Assignment 1

Hrithik Maheshwari
2017CS10335

February 12, 2020

1) Linear Regression

In this part of the assignment we simply loaded the Data and implemented Batch Gradient Descent. The Loaded data was first normalized by assuming it to be a gaussian distribution and the Mean,Variance of resulting data to be 0,1 and respectively.
Although the Mean and variance of the given data came out to be 8.062 and 2.969156

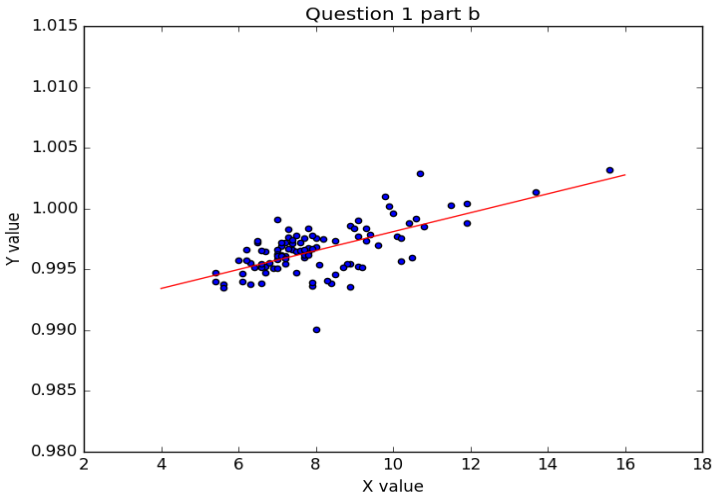
a) Implementing batch gradient Descent

Learning rate : 0.1
Thetha0 : 0.99659363
Thetha1 : 0.00134016
The termination condition in this learnning was when error term is less than 0.0000001 or the change in error is less than 0.0000000001.
The snippets of used Normalize Function,error function and termination condition are

```
25  def error(x,y,thetha):
26      dif = y - np.dot(x,thetha)
27      return (np.sum(dif**2)/(2*x.shape[0]))
28
29  def normalize(x):
30      mean = np.sum(x)/x.shape[0]
31      mean_vector = x - mean
32      variance = np.sum(mean_vector**2)/x.shape[0]
33      x = (x-mean)/math.sqrt(variance)
34      return x,mean,variance
35
63  while(error(x,y,new_thetha) > 0.0000001 and check):
70      check = abs(error(x,y,new_thetha) - error(x,y,thetha)) > 0.0000000001
```

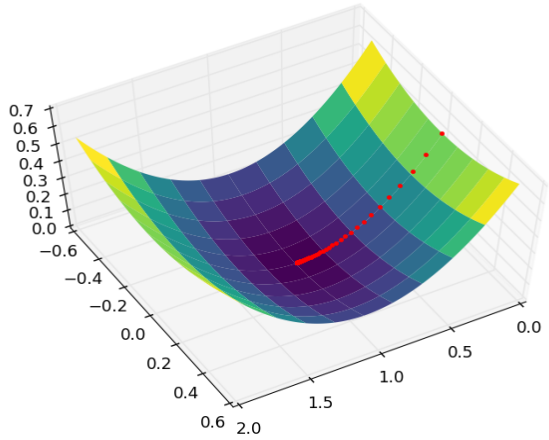
Plotting the Graph

Thetha0 : 0.99659363
Thetha1 : 0.00134016
Note But these thetha0 and thetha1 are calculated for the normalized data. The unnormalized thetha0 and thetha1 are given below and the line in the below graph is plotted according to the Unnormalized from
Not Normalized thetha0 = thetha0 - thetha1*(mean/Deviation) = 9.90323404*exp(-1)
Not Normalized thetha1 = thetha1/standard Deviation = 7.77750465*exp(-4)



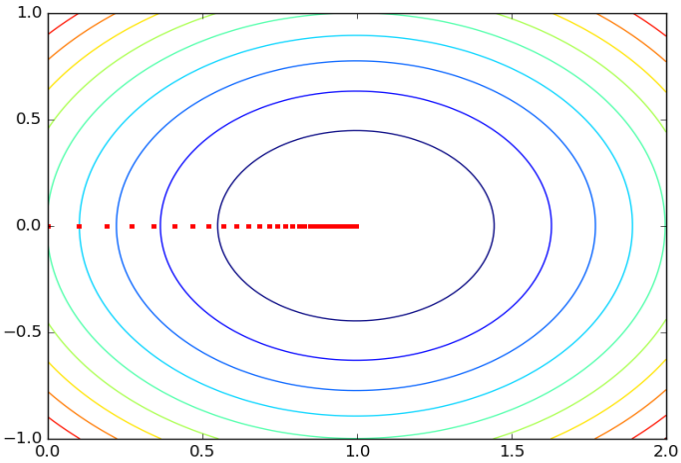
3 Dimensional Mesh

The Animation of 3d mesh can not be put into the pdf but the final state of the 3D mesh looked like the figure given below.



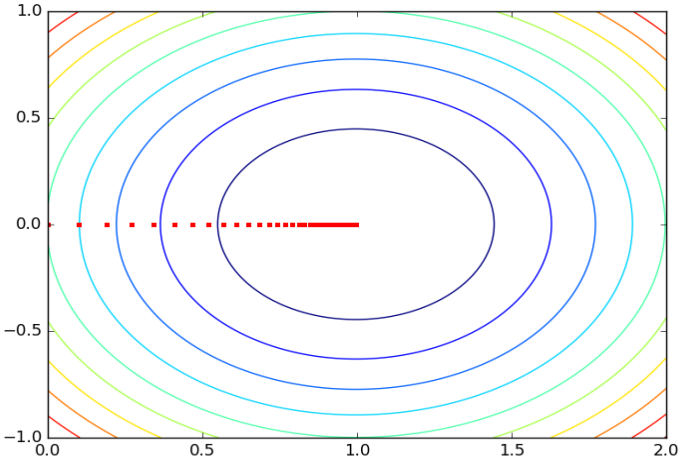
Contours Of the Error Function

Again as it was a animation which can be visualized when running the code but the end result looked like this



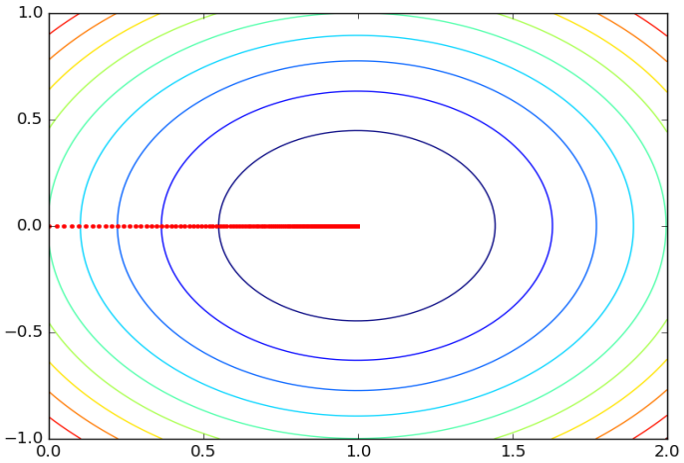
Contours of the Error function with diffrent learning rates

The contour with learning rate 0.1 was the fastest to converge in around 100 iterations

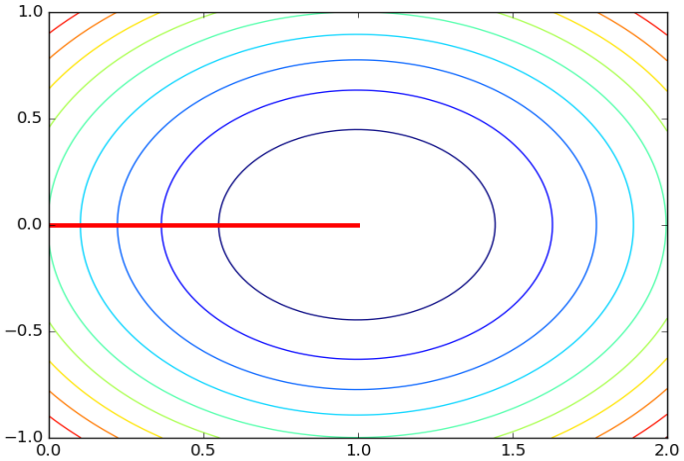


The contour with learning rate 0.025 was the fast to converge but not as fast as n=0.1 in

around 400 iterations



The contour with learning rate 0.001 was the slowest to converge in around 8000 iterations



Stochastic Gradient Descent

0.1 Part a

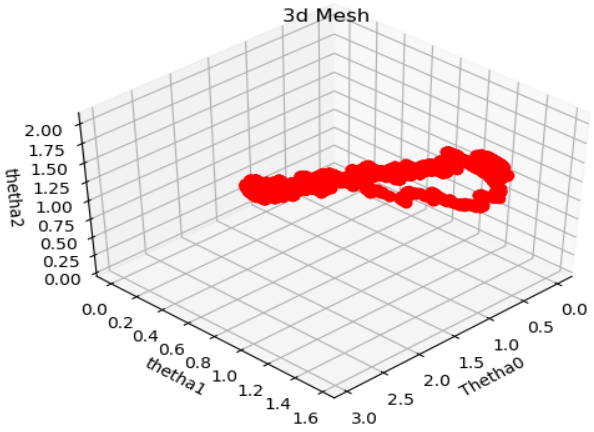
In the given part we used the normal function of python to generate the X1 and X2 and then with the help of given thetha we calculated the X.Thetha+ errorTerm. (Error term is calculated diffrently)

0.2 Part b, c and d

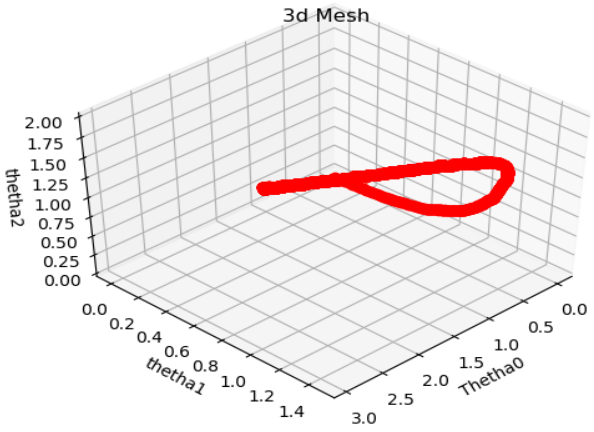
Error on Test data from original thetha is 0.98294

- **Batch = 1**

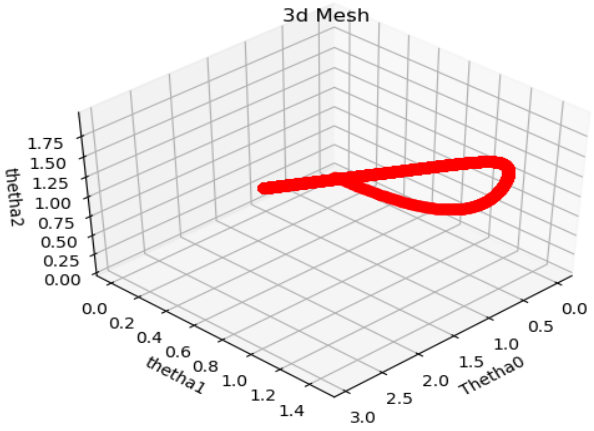
- **Convergence Criteria :** When the difference between the average of error from last 5000 batches and last last 5000 batches is less than 0.000001
- **Convergence Speed : It took some time but the number of iternations were very less**
- **Reported Thetha :** Thetha0:3.026 Thetha1:1.035 Thetha2:1.938
- **Error on Test Data:** 1.2
- **3D plot of thetha:**



- **Batch = 100**
  - **Convergence Criteria :** When the difference between the average of error from last 500 batches and last last 500 batches is less than 0.000001
  - **Convergence Speed :** It converged very quickly in less than 3 iterations of whole training data
  - **Reported Thetha :** Thetha0:2.9786 Thetha1:0.9983 Thetha2:0.98294
  - **Error on Test Data:** 0.98404
  - **3D plot of thetha:**

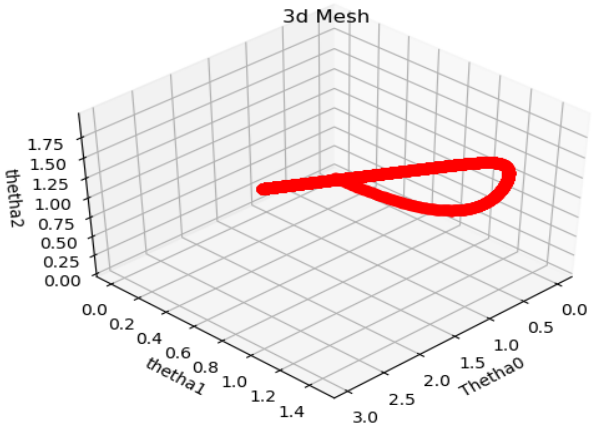


- **Batch = 10000**
  - **Convergence Criteria :** When the difference between the average of error from last 5 batches and last last 5 batches is less than 0.000001
  - **Convergence Speed :** It took around 150 iterations on the whole data to get converge
  - **Reported Thetha :** Thetha0: 2.9391 Thetha1: 1.012 Thetha2: 1.994
  - **Error on Test Data:** 0.99348
  - **3D plot of thetha:**



- **Batch = 1000000**
  - **Convergence Criteria :** When the change in error data is less than 0.0000001
  - **Convergence Speed :** It converged very quickly in less than 3 iterations of whole training data

- **Reported Thetha :** Thetha0: 2.9658 Thetha1: 1.007 Thetha2: 1.9977
- **Error on Test Data: 0.9866**
- **3D plot of thetha:**



Logistic Regression

Part a

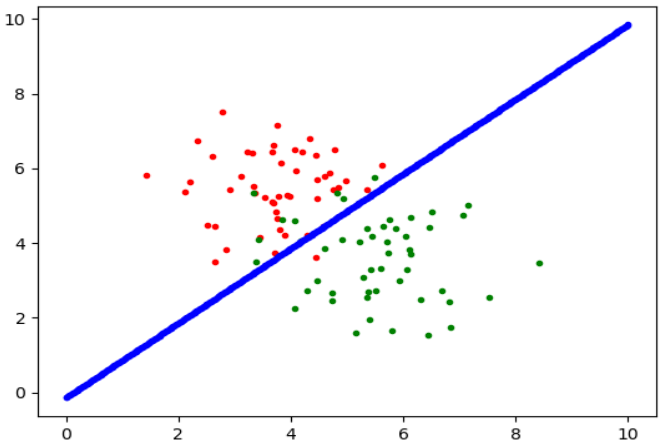
The trained thetha are coming out to be following after normanilizing of the data

- **Thetha0 : 0.4125**
- **Thetha1 : 2.588**
- **Thetha2 : -2.725**

Part b

The given graph is plotted with X1 on the x axis and X2 on the y axis and the green point representing where Y is 1 and Red denoting where Y is 0. The grsph is plotted by converting the trained Thetha according to the mean and variance of data

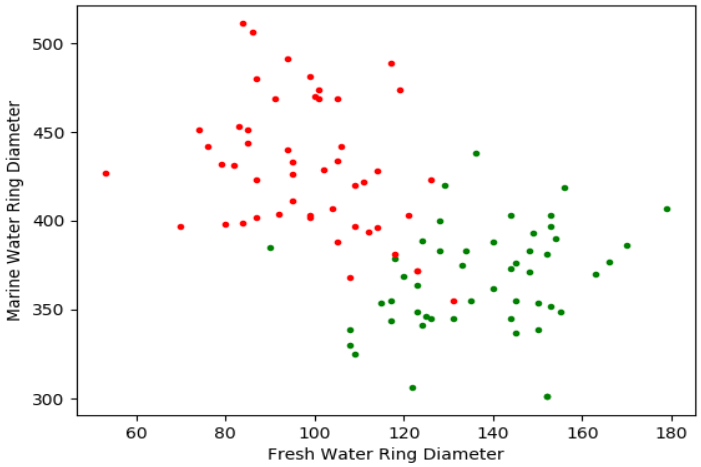
- **Thetha0 : 0.2232**
- **Thetha1 : 1.9626**
- **Thetha2 : -1.964**

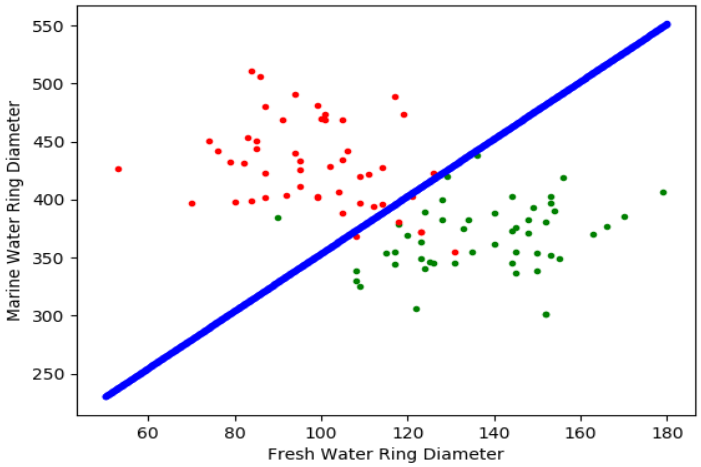


GDA

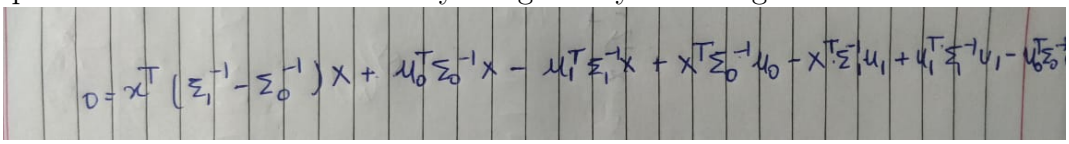
The following constant value are formed:
U0 - [0.75043876 -0.68217391]
U1 - [-0.75935336 0.68826087]
Sigma
[0.42907761 -0.02246447
[-0.02246447 0.53083648]]
Sigma0
[0.47696776 0.10988238
[0.10988238 0.41370302]]
Sigma1
[0.38118746 -0.15481131

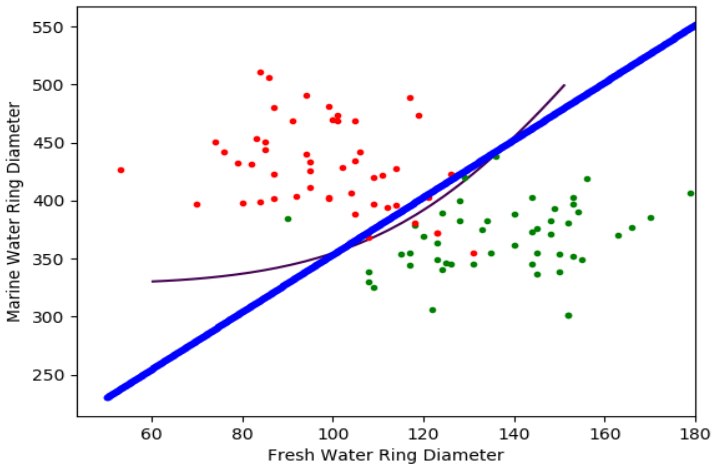
[-0.15481131 0.64796994]]
Slope of Normalized Line - 1.390828398749756
Intercept of Normalized Line - -0.04639031941782183
Slope of Original Line - 2.416898
Intercept of Original Line - 106.67832





It is seen that quadratic function tries to cover more and more points accurately. The quadratic function and boundary are given by two images below




Marine Water Ring Diameter
Fresh Water Ring Diameter
550
500
450
400
350
300
250
60
80
100
120
140
160
180