Project summary

A federated learning system (FedAvg) in Rust. A simple MNIST linear model is trained locally on multiple “clients,” and a “server” aggregates parameters via averaging.

Core pieces:

- `common.rs`: Linear model, training loop (ndarray), FedAvg, accuracy.
- `data.rs`: Loads and splits MNIST into client subsets.
- `simple_demo.rs`: Runs a full local simulation of federated rounds.
- `client.rs` / `server.rs`: gRPC-oriented client/server (feature-gated).

How to use

- Build:
  ```bash
  cargo build
  ```
- Run the demo (recommended):
  ```bash
  cargo run --bin simple_demo
  ```
  This:
  - Initializes a global model
  - Splits MNIST among 3 clients
  - Trains locally on each client
  - Aggregates via FedAvg for several rounds
  - Prints accuracies

- Networking (advanced, optional):
  - The gRPC `client`/`server` are behind a feature flag and need service wiring to fully work with tonic.
  - Build with the flag:
    ```bash
    cargo build --features grpc
    ```
  - You will need to implement/plug in tonic service glue or generate code from `proto/federated_learning.proto` to run true client/server.

- TLS note: `reqwest` uses Rustls here, so no OpenSSL setup is required.

- Quick recap:

- For a smooth, self-contained run, use `simple_demo`.
- Use the `grpc` feature only if you plan to complete the networked path.