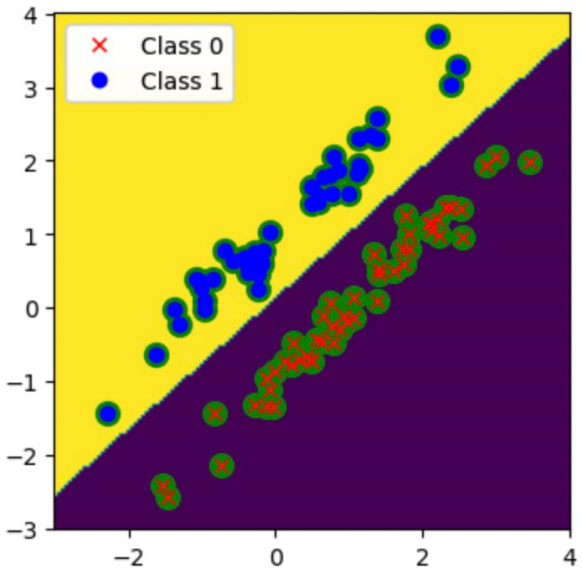
EE559-Homework#6-Lei Lei

1. SVM using sklearn

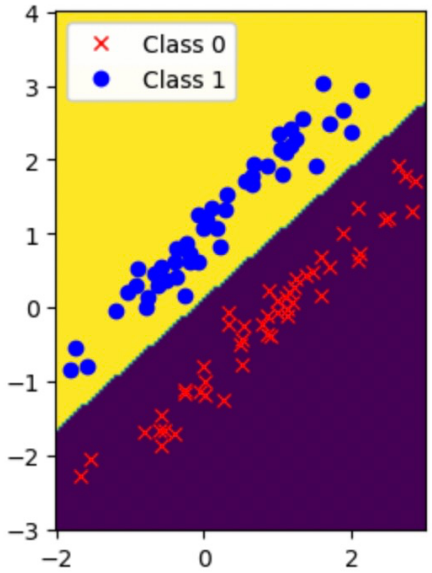
(a) When C = 0.01:

```
train acc = 1.0, test acc = 1.0
weight vector w =  [[-0.47617753  0.53480959]]
offset w0 = [-0.06369157]
```

Train dataset1:



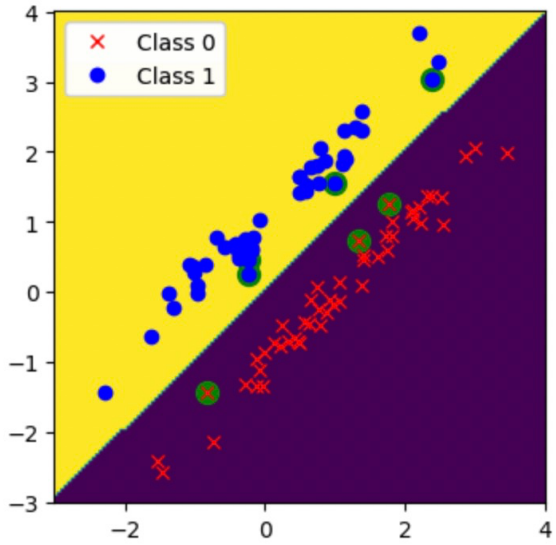
Test dataset1:



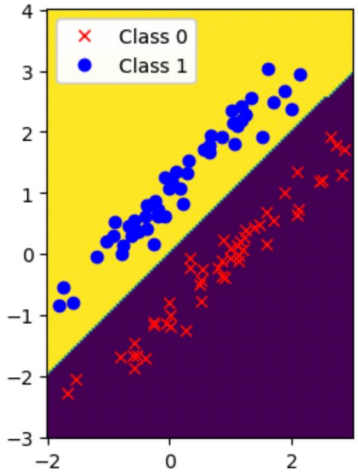
When C = 1:

```
train acc = 1.0, test acc = 1.0
weight vector w =  [[-1.5291679   1.54596349]]
offset w0 = [-0.04317366]
```

Train dataset1:



Test dataset1:



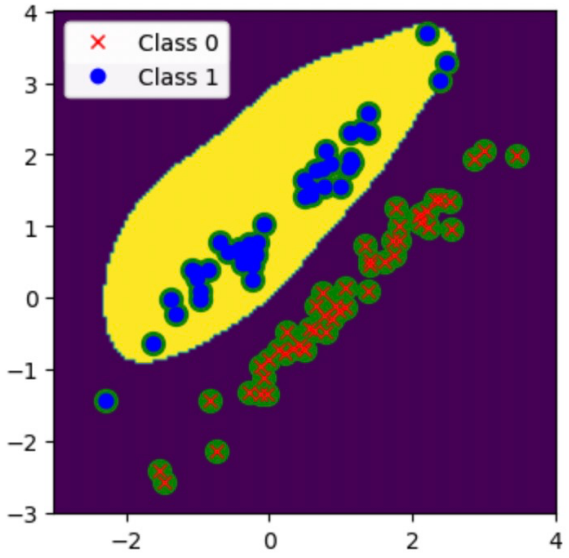
Discuss your results and explain the performance and the differences for the different values of C.

Obviously, the dataset1 can be classified 100% correct because the data is linearly separatable. By plotting the support vectors, we can find that, when C = 0.01, every data point is considered as a support vector while there are only a few data points are considered as support vectors when C = 1. Because the value of C stands for a regularization parameter in the SVM. The larger values of C, a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin.

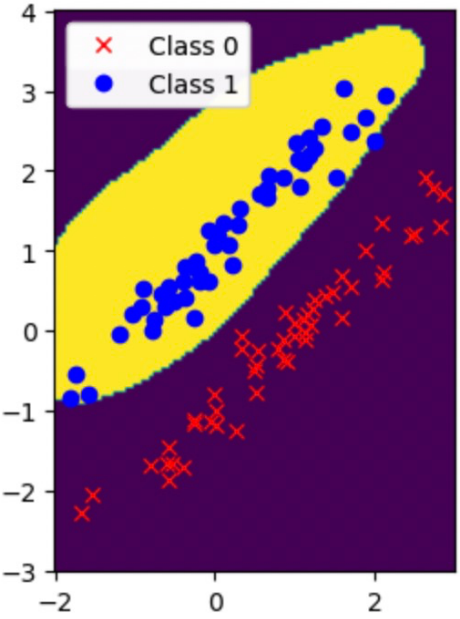
(b) C = 0.01, γ = 1

```
train acc = 0.99, test acc = 0.99
```

Train dataset1:



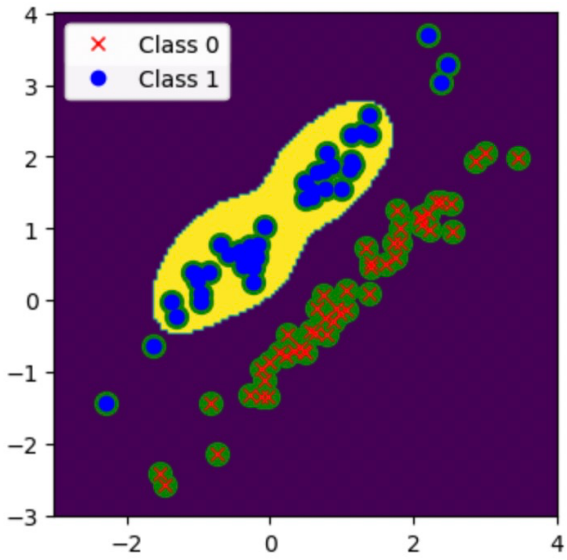
Test dataset1:



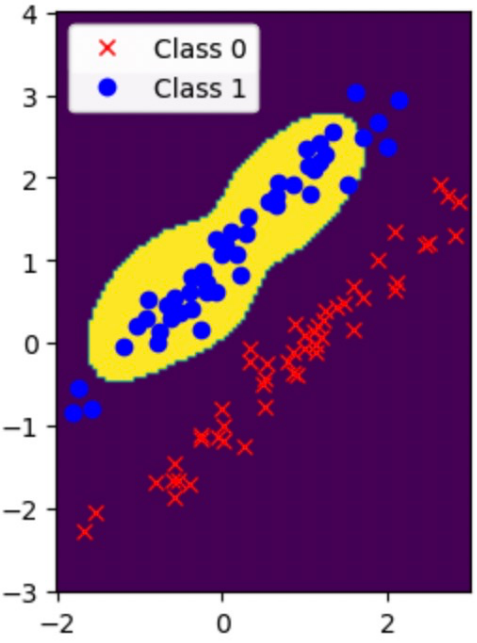
C = 0.01, γ = 3

```
train acc = 0.95, test acc = 0.92
```

Train dataset1:



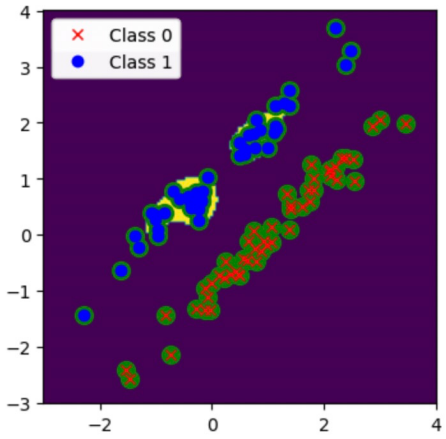
Test dataset1:



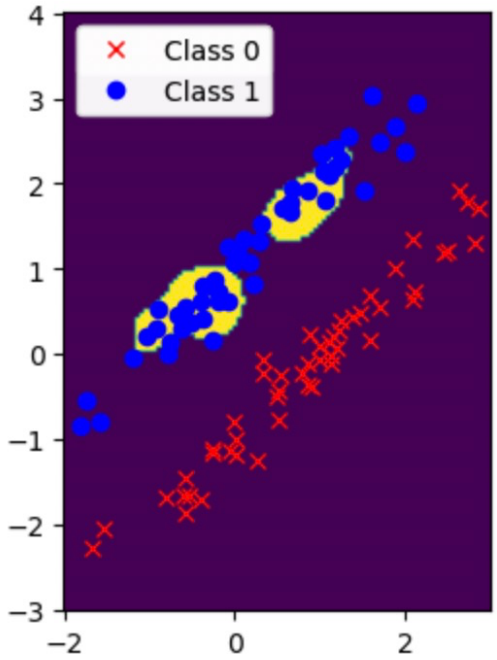
C = 0.01, γ = 10

```
train acc = 0.88, test acc = 0.77
```

Train dataset1:



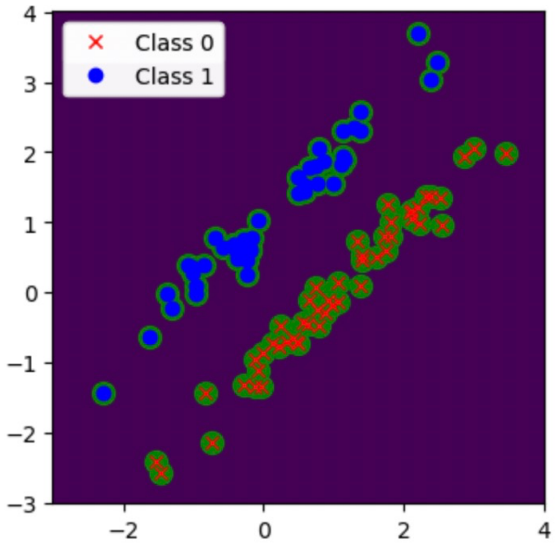
Test dataset1:



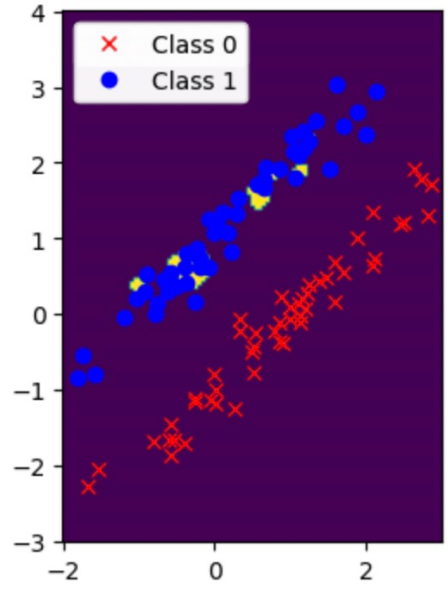
C = 0.01, γ = 50

```
train acc = 0.79, test acc = 0.6
```

Train dataset1:



Test dataset1:



Explain the linearity or nonlinearity of the decision boundary and explain the difference in decision regions for the various values of γ. State where (if anywhere) you observe underfitting or overfitting.

The decision boundary is non-linear due to the RBF kernel.

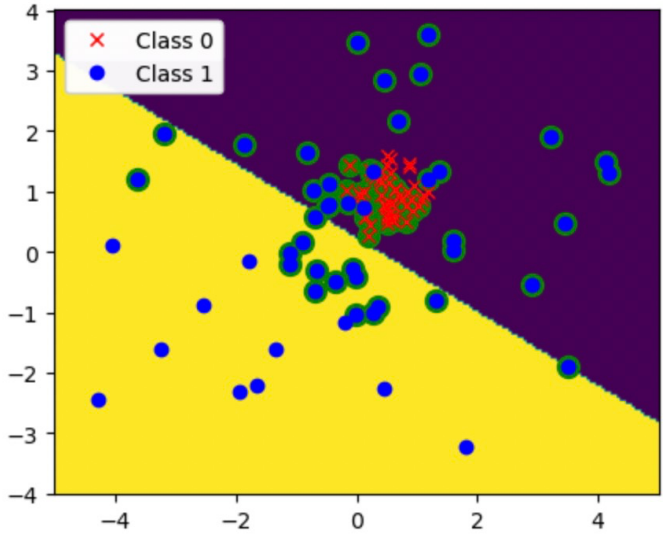
The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors. Therefore, the larger the gamma is, the radius of the area of influence of the support vectors only includes the support vector itself. As the gamma getting larger, the decision boundary is getting closer to the support vectors.

When gamma equals and larger then 10, there exist underfitting according to the low accuracy of both train dataset and test dataset.

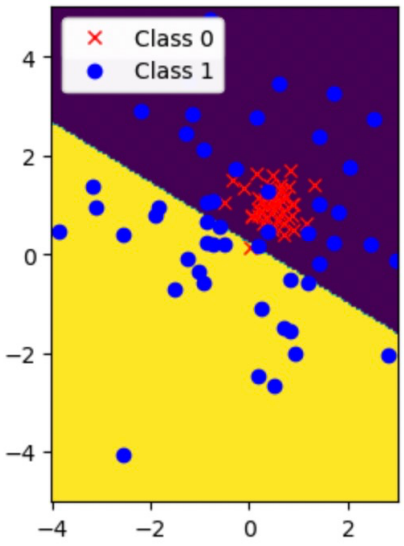
(c) C = 1

```
train acc = 0.76, test acc = 0.74
weight vector w =  [[-0.48069371 -0.78654222]]
offset w0 = [0.1697672]
```

Train dataset3:



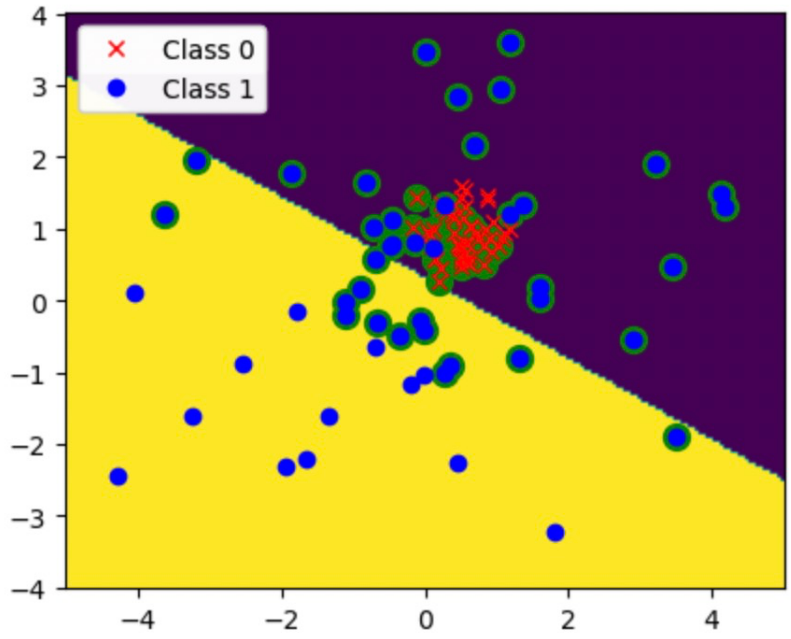
Test dataset3:



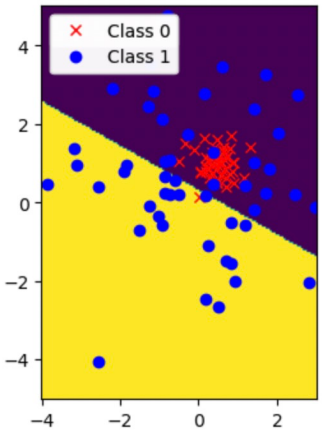
C = 100

```
train acc = 0.77, test acc = 0.75
weight vector w =  [[-0.49088354 -0.86901288]]
offset w0 = [0.26395834]
```

Train dataset3:



Test dataset3:

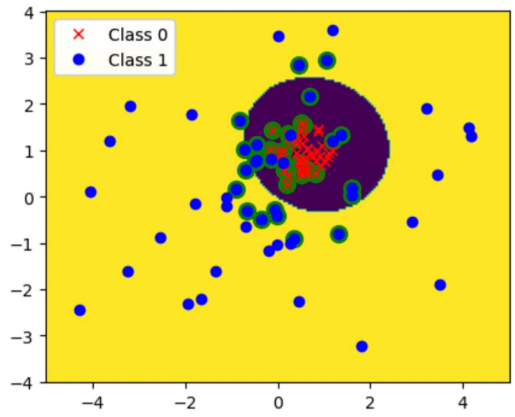


The distribution of dataset3 is not linear. Therefore, the linear kernel cannot classify the data 100% correctly and the performance of both values of C is almost the same.

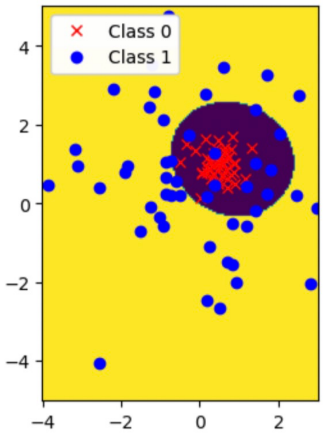
(d) C = 1, γ = 0.1

```
train acc = 0.89, test acc = 0.89
```

Train dataset3:



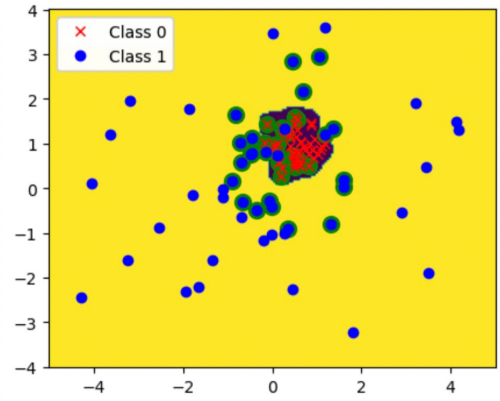
Test dataset3:



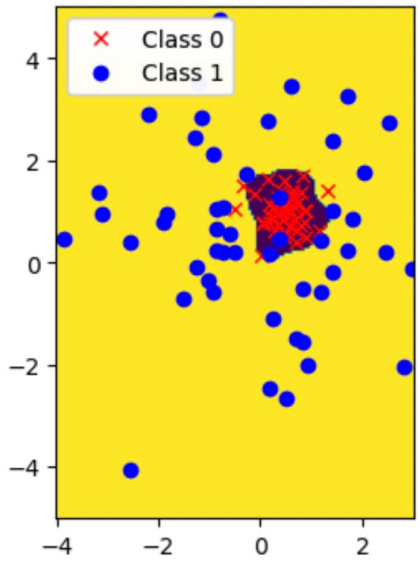
C = 1, γ = 10

```
train acc = 0.98, test acc = 0.94
```

Train dataset3:



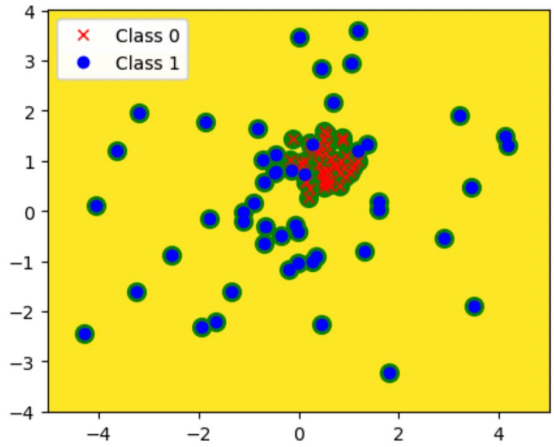
Test dataset3:



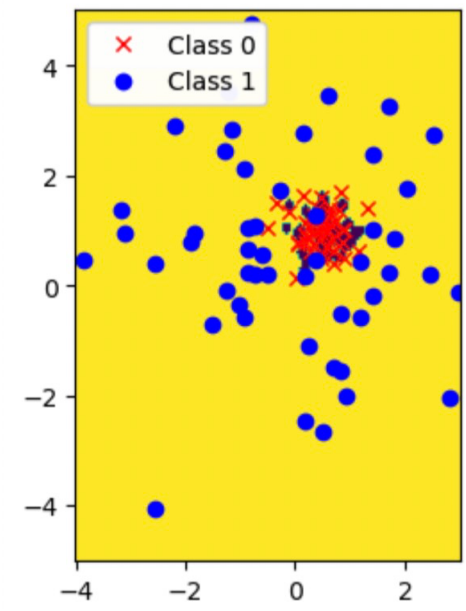
C = 1, γ = 200

```
train acc = 0.98, test acc = 0.77
```

Train dataset3:



Test dataset3:



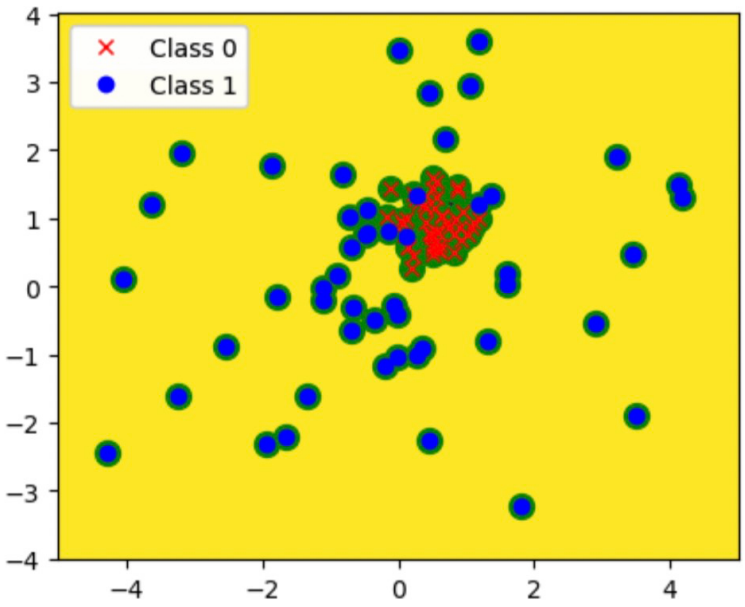
Explain the difference in decision regions for the different values of γ. Do you observe any overfitting or underfitting for any of the given values of γ?

The larger the value of gamma is, the closer the decision boundary are to the support vectors. When the gamma is too large (e.g. 200), there existing overfitting according to the high accuracy of train dataset and low accuracy of test dataset.

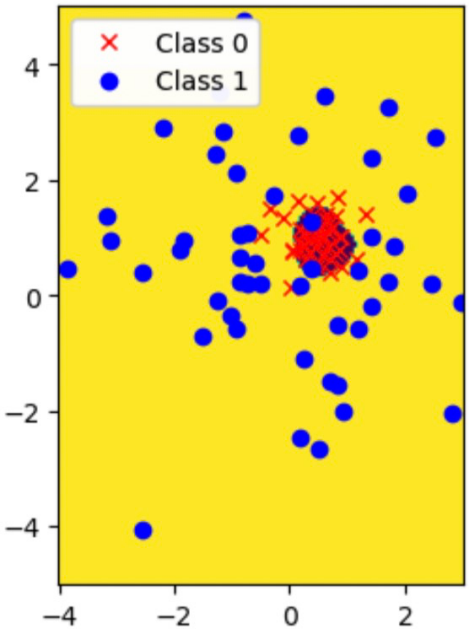
(e) C = 0.01, γ = 10

```
train acc = 0.88, test acc = 0.86
```

Train dataset3:



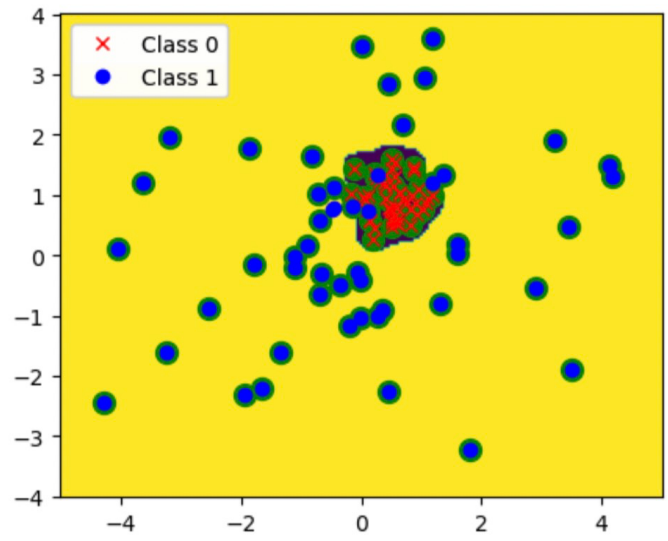
Test dataset3:



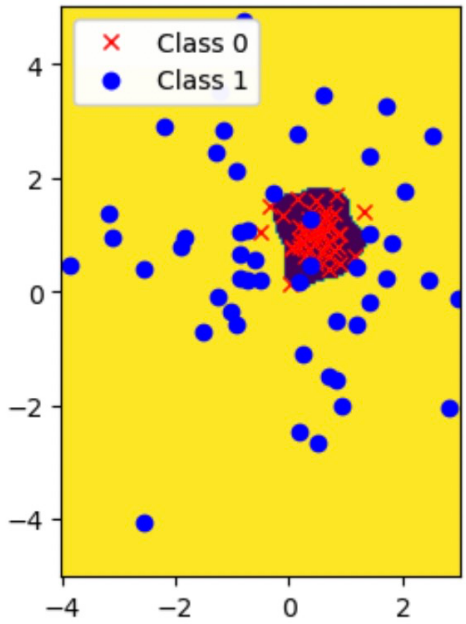
C = 1, γ = 10

```
train acc = 0.98, test acc = 0.94
```

Train dataset3:



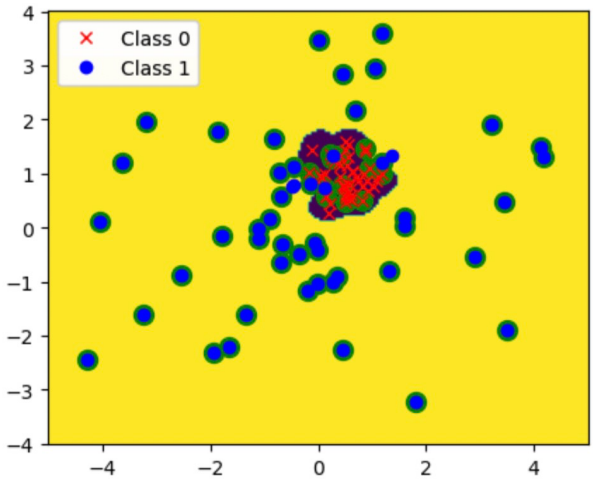
Test dataset3:



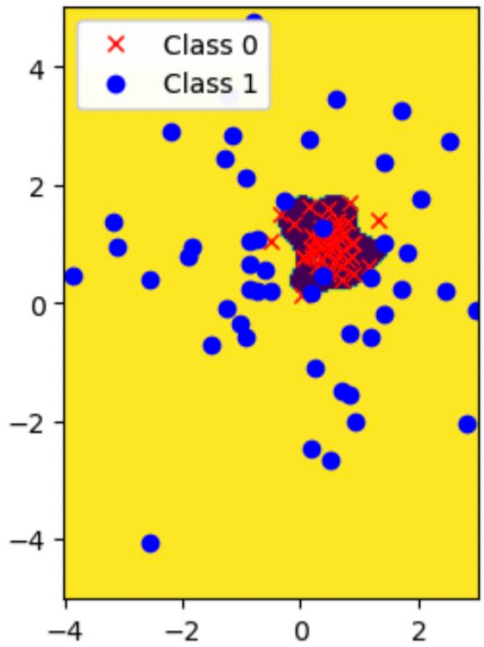
C = 100, γ = 10

```
train acc = 0.98, test acc = 0.95
```

Train dataset3:



Test dataset3:



Explain your observations in the different decision boundaries and the support vectors for the different values of C.

While the increasing of the value of C, the margin between the datapoints from class 0 and the decision boundary become larger. For a small value of C, every datapoints in class 0 are selected as support vectors. For a large value of C, only a few datapoints near the boundary are selected.

2. Backprop Initialization for Multi-class classification

1. What is the dimension of each of the following quantities:
$\frac{\partial \hat{\underline{y}}}{\partial \underline{a}}$, $\nabla_{\hat{\underline{y}}} J$ and $\underline{\delta}^{(L)}$?

(a) $\hat{\underline{y}} = \underline{z}^{(L)} = \text{SoftMax}(\underline{a}) = \underline{t}(\underline{a})$

$$\frac{\partial \underline{t}(\underline{a})}{\partial \underline{a}} = \begin{bmatrix} \frac{\partial t(\underline{a})}{\partial a_1} \\ \frac{\partial t(\underline{a})}{\partial a_2} \\ \vdots \\ \frac{\partial t(\underline{a})}{\partial a_C} \end{bmatrix} \quad \text{for } \underline{a} \in \mathbb{R}^C$$

Also, we should extend this to the derivative of a vector-valued function with respect to a vector argument.

$$\therefore \frac{\partial \underline{t}(\underline{a})}{\partial \underline{a}} = \begin{bmatrix} \frac{\partial t_1(\underline{a})}{\partial a_1} & \cdots & \frac{\partial t_C(\underline{a})}{\partial a_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial t_1(\underline{a})}{\partial a_M} & \cdots & \frac{\partial t_C(\underline{a})}{\partial a_C} \end{bmatrix}, \; \underline{t}(\underline{a}) \text{ mapping } \mathbb{R}$$

∴ The dimension of $\frac{\partial \hat{\underline{y}}}{\partial \underline{a}}$ is $C \times C$.

(b). $J = -\sum_{i=1}^{C} y_i \ln \hat{y}_i$

$$\nabla_{\hat{\underline{y}}} J = \begin{bmatrix} \frac{\partial J}{\partial \hat{y}_1} \\ \frac{\partial J}{\partial \hat{y}_2} \\ \vdots \\ \frac{\partial J}{\partial \hat{y}_C} \end{bmatrix}$$

∴ The dimension of $\nabla_{\hat{\underline{y}}} J$ is $C \times 1$

(c) $\delta = \nabla_{\underline{a}} J$

$= \underbrace{\frac{\partial \hat{\underline{y}}}{\partial \underline{a}}}_{C \times C} \times \underbrace{\nabla_{\hat{\underline{y}}} J}_{C \times 1}$

∴ The dimension of δ is $C \times 1$

2. Find $\frac{\partial \hat{\underline{y}}}{\partial \underline{a}}$

$\hat{\underline{y}} = \underline{v}^{(L)} = \underline{\text{SoftMax}}(\underline{a}) = \underline{f}(\underline{a})$

$$\frac{\partial \hat{\underline{y}}}{\partial \underline{a}} = \frac{d\,\underline{\text{SoftMax}}(\underline{a})}{d(\underline{a})} = \begin{bmatrix} \frac{\partial f_1(\underline{a})}{\partial a_1} & \cdots & \frac{\partial f_C(\underline{a})}{\partial a_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1(\underline{a})}{\partial a_C} & \cdots & \frac{\partial f_C(\underline{a})}{\partial a_C} \end{bmatrix}$$

$f_j(\underline{a}) = \frac{e^{a_j}}{D(\underline{a})} \quad D(\underline{a}) = \sum_i^C e^{a_i}$

$\frac{\partial f_j(\underline{a})}{\partial a_0} = \frac{\frac{\partial a_j}{\partial a_i}}{D(\underline{a})} - \frac{e^{a_j}}{[D(\underline{a})]^2} \cdot \frac{\partial D(\underline{a})}{\partial a_i}$

$\frac{\partial D(\underline{a})}{\partial a_i} = e^{a_i}$

$\frac{\partial a_j}{\partial a_i} = e^{a_i} \delta_{i,j}$ where $\delta_{i,j} = 0$ if $i \neq j$ and 1 if $i = j$

$\therefore \frac{\partial f_j(\underline{a})}{\partial a_i} = \frac{e^{a_j}}{D(\underline{a})} \delta_{i,j} - \frac{e^{a_j}}{[D(\underline{a})]^2} \cdot e^{a_i} = f_j(\underline{a}) \delta_{i,j} - f_j(\underline{a}) f_i(\underline{a})$

$= \begin{cases} f_i(\underline{a})[1 - f_i(\underline{a})] & i = j \\ -f_i(\underline{a}) f_j(\underline{a}) & i \neq j \end{cases}$

$$\therefore \left[\frac{\partial \hat{\underline{p}}}{\partial \underline{a}}\right]_{i,j} = \begin{cases} \hat{p}_i(1-\hat{p}_i) & i=j \\ -\hat{p}_i\hat{p}_j & i\neq j \end{cases}$$

$$\therefore \frac{\partial \hat{\underline{p}}}{\partial \underline{a}} = \begin{bmatrix} \hat{p}_1(1-\hat{p}_1) & -\hat{p}_1\hat{p}_2 & \cdots & -\hat{p}_1\hat{p}_c \\ -\hat{p}_1\hat{p}_2 & \hat{p}_2(1-\hat{p}_2) & \cdots & -\hat{p}_2\hat{p}_c \\ \vdots & & \ddots & \vdots \\ -\hat{p}_1\hat{p}_c & \cdots & -\hat{p}_c\hat{p}_{c-1} & \hat{p}_c(1-\hat{p}_c) \end{bmatrix}$$

3. Find $\nabla_{\hat{\underline{p}}} J$

$$J = -\sum_{i=1}^{c} p_i \ln \hat{p}_i$$

$$\nabla_{\hat{p}_i} J = \frac{\partial}{\partial \hat{p}_i}\left(-\sum_{i=1}^{c} p_i \ln \hat{p}_i\right)$$

$$= \frac{\partial}{\partial \ln \hat{p}_i}\left(-\sum_{i=1}^{c} p_i \ln \hat{p}_i\right) \cdot \frac{\partial \ln \hat{p}_i}{\partial \hat{p}_i}$$

$$= -p_i \cdot \frac{1}{\hat{p}_i} = -\frac{p_i}{\hat{p}_i}$$

$$\therefore \nabla_{\hat{\underline{p}}} J = \begin{bmatrix} -p_1/\hat{p}_1 \\ -p_2/\hat{p}_2 \\ \vdots \\ -p_c/\hat{p}_c \end{bmatrix} = -\begin{bmatrix} p_1/\hat{p}_1 \\ p_2/\hat{p}_2 \\ \vdots \\ p_c/\hat{p}_c \end{bmatrix}$$

4. Show that $\underline{\delta}^{(L)} = \hat{\underline{p}} - \underline{p}$

$$\delta = \nabla_{\underline{a}} J = \frac{\partial \hat{\underline{p}}}{\partial \underline{a}} \cdot \nabla_{\hat{\underline{p}}} J$$

$$= \begin{bmatrix} -(1-\hat{p}_1)p_1 + \hat{p}_1 p_2 + \hat{p}_1 p_3 + \cdots \hat{p}_1 p_c \\ \hat{p}_2 p_1 - (1-\hat{p}_2)p_2 + \hat{p}_2 p_3 + \cdots \hat{p}_2 p_c \\ \vdots \\ \hat{p}_c p_1 + \hat{p}_c p_2 + \hat{p}_c p_3 + \cdots -(1-\hat{p}_c)p_c \end{bmatrix}$$

Consider the first element in this vector.

$$\delta_0 = -(1-\hat{p}_1)p_1 + \hat{p}_1 p_2 + \hat{p}_1 p_3 + \cdots \hat{p}_1 p_c$$

$$= -p_1 + \hat{p}_1 \sum_{m=1}^{c} p_m \quad \text{where } p_m \text{ is the label}$$

$$\therefore \sum_{m=1}^{c} p_m = 1$$

$$\therefore \delta_0 = \hat{p}_1 - p_1$$

And apply this to each row of the vector, hence

$$\therefore \delta = \hat{\underline{p}} - \underline{p}$$

$$\delta^{(L)} = \hat{\underline{p}} - \underline{p}$$

3. Backprop by Hand

1. Forward Computation

$$\underline{a}^{(1)} = \underline{\underline{W}}^{(1)} \cdot \underline{x} + \underline{b}^{(1)} = \begin{bmatrix} 1 & -2 & 1 \\ 3 & 4 & -2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} 4 \\ -3 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \end{bmatrix} = \begin{bmatrix} 5 \\ -5 \end{bmatrix}$$

$$\underline{v}^{(1)} = \text{Relu}(a^{(1)}) = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$$

$$\dot{\underline{v}}^{(1)} = \begin{bmatrix} 1 \\ 1/2 \end{bmatrix}$$

$$a^{(2)} = \underline{\underline{W}}^{(2)} \cdot \underline{v}^{(1)} + b^{(2)} = \begin{bmatrix} 1 & -2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 5 \\ 15 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ 15 \end{bmatrix}$$

$$\underline{v}^{(2)} = \text{Relu}(a^{(2)}) = \begin{bmatrix} 6 \\ 15 \end{bmatrix}$$

$$\dot{\underline{v}}^{(2)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\underline{a}^{(3)} = \underline{\underline{W}}^{(3)} \underline{v}^{(2)} + b^{(3)} = \begin{bmatrix} 2 & 2 \\ 3 & -3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 6 \\ 15 \end{bmatrix} + \begin{bmatrix} 0 \\ -4 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} 42 \\ -27 \\ 27 \end{bmatrix} + \begin{bmatrix} 0 \\ -4 \\ -2 \end{bmatrix} = \begin{bmatrix} 42 \\ -31 \\ 25 \end{bmatrix}$$

$$\underline{v}^{(3)} = \text{SoftMax}(\underline{a}^{(3)}) = \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ 4.139 \times 10^{-8} \end{bmatrix}$$

2. Back-propagation Computation

$$\delta^{(3)} = \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ 4.139 \times 10^{-8} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ -0.999 \end{bmatrix}$$

$$\underline{\underline{W}}^{(3)}(1) = \underline{\underline{W}}^{(3)} - \eta\, \delta^{(3)} \left[v^{(2)}\right]^T$$

$$= \begin{bmatrix} 2 & 2 \\ 3 & -3 \\ 2 & 1 \end{bmatrix} - 0.5 \times \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ -0.999 \end{bmatrix} \begin{bmatrix} 6 & 15 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 2 \\ 3 & -3 \\ 2 & 1 \end{bmatrix} - 0.5 \times \begin{bmatrix} 5.994 & 14.985 \\ 1.1874 \times 10^{-31} & 2.9685 \times 10^{-31} \\ -5.994 & -14.985 \end{bmatrix}$$

$$= \begin{bmatrix} -0.997 & -5.4925 \\ 3 & -3 \\ 4.997 & 8.49925 \end{bmatrix}$$

$$\underline{b}^{(3)}(1) = \underline{b}^{(3)} - \eta\, \delta^{(3)} = \begin{bmatrix} 0 \\ -4 \\ 2 \end{bmatrix} - 0.5 \times \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ -0.999 \end{bmatrix}$$

$$= \begin{bmatrix} -0.4995 \\ -4 \\ -1.50005 \end{bmatrix}$$

$$\delta^{(2)} = \dot{v}^{(2)} \odot \left[\left(\underline{\underline{W}}^{(3)}\right)^T \delta^{(3)}\right]$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} \odot \begin{bmatrix} 2 & 3 & 2 \\ 2 & -3 & 1 \end{bmatrix} \begin{bmatrix} 0.999 \\ 1.979 \times 10^{-32} \\ -0.999 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} \odot \begin{bmatrix} 0 \\ 0.999 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0.999 \end{bmatrix}$$

$$\underline{\underline{W}}^{(2)}(1) = \underline{\underline{W}}^{(2)} - \eta\, \delta^{(2)} \left[v^{(1)}\right]^T$$

$$= \begin{bmatrix} 1 & -2 \\ 3 & 4 \end{bmatrix} - 0.5 \cdot \begin{bmatrix} 0 \\ 0.999 \end{bmatrix} \begin{bmatrix} 5 \\ 0 \end{bmatrix}^T$$

$$= \begin{bmatrix} 1 & -2 \\ 3 & 4 \end{bmatrix} - 0.5 \begin{bmatrix} 0 & 0 \\ 4.995 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -2 \\ 0.5025 & 4 \end{bmatrix}$$

$$\vec{b}^{(2)}(1) = \vec{b}^{(2)} - \eta \delta^{(2)}$$

$$= \begin{bmatrix} 1 \\ 0 \end{bmatrix} - 0.5 \begin{bmatrix} 0 \\ 0.999 \end{bmatrix} =$$

$$= \begin{bmatrix} 1 \\ -0.4995 \end{bmatrix}$$

$$\vec{\delta}^{(1)} = \vec{\lambda}^{(1)} \odot \left[(\vec{m}^{(2)})^{\perp} \delta^{(2)} \right]$$

$$= \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \odot \begin{bmatrix} 1 & 3 \\ -2 & 4 \end{bmatrix} \begin{bmatrix} 0 \\ 0.999 \end{bmatrix} =$$

$$= \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \odot \begin{bmatrix} 2.997 \\ 3.996 \end{bmatrix} =$$

$$= \begin{bmatrix} 2.997 \\ 1.998 \end{bmatrix}$$

$$\vec{m}^{(1)}(1) = \vec{m}^{(1)} - \eta \delta^{(1)} \left[\vec{x} \right]^{\perp}$$

$$= \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \\ 3 & 4 & -2 \end{bmatrix} - 0.5 \begin{bmatrix} 2.997 \\ 1.998 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & -2 & 1 \\ 3 & 4 & -2 \end{bmatrix} - 0.5 \begin{bmatrix} 2.997 & -2.997 & 2.997 \\ 1.998 & -1.998 & 1.998 \end{bmatrix} =$$

$$= \begin{bmatrix} -0.4985 & -0.5015 & -0.4985 \\ 2.001 & 4.999 & -2.999 \end{bmatrix}$$

$$\vec{b}^{(1)}(1) = \vec{b}^{(1)} - \eta \delta^{(1)}$$

$$= \begin{bmatrix} 1 \\ -2 \end{bmatrix} - 0.5 \begin{bmatrix} 2.997 \\ 1.998 \end{bmatrix} =$$

$$= \begin{bmatrix} -0.4985 \\ -2.999 \end{bmatrix}$$

3. $\underline{a}^{(3)} = \underline{\underline{W}}^{(3)} \underline{v}^{(2)} + \underline{b}^{(3)}$

$= \underline{\underline{W}}^{(3)} \max(0, \underline{\underline{W}}^{(2)} \underline{v}^{(1)} + \underline{b}^{(2)}) + \underline{b}^{(3)}$

$= \underline{\underline{W}}^{(3)} \max(0, \underline{\underline{W}}^{(2)} \max(0, \underline{\underline{W}}^{(1)} \underline{x} + \underline{b}^{(1)}) + \underline{b}^{(2)}) + \underline{b}^{(3)}$

For those specitic input :

$\underline{a}^{(3)} = \underline{\underline{W}}^{(3)} \left[\underline{\underline{W}}^{(2)} (\underline{\underline{W}}^{(1)} \underline{x} + \underline{b}^{(1)}) + \underline{b}^{(2)} \right] + \underline{b}^{(3)}$

$= \underline{\underline{W}}^{(3)} \left[\underline{\underline{W}}^{(2)} \underline{\underline{W}}^{(1)} \underline{x} + \underline{\underline{W}}^{(2)} \underline{b}^{(1)} + \underline{b}^{(2)} \right] + \underline{b}^{(3)}$

$= \underline{\underline{W}}^{(3)} \underline{\underline{W}}^{(2)} \underline{\underline{W}}^{(1)} \underline{x} + \underline{\underline{W}}^{(3)} \underline{\underline{W}}^{(2)} \underline{b}^{(1)} + \underline{\underline{W}}^{(3)} \underline{b}^{(2)} + \underline{b}^{(3)}$

$\therefore \underline{\underline{W}}_{eff} = \underline{\underline{W}}^{(3)} \underline{\underline{W}}^{(2)} \underline{\underline{W}}^{(1)}$

$\underline{b}_{eff} = \underline{\underline{W}}^{(3)} \underline{\underline{W}}^{(2)} \underline{b}^{(1)} + \underline{\underline{W}}^{(3)} \underline{b}^{(2)} + \underline{b}^{(3)}$

$\therefore$ No, these effective weights and bias are not a function of nodes in $\underline{x}$.

As we can see, $\underline{\underline{W}}_{eff}$ corresponds to the composition of all the weight matrices in the network and the bias $\underline{b}_{eff}$ also corresponds to the composition of weight matrices in the network (except the first one) and all of the bias vectors.

Comparing to a linear model, the mapping $\underline{\underline{W}}\underline{x} + \underline{b}$ affects on every node in $\underline{x}$ for fixed $\underline{\underline{W}}$ and $\underline{b}$. But for the Relu activation, when the pre-activation to a Relu is negative, the node will be removed from the network. Therefore, the ANN with Relu activations can learn non-linear distribution of input $\underline{x}$ by selectively activating different sets of nodes in input $\underline{x}$.