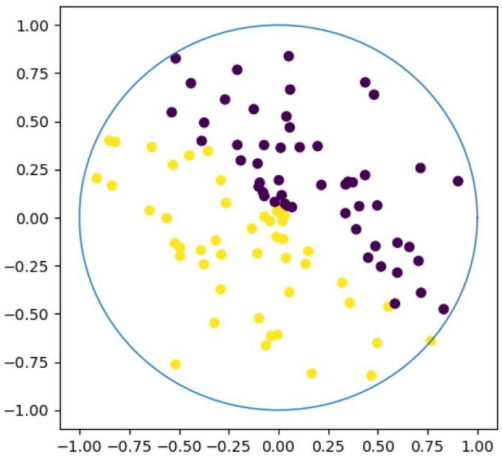
Code:

```
import numpy as np
import numpy.random
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
import math



def generate_data_set_with_labels(n):
    '''
    generates random data set of size n by sampling points uniformly
    at random within the unit sphere in the Euclidean plane and labels
    the points according to a randomly chosen linear decision boundary
    '''
    # generate two points which will represent linear decision boundary

    data_set = []
    labels = []

    r = np.random.uniform(0,1.0, 2)
    theta = np.random.uniform(0,2 * math.pi, 2)
    x1 = r[0] * math.cos(theta[0])
    x2 = r[1] * math.cos(theta[1])
    y1 = r[0] * math.sin(theta[0])
    y2 = r[1] * math.sin(theta[1])
    m = (y1-y2)/(x1-x2)
```

```
    for i in range (n):
        r = np.random.uniform(0,1.0)
        theta = np.random.uniform(0,2 * math.pi)
        x = r * math.cos(theta)
        y = r * math.sin(theta)
        data_set.append(np.array([x,y]))
        if x - x1 > m*(y - y1):
            labels.append(-1)
        else:
            labels.append(1)

    np_data_set = np.array(data_set)
    np_labels = np.array(labels)
    return np_data_set, np_labels



def plot_data(data, labels):
    plt.figure()
    t = np.linspace(0,math.pi*2,100)
    plt.plot(np.cos(t), np.sin(t), linewidth=1)
    plt.scatter(data[:, 0], data[:, 1], c=labels)
    plt.show()



def plot_mistakes(mistakes):
    plt.figure()
    lst = []
    for i in range(100):
```

```
        lst.append(i)
    plt.scatter(lst, mistakes)


    plt.show()



def perceptron_algorithm(data, labels):
    w = np.zeros((100, 2))
    t = 0
    mistakes = []
    mistakes_counter = 0
    for x_t in data:

        p_t = np.sign(np.dot(w[t],x_t))
        y_t = labels[t]
        if p_t != y_t:
            mistakes_counter += 1
        mistakes.append(mistakes_counter)


        if t+1<100:
            if p_t*y_t <= 0:
                w[t+1] = w[t] + y_t*x_t
            else:
                w[t+1] = w[t]
        t += 1
    return mistakes
```

```
data, labels = generate_data_set_with_labels(n=100)
print(data.size, labels.size, np.sign(-2))
plot_data(data, labels)
mistakes = perceptron_algorithm(data, labels)
plot_mistakes(mistakes)
```



Mistakes: