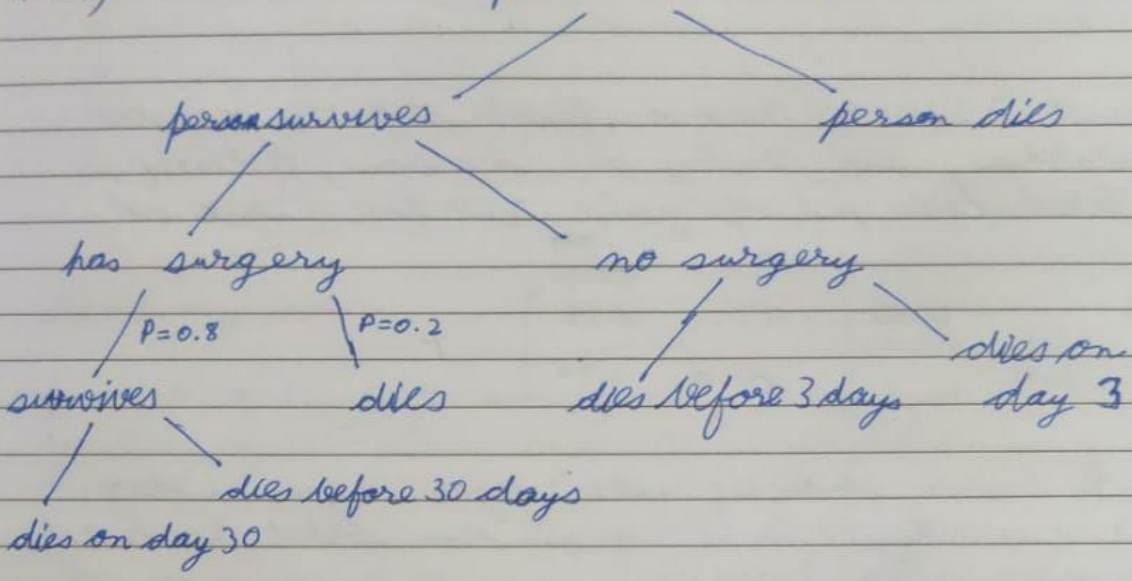
## Section A

13/10/22 Date .....

ML Assignment - 2

#1 1)




3) given, $P(\text{test +ve} \mid \text{survives}) = 0.95$

$P(\text{test +ve} \mid \text{dies}) = 0.05$

$P(\text{survives}) = 0.8$

We only ~~use~~ calculate with the ~~positive~~ positive ~~result~~ test results as we do not take into account the negative test scores.

To find ⇒ $P(\text{survives} \mid \text{test +ve})$ = (Bayes' theorem) $\frac{P(\text{test +ve} \mid \text{survives})}{P(\text{test +ve})} \times P(\text{survives})$

$= \frac{0.95 \times 0.8}{0.95 \times 0.8 + 0.05 \times 0.2}$

$= \frac{0.76}{0.77}$

$= 98.7\%$.

Spiral

Date .....

4) Yes, the surgery should be performed as chance of survival = 98.7%.

5) To check whether the test should be conducted before the operation, we need to find the probability that the test is positive, disease free and surgery is successful.

⇒ we need to find $P(\text{test +ve}) \times P(\text{survives, no disease} \mid \text{test +ve})$

We assume that test is conducted and -ve test is ignored.

Now, we find $P(\text{test +ve}) \times P(\text{survives} \mid \text{test +ve}) \times P(\text{no disease})$

$P(\text{no disease}) = 1 - 0.005 = 0.995$ (from test)

Plugging in all values,

$P = 0.77 \times 0.987 \times 0.995$

$= 0.75$

∴ the probability is less than 0.8 ⇒ test should not be conducted.

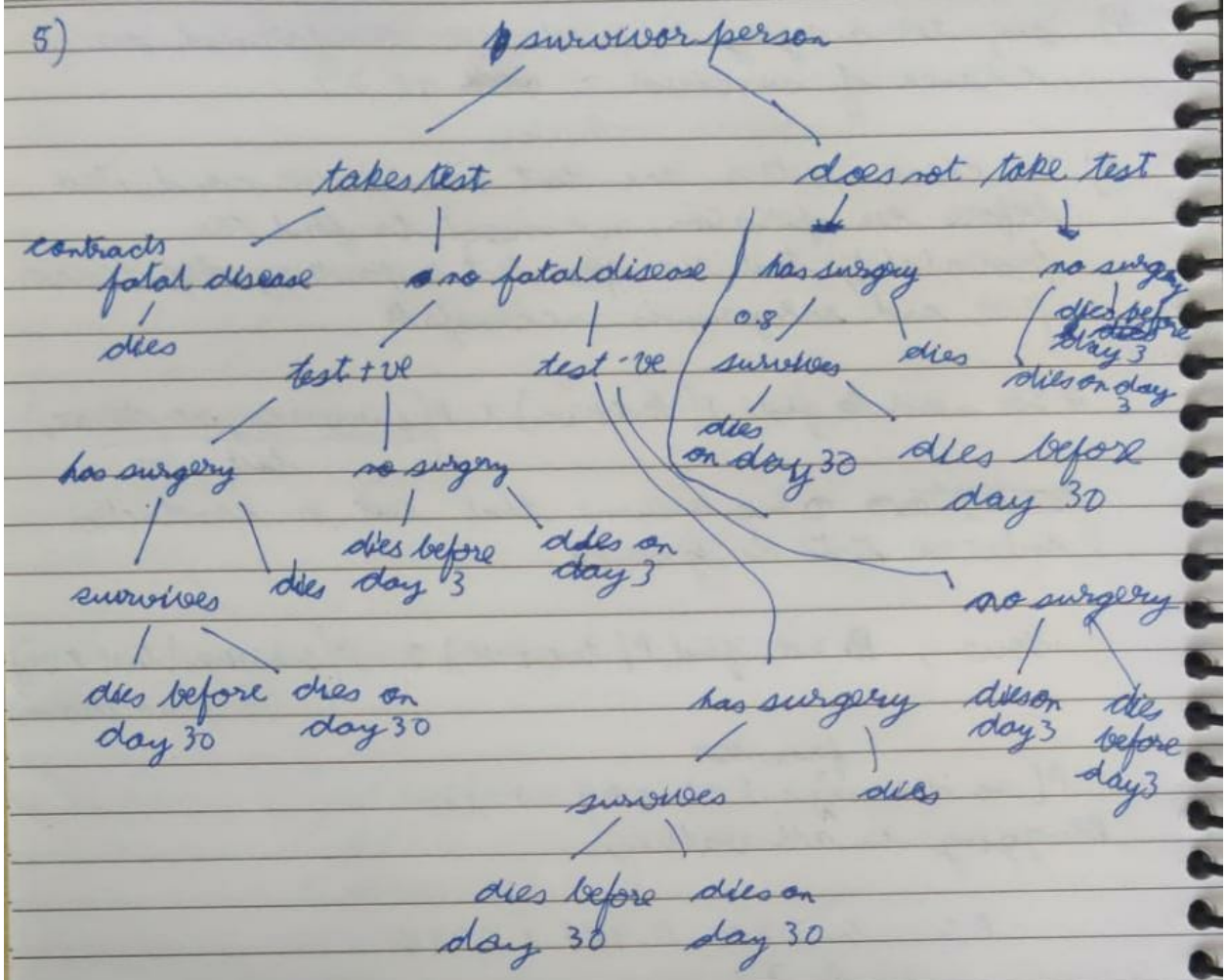
2) Given $L(30) = 1$, $L(0) = 0$, we assume L is a linear function and so every day adds $\frac{1}{30}$ to a patient's living function.
If the patient dies without surgery ⇒ $L(x) = L(3) = 0.1$
If patient lives for 'd' days after surgery ⇒ $L(x+d) = 0.1 + \frac{d}{30}$
For the surgery to be useful, the patient has to survive at least 3 more days ⇒ $d = 3$ ⇒

∴ $L(6) = 0.1 + 0.1 = \underline{0.2}$

Spiral

Date .....

5)

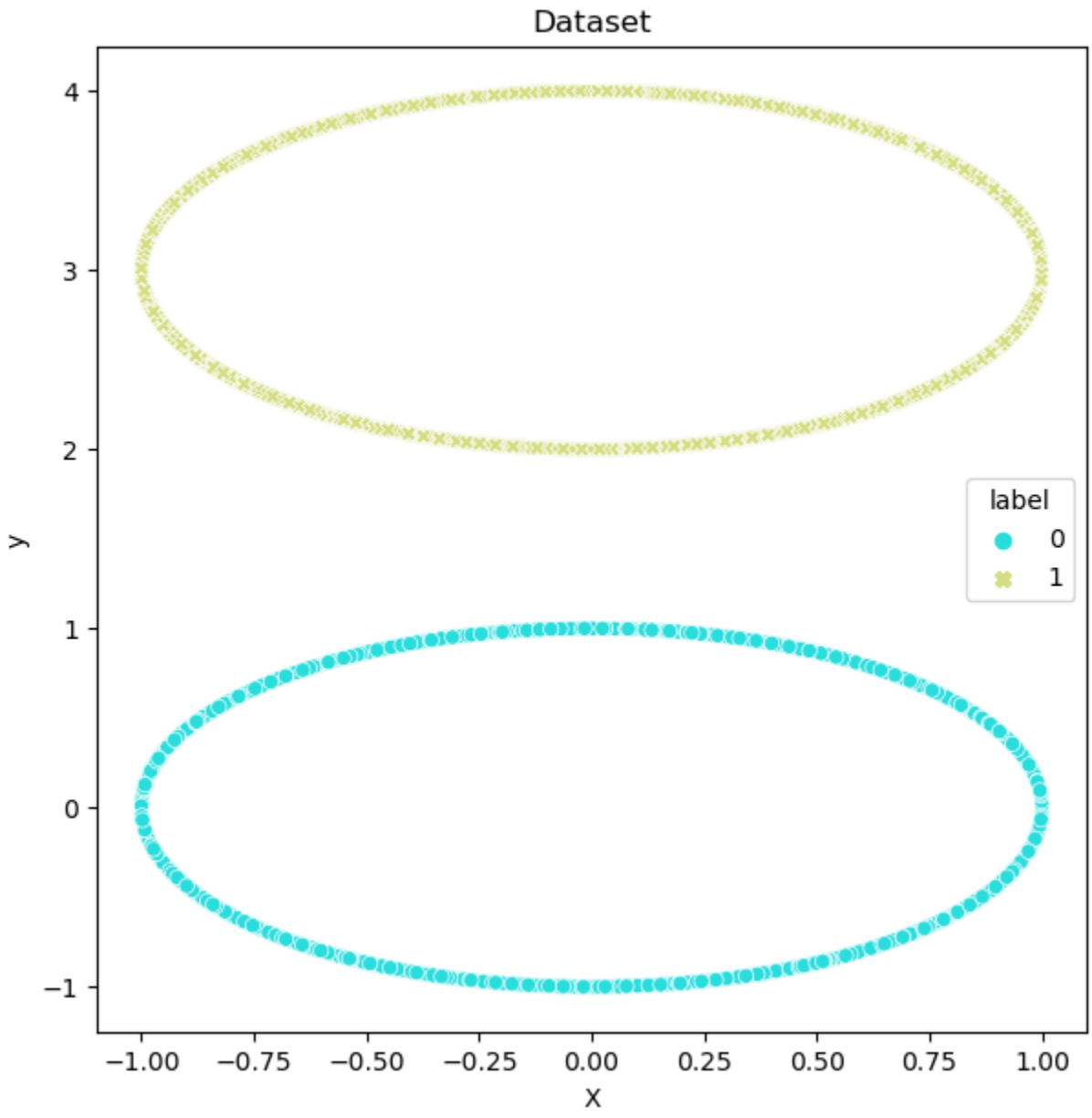



Spiral

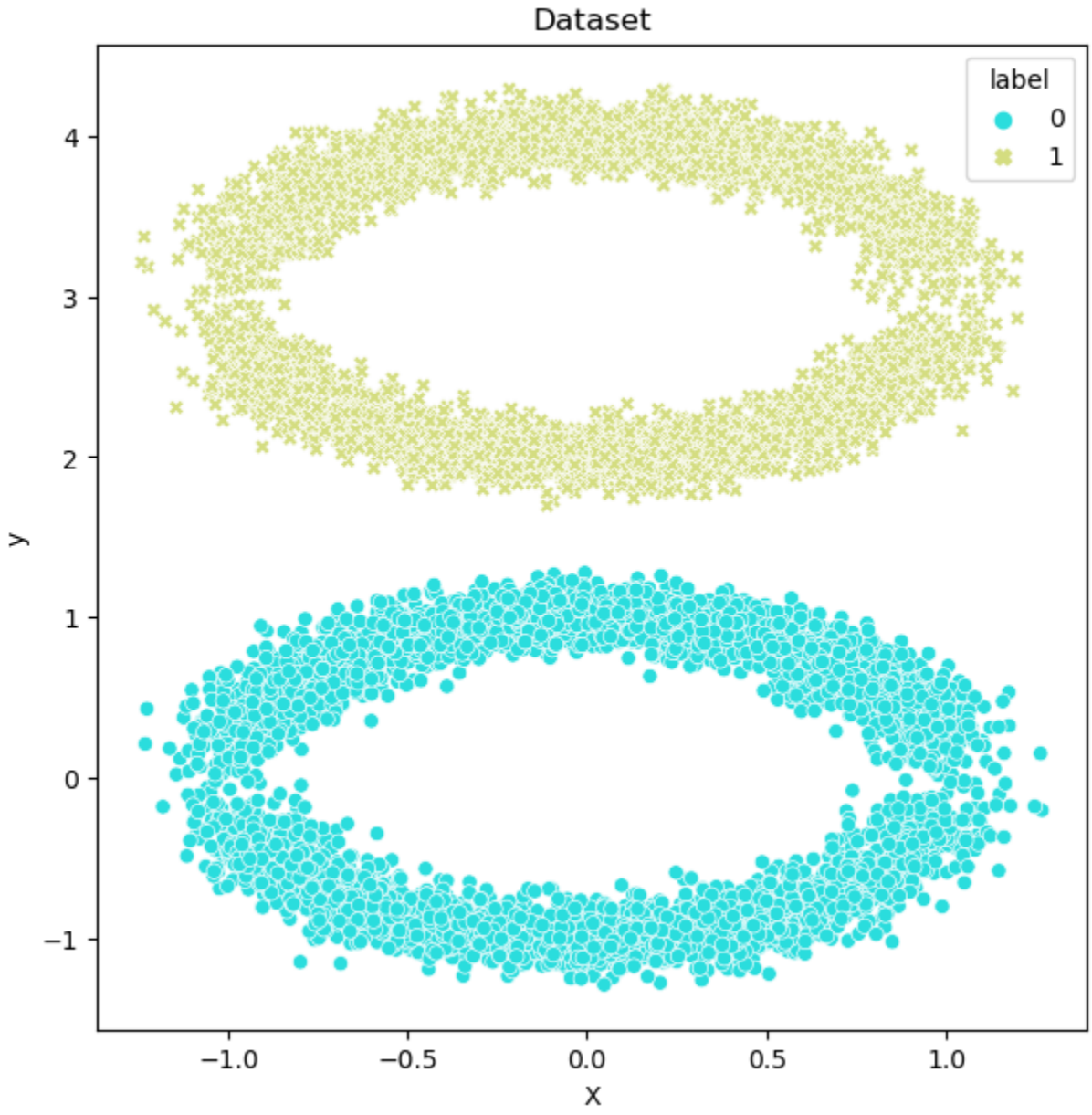
## Section B

### Part 2

Dataset without noise:

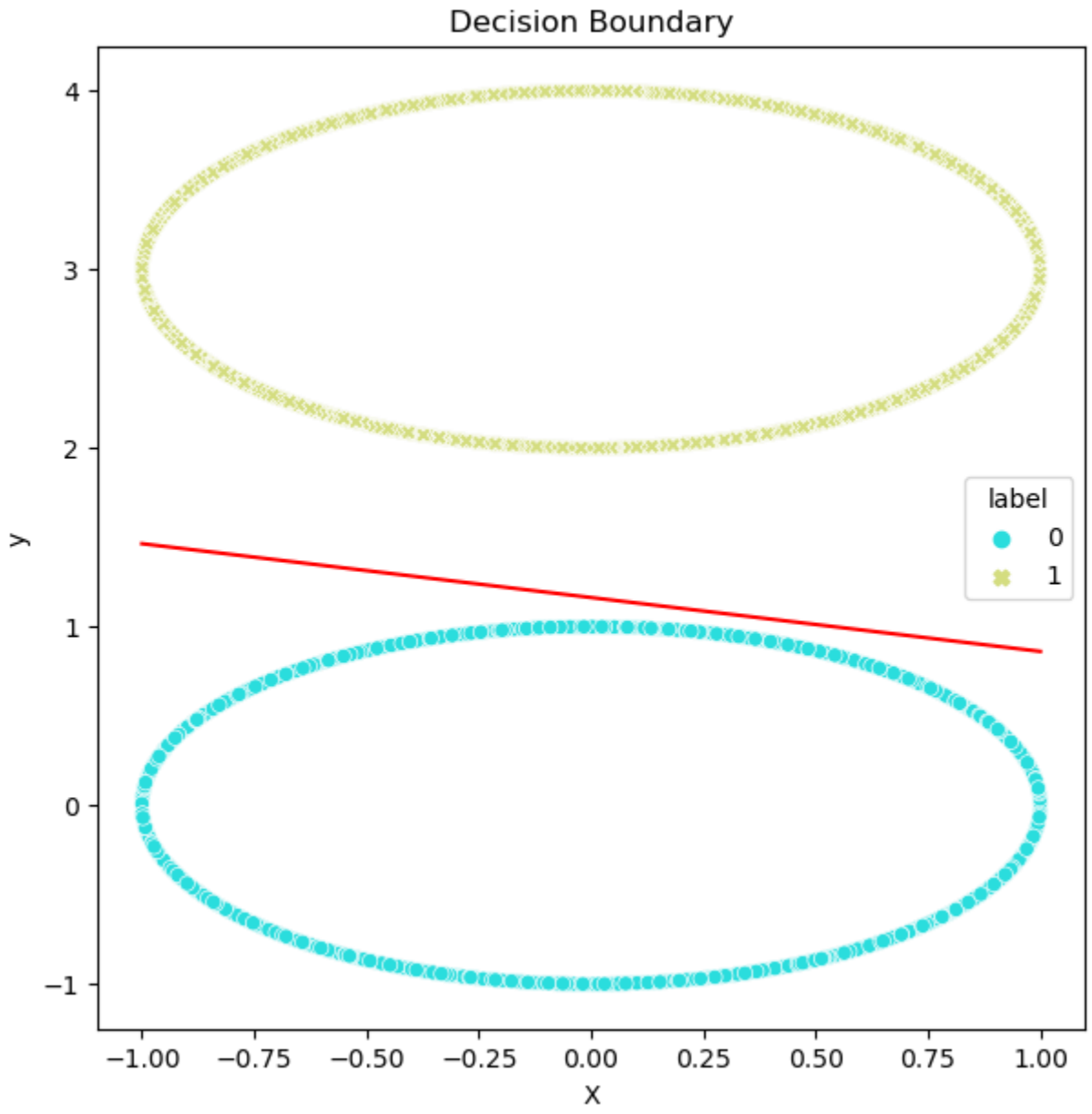


Dataset with noise:

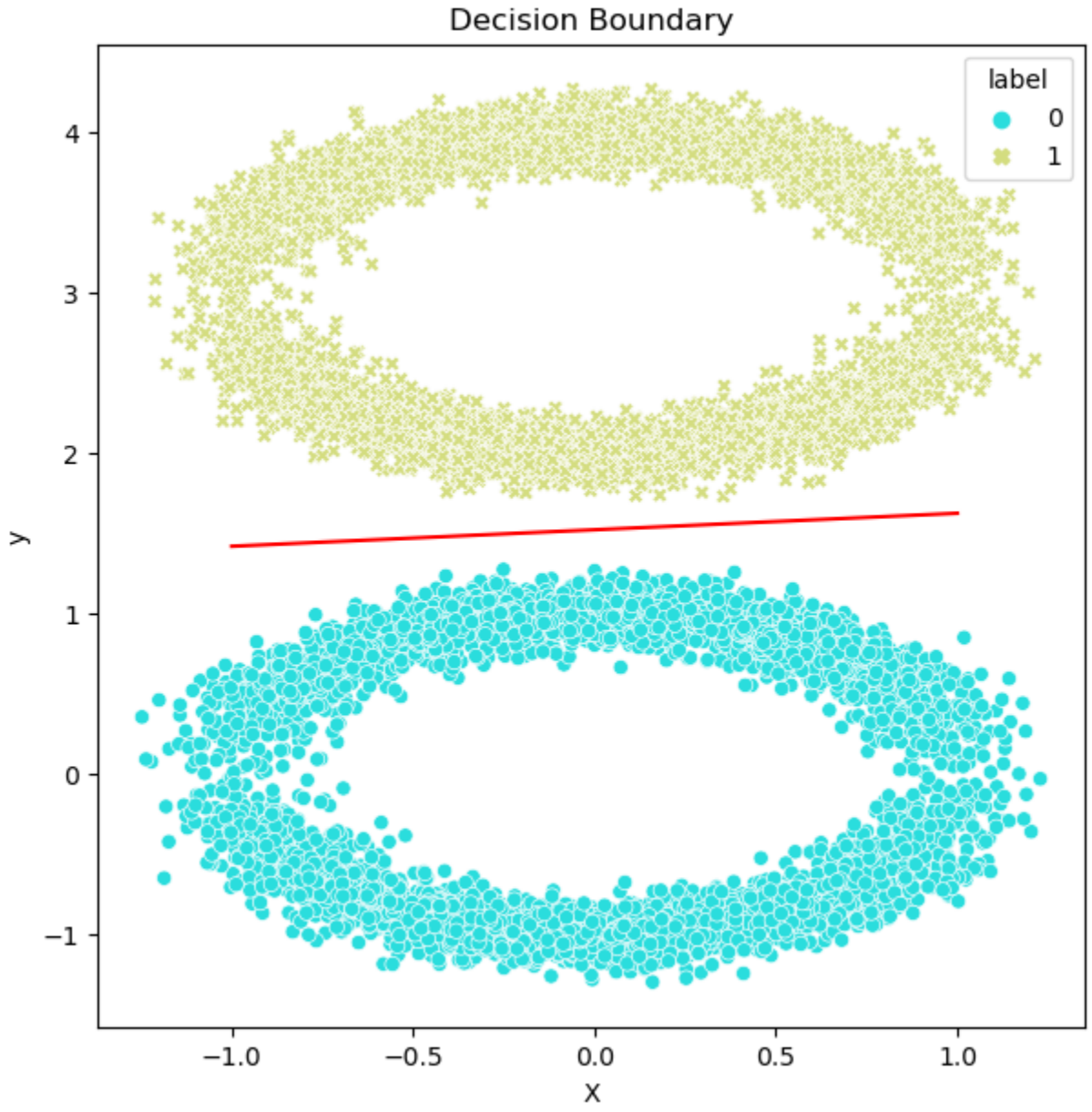


## Part 3

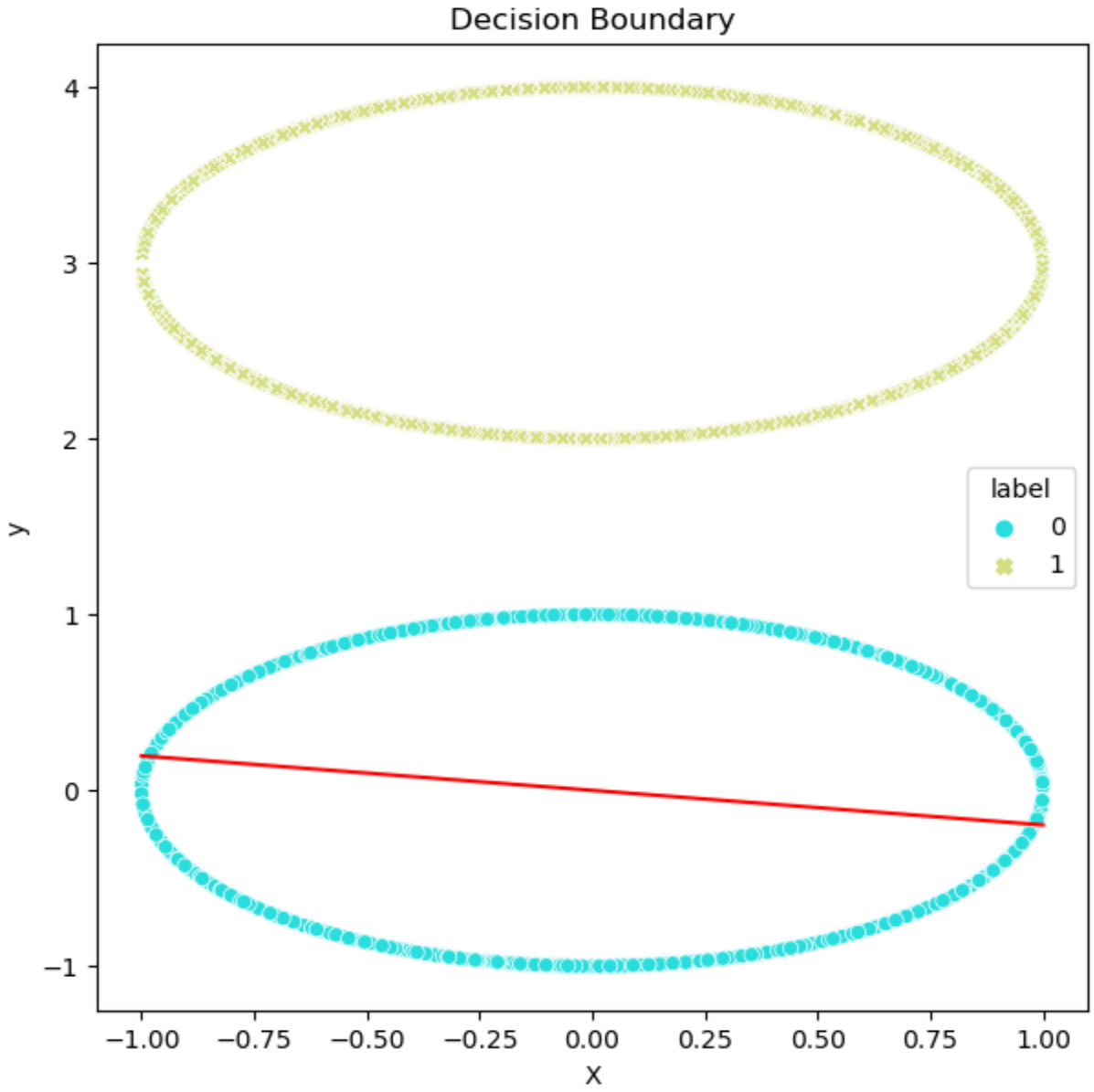
Decision boundary on the dataset without noise:



Decision boundary on the dataset without noise:



## Part 4



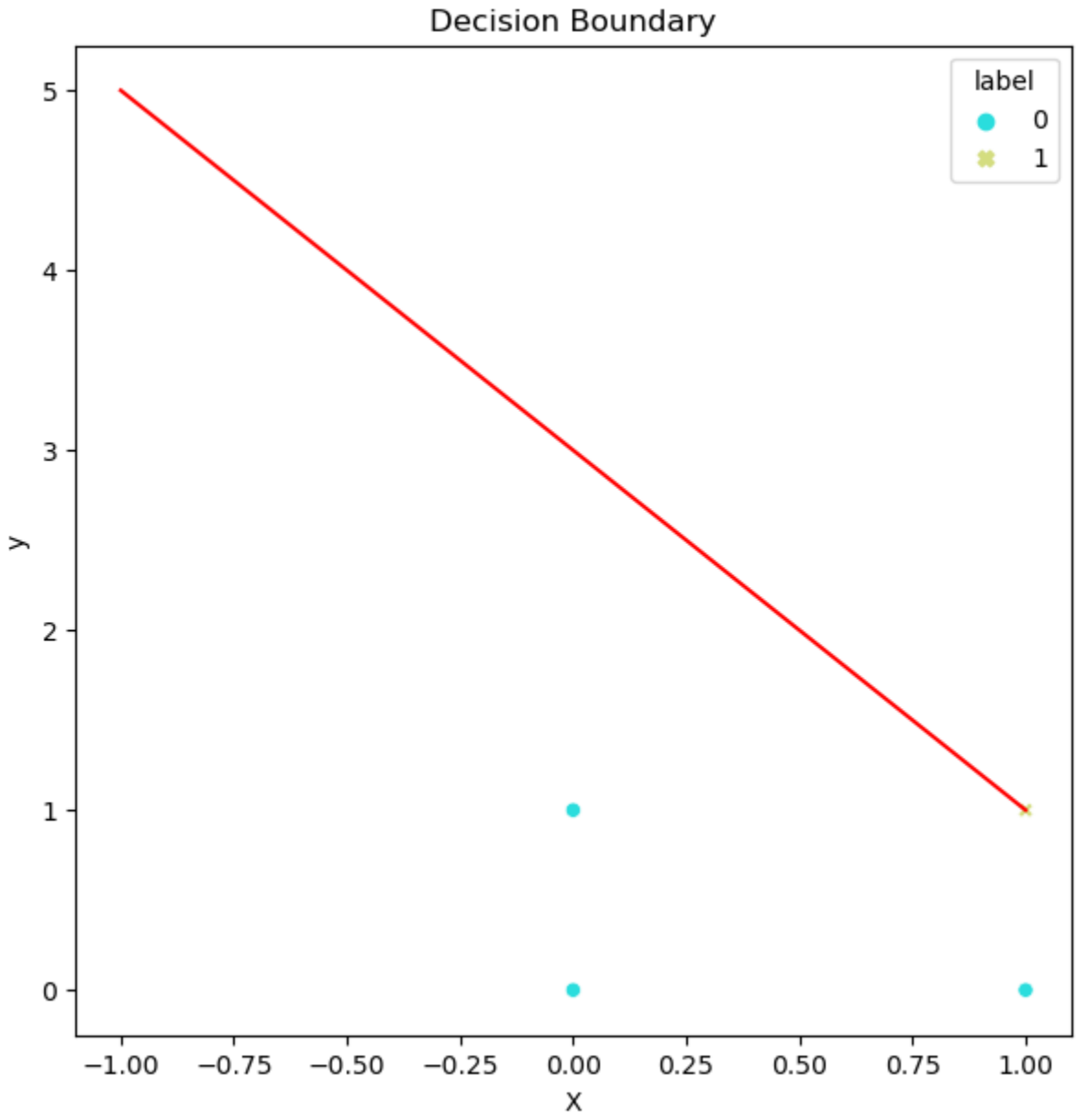

In part 2, the decision boundary correctly classifies the dataset, whereas the decision boundary in part 3 is incorrect (does not exist). This is because we set the bias to 0 after the training and convergence of the perceptron algorithm, which gives the correct weights and bias to create the correct decision boundary, as shown in part 2. However, by setting the bias to 0, we essentially remove the intercept from the line equation ($y = w^T x + b$), because of which the line passes through (0,0), i.e. origin. Thus, fixed bias perceptron is unable to distinguish between the classes.

As the circle with label 0 is centered on the origin, the line will always pass through that circle, and so, the decision boundary does not exist. The same happens in part 5 for fixed bias with AND, OR and XOR properties. No decision boundary is created when the bias is set to 0 after the convergence of the perceptron algorithm.

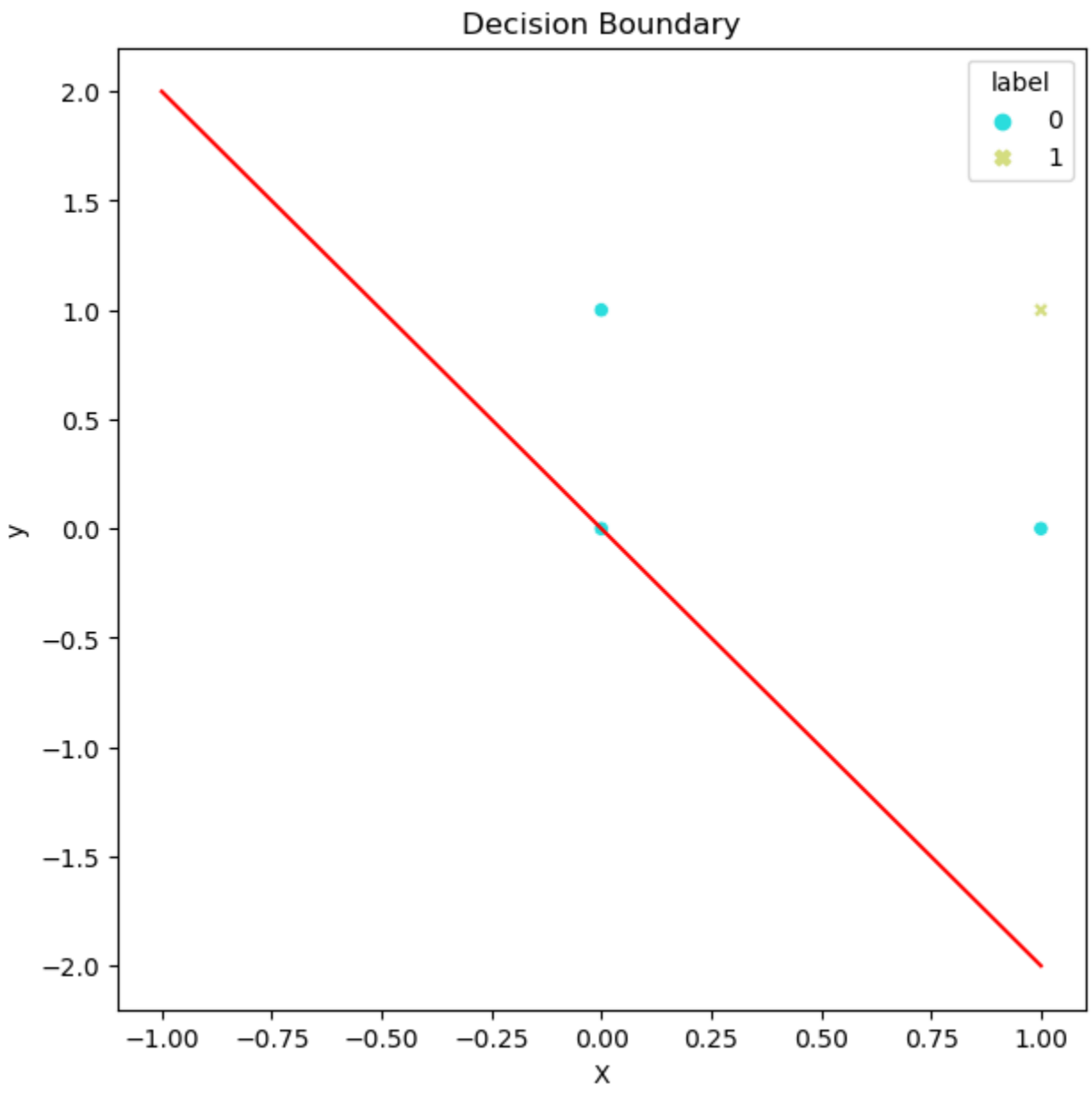
## Part 5

### AND Dataset

Learnable Bias:



Fixed Bias:

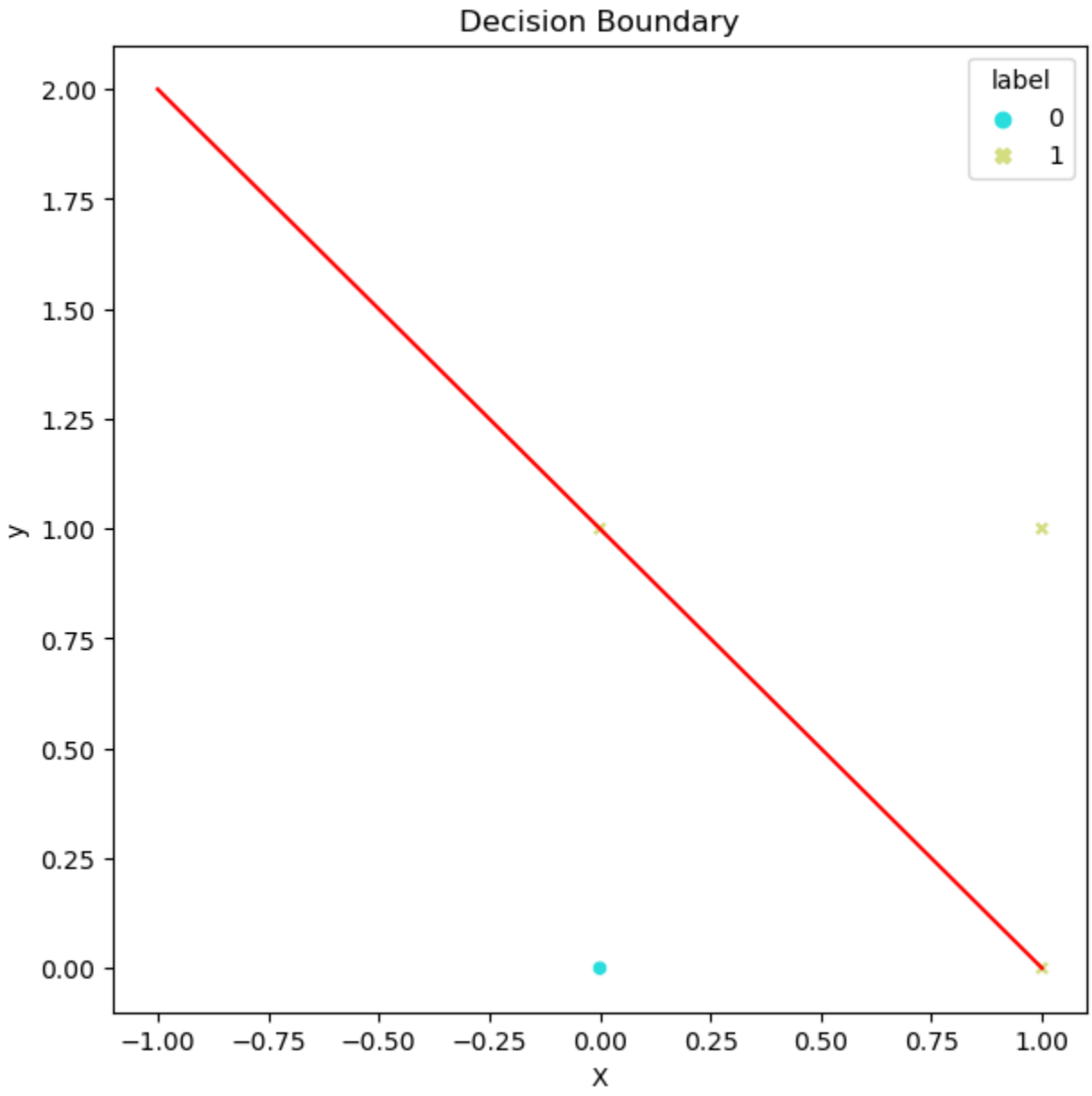



Decision boundary exists in the case of learnable bias. The line passes through the point because of the implementation done in the activation function.

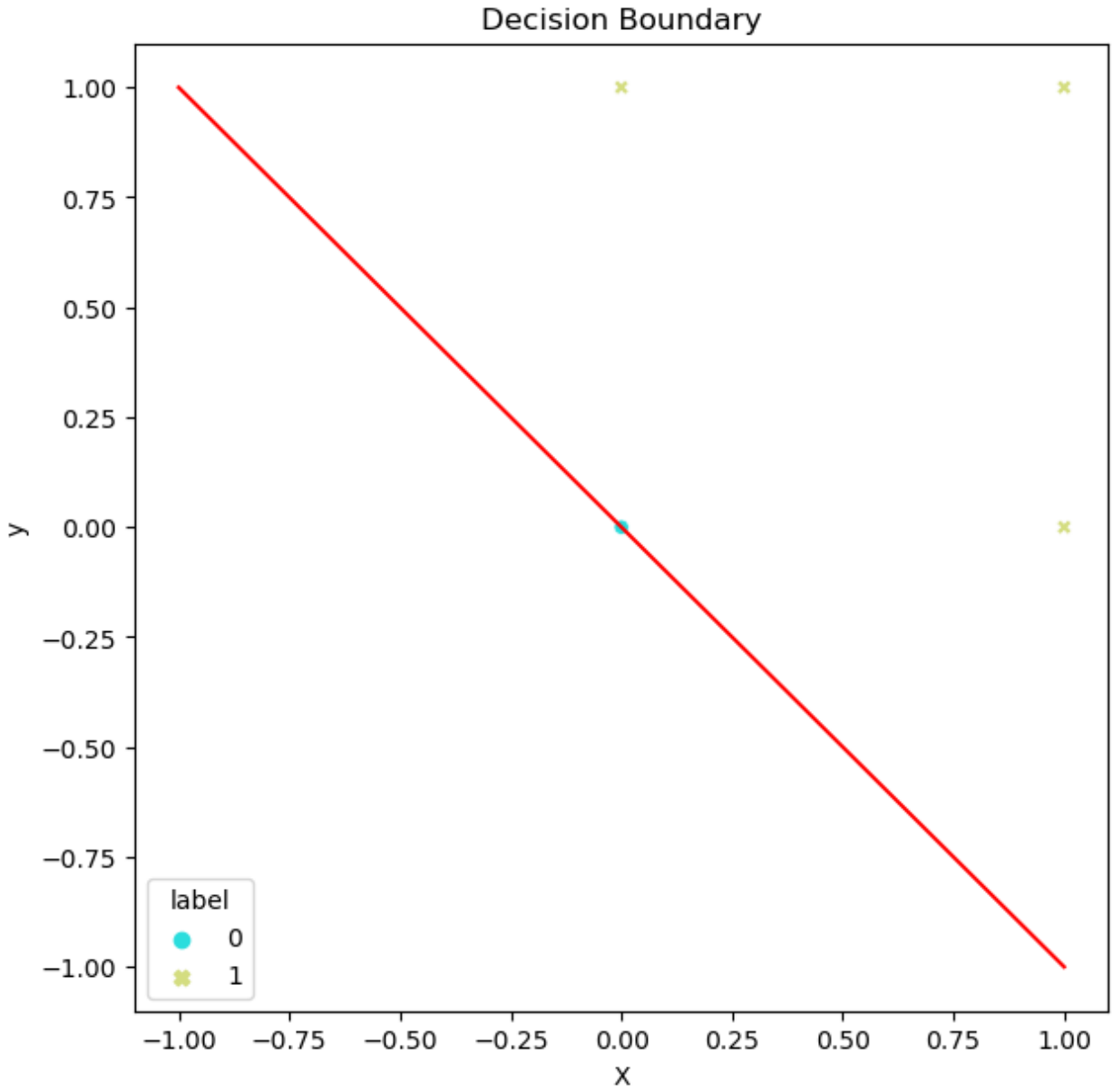
Decision boundary does not exist in the case of fixed bias.

**OR Dataset**

Learnable Bias:



Fixed Bias:

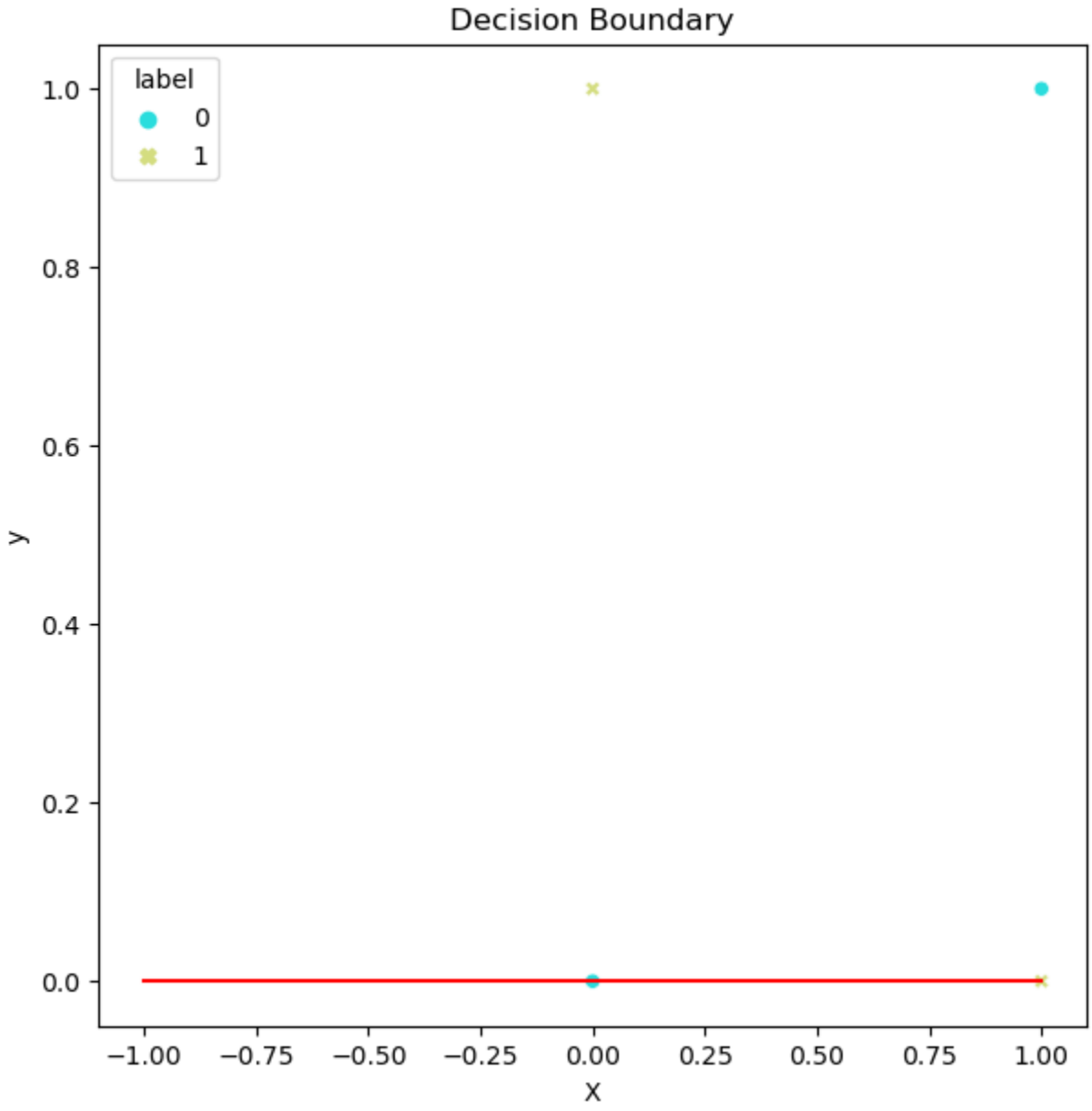



Decision boundary exists in the case of learnable bias. The line passes through the point because of the implementation done in the activation function.

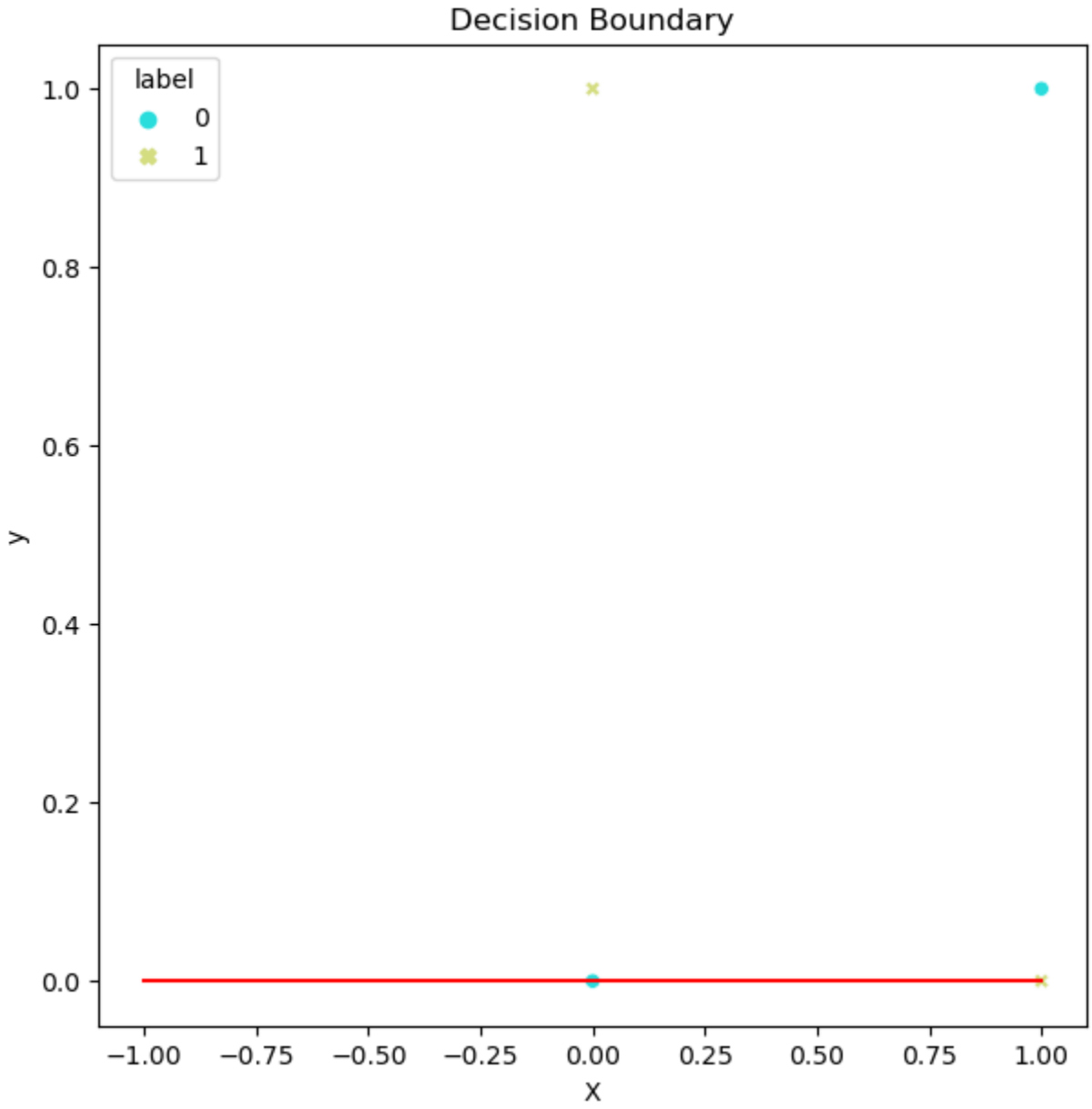
Decision boundary does not exist in the case of fixed bias. It passes through the point because the point lies on the origin, and the line also passes through it.

**XOR Dataset**

Learnable Bias:



Fixed Bias:




Decision boundary does not exist for the XOR dataset, as it is not linearly separable.

## Part 6

Given a point (x, y), we use ‘x’ and ‘y’ in the plane equation and apply our activation function to the output. Depending on the implementation, we set the class to 1 if the value >= 0; else, we set it to 0 in this case.

# Section C

## Part 1

```
Percentage:  20%|██        | 1/5 [00:28<01:52, 28.22s/depth]

For Depth: 4, accuracy of Gini: 0.9856984164716208, Entropy: 0.9856984164716208 with the test set
For Depth: 4, accuracy of Gini: 0.9858561616438669, Entropy: 0.9858561616438669 with the validation set

Percentage:  40%|████      | 2/5 [01:08<01:46, 35.49s/depth]

For Depth: 8, accuracy of Gini: 0.986331553540082, Entropy: 0.9859795567583383 with the test set
For Depth: 8, accuracy of Gini: 0.9864892972651741, Entropy: 0.9861715866104387 with the validation set

Percentage:  60%|██████    | 3/5 [01:50<01:16, 38.20s/depth]

For Depth: 10, accuracy of Gini: 0.9866881217086015, Entropy: 0.9872481165886483 with the test set
For Depth: 10, accuracy of Gini: 0.9869555776505411, Entropy: 0.9873830013371275 with the validation set

Percentage:  80%|████████  | 4/5 [02:40<00:43, 43.02s/depth]

For Depth: 15, accuracy of Gini: 0.9876275416910474, Entropy: 0.988068680514921 with the test set
For Depth: 15, accuracy of Gini: 0.9879064239265837, Entropy: 0.9882195632049919 with the validation set

Percentage: 100%|██████████| 5/5 [03:32<00:00, 42.41s/depth]

For Depth: 20, accuracy of Gini: 0.9863498390871855, Entropy: 0.98615555514921 with the test set
For Depth: 20, accuracy of Gini: 0.9866767236945864, Entropy: 0.9864047268031223 with the validation set
```

We can see from the above data that we get the best accuracy by the **Entropy** tree with depth=**15**.
This could be because the trees having a max depth of < 15 might not have been able to generate a generalised model and could be underfitting. Alternatively, the tree with depth=20 might be overfitting the data and hence giving lower accuracy on the unseen data.
Testing set accuracy: **98.8068680514921%**
Validation set accuracy: **98.82195632049919%**

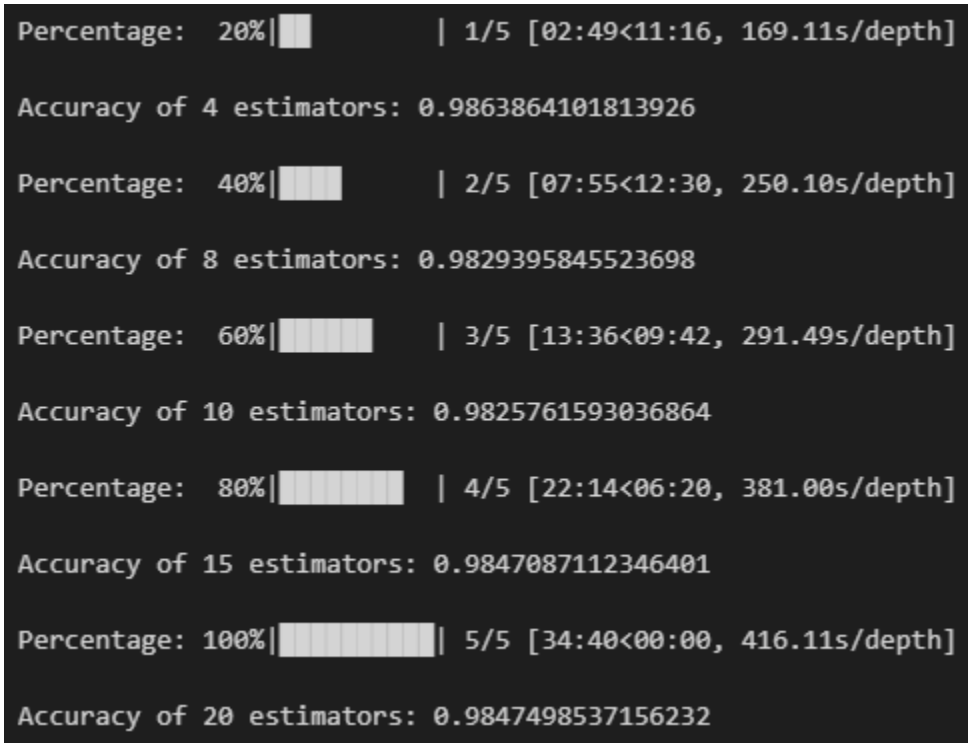
## Part 2

```
Percentage: 100%|██████████| 100/100 [08:24<00:00,  5.05s/stump]

Testing set accuracy from ensemble learning: 0.9859795567588883
Validation set accuracy from ensemble learning: 0.9861258411351668
```

Testing set accuracy: **98.59795567588883%**
Validation set accuracy: **98.61258411351668%**
We have done ensembling by using 100 “stumps” of Decision trees with max depth=3. We can see that this random forest having depth=3, outperforms all the decision trees made with Gini and Entropy, having depth=4. This shows an increase in accuracy.

## Part 3




Since the tree is already overfitted, having max depth=15. If we increase the number of estimators, it could lead to a drop in accuracy. So having the highest accuracy with 4 estimators is the best in this case. Also, it is higher than the accuracy from Random Forest tree done through ensembling.