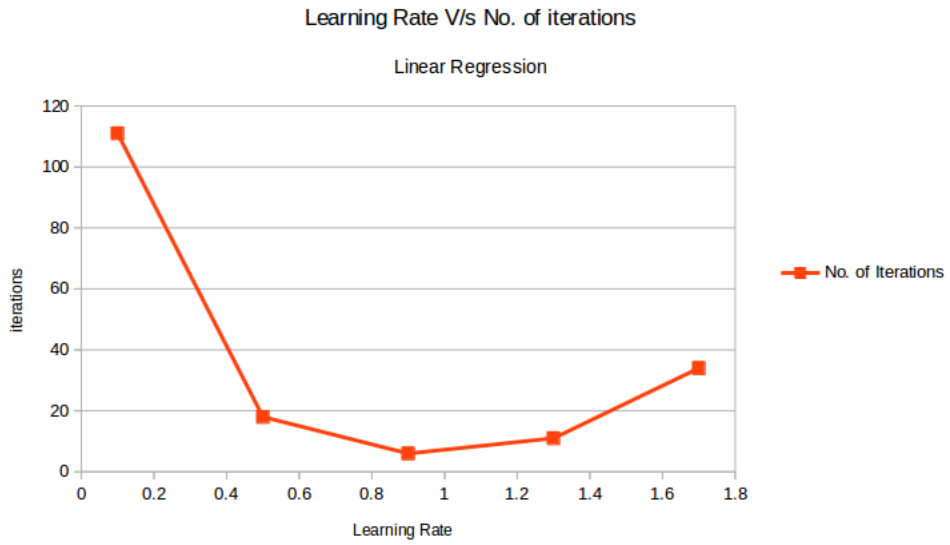
Sachin Kumar Prajapati
2016CS10355
Group-3

COL774 Assignment-1 Report

Q1: Linear Regression

- The first part is the implementation of batch gradient descent method for linear regression. After taking the input of training data, the data is normalised and a 2-dimensional vector θ is chosen, which is initialised to 0.
- After this we apply the gradient descent method to minimise $J(\theta)$. At each iteration, ∇J_θ is evaluated using all training examples and θ is updated.
- The **criteria for stopping** is that if the maximum change in any component of θ is less than ϵ(which is set to 10^{-5}), then we could stop our gradient descent step as θ is almost converged.
- **Learning Rate** is given as a input to the program(typical values $\sim \{0.01, 1.7\}$). The no. of iterations taken to converge differ with the learning rate. They algorithm may even not converge if the learning rate is very high, which happens in the case when learning rate is 2.1 or 2.5. In such cases the program prints a message that the values of θ are not converging.
- Also, in the animation which shows the movement of θ on the contour planes. It could be observed that as we increase the learning rate, θ reaches to its optimal value quickly but beyond a certain learning rate the value of theta overshoots from the optimal value to other direction and it takes more iterations to comeback to the optimal value.
- The predicted θ for this data are: $\theta_0 = 0.9903$, $\theta_1 = 0.00077$.
- The trend of no. of iterations V/s learning rate is shown in the figure below.



Sachin Kumar Prajapati
2016CS10355
Group-3

COL774 Assignment-1 Report

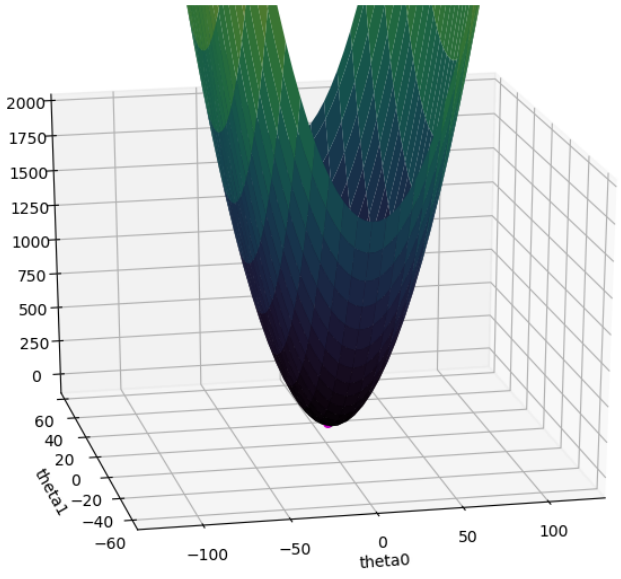


Figure 1: 3D- Mesh for the error function

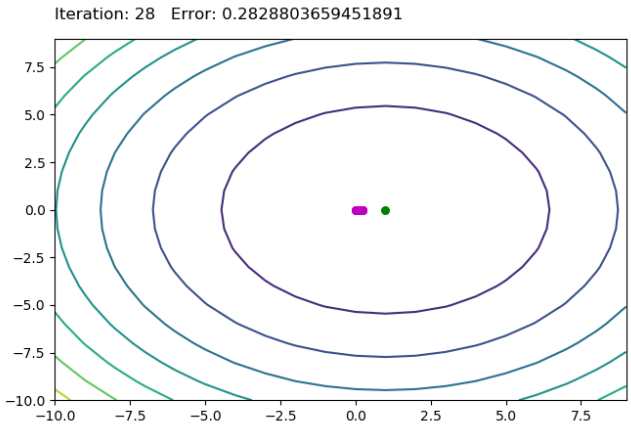


Figure 2: Contour Plots for θ parameter

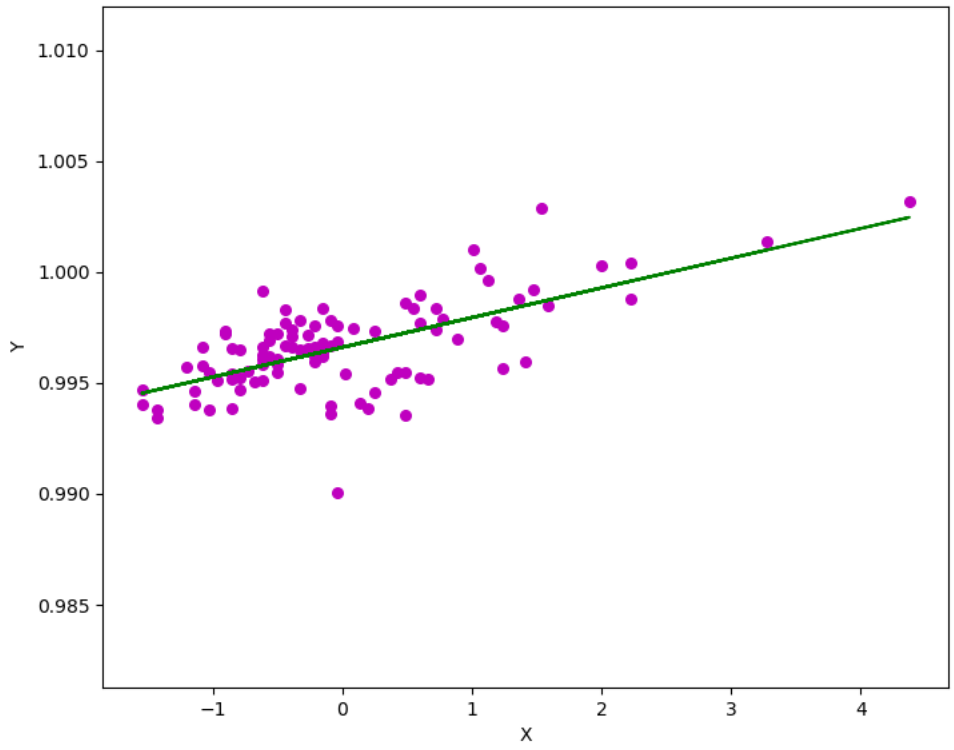


Figure 3: Function predicted by the algorithm.

Sachin Kumar Prajapati
2016CS10355
Group-3

COL774 Assignment-1 Report

Q2: Weighted Linear Regression

- In this part we had to implement weighted linear regression. There is not training phase in this algorithm. The training is done only at time when we have a test data.
- To calculate the prediction at a test point. We calculate the weights of each point in the data set corresponding to that point(x) by using this formula $w(i) = \exp(\frac{-(x-x_i)^2)}{2\tau^2})$
- After calculating the weights we generate a m*m diagonal matrix of the weights(w(i)'s).
- The prediction of θ is then calculated using the formula

$$\theta = (X^\intercal W X)^{-1} X^\intercal W Y$$

- For testing the implementation, several points are generated in the range of values of x. The prediction is done at these points and the graph is plotted.
- The **Bandwidth parameter(τ)**has a major role in determining the correctness of the model. As τ is increased the weighted linear regression starts giving the results close to linear regression because as τ is increased the locality which is used to predict the value of a test point increases and similarly as τ is decreased the model becomes more and more accurate as the locality considered in predicting values is very close to test point. For this data the value of $\tau = 0.8$ or 0.3 works the best. These graphs show different models generated using different values of τ.

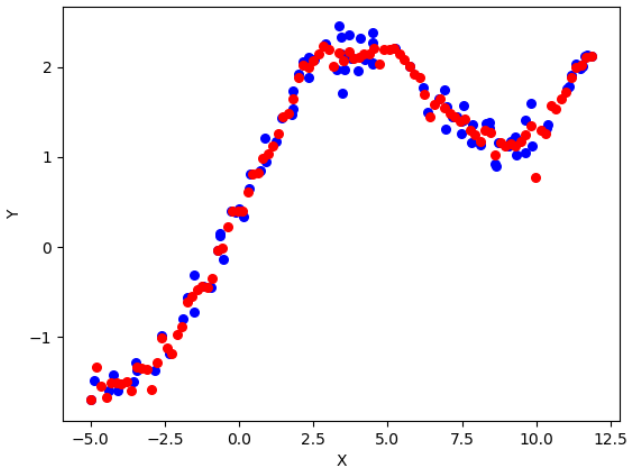


Figure 4: Model Prediction: $\tau = 0.1$

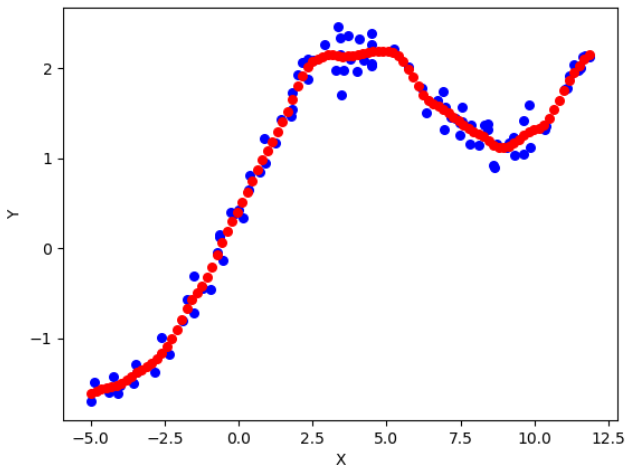


Figure 5: Model Prediction: τ=0.3

Sachin Kumar Prajapati
2016CS10355
Group-3

COL774 Assignment-1 Report

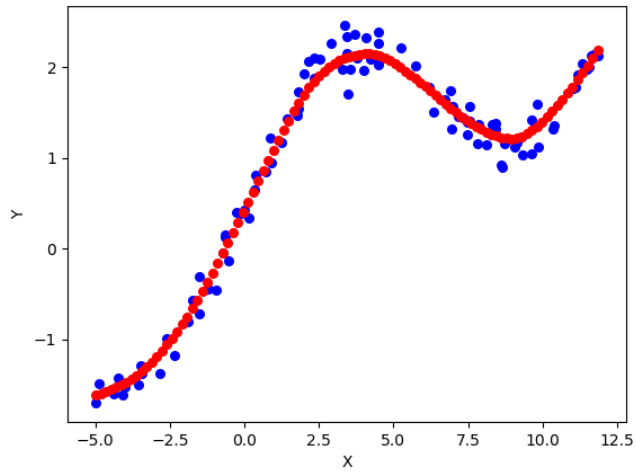


Figure 6: Model Prediction: τ=0.8

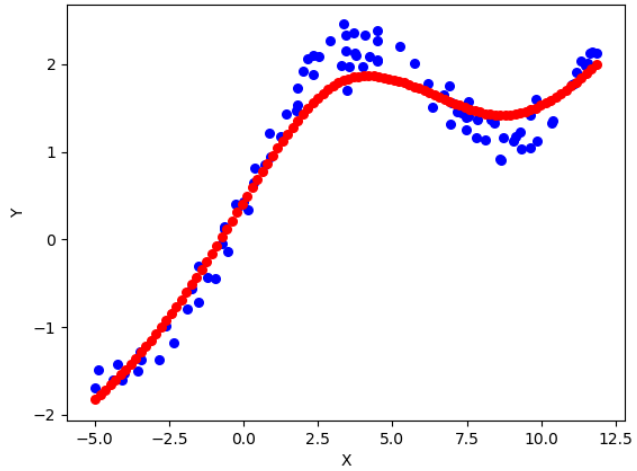


Figure 7: Model Prediction: $\tau = 2$

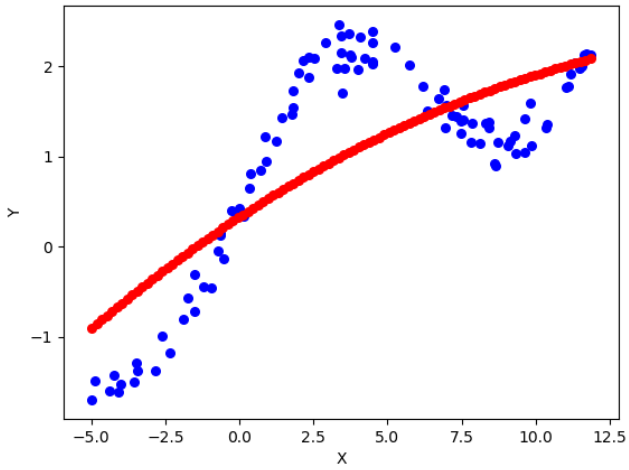


Figure 8: Model Prediction: τ=10

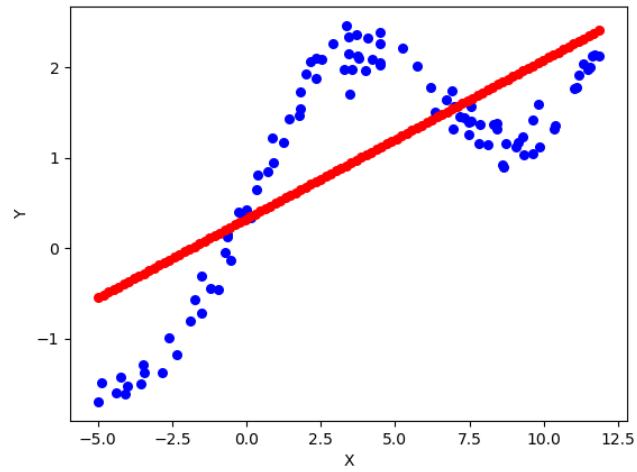


Figure 9: Model Prediction: unweighted(all weights equal)

Q3: Logistic Regression

- In this part we had to implement logistic regression for a classification problem. After taking the input the data is first normalised.
- In this implementation we use the Newton's Method to find the maxima of Log likelihood. To maximise $LL(\theta)$ we use the Newton's method compute the zeroes of $LL(\theta)'$

and as $LL(\theta)$ is a concave function, the local maxima we find using Newton's method is the global maxima.

- For updating the θ we have to compute the θ' and the Hessian Matrix(H) for each iteration. For computing the Hessian we use

$$H = -(X^\intercal DX)$$

$$D = diag(sigmoid(X)(1 - sigmoid(X)))$$

- The **criteria for stopping** is that if the maximum change in any component of θ is less than ϵ(which is set to 10^{-8}), then we could stop as θ is almost converged.
-
- The predicted θ for this data are: $\theta_0 = 0.2232$, $\theta_1 = 1.9626$, $\theta_2 = -1.9648$.
- After finding the optimal value of θ, we could plot the seperator for denormalised data.

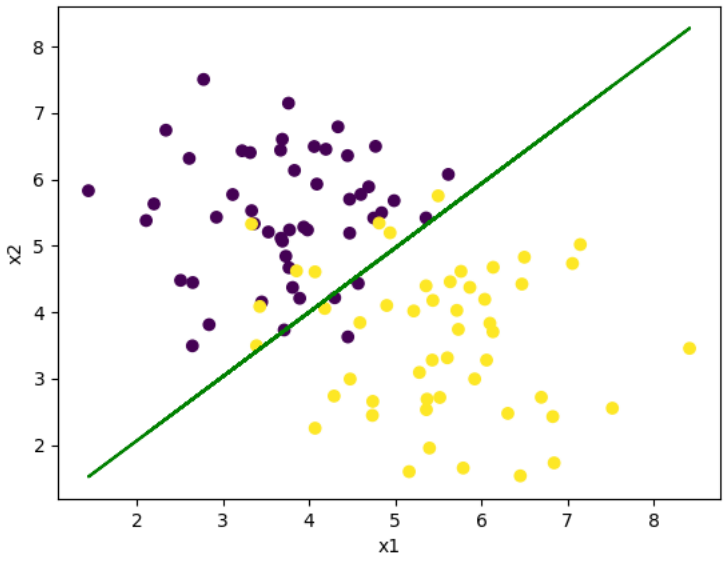


Figure 10: Function predicted by the logistic model.

Q4: Gaussian Discrmimant Analysis(GDA)

- In this part we had to implement GDA for a classification problem.

- In this algorithm we make prior assumptions about the data. We assume $p(y) \sim Bernoulli(\phi)$ and $p(x|y) \sim N(\mu, \Sigma)$
- We can also make the assumptions of $\Sigma_0 = \Sigma_1 = \Sigma$, using this assumptions the separator comes out to be linear.
- In the above case the equation of seperator comes out to be:-

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

- If we drop this assumption, the separator comes out to be a conic section(hyprebola in this case):-

$$x^T(\Sigma_0^{-1} - \Sigma_1^{-1})x + 2(\mu_1^T\Sigma_1^{-1} - \mu_0^T\Sigma_0^{-1})x + (\mu_0^T\Sigma_0^{-1}\mu_0 - \mu_1^T\Sigma_1^{-1}\mu_1) - 2\log(\frac{(1-\phi) \mid \Sigma_1 \mid^{0.5}}{(\phi) \mid \Sigma_0 \mid^{0.5}}) = 0$$

- Parameters for the separator:-

$$\mu_0 = \begin{bmatrix} 98.38 \\ 429.66 \end{bmatrix}, \mu_1 = \begin{bmatrix} 137.46 \\ 366.62 \end{bmatrix}, \Sigma = \begin{bmatrix} 287.482 & -26.748 \\ -26.748 & 1123.25 \end{bmatrix},$$

$$\Sigma_0 = \begin{bmatrix} 255.3956 & -184.3308 \\ -184.3308 & 1371.1044 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 319.5684 & 130.8348 \\ 130.8348 & 875.3956 \end{bmatrix}$$

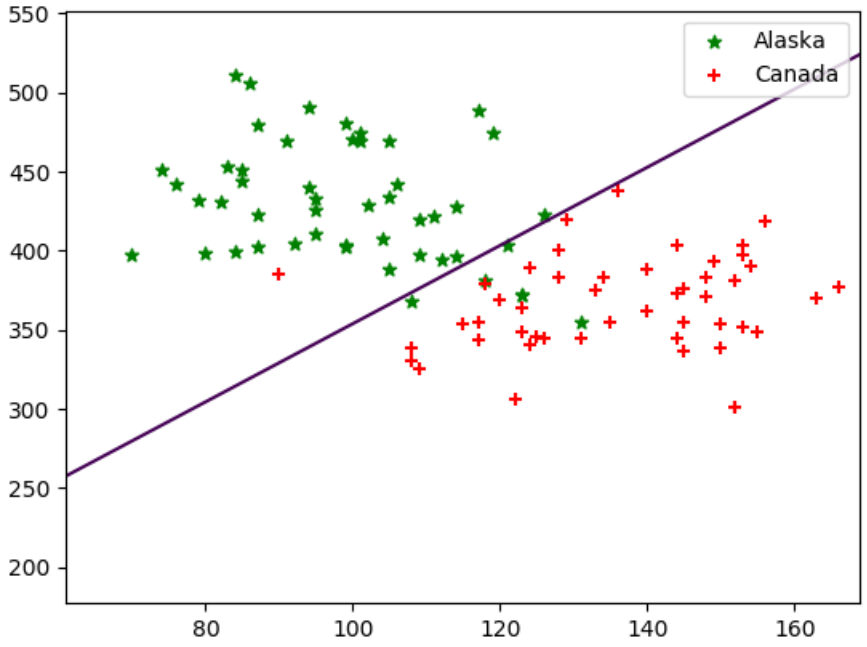


Figure 11: Linear Seperator assuming $\Sigma_0 = \Sigma_1 = \Sigma$

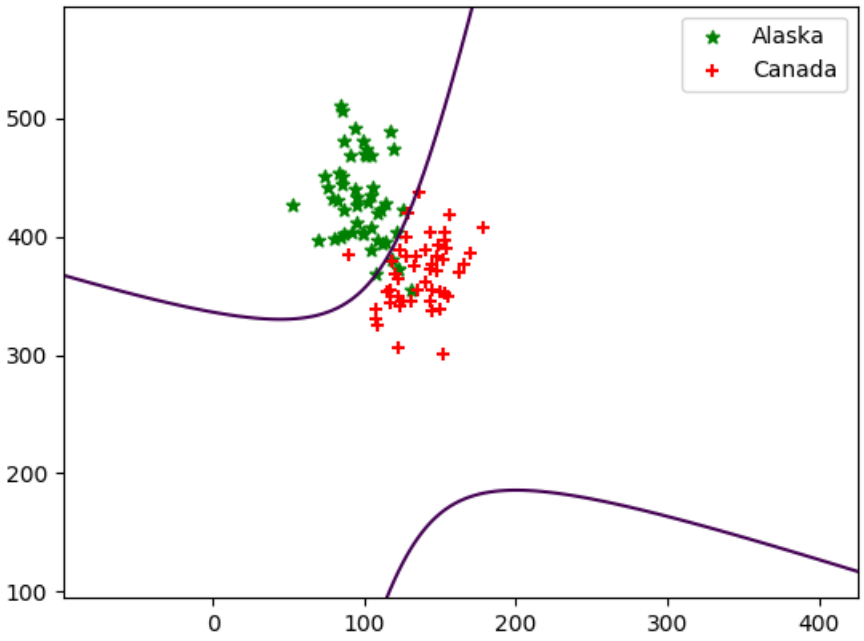


Figure 12: Conic Separator assuming co variance matrix

- It is observed that for this data set both the linear and hyperbolic separator work fine because in this case Σ is close to both Σ_0 and Σ_1. But in case of linear separator we make a extra assumptions that both the co variance matrix are same, which might not be the case all time. So, in those cases conic separator may work better than linear separator.