Credit Card Fraud Detection

A Business Analytics Project

Group 25
Tran Quang Trong 20235565
Pham Quoc Thai 20235558
Nguyen Khac Viet Anh 20235471
Le Thanh Vinh 20200668

December 20, 2025

Contents

1 Executive Summary

This report presents a business-focused analysis of a credit card fraud detection project using historical transaction data from European cardholders. The dataset contains 284,807 transactions made during a two-day period in September 2013, with only 492 labeled as fraudulent (approximately 0.172% of all transactions). This extreme imbalance reflects a typical real-world scenario: fraud events are rare but can generate disproportionately high financial and reputational losses.

From a business perspective, the project pursues three major objectives:

- Understand behavioral patterns of fraudulent versus legitimate transactions, especially in terms of transaction amount and time-of-day activity.
- Build a fraud detection model that achieves high recall on fraudulent transactions while keeping false positives at a manageable level, to avoid unnecessary friction for legitimate customers.
- Deploy the model into a simple, interactive user interface (UI) that allows analysts or operations teams to upload transaction files, obtain fraud risk scores, and review the most suspicious transactions.

Using a combination of feature engineering (including time-of-day features and anomaly-based risk scores), Logistic Regression, and an XGBoost gradient boosting model, the solution achieves strong performance on a time-based hold-out test set:

- Logistic Regression: fraud recall of approximately 87.9%, with ROC AUC around 0.95.
- XGBoost: more balanced precision–recall trade-off for the fraud class, with ROC AUC around 0.96 on the test set.

Finally, the best-performing model (XGBoost) is integrated into a `Streamlit` web application. The app allows end users to upload raw CSV transaction data (with features matching the original dataset), automatically performs all necessary feature engineering, and returns both fraud probabilities and binary flags, together with summary metrics and a ranked list of the most suspicious transactions.

2 Business Problem and Objectives

2.1 Business Context

Credit card fraud presents a persistent threat to financial institutions. Even though fraudulent transactions represent a very small fraction of total transaction volume, each case can result in direct financial losses, chargeback costs, and damage to customer trust. Traditional rule-based systems often struggle to keep up with evolving fraud tactics and may either miss sophisticated fraud attempts or produce excessive false alarms.

In this context, we analyze a real-world dataset of card-present transactions recorded over a two-day period. The data is highly imbalanced, with only 492 frauds out of 284,807 transactions (0.172%). The challenge is to detect as many fraudulent transactions as possible, while minimizing disruption to legitimate customers.

2.2 Business Objectives

From a business analytics perspective, the project is designed to achieve the following objectives:

1. **Understand fraud behavior**: Identify temporal and behavioral patterns that differentiate fraudulent from normal transactions (for example, time-of-day and anomaly patterns).
2. **Develop a robust detection model**: Train and evaluate machine learning models capable of providing a fraud risk score per transaction, with a focus on high recall for the fraud class.
3. **Enable operational use**: Package the detection capability into an interactive tool that can be used by risk analysts or operational teams to inspect new transaction batches and prioritize investigations.

3 Data Description and Preparation

3.1 Dataset Overview

The dataset consists of credit card transactions made by European cardholders in September 2013. The key characteristics of the dataset are:

- Time span: two consecutive days of transaction activity.
- Number of transactions: 284,807.
- Number of fraudulent transactions: 492 (0.172% of total).
- Input variables: 30 features, including:
  - **V1–V28**: numerical features obtained via Principal Component Analysis (PCA) on the original, confidential attributes.

  - **Time**: the number of seconds elapsed between each transaction and the first transaction in the dataset.
  - **Amount**: the transaction amount.

- Target variable:
  - **Class**: 1 for fraudulent transactions, 0 for non-fraudulent transactions.

Because the original feature names and meanings are not accessible, most behavioral interpretation must rely on the engineered features, the raw `Time` and `Amount` fields, and aggregate patterns rather than individual PCA dimensions.

3.2 Train–Test Split Strategy

To better approximate a real-world production setting, the dataset is split into training and testing sets based on time order instead of random shuffling:

- **Training set** (`df_train`): first half of the transactions (approximately the first day).
- **Test set** (`df_test`): second half of the transactions (approximately the second day).

This temporal split ensures that all model training is performed on past data, and evaluation is conducted on future data, which better reflects how a model would operate in a live environment.

The number of frauds in each subset is:

- Training set: 269 frauds.
- Test set: 223 frauds.

3.3 Handling Missing Values and Data Quality

Both training and testing subsets are checked for missing values. The dataset contains no missing values in either set, and thus no explicit imputation is required.

In the deployment pipeline, for robustness, numeric columns in uploaded CSV files are filled with zero for missing values before processing; however, this is a defensive design choice rather than a requirement of the original dataset.

3.4 Feature Scaling

Because the PCA components `V1-V28` are derived from standardized variables, they are already approximately on the same scale. However, the raw `Amount` feature is highly skewed and has a much larger range compared to the PCA components.

To avoid letting very large transaction amounts dominate distance-based or gradient-based models, the `Amount` feature is scaled using the **RobustScaler**:

- RobustScaler uses the median and interquartile range (IQR), making it resistant to extreme outliers that are common in financial transaction data.
- The scaled feature is stored as `scaled_amount`, and the original `Amount` column is dropped from the training and test sets.

The trained `RobustScaler` object is saved to disk (`robust_scaler.pkl`) so that the same transformation can be applied consistently during model evaluation and later in the deployment UI.

4 Feature Engineering and Exploratory Analysis

4.1 Transaction Amount Distribution

An initial exploratory data analysis reveals that the transaction amount in the training set is extremely right-skewed, with a long tail of large transactions. This is typical for financial data: most transactions are small, while a few transactions involve much larger amounts.

A boxplot of transaction amount by class (fraud vs. non-fraud) shows that:

- Fraudsters in this dataset do not systematically use unusually large amounts.
- Many fraudulent transactions have relatively small or moderate amounts, which helps them blend into normal, everyday spending patterns.

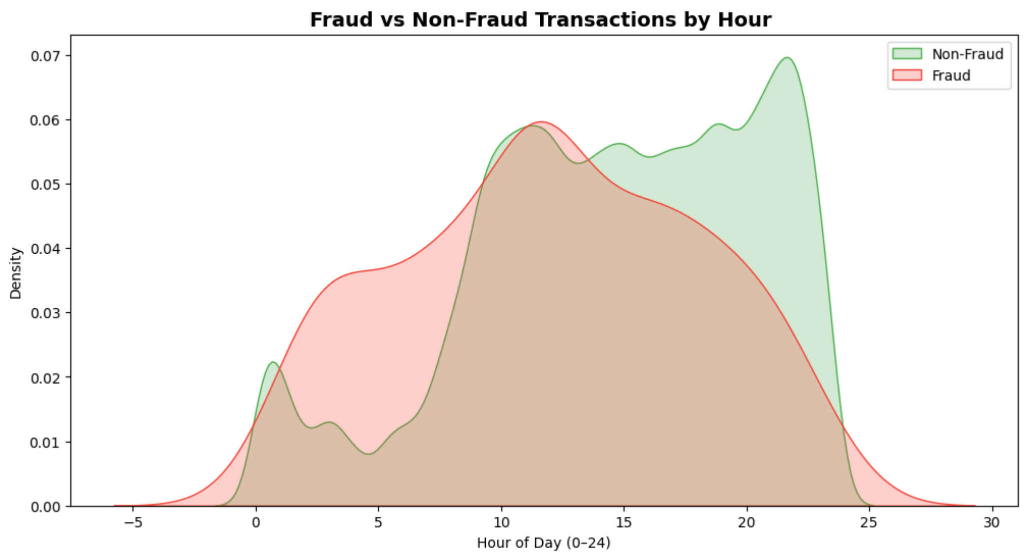
From a business perspective, this insight suggests that relying solely on very high amounts as a rule for fraud detection would miss a substantial portion of fraudulent activity. Instead, more subtle patterns and combinations of features need to be considered.

4.2 Time-of-Day Features

The original `Time` feature measures the elapsed seconds since the first transaction in the dataset, which is not directly interpretable from a behavioral standpoint. To extract more meaningful patterns, **hour-of-day** is computed as:

$$\text{hour\_of\_day} = \left(\frac{\text{Time}}{3600}\right) \bmod 24.$$

Kernel density plots for `hour_of_day` by class show the following patterns:



- Between approximately **1 AM and 5 AM**, the density of fraudulent transactions increases while legitimate transaction activity decreases.
- From around **8 AM to 8 PM**, fraud density remains elevated and roughly follows the trend of normal transactions, indicating that fraudsters also operate during normal business hours, likely to camouflage their behavior.

These patterns support the intuition that fraud risk can be time-dependent:

- Off-peak hours may be attractive for fraudsters due to lower monitoring and fewer manual reviews.
- During peak hours, fraud may be disguised within heavy legitimate traffic.

4.2.1 Cyclical Encoding of Time

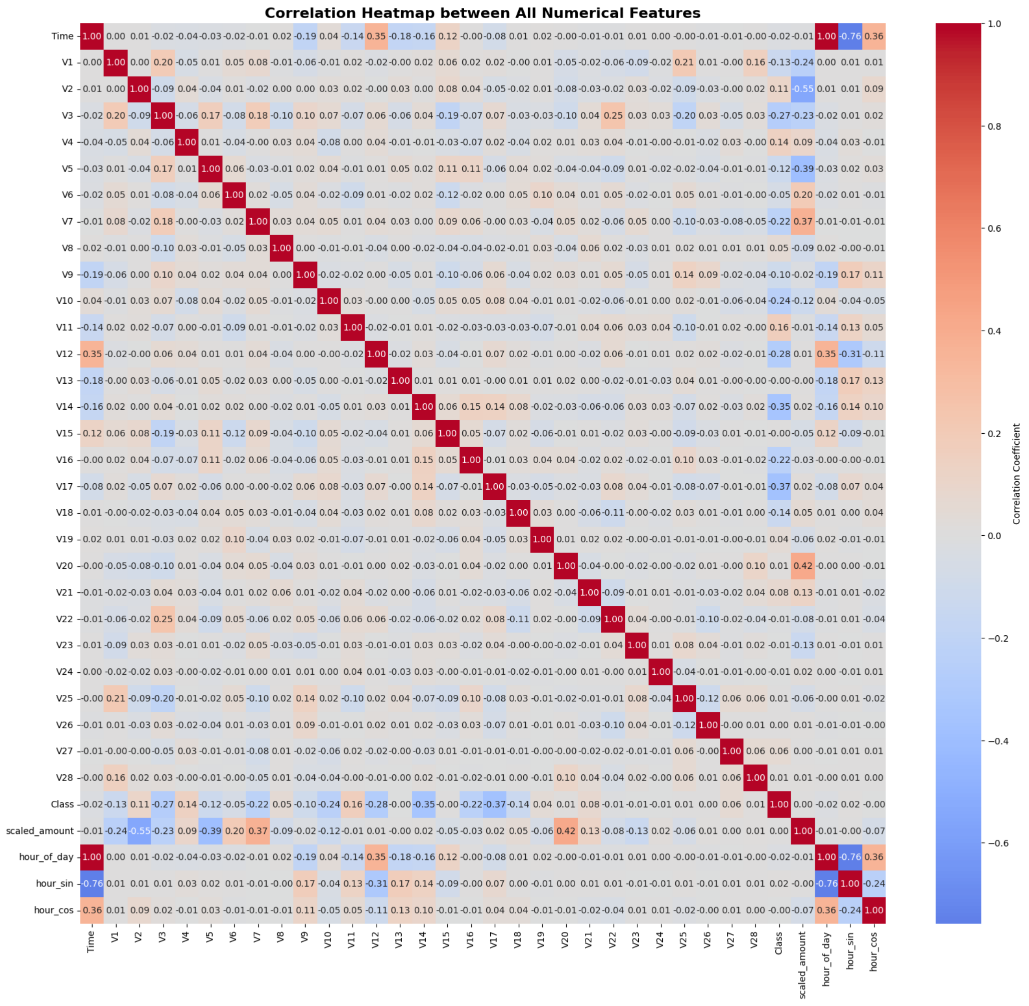
Hour-of-day is a cyclic variable: after 23:00 comes 00:00, and hours near the boundary (e.g., 23 and 0) should be considered close. Using raw numeric values $[0, 23]$ introduces artificial discontinuities. To address this, the following cyclical encoding is used:

$$\text{hour\_sin} = \sin\left(2\pi \cdot \frac{\text{hour\_of\_day}}{24}\right), \quad \text{hour\_cos} = \cos\left(2\pi \cdot \frac{\text{hour\_of\_day}}{24}\right).$$

This representation captures the circular nature of time-of-day and provides a more faithful geometric structure for models based on distances or linear combinations.

Correlation analysis shows that `hour_sin` provides sufficient information and is strongly related to hour-of-day patterns, while `hour_cos` is less critical in this context. As a result, `hour_cos` and the intermediate `hour_of_day` column are dropped before model training, leaving `hour_sin` as the main time-of-day feature.

4.3 Correlation Structure of PCA Features



A correlation heatmap of all numerical features in the training data reveals:

- The PCA-generated components `V1-V28` exhibit correlations close to zero with each other, as expected, reflecting minimal multicollinearity.
- Several components, notably `V3, V7, V10, V12, V14, V16, V17`, show stronger relationships with the target class and with each other in terms of capturing anomalous behavior.

These observations motivate their use as a subset of features for anomaly detection, described in the next subsection.

4.4 Anomaly-Based Risk Score (Isolation Forest)

To complement the supervised classification models, an anomaly detection approach is employed using an **Isolation Forest**. The method is fitted on a subset of PCA components:

$$\{V3, V7, V10, V12, V14, V16, V17\}.$$

The Isolation Forest assigns an anomaly score to each transaction; this score is transformed into a **risk score** feature:

$$\text{risk\_score\_if} = -\text{decision\_function}(\cdot),$$

so that higher values correspond to a higher likelihood of being anomalous (and potentially fraudulent).

A boxplot of `risk_score_if` by class shows:

- Fraudulent transactions (Class = 1) have significantly higher risk scores, typically around 0.08–0.16.
- Normal transactions (Class = 0) cluster around lower risk scores (approximately -0.20 to -0.15).
- There is some overlap, as expected, but the separation of medians is substantial, indicating that `risk_score_if` is a highly informative feature.

The fitted Isolation Forest model is saved (`isolation_forest.pkl`) and is reused for scoring new data in both the test phase and the deployed application.

4.5 Final Feature Set for Modeling

After all transformations, the final feature set used for supervised modeling includes:

- PCA components: `V1-V28`.
- Engineered amount feature: `scaled_amount`.
- Engineered time feature: `hour_sin`.
- Anomaly-based feature: `risk_score_if`.

The columns `Time`, `hour_of_day`, and `hour_cos` are dropped prior to model training and evaluation.

5 Modeling Approach

5.1 Problem Framing and Evaluation Metrics

The problem is framed as a binary classification task:

- **Positive class**: fraudulent transactions (Class = 1).
- **Negative class**: legitimate transactions (Class = 0).

Given the extreme class imbalance and the business context, the following metrics are prioritized:

- **Recall for the fraud class (Class = 1)**: proportion of true frauds correctly detected. Missing a fraud (false negative) can result in direct financial loss.
- **Precision for the fraud class**: proportion of predicted frauds that are truly fraudulent. Low precision implies many false positives, which can cause unnecessary customer friction and operational costs.
- **ROC AUC**: global measure of ranking quality across thresholds.
- **Confusion matrix**: to explicitly observe the trade-off between false positives and false negatives.

Two main supervised models are built and evaluated:

1. Logistic Regression with class weighting.
2. XGBoost (Extreme Gradient Boosting) with a scaling factor for the positive class.

5.2 Logistic Regression with Class Weighting

5.2.1 Model Configuration

A Logistic Regression model is trained on the full training set using:

- Solver: `lbfgs`.
- Maximum iterations: 1000.
- `class_weight` = `balanced`, which automatically up-weights the minority class (fraud) in proportion to the class imbalance.

Table 1: Logistic Regression – Test Confusion Matrix

	Predicted Non-Fraud	**Predicted Fraud**
Actual Non-Fraud	137,459	4,722
Actual Fraud	27	196

5.2.2 Test Performance

On the time-based test set, the model yields the following confusion matrix and metrics: The classification report for the test set shows approximately:

- Fraud precision: 0.04.
- Fraud recall: 0.88.
- Overall ROC AUC: ≈ 0.95.

From a business perspective:

- The model is very aggressive in flagging potential fraud, successfully detecting nearly 88% of true fraudulent transactions.
- However, the precision for the fraud class is low, meaning that many legitimate transactions are being flagged as suspicious (false positives).

This setting might be acceptable in a second-layer review process, where flagged transactions trigger a lightweight verification (such as an SMS confirmation). But as a standalone front-line decision engine, this level of false positives may introduce notable customer friction.

5.3 XGBoost Gradient Boosting Model

5.3.1 Model Configuration

To achieve a better balance between detection and false alarms, an XGBoost model is trained with the following configuration:

- Objective: `binary:logistic`.
- Evaluation metric: `auc`.
- Maximum tree depth: 6.
- Learning rate: 0.1.
- Subsample ratio: 0.8.

- Column subsample ratio: 0.8.
- `scale_pos_weight`: computed as the ratio of non-fraud to fraud samples in the training subset, to counter the class imbalance.

The training data is further split into training and validation subsets using `train_test_split` with stratification on the target class. Early stopping is applied with:

- Maximum of 500 boosting rounds.
- Early stopping after 20 rounds without improvement in validation AUC.

5.3.2 Validation Performance

On the validation set, the XGBoost model achieves:

- High overall accuracy ($> 99\%$).
- Fraud recall above 85%.
- ROC AUC on validation around 0.96.

This indicates strong discrimination power and good calibration for ranking transactions by risk level.

5.3.3 Test Performance

On the time-based test set, the model yields the following confusion matrix:

Table 2: XGBoost – Test Confusion Matrix

	Predicted Non-Fraud	**Predicted Fraud**
Actual Non-Fraud	141,816	365
Actual Fraud	48	175

The corresponding test metrics for the fraud class are approximately:

- Fraud precision: 0.32.
- Fraud recall: 0.78.
- ROC AUC: ≈ 0.96.

Compared to the Logistic Regression model:

- Fraud recall decreases slightly (from $\sim$88% to $\sim$78%), meaning more frauds are missed.

- Fraud precision increases substantially (from ∼4% to ∼32%), meaning that a much larger portion of flagged transactions are truly fraudulent.

From a business standpoint, this represents a more attractive trade-off for many use cases: fewer legitimate customers are disrupted, and the operational review workload is focused on a smaller, higher-quality set of alerts.

5.4 Model Selection

Given the trade-offs:

- Logistic Regression is useful when the primary objective is *maximum recall* and the business is willing to tolerate a high volume of false positives.
- XGBoost provides a more balanced solution, with strong detection capability and significantly improved precision.

In this project, the XGBoost model is chosen as the primary model for deployment, due to its superior overall balance and higher ROC AUC on the time-based test set.

6 Deployment and User Interface

6.1 Deployment Artifacts

The final deployed solution consists of:

- `robust_scaler.pkl`: pre-fitted RobustScaler for the `Amount` feature.
- `isolation_forest.pkl`: Isolation Forest model for computing `risk_score_if`.
- `xgb_fraud_model.json`: trained XGBoost model.

These artifacts are loaded by the application to ensure consistency between training, evaluation, and production scoring.

6.2 Streamlit Web Application

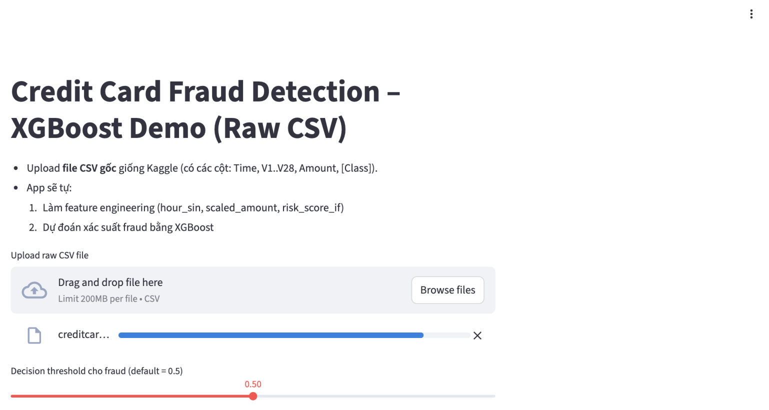
A simple `Streamlit` web application is developed to serve as a demonstration UI for fraud risk scoring. The main capabilities of the app include:

- **File upload**: users can upload a raw CSV file with the same schema as the original dataset (including `Time`, `V1-V28`, and `Amount`, optionally `Class`).
- **Automatic preprocessing**:

  - Deduplicate rows and reset indices.
  - Fill missing numerical values with zero (for robustness).
  - Apply the saved RobustScaler to compute `scaled_amount`.
  - Derive `hour_of_day` from `Time` and compute `hour_sin`.
  - Use the Isolation Forest to compute `risk_score_if`.
  - Drop raw `Amount`, `Time`, and intermediate time-of-day columns.

- **Fraud scoring**:
  - Generate fraud probabilities using the XGBoost model.
  - Convert probabilities to binary fraud flags based on a user-controlled decision threshold.

- **Interactive threshold selection**: the application allows the user to adjust the decision threshold (default 0.5), enabling exploration of different trade-offs between recall and precision.



Credit Card Fraud Detection – XGBoost Demo (Raw CSV)

- Upload **file CSV gốc** giống Kaggle (có các cột: Time, V1..V28, Amount, [Class]).
- App sẽ tự:
  1. Làm feature engineering (hour_sin, scaled_amount, risk_score_if)
  2. Dự đoán xác suất fraud bằng XGBoost

Upload raw CSV file

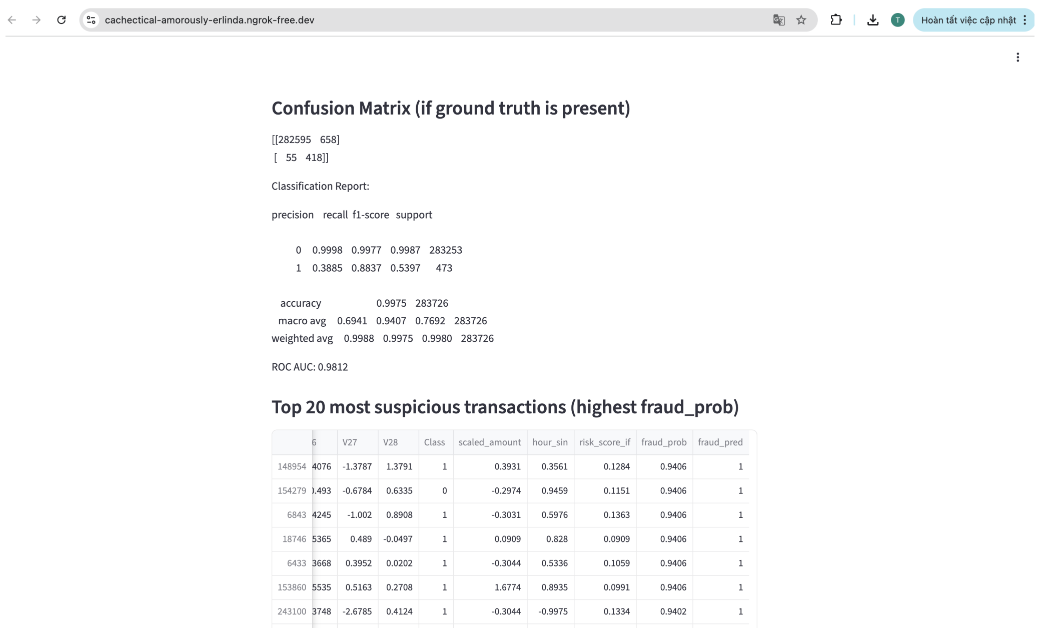
Drag and drop file here
Limit 200MB per file • CSV

creditcar…

Decision threshold cho fraud (default = 0.5)

0.50

- **Results visualization**:
  - Show a preview of processed data, including engineered features.
  - Display the number of total transactions and the number of transactions flagged as fraud.
  - If the ground-truth `Class` column is present, display the confusion matrix, classification report, and ROC AUC.



Confusion Matrix (if ground truth is present)

```
[[282595   658]
 [   55   418]]

Classification Report:

precision    recall  f1-score   support

       0    0.9998    0.9977    0.9987    283253
       1    0.3885    0.8837    0.5397       473

  accuracy                        0.9975    283726
  macro avg    0.6941    0.9407    0.7692    283726
weighted avg    0.9988    0.9975    0.9980    283726

ROC AUC: 0.9812
```

Top 20 most suspicious transactions (highest fraud_prob)

	6	V27	V28	Class	scaled_amount	hour_sin	risk_score_if	fraud_prob	fraud_pred
148954	4076	-1.3787	1.3791	1	0.3931	0.3561	0.1284	0.9406	1
154279	).493	-0.6784	0.6335	0	-0.2974	0.9459	0.1151	0.9406	1
6843	4245	-1.002	0.8908	1	-0.3031	0.5976	0.1363	0.9406	1
18746	5365	0.489	-0.0497	1	0.0909	0.828	0.0909	0.9406	1
6433	3668	0.3952	0.0202	1	-0.3044	0.5336	0.1059	0.9406	1
153860	5535	0.5163	0.2708	1	1.6774	0.8935	0.0991	0.9406	1
243100	3748	-2.6785	0.4124	1	-0.3044	-0.9975	0.1334	0.9402	1

– Highlight the top 20 most suspicious transactions sorted by fraud probability.

This UI turns the underlying model into a practical tool that risk analysts or operations staff can use to inspect new transaction batches, perform what-if analysis on thresholds, and prioritize investigations based on model scores.

7 Business Interpretation and Recommendations

7.1 Key Insights from the Data

From the exploratory analysis and modeling results, several meaningful business insights emerge:

- **Fraud is rare but impactful**: although only about 0.172% of transactions are labeled as fraud, the associated financial and reputational risks justify the use of advanced detection models.
- **Fraudsters favor inconspicuous amounts**: fraudulent transactions do not always correspond to extreme transaction amounts; many frauds use relatively small amounts to avoid triggering simple amount-based rules.
- **Time-of-day matters**: fraud occurrences increase during off-peak hours (approximately 1–5 AM), when normal customer activity is low. However, fraud is also present throughout the day, especially during business hours, likely to blend in with regular activity.

- **Anomaly-based risk scores are valuable**: the Isolation Forest-derived `risk_score_if` feature shows strong separation between fraud and non-fraud, providing an additional signal beyond raw features.

7.2 Operational Recommendations

Based on the final XGBoost model and its deployment:

- **Implement a risk-based decision process**:
  - Use the model's fraud probability as a *risk score*.
  - Define multiple action bands:
    * Very high risk: automatically block or require strong customer authentication (e.g., two-factor verification).
    * Medium risk: flag for manual review or delayed settlement.
    * Low risk: allow transaction with standard monitoring.
- **Calibrate decision thresholds to business risk appetite**:
  - For higher protection, lower the threshold and accept more false positives.
  - For smoother customer experience, increase the threshold and reduce false positives, at the cost of missing some frauds.
- **Prioritize off-peak monitoring**: allocate additional automated or manual monitoring capacity to early-morning hours when fraud risk is relatively higher.
- **Integrate anomaly scores into existing rule engines**: the `risk_score_if` can be combined with existing rule-based checks to create hybrid rules (for example, high anomaly score and repeated small transactions in a short time window).

7.3 Stakeholders and Use Cases

The solution can support several stakeholders:

- **Fraud and risk teams**: investigate flagged transactions and refine operational thresholds.
- **Customer service**: manage customer communication for blocked or challenged transactions with higher confidence.
- **Product and strategy teams**: evaluate the impact of different fraud control policies on customer experience and loss reduction.

8 Limitations and Future Work

While the current solution demonstrates strong performance and a practical deployment path, several limitations and opportunities for improvement remain:

- **Limited feature set and anonymization**: due to confidentiality constraints, only PCA-transformed features are available. Access to more granular features (merchant category, geography, device identifiers, customer history) could significantly enhance model performance and interpretability.
- **Short time horizon**: the dataset covers only two days of activity. Extending the analysis to longer periods would enable robust analysis of seasonal patterns, customer-level behavior, and model stability over time.
- **Cost-sensitive optimization**: while the current model is tuned with standard metrics (recall, precision, ROC AUC), future work could incorporate explicit cost functions reflecting monetary loss for false negatives and operational costs for false positives.
- **Model explainability**: adding model explanation tools (such as SHAP values) would help analysts understand which features drive each fraud alert, improving trust and facilitating policy decisions.
- **Continuous monitoring and retraining**: in a real production setting, the model should be periodically monitored for performance drift and retrained as fraud patterns evolve.

9 Conclusion

This project demonstrates how a combination of feature engineering, anomaly detection, and supervised learning can be used to build an effective credit card fraud detection system. By translating raw transaction data into actionable risk scores and deploying a usable interface, the solution provides tangible value to business stakeholders:

- improved detection of fraudulent transactions,
- more efficient allocation of investigation resources, and
- better control over the trade-off between customer experience and fraud losses.

Although based on anonymized and limited-duration data, the methodology can be extended and adapted to richer, real-world environments. With further enhancements in feature design, cost-sensitive optimization, and model governance, the approach outlined here can form a solid foundation for an enterprise-grade fraud detection capability.