1. if $\vec{A}$ & $\vec{B}$ are the vectors then the cosine similarity between these vectors will be

$$D = \frac{\vec{A} \cdot \vec{B}}{|A| \cdot |B|}$$

$$= \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Demonstrating that the value can be

-1 : $[a\ b\ c\ d]^T$ & $[-a\ -b\ -c\ -d]^T$.

$$\Rightarrow D = \frac{-a^2 - b^2 - c^2 - d^2}{\sqrt{a^2+b^2+c^2+d^2}\sqrt{a^2+b^2+c^2+d^2}}$$

$$= \frac{-1(a^2+b^2+c^2+d^2)}{(a^2+b^2+c^2+d^2)}$$

$$= -1$$

0 : $[a\ -a\ b\ -b]^T$ & $[a\ a\ b\ b]$.

$$\Rightarrow D = \frac{a^2 - a^2 + b^2 - b^2}{\sqrt{2a^2+2b^2}\sqrt{2a^2+2b^2}} = 0$$

$$I = [a\ b\ c\ d][a\ b\ c\ d]^T$$

$$D = \frac{a^2+b^2+c^2+d^2}{\sqrt{a^2+b^2+c^2+d^2}\cdot\sqrt{a^2+b^2+c^2+d^2}}$$

$$= \frac{a^2+b^2+c^2+d^2}{a^2+b^2+c^2+d^2} = 1$$

2. a) if all the sample are multiplied by the same scaler then

$x_i \rightarrow \alpha x_i$

where $\alpha \in R^+$

So if $w^T x_i \geq 0$

then $\alpha w^T x_i \geq 0$

or if $w^T x_i < 0$

then $\alpha w^T x_i < 0$.

hence, the accuracy of the model won't change.

b) Similar to above example it does not depend if all the sample are multiplied to same scaler # or to different scaler until the scaler is changing the sigh of $w^T x$ the accuracy will not change.

and since $\alpha \in R^+$ then the accuracy will be same.

c) When the sample are multiplied by orthogonal matrix A, then the angle between do not change & also the length of the vectors do not change but angle of whole all the points change with x-axis (or fixed axis) but not relative to each other, ~~hence there will be no change in accuracy of the model.~~

hence the points may cross-over to the other side of line resulting in change of ~~accuracy~~ accuracy

d) if the matrix is rank deficient it means that the columns are dependent this means few are independent also and this will make the accuracies change.

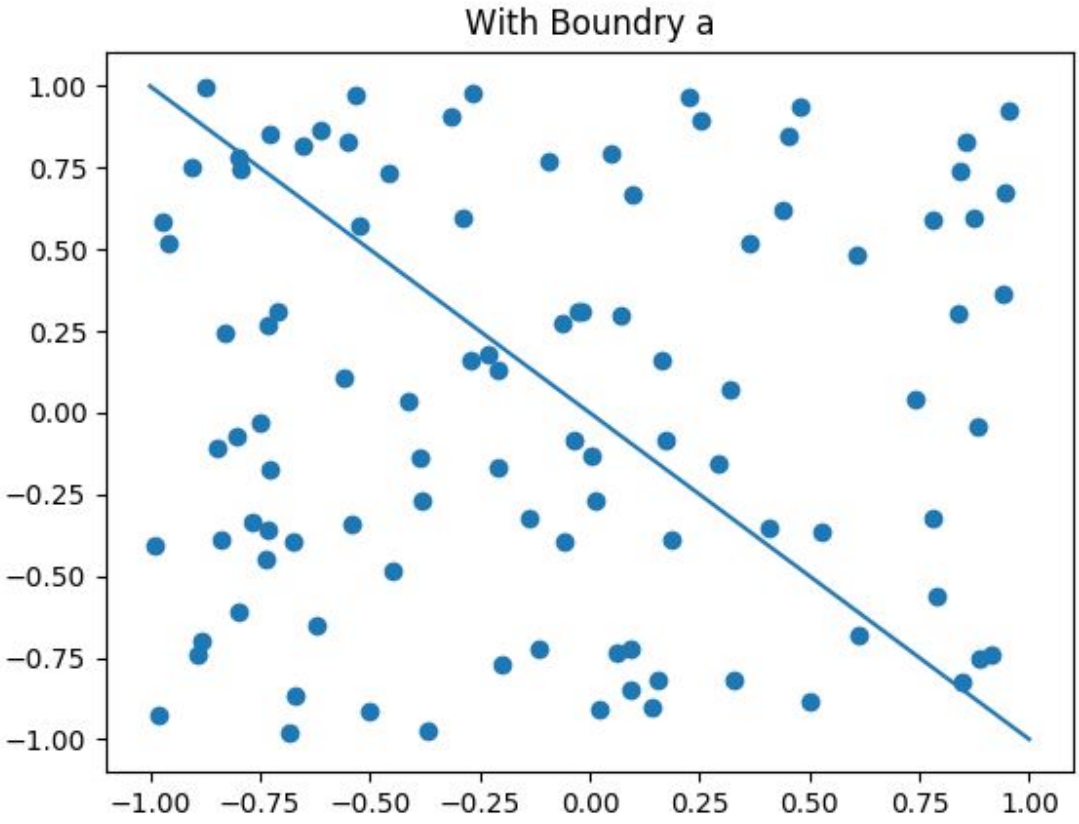
```
# this is the Code for 3rd part of the Assignment

import random
import matplotlib.pyplot as plt
import numpy as np

def RC_Multiply(a,b):
	sum = int("0",10)
	for i in range(3):
		sum += a[i]*b[i]
	return sum;

Data = []
plt_dt = []
a = 0
for i in range(100):
	r_fra1 = random.randint(-1000,1000)/1000;
	r_fra2 = random.randint(-1000,1000)/1000;
	dat_plt = [r_fra1, r_fra2]
	plt_dt.append(dat_plt)
	a = [r_fra1, r_fra2,1]
	if(i < 50):
		clas = "A"
	else:
		clas = "B"
	Data.append([a,clas])

w1 = [1,1,0]  #plot this line
w2 = [-1,-1,0]
w3 = [0,0.5,0]
w4 = [1, -1,5] # plot this line
w5 = [1,1,0.3] # plot this line

acc_w1 = 0;
acc_w2 = 0;
acc_w3 = 0;
acc_w4 = 0;
acc_w5 = 0;
for i in range(100):
	if(RC_Multiply(w1,Data[i][0]) > 0 and Data[i][1]=="A"):
		acc_w1 += 1;
	elif(RC_Multiply(w1,Data[i][0]) <= 0 and Data[i][1]=="B"):
		acc_w1 += 1;
```

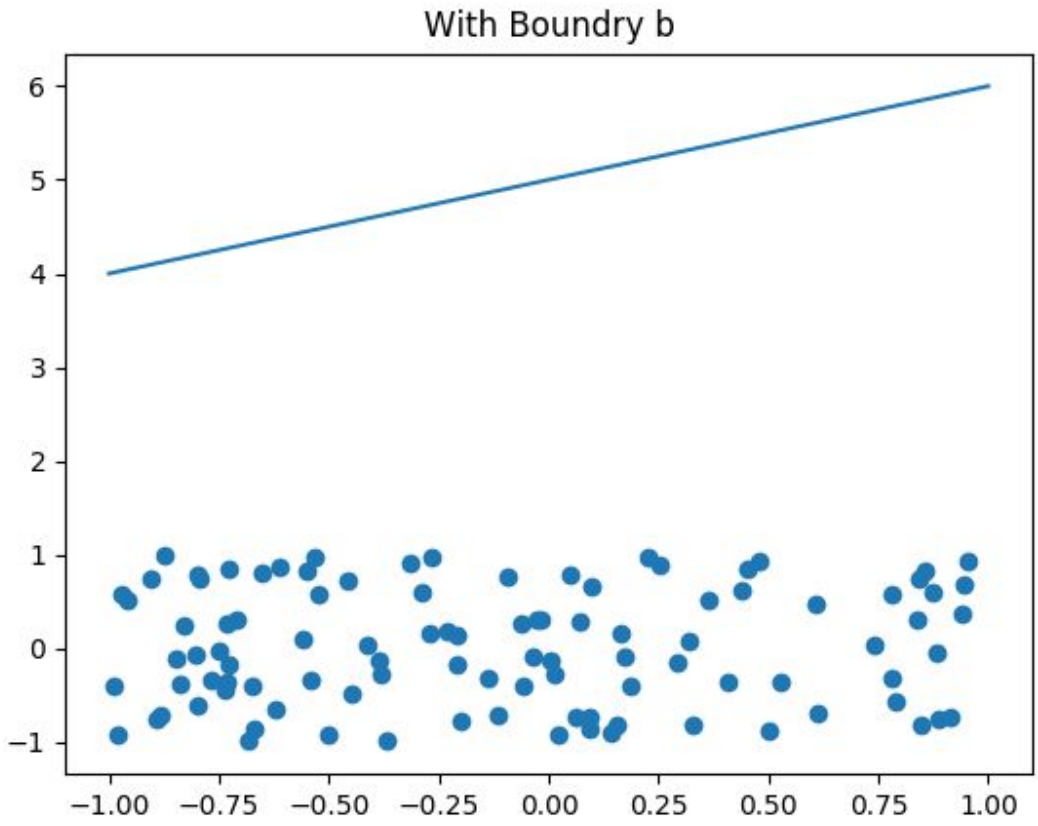
```
        if(RC_Multiply(w1,Data[i][0]) > 0 and Data[i][1]=="A"):
                acc_w2 += 1;
        elif(RC_Multiply(w2,Data[i][0]) <= 0 and Data[i][1]=="B"):
                acc_w2 += 1;
        if(RC_Multiply(w3,Data[i][0]) > 0 and Data[i][1]=="A"):
                acc_w3 += 1;
        elif(RC_Multiply(w3,Data[i][0]) <= 0 and Data[i][1]=="B"):
                acc_w3 += 1;
        if(RC_Multiply(w4,Data[i][0]) > 0 and Data[i][1]=="A"):
                acc_w4 += 1;
        elif(RC_Multiply(w4,Data[i][0]) <= 0 and Data[i][1]=="B"):
                acc_w4 += 1;
        if(RC_Multiply(w5,Data[i][0]) > 0 and Data[i][1]=="A"):
                acc_w5 += 1;
        elif(RC_Multiply(w5,Data[i][0]) <= 0 and Data[i][1]=="B"):
                acc_w5 += 1;

print("PART (i)")
print("Accuracy of a : "+str(acc_w1)+"%")
print("Accuracy of b : "+str(acc_w2)+"%")
print("Accuracy of c : "+str(acc_w3)+"%")
print("Accuracy of d : "+str(acc_w4)+"%")
print("Accuracy of e : "+str(acc_w5)+"%")

print("PART (ii)")
fig, ax = plt.subplots()
data_plot = np.array(plt_dt);
x,y = data_plot.T
ax.scatter(x,y)
x = np.array(range(-1,2))
y = (w1[0]/-w1[1])*x + (w1[2]/-w1[1])
la = ax.plot(x,y,label='line for a')
ax.set_title('With Boundry a')

fig,bx = plt.subplots()
data_plot = np.array(plt_dt);
x,y = data_plot.T
bx.scatter(x,y)
x = np.array(range(-1,2))
y = (w4[0]/-w4[1])*x + (w4[2]/-w4[1])
bx.plot(x,y,label='line for d')
bx.set_title('With Boundry b')
```

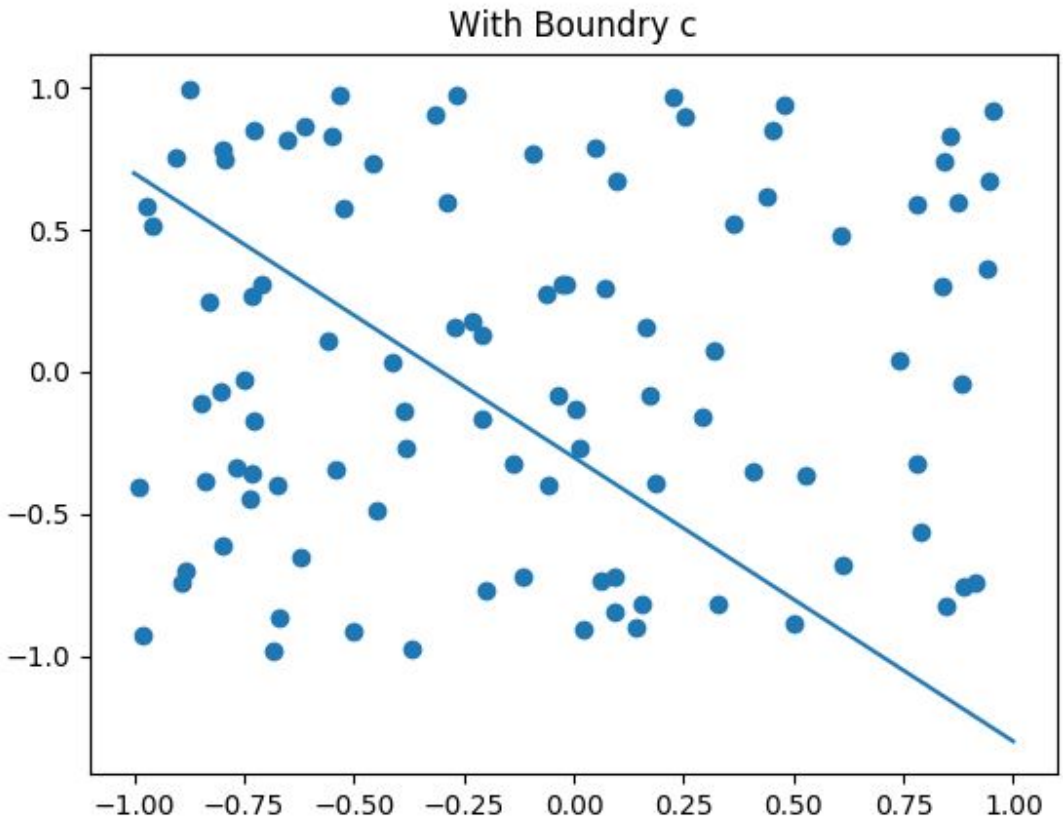
```
fig,cx = plt.subplots()
data_plot = np.array(plt_dt);
x,y = data_plot.T
cx.scatter(x,y)
x = np.array(range(-1,2))
y = (w5[0]/-w5[1])*x + (w5[2]/-w5[1])
le = cx.plot(x,y,label='line for e')
cx.set_title('With Boundry c')
plt.show()
```

PART (i)
Accuracy of a : 55%
Accuracy of b : 53%
Accuracy of c : 54%
Accuracy of d : 50%
Accuracy of e : 51%
PART (ii)

These almost show near about 50 % accuracy and since the smamples are distributed hence it is expected to show that the samples are 50 % of the time classified correclty.


With Boundry a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


With Boundry b
6
5
4
3
2
1
0
−1
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00



The accuracies are random despite