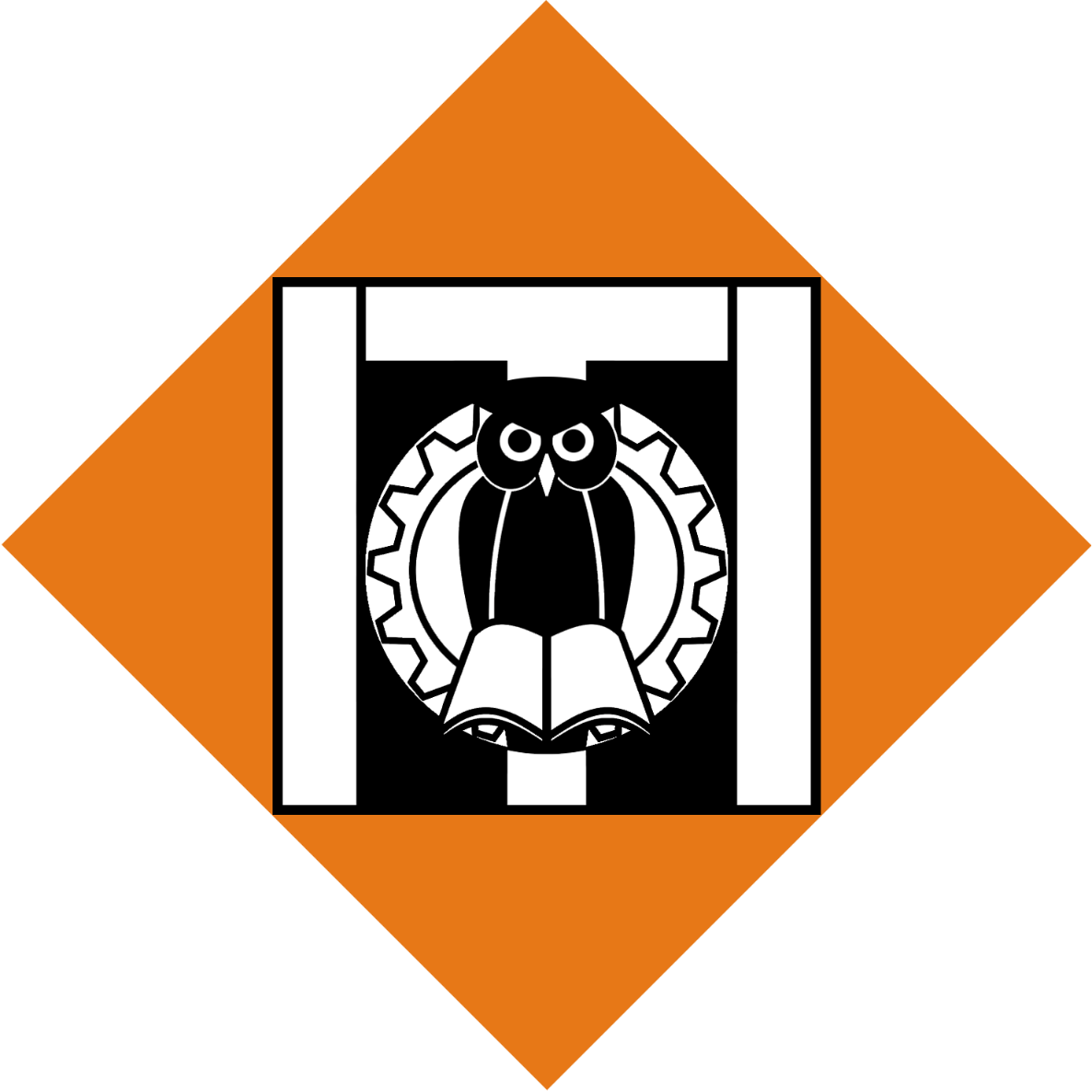
LAPORAN KECERDASAN BUATAN

“KNN WITH PYTHON”

Ditulis oleh:

IKHWAN EL FARIS (1152200009)

JONATHAN NATANNAEL ZEFANYA (1152200024)

DAFFA DANINDRA (1152200028)

DETHALISA AURA KIRANA (1152200037)

# **BAB I**

**PENDAHULUAN**

## **1.1 Masalah**

1. Bagaimana cara mengimplementasikan algoritma K-Nearest Neighbors (KNN) secara manual tanpa menggunakan library eksternal **KNearestNeighborClassifier** dalam Python?
2. Seberapa akurat dan efisien implementasi KNN manual dalam mengklasifikasikan data penyakit liver dibandingkan dengan menggunakan library yang sudah ada?
3. Bagaimana menyusun kode Python untuk KNN yang mudah dipahami, dioptimalkan, dan tetap memberikan hasil yang akurat dalam diagnosis penyakit liver?
4. Bagaimana cara mengolah dan menyiapkan dataset penyakit liver agar dapat digunakan dalam algoritma KNN manual?

## **1.2 Tujuan**

* 1. Mengimplementasikan algoritma K-Nearest Neighbors (KNN) secara manual dalam Python tanpa menggunakan library eksternal **KNearestNeighborClassifier**.
  2. Membandingkan akurasi dan efisiensi antara implementasi KNN manual dan implementasi KNN menggunakan library **KNearestNeighborClassifier**.
  3. Membuat kode Python untuk KNN agar memberikan hasil yang akurat dalam diagnosis penyakit liver.
  4. Mengolah dan menyiapkan dataset penyakit liver agar dapat digunakan dalam algoritma KNN manual dengan benar.

# **BAB II**

**DASAR TEORI**

## **2.1 K-Nearest Neighbors (KNN)**

Algoritma K-Nearest Neighbors (KNN) adalah metode klasifikasi yang sederhana dan efektif dalam machine learning. KNN bekerja dengan mencari K tetangga terdekat dari data baru berdasarkan jarak (misalnya jarak Euclidean) antara data baru dan data dalam training set. Kelas dari data baru ditentukan oleh mayoritas kelas dari K tetangga terdekat tersebut. Implementasi manual KNN dalam Python melibatkan beberapa langkah penting: pengolahan dan pembersihan data, perhitungan jarak, pemilihan tetangga terdekat, dan klasifikasi berdasarkan kelas mayoritas.

Kelebihan dari KNN termasuk kesederhanaan dan kemudahan implementasi, sementara kekurangannya adalah ketidakefisienan pada dataset besar dan sensitif terhadap skala data serta outliers. Dalam proyek ini, algoritma KNN akan diterapkan untuk mengklasifikasikan data penyakit liver, yang melibatkan atribut kesehatan hati pasien, guna memahami cara kerja algoritma ini dan mengoptimalkannya untuk hasil klasifikasi yang akurat.

# **BAB III**

**DATA DAN METODOLOGI**

## **3.1 Data**

### 3.1.1 Sumber Data

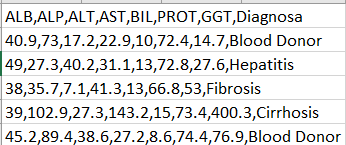
Kumpulan data berisi nilai laboratorium donor darah dan pasien Hepatitis C serta nilai demografi seperti usia, Yang dibuat oleh Ralf Lichtinghagen, Frank Klawonn, Georg Hoffman.

Berikut link dari data yang digunakan: https://archive.ics.uci.edu/dataset/571/hcv+data

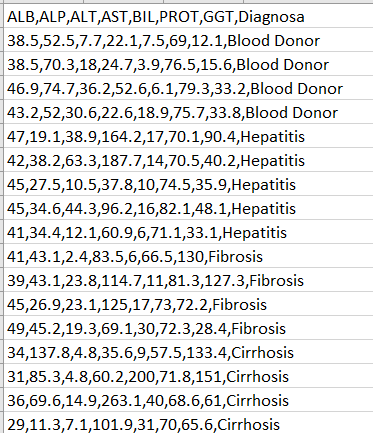
### 3.1.2 Pre-processing Data

**Gambar 3.1 Pre-Processing Data**

### 3.1.3 Data Testing dan Data Training

- Data Testing:

**Gambar 3.2 Data Testing**

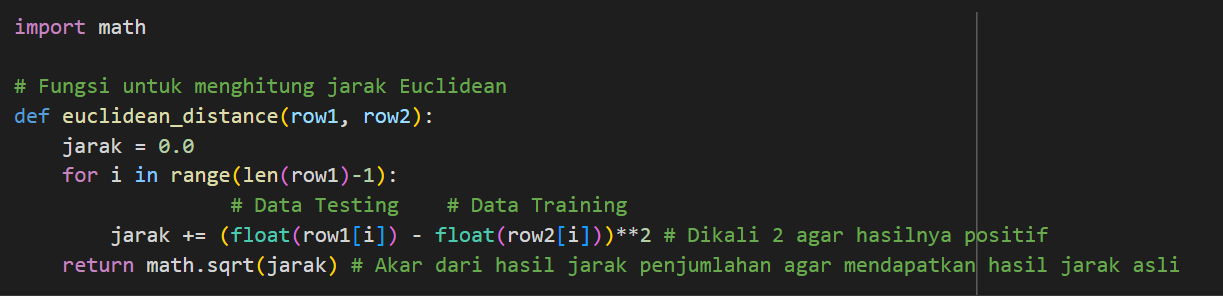
* Data Training:

**Gambar 3.3 Data Training**

## **3.2 Metodologi**

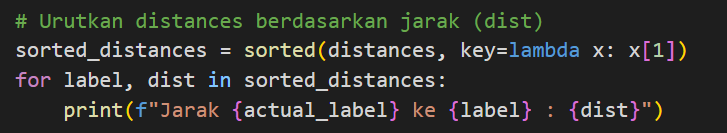
### 3.2.1 Perhitungan Jarak

Kode ini berfungsi untuk menghitung jarak Euclidean antara dua baris data `row1` dan `row2`. Dimulai dengan mengimpor modul `math` untuk menggunakan fungsi `sqrt`, kemudian mendefinisikan fungsi `euclidean\_distance` yang menerima dua argumen. Variabel `jarak` diinisialisasi dengan nilai 0.0 untuk menyimpan hasil penjumlahan kuadrat selisih setiap elemen dari `row1` dan `row2`, kecuali elemen terakhir yang diasumsikan sebagai label atau output. Loop `for` digunakan untuk iterasi melalui elemen-elemen tersebut, dimana selisih antara elemen ke-i dari `row1` dan `row2` dihitung, dikuadratkan, dan ditambahkan ke variabel `jarak`. Setelah loop selesai, akar kuadrat dari `jarak` dihitung menggunakan `math.sqrt(jarak)` untuk mendapatkan jarak Euclidean yang sebenarnya, dan hasil ini dikembalikan sebagai output dari fungsi.



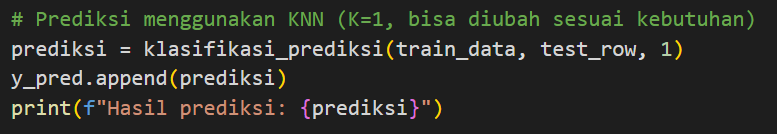
**Gambar 3.4 Perhitungan Jarak**

### 3.2.2 Sorting Data

Kode ini mengurutkan daftar jarak (`distances`) berdasarkan nilai jarak (elemen kedua setiap pasangan) menggunakan `sorted()` dengan parameter `key=lambda x: x[1]`, kemudian iterasi melalui daftar jarak yang sudah diurutkan dan mencetak jarak antara `actual\_label` dan `label` untuk setiap pasangan jarak yang diurutkan.

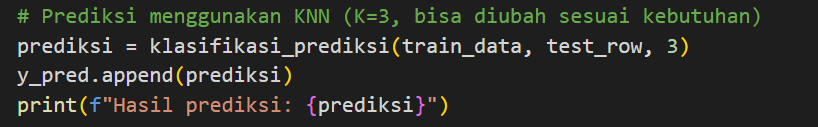
**Gambar 3.5 Sorting Data**

### 3.2.3 Perhitungan 1NN

Kode ini menggunakan algoritma K-Nearest Neighbors (KNN) untuk membuat prediksi dengan parameter K=1, yang berarti hanya satu tetangga terdekat yang dipertimbangkan. Fungsi `klasifikasi\_prediksi` dipanggil dengan argumen `train\_data`, `test\_row`, dan `1` untuk menghasilkan prediksi berdasarkan data pelatihan dan baris uji. Hasil prediksi kemudian ditambahkan ke dalam daftar `y\_pred` dan dicetak dengan menggunakan `print`.

**Gambar 3.6 Perhitungan 1NN**

### 3.2.4 Perhitungan 3NN

Kode tersebut menggunakan algoritma K-Nearest Neighbors (KNN) untuk membuat prediksi dengan parameter K=3, yang berarti tiga tetangga terdekat yang dipertimbangkan. Fungsi `klasifikasi\_prediksi` dipanggil dengan argumen `train\_data`, `test\_row`, dan `3` untuk menghasilkan prediksi berdasarkan data pelatihan dan baris uji. Hasil prediksi kemudian ditambahkan ke dalam daftar `y\_pred` dan dicetak dengan menggunakan `print`.

**Gambar 3.7 Perhitungan 3NN**

### 3.2.5 Confusion Matrix dan Accuracy

### 

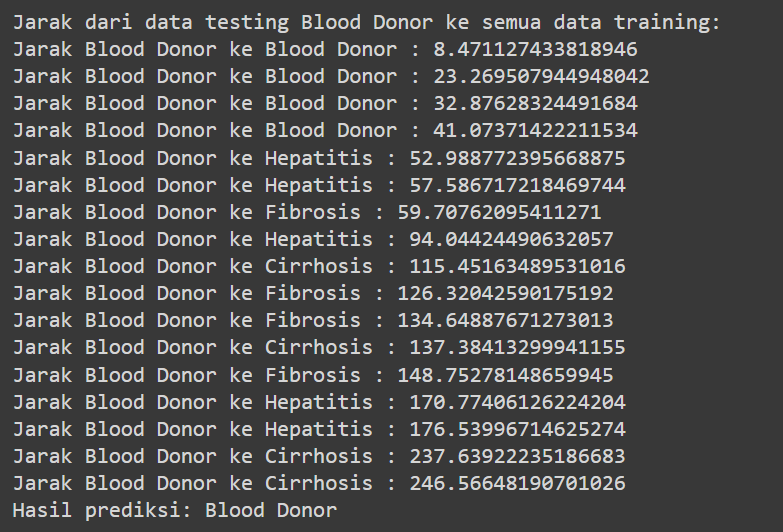
### Gambar 3.8 Perhitungan CM dan AC

# **BAB IV**

**HASIL DAN ANALISA**

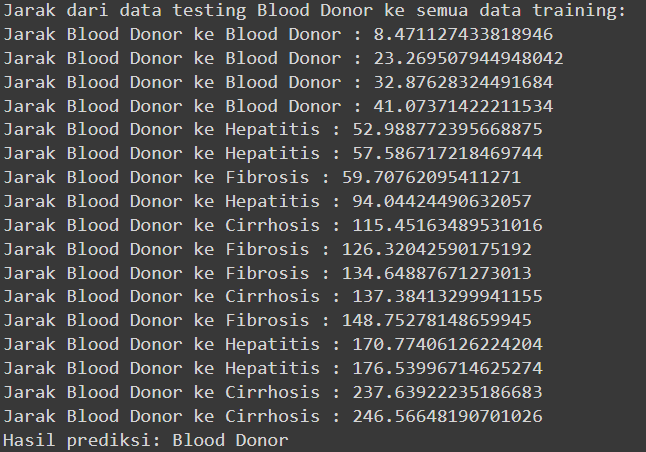
## **4.1 Hasil**

### 4.1.1 Hasil Perhitungan Jarak dan Sorting Data



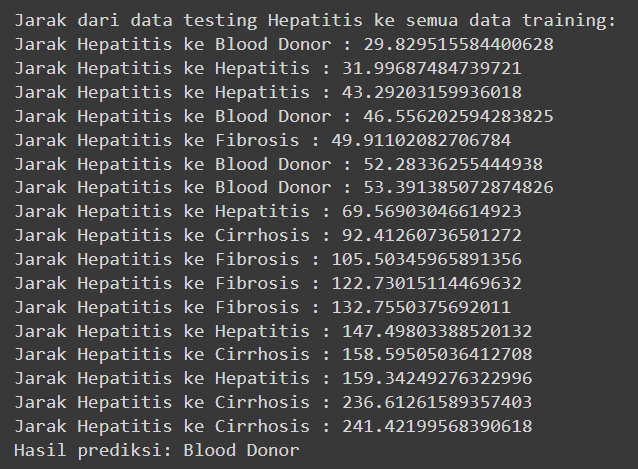
**Gambar 4.1 Hasil Perhitungan Jarak dan Sorting Data**

### 4.1.2 Hasil Perhitungan 1NN

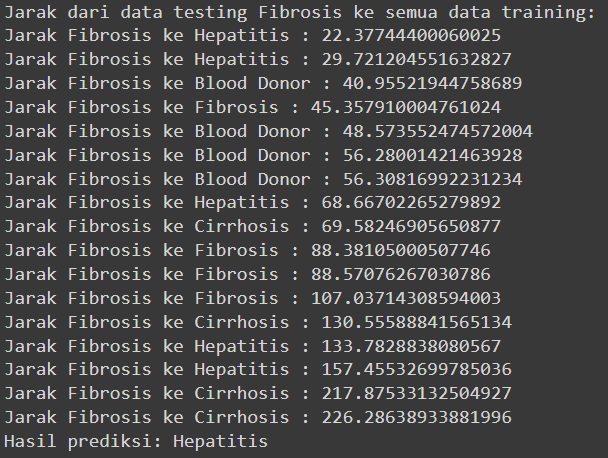
- Hasil 1 Perhitungan 1NN

**Gambar 4.2 Hasil 1 Perhitungan 1NN**

* Hasil 2 Perhitungan 1NN

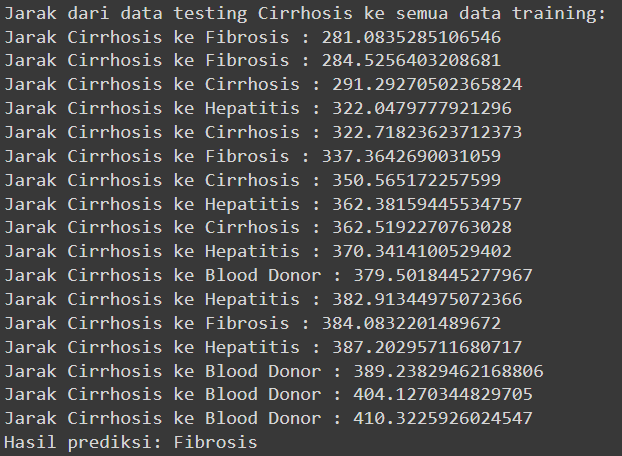


**Gambar 4.3 Hasil 2 Perhitungan 1NN**

* Hasil 3 Perhitungan 1NN

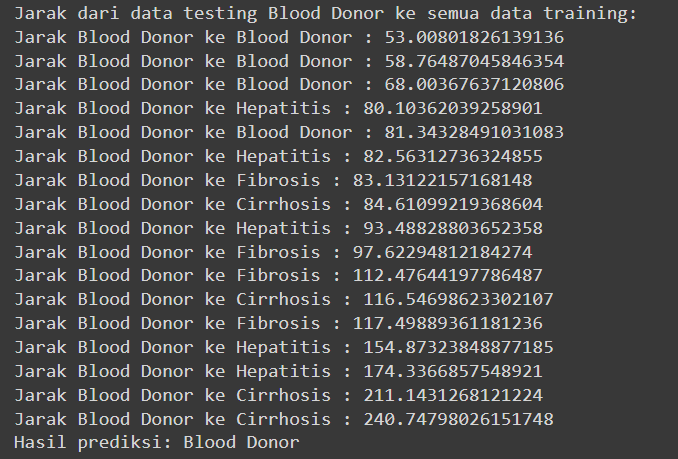
**Gambar 4.4 Hasil 3 Perhitungan 1NN**

* Hasil 4 Perhitungan 1NN



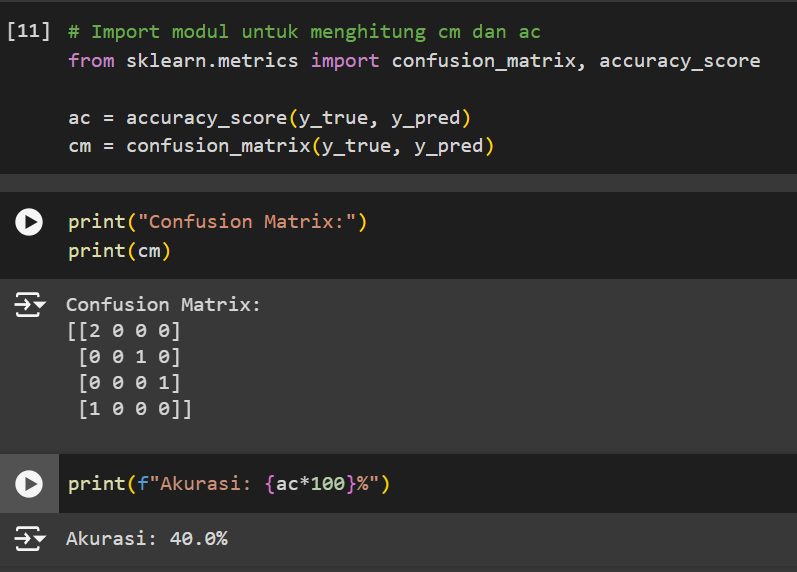
**Gambar 4.5 Hasil 4 Perhitungan 1NN**

* Hasil 5 Perhitungan 1NN



**Gambar 4.6 Hasil 5 Perhitungan 1NN**

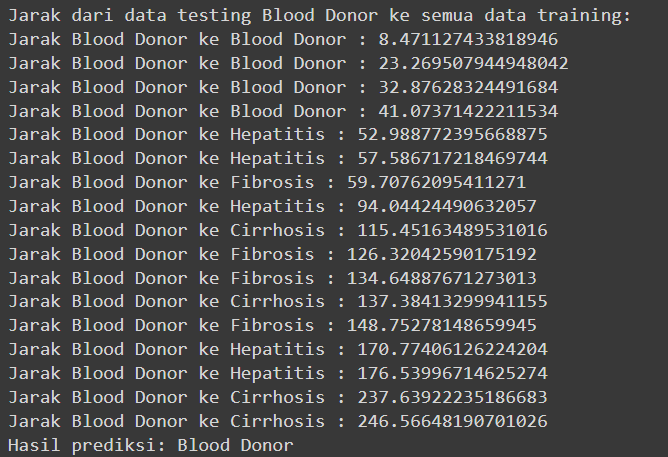
* Hasil *Cofusion Matrix* dan *Accuracy*



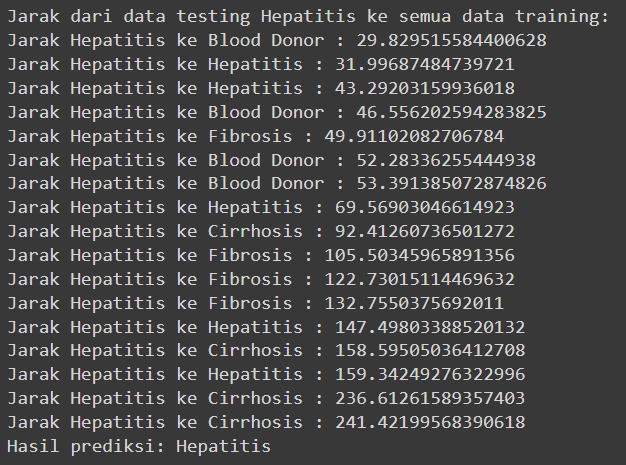
**Gambar 4.7 Hasil perhitungan AC dan CM**

### 4.1.3 Hasil Perhitungan 3NN

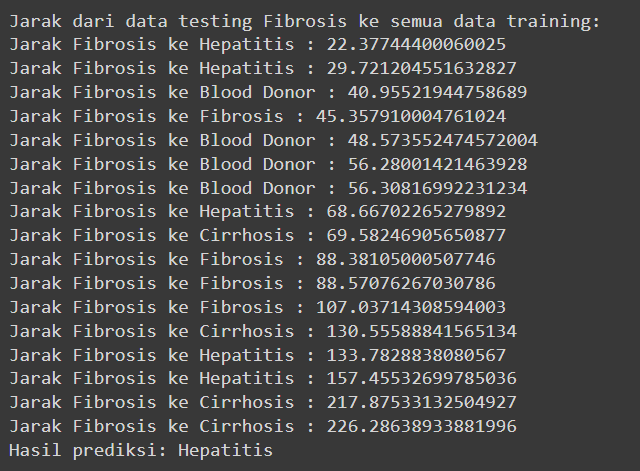
Pada perhitungan 3NN prediksi terpilih melalui hasil voting dari 3 data teratas atau terdekat.

- Hasil 1 Perhitungan 3NN

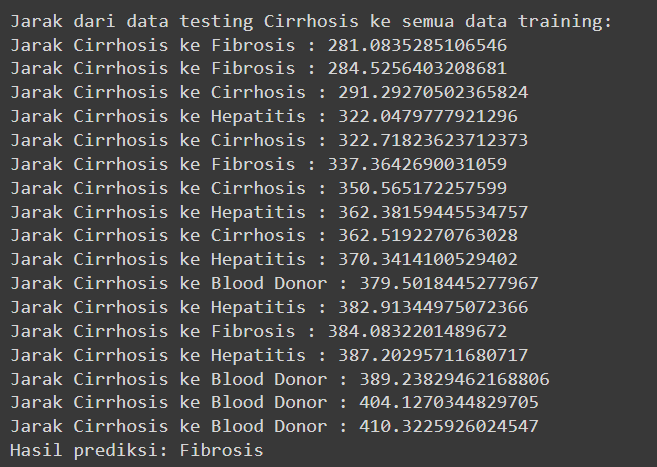
**Gambar 4.7 Hasil 1 Perhitungan 3NN**

* Hasil 2 Perhitungan 3NN

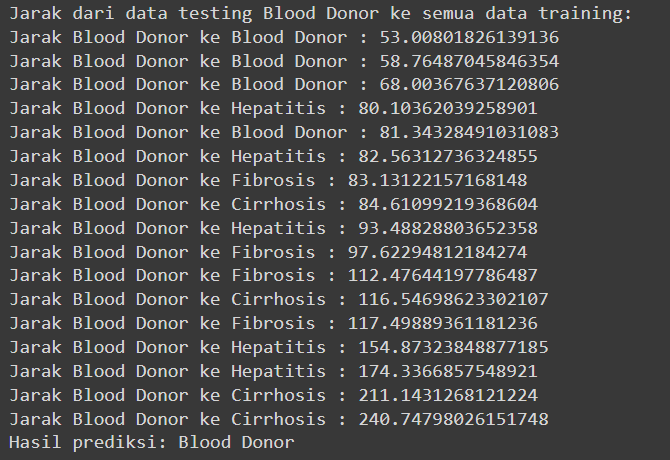
**Gambar 4.8 Hasil 2 Perhitungan 3NN**

* Hasil 3 Perhitungan 3NN

**Gambar 4.9 Hasil 3 Perhitungan 3NN**

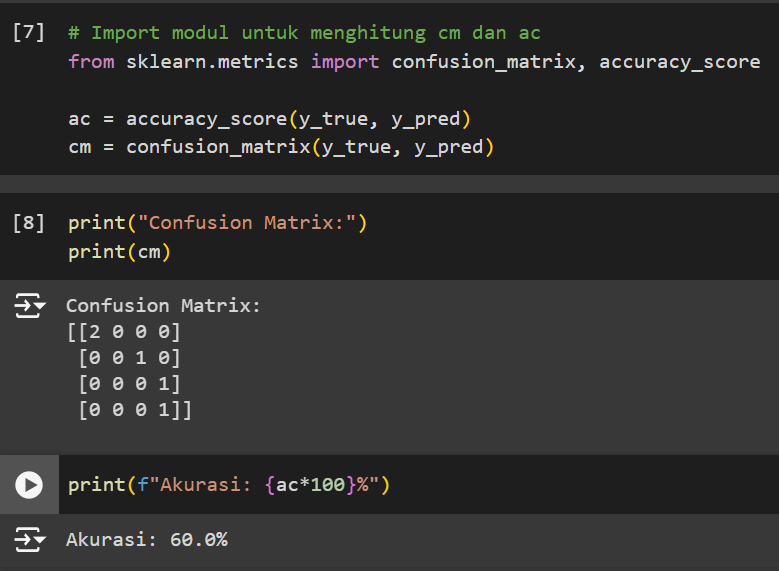
* Hasil 4 Perhitungan 3NN

**Gambar 4.10 Hasil 4 Perhitungan 3NN**

* Hasil 5 Perhitungan 3NN

**Gambar 4.11 Hasil 5 Perhitungan 3NN**

* Hasil *Cofusion Matrix* dan *Accuracy* perhitungan 3NN



**Gambar 4.12 Hasil CM dan AC**

# **BAB V**

**KESIMPULAN**

## **5.1 Kesimpulan**

Dalam proyek ini, tujuan utama kami adalah mengimplementasikan algoritma K-Nearest Neighbors (KNN) secara manual menggunakan Python tanpa library eksternal. Tujuan ini berhasil dicapai dengan merancang dan menguji algoritma KNN untuk klasifikasi data penyakit liver. Implementasi dilakukan dengan menghitung jarak Euclidean, mengurutkan jarak, dan melakukan prediksi berdasarkan tetangga terdekat. Hasil dari perhitungan 1NN dan 3NN menunjukkan bahwa algoritma KNN manual mampu memberikan hasil yang akurat dalam diagnosis penyakit liver, sesuai dengan hasil yang diharapkan. Analisa dari perhitungan dan prediksi menunjukkan bahwa algoritma ini dapat bekerja dengan baik dalam mengklasifikasikan data berdasarkan atribut kesehatan hati pasien.

# **Referensi**

* https://www.kaggle.com/code/mikhailsukhanov55/knn-without-libraries
* https://stackoverflow.com/questions/67744245/how-to-create-a-knn-function-without-a-library
* https://medium.com/analytics-vidhya/implementing-k-nearest-neighbours-knn-without-using-scikit-learn-3905b4decc3c
* https://www.kaggle.com/code/lohitha17/knn-without-scikit-learn

# **SIAPA MENGERJAKAN APA:**

1. Jonathan Natannael Zefanya (1152200024) – Mengerjakan Program
2. Ikhwan El Faris (1152200009) – Mencari Dan Memberikan Algoritma Program
3. Daffa Danindra (1152200028) – Mengambil Dan Membuat Seluruh Data Yang Dibutuhkan, Mengerjakan Laporan
4. Dethalisa Aura Kirana (1152200037) – Mengerjakan Laporan