COL774 Assignment 1

Yatharth Kumar, 2020CS10413

September 2, 2023

1 Linear Regression

1.1 Final Parameters

- Learning Rate = 0.01
- Stopping Criteria, $\epsilon = 0.0000000000001$
- $\theta = [0.99661018, 0.00134018]$

1.2 Hypothesis Function Plot

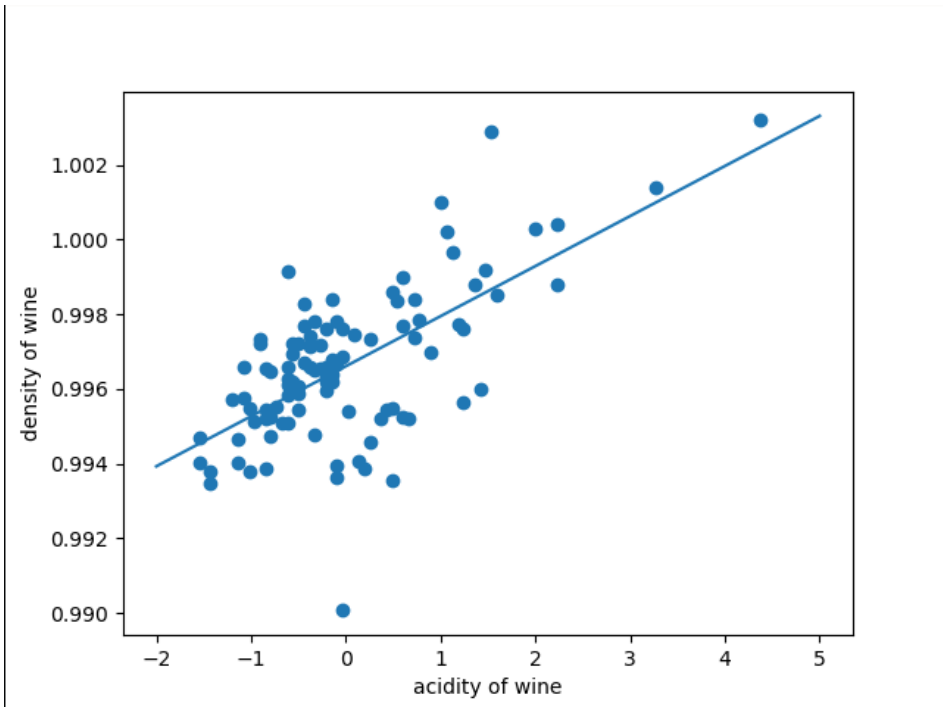

Figure 1: Hypothesis Function

1.3 Cost Function Plot

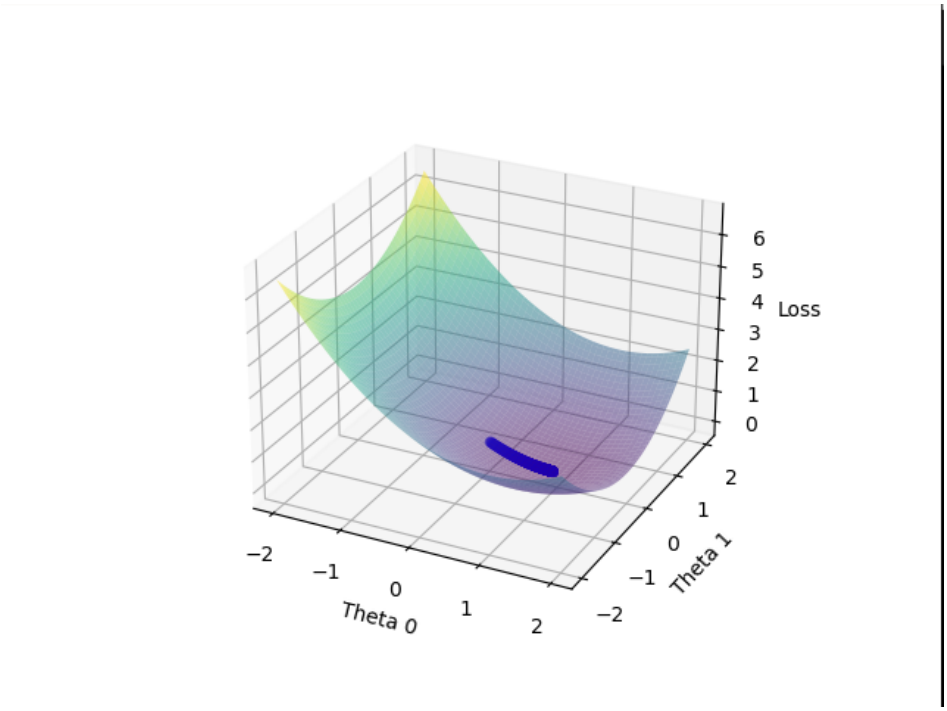

Figure 2: Cost Function

1.4 Contour Plot

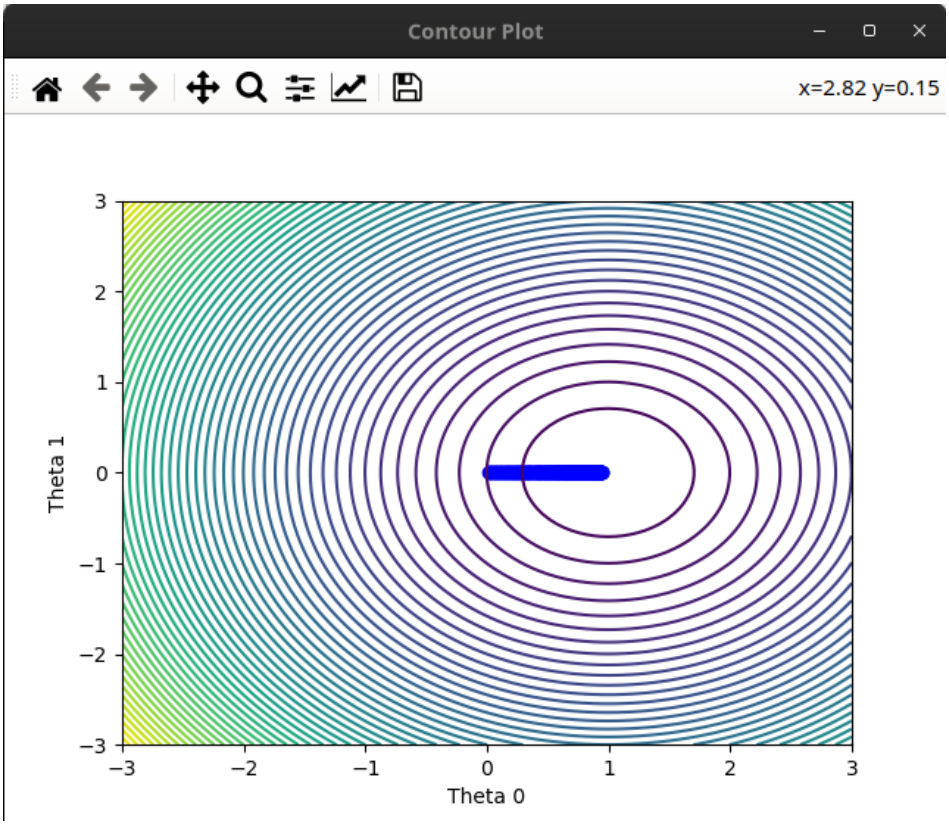

Figure 3: Contour

1.5 Different Step Size Analysis

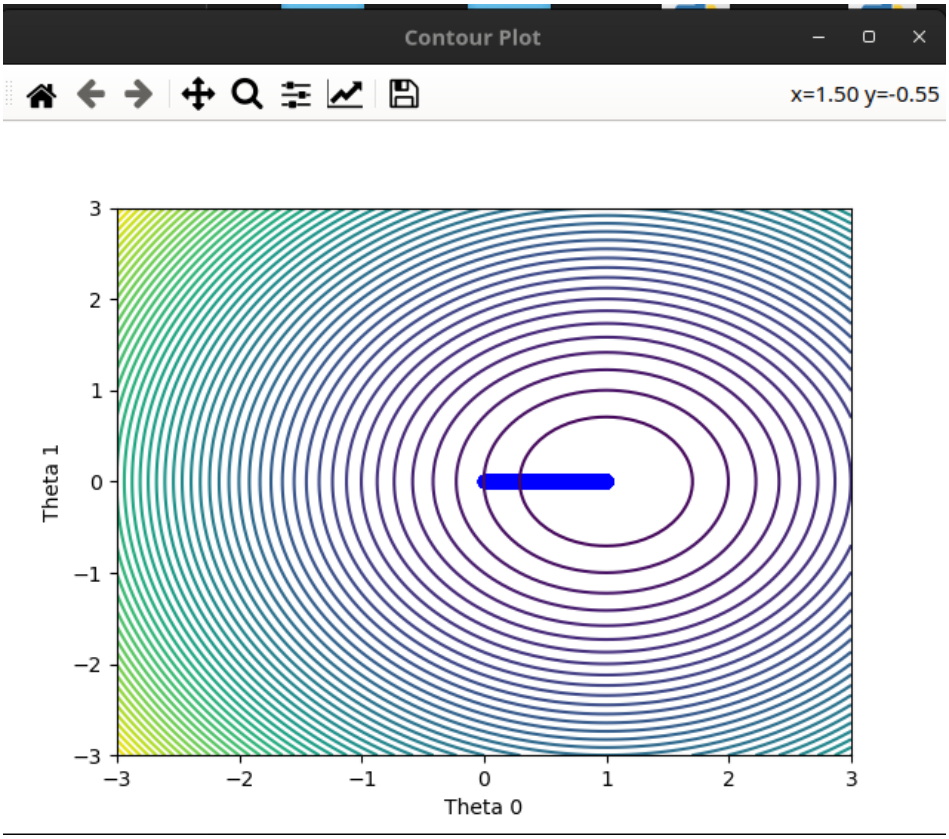

Figure 4: $\eta = 0.001$

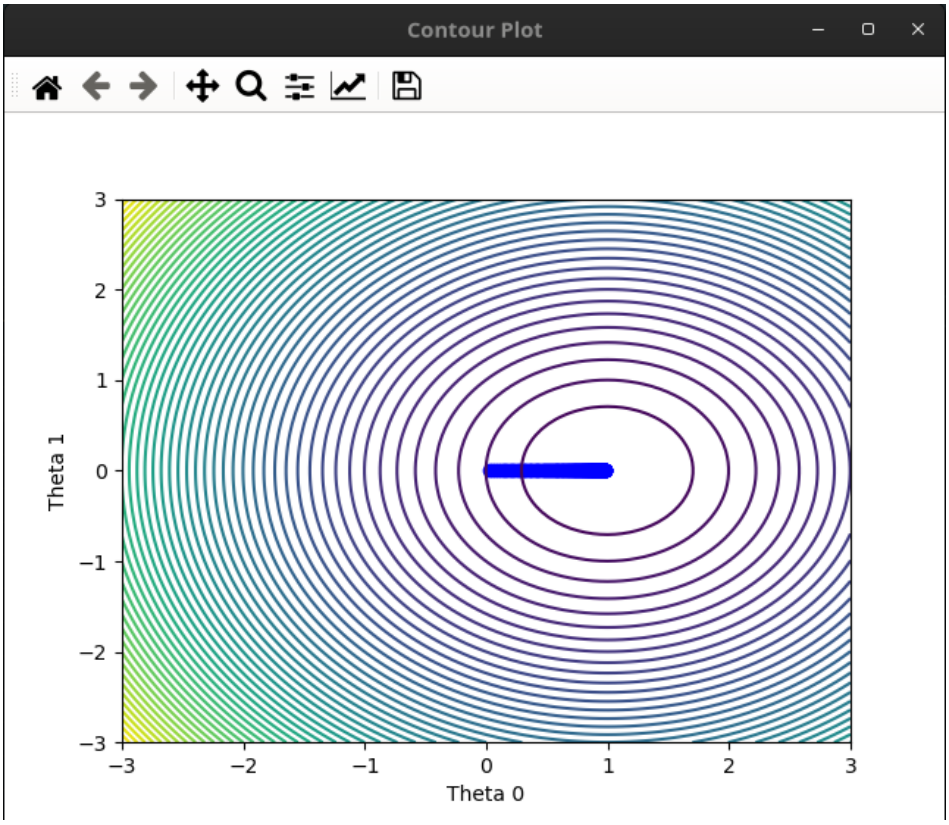

Figure 5: $\eta = 0.025$

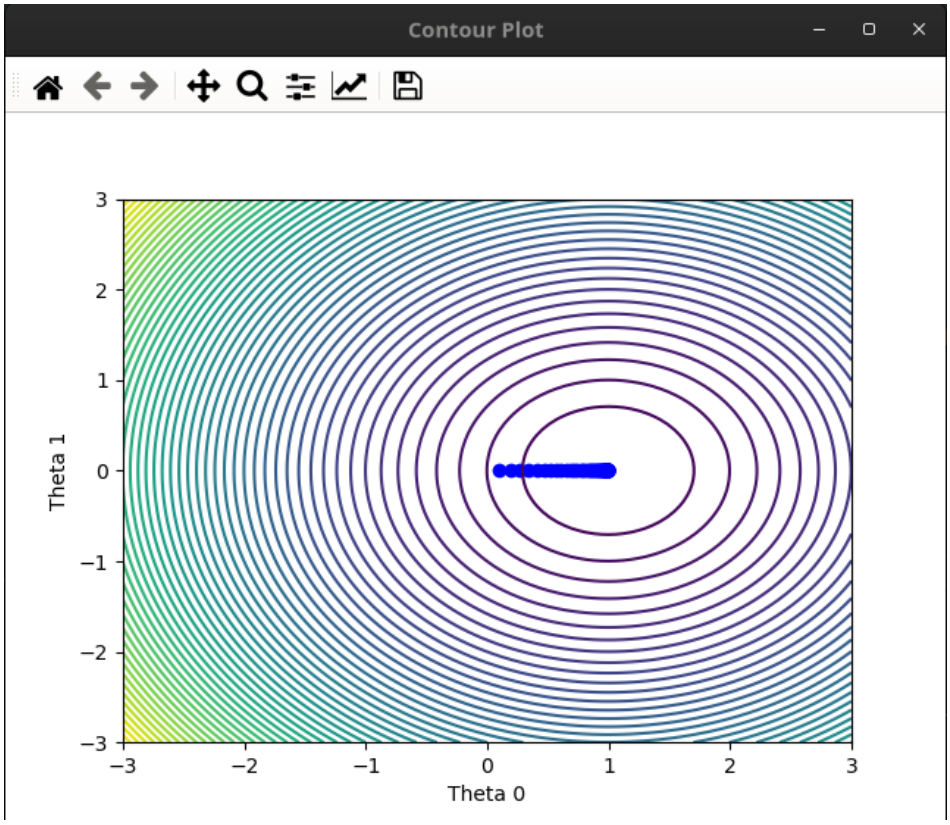

Figure 6: $\eta = 0.1$

Increasing the learning rate results in larger steps toward the local optimum, potentially leading to quicker convergence. However, a high learning rate can also cause overshooting, causing the optimization process to oscillate around the optimum rather than settling into it.

2 Sampling and Stochastic Gradient Descent

2.1 Parameters Learned

- r = 1
  - θ : $[3.00628859, 0.95340697, 2.03434063]$
  - iterations = 51300
- r = 100
  - θ : $[2.97696983, 1.00400722, 1.99806896]$
  - iterations = 17700
- r = 10000
  - θ : $[2.95488385, 1.07485514, 1.97378174]$
  - iterations = 7700
- r = 1000000
  - θ : $[2.9940582, 1.0408797, 2.01411928]$
  - iterations = 5810

Convergence Condition: I terminate the iterative process when the absolute difference between the average error of the last k iterations at time t and the average error of the last k iterations at time t - 1 becomes smaller than a certain. Here, the value of k is determined as the minimum of 100 and the number of batches. For my simulations, I employed different thresholds, namely [1e-10, 1e-8, 1e-5, 1e-5] for batch sizes [1, 100, 10000, 1000000] respectively.

2.2 Convergence Analysis

The mini-batch algorithm's convergence behavior varies with batch size due to its inherent stochasticity, where data point order changes with each run. While parameter values differ across batch sizes, they consistently approximate the actual values. Larger batch sizes result in extended convergence times but require fewer iterations. This delay arises from the increased computational complexity of updating θ for larger batches. In contrast, smaller batch sizes yield more random θ updates, sometimes leading to steps in the opposite direction of the optimum, as observed in the $r = 1$ plot. Conversely, with a batch size of $r = 1,000,000$, θ's movement consistently progresses toward the optimum, showcasing the reduction in randomness associated with larger batch sizes.

Batch Size	Iterations	Test Error
1	51300	1.1535088537521576
100	17700	0.9842315688825469
10000	7700	1.3005863524433212
1000000	5810	1.072246350005552

Table 1: Performance Metrics for Different Batch Sizes

2.3 Plots

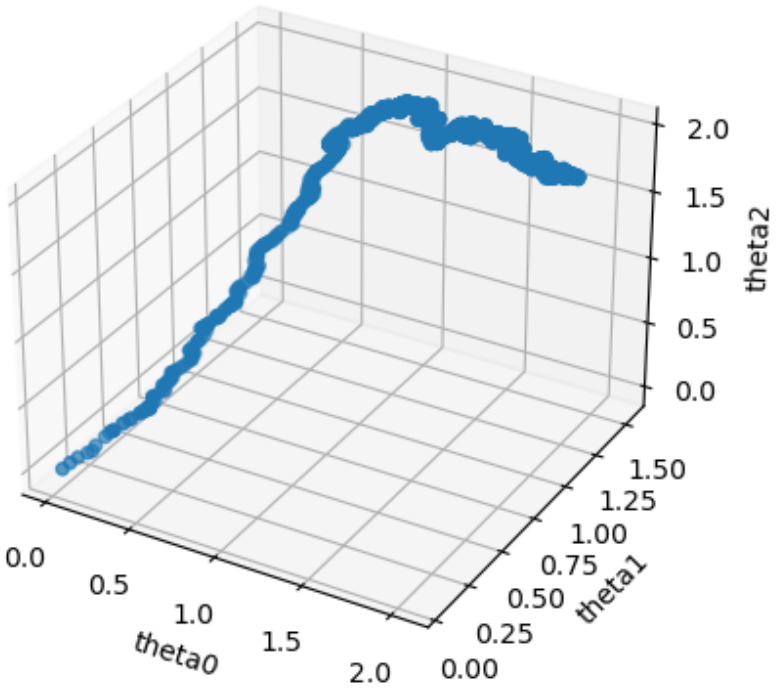

Figure 7: r = 1

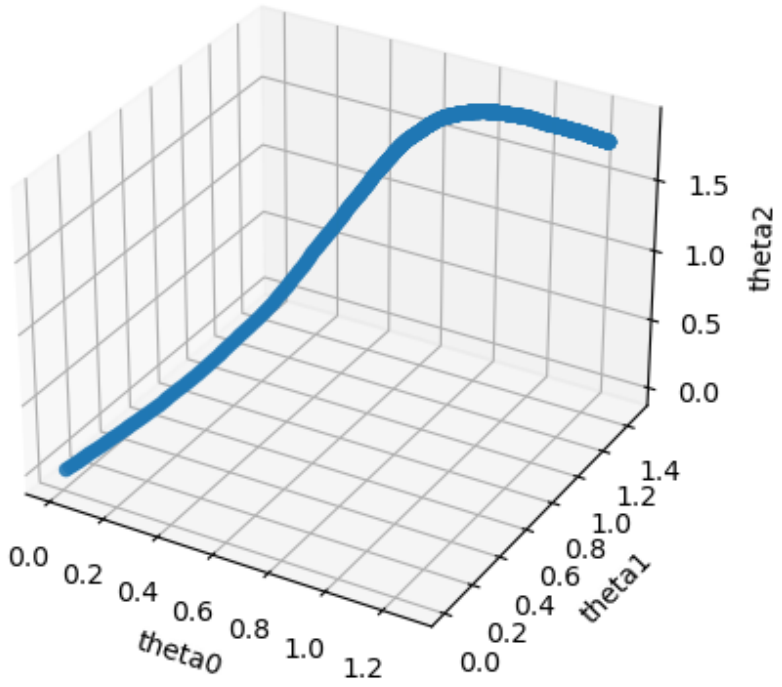

Figure 8: r = 100

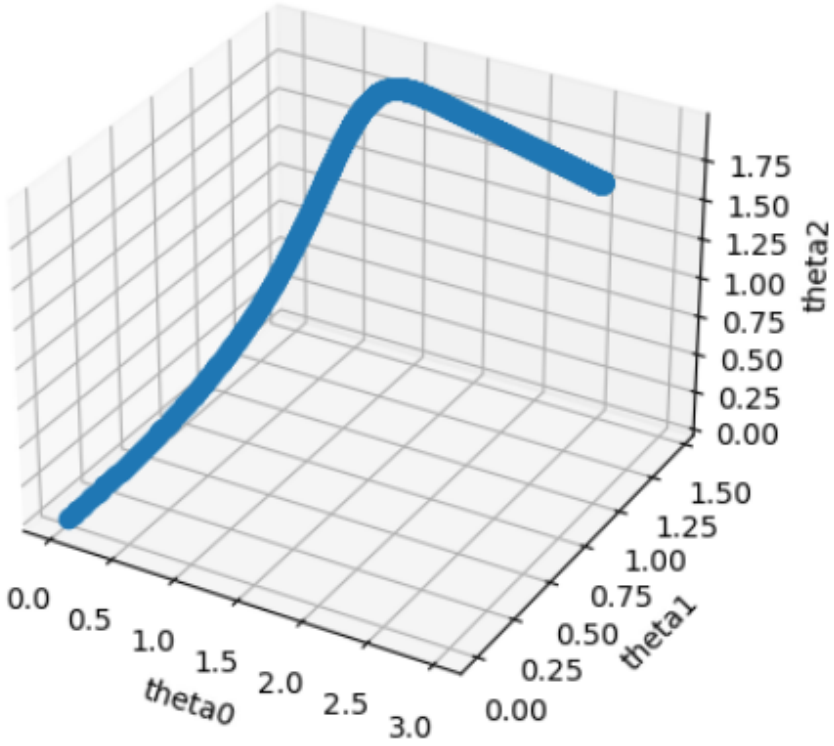

Figure 9: r = 10000

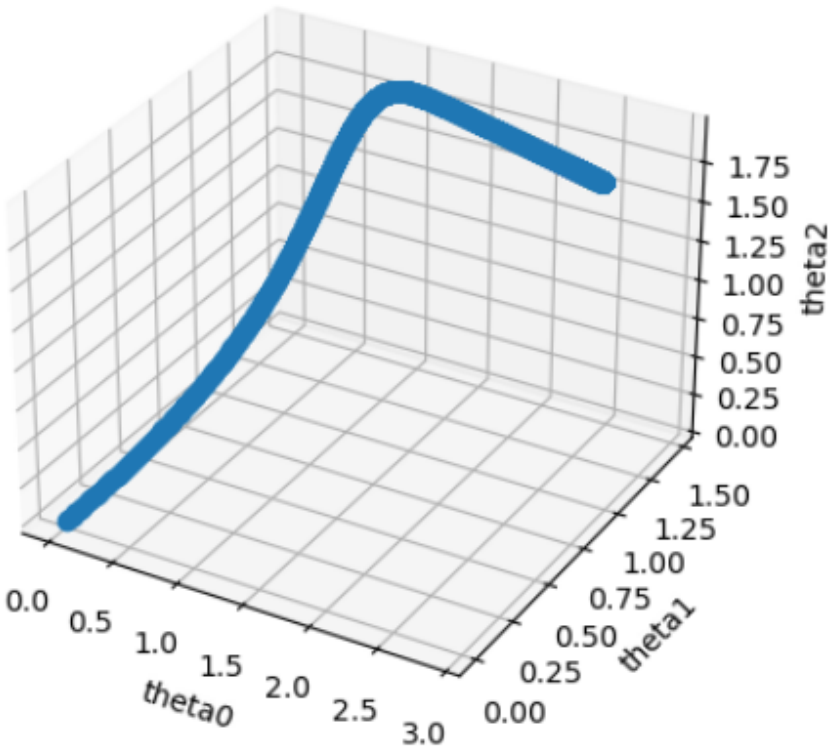

Figure 10: r = 1000000

In the first case, a small batch size resulted in significantly larger iterations. As batch size increased, the trend consistently showed a decrease in the number of iterations. This phenomenon is primarily attributed to larger batches enabling more informed decisions during gradient descent, reducing the likelihood of incorrect or random steps and allowing for direct progress toward optimal values. However, the trade-off is longer execution times due to the increased computational load per iteration. While Stochastic Gradient Descent may exhibit different behavior, in our case, higher batch sizes demonstrated lower error rates, thanks to their improved accuracy in gradient descent. Any deviations from this trend were typically due to early or randomized convergence instances with smaller batch sizes.

3 Logistic Regression

3.1 Final Parameters

$$Parameter, \theta = [0.40125316, 2.5885477, -2.72558849]$$

$$\text{Stopping Criteria}, \epsilon : 0.0000000000001$$

3.2 Decision Boundary

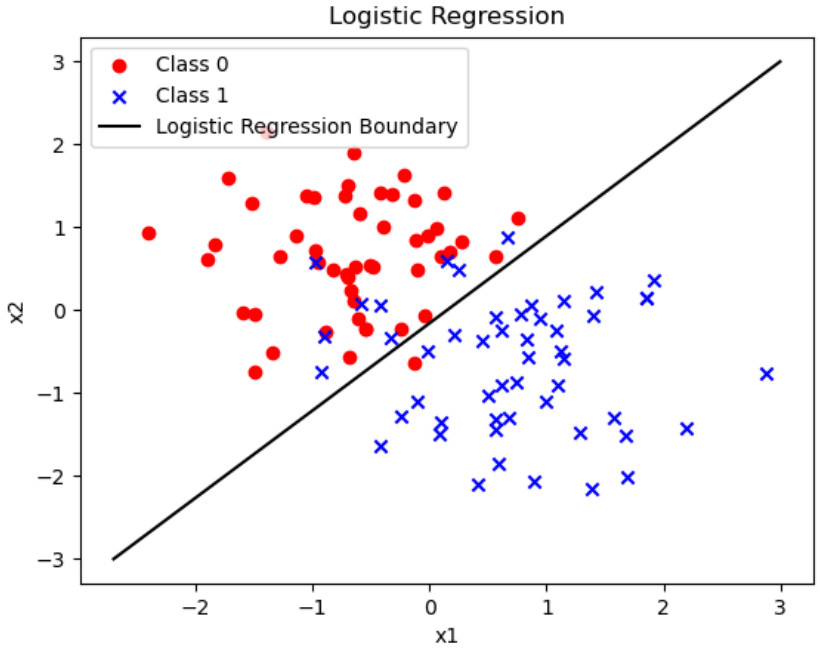

Figure 11: Logistic Regression

4 Gaussian Discriminant Analysis

4.1 Final Parameter Values With Same Covariance Matrix

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

4.2 Training Data Plot

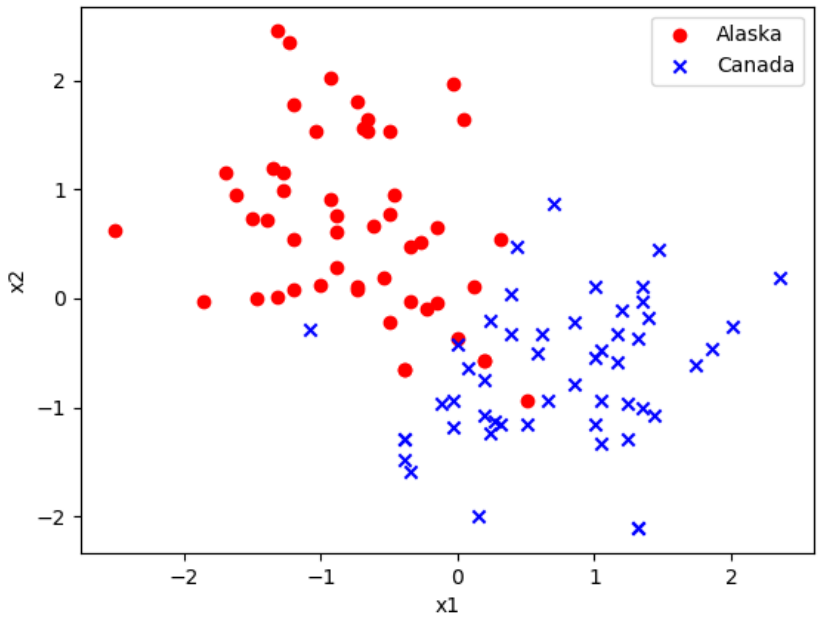

Figure 12: Input Data

4.3 Linear Decision Boundary

Linear Boundary Equation:

$$(\mu_1 - \mu_0)\Sigma^{-1}x + \log\frac{\phi}{1-\phi} + \frac{1}{2}(\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1) = 0$$

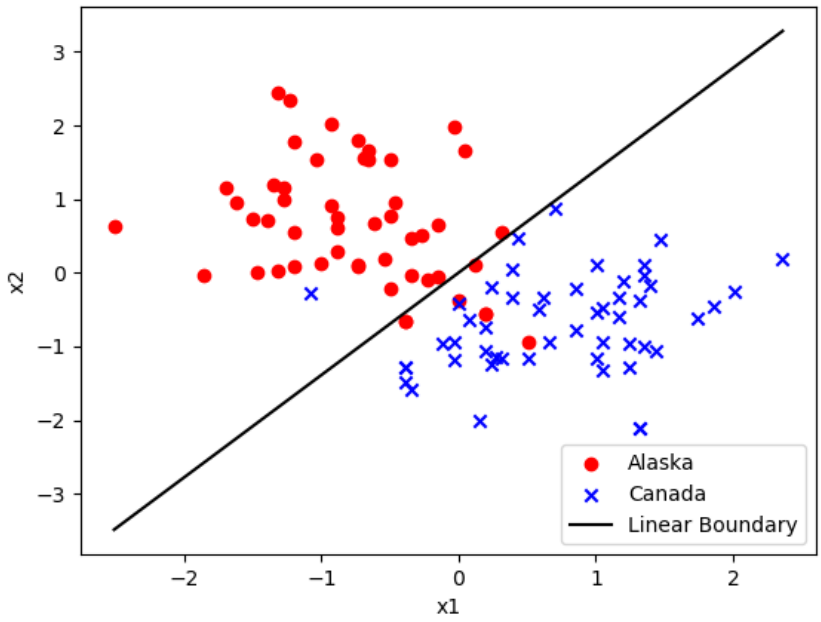

Figure 13: GDA with Linear Boundary

4.4 Final Parameter Values With Different Covariance Matrix

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

4.5 Quadratic Decision Boundary

Quadratic Boundary Equation:

$$\frac{1}{2}x^T(\Sigma_0^{-1} - \sigma_1^{-1})x + (\mu_1^T\Sigma_1^{-1} - \mu_0^T\Sigma_0^{-1})x + \log\frac{\phi}{1-\phi} + \frac{1}{2}\log\frac{|\Sigma_0|}{|\Sigma_1|} + \frac{1}{2}(\mu_0^T\Sigma_0^{-1}\mu_0 - \mu_1^T\Sigma_1^{-1}\mu_1) = 0$$

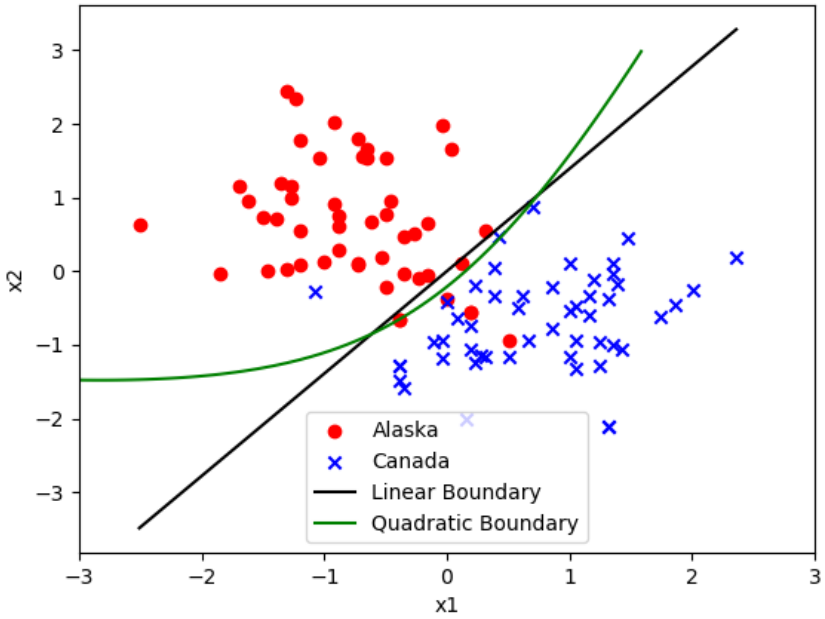

Figure 14: GDA with Quadratic and Linear Boundary

4.6 Analysis of Linear and Quadratic Decision Boundary

The quadratic separator outperforms the linear separator in classifying test data due to its greater flexibility. This flexibility stems from the assumption of different covariance matrices for the classes, which provides a more accurate representation of their relationships with data points. The quadratic curve better fits the training data, particularly in regions where the linear separator struggles, showcasing its adaptability. This improvement is linked to not assuming identical covariance matrices for both classes in the quadratic model, reducing simplifications. In essence, quadratic curves offer superior boundary modeling compared to linear curves, as observed in the plotted graph.

In my simulations, the quadratic curve better fits the given training data, especially in the middle portion where the linear separator misclassifies some of the Alaska Class points. The quadratic curve's flexibility allows it to adapt to these points and adjust the boundary accordingly. This improvement may be attributed to not assuming identical covariance matrices for both classes in the quadratic case, reducing a simplifying assumption.