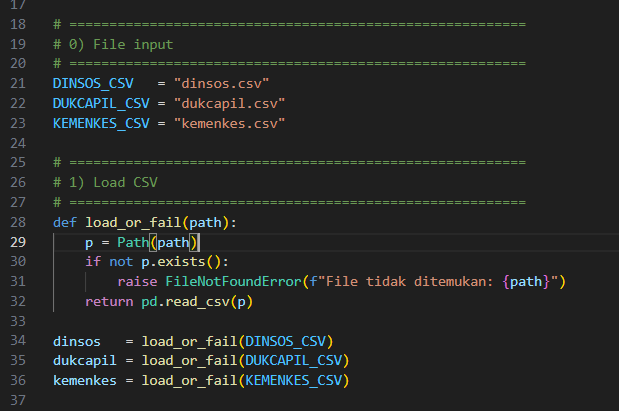
Topik : 4.4. Finalisasi Prototipe

Objective : Siapkan pipeline end-to-end: dataset → FL → DP → hasil

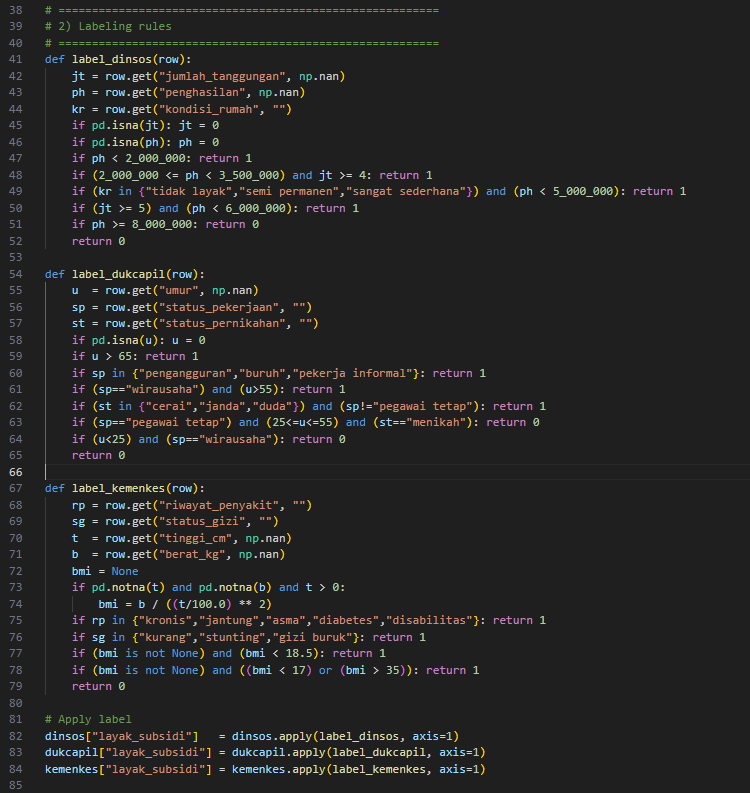
Task : Uji ulang semua modul, pastikan reproducible



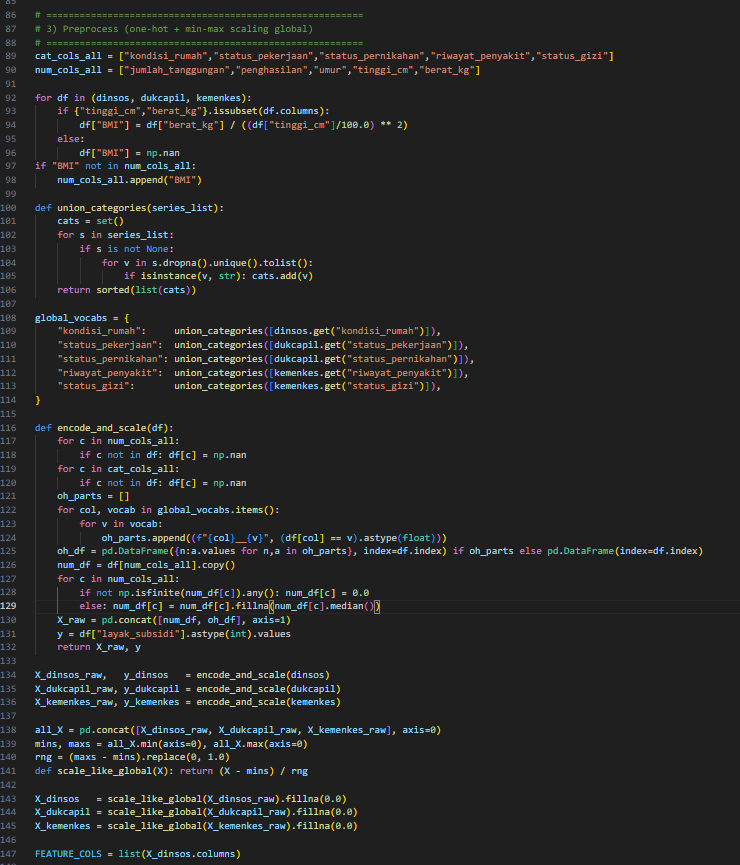
1. Dataset (Load Data)



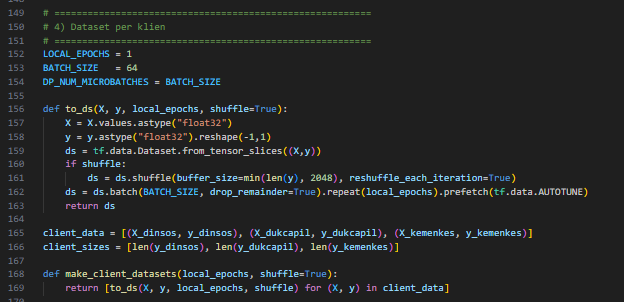
1. Labeling



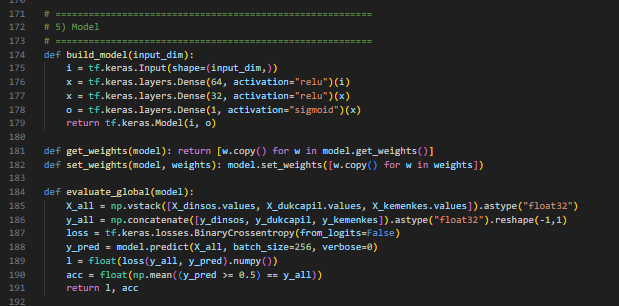
1. Preprocessing



1. Dataset per klien



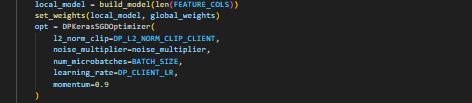
1. Model

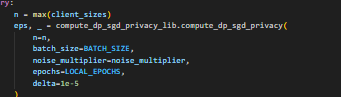


1. Federated learning

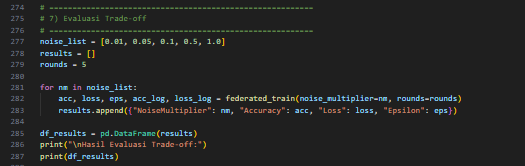


1. Differential Privacy

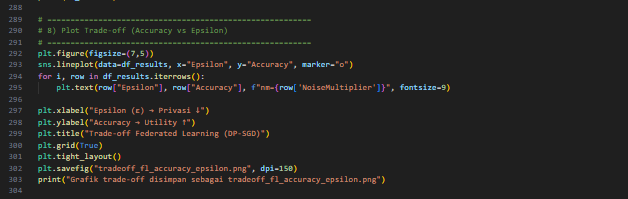




1. Evaluasi



1. Plot Trade-off



**Pipeline end-to-end :**

1. Dataset (load data)

Tahap pertama Adalah mengambil data mentah dari tiga client :

* Dinsos 🡪 berisi informasi jumlah\_tanggungan, penghasilan, kondisi rumah
* Dukcapil 🡪 berisi informasi umur, status\_pekerjaan, status\_pernikahan
* Kemenkes 🡪 berisi informasi Riwayat\_penyakit, status\_gizi, tinggi, berat

Data ini di-*load* dari file CSV menggunakan pandas.read\_csv(). Jika file tidak ada, program berhenti agar reproducibility terjaga.

1. Labeling

Data mentah tidak memiliki target langsung. Maka dibuat aturan untuk menentukan label layak\_subsidi (1 = layak, 0 = tidak layak).

* Output tahap ini : kolom baru layak\_subsidi pada tiap dataset

1. Pre-processing

Agar model dapat dilatih :

* Numerik : nilai yang hilang diisi mendian, lalu di-*scale* pakai min-max normalisasi global
* Kategori : diubah menjadi one-hot encoding berdasarkan vocab gabungan (supaya konsisten antar klien)
* Fitur Tambahan : BMI dihitung dari tinggi & berat badan

Output : matriks fitur numerik dan vector label yang sudah bersih

1. Dataset per Klien

Setiap sumber data dianggap sebagai client dalam Federated Learning.

* Data tiap klien dikonversi ke tf.data.Dataset kemudian dilakukan batching, shuffling, repeat sesuai dengan local epochs
* Tiga dataset klien siap dipakai dalam loop training

Hal ini mensimulasikan kondisi nyata Dimana data tidak dikumpullkan disatu tempat, melainkan dilatih secara terdistribusi

1. Model

Menggunakan arsitektur :

* Input layer sesuai jumlah fitur
* Hidden layer : 64 neuron (ReLU) 🡪 32 neuron (ReLU)
* Output layer : 1 neuron sigmoid (binary classification)

Penerapan model yang cukup ringan sehingga cocok untuk melakukan federated learning.

1. Federated Learning

Implementasi manual Federated Averaging (FedAvg) :

* Tiap klien dimulai dari bobot global
* Klien melatih model local pada datasetnya dengan optimizer DP-SGD
* Delta bobot hasil training di-clip dengan L2 norm agar tidak terlalu besar
* Server melakukan aggregasi delta bobot dari semua klien (weighted average berdasarkan ukuran dataset
* Update bobot global 🡪 looping untuk beberapa round

1. Differential privacy

Untuk menjamin privasi data tiap klien, maka digunakan :

* DP-SGD Optimizer 🡪 menambahkan *noise Gaussian* ke gradien, plus clipping gradien
* Menggunakan parameter penting
  + L2\_norm\_clip 🡪 batas clipping gradien
  + Noise\_multiplier 🡪 besar noise yang ditambahkan
* Estimasi privasi dihitung dengan menggunakan compute\_dp\_sgd\_privacy 🡪 menghasilkan nilai ε (epsilon), semakin kecil artinya privasi lebih kuat

1. Evaluasi

Setelah setiap federated round :

* Model global dievaluasi pada gabungan semua data klien
* Metrics : accuracy dan loss
* Disimpan log per round agar bisa dianalisis konsistensinya
* Dicatat pula nilai rata – rata akurasi/loss untuk tiap noise multiplier

1. Plot Trade-off

Hasil evaluasi divisualisasikan :

* Grafik Accuracy vs Epsilon (ε) 🡪 menunjukkan trade-off antara utility(akurasi) dan privacy **(ε)**.
* Tiap titik diberi label nm = noise\_multiplier
* Grafik disimpan dalam file PNG agar reproducible