Fraud Detection Executive Summary

Author: Gurdeep Singh

I developed an end-to-end fraud detection solution using the provided transaction dataset. The work includes efficient data loading and cleaning, feature engineering (time, velocity, and amount transforms), baseline and advanced model training (interpretable Logistic Regression, and LightGBM for final scoring), and model evaluation using ROC AUC and Precision-Recall AUC. Operational recommendations and a measurement plan are included for production adoption.

Dataset: ~6.3M rows, 10 columns (CSV provided). - Deliverables: This Jupyter Notebook (analysis, code, plots, artifacts)

Approach & Methods: - Data cleaning: median imputation for numerics, 'missing' for categoricals, and outlier capping (1st/99th percentiles). - Feature engineering: amount_log, hour, dayofweek, secs_since_prev (velocity), and categorical encodings. - Modeling: baseline Logistic Regression (interpretable) and LightGBM (final). RandomForest used for fast iteration.

Key Findings:
- Final model: RandomForest.
- Performance: ROC AUC = 0.9961, PR AUC = 0.8302.
- Top predictors: amount_log, secs_since_prev (transaction velocity), hour (time of day), merchant/device features.

Recommendations (production): 1. Real-time scoring API for near-real-time decisioning. 2. Hybrid rules + ML approach (blacklists, velocity rules before ML to reduce risk). 3. Device fingerprinting, 2FA for suspicious flows, and adaptive thresholds by customer segment. 4. Logging & feedback loop to capture confirmed fraud labels for continuous retraining. 5. Monitoring for model drift and business KPIs (fraud prevented, false positives, customer friction).

Measurement Plan: - Use A/B testing or canary rollout to measure reduction in confirmed fraud and impact on conversion. - Key metrics: confirmed fraud saved (monetary), false positive rate, PR AUC change, manual review workload. - Run offline backtests on historical confirmed labels before full rollout.