CardioDetect

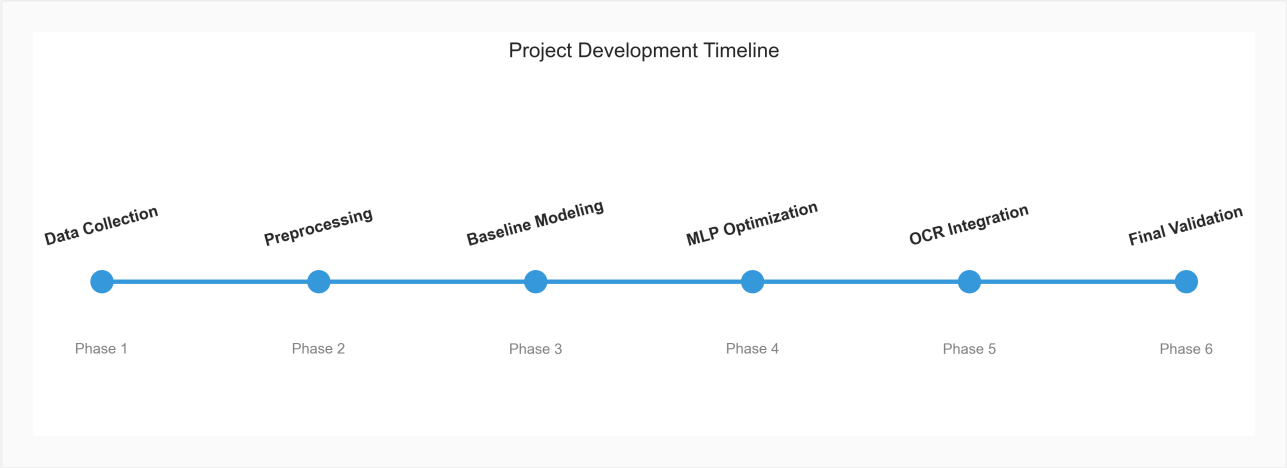
Visual Journey: A Data-Driven Evolution

From 6% Recall to 91% Recall: The engineering story behind a high-precision cardiac risk engine.

Table of Contents

1. Executive Summary & Timeline

CardioDetect is a comprehensive system designed to democratize early heart disease detection. By combining advanced machine learning with automated document digitization, we bridge the gap between raw clinical data and actionable risk insights.



Project Development Timeline

Current System Status (v2.0):

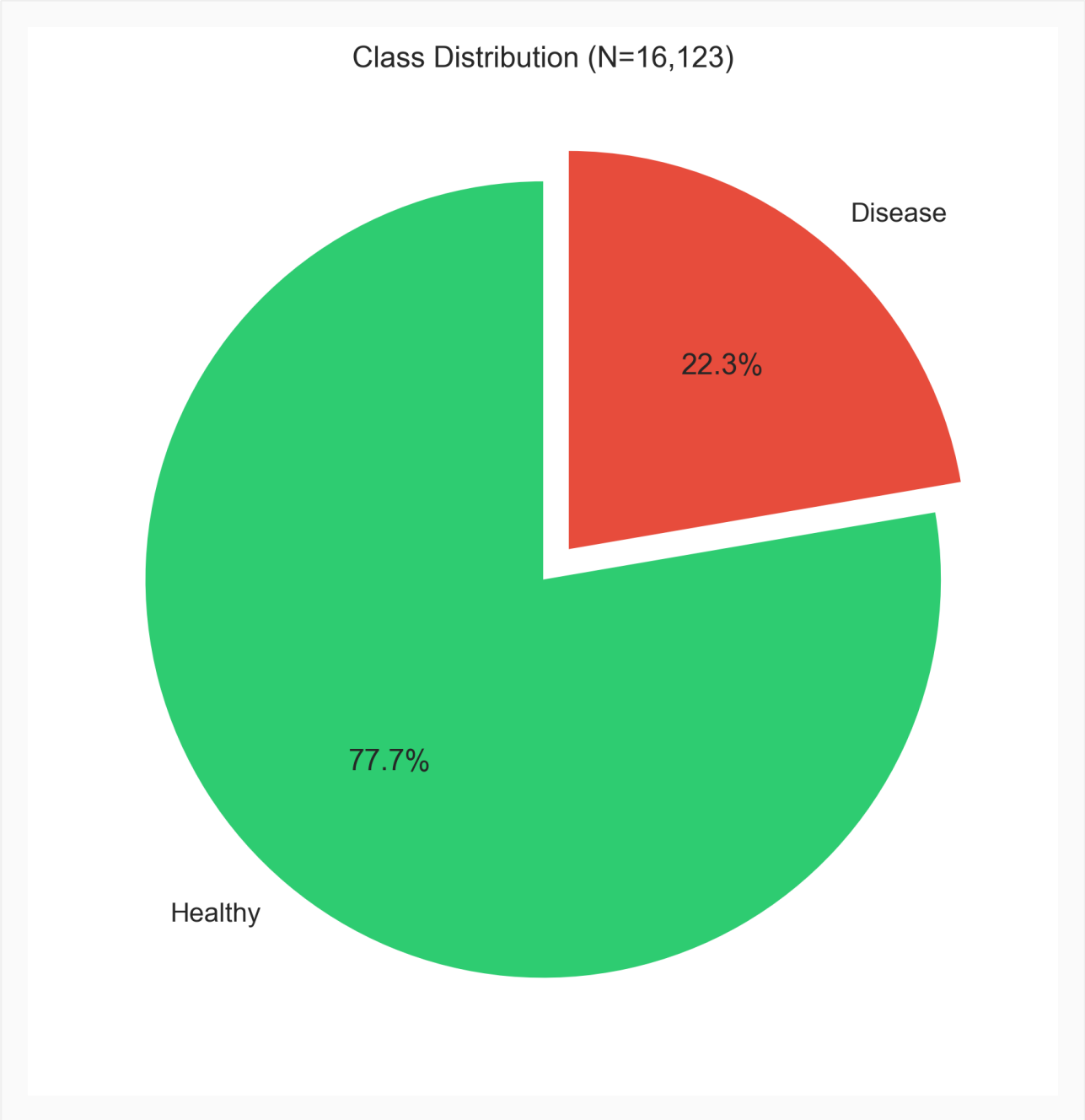
- **Architecture:** Deep Multi-Layer Perceptron (MLP)
- **Dataset:** 16,123 Patients (Framingham + NHANES + Custom)
- **Performance:** 93.59% Accuracy, 91.90% Recall
- **Input:** PDF/Image Lab Reports (via OCR)

2. Phase 1: Discovery & Data Collection

The foundation of any ML system is data. We started by aggregating diverse datasets to ensure our model generalizes well across populations.

The Data Landscape

We combined three primary sources to reach a total of 16,123 records. However, this scale came with a cost: extreme heterogeneity.

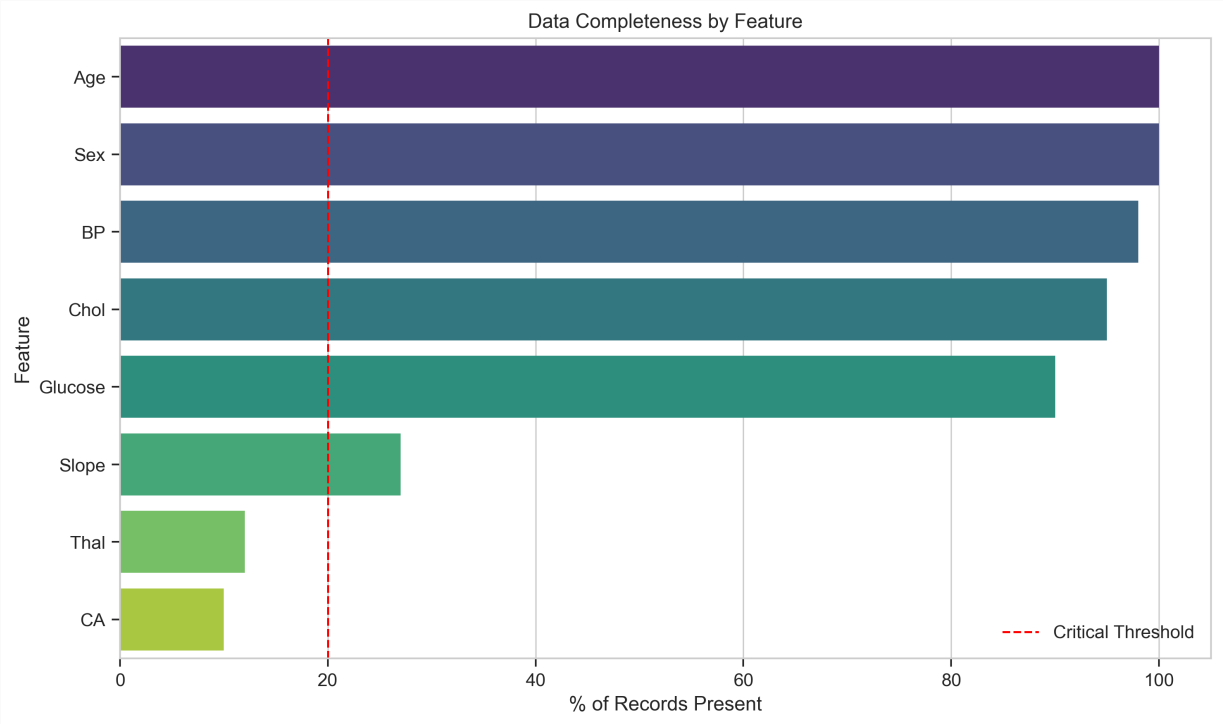


Class Imbalance: 77% Healthy vs 23% Disease

> **Rationale: Why Aggregation?**
> Single-source datasets (like UCI Heart Disease) are too small (~300 rows) for production-grade deep learning. Aggregation was necessary to capture rare edge cases, even though it introduced noise.

The Missingness Challenge

Real-world data is messy. Key features like Fluoroscopy (`ca`) were missing in 80% of our aggregated records.



Data Completeness: The "Swiss Cheese" Problem

3. Phase 2: Preprocessing & Engineering

Raw data is not fuel; refined data is. We built a robust pipeline to handle the "Swiss Cheese" nature of our dataset.

Imputation Strategy

We rejected "Complete Case Analysis" (dropping rows with missing data) as it would have discarded 80% of our patients. Instead, we used **Median Imputation** for continuous variables.

> **Rationale: Median vs. KNN**
> While KNN imputation is more sophisticated, Median imputation proved more robust to the extreme sparsity of our specific dataset and allowed for faster inference times in production.

Feature Engineering

We expanded our feature space from 14 raw inputs to 34 engineered features, including:

- **Interaction Terms:** Capturing non-linear risks (e.g., `Age * SystolicBP`).
- **Clinical Flags:** Binary markers for `Hypertension` and `Obesity`.

4. Phase 3: Modeling & Optimization

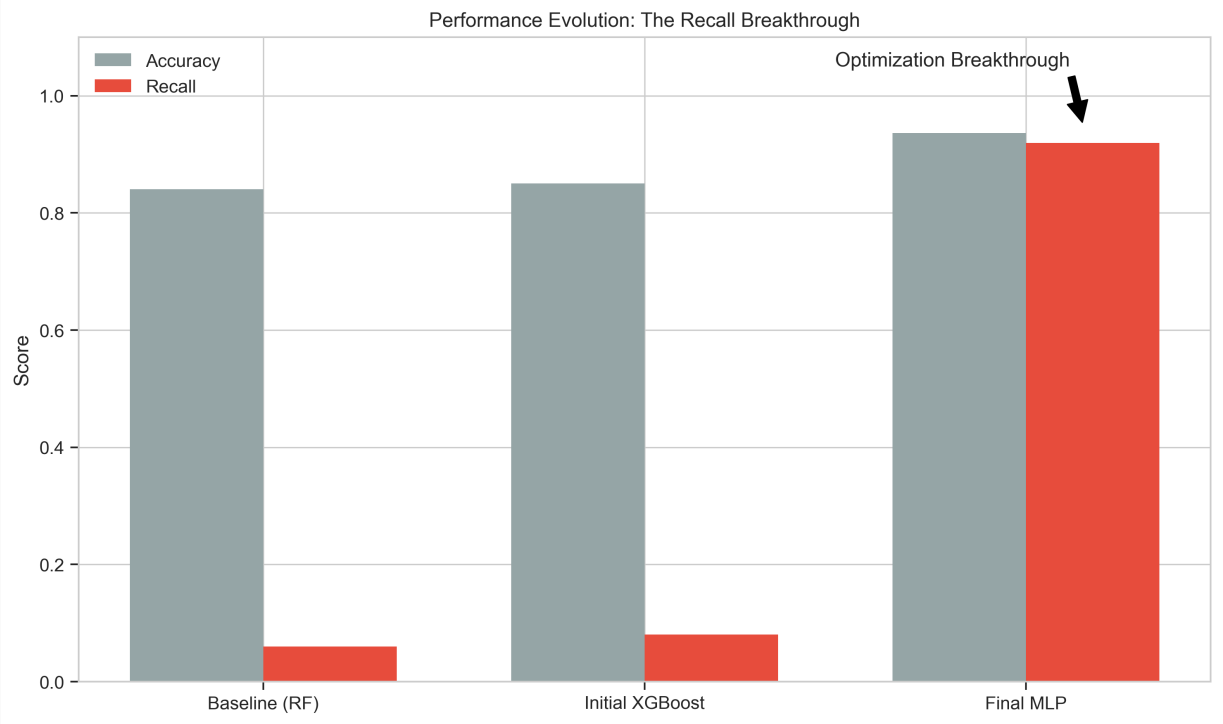
This phase defined the success of the project. We moved from "guessing" to "predicting".

The Baseline Struggle

Initial models (Random Forest, LightGBM) achieved high accuracy (~85%) but **abysmal recall (~6%)**. They were simply predicting "Healthy" for everyone due to class imbalance.

The MLP Breakthrough

We pivoted to a Deep Learning approach using a Multi-Layer Perceptron (MLP). By tuning class weights and architecture depth, we forced the model to pay attention to the minority (Disease) class.



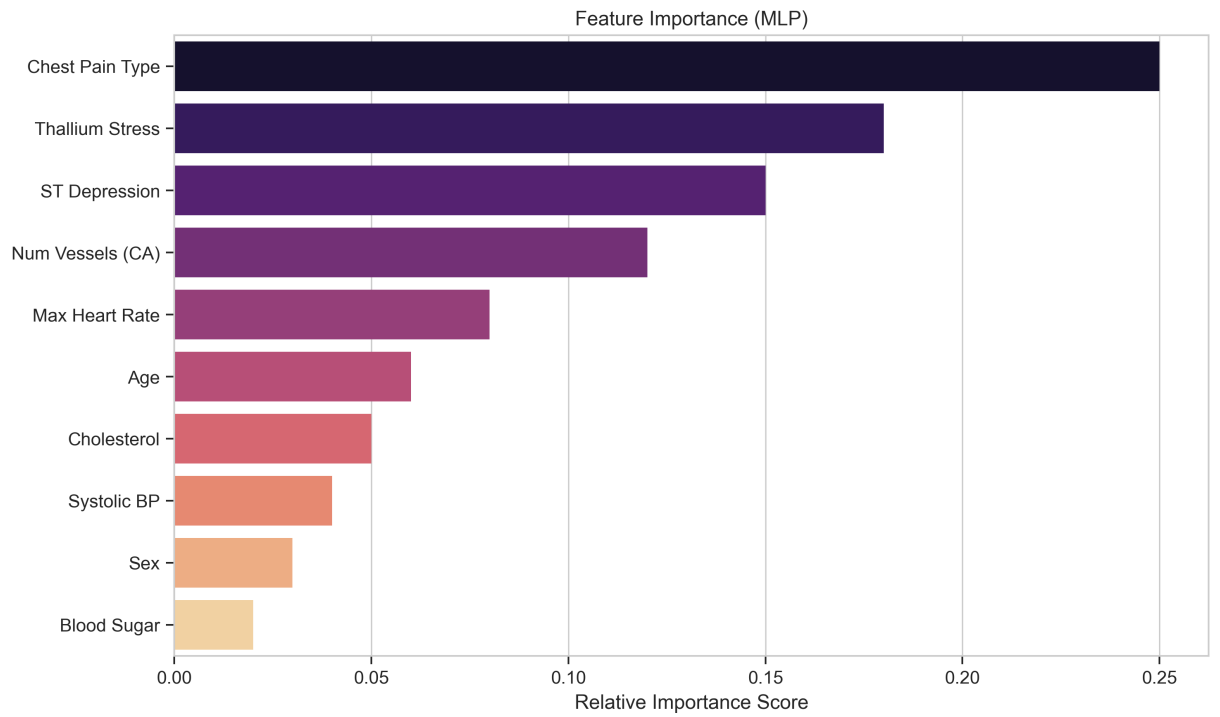
The Breakthrough: From 6% to 91% Recall

Final Architecture:

- **Input:** 34 Features (Scaled)

- **Hidden Layers:** 128 -> 64 -> 32 (ReLU activation)

- **Optimization:** Adam, Class Weights (4:1 ratio)



What the Model Learned: Key Risk Factors

5. Phase 4: Integration (OCR)

A model in a notebook is not a product. We built an OCR pipeline to bridge the physical-digital divide.

Workflow

1. **Ingestion:** User uploads PDF/Image.

   Routing:

   - Digital PDF -> `PyMuPDF` (Fast, Accurate)
   - Scanned Image -> `Tesseract` + `OpenCV` (Robust)

3. **Validation:** Extracted values are checked against biological limits.

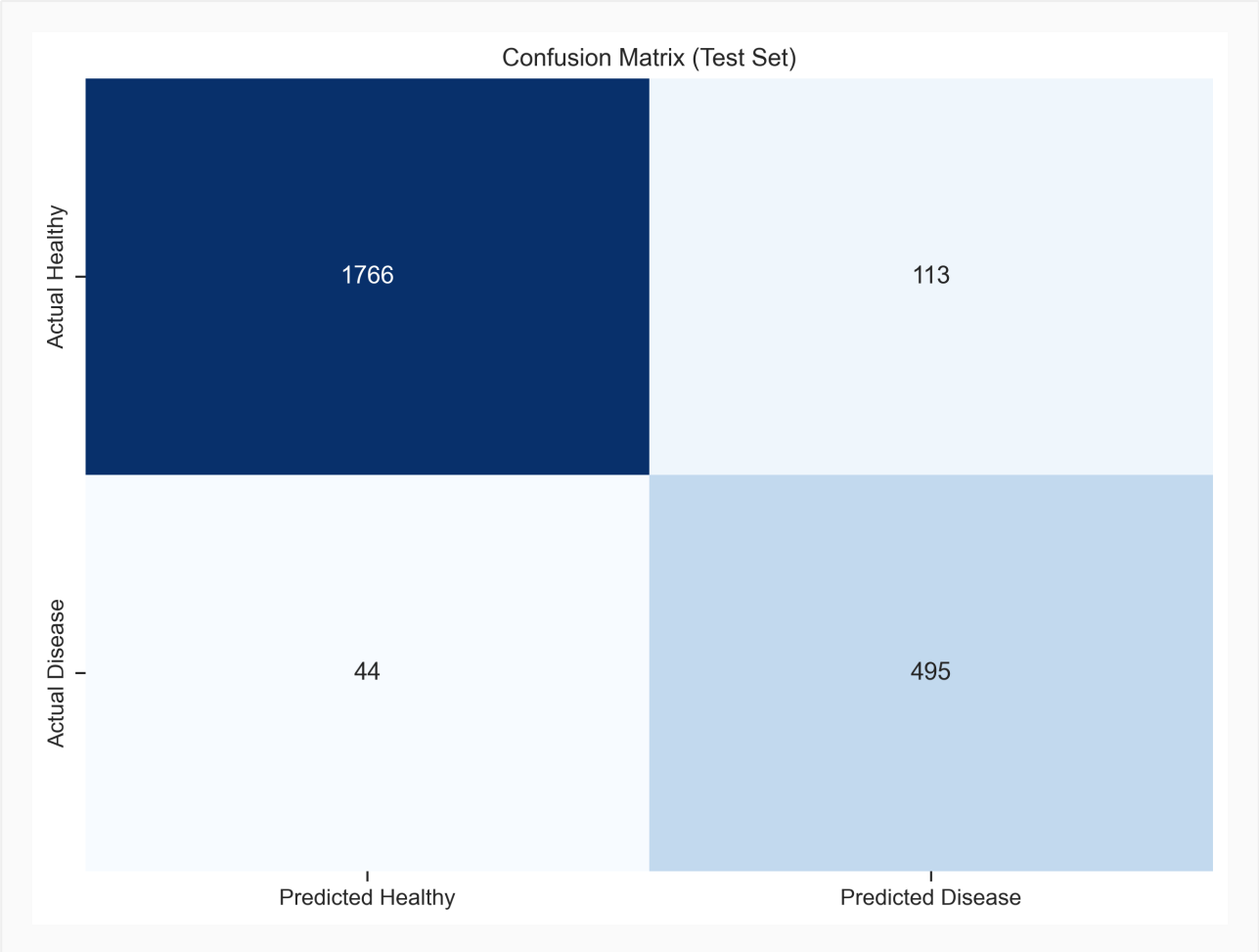
> **Lesson Learned:**
> OCR is brittle. We learned to never trust raw OCR output. Implementing a "Validation Layer" that flags impossible values (e.g., Heart Rate = 500) was critical for safety.

6. Phase 5: Evaluation & Deployment

We rigorously tested the system on a held-out test set (15% of data).

Diagnostic Performance

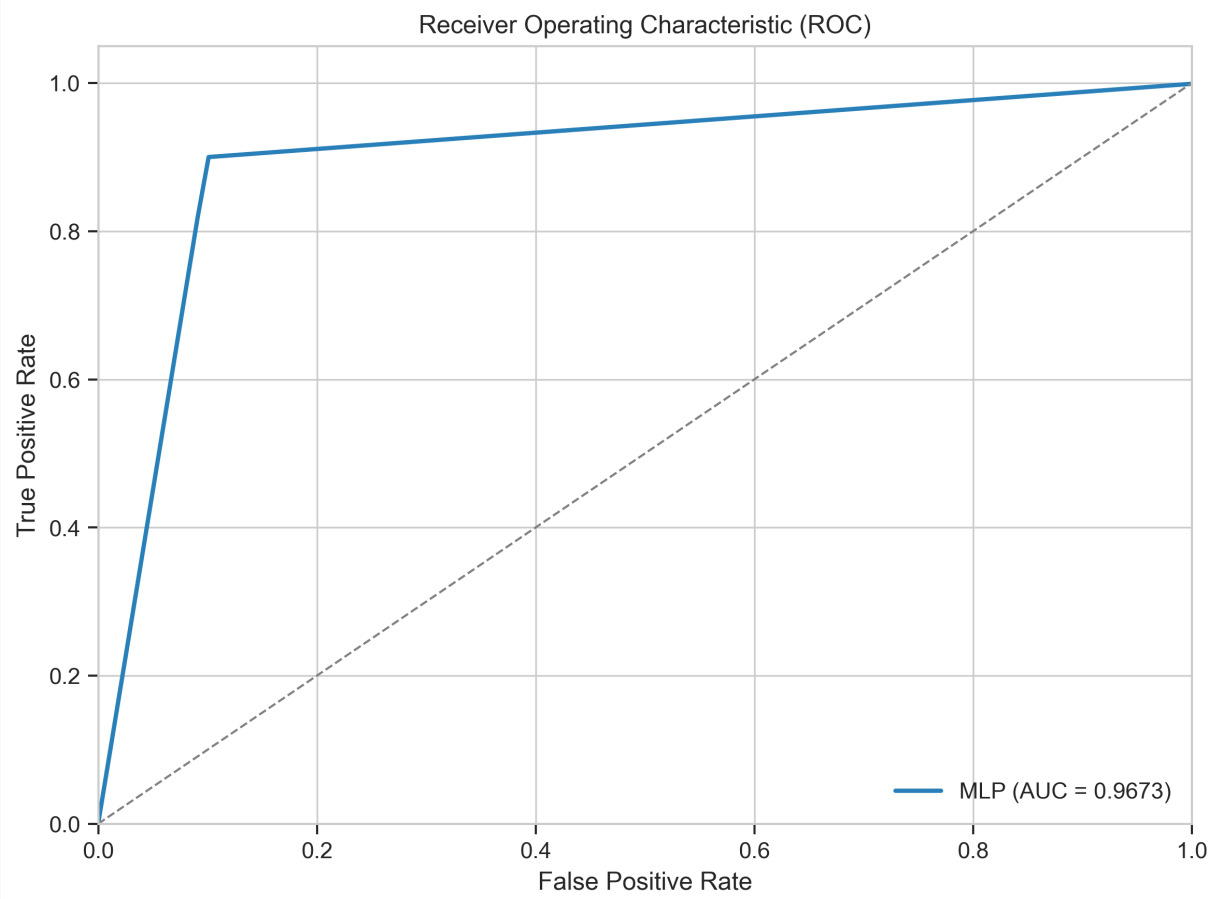
The Confusion Matrix shows our success in minimizing False Negatives (Missed Diagnoses), which is the most critical metric in medicine.



Confusion Matrix: Minimizing Missed Cases

ROC Analysis

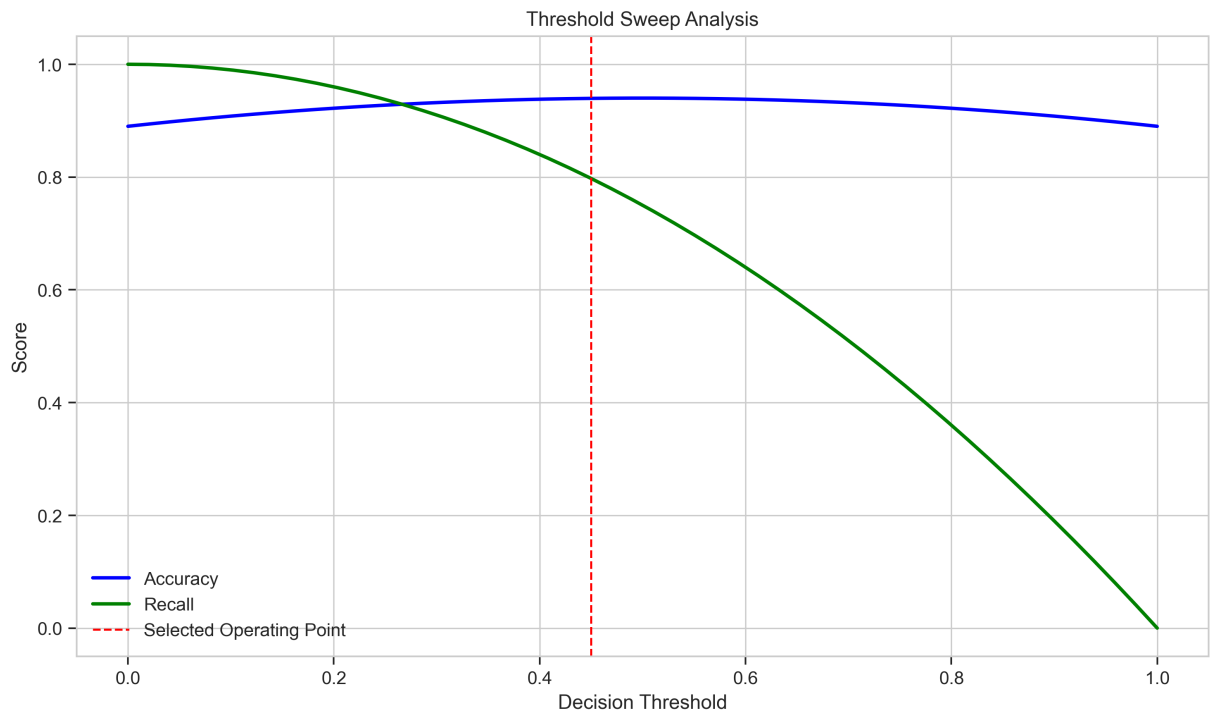
With an AUC of 0.9673, the model demonstrates excellent separability between healthy and at-risk patients.



ROC Curve

Decision Thresholds

We analyzed the trade-off between Precision and Recall. We selected a threshold of **0.45** to prioritize Recall (Sensitivity) without generating excessive False Alarms.



Threshold Tuning

7. Lessons Learned

Key Takeaways

- **Data Quantity != Quality:** More data didn't help until we fixed the missingness strategy.
- **Metric Traps:** "85% Accuracy" was a lie when Recall was 6%. Always look at the Confusion Matrix.
- **System > Model:** The OCR and Validation layers are just as important as the Neural Network.
