ARUM YUMNA ZAHRAH
IFIK 40-01 / 1301164686

LAPORAN TUGAS KECERDASAN BUATAN
"KLASIFIKASI MENGGUNAKAN ALGORITMA K-NEAREST NEIGHBOR"

1. Analisis Masalah

   Diberikan suatu dataset berupa file Data Latih (DataTrain_Tugas2_AI.csv) yang berisi 4000 data dengan 4 atribut bilangan riil yang memiliki 2 kelas. System klasifikasi akan mengklasifikasikan Data Uji (DataTest_Tugas2_AI.csv) yang berisi 1000 data uji menggunakan algoritma KNN (K-Nearest Neighbor). System klasifikasi akan membaca masukan file data uji dan mengeluarkan output berupa file berisi 1000 data prediksi kelas 0/1 sesuai dengan urutan.

2. Strategi Penyelesaian Masalah

   Untuk menentukan nilai K agar mendapatkan nilai akurasi optimal (paling baik), langkah-langkah yang digunakan adalah sebagai berikut:

   1. Load file Data Train dan Data Test. Kemudian buat list sesuai dengan atribut masing-masing data.

```
datatrain = {
    "Atribut1":[],
    "Atribut2":[],
    "Atribut3":[],
    "Atribut4":[],
    "Kelas":[]
}

datatest = {
    "Atribut1":[],
    "Atribut2":[],
    "Atribut3":[],
    "Atribut4":[],
}
```

```
def loadfile():
    file = open("DataTrain_Tugas2_AI.csv", "r")
    for line in file:
        data = []
        data = line.split(",")
        datatrain["Atribut1"].append(data[0])
        datatrain["Atribut2"].append(data[1])
        datatrain["Atribut3"].append(data[2])
        datatrain["Atribut4"].append(data[3])
        datatrain["Kelas"].append(data[4])

    datatrain["Atribut1"].remove(datatrain["Atribut1"][0])
    datatrain["Atribut2"].remove(datatrain["Atribut2"][0])
    datatrain["Atribut3"].remove(datatrain["Atribut3"][0])
    datatrain["Atribut4"].remove(datatrain["Atribut4"][0])
    datatrain["Kelas"].remove(datatrain["Kelas"][0])

    datatrain["Atribut1"] = list(map(float, datatrain["Atribut1"]))
    datatrain["Atribut2"] = list(map(float, datatrain["Atribut2"]))
    datatrain["Atribut3"] = list(map(float, datatrain["Atribut3"]))
    datatrain["Atribut4"] = list(map(float, datatrain["Atribut4"]))
    datatrain["Kelas"] = list(map(int, datatrain["Kelas"]))
    file.close()

    file = open("DataTest_Tugas2_AI.csv", "r")
    for line in file:
        data = []
        data = line.split(",")
        datatest["Atribut1"].append(data[0])
        datatest["Atribut2"].append(data[1])
        datatest["Atribut3"].append(data[2])
        datatest["Atribut4"].append(data[3])

    datatest["Atribut1"].remove(datatest["Atribut1"][0])
    datatest["Atribut2"].remove(datatest["Atribut2"][0])
    datatest["Atribut3"].remove(datatest["Atribut3"][0])
    datatest["Atribut4"].remove(datatest["Atribut4"][0])

    datatest["Atribut1"] = list(map(float, datatest["Atribut1"]))
    datatest["Atribut2"] = list(map(float, datatest["Atribut2"]))
    datatest["Atribut3"] = list(map(float, datatest["Atribut3"]))
    datatest["Atribut4"] = list(map(float, datatest["Atribut4"]))
    file.close()
```

   2. Buat fungsi untuk menghitung jarak dengan setiap neighbor menggunakan rumus Euclidean Distance.

$$d(x_1, x_2) = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + \ldots + (x_{1p} - x_{2p})^2} = \sqrt{\sum_{j=1}^{p} (x_{1j} - x_{2j})^2}$$

```
def distance(datatest, datatrain, k):
    hasil1 = []
    for i in range(len(datatest["Atribut1"])):
        for j in range(len(datatrain["Atribut1"])):
            hasil2 = math.sqrt((datatrain["Atribut1"][j]-datatest["Atribut1"][i])**2+(datatrain["Atribut2"][j]-datatest["Atribut2"][i])**2+(datatrain["Atribut3"][j]-datatest["Atribut3"][i])**2+(datatrain["Atribut4"][j]-datatest["Atribut4"][i])**2)
            hasil1.append(hasil2)
        yield KNN(k, hasil1, datatrain)
        hasil1 = []
```

   3. Tentukan data tersebut masuk ke kelas 0 atau 1 berdasarkan jarak neighbor.

```
def KNN(k, hasil1, datatest):
    hasil2 = []
    minimum = []
    rank = []
    for i in hasil1:
        hasil2.append(i)
    for i in range(k):
        minimum.append(min(hasil2))
        hasil2.remove(min(hasil2))
    idxhasil = list(map(hasil1.index,minimum))
    kelas = list(map(lambda x: datatest["Kelas"][x],idxhasil))
    rank = kelas.count(1)
    if rank > (k/2):
        return 1
    else:
        return 0
```

4. Tentukan nilai K optimal berdasarkan nilai akurasi yang paling optimal. Hitung nilai akurasi menggunakan rumus akurasi dengan range nilai K antara 3 hingga 8 dengan selisih 2, lalu tentukan nilai akurasi yang paling baik. Maka, kita dapat menentukan nilai K yang dicari.

```
def nilaiK(datatrain, datatest):
    k = []
    accuracy = []
    for i in range(3, 8, 2):
        kelas = list(distance(datatrain, datatest, i))
        accuracy1 = 0
        for j in range(len(kelas)):
            if kelas[j] == datatrain["Kelas"][j]:
                accuracy1+=1
            else:
                continue
        accuracy2 = (accuracy1/len(kelas))*100
        k.append(i)
        accuracy.append(accuracy2)
    idx = accuracy.index(max(accuracy))
    print(range)
    print("K = ", k[idx])
    print("Akurasi = ", accuracy[idx])
    return k[idx]
```

5. Panggil semua fungsi pada main sesuai dengan tujuan fungsi, lalu tulis hasil klasifikasi pada file Prediksi_Tugas2AI_1301164686.csv.

```
def main():
    loadfile()

    k = nilaiK(datatrain, datatrain)
    datatest["Kelas"] = list(distance(datatest, datatrain, k))
    file = open("Prediksi_Tugas2AI_1301164686.csv", "w")
    file.write("Atribut1, Atribut2, Atribut3, Atribut4, Kelas\n")
    for i in range(len(datatest["Atribut1"])):
        data = str(datatest["Atribut1"][i]) +","+ str(datatest["Atribut2"][i]) +","+ str(data
test["Atribut3"][i]) +","+ str(datatest["Atribut4"][i]) +","+ str(datatest["Kelas"][i]) +"\n"
        file.write(data)
    file.close

if __name__ == "__main__":
    main()
```

```
<class 'range'>
K =  3
Akurasi =  78.57499999999999
```

Berdasarkan running diatas, hasil yang didapatkan adalah nilai K optimum = 3 dengan nilai akurasi = 78.57499.