Practical No.6

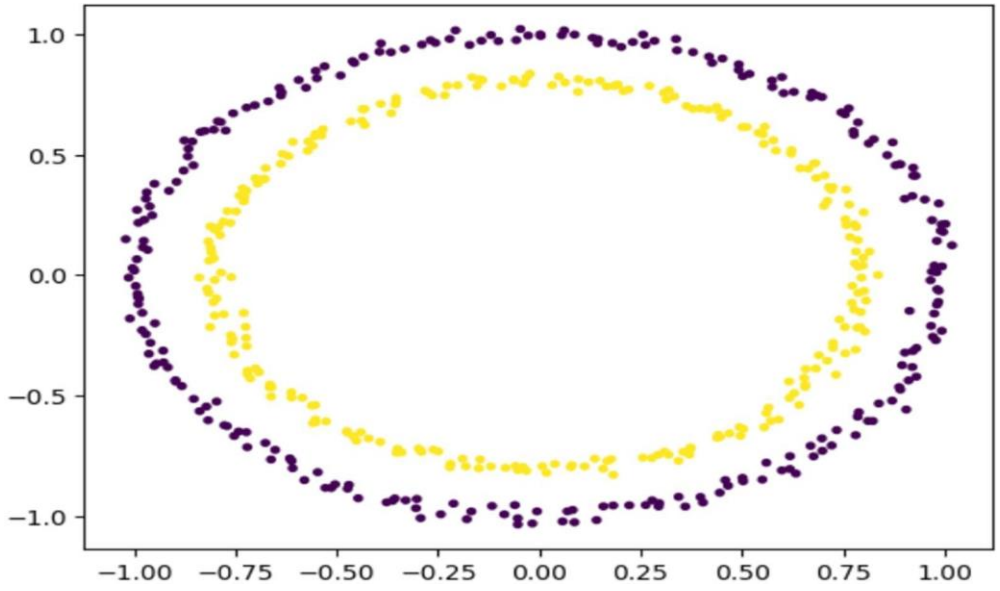
Aim: Design and implement SVM for classification with the proper data set of your choice. Comment on Design and Implementation for Linearly non separable Dataset.

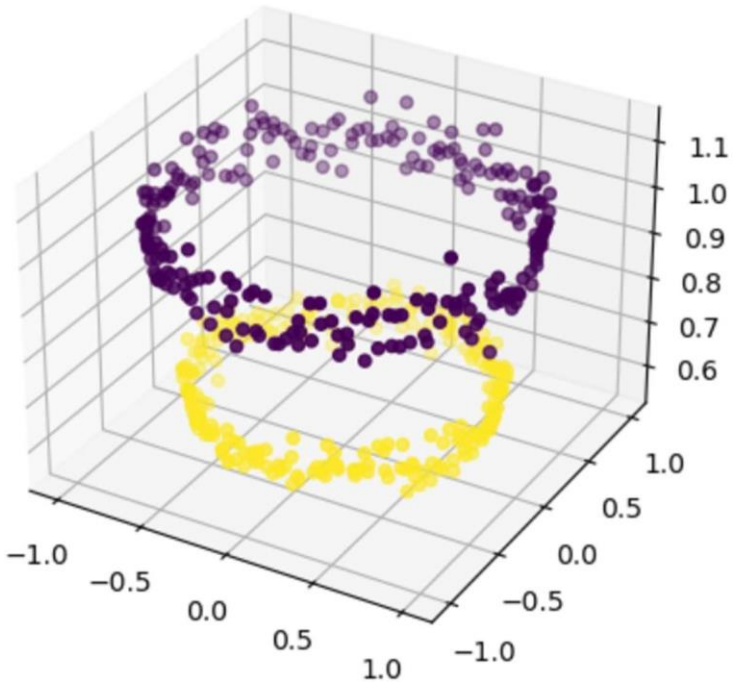
Code:

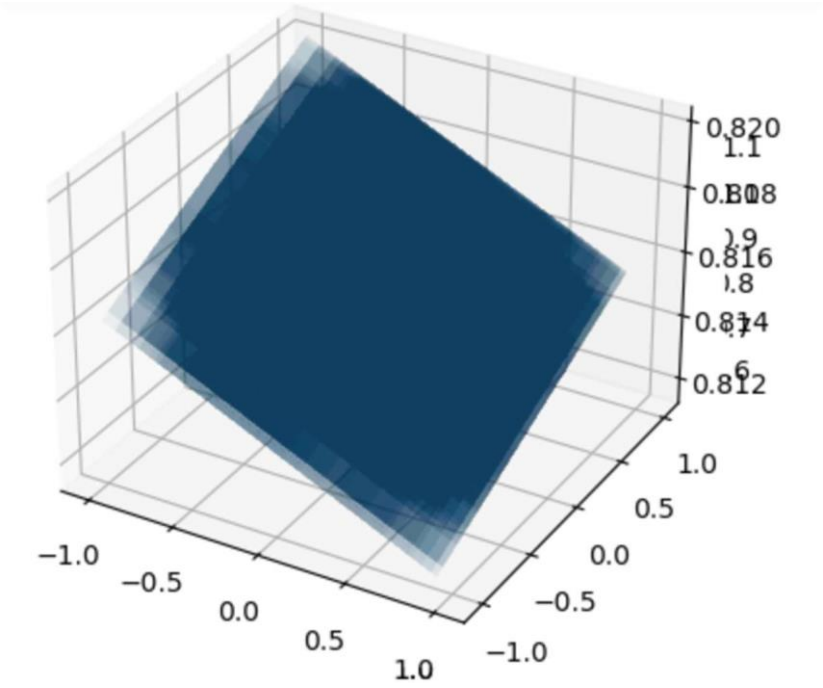
```
import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import make_circles

from mpl_toolkits.mplot3d import Axes3D

X, Y = make_circles(n_samples = 500, noise = 0.02)

plt.scatter(X[:, 0], X[:, 1], c = Y, marker = '.')

plt.show()

X1 = X[:, 0].reshape((-1, 1))

X2 = X[:, 1].reshape((-1, 1))

X3 = (X1**2 + X2**2)

X = np.hstack((X, X3))

fig = plt.figure()

axes = fig.add_subplot(111, projection = '3d')

axes.scatter(X1, X2, X1**2 + X2**2, c = Y, depthshade =
True)

plt.show()

from sklearn import svm

svc = svm.SVC(kernel = 'linear')

svc.fit(X, Y)

w = svc.coef_

b = svc.intercept_

x1 = X[:, 0].reshape((-1, 1))

x2 = X[:, 1].reshape((-1, 1))
```

```
x1, x2 = np.meshgrid(x1, x2)

x3 = -(w[0][0]*x1 + w[0][1]*x2 + b) / w[0][2]

fig = plt.figure()

axes2 = fig.add_subplot(111, projection = '3d')

axes2.scatter(X1, X2, X1**2 + X2**2, c = Y, depthshade =
True)

axes1 = fig.add_subplot(111, projection = '3d')

axes1.plot_surface(x1, x2, x3, alpha = 0.01)
plt.show()
```

Output:




1.1
1.0
0.9
0.8
0.7
0.6
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


0.820
1.1
0.816
1.8
0.812
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

Practical No.7

Aim: Implement a basic not gate using perceptron.

Code:

```
import numpy as np

def unitStep(v):
    if v >= 0:
        return 1
    else:
     return 0

def perceptronModel(x, w, b):
    v = np.dot(w, x) + b
    y = unitStep(v)
    return y

def NOT_logicFunction(x):
    w = -1
    b = 0.5
 return perceptronModel(x, w, b)

test1 = np.array(1)

test2 = np.array(0)

print("NOT({}) = {}".format(1, NOT_logicFunction(test1)))

print("NOT({}) = {}".format(0, NOT_logicFunction(test2)))
```

Output:

```
NOT(1) = 0
NOT(0) = 1
```