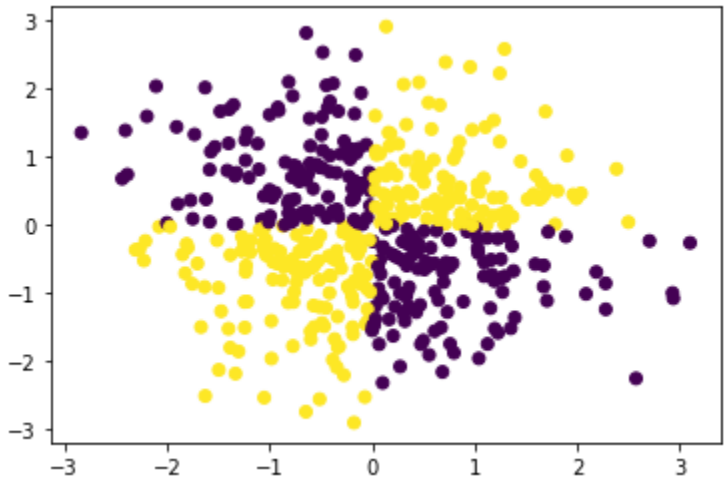
NISARG UPADHYAYA - 19CS30031

Assignment 2 Q9

Link to colab:
https://colab.research.google.com/drive/1ZCiMFgl49siSRzz1ieVbFSwKkpBCRiuC?usp=sharing

Distribution of random points generated:



Part A.

Coefficients generated using least squares

```
[-1.18510686e-02 -4.15105479e-04  1.23373406e-02  6.63671035e-01
  7.44812004e-02 -1.50746503e-02]
```

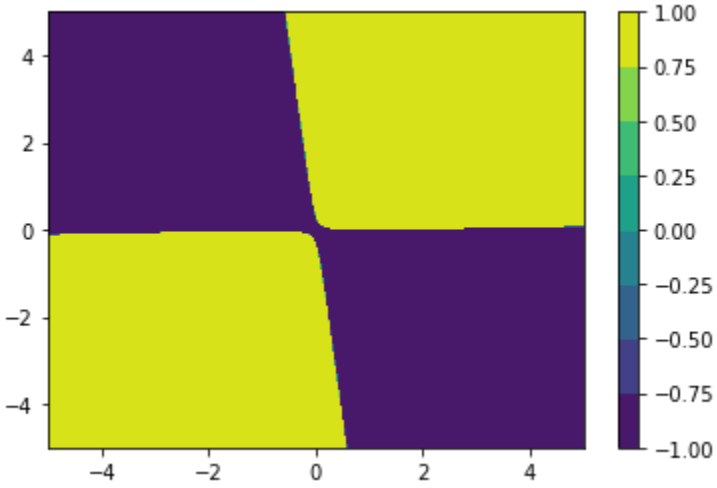
Confusion matrix

```
[[228  18]
 [ 13 241]]

Error rate: 6.2%
```

Part B.

Contour



Part C.

If the polynomial is g(x1, x2) = x1 * x2, then the output is sign(x1 * x2). This will obviously yield zero error and classify points correctly because that was the initial assumption. It can be seen that here the coefficient of the term x1*x2 is 1. From the parameters we estimated, it can be seen that the highest weighted coefficient is that of x1 * x2 with a weight of `6.63671035e-01` which is higher than all others. Hence, it is a very good estimator for our classification purpose.

CODE

```
X = np.random.normal(size=(500,2))
Y = np.sign(np.prod(X, 1))

plt.scatter(X[:,0], X[:,1], c=Y, cmap='viridis')
plt.show()
```

```
X_final = np.ndarray((500, 6), dtype = np.float64)
for i in range(500):
   X_final[i][0] = 1
   X_final[i][1] = X[i][0]
   X_final[i][2] = X[i][1]
   X_final[i][3] = X[i][0]*X[i][1]
   X_final[i][4] = X[i][0]**2
   X_final[i][5] = X[i][1]**2

coeff = np.linalg.inv(X_final.T @ X_final) @ X_final.T @ Y
print(coeff)

pred = np.sign(X_final @ coeff)

cm = confusion_matrix(Y, pred)
print(cm)

print(f"Error rate: {(cm[0][1]+cm[1][0])/5}%")
```

```
x = np.linspace(-5, 5, 500)
y = np.linspace(-5, 5, 500)

z = np.ndarray((500, 500), dtype = np.float64)

for x_ind, i in enumerate(x):
   for y_ind, j in enumerate(y):
       z[x_ind][y_ind] = np.sign(coeff[0] + (coeff[1]*i) + (coeff[2]*j) +
(coeff[3]*i*j) + (coeff[4]*(i**2)) + (coeff[5]*(j**2)))

fig, ax = plt.subplots()
cs = ax.contourf(x, y, z)
fig.colorbar(cs)
plt.show()

print(Y.shape)
```