DHRUV GUPTA

2023EE30858

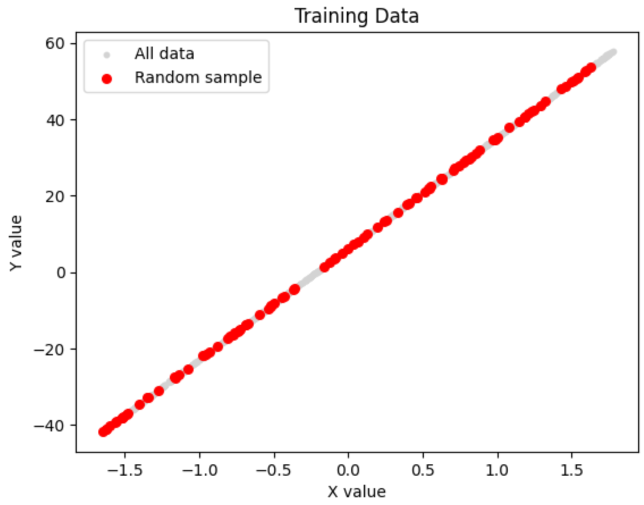
Assignment 1 Report

PART – 1(Linear Regression)

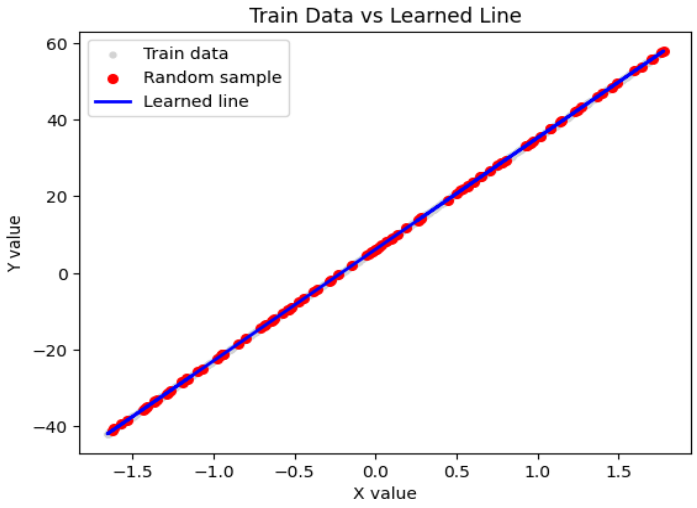
Part 1.1

- Learning Rate used: 0.1
- Learned Parameters (Theta(θ)): [6.21867784 29.06475432]
- (θ_0 = Bias, θ_1=Slope)
- Stopping Criteria: $| J_\theta(t) - J_\theta(t-1) | < 10^{-10}$

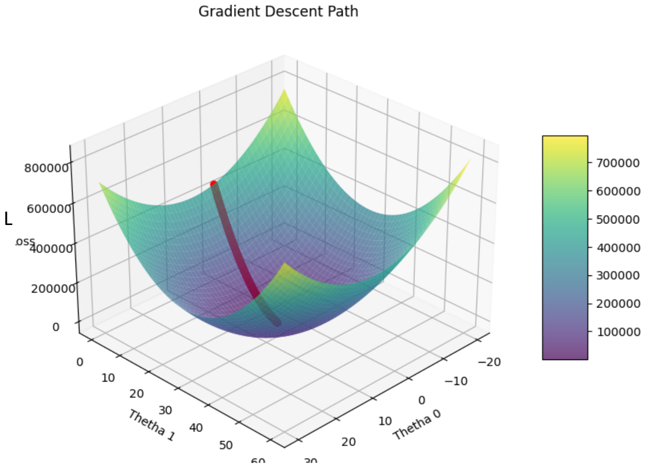
Part 1.2(Plots)



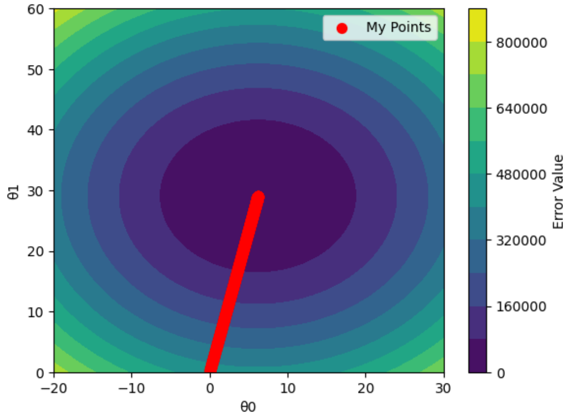
(a) Given Data



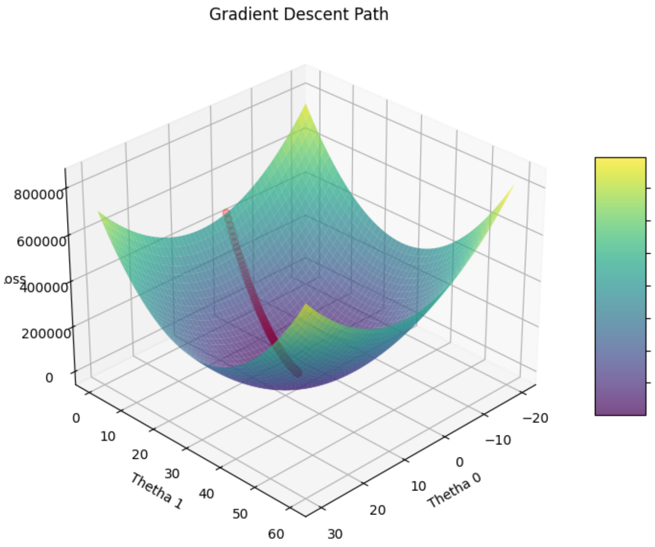
(b) Learned line and Train Data



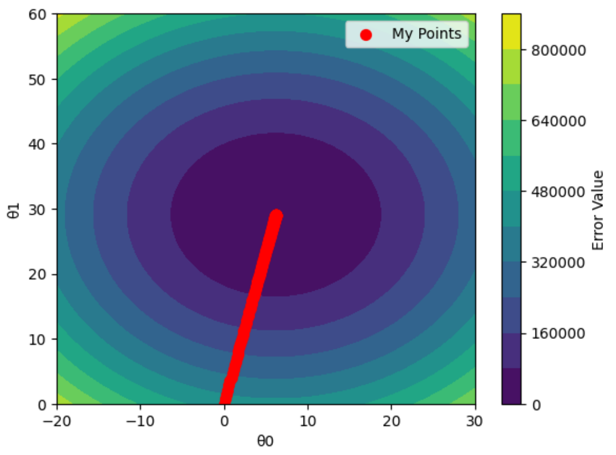
(c) Error function with learnt model LR=0.001



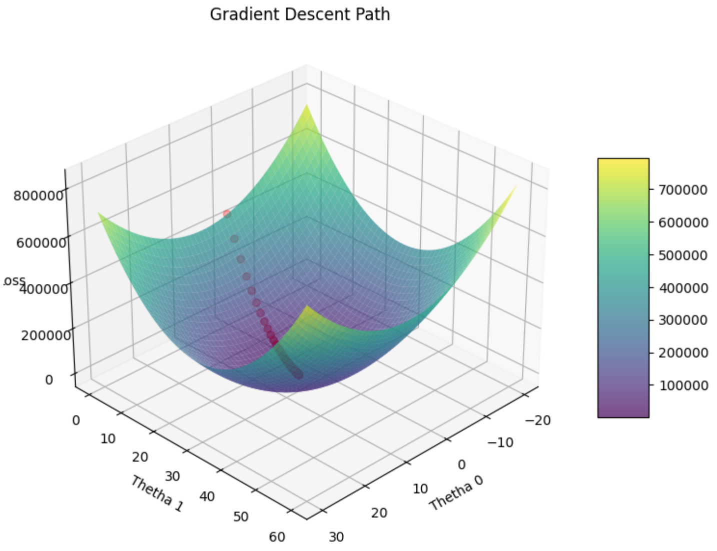
(d) Contour, LR = 0.001



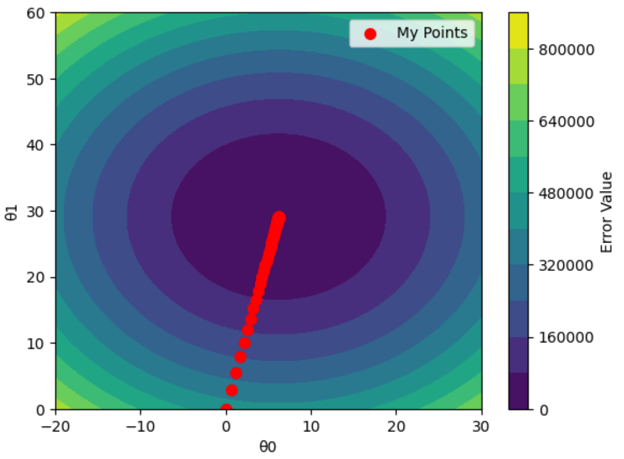
(e) Error function with learnt model LR=0.025



(d) Contour, LR = 0.025



(e) Error function with learnt model LR=0.1



(d) Contour, LR = 0.1

Conclusion:

With an increased learning rate, the convergence is faster (the number of iterations decreases) and jumps towards minima increases, but the risk of skipping the minima increases. A larger learning rate is prone to high error, and smaller Learning rate is prone to very low convergence speed and, hence, more time. A wise choice of learning rate is necessary for good working of a model w.r.t. to both training and accuracy, therefore from the plots LR = 0.1 seems a wise choice. Considering the tradeoff between time and skipping condition of minimum, stopping criteria is chosen to provide low loss and reasonable time for convergence. In contour animations, the contour ends very early for 0.001 and 0.025 learning rates because, I have taken only 50 points to show the reasonable animation, and since gradient descent is slow for smaller LR, therefore animation is congested and does not converge.

PART-2(Stochastic Gradient Descent)

Part-2.1

- For the convergence check I calculate the norm of $|| \nabla f(\theta_{t,r}) || < 10^{-6}$, or else if maximum epoch specifies the over, I only run 10^5 iterations and stop even if norm specification is not satisfied. $J_{\theta,\, r}$ (t) is calculated after updating the **θ** parameters.
- Batch size of 8000 seems to be most optimal choice.
- Error with original parameters (θ = [3,2,1)), Train Loss = 1.99695609, Test Loss = 1.99743816

Part -2.2

Learning rate(η) = 0.001

Batch Size(r)	Learned Parameters (θ)	Test Loss	Train Loss	Iterations	Time(seconds)
1	$[2.80872619\ 1.01578505\ 2.01039561]^T$	2.01182295	2.00906929	100000	3
80	$[2.99802325\ 1.0035842\ \ 2.00409566]^T$	1.99751224	1.99702381	100000	3.6
8000	$[3.00047457\ 0.99905255\ 2.00205049]^T$	1.99747917	1.99694505	100000	33.8
800000	$[2.99695436\ 1.00010807\ 2.00137266]^T$	1.99748852	1.99694681	100000	850

Part-2.3

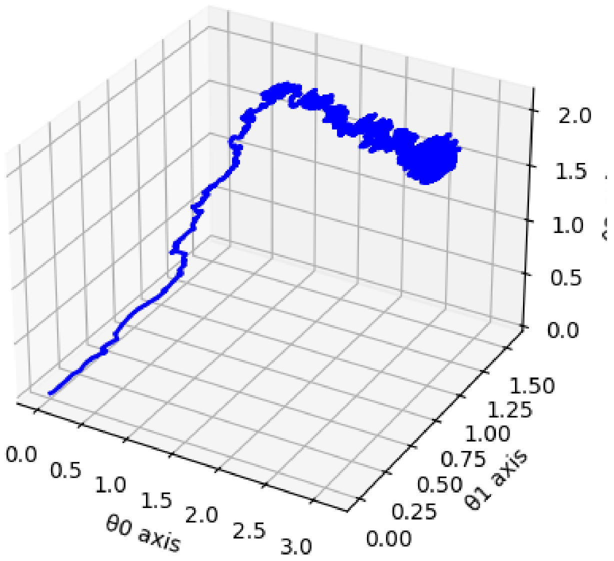
The Parameters learned from the closed form solution for linear regression using - $\boxed{\theta = (X^TX)^{-1}X^TY}$

are - $\boxed{[3.00054272\ 0.99932353\ 2.00163334]^T}$ and the corresponding loss is 1.99746348.

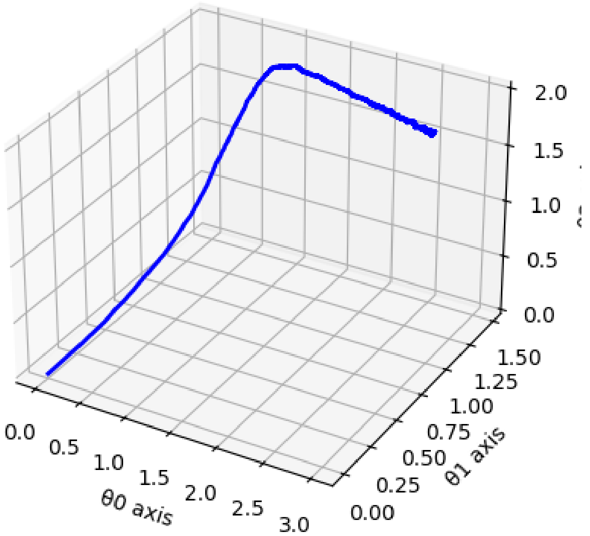
Conclusion and Inference

For the smaller batch size the 3-D θ curve shows random oscillations(may escape minima) and not a continuous curve(proper convergence), the smoothness of 3-D curve increases with the increase in the batch size, this is because more data points are taken into account for a single update of parameters , therefore gradient descends holds more information and takes θ towards minima and not particularly for each data point as the case is for r=1, therefore the movement smoothens out . R= 8000 seems to be the best choice among the different batch sizes as same number of updates of θ, the r=8000 goes close to convergence smoothly and not in an oscillatory motion , whereas lower batch size may deviate causing random oscillations and therefore noisy updates . As batch size increases the same number of updates in θ take large time therefore not an optimal choice for LR = 800000. Test errors are close to the original hypothesis error. For larger batch size, more epochs are required for the convergence because for same number of epochs, θ is updated fewer times for larger batch size than for smaller batch size. Large batches may train faster per epoch but risk worse test accuracy (due to not proper convergence).

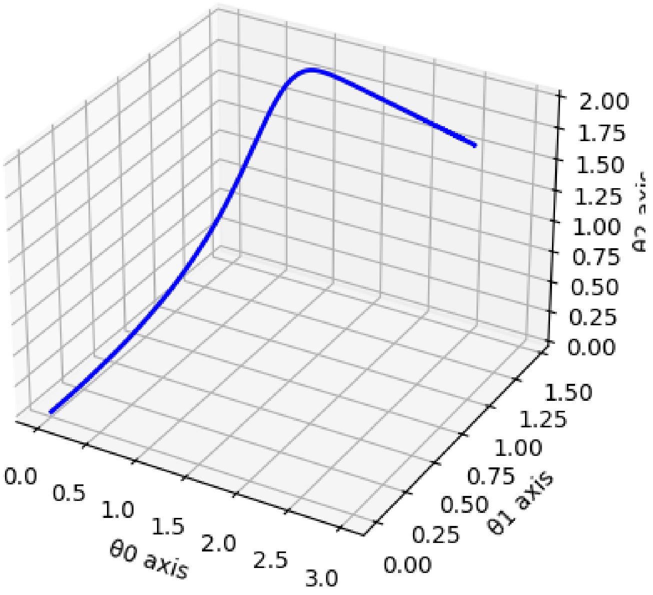
Part – 2.5 (Graphs)



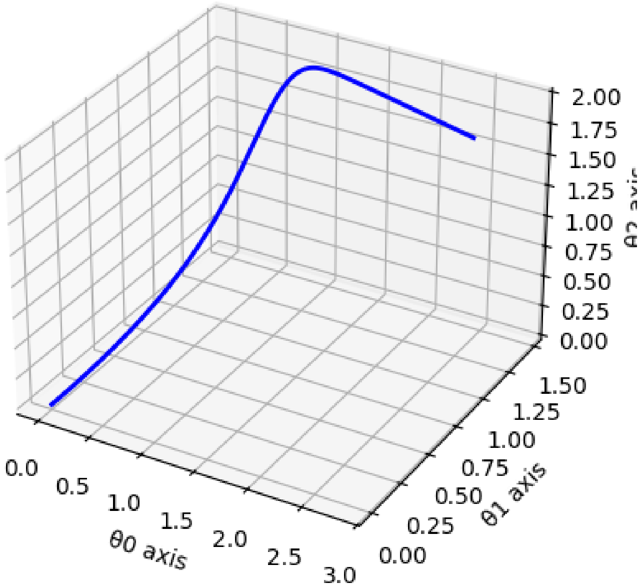
Batch Size = 1



Batch Size = 80



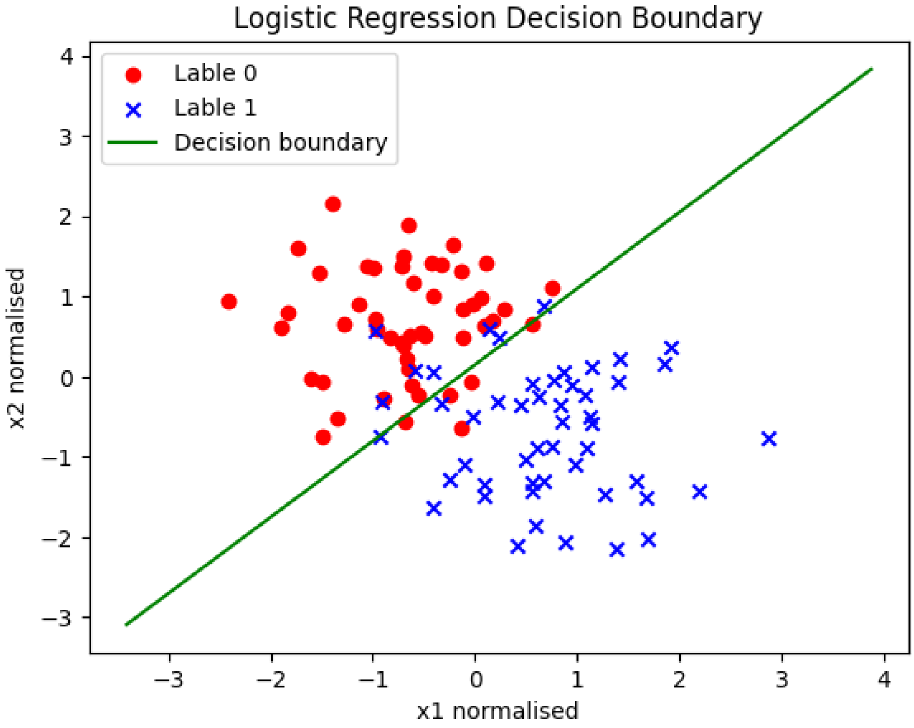
Batch Size = 8000



Batch Size = 800000

PART-3(Logistic Regression)

- Learned Parameters (Theta(θ)) = $(0.40125316, 2.5885477, -2.72558849)^T$
- Convergence Conditions: $|| \nabla f(\theta_t) ||_2 < 10^{-12}$



PART-4 (Gaussian Discriminant Analysis)

Part 4.1

- Assuming $\Sigma_0 = \Sigma_1 = \Sigma$, Alaska as 0 and Canada as 1.
- μ_0 = *[-0.75529433 0.68509431]*, μ_1= *[0.75529433 -0.68509431]*
- Σ =   *[[0.42953048 -0.02247228]*
  [-0.02247228 0.53064579]]

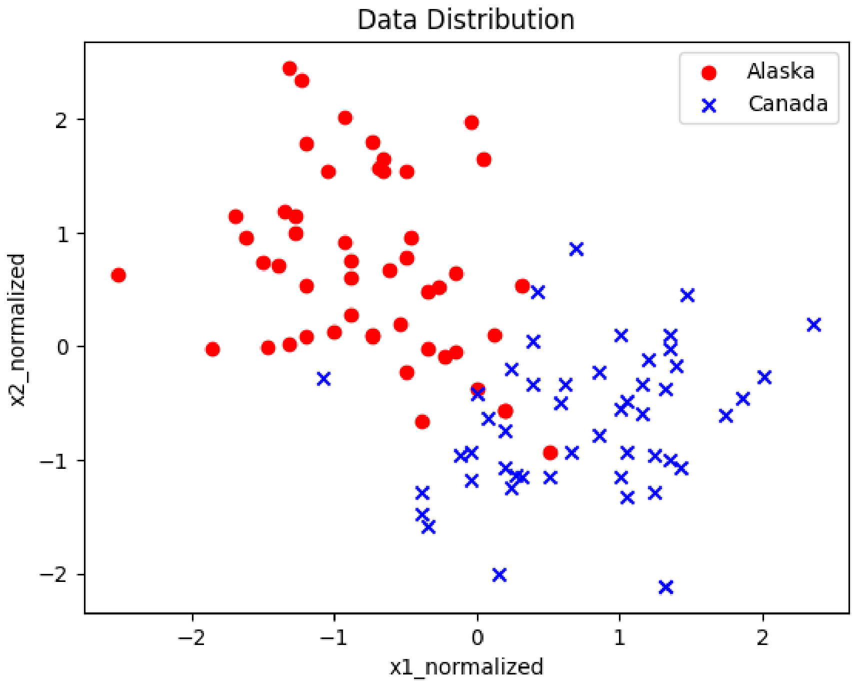
Part 4.4

Σ_0 = *[[0.38158978 -0.15486516]* Σ_1 = *[[0.47747117 0.1099206]*

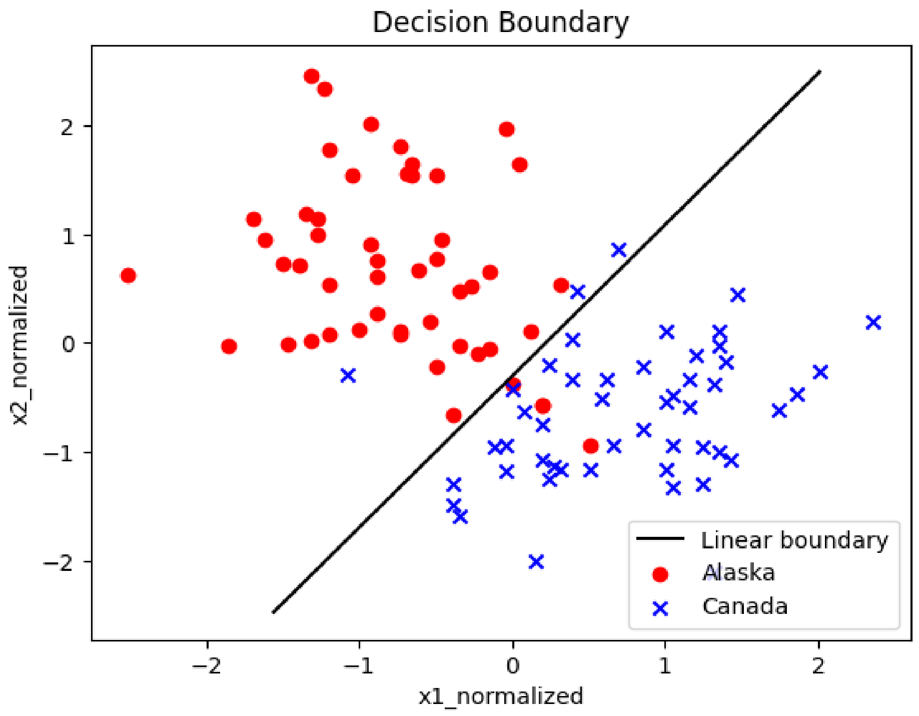
[-0.15486516 0.64773717]]   *[0.1099206 0.41355441]]*

- μ_0 = *[-0.75529433 0.68509431]*, μ_1= *[0.75529433 -0.68509431]*
- ϕ = *0.5* (corresponding to label 0)

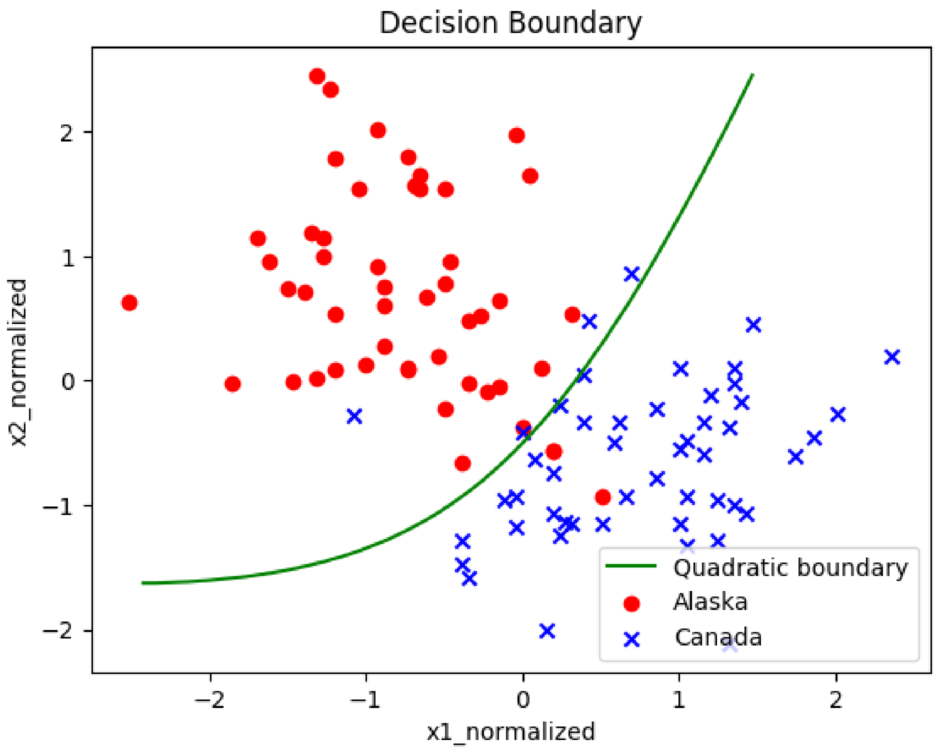
Plots



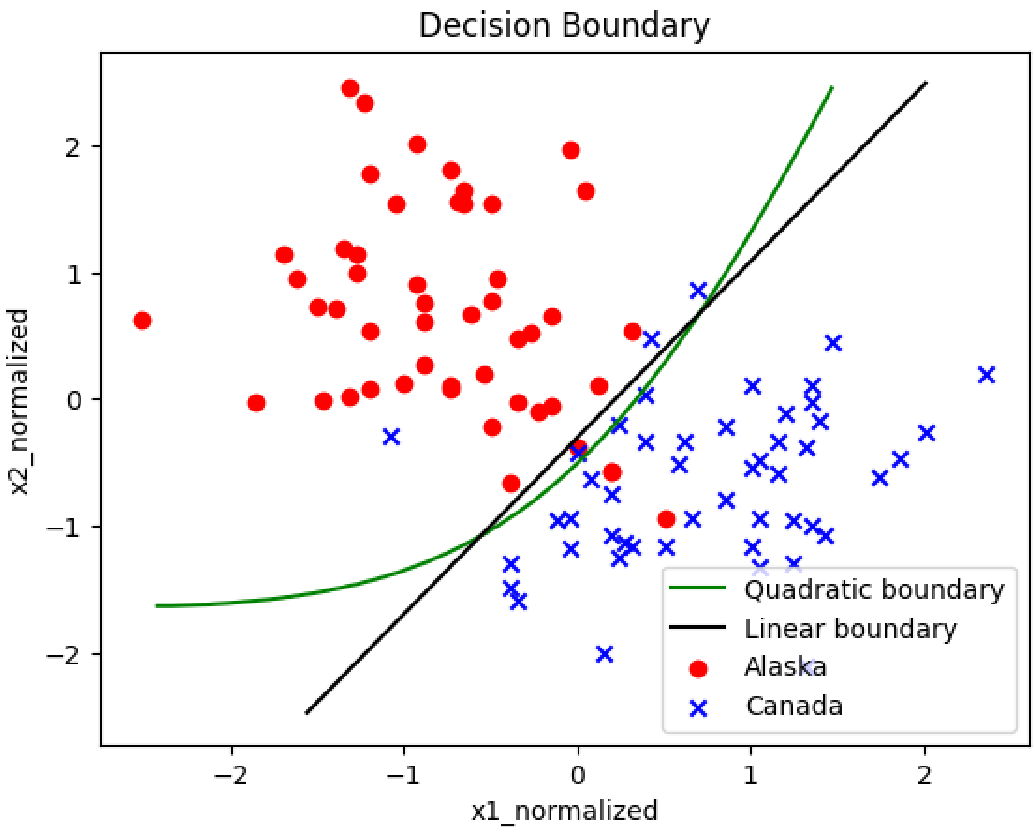
(a) Given Data



(b)Linear Decision boundary for $\Sigma_0 = \Sigma_1 = \Sigma$



(c) Quadratic Decision Boundary $\Sigma_0 != \Sigma_1$



(d) Both Quadratic and Linear Boundaries

Conclusion and Inferences

We can see that a few misclassifications by linear line are correctly classified by the quadratic boundary , thus improving accuracy over logistic regression line , for same covariance the line is same as that of logistic regression. It is likely that curve boundaries may sometimes overfit the data.

Quadratic Boundary Equation

$$(x-\mu_0)^T\Sigma_0^{-1}(x-\mu_0)-(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)+\ln\frac{|\Sigma_0|}{|\Sigma_1|}-2\ln\frac{\pi_0}{\pi_1}=0$$

Linear Boundary equations($\Sigma_0 = \Sigma_1 = \Sigma$) similar to $\theta^T x+b=0$, as in Logistic regression.

$$(\mu_1-\mu_0)^T\Sigma^{-1}x+\left[-\frac{1}{2}(\mu_1^T\Sigma^{-1}\mu_1-\mu_0^T\Sigma^{-1}\mu_0)+\ln\frac{\pi_1}{\pi_0}\right]=0$$