history2 = gradient_descent(Q3, Q_3, price, price2, 0.5, 1, t
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-321-85e6285f0ed6> in <module>
----> 1 theta, cost_history, cost_history2 = gradient_descent(Q3, Q_3, price, price2,
0.5, 1, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-303-7b40cbdda848> in gradient_descent(X, X2, Y, Y2, lamba, Reg, theta, al
pha, iterations)
```

```
     13             theta = theta * (1-alpha *(lamba/M))- sum_delta
     14 
---> 15         cost_history[i] = (compute_cost(X, Y,lamba, theta, Reg))
     16         cost_history2[i] = (compute_cost(X2,Y2, theta))
     17 

TypeError: compute_cost() takes 4 positional arguments but 5 were given
```

In [322…

```python
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Penalty of Standardization Graph')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-322-79dbe69dd983> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [ ]:

```python

```

In [323…

```python
#3b
```

In [324…

```python
Q3b = np.hstack((X_0,X_1,X_2,X_3,X_4,X_5,X_6,X_7,X_8,X_9,X_10,X_11))
Q_3b = np.hstack((X_0,X_1,X_2,X_3,X_4,X_5,X_6,X_7,X_8,X_9,X_10,X_11))

Q3S = np.hstack((S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11))
Q_3S = np.hstack((S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7,S_8,S_9,S_10,S_11))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-324-f15cc9a72dd1> in <module>
      3 
      4 Q3S = np.hstack((S0,S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11))
----> 5 Q_3S = np.hstack((S_0,S_1,S_2,S_3,S_4,S_5,S_6,S_7,S_8,S_9,S_10,S_11))

NameError: name 'S_0' is not defined
```

In [325…

```python
theta = np.zeros(12)
theta2 = np.zeros(12)
iterations = 1500;
alpha = 0.001;
```

In [326…

```python
theta, cost_history, cost_history2 = gradient_descent(Q3b, Q_3b, price, price2, 0.25, 1
print('Final value of theta =', theta)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-326-90ce5b702dcc> in <module>
----> 1 theta, cost_history, cost_history2 = gradient_descent(Q3b, Q_3b, price, price2,
0.25, 1, theta, alpha, iterations)
      2 print('Final value of theta =', theta)
      3 print('cost_history =', cost_history)

<ipython-input-303-7b40cbdda848> in gradient_descent(X, X2, Y, Y2, lamba, Reg, theta, al
pha, iterations)
     13             theta = theta * (1-alpha *(lamba/M))- sum_delta
     14 
---> 15         cost_history[i] = (compute_cost(X, Y,lamba, theta, Reg))
     16         cost_history2[i] = (compute_cost(X2,Y2, theta))
     17 

TypeError: compute_cost() takes 4 positional arguments but 5 were given
```

In [327…

```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Penalty of Normalization Graph')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-327-2caec9dd0450> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```

In [328…

```
theta1, cost_history, cost_history2 = gradient_descent(Q3S, Q_3S, price, price2, 0.1, 1
print('Final value of theta =', theta1)
print('cost_history =', cost_history)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-328-7d70ef88438f> in <module>
----> 1 theta1, cost_history, cost_history2 = gradient_descent(Q3S, Q_3S, price, price2
, 0.1, 1, theta, alpha, iterations)
      2 print('Final value of theta =', theta1)
      3 print('cost_history =', cost_history)

NameError: name 'Q_3S' is not defined
```

In [329…

```
plt.plot(range(1,iterations+1),cost_history,color='blue')
plt.plot(range(1,iterations+1),cost_history2,color='red')
plt.rcParams["figure.figsize"]=(10,6)
plt.grid()
plt.xlabel('Number of iterations')
plt.ylabel('Cost (J)')
plt.title('Penalty of Standardization Graph')
```

```
---------------------------------------------------------------------------
```

```
NameError                                 Traceback (most recent call last)
<ipython-input-329-79dbe69dd983> in <module>
----> 1 plt.plot(range(1,iterations+1),cost_history,color='blue')
      2 plt.plot(range(1,iterations+1),cost_history2,color='red')
      3 plt.rcParams["figure.figsize"]=(10,6)
      4 plt.grid()
      5 plt.xlabel('Number of iterations')

NameError: name 'cost_history' is not defined
```