OVER , UNDER SAMPLING

```
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=1000,
    n_features=5,
    n_informative=2,
    n_classes=2,
    random_state=888
)



import pandas as pd
dataset = pd.DataFrame(X)
dataset.columns = ['X11', 'X22', 'X33', 'X44', 'X55']
dataset['y'] = y
dataset.info()


dataset['y'].value_counts()



ros = RandomOverSampler()
X_ros, y_ros = ros.fit_resample(X, y)
print(Counter(y_ros))
Counter({1: 9847, 2: 9867})




rus = RandomUnderSampler()
X_rus, y_rus = rus.fit_resample(X, y)
print(Counter(y_rus))
Counter({1: 175, 0: 145})
```

ACTCATION FUNCTION

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


x = np.linspace(-15, 15)
plt.plot(x, binaryStep(x))
plt.axis('tight')
plt.title('BinaryStep activation Function ')
plt.show()



def linear(x):
    ''' y = f(x) It returns the input as it is'''
    return x


x = np.linspace(-11, 11)
plt.plot(x, linear(x))
plt.axis('tight')
plt.title('Linear Activation Function ')
plt.show()



ok def sigmoid(x):
    ''' It returns 1/(1+exp(-x)). where the values lies between zero and one '''

    return 1/(1+np.exp(-x))



x = np.linspace(-22, 12)
plt.plot(x, sigmoid(x))
plt.title('Sigmoid')
plt.show()
```

```
def RELU(x):
    ''' It returns zero if the input is less than zero otherwise it returns the given input. '''
    x1=[]
    for i in x:
        if i<0:
            x1.append(0)
        else:
            x1.append(i)

    return x1


x = np.linspace(-15, 15)
plt.plot(x, RELU(x))
plt.title('RELU')
plt.show()
```