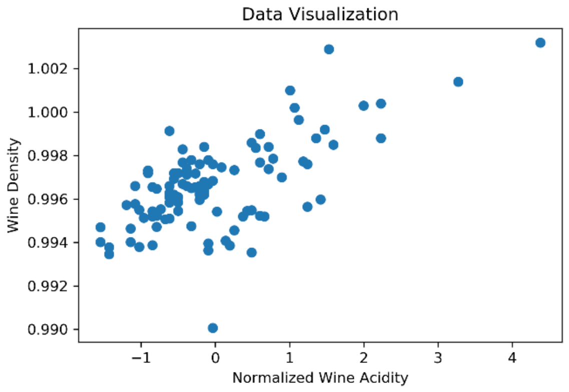
Q 1. Linear Regression

Data Visualization of Normalized Wine Data:



a) Implement Gradient Descent

Learning Rate: 0.2

Number of Iterations: 81

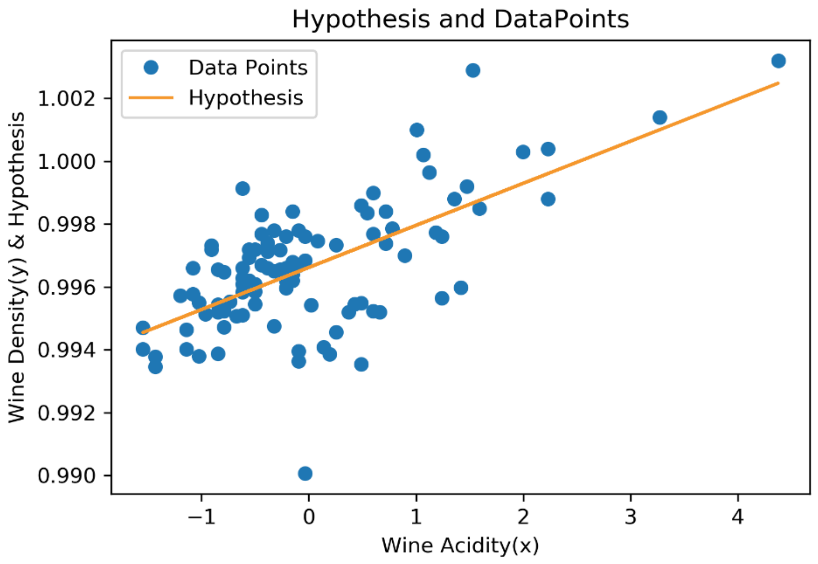
Final Parameters i.e. Theta values

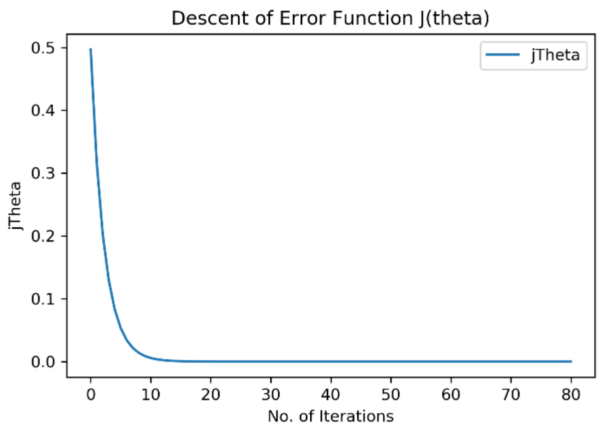
θ = [θ_0 = 0.99662009, **θ_1** = 0.0013402**]**

Stopping Criteria **abs(** $J(\theta^{t+1}) - J(\theta^t)$ **) < ε:** **ε = 1.1e-16**

b) Hypothesis Learned for Linear Regression Model through Gradient Descent

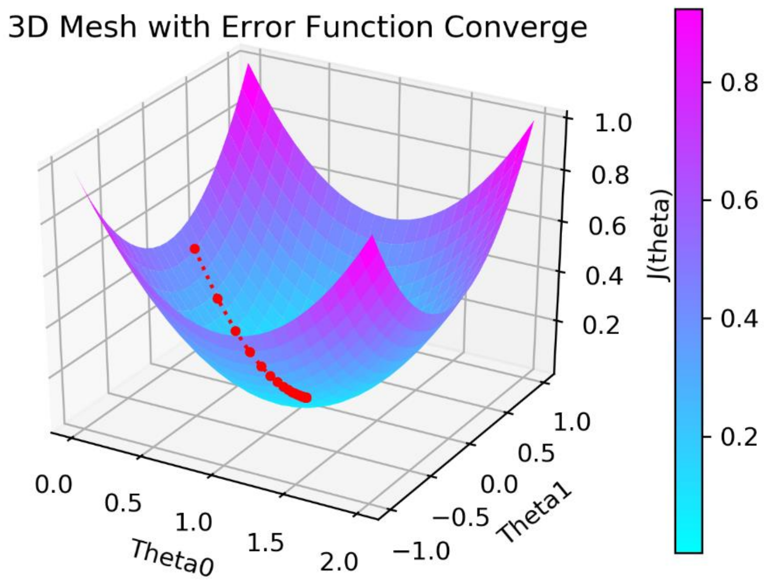
Learning Rate: 0.2





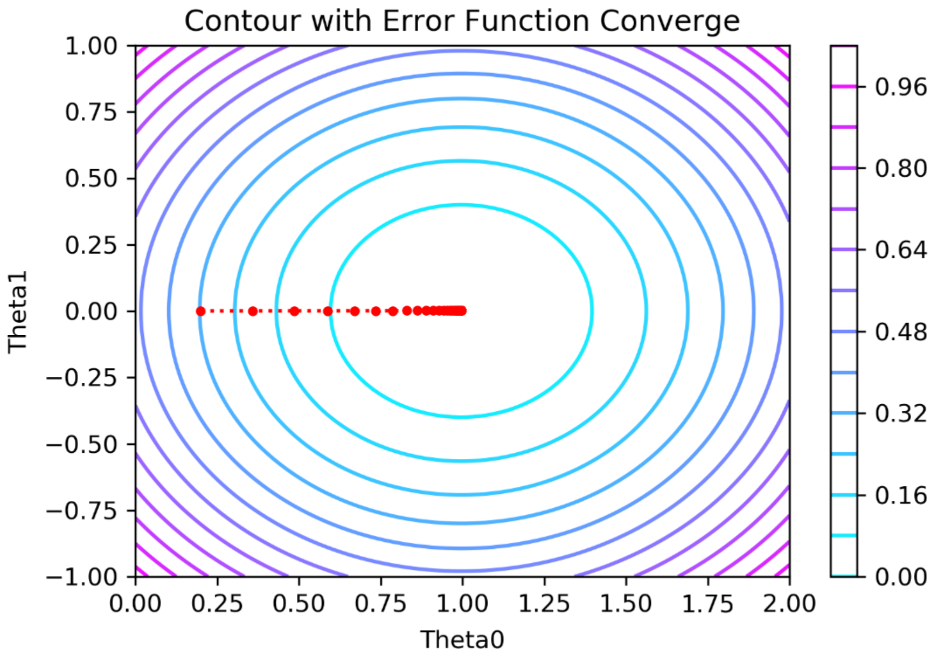
c) 3D Surface Plot

Learning Rate = 0.2



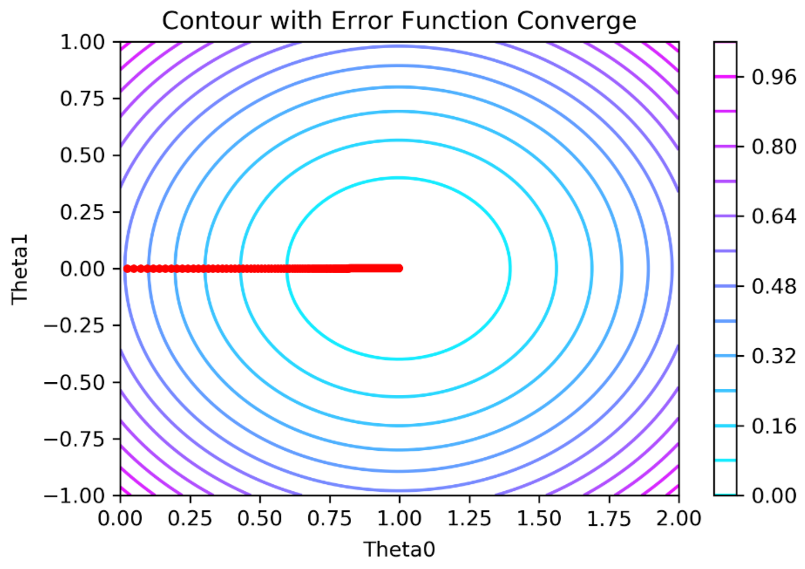
d) Contour Plot

Learning Rate = 0.2

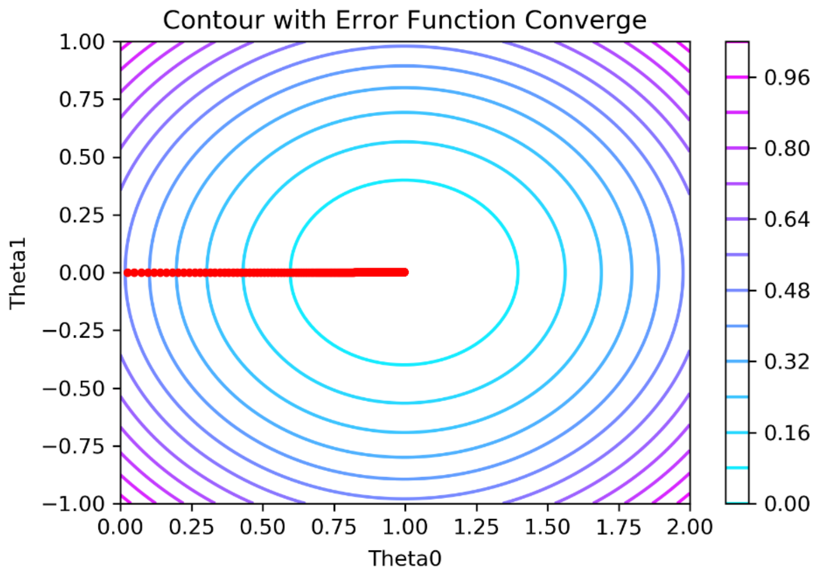


e) Contour plots for different Learning Rates

a. Learning Rate **$\eta = 0.001$**
   Number of Iterations: 10000
   Final Parameters i.e. Theta values
   θ = [$\boldsymbol{\theta_0}$ = 0. 99657508, $\boldsymbol{\theta_1}$ = 0. 00134014]
   Stopping Criteria **abs($J(\theta^{t+1})$ - $J(\theta^{t})$)** < ε: ε = 8.1e-16



b. Learning Rate **$\eta = 0.025$**
   Number of Iterations: 564
   Final Parameters i.e. Theta values
   θ = [$\boldsymbol{\theta_0}$ = 0. 99661947, $\boldsymbol{\theta_1}$ = 0. 0013402]
   Stopping Criteria **abs($J(\theta^{t+1})$ - $J(\theta^{t})$)** < ε: ε = 1.1e-14



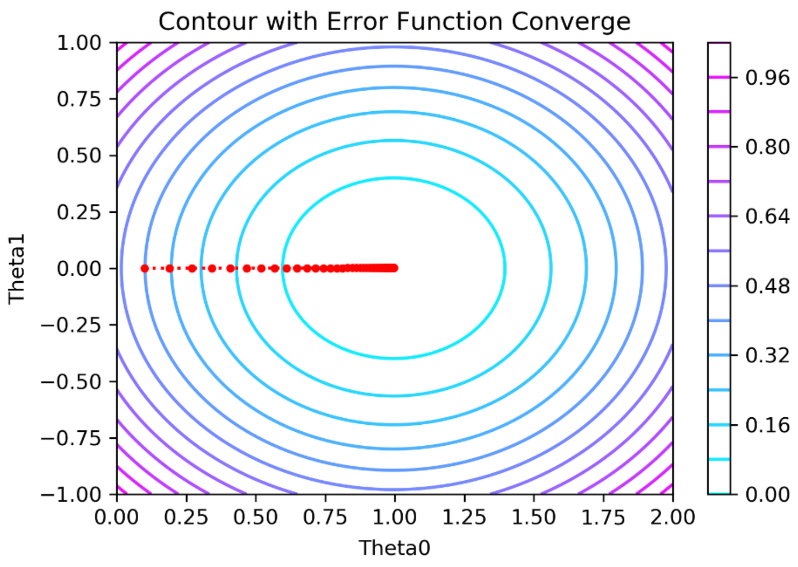
c. Learning Rate **$\eta = 0.1$**

Number of Iterations: 166

Final Parameters i.e. Theta values

θ = [$\boldsymbol{\theta_0}$ = 0. 99662007, $\boldsymbol{\theta_1}$ = 0. 0013402]

Stopping Criteria **abs($J(\theta^{t+1})$ - $J(\theta^{t})$)** < ε: **ε** = 1.1e- 16



Observations

- As we can see for increasing values of Learning Rate **η,** the number of steps taken to converge decreases.

  η = 0.001 Number of Iterations = 10000

  η = 0.025 Number of Iterations = 564

  η = 0.1 Number of Iterations = 166

  η = 0.2 (Chosen by me) Number of Iterations = 81

- If we look at the contour plots, the larger the learning rate the larger steps taken by Gradient Descent to converge to the optimum values of **θ.**
- But if we increase learning rate further, the Gradient Descent Algorithm will diverge after certain value of learning rate.
- If we further increase the learning rate, then we will observe that algorithm diverges for the values of learning rate greater than 1.9 it will diverge.

Q 2. Sampling and Stochastic Gradient Descent

a) **Sampling of Data**

b) **SGD for different batch size keeping Learning Rate η = 0.001**

a. Batch Size = 1

Number of Epoch (pass over data): **0.014811**

Run Time = **0.23Sec**

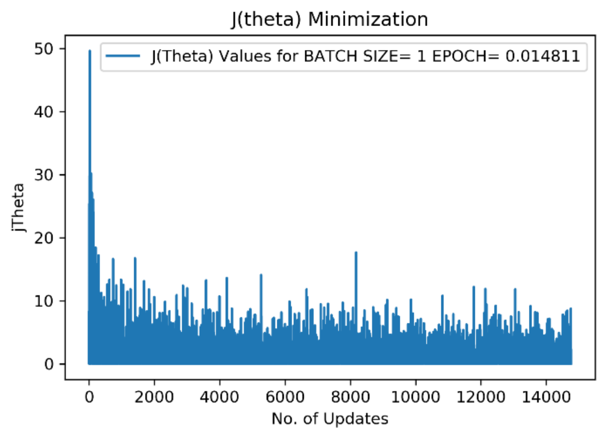
Final Parameters i.e. Theta values

θ = [**θ_0** = 2.9318233774933407, **θ_1** = 1.0108940147060694, **θ_2** = 1.9906446823652668]

Stopping Criteria:

- **abs($J(\theta^{t+1}) - J(\theta^t)$)** < ε: **ε = 2.1e-05 OR**
- **Max Iterations = 20000 (20000 * 1 => 0.02 Pass of Data)**

Plot of J(θ) Values of Each Update



b. Batch Size = 100

Number of Epoch (pass over data): **2.0**

Run Time = **0.33Sec**

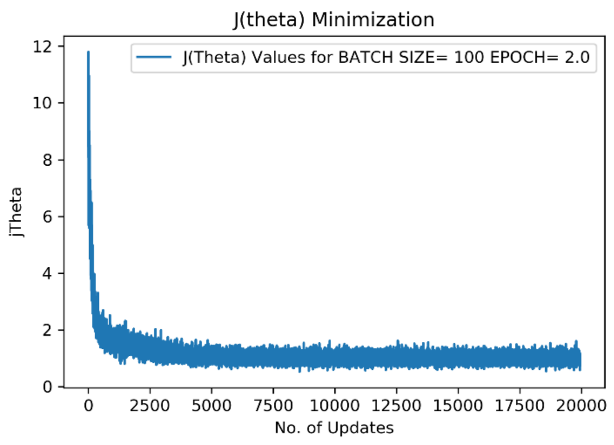
Final Parameters i.e. Theta values

θ = [**θ_0** = 2. 987854759450669, **θ_1** = 1. 0021688211937427, **θ_2** = 2. 0023862582053438]

Stopping Criteria:

- **abs($J(\theta^{t+1}) - J(\theta^t)$)** < ε: **ε = 1.1e-05 OR**
- **Max Iterations =** 20000 (20000 * 100 **=> 2 Pass of Data**)

Plot of J(θ) Values of Each Update



c. Batch Size = 10000

Number of Epoch (pass over data): **200.0**

Run Time = **2.62Sec**

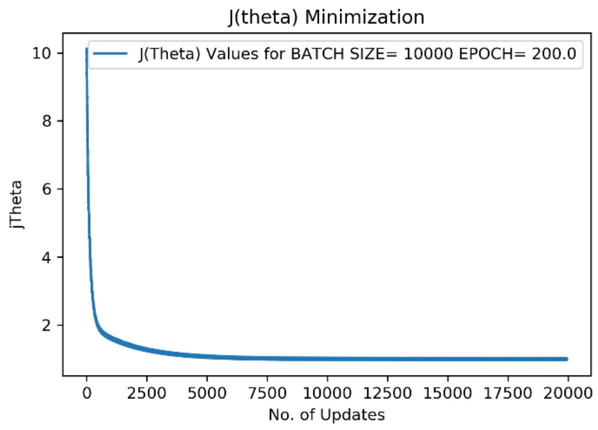
Final Parameters i.e. Theta values

θ = [**θ_0** = 2. 987813051086349, **θ_1** = 1. 0030629356527994, **θ_2** = 1. 9992223496832995]

Stopping Criteria:

- **abs($J(\theta^{t+1})$ - $J(\theta^t)$)** < ε: **ε** = 1.1e-06 **OR**
- **Max Iterations =** 20000 (20000 * 10000 => **200 Pass of Data**)

Plot of J(θ) Values of Each Update



d. Batch Size = 1000000

Number of Epoch (pass over data): **20000.0**

Run Time = **269.78Sec**

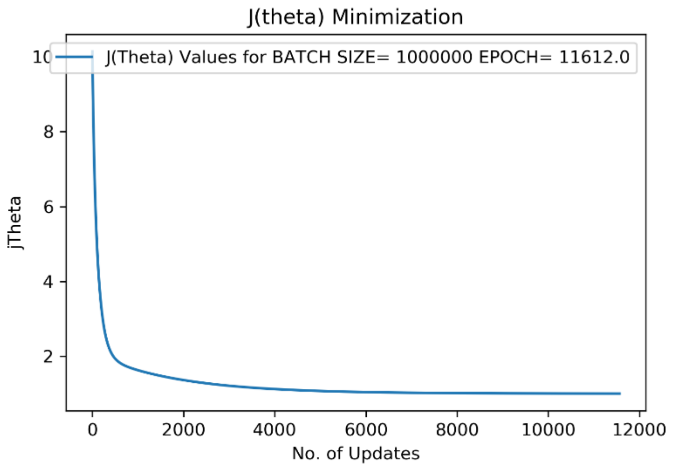
Final Parameters i.e. Theta values

θ = [**θ_0** = 2. 8809504782350537, **θ_1** = 1. 0264428327630841, **θ_2** = 1. 9912329531896353]

Stopping Criteria:

- **abs($J(\theta^{t+1})$ - $J(\theta^t)$)** < ε: **ε** = 1.1e-06 **OR**
- **Max Iterations =** 20000 (20000 * 10000 => **20000 Pass of Data**)

Plot of J(θ) Values of Each Update



Note: The **J(θ)** descent has **become smoother with increase** in Batch Size. For Batch Size = 1, it is very irregular for each update. But as we increase the values of batch size from 1 to 1000000 this graph has become smoother and for Batch Size = 1000000 it is same as that of Batch gradient descent.

c)

Do different algorithms in the part above (for varying values of r) converge to the same parameter values?

Ans)

- In all the four cases, for Learning Rate = 0.001, theta converge to the values which are very close to each other, but not exactly same as we have decided different stopping criteria for all the four cases. And these values of Theta Parameter is very close to original values of Theta through which we have sampled the data.
- If we run the algorithm long enough, they will reach to almost the original values of Theta.

How much different are these from the parameters of the original hypothesis from which the data was generated?

Ans)

- They are very close to original hypothesis. You can see and compare the values below for each batch size with original hypothesis Theta values. Therefore, we can verify that SGD gives good enough results when we have large dataset in very less time.
  Original Hypothesis:
  θ = [θ_0 = 3, θ_1 = 1, θ_2 = 2]

  Batch Size = 1
  θ = [θ_0 = 2.9318233774933407, θ_1 = 1.0108940147060694, θ_2 = 1.9906446823652668]

  Batch Size = 100
  θ = [θ_0 = 2. 987854759450669, θ_1 = 1. 0021688211937427, θ_2 = 2. 0023862582053438]

  Batch Size = 10000
  θ = [θ_0 = 2. 987813051086349, θ_1 = 1. 0030629356527994, θ_2 = 1. 9992223496832995]

  Batch Size = 1000000
  θ = [θ_0 = 2. 8809504782350537, θ_1 = 1. 0264428327630841, θ_2 = 1. 9912329531896353]

Comment on the relative speed of convergence and also on number of iterations in each case.

Ans)

- Relative speed of convergence can be decided by Run Time in each case and also by number of iterations i.e. **epoch / pass over whole data**. Let's see the data in each case first then well discuss the results
- **Batch Size = 1 |** Epoch/Iterations: **0.014811** Run Time = **0.23Sec**
- **Batch Size = 100 |** Epoch/Iterations: **2.0** Run Time = **0.33Sec**
- **Batch Size = 10000 |** Epoch/Iterations: **200.0** Run Time = **2.62Sec**
- **Batch Size = 1000000 |** Epoch/Iterations: **20000.0** Run Time = **269.78Sec**
  As we can see that with increasing batch size number of iterations and runtime increases drastically, as the rows of data i.e. Batch size per update increases.

It takes O(b*n) time per update of theta, where b is batch size and n is number of features.

Also, we can see that with increasing the batch size from 1 to 1000000 the algorithm tends to become Batch Gradient Descent from Stochastic Gradient Descent and time complexity approaches O(m*n) where m is number of examples.

Report the error on a new test data of 10,000 samples provided in the file named q2test.csv.

Mean Square Error with Original Hypothesis: 1.96589384

Mean Square Error (MSE) for following cases:

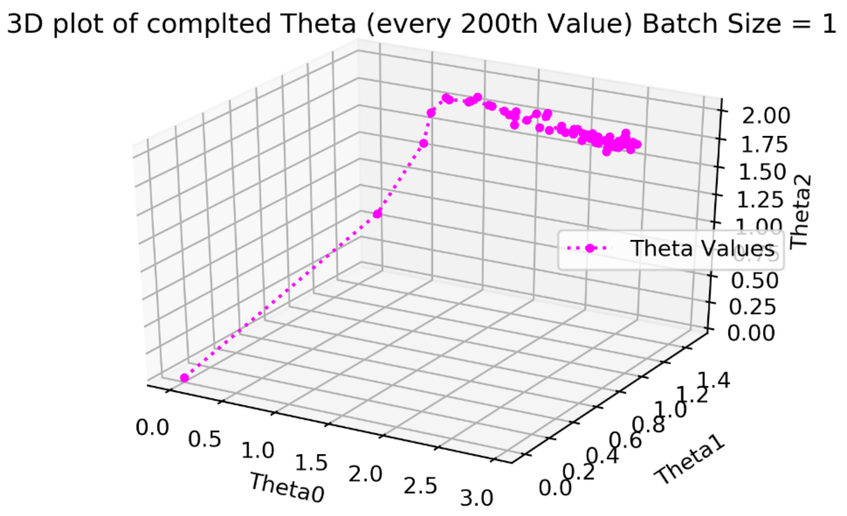
- **Batch Size = 1 |** MSE: **1.99078244** Learned – Original Error = **0.0248886**
- **Batch Size = 100 |** MSE: **1. 96630546** Learned – Original Error = **0. 00041162**
- **Batch Size = 10000 |** MSE: **1. 96693886** Learned – Original Error = **0. 00104501**
- **Batch Size = 1000000 |** MSE: **2.05055261** Learned – Original Error = **0. 08465877**

Comments:

- The difference of error is decreased from batch size 1 to 10000. As the algorithm learned better in every increased batch size. But it increased for 1000000 case because it was taking a lot of time to converge to the close to the original values. Therefore, we had to stop it after 20000 epochs.
- If we need to converge it faster than we will have to increase the learning rate. The error value is so big because as per my observation the test data is not from the same mean and variance as that of sampled data.

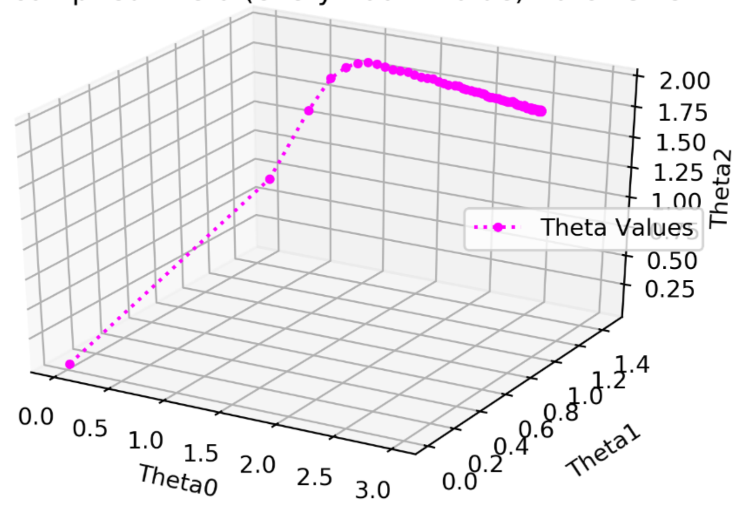
d) plot the movement of θ on a 3D space.

- **Batch Size = 1**



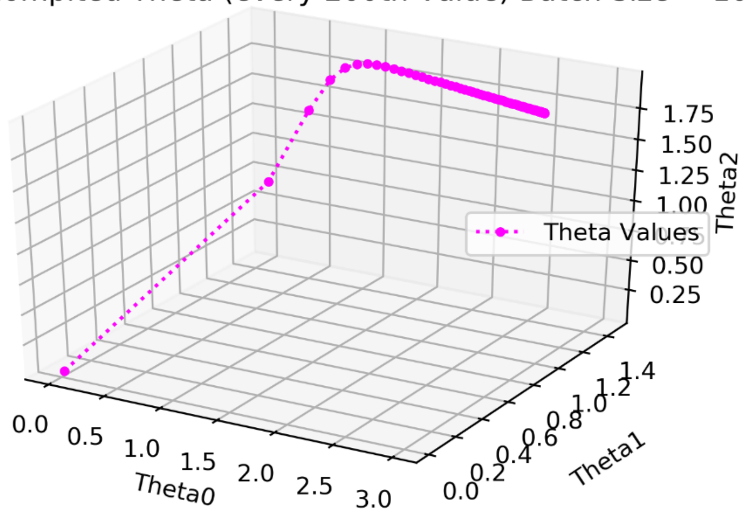
- **Batch Size = 100**

3D plot of complted Theta (every 200th Value) Batch Size = 100

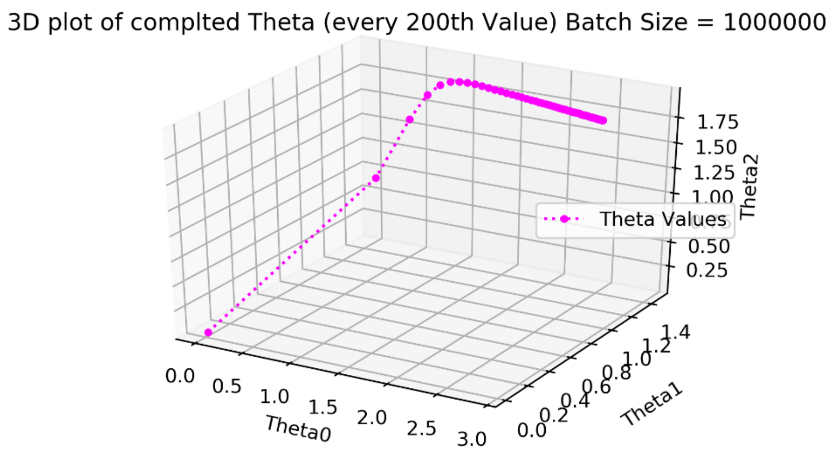


- **Batch Size = 10000**

3D plot of complted Theta (every 200th Value) Batch Size = 10000



- **Batch Size = 1000000**



Comments:

- For Batch Size = 1 the values of theta are not regular it is a zig-zag graph towards to convergence point.
- But it will become smoother with every increase in batch size.
- First few updates take larger step and then the steps become very small as we approach the convergence point.
- This makes intuitive sense as the gradient descent makes linear steps so change with every update becomes very small as we approach the convergence points.

Q 3. Logistic Regression & Newton's Method

a) Implement Newton's Method of Optimization

- Number of Iterations to Converge = **9**
- Final Set of Parameters **θ:**

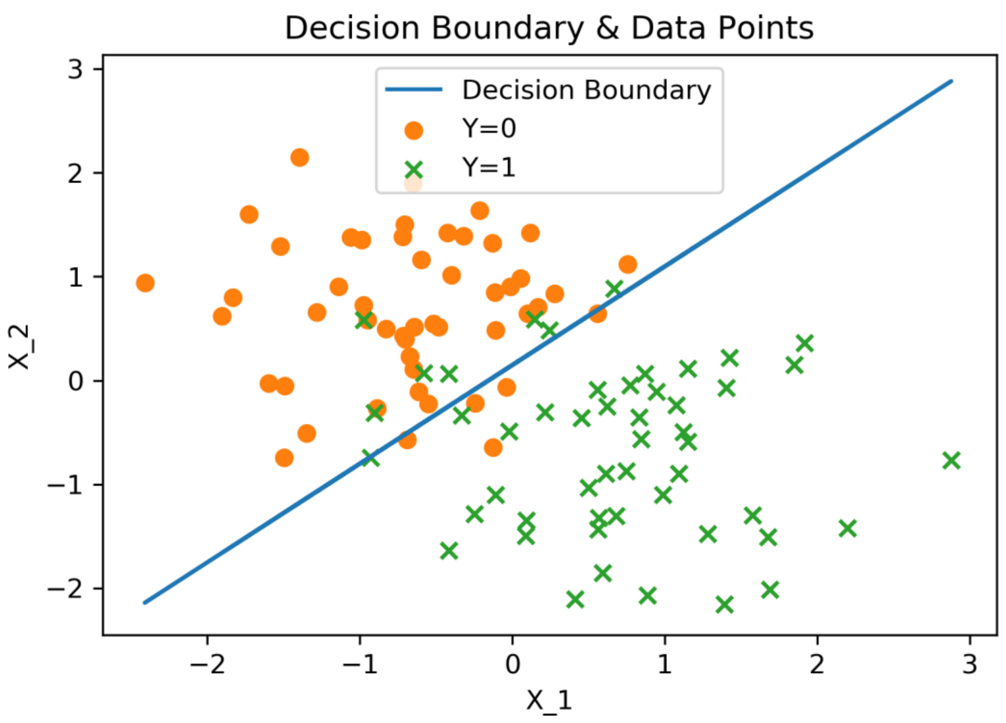
  θ_0 = 0. 40125316

  θ_1 = 2.5885477

  θ_2 = $-$ 2.72558849
- Stopping Criteria:

  abs($\nabla_\theta J(\theta)$) < ε where **ε = 1.1e-15**

b) Plot Decision Boundary



Q 4. Gaussian Discriminant Analysis

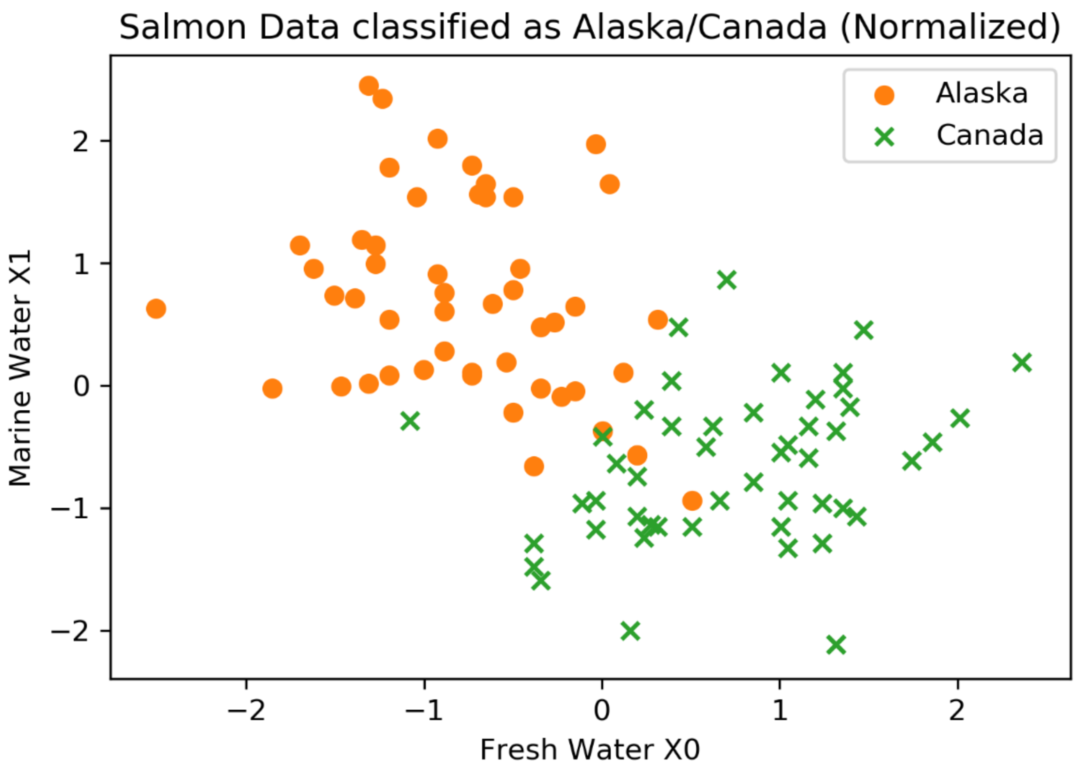
a) Calculate values of parameters $\mu_0, \mu_1, \Sigma_0, \Sigma_1$

a. μ_0 = [-0.75529433 0.68509431]

b. μ_1 = [0.75529433 -0.68509431]

c. $$\Sigma_0 = \Sigma_1 = \Sigma = \begin{bmatrix} 0.42953048 & -0.02247228 \\ -0.02247228 & 0.53064579 \end{bmatrix}$$

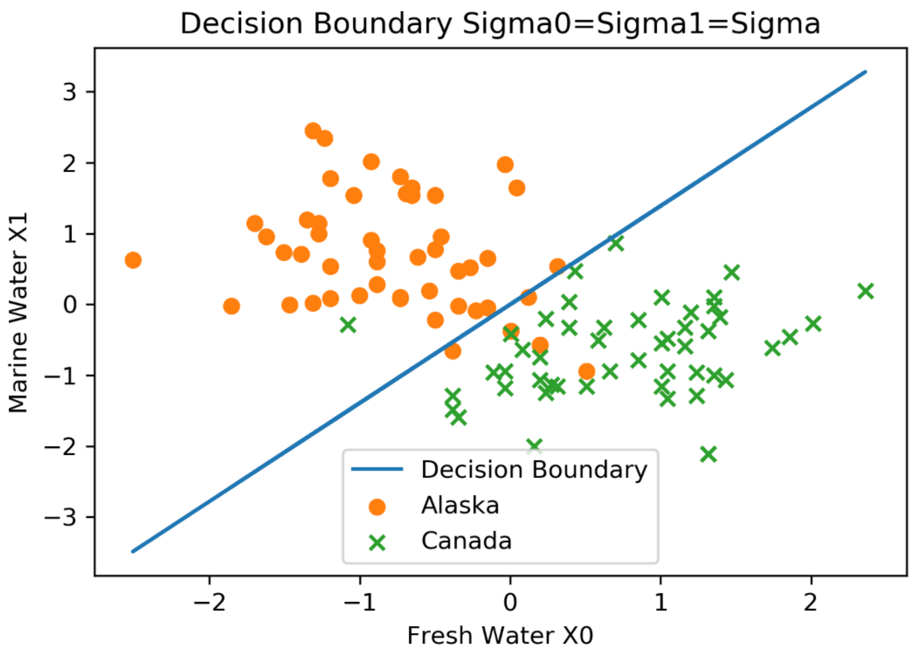
b) Plot training data with different markers for different labels



c) Equation of Decision Boundary for $\sum_0 = \sum_1 = \sum$

$$2 * (\boldsymbol{\mu_0}^T \Sigma^{-1} - \boldsymbol{\mu_1}^T \Sigma^{-1}) \mathbf{X} - (\boldsymbol{\mu_0}^T \Sigma^{-1} \boldsymbol{\mu_0} - \boldsymbol{\mu_1}^T \Sigma^{-1} \boldsymbol{\mu_1}) + 2 * \log(\frac{1-\varphi}{\varphi}) = 0$$

$*This\ is\ a\ Linear\ Equation\ in\ terms\ of\ X$



d) Calculate values of parameters $\mu_0, \mu_1, \sum_0, \sum_1$

a. μ_0 = [-0.75529433   0.68509431]

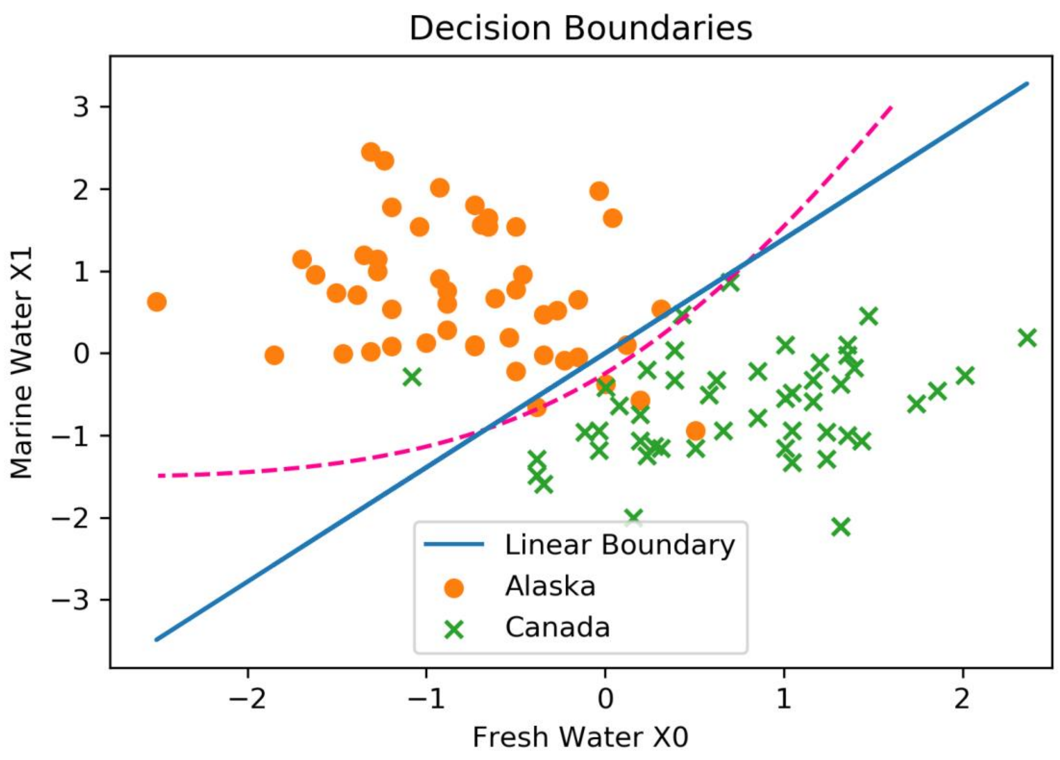
b. μ_1 = [0.75529433   -0.68509431]

c. $\sum_0 = \begin{bmatrix} 0.38158978 & -0.15486516 \\ -0.15486516 & 0.64773717 \end{bmatrix}$

d. $\sum_1 = \begin{bmatrix} 0.47747117 & -0.1099206 \\ -0.1099206 & 0.41355441 \end{bmatrix}$

e) Quadratic Decision Boundary

$$Z = \frac{1}{2}\left[(x - \mu_0)^T {\Sigma_0}^{-1}(x - \mu_0) \; - \; (x - \mu_1)^T {\Sigma_1}^{-1}(x - \mu_1)\right] - \log\left(\frac{1-\varphi}{\varphi}\right) - \frac{1}{2}\,\log\frac{|\Sigma_0|}{|\Sigma_1|} \; = \; 0$$



f) Observations:

- From the above plot of the two decision boundaries together shows that quadratic decision boundary classifies some of the points which were falsely classified when we consider the linear decision boundary.
  Therefore, quadratic decision boundary is slightly better in this case.
- our assumption that data belongs to gaussian, also does fairly well.