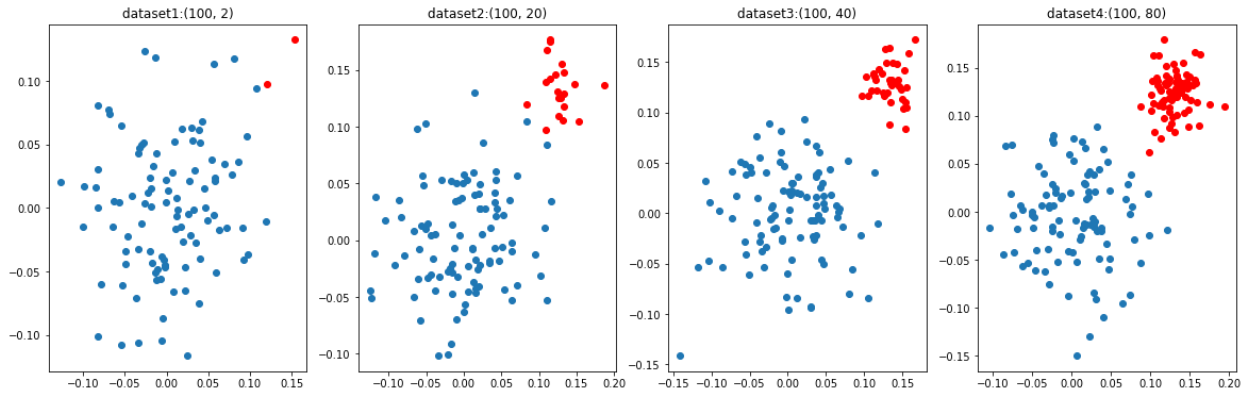
TASK 1

Applying SVM on Imbalanced Dataset

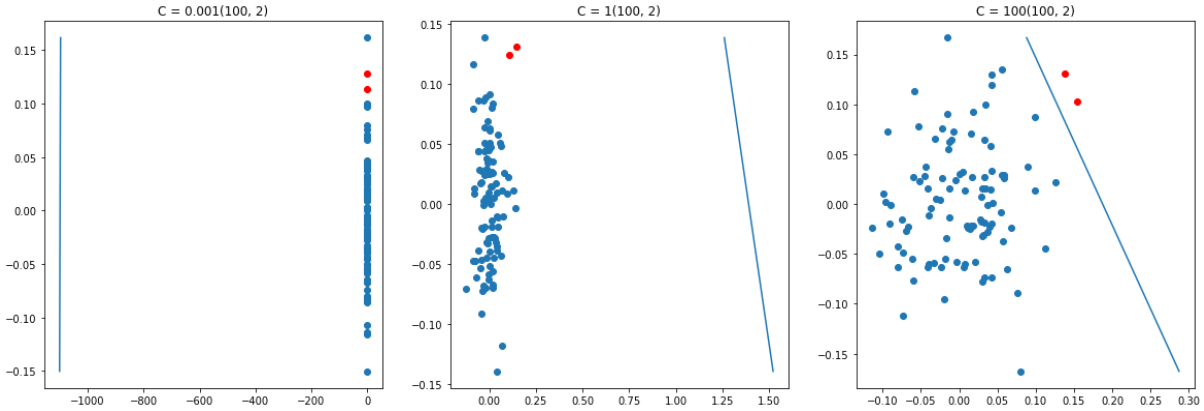
In [1]:
```
#Creating  the Imbalanced datasets each of these will have a positive and negative classes
#Dataset-1: 100 positive and 2 Negative points
#Dataset-2: 100 positive and 20 Negative points
#Dataset-3: 100 positive and 40 Negative points
#Dataset-4: 100 positive and 80 Negative points
#creating these datasets and plotting them

import numpy as np
import matplotlib.pyplot as plt
ratios=[(100,2),(100,20),(100,40),(100,80)]
plt.figure(figsize=(20,6))
#https://www.geeksforgeeks.org/enumerate-in-python/
#https://www.geeksforgeeks.org/matplotlib-pyplot-subplot-function-in-python/
for i,j in enumerate(ratios):
    plt.subplot(1,4,i+1)
 #Lets take this in Normal distributed with mean and variance so that it will be easy for us
    X_p=np.random.normal(0,0.05,size=(j[0],2))
    X_n=np.random.normal(0.13,0.02,size=(j[1],2))
    y_p=np.array([1]*j[0]).reshape(-1,1)
    y_n=np.array([0]*j[1]).reshape(-1,1)
#https://www.geeksforgeeks.org/numpy-vstack-in-python/
    X=np.vstack((X_p,X_n))
    y=np.vstack((y_p,y_n))
    plt.title("dataset"+str(i+1)+":"+str(j))
    plt.scatter(X_p[:,0],X_p[:,1])
    plt.scatter(X_n[:,0],X_n[:,1],color='red')
plt.show()
```



In [2]:
```
#Now we will consider three different values for our regularization c=0.001,1,100 and observe how the hyperpl
#with the changing regularization term on our imbalanced dataset

#https://stackoverflow.com/questions/65584316/how-to-plot-decision-boundaries-of-svm-with-different-kernels-3
# for the separating hyper plane ax+by+c=0, the weights are [a, b] and the intercept is c
    # to draw the hyper plane we are creating two points
    # 1. ((b*min-c)/a, min) i.e ax+by+c=0 ==> ax = (-by-c) ==> x = (-by-c)/a here in place of y we are keepin
    # 2. ((b*max-c)/a, max) i.e ax+by+c=0 ==> ax = (-by-c) ==> x = (-by-c)/a here in place of y we are keepin

def draw_line(coef,intercept,mi,ma):
    points=np.array([[((-coef[1]*mi-intercept)/coef[0]),mi],[((-coef[1]*ma-intercept)/coef[0]),ma]])
    plt.plot(points[:,0],points[:,1])
```

In [ ]:

In [4]:
```
#now considering different values of regularization
from sklearn import svm
c = [0.001,1,100]
plt.figure(figsize = (20,30))
ratios = [(100,2), (100, 20), (100, 40), (100, 80)]
num=1
for j,i in enumerate(ratios):
    for k in range(0, 3):
        model=svm.LinearSVC(C=c[k])
        plt.subplot(4, 3, num)
        num=num+1
        X_p=np.random.normal(0,0.05,size=(i[0],2))
        X_n=np.random.normal(0.13,0.02,size=(i[1],2))
        y_p=np.array([1]*i[0]).reshape(-1,1)
        y_n=np.array([0]*i[1]).reshape(-1,1)
        X=np.vstack((X_p,X_n))
        y=np.vstack((y_p,y_n))
        model.fit(X,y)
        plt.scatter(X_p[:,0],X_p[:,1])
        plt.scatter(X_n[:,0],X_n[:,1],color='red')
        plt.title('C = '+ str(c[k])+str(i))
        draw_line(coef=model.coef_[0],intercept=model.intercept_,ma=max(X[:,1]), mi= min(X[:,1]))

    plt.show()
```



As c is very small model is unable to classify the data hyperplane is far from the actual points so it might be balanaced or unbalanced nothing being particular when c is small

#when c is 1 Hyperplane is far from the points and this model cannot classify Imbalanced Dataset but as we seen for balanaced #Data set it is working somewhat but not perfectly classifying

when c is 100 The model is not able to classify the highly imbalanced dataset as even with a high value of c. So we can conclude that this model does not work well, or it's not recommended to use this when we have a highly imbalanced dataset.

In [ ]: