HW 4

Statistical Learning

Submitted by

Md Salman Rahman

Graduate Student

University of Texas Rio Grande Valley

Submitted to

Dr. Tamer Oraby

November 28, 2021

Contents

1 Problem 1

Upon the arrival of new seven contestants: contestant 4 performing logistic regression, contestant 5 performing LDA, contestant 6 performing QDA, contestant 7 performing naive Bayes, contestant 8 performing support vector machine, contestant 9 performing random forest and contestant 10 performing neural network, the judge would like to reopen the contest. This time, plots of the decision boundaries are required. Please add those four methods to the contest and compare the ten contestants.

Solution

1.1 Decision Boundary of 10 Contestant

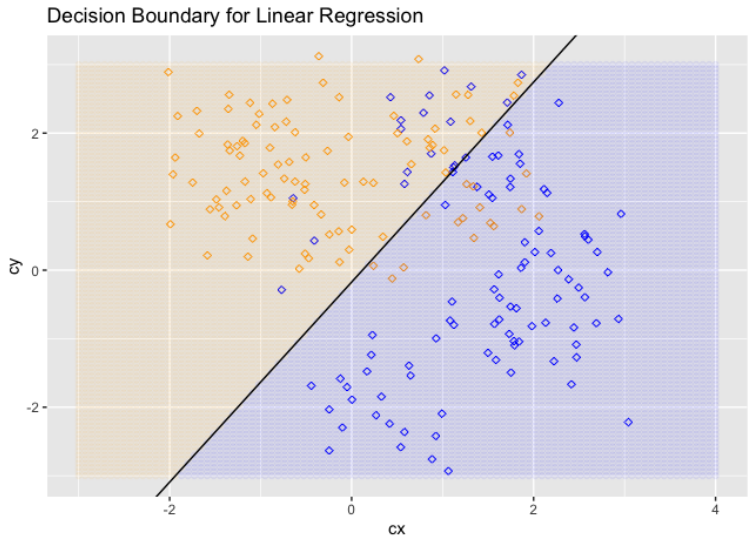


Figure 1: Decision Boundary of Contestant 1 (linear regression)

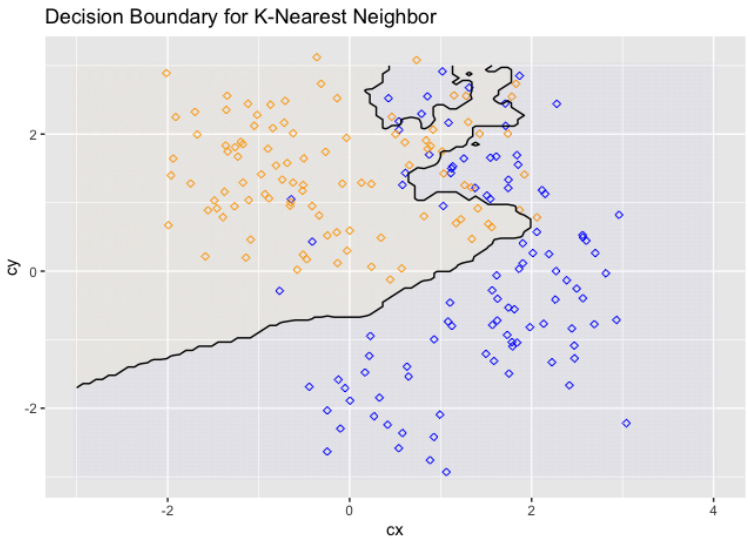


Figure 2: Decision Boundary of Contestant 2 (KNN)

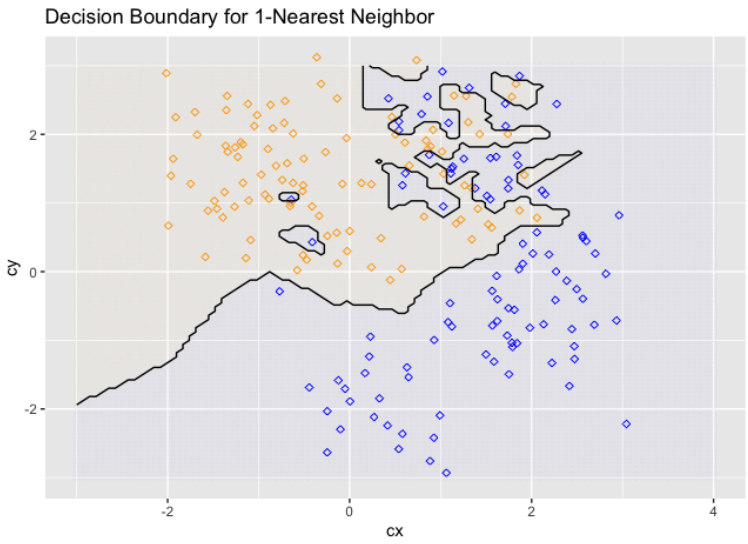


Figure 3: Decision Boundary of Contestant 3 (1-NN)

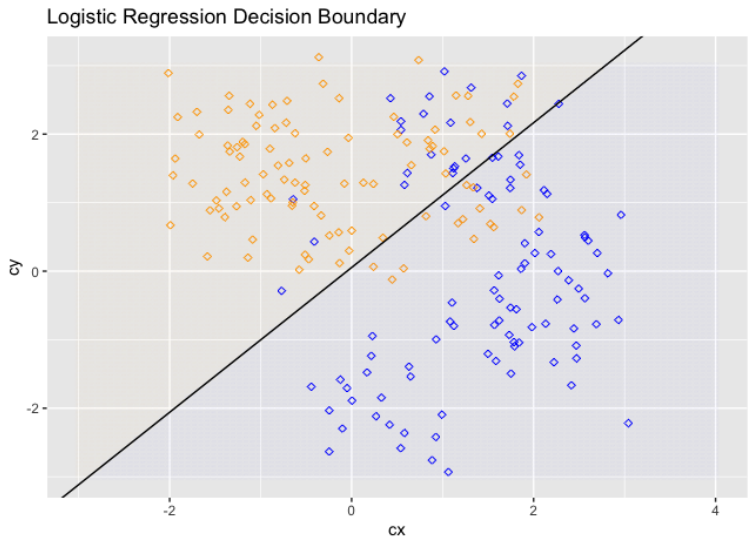


Figure 4: Decision Boundary of Contestant 4 (Logistic Regression)

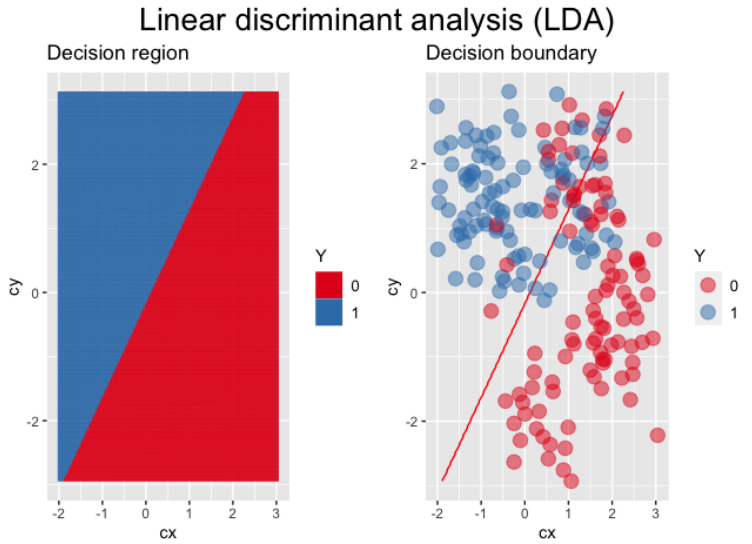


Figure 5: Decision Boundary of Contestant 5 (LDA)

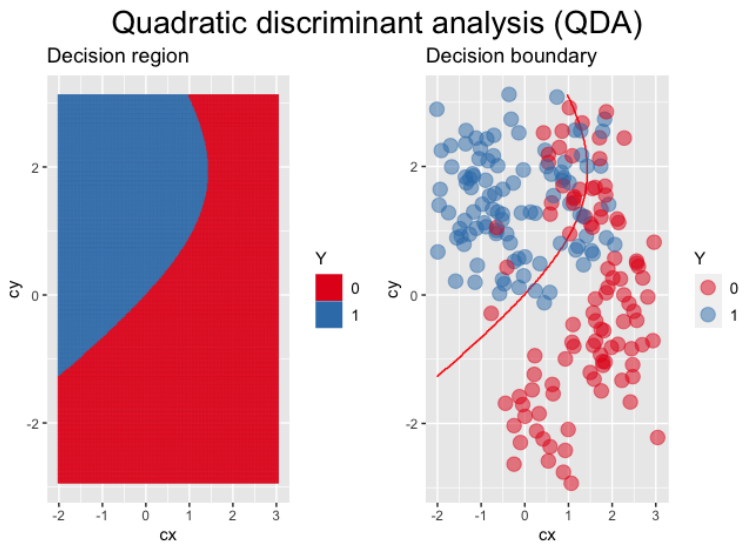


Figure 6: Decision Boundary of Contestant 6 (QDA)

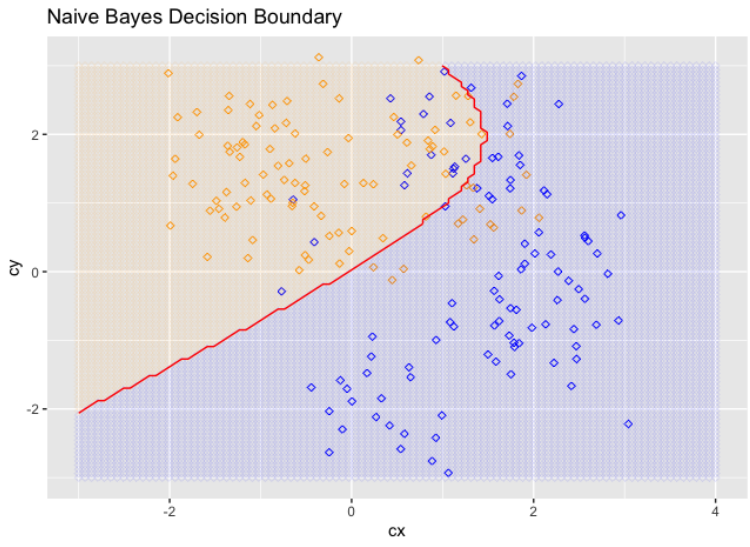


Figure 7: Decision Boundary of Contestant 7 (Naive Bayes)

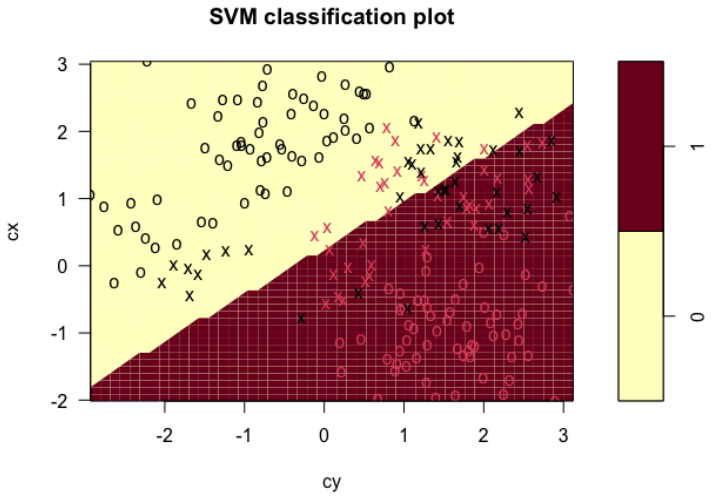


Figure 8: Decision Boundary of Contestant 8 (SVM)

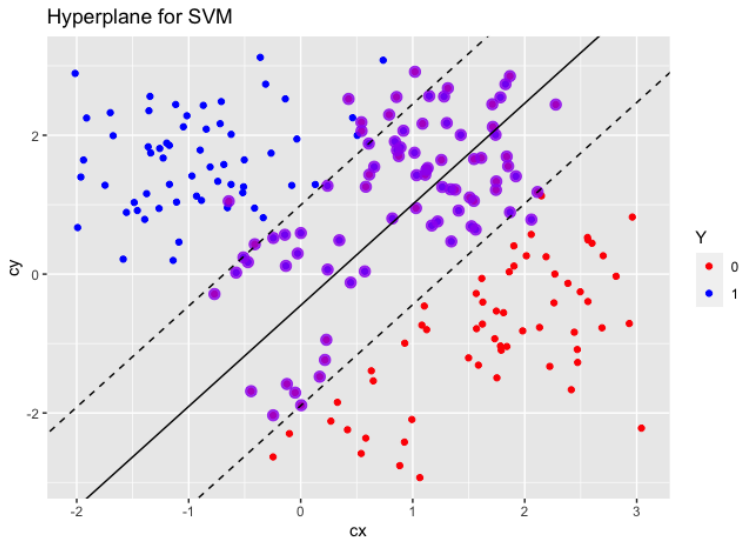


Figure 9: SVM Classifier Contestant 8

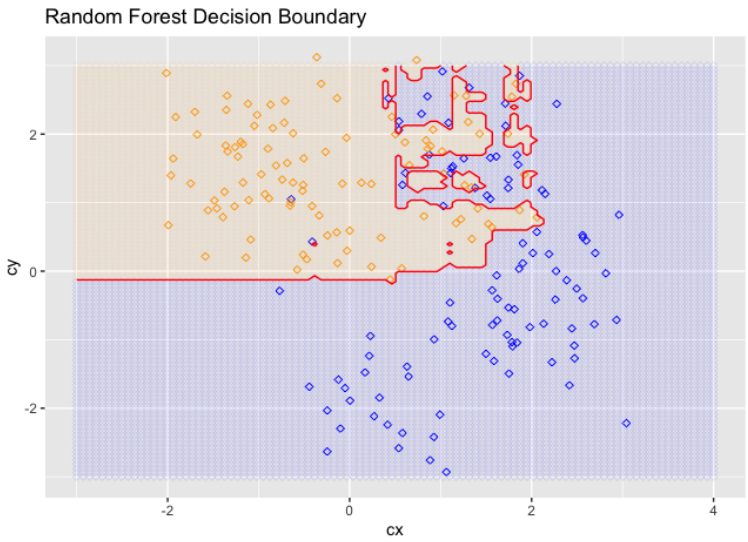


Figure 10: Decision Boundary of Contestant 9 (RF)

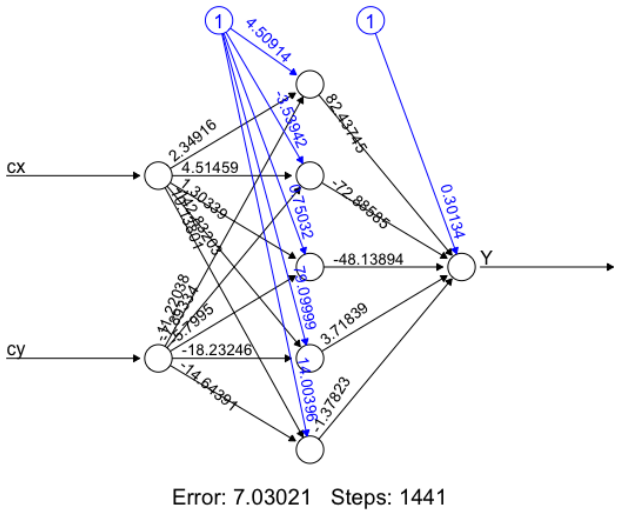


Figure 11: Neural Network Architecture with 5 Hidden Neuron and logistic Activation Function (Contestant 10)

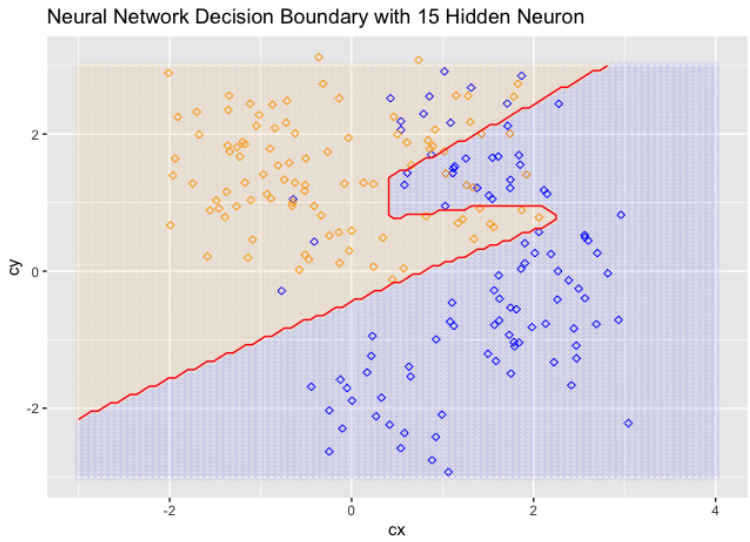


Figure 12: Decision Boundary of Contestant 10 (Neural Network)

1.2 Judge Role and Comparison

In Table 1, the notation C1 to C10 indicate contestant 1 to 10. As a judge, I will declare both contestant 6, 7, and 8 as the winner as they have the smallest testing error. The K nearest neighbour (contestant 2) performs worst among all the contestant. Also, I apply neural network with 50, 100, and 500 hidden neuron, and as the layer and neuron increase the error decrease but it seems to be over fit on testing data. All in all, as a judge if I have to choose one, I will choose contestant 7 who train the model with naive bayes. But, contestant 6 and 8 also perfoms extremely well.

Table 1: Performance Evaluation of Different Model

Criteria	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Testing Error	0.1	0.2	0.1	0.15	0.1	0.05	0.05	0.05	0.15	0.2

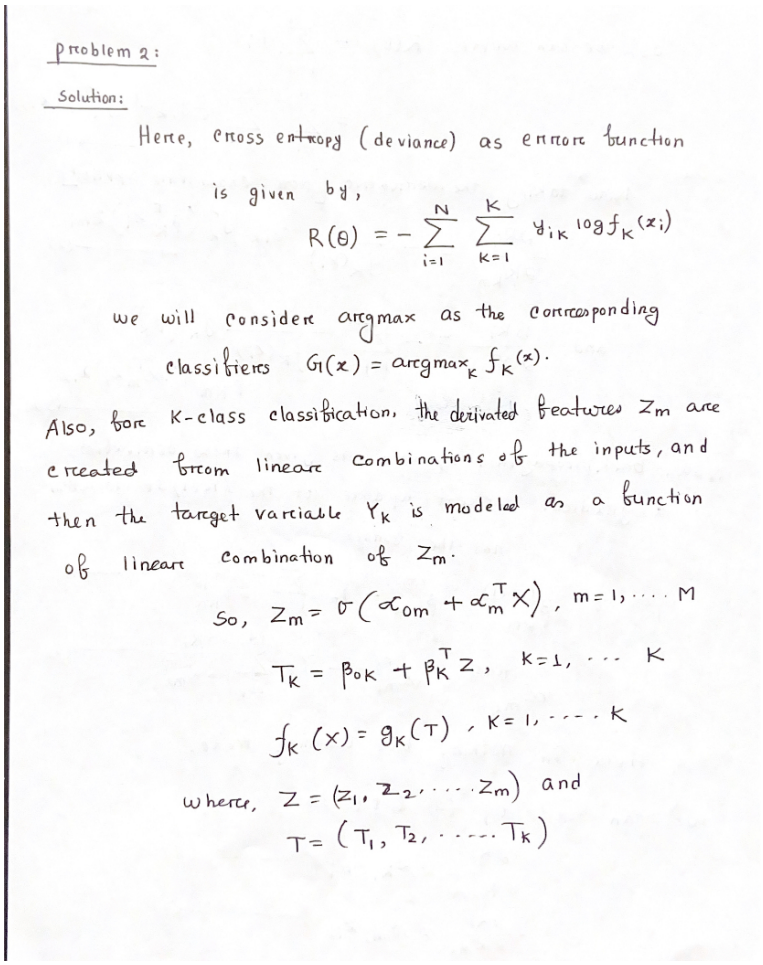
R Code

R code for the all the contestant and judge is uploaded in the Blackboard.

2 Problem 2

Derive the forward and backward propagation equations for the cross-entropy loss function.

Solution



Problem 2:

Solution:

Here, cross entropy (deviance) as error function is given by,

$$R(\theta) = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik} \log f_k(x_i)$$

we will consider argmax as the corresponding classifiers $G(x) = \text{argmax}_k f_k(x)$.

Also, for K-class classification, the derivated features Z_m are created from linear combinations of the inputs, and then the target variable Y_k is modeled as a function of linear combination of Z_m.

So, $Z_m = \sigma(\alpha_{0m} + \alpha_m^T X)$, $m = 1, \ldots M$

$T_k = \beta_{0k} + \beta_k^T Z$, $k = 1, \ldots K$

$f_k(X) = g_k(T)$, $k = 1, \ldots K$

where, $Z = (Z_1, Z_2, \ldots Z_m)$ and

$T = (T_1, T_2, \ldots T_k)$

So, we can write, $R(\theta) = \sum_{i=1}^{N} R_i$

$$= \sum_{i=1}^{N} \sum_{K=1}^{K} (-y_{iK} \log(f_K(x_i))) \quad \text{——(1)}$$

Now taking the derivative, [Note: chain rule applied]

$$\frac{\partial R_i}{\partial \beta_{Km}} = -\frac{1}{f_K(x_i)} y_{iK} g_K'(\beta_K^T z_i) z_{mi} \quad \text{——(2)}$$

and $$\frac{\partial R_i}{\partial \alpha_{mL}} = -\sum_{K=1}^{K} \frac{y_{iK}}{f_K(x_i)} g_K'(\beta_K^T z_i) \beta_{Km} \sigma'(\alpha_m^T x_i) x_{iL} \quad \text{——(3)}$$

Now, Backpropagation update as given these derivatives, a gradient descent update at the $(r+1)$st iteration has the form:

$$\beta_{Km}^{(r+1)} = \beta_{Km}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{Km}^{r}} \quad \text{——(4)}$$

$$\alpha_{mL}^{(r+1)} = \alpha_{mL}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R_i}{\partial \alpha_{mL}^{(r)}} \quad \text{——(5)}$$

Here γ_r nothing but learning rate.

We can write eq (2) & (3) as:

$$\frac{\partial R_i}{\partial R_{Km}} = \delta_{Ki} z_{mi} \quad \text{——(6)}$$

$$\frac{\partial R_i}{\partial \alpha_{mL}} = s_{mi} x_{iL} \quad \text{——(7)}$$

δ_{ki} = errrors from the current model at output unit

s_{mi} = errrors from current model at hidden layer unit.

These errrors satisfy this equation

$$s_{mi} = \sigma'(\alpha_{0m} + \alpha_m^T x_i) \sum_{k=1}^{K} \beta_{km} \delta_{ki} \quad \text{——— (8)}$$

eq (8) is known as backpropagation equation.

In forwarded pass, the current weights are fixed and the predicted values $\hat{f}_k(x_i)$ are computed from eq (8).

In Backward pass, errror δ_{ki} are computed and back propagated via eq (8) to give the errror s_{mi}.

In both pass, errrors are then used to compute the gradients for the updates in eq (4) & (5) via eq (6) & (7).