```
# laplaceTrainer.py
# Laplace Trainer

import numpy
class LaplaceTrainer(object):

    def loadTrainDataSet(self, inputData):
        #Parameter initialization
        self.cxy = numpy.ones((inputData.inputNum,2,2))
        self.cy = numpy.ones(2)*2
        self.pxy = numpy.zeros((inputData.inputNum,2,2))
        self.py = numpy.zeros(2)
        self.px_y = numpy.zeros((inputData.inputNum,2,2))
        # Count all the X_i
        for i in range(0,len(inputData.rowInputList)):
            for j in range(0, len(inputData.rowInputList[i])):
                for k in range(0,len(inputData.rowOutputList[i])):
                    self.cxy[j][int(inputData.rowInputList[i][j])][int(inputData.rowOutputList[i][k])] += 1
        # Calculate probabilities
        self.pxy = self.cxy / (inputData.rowNum+2*2)
        #Count all the Y
        for i in range(0,len(inputData.rowOutputList)):
            self.cy[int(inputData.rowOutputList[i][0])] += 1
        # Calculate probabilities
        self.py = self.cy/ (inputData.rowNum+2*2)
        py2 = numpy.zeros((inputData.inputNum,2,2))
        for i in range(0, (self.pxy.shape)[0]):
            for j in range(0, (self.pxy.shape)[1]):
                for k in range(0, (self.pxy.shape)[2]):
                    py2[i][j][k] = self.py[k]
        self.px_y = self.pxy / py2

    def loadTestDataSet(self, testData):
        # Parameter initialization
        self.ty = numpy.zeros(testData.rowNum)
        self.correct = 0
        self.correctRate = float(0)
        self.rowMax_0 = numpy.zeros(testData.rowNum)
        self.rowMax_1 = numpy.zeros(testData.rowNum)
        for i in range(0, len(testData.rowInputList)):
            for j in range(0, len(testData.rowInputList[i])):
                #Load the current probability of Y = 0 given certain X
                px_yNow_0 = self.px_y[j][int(testData.rowInputList[i][j])][0]
                if px_yNow_0 > 0:
                    self.rowMax_0[i] += numpy.log(px_yNow_0)
                else:
                    #Use -10 to replace -inf
                    self.rowMax_0[i] += -10
                # Load the current probability of Y = 0 given certain X
                px_yNow_1 = self.px_y[j][int(testData.rowInputList[i][j])][1]
                if px_yNow_1 > 0:
                    self.rowMax_1[i] += numpy.log(px_yNow_1)
                else:
                    #Use -10 to replace -inf
                    self.rowMax_1[i] += -10
            self.rowMax_0[i] += numpy.log(self.py[0])
            self.rowMax_1[i] += numpy.log(self.py[1])
            if self.rowMax_0[i] > self.rowMax_1[i]:
                    self.ty[i] = 0
            else:
                    self.ty[i] = 1
            if self.ty[i] == testData.rowOutputList[i][0]:
                    self.correct += 1

        self.correctRate = self.correct / float(testData.rowNum)

    def printPx_y(self, x_i,x, y):
        output = self.px_y[x_i][x][y]
        print("Laplace: P(X[%d] = %d|Y= %d) = %f"%(x_i, x, y, output))

    def printAccuracy(self):
        print("Laplace: Accuracy = %f"%(self.correctRate))

    def printPY(self,y):
        print("Laplace: P(Y = %d) = %f"%(y, self.py[int(y)]))
```