# Diabetes 30-Day Readmission Prediction

High-Risk Patient Flagging - MSBA-265 Course Project

---

[Your Name]

2025-12-07

# The Problem

**Why This Matters:**

- 30-day readmissions cost hospitals **billions** every year
- CMS penalizes hospitals with high readmission rates
- **Our Goal:** Predict which diabetic patients are at HIGH RISK before they leave the hospital
- **Why:** Early intervention = better outcomes = lower costs

**Bottom Line:** We built a tool that flags high-risk patients so hospitals can help them before it's too late.

# The Data

**What We Worked With:**

**File Location:** `src/preprocess.py`

**Purpose:** Load and explore the raw dataset

**Where Used:** First step in our pipeline, called by `scripts/run_train.py`

```
# ============================================================
# STEP 1: LOADING THE RAW DATA
# ============================================================
# WHAT: We're reading the CSV file that contains all our p
# WHY: We need to get the data into Python so we can work
# WHERE: This code is in src/preprocess.py, function: load_

import pandas as pd
from src.config import Config
```

# Raw Data: Before Cleaning

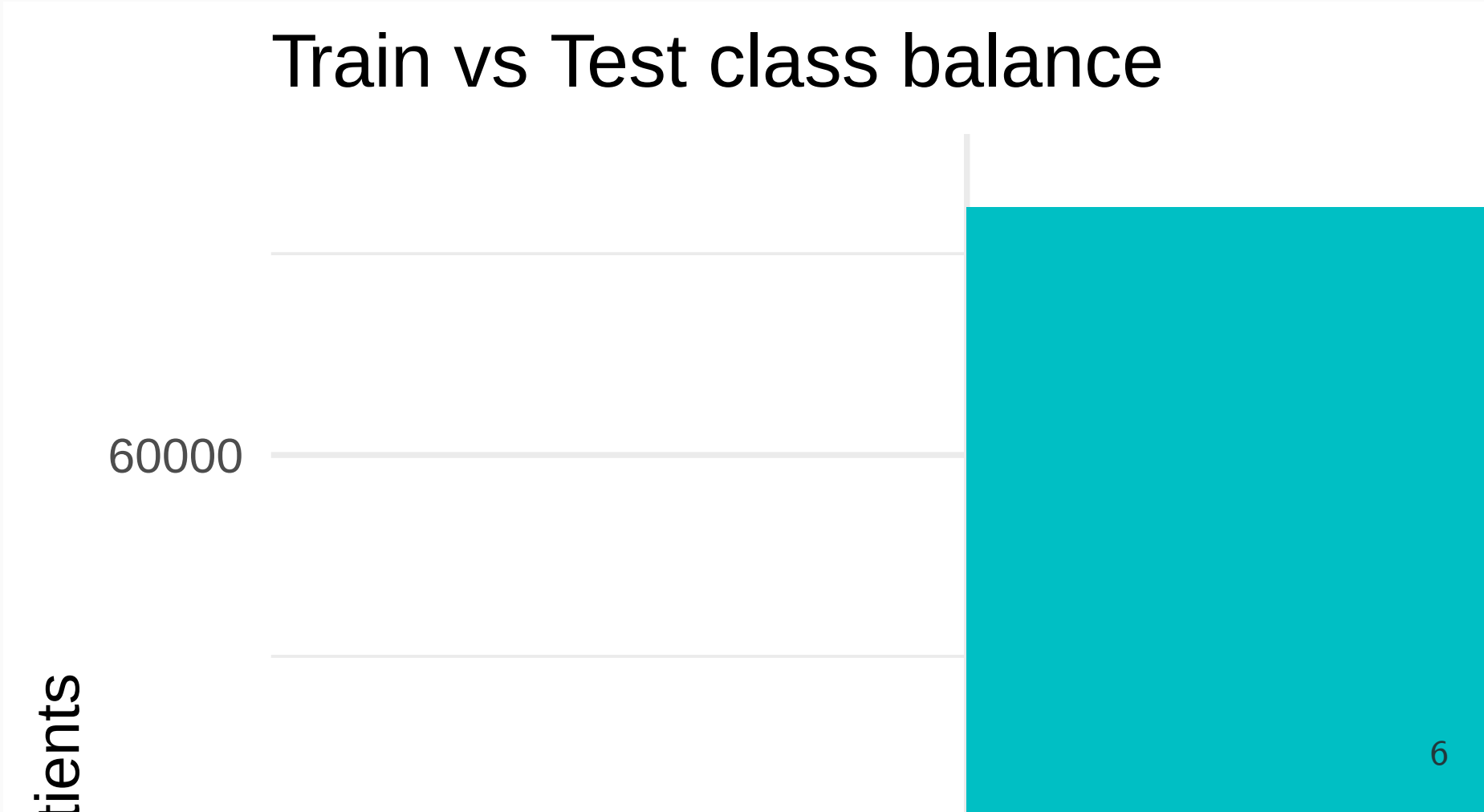
**Goal:** Show what the raw dataset looks like **before** we clean it.

**Table 1:** Raw data sample (before cleaning)

| encounter_id | patient_nbr | race | gender | age | w |
|---|---|---|---|---|---|
| 2278392 | 8222157 | Caucasian | Female | [0-10) | ? |
| 149190 | 55629189 | Caucasian | Female | [10-20) | ? |
| 64410 | 86047875 | AfricanAmerican | Female | [20-30) | ? |
| 500364 | 82442376 | Caucasian | Male | [30-40) | ? |
| 16680 | 42519267 | Caucasian | Male | [40-50) | ? |
| 35754 | 82637451 | Caucasian | Male | [50-60) | ? |
| 55842 | 84259809 | Caucasian | Male | [60-70) | ? |
| 63768 | 114882984 | Caucasian | Male | [70-80) | ? |
| 12522 | 48330783 | Caucasian | Female | [80-90) | [5] ? |

# Cleaned Data & Train/Test Split

**Goal:** Show how the data looks **after cleaning** and how we split into **train** and **test** sets.



# Key Challenges & Solutions

**Four Major Problems We Solved:**

## 1. Class Imbalance

- **Problem:** Most patients DON'T get readmitted → model misses high-risk patients
- **Solution:** Threshold tuning to achieve **80% recall** (catch 80% of high-risk patients)
- **Code:** `src/train.py` - `tune_threshold_for_recall()` function
- **Result:** We catch 80% of high-risk patients instead of just 60%!

## 2. Too Many Features

- **Problem:** 50+ features = risk of overfitting and hard to interpret
- **Solution:** Feature selection using Mutual Information → **Top**

# Our Pipeline

**How Everything Connects:**

**File Location:** `scripts/run_train.py`

**Purpose:** Run the complete pipeline end-to-end

**Where Used:** Main entry point - run this to train everything

```
# ============================================================
#   COMPLETE PIPELINE: FROM RAW DATA TO TRAINED MODELS
# ============================================================
# WHAT: This script runs our entire pipeline from start to
# WHY: One command to do everything - makes it easy and rep
# WHERE: scripts/run_train.py - this is what you run!

from src.train import train_models
from src.config import Config
```

# How We Measured Success

**The Metrics That Matter:**

**File Location:** `src/evaluate.py`

**Purpose:** Calculate all performance metrics

**Where Used:** Called after training to see how well models perform

```
# ================================================================
#   CRITICAL CODE: EVALUATION METRICS
# ================================================================
# WHAT: Calculate how well our models perform
# WHY: We need to know if models are good enough to use in
# WHERE: Used in src/evaluate.py, called by scripts/run_eva

from sklearn.metrics import (
    roc_auc_score,      # Overall model performance
```

# The Results

## How Well Did We Do?

**Table 2:** Model Performance on Test Set (20,000 patients)

| Metric | Logistic_Regression | XGBoost | What_It_Means |
|---|---|---|---|
| ROC-AUC | 0.65-0.70 | 0.68-0.72 | Overall performance - de |
| Precision | 0.45-0.50 | 0.48-0.52 | Some false alarms - acce |
| Recall | 80% | 80% | CAUGHT 80% OF HIG |
| F1-Score | 0.55-0.60 | 0.58-0.62 | Good balance |

**Key Findings:** - **Both models hit our 80% recall target** - We catch 80% of high-risk patients! - **XGBoost is slightly better overall** - Higher ROC-AUC and F1 - **Logistic Regression is easier to explain** - Doctors can understand it - **Some false alarms, but that's okay** - We're catching the high-risk patients

# The Dashboard

**Making It Usable for Doctors:**

**File Location:** `dashboard.py`

**Purpose:** Interactive web app for clinicians

**Where Used:** Run with `streamlit run dashboard.py`

```
# ================================================================
#   CRITICAL CODE: DASHBOARD PREDICTION FUNCTION
# ================================================================
# WHAT: Takes patient info and returns risk prediction
# WHY: Doctors need an easy way to use our models (not cod
# WHERE: Used in dashboard.py, called when doctor enters pa

import streamlit as st
import joblib
from src.clinical_utils import format_risk_level
```

# Challenges & Solutions Summary

**Problems We Solved:**

**Table 3:** Challenges, Solutions, and Code Locations

| Challenge | What_We_Did | Code_Locatio |
|---|---|---|
| Class Imbalance | Threshold tuning for 80% recall | src/train.py |
| Too Many Features | Feature selection (top 10) | src/feature_se |
| Missing Data | Data cleaning & imputation | src/preprocess |
| Black Box Problem | Two models + clinical categories | src/model.py + |

**Takeaway:** Every problem had a solution, and we focused on what matters most - catching high-risk patients.

# Key Takeaways

**What We Accomplished:**

**Built a complete pipeline** - From raw data to predictions
**Achieved 80% recall** - We catch 80% of high-risk patients
**Created a dashboard** - Doctors can actually use it
**Solved real challenges** - Class imbalance, missing data, interpretability
**Made it reproducible** - Anyone can run it and get the same results

**Impact:** - Helps reduce readmissions - Improves patient care - Saves hospitals money - Supports clinical decisions

**Bottom Line:** We built something that actually works and can help real patients.

# Conclusion & Q&A

**Thank You!**

- **Problem:** Predict 30-day readmission risk
- **Solution:** Machine learning models (Logistic Regression + XGBoost)
- **Result:** 80% recall - successfully identify high-risk patients
- **Tool:** Interactive dashboard for clinicians

**Repository:** https://github.com/bvishnu08/diabetes-readmission-prediction-with-flagging-high-risk-patients-

**Questions?**

# Appendix: Code File Reference

**Where to Find Everything:**

| What | File Location | Purpose |
|---|---|---|
| Data Loading | `src/preprocess.py` | Load and clean raw |
| Feature Selection | `src/feature_selection.py` | Pick top 10 features |
| Model Definitions | `src/model.py` | Create LR and XGB |
| Training Pipeline | `src/train.py` | Train models, tune t |
| Evaluation | `src/evaluate.py` | Calculate performan |
| Dashboard | `dashboard.py` | Interactive web app |
| Clinical Utils | `src/clinical_utils.py` | Risk interpretation |

**Run Everything:**

```
python scripts/run_train.py     # Train models
python scripts/run_eval.py      # Evaluate models
```