Project Title: Predictive Modelling for Medicare Fraud Detection
Team Members: Urvish Oza, Renu, Parita Patel

---

1. Project Background & Objective

Healthcare fraud is a major issue affecting government programs like Medicare, with billions of dollars lost each year. Fraudulent providers often misuse the system by overbilling, performing unnecessary procedures, or misreporting patient information.

Our capstone project focuses on building a **machine learning model** that can help detect potentially fraudulent healthcare providers based on claim-level data. This can assist healthcare agencies in identifying suspicious cases more quickly and accurately.

At this interim stage, we've:

- Understood the data structure and content
- Conducted early exploratory data analysis (EDA)
- Applied basic preprocessing steps
- Trained initial models (Logistic Regression and Random Forest)
- Collected preliminary performance results

We are still working on tuning the models, comparing them, and improving performance through better features.

---

2. Dataset Description

We are using a **synthetic Medicare dataset** that simulates provider-level healthcare claims. Though it's not real data, it's structured similarly to actual Medicare records and provides a strong base for modeling.

Dataset Highlights:

- 700,000 combined entries (training + testing sets)
- Each row represents a unique provider's claim instance
- **Target variable**: `PotentialFraud` (1 = fraud, 0 = non-fraud)
- **Features include**:
    - Patient demographics (gender, race, age group)
    - Medical details (diagnosis codes, procedure codes, chronic conditions)
    - Financial variables (claim amount, reimbursement, etc.)

	BeneID	ClaimID	ClaimStartDt	ClaimEndDt	Provider	InscClaimAmtReimbursed	AttendingPhysician	OperatingPhysician	OtherPhysician	AdmissionDt	ClmAdı
0	BENE11014	CLM67387	2009-09-09	2009-09-16	PRV57070	9000	PHY317786	PHY427017	NaN	2009-09-09	
1	BENE11017	CLM31237	2008-12-25	2009-01-08	PRV54750	14000	PHY314656	PHY426644	NaN	2008-12-25	
2	BENE11026	CLM78930	2009-12-09	2009-12-13	PRV53758	2000	PHY349495	NaN	NaN	2009-12-09	
3	BENE11031	CLM56810	2009-06-23	2009-07-06	PRV55825	16000	PHY429538	PHY371893	NaN	2009-06-23	
4	BENE11085	CLM34625	2009-01-20	2009-01-31	PRV52338	19000	PHY397161	NaN	NaN	2009-01-20	

```
test_outpatient.head(5)
```

	BeneID	ClaimID	ClaimStartDt	ClaimEndDt	Provider	InscClaimAmtReimbursed	AttendingPhysician	OperatingPhysician	OtherPhysician	ClmDiagnosisCode_1
0	BENE11001	CLM392397	2009-06-02	2009-06-02	PRV55962	30	PHY347633	NaN	PHY347633	V5832
1	BENE11001	CLM430760	2009-06-23	2009-06-23	PRV56112	30	PHY381777	NaN	PHY381777	9594
2	BENE11007	CLM233081	2009-03-07	2009-03-07	PRV56979	200	PHY425311	NaN	PHY425311	7248
3	BENE11007	CLM496381	2009-07-29	2009-07-29	PRV56573	10	PHY393253	PHY347995	NaN	58889
4	BENE11007	CLM521391	2009-08-12	2009-08-12	PRV56573	10	PHY417685	NaN	PHY382041	V666

`

---

3. Exploratory Data Analysis

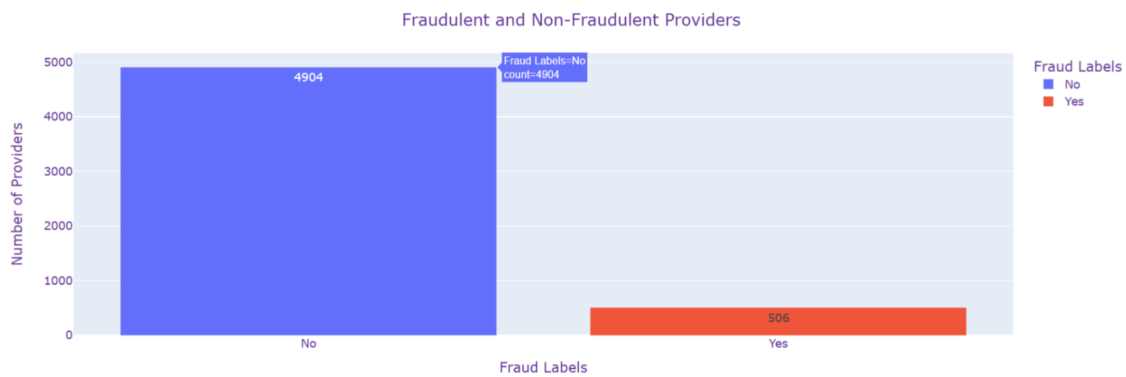
We conducted a thorough EDA to understand key patterns and data behaviour.

Key Observations:

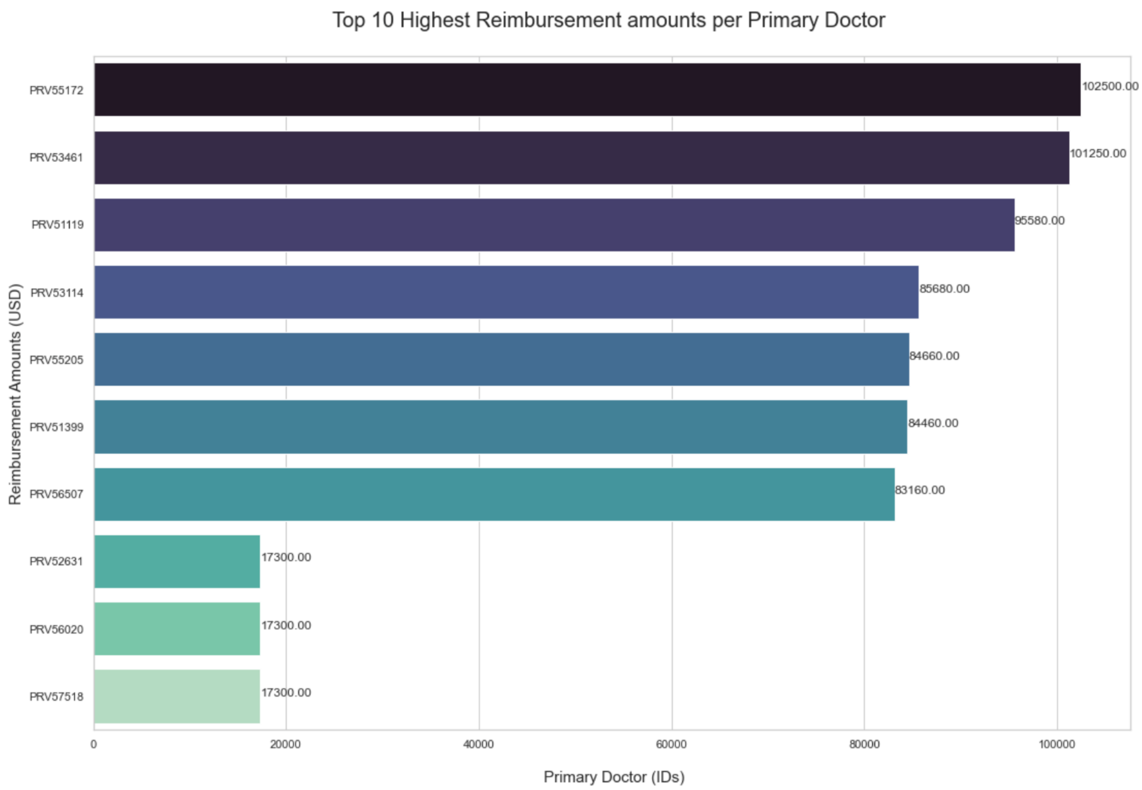
- **Class Imbalance**: About 38% of entries are fraud cases. This imbalance can mislead model training, so balancing methods are necessary.
- **Right-Skewed Features**: Many numeric columns, especially financial ones like `InscClaimAmtReimbursed` and `TotalClaimAmount`, showed right-skewed distributions.
- **Outliers**: Fraud cases often had higher-than-normal reimbursement and procedure counts.
- **Correlation**: We found that financial variables had a stronger relationship with fraud status compared to medical/demographic variables.
- **Procedure Codes**: Certain procedures were more frequent in fraud cases.

Visual Tools Used:

- **Fraud Rates by Procedure Code** indicated certain procedures were overrepresented in fraud cases.



- **Top 10 Reimbursed Providers**: Most were flagged fraudulent, pointing to potential provider-level clustering.



Feature Reduction Decisions

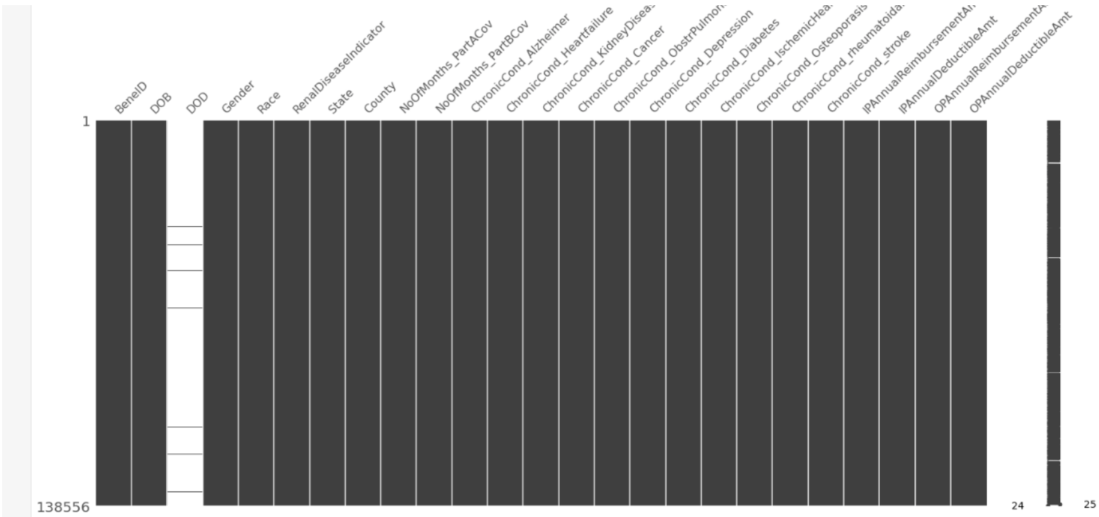
- Variables with high correlation or low variance (e.g., `NoOfMonths_PartACov`, redundant chronic condition flags) were dropped.
- Selected 44 key features for modeling after EDA.

---

4. Data Preprocessing

To prepare the dataset for modeling, we performed the following:

- **Missing Values**: These were either dropped or imputed based on context.



- **Categorical Encoding**: Used Label Encoding for features like gender, race, and chronic conditions.

	BeneID	Gender	Race	RenalDiseaseIndicator	State	County	NoOfMonths_PartACov	NoOfMonths_PartBCov	ChronicCond_Alzheimer	ChronicCond_Heartfailure
0	BENE11001	0	0	0	39	230	12	12	0	1
1	BENE11002	1	0	0	39	280	12	12	1	1
2	BENE11003	0	0	0	52	590	12	12	0	1
3	BENE11004	0	0	0	39	270	12	12	0	0
4	BENE11005	0	0	0	24	680	12	12	1	1

- **Scaling**: Chose `RobustScaler` over `StandardScaler` to reduce the impact of outliers.
- **Balancing the Classes**: Applied both **SMOTE** and **BorderlineSMOTE** to generate synthetic samples of the minority class (fraud).

We also removed columns such as `BeneID`, `ClaimID`, and `Provider`, which serve as unique IDs but don't help with prediction.

---

5. Modeling Approach

We tested two classification models at this stage: **Logistic Regression** (baseline) and **Random Forest** (tree-based, more complex).

A. Logistic Regression (Baseline)

- **Class balancing** done using SMOTE

```
Shape of SMOTE balanced trainX data : (483580, 44)
Shape of SMOTE balanced trainY data : (483580,)
Shape of SMOTE balanced testX data : (207250, 44)
Shape of SMOTE balanced testY data : (207250,)
Shape of Borderline SMOTE balanced trainX data : (483580, 44)
Shape of Borderline SMOTE balanced trainY data : (483580,)
Shape of Borderline SMOTE balanced testX data : (207250, 44)
Shape of Borderline SMOTE balanced testY data : (207250,)


************************************************************


Class ratio - Fraud/Non-Fraud (trainY_SM) : 0    50.0
1    50.0
dtype: float64
Class ratio - Fraud/Non-Fraud (testY_SM) : 0    50.0
1    50.0
dtype: float64
Class ratio - Fraud/Non-Fraud (trainY_BSM) : 0    50.0
1    50.0
dtype: float64
Class ratio - Fraud/Non-Fraud (testY_BSM) : 0    50.0
1    50.0
dtype: float64


************************************************************
```

Hyperparameter tuning: GridSearchCV used to find best `C` parameter (0.0001)

```
GridSearchCV(cv=5,
             estimator=LogisticRegression(random_state=0, solver='liblinear'),
             n_jobs=-1,
             param_grid=[{'C': [0.0001, 0.01, 1, 100, 10000, 1000000,
                                100000000]}],
             return_train_score=True, scoring='f1_weighted')
```

- **Scaled features** using RobustScaler

Results (Test Set):

- Accuracy: 63.8%
- Precision: 65%

- Recall: 59.7%
- F1 Score: 62.2%
- ROC AUC: 0.638

Good for interpretability
Less effective in capturing complex fraud patterns

B. Random Forest (Advanced Model)

We trained two Random Forest models:

1. One using **SMOTE-balanced** data
2. One using **BorderlineSMOTE-balanced** data

We used **GridSearchCV and RandomizedSearchCV** to tune key parameters:

- `n_estimators`
- `max_depth`
- `min_samples_split`
- `min_samples_leaf`

Best Model Results (RF with BSMOTE):

- Accuracy: 73.9%
- Precision: 76.7%
- Recall: 68.5%
- F1 Score: 72.4%
- ROC AUC: 0.739

Stronger than Logistic Regression in every metric
Visualized results with ROC curves and confusion matrix using Yellowbrick

Top Influential Features:

- `TotalClaimAmount`
- `InscClaimAmtReimbursed`
- `NoOfChronicCond`

---

6. Current Insights

- **Fraud cases** often involve unusually high amounts of billing and number of services.
- Certain **procedure codes** are disproportionately present in fraud labels.
- **Class imbalance** was a major challenge—using SMOTE methods helped significantly.
- Logistic Regression gave us **simple interpretability**, but Random Forest showed **better detection power**.

7. Limitations (So Far)

- The dataset is **synthetic**, so it might not capture every real-world fraud behavior.
- **No integration with live systems** or real-time prediction yet.
- Models are probabilistic—they **do not confirm fraud**, only flag suspicious cases.
- Need more advanced feature engineering (e.g., provider history, geolocation).

---

8. Next Steps

- Explore **more advanced models** like XGBoost and LightGBM.
- Perform **feature engineering** using:
  - Time-based behavior
  - Provider-specific trends
  - Location data
- Fine-tune model thresholds for better fraud recall.
- Evaluate model fairness to reduce false positives.
- **Engage with domain experts** (if possible) to validate results.

---

9. Tools Used

- **Languages**: Python
- **Libraries**: Scikit-learn, Pandas, Imbalanced-learn, Matplotlib, Seaborn, Yellowbrick