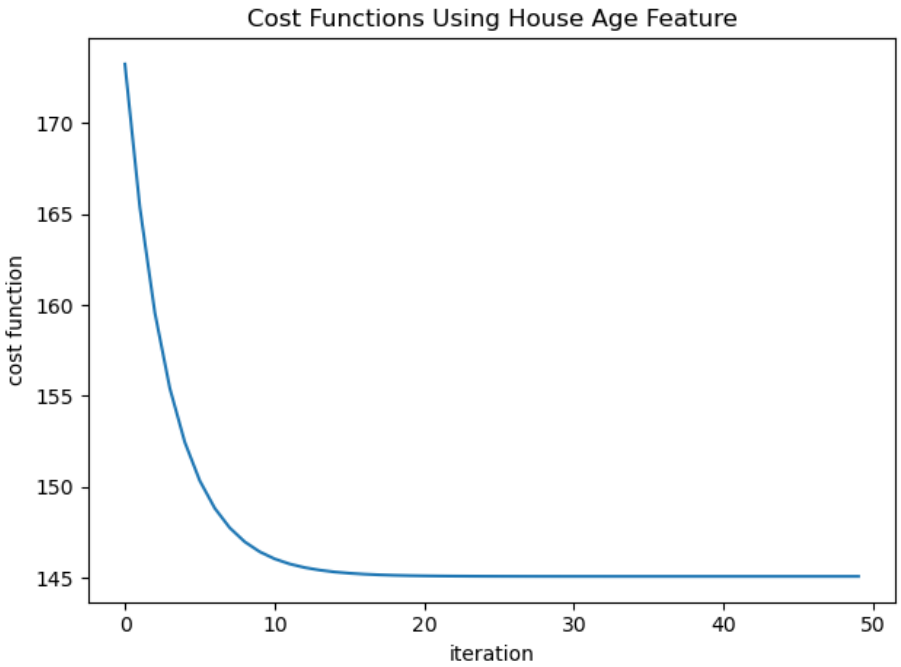
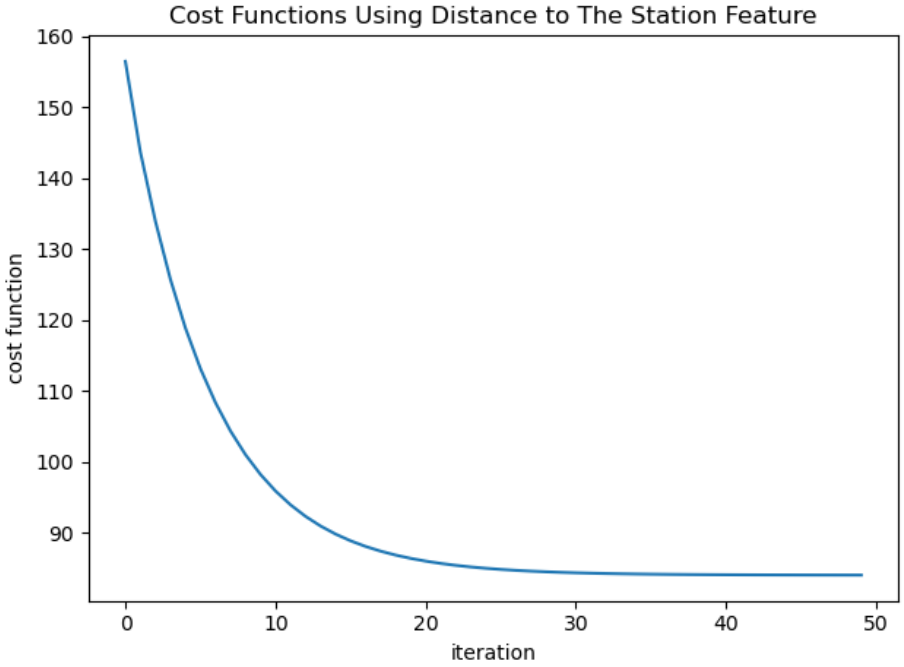
COMP 9417 Homework 1

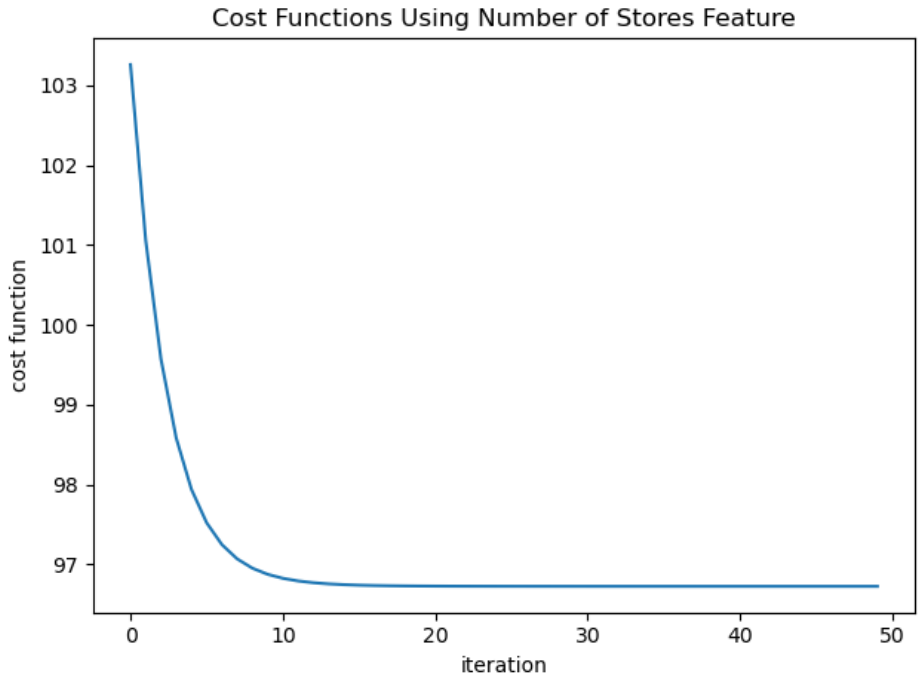
term 1 2020

Ziyi Wang z5248041

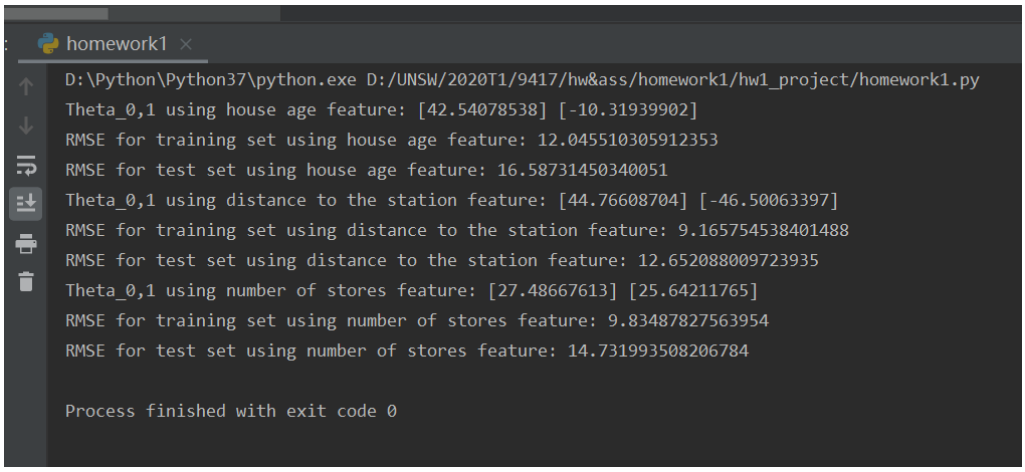
1. θ0, θ1 for step 3 using house age feature: 42.541 and -10.319
2.







3. RMSE for training set using house age feature: 12.045
4. RMSE for test set using house age feature: 16.587
5. RMSE for test set using distance to the station feature: 12.652
6. RMSE for test set using number of stores feature: 14.732
7.



By comparing the RMSE for test set using different features, we get that RMSE for test set using distance to the station feature is the smallest, followed by RMSE for test set using number of stores feature and then RMSE for test set using house age feature. Hence the model using distance to the station feature ranks the first. The model using number of stores feature ranks the second. The model using house age feature ranks the last.

The code.

```
1.  import pandas
2.  import numpy
3.  import math
4.  from matplotlib import pyplot as plt
5.
6.
7.  def normalization(df):
8.      return (df-df.min())/(df.max()-df.min())
9.
10. def SGD(x_training, y_training):
11.     theta_0 = -1
12.     theta_1 = -0.5
13.     loss = []
14.     y_pred = numpy.zeros((300, 1))
15.
16.     for i in range(50):
17.         for j in range(300):
18.             y_pred[j][0] = x_training[j] * theta_1 + theta_0
19.             theta_0 = theta_0 + 0.01 * (y_training[j] - y_pred[j][0])
20.             theta_1 = theta_1 + 0.01 * (y_training[j] - y_pred[j][0]) * x_tr
    aining[j]
21.
22.         for k in range(300):
23.             y_pred[k][0] = x_training[k] * theta_1 + theta_0
24.
25.         s = (y_training - y_pred) * (y_training - y_pred)
26.         s_list = s.tolist()
27.         sl = []
28.         for n in s_list:
29.             for m in n:
30.                 sl.append(m)
31.         loss.append(sum(sl) / 300)
32.     return theta_0, theta_1, loss
33.
34. def RMSE(x_test, y_test, theta_0, theta_1):
35.     y_pred = numpy.zeros((100, 1))
36.     for i in range(100):
37.         y_pred[i][0] = x_test[i] * theta_1 + theta_0
38.     s = (y_test - y_pred) * (y_test - y_pred)
39.     s = s.tolist()
40.     sl = []
41.     for i in s:
42.         for j in i:
```

```
            sl.append(j)
    result = math.sqrt(sum(sl) / 100)
    return result

file = pandas.read_csv('house_prices.csv')
df = pandas.DataFrame(file)
df_curr = df.drop('No', axis = 1)
y = df_curr[['house price of unit area']]
x = df_curr[['house age', 'distance to the nearest MRT station', 'number of convenience stores']]

x_preprocessed = normalization(x)

x_training = x_preprocessed[0: 300]
x_test = x_preprocessed[300: 400]
y_training = (y[0: 300]).values
y_test = (y[300: 400]).values

x_hatrn = x_training[['house age']].values
theta_0_ha, theta_1_ha, loss_ha = SGD(x_hatrn, y_training)
plt.title('Cost Functions Using House Age Feature')
plt.xlabel('iteration')
plt.ylabel('cost function')
plt.plot(loss_ha)
plt.show()
RMSE_hatrn = math.sqrt(loss_ha[49])
print('Theta_0,1 using house age feature:', theta_0_ha, theta_1_ha)
print('RMSE for training set using house age feature:', RMSE_hatrn)

x_hatst = x_test[['house age']].values
RMSE_hatst = RMSE(x_hatst, y_test, theta_0_ha, theta_1_ha)
print('RMSE for test set using house age feature:', RMSE_hatst)

x_dtstrn = x_training[['distance to the nearest MRT station']].values
theta_0_dts, theta_1_dts, loss_dts = SGD(x_dtstrn, y_training)
plt.title('Cost Functions Using Distance to The Station Feature')
plt.xlabel('iteration')
plt.ylabel('cost function')
plt.plot(loss_dts)
plt.show()
RMSE_dtstrn = math.sqrt(loss_dts[49])
print('Theta_0,1 using distance to the station feature:', theta_0_dts, theta_1_dts)
```

```
84. print('RMSE for training set using distance to the station feature:', RMSE_d
    tstrn)
85.
86. x_dtstst = x_test[['distance to the nearest MRT station']].values
87. RMSE_dtstst = RMSE(x_dtstst, y_test, theta_0_dts, theta_1_dts)
88. print('RMSE for test set using distance to the station feature:', RMSE_dtsts
    t)
89.
90. x_ncstrn = x_training[['number of convenience stores']].values
91. theta_0_ncs, theta_1_ncs, loss_ncs = SGD(x_ncstrn, y_training)
92. plt.title('Cost Functions Using Number of Stores Feature')
93. plt.xlabel('iteration')
94. plt.ylabel('cost function')
95. plt.plot(loss_ncs)
96. plt.show()
97. RMSE_ncstrn = math.sqrt(loss_ncs[49])
98. print('Theta_0,1 using number of stores feature:', theta_0_ncs, theta_1_ncs)

99. print('RMSE for training set using number of stores feature:', RMSE_ncstrn)

100.
101. x_ncstst = x_test[['number of convenience stores']].values
102. RMSE_ncstst = RMSE(x_ncstst, y_test, theta_0_ncs, theta_1_ncs)
103. print('RMSE for test set using number of stores feature:', RMSE_ncstst)
```