Assignment 3 – Machine Learning

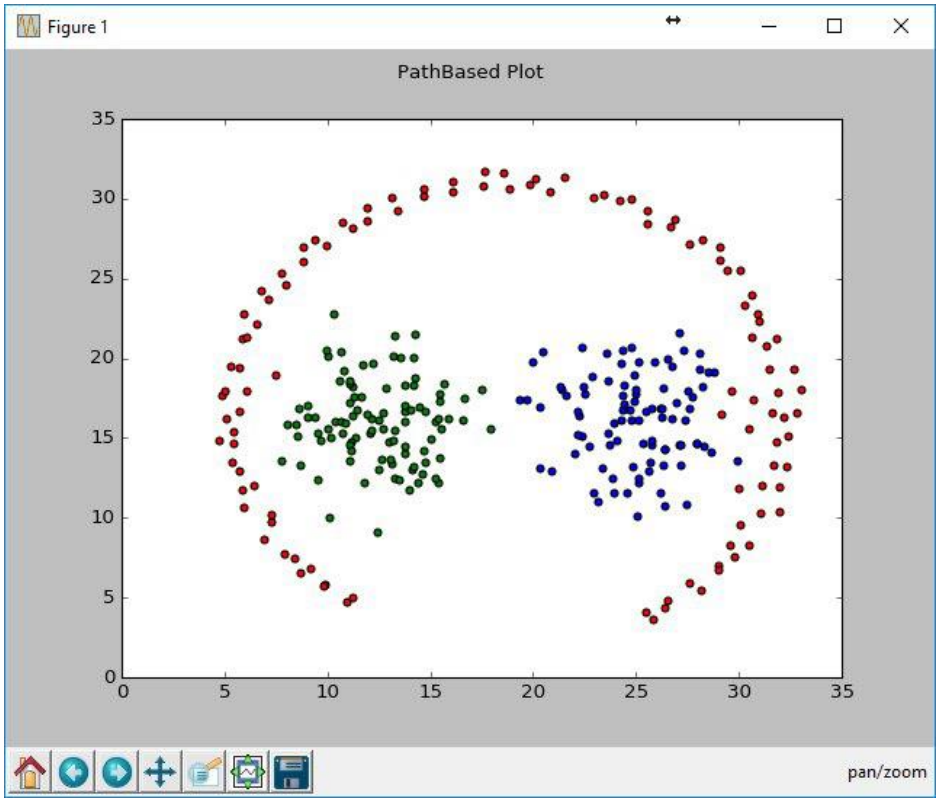
Satria Nur Hidayatullah

1103130035

1. Fungsi bernama : performance_calculator
   List Atribute:
   - Types: [0 = F1-Micro, 1 = F2-Macro, 2 = Simple Accuracy
   - Target: vector berisi target dari dataset
   - Predict: vector berisi prediksi hasil classifier

```
# types: 0 = F1-Micro, 1 = F1-Macro, 2 = Simple Accuracy
def performance_calculator(types, target, predict):
    totaldata = target.shape[0]
    if types == 2:
        totaltrue = 0.0
        for i in xrange(totaldata):
            if target[i] == predict[i]:
                totaltrue += 1
        return totaltrue / totaldata
    else:
        list_class = np.unique(target)
        confmatrix = np.array([np.array([0.0 for i in xrange(4)]) for i in xrange(list_class.shape[0])])
        for i in xrange(totaldata):
            confmatrix += [0,0,0,1]
            chosen = np.where(list_class==target[i])
            if target[i] == predict[i]:
                confmatrix[chosen] += [1,0,0,-1]
            else:
                confmatrix[chosen] += [0,1,0,-1]
                confmatrix[np.where(list_class==predict[i])] += [0,0,1,-1]
        if types == 0:
            confmatrix = confmatrix.sum(axis=0)
            precision = confmatrix[0] / (confmatrix[0] + confmatrix[2])
            recall = confmatrix[0] / (confmatrix[0] + confmatrix[1])
        else:
            precision = (confmatrix[:,0] / (confmatrix[:,0] + confmatrix[:,2])).sum()/confmatrix.shape[0]
            recall =  (confmatrix[:,0] / (confmatrix[:,0] + confmatrix[:,1])).sum()/confmatrix.shape[0]
        return 2*(precision * recall)/(precision + recall)
```

2. Naïve Bayes
   a. Visualisasi scatter plot data pathbased



b. Menggunakan Naïve bayes untuk classifier pathbased

i. Fungsi bernama: naïve_learn

Parameter: data(yaitu berisi data train), targetcol(kolom yang menyatakan target class)

Output: Prior Probability, Standar deviasi dari tiap kolom, rata-rata dari tiap kolom

Fungsi ini dikhususkan untuk learning naïve bayes pada data kontinu(semua kolom)

```
def naive_learn(data, targetcol):
    list_class = np.unique(data[:,targetcol].astype('i'))
    dataperclass,prior,mean,std = {},{},{},{}
    likelihood = []

    for i in list_class:
        dataperclass[i] = data[data[:,targetcol] == i]
        prior[i] = dataperclass[i].shape[0] / (data.shape[0] + 0.0)
        mean[i] = [np.mean(dataperclass[i][:,j]) for j in xrange(dataperclass[i].shape[1] - 1)]
        std[i] = [np.std(dataperclass[i][:,j]) for j in xrange(dataperclass[i].shape[1] - 1)]

    return prior,mean,std
```

ii. Fungsi bernama: naïve_bayes

Parameter: data(berisi data test), model(berisi probabilitas yang didapatkan dari fungsi learn)

Output: Vektor berisikan prediksi class tiap data test

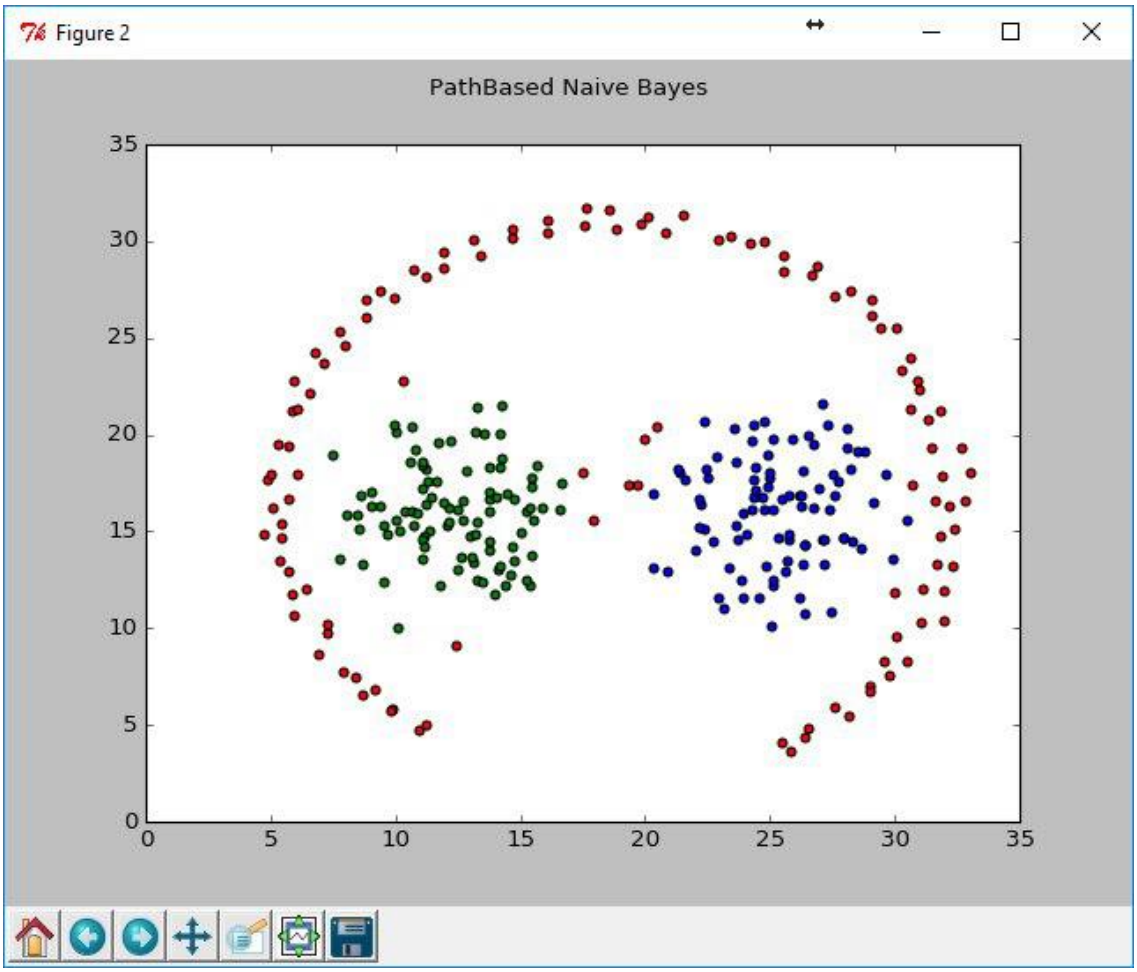
Fungsi ini dikhususkan untuk testing naïve bayes pada data kontinu(semua kolom)

```
def naive_bayes(data, model):
    prior, mean, std = model
    classification_result = []
    gausbayes = lambda m,sd,x: (1/(sd*np.sqrt(2*np.pi)))*np.exp(-(((x-m)**2)/(2*sd**2)))
    likelihood = []

    for values in data:
        likelihood.append({i: [gausbayes(mean[i][j],std[i][j],values[j]) for j in xrange(len(mean[i]))] for i in prior.keys()})

    for x in xrange(data.shape[0]):
        posterior = {}
        for i in prior.keys():
            posterior[i] = np.log(prior[i]) + np.sum(np.log(likelihood[x][i]))
        classification_result.append(max(posterior, key=posterior.get))
    return classification_result
```

iii. Hasil plot dengan klasifikasi naïve bayes



iv. Dari kedua grafik scatter plot diatas, terlihat bahwa dengan menggunakan classifier Naïve bayes cukup untuk melakukan klasifikasi pada dataset pathbased dengan akurasi yang tinggi, karena terlihat bahwa hanya beberapa titik saja yang tidak sama.

c. 96%

d. Hasil decision boundary menggunakan model yang didapat dari Naïve bayes: