

- Research-Grade Fraud Detection System (FYP 2025)
 - Overview
 - 🚀 Key Features
 - 1. Advanced Neural Network Architecture
 - 2. Comprehensive Diagnostics Pipeline (09_comprehensive_diagnostics.ipynb)
 - 3. Explainable AI (XAI)
 - 📈 Model Performance
 - 📁 Project Structure
 - 🛠 Usage Guide
 - 1. Environment Setup
 - 2. Running the Pipeline
 - 📈 Diagnostics Report
 - ⚠ Notes for Evaluators

Research-Grade Fraud Detection System (FYP 2025)

Overview

This project implements a robust, academically defensible, and explainable **Credit Card Fraud Detection System**. It addresses the critical challenges of extreme class imbalance (0.17% fraud) and high-stakes decision-making using a methodologically rigorous pipeline.

Key highlights include a **refactored Neural Network** with Binary Focal Loss, a **12-Step Diagnostics Pipeline**, and full **Model Explainability** using SHAP and LIME.

🚀 Key Features

1. Advanced Neural Network Architecture

We moved beyond standard implementations to design a Fraud-Specific Neural Network:

- **Loss Function: Binary Focal Loss** ($\gamma = 2.0, \alpha = 0.25$) to down-weight easy negatives and focus training on hard-to-classify fraud cases.
- **Architecture:** Simplified MLP (64 → 32 units) with strong regularization (L2 + Dropout) to prevent overfitting on tabular data.
- **Optimization:** Adam optimizer with Class-Weighted training removed in favor of Focal Loss for better stability.

2. Comprehensive Diagnostics Pipeline (09_comprehensive_diagnostics.ipynb)

A production-grade validation suite ensuring model reliability:

- **Data Integrity:** Automated checks for drift and imbalance.
- **Stability:** Cross-validation and learning curve analysis.
- **Business Impact:** Cost-sensitive threshold optimization.
- **Calibration:** Assessment of predicted probabilities vs. actual risk.

3. Explainable AI (XAI)

To ensure regulatory compliance and trust:

- **SHAP (Global):** Quantifies feature impact across the entire dataset.
- **LIME (Local):** Explains *individual* fraud predictions for human review.



Model Performance

Model	ROC-AUC	PR-AUC	Test F1 (Fraud)	Key Characteristic
XGBoost	0.981	0.865	0.84	Best overall performance; excellent precision-recall balance.
Random Forest	0.975	0.850	0.82	Robust baseline; highly interpretable.

Model	ROC-AUC	PR-AUC	Test F1 (Fraud)	Key Characteristic
Neural Network	0.965	0.820	0.79	Tailored for sensitivity; stable training via Focal Loss.
Logistic Reg	0.940	0.760	0.11	High recall but very low precision (many false alarms).

Note: Metrics primarily optimized for PR-AUC due to extreme imbalance.

📁 Project Structure

```
.
├── models/                      # Saved models (Keras .h5, Joblib .pkl)
├── notebooks/                   # Interactive Analysis & Training
│   ├── 01_data_loading_and_eda.ipynb    # Initial Exploration
│   ├── 02_data_preprocessing.ipynb      # Scaling & Split
│   ├── 06_neural_network.ipynb         # Refactored NN Training
│   ├── 07_model_evaluation.ipynb       # Comparative Analysis
│   ├── 08_explainability.ipynb        # SHAP & LIME
│   └── 09_comprehensive_diagnostics.ipynb # 12-Step Validation Suite
└── results/
    ├── diagnostics/                 # Automated Reports & Figures
    │   ├── figures/                  # 12+ Generated diagnostic plots
    │   └── diagnostics_report.xlsx  # Consolidated metrics
    └── metrics/                    # Raw JSON logs
└── requirements.txt              # Python dependencies
```

🔧 Usage Guide

1. Environment Setup

```
# Create virtual environment
python -m venv venv
venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt
```

2. Running the Pipeline

The project is modular. You can run individual notebooks, but the recommended flow is:

- 1. Train Neural Network:** Run `06_neural_network.ipynb` to train the advanced model with Focal Loss.
 - 2. Evaluate All Models:** Run `07_model_evaluation.ipynb` to compare NN, XGBoost, and RF.
 - 3. Run Diagnostics:** Run `09_comprehensive_diagnostics.ipynb` to generate the full PDF-ready report and Excel metrics.
 - 4. Explain Predictions:** Run `08_explainability.ipynb` to analyze specific fraud cases.
-



Diagnostics Report

The `09_comprehensive_diagnostics.ipynb` notebook automatically generates:

- Drift Analysis:** Checking if train/test data distributions match.
- Learning Curves:** Diagnosing bias vs. variance.
- Cost Analysis:** Estimating financial loss at different thresholds.

Output Location: `results/diagnostics/`



Notes for Evaluators

- Class Imbalance:** We explicitly avoided over-reliance on SMOTE for the Neural Network, proving that custom loss functions (Focal Loss) provide superior stability for this specific architecture.
- Metric Selection:** We prioritize **PR-AUC (Precision-Recall AUC)** over ROC-AUC, as ROC can be misleadingly optimistic in highly imbalanced fraud datasets.