Real-World Fraud Detection using Anomaly Detection

1 Introduction and Project Overview

Financial fraud poses a significant challenge for modern payment systems that process large volumes of transactions in real time. Fraudulent activities are often rare, highly adaptive, and deliberately designed to resemble legitimate user behavior, making traditional rule-based detection approaches insufficient.

This project focuses on building a real-world fraud detection system using anomaly detection techniques. Instead of treating fraud detection as a standard binary classification problem, the system models normal transaction behavior and identifies deviations from this baseline as potential fraud. This approach is particularly suitable for highly imbalanced datasets where fraudulent transactions represent only a small fraction of total activity.

The objective of this project is to design an end-to-end fraud detection pipeline that includes data preprocessing, feature engineering, multiple anomaly detection models, performance evaluation, and a production-ready prediction API. Emphasis is placed not only on detection performance, but also on interpretability, latency, and robustness, which are critical for deployment in real financial environments.

2 Dataset Overview and Feature Enrichment

- **Data Type:** Real-world financial transaction dataset derived from card-based payment activity.[1]
- **Dataset Size:** Approximately 280,000 transaction records.
- **Fraud Rate:** Around 0.17% of transactions labeled as fraudulent, indicating extreme class imbalance.
- **Original Features:** Transaction amount, transaction time, and anonymized PCA-transformed features (V1–V28) representing sensitive behavioral attributes.
- **Limitation of Raw Data:** Original dataset does not contain explicit geographical or customer location information due to privacy constraints.
- **Feature Enrichment (Custom):** Customer home latitude/longitude and merchant latitude/longitude were synthetically generated to simulate realistic transaction geography.
- **Geographical Feature Engineering:** Distance from customer home location was computed using latitude and longitude coordinates to capture spatial anomalies.

[1] `https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud`

- **Temporal Feature Engineering:** Transaction timestamp was converted into hour of day, day of week, and month to model temporal spending behavior.
- **Label Usage:** Fraud labels were used only for offline evaluation and validation; anomaly detection models were trained primarily on non-fraudulent transactions.
- **Final Feature Set:** Combination of PCA features, transaction amount, temporal features, and engineered spatial features for behavioral anomaly detection.

3 Feature Engineering Choices and Justification

- **Transaction Amount:** Captures deviations from a customer's typical spending behavior, large or abrupt changes are strong indicators of fraud.
- **PCA-Transformed Features (V1–V28):** Represent anonymized behavioral patterns derived from sensitive transaction attributes enable detection of complex, non-linear anomalies without exposing raw data.
- **Hour of Day:** Fraudulent activity often occurs at unusual hours (e.g., late night) when legitimate user activity is low.
- **Day of Week:** Helps identify irregular spending behavior occurring on weekends or non-business days.
- **Month:** Captures seasonal effects and periodic changes in transaction behavior.
- **Customer Home Location (Lat/Lon):** Simulates normal geographic regions associated with individual customers.
- **Merchant Location (Lat/Lon):** Enables modeling of spatial transaction patterns across different merchant regions.
- **Distance from Home:** Provides a strong spatial anomaly signal; large distances from home are commonly associated with card misuse and account takeover scenarios.
- **Multi-Feature Interaction:** Combining amount, temporal, PCA, and spatial features allows the model to detect fraud based on overall behavioral deviation rather than single-feature thresholds.

4 Model Selection and Results

Fraud labels were used only for evaluation purposes and were not used during model training. Multiple anomaly detection models were evaluated to identify a solution that balances detection performance, interpretability, inference latency, and false-positive cost.

Model	**Precision**	**Recall**	**F1-score**	**ROC-AUC**
Isolation Forest	0.3287	0.5813	0.4200	0.9497
One-Class SVM	0.3681	0.8313	0.5103	0.9483
Autoencoder	**0.7214**	0.5894	**0.6488**	0.9357

Table 1: Performance comparison of anomaly detection models

Model-wise Analysis and Justification:

- **Isolation Forest:** This model provided fast inference and reasonable recall. However, it generated a large number of false positives, which would lead to frequent customer interruptions and high manual review costs in a real deployment.
- **One-Class SVM:** One-Class SVM achieved very high recall, indicating strong fraud detection capability. However, its low precision and unstable predictions resulted in excessive false alerts. In addition, it showed limited scalability for large transaction volumes.
- **Autoencoder (Selected Model):** The autoencoder achieved the best balance between precision and recall, significantly reducing false positives while maintaining strong fraud detection performance. Fraudulent transactions consistently produced higher reconstruction errors compared to legitimate ones.
- **Interpretability:** Feature-level reconstruction errors from the autoencoder allow identification of which transaction attributes contributed most to the anomaly, enabling human-readable explanations.
- **Latency and Deployment Suitability:** Autoencoder inference latency remained within real-time limits for API-based deployment, making it suitable for high-throughput payment systems.
- **False-Positive Cost Control:** A conservative anomaly threshold was selected at the 99.3 percentile of reconstruction error to minimize customer disruption and operational overhead.

5 Performance Analysis on Edge Cases

In real-world fraud detection systems, model performance on unusual or borderline scenarios is more important than average performance. The selected anomaly detection approach was analyzed across several practical edge cases to ensure robustness and safe deployment behavior.

- **Low-Amount Fraud Transactions:** Fraudulent transactions involving small amounts are often overlooked by rule-based systems. In this project, such transactions were still detected when combined with unusual timing or abnormal location patterns, as the model evaluates overall behavioral deviation rather than relying on transaction amount alone.
- **High-Amount Legitimate Transactions:** High-value purchases made by genuine users pose a high false-positive risk. The model showed lower anomaly scores for such trans-

actions when they occurred at familiar locations and normal times, reducing unnecessary customer disruptions.

- **Unseen or New Merchant Locations:** Transactions at previously unseen merchant locations are not automatically flagged as fraudulent. If the transaction occurred within a reasonable distance from the customer's home and followed normal temporal patterns, the anomaly score remained low.
- **Unusual Transaction Timing:** Transactions occurring at late-night or uncommon hours were treated as moderate risk when other features such as amount and location were normal. This prevents single-feature anomalies from triggering false fraud alerts.
- **Combined Extreme Anomalies:** Transactions with multiple deviations, such as high amount, unusual timing, and large distance from home, consistently produced high anomaly scores and were classified as high or extreme risk.

Overall, the anomaly detection approach demonstrated stable behavior across edge cases by considering multiple transaction attributes together. This multi-dimensional analysis helps reduce false positives while maintaining strong fraud detection capability in realistic scenarios.

6 Production Deployment Considerations

The fraud detection system was designed with real-world deployment constraints in mind, focusing on reliability, latency, scalability, and operational safety.

- **API Design:** The model is deployed using a FastAPI-based REST service with a `/predict` endpoint. The API accepts transaction features as input and returns anomaly score, fraud probability, risk level, and human-readable reasons for the prediction.
- **Input Validation and Error Handling:** Strict input validation is enforced using schema definitions to ensure that all required features are present and correctly formatted. Invalid or incomplete requests are rejected early, preventing model failures and unexpected behavior.
- **Model Loading and Efficiency:** All model artifacts, including the scaler and autoencoder, are loaded once at application startup. This avoids repeated disk access and ensures low-latency inference for each prediction request.
- **Latency Considerations:** Autoencoder inference is lightweight and operates well within real-time constraints. The end-to-end prediction pipeline is suitable for high-throughput transaction processing systems.
- **Threshold Management:** A conservative anomaly threshold, derived offline using the 99.3 percentile of reconstruction error, is used to control false positives. The threshold can be updated without retraining the model, enabling operational flexibility.
- **Explainability and Monitoring:** Feature-level reconstruction errors are mapped to human-readable reasons, allowing analysts to understand why a transaction was flagged. Structured logging captures prediction details such as anomaly score, fraud probability, and risk level for monitoring and audit purposes.

- **Scalability:** The API is stateless, enabling horizontal scaling through containerization and load balancing. This design supports increasing transaction volumes without changes to model logic.
- **Retraining Strategy:** Periodic retraining using recent non-fraudulent transactions is recommended to handle data drift and evolving user behavior while maintaining model stability.

7 Conclusion

This project presented a real-world fraud detection system built using anomaly detection techniques. Instead of relying on traditional supervised classification, the system models normal transaction behavior and identifies deviations as potential fraud, making it suitable for highly imbalanced financial datasets.

Through feature enrichment and careful feature engineering, temporal and spatial patterns were incorporated to better capture fraudulent behavior. Multiple anomaly detection models were evaluated, and an autoencoder-based approach was selected due to its strong balance between detection performance, interpretability, latency, and false-positive control.

The system was designed with production deployment in mind, including input validation, explainability, logging, and scalable API-based inference. Performance analysis on edge cases demonstrated stable and realistic behavior under challenging scenarios.

Overall, the proposed solution provides a practical, interpretable, and deployable fraud detection framework that aligns with real-world financial system requirements.