Q1: Linear Regression

a) Implemented batch gradient descent for optimizing $J(\theta)$. Tried the following loss functions at a learning rate of 0.019:

1. **Argmax(absolute(Gradient))**

2. **Change in theta**

3. **Absolute Error i.e absolute value of $J(\theta)$.**

    a. If threshold is set too low 1.0e-5, the algorithm (loss < threshold) even after several thousand iterations because loss became stagnant at **0.0001194789810984.**

4. **Change in Error $J(\theta)$**

    a. This performed best for even very low thresholds because as seen above the error becomes stagnant quite early and the algorithm converges.

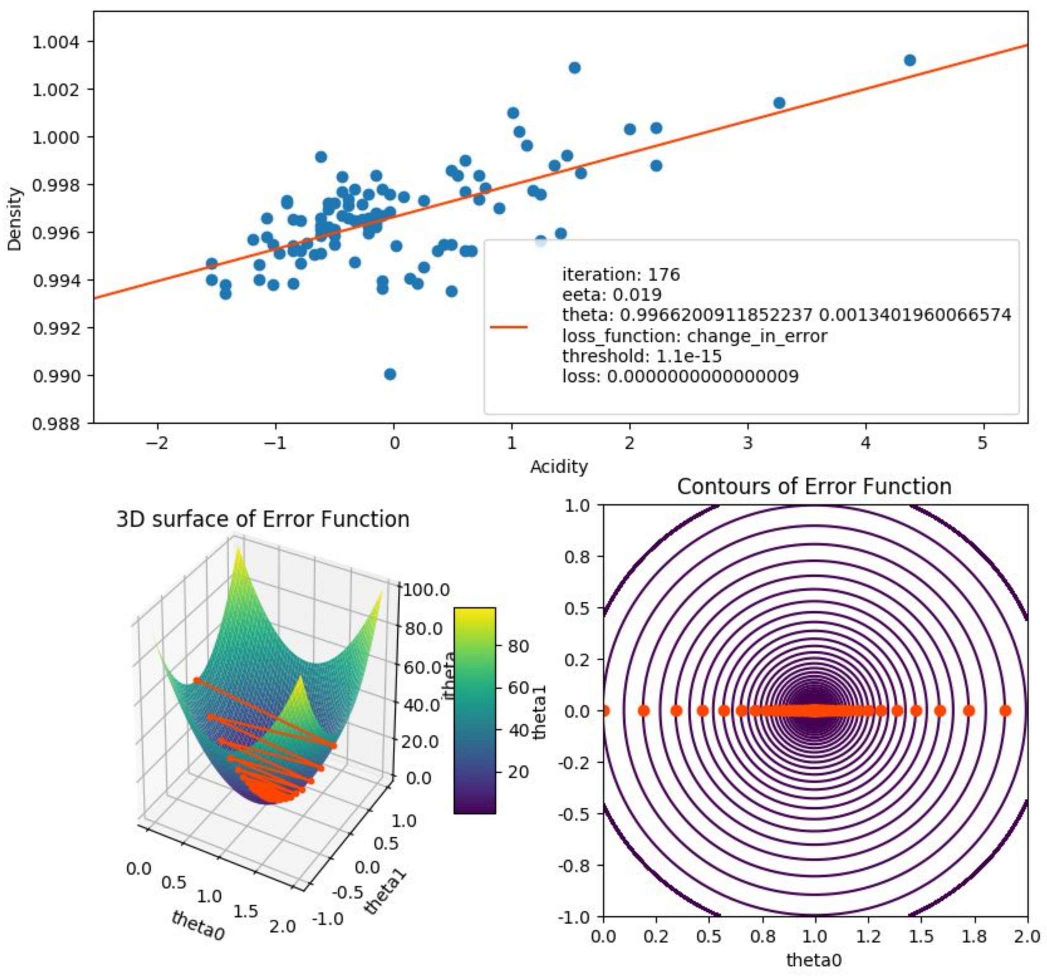
    b. The loss became less than threshold of 1.0e-15 just after 176 iterations.

The final values of θ using Change in Error $J(\theta)$ as the loss function is:

Number of Iteration: 176
Learning rate: 0.019
θ : 0.9966200911852237, 0.0013401960066574
loss_function: change_in_error
threshold: 1.1e-15
loss: 0.0000000000000009
Stopping criteria: loss < threshold

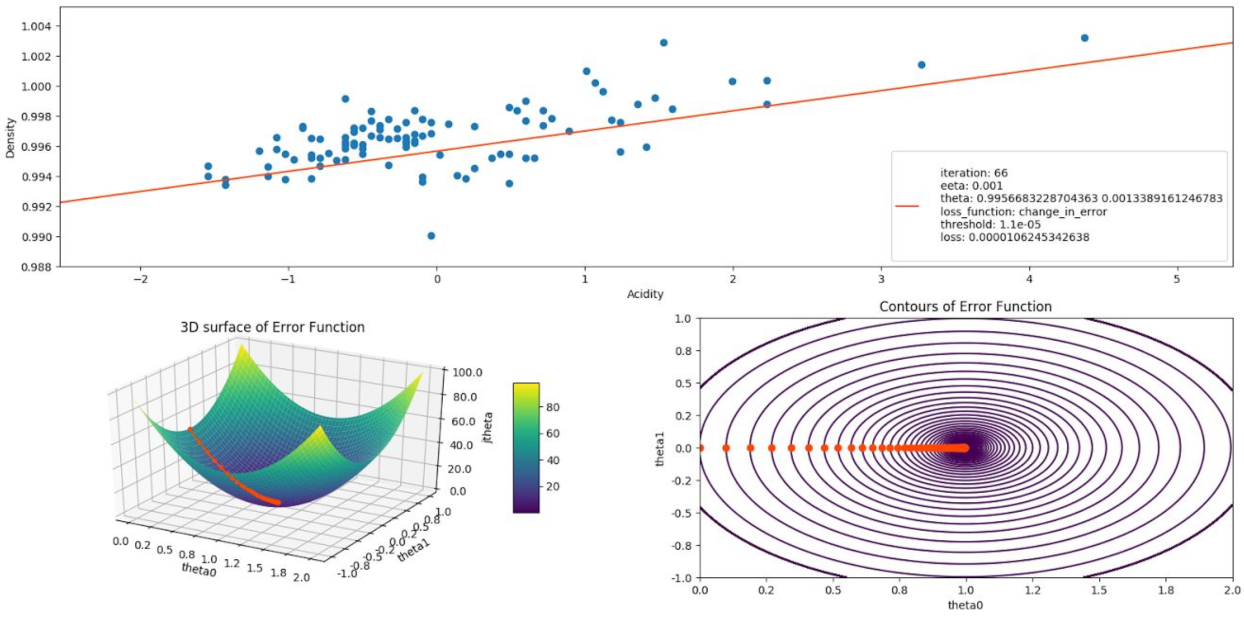
Note: A folder in Q1 directory contains a one plot corresponding to each loss functions

b) c) d)



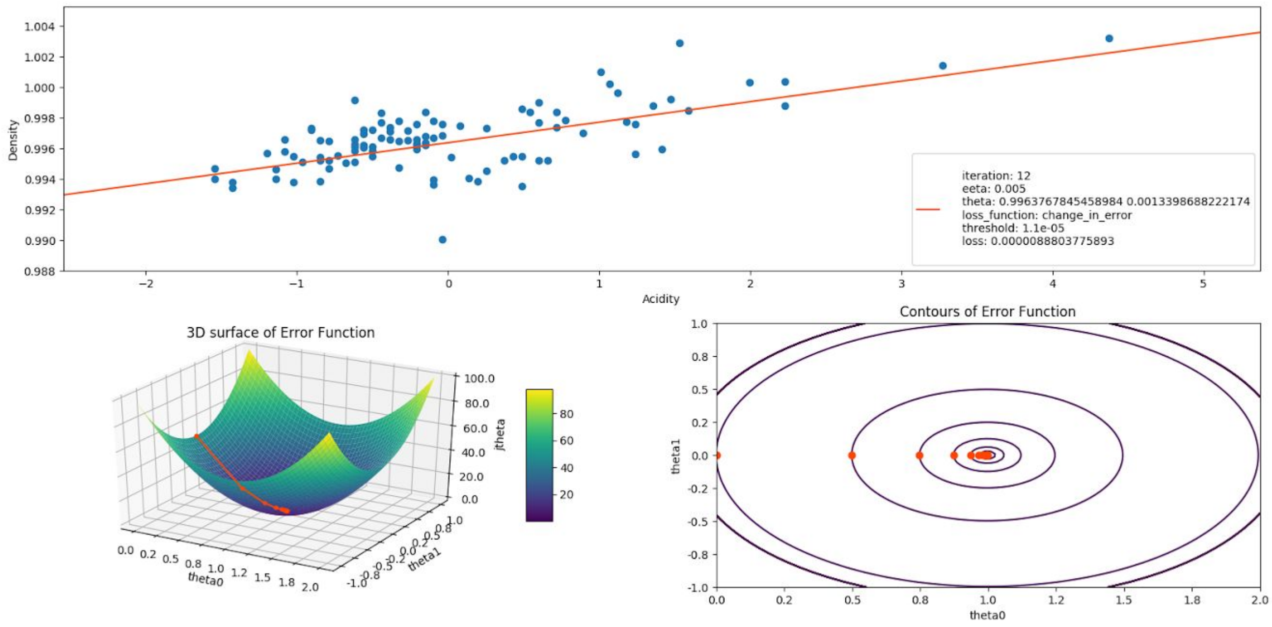
```
iteration: 176
eeta: 0.019
theta: 0.9966200911852237 0.0013401960066574
loss_function: change_in_error
threshold: 1.1e-15
loss: 0.0000000000000009
```

e)



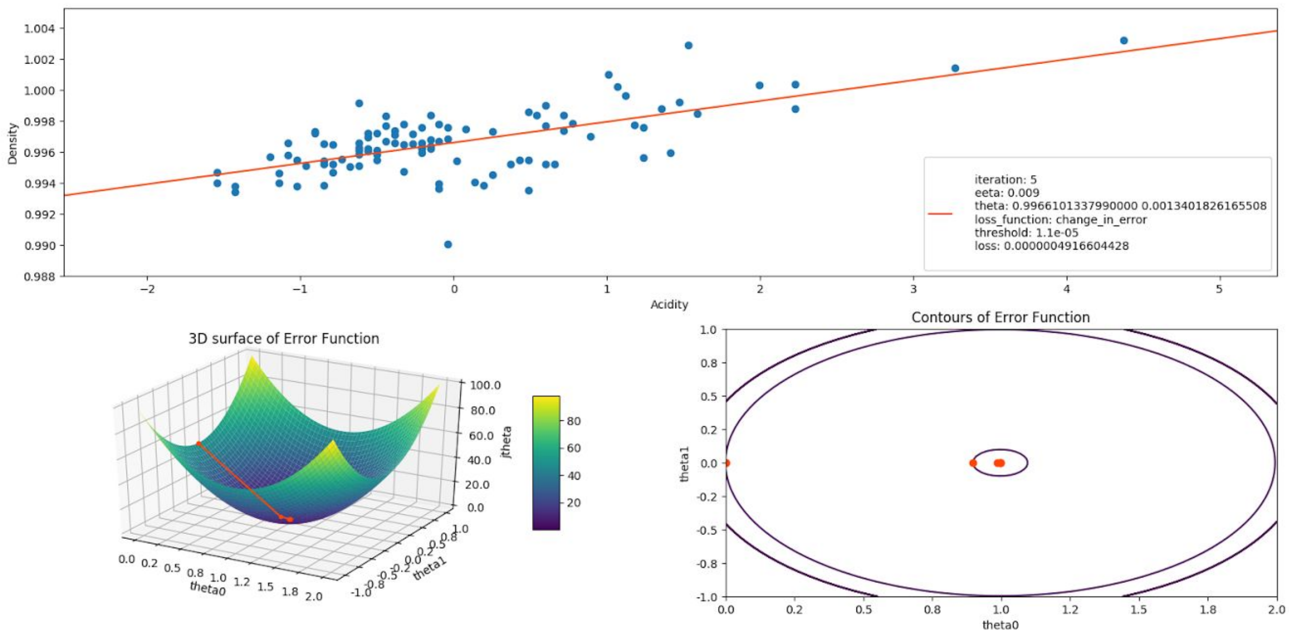
```
iteration: 66
eeta: 0.001
theta: 0.9956683228704363 0.0013389161246783
loss_function: change_in_error
threshold: 1.1e-05
loss: 0.0000106245342638
```

Learning rate = 0.001

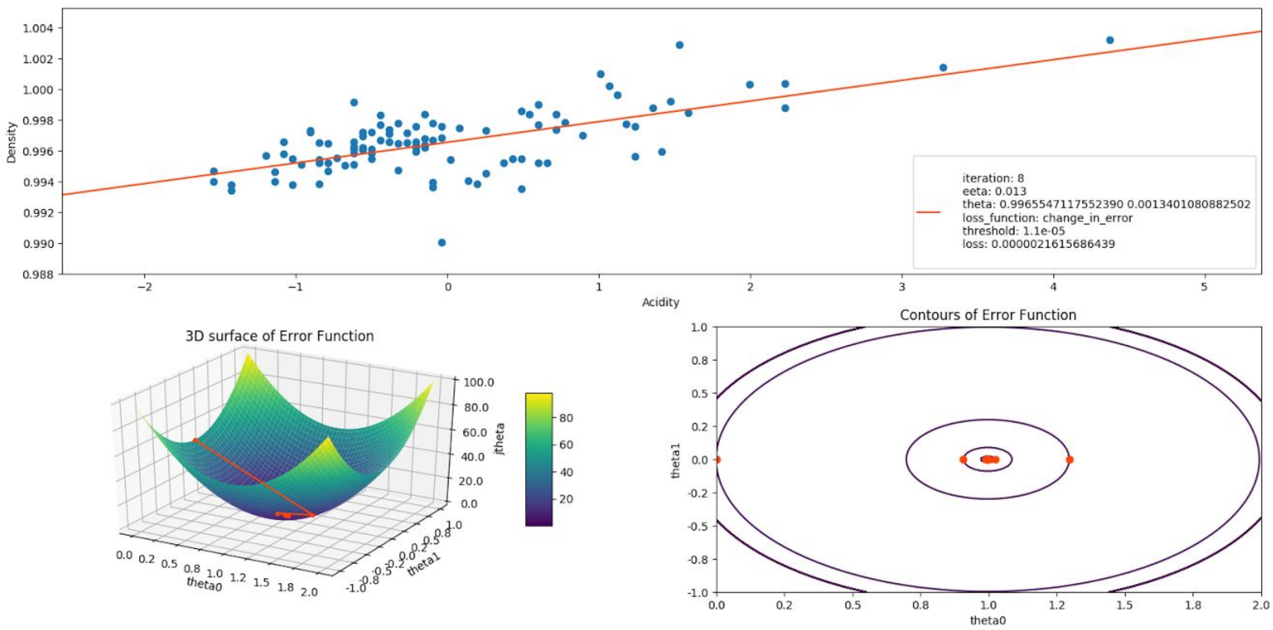


```
iteration: 12
eeta: 0.005
theta: 0.9963767845458984 0.0013398688222174
loss_function: change_in_error
threshold: 1.1e-05
loss: 0.0000088803775893
```

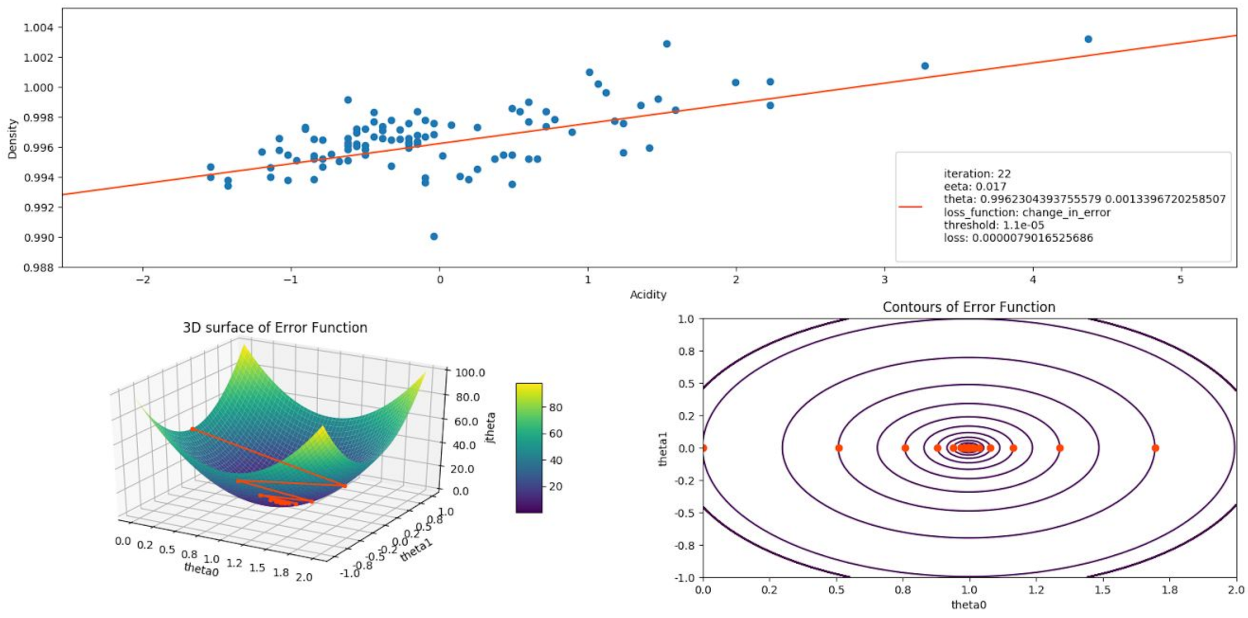
Learning rate = 0.005



Learning rate = 0.009

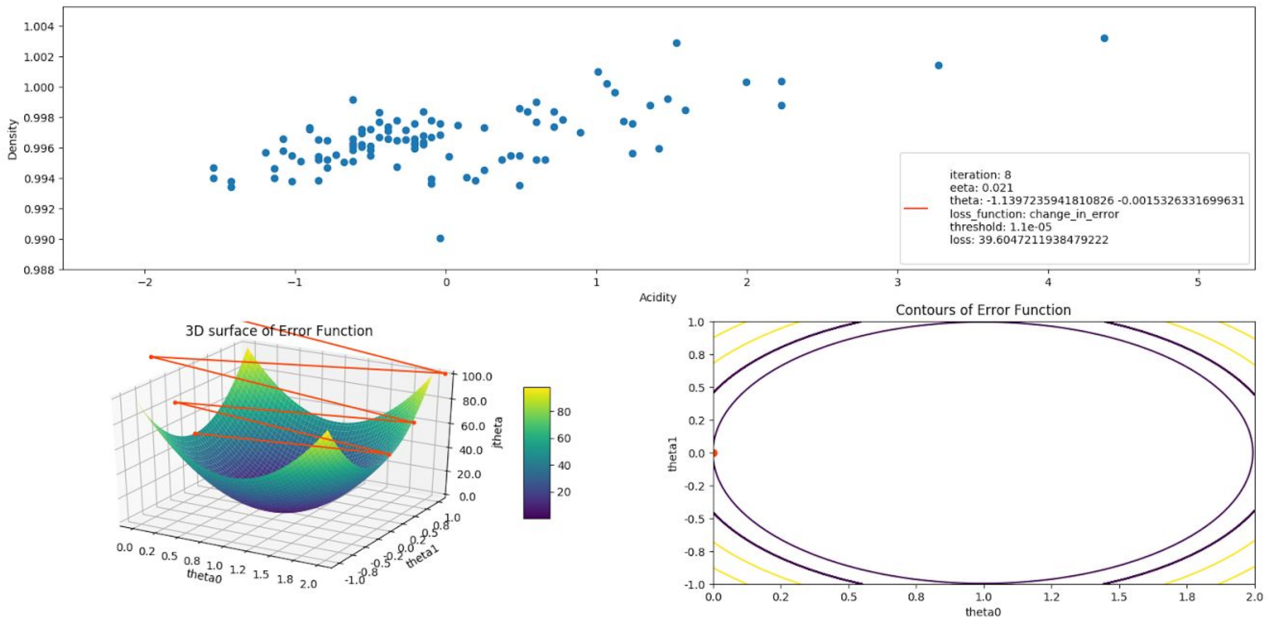


Learning rate = 0.0.13



iteration: 22
eeta: 0.017
theta: 0.9962304393755579 0.0013396720258507
loss_function: change_in_error
threshold: 1.1e-05
loss: 0.0000079016525686

Learning rate = 0.017



iteration: 8
eeta: 0.021
theta: -1.1397235941810826 -0.0015326331699631
loss_function: change_in_error
threshold: 1.1e-05
loss: 39.60472119384792222

Learning rate = 0.021

Observation

As can be seen from the above plots (for the threshold of 1.0e-5)

At first on increasing the learning rate the number of iterations required for convergence decreases. **Number of iterations decreases from 66 to 12 to 5** on increasing learning rate from 0.001 to 0.005 to 0.009. This is because we are making bigger updates to θ per iterations.

But after that on increasing the learning rate further to 0.013 and 0.017 the number of i**terations required starts increasing again to 8 and then to 22.** This, as can be seen from the surface plot, is due to the fact that the learning rate is big enough to make the candidate theta jump to the other side of the minima. Thus algorithm instead of descending to the minima, oscillates its way to the minima, thus requiring more iterations.

On further increasing the learning rate, the jumps in theta in either directions are big enough to make the algorithm diverge.

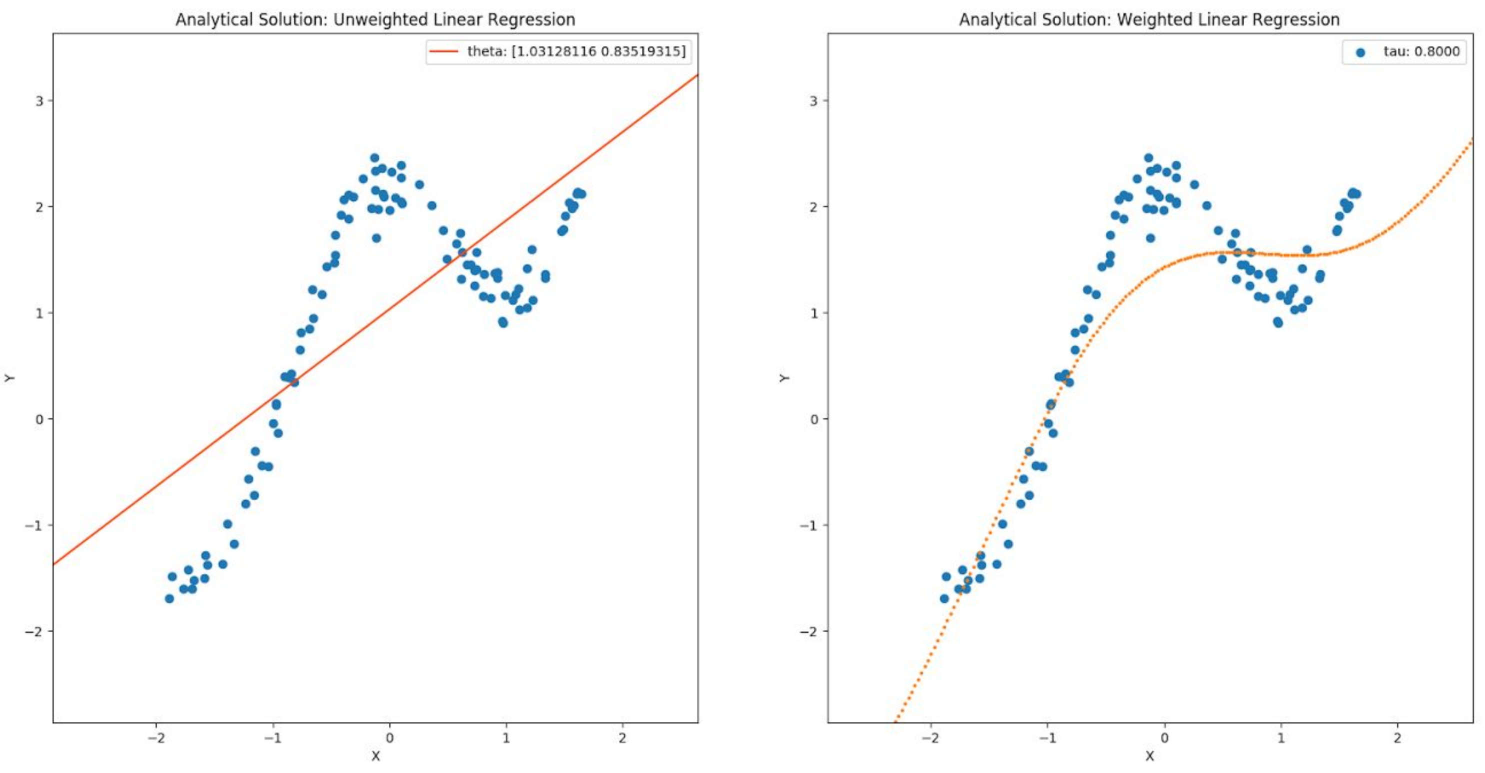
Q2: Locally weighted Linear Regression

a) Equation Used:

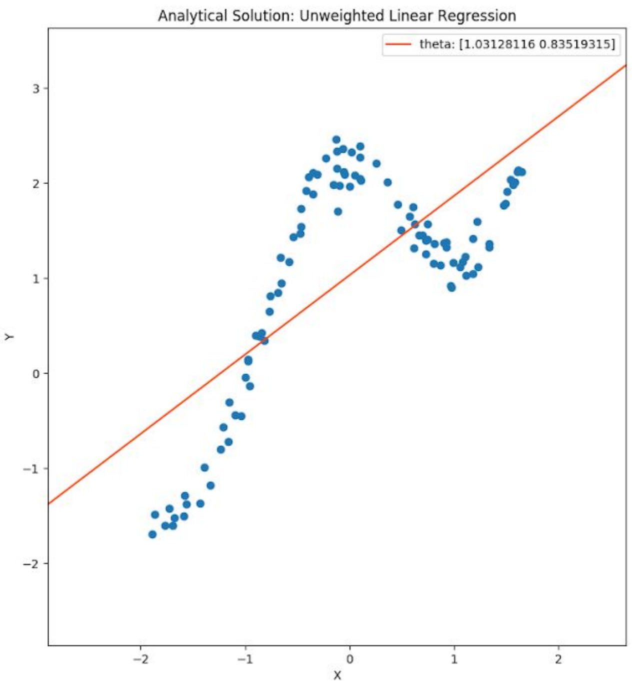
$$\theta = (X^T X)^{-1} X^T Y$$

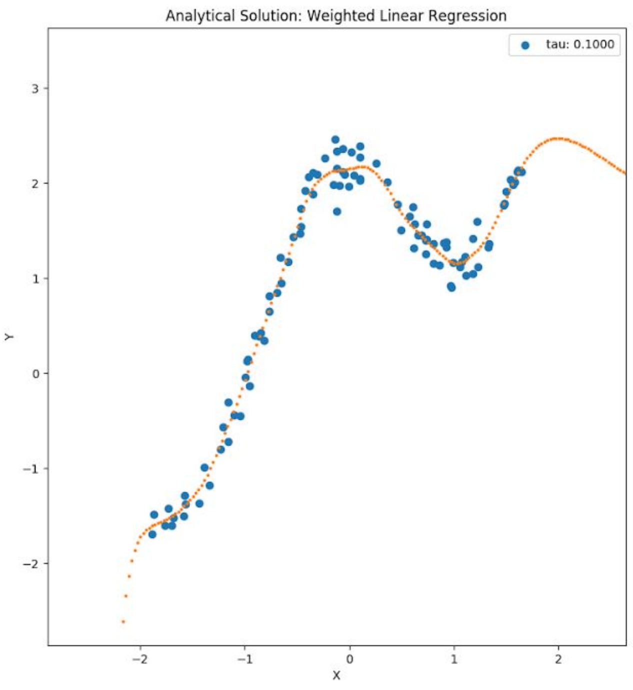
b) Equation Used:

$$\theta = (X^T W X)^{-1} X^T W Y$$

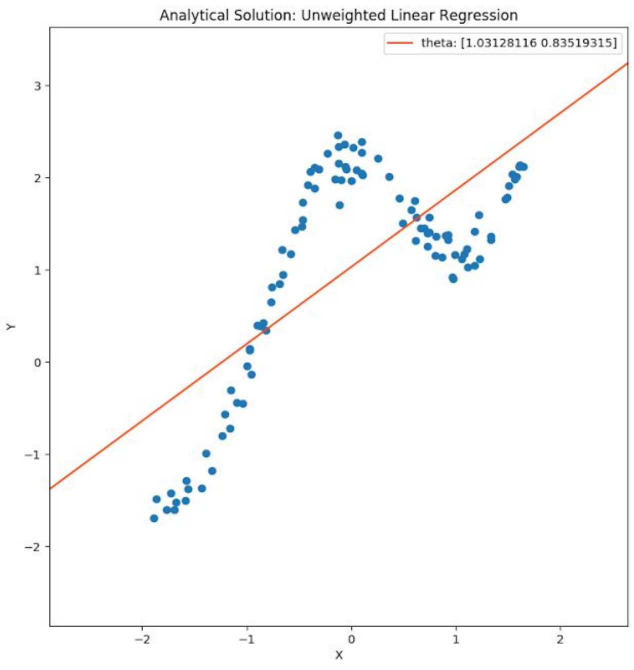


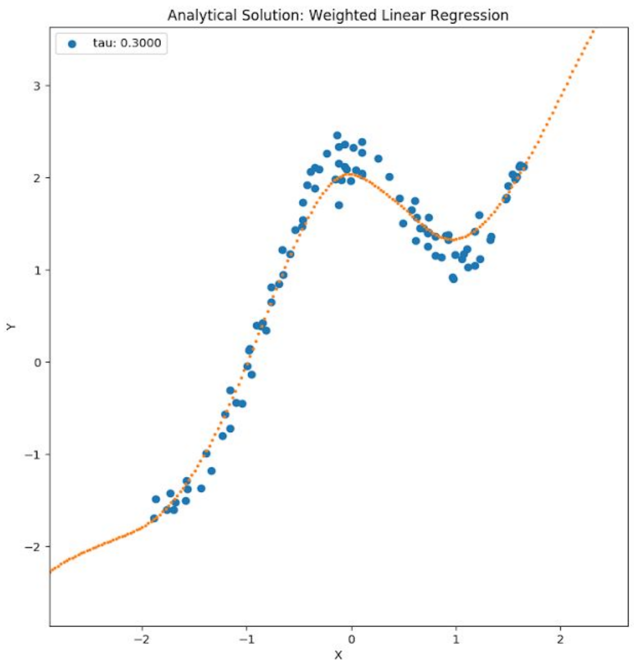
Plot for τ = 0.8



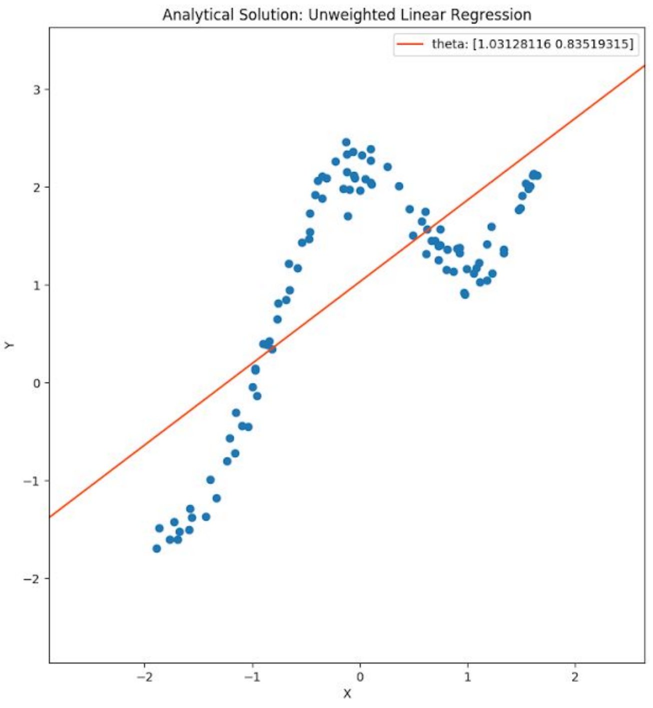


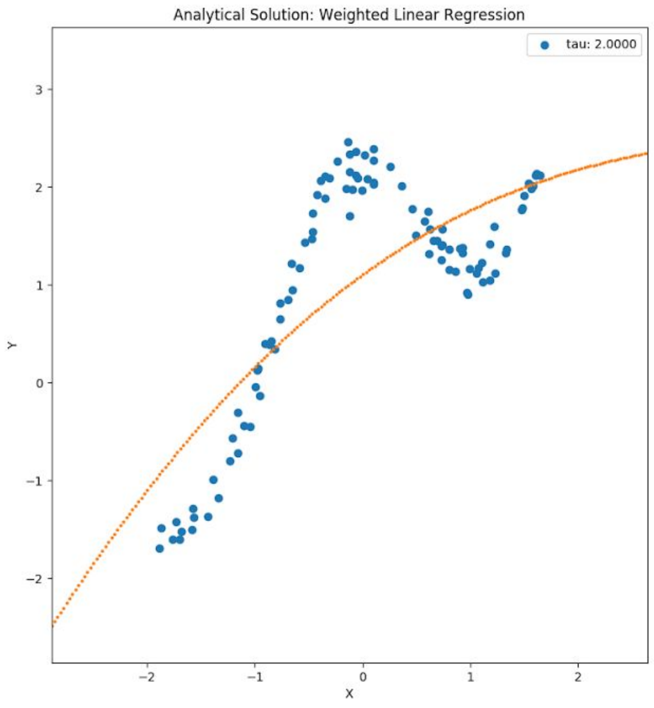
$\tau = 0.10$



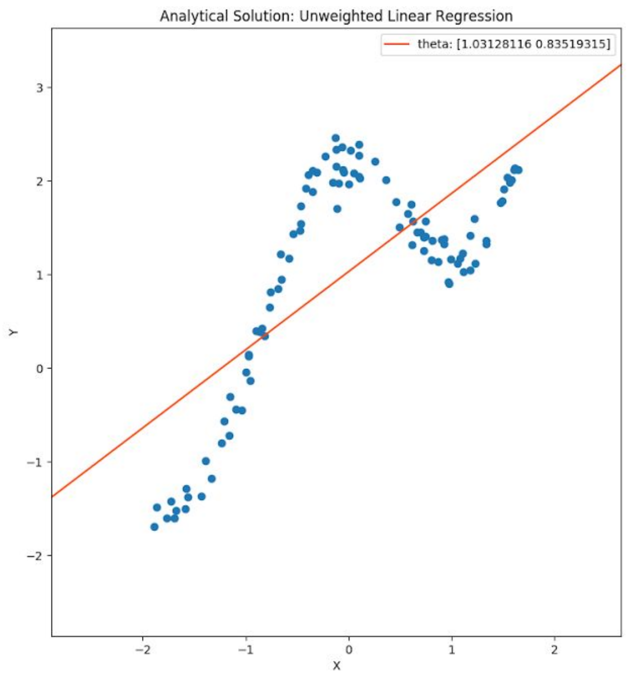


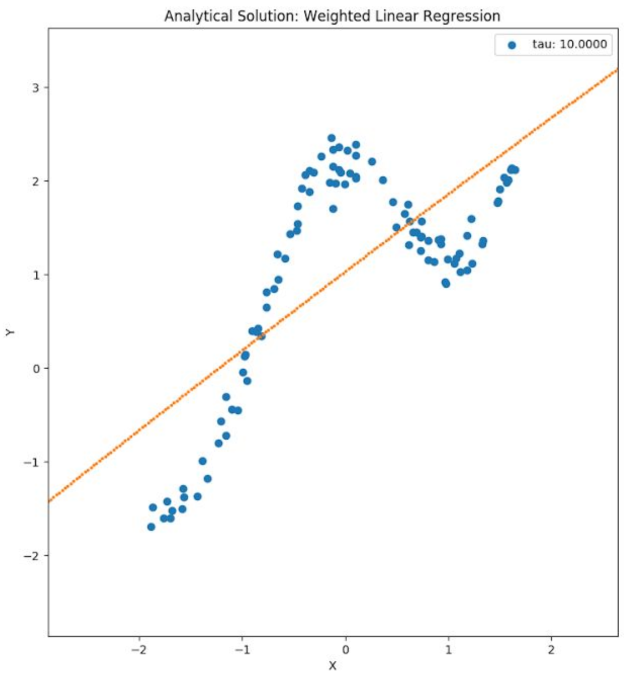
$\tau = 0.30$


Analytical Solution: Unweighted Linear Regression
theta: [1.03128116 0.83519315]
X
Y


Analytical Solution: Weighted Linear Regression
tau: 2.0000
X
Y

τ = 2.0


Analytical Solution: Unweighted Linear Regression
theta: [1.03128116 0.83519315]
X
Y


Analytical Solution: Weighted Linear Regression
tau: 10.0000
X
Y

τ = 10.0

Observation

We can see that for **small values** of τ (e.g 0.10), the curve obtained from locally weighted gradient descent seems to **overfit** the data, following the data too tightly.

For **large value** of τ (e.g for 2 and 10), the curve starts to **underfit** and starts becoming more and more **linear**. As can be seen the curve for τ = 10, is completely linear and is <u>identical to the line obtained by the unweighted gradient descent algorithm.</u>

Q3: Logistic Regression

a) Equations Used:

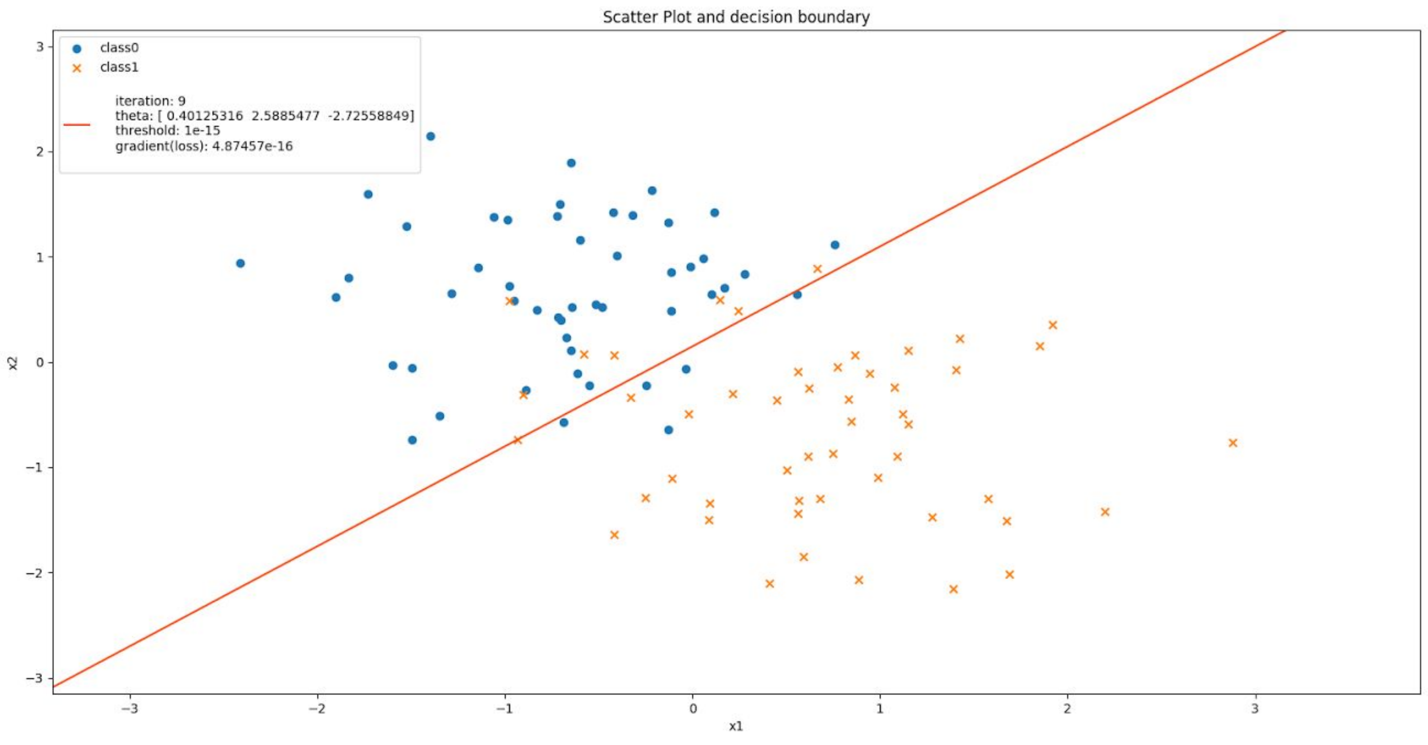
$$Gradient_j = \sum_{i=1}^{m} [\, y^i - g(\theta^T x^i) \,]\, x_j$$

$$Hessian_{ij} = \sum_{k=1}^{m} x_i^k x_j^k \,[\, g(\theta^T x^k) - 1 \,]\, g(\theta^T x^k)$$

Final Values of θ using newton's update method are

Number of Iterations: 9
θ : 0.40125316, 2.5885477, -2.72558849
Threshold: 1e-15
Gradient(loss): 4.87457e-16
Stopping Criteria: loss < threshold

b) Scatter Plot of the training data and the decision boundary



Q4 Gaussian Discriminant Analysis

a)

- $\mu_0 = \begin{bmatrix} -0.75529433 \\ 0.68509431 \end{bmatrix}$
- $\mu_1 = \begin{bmatrix} 0.75529433 \\ -0.68509431 \end{bmatrix}$
- $\Sigma = \begin{bmatrix} 0.42953048, & -0.02247228 \\ -0.02247228, & 0.53064579 \end{bmatrix}$
- $\phi = 0.5$

d)

- $\mu_0 = \begin{bmatrix} -0.75529433 \\ 0.68509431 \end{bmatrix}$
- $\mu_1 = \begin{bmatrix} 0.75529433 \\ -0.68509431 \end{bmatrix}$
- $\Sigma_0 = \begin{bmatrix} 0.38158978, & -0.15486516 \\ -0.15486516, & 0.64773717 \end{bmatrix}$
- $\Sigma_1 = \begin{bmatrix} 0.47747117, & 0.1099206 \\ 0.10992060, & 0.41355441 \end{bmatrix}$
- $\phi = 0.5$

b) c) e)

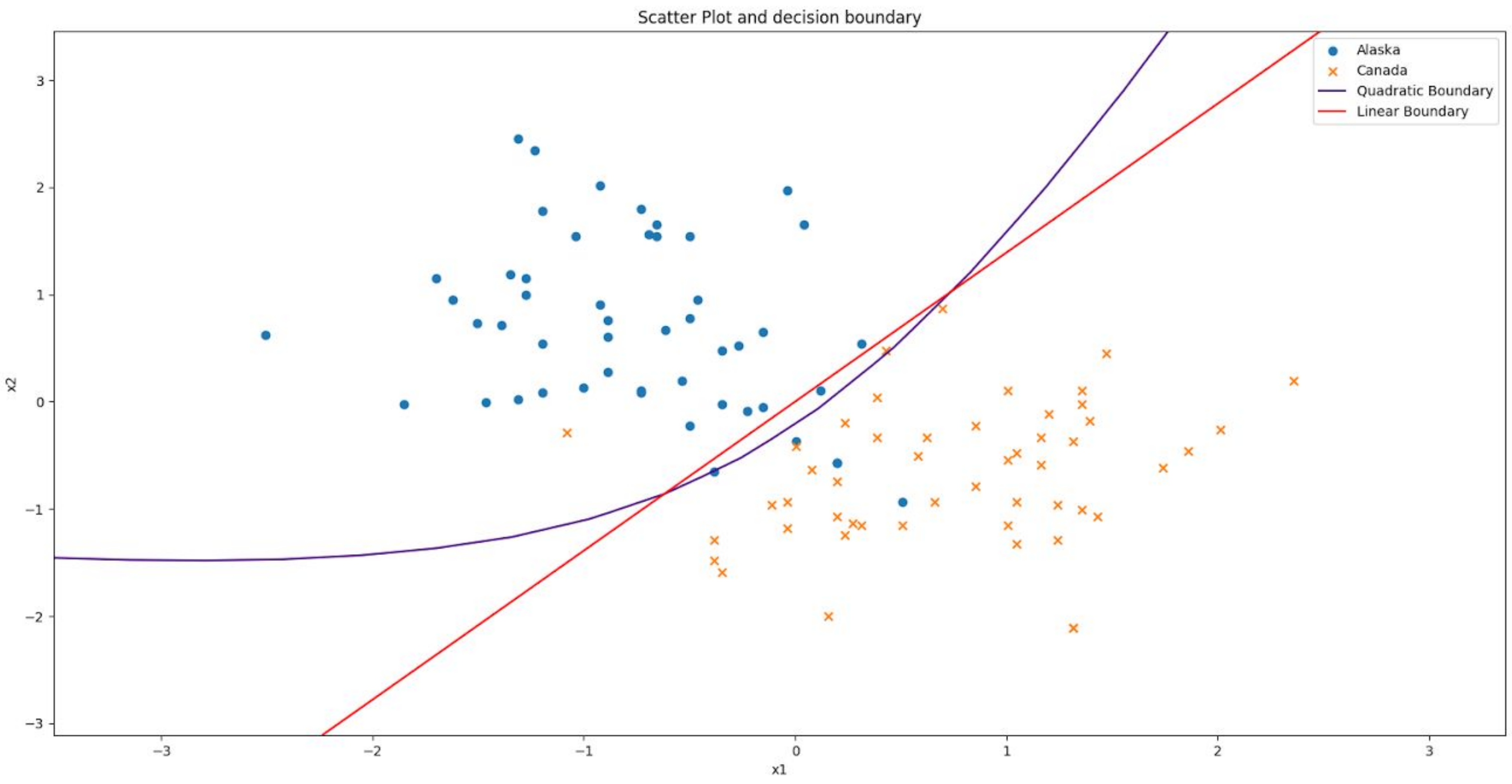
Equation of the boundary used:

$$\frac{1}{2}\left[(x - \mu_1)^T \Sigma_1^{-1} (x - \mu_1) - (x - \mu_0)^T \Sigma_0^{-1} (x - \mu_0)\right] - \log\left(\frac{\phi}{1-\phi}\right) + \log\left(\left(\frac{\Sigma_1}{\Sigma_0}\right)^{\frac{1}{2}}\right) = 0$$

In the above equation if we put $\Sigma_0 = \Sigma_1 = \Sigma$, then, it depicts the equation of the linear boundary/separator, else it depicts the equation of the quadratic boundary.

The decision but can be plotted by thinking it as the contour of the following 3D surface at z = 0 !!

$$Z = \frac{1}{2}\left[(x - \mu_1)^T \Sigma_1^{-1} (x - \mu_1) - (x - \mu_0)^T \Sigma_0^{-1} (x - \mu_0)\right] - \log\left(\frac{\phi}{1-\phi}\right) + \log\left(\left(\frac{\Sigma_1}{\Sigma_0}\right)^{\frac{1}{2}}\right)$$



Scatter Plot of training data and the decision boundaries

e) Observation:

1. There are two points, one of salmon from Alaska and one from canada that are on the different sides of the two boundaries.
2. Linear boundary classifies both of them as being from Canada, Quadratic boundary classifies both of them as being from Alaska.
3. Linear boundary seems to be better as there seems to be too few points to visually/intuitively justify the curvature of the decision boundary