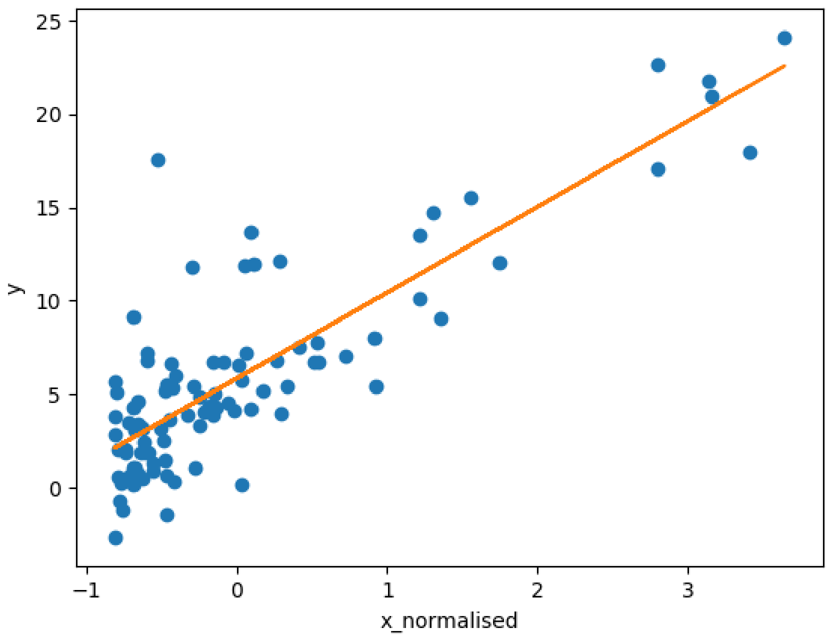
Q1.

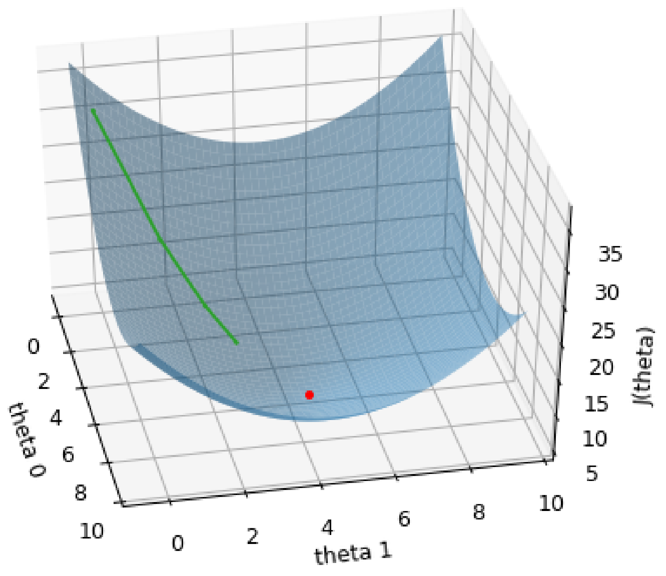
a)

- **Learning Rate:** Anything between 1e-2 to 1.9 converges in a few seconds.
- **Stopping criteria:** Norm of Gradient of Loss <1e-10 works well (anything between 1e-2 and 1e-10 also works equally well)
- **Final parameters obtained:** y = 5.83905413026 + 4.59297748099 x_normalised ($\theta_0 + \theta_1 x$)

b)

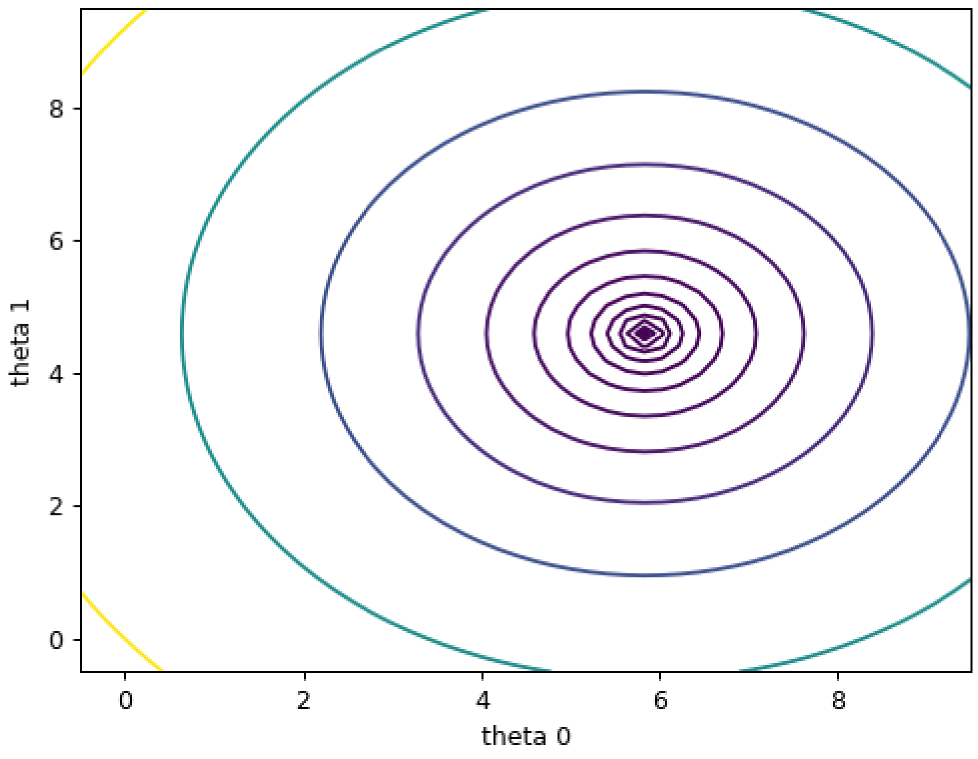


c)



Red point is the minima. Green path is the descent. (Animation in iPython Notebook)

d)
One instance of the contour map (Animations in iPython notebook).



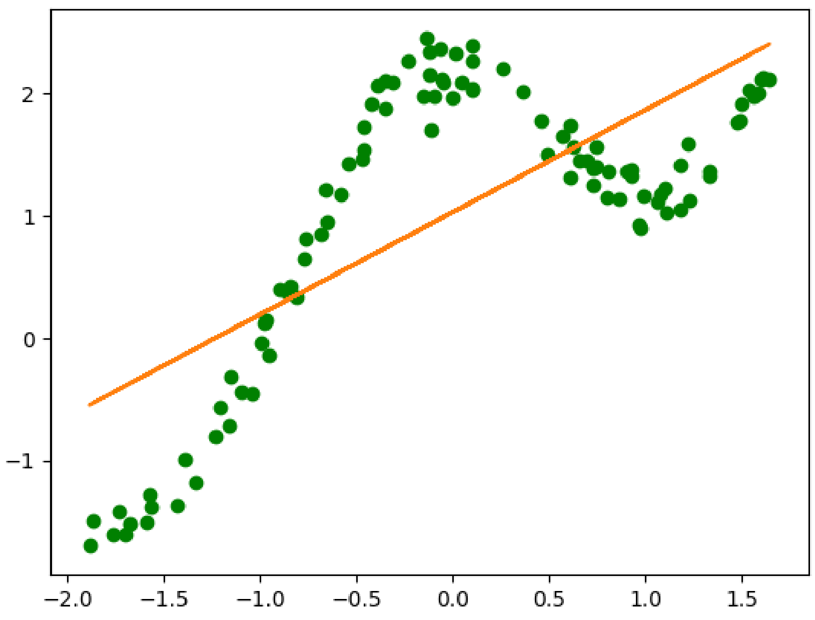
e) By looking at the contour animations (For animations see iPython notebook) we observe:

- For a learning rate like 1, the loss descends very rapidly and converges in 3 to 4 steps.
- For 0.1, 0.5, 0.9, 1.3, the contours ALWAYS go down (radius decreases between consecutive steps), but in different ways. For higher LRs, the loss oscillates around the minima and for lower LRs, the loss gradually goes towards minima.
- The gap between successive contour lines goes down as we converge (because of our optimisation rule). For LR ~ 1, The gap between successive contour lines is high but for 1.3 or 0.1 it is very small because gradient descent is very slow for small LR, and it oscillates a lot for high LR).
- For 2.1 and 2.5 we see that the contour lines increase in radius (i.e. the loss diverges). This means that the LR is too high and should be reduced.

Q2

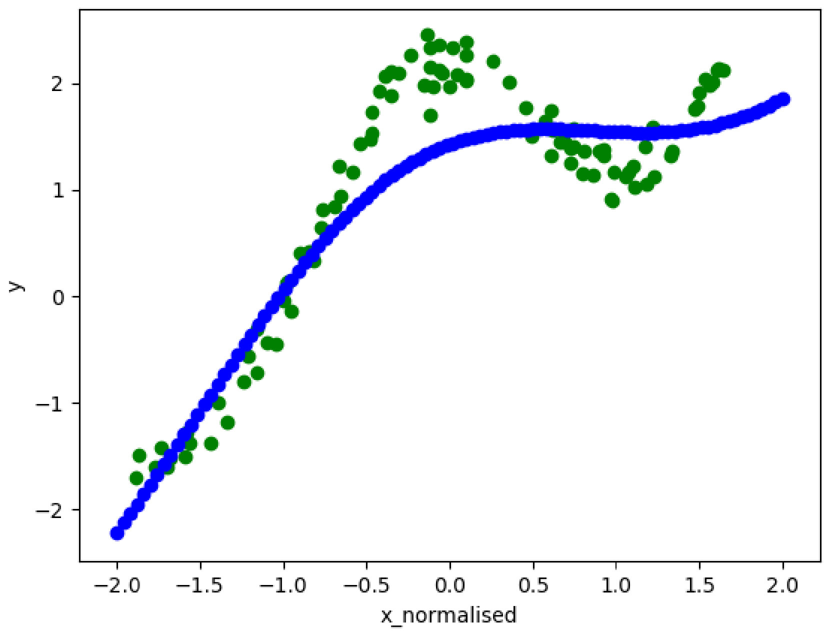
a)

Using unweighted regression: y = 0.835188895313 x_normalised + 1.03127983079

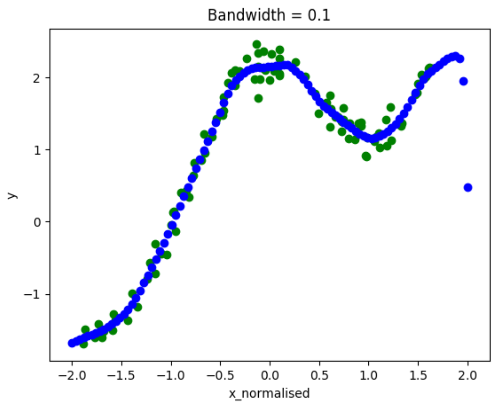


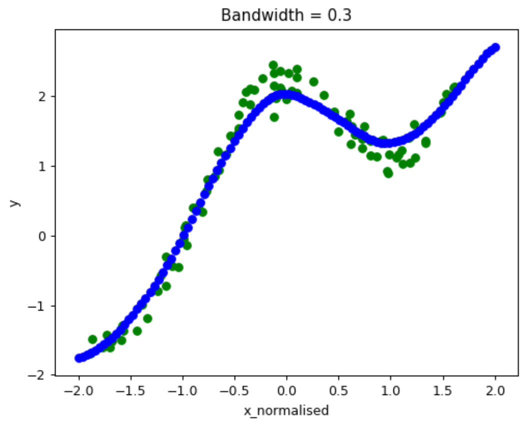
b)

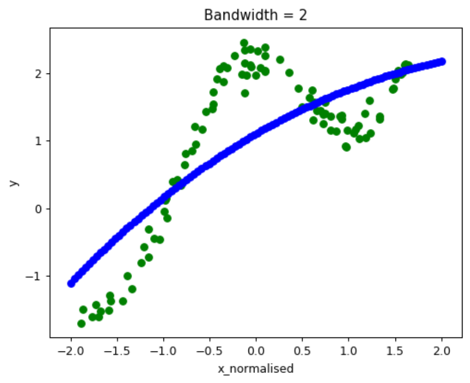
After implementing locally weighted linear regression and plotting at points uniformly sampled in range of train data: (Bandwidth = 0.8)

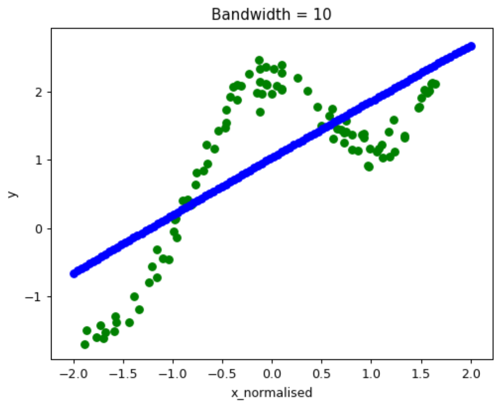


c) Plotting for different bandwidths:









We can see that for small bandwidth, the predictions for test data overfit badly. They fail to capture global properties of data and just replicate the y of nearest points. This will probably not generalize well to test data. It is also highly susceptible to local noise.

For high bandwidths, the prediction plot becomes almost equivalent to unweighted linear regression. It does not capture the local properties of data at all. This is again bad as the data distribution given is clearly non linear.

For bandwidth between the extremes, we see that the curve captures local properties of data without overfitting much. Hence a lot of hyperparameter tuning is necessary in this method (also validation on a held out set).

I think the value of 0.4 to 0.7 is the best for this data.

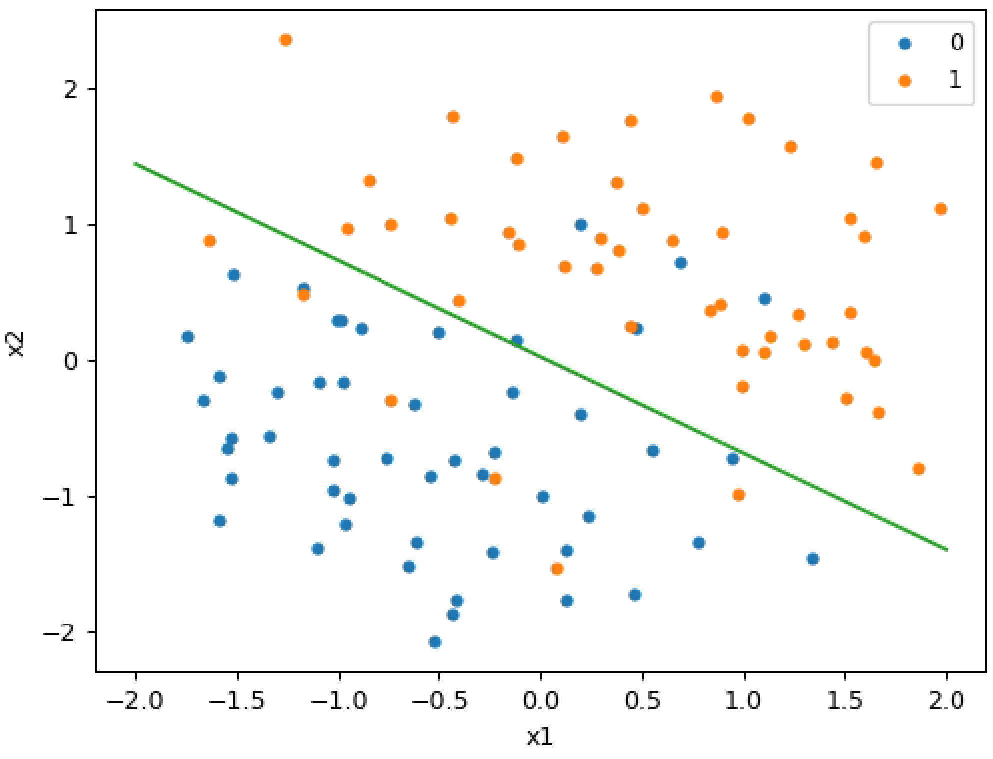
Q3.

a)

The equation of the separating line is:

$$-0.0471757732513 + 1.46005895612\ x1 + 2.06586134271\ x2 = 0$$

b)

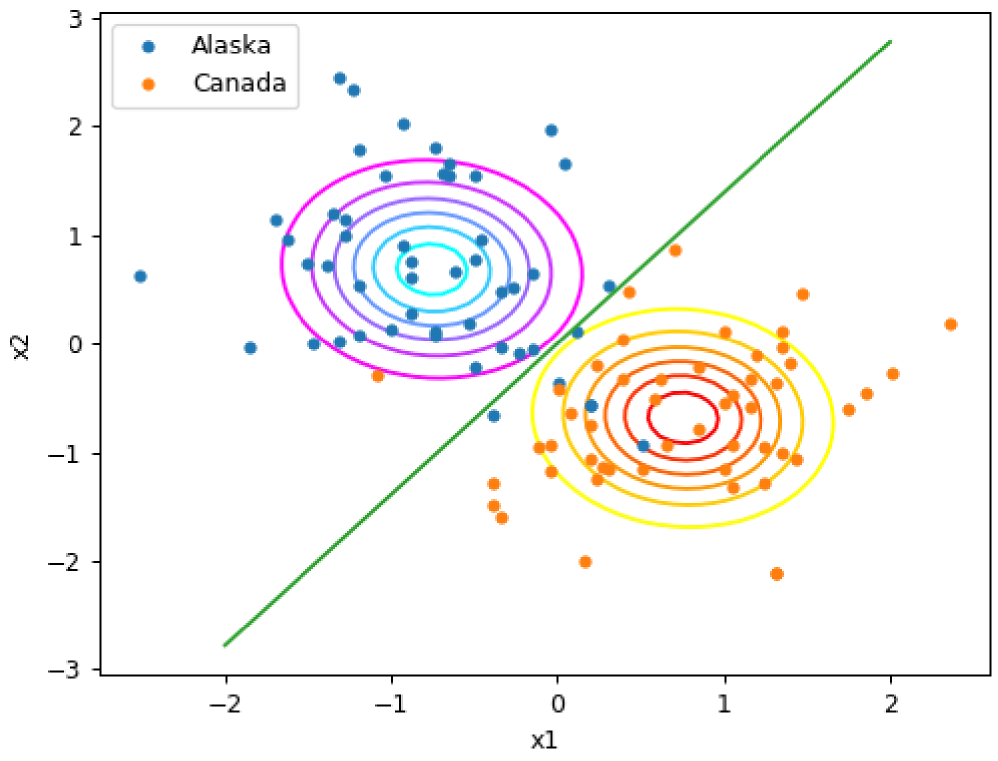


Q4

a)

Phi = 0.5
Mu_0 = [-0.75529433 0.68509431]
Mu_1 = [0.75529433 -0.68509431]
Sigma = [[0.42953048 -0.02247228]
[-0.02247228 0.53064579]]

b)



c)

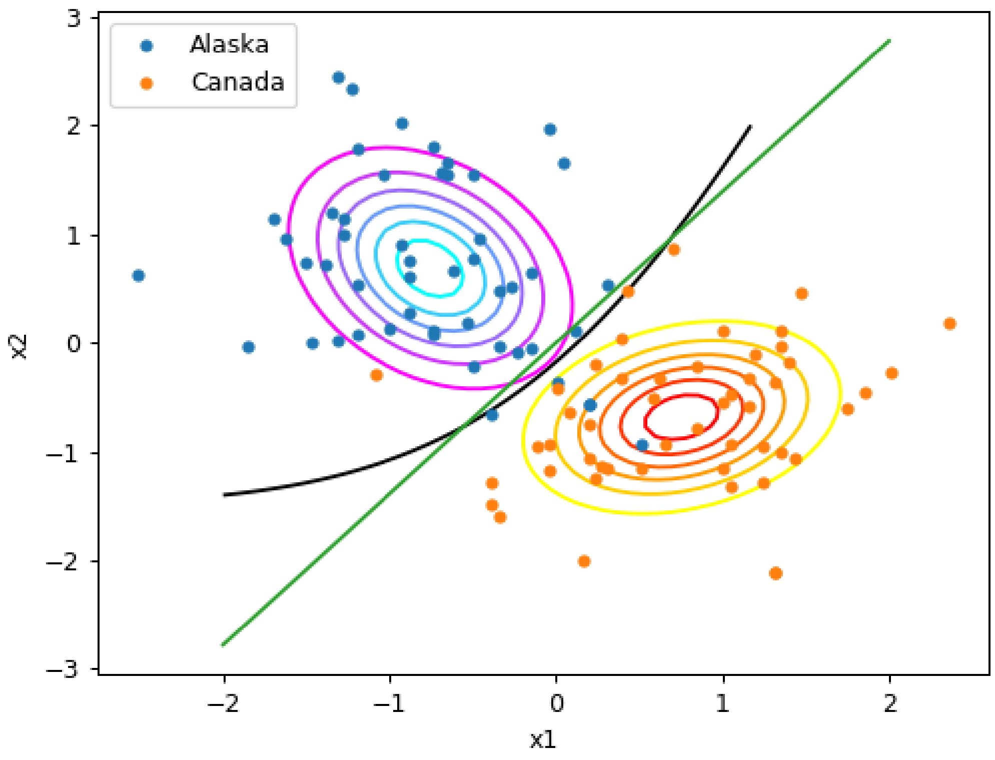
$$2(\mu_0^T - \mu_1^T)\Sigma^{-1}X - (\mu_0^T\Sigma^{-1}\mu_0 - \mu_1^T\Sigma^{-1}\mu_1 - 2ln(\frac{1-\phi}{\phi})) = 0$$

d)

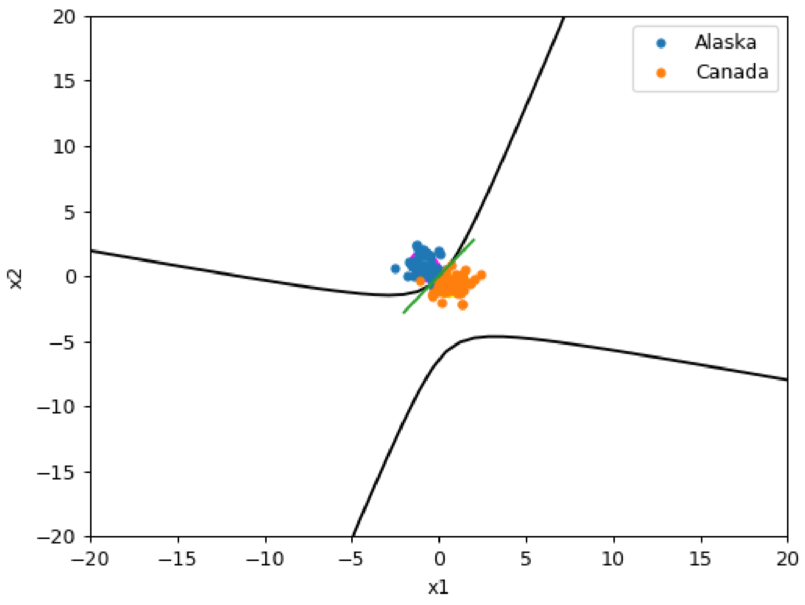
Mu_0 = [-0.75529433 0.68509431]
Mu_1 = [0.75529433 -0.68509431]
Sigma_0 = [[0.38158978 -0.15486516]
[-0.15486516 0.64773717]]
Sigma_1 = [[0.47747117 0.1099206]
[0.1099206   0.41355441]]

e)

$$X^T(\Sigma_0^{-1}-\Sigma_1^{-1})X-2(\mu_0^T\Sigma_0^{-1}-\mu_1^T\Sigma_1^{-1})X-(\mu_0^T\Sigma_0^{-1}\mu_0-\mu_1^T\Sigma_1^{-1}\mu_1-2ln(\frac{1-\phi}{\phi}))=0$$



Zoomed out view:



Hence the quadratic is a hyperbola.

f) It seems that the hyperbola is tangential to both the contours at its vertex. This is why keeping sigma's equal degenerates the hyperbola into a line (in part d). For the region between the centres of the two gaussians (around the area where line and curve intersect) we can see that the curve is able to attribute more points to correct class than the straight line which misses some points in the middle from Alaska and classifies them as Canada.