Nama: Tri Ayu Syifa'ur Rohmah

Kelas: IF-42-04

NIM : 1301180254

1. Pemilihan ukuran jarak yang digunakan

   Metode ukuran jarak yang digunakan adalah Euclidean distance, karena pada permasalahan program diabetes, Euclidean distance paling cocok dan umum digunakan, serta implementasi ke code nya lebih mudah. Jarak yang dihitung dimulai dari indeks ke 0 sampai ke 7 (ada 8 kolom).

```
def euclideandistance(test,train):
  res = (((test[0]-train[0])**2) + ((test[1]-train[1])**2) + ((test[2]-train[2])**2) + ((test[3]-train[3])**2) + ((test[4]-train[4])**2) + ((test[5]-train[5])**2)
  return math.sqrt(res)
print (euclideandistance([7,13,4,8,6,7,4,5],[4,7,3,8,7,8,10,5]))
```

```
9.16515138991168
```

2. Teknik prapremrosesan data dan Teknik rekayasa fitur

```
[ ] zeronotaccepted = ['Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'Insulin','Pregnancies']
```

```
[ ] for column in zeronotaccepted:
      data[column] = data[column].replace(0, np.NaN)
      mean = int(data[column].mean(skipna=True))
      data[column] = data[column].replace(np.NaN, mean)
    print(data)
```

```
     Pregnancies  Glucose  ...  Age  Outcome
0            6.0    148.0  ...   50        1
1            1.0     85.0  ...   31        0
2            8.0    183.0  ...   32        1
3            1.0     89.0  ...   21        0
4            4.0    137.0  ...   33        1
..           ...      ...  ...  ...      ...
763         10.0    101.0  ...   63        0
764          2.0    122.0  ...   27        0
765          5.0    121.0  ...   30        0
766          1.0    126.0  ...   47        1
767          1.0     93.0  ...   23        0

[768 rows x 9 columns]
```

Pada Teknik praperosesan dihilangkan nilai 0 pada kolom glucose, bloodpressure, skinthicknes, BMI, insulin, dan pregnancies karena nilai 0 dianggap tidak masuk akal.

3. Strategi penggunaan algoritma kNN dan pemilihan nilai k terbaik

```
def fungsiKNN(k, train,test, ytrain):
  yprediksi = []
  for i in range(len(test)):
    distance = []
    for j in range(len(train)):
      distance.append([euclideandistance(test[i],train[j]),ytrain[j]])
    sortdistance = sorted(distance)
    for j in range(0,k):
      no = 0
      yes = 0
      if sortdistance[j][1] == 0:
        no += 1
      elif sortdistance[j][1] == 1:
        yes += 1
    if yes > no:
        yprediksi.append(1)
    else:
        yprediksi.append(0)
  return yprediksi
p1 = fungsiKNN(1,train1,test1,ytrain1)
print(p1)
print(len(p1))
```

```
[0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0,
154
```

```
samevalues1 = 0
for i in range(len(prediksi1)):
  if prediksi1[i] == ytest1[i]:
    samevalues1 += 1
samevalues2 = 0
for i in range(len(prediksi2)):
  if prediksi2[i] == ytest2[i]:
    samevalues2 += 1
samevalues3 = 0
for i in range(len(prediksi3)):
  if prediksi3[i] == ytest3[i]:
    samevalues3 += 1
samevalues4 = 0
for i in range(len(prediksi4)):
  if prediksi4[i] == ytest4[i]:
    samevalues4 += 1
samevalues5 = 0
for i in range(len(prediksi5)):
  if prediksi5[i] == ytest5[i]:
    samevalues5 += 1
hasil1 = (samevalues1/len(prediksi1))*100
hasil2 = (samevalues2/len(prediksi2))*100
hasil3 = (samevalues3/len(prediksi3))*100
hasil4 = (samevalues4/len(prediksi4))*100
hasil5 = (samevalues5/len(prediksi5))*100

isiakurasi = (hasil1+hasil2+hasil3+hasil4+hasil5)/5
print(isiakurasi)
```

```
75.9536541889483
```

Mencari nilai jarak antara satu data ke data lain, kemudan jarak tersebut diurutkan dari nilai terkecil ke terbesar kemudian dilakukan pengalamatan terhadap data sebanyak k buah. Dari hasil percobaan untuk mendapatkan nilai k terbaik didapatkan hasil bahwa semakin kecil nilai k nya maka semakin besar nilai akurasinya. Sehingga nilai k terbaik adalah 1, dengan nilai akurasi 75.9536541889483