LAPORAN TUGAS PEMROGRAMAN 03 – LEARNIG
CII-2M3 Pengantar Kecerdasan Buatan
Semester Genap 2021/2022

Oleh:

Citakamalia	(1301204122)
Indah Putri Maharani	(1301204181)
Iqlima Putri Hawa	(1301204337)

JURUSAN S1 - INFORMATIKA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

2022

DAFTAR ISI

BAB I

1.1 Latar Belakang

Pada masa seperti ini yang dimana banyaknya informasi dan dokumen yang tersedia sangat mendorong para pengguna untuk mencari cara lebih cepat dalam mendapatkan informasi dan dokumen yang dibutuhkan. Jika waktu pencarian terlalu lama, maka manfaat dari informasi yang diperoleh dapat berkurang. Hal ini dikarenakan informasi yang diperoleh sudah masuk waktu yang sudah tidak berguna atau tidak valid. Klasifikasi dokumen dapat membantu proses pencarian sebuah dokumen dengan cepat dan tepat. Klasifikasi dokumen mengelompokan dokumen yang sesuai dengan kategori yang terkandung pada dokumen tersebut. Permasalahan klasifikasi dokumen bisa diselesaikan dengan banyak metode, salah satu diantaranya adalah *K-Nearest Neighbor* (KNN).

1.2 Dasar Teori

Algoritma *K-Nearest Neighbor* (KNN) adalah sebuah metode klasifikasi terhadap sekumpulan data berdasarkan pembelajaran data yang sudah terklasifikasikan sebelumya. Algoritma KNN bekerja dengan mengambil sejumlah K data terdekat (tetangganya) sebagai acuan untuk menentukan kelas dari data baru. Algoritma ini mengklasifikasikan data berdasarkan similarity atau kemiripan atau kedekatannya terhadap data lainnya. Dalam *K-Nearest Neighbor*, data point yang berada berdekatan disebut “neighbor” atau “tetangga”. Tujuan dari algoritma ini adalah untuk mengklasifikasikan obyek baru berdasarkan atribut dan sample-sample dari training data. Secara umum, cara kerja algoritma KNN adalah sebagai berikut:

1. Menentukan jumlah tetangga (K) yang akan digunakan untuk pertimbangan penentuan kelas.
2. Menghitung jarak dari data baru ke masing-masing data point di dataset menggunakan Euclidean.
3. Ambil sejumlah K data dengan jarak terdekat, kemudian tentukan kelas dari data baru tersebut.

Euclidean Distance digunakan untuk menghitung jarak antara dua titik pada algoritma KNN dengan menjumlahkan seluruh selisih dari atribut yang telah dikuadratkan, kemudian diakarkan.

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

Manhattan Distance digunakan untuk menentukan kesamaan antara dua buah objek. menggunakan sebuah formula yang menjumlahkan seluruh selisih dari atribut yang telah dimutlakkan.

$$d(i,j) = \left|x_{i1} - x_{j1}\right| + \left|x_{i2} - x_{j2}\right| + \ldots + \left|x_{ip} - x_{jp}\right|$$

Minkowski Distance atau metrik Minkowski adalah metrik dalam ruang vektor bernorma yang dapat disebut sebagai generalisasi *Euclidean Distance* dan *Manhattan Distance* dengan menjumlahkan seluruh selisih dari atribut yang telah dimutlakkan dan dipangkatkan dengan nilai h, kemudian diakarkan dengan nilai h tersebut.

$$d(i,j) = \sqrt[h]{\left|x_{i1} - x_{j1}\right|^h + \left|x_{i2} - x_{j2}\right|^h + \ldots + \left|x_{ip} - x_{jp}\right|^h}$$

Supremum Distance menggunakan sebuah formula yang mencari nilai maksimum pada selisih dari atribut yang telah dimutlakkan.

$$d(i,j) = \lim_{h\to\infty}\left(\sum_{f=1}^{p}\left|x_{if} - x_{jf}\right|^h\right)^{\frac{1}{h}} = \max_{f}^{p}\left|x_{if} - x_{jf}\right|$$

BAB II

2.1 Deskripsi Masalah

Diberikan file **traintest.xlsx** yang terdiri dari dua sheet: **train dan test**, yang berisi dataset untuk problem klasifikasi biner (*binary classification*). Setiap record atau baris data dalam dataset tersebut secara umum terdiri dari nomor baris data (id), fitur input ($x1$ sampai $x3$), dan output kelas (y). Fitur input terdiri dari nilai-nilai integer dalam range tertentu untuk setiap fitur. Sedangkan output kelas bernilai biner (0 atau 1).

id	x1	x2	x3	y
1	60	64	0	1
2	54	60	11	0
3	65	62	22	0
4	34	60	0	1
5	38	69	21	0

Sheet train berisi 296 baris data, lengkap dengan target output kelas (y). Gunakan sheet ini untuk tahap pemodelan atau pelatihan (*training*) model sesuai metode yang Anda gunakan. Adapun sheet test berisi 10 baris data, dengan output kelas (y) yang disembunyikan. Gunakan sheet ini untuk tahap pengujian (*testing*) model yang sudah dilatih. Nantinya output program Anda untuk data uji ini akan dicocokkan dengan target atau kelas sesungguhnya.

2.2 Analisis Masalah

Program ini secara umum memiliki dua tahap, pelatihan (*training*) dan pengujian (*testing*). Pada tahap *training*, akan dihasilkan output berupa model yang sesuai dengan metode yang digunakan. Sedangkan pada tahap *testing* akan dihasilkan output berupa kelas (0 atau 1).

2.3 Proses yang Dibangun

1. Membaca File

Dalam proses ini kami menggunakan library pandas untuk membaca file excel traintest.xlsx yang nantinya akan digunakan sebagai bahan data penganalisisan. Kami juga menggunakan libraty math untuk melakukan operasi perhitungan matematika.

```
[ ]  import pandas as pd
     import numpy as np
     import xlwt
     import math
```

Import file train.xlsx yang berada di g-drive.

```
[ ]  !gdown --id 11K4sdfBlT6X0TyiF8f49OV0okEG53kmk

     /usr/local/lib/python3.7/dist-packages/gdown/cli.py:131: FutureWarning: Option `--id` was
       category=FutureWarning,
     Downloading...
     From: https://drive.google.com/uc?id=11K4sdfBlT6X0TyiF8f49OV0okEG53kmk
     To: /content/train.xlsx
     100% 23.4k/23.4k [00:00<00:00, 23.4MB/s]
```

Import file test.xlsx yang berada di g-drive.

```
[ ]  !gdown --id 13_3-CRFbeXcuip0CcyqkSiJYVjSiETbv

     /usr/local/lib/python3.7/dist-packages/gdown/cli.py:131: FutureWarning: Option `--id` was
       category=FutureWarning,
     Downloading...
     From: https://drive.google.com/uc?id=13_3-CRFbeXcuip0CcyqkSiJYVjSiETbv
     To: /content/test.xlsx
     100% 19.6k/19.6k [00:00<00:00, 25.6MB/s]
```

2. Perhitungan rumus jarak

Pada laporan ini, kami menggunakan empat model untuk menghitung ukuran jarak atribut, yaitu *Euclidean Distance*, *Manhattan Distance*, *Minkowski Distance*, dan *Supremum Distance*. Implementasi pada program sebagai berikut:

a. *Euclidean Distance*

```
[ ]  def euclidean(x, y):
       dist = 0
       for i in range(len(y)):
         dist += ((x[i] - y[i]) ** 2)
       return dist ** (1/2)
```

b. *Manhattan Distance*

```
[ ]  def manhattan(x, y):
       dist = 0
       for i in range(len(y)):
         dist += abs((x[i] - y[i]))
       return dist
```

c. *Minkowski Distance*

```
[ ]  def minskowski(x, y, k=2):
        dist = 0
        for i in range(len(x)):
            dist += abs(x[i] - y[i]) ** k
        return dist ** (1/k)
```

d. *Supremum Distance*

```
[ ]  def supremum(x, y):
       dist = []
       for i in range(len(x)):
         dist.append(abs(x[i] - y[i]))
       return max(dist)
```

BAB III

3.1 Implementasi Program

a. Inisialisasi

Disini kami membuat variabel yang bernama “data” yang dimana isinya merupakan sebuah array yang berfungsi sebagai inputan data frame yang nantinya akan dibandingkan dengan data train. Selain itu kami juga membuat beberapa array kosong yang akan menyimpan hasil dari perhitungan setiap rumusnya.

```
[ ]  # Input Data Frame (x1,x2,x3,y)
     data = [5, 6, 7, 8]
     pd.set_option('display.width', 200)
     pd.set_option('display.max_columns', None)

     def mainprogram(X):
       # Mendefinisikan array
       euclidean_list = []
       manhattan_list = []
       minkowski_list = []
       supremum_list = []
```

b. Proses penghitungan data

Memasukan data train dan data test kedalam rumus. Disini akan dilakukan perhitungan dari setiap rumusnya. Kemudian akan dimasukan hasil perhitungan tadi kedalam template list yang nantinya akan dituang kedalam file excel.

```
# Proses penghitungan Data
for x in temp:
  euclidean_list.append(euclidean(x, data))
  manhattan_list.append(manhattan(x, data))
  minkowski_list.append(minskowski(x, data))
  supremum_list.append(supremum(x, data))

train["Euclidean Distance"] = euclidean_list
train["Manhattan Distance"] = manhattan_list
train["Minkowski Distance"] = minkowski_list
train["Supremum Distance"]  = supremum_list
```

c. Proses pengurutan data da mengimport hasil data

Pada proses ini akan dilakukan penuangan hasil perhitungan sebelumnya ke dalam excel.

```
    # Sorting Data dan Mengimport hasil data ke Excel
    EuclideanData = train.sort_values(by=['Euclidean Distance'])[:X]
    EuclideanData['id'].to_excel('Euclidean.xls', index=False, header=True)

    ManhattanData = train.sort_values(by=['Manhattan Distance'])[:X]
    ManhattanData['id'].to_excel('Manhattan.xls', index=False, header=True)

    MinkowskiData = train.sort_values(by=['Minkowski Distance'])[:X]
    MinkowskiData['id'].to_excel('Minkowski.xls', index=False, header=True)

    SupremumData = train.sort_values(by=['Supremum Distance'])[:X]
    SupremumData['id'].to_excel('Supremum.xls', index=False, header=True)

    return ManhattanData, EuclideanData, MinkowskiData, SupremumData

EuclideanData, ManhattanData, MinkowskiData, SupremumData = mainprogram(6)
```

3.2 Output Program

a. Hasil terbaik data train.xlsx menggunakan Euclidean distance

```
==== Euclidean Distance ====
        id    x1    x2    x3    y  Euclidean Distance  Manhattan Distance  Minkowski Distance  Supremum Distance
5      6.0  33.0  58.0  10.0  1.0           59.548300                90.0           59.548300               52.0
82    83.0  31.0  59.0   2.0  1.0           59.657355                91.0           59.657355               53.0
105  106.0  30.0  62.0   3.0  1.0           61.854668                92.0           61.854668               56.0
89    90.0  37.0  59.0   6.0  1.0           62.313722                93.0           62.313722               53.0
256  257.0  34.0  61.0  10.0  1.0           62.641839                94.0           62.641839               55.0
285  286.0  31.0  65.0   4.0  1.0           64.923031                95.0           64.923031               59.0
```

b. Hasil terbaik data train.xlsx menggunakan Manhattan Distance

```
==== Manhattan Distance ====
        id    x1    x2    x3    y  Euclidean Distance  Manhattan Distance  Minkowski Distance  Supremum Distance
5      6.0  33.0  58.0  10.0  1.0           59.548300                90.0           59.548300               52.0
82    83.0  31.0  59.0   2.0  1.0           59.657355                91.0           59.657355               53.0
284  285.0  34.0  59.0   0.0  0.0           61.343296                97.0           61.343296               53.0
65    66.0  33.0  60.0   0.0  1.0           61.627916                96.0           61.627916               54.0
112  113.0  37.0  58.0   0.0  1.0           61.854668                98.0           61.854668               52.0
105  106.0  30.0  62.0   3.0  1.0           61.854668                92.0           61.854668               56.0
```

c. Hasil terbaik data train.xlsx menggunakan Minkowski Distance

```
==== Minkowski Distance ====
        id    x1    x2    x3    y  Euclidean Distance  Manhattan Distance  Minkowski Distance  Supremum Distance
5      6.0  33.0  58.0  10.0  1.0           59.548300                90.0           59.548300               52.0
82    83.0  31.0  59.0   2.0  1.0           59.657355                91.0           59.657355               53.0
284  285.0  34.0  59.0   0.0  0.0           61.343296                97.0           61.343296               53.0
65    66.0  33.0  60.0   0.0  1.0           61.627916                96.0           61.627916               54.0
112  113.0  37.0  58.0   0.0  1.0           61.854668                98.0           61.854668               52.0
105  106.0  30.0  62.0   3.0  1.0           61.854668                92.0           61.854668               56.0
```

d. Hasil terbaik data train.xlsx menggunakan Supremum Distance

```
==== Supremum Distance ====
        id    x1    x2   x3    y  Euclidean Distance  Manhattan Distance  Minkowski Distance  Supremum Distance
170  171.0  50.0  58.0  1.0  1.0           69.382995                110.0           69.382995               52.0
60    61.0  41.0  58.0  0.0  1.0           64.015623                102.0           64.015623               52.0
26    27.0  53.0  58.0  4.0  0.0           71.281134                111.0           71.281134               52.0
144  145.0  53.0  58.0  1.0  1.0           71.365258                113.0           71.365258               52.0
138  139.0  42.0  58.0  0.0  1.0           64.583280                103.0           64.583280               52.0
136  137.0  46.0  58.0  2.0  0.0           66.887966                106.0           66.887966               52.0
```

BAB IV

4.1. **K**esimpulan

Dari penjelasan dan hasil output program dapat disimpulkan bahwa metode KNN cukup efisien untuk menyelesaikan persoalan problem klasifikasi biner (*binary classification*).

4.2. **Peran Anggota**

Source code program : Citakamalia, Indah Putri Maharani, Iqlima Putri Hawa

Laporan : Citakamalia, Indah Putri Maharani, Iqlima Putri Hawa

4.3. Referensi

18.04.103_jurnal_eproc.pdf. e-Proceeding of Engineering : Vol.5, No.1 (Maret 2018). Klasifikasi Dokumen Menggunakan Metode k-Nearest Neighbor (KNN) dengan Information Gain. Diakses pada 17 Juni 2022, dari file:///C:/Users/user/Downloads/18.04.103_jurnal_eproc.pdf

ilmudatapy.com(). Algoritma K-Nearest Neighbor (KNN) untuk Klasifikasi. Diakses pada 17 Juni 2022, dari https://ilmudatapy.com/algoritma-k-nearest-neighbor-knn-untuk-klasifikasi/

Siti Saadah (). 11-Nearest Neighbor. Diakses pada 16 Juni 2022, dari 12 - Nearest Neighbor (1).pptx

4.4 Video Presentasi

https://drive.google.com/file/d/1Ffd5dCBtKagQuHzxU3N1T9WcddiUm7It/view?usp=sharing