Credit Card Fraud Detection System

Introduction

In today's digital economy, credit card fraud has become a growing concern, impacting individuals and financial institutions alike. This project aims to detect fraudulent credit card transactions using machine learning techniques. Unlike theoretical implementations, this was developed as a real-world, end-to-end system with a responsive interface, allowing both real-time and batch processing of transactions.

Abstract

The project utilizes the widely used Kaggle Credit Card Fraud Detection Dataset, which contains anonymized features resulting from a PCA transformation and includes severe class imbalance-only 0.17% of transactions are fraudulent. The challenge was not only in model selection but also in ensuring that the model doesn't ignore rare fraud cases while avoiding false alarms.

To solve this, multiple steps were taken-from robust preprocessing to model training, hyperparameter tuning, and deploying the solution using Streamlit. SHAP values were integrated for interpretability, ensuring transparency in each prediction.

Tools Used

- Python (Pandas, NumPy, Scikit-learn, XGBoost, LightGBM, CatBoost)
- Jupyter Notebook
- Streamlit (for web app interface)
- SHAP (for explainable AI)
- Matplotlib & Seaborn (for visualizations)
- SMOTE (to handle class imbalance)
- Joblib (for saving models and scalers)
- GitHub + Streamlit Cloud (for deployment)

Steps Involved in Building the Project

1. Data Understanding & Cleaning
   - Used the Kaggle dataset with anonymized features.
   - Visualized and analyzed key distributions (Time, Amount).

2. Feature Engineering
   - Scaled Time and Amount separately using RobustScaler.

- Dropped original unscaled versions to reduce skew impact.

3. Handling Class Imbalance
   - Applied SMOTE to oversample the minority (fraudulent) class.
   - Ensured the model would not be biased toward the majority class.

4. Model Building & Evaluation
   - Trained several models including Logistic Regression, Decision Tree, Random Forest.
   - Chose an ensemble Voting Classifier combining XGBoost, LightGBM, and CatBoost.
   - Tuned for high recall to minimize false negatives.

5. Interpretability with SHAP
   - Integrated SHAP values to explain each prediction (Waterfall & Bar plots).

6. Deployment
   - Built a Streamlit app with options for real-time prediction and batch processing.
   - Added dynamic visualizations post-prediction.

Conclusion

This project bridges the gap between machine learning experimentation and production-level deployment. It not only addresses a critical real-world problem - credit card fraud - but also focuses on interpretability, recall-focused evaluation, and usability. The app allows users to interact with the model meaningfully and transparently, which is crucial in high-risk applications like fraud detection.

The full system is live and ready for demonstration and further enhancement.