HealthCare Provider Fraud Detection

- Satya Venkataswamy

Objectives and Overview

OBJECTIVE

The primary objective is a **binary classification task** of each of the provider as either **a Fraud or a Non-Fraud**. Since the data is related to fraud, the dataset is **imbalanced (Positive : Negative= 60:40).**

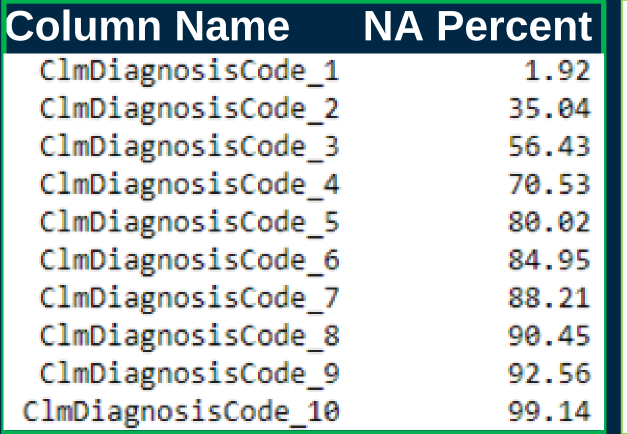
Evaluation Metric

- **Precision and Recall**: Since we are dealing with a fraud identification problem with a **class imbalance**, metrics such as **Accuracy does not precisely** evaluate the models.
- **ROC/AUC curve:** Primarily used the ROC curve to arrive at the best probability value
- **F1 score**: As both precision and recall are important

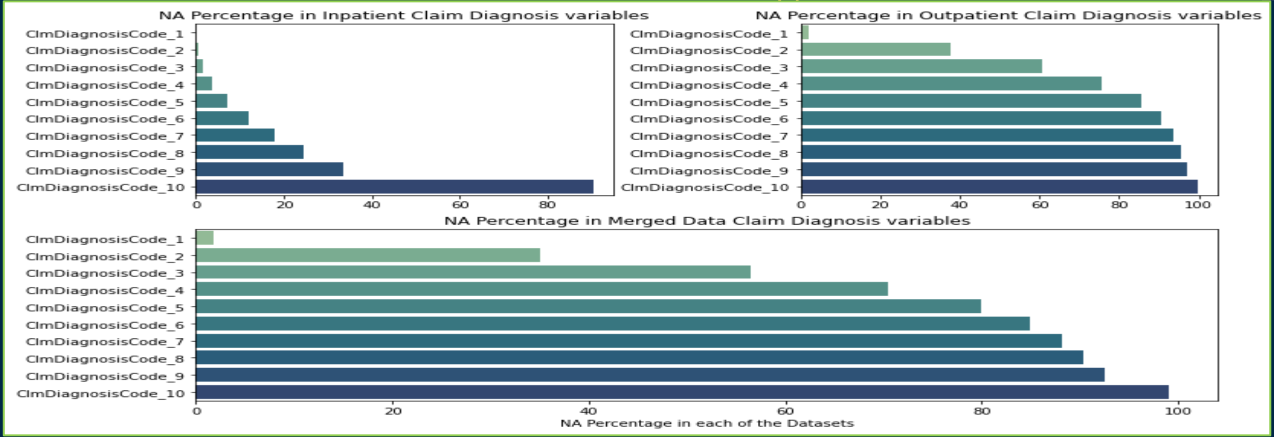
Data Overview

- The data was provided as **3 separate datasets – Outpatient Data**, **Inpatient Data** and **Beneficiary Data**
- A total of **500,000+ Observations and 55 attributes** were provided: **21 Features Categorical** in nature

Imputation And Feature Engineering



Column Name	NA Percent
ClmDiagnosisCode_1	1.92
ClmDiagnosisCode_2	35.04
ClmDiagnosisCode_3	56.43
ClmDiagnosisCode_4	70.53
ClmDiagnosisCode_5	80.02
ClmDiagnosisCode_6	84.95
ClmDiagnosisCode_7	88.21
ClmDiagnosisCode_8	90.45
ClmDiagnosisCode_9	92.56
ClmDiagnosisCode_10	99.14



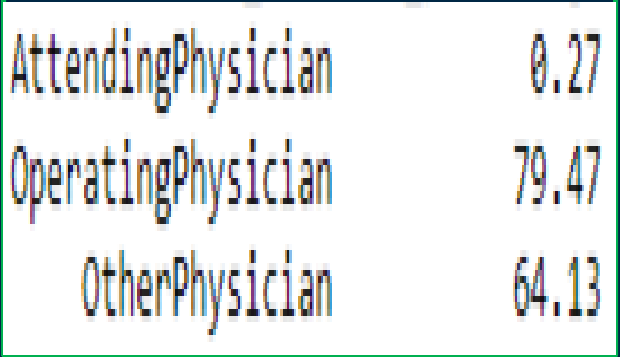
Key Observations

- At the first glance it seems that there is a very high percentage of missing values in the merged dataset.
- A deeper secondary research revealed that Inpatients are admitted with complicated ailments than Outpatients
- High percentage of the NA values in the merged datasets is not due to missing data but due to the reason that the size of the Outpatient dataset is much higher than the Inpatient dataset

Feature Engineering

- Each of the claim diagnosis codes indicates a different diagnosis hence the counting the number of diagnosis performed effectively captures the information carried by the 10 different claim procedure columns.
- I have created a new feature capturing the number of diagnosis performed for each of the patients. Higher the number of procedures it is highly likely that higher is the complexity of the case.

Imputation And Feature Engineering



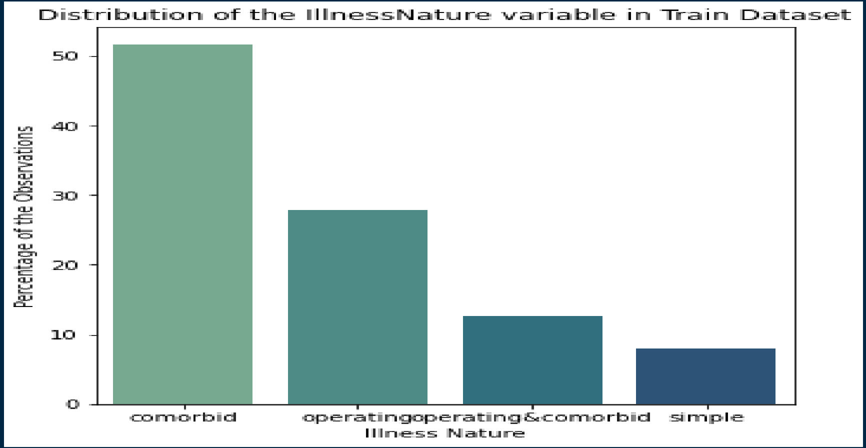
Column Name	NA Percent
AttendingPhysician	0.27
OperatingPhysician	79.47
OtherPhysician	64.13

Key Observations

- NA values in Operating Physician and Other Physician datasets doesn't mean that the data is missing.
- As per my secondary research, Operating Physicians are involved in cases where a surgery or other complications are involved
- OtherPhysicians are involved in cases where the patient has co-morbidities

Feature Engineering

- A new column has been created to capture the nature of illness of the patient. This column will be categorical and will have the below categories: Simple, Operating, Comorbid, Operating & comorbid.
- Comorbid condition or the illness nature has the highest percentage of the observations in the overall dataset followed by the Operating illness nature.



Data Pre-processing

Converting the Categorical Columns into Numerical Columns

- Majority of the categorical columns have more than 50-100 categories in each of the columns
- Adopting a one-hot encoding to convert the categorical columns into numerical columns could lead to creation of a lot of columns leading to Curse of Dimensionality
- Used response coding in order to convert categorical columns to numerical ones.

Response Coding

- Response Coding: Calculating the probabilities of each of the categories in a column. Probability is calculated as follows
- P(x=c1/y='yes') which is Probability of category in column X, given the Y variable belongs to class 'Yes' and class 'No'
- P(x=c1/y='yes')= (Number of Occurrences of C1 where Y belongs to 'yes' class) divided by (Number of Occurrences of where Y='yes' + Number of Occurrences of where Y='No')

Experimentation and Model Selection

Experiments Performed

- **Experiment 1:** Supervised Classification Models **WITHOUT** adjusting the **class weights (Class Imbalance)**
- **Experiment 2: Adjusting for Class Imbalance** by choosing the appropriate **Class Weights parameter**
- **Experiment 3: Calculate VIF** values to look for Dummy Variable Trap and **remove features with very high VIF values**
- **Experiment 4:** Using **SMOTE Oversampling** technique in order to create a **dataset corrected for Data imbalance**

Supervised Models Used across all the above Experiments

- **Logistic Regression:** Since the dataset is huge and computational simplicity, I chose Logistic Regression over SVM amongst the Linear classifiers.
- **XGBoost Classifier:** The results of logistic regression indicated more bias and less variance hence chose boosting in order to reduce the bias.
- **LightGBM Classifier:** Secondary research indicated that Light GBM implementation of Boosting performed better than the XGBoost implementation hence experimented with LightGBM.
- **Random Forests Classifier**: Experimented with RF just to validate that my previous assumption that Boosting will perform better than Bagging due to the high bias and low variance problem

Results of Best Experiment – Experiment 1

Model	Train Data		Cross-validate Data	
	Precision	Recall	Recall	Precision
Logistic Regression	0.62996	0.61273	0.62978	0.61275
XGBoost	0.863	0.862	0.866	0.870
Light GBM	0.997118	0.997119	0.97790	0.97794
Random Forests	0.87742	0.87820	0.87157	0.87152

Questions?

THANKS