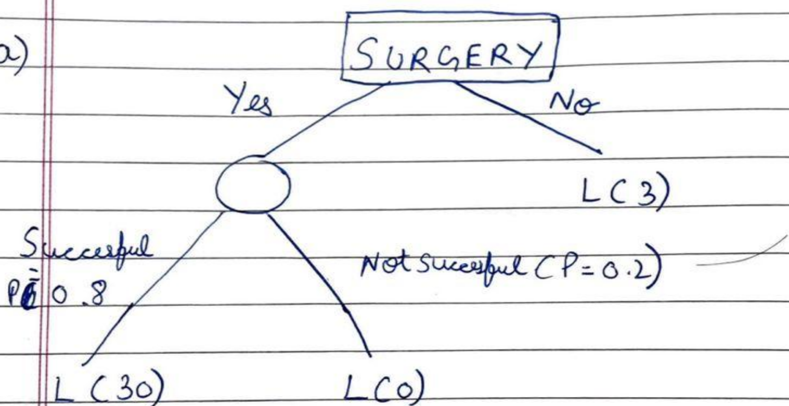
ASSIGNMENT 2 REPORT
VASU KHANNA
2020483

SECTION A

Assignment 2

Section A

a)



b) We are given that $L(0) = 0$ and $L(30) = 1$

we have $y = \frac{x}{30}$ when $x = 3$

$$y = \frac{3}{30}$$

$$y = 0.1$$

Probability of patient living for 3 days =

$L(3) = 0.1$

c) ~~P(surv~~

P(survived / test yields +ve)

$$= \frac{P(\text{test yields +ve / survive}) \times P(\text{survives})}{P(\text{+ve test / survive})\ P(\text{survive}) + P(\text{+ve / not survive}) + P(\text{not survive})}$$

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

$$= \frac{0.95 \times 0.8}{7.6 + 0.1}$$

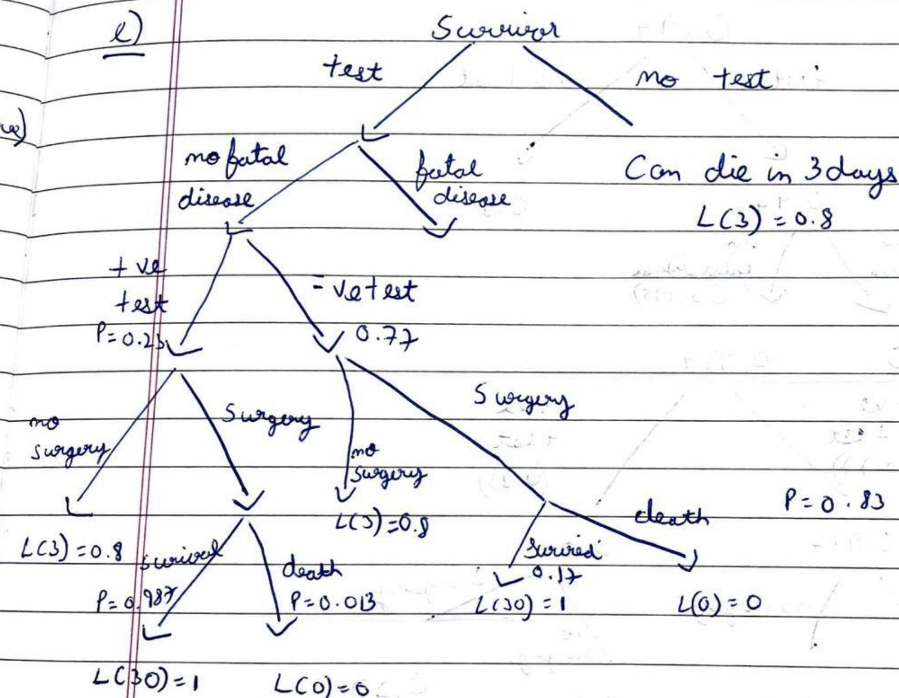
$$= \frac{76}{77} = 0.99$$

Page

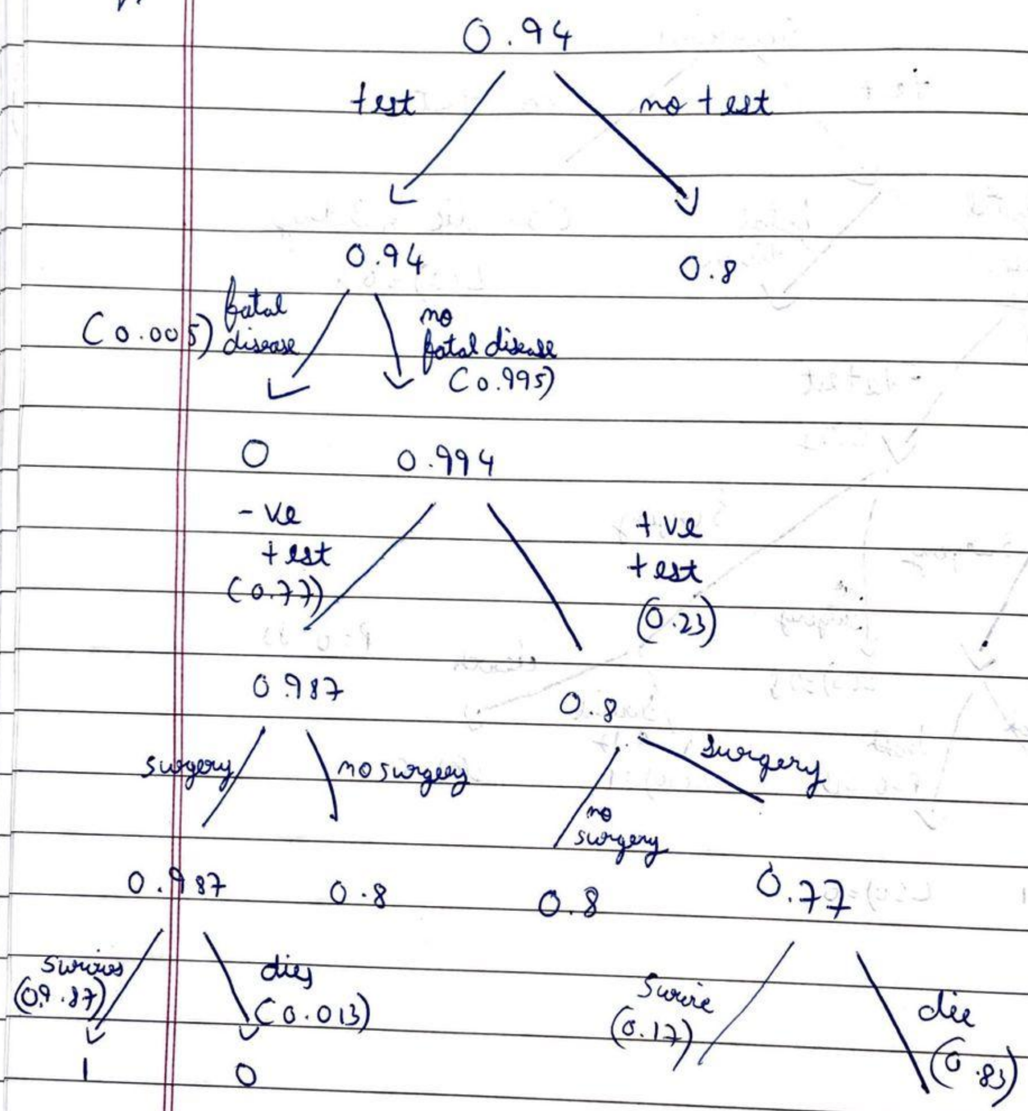
d) Yes because the surgery should be performed as the chances of survival is high (as we calculated above the chances of survival are 99%.

ould
of
tree above)

e)



f)


0.94
test
no test
0.94
0.8
(0.005) fatal disease
no fatal disease (0.995)
0
0.994
-ve test (0.77)
+ve test (0.23)
0.987
0.8
surgery
no surgery
no surgery
surgery
0.987
0.8
0.8
0.77
survives (0.987)
dies (0.013)
1
0
survive (0.17)
die (0.83)

$P(\text{catching disease while testing}) = 0.005$

$P(\text{Survive} / \text{Positive}) = 0.987$

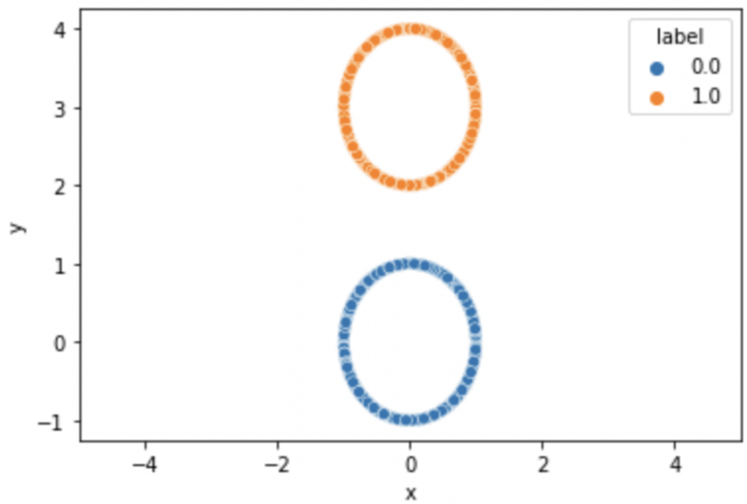
$P(\text{Survive} / +\text{ve disease II}) = 0.005 \times 0.987$
$= 0.004935$

Since P (Survive / + ve disease II) is very low we should test before surgery.

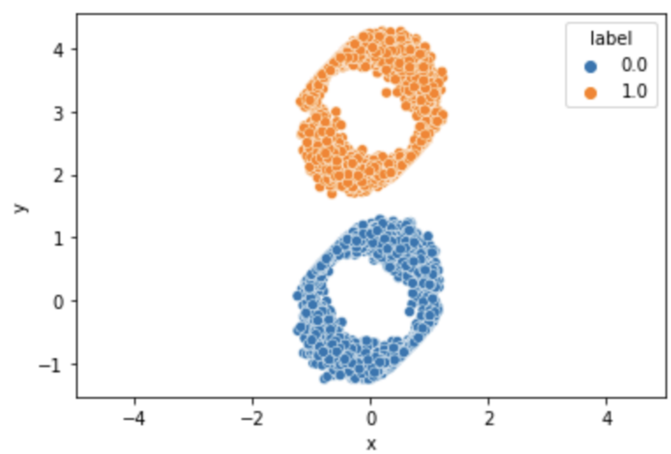
SECTION B

1) We created two data sets(with and without noise) using the circle equation and the constraints provided and then plotted the data with noise and without noise below are the results for the same:
2)

Data without noise



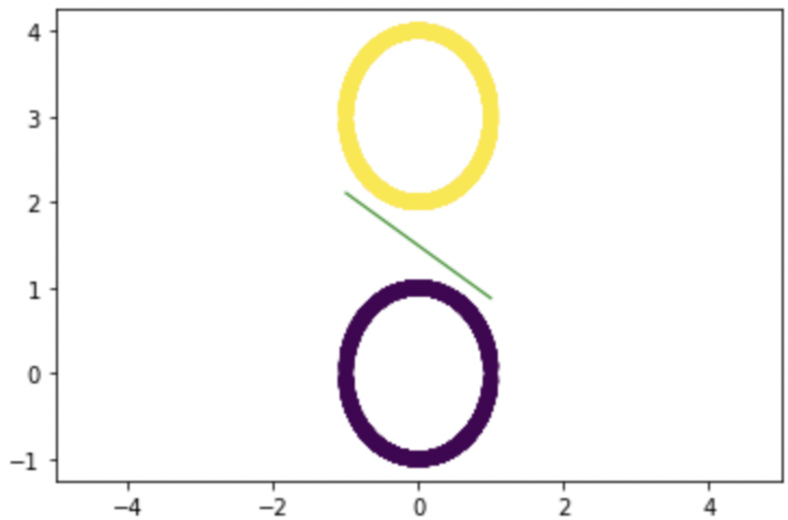
Data with noise



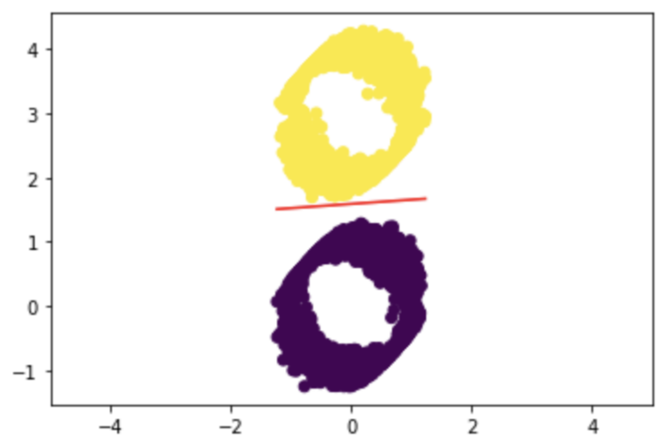
3) We trained our model using the perceptron algorithm in the utils.py file and then imported it Into the main.py file. After importing, we plotted the data(with and without noise), and along with that, we also plotted the decision boundary to check whether our model is classified between the two classes correctly or not.

Below are the resulting plots for the same:

The plot of perceptron model decision boundary in the case of data without noise:



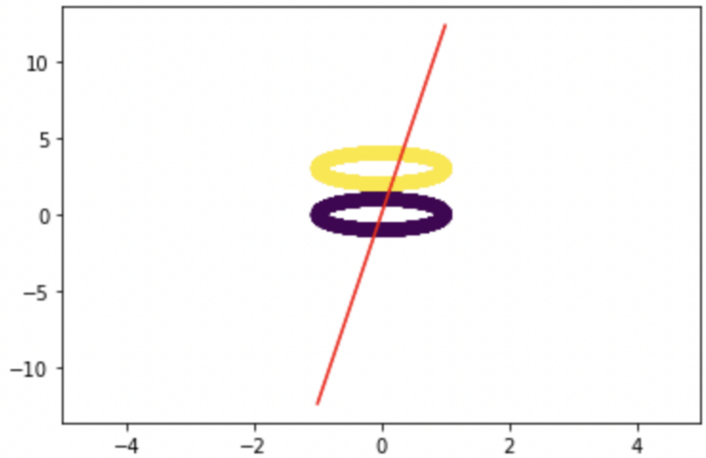
The plot of perceptron model decision boundary in the case of data with noise:



As we can see from the above plots that our perceptron model is working well, and we are getting a decision boundary that clearly separates the two classes.

4) Now we were told to keep the bias fixed, below is the graph for keeping the bias fixed in case of perceptron model decision boundary with data without noise:

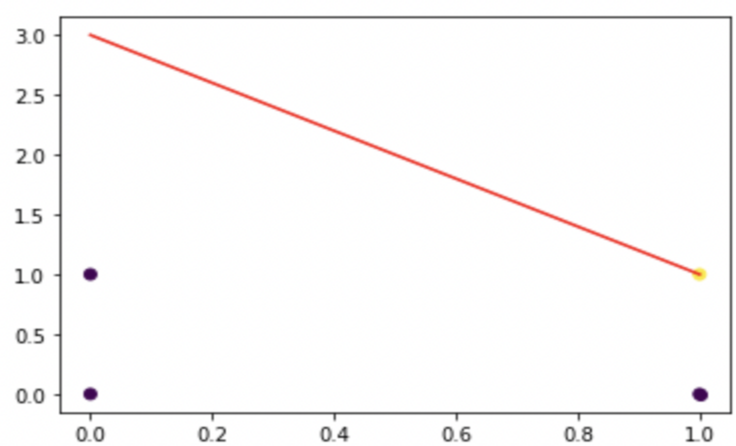
The plot for the decision boundary when we keep the bias fixed at 0



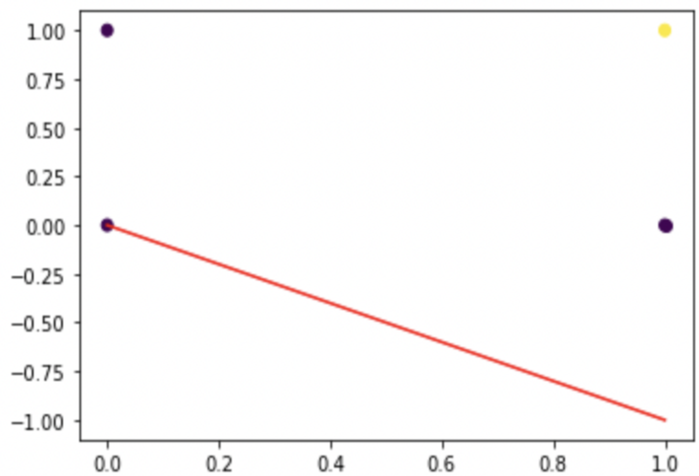
The fixed bias model produced an improper decision boundary that passed through the classes. This occurred because we could not adjust the bias even when the classifier provided an incorrect response. During training, we attempt to draw a line that separates the two classes. But, in this situation, because we can only adjust the value of the weights (theta) and not the bias, this forcibly resulted in a line that passes through the origin. As a consequence, we cannot get a line that divides the two classes, which is why we get an invalid decision boundary. While on the other hand, when we had the freedom to update the bias, we got a decision boundary which separated the classes.

5) In this, we created datasets of and, or, xor gates using pandas data frame and then plotted the graphs for the decision boundaries of the same both in the case of fixed and learnable variance.
Below are the graphs for the same:

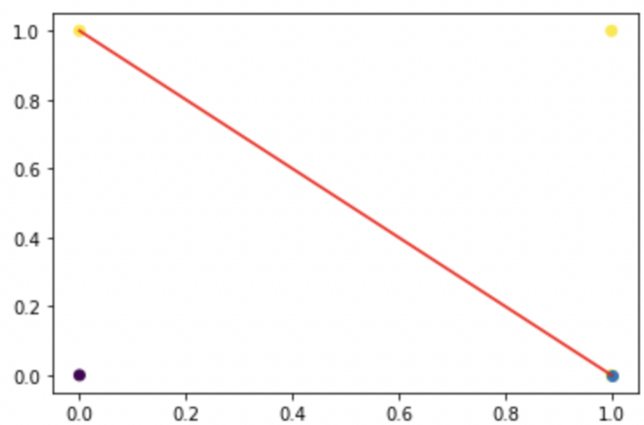
AND GATE WITH LEARNABLE BIAS



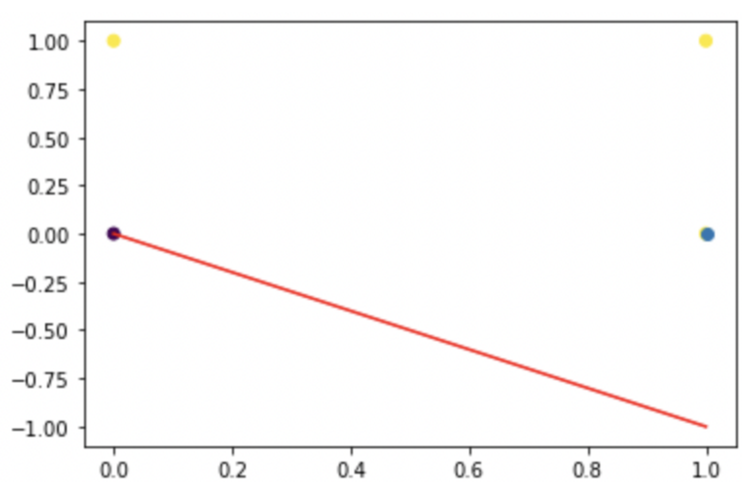
AND GATE WITH FIXED BIAS



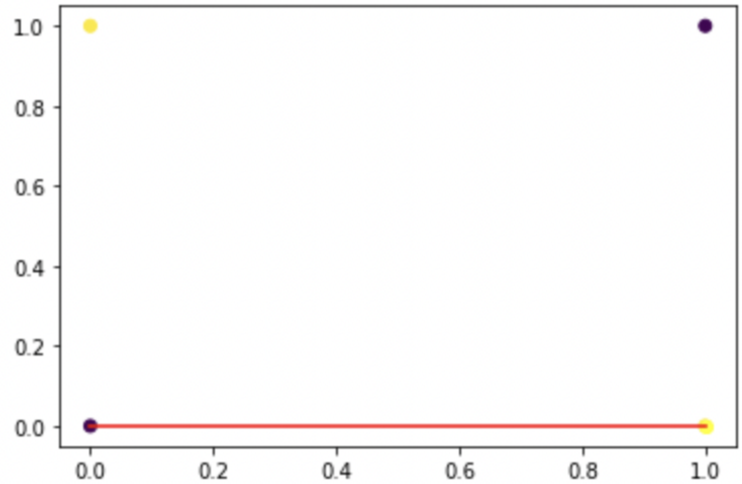
OR GATE WITH LEARNABLE BIAS



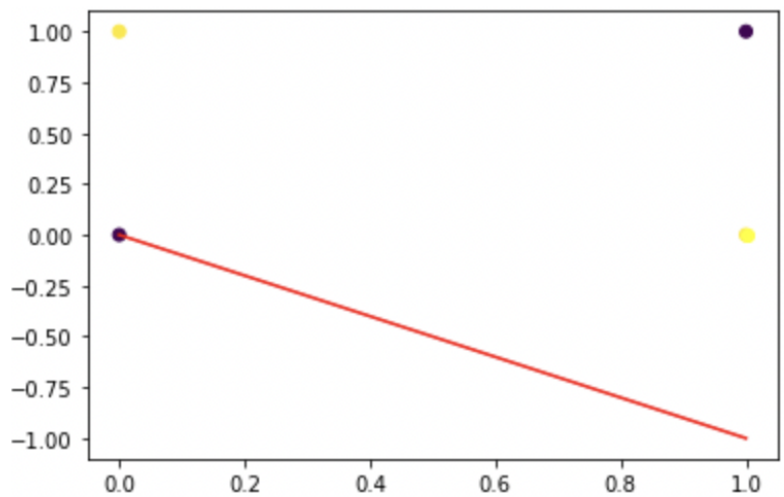
OR GATE WITH FIXED BIAS



XOR GATE WITH VARIABLE BIAS



XOR GATE WITH FIXED BIAS



In the above cases, we observe that when we take the bias as variable, we get a correct decision boundary, while when we fix the bias, we get an incorrect decision boundary(as explained above)

6)

For this, we can take a point from a class and then put it into the equation of hyperplane and then observe the output(whether we are getting a positive output or a negative output) and on the bases of that, we can decide which point belongs to which class. Also, we should consider a strong assumption while performing this practice that the data should be strictly linearly separable.

SECTION C

a) I have used the mean method for handling null values. I have taken the mean of all the columns and then replaced the null values with the mean of the columns below is the output for the same:

```
Replace null value from column year  with  2014
Replace null value from column day  with  181.457211016434
Replace null value from column length  with  45.00859293920486
Replace null value from column weight  with  0.5455192341637917
Replace null value from column count  with  721.6446428957139
Replace null value from column looped  with  238.50669884461772
Replace null value from column neighbors  with  2.206516137946451
Replace null value from column income  with  4464889007.1859665
```

Also, I have removed the address column from the data frame as it had string values while other columns had numerical values.

Now we had to train a decision tree using entropy and the Gini index method while changing the max depth[4,8,10,15,20] and check which method is performing better below are the results:

```
For depth =  4  ****Gini results****
Accuracy score :  98.5698449160581

For depth =  4  ****Entropy results****
Accuracy score :  98.5698449160581

For depth =  8  ****Gini results****
Accuracy score :  98.56367355801649

For depth =  8  ****Entropy results****
Accuracy score :  98.56344498920012

For depth =  10  ****Gini results****
Accuracy score :  98.57281631067073

For depth =  10  ****Entropy results****
Accuracy score :  98.57190203540532

For depth =  15  ****Gini results****
Accuracy score :  98.5730448794871

For depth =  15  ****Entropy results****
Accuracy score :  98.5609307322202

For depth =  20  ****Gini results****
Accuracy score :  98.57007348487446

For depth =  20  ****Entropy results****
Accuracy score :  98.54401663980983
```

We can see that for the max depth 15, we get the best results also we can observe that gini method performs better than the entropy method as it wins the majority of the time.

c) in this part, we had to use the AdaBoost technique and below were the results for the same:

```
 4 th estimator
Score  94.8789156695352


 8 th estimator
Score  95.3369675775134


 10 th estimator
Score  94.71617467228945


 15 th estimator
Score  96.59683889326979


 20 th estimator
Score  97.50037142432659
```

We can see that in this case, we get the highest accuracy for the 20th estimator. Also, AdaBoost and the random forest give similar accuracies, but AdaBoost is a slightly better method because, in AdaBoost, we train the data to correct each other's errors.