Machine Learning
Assignment 1
By: Chavan Prathamesh Vasant
Roll no: 15CS30010

Data preprocessing:

Data is read into pandas dataframe from the excel sheet and immediately converted into numpy arrays.
To handle a bias theta_0, we add another column to X(input values) containing all ones.
But before adding, we do mean normalization of the given data(both testing and training data).

A. Gradient descent was used to calculate the values of theta.
Finally we obtained the value of our hypothesis function as:
Finally Learned Values of theta are:
h_theta(X) = 2297.9064171 * sqft' + 38196.33043568 * floors' + 7629.9444942 * bedrooms' + 41309.34747318 * bathrooms' + 492937.52512177

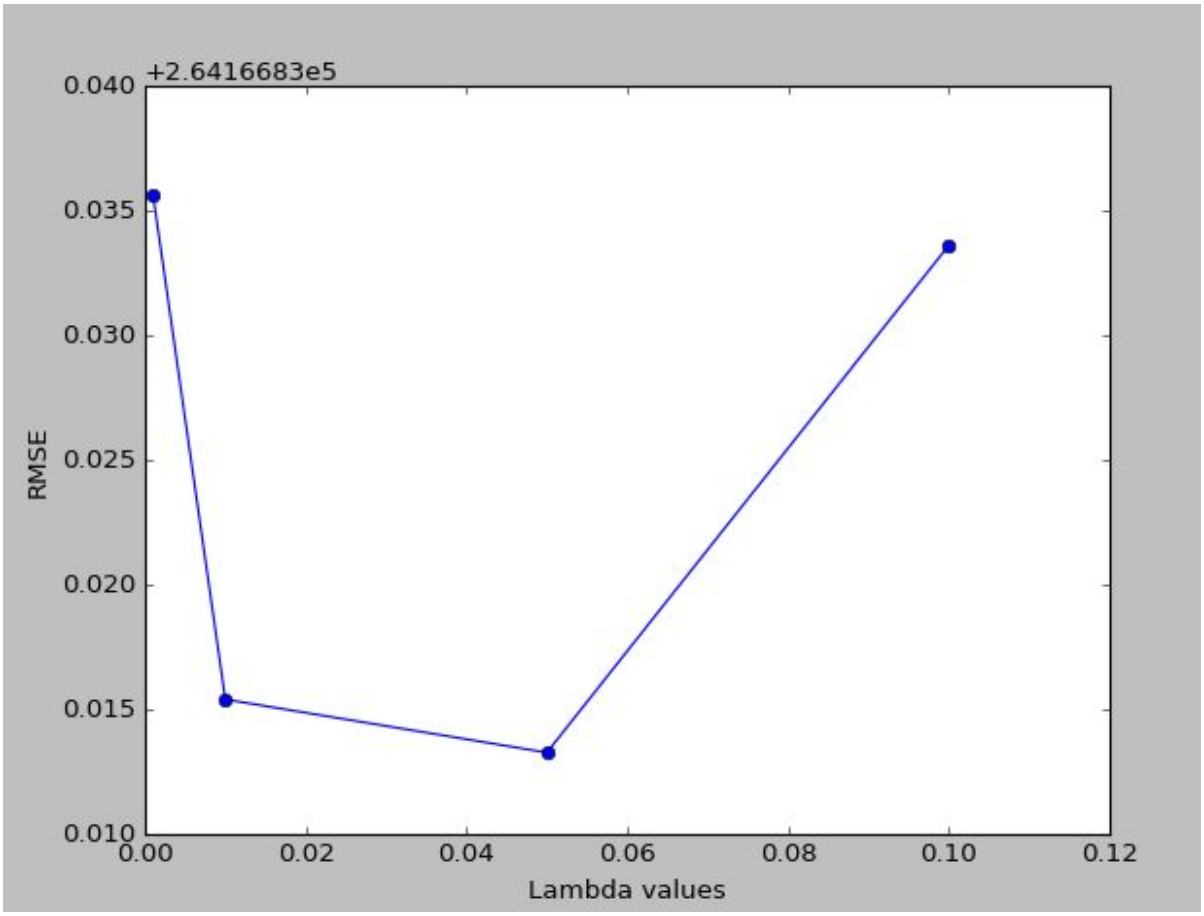
where:
sqft' = (sqft - 15106.9675658) / 1650839.0
floors' = (floors - 1.49430898071) / 2.5
bedrooms' = (bedrooms - 3.3708416231) / 33.0
bathrooms' = (bathrooms - 2.11475732198) / 8.0

When regression is applied:

When alpha = 0.05 and lambda = 0.05
RMSE finally obtained is 264166.843269

Hence final hypothesis function obtained is:
h_theta(X) = sqft' * 2297.73026681 + floors' * 38196.2720313 + bedrooms' * 7630.45084976 + bathrooms' * 41308.90913604 + 492937.53729354

where:
sqft' = (sqft - 15106.9675658) / 1650839.0
floors' = (floors - 1.49430898071) / 2.5
bedrooms' = (bedrooms - 3.3708416231) / 33.0
bathrooms' = (bathrooms - 2.11475732198) / 8.0



B. (i) gradient descent with learning rate of 0.05

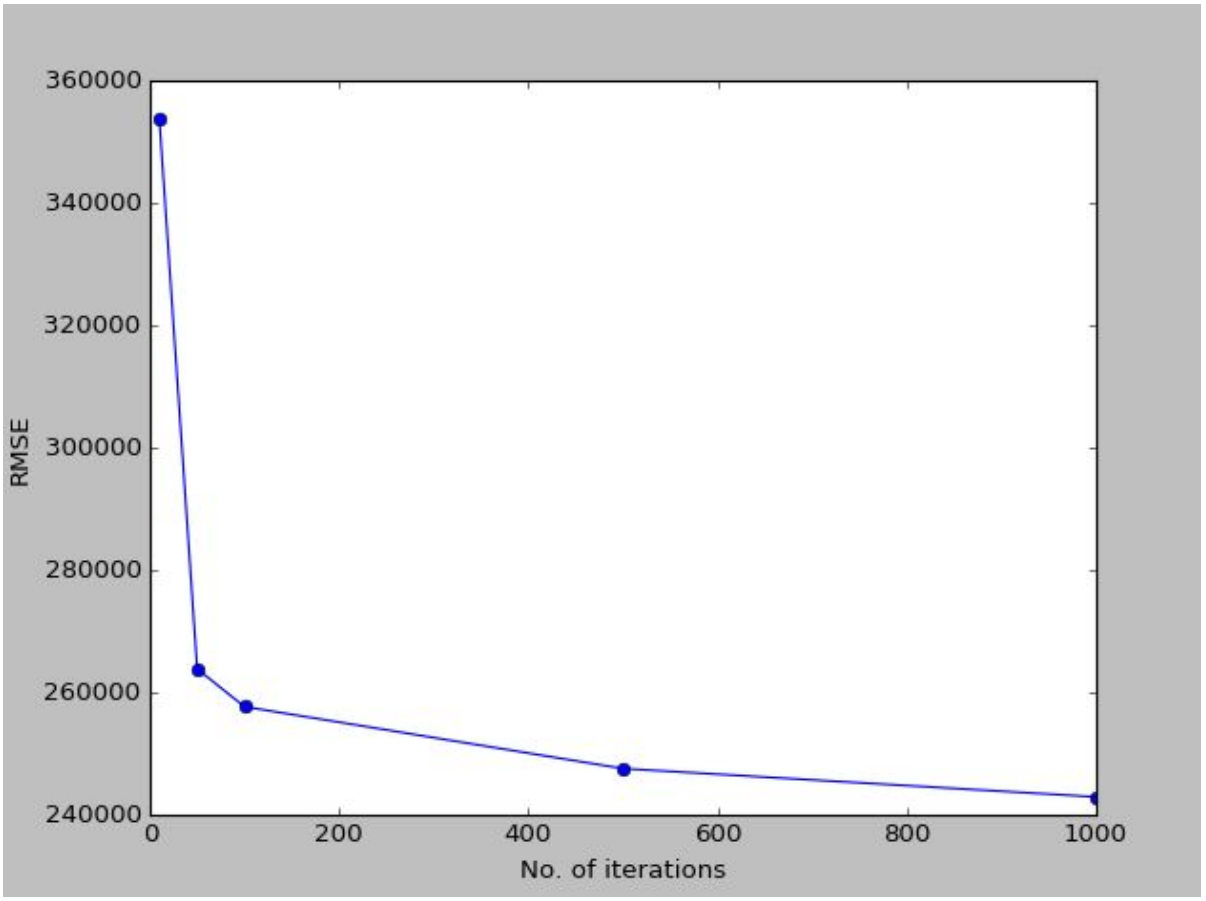
After thousand iterations:

RMSE finally obtained is 242917.232716

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4]

theta[] = [36659.92793143 346969.55343873 114849.19959393 623011.70723612
545696.46596066]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0
d = (bathrooms - 2.11475732198) / 8.0



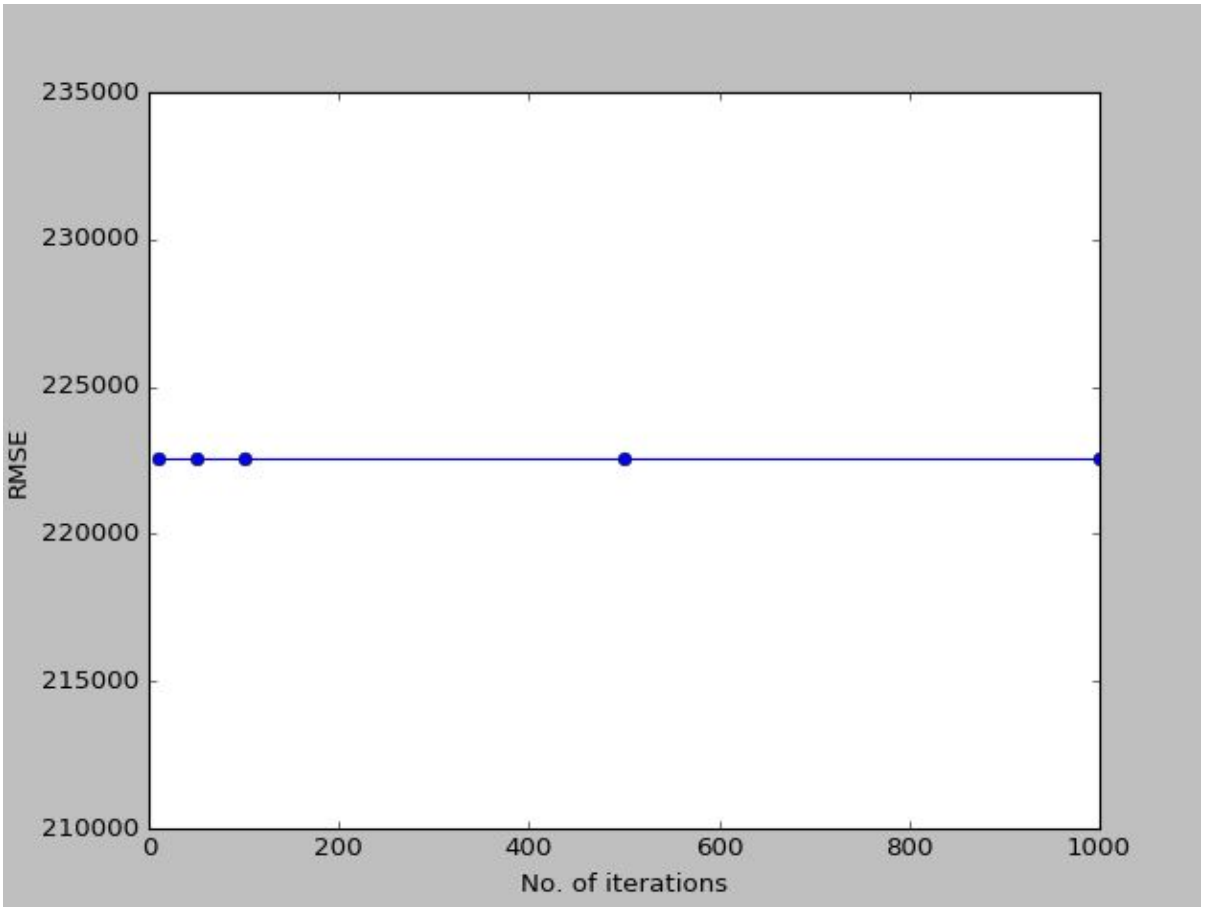
(ii) iterative re-weighted least square method

RMSE finally obtained is 222550.30125

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4]

theta[] = [15630.55883872 10871.68870279 16248.61794073 178987.96041402
546372.98102852]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0
d = (bathrooms - 2.11475732198) / 8.0



Conclusion:
I would prefer to use IRLS method for finding the optimized hypothesis function, as we can see that it gives the least amount of RMSE, and doesn't even require any iterations. Although We can also note that if the training data is very very big, computation using the IRLS method would be very costly, where as in gradient descent, we will still have options of stochastic gradient descent, and obtain good enough values of theta for our hypothesis function.

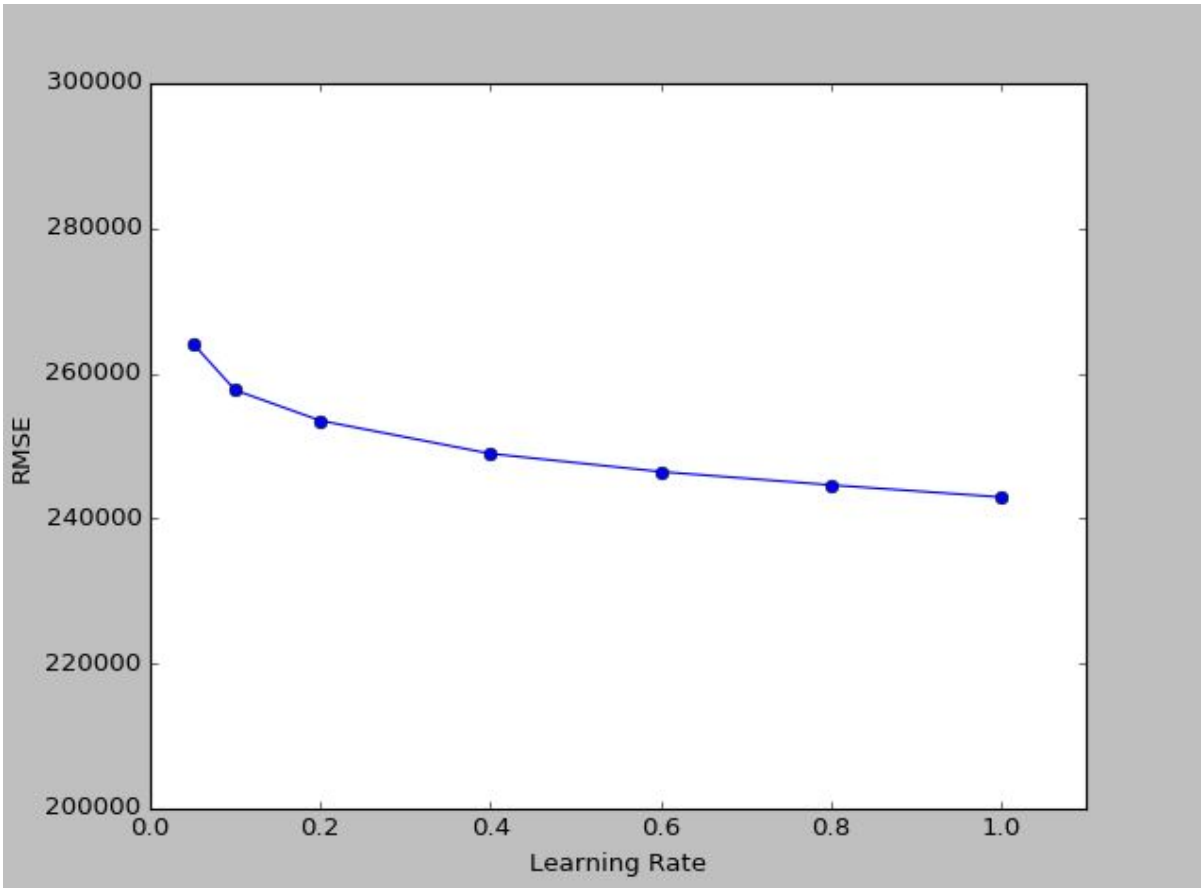
C. i) Linear:

When alpha = 0.05
RMSE finally obtained is 264166.861479

Hence final hypothesis function obtained is:
h_theta(X) = sqft' * 2297.68444117 + floors' * 38196.08889875 + bedrooms' * 7630.37229123 + bathrooms' *41308.72424696 + 492937.50683895

where:
sqft' = (sqft - 15106.9675658) / 1650839.0
floors' = (floors - 1.49430898071) / 2.5
bedrooms' = (bedrooms - 3.3708416231) / 33.0

bathrooms' = (bathrooms - 2.11475732198) / 8.0



C. ii) Quadratic

When learning rate = 0.05

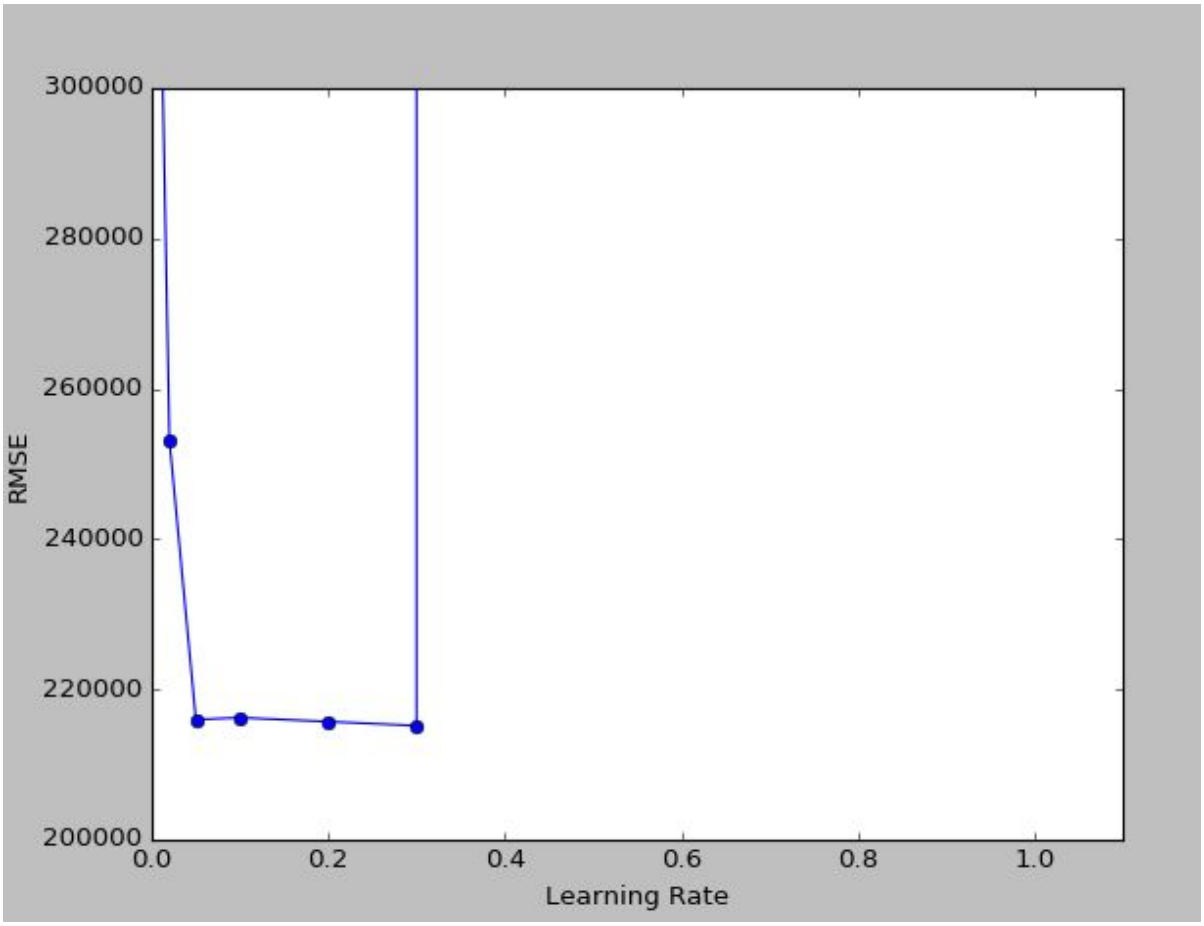
RMSE finally obtained is 215994.773035

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4] + theta[5] * a * a + theta[6] * b * b + theta[7] * c * c + theta[8] * d * d + theta[9] * a * b + theta[10] * a * c + theta[11] * a * d + theta[12] * b * c + theta[13] * b * d + theta[14] * c * d

Where theta[] = [2886.19335823 -7615.6753166 3394.46024755 44199.45801445
502255.99340829 -10095.34191252 -13421.36989322 -7866.97847001
88441.94622803 8970.04781443 1928.68641388 11492.98410826
12084.51352257 29585.13135829 47046.08513199]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0

d = (bathrooms - 2.11475732198) / 8.0



C iii) Cubic

When learning rate = 0.05

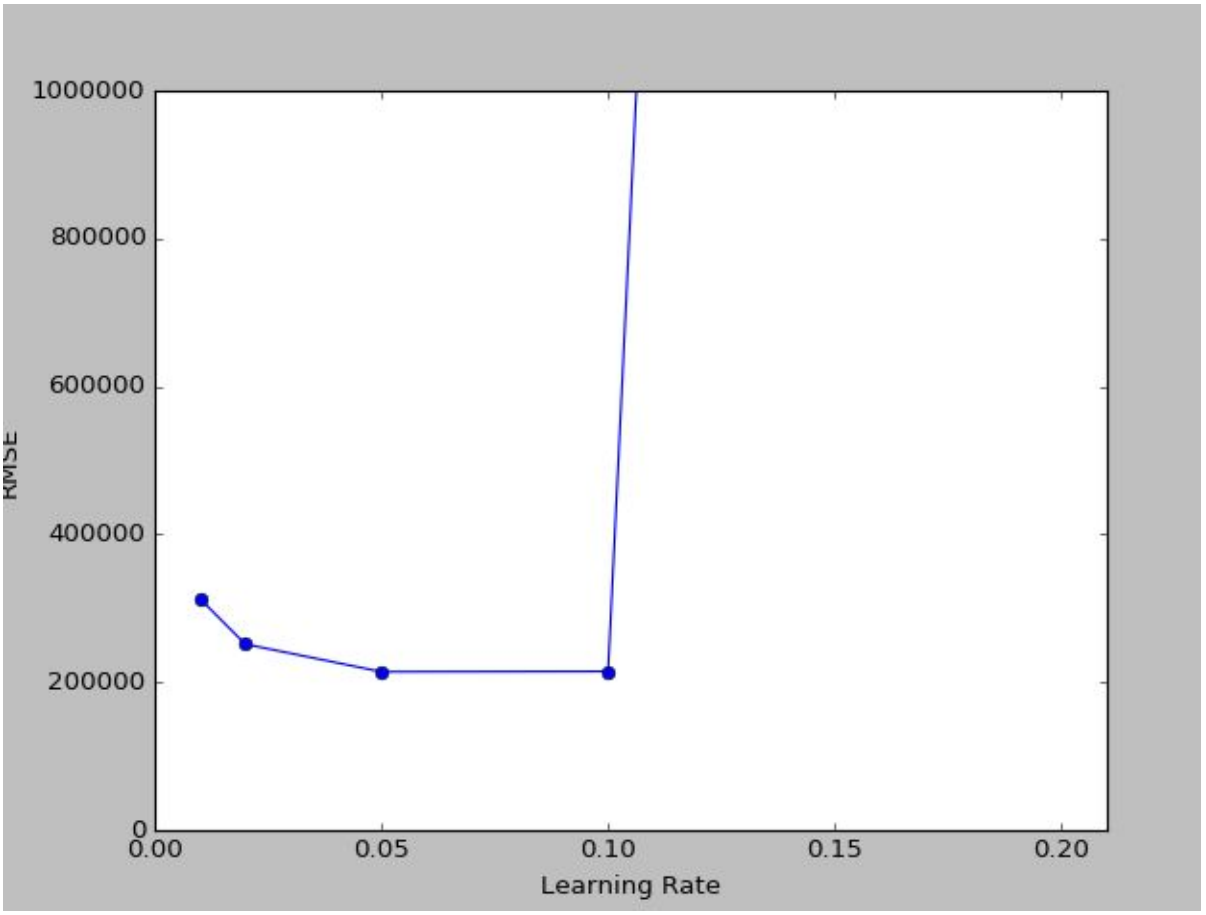
RMSE finally obtained is 214146.781654

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4] + theta[5] * a * a + theta[6] * b * b + theta[7] * c * c + theta[8] * d * d + theta[9] * a * b + theta[10] * a * c + theta[11] * a * d + theta[12] * b * c + theta[13] * b * d + theta[14] * c * d + theta[15] * a * a * a + theta[16] * b * b * b + theta[17] * c * c * c + theta[18] * d * d * d + theta[19] * a * a * b + theta[20] * a * a * c + theta[21] * a * a * d + theta[22] * b * b * a + theta[23] * b * b * c + theta[24] * b * b * d + theta[25] * c * c * a + theta[26] * c * c * b + theta[27] * c * c * d + theta[28] * d * d * a + theta[29] * d * d * b + theta[30] * d * d * c + theta[31] * a * b * c + + theta[32] * a * b * d + + theta[33] * a * c * d + theta[34] * b * c * d

theta[] = [6.25222329e+03 -4.10255069e+03 1.99144061e+03 2.11204257e+04
5.01982645e+05 -6.75295021e+02 -9.60583341e+03 -4.38367369e+03
4.23753351e+04 8.22721391e+03 1.90711602e+03 5.22021600e+03
6.46553964e+03 9.87180374e+03 1.59480853e+04 5.40561838e+03
-1.44697826e+04 -4.75908615e+03 5.93332699e+04 -2.53572391e+03
-4.87266329e+03 -9.10386034e+03 8.36298014e+03 2.65365528e+03

1.12029507e+03 -5.06158657e+03 3.65982645e+03 -4.53869537e+02
-1.44208741e+03 3.13355790e+04 3.00616168e+04 1.64292600e+03
1.64200071e+03 -3.54844283e+03 1.48204028e+04]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0
d = (bathrooms - 2.11475732198) / 8.0



Conclusion:
In the above case, our RMSE value decreases with increase in degree of hypothesis function. Hence in the above case, I shall prefer to use the cubic hypothese function.
We may see that as we increase the no. of parameters in theta, and increase the degree of our hypothesis function, we can get better results. But we shall also keep in mind the problem of overfitting whole doing so.

D. i) mean absolute error

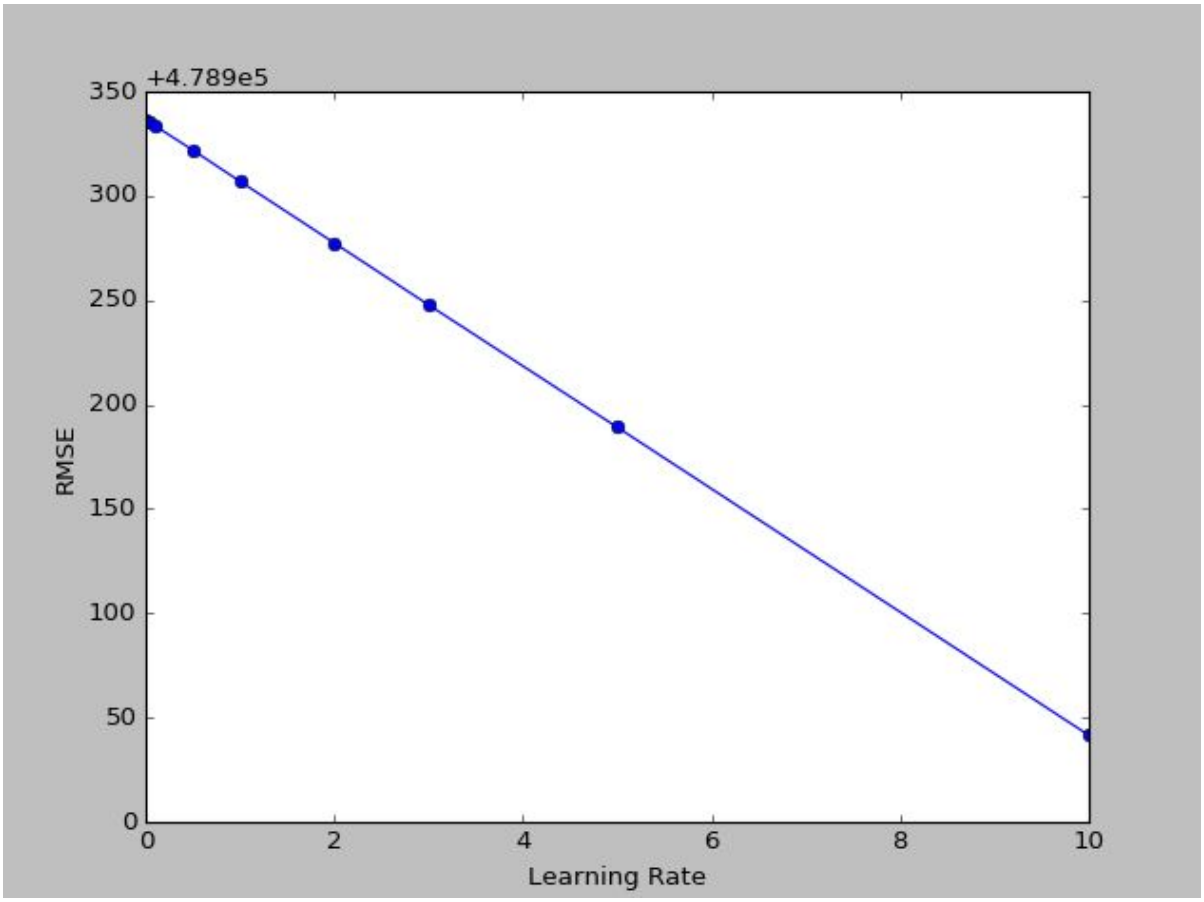
When learning rate = 0.05

RMSE finally obtained is 479236.091159

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4]

theta[] = [0.46119203 0.86593926 0.0390826   0.98914403 1.11402277]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0
d = (bathrooms - 2.11475732198) / 8.0



D. ii) mean squared error

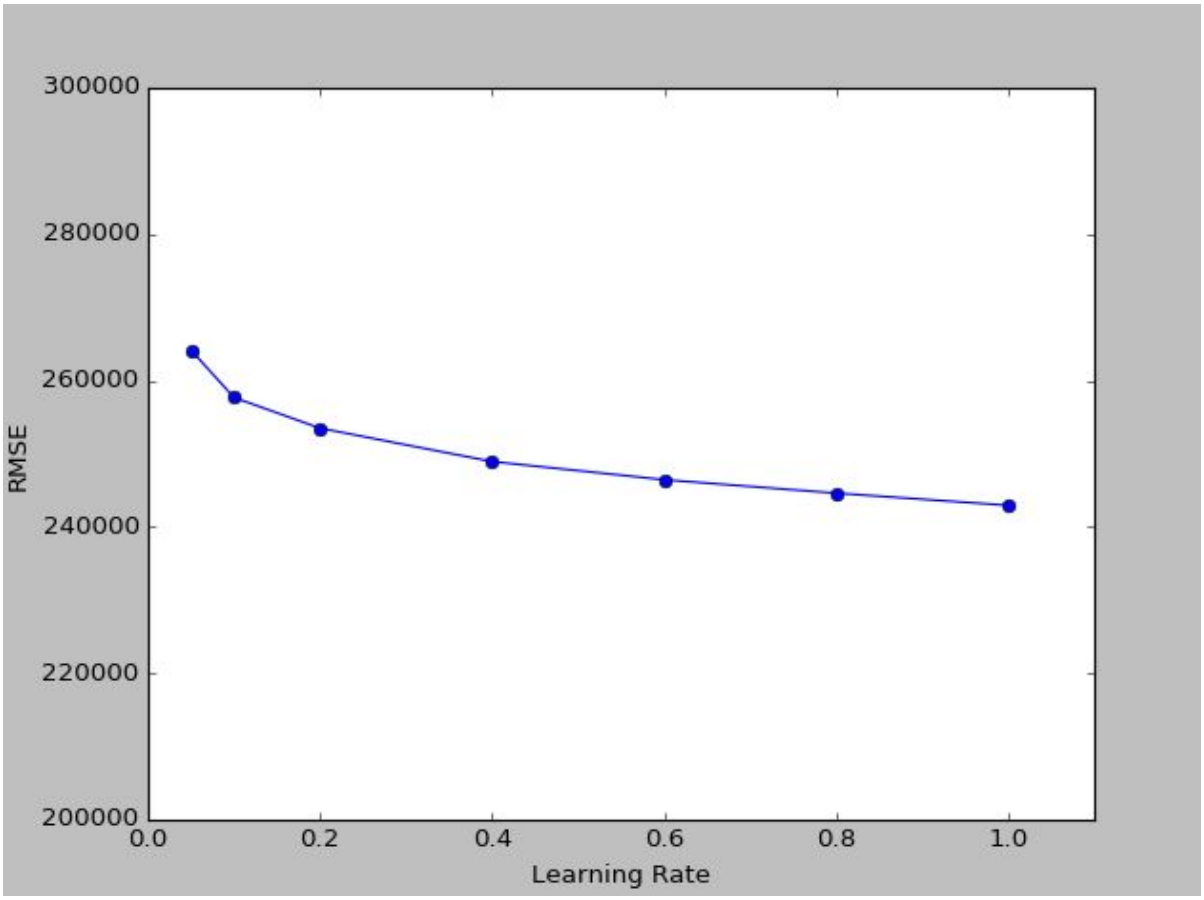
When learning rate = 0.05

RMSE finally obtained is 264166.869672

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4]

theta[] = [2297.72820361   38195.47922716   7630.56523892   41309.1981961
  492937.46103593]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0
d = (bathrooms - 2.11475732198) / 8.0



D. iii) mean cubic error

When learning rate = 0.01

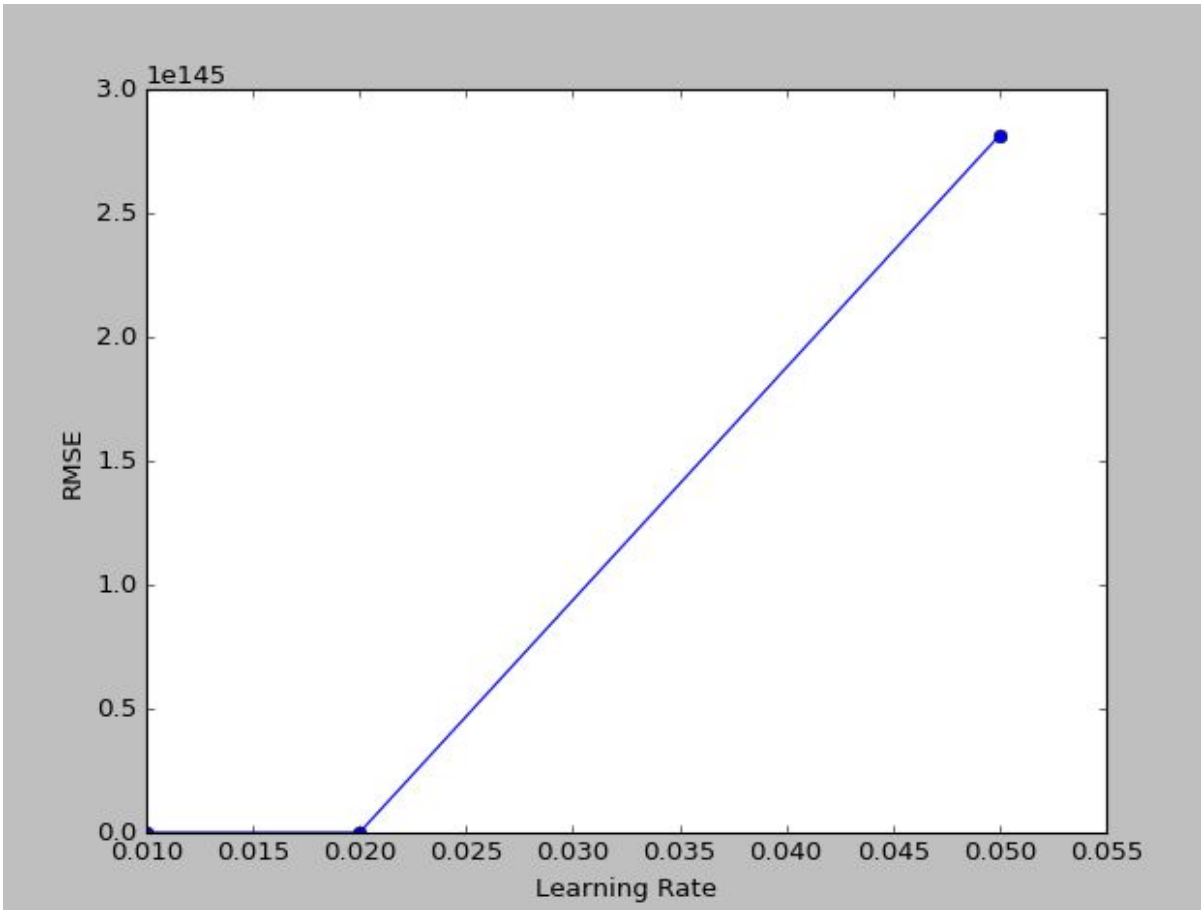
RMSE finally obtained is 6.05210284878e+123

h_theta(X) = theta[0] * a + theta[1] * b + theta[2] * c + theta[3] * d + theta[4]

theta[] = [-2.97955313e+120 2.30022038e+122 1.78729960e+120 6.11695667e+121
 -8.58285607e+123]

where:
a = (sqft - 15106.9675658) / 1650839.0
b = (floors - 1.49430898071) / 2.5
c = (bedrooms - 3.3708416231) / 33.0

d = (bathrooms - 2.11475732198) / 8.0



Conclusion:

For the above cases, we may see that training the data using the mean cubic error is very difficult, as it enlarges chances of data getting overflowed. Another problem with it is that, we can't have a high learning rate, hence learning will occur very slowly.

For the case of mean absolute error, we can clearly see learning rate need to be increased to significantly high value to get decent value of error. Yet, it still cannot be compared with the mean square error cost function's trained model.

Hence, I shall prefer to use mean squared error function as my cost function.