TUGAS 3

Laporan Observasi

Ditujukan untuk memenuhi salah satu tugas mata kuliah Pengantar Kecerdasan Buatan

Oleh:
Muhammad Daffa Regenta S (1301184291)

IF-42-08

PROGRAM STUDI INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2020

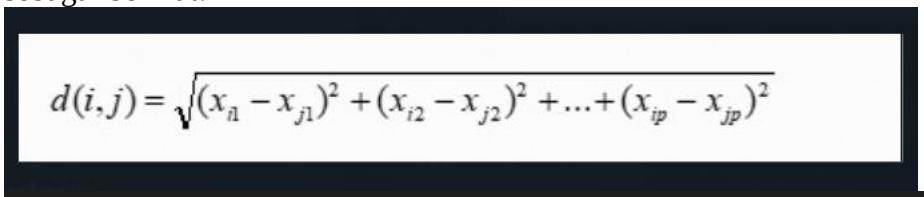
1. **Pendahuluan**

   Membuat sebuah system menggunakan basis K-Nearest Neighbor untuk estimasi nilai K terbaik menggunakan skema 5-Fold Cross Validation.

2. **Penilitan dan Pembahasan**

   2.1 Perhitungan jarak

   Metode yang digunakan menghitung jarak adalah metode Euclidean dengan rumus sebagai berikut.



$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \ldots + (x_{ip} - x_{jp})^2}$$

```python
#Menghitung Jarak
#Muhammad Daffa Regenta S - 1301184201 - IF4208
def euclideanDistance(data1,data2):
  sum = pow((data1['Pregnancies'] - data2['Pregnancies']),2) + pow((data1['Glucose'] - data2['Glucose']),2) + pow((data1['BloodPressure'] - data2['BloodPressure']),2) +
        pow((data1['SkinThickness'] - data2['SkinThickness']),2) + pow((data1['Insulin'] - data2['Insulin']),2) + pow((data1['BMI'] - data2['BMI']),2) + pow((data1['DiabetesPedigreeFunction'] -
        data2['DiabetesPedigreeFunction']),2) + pow((data1['Age'] - data2['Age']),2)
  return math.sqrt(sum)
```

   2.2 Pre-processing Data

   Metode menggunakan normalisasi, cara ini digunakan untuk memperkecil nilai, serta dapat mewakili nilai data asli nya.

$$x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}}$$

```python
#Muhammad Daffa Regenta S - 1301184291 - IF4208
abet['Pregnancies'] = abet['Pregnancies']/(abet['Pregnancies'].max())
abet['Glucose'] = abet['Glucose']/(abet['Glucose'].max())
abet['BloodPressure'] = abet['BloodPressure']/(abet['BloodPressure'].max())
abet['SkinThickness'] = abet['SkinThickness']/(abet['SkinThickness'].max())
abet['Insulin'] = abet['Insulin']/(abet['Insulin'].max())
abet['BMI'] = abet['BMI']/(abet['BMI'].max())
abet['DiabetesPedigreeFunction'] = abet['DiabetesPedigreeFunction']/(abet['DiabetesPedigreeFunction'].max())
abet['Age'] = abet['Age']/(abet['Age'].max())
abet
```

2.3 kNN

Pemilihan nilai K terbaik dengan akurasi dengan klasifikasi kNN dengan mengoutputkan nilai K.

```
Nilai K 1 Dengan Akurasi  68.21784666945958
Nilai K 3 Dengan Akurasi  72.23963133640552
Nilai K 5 Dengan Akurasi  73.27691663175534
Nilai K 7 Dengan Akurasi  73.27943024717219
Nilai K 9 Dengan Akurasi  73.79639715123587
Nilai K 11 Dengan Akurasi  75.09509844993715
Nilai K 13 Dengan Akurasi  75.48554671135317
Nilai K 15 Dengan Akurasi  74.96857980728949
Nilai K 17 Dengan Akurasi  73.41013824884793
Nilai K 19 Dengan Akurasi  74.05781315458734
Nilai K 21 Dengan Akurasi  74.18684541265186
Nilai K 23 Dengan Akurasi  73.7980728948471
Nilai K 25 Dengan Akurasi  73.40930037704231
Nilai K 27 Dengan Akurasi  74.18684541265186
Nilai K 29 Dengan Akurasi  73.66820276497695
Nilai K 31 Dengan Akurasi  73.5374947633012
Nilai K 33 Dengan Akurasi  73.92542940930036
Nilai K 35 Dengan Akurasi  73.6648512777545
Nilai K 37 Dengan Akurasi  73.66652702136572
Nilai K 39 Dengan Akurasi  74.05697528278174
```

```python
#Muhammad Daffa Regenta S - 1301184291 - IF4208
#kNN
def klasifikasi(objek1,objek2,k):
  data1,data2 = [], []
  for i in range(len(objek2)):
    temp1, temp2 = [], []
    for j in range(len(objek1)):
      total = euclideanDistance(objek1[j],objek2[i])
      temp1.append(total)

    data1.append(objek2[i]['Outcome'])
    hasil = np.argsort(temp1)[:k]

    for l in hasil:
      temp2.append(objek1[l]['Outcome'])
    select = Counter(temp2).most_common(1)
    data2.append(select[0][0])
  acy = accuracy_score(data1,data2)
  return acy
```

2.4 Perhitungan rata rata akurasi kNN menggunakan 5-fold Cross-Validation beserta output.

Setelah menghitung menggunakan algortima kNN maka menghasilkan seperti berikut.

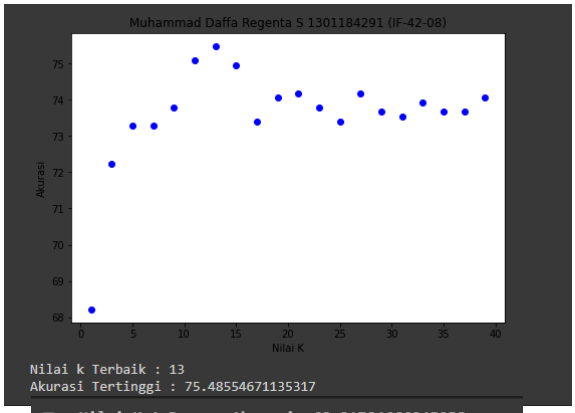
```python
#Muhammad Daffa Regenta S - 1301184291 - IF4208
#kNN
def klasifikasi(objek1,objek2,k):
  data1,data2 = [], []
  for i in range(len(objek2)):
    temp1, temp2 = [], []
    for j in range(len(objek1)):
      total = euclideanDistance(objek1[j],objek2[i])
      temp1.append(total)

    data1.append(objek2[i]['Outcome'])
    hasil = np.argsort(temp1)[:k]

    for l in hasil:
      temp2.append(objek1[l]['Outcome'])
    select = Counter(temp2).most_common(1)
    data2.append(select[0][0])
  acy = accuracy_score(data1,data2)
  return acy
```

```python
#main

temp_acy, temp_k = [], []
k = 1
for hitung in range(0,20):
  skuy1 = klasifikasi(dataset_training1,dataset_testing1,k)
  skuy2 = klasifikasi(dataset_training2,dataset_testing2,k)
  skuy3 = klasifikasi(dataset_training3,dataset_testing3,k)
  skuy4 = klasifikasi(dataset_training4,dataset_testing4,k)
  skuy5 = klasifikasi(dataset_training5,dataset_testing5,k)
  hasil = ((skuy1 + skuy2 + skuy3 + skuy4 + skuy5)/5)*100
  print('Nilai K',k,'Dengan Akurasi ',hasil)
  temp_k.append(k)
  k+=2
  temp_acy.append(hasil)
```




```
Nilai k Terbaik : 13
Akurasi Tertinggi : 75.48554671135317
```

```
Nilai K 1 Dengan Akurasi  68.21784666945958
Nilai K 3 Dengan Akurasi  72.23963133640552
Nilai K 5 Dengan Akurasi  73.27691663175534
Nilai K 7 Dengan Akurasi  73.27943024717219
Nilai K 9 Dengan Akurasi  73.79639715123587
Nilai K 11 Dengan Akurasi  75.09509844993715
Nilai K 13 Dengan Akurasi  75.48554671135317
Nilai K 15 Dengan Akurasi  74.96857980728949
Nilai K 17 Dengan Akurasi  73.41013824884793
Nilai K 19 Dengan Akurasi  74.05781315458734
Nilai K 21 Dengan Akurasi  74.18684541265186
Nilai K 23 Dengan Akurasi  73.7980728948471
Nilai K 25 Dengan Akurasi  73.40930037704231
Nilai K 27 Dengan Akurasi  74.18684541265186
Nilai K 29 Dengan Akurasi  73.66820276497695
Nilai K 31 Dengan Akurasi  73.5374947633012
Nilai K 33 Dengan Akurasi  73.92542940930036
Nilai K 35 Dengan Akurasi  73.6648512777545
Nilai K 37 Dengan Akurasi  73.66652702136572
Nilai K 39 Dengan Akurasi  74.05697528278174
```