COL - 774 Machine Learning (Assignment -1)

Om Prakash (2019MCS2567)

Monday, 10 Feb, 2020

1 Linear Regression

1.1 Que.1(a)

Learning Rate = 0.001
Theta 0 : 0.99661694
Theta 1 : 0.00134019
Final Cost : 1.1947947960167756e-06
No. of iterations : 12656
Stopping Criteria: 0.00000000000001: (10 ^−14) (If the difference between the two consecutive values of error function J(θ) is less than 10^-14)

1.2 Que.1(b)

Hypothesis function is a straight line and can be written in the following form:
y = mx + c, where m = 0.00134019 and c = 0.99661694

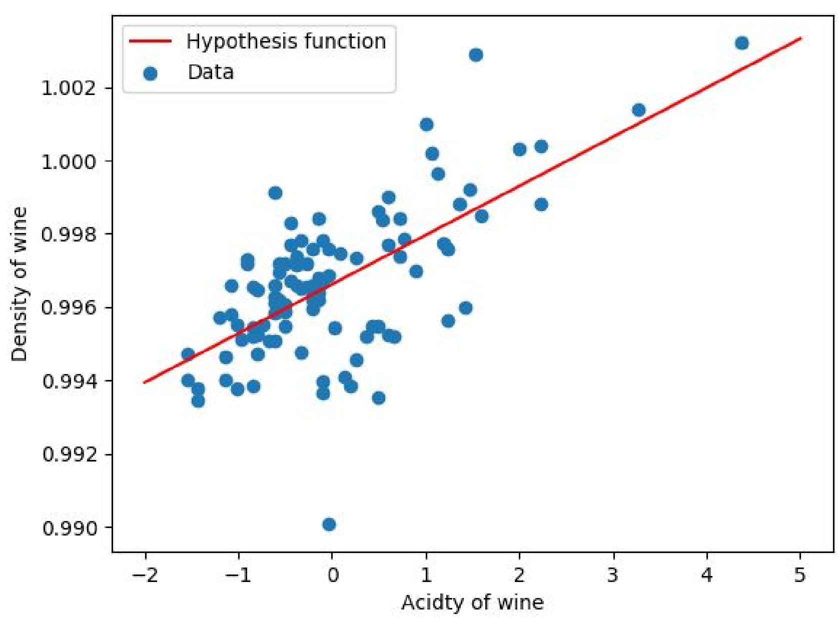


Figure 1: Linear Regression using η = 0.001

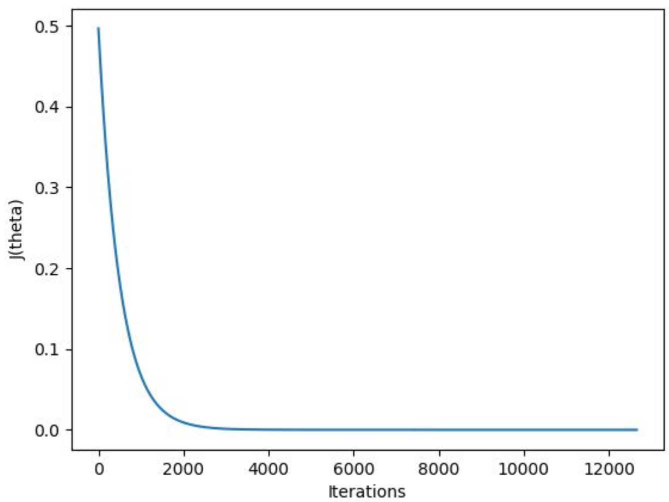


Figure 2: Error Function using θ0 = 0.99661694 and θ1 = 0.00134019

1.3 Que.1(c)

3 dimensional mesh showing the error function J(θ) on z-axis and the parameters in the x-y plane.

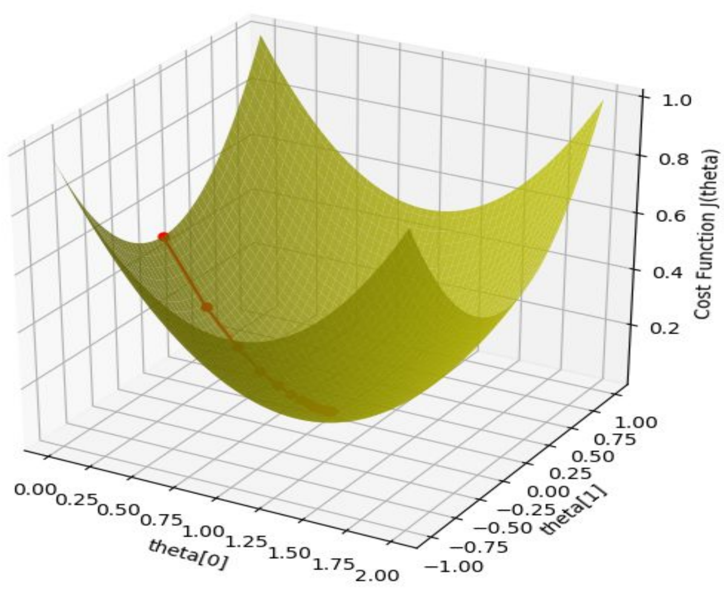


Figure 3: 3D Mesh grid and convergence of batch gradient descent for η = 0.25

1.4 Que1.(d)

The contours of the error function at each iteration of the gradient descent. Following are the learned values:

Final Theta : [0.99661997 0.0013402]

Final Cost : 1.1947898199531533e-06

No. of iterations : 55

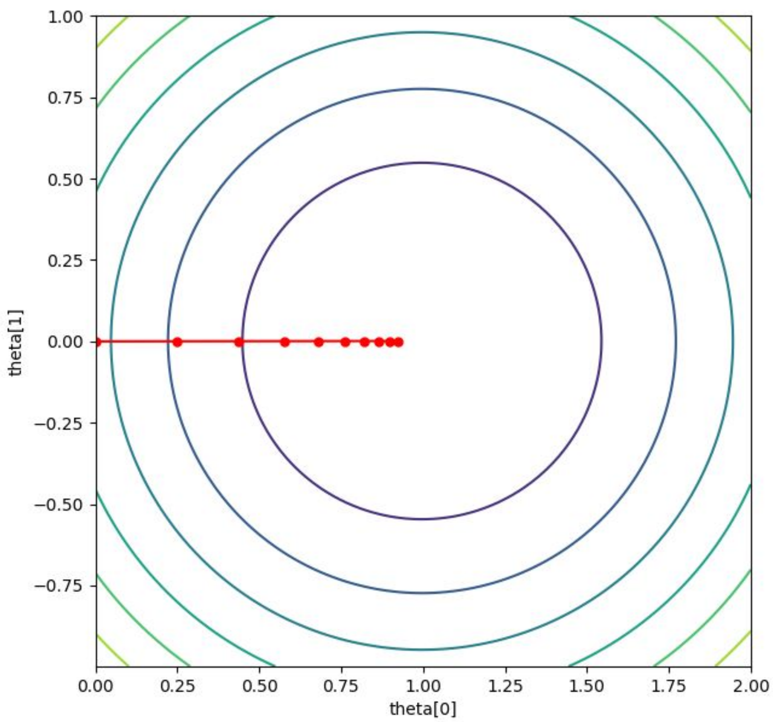


Figure 4: Contour representation of gradient descent at η = 0.25

1.5 Que1.(e)

With $\eta = 0.001$, error function converges very slowly and decreases at a very minimal rate. Following are the learned parameters:

Final Theta : [0.99661694 0.00134019]

Final Cost : 1.1947947960167756e-06

No. of iterations : 12656

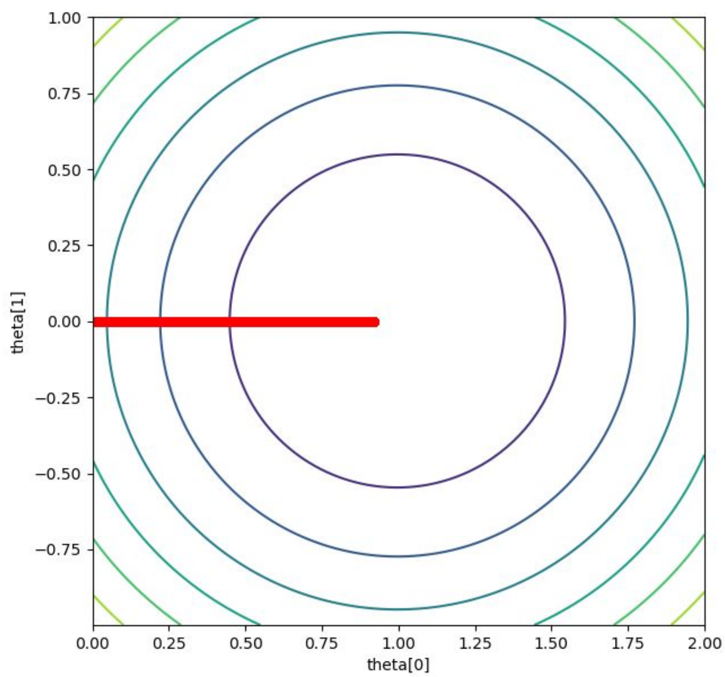


Figure 5: Contour representation of gradient descent at $\eta = 0.001$

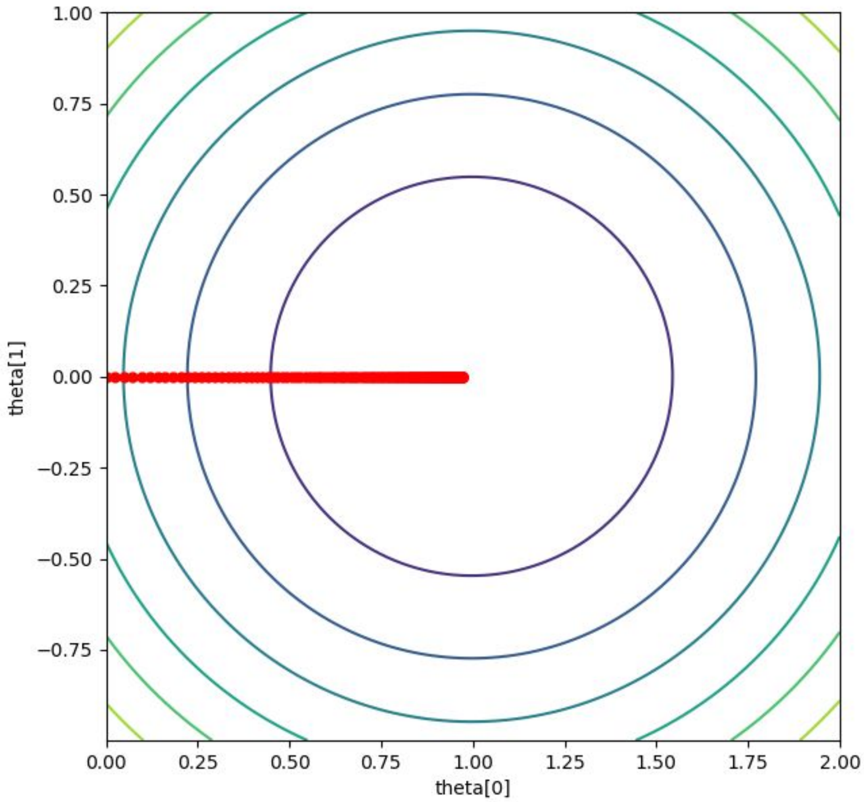


Figure 6: Contour representation of gradient descent at η = 0.025

Following are the learned parameters:

Final Theta : [0.99661949 0.0013402]

Final Cost : 1.1947899977272803e-06

No. of iterations : 565

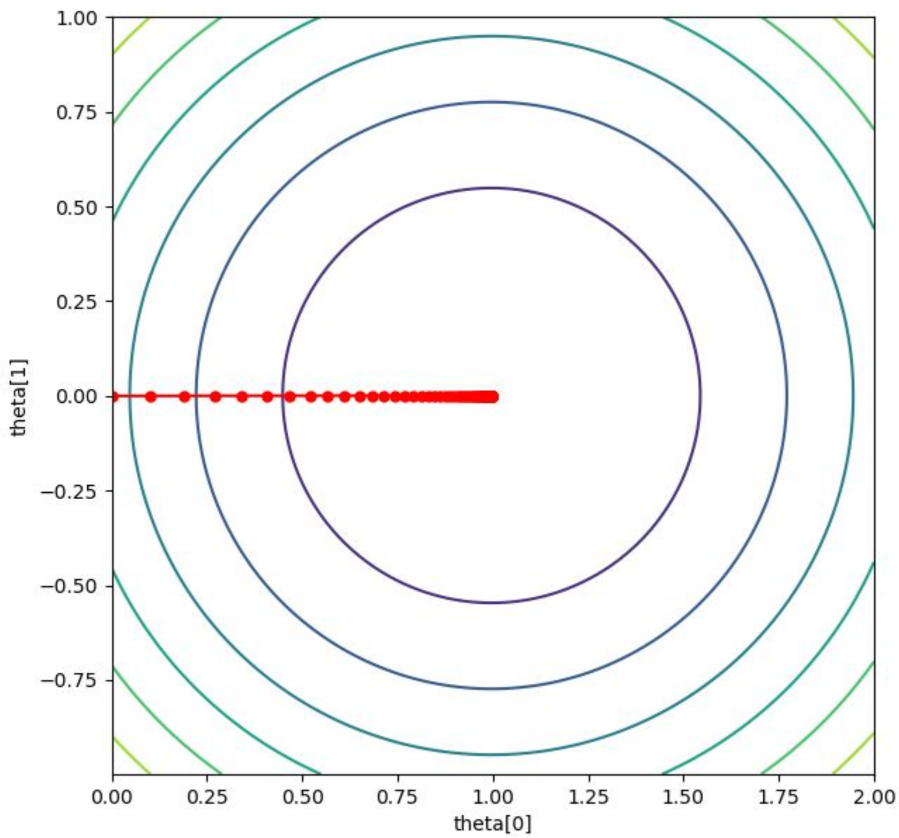


Figure 6: Contour representation of gradient descent at η = 0.1

With η = 0.1, error function converges rapidly and decreases at a very faster rate, following are the values generated:

Final Theta : [0.99661981 0.0013402]

Final Cost : 1.1947898516642842e-06

No. of iterations : 143

Conclusion:

- It has been observed if the learning rate is larger then the error function converges faster with lesser number of iterations required.
- If the learning rate is smaller then the error function converges slower and needs more number of iterations as shown above.
- If the learning rate is very high then the error function may not converge and overshoot.
- If learning parameters are too high then python may result into nan values as can't handle such big values.

2. **Stochastic Gradient Descent**

2.1 Que.2(a):

Sample 1 million data points:

```
x1 = np.random.normal(3,2,sample_count).reshape(100000,1)
x2 = np.random.normal(-1,2,sample_count).reshape(100000,1)
e =  np.random.normal(0,math.sqrt(2),sample_count).reshape(100000,1)
```

2.2 Que.2(b):

Following are the learned attributes by the SGD algorithm for different batch size.

No. of samples : 1000000 Batch Size : 1 Total Iterations: 50000
Final Theta :
[[2.99968088]
 [0.95472704]
 [1.97118111]]
Execution Time in seconds : 0.88
Converge Criteria: Max_iterations==50000 or the difference between the average cost on 1000 examples is less than 10^-3.

No. of samples : 1000000 Batch Size : 100 Total Iterations: 20000
Final Theta :
[[2.98449633]
 [1.00397983]
 [1.99754537]]
Execution Time in seconds : 0.39
Converge Criteria: max_iterations==100000 or or the difference between the average cost on 1000 examples is less than 10^-3.

No. of samples : 1000000 Batch Size : 10000 Total Iterations: 26000
Final Theta :
[[2.99101259]
 [1.0017618]
 [1.99969984]]
Execution Time in seconds : 4.09
Converge Criteria: The difference between the average cost on 1000 examples is less than 10^-6.

No. of samples : 1000000 Batch Size : 1000000 Total Iterations: 26000
Final Theta :
[[2.99776263]
 [1.00061538]
 [1.9999845]]
Execution Time in seconds : 339.5
Converge Criteria: The difference between the average cost on 1000 examples is less than 10^-6.

2.3 Que.2(c):

a. Do different algorithms in the part above (for varying values of r) converge to the same parameter values?
   Ans: No, as shown above, values are different for different r values.
b. How much different are these from the parameters of the original hypothesis from which the data was generated?
   Ans: As shown above, the difference is very minimal, they almost converged to the original theta parameters.
c. Comment on the relative speed of convergence and also on the number of iterations in each case?
   Ans:
   r = 1, converge very fast. Max number of iterations = 50,000, Epoch = 1, though it converged even in less than 50000 iterations, it took 0.88 seconds of time for convergence.
   r = 100, faster than the previous one, max iterations = 20000,Epoch = 1, it took 0.39 seconds of time for convergence.
   r = 10000, Execution Time in seconds : 4.09, Total Iterations: 26000, No of Epochs : 260
   r = 10,00,000, The slowest one, Execution Time in seconds : 339.5,
   total iterations = 26000, No of Epochs : 26000

d. Test error with respect to the prediction of the original hypothesis, and compare with the error obtained using learned hypothesis in each case?

Ans:

1. Batch Size = 1
   original cost : 0.9829469215000001
   My Theta : [2.99968088 0.95472704 1.97118111]
   My Cost : 1.1337547186188743
   Error difference : 0.15080779711887426

2. Batch Size = 100
   original cost : 0.9829469215000001
   My Theta : [2.98449633 1.00397983 1.99754537]
   My Cost : 0.9842663233575532
   Error difference : 0.001319401857553082

3. Batch Size = 10000
   original cost : 0.9829469215000001
   My Theta : [2.99101259 1.0017618 1.99969984]
   My Cost : 0.9831158377688991
   Error difference : 0.0001689162688990331

4. Batch Size = 10,00,000
   original cost : 0.9829469215000001
   My Theta : [2.99776263 1.00061538 1.9999845]
   My Cost : 0.9829446735402787
   Error difference : -2.2479597213687086e-06

3. **Logistic Regression**
Logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

Following is the log likelihood function of logistic regression:

$$L(\theta) = \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

Following is the Newton's method:

$$\theta^{(t+1)} = \theta^{(t)} - H^{-1} \nabla_\theta L(\theta)$$

3.1 Que.3(a):

Following are the values of theta which are learned by the algo:

[[0.40125316]
[2.5885477]
[-2.72558849]]

3.2 Que.3(b):

A straight line showing the boundary separating the region where h(x) > 0.5 from where h(x) ≤ 0.5.)

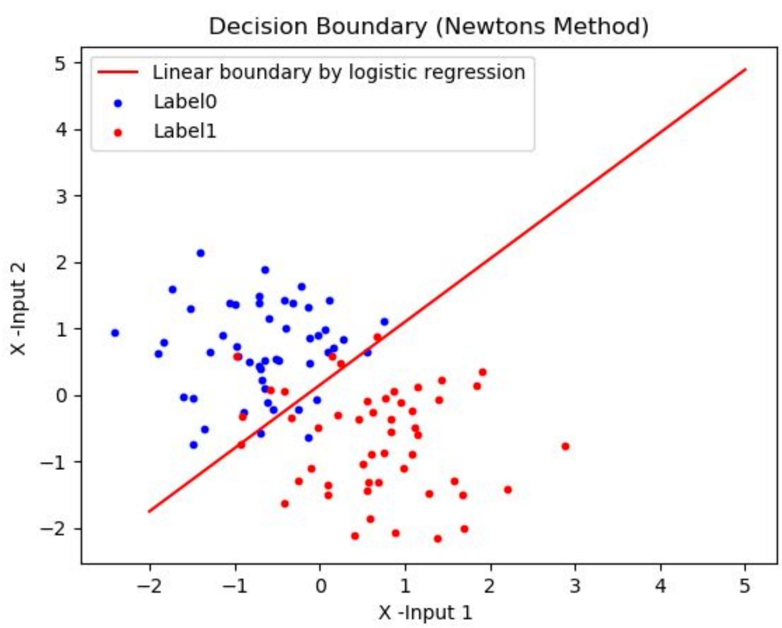


Figure 7: Logistic Regression using Newton's method

4. **Gaussian Discriminant Analysis (GDA)**
The model is:

$$\begin{aligned} y &\sim \text{Bernoulli}(\phi) \\ x|y=0 &\sim \mathcal{N}(\mu_0, \Sigma) \\ x|y=1 &\sim \mathcal{N}(\mu_1, \Sigma) \end{aligned}$$

Writing out the distributions, this is:

$$\begin{aligned} p(y) &= \phi^y(1-\phi)^{1-y} \\ p(x|y=0) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\right) \\ p(x|y=1) &= \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\right) \end{aligned}$$

The log-likelihood of the data is given by

$$\begin{aligned} \ell(\phi, \mu_0, \mu_1, \Sigma) &= \log \prod_{i=1}^{m} p(x^{(i)}, y^{(i)}; \phi, \mu_0, \mu_1, \Sigma) \\ &= \log \prod_{i=1}^{m} p(x^{(i)}|y^{(i)}; \mu_0, \mu_1, \Sigma) p(y^{(i)}; \phi). \end{aligned}$$

By maximizing ℓ with respect to the parameters, we find the maximum likelihood estimate of the parameters to be:

$$\begin{aligned} \phi &= \frac{1}{m}\sum_{i=1}^{m} 1\{y^{(i)}=1\} \\ \mu_0 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)}=0\}x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)}=0\}} \\ \mu_1 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)}=1\}x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)}=1\}} \\ \Sigma &= \frac{1}{m}\sum_{i=1}^{m}(x^{(i)}-\mu_{y^{(i)}})(x^{(i)}-\mu_{y^{(i)}})^T. \end{aligned}$$

Note that covariance matrix Σ is identical here.

4.1 Que.4(a)

Both the classes have the same covariance matrix i.e. Σ0 = Σ1 = Σ.

$$\begin{aligned}
\phi &= \frac{1}{m}\sum_{i=1}^{m} 1\{y^{(i)} = 1\} \\
\mu_0 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 0\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 0\}} \\
\mu_1 &= \frac{\sum_{i=1}^{m} 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^{m} 1\{y^{(i)} = 1\}} \\
\Sigma &= \frac{1}{m}\sum_{i=1}^{m} (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T.
\end{aligned}$$

μ0 : [-0.75529433 0.68509431]
μ1 : [0.75529433 -0.68509431]
Sigma (Σ) :
[[0.42953048 -0.02247228]
[-0.02247228 0.53064579]]

4.2 Que.4(b)

The training data corresponding to the two coordinates of the input features, Note that Alaska is considered 0 here and Canada is considered as 1.

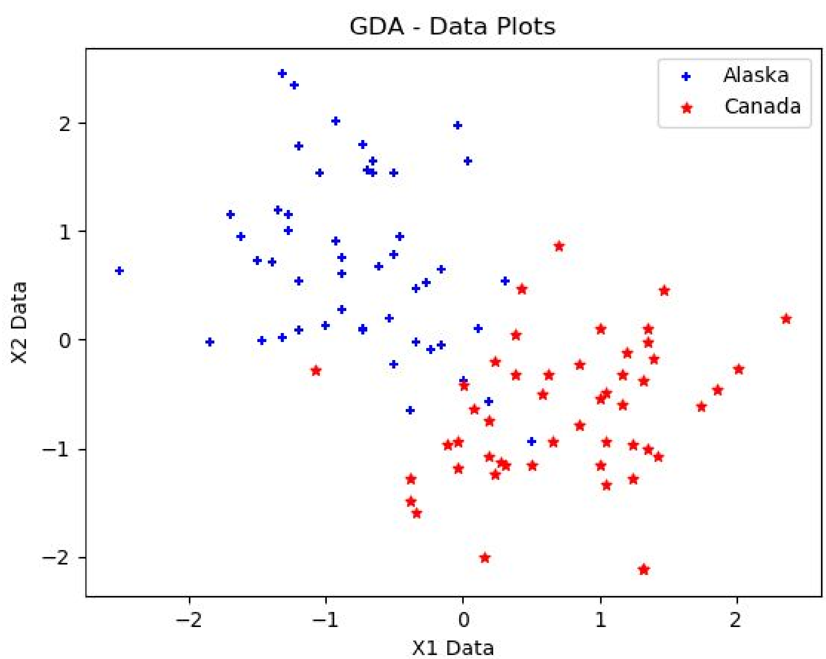


Figure 8: Training data corresponding to the two coordinates of the input features

4.3 Que.4(c)

Equation of the boundary separating the two regions in terms of the parameters μ0, μ1 and Σ, when the two classes have identical covariance matrix. Following is the equation of boundary when the two covariance matrices are the same.

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

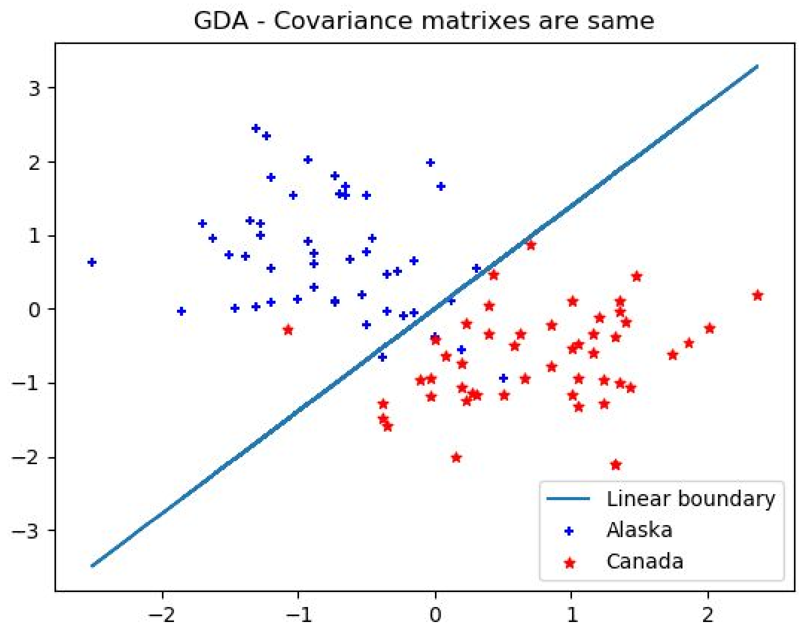


Figure 9: Linear separator when the two classes have identical covariance matrix

4.4 Que.4(d)

Following are the different parameters learned by the algorithm when the two covariance Σ0 and Σ1 matrices are the same.

Σ0 : [[0.38158978 -0.15486516]
[-0.15486516 0.64773717]]

Σ1 : [[0.47747117 0.1099206]
[0.1099206 0.41355441]]

μ0 : [[-0.75529433 0.68509431]]
μ1 : [0.75529433 -0.68509431]

4.5 Que.4(e)

The maximum-likelihood estimate of the covariance matrix Σ0 can be derived using the equation:

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

It's the same for Σ1 but Σ0 will be replaced with Σ1.

The equation for the quadratic boundary separating the two regions:Σ1

$$\frac{1}{2}X^T(\sum_1^{-1} - \sum_0^{-1})X + (\mu_0^T \sum_0^{-1} - \mu_1^T \sum_1^{-1})X - (\log(\frac{\phi}{1-\phi}) + \frac{1}{2}\log(\frac{|\sum_0|}{|\sum_1|})) + \frac{1}{2}(\mu_1^T \sum_1^{-1} \mu_1 - \mu_0^T \sum_0^{-1} \mu_0) = 0$$

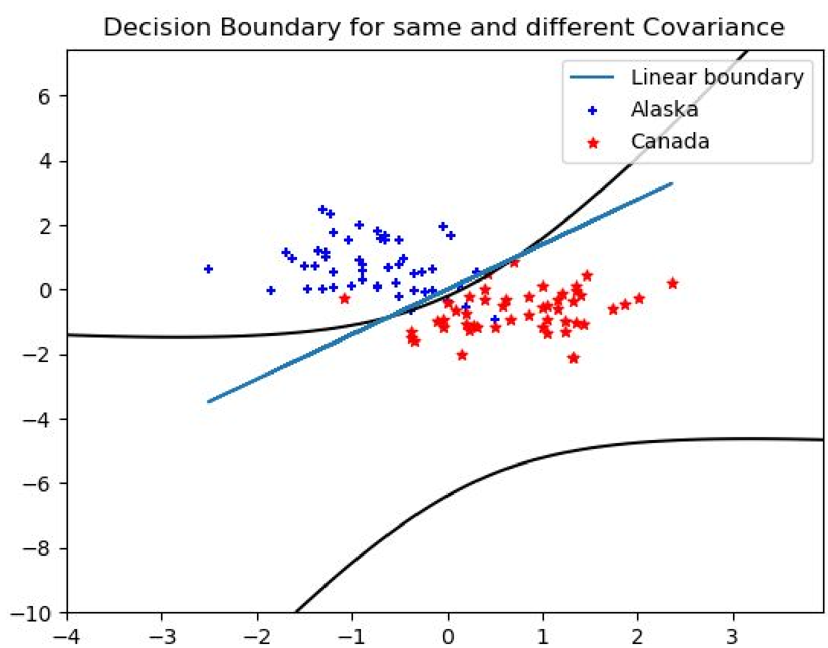
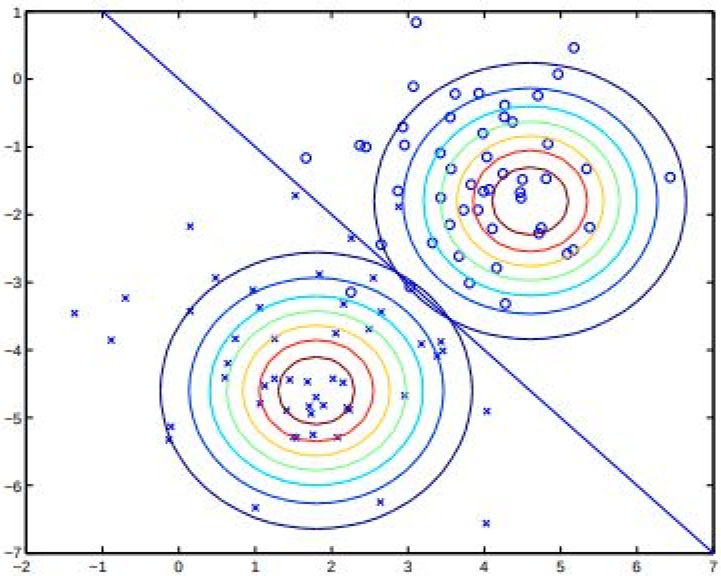


Figure 10: Quadratic boundary the covariance matrix is different

4.6 Que.4(f)



Note that the two Gaussians have contours that are the same shape and orientation, since they share a covariance matrix Σ, but they have different means μ0 and μ1. Also shown in the figure is the straight line giving the decision boundary at which p(y = 1|x) = 0.5. . On one side of the boundary, we'll predict y = 1 to be the most likely outcome, and on the other side, we'll predict y = 0.
The quadratic boundary by GDA offers a non linear way of classifying data. Although it computes an additional sigma parameter, it provides a better estimate of the decision boundaries.

Reference: Andrew NG Notes