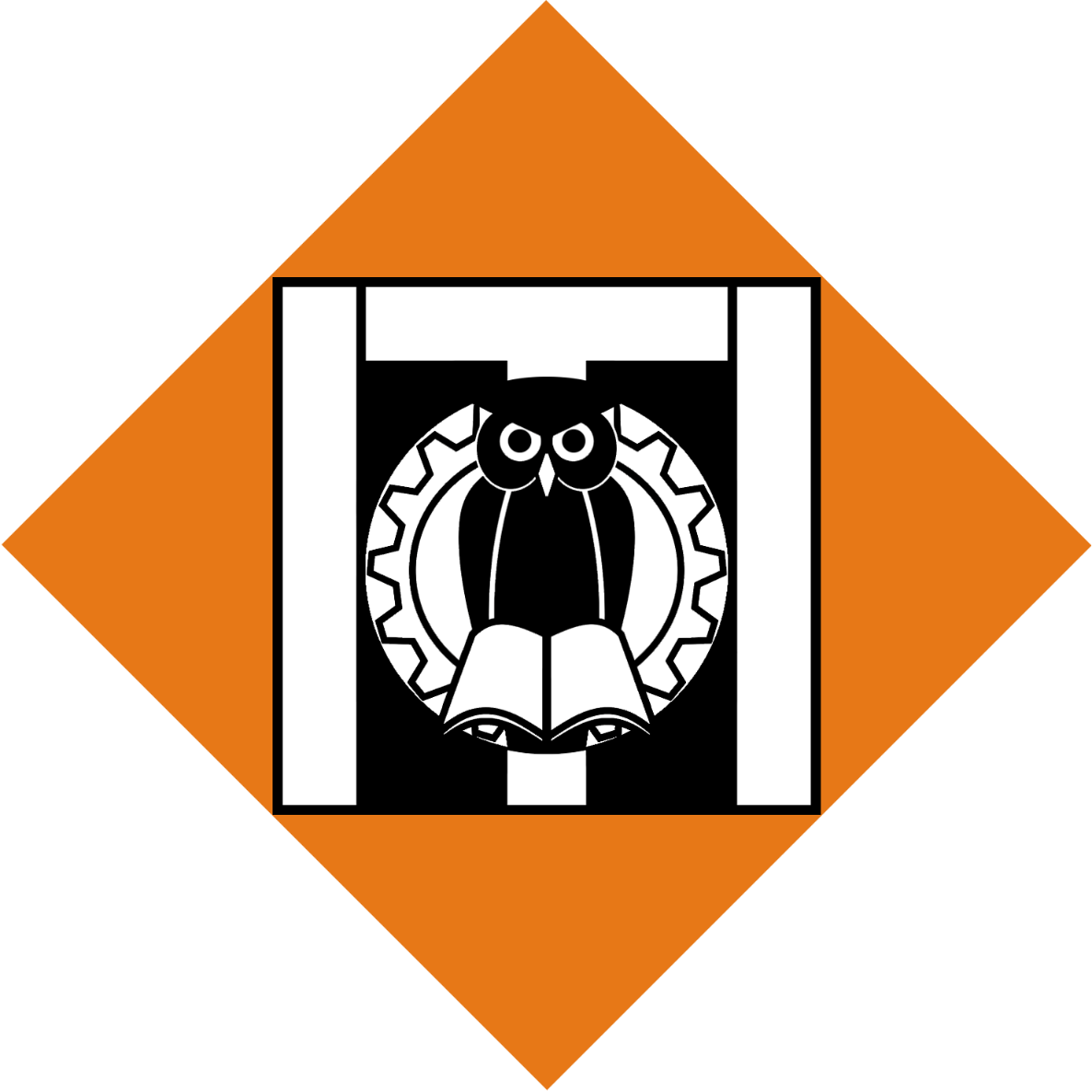
LAPORAN KECERDASAN BUATAN

“NAÏVE BAYES 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 Naïve Bayes secara manual tanpa menggunakan library eksternal dalam Python?
2. Seberapa akurat dan efisien implementasi Naïve Bayes manual dalam mengklasifikasikan dataset Pendapatan Orang Dewasa dibandingkan dengan menggunakan library yang sudah ada?
3. Bagaimana menyusun kode Python untuk Naïve Bayes yang mudah dipahami, dioptimalkan, dan tetap memberikan hasil yang akurat dalam menghitung probabilitas dataset Pendapatan Orang Dewasa?
4. Bagaimana cara mengolah dan menyiapkan dataset Pendapatan Orang Dewasa agar dapat digunakan dalam Naïve Bayes?

## **1.2 Tujuan**

* 1. Mengimplementasikan Naïve Bayes secara manual dalam Python tanpa menggunakan library eksternal.
  2. Membandingkan akurasi dan efisiensi antara implementasi Naïve Bayes manual dan implementasi Naïve Bayes menggunakan library **eksternal**.
  3. Membuat kode Python untuk Naïve Bayes agar memberikan hasil yang akurat dalam menghitung probabilitas dataset Pendapatan Orang Dewasa.
  4. Mengolah dan menyiapkan dataset Pendapatan Orang Dewasa agar dapat digunakan dalam perhitungan Naïve Bayes manual dengan benar.

# **BAB II**

**DASAR TEORI**

## **2.1 Naïve Bayes**

Naïve Bayes adalah salah satu algoritma klasifikasi yang menggunakan prinsip probabilitas. Algoritma ini didasarkan pada Teorema Bayes yang mengkalkulasi probabilitas suatu kelas berdasarkan fitur-fitur yang ada, dengan asumsi bahwa setiap fitur bersifat independen satu sama lain. Asumsi independensi inilah yang membuat algoritma ini disebut "naïve". Probabilitas adalah konsep dasar dalam statistika yang mengukur kemungkinan terjadinya suatu kejadian. Dalam konteks machine learning, probabilitas sering digunakan untuk memperkirakan kelas atau label suatu data berdasarkan fitur-fitur yang dimilikinya.

Naïve Bayes bekerja dengan menghitung probabilitas prior dari setiap kelas, serta probabilitas likelihood dari fitur-fitur yang diberikan kelas tertentu. Probabilitas posterior, yang merupakan probabilitas suatu kelas diberikan fitur-fitur tertentu, kemudian dihitung dan digunakan untuk memprediksi kelas dengan probabilitas tertinggi. Meskipun asumsi independensi antara fitur seringkali tidak realistis dalam kasus dunia nyata, Naïve Bayes tetap menjadi algoritma yang efektif dan efisien, terutama untuk dataset besar dan situasi dimana kecepatan prediksi menjadi faktor penting.

1. Keuntungan
   1. Mudah diimplementasikan.
   2. Akurasi bagus dalam banyak kasus.
2. Kerugian:
   1. Asumsi bahwa : Masing-masing ciri tidak saling tergantung (independent) satu sama lain, sehingga tidak cocok untuk kasus kasus tertentu yang membutuhkan kebergantungan antar ciri dan kelas.

Diberikan data testing X, maka probabilitas (posteriori probability) dari sebuah hipotesa H, P(H|X) adalah:

P ( H | X ) = P ( X | H ) P ( H ) / P ( X ) = P ( X | H ) × P ( H ) / P ( X )

Secara informal dapat ditulis:

posteriori = likelihood x prior/evidence

X adalah data (“evidence”) yang klasifikasinya tidak diketahui.

H adalah hipotesa bahwa X masuk klasifikasi C.

Klasifikasi adalah menentukan P(H|X) atau yang disebut dengan posteriori probability yaitu probabilitas dari sebuah hipotesa diberikan data X.

P(H) (prior probability), adalah probabilitas awal.

P(X): Probabilitas dari data yang akan dicari klasifikasinya.

P(X|H) (likelihood), the probabilitas data X jika diberikan given that the hipotesa H.

# **BAB III**

**DATA DAN METODOLOGI**

## **3.1 Data**

### 3.1.1 Sumber Data

Kumpulan data berisi Pendapatan tahunan seseorang dihasilkan dari berbagai faktor. Secara intuitif dipengaruhi oleh tingkat pendidikan individu, usia, jenis kelamin, pekerjaan, dan lain-lain, Yang dibuat oleh “Tidak Dikenal”.

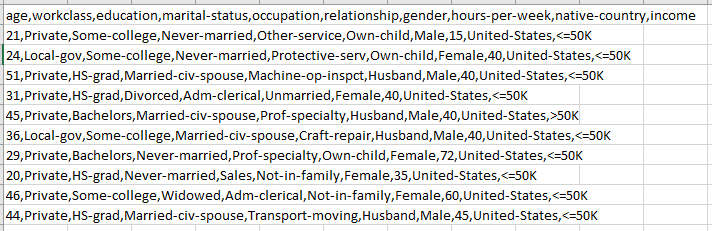
Berikut link dari data yang digunakan:

https://www.kaggle.com/datasets/wenruliu/adult-income-dataset

### 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**

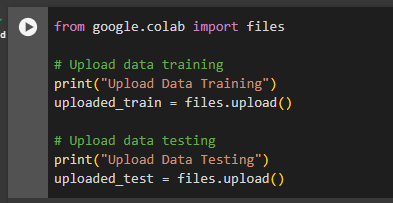
* Data Training:

**Gambar 3.3 Data Training**

## **3.2 Metodologi**

### 3.2.1 Mengupload Data

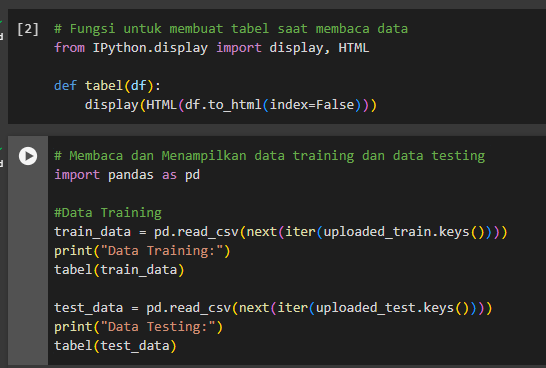
Kode tersebut merupakan bagian dari program Python yang digunakan untuk mengunggah file data ke lingkungan Google Colab. Pertama, modul `files` dari `google.colab` diimpor untuk memungkinkan interaksi dengan sistem file Colab. Setelah itu, kode mencetak pesan "Upload Data Training" untuk memberi tahu pengguna bahwa mereka perlu mengunggah file data pelatihan. Fungsi `files.upload()` kemudian dipanggil untuk membuka dialog pengunggahan file, memungkinkan pengguna untuk memilih dan mengunggah file data pelatihan mereka. Proses ini diulangi untuk file data pengujian, dengan mencetak pesan "Upload Data Testing" dan memanggil `files.upload()` lagi untuk mengunggah file data pengujian. File yang diunggah akan disimpan dalam variabel `uploaded\_train` dan `uploaded\_test` sebagai dictionary yang berisi nama file dan konten file tersebut.



**Gambar 3.4 Mengupload Data**

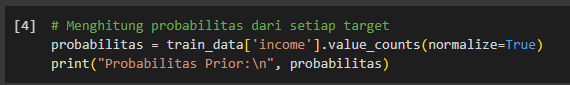
### 3.2.2 Membuat table dan membaca,menampilkan data.

Kode tersebut merupakan bagian dari program Python yang digunakan untuk membaca dan menampilkan data dalam bentuk tabel di lingkungan Jupyter Notebook atau Google Colab. Pertama, modul `display` dan `HTML` dari `IPython.display` diimpor untuk menampilkan tabel HTML. Fungsi `tabel` didefinisikan untuk mengkonversi DataFrame pandas menjadi HTML dan menampilkannya tanpa indeks. Selanjutnya, modul `pandas` diimpor sebagai `pd` untuk memudahkan manipulasi data. Data pelatihan dibaca dari file CSV yang telah diunggah sebelumnya menggunakan `pd.read\_csv` dan disimpan dalam `train\_data`. Data ini kemudian ditampilkan dengan mencetak "Data Training:" dan memanggil fungsi `tabel` untuk menampilkan DataFrame. Proses serupa dilakukan untuk data pengujian, yang dibaca dan ditampilkan dengan cara yang sama.



**Gambar 3.5 membuat table dan membaca, menampilkan data.**

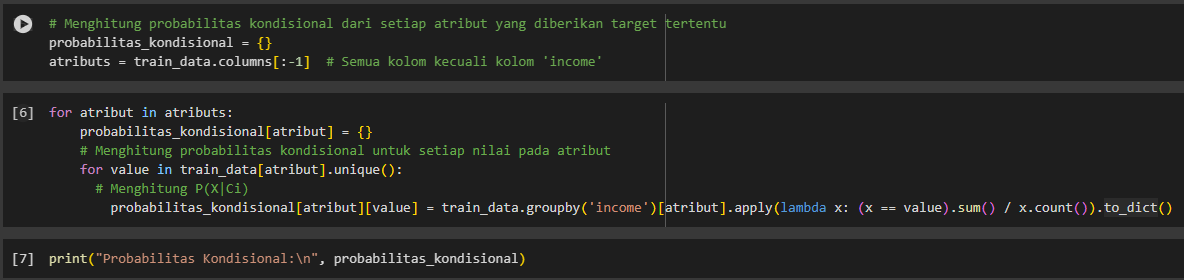
### 3.2.3 Menghitung probabilitas dari setiap target

 Kode tersebut menghitung probabilitas prior dari setiap kategori dalam kolom `income` pada data pelatihan `train\_data`. Dengan menggunakan metode `value\_counts` dari pandas dan mengatur parameter `normalize=True`, kode ini mengkalkulasi proporsi atau probabilitas dari setiap nilai unik dalam kolom `income`. Hasilnya adalah seri pandas yang menyimpan probabilitas masing-masing kategori. Kemudian, hasil ini dicetak dengan judul "Probabilitas Prior:" diikuti oleh nilai-nilai probabilitas yang telah dihitung, memberikan gambaran tentang distribusi awal kelas-kelas dalam data pelatihan.

**Gambar 3.6 Menghitung probabilitas dari setiap target.**

### 3.2.4 Perhitungan Probabilitas Kondisional

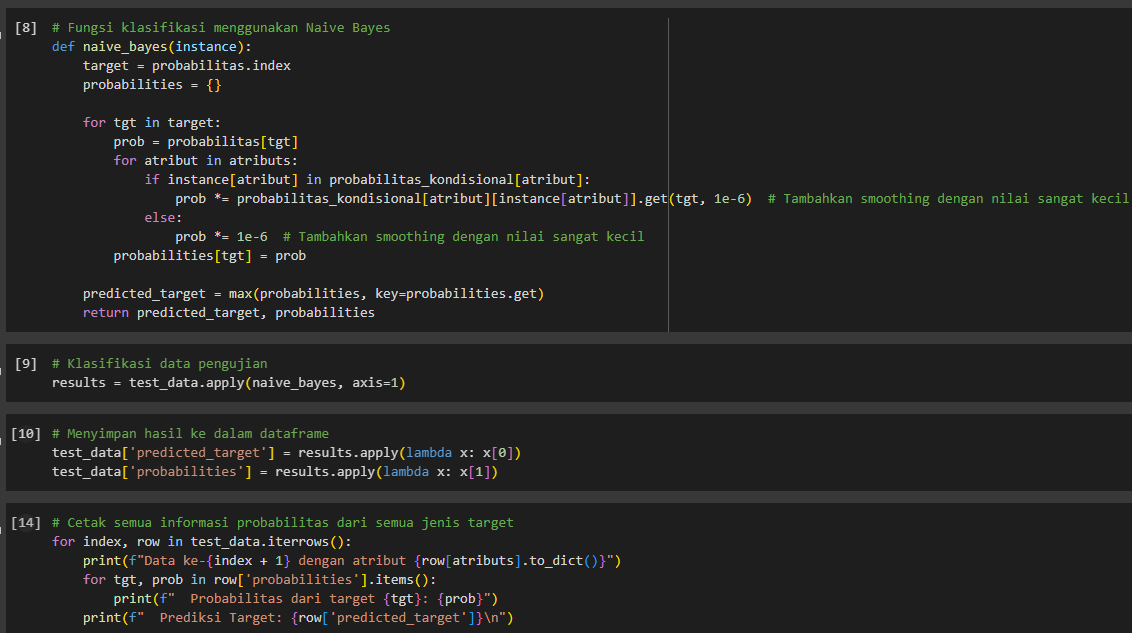
Kode tersebut menghitung probabilitas kondisional dari setiap nilai atribut yang diberikan target tertentu dalam data pelatihan `train\_data`. Pertama, sebuah dictionary kosong `probabilitas\_kondisional` diinisialisasi untuk menyimpan hasil perhitungan. Kode ini kemudian iterasi melalui semua kolom atribut kecuali kolom 'income'. Untuk setiap atribut, sebuah sub-dictionary dibuat untuk menyimpan probabilitas kondisional dari setiap nilai dalam atribut tersebut. Kemudian, untuk setiap nilai unik dalam atribut, probabilitas kondisional dihitung dengan membandingkan jumlah kemunculan nilai tersebut terhadap total kemunculan dalam kelompok target `income`. Hasilnya disimpan dalam bentuk dictionary yang mencakup semua kombinasi nilai atribut dan kategori target. Terakhir, hasil probabilitas kondisional tersebut dicetak untuk ditampilkan.



**Gambar 3.7 Perhitungan Probabilitas Kondisional**

### 3.2.5 Klasifikasi Data dan Prediksi Target

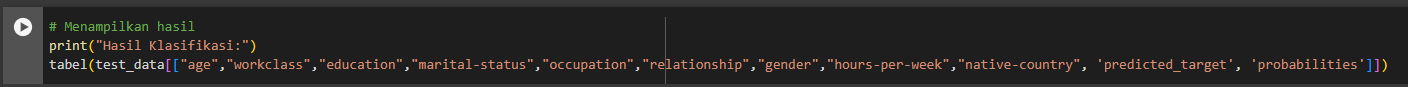
Kode tersebut melakukan klasifikasi menggunakan algoritma Naïve Bayes. Pertama, fungsi `naive\_bayes` didefinisikan untuk mengklasifikasikan sebuah instance (baris data). Fungsi ini menghitung probabilitas dari setiap target (kelas) dengan mengalikan probabilitas prior target dengan probabilitas kondisional dari setiap atribut yang diberikan target. Untuk menghindari probabilitas nol, digunakan smoothing dengan nilai sangat kecil (1e-6). Setelah menghitung probabilitas untuk setiap target, target dengan probabilitas tertinggi dipilih sebagai prediksi. Selanjutnya, data pengujian `test\_data` diklasifikasikan dengan menerapkan fungsi `naive\_bayes` pada setiap baris data. Hasil prediksi dan probabilitas disimpan dalam kolom baru di `test\_data`. Terakhir, hasil klasifikasi dan probabilitas untuk setiap instance data pengujian dicetak, menunjukkan atribut-atribut instance, probabilitas setiap target, dan prediksi akhir.



**Gambar 3.8 Klasifikasi Data dan Prediksi Target.**

### 3.2.6 Hasil Klasifikasi

Kode tersebut digunakan untuk menampilkan hasil klasifikasi dalam bentuk tabel dengan kolom-kolom tertentu. Baris `print("Hasil Klasifikasi:")` digunakan untuk mencetak teks "Hasil Klasifikasi" sebagai judul atau pengantar hasil yang akan ditampilkan. Kemudian, fungsi `tabel()` digunakan untuk menampilkan data yang ada pada variabel `test\_data` dengan hanya menampilkan kolom-kolom yang terpilih, yaitu "age" (usia), "workclass" (kelas pekerjaan), "education" (pendidikan), "marital-status" (status perkawinan), "occupation" (pekerjaan), "relationship" (hubungan), "gender" (jenis kelamin), "hours-per-week" (jam kerja per minggu), "native-country" (negara asal), "predicted\_target" (target yang diprediksi), dan "probabilities" (probabilitas). Kolom-kolom ini mencakup informasi demografis, hasil prediksi klasifikasi, dan probabilitas terkait dari model yang digunakan.



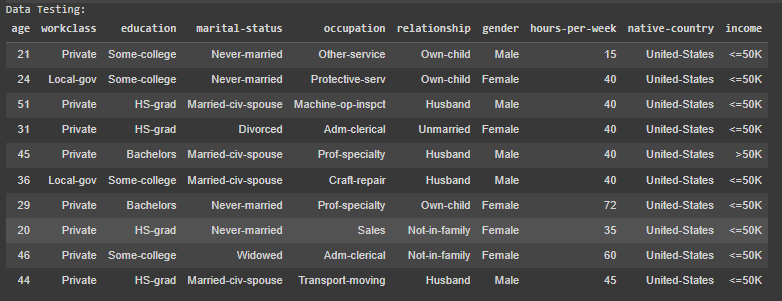
**Gambar 3.9 Hasil Klasifikasi.**

# **BAB IV**

**HASIL DAN ANALISA**

## **4.1 Hasil**

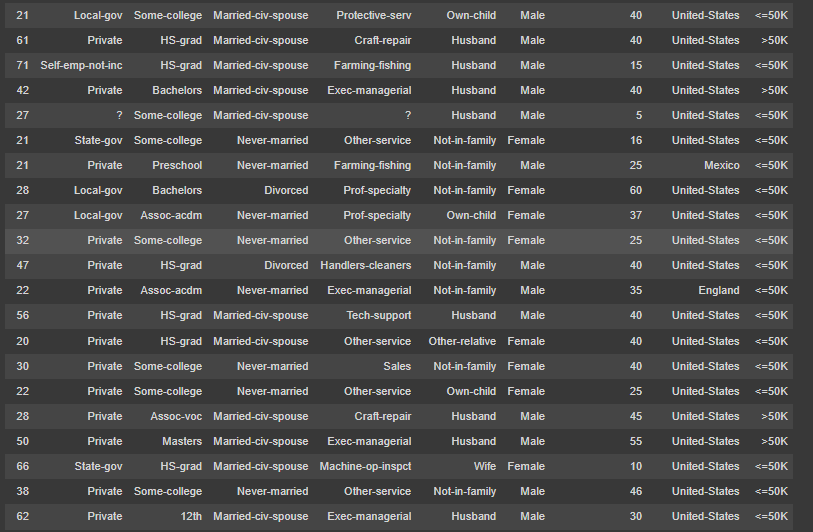
### 4.1.1 Hasil Menampilkan Data Testing



**Gambar 4.1 Hasil Menampilkan Data Testing**

### 4.1.2 Hasil Menampilkan Data Training

**Gambar 4.2 Hasil Menampilkan Data Training.**



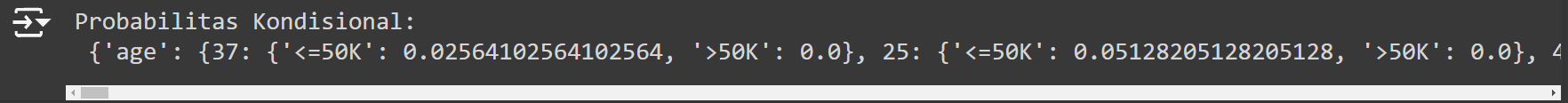
**Gambar 4.3 Hasil Menampilkan Data Training.**

### 4.1.3 Hasil Perhitungan Probabilitas Prior

**Gambar 4.4 Hasil Perhitungan Probabilitas Prior**

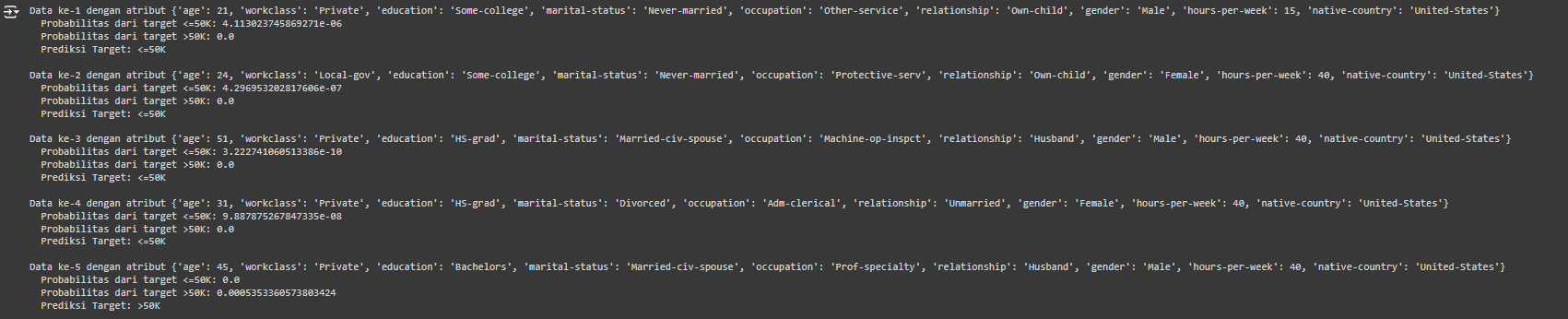
### 4.1.4 Hasil Perhitungan Probabilitas Kondisional

Pada perhitungan ini menampilkan hasil yang cukup Panjang.

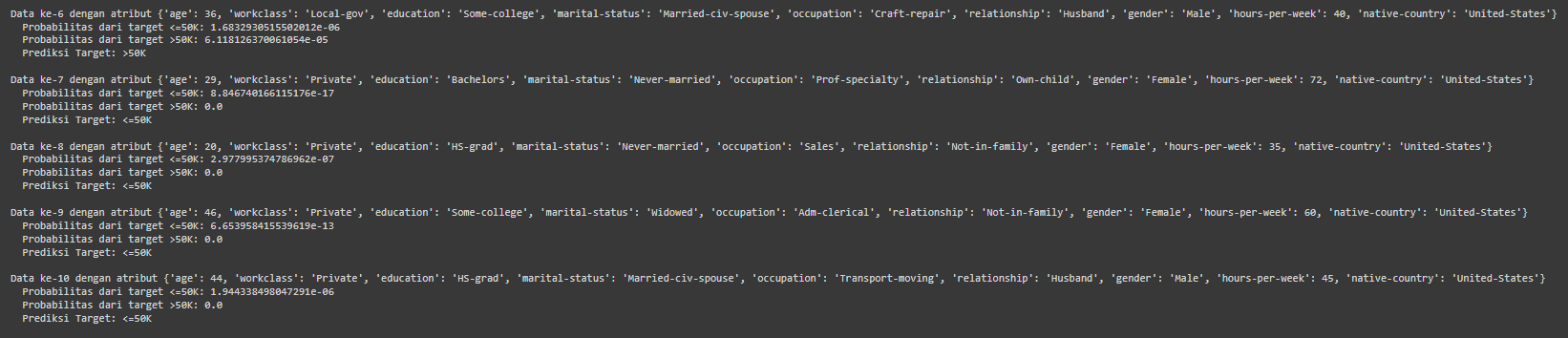


**Gambar 4.5 Hasil Perhitungan Probabilitas Kondisional.**

### 4.1.5 Hasil Probabilitas dan Prediksi Target

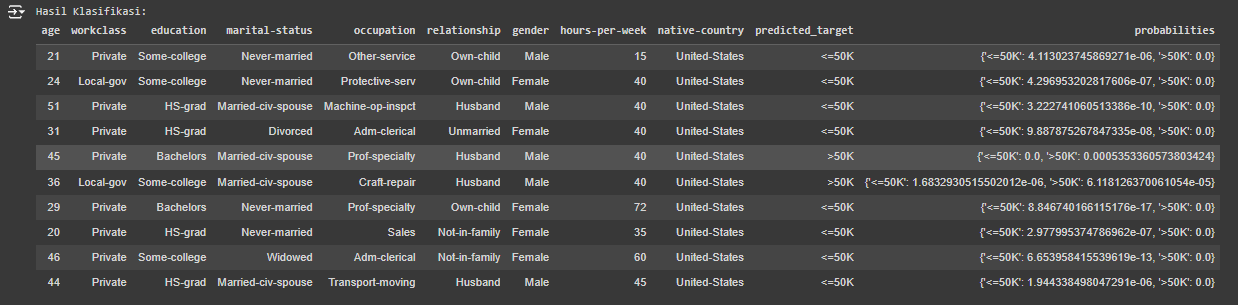


**Gambar 4.6 Hasil Probabilitas dan Prediksi Target.**

****

**Gambar 4.7 Hasil Probabilitas dan Prediksi Target.**

### 4.1.6 Hasil Klasifikasi



**Gambar 4.8 Hasil Klasifikasi.**

# **BAB V**

**KESIMPULAN**

## **5.1 Kesimpulan**

Kesimpulan dari implementasi algoritma Naïve Bayes ini menunjukkan bahwa algoritma tersebut efektif dalam klasifikasi data pendapatan orang dewasa, mampu memberikan prediksi yang cukup akurat meskipun menggunakan asumsi independensi antar fitur yang sederhana. Implementasi manual Naïve Bayes dalam Python tanpa menggunakan library eksternal menuntut pemahaman mendalam tentang dasar teori dan probabilitas, melibatkan perhitungan probabilitas prior dan kondisional serta penerapan smoothing untuk menghindari probabilitas nol. Hasil klasifikasi mengindikasikan bahwa prediksi yang dihasilkan oleh model konsisten dengan data uji yang digunakan. Meskipun Naïve Bayes mudah diimplementasikan dan efisien dalam hal waktu komputasi, asumsi independensi fitur dapat menjadi kelemahan ketika fitur saling bergantung, yang dapat mempengaruhi akurasi prediksi. Algoritma ini cocok digunakan dalam berbagai aplikasi klasifikasi dengan dataset yang besar dan membutuhkan kecepatan prediksi yang tinggi. Penelitian ini memberikan wawasan berharga tentang implementasi dan aplikasi algoritma Naïve Bayes dalam dunia nyata, serta mengidentifikasi area di mana algoritma ini dapat ditingkatkan atau digunakan secara lebih efektif.

# **Referensi**

* <https://www.kaggle.com/datasets/wenruliu/adult-income-dataset>
* Naive Bayes Classification dan Contoh Perhitungannya.pdf (Materi Naïve Bayes pada matakuliah Kecerdasan Buatan)

# **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