1. Introduction

Healthcare fraud continues to create major financial burdens for national insurance systems, costing billions annually. Detecting fraudulent providers is challenging because fraud patterns are subtle, inconsistent, and hidden within vast amounts of legitimate claims data.

This project builds a complete end-to-end machine learning pipeline to detect potential fraudulent healthcare providers using Medicare claims data. The work includes:

- Data exploration
- Cleaning and preprocessing
- Feature engineering
- Provider-level aggregation
- Model selection and training
- Evaluation and error analysis
- Business and operational recommendations

All work follows a clear scientific reasoning process, with documentation of experiments and decisions.

---

2. Data Overview

The dataset consists of four main components:

2.1. Beneficiary Data

Contains demographic and chronic condition indicators for each beneficiary.

2.2. Inpatient Claims

Includes hospitalization records, admission/discharge dates, diagnosis codes, procedure codes, and reimbursed amounts.

2.3. Outpatient Claims

Includes visit-level provider billing data for non-hospital services.

2.4. Provider Labels

Indicates whether a provider is tagged as **PotentialFraud = Yes/No**.

---

3. Data Exploration

The initial exploration focused on:

✔ Checking dataset sizes

- 138k+ beneficiaries
- 40k inpatient claims
- 517k outpatient claims
- 5,410 providers

✔ Inspecting null values

Inpatient and outpatient datasets contain significant missingness in physician fields and diagnosis codes. These missing values are expected in claims data and are not detrimental to modeling once aggregated.

✔ Evaluating schema correctness

Date fields in inpatient/outpatient claims were stored as strings and required conversion to `datetime`.

✔ Understanding label imbalance

Fraud labels distribution:

- **Yes (Fraud): 506 providers**

- **No (Legitimate): 4904 providers**

This ~1:10 imbalance directly influenced model selection and evaluation strategy.

---

4. Preprocessing Steps

These preprocessing operations were applied:

4.1. Date Parsing

`ClaimStartDt`, `ClaimEndDt`, `AdmissionDt`, and `DischargeDt` were converted to datetime format.

4.2. Deriving Length of Stay (LOS)

For inpatient claims:

- `LOS = (DischargeDt - AdmissionDt).days`

This became one of the strongest fraud indicators.

4.3. Handling Missing Values

Because the analysis aggregates to provider-level statistics, missing diagnosis/procedure codes do not hinder modeling. Only LOS rows with invalid or negative durations were removed.

4.4. Merging Claims With Provider IDs

In both inpatient and outpatient datasets, the `Provider` column was renamed to `ProviderID` to match labels.

---

5. Feature Engineering

This is the most important step in the project.

Claims-level data was aggregated **per provider**, producing a single row per provider.

5.1. Inpatient Features

For each provider, the following were created:

- `INP_LOS_mean`
- `INP_LOS_max`
- `INP_LOS_min`
- `INP_InscClaimAmtReimbursed_mean`
- `INP_InscClaimAmtReimbursed_sum`
- `INP_ClaimID_count`

5.2. Outpatient Features

- `OUT_InscClaimAmtReimbursed_mean`
- `OUT_InscClaimAmtReimbursed_sum`
- `OUT_ClaimID_count`

5.3. Label Merge

Provider-level aggregates were joined with:

- `ProviderID, PotentialFraud`

5.4 Output File

The final engineered dataset was exported as:

provider_features_final.csv
(Required by Phase 1 rubric)

---

6. Modeling Methodology

Models were trained inside `02_modeling.ipynb`.

6.1. Feature Scaling

Required only for Logistic Regression:

- StandardScaler used
- Random Forest does not require scaling

6.2. Train/Test Split

- `75% training`
- `25% testing`
- `stratified by fraud label`

6.3. Models Tested

Two baseline models were selected:

Logistic Regression

- Interpretable
- Good for linear feature relationships
- Acts as a baseline comparison

Random Forest

- Handles nonlinear relationships
- Robust to noise

- Good performance on tabular data

---

7. Hyperparameter Experiments

7.1 Logistic Regression Experiments

Parameter	Tried	Reason
Penalty	l2	stable baseline
C	0.5, 1.0, 2.0	controls regularization
Solver	liblinear, lbfgs	compatibility testing

Best: `C=1.0`, `l2 penalty`

7.2 Random Forest Experiments

Parameter	Tried	Reason
n_estimators	100, 200, 300	more trees → better stability

max_depth	None, 10, 20	control overfitting
min_samples_split	2, 5	check generalization

Best:

- `n_estimators=200`
- `max_depth=None`

Random Forest clearly outperformed Logistic Regression.

---

8. Final Model Performance

Random Forest Results (Best Model)

Metric	S
Accuracy	9
Precision (Fraud)	0.

Recall (Fraud)	0.
F1 Score	0.

Recall matters most—missing fraudulent providers is costly.
Capturing **67%** is strong given imbalanced data.

---

9. Error Analysis

9.1 False Positives

Providers incorrectly flagged as fraud.

Patterns:

- unusually high reimbursement
- but normal LOS and visit patterns

Implication:

- Audit cost increases
- However, low risk financially

9.2 False Negatives

Fraudulent providers the model failed to detect.

Patterns:

- moderate reimbursement totals
- small frequent claims overlooked
- less extreme LOS patterns

Implication:

- Real financial loss
- These cases highlight the limits of simple features
- Future work should improve recall further

---

10. Feature Importance Insights

Top features (from Random Forest):

1. **INP_LOS_max**
2. **INP_InscClaimAmtReimbursed_sum**
3. **OUT_ClaimID_count**
4. **OUT_InscClaimAmtReimbursed_sum**
5. **INP_LOS_mean**

Fraudulent providers tend to:

- keep patients longer
- bill higher reimbursements
- submit more outpatient claims

---

11. Business Impact

The model helps:

- Prioritize investigations
- Reduce fraudulent payments
- Protect Medicare funds
- Automatically flag high-risk providers
- Improve operational auditing efficiency

Given limited resources, a fraud-recall rate of **67%** significantly improves targeting.

---

12. Future Work

To enhance performance:

✔ Add XGBoost / LightGBM

Advanced tree models outperform Random Forest on structured data.

✔ Apply SMOTE

Balances fraud vs. non-fraud classes.

✔ Generate diagnosis code embeddings

Transform ICD codes into vector representations.

✔ Deploy model

Expose REST API for real-time fraud scoring.

---

13. Conclusion

This project successfully developed a full machine learning pipeline capable of identifying high-risk healthcare providers. The modeling process, feature engineering, and evaluation steps produce a strong, defendable fraud-detection system with meaningful business value.

The Random Forest model provides the best trade-off between accuracy, fraud recall, and robustness, and serves as a strong foundation for future enhancements.

-