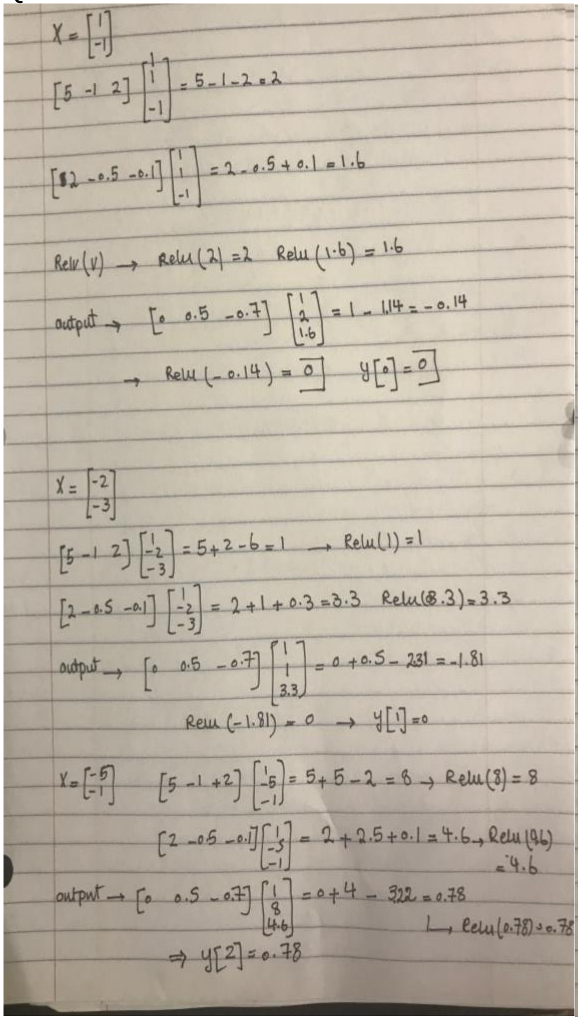
Question 1:



$X = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$

$\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} = 5 - 1 - 2 = 2$

$\begin{bmatrix} \cancel{8}2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} = 2 - 0.5 + 0.1 = 1.6$

Relu(v) $\rightarrow$ Relu(2) = 2 Relu(1.6) = 1.6

output $\rightarrow$ $\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 1.6 \end{bmatrix} = 1 - 1.14 = -0.14$

$\rightarrow$ Relu(-0.14) = $\boxed{0}$ $y[0] = \boxed{0}$

$X = \begin{bmatrix} -2 \\ -3 \end{bmatrix}$

$\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ -3 \end{bmatrix} = 5 + 2 - 6 = 1 \rightarrow$ Relu(1) = 1

$\begin{bmatrix} 2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ -3 \end{bmatrix} = 2 + 1 + 0.3 = 3.3$ Relu(3.3) = 3.3

output $\rightarrow$ $\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 3.3 \end{bmatrix} = 0 + 0.5 - 2.31 = -1.81$

Relu(-1.81) = 0 $\rightarrow$ $y[1] = 0$

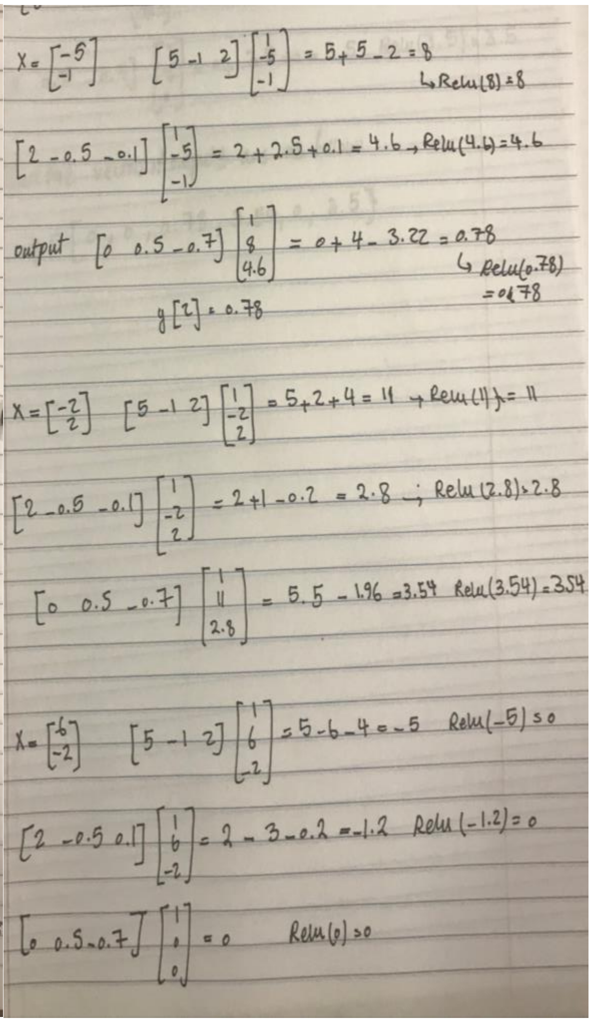
$X = \begin{bmatrix} -5 \\ -1 \end{bmatrix}$ $\begin{bmatrix} 5 & -1 & +2 \end{bmatrix} \begin{bmatrix} 1 \\ -5 \\ -1 \end{bmatrix} = 5 + 5 - 2 = 8 \rightarrow$ Relu(8) = 8

$\begin{bmatrix} 2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ -5 \\ -1 \end{bmatrix} = 2 + 2.5 + 0.1 = 4.6 \rightarrow$ Relu(4.6) = 4.6

output $\rightarrow$ $\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 8 \\ 4.6 \end{bmatrix} = 0 + 4 - 3.22 = 0.78$

$\hookrightarrow$ Relu(0.78) = 0.78

$\Rightarrow y[2] = 0.78$



$X = \begin{bmatrix} -5 \\ -1 \end{bmatrix}$ $\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -5 \\ -1 \end{bmatrix} = 5 + 5 - 2 = 8$

$\hookrightarrow$ Relu(8) = 8

$\begin{bmatrix} 2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ -5 \\ -1 \end{bmatrix} = 2 + 2.5 + 0.1 = 4.6 \rightarrow$ Relu(4.6) = 4.6

output $\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 8 \\ 4.6 \end{bmatrix} = 0 + 4 - 3.22 = 0.78$

$\hookrightarrow$ Relu(0.78) = 0.78

$y[2] = 0.78$

$X = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$ $\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix} = 5 + 2 + 4 = 11 \rightarrow$ Relu(11) = 11

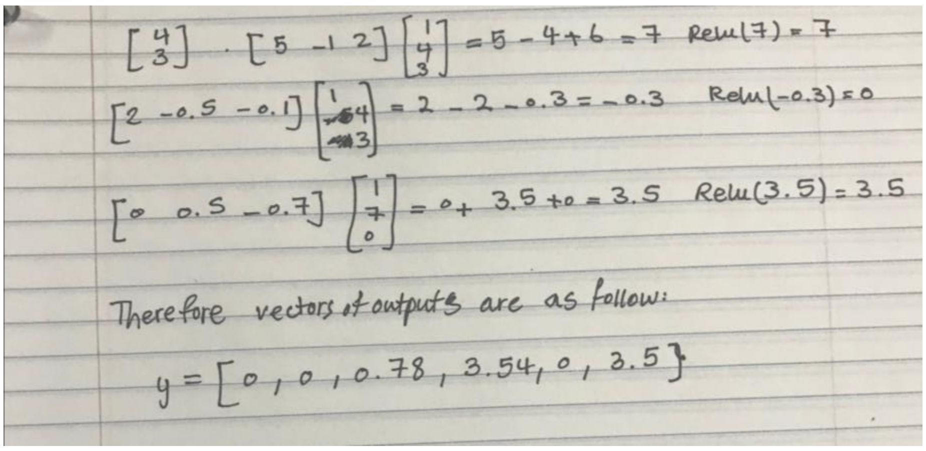
$\begin{bmatrix} 2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 2 \end{bmatrix} = 2 + 1 - 0.2 = 2.8 \rightarrow$ Relu(2.8) = 2.8

$\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 11 \\ 2.8 \end{bmatrix} = 5.5 - 1.96 = 3.54$ Relu(3.54) = 3.54

$X = \begin{bmatrix} -6 \\ -2 \end{bmatrix}$ $\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 6 \\ -2 \end{bmatrix} = 5 - 6 - 4 = -5$ Relu(-5) = 0

$\begin{bmatrix} 2 & -0.5 & 0.1 \end{bmatrix} \begin{bmatrix} 1 \\ 6 \\ -2 \end{bmatrix} = 2 - 3 - 0.2 = -1.2$ Relu(-1.2) = 0

$\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = 0$ Relu(0) = 0



$\begin{bmatrix} 4 \\ 3 \end{bmatrix}$ $\cdot$ $\begin{bmatrix} 5 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} = 5 - 4 + 6 = 7$ Relu(7) = 7

$\begin{bmatrix} 2 & -0.5 & -0.1 \end{bmatrix} \begin{bmatrix} 1 \\ \cancel{6}4 \\ \cancel{4}3 \end{bmatrix} = 2 - 2 - 0.3 = -0.3$ Relu(-0.3) = 0

$\begin{bmatrix} 0 & 0.5 & -0.7 \end{bmatrix} \begin{bmatrix} 1 \\ 7 \\ 0 \end{bmatrix} = 0 + 3.5 + 0 = 3.5$ Relu(3.5) = 3.5

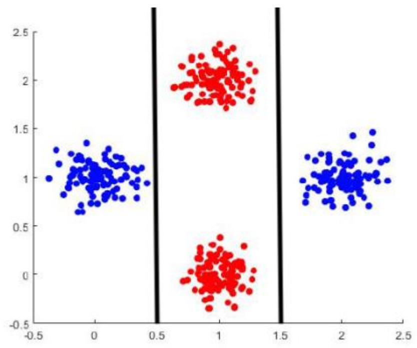
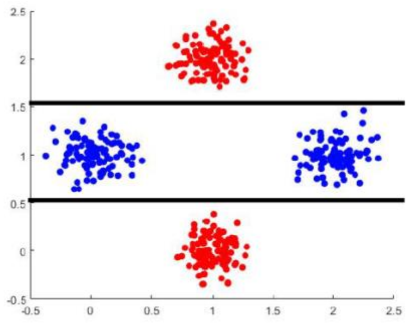
Therefore vectors of outputs are as follow:

$y = [0, 0, 0.78, 3.54, 0, 3.5\}$

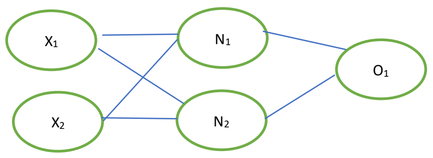
Question 2:

Dataset 1:

For both examples, one hidden, one input and one output layer has been designed. The hidden layer consists of two neurons to do the classification task. The decision boundaries have been shown in the following figure:

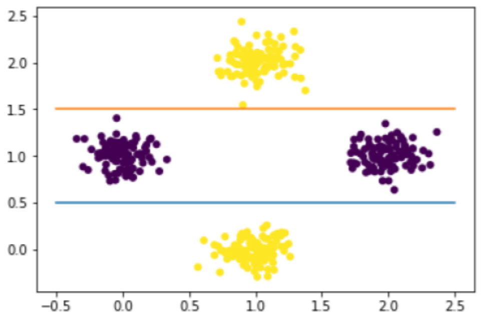
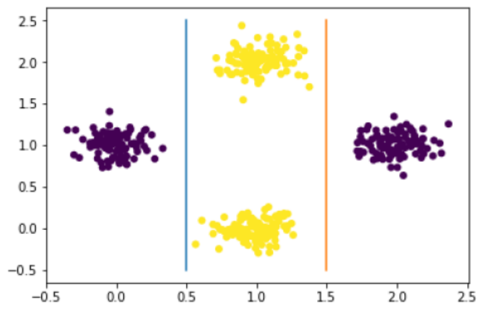
For left picture, Weights of first and second layer are [0,0,2,2] and [-2,2] respectively. The biases of hidden layer are [-1, -3] and bias of the output layer is 1. For the right picture, weights of first and second layer are [2,2,0,0] and [-2,2]. The biases are the same as the other picture. The underlying neural network architecture has been shown in the following figure:



Sigmoid activation function has been used in all layers.

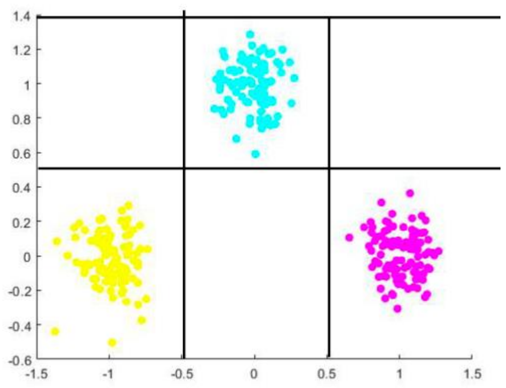
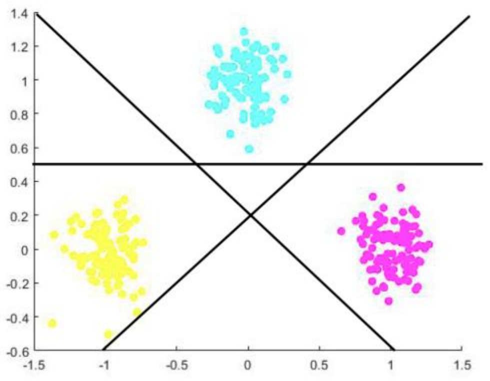
The results of our classification method have been shown in the following plot. Our picked decision boundaries were successful in classifying the points most of the time. The accuracy on both examples were 100% which shows the validity of our decision boundaries.

The high accuracy could be due to the fact that the data from both classes are perfectly separable and we could discriminate the differences between them with two simple lines in 2D.

Dataset2:

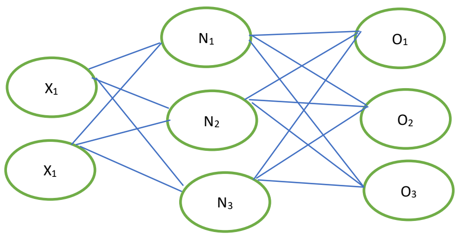
In this dataset, we are dealing with multi-class classification problem in which we have more than two distinct classes. We should pick enough decision boundaries so that we could guarantee that each of the classes are discriminated from the other two classes. Therefore, I picked different number of lines in each example to separate the classes. The examples are shown on the following pictures:

The weights and biases for each picture has been shown in Table 1. The corresponding weight and for each neuron in each layer has been mentioned separately within a bracket.

Name	Right picture	Left picture
First layer weights	[2,0],[2,0],[2,10],[0,10]	[-4,5],[4,5],[0,2]
First layer biases	[-1,1,-4,4]	[-1,-1,-1]
Second layer weights	[1,1,0,0],[0,1,1,0],[1,0,-1,0]	[3,-4,-1],[-4,1,-3],[0.5,1,0]
Second layer biases	[-1,-1,0]	[3,4,8]

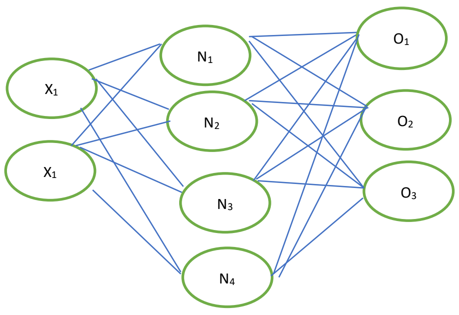
The underlying neural network architecture of the first example (left picture) has been shown in the following figure:



Sigmoid activation function has been used in the hidden layers. However, as we need to mark classes between 0 to 2, we used SoftMax activation function on the output layer to come up with the class labels for each point which has the highest probability among all of the classes.

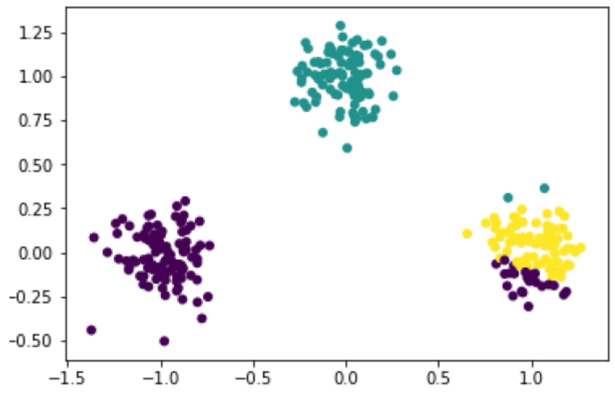
$$Sigmoid: \quad \sigma(v) = \frac{1}{1+e^{-v}}$$
$$Softmax: \sigma(v)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

The classes are very close to each other, so this could be a possible reason that sometimes algorithm was not able the classify the point correctly. After finding these examples and implementation I would think that it is a better solution to apply another function like ReLU or Tanh before applying SoftMax. Therefore, we used ReLU function to estimate the hidden layers output and then pass it to the next layer.
For the second example we tried to classify classes with three different lines from the first example. However, there were multiple overlaps between two classes which was hindering us from getting more than 50% accuracy. Therefore, four lines as it is shown in the left figure has been selected. The underlying neural network architecture for left picture has been shown in the following figure:



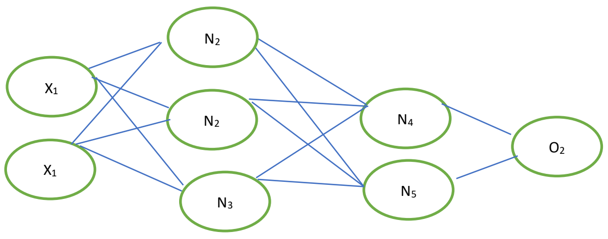
To reduce the complexity of the problem, I used sigmoid as an activation function in hidden layers. After finding the inputs of N1, N2, N3 and N4 we find three planes which distinguish them and used their weights for the output layer computations. Softmax was used in the output layer to classify all three classes

The result of our classification has been shown in the following scatter plot:

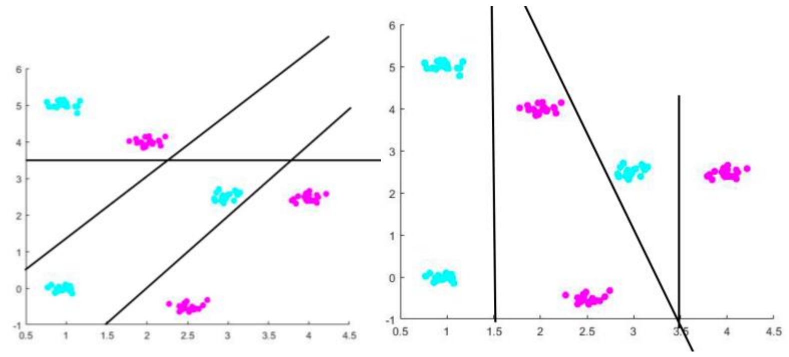


My results showed that second example was more successful in separating different classes. This observation thought me that adding more neurons to the hidden layer helped to solve the problem better. As you see we got much more better results with 3 neurons which is due to the decision boundaries we found. There are couple of pointes in outer left part which was not classified correctly. According to Jung et al. [1], importance of underlying decision boundary is something which is missed sometimes. I think this could related to misclassification we had in our problem. A better decision boundary to discriminate the class 0 and class 2 purposefully could help to mitigate this problem.

Dataset 3:
In this problem, we are looking into binary classification problem. However, its major difference with the first dataset is that we should pick decision boundaries more carefully as many of the points from two different classes are pretty close. I picked couple of examples which lead to the overlapping between red points around (3,2.5) and blue points around (4,2.5). The following pictures shown the two examples I found. The first example reached 100% accuracy which could be due the fact that it is separating the challenging points of (3,2.5) and (4,2.5) more clearly. In the second example which accuracy was around 84, he points which are around (4,2.5) was not recognized correctly. This could be related to how much these points are close to the other class in the neighborhood. Moreover, existence of a decision boundary which completely separate these classes could be a solution. My approach in this question was to use 3 neurons in the hidden layer to take the feature space data into 3D. Then using CPM 3D plotter, the points would be visualized to extract the information about the plane which should separate them. The information I would extract consists of normal vector which is orthogonal to the plane and a point coordinates in that plane. As third dataset was a little more challenging, I decided to not include more complexity which is not helpful for solving the problem. Therefore, severe sigmoid has been selected as an activation function in all layers. The proposed architecture has been shown in the following diagram. The main reason I picked two hidden layers for this problem is that it was not feasible to completely separate the red and blues using one 3D planes. Therefore, I picked two 3D plane which helped me in transforming my problem into 2D space.



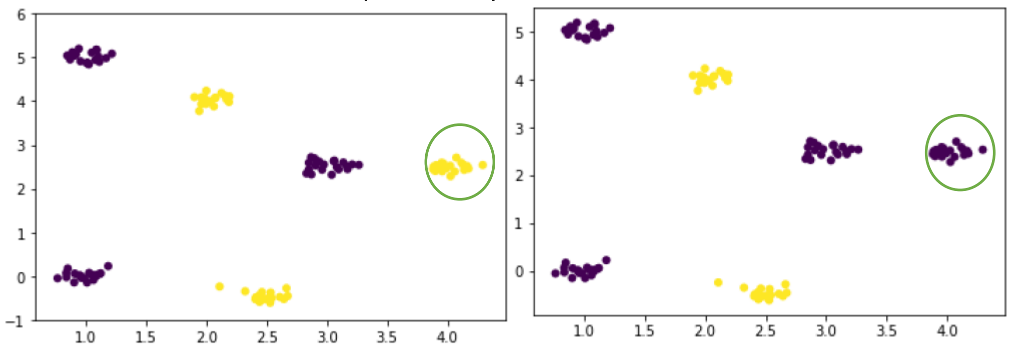
The following pictures are the scatterplots after the classification.



The weights and biases for each layer in the first and second example has been shown to clarify the design process.

Name	Left picture	Right picture
First layer weights	[2,0],[4,1],[2,0],	[0,2],[1,-2],[-6,4]
First layer biases	[-7,-13,-3]	[-9,4,3]
Second layer weights	[0,2,-2],[1,2,1]	[0,2,-2],[1,2,1]
Second layer biases	[-1,-3]	[-1,-3]
Output layer weights	[2,2]	[2,2]
Output layer Bias	-1	-1

The scatter plots which we discussed above is shown in below. As you could see the only difference is on the red class in outer left part of the picture.



References:

Lee, C., Jung, E., Kwon, O., Park, M. and Hong, D., 2004. Decision boundary formation of neural networks. *WSEAS Transactions on Computers*, *3*(4), pp.853-861.