CODE

```
import math
import csv
import random

def splitDataset(datasat, train_data, test_data):
	for i in range(len(dataset)):
		if(i<=int(0.75*len(dataset))):
		#if(i<len(dataset)-1):
			train_data.append(dataset[i])
		else:
			test_data.append(dataset[i])
	print("The Training Data is :")
	for row in train_data:
		print(row)
	print("\nThe Test Data is :")
	for row in test_data:
		print(row)

def separateByClass(dataset):
	separated = {}
	for i in range(len(dataset)):
		row = dataset[i]
		if (row[-1] not in separated):
			separated[row[-1]] = []
		separated[row[-1]].append(row)
	return separated

def mean(numbers):
	if(len(numbers)==0):
		return 0
	else:
		return sum(numbers)/float(len(numbers))

def stdev(numbers):
	avg = mean(numbers)
	if(len(numbers)-1==0):
		return 0;
	else:
		return math.sqrt(sum([pow(x-avg,2) for x in numbers])/float(len(numbers)-1))

def summarize(dataset):
	summaries = [(mean(attribute), stdev(attribute)) for attribute in zip(*dataset)[:-1]]
	#del summaries[-1]
	return summaries

def summariseData(dataset):
	separated = separateByClass(dataset)
	summary = {}
	for key, value in separated.iteritems():
		summary[key] = summarize(value)
	return summary


def calculateProbability(x, mean, stdev):
	if(2*math.pow(stdev,2) == 0):
		exponent = math.exp(0)
```

```
        else:
                exponent = math.exp(-(math.pow(x-mean,2)/(2*math.pow(stdev,2))))
        if(stdev==0):
                return 0;
        else:
                return (1 / (math.sqrt(2*math.pi) * stdev)) * exponent

def calculateClassProbabilities(summaries, inputVector):
        probabilities = {}
        for classValue, classSummaries in summaries.iteritems():
                probabilities[classValue] = 1
                for i in range(len(classSummaries)):
                        mean, stdev = classSummaries[i]
                        x = inputVector[i]
                        probabilities[classValue] *= calculateProbability(x, mean, stdev)
        return probabilities

def predict(summaries, inputVector):
        probabilities = calculateClassProbabilities(summaries, inputVector)
        bestLabel, bestProb = None, -1
        for classValue, probability in probabilities.iteritems():
                if bestLabel is None or probability > bestProb:
                        bestProb = probability
                        bestLabel = classValue
        return bestLabel

def getPredictions(summary, testSet):
        predictions = []
        for i in range(len(testSet)):
                result = predict(summary, testSet[i])
                predictions.append(result)
        return predictions

def getAccuracy(testSet, predictions):
        correct = 0
        for i in range(len(testSet)):
                if testSet[i][-1] == predictions[i]:
                        correct += 1
        return (correct/float(len(testSet))) * 100.0

#Loading and splitting dataset
data = csv.reader(open("shoeNew.csv"),delimiter=",")
dataset=list()
train_data=list()
test_data=list()
for row in data:
        dataset.append(row)
for i in range(len(dataset)):
                dataset[i] = [float(x) for x in dataset[i]]
splitDataset(dataset, train_data, test_data)
#Summarise Data
summary = summariseData(train_data)
#Make prediction
prediction = getPredictions(summary, test_data)
#Get Accuracy
accuracy = getAccuracy(test_data, prediction)
print("\nThe Accuracy is :\t"+str(accuracy)+"%\n")
```

OUTPUT

The Training Data is :
[5.75, 11.0, 1.0]
[5.83, 10.0, 1.0]
[5.33, 7.0, 2.0]
[5.33, 6.0, 2.0]
[6.0, 13.0, 1.0]
[4.92, 4.0, 2.0]
[6.0, 12.0, 1.0]
[5.75, 12.0, 1.0]
[5.42, 9.0, 2.0]
[5.5, 8.0, 2.0]
[5.25, 6.0, 2.0]
[4.83, 5.0, 2.0]
[5.5, 8.0, 1.0]
[5.75, 8.0, 1.0]
[6.0, 11.0, 1.0]
[5.92, 10.0, 1.0]

The Test Data is :
[6.08, 12.0, 1.0]
[6.17, 12.0, 1.0]
[6.25, 12.0, 1.0]
[5.75, 11.0, 1.0]
[6.0, 10.0, 2.0]

The Accuracy is : 80.0%