

# HEALTHCARE PROVIDER FRAUD DETECTION ANALYSIS

## Project Report

Akilesh Vishnu Mohan Raj  
Kirthana Shri Chandra Sekar  
Neeraj Kamal Rangwani  
Samarth Saxena

[mohanraj.a@northeastern.edu](mailto:mohanraj.a@northeastern.edu)  
[chandrasekar.k@northeastern.edu](mailto:chandrasekar.k@northeastern.edu)  
[rangwani.n@northeastern.edu](mailto:rangwani.n@northeastern.edu)  
[saxena.sam@northeastern.edu](mailto:saxena.sam@northeastern.edu)

Percentage of Effort Contributed by Student1: 25%

Percentage of Effort Contributed by Student2: 25%

Percentage of Effort Contributed by Student3: 25%

Percentage of Effort Contributed by Student4: 25%

Signature of Student 1:	<u>Akilesh Vishnu Mohan Raj</u>
Signature of Student 2:	<u>Kirthana Shri Chandra Sekar</u>
Signature of Student 3:	<u>Neeraj Kamal Rangwani</u>
Signature of Student 4:	<u>Samarth Saxena</u>

Submission Date: 04/28/2023

## **Table of Contents**

1. Problem Setting	3
2. Problem Statement	3
3. Data Sources	4
4. Data Description	4
5. Data Preprocessing	4
6. Data Exploration	6
7. Data Partitioning	10
8. Data Mining Models/Methods	10
9. Project Results	14
10. Future Scope	15

## **1. Problem Setting**

Provider Fraud is a major problem that continues to plague the Medicare system. In fact, according to the government, fraudulent activities in Medicare claims have caused a significant increase in the overall Medicare spending. Healthcare fraud is a type of organized crime that involves different parties, such as providers, physicians, and beneficiaries, working together to commit fraudulent activities. Through careful analysis of Medicare data, many physicians have been found to engage in fraudulent activities. They often use ambiguous diagnosis codes to justify the use of expensive procedures and drugs. Insurance companies are particularly vulnerable to these practices, which have led to increased insurance premiums and a general rise in healthcare costs.

Healthcare fraud and abuse come in various forms, but some of the most common types of provider frauds include:

- Billing for services that were never provided.
- Submitting duplicate claims for the same service.
- Misrepresenting the services provided.
- Charging for a more expensive or complex service than what was actually provided.
- Billing for a covered service when the service provided was not covered.

These fraudulent activities are not only illegal but also have serious consequences on the Medicare system, patients, and the healthcare industry as a whole. Therefore, it is essential to put in place measures to prevent and detect healthcare fraud and abuse.

## **2. Problem Statement**

The primary objective is to predict potentially fraudulent healthcare providers by analyzing their claims history and identifying key variables for detecting fraudulent behavior. It also involves studying fraudulent patterns to gain insights into future behavior. The ultimate goal is to improve the detection and prevention of healthcare fraud and abuse.

### **3. Data Sources**

The dataset for this project was obtained from Kaggle and includes information on healthcare providers in USA. By analyzing this data, we aim to identify potentially fraudulent healthcare providers and gain insights into the patterns and variables that can help detect and prevent healthcare fraud and abuse.

Link: <https://www.kaggle.com/datasets/rohitrox/healthcare-provider-fraud-detection-analysis>

### **4. Data Description**

The dataset consists of 558K records and 56 columns. It includes information on healthcare providers' Inpatient claims, Outpatient claims, and Beneficiary details. Here is a brief overview of each category:

- A) *Inpatient Data*: This data provides information on claims filed for patients who have been admitted to hospitals. It includes details such as admission and discharge dates, as well as admission diagnosis codes.
- B) *Outpatient Data*: This data provides information on claims filed for patients who have received healthcare services at hospitals but were not admitted as inpatients.
- C) *Beneficiary Details Data*: This data contains details on beneficiary KYC (know your customer) information, such as their health conditions and the region they belong to.

### **5. Data Preprocessing**

#### **#1 - Merging all csv files**

To create a comprehensive dataset for this project, we merged the CSV files for Inpatient claims, Outpatient claims, and Beneficiary details based on both the Beneficiary ID and Provider ID. We used the **merge()** function to perform the merging operation. To check the resulting size of the dataset, we used the shape function, which returns the number of rows and columns in the DataFrame. The resulting merged dataset contains **558,211** rows and **56** columns.

## #2 - Checking the datatype

The method **patient.info()** provides useful information about the data, such as the data types, number of non-null values, range index, and memory usage for each variable in the dataset. This information can be useful for understanding the structure of the data and identifying any potential issues, such as missing or incorrect data types.

Data columns (total 56 columns):			
#	Column	Non-Null Count	Dtype
0	BeneID	558211 non-null	object
1	ClaimID	558211 non-null	object
2	ClaimStartDt	558211 non-null	object
3	ClaimEndDt	558211 non-null	object
4	Provider	558211 non-null	object
5	InscClaimAmtReimbursed	558211 non-null	int64
6	AttendingPhysician	556703 non-null	object
7	OperatingPhysician	114447 non-null	object
8	OtherPhysician	199736 non-null	object
9	AdmissionDt	40474 non-null	object
10	ClmAdmitDiagnosisCode	145899 non-null	object
11	DeductibleAmtPaid	557312 non-null	float64
12	DischargeDt	40474 non-null	object
13	DiagnosisGroupCode	40474 non-null	object
14	ClmDiagnosisCode_1	547758 non-null	object
15	ClmDiagnosisCode_2	362605 non-null	object
16	ClmDiagnosisCode_3	243055 non-null	object
17	ClmDiagnosisCode_4	164536 non-null	object
18	ClmDiagnosisCode_5	111924 non-null	object
19	ClmDiagnosisCode_6	84392 non-null	object
20	ClmDiagnosisCode_7	66177 non-null	object
21	ClmDiagnosisCode_8	53444 non-null	object
22	ClmDiagnosisCode_9	41815 non-null	object
23	ClmDiagnosisCode_10	5010 non-null	object
24	ClmProcedureCode_1	23310 non-null	float64
25	ClmProcedureCode_2	5490 non-null	float64
26	ClmProcedureCode_3	969 non-null	float64
27	ClmProcedureCode_4	118 non-null	float64
28	ClmProcedureCode_5	9 non-null	float64
29	ClmProcedureCode_6	0 non-null	float64
30	type	558211 non-null	object
31	DOB	558211 non-null	object
32	DOD	4131 non-null	object
33	Gender	558211 non-null	int64
34	Race	558211 non-null	int64
35	RenalDiseaseIndicator	558211 non-null	object
36	State	558211 non-null	int64
37	County	558211 non-null	int64
38	NoOfMonths_PartACov	558211 non-null	int64
39	NoOfMonths_PartBCov	558211 non-null	int64
40	ChronicCond_Alzheimer	558211 non-null	int64
41	ChronicCond_Heartfailure	558211 non-null	int64
42	ChronicCond_KidneyDisease	558211 non-null	int64
43	ChronicCond_Cancer	558211 non-null	int64
44	ChronicCond_ObstrPulmonary	558211 non-null	int64
45	ChronicCond_Depression	558211 non-null	int64
46	ChronicCond_Diabetes	558211 non-null	int64
47	ChronicCond_IschemicHeart	558211 non-null	int64
48	ChronicCond_Osteoporosis	558211 non-null	int64
49	ChronicCond_rheumatoidarthritis	558211 non-null	int64
50	ChronicCond_stroke	558211 non-null	int64
51	IPAnnualReimbursementAmt	558211 non-null	int64
52	IPAnnualDeductibleAmt	558211 non-null	int64
53	OPAnnualReimbursementAmt	558211 non-null	int64
54	OPAnnualDeductibleAmt	558211 non-null	int64
55	PotentialFraud	558211 non-null	object
dtypes: float64(7), int64(22), object(27)			
memory usage: 242.8+ MB			
None			

## #3 - Handling NA values

Using the **'isnull().sum()'** function, we have identified any null or missing values present in the data and dropped those columns which have NA values.

BeneID	0.0	AdmissionDt	92.7
ClaimID	0.0	DischargeDt	92.7
ClaimStartDt	0.0	DiagnosisGroupCode	92.7
ClaimEndDt	0.0	DOB	0.0
Provider	0.0	DOD	99.3
InscClaimAmtReimbursed	0.0	Gender	0.0
AttendingPhysician	0.3	Race	0.0
OperatingPhysician	79.5	RenalDiseaseIndicator	0.0
OtherPhysician	64.2	State	0.0
ClmDiagnosisCode_1	1.9	County	0.0
ClmDiagnosisCode_2	35.0	NoOfMonths_PartACov	0.0
ClmDiagnosisCode_3	56.5	NoOfMonths_PartBCov	0.0
ClmDiagnosisCode_4	70.5	ChronicCond_Alzheimer	0.0
ClmDiagnosisCode_5	79.9	ChronicCond_Heartfailure	0.0
ClmDiagnosisCode_6	84.9	ChronicCond_KidneyDisease	0.0
ClmDiagnosisCode_7	88.1	ChronicCond_Cancer	0.0
ClmDiagnosisCode_8	90.4	ChronicCond_ObstrPulmonary	0.0
ClmDiagnosisCode_9	92.5	ChronicCond_Depression	0.0
ClmDiagnosisCode_10	99.1	ChronicCond_Diabetes	0.0
ClmProcedureCode_1	95.8	ChronicCond_IschemicHeart	0.0
ClmProcedureCode_2	99.0	ChronicCond_Osteoporosis	0.0
ClmProcedureCode_3	99.8	ChronicCond_rheumatoidarthritis	0.0
ClmProcedureCode_4	100.0	ChronicCond_stroke	0.0
ClmProcedureCode_5	100.0	IPAnnualReimbursementAmt	0.0
ClmProcedureCode_6	100.0	IPAnnualDeductibleAmt	0.0
DeductibleAmtPaid	0.2	OPAnnualReimbursementAmt	0.0
ClmAdmitDiagnosisCode	73.9	OPAnnualDeductibleAmt	0.0
type	0.0	PotentialFraud	0.0

## #4 - Handling data type

The dates in the original dataset were stored as strings (data type: 'object'), which is not the ideal format for performing date-related operations. Therefore, the dates were converted into a datetime format using the pandas method **pd.to\_datetime()**.

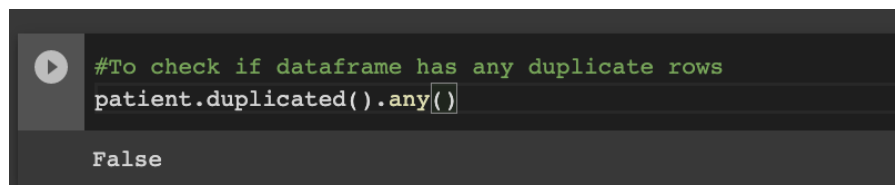
After converting the dates into datetime format, the Claim Start Date and Claim End Date columns were used to calculate the total number of claim days by taking the difference between the two dates. This helps in understanding the duration of the claims filed by each provider and identifying any unusual patterns or outliers in their claims history.

```
patient['#of_Claimdays'] = ((pd.to_datetime(patient['ClaimEndDt']) -  
pd.to_datetime(patient['ClaimStartDt'])) / np.timedelta64(1, 'D')).astype('int32')
```

## #5- Checking for duplicate rows

To check for duplicate rows in the patient dataframe, the **duplicated().any()** method can be used. This method returns a boolean value indicating whether there are any duplicate rows in the dataframe.

After applying to the patient dataframe, it was found that there are no duplicate values in the dataset.



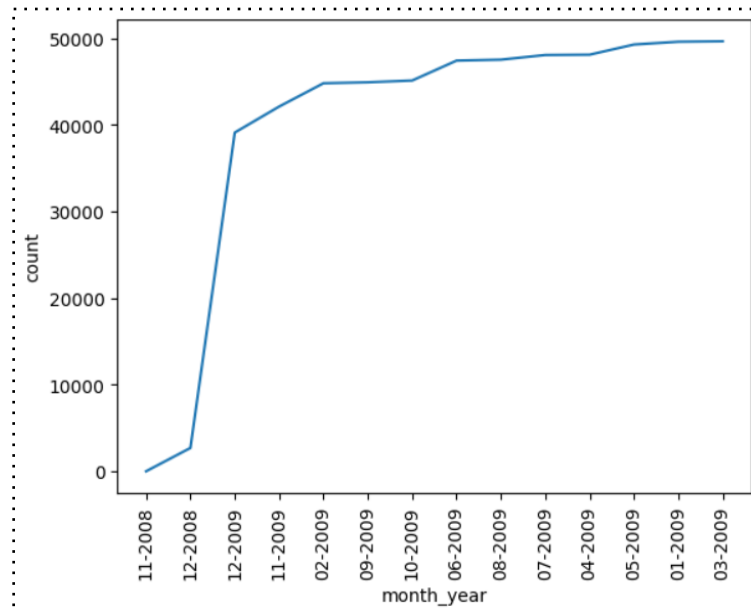
```
#To check if dataframe has any duplicate rows  
patient.duplicated().any()  
  
False
```

New shape of the dataframe patient has 558211 rows and 32 columns.

## 6. Data Exploration

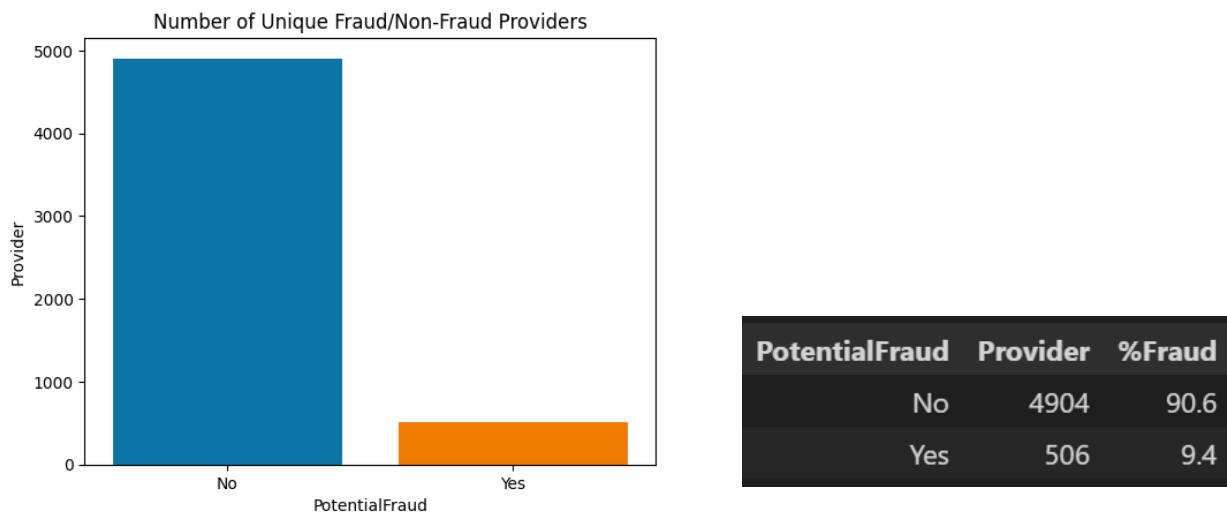
Data exploration is the process of gaining an understanding of the data, identifying patterns and relationships, and detecting potential issues or anomalies. It involves various techniques such as **visualizations**, statistical summaries, and data manipulation to extract insights from the dataset. We can develop an intuition about the data, validate assumptions, and guide the selection of appropriate models and techniques for further analysis. Through data exploration, one can identify important variables, outliers, missing values, and other factors that can impact the quality and accuracy of the analysis.

- A. By visualizing the claims made over time, we can identify monthly trends, and it is observed that the trend has been **continuously increasing** with a positive slope. This might suggest that fraudulent claims have been increasing over time



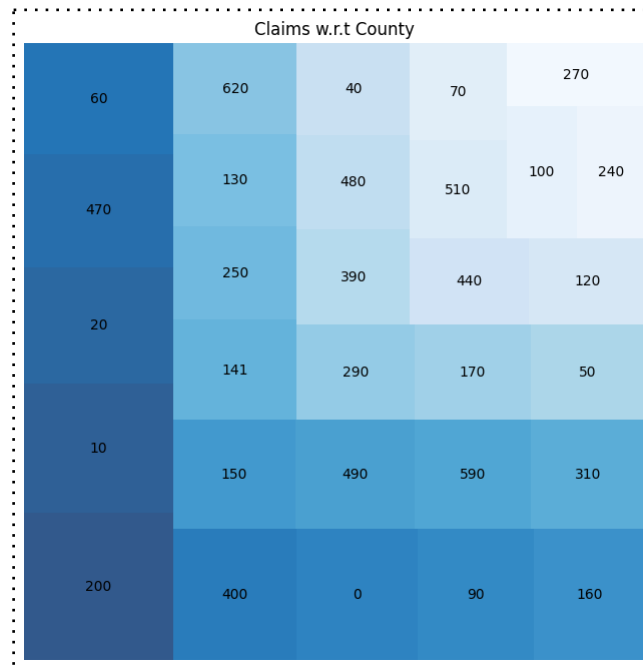
**FIG1 : Trend of beneficiaries taking claim over the months**

- B. Through analysis of the total number of potential fraudulent providers and legitimate providers, it was discovered that approximately 10% of healthcare service providers may be fraudulent. This finding highlights the need for further investigation and identification of potential fraudulent providers in the healthcare industry.



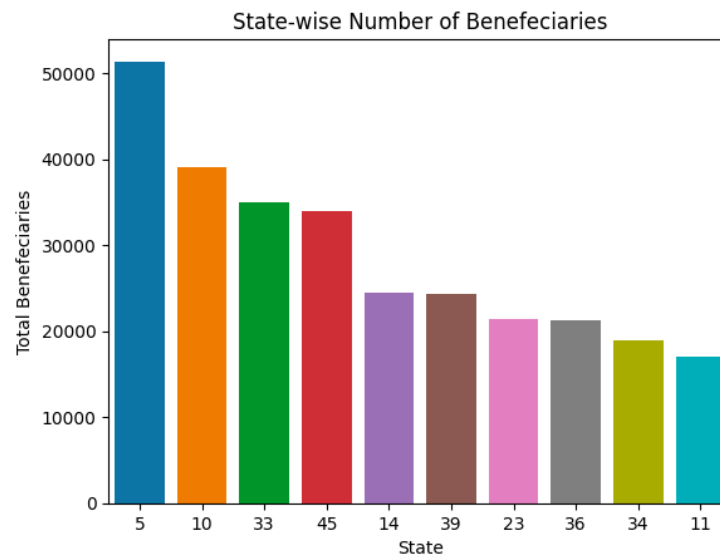
**FIG 2: Distribution of Fraud and Non-Fraud Providers**

C. Further assessing the countries with the highest number of claims by plotting and analyzing the trends, we can observe the counties with the most claims. The counties with a high density of claims are likely to have made the most claims.



**FIG 3: Distribution of Fraud and Non-Fraud Providers**

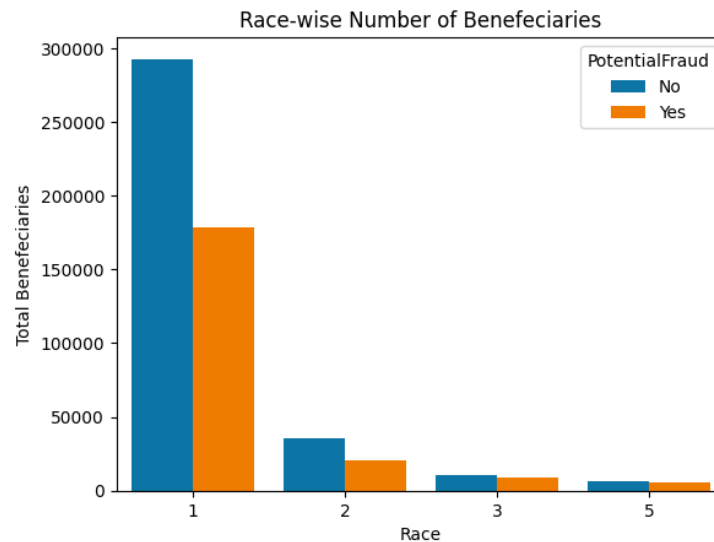
D. We examined the top 10 states with the highest number of beneficiaries to focus our analysis on areas with potentially higher risk for fraud. By identifying patterns and trends, we gained insights into types of services claimed, providers involved, and beneficiaries. This can inform targeted fraud prevention efforts in high-risk areas.



**FIG 4: Top 10 States with Most Beneficiaries**

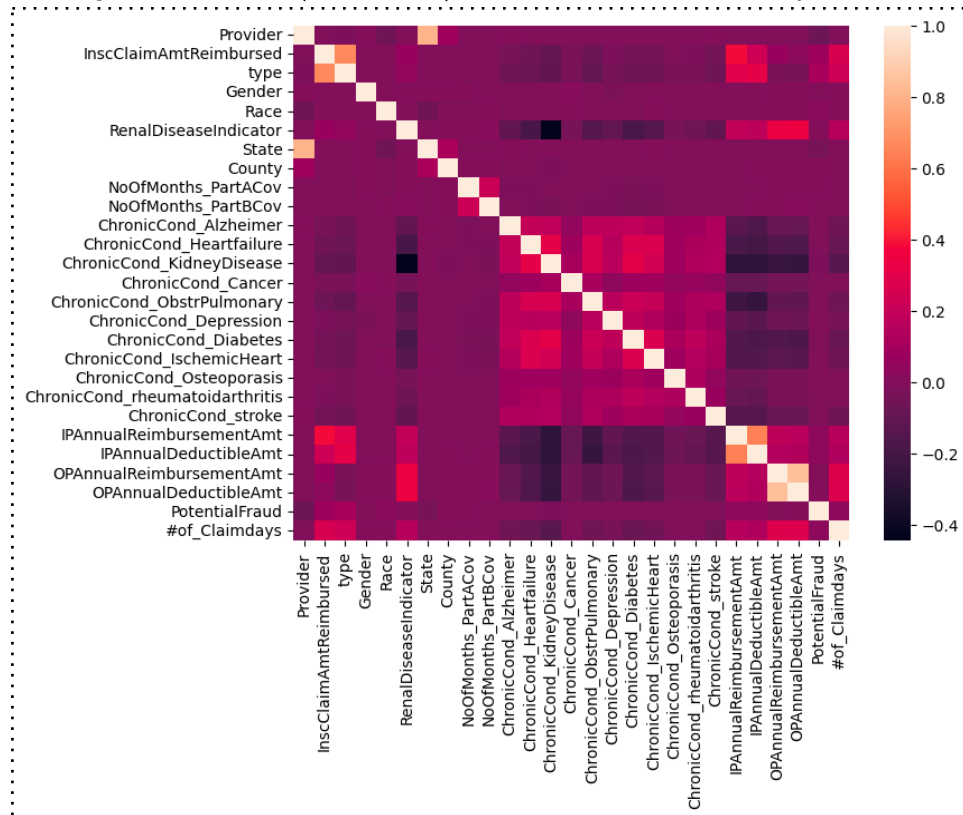


- E. Examining the claims made by race, we observe that the individuals classified under race category 1 have made the maximum number of claims. Further investigation into this race category can provide insights into the underlying factors contributing to the high number of claims made by this group.



**FIG 5: Beneficiaries by Race**

- F. By creating a correlation plot between variables, we observed that most columns do not have a high correlation with each other, while a few columns do. Based on this, we can remove the columns with high correlation (above 0.8) to avoid multicollinearity issues in our analysis.



**FIG 6 : Correlation Plot**

## 7. Data Partitioning

- The predictor variables are represented using the variable 'X', while the target variable 'PotentialFraud' is represented using the variable 'Y'. The dataset was split into training and testing sets using the 'train\_test\_split' method from the sklearn library.
- The **holdout** method was used to split the data, with **80%** of the dataset used for training the classification model and the remaining **20%** used to evaluate model performance.
- The resulting 'x\_train' dataset contains 446k records and 25 predictor variables, while the 'x\_test' dataset contains 111k records and 25 predictor variables. 'Y\_train' contains 446k records, and 'Y\_test' contains 111k records, representing the target variable.

## 8. Data Mining Models/Methods

**Model Evaluation:** The aim is to predict healthcare provider fraud using the dataset. Relevant features were selected after data wrangling, and the dataset was split into training and validation data with a 70/30 ratio. Various classification algorithms were tested using different metrics, and the algorithm with the best **performance** on **validation** data was selected for predicting potential fraud.

### 1. Logistic Regression

Logistic regression is a statistical algorithm used for binary classification. It estimates the probability of classifying an event as either true or false, based on a given dataset of independent variables. The algorithm uses a threshold cutoff on the probabilities for classifying into either of the classes. The outcome function called logit is used to model a linear function of predictors.

#### **Advantages:**

- It is easier to implement, interpret, and very efficient to train, even for large datasets.
- It makes no assumptions about the distributions of classes in feature space, making it flexible in handling different types of data.
- It can interpret model coefficients as indicators of feature importance, providing insights into the effect of the independent variables on the dependent variable.

**Disadvantages:**

- It constructs linear boundaries, limiting its ability to model complex relationships between the independent and dependent variables.
- It assumes linearity between the dependent variable and independent variables, which may not be valid in all cases.
- It requires little or no multicollinearity between independent variables, meaning that highly correlated features can negatively affect its performance.

**Implementation:**

The base logistic regression model was executed resulting in an accuracy score of 0.62, f1 score of 0.06, Precision score of 0.58 and Recall Score of 0.03.

```
Accuracy Score - 0.6217228128946732
F1 score - 0.06121904592539902
Precision Score - 0.5822410147991544
Recall score - 0.03230801717463223
```

## **2. Decision Tree Classification**

The decision tree algorithm can be used to classify new healthcare providers as fraudulent or non-fraudulent by following the path of the tree from the root node to a leaf node. Each internal node represents a decision based on the values of a feature, and each leaf node represents a predicted class label. To classify a new provider, the decision tree algorithm evaluates the feature values at each internal node and selects the appropriate branch until it reaches a leaf node with a predicted class label.

**Advantages:**

- Easy to understand and interpret
- Can handle both numerical and categorical data
- Can handle missing values by using splits to fill in missing values or by ignoring missing values altogether
- Able to identify important features for classification

**Disadvantages:**

- Prone to overfitting when the tree is too complex
- Unstable and less dependable when data even slightly changes
- Biased towards features with many categories or towards features with many levels
- May not always provide the most accurate results compared to other classification algorithms

**Implementation:**

After executing the decision tree model, an accuracy score of 0.76, f1 score of 0.65, Precision score of 0.74 and Recall Score of 0.58 was obtained.

```
Accuracy Score - 0.7629497595012674
F1 score - 0.6510922730089255
Precision Score - 0.7430935901294011
Recall score - 0.5793622861969452
```

### **3. Random Forest**

The random forest algorithm can be used to classify healthcare providers as fraudulent or non-fraudulent by learning patterns and features in the data that distinguish between the two classes. The algorithm creates a collection of decision trees, each using a subset of the features and data. When a new claim is presented to the random forest, the algorithm uses each decision tree to predict whether the healthcare provider is fraudulent or not. The final output of the random forest is determined by aggregating the predictions of all the decision trees.

**Advantages:**

- Handles high-dimensional feature spaces and is good at identifying complex, non-linear relationships between features.
- Can handle missing data and maintain accuracy with large amounts of noisy data.
- Computationally efficient, especially when dealing with large datasets, as each decision tree can be trained in parallel.
- Can provide feature importance estimates.

**Disadvantages:**

- More trees slow down the model and are prone to overfitting.
- Involves combining the outputs of multiple decision trees, making it difficult to interpret.
- Random forests can be sensitive to the choice of hyperparameters.

**Implementation:**

After executing the random forest model, an accuracy score of 0.89, f1 score of 0.85, Precision score of 0.89 and Recall Score of 0.81 was obtained.

```
Accuracy Score - 0.8917083919278415
F1 score - 0.8516091022903013
Precision Score - 0.892904022855378
Recall score - 0.8139649468571831
```

#### 4. XGBoost Classification

XGBoost (Extreme Gradient Boosting) is a highly efficient and flexible gradient boosting algorithm that is used for classification of healthcare providers as fraud or not by learning patterns. The algorithm is based on the concept of building many weak decision trees, known as "base learners," and then combining their predictions in a way that reduces the overall error. XGBoost trains these base learners in a sequential manner, where each base learner attempts to correct the mistakes made by the previous base learner, leading to a sequence of increasingly accurate predictions. To prevent overfitting and improve the generalization performance of the model, the algorithm uses a technique called "regularization."

**Advantages:**

- Has high predictive accuracy due to its ability to model complex nonlinear relationships in data.
- Handles large datasets with a large number of features.
- Regularization helps prevent overfitting and improves generalization performance.
- Has a built-in mechanism to handle missing values in the data.
- Supports parallel processing, making it faster than some other machine learning algorithms.

- Provides feature importance scores, which can help identify the most important features in the dataset.

**Disadvantages:**

- Requires careful tuning of hyperparameters to achieve optimal performance.
- Can be prone to overfitting if hyperparameters are not properly tuned.
- Require more computational resources than some other machine learning algorithms.

**Implementation:**

- After executing the XGBoost model, an accuracy score of 0.79, f1 score of 0.67, Precision score of 0.82 and Recall Score of 0.57 was obtained.

```
Accuracy Score - 0.7878774307390521
F1 score - 0.672030799911367
Precision Score - 0.8200561057221076
Recall score - 0.5692733628962249
```

## **9. Project Results**

The following performance metrics were calculated to evaluate the classification performance of the model on the holdout dataset:

- **Accuracy:** the proportion of correct predictions to the total number of predictions made by the model.
- **Precision:** the proportion of true positive predictions to the total number of positive predictions made by the model.
- **Recall/Sensitivity:** the proportion of true positive predictions to the total number of actual positive cases in the dataset.
- **F1-score:** the harmonic mean of precision and recall, which provides a balance between these two metrics.

	Model	Accuracy	F1 score	Precision	Recall
0	Logistic Regression	0.62	0.06	0.58	0.03
1	Decision Tree	0.76	0.65	0.74	0.58
2	Random Forest	0.89	0.85	0.89	0.81
3	XG Boost	0.79	0.67	0.82	0.57

Overall, the random forest model has the highest accuracy and F1 score, indicating better overall performance. It also has the highest precision and recall values, which means it has the best balance between correctly identifying fraudulent healthcare providers and minimizing false positives. The decision tree model also performs reasonably well, but is outperformed by the random forest model. The logistic regression model has the lowest accuracy, F1 score, precision and recall values, indicating poor performance for this task.

## **10. Future Scope**

### **>> Model Improvement**

Based on the performance of the model, there is room for improvement by adding more fraud data to the training dataset to help predict unseen fraudulent behavior, implementing ensembling methods with parameter tuning, and vectorizing medical codes (ICD 9 codes) with Count Vectoriser.

### **>> Business Recommendation and Improvement**

The above model can help insurance companies scrutinize claims more thoroughly, and further improvements can aid the government in making decisions against fraudulent health providers and amending rules and regulations in this domain.

Additionally, improving the model can aid in detecting networks of fraudulent physicians, providers, and beneficiaries. Ultimately, reducing fraud can lead to a healthier economy by lowering inflation caused by fraudulent claims and reducing insurance premiums, making healthcare more affordable for all.