Insurance Fraud Detection with AI

Chrisostomos–Panagiotis Stamou

August 25, 2025

Project, Motivation, and AI Intervention

Insurance companies incur significant losses from fraudulent claims. Fraud prevalence is relatively low (about 10%), but the cost per case is high. Manually reviewing every claim is impractical. **Goal:** use AI to assign a fraudrisk score to each claim so investigators can focus on the riskiest subset.

AI intervenes by learning patterns that differentiate fraud from genuine claims and producing *probabilities* of fraud. These probabilities are then converted into a review list by selecting the top fraction of claims (a "flag rate") that matches operational capacity.

The code, notebook and details of these project can found on the public GitHub repository: https://github.com/ChrisStamou/AI-Case-Insurance-Fraud-Detection-with-AI.

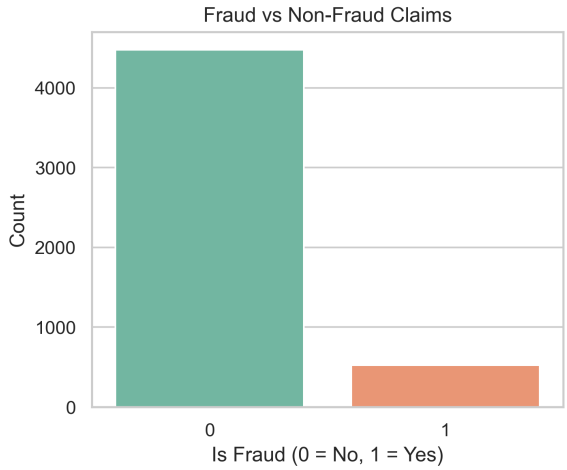
Synthetic Data and Current Situation

We created a synthetic dataset representing a midsized Dutch motor insurance company with **5,000** claims. The target variable is `is_fraud` (1 = fraud, 0 = genuine) with an overall rate of $\approx 10\%$. Features include:

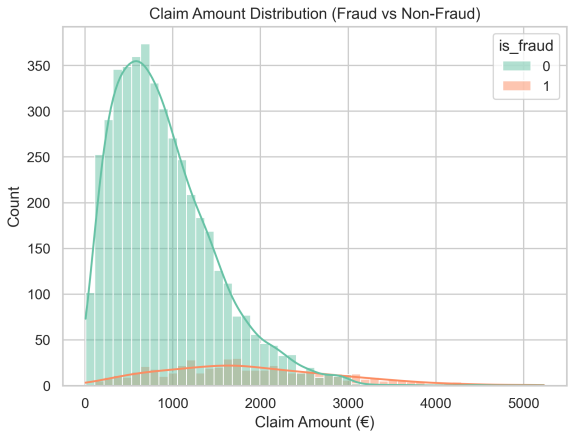
- **claim_amount**: skewed, average $\sim$ €1,000.
- **days_to_file**: 0–60 days from incident to filing.
- **channel**: {agent, online, phone}.
- **policy_tenure_yrs**: 0–15.
- **has_prior_claims**: {0,1}.
- **region**: {NLNorth, West, South, East}.
- **vehicle_age_yrs**: 0–20.
- **claim_type**: {collision, theft, weather, glass}.

Exploratory View (Synthetic)

Below are example figures to characterise the current situation.

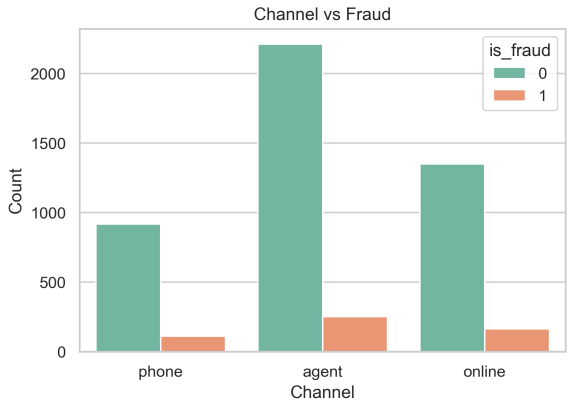


(a) Fraud vs Non-Fraud counts

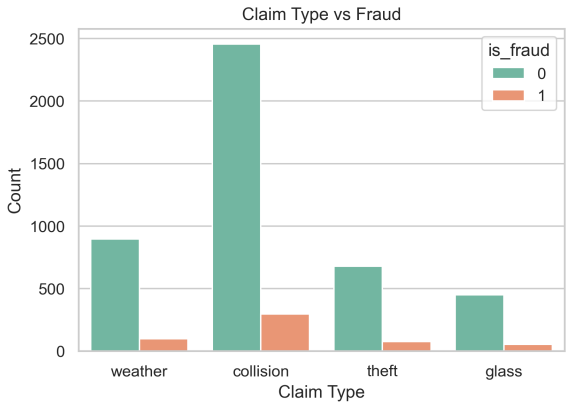


(b) Claim amount distribution

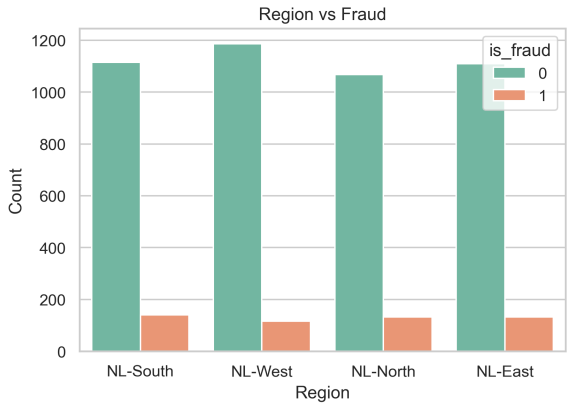
(c) Days to file



(d) Channel vs Fraud



(e) Claim type vs Fraud



(f) Region vs Fraud

Figure 1: Exploratory overview of the synthetic dataset.

AI Approach and Modeling Framework

Stack: Python (`pandas`, `numpy`, `matplotlib`), `scikitlearn`. **Models tested:** (i) Logistic Regression (LR), (ii) Random Forest (RF).

Build Process

1. **Split:** train 80%, test 20% (holdout test used only for final evaluation).
2. **Preprocessing:**
   - Numeric features scaled (StandardScaler).
   - Categorical features onehot encoded.
3. **Training:** fit LR and RF on the training data.
4. **Scoring:** compute *probabilities* $\hat{p} = \mathbb{P}(\text{fraud} \mid \text{features})$ on the test set, with $\hat{p} \in [0, 1]$; higher values mean higher fraud risk. We sort by $\hat{p}$ and flag the top $K\%$ (e.g., 25%).

Why These Models

LR offers a strong linear baseline and is wellcalibrated on small/medium datasets. **RF** captures nonlinearities and interactions without manual feature engineering.

Evaluation: What We Measure and Why

- **ROC AUC** (Area Under the ROC Curve): probability that a randomly chosen fraud case is ranked higher than a randomly chosen genuine case. Higher is better; thresholdindependent.
- **Precision**: among flagged claims, the fraction that are truly fraud. Important when investigation capacity is limited.
- **Recall**: among all fraud cases, the fraction correctly flagged. Important to avoid missing fraud.
- **F1**: harmonic mean of precision and recall; balances both.

Flag Rate and Threshold.

Investigators can review only a fraction of claims. We sort testset claims by predicted $\hat{p}$ and **flag the top** $K\%$ (e.g., $K = 25\%$). This induces a probability threshold that aligns with capacity. We then compute precision/recall/F1 at that operating point.

Results on Hold-out Test Set

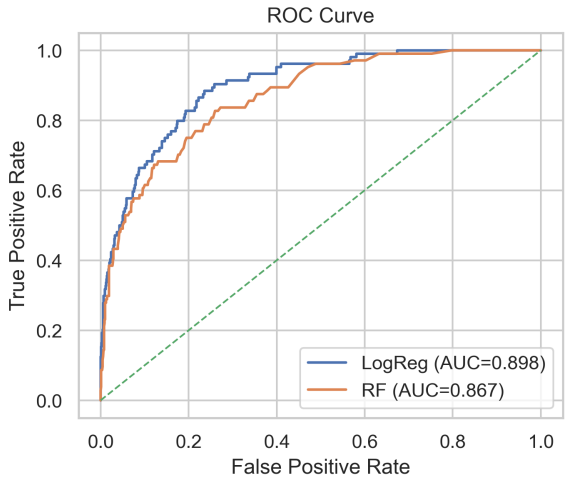
Test size: $n = 1{,}000$ claims. Metrics at **25%** flag rate:

Logistic Regression

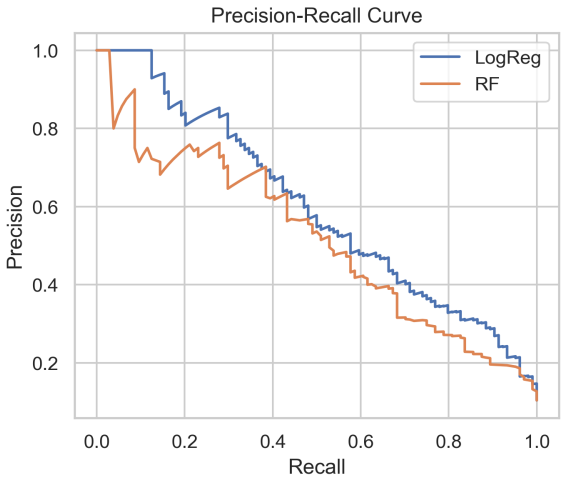
- ROC AUC: **0.898**
- Precision: **0.332**
- Recall: **0.798**
- F1: **0.469**

Random Forest

- ROC AUC: **0.867**
- Precision: **0.308**
- Recall: **0.750**
- F1: **0.437**



(a) ROC curves: LR vs. RF (test set).



(b) Precision–Recall curves (test set).

Figure 2: Model comparison: ROC (left) and Precision–Recall (right).

Interpretation

In this synthetic setting:

- **LR outperformed RF** on AUC and at the 25% flag operating point (higher precision and recall).
- Operating at 25% means investigators review the riskiest quarter of claims instead of all claims, while still capturing ∼80% of fraud cases.
- A simple, interpretable baseline (LR) is effective when paired with proper preprocessing and capacity-aligned thresholding.

Time & Workforce Impact — Manual vs AI (LogReg @ 25% flag rate)

What this means in numbers.

- **Workload reduction:** from 1,000 investigated claims to 250 (−75%); that's a **4**× smaller review queue.
- **Precision improvement:** from 10% to 33% — an absolute increase of +**23** percentage points and a **3.3**× relative improvement.
- **Throughput:** investigator-days drop from 5,000 to 1,250 (−3,750), and calendar time (50 investigators) from 100 to 25 days.

Table 1: Time & Workforce Impact — Manual vs AI (LogReg @ 25% flag rate)

Metric	**Manual (No AI)**	**With AI**	**Δ (Savings/Change)**
Claims investigated	1,000	250	−75%
Investigator-days (5 days/claim)	5,000	1,250	−3,750 (−75%)
Calendar days @ 50 investigators	100	25	−75 days (−75%)
Fraud caught (cases)	100 (100% recall)	∼ 80 (80% recall)	−20
Precision (fraud hit rate)	10%	33%	×3.3
Inv-days per fraud caught	50.0	15.6	−69%

- **Coverage:** recall stays high at ∼ 80%, so most fraudulent cases are still detected even with a much smaller queue.

The relation between precision and recall for any flag rate for both models is illustrated below:

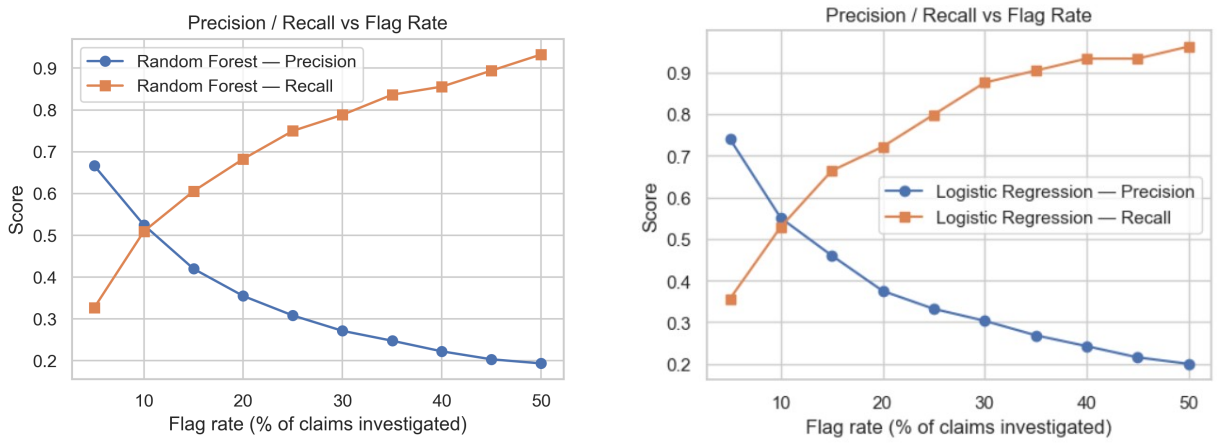


Figure 3: Precision and Recall relation for all flag rates across both RF and LR models.

Take-Home Message

This project demonstrates that even a simple, interpretable model such as Logistic Regression, when aligned with operational constraints, can drastically reduce investigation workload while maintaining high fraud detection, proving that AI can deliver immediate and tangible value to business processes.