Federated Learning Pipeline

Step-by-step execution guide for data preparation, partitioning, and experiment runs

---

Step 0: Activate Virtual Environment

powershell

```
.\virt\Scripts\Activate.ps1
```

Step 1: Clean Previous Results

Optional — Fresh Start

powershell

```
Remove-Item data\processed -Recurse -Force -ErrorAction SilentlyContinue
Remove-Item data\partitions -Recurse -Force -ErrorAction SilentlyContinue
Remove-Item results -Recurse -Force -ErrorAction SilentlyContinue
```

Step 2: Download Raw Data

Skip if already downloaded.

powershell

```
python download_data.py
```

Step 3: Process & Harmonize Raw Data

Loads BRFSS (1.16 GB) + NHANES → harmonizes features → train/test split → saves CSVs + scaler.

powershell

```
python -m src.data_prep --output data/processed
```

Step 4: Partition Data for Federated Learning

Dirichlet non-IID split → 5 BRFSS + 5 NHANES hospital nodes.

powershell

```
python -m src.partition --input data/processed --num-clients 10 --alpha 0.5
```

Step 5: Run Quick Test

Runs: Centralized → Local-Only → FedAvg → FedAvg+DP → Full System → Sweeps (all abbreviated). 5 rounds, 3 clients — validates everything works.

powershell

```
python -m experiments.run_all --quick
```

Step 6: Run Full Experiment Suite

> *Full 50-round training for all baselines + privacy-utility sweep + non-IID robustness sweep. Production run — takes several hours.*

powershell

```
python -m experiments.run_all
```

Or Run Individual Experiments

powershell

```
# Centralized baseline (upper bound)
python -m experiments.centralized

# Local-only baseline (lower bound)
python -m experiments.fedavg --baseline local_only

# Vanilla FedAvg (no DP)
python -m experiments.fedavg --baseline fedavg

# FedAvg + DP
python -m experiments.fedavg --baseline fedavg_dp

# Full system (FedProx + DP + Compression)
python -m experiments.run_all --quick   # use --quick for fast validation
```

Quick Copy-Paste (All Steps at Once)

powershell

```
.\virt\Scripts\Activate.ps1

# Clean slate
Remove-Item data\processed -Recurse -Force -ErrorAction SilentlyContinue
Remove-Item data\partitions -Recurse -Force -ErrorAction SilentlyContinue
Remove-Item results -Recurse -Force -ErrorAction SilentlyContinue

# Pipeline
python -m src.data_prep --output data/processed
python -m src.partition --input data/processed --num-clients 10 --alpha 0.5
python -m experiments.run_all --quick
```

Note: Steps 3 & 4 (data prep + partition) are also automatically handled by `run_all` if `data/processed` doesn't exist. But running them separately lets you validate each stage independently before committing to the full experiment run.