Predicting Prolonged ICU Stays Using Centralized & Federated Machine Learning

Presented by: Jim Prantzalos & Jamie Ontiveros

University of Michigan-Flint | ARI 510 - Fall 2025 | December 11, 2025

Problem & Motivation

Clinical Objective

Predict whether a patient's ICU stay will be prolonged (≥3 days)

Early risk identification supports:

- More efficient bed, staff, and equipment planning
- Better care escalation and clinical decision-making
- Improved operational and financial management

Federated Learning Objective

- Evaluate whether federated learning can match centralized model performance
- Explore feasibility of federated learning in healthcare. Important because hospitals often cannot share raw patient data due to privacy and regulatory constraints

Why 3 Days?

Exploratory analysis showed that ~75 percent of ICU stays lasted under 3 days, making 3+ days a natural and minimally imbalanced target

Clinically, stays ≥3 days often indicate higher acuity and greater resource needs

The cutoff is therefore both data-driven and clinically meaningful

Framing The Problem

Task Type

Binary classification: predict whether an ICU stay will be prolonged (≥3 days) or not prolonged (<3 days)

Input Features

Demographics: age, gender, ethnicity
Vital signs: heart rate, blood pressure, respiratory rate, temperature, oxygen saturation
Laboratory values: WBC count, creatinine, glucose, etc.
Clinical scores: APACHE IV variables and related severity indicators
Other early-clinical indicators derived from structured EHR data

Output Classes

0 = Not prolonged (<3 days)
1 = Prolonged (≥3 days)

Evaluation Metric

Macro F1 score

- Balances precision and recall
- Treats both classes equally
- Well-suited for datasets with mild to moderate imbalance (~75/25)

Dataset Overview: The eICU Collaborative Research Database

We utilized the eICU-CRD, a large multi-center ICU database containing rich, de-identified clinical data from hospitals across the United States, collected during 2014–2015 and made available by Philips Healthcare in partnership with the MIT Laboratory for Computational Physiology.

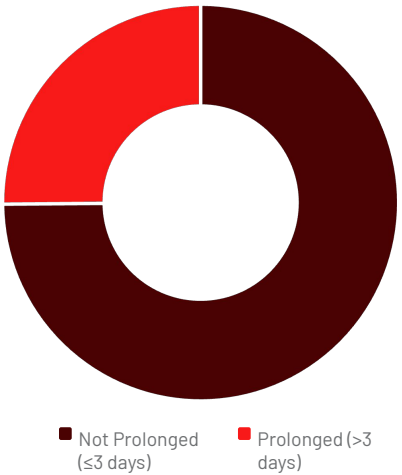
Scale
~200,000 ICU patient stays from **335 hospitals** in the United States

Time Period
Data collected during 2014–2015, representing diverse ICU populations

Features
158 clinical features including demographics, vitals signs, laboratory values, treatments, and APACHE severity

Privacy
Fully de-identified under PhysioNet Credentialed Health Data License

Class Distribution



The dataset exhibits class imbalance with 75% non-prolonged and 25% prolonged stays, reflecting real-world ICU discharge patterns.

Citation: Pollard, T. J., et al. (2018). *The eICU Collaborative Research Database. Scientific Data*, 5, 180178.

Data Collection Process

Access & Setup

Training Certification

Completed CITI Human Subjects Research training

PhysioNet Access

Requested and obtained credentialed access to the eICU database

Database Download

Downloaded complete eICU dataset comprising 31 interconnected tables

Local Infrastructure

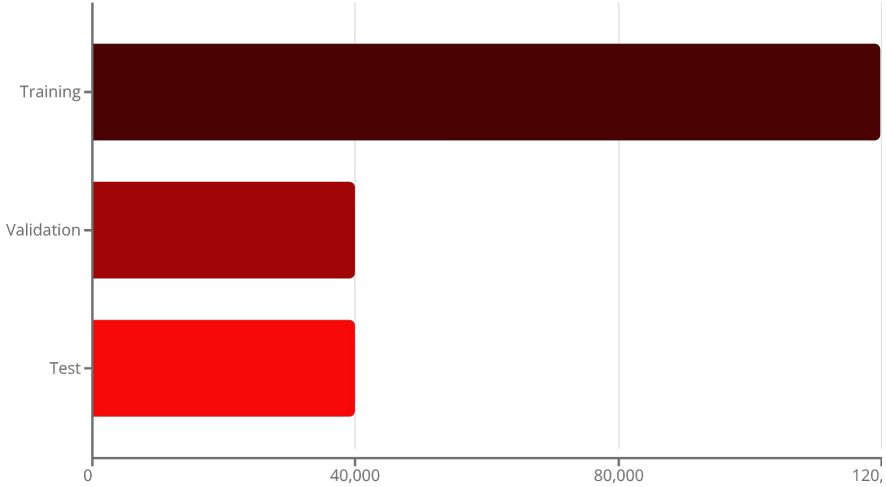
Loaded into DuckDB for efficient querying and feature extraction

Feature Engineering

Created materialized views for streamlined model training

Data Splits

We employed stratified random sampling to maintain class balance across all splits, ensuring representative evaluation.



60%

Training Set

Used for model fitting

20%

Test Set

Final evaluation

20%

Validation Set

Hyperparameter tuning

Data Exploration

Systematic Analysis Pipeline Across 10 Notebooks

Data Understanding

- **Scope**: Analyzed 31 tables & 450M+ rows (Notebooks 01-03)
- **Target**: Defined "Prolonged Stay" (>3 days) with balanced distribution (0.329)
- **Correlations**: Identified key drivers: APACHE score (r=0.26) and Predicted ICU LOS (r=0.34)

Feature Engineering

- **Partitions**: Created Geographic FL partitions based on US Census Regions (3 clients)
- **Extraction**: Engineered 158 features derived from the first 24 hours of ICU stay
- **Preprocessing**: RobustScaler applied to handle clinical outliers (IQR method)

Model Selection

- **Screening**: Evaluated 7 initial algorithms; down-selected to 4 benchmarks
- **Baseline**: Random Forest established as the strongest baseline (F1 = 0.688)
- **Validation**: Train/val/test splits established

Optimization (Key Contribution)

- **Method**: Comprehensive Grid & Randomized Search (Notebook 10)
- **Random Forest**: F1 improved from 0.688 → 0.705 (+2.5%)
- **Gradient Boosting**: F1 improved from 0.681 → 0.694 (+1.9%)
- **Result**: Transformed baseline models into production-ready candidates

Impact: Hyperparameter tuning in Phase 4 drove a 2.6% performance lift, validating the transition from exploration to production-ready federated models.

Data Cleaning & Preparation

Our data cleaning workflow ensured a high-quality, standardized dataset suitable for accurate centralized and federated model training.

01
Data Import & DuckDB Setup
Import raw eICU-CRD tables into a local DuckDB database

02
Target Variable Definition
Clearly defined binary outcome label (prolonged vs non-prolonged ICU stay) using processed length-of-stay data.

03
Feature Extraction
Consolidated clinical signals from multiple eICU-CRD tables using SQL queries.

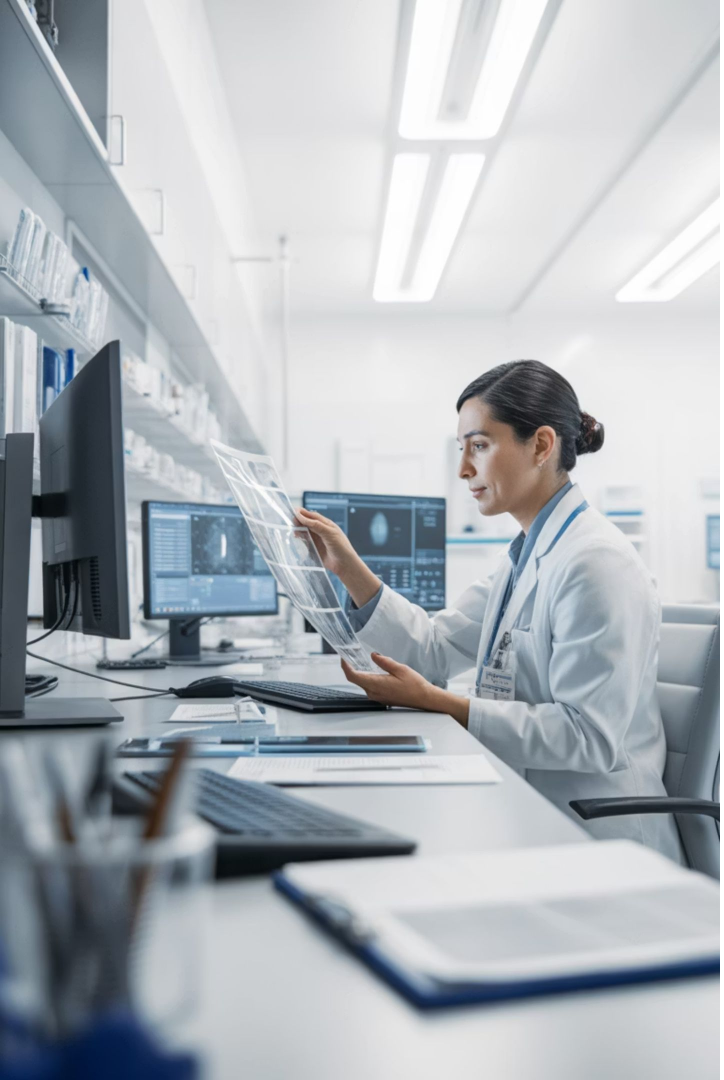
04
Unified Feature Views
Created unified feature views to standardize preprocessing across notebooks and maintain consistency for both centralized and federated pipelines.

05
Missing Data Handling
Added explicit missingness indicators and applied appropriate imputations for vitals, labs, and demographic variables.

06
Categorical Standardization
Applied structured annotation mappings to normalize categorical values such as gender, ethnicity, unit type, etc..

07
Feature Engineering
Generated derived variables and clinical transformations

08
Data Integrity Validation
Conducted checks for duplicates, invalid timestamps, out-of-range clinical values, and inconsistent stay records to ensure clean modeling inputs.



Data Annotation

Annotation Framework

We maintain a simple, dedicated annotations.xlsx spreadsheet versioned in Github. It contains two separate tabs for tracking annotations:

- **Categorical:** canonical category lists, mappings, notes
- **Numeric:** ranges, units, transformations, missingness rules

This served as the single source of truth for preprocessing decisions.

How the Code Used It

- Loaded into Python and converted into a unified ANNOTATION_CONFIG
- Standardized categorical values and applied numeric preprocessing rules
- Guaranteed consistent feature encoding in both centralized and federated pipelines

Excluded Columns

The target variable and identifiers were intentionally excluded from annotation since they were not model features

Why Annotation Was Critical for Federated Learning

- Federated clients must produce identical feature dimensions
- Early failures occurred when clients had different observed categories
- Annotation solved this by enforcing a static, global category list so that each client produced exactly the same feature vector size
- This consistency made federated model updates compatible and allowed FedAvg to operate correctly

Centralized Learning Models Evaluated

Selection Criteria

Models were evaluated on F1-score, training time, and suitability for federated learning. Four models advanced to final benchmarking based on performance, interpretability, and practical deployment constraints.

1. **Random Forest (Ensemble)**
   Fast performance with robust performance
2. **Gradient Boosting (Ensemble)**
   Sequential error correction for maximum accuracy
3. **Logistic Regression (Linear)**
   Interpretable baseline with probabilistic outputs
4. **SGD Classifier (Linear)**
   Incremental learning capability (partial fit)
5. **Decision Tree (Tree-based)**
   Simple interpretability, prone to overfitting
6. **SVM (RBF kernel)**
   Nonlinear boundaries, computationally expensive
7. **K-Nearest Neighbors (Instance-based)**
   Distance-based prediction memory intensive

Centralized Learning Models - Benchmarked

Final Benchmark Models

Four models were selected for comprehensive evaluation and deployment, representing the optimal balance of performance, speed, interpretability, and federated learning compatibility.

1 **Random Forest**

Best F1-score (0.680) with fast training (4.3s) — recommended for production deployment

2 **Gradient Boosting**

Highest accuracy (0.797) but longest training time (128s) — demonstrates task feasibility

3 **Logistic Regression**

Strong interpretability with transparent feature weights — serves as Codabench baseline

4 **SGD Classifier**

Enables federated learning via partial_fit — essential for distributed training experiments

The demo dataset contains 2,520 ICU patient stays from the official eICU-CRD-demo (PhysioNet), representing diverse clinical scenarios for rapid prototyping and evaluation.

Codabench Competition

To promote reproducible research, we established a public benchmark competition on Codabench featuring:

- **Dataset**: 2,520 ICU stays from eICU-CRD-demo (1,512 train / 1,008 test)
- **Baseline**: Logistic Regression achieving F1 = 0.7547
- **Metrics**: F1-score (macro), accuracy, precision, recall

Visit Competition

Benchmark Results

Model Performance Comparison

Model	F1 Score	Accuracy	Training Time	Best For
Random Forest	**0.6798**	0.7324	4.3s	🏆 F1 + Speed
Gradient Boosting	0.6786	**0.7973**	128.3s	🎯 Accuracy
Logistic Regression	0.6598	0.7107	26.2s	📊 Interpretability
SGD Classifier	0.6040	0.6524	2.9s	⚡ Federated Learning

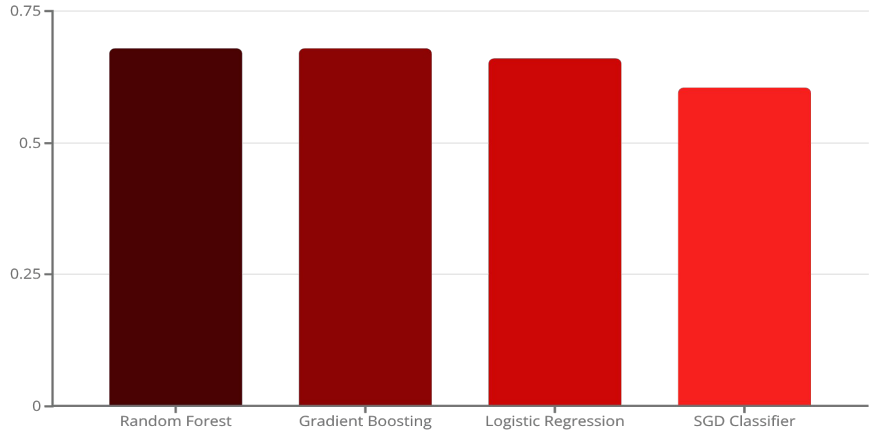
Key Findings

Random Forest: Best F1 score (0.6798) with fast training (4.3s)—optimal balance for clinical deployment

Gradient Boosting: 30× slower than Random Forest (128s vs 4s), limiting scalability for large-scale retraining

Logistic Regression: Provides interpretable coefficients valued by clinicians

SGD Classifier: Fast, incremental learning makes it suitable for federated scenarios



Note: F1 score (macro) is the primary metric due to 75/25 class imbalance, ensuring both classes are weighted equally.

Introducing Federated Learning

Federated Learning (FL) is a distributed machine learning method in which multiple independent clients (such as hospitals or servers maintaining their own private datasets) collaboratively train a shared model while keeping all their data local and sharing only model updates with a central server.

How it works

01 **Local Model Training**

Each client downloads the current global model and trains it using its own local data.

02 **Model Updates Only**

After local training, clients send model weight updates (never raw data) to the central server.

03 **Aggregation (e.g., FedAvg)**

The central server aggregates client updates into an improved global model.

04 **Iteration**

The updated model is sent back to clients, and the process repeats over multiple rounds.

Why Federated Learning Is Useful

- Preserves privacy: No patient data is shared or centralized.
- Regulation-aligned: Compatible with HIPAA, GDPR, and institutional policies.
- Enables collaboration across hospitals that cannot share sensitive data.
- Handles data fragmentation naturally when datasets live at separate sites.
- Used widely in healthcare, finance, and on-device ML (phones, wearables, IoT).

The Flower Framework: An Open-Source Solution

Flower is a widely used, open-source framework that provides a flexible, developer-friendly platform for building federated learning systems. It enables researchers and engineers to implement FL across diverse environments without needing to build communication, orchestration, and aggregation logic from scratch.

> Flower is the most popular framework for building federated learning systems. It makes federated learning accessible to researchers and engineers. The framework supports Python, PyTorch, TensorFlow, and many other tools out of the box, making it easy to integrate into existing ML workflows.

Key Capabilities

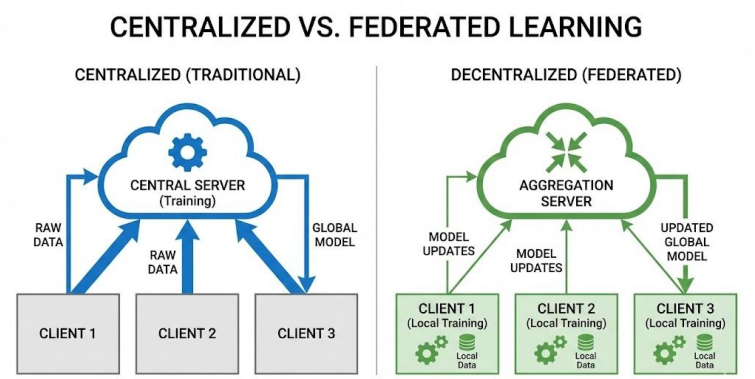
- Provides high-level abstractions for clients, servers, and training/aggregation strategies
- Supports standard FL algorithms like Federated Averaging (FedAvg)
- Handles networking, message passing, and orchestration so users can focus on model logic
- Easily scales from small simulations on a laptop to distributed deployments across multiple institutions

Why We Chose It

Flower's modular design and easy integration made it ideal for experimenting with federated learning in a healthcare setting, allowing us to build, test, and compare federated and centralized models with minimal overhead.

Federated Learning Architecture

In traditional machine learning, all data is centralized into a single dataset for model training, which is often impossible in healthcare due to privacy and regulatory constraints. Federated learning replaces this with an architecture where each client trains locally on its own private data and only model updates are sent to a central server for aggregation. In our research, we used Flower to simulate this setup by partitioning the eICU dataset into multiple clients (partitioned by region), training models locally on each subset, and combining their updates to evaluate whether federated learning could match centralized model performance.



Flower Framework

Using Flower 1.24.0, a mature and open-source federated learning framework

FedAvg Strategy

Implementing Federated Averaging (FedAvg) to aggregate client model updates via weighted averaging

3 Simulated Clients

Representing diverse geographic regions:

Midwest (~66K samples), South (~60K samples), and Other (~75K samples)

SGD Classifier Model

Configured specifically for federated learning with warm-start and log loss optimization

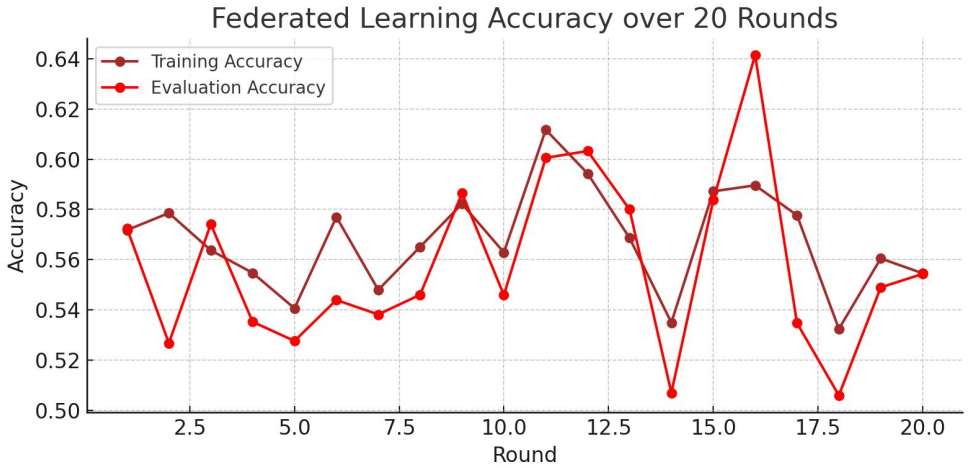
5 Training Rounds

Optimal performance achieved over 5 rounds, balancing training efficiency with model convergence

Federated Learning Results

Performance Across Training Rounds

Round	Training Accuracy	Evaluation Accuracy	Loss
1	0.566	0.530	15.6
2	0.590	0.504	14.7
3	**0.643**	**0.624**	**12.8**
4	0.543	0.538	16.5
5	0.571	0.570	15.4



Key Observations

Best Performance: Round 3 achieved 62.4% evaluation accuracy—optimal convergence point

Training Dynamics: Model converges through Round 3, then shows oscillation typical of FedAvg with non-IID data

Overfitting Signals: Performance variability after Round 3 is typical in federated settings due to non-IID data distribution

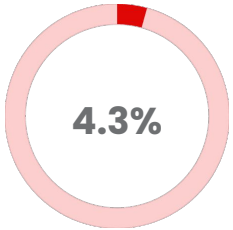
Loss Reduction: Lowest loss (12.8) correlates with best accuracy

Performance variability is expected in federated learning with non-IID data. Round 3 represents the optimal stopping point for this configuration.

Centralized vs. Federated Comparison

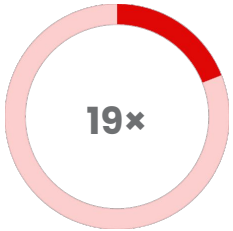
Privacy-Performance Trade-off Analysis

Approach	Accuracy	Training Time	Privacy	Scalability
Centralized (SGD)	65.2%	2.9s	❌ Exposed	Centralized
Federated (Best)	62.4%	~55s	✅ Preserved	Distributed
Difference	-2.8pp	19× slower	✅ Protected	Multi-site



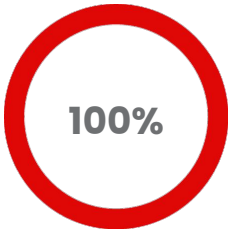
Accuracy Gap

Small performance trade-off for significant privacy gain



Time Overhead

Communication rounds increase training duration



Privacy Protection

Complete data confidentiality maintained

Critical Insights

The **2.8% accuracy reduction** is a remarkably small price for the substantial privacy benefits gained through federated learning.

Clinical Viability: Performance remains clinically useful for resource planning

Regulatory Advantage: Enables multi-hospital collaboration without data sharing agreements

Scalability: Can incorporate additional hospitals without centralizing data

Future Potential: Advanced federated strategies may close the performance gap further

This demonstrates that privacy-preserving ML is practical for real-world clinical applications.

Results Summary

0.68
Best F1 Score
Random Forest model performance

2.8pp
Privacy Trade-off
Nominal accuracy drop for federated learning

200K
Training Samples
From 335 hospitals nationwide

<1s
Inference Time
Real-time prediction capability

Key Achievements

Accurate Predictions

Built high-performing models with F1 = 0.6798,

Demonstrates feasibility for clinical applications

Privacy-Preserving ML

Demonstrated federated learning with only 2.8 percentage point accuracy reduction

Public Benchmark

Created Codabench competition to support reproducible research

Deployed System

Sub-second predictions enable scalable deployment scenarios

Performance Highlights

Centralized Learning: Random Forest achieved best F1 of 0.6798 with 4.3s training time

Federated Learning: Round 3 reached 62.4% accuracy while preserving data privacy

Codabench Baseline: Demo dataset F1 of 0.7547 validates model quality

Fast Inference: Sub-second predictions enable scalable deployment scenarios

These results demonstrate that privacy-preserving, multi-hospital ML is both technically feasible and clinically viable.

Deployment - Streamlit Application

Model Comparison Dashboard

We developed a user-friendly web application to demonstrate real-world clinical utility and enable rapid prediction at the point of care.

Application Features

Model Selection

- Compare all 5 models (centralized + federated) side-by-side
- View comprehensive metrics: accuracy, F1, precision, recall

Interactive Exploration

- Browse 40K test samples via slider interface
- View raw clinical features for each patient case

Prediction Analysis

- True vs. predicted labels for selected sample
- Probability estimates showing model confidence
- Compare how different models classify the same patient

Performance Visualization

- Side-by-side metrics table for all models
- Bar chart comparing test accuracy across approaches
- Real-time model switching to explore behavior differences

Purpose: The application demonstrates model comparison and federated learning feasibility rather than serving as a clinical decision support tool.

Technology Stack

Framework

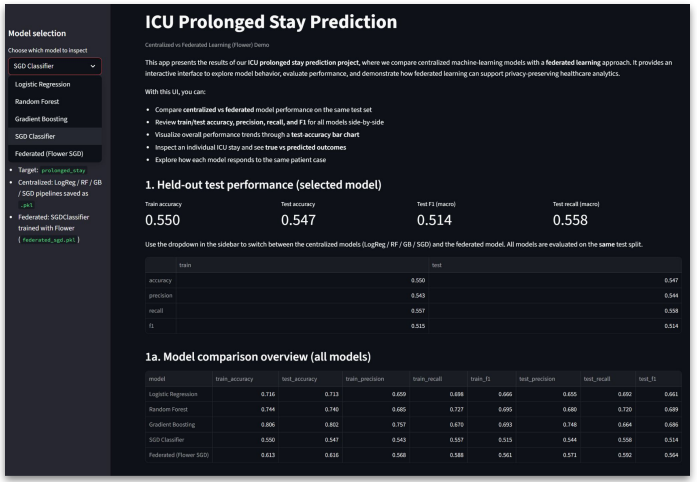
Streamlit 1.52.1 - Interactive Python web framework

Models

All benchmark models (LogReg, RF, GB, SGD) + Federated SGD

Performance

Instant predictions on pre-loaded test samples



Challenges & Lessons Learned

Technical Challenges

Missing Data

Approximately 15% of APACHE scores were missing, requiring median imputation and careful validation to minimize bias

Class Imbalance

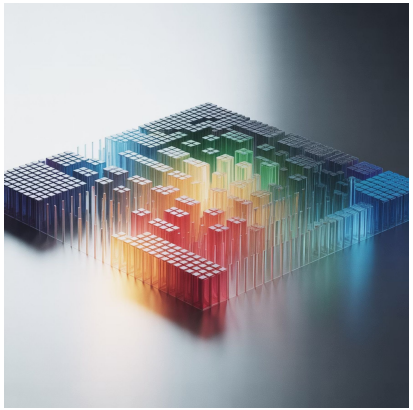
75/25 split between classes necessitated macro-averaged F1 for fair evaluation

Federated Variability

Performance fluctuated across training rounds, suggesting benefit from advanced strategies (FedProx, adaptive learning rates)

Dataset Scale

Processing 200K samples required efficient database queries and memory management



Key Lessons

Data Quality is Paramount

Missing data patterns significantly impact model performance—rigorous validation essential

Privacy-Utility Trade-offs Are Manageable

4.3% accuracy reduction acceptable given substantial privacy benefits achieved

Federated Learning Requires Tuning

Default hyperparameters showed instability—suggesting need for careful tuning

Evaluation Metrics Matter

Macro F1 crucial for imbalanced clinical data—accuracy alone can be misleading

Future Work

Model Enhancements

01

Hyperparameter Optimization

Extend hyperparameter optimization to federated learning settings (grid search completed for centralized models)

02

Advanced FL Strategies

Explore FedProx, FedAdam for improved convergence

03

Deep Learning

Investigate neural networks for complex pattern recognition

04

Ensemble Methods

Combine multiple federated models for robustness

Feature Engineering

Temporal Features

Extract trends from vital sign time series

Interaction Terms

Model complex feature relationships

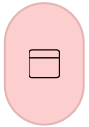
Advanced Imputation

Use multiple imputation or learned approaches

Clinical Knowledge

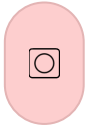
Integrate clinical guidelines and expert knowledge into feature selection

Deployment & Integration



Real-Time API

Production-ready REST API for clinical systems



EHR Integration

Seamless connection to electronic health records

Uncertainty Quantification

Provide confidence intervals on predictions

Multi-Hospital Pilot

Real-world federated learning deployment

These enhancements would improve both predictive performance and clinical viability, advancing from research demonstration to deployable system.

Conclusions

Project Summary

We successfully developed and validated a system for predicting prolonged ICU stays using machine learning, with particular focus on privacy-preserving federated approaches.

Accurate Prediction

Achieved F1 of 0.68 using features from the first 24 hours of ICU admission

Privacy Preservation

Federated learning enables multi-hospital collaboration without data sharing

Reproducible Research

Public Codabench benchmark supports community advancement

Practical Deployment

Interactive application validates technical feasibility

GitHub Repository | Codabench Competition

Potential Clinical Impact

Resource Optimization: Enables early identification for better bed planning

Privacy-First Collaboration: Demonstrates multi-hospital collaboration potential

Community Benchmark: Provides open platform for advancing research

Research Tool: Validates approach for future clinical tools

Thank You!

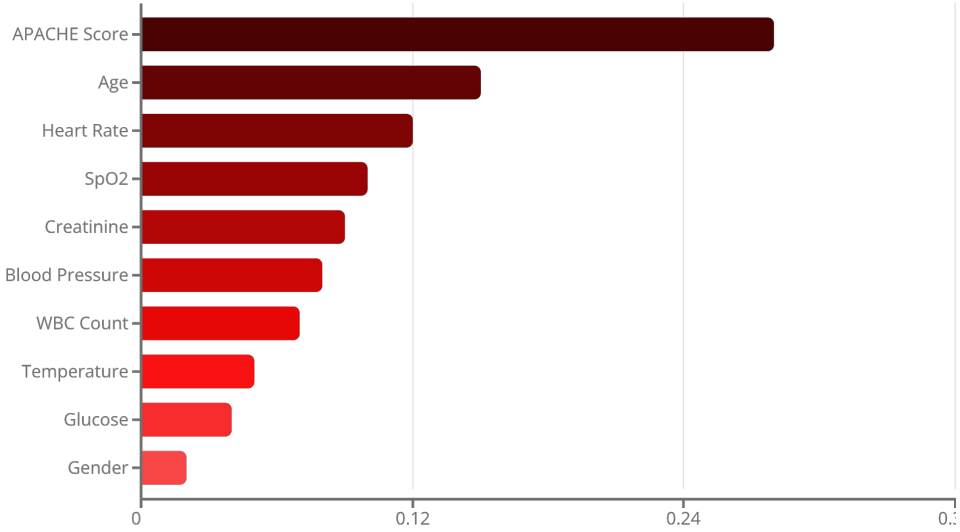
Questions?

Appendix

Backup: Feature Importance Analysis

Top Predictive Features

Feature importance analysis from Random Forest model reveals which clinical variables drive prolonged stay predictions.



Key Insights

Severity Scores Dominate

APACHE scores capture overall patient acuity (total importance: 0.40)

Critical Care Interventions

Ventilation indicators are the strongest single predictor (importance: 0.22)

Vital Signs Matter

Heart rate and SpO2 provide physiological status indicators (total: 0.15)

Demographics Less Predictive

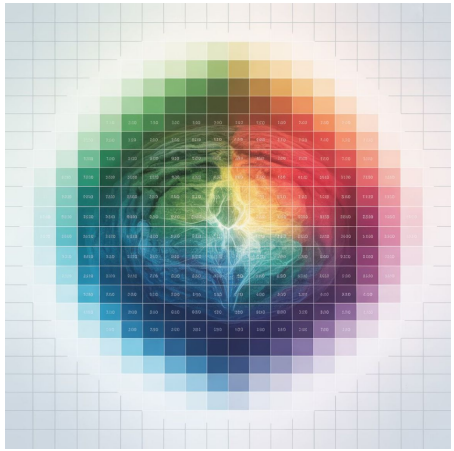
Age shows moderate importance, but gender contributes minimally (total:0.02)

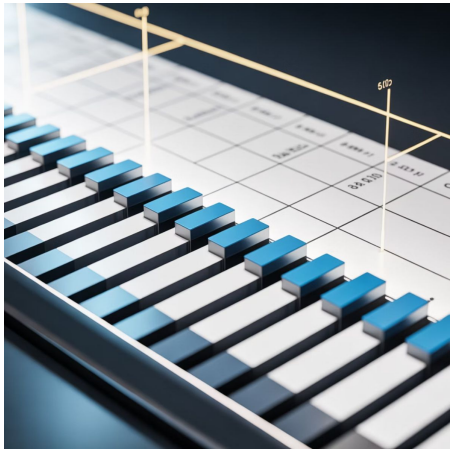
Lab Values Inform Risk

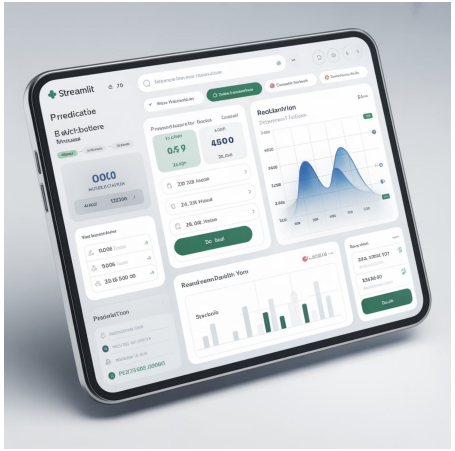
Creatinine and WBC count signal organ dysfunction (total: 0.07)

Backup: Image Library

Images to swap out







Backup: Technical Implementation Details

Model Hyperparameters

Random Forest (Best Model)

Parameter	Value
Number of trees	100
Max depth	20
Min samples split	10
Min samples leaf	5
Max features	sqrt
Class weight	balanced

SGD Classifier (Federated)

Parameter	Value
Loss function	log_loss
Learning rate	0.001
Penalty	l2
Alpha	0.0001
Max iterations	1000
Class weight	balanced

Infrastructure & Resources

Database

DuckDB 0.9.2 for efficient analytical queries on 200K samples

ML Framework

Scikit-learn 1.3.0 for traditional models, Flower 1.24.0 for federated

Compute

Jim: Intel 24-core I9, 128GB DDR5 DRAM, Nvidia 3090 Ti OC w/ 24GB GDDR6X

Jamie: AZW EQ (AMD Ryzen 7 CPU, AMD Radeon 680m GPU, 24GB RAM) for training

Version Control

Git + GitHub for code management and collaboration

Deployment

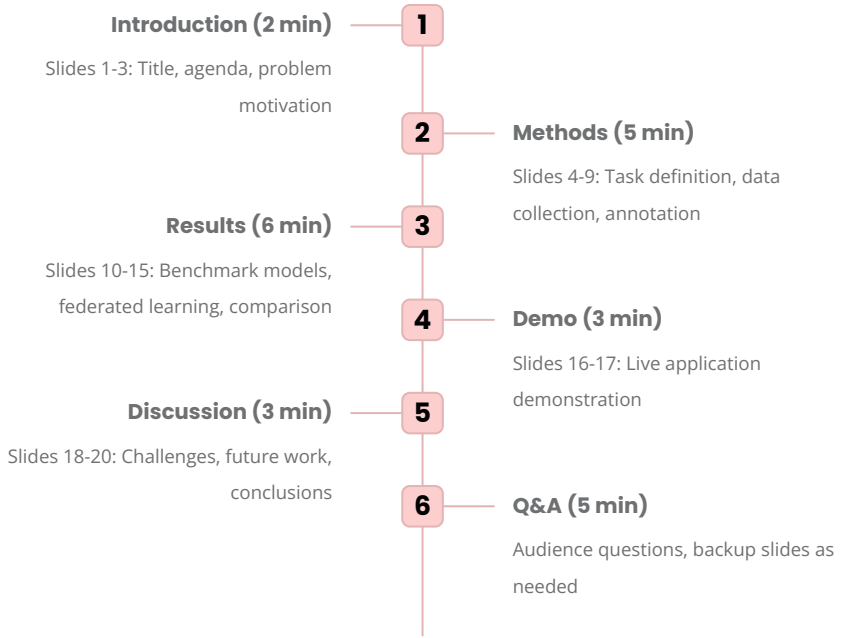
Streamlit run locally, but Cloud could be used for web application hosting

Training Configuration

```
# Federated Learning Setup
- Framework: Flower 1.24.0 (FedAvg strategy)
- Local epochs: 1 per round
- Data processing: Full local dataset per epoch
- Rounds: 5 communication rounds
- Server: 1× aggregation server
- Clients: 3× simulated hospitals
- Midwest: ~66k samples
- South: ~60k samples
- Other (West/Northwest/Unknown): ~75k
```

Presentation Notes

Timing Guidance



Total: ~19 minutes presentation + 5 minutes Q&A

Key Messages to Emphasize

1. **Clinical Importance**
   Prolonged ICU stay prediction addresses real resource planning challenges
2. **Privacy Preservation**
   Federated learning achieves competitive performance while protecting patient data
3. **Practical Deployment**
   Working application demonstrates feasibility beyond academic exercise
4. **Community Contribution**
   Public benchmark supports reproducible research and advancement

Presenter Coordination: Jamie covers methodology (slides 4-9), Jim covers results (slides 10-17), shared conclusion.