Credit Card Fraud Detection Report

Executive Summary

As a new player in the credit card industry, [Company Name] aims to establish itself as the safest option for customers in the United States. To achieve this, we have developed a machine learning model to identify fraudulent credit card transactions. The model is designed to prioritise fraud detection while accepting some false positives to ensure maximum security for customers.

After testing multiple machine learning models, **XGBoost** has emerged as the most effective in balancing fraud detection accuracy and recall. This report outlines the results of our analysis, key insights from model performance, and recommendations for deployment and continuous improvement.

Problem Overview

Credit card fraud is a significant challenge for financial institutions, causing billions in losses annually. Given the competitive landscape, **[Company Name] must demonstrate robust fraud detection capabilities to build trust and ensure the safety of its cardholders.**

The primary challenge is to **detect fraudulent transactions effectively without incorrectly blocking too many legitimate ones.** Since fraud is rare, it is crucial to develop a model that can correctly identify fraudulent transactions while maintaining high precision and recall.

Our executive team has emphasised **err on the side of caution**, meaning we should prioritise catching fraudulent transactions even if some legitimate ones are mistakenly flagged.

Key Insights from the Model

Fraud Detection Accuracy

We tested three machine learning models: **Random Forest, Gradient Boosting, and XGBoost**, evaluating them at both default and adjusted thresholds to maximise fraud detection.

Best Model: XGBoost

- **Precision: 43% (default) → 25% (threshold = 0.3)**
- **Recall: 85% (default) → 91% (threshold = 0.3)**
- **F1-score: 57% (default) → 39% (threshold = 0.3)**
- **Accuracy: 99% (default) → 98% (threshold = 0.3)**

Comparison with Other Models

1. **Random Forest**
   - ○ At the default threshold, it had high **precision (83%)** but lower recall (65%), meaning it was **good at avoiding false positives but missed many fraud cases**.
   - ○ At the adjusted threshold (0.3), recall increased to **77%**, but precision dropped to **68%**.
2. **Gradient Boosting**
   - ○ High recall (87%) but very low precision (21%) at the default threshold.
   - ○ At a threshold of 0.3, it had **extremely low precision (8%)**, meaning most flagged transactions were false positives.
3. **XGBoost**
   - ○ **Balanced precision and recall**, making it the best option.
   - ○ **Recall was significantly higher (85%-91%) than other models, meaning it detected more fraud cases.**

Model Performance in the Context of Business Goals

- **Prioritising Recall:** Since we want to catch as many fraud cases as possible, XGBoost performs best by identifying **91% of fraudulent transactions at a 0.3 threshold**.
- **Trade-off with Precision:** While we expect more false positives, this aligns with the company's risk-averse approach.

Business Impact

Fraud Prevention

By detecting fraudulent transactions early, [Company Name] can significantly reduce potential financial losses. A high-recall model ensures that most fraudulent transactions are flagged before they cause harm.

Customer Trust and Retention

Customers are more likely to trust a credit card that proactively detects fraud. By prioritising security, the company strengthens its reputation as a safe financial institution.

Operational Efficiency

Automating fraud detection reduces the need for manual review of transactions, **cutting operational costs and enabling faster response times** to potential fraud cases.

Recommendations

Fine-Tuning the Model

1. **Further Improve Recall**
    - Adjust the classification threshold to **flag more potential fraud cases**, increasing recall.
    - Introduce **ensemble methods** to combine the strengths of multiple models.
2. **Refine Features**
    - Improve fraud detection by incorporating **real-time transaction patterns** and external fraud databases.

Real-Time Deployment

1. **Deploy Model into Transaction Processing System**
    - The model should run in real-time to flag fraudulent transactions before approval.
    - **Transactions flagged as suspicious should trigger security measures**, such as customer verification.
2. **Integration with Existing Security Measures**
    - Combine machine learning predictions with **customer spending habits, location tracking, and anomaly detection** to improve accuracy.

Continuous Monitoring & Model Updates

1. **Retraining with New Data**
    - Fraud patterns evolve, so **periodic model retraining is essential** to adapt to emerging fraud techniques.
2. **Monitor Model Performance**
    - Track **false positive rates and customer complaints** to refine model thresholds.

Customer Communication Strategy

1. **Handling False Positives**

   ○ Notify customers when a transaction is flagged and provide **an easy verification process** to resolve false alarms.

2. **Improve Customer Experience**

   ○ Offer customers additional security options, such as **temporary transaction limits or two-factor authentication for flagged transactions**.

Conclusion

Our fraud detection analysis has shown that **XGBoost is the best model for identifying fraudulent transactions while maintaining a balance between precision and recall.**

By **prioritising recall**, the company can **catch more fraud cases and prevent financial losses** while accepting some false positives as a trade-off.

Next Steps

- **Deploy XGBoost in real-time fraud detection.**
- **Adjust classification thresholds to maximize recall.**
- **Continuously retrain and monitor the model.**
- **Enhance customer communication strategies to handle false positives efficiently.**

With these steps, **[Company Name] will strengthen its position as the safest credit card provider** while ensuring a seamless customer experience.

Model and Data Access

The trained model, code, and dataset are available for further review at:
https://github.com/luluhsu727/data-science-portfolio/tree/main/Fraud%20Detection