Name: Sayantan Banerjee

Roll No: 2018IMT-093

Course: Machine Learning Lab

Course Code: ITIT - 4107

Deadline : 18 September 2021

Importing necessary library for completing the experiments. Libraries used are NumPy, scikit-learn, matplotlib, seaborn

```
import numpy as np
import scipy as sp
import sklearn as sk
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import make_moons, make_blobs, make_circles
```

1a

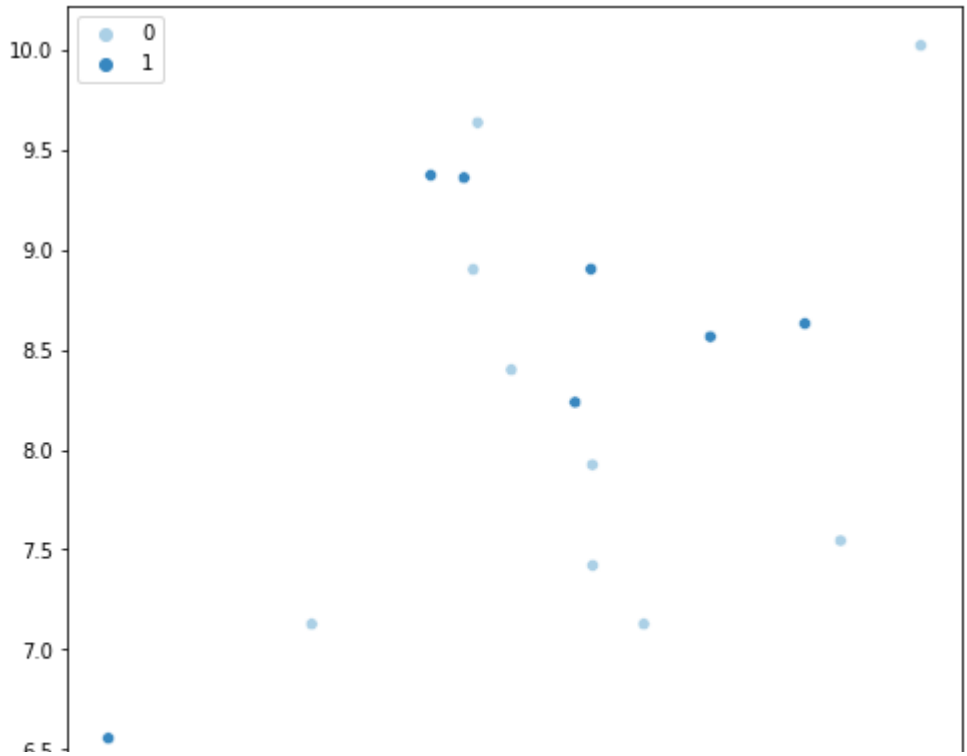
Aim

Create a random dataset with two input features and one ouptut feature (class labels). The two input features shall be random variables sampled from a gaussian distribution with mean 8 , standard deviation of 1.5 as well as the output feature may be sampled from a binomial distribution with probability of 1 as 0.6. Create 20 instances of the sampled data and plot the same.

Procedure

I used random distribution from numpy library to generate the sample from distributio. After generating the sample from the corresponding I plotted them along with their class output.

```
fig = plt.figure(figsize=(8,8))
X = np.random.normal(8, 1.5, ((20, 2)))
Y = np.random.binomial(1, 0.6, 20)
sns.scatterplot(x = X[:, 0], y = X[:, 1], hue = Y, palette='Blues')
plt.show()
```



1b

Aim

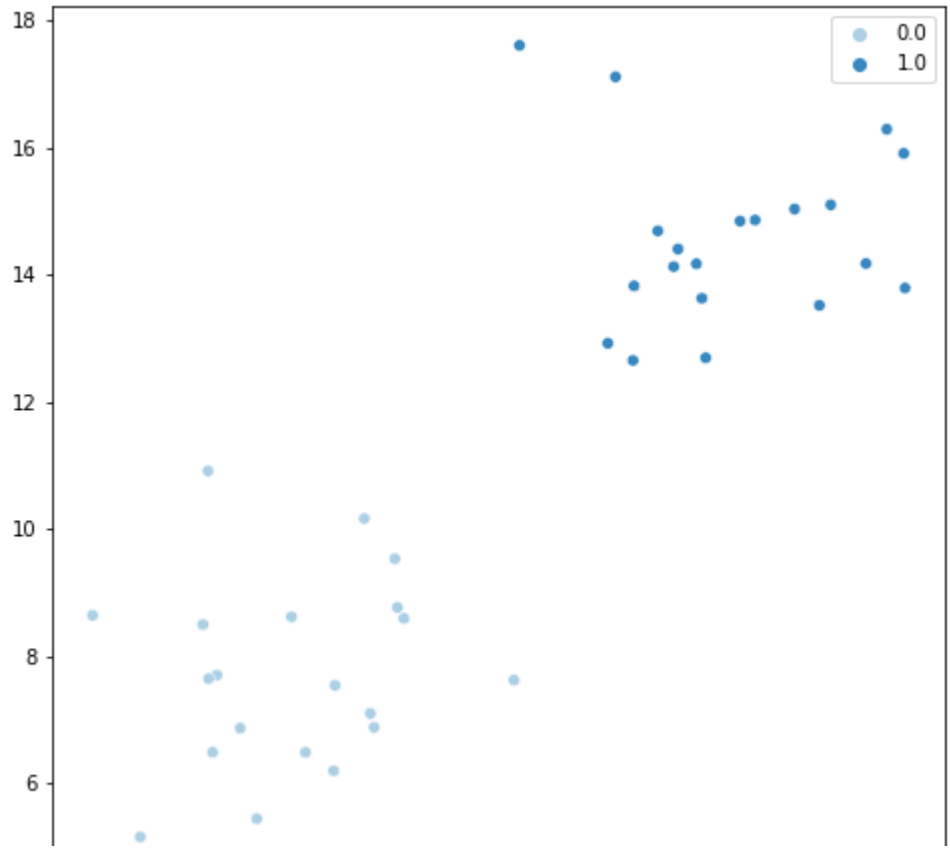
Create a random dataset with two input features and one ouptut feature. Create 20 instances of data by sampling the two input features from a gaussian distribution with mean 8 , standard deviation of 1.5. Label these instances as 0. Generate another 20 instances of data by sampling the two input features from a gaussian distribution with mean 15 and standard deviation 1.5 and label these instances as 1. Plot these.

Procedure

Initially I generated the random number which were normally distributed for output class 0. And the similarly for the output class 1. After that I combined the both randomly generated sequence and come up with final sequence termed as **X1** To visualise the dataset, I plotted the data along the axis (as it is 2-dimensional in this case) and used different colors to depict data point of different class

```
X1_zero = np.random.normal(8, 1.5, (20, 2))
Y1_zeros = np.zeros(20)
X1_ones = np.random.normal(15, 1.5, (20, 2))
Y1_ones = np.ones(20)
X1 = np.concatenate((X1_zero, X1_ones), axis=0)
Y1 = np.concatenate((Y1_zeros, Y1_ones), axis=0)

fig = plt.figure(figsize=(8,8))
sns.scatterplot(x = X1[:, 0], y = X1[:, 1], hue=Y1, palette="Blues")
plt.show()
```



1c

Aim

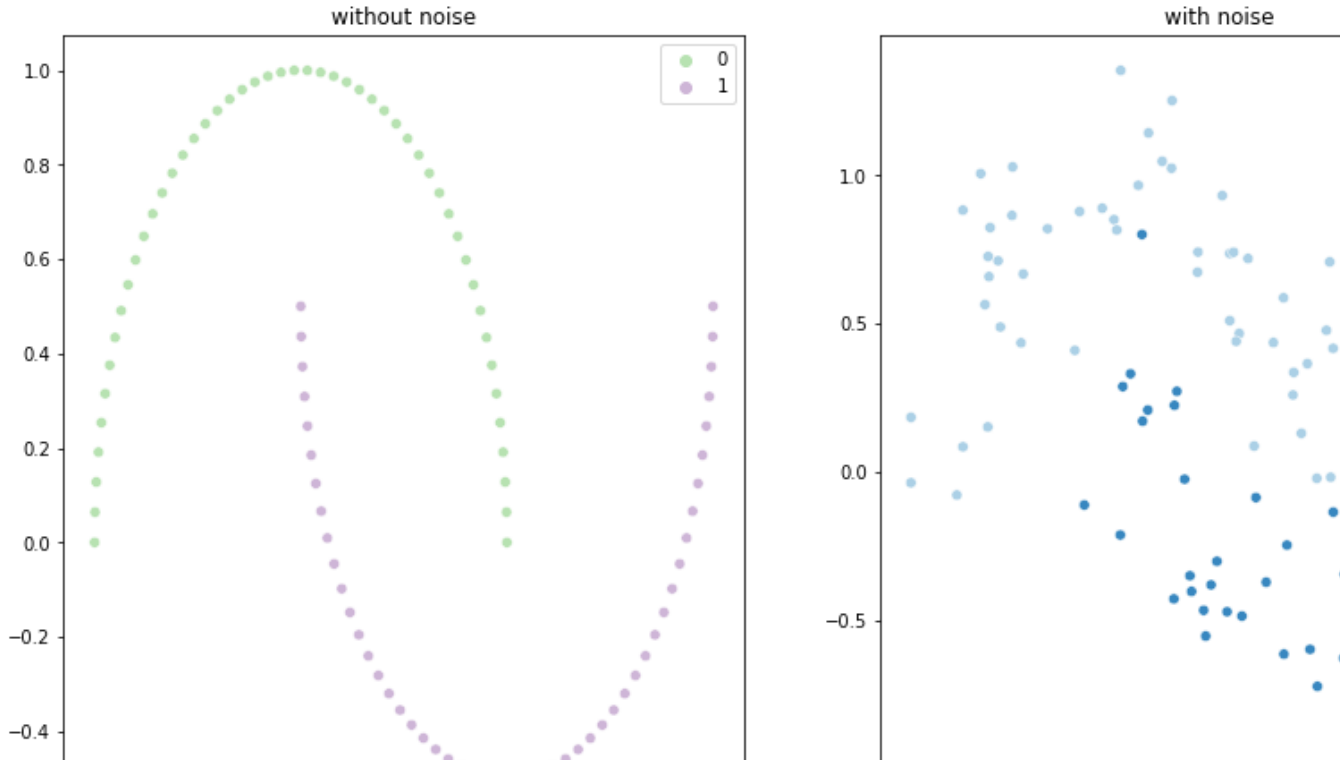
Use the make_moon, make_circles, make_blobs functions to create from sklearn library to generate datasets and plot them.

Procedure

For completition of this part I created two figure for each methods. For each functions I generated two figures one without noise to show the exact nature of dataset generated by the function. And other figure with noise in it.

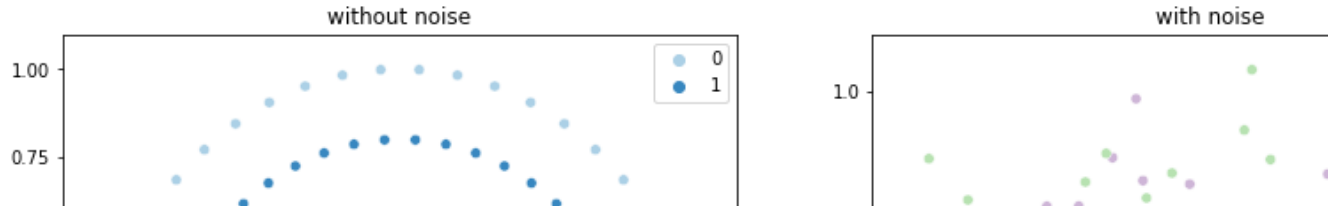
```
X2, Y2 = make_moons(random_state=1, n_samples=100)
X2_noise, Y2_noise = make_moons(noise=0.2, random_state=1, n_samples=100)

fig, ax = plt.subplots(1,2, figsize=(15,8))
sns.scatterplot(ax=ax[0], x=X2[:, 0], y = X2[:, 1], hue=Y2, palette="PRGn_r")
sns.scatterplot(ax=ax[1], x=X2_noise[:, 0], y = X2_noise[:, 1], hue=Y2_noise, palette="Blues"
ax[0].set_title("without noise")
ax[1].set_title("with noise")
plt.show()
```




```
X3, Y3 = make_circles(n_samples=100, random_state=1)
X3_noise, Y3_noise = make_circles(n_samples=100, noise=0.2, random_state=1)

fig, ax = plt.subplots(1,2, figsize=(15,8))
sns.scatterplot(ax=ax[0], x=X3[:, 0], y = X3[:, 1], hue=Y3, palette="Blues")
sns.scatterplot(ax=ax[1], x=X3_noise[:, 0], y = X3_noise[:, 1], hue=Y3_noise, palette="PRGn_r
ax[0].set_title("without noise")
ax[1].set_title("with noise")
plt.show()
```




```
X4, Y4 = make_blobs(n_samples=100, centers=4, n_features=3, random_state=1)
X4_noise, Y4_noise = make_blobs(n_samples=100, centers=4, n_features=3, cluster_std=2, random
```

```
fig, ax = plt.subplots(1,2, figsize=(15,8))
sns.scatterplot(ax=ax[0], x=X4[:, 0], y = X4[:, 1], hue=Y4)
sns.scatterplot(ax=ax[1], x=X4_noise[:, 0], y = X4_noise[:, 1], hue=Y4_noise)
ax[0].set_title("with std of cluster = 1")
ax[1].set_title("with std of cluster = 2")
plt.show()
```