Indian Institute of Technology, Delhi
Computer Science
Name : Divyansh Mittal
Entry No: 2020CS10342

Semester 1, 2023
Assignment 1

COL774: Machine Learning

Question 1.a

> Implement batch gradient descent method for optimizing J(θ). Choose an appropriate learning rate and the stopping criteria (as a function of the change in the value of $J(\theta)$). You can initialize the parameters as $\theta =\sim \vec{0}$ (the vector of all zeros). Do not forget to include the intercept term. Report your learning rate, stopping criteria and the final set of parameters obtained by your algorithm.

- **Learning Rate:** 0.01
- **Stopping Criteria:** $|J(\theta_{t+1}) - J(\theta_t)| \leq 10^{-8}$
- **Parameters:** $\vec{\theta} = [0.0013402, \quad 0.99662007]$

Question 1.b

> Plot the data on a two-dimensional graph and plot the hypothesis function learned by your algorithm in the previous part.

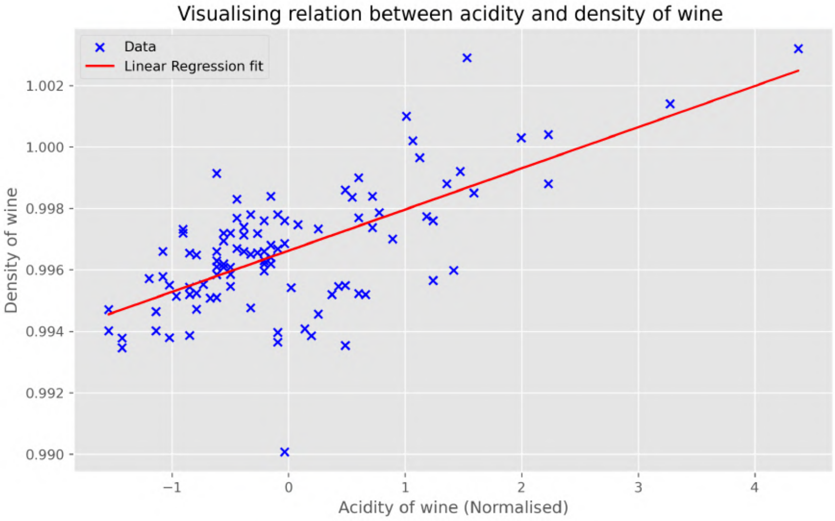


Figure 1: Data and Hypothesis Function

Question 1.c

> Draw a 3-dimensional mesh showing the error function ($J(\theta)$) on z-axis and the parameters in the x-y plane. Display the error value using the current set of parameters at each iteration of the gradient descent. Include a time gap of 0.2 seconds in your display for each iteration so that the change in the function value can be observed by the human eye.

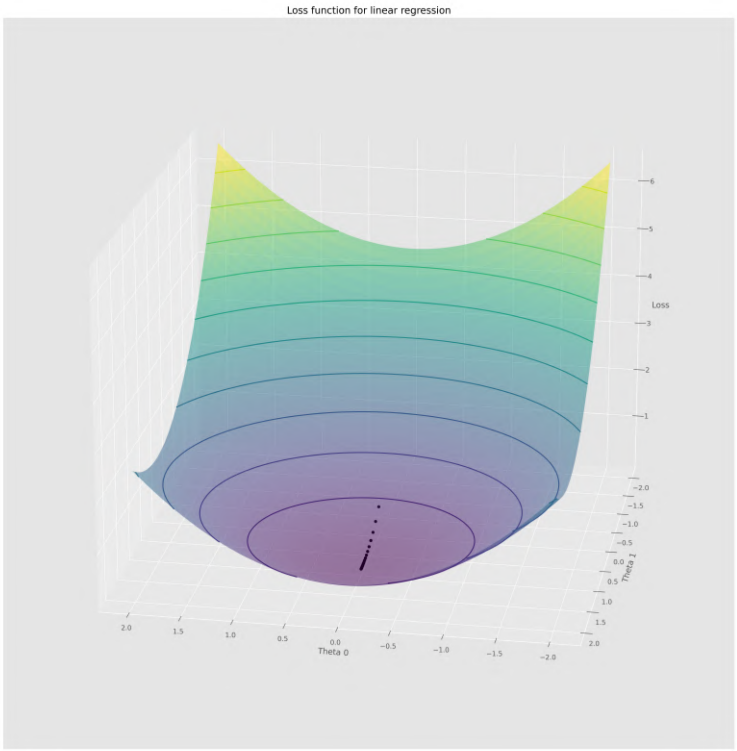


Figure 2: 3D-Loss Visualisation

Question 1.d

Repeat the part above for drawing the contours of the error function at each iteration of the gradient descent. Once again, chose a time gap of 0.2 seconds so that the change be perceived by the human eye.(Note here plot will be 2-D)

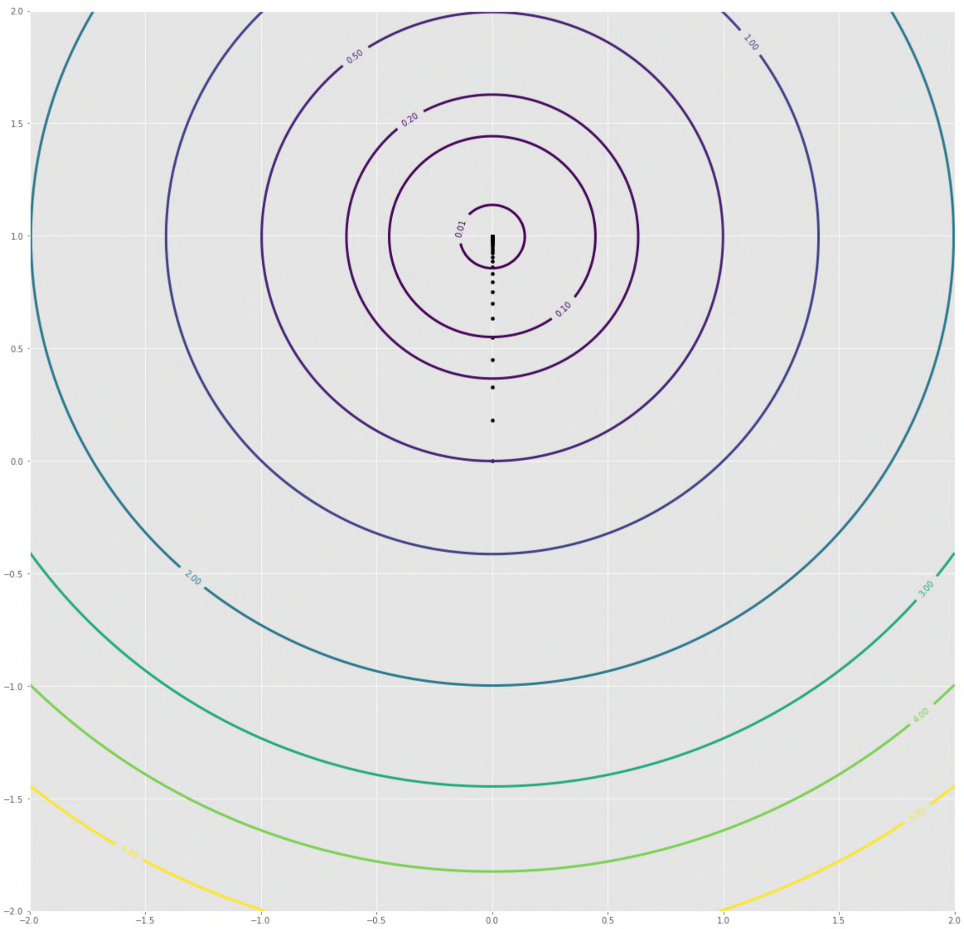
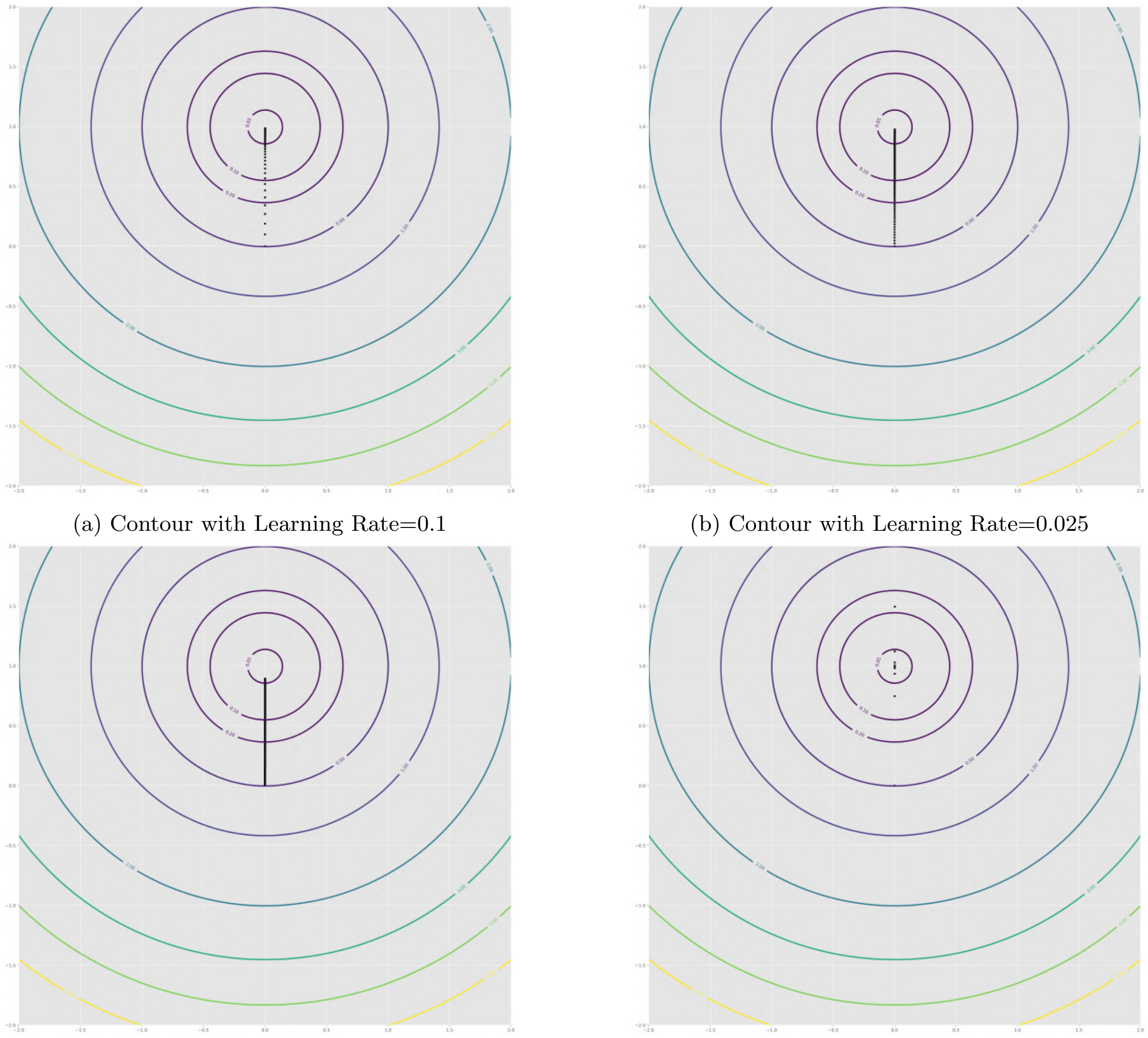


Figure 3: Contour plots

Question 1.e

Repeat the part above (i.e. draw the contours at each learning iteration) for the step size values of $\eta = \{0.001, 0.025, 0.1\}$. What do you observe? Comment.



(a) Contour with Learning Rate=0.1

(b) Contour with Learning Rate=0.025

(c) Contour with Learning Rate=0.001

(d) Contour with Learning Rate=1.5

Figure 4: Contour Plots with Different Learning Rates

For the learning rates much smaller than 1, we see that the parameter $\theta = \vec{0}$, slowly and monotonically move towards the parameters, which minimise the cost. We see, that as the learning rate decreases, the number of steps we take drastically increases. Eg for a learning rate of 0.001, the consecutive points are so close by, that they are almost indistinguishable.

When we have a learning rate, that has a large value, eg here 1.5, we see oscillations while moving towards the optima, and thus, it indicates that this learning rate is too large, for our problem.

Question 2.a

Sample 1 million data points taking values of $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix}$, $x_1 \sim \mathcal{N}(3, 4)$ and $x_2 \sim \mathcal{N}(-1, 4)$ independently, and noise variance in y, $\sigma^2 = 2$

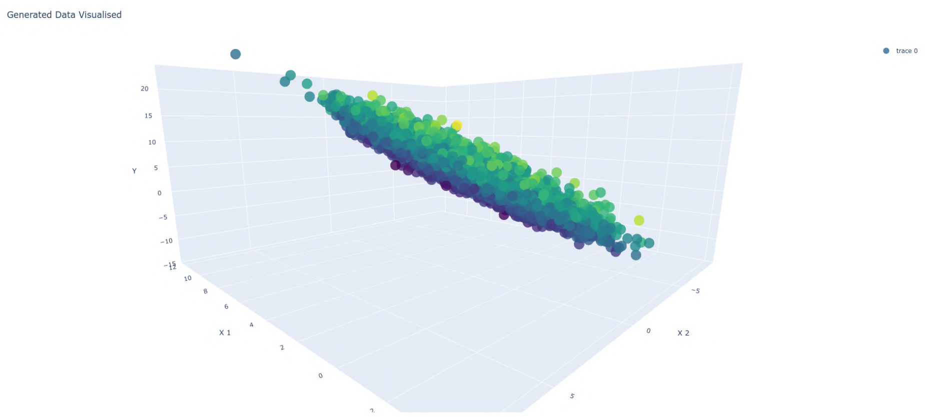


Figure 5: Subset of generated data visualised

While learning the data, I do not shuffle the data, because with shuffling the algorithm took too long.

Question 2.b

> Implement Stochastic gradient descent method for optimizing J(θ). Relearn $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{bmatrix}$ using sampled data points of part a) keeping everything same except the batch size. Keep η= 0.001 and initialize $\forall j, \theta = \vec{0}$. Report the θ learned each time separately for values of batch size $r = \{1, 100, 10000, 1000000\}$. Carefully decide your convergence criteria in each case. Make sure to watch the online video posted on the course website for deciding the convergence of SGD algorithm

1. Theta learned: [3.00560471 0.99705032 2.02474168] for batch size: 1, with a threshold of: 0.0005, by taking average of over 1000 batches in 193 iterations [1]
2. Theta learned: [2.99813167 1.00271165 1.99749511] for batch size: 100, with a threshold of: 0.0005, by taking average of over 1000 batches in 25 iterations
3. Theta learned: [2.92334463 1.01793219 1.99488172] for batch size: 10000, with a threshold of: 5e-05, by taking average of over 100 batches in 132 iterations
4. Theta learned: [2.96490518 1.00892165 1.99797948] for batch size: 1000000, with a threshold of: 1e-07, in 16034 iterations

Question 2.c

> Do different algorithms in the part above (for varying values of r) converge to the same parameter values? How much different are these from the parameters of the original hypothesis from which the data was generated? Comment on the relative speed of convergence and also on number of iterations in each case. Next, for each of learned models above, report the error on a new test data of 10,000 samples provided in the file named q2test.csv. Note that this test set was generated using the same sampling procedure as described in part (a) above. Also, compute the test error with respect to the prediction of the original hypothesis, and compare with the error obtained using learned hypothesis in each case. Comment.

- For test data error is 1.01130246422 for batch size: 1, threshold: 0.0005, by taking average of over 1000 batches in 193 iterations

[1] Here I call 1 iteration as going over 1 entire window over which average is taken. To convert it into epochs just do $\frac{\text{Batch size} \cdot \text{Average over}}{\text{Total Data}}$

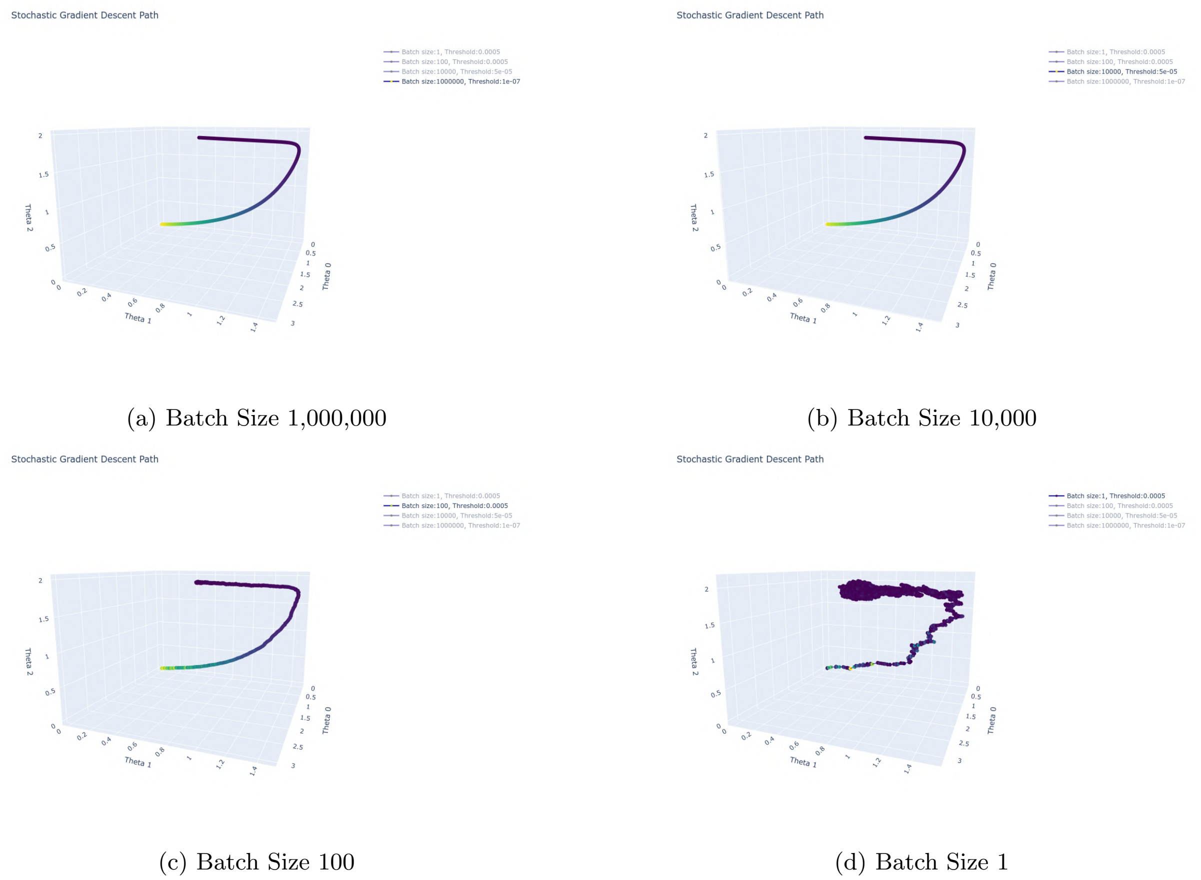
- For test data error is 0.98382959683 for batch size: 100, threshold: 0.0005, by taking average of over 1000 batches in 25 iterations
- For test data error is 1.00170744856 for batch size: 10000, threshold: 5e-05, by taking average of over 100 batches in 132 iterations
- For test data error is 0.98733342229 for batch size: 1000000, threshold: 1e-07 in 16034 iterations

With the original hypothesis, the error is 0.9829469215

For the different values of r, by carefully choosing the threshold, we can learn theta which is very close to the original parameters, from which the data was generated. But if we did not tune this threshold properly, the convergence and convergence to the correct parameter becomes very difficult. If our threshold is too small, for smaller batch sizes like 1, it does not converge. For a larger batch size like 1M, if the threshold is too big, it stops before converging. For both extremes of batch sizes, the number of iterations required is higher.

Question

> In the 3 dimensional parameter space(θ_j on each axis), plot the movement of θ as the parameters are updated (until convergence) for varying batch sizes. How does the (shape of) movement compare in each case? Does it make intuitive sense? Argue.



(a) Batch Size 1,000,000

(b) Batch Size 10,000

(c) Batch Size 100

(d) Batch Size 1

Figure 6: Path of stochastic descent for the different batch sizes

It makes intuitive sense. The larger the batch size gets, the smoother is the convergence. For a batch size = 1, we have the most jagged path that the algorithm takes.

Question 3.a

> The files logisticX.csv and logisticY.csv contain the inputs ($x^{(i)} \in \mathcal{R}^2$) and outputs ($y^{(i)} \in \{0, 1\}$) respectively for a binary classification problem, with one training example per row. Implement1 Newton's method for optimizing $\mathrm{L}(\theta)$, and apply it to fit a logistic regression model to the data. Initialize Newton's method with $\theta =\sim \vec{0}$ (the vector of all zeros). What are the coefficients θ resulting from your fit? (Remember to include the intercept term.)

The final parameters obtained are

$$\vec{\theta} = [0.40125316, 2.5885477, -2.72558849]$$

With logistic regression, we reach these parameters very quickly, and the value converges within 8 iterations, after which the difference in the cost function is less than 10^{-10}

Question 3.b

> Plot the training data (your axes should be x1 and x2 , corresponding to the two coordinates of the inputs, and you should use a different symbol for each point plotted to indicate whether that example had label 1 or 0). Also plot on the same figure the decision boundary fit by logistic regression. (i.e., this should be a straight line showing the boundary separating the region where $h(x) > 0.5$ from where $h(x) \leq 0.5$.)

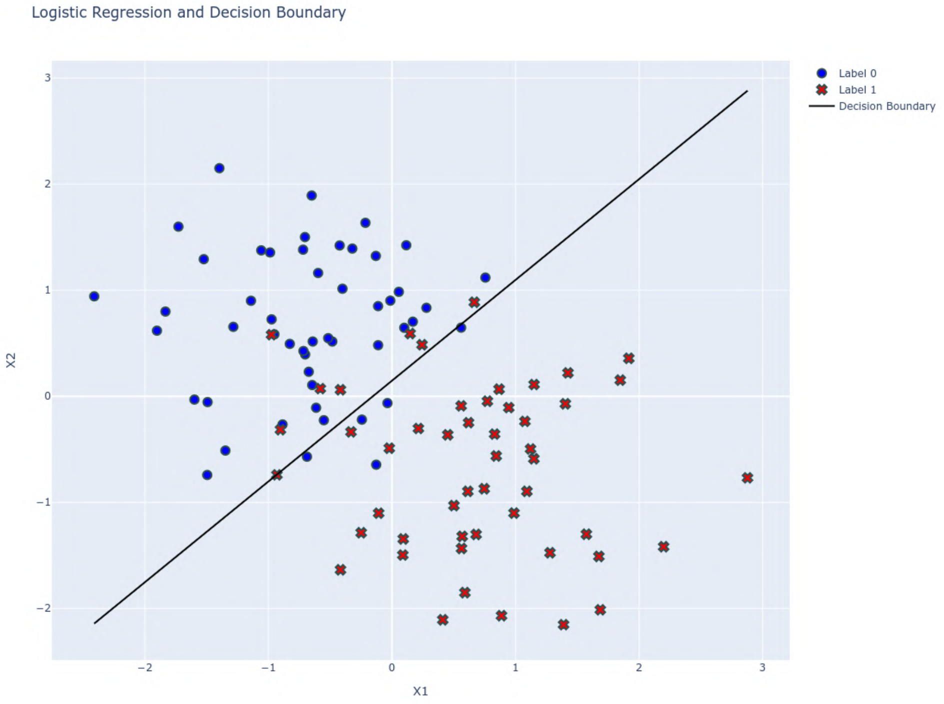


Figure 7: Logistic Regression

Question 4.a

> Implement Gaussian Discriminant Analysis using the closed form equations described in class. Assume that both the classes have the same co-variance matrix i.e. $\Sigma_0 = \Sigma_1 = \Sigma$. Report the values of the means, μ_0 and μ_1, and the co-variance matrix Σ

1. μ_0 = [-0.75529433, 0.68509431]

2. μ_1 = [0.75529433, -0.68509431]

3. $\Sigma = \begin{bmatrix} 0.42953048 & -0.02247228 \\ -0.02247228 & 0.53064579 \end{bmatrix}$

Question

Plot the training data corresponding to the two coordinates of the input features, and you should use a different symbol for each point plotted to indicate whether that example had label Canada or Alaska.

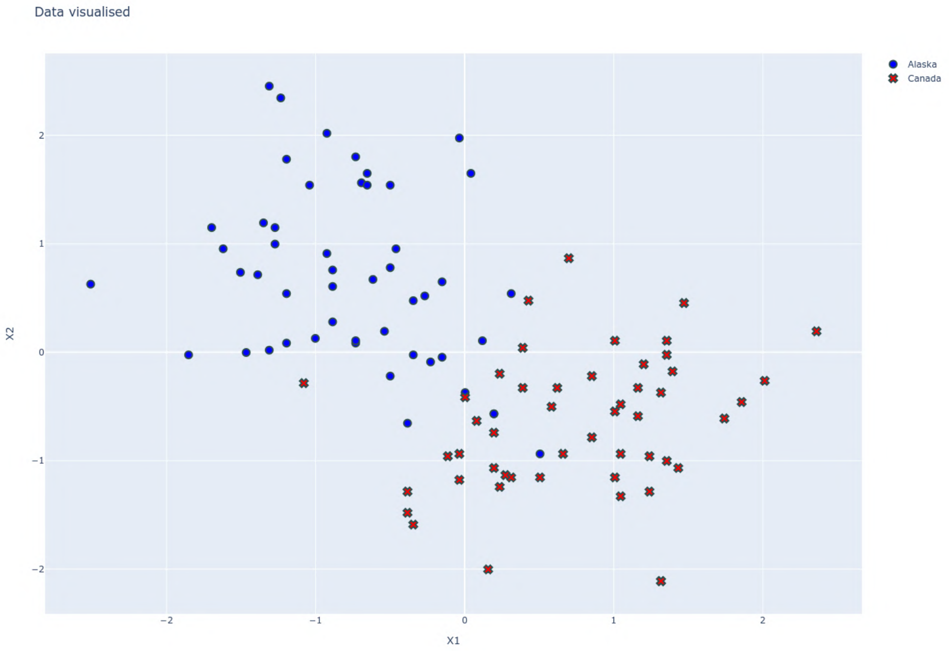


Figure 8: Data Visualised

Question

Describe the equation of the boundary separating the two regions in terms of the parameters μ_0, μ_1, and Σ. Recall that GDA results in a linear separator when the two classes have identical covariance matrix. Along with the data points plotted in the part above, plot (on the same figure) decision boundary fit by GDA

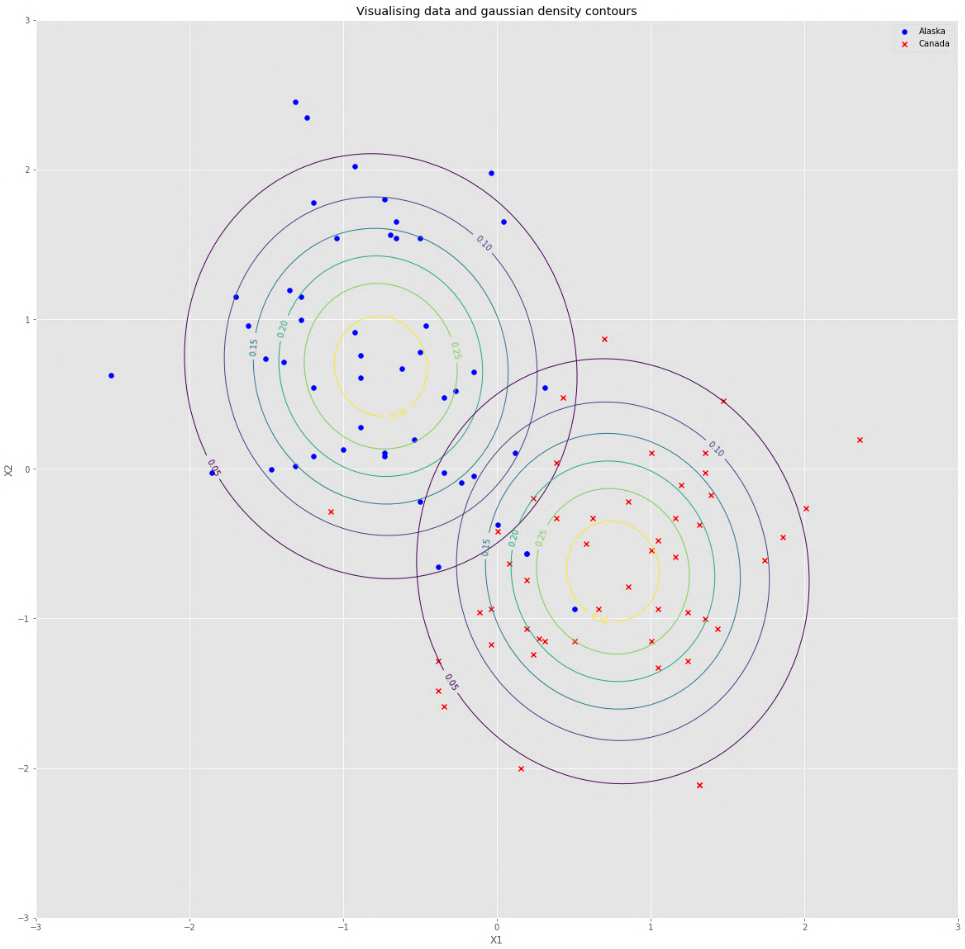


Figure 9: Contour Visualised

Using the premise of GDA, we know that

$$p(x|y=y^{(i)}) \sim \mathcal{N}(\mu_{y^{(i)}}, \Sigma) \to \frac{1}{\sqrt{(2\pi)^m|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu_{y^{(i)}})^T\Sigma^{-1}(x-\mu_{y^{(i)}})\right)$$

The log of this quantity is

$$\begin{aligned}\log\left(p(x|y=y^{(i)})\right) &= \left(-\frac{1}{2}(x-\mu_{y^{(i)}})^T\Sigma^{-1}(x-\mu_{y^{(i)}})\right) - \frac{m}{2}\log(2\pi) - \frac{1}{2}\log\left(|\Sigma|\right) \\ &= -\frac{1}{2}\left(x^T\Sigma^{-1}x + \mu^T_{y^{(i)}}\Sigma\mu_{y^{(i)}} - 2\mu^T_{y^{(i)}}\Sigma^{-1}x\right) - \frac{m}{2}\log(2\pi) - \frac{1}{2}\log\left(|\Sigma|\right)\end{aligned}$$

and the posterior probability is given by $p(y=y^{(i)}|x) = \frac{p(x|y^{(i)})\cdot p(y=y^{(i)})}{p(x)}$ To classify a point as 0 or 1, we look if $p(y=0|x) > p(y=1|x)$ or $p(y=0|x) < p(y=1|x)$ respectively.
So let us look at the quantity and assugming $p(y=0) = p(y=1)$[2]

[2] The data provided has 50 points for each of the data class, so this is a correct assumption to make

$$
\begin{aligned}
\log\left(\frac{p(y=0|x)}{p(y=1|x)}\right) &= \log\left(\frac{\frac{p(x|y=0)\cdot p(y=0)}{p(x)}}{\frac{p(x|y=1)\cdot p(y=1)}{p(x)}}\right) \\
&= -\frac{1}{2}\left((x^T\Sigma^{-1}x + \mu_0^T\Sigma^{-1}\mu_0 - 2\mu_0^T\Sigma^{-1}x) - (x^T\Sigma^{-1}x + \mu_1^T\Sigma^{-1}\mu_1 - 2\mu_1^T\Sigma^{-1}x)\right) \\
&= \left((\mu_0 - \mu_1)^T\Sigma^{-1}\right)x + \frac{1}{2}\left(\mu_1^T\Sigma^{-1}\mu_1 - \mu_0^T\Sigma^{-1}\mu_0\right)
\end{aligned}
$$

For our decision boundary, we have $p(y=0|x) = p(y=1|x)$, and thus are the points, where the above quantity is 0. The above equation then becomes of the form $w \cdot x + c$, which is linear.

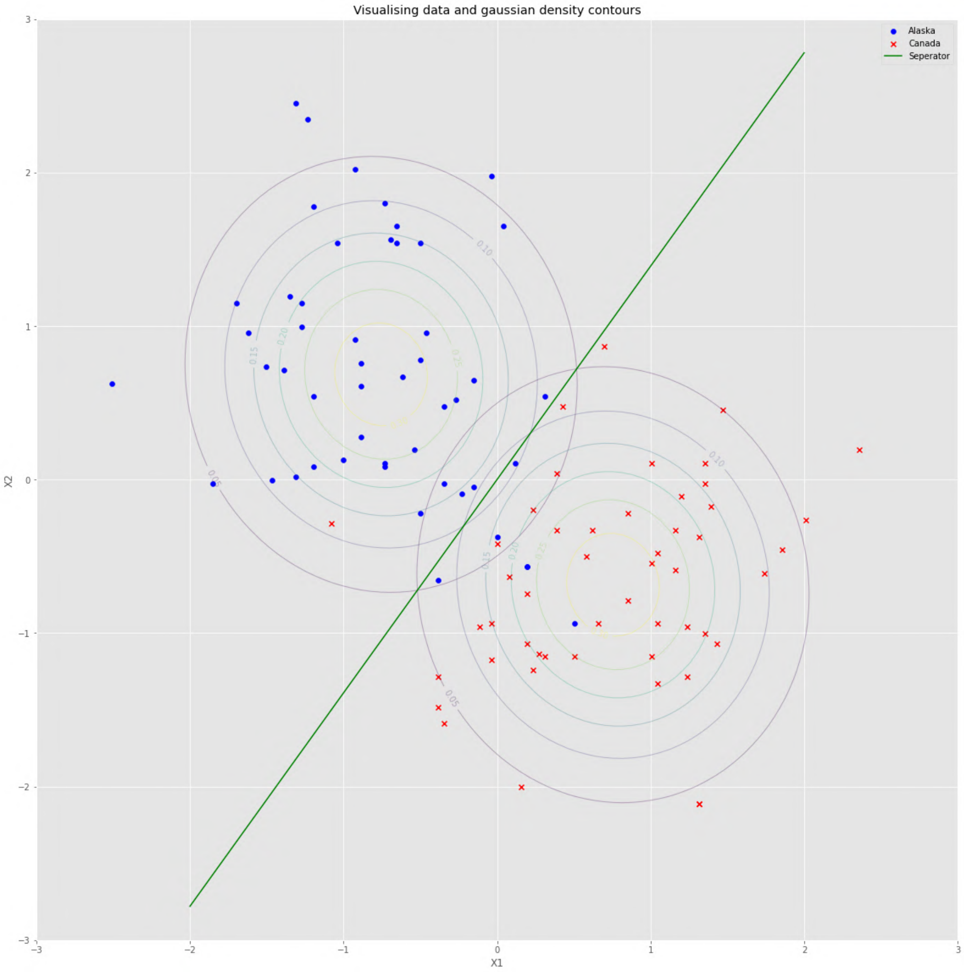


Figure 10: Plotting this seperator with the data

Question

In general, GDA allows each of the target classes to have its own covariance matrix. This results (in general) results in a quadratic boundary separating the two class regions. In this case, the maximum-likelihood estimate of the co-variance matrix Σ_0 can be derived using the equation

$$
\Sigma_0 = \frac{\sum_{i=1}^m 1\{y^{(i)} = 0\}(x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T}{\sum_{i=1}^m 1\{y^{(i)} = 0\}}
$$

And similarly, for Σ_1. The expressions for the means remain the same as before. Implement GDA for the above problem in this more general setting. Report the values of the parameter estimates i.e. $\mu_0, \mu_1, \Sigma_0, \Sigma_1$

The value for μ_0 and μ_1 remain the same and are:

1. μ_0 = [-0.75529433, 0.68509431]
2. μ_1 = [0.75529433, -0.68509431]
3. For Alaska(Σ_0): $\begin{bmatrix} 0.38158978 & -0.15486516 \\ -0.15486516 & 0.64773717 \end{bmatrix}$
4. For Canada(Σ_1): $\begin{bmatrix} 0.47747117 & 0.1099206 \\ 0.1099206 & 0.41355441 \end{bmatrix}$

Question

Describe the equation for the quadratic boundary separating the two regions in terms of the parameters μ_0, μ_1 and Σ_0, Σ_1. On the graph plotted earlier displaying the data points and the linear separating boundary, also plot the quadratic boundary obtained in the previous step.

Similar to the previous analysis, we now look at

$$\begin{aligned}
\log\left(\frac{p(y=0|x)}{p(y=1|x)}\right) &= \log\left(\frac{\frac{p(x|y=0)\cdot p(y=0)}{p(x)}}{\frac{p(x|y=1)\cdot p(y=1)}{p(x)}}\right) \\
&= -\frac{1}{2}\left((x^T\Sigma_0^{-1}x + \mu_0^T\Sigma_0^{-1}\mu_0 - 2\mu_0^T\Sigma_0^{-1}x) - (x^T\Sigma_1^{-1}x + \mu_1^T\Sigma_1^{-1}\mu_1 - 2\mu_1^T\Sigma_1^{-1}x)\right) + \frac{1}{2}\log(\frac{|\Sigma_1|}{|\Sigma_0|}) \\
&= \frac{1}{2}\left(x^T(\Sigma_1^{-1} - \Sigma_0^{-1})x\right) + \left(\mu_0^T\Sigma_0^{-1} - \mu_1^T\Sigma_1^{-1}\right)x + \frac{1}{2}\left(\mu_1^T\Sigma_1^{-1}\mu_1 - \mu_0^T\Sigma_0^{-1}\mu_0\right)
\end{aligned}$$

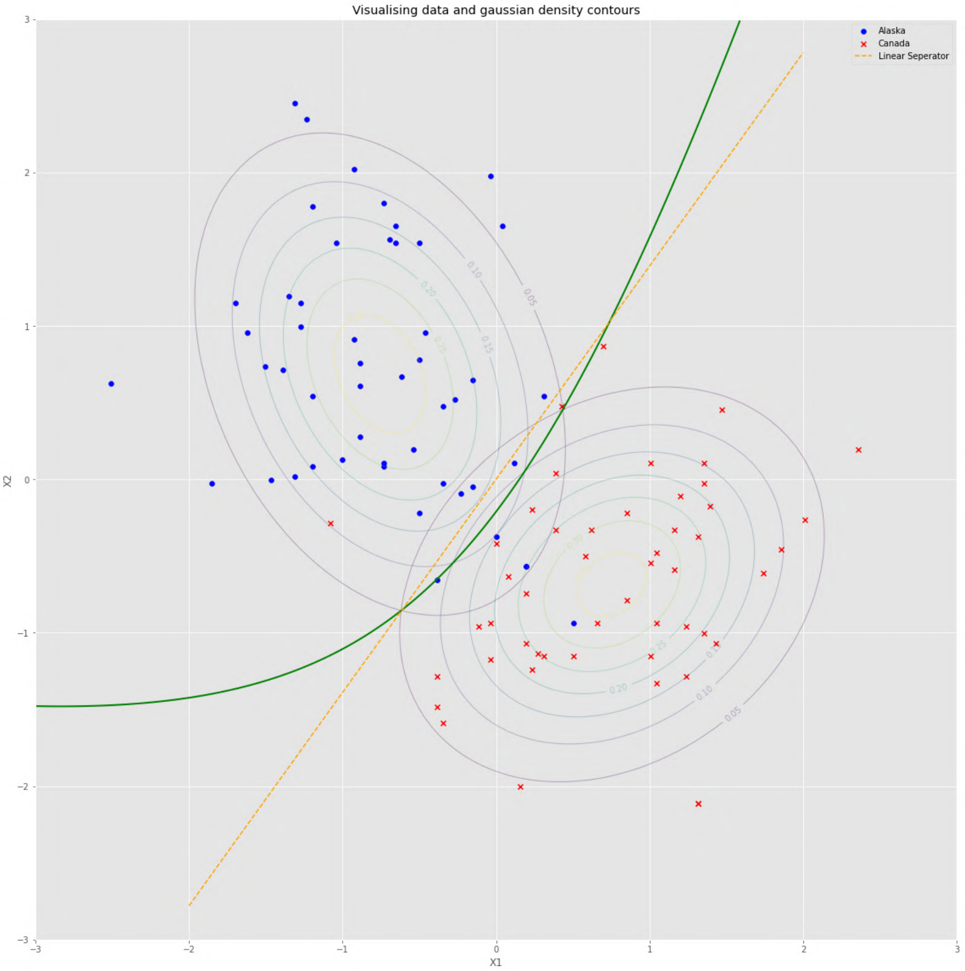


Figure 11: Quadratic Boundary

Question

> Carefully analyze the linear as well as the quadratic boundaries obtained. Comment on your observations.

We find, that the quadratic boundary is much better at capturing the seperation. The assumption of taking the same covariance matrix for both the distributions appear to be less accurate. The data seems to reveal, for Alaska, we have a higher positive correlation between the x_1 and the x_2 feature, whereas for Canada, this is slightly negative. Thus the gaussian, that fits both these distributions is much different. This is much better captured by the quadratic seperator we get. Certain points which were mis-labeled by a linear seperator, got correctly labeled by a quadratic classifier.