Medical Fraud Detection

Team Members:

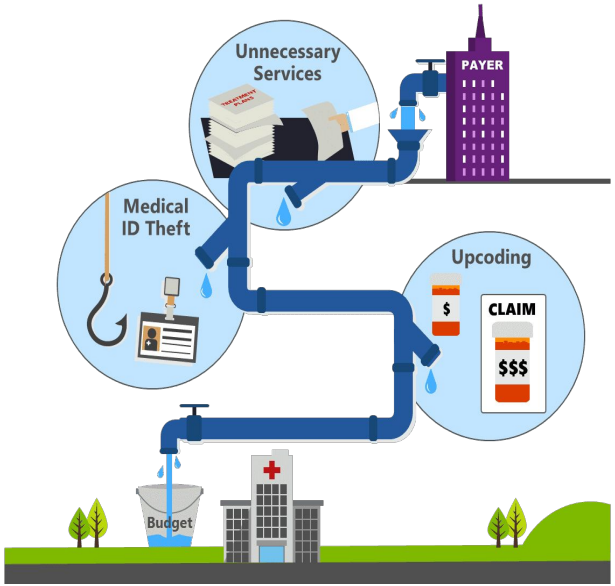
Neel Shah
Sagnik Ghosal
Prerit Chaudhary
Ishank Vasania

Background and Problem Statement

Provider fraud is a significant issue in the Medicare system, leading to increased healthcare costs due to fraudulent claims. These fraudulent claims can involve providers, physicians, and beneficiaries colluding to submit false claims for medical services. These bad practices impact insurance companies heavily, leading to increased insurance premiums and making healthcare more expensive for everyone.

We worked on building a tool that can identify identify fraudulent claims and also get insights about key factors contributing to frauds. The Infrastructure consists of 2 main parts:

- Visualization
- Prediction



Data

Beneficiary Data

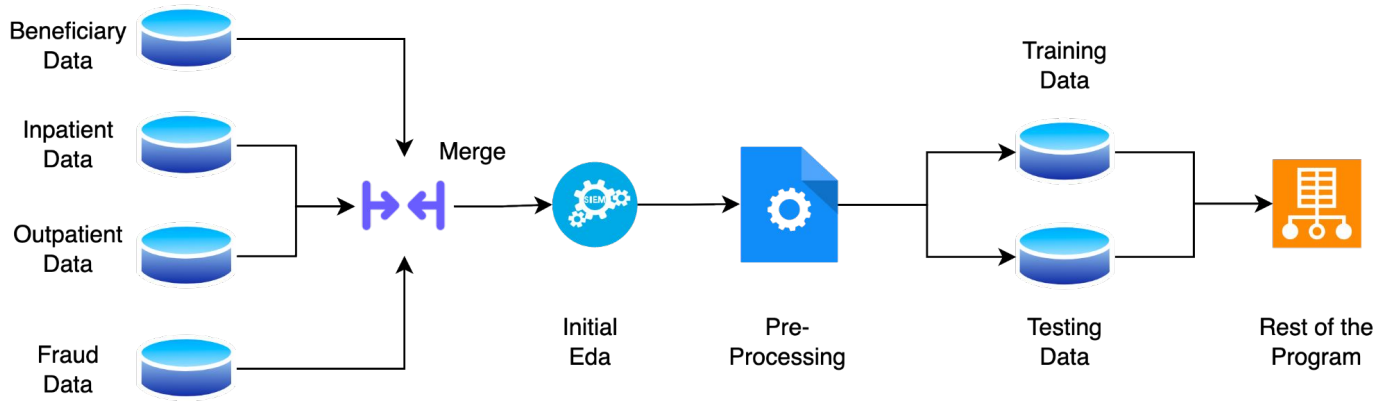
Inpatient Data

Outpatient Data

Fraud Data

- **Beneficiary Data**:
  - Demographic details of healthcare insurance plan beneficiaries
  - 138,556 rows and 25 attributes.
  - Beneficiary ID, DOB, gender, country, existing medical conditions etc.
- **Inpatient Data**:
  - Medical data of patients hospitalized for a night or longer.
  - 40,474 rows and 30 attributes.
  - Beneficiary ID, Provider ID, Claim start date, attending physician etc.
- **Outpatient Data**:
  - Medical data of patients not hospitalized overnight or longer.
  - 517,737 rows and 27 attributes
  - Beneficiary ID, Provider ID, Claim start date, attending physician etc.
- **Fraud Data**:
  - Fraud Yes/No Data
  - CSV file having 5410 rows and 2 attributes, which are Provider id, Fraud yes or no.

Data Flow



Merge

- Fraud, Beneficiary, inpatient, and Outpatient - based on the BeneID, Provider and other required columns,
- Saves as a CSV file in the data folder.

EDA

- Performs initial EDA on a merged dataset to provide insights
- Unique values, Dimensions, Missing data, Admission analysis, Correlation, and Visualization Plots

Preprocessing

- Module for data preprocessing for fraud detection
- Data Encoding, Categorical data, Derived Metrics
- Train - Test Data

Process Flow

Data Processing | Modeling | UI Design | Testing and CI

Healthcare Data
- Data Ingestion
- Data Cleaning
- Data Processing
- Exploratory Data Analysis
- Train - Validation - Test Data

Supervised:
- Logistic Regression
- Random Forest
- XGBoost

Unsupervised:
- Isolation Forest
- CPOD

Home Page:
- Fraud Visualizations
- Prediction console

Prediction Page:
- Model Performance
- Visualization
- Prediction Upload and Download

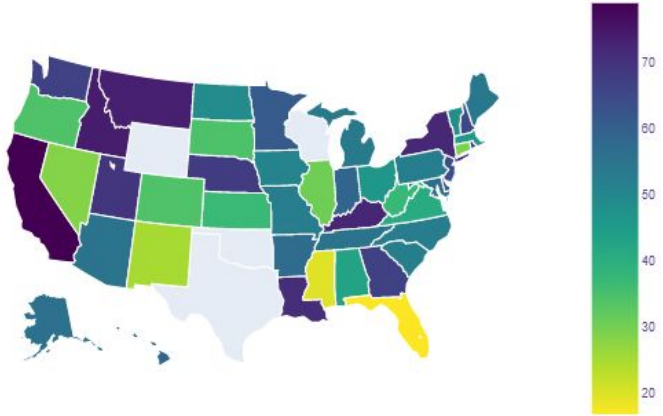
Testing:
- Unit Tests
- Smoke Tests
- Edge Tests

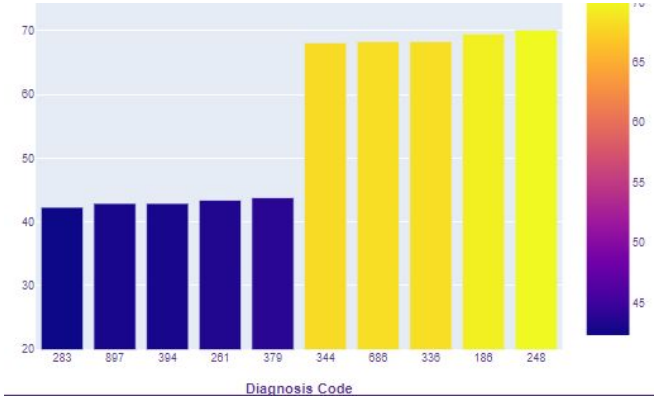
GIT Actions:
- Environment
- Build
- Unit Tests
- Pylint

Use Cases 1

Visualizing Fraud Patterns in Medicare Claims Data

- **Objective***:* The objective of this use case is to create informative and intuitive visualizations that can effectively showcase the patterns and trends of fraudulent activities within Medicare claims data. The goal is to help identify potential fraudulent cases and make data-driven decisions to prevent and reduce fraud.





Use Case 2

Identifying Potentially Fraudulent Claims for Review in a New Dataset

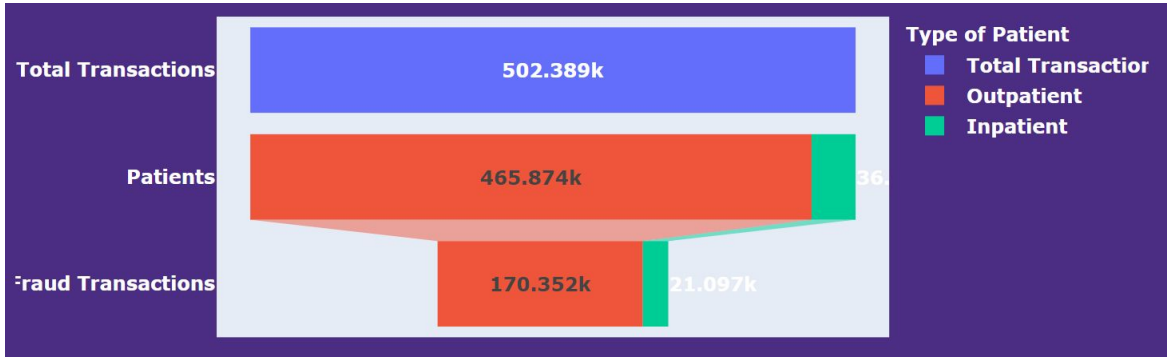
- **Objective***:* The objective for this use case is to develop a fraud detection system that can analyze a new dataset of medical claims and identify potentially fraudulent claims for review. The system should use machine learning algorithms to identify patterns and anomalies in the data and generate a list of flagged claims.

- **Expected Interactions**: The user's primary objective is to identify potentially fraudulent claims for review in a new dataset. The user would first need to upload the new dataset containing medical claims data to the system. Once the detection process is complete, the system would generate a list of potentially fraudulent claims. The user would then review these flagged claims and instigate appropriate investigations.

Use Case 3

Analyzing Medicare Claims with Visualizations for a New Dataset

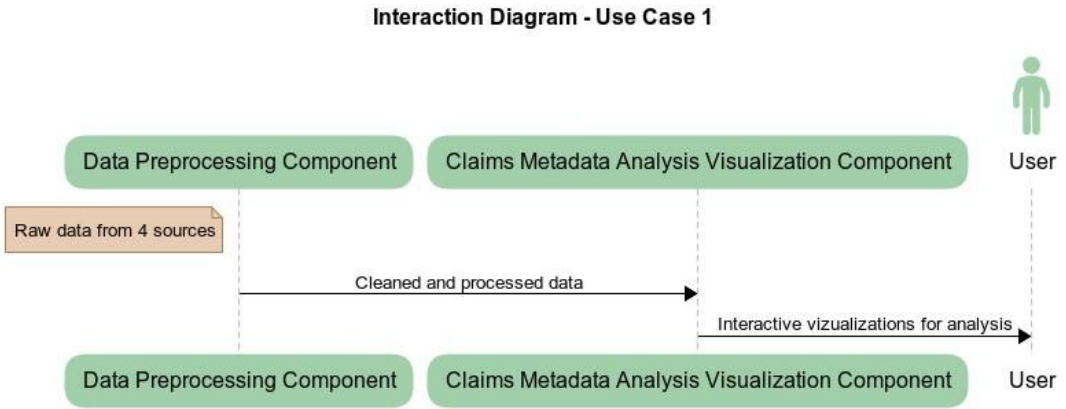
- **Objective***:* The objective of this use case is to analyze a new dataset of Medicare claims using visualizations such as bar charts, pie-charts, and funnel charts. The analysis will focus on metrics related to the percentage of fraud and non-fraud, as well as the number of fraudulent cases in inpatient and outpatient cases.



Design

Use Case 1: Visualizing Fraud Patterns in Medicare Claims Data

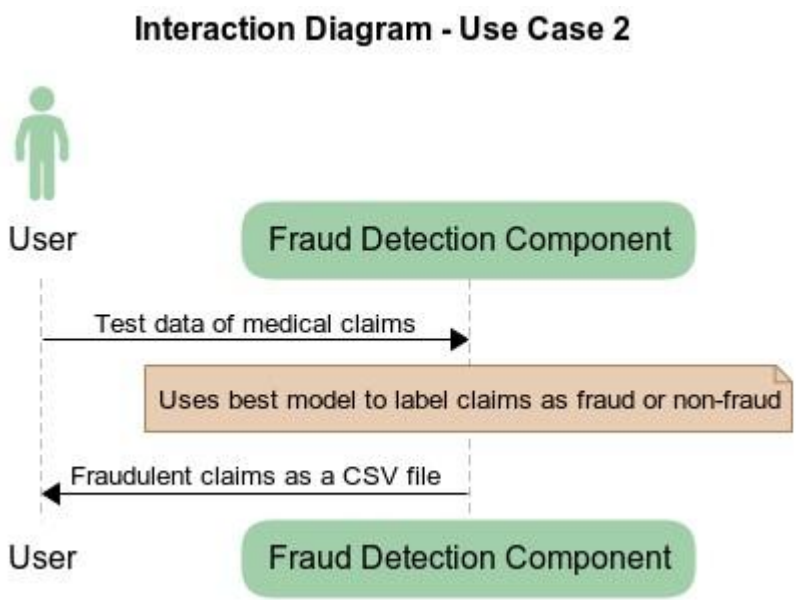
- The Data Preprocessing Component is responsible for taking in raw data from 4 sources and processes it for analysis.
- The preprocessed data is passed onto the Claims Metadata Analysis Visualization Component, which generates visualizations to provide valuable insights.



Design

Use Case 2: Identifying Potentially Fraudulent Claims for Review in a New Dataset

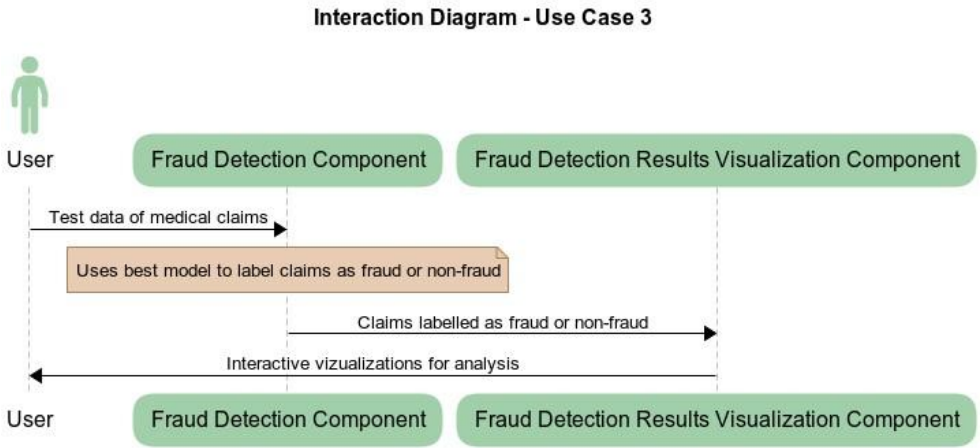
- The Fraud Detection Component performs fraud detection on a new test dataset provided by the user.
- It uses the best fraud detection model to label each claim in the test dataset with a 0 or 1, where 0 indicates a non-fraudulent claim and 1 indicates a fraudulent claim.
- The claims that are labeled as 1 are then written to a CSV file, which can be downloaded by the user by clicking on the download button.



Design

Use Case 3: Analyzing Medicare Claims with Visualizations for a New Dataset

- Fraud Detection Component is responsible for analyzing a new test dataset provided by the user.
- It utilizes the best fraud detection model to label each claim in the test dataset with either 0 (non-fraudulent) or 1 (fraudulent).
- The labelled data is passed to the Fraud Detection Results Visualization Component, which generates visualizations to offer various insights.



Demo

Visualize the tool at: **http://127.0.0.1:5000/**

Link to recording of the demo:
https://drive.google.com/file/d/1fK4CRSSY7eo01fwx88FSXHJc1CeLT9lQ/view?usp=sharing

Lessons Learned and Future Work

Software engineering lessons learned from the project:

- Deciding the scope of the project
- Implementing and deploying trained models in the backend for a tool
- Importance of having modular code to enhance reusability
- Writing test cases which can capture corner cases

The future releases may include the following:

- Improve latency time or response time for dynamically uploaded data
- Improve prediction performance by including powerful yet computationally lightweight ML and DL models for improved performance
- Dynamic UI for different resolutions

Thank You!