MODEL FOR ASSESSING FRAUDULENT MEDICAL CLAIMS: AN APPLICATION OF MACHINE LEARNING ALGORITHMS IN HEALTH CARE.

By

FRANCIS ADRIANO OUNDO

17/01050

Masters of Science in Data Analytics

KCA UNIVERSITY

2019

FACULTY OF COMPUTING & INFORMATION MANAGEMENT

RESEARCH PROJECT

ON

MODEL FOR ASSESSING FRAUDELENT MEDICAL CLAIMS: AN APPLICATION OF MACHINE LEARNING ALGORITHMS IN HEALTH CARE.

BY

FRANCIS ADRIANO OUNDO

A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE IN DATA ANALYTICS IN THE FACULTY OF COMPUTING AND INFORMATION MANAGEMENT AT KCA UNIVERSITY

SUPERVISED

BY

DR. MWENDIA

OCTOBER 2019

DECLARATION

I, Francis Adriano Oundo, the undersigned, hereby declare that this research proposal is my original work and that it has not been presented to any other University, College, or Institution of Higher Learning other than KCA University.

…… …….

Francis Adriano Oundo Date

Student No: MDA-17/01050

This project has been presented for examination with my approval as the appointed supervisor.

Signed …… Date ……..

Signed …… Date ……...

Dean,

Faculty of Computing & Information Management KCA University

ABSTRACT

The insurance industry is experiencing challenges as claim verification has remained to be a manual process due to its nature that requires human observation. Health insurance cost has endlessly been increasing with time. In Kenya, the majority of citizens cannot afford healthcare services as most are not insured. The insurance cost is so high with low per capita income. The high cost is due to administrative cost that results from inefficient operational processes that are prone to errors and fraudulent claims. We propose a machine learning model that can be extended to a system which will create claim assessment automation that will greatly reduce the administrative resources required to process medical claims. The model will learn from previous data patterns and predict the correctness or review of the claim thus reducing errors that are prone to manual processes. Using machine learning techniques for classification like Logistic Regression and SVM we are able to classify a correct claim or a claim that will be called for review by the auditors. We will show the significance of our model and how it achieves better precision and accuracy than the manual process.

Acknowledgments

Throughout the writing of this dissertation I have received a lot of assistance.

I would like to thank my supervisor Dr. Mwendia, who has given me unwavering support in all the steps of this research.

I would also want to thank all my lecturers who taught me throughout my master's study and my classmates who supported me and shared this journey with me.

To my family (*Ann, Gael and Rubia)* without whom I would have finished much earlier! and friends, thank you for supporting me in my endeavor to make a world a better place.

Dedication

It’s my gratefulness and warmest regard that I dedicate this dissertation;

To my wife Ann, and son Ambanda without whom this work would have been completed somehow earlier!!

Acronyms and Abbreviations

AKI -Association of Kenyan Insurers

EACC -Ethics and Anti-corruption Commission

L-R -Logistic Regression

MOH – Ministry of Health

SDG -Sustainable Development Goals

SVM -Support Vector Machines

TAT -Turn Around time

KYC -Know Your Customer

UHC -Universal Health Coverage

WHO -World Health Organization

Operational Definition of Terms

Table of Contents

List of Figures

List of Tables

CHAPTER ONE:

INTRODUCTION

1.1 Background

Healthcare insurance is an organized financial plan that assists households and individual persons to put aside resources that can offset the cost of healthcare in case of an illness. It bases its principle on aggregating the funds together and entrusting of such funds to a third party who pays the members that contribute to the pool. These third parties are organized as government entity, employer institution, insurance scheme or an individual provider (Kraushaar, 1994).

When the contributing members pool these resources together in event of the risk happening, the insurance scheme has potential of recovering the cost as they use the funds in the pool to offset the individual cases (Shaw, 1998). When many people come together and harness the risk in large healthcare expenditure, healthcare insurance can ensure that healthcare cost is affordable to all. In 2010 the World Health Organization listed fraud as one of top ten leading inefficiencies in healthcare. (WHO, 2010). And recent studies estimate that about $487 billion is lost to fraud (Im Gee, 2014).

Fraud, wastage, and abuse are widespread in Kenya's healthcare. According to the Insurance Regulatory Authority, in 2012, 143 cases of medical fraud were recorded and Sh.253.6 million was lost. And out of this amount only Sh.5.2 million was recovered. The class of business was at 81% in 2010 having an average loss ratio of 78% in five years up to 2010. (AKI, Insurance Industry Annual Report., 2013).

Medical Gross Written Premium hit Kes.38.42 billion in 2017, about Kes.36 million less in previous year. Medical insurance had an increase of 14.63% claims in 2017 compared to previous year at Kes.20.56 billion. Making the technical loss ratio of 73.38% to be the highest in Non-Life insurance business. This means out of every one shilling premium paid 73.38 cents were claimed. (AKI, Insurance Industry Annual Report., 2017)

Total re-insurance premiums paid in 2017 was kes.37.52 billion as compared to kes.31.41 billion in previous year. Medical reinsurance premiums was Kes.13 billion being the highest of the non-life insurance business at 34.75%. The loss ratio of 73.38% and an expense ratio of 28.45% raised the combined ratio of medical insurance to 101.39%. Making medical insurance in 2017 to register a net loss of Kes.513, 887,486. (AKI, Insurance Industry Annual Report.,

2017). There is a high administrative cost for medical claim processing and these cost results from extra administrative cost to re-process the incorrectly captured claims, overpayments of claims which also add the extra cost of resources to initiate the refund process, underpayments that result in fines and the extra cost of credit.

Most common types of insurance fraud in Kenya as postulated in (AKI, Insurance Industry Annual Report., 2017) are diagnosis manipulation, provision of generic drugs instead of branded drugs, over servicing, membership substitution, fee splitting, falsifying claims or altered invoices and non-disclosure of prior ailments. Most of these fraud cases are perpetuated by healthcare providers.

Patients using insurance cards are charged up to 50 times more than those paying in cash for the same procedure at the same hospital. The highly exaggerated cost of healthcare by health service providers when they bill patients with insurance cards is a cause for worry by the Ministry of Health. A report released recently by the Ethics and Anti-corruption Commission (EACC) revealed that there is wide variation in what different hospitals charge for the same medical procedures (BD, 2018). Medical providers unjustifiably claim that the variations in pricing are as a result of terms of payments. Therefore they add the cost of credit to their pricing when dealing with insured patients as the insurers take a longer time to pay the providers.

As observed by (Orayo, 2017) medical claims are done manually in Kenya. In 2016 around 10 million medical claims were presented to private insurance companies making a rough estimate of 40,000 claims per day. Each of these claim form is accompanied by an invoice(s) which is more likely followed by test request form from the lab or a prescription, these averages four pieces of paper per claim. Considering the best 10 first tier medical insurance companies, they receive almost 150,000 pieces of paper claims in a day. Which have to be paid as agreed with the customers according to policy rules. The information presented in those papers has to be manually recaptured into the insurer's systems before any payment can be appropriately made. These could take up to 90 days depending on the availability of resources and the capabilities of insurer's systems. Some of these claims could be processed and paid. Some of the claims may be incorrectly captured or even misplaced thus making the payments to be delayed further.

In this study we intend to use supervised learning techniques. In this classification problem we will employ binary logistic regression as our main algorithm of choice for model construction. Logistic Regression is among the most popular methods used to fit models for

categorical data, it fits binary response data in Data Modeling. It is the most important (and probably most used) member of a class of models called generalized linear models (Brid, 2018)

Logistic regression does not encompass linear relationship between dependent and independent variables. It usually handles different types of relationships as it applies a non-linear log transformation to the predicted odds ratio.

1.2: **Problem statement**

Insurance schemes are obliged to verify authenticity of claims. This task ties down several employees, especially for top tier insurance companies. This process is extremely cumbersome, most claims are potentially incorrect and have to rely on the administrative staff to check in detail for validation based on insurers specific rule book. These rules are based on claim information and any available data from patient's previous engagements, the staff draws based on their experience and expertise to decide on whether to accept, deny or probe for further audit. (Mohit Kumar, 2014).

Claims that are denied during the evaluation and payment process are costly for providers and brings a lot of pain for patients who have to pay out-of-pocket or providers are forced to write off as losses. The manual nature of existing claims management processes, which usually comprises of analysts and rules making choices about which claims to work for resubmission with their limited time. Most related studies have focused on feature acquisition where the goal is to select features with most informative gain to obtain during training (M. Saar-Tsechansky, 2009) and budgeted learning where the objective is to learn the most accurate or active classifier and rank based on a fixed learning budget for acquiring training data which explains the idea of exploration from reinforcement learning (Sculley., 2010).

None of this research we are aware of focusses on using a classifier and obtaining classification result to optimize the efficiency of a domain expert in accepting or denying healthcare claim, like our study. Our study will try to bridge the gaps that exist in finding a solution for class imbalance in fraud detection.

Classifiers generally perform poorly on imbalanced datasets because they are designed to generalize from sample data and output the simplest hypothesis that best fits the data, based on the principle of Occam's razor. This principle is embedded in the inductive bias of many

machine learning algorithms including decision trees, which favor shorter trees over longer ones. With imbalanced data, the simplest hypothesis is often the one that classifies almost all instances as negative. (Akbani, 2004)

Class imbalance is not only the major concern in fraud detection system. Overlapping of the genuine and fraudulent classes due to limited information about the transaction records is another problem in the classification task (Shakya, 2018) and most machine learning algorithms underperform under these scenarios.

1.4: Proposed Solution

Machine learning model will be used to automate claims assessment and routing based on existing fraud patterns. This process will flag claims that are fraudulent and incorrectly captured claims for review, it will also automatically identify good transactions and streamline their approval and payment. This allows the providers and payers to spend their valuable time on likely valid claims that yield most value to them. This will lower the expense ratio for medical insurance and lower medical reinsurance premiums ceded due to reduced risk margin thus lowering total cost of medical insurance. The TAT for medical claim clearing will be greatly reduced hence avoiding late payments from insurance companies.

1.5: Research Objectives

To establish an appropriate model for assessing medical insurance claims.

1. ***Objective 1****: To investigate the identified independent variables that can be used to generate a model for assessing medical insurance claims.*

Research questions:

a) What are independent variables in a patient's information?
b) What are independent variables for the providers' information?
c) What are the independent variables in claim details for authorizations and medical necessity?

2. ***Objective 2****: Investigate an appropriate machine learning algorithm that can use the identified factors to assessing insurance claims*

Research questions:

a) What is the appropriate classification algorithm that will be used for assessing medical claims?

b) Why is the selected algorithm more suited?

3. ***Objective 3:*** *To train the algorithm to generate an appropriate model for assessing insurance claims.*

Research questions:

a) How efficient is the algorithm with selected features in reducing training time?

4. ***Objective 4:*** *To Evaluate the performance of the established model*

Research Questions:

a) How accurate is the established model?

b) What are the performance metrics used for the established model?

1.7: Motivation of the Study

The cumbersome nature of manual processing of medical claims raises the cost of healthcare and reduces the quality of medical care provided. Machine learning methods could provide a way to find patterns automatically and reasons about data, this in return allows healthcare professionals to focus on personalized care called precision medicine. Previously more attention has been paid on using machine learning on patient's records to predict or classify their ailments. Little in terms of research has been done for medical claims. On this backdrop, we decided to propose a faster approach of clearing medical claims using machine learning model. This will expedite the process of clearing medical claims as the model learns from previous data patterns and can predict based on the nature of the medical claim whether the claim will be correct or reviewed thus reducing time and resource wastage that translates to affordable medical care.

1.8: Research Scope

Our study will span data from one insurance firm in India and cover out-patients section only

1.9: Justification/Significance of the Study

1.9.1 Healthcare provider

- By using claims processing model we could reduce processing costs, speeds up claims resolution cycles which increases revenue.
- By eliminating manual touch points for higher data accuracy, it will increase productivity and efficiency, enhanced processing speed, enhanced information accessibility, and reduced processing cost
- Improve customer satisfaction in enhancing communication and reducing data entry errors.
- The operational costs can be reduced by lowering staffing requirements cost while handling more claims.
- Mitigate risk and promote compliance capacity when it eliminates manual prone error work, reduces human exposure to sensitive customer data.

1.9.2 Government/Regulator

- Allow government/ regulator to leverage integrated data for proper planning in healthcare and insurance departments.
- Achieve its mission of universal and affordable healthcare, by increasing medical insurance penetration due to reduced healthcare premium cost.
- Improves data access, integration, and compatibility by allowing flow of information from variety of devices to both external and internal processes, this enables regulators to smoothly monitor these process.

1.9.3 Patient

- Affordable healthcare services due to reduced cost of insurance.
- Improved and efficient experience from the providers.

CHAPTER TWO

LITERATURE REVIEW

2.1: Introduction

Universal health coverage (UHC) is SDG number 3, being a priority policy agenda worldwide. There is universal need to access improved quality healthcare services and to protect the population from catastrophic and impoverishing health care costs. In Kenya, UHC is among the top four agenda that the President launched as the blueprint for his administration. Therefore there is urgent need to access the healthcare services in equitable and sustainable fashion and not based on ability to pay. (WHO, 2010)

There upsurge in health insurance costs across the world has been consistent in recent years. These increased costs have been forced down to consumers thereby making healthcare services unaffordable. According to (NCHC, 2010), over the last decade there has been an upward increase of health insurance premiums of 131%. Most of these increases has been due to healthcare providers increasing administrative costs of insurance.

In Kenya, there is need to emphasize the usefulness of financial protection to UHC. About 83% of Kenyans lack financial protection from health care costs, which forces about 1.5 million people into poverty each year (MOH, 2010). The higher medical premiums are as a result of a direct effect on the administrative cost incurred during the medical insurance processing procedure, and fraud. Therefore for the realization of UHC, the medical insurance industry players must devise novice means of doing business with minimal administrative cost. In this chapter, we are going to cover the key factors and attributes for assessing medical insurance claims, the machine learning algorithms suitable for the attributes, the application for the selected algorithms, and the existing methods for assessing the developed models, empirical literature, conceptual framework and operationalization of the variables in conceptual framework.

2.1: independent variables that are considered when classifying health insurance claims

There are myriad factors that are considered for classifying health insurance claims if they are either correct for payment or they will be denied. First, the provider must show proof that the patient/beneficiary attended the provider facility for the service rendered. This is ensured by sending a verification code to the insurer at the time the beneficiary's medical card is swiped on the providers' system. Most medical claims are denied as a result of various reasons such as; Missing information or incorrect filled patient demographic data and technical errors, claim submission that are duplicated, services that are not covered by the payer(restrictions), lapsed time limit for claim submission and lack of prior authorization or referral especially for drugs dispensed. (Ben Colton, 2015). These reasons are clustered into four major factors;

2.1.1: Demographic/Patient information

By not accurately collecting patient information and verifying insurance coverage may lead a claim denial. For instance if hysterectomy was mistakenly claimed for a male, it will be denied. Patient information is critical in identifying the patient and ensuring that the patient at hand is the verified patient who deserves the care/service given. There are various attributes that can help identify the patient with certainty. Administrative staff should ensure the verification of all demographic information and the information reported on forms should match patient identity and insurance cards. The data should be accurately entered into the billing system. (Ben Colton, 2015)

2.1.2: Policy information

According to (Ben Colton, 2015) most medical claims are denied due to ineligibility of the patient. Policy information is very crucial for understanding the eligibility of the insured person. This information outlines the cover limits for a patient and highlights their non-covered services. If a particular medical service is not included in the policy plan, then that particular service cannot be provided. Benefit information should be checked and verified before the services are rendered. The patient's last physical exam date should be checked and other annual services to be sure they are covered at the time of the current visit. Also, ensure the patient has not exceeded the maximum number of sessions allowed in a given calendar. (Ben Colton, 2015). The insurances rules are common and specific terms specifies terms of insurance contract of the beneficiary. (Ilker Kose, 2015).

2.1.3: Provider information

According to (Qi Liu, 2013), the claim data have two characteristics. First, they contain attributes to describe service providers' behavior. Second, each claim uniquely identifies service provider by unique identifier. When we use the unique identifiers to link different claims, we can obtain a round view of a service provider's behaviors over time and across different insurance subscribers. The rounded views help significantly in identifying the payments pattern for the claim and fraud committed by service providers.

2.1.4: Claim information

These are sometimes called the medical rules, which checks whether the provided health service is appropriate for the specified age, sex or other attributes of the insured individual. The medical cases are difficult to model use cases for many possible outcomes, as the claim data set includes limited data due to practices and uncooperative nature of medical providers to include many attributes. (Ilker Kose, 2015). The claim information contains the clinical report. This report is summary of the care or service(s) that is given to a patient. A clinical instance is a process instance made of set of activities that are logical units of work performed by medical practitioners. Like, treatment may involve several activities. These activities, may be performed in sequence, concurrently, or repeatedly. "For example, before a health practitioner gives any therapeutic intervention, diagnosis activities are usually performed to verify the condition of a patient. More than one of these interventions may be performed concurrently in order to increase the curative effect in some cases. By carefully mining these activities, fraudulent behaviors can be distinguished from normal activities and payment pattern can be discerned depending on the previous patient's data." (Qi Liu, 2013)

2.2: Attributes for assessing medical insurance claims.

The data used in assessing medical insurance claims can be obtained from insurance warehouse with several tables. The required attributes can be selected and merged from various table to identify the key features that are needed for the model. According to (Mohit Kumar, 2014) the evaluation criterion was first, the patient (beneficiary) information; this study outlined attributes such as; a unique patient identifier, age, sex, marital status, employer number etc. Secondly, they looked at the policy (eligibility/beneficiary verification) information; the attributes considered were medical insurance limit, number of the dependent beneficiaries, card

restrictions, number of times the card has been used in a given year, co-insurance. Thirdly they evaluated the provider (Healthcare) information; which included a unique number of provider institution, the attending physician, county code etc. Then finally they evaluated the Claim information; with attributes like admission date, date claim was made, admission type, diagnosis code (ICD-10_CPT), treatment code (ICD-10-CPT), amount billed, co-payment.

The attributes were preprocessed in Ski-Kit Learn module of Python using PCA to evaluate the attributes with the high information gain to be included in the model.

2.3: Overview of Machine Learning Algorithms

a) Supervised Learning algorithms

Supervised learning methods are employed in predictive analytics. In this study, we intend to predict the outcome of the healthcare claim if either it will be paid or denied. We examine classification models like Logistic Regression and Support Vector Machines (SVM). Logistic Regression applies logistic function to classify tasks. It applies a linear equation with independent predictors to predict a value.

$$h = g(z) = \frac{1}{1 + e^{-z}}$$

Equation 1

It takes the output (z) of the linear equation and gives to the function g(x) which returns a squashed value h, the value h will lie in the range of 0 to 1.

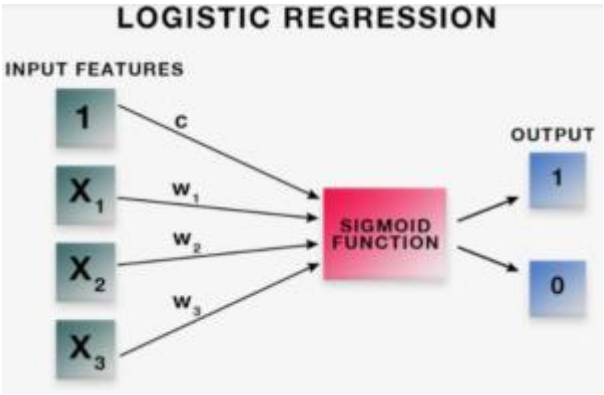


Figure 2.1 1 Logistic Regression

$$y = \text{logistic}\ (c + x_1{*}w_1 + x_2{*}w_2 + x_3{*}w_3 + \ldots\ldots\ldots + x_n{*}w_n)$$

$$y = 1 / 1 + e\ [-\ (c + x_1{*}w_1 + x_2{*}w_2 + x_3{*}w_3 + \ldots\ldots\ldots + x_n{*}w_n)]$$

Equation 2

Logistic function is given by **y = 1 / 1+ e-x** Equation 3

The SVM, on the other hand, use the feature weights learned by the SVM to generate the explanations. “Linear SVMs use a prediction function of the form:

Prediction(x) = $sign(w^T x + b) = sign(\sum_i w_i * x_i + b)$ with features i. Equation 4

Features with large (absolute) values have a large effect on the classification while features with weights close to zero have little effect.” (K. Shih, 2002) Has shown that the weighty features are influential in determining the width of the margin when learning with SVMs.

b) Unsupervised Learning Algorithms

In unsupervised learning, Dimensionality Reduction is applied to select the most informative features during training. Principle Component Analysis (PCA) determines the feature with high information gain so as to avoid overfitting on the outcome of the algorithm.

c) Ensemble Learning Algorithms

“Ensemble learning is employed as a descriptive analytics. Random Forest, for example, generates a large number of trees. The class value appearing most frequently among these trees is the class predicted as output from the model. As an ensemble learning method, Random Forest is an aggregation of various tree predictors. Each tree within the forest is dependent upon the values dictated by a random vector that is independently sampled and where each tree is equally distributed among the forest.” (Apache, 2018). The Random Forests ensemble inserts randomness into the training process which can minimize overfitting and is fairly robust to imbalanced data. (Richard A. Bauder, 2018).

2.4: Application of Machine Learning algorithms in developing a model for assessing Insurance claims

In one study (Richard A. Bauder, 2018) incorporated a two-step approach in detecting Medicare fraud, per provider type. Their method used a multivariate regression model from which the residuals are passed into a Bayesian probability model. In another study by (R. A. Bauder and T. M. Khoshgoftaar, 2017) they used multivariate regression to establish a baseline for expected Medicare payments, per provider type, to compare against actual payments to flag possible fraud.

Our research problem can be formulated based on review identification as a binary classification task and predict whether a claim will end up requiring review in the future or not. Confidence score of the binary classifier can be used to prioritize and order the claims for review by the auditors. Using classification algorithms SVM and logistic regression to gauge the accuracy of each technique using the same variables. "Logistic Regression (LR) (Apache, 2018) predicts probabilities for which class a categorical dependent variable belongs to by using a set of independent variables employing a logistic function. LR uses a sigmoidal (logistic) function to generate values that can be interpreted as class probabilities.

SVMs have been shown to be robust for large data mining tasks with large feature sets." (Mohit Kumar, 2014). Since SVMs are not able to handle categorical data, binary features are formed from categorical features. All features are based on the information that is available at the point in time when the claim was adjudicated and ready for payment, from these features, the algorithm predicts before payments on whether to pay or review the claim further. The study is unique in that it focuses more on predicting if the claim will be correct. From the real data, the usual class distribution in data is approximately 2-5% review and 95-98% correct claims which is the overall distribution of the entire population of claims. (Mohit Kumar, 2014). This brings class imbalance that will be addressed by using the class distribution of features and class labels differently. (Branting, Reeder, Gold, & and Champney, 2016). For the accuracy of this model, it prioritizes the claims based on the classification confidence and aims to optimize the predictions for the top-ranked claims. This is justified in real life as the daily volume of claims coming into the pipeline for adjudication is huge.

2.5: Existing methods for assessing the developed models.

The classification models are evaluated using the Area Under receiver operating characteristic Curve (AUC) performance metric (Bekkar, Djemaa, & and Alitouche, 2013). AUC is used to assess the capabilities of binary classification methods. "The ROC curve is used to characterize the tradeoff between true positive rate and false positive rate, depicting a learner's performance across all decision thresholds. Perfect classification is indicated by an AUC value of 1, with a range from 0 to 1. Due to the severe class imbalance of the testing data, AUC is used as the performance measure for our experiment."(Jeni, Cohn, & and De La Torre, 2013).

Cross-validation especially stratified k-fold cross-validation has been robust in evaluating models. “In a case where k = 5, stratification ensures all folds have class representation matching the ratio of the original data, which is important when dealing with largely imbalanced data. The training data is evenly divided into fivefold where fourfold will be used for training the model and the remaining fold tests the model. This process is repeated 5 times allowing each fold an opportunity as the test fold, ensuring the entire dataset is fully leveraged being used in training and validation. The percentage split labeled data in 70%-30% training and test sets in randomness.” (Richard A. Bauder, 2018).

2.6: Empirical Literature Review

Expenses are the costs associated with acquiring, underwriting and servicing the business. In 2017, expenses were recorded at KES 36.14 billion compared to KES 34.66 billion in 2016. The total expenses expressed as a percentage of GWP was 28.7percent. Medical insurance registered the third highest expense ratio with close to Kes.7 billion. (AKI, Insurance Industry Annual Report., 2017). Most of these costs are as a result of the administrative cost of review. There exists two industry practices prevalent currently for identifying payment errors: random quality control audits and hypothesis (rule) based queries. Using random audits is not very effective at identifying review since the majority of claims are correct and most of the effort spent on audits is wasted. In previous studies, it had been found that between 2% and 5% of the claims audited are rework, making 95% to 98% of the effort spent in random audits a waste. (Mohit Kumar, 2014). In their research Data Mining to Predict and Prevent Errors in Health Insurance Claims Processing. (Mohit Kumar, 2014)

The study showed that system produced a higher precision (hit rate) over existing approaches which was accurate and potentially resulted in savings over $ 15 million every year for a typical insurer. This, reduced healthcare costs as well as helped make the healthcare process smoother. In their study (Mohit Kumar, 2014) formulated multi-class classifier ones Vs all strategy. This has a limitation of training only one model if there are only two classes. In this study using one Vs one multi-class strategy ensures that classifiers are constructed and each one train data from two classes.

2.7: Critique of Relevant Literature.

There are very few easily accessible literature materials on medical claim processing using machine learning. Outside the research community, there are commercial product vendors whose model are not well explained. These vendors are using machine learning techniques incorporated in their systems for medical claim clearing. Like H20.ai, Kofax and ClaimVantage. As (Dr. Daniel Schlegel, 2019) observes, one should estimate the benefits of automation on a process-by-process basis. If resources can be relieved of tasks through automation, where will their freed-up time be spent? How does improved accuracy or increased speed translate into value? By answering these questions is when we will use technology in a more ethical manner and bring value to society.

2.8: Conceptual Framework

Increasing health care costs has become a critically important public policy challenge around the world (WHO, World Report on Health Policy and Systems Research, 2017). Insurance companies in medical claim processing waste valuable time in manually evaluating many claims adjudicated for payments. The interactions of the independent variables as outlined in the table helps identify the intended target dependent variable. The moderating variables are the factors that are beyond the control of the model since the biometric identification is limited to a beneficiary with 5 years and above, then their unique identification is beyond this model. Another moderating variable will be a new and unique diagnosis (new diseases that are not captured in ICD-10).

Figure 2.1 2 Conceptual Framework

2.9: Operationalization of Variables in the Conceptual Framework.

Table 2.1 1 Operationalization of variables

Independent variables	**Attributes (indicators)**	**Data (values)**
Demographic/Patient information	Age	Numerical value
	Gender	Male or female
	Marital status	Married or single
	Employer Identifier	Value
	Patient unique identifier	Value
Policy information	Cover limit	Numerical value >0
	Card restrictions	String value
	Donor beneficiary	Numerical value>0
	Usage per year	Numerical value
Provider information	Provider unique identifier	Value
	Attending physician identifier	Value
	A physical location of the provider	Address value
	Provider status	Public or private
Claims details	Admission date	Date time
	Date claim was made	Date time
	Admission type	Value
	Diagnosis type code (ICD-10)	Value
	Treatment code (CPT)	Value
	Amount Billed	Numerical value>0
	Co-payment plan	Specified value

After collection and labeling of data, feature construction takes place. The four classes of information in each claim are: Member information, Provider information, Policy Information, and Claim Details. The features are extracted out of these classes of information. Member and Provider information narrates the information about the Member (patient), and the provider (hospital, doctor, or medical facility). The Claim Details outlines the information about the entire claim as well. Contract information, Diagnosis codes, Amount billed, Dates of service are some examples of data fields in the claim header. The other source is the Provider Details. This gives the details of the facility and physician attended. In each procedure, the claim detail contains the amount billed, the procedure code (CPT), the count for the procedure done (quantity). As the class focuses on predicting the whether the entire claim will be correct or review, the claim details are aggregated to the overall claim level. In example, features are created using Min, Max, Average and standard deviation functions for each numeric data field in each line. (Mohit Kumar, 2014). The class labels are constructed from the data features.

2.10: Summary of Literature Reviewed.

From the literature reviewed in this study, it is evident that the problem exists in the medical industry that is raising the administrative cost for claim processing caused by inefficiencies in operational procedures. This manual process, with big amounts of data to sieve through, becomes tedious and inefficient compared to more automated data mining and machine learning approaches for detecting fraud and claim processing. (Clara, 2013). The information within healthcare continues to increase due to technological advances allowing for the storage of high-volume information, such as in Electronic Health Records (EHR), when we use "Big Data." as technology advances and its use increases, so does the ability to perform data mining and machine learning on Big Data, which can improve the state of healthcare and medical insurance programs for patients to receive quality medical care.

2.11: Research Gaps.

The existing studies have focused on applying different algorithms on medical fraud detections. (R. A. Bauder and T. M. Khoshgoftaar, 2017) Big Data fraud detection using multiple Medicare data sources observed that logistic regression was more accurate for combined data than random forests. Their research examined the performance of different algorithms when used on

combined data from different sources for fraud detection. (Mohit Kumar, 2014) Studied the use of SVM on medical claim processing and predict rework. Their research allowed health insurance companies to save tens of millions of dollars each year. There have been gaps in Kenyan medical insurance industry in adopting automation in medical processing and this presents the opportunity to study the application of machine learning to medical claim processing in local contest using the local available data.

CHAPTER THREE:

METHODOLOGY

3.1 Introduction

In this chapter the study covered sequentially the methods we used to achieve our study objectives. This chapter was integral in our study as it outlined the process through which we achieved our study objectives.

3.2 Methods for achieving Objective 1 and 2

3.2.1: Target population

The data used in this research was used for research in India and was released on Kaggle an online portal for community of data scientist. The dataset captured mostly elderly persons with chronic diseases and their claim behaviors for a period of 3 years. The study used this data in research as its prudent to for any civilized society to take care of the aging population with utmost honesty and integrity.

3.2.2: Sampling Design

The data that was used in this study was obtained from secondary data sources. The study sampled the data as a whole representative of our target population which spanned the entire outpatients' data for the last 3 years from a local insurance company. This data had several hundred thousand records and over one hundred features or columns. For the purposes of this research we focused on;

- Outpatient data

This provided the details of the patients who visited the hospitals and were not admitted in them. The datasets in outpatient file contained train and test datasets, each with elaborate features describing the details of the claim. The dataset had 517,737 records and 27 feature columns.

- Beneficiary data detail

Which contained the details of the beneficiary of each claim, their identifying KYC details for each beneficiary. This also contained both train and test datasets. This data set had 138,556 records and 25 feature columns.

- Train data

This file contained the provider unique identifier and the class label. The class label is column Potential Fraud, which is Yes or No coded as either 1 or 0 respectively. If it is fraudulent as Yes

then the claim is reviewed and if it's No then the claim is correct to be paid. The study was examining whether the claim will be correctly identified for payment or will be reviewed, hence we aggregated data at unique Claim ID for each claim made by the Provider. We had 5410 unique providers each handling different claims.

3.2.3: Data collection

The data was collected from online desk research, where data mining techniques were used to acquire the required data for our research.

This was data introduction stage, the data was captured from the claims processing pipeline at the time when it is priced and ready for finalization and payment. The data that was operated on was the entire claims data warehouse which contained all the claims that had been submitted and processed in the past 3 years. The data used was aggregated from various sources from insurance pipeline and combined to form master data. We assigned labels to train our model, *correct* claim or *review*. This spanned a given timeframe for the entire outpatient's data in warehouse for healthcare claims data.

3.2.4 Data Analysis Methods

Descriptive analysis techniques

Using Sci-kit learn, pandas library and data visualization library called seaborn and matplotlib were imported. Combining seaborn and pandas executes most interactive data visualization. The study used countplot function to visualize the descriptive statistics of the data. Then analyzed relationships between different features and how they contribute to predicting target variable. The skewness and correlations of the independent variables is a key factor to the outcome variable.

Qualitative content analysis

The study used previous research in literature review to identify machine learning algorithms that are suitable for our data. From the literature review it can be widely accepted that for a binary classification problems, supervised machine learning algorithms are more suitable. Our research problems required a solution that could classify healthcare claims for payment or review. The claims data usually is imbalanced and with positive class as a minority class.

3.3: Methods for achieving object 3 and 4

The study used Knowledge Discovery in Databases (KDD) Process by Fayyad et al. (1996) to outline our data mining cycle process.

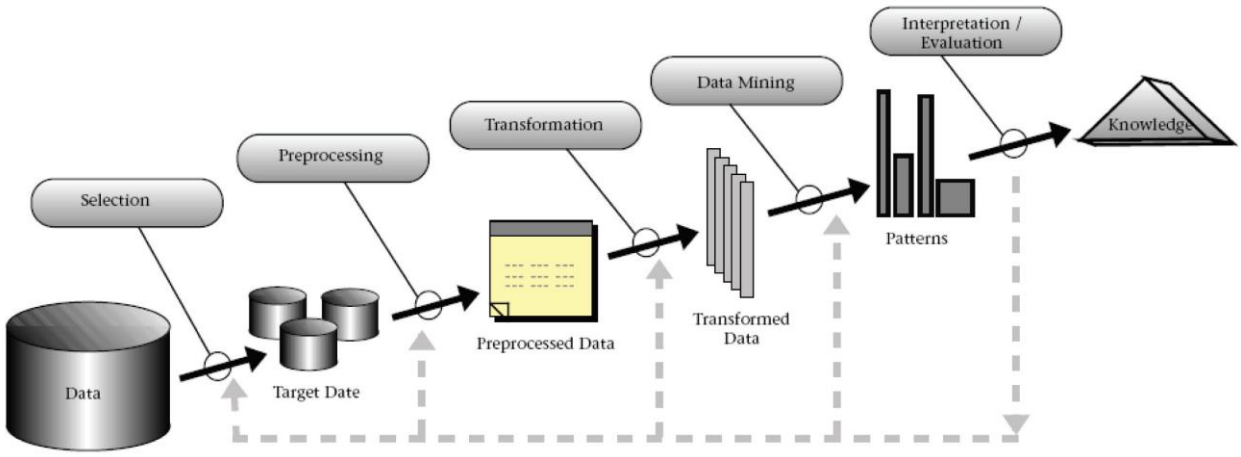


Figure 3.1 1 Model process cycle

3.3.1: Data selection

The research developed an understanding of the domain knowledge by intensively researching on the insurance data. We joined and merged provider data file on our main outpatient data. This was useful as the provider data consisted our target variable.

```
## Merge Train_PatientDetaildata with fraudelent providers using "Provider" as joining key
Train_PatientDetail_labeldata=pd.merge(Train, Train_PatientDetaildata, on='Provider')
Test_PatientDetail_labeldata=pd.merge(Test, Test_PatientDetaildata, on='Provider')
```

And then did descriptive statistics to understand our data at hand using pandas in python.

```
####DESCRIPTIVE STATISTICS
describe=Train_PatientDetail_labeldata.describe()
class_counts=Train_PatientDetail_labeldata.groupby('PotentialFraud').size()
correlations=Train_PatientDetail_labeldata.corr(method='pearson')
skew=Train_PatientDetail_labeldata.skew()
```

3.3.2: Data Preprocessing

I. Basic data cleaning

The data from the data warehouse or database was carefully cleaned by following techniques.

- ✓ Dealing with data types

Claims data may can be in form of numerical, categorical or ordinal. Our machine learning algorithm could only incorporate numerical data. Therefore the categorical and ordinal data was transformed into binary as a numerical class. This was done by technique called Get Dummy in pandas.

- ✓ Handling missing data

Machine learning algorithms cannot handle the missing values in data, usually there are two solution when dealing with missing data; first we removed entirely the features/observations with the missing values. This could potentially harm our model by introducing biases in our data. The other solution which is dependable is usually to use imputation. This is where we replaced the missing value with another value. The strategies we used was by evaluating either mean, medium or highest frequency of the given feature. In Sci-Kit Learn we imported Imputer module and fit the missing data with the desired strategy.

```
Train_PatientDetail_labeldata[cols1] = Train_PatientDetail_labeldata[cols1].fillna(value=0)
Test_PatientDetail_labeldata[cols1]=Test_PatientDetail_labeldata[cols1].fillna(value=0)
```

- ✓ Data standardization

The data was normalized to normal distribution with unit variance. Using sci-kit learn Standard scaler.

```
sc=StandardScaler(copy=True, with_mean=False, with_std=False)
sc.fit(X)
X_std=sc.transform(X)
```

a. Data exploration

The data was explored by understanding the domain under which the data is extracted. Data exploration was achieved through various ways.

- ✓ Outlier detection

An outlier is an observation that deviates drastically from other observations in a dataset. This drastic deviation could as a result of natural occurrence or an error. Natural occurrence in data is not normally problematic but could skew the model by affecting the slope. An error in data could indicate the bad quality of data. There are different approaches to correct outliers in data. Using Tukey Inter-Quartile Range (Tukey IQR) method identifies extreme values in dataset. The outliers are defined as values below q1-1.5(q3-q1) or values above q3+1.5(q3-q1) where 1.5 is an arbitrary number.

The other approach that was used was Kernel Density Estimation. This is a non-parametric way to estimate probability function of a given feature. It captures the outlier in bimodal distributor.

- ✓ Distribution of Features

A histogram is a simple representation of the distribution of values of a given feature. X- axis usually represents the value bins while Y-axis represents the frequency of the observation falling into that bin. This allowed us to see in summery the distributions of the features selected for the model.

3.3.3: Data Transformation

Feature Construction

After collecting and labeling data, variable attributes were constructed. The feature information we dealt with included; Member (patient) information, Provider (Facility, Doctor) information, Claim details (Contract information, Amount billed, Diagnosis codes, Dates of service etc). These were operationalized into distinct feature variables spanning our columns dataset. more features were created from already existing columns by either averages or sum of the features.

```
##AVERAGE FEATURES BASED ON GROUPING VARIABLES
###Provider
Train_PatientDetail_labeldata["PerProviderAvg_DeductibleAmtPaid"]=Train_PatientDetail_labeldata.groupby('Provider')['De
Train_PatientDetail_labeldata["PerProviderAvg_IPAnnualReimbursementAmt"]=Train_PatientDetail_labeldata.groupby('Provide
Train_PatientDetail_labeldata["PerProviderAvg_IPAnnualDeductibleAmt"]=Train_PatientDetail_labeldata.groupby('Provider')
Train_PatientDetail_labeldata["PerProviderAvg_OPAnnualReimbursementAmt"]=Train_PatientDetail_labeldata.groupby('Provide
Train_PatientDetail_labeldata["PerProviderAvg_OPAnnualDeductibleAmt"]=Train_PatientDetail_labeldata.groupby('Provider')
```

Feature Selection

Univariate feature selection examines each feature individually to determine the strength of the relationship of the feature with the response variable. Because the features were categorical, we computed the chi-square between each feature and the target vector.

Univariate feature selection works by selecting the best features based on univariate statistical tests. We compare each feature to the target variable, to see whether there is any statistically significant relationship between them.

To use chi-square for feature selection, we calculated chi-square between each feature and the target output and select the desired number of features with the best chi-square scores (z).

Feature selection algorithm, was applied to recognize important features showing strong correlation. The algorithm considered one attribute at a time to see how well each predictor alone (feature) predicted the target variable (output). The importance value of each variable was then calculated as (1-p) where p was the p value of the appropriate test of association between the candidate predictor and the target variable. (Assoc Prof. Dr. Othman Ibrahim, 2014) .

Using SelectKBest in feature selection module in sci-kit learn, we fitted chi-square which is a function that returns calculated chi-square matrix for the features. Therefore we can ascertain the valuable features in our training data that has high ranking and are more important towards achieving the target variable.

```
# FEATURE SELECTION

from sklearn.feature_selection import SelectKBest
# from sklearn.feature_selection import f_classif
from sklearn.feature_selection import chi2

bestfeatures = SelectKBest(score_func=chi2, k=10)
# bestfeatures = f_classif(X, y)
fit = bestfeatures.fit(X_std, y)

dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X_std.columns)

#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
print(featureScores.nlargest(10,'Score'))  #print 10 best features
```

a. Class imbalance

Because our dataset was highly imbalanced we employed a technique in Sci-Kit Learn where we combined two methods of oversampling called Synthetic Minority Oversampling technique (SMOTE) and Tomek Link Removal.

SMOTE

Is a popular technique that helps to rebalance data as aims to create new minority class examples (synthetic instances) by interpolating between several nearest minorities examples rather than by oversampling with replacement. Depending upon the amount of oversampling required, nearest neighbors of minority examples are randomly selected. (Shakya, 2018).

TOMEK LINK REMOVAL

Tomek Link is a pair of examples of different classes which are each other's nearest neighbors. Given two samples E1 and E2 belonging to different classes, a pair (E1, E2) is a Tomek Link if there's not a sample E3 such that the distance between E1 and E3 is less than that of E1 and E2 or the distance between E2 and E3 is less than that of E1 and E2. Removing Tomek links can be considered as an under sampling approach, where the majority sample in the Tomek link is eliminated. (Shakya, 2018).

SMOTETomek

Combining SMOTE and Tomek Link Removal is a powerful approach to balance the class distributions. However, while creating new synthetic minority examples, the minority class cluster might invade the majority class space. Providing such data to the model can lead to overfitting. Hence, to mitigate such a situation, both SMOTE and Tomek Link removal approach can be applied for balancing the class distribution. In this process, the original training dataset is first oversampled using SMOTE, and then Tomek Link removal is applied to the resulting dataset producing a balanced dataset. (Shakya, 2018). SMOTE does oversampling and Tomek does cleaning over Tomek links.

3.3.4: Data mining

Model building

After preprocessing and labelling our data, the model was built using Sci-Kit Learn library in python to fit a Binary Logistic Regression model.

```
###LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegressionCV
log=LogisticRegressionCV(Cs=10, fit_intercept=True, cv=10, dual=False, penalty='l2', scoring=None, solver='lbfgs', tol=
```

3.3.5: Interpretation/Evaluation

Our model evaluation was done by interpreting the results by cross validating with other literature.

Performance metrics

In accessing medical claim processing we were presented with two-class classification, either correct claim or review claim. In our study, the area of interest is correct claim as we trying to expedite the process of clearing medical claims so as to fasten the payment process to the provider and reduce the cumbersome and tiresome process to the insurer.

There is confusion matrix for each model and that is used to assess the performance of learners. Confusion matrices compares actual counts against predicted counts. From the resultant matrices, we employed AUC to measure claim clearing performance. AUC is the Area under the Receiver Operating Characteristic (ROC) curve, where ROC is the comparison between false positive (fall-out) and true positive (recall). Recall is calculated by TP/TP+FN and fall-out is calculated by FP/FP+TN. (Richard A. Bauder, 2018).

Figure 3.1 2 Table summary of suggested methodology by author.

Objectives	**Methodology**
1.To investigate and identify independent variables that can be used to generate a model for assessing medical insurance claims	(R. A. Bauder and T. M. Khoshgoftaar, 2017).
2. Investigate an appropriate machine learning algorithm that can use the identified factors to assessing insurance claims.	(Brid, 2018). Logistic regression. (Mohit Kumar, 2014). Online desk research.
3. To train the algorithm to generate an appropriate model for assessing insurance claim	(Bekkar, Djemaa, & and Alitouche, 2013) Applying Support Vector Machines to Imbalanced Data Sets. (Rehen Akbani, 2004)
4. To Evaluate the performance of the established model	(Assoc Prof. Dr. Othman Ibrahim, 2014). (Jeni, Cohn, & and De La Torre, 2013)

CHAPTER FOUR

DATA ANALYSIS, FINDINGS AND DISCUSSIONS

4.1: Introduction

In this chapter we are going to discuss results and findings as examined from the methodology. The data used was retrieved from public use case file in an online portal which was released on Kaggle, a portal for data scientist. The data describes the patient information for the elderly persons in India seeking medical assistance especially for the chronic illness. Due to sensitivity of the information contained in the medical data most identifying features were removed or coded to safeguard the privacy of the individuals.

All the datasets were merged and through various preprocessing procedures obtained desired data for our algorithms training as the study will further explain on the Model.

Our datasets was imbalanced since majority of the claims were correct with small percentage as review. The study will discuss in detail on how we handled the imbalanced data.

4.2: Descriptive statistics/Response rate

All the data were rolled from different angles so the study could understand the nature of the data. Then examined the data types of our features. Most features are continuous variables.

The class variable is highly imbalanced as shown in the figure. The study used a technique in scikit-learn called SMOTETomek to solve the nature of the imbalanced data.

```
PotentialFraud
No     328343
Yes    189394
dtype: int64
```

Figure 4.1 1 Distribution of Potential Fraud class

```
Percent Distribution of Potential Fraud class:-
 No     63.418879
Yes    36.581121
Name: PotentialFraud, dtype: float64
```

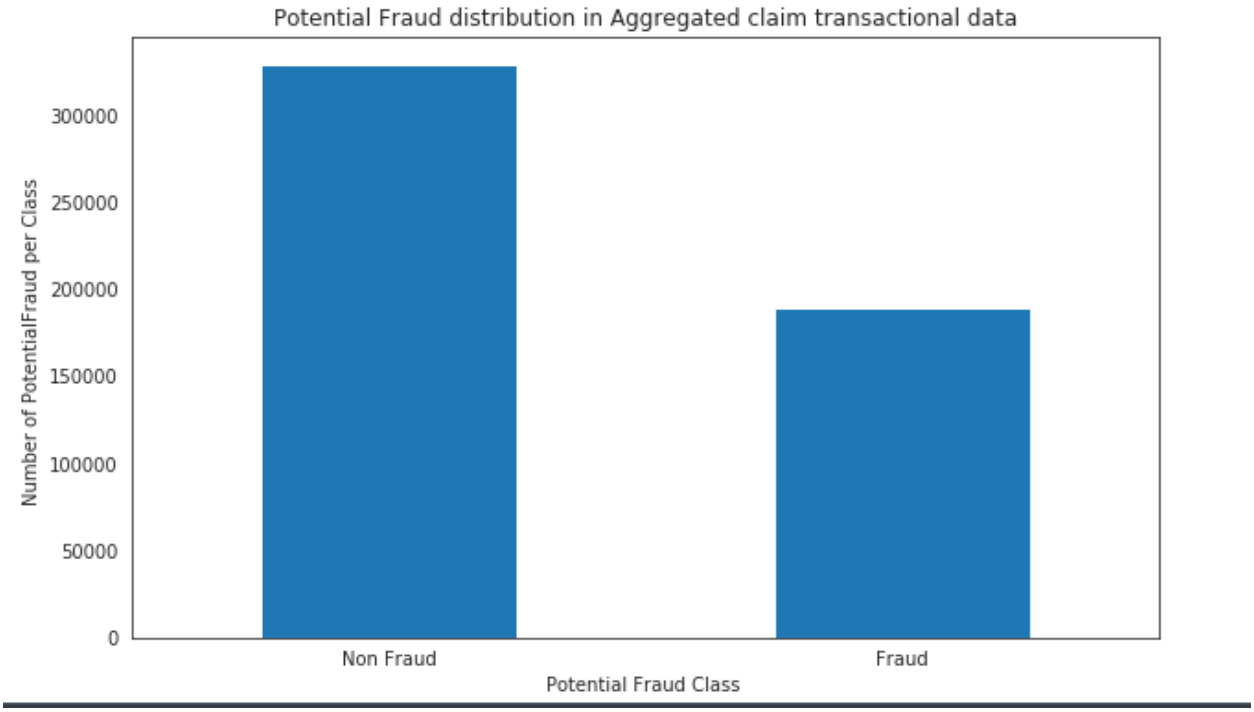


Figure 4.1 2 Percent Distribution of Potential Fraud class

4.3: Demographic Information

The data was multi-dimensional and had various features that explained the demographic information of the patients. Beneficiary information is one of the independent variables that contribute to generating the model that can be used to assess or classify medical claim. Demography of the patient gives a lot of information and intuition on how the claim will be classified. We looked at the distribution of the state and race of the beneficiary in relation to fraudulent claims and the race with most fraudulent claims. State 5, 10, and 33 had majority of the beneficiary enrolled. State 5 still had the highest percentage of the fraudulent claims. State 3 had the highest proportion of fraudulent claims as compared to the beneficiary distribution.

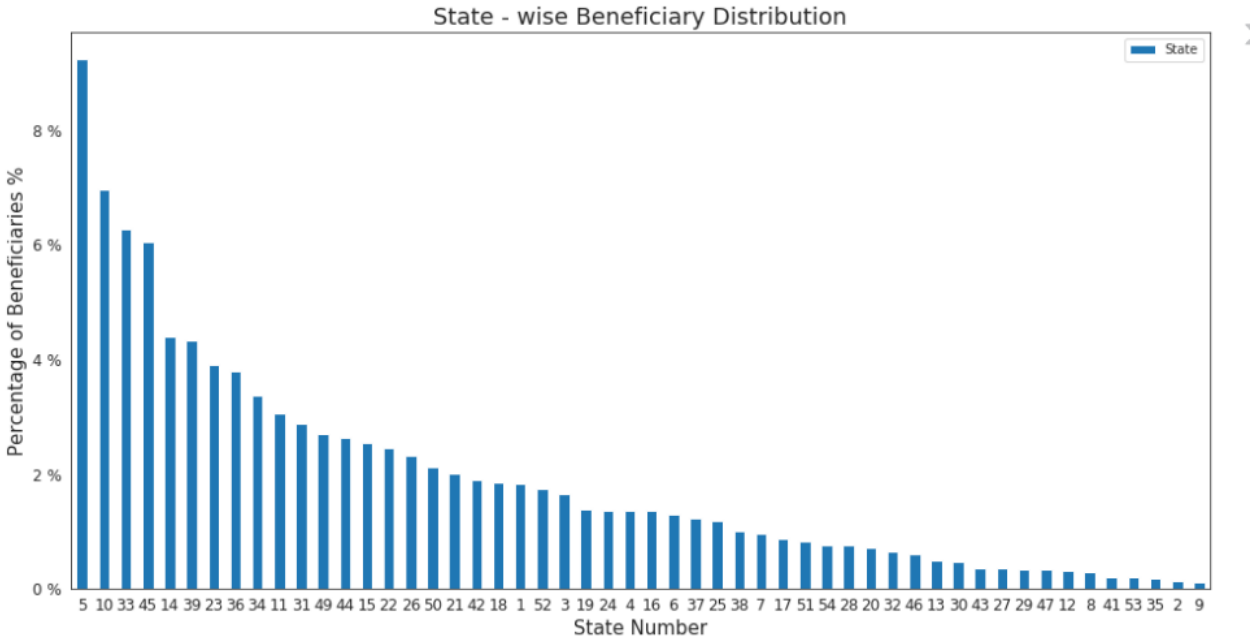


Figure 4.1 3 State-wise Beneficiary Distribution

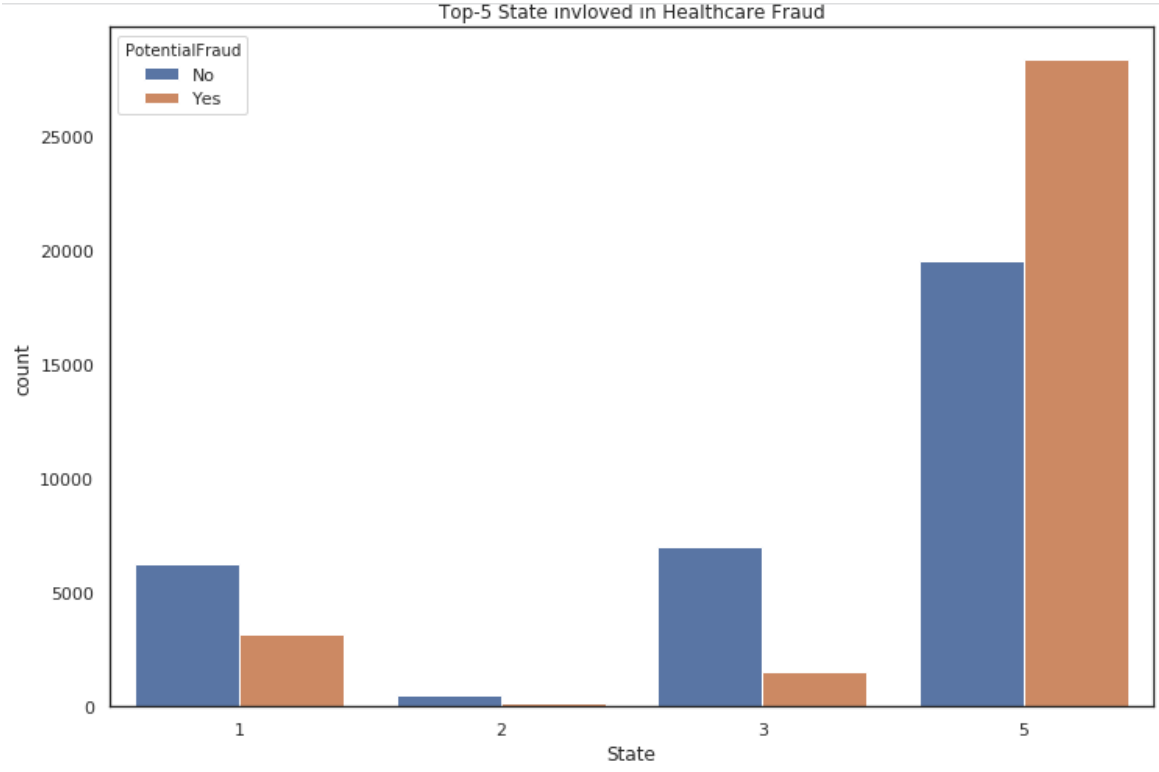


Figure 4.1 4 Top-5 State involved in Healthcare Fraud

Majority of the beneficiary came from Race 1 with over 80% of the dataset. Race 1 and 2 contained over 90% of the population.

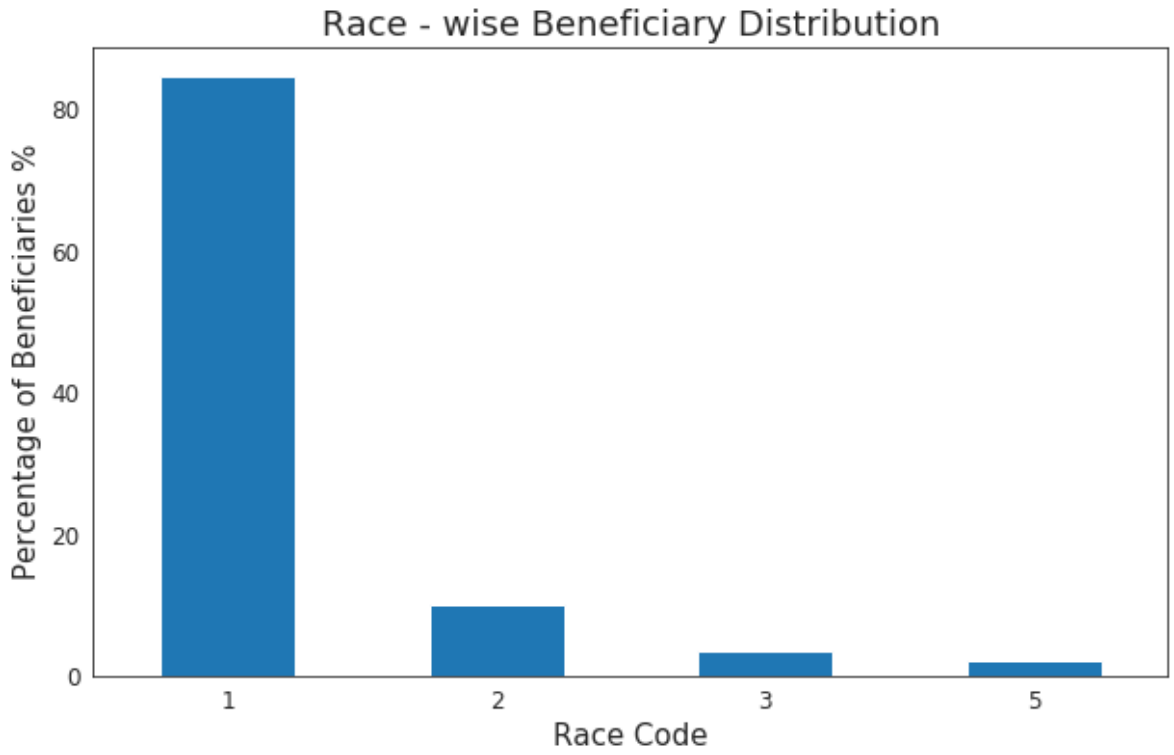


Figure 4.1 5 Race-wise Beneficiary Distribution

As per below table, we can see that Race 1 had the highest proportion of the fraudulent claims.

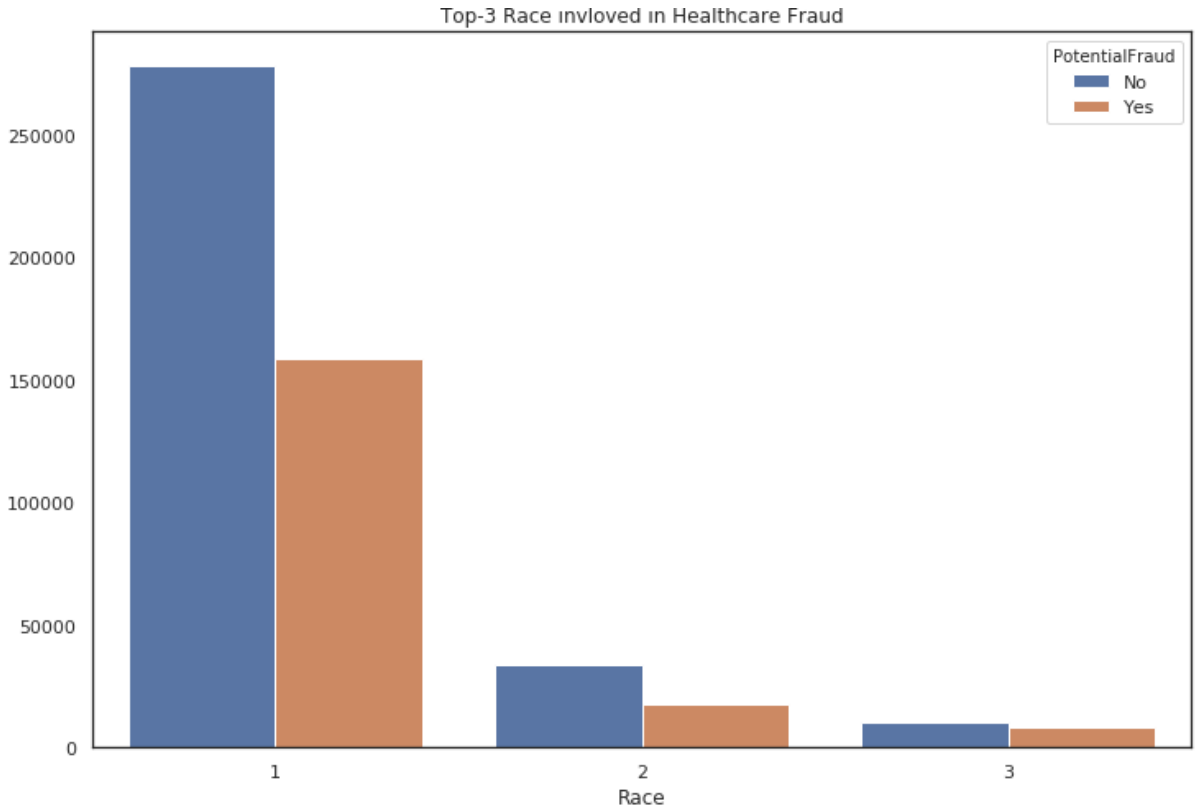


Figure 4.1 6 Top-3 Race involved in Healthcare Fraud

4.4: Independent Variables

The study variables observed from the literature review were highlighted as part of the observation that answered our research questions. Patient information consists of the identifying features that tells us the overview of the beneficiary. We will discuss research findings that answers the research objectives and research questions as postulated in chapter one.

4.4.1: Objective one results

The dataset had these independent variables that were used to predict the target variable. The study examined the presence of these variables and tabulated the top features as shown in the graphs below.

.

a) Provider information

As discussed in literature review, the provider information is integral independent variable that determine our target variable, we analyzed the top providers who were involved in fraud;

There were various providers in the dataset, but analyzed the top 10 providers who were involved in the fraud cases. These unique providers presents us with overall intuition on how to deal with them when processing the medical claims.

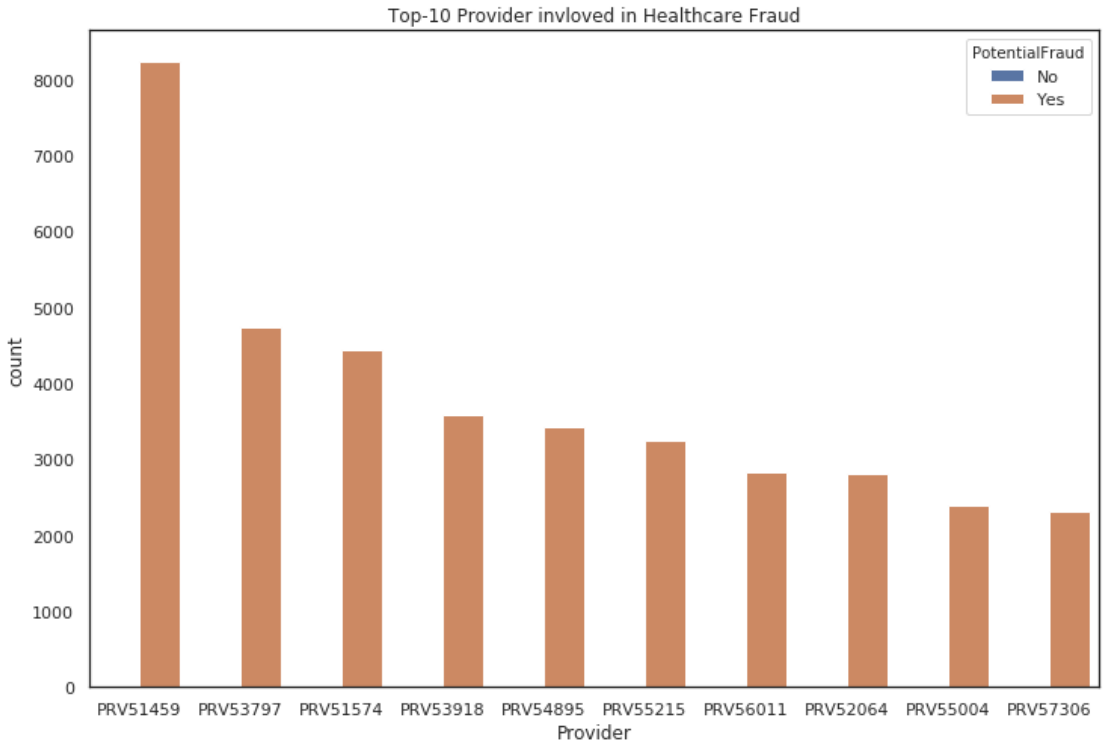


Figure 4.1 7 Top-10 Provider involved in Healthcare Fraud

Provider information had several independent variables (features) that determined our target variables as outlined in literature review. Among the top features that determined the outcome variable in provider information were;

Attending physician

These are medics who offer medical services to the patient. The study analyzed the trend behavior of these medics and tabulated in graph below the top 10 medics who were involved in fraudulent claims. Every physician has a unique identifier and handles various patients at various providers.

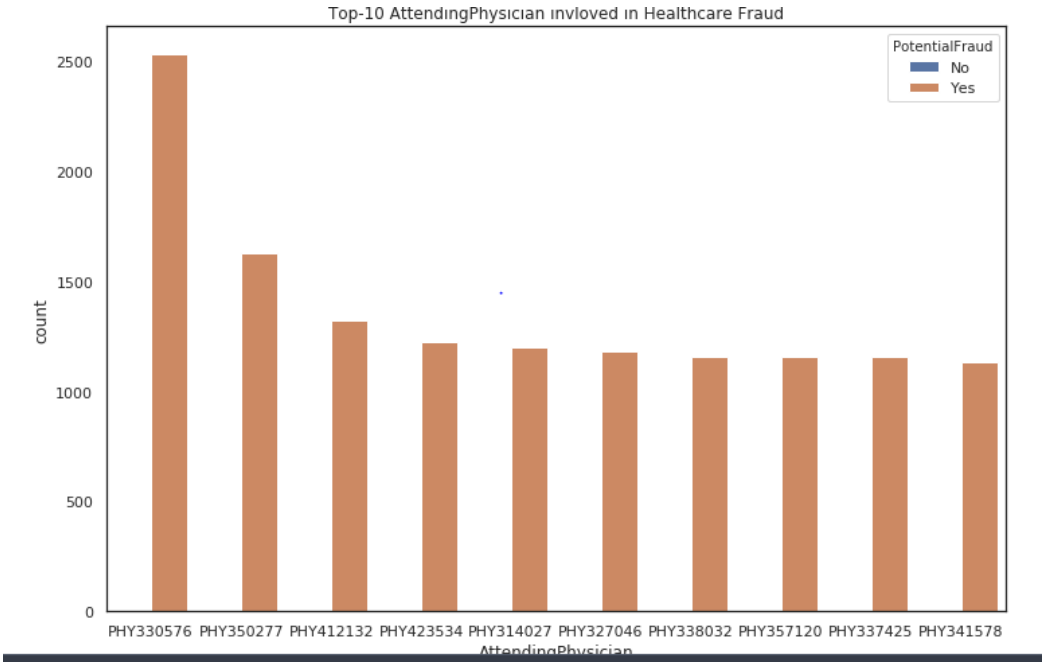


Figure 4.1 8 Top-10 Attending Physician involved in Healthcare Fraud

Operating physician

Operating physicians offer operating procedures to patients who required these services. Compared to attending physicians the operating physicians handled fewer cases and had low ratio of fraudulent cases.

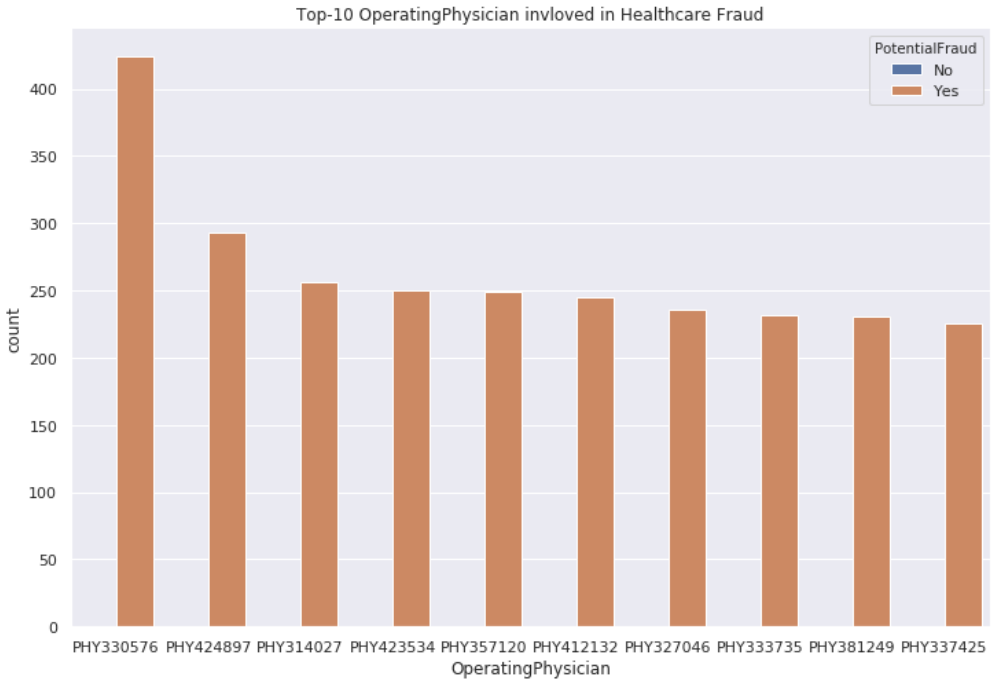


Figure 4.1 9 Top-10 Operating Physician involved in Healthcare Fraud

b) Claim details

Claim details is the treatment that is prescribed to the patient/beneficiary. This involves diagnosis and procedures performed on the beneficiary. The graphs below shows that Diagnosis 4019 was the leading diagnosis that lead to fraudulent claims and was the most performed diagnosis overall. The outpatient deductible amount that had highest proportion was lower amount from 0-20 usd.

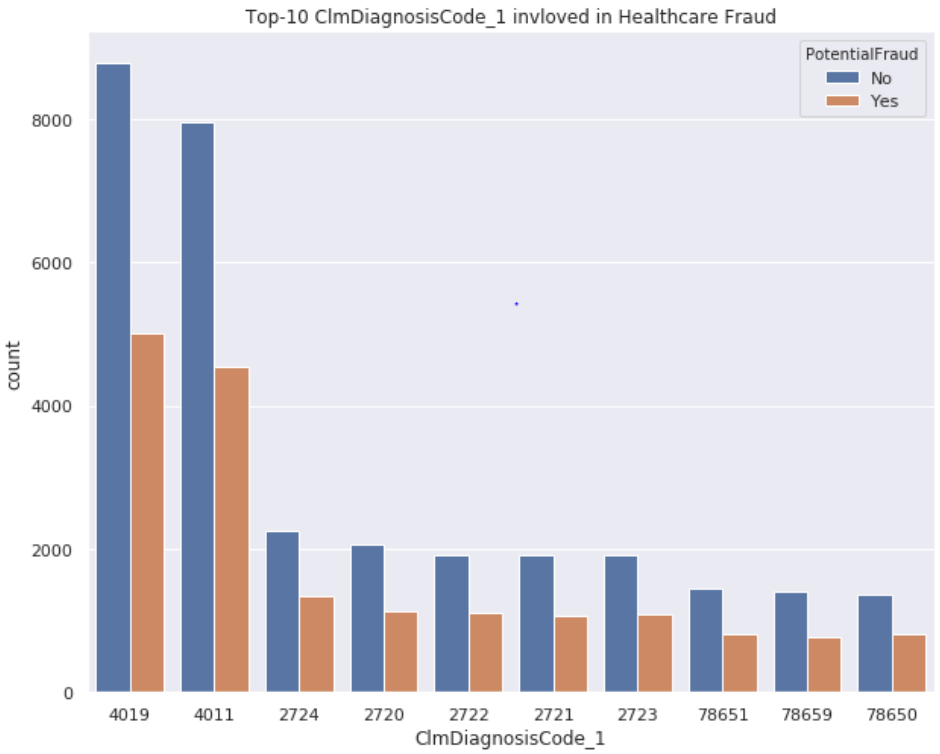


Figure 4.1 10 Top-10 Claim Diagnosis involved in Healthcare Fraud

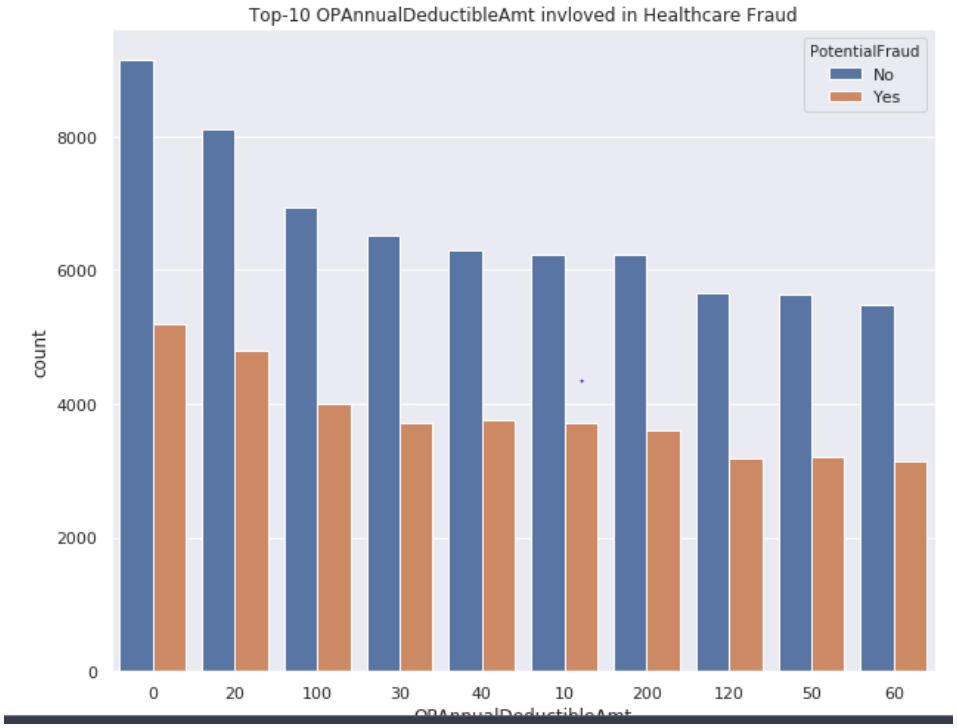
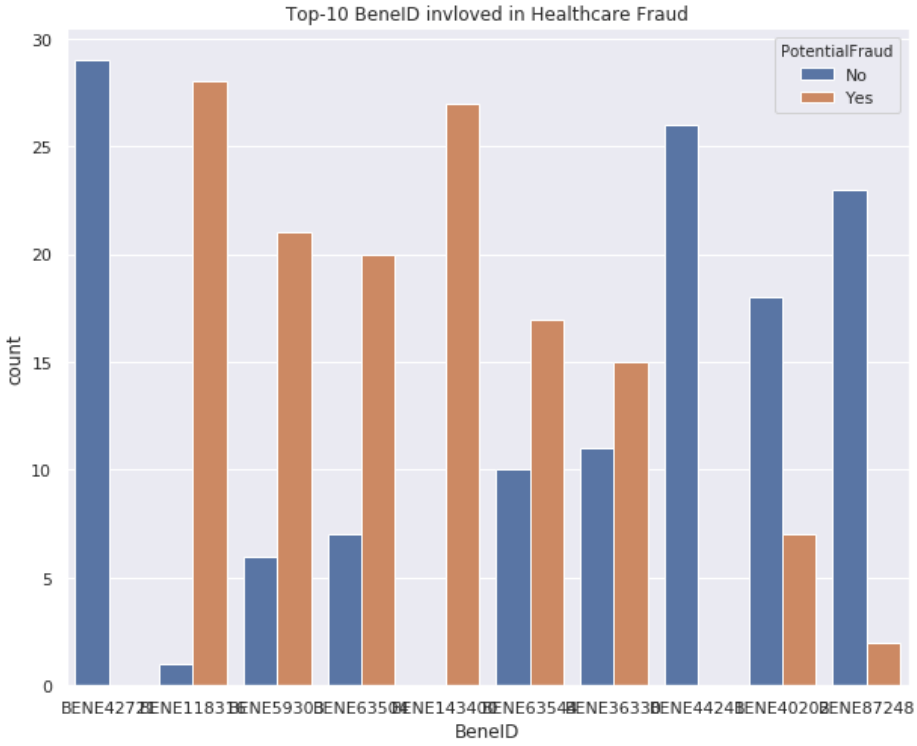


Figure 4.1 11 Top-10 Outpatient Annual deductible amount involved in Healthcare Fraud

c) Patient information

Patient/beneficiary information helps in identifying the right beneficiary. For every unique beneficiary could seek various medical assistance from different providers and attended by different physicians. This culminated in a trend that shows the beneficiary fraudulent behavior. The most treated beneficiary did not show any trend of fraudulent behavior. But BENE1183 had the highest proportion of fraudulent cases. Out of 28 claims only 1 was not a fraud case.



c

Figure 4.1 12 Top-10 Bene ID involved in Healthcare Fraud

Age

Our dataset shows that majority of the persons seeking medical services were elderly.

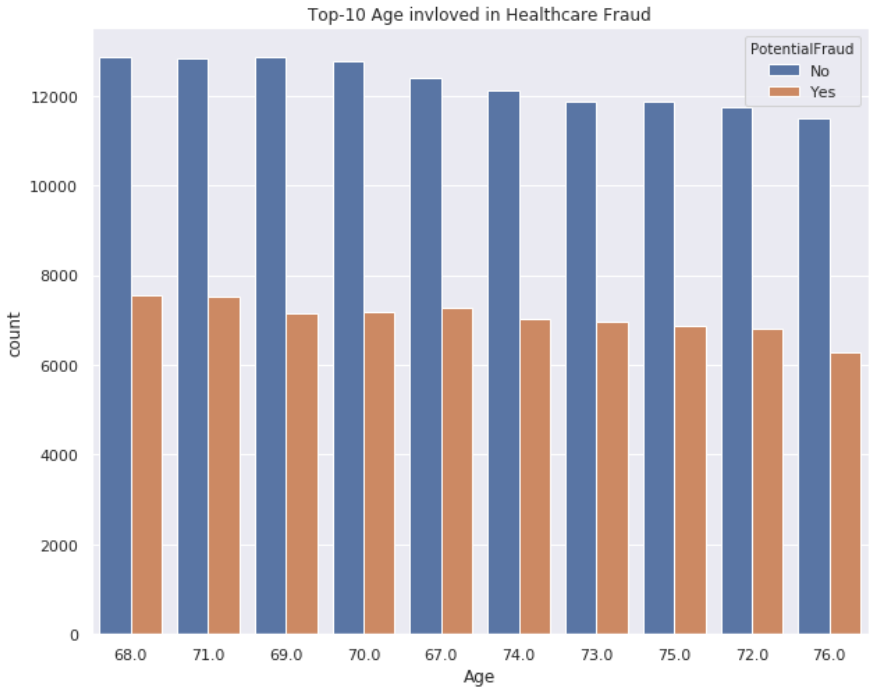


Figure 4.1 13 Top-10 Age involved in Healthcare Fraud

Race

Most beneficiary who made medical claims were from one race.

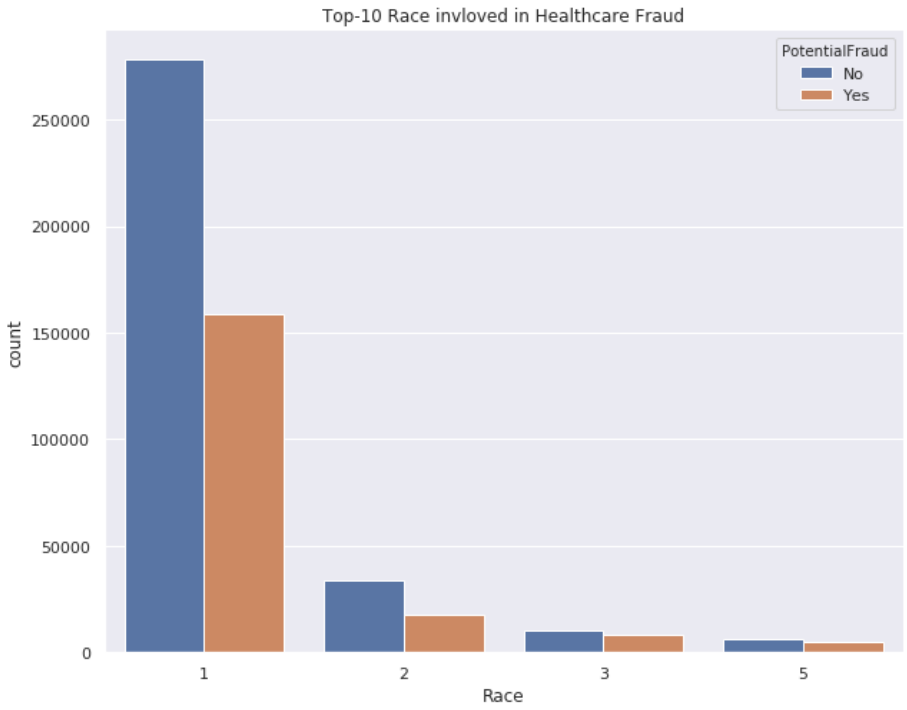


Figure 4.1 14 Top-10 Race involved in Healthcare Fraud

d) Policy information

From the literature review, policy information is useful in determining the admission of the beneficiary at the health facility, this information is highly useful in identification by the provider before any health service is provided. This is done at the point of admission to the providers' health facility and thus has no impact in determining whether the claim status will be correct or review. In our dataset the provider information was not included and thus had no impact in our model.

4.4.2: Objective 2 results

The choice of machine learning algorithm used was dependent on the nature of the dataset used. The insurance claim dataset is data with high correlation. The features are correlated and there is class imbalance. The majority label are for the one class, NO fraudulent claim. From the literature review we had examined the validation of classification algorithms to be used. This is a binary classification task in supervised machine learning.

The most suited classification algorithm chosen was Logistic Regression and Support Vector Machines.

a) **Logistic Regression**

Logistic Regression is among the most popular methods used to fit models for categorical data, it fits binary response data in Data Modeling. It is the most important (and probably most used) member of a class of models called generalized linear models (Brid, 2018)

Logistic regression does not encompass linear relationship between dependent and independent variables. It usually handles different types of relationships as it applies a non-linear log transformation to the predicted odds ratio.

Logistic regression uses maximum likelihood estimates, this requires large samples for estimation. The independent variables in logistic regression does not support multi collinearity

We used stepwise method to estimate the logistic regression by including all significant variables, this helped us avoid both under fitting and overfitting.

b) **Support Vector Machines**

We specifically chose SVM to attack the problem of imbalanced data because SVM is based on strong theoretical foundations (Vapnik, 1995) and our empirical results show that it performs well with moderately imbalanced data even without any modifications.

Its unique learning mechanism makes it an interesting candidate for dealing with imbalanced datasets, since SVM only takes into account those instances that are close to the boundary, i.e. the support vectors, for building its model. This means that SVM is unaffected by non-noisy negative instances far away from the boundary even if they are huge in number. (Akbani, 2004).

The study used two methods to examine the suitability of the algorithm to be used. Content analysis was first used to establish the goodness of the algorithm in respect to various content based metrics according the data at hand. Speed, effect on imbalanced data, speed vis-à-vis accuracy were factors of choice. Then the results were tabulated as shown in Table 4.1 1.

Content analysis

Table 4.1 2 Quantitative content analysis

Algorithm	Speed	Multi-collinearity	Accuracy	Imbalanced data	Sensitivity to outliers	If n=features, m=samples n=1-1000 and m=50,000+
SVM	No if with kernel	Yes	Yes	Yes	No	Yes without kernel(reduces speed
Binary LR	Yes	No	Yes	Yes	Yes	Yes

The study did a comparative analysis using the performance metrics as discussed in literature review and settled on the algorithm with best performance as per the problem requirement. The Table 4.1 3 below was tabulated and as discussed we wanted a better F-1 score and better accuracy.

Table 4.1 4 Qualitative content analysis

ALGORITHM	**RECALL**	**F-1 SCORE**	**AUC**	**ACCURACY**
Binary L-R	0.762	0.617	0.807	0.922
SVM	0.816	0.602	0.846	0.88

From the content analysis we selected Binary Logistic Regression as our algorithm of choice

4.4.3: Objective 3 results

We trained our selected algorithm on the dataset and looped through the whole data mining process. This objective allowed us to explore through various data mining techniques in preparation of fitting our desired algorithm to train the model following Fayyad's et al. (1996) KDD process model as outlined in the methodology.

4.6: Model

1. **Data selection**

We selected our data from data repository on Kaggle, a community of data scientist online portal. We analyzed our data to understand the domain knowledge of the data. We set the target variable as fraudulent claim where we labeled as either fraud or non-fraud Yes or No.

2. **Data preprocessing**

We imported data into our working environment using pandas in Python 3. Pandas library allows us in data processing in csv format. Alongside pandas we imported other required libraries and then loaded files data to our Data Frame.

```
Shape of Train_Beneficiarydata data : (138556, 25)
Shape of Train_Outpatientdata data : (517737, 27)
Shape of Train data : (5410, 2)
```

Figure 4.1 15 Shape of Data

We ascertained the coherence of data by checking data types in each column. We ensured that there were no missing values in our data.

3. Data transformation

a. Feature Engineering

We aggregated the data using various feature engineering techniques in python environment to form one singular data frame for both train and test datasets. Test dataset has no label column and that is what we used to test the data.

```
Test_ProviderWithPatientDetailsdata shape- (125841, 137)
Train_ProviderWithPatientDetailsdata shape- (517737, 138)
```

Figure 4.1 16 Shape of Aggregated data

We aggregated our data to providers' level as the medical claim is done by the provider. We removed some features that had no correlations.

```
Train_data shape (517737, 111)
Test_data shape (125841, 110)
```

Figure 4.1 17 Shape of preprocessed Train and Test Data

We dummified categorical variables and converted target variables 'YES' to 1s and 'NO' to 0s and did train validation split where we specified feature variables and target variable.

	InscClaimAmtReimbursed	DeductibleAmtPaid	Gender	Race	\
0	30	0	2	1	
1	1600	0	1	1	
2	200	0	1	1	
3	1900	0	1	1	
4	400	0	1	1	

	NoOfMonths_PartACov	NoOfMonths_PartBCov	ChronicCond_Alzheimer	\
0	12	12	0	
1	12	12	1	
2	12	12	0	
3	12	12	0	
4	12	12	0	

	ChronicCond_Heartfailure	ChronicCond_KidneyDisease	ChronicCond_Cancer	\
0	0	0	0	

Figure 4.1 18 Dummified Data

The data was then standardized to normal distribution using Standard Scaler in Sci-Kit Learn python environment.

```
[[3.000e+01 0.000e+00 2.000e+00 ... 1.000e+00 1.000e+00 1.000e+00]
 [1.600e+03 0.000e+00 1.000e+00 ... 2.000e+00 1.000e+00 1.000e+00]
 [2.000e+02 0.000e+00 1.000e+00 ... 1.000e+00 1.000e+00 1.000e+00]
 ...
 [1.190e+04 0.000e+00 2.000e+00 ... 2.000e+00 2.000e+00 2.000e+00]
 [1.403e+04 0.000e+00 1.000e+00 ... 2.000e+00 2.000e+00 2.000e+00]
 [1.130e+04 0.000e+00 1.000e+00 ... 2.000e+00 2.000e+00 2.000e+00]]
```

Figure 4.1 19 Standardized Data

b. Feature Selection

Using SelectKBest in feature selection module in sci-kit learn, we fitted chi-square which is a function that returns calculated chi-square matrix for the features. Therefore we can ascertain the valuable features in our training data that has high ranking and are more important towards achieving the target variable when we chose k=20 (20 best features) are as shown

	Specs	Score
80	ClmCount_Provider_AttendingPhysician	59411.164822
81	ClmCount_Provider_OtherPhysician	23830.306758
89	ClmCount_Provider_ClmDiagnosisCode_2	20013.299026
79	ClmCount_Provider_BeneID	15581.870187
88	ClmCount_Provider_ClmDiagnosisCode_1	13736.701991
90	ClmCount_Provider_ClmDiagnosisCode_3	12206.952882
82	ClmCount_Provider_OperatingPhysician	8462.498701
91	ClmCount_Provider_ClmDiagnosisCode_4	6657.214924
83	ClmCount_Provider_ClmAdmitDiagnosisCode	6294.863334
92	ClmCount_Provider_ClmDiagnosisCode_5	3333.946502
93	ClmCount_Provider_ClmDiagnosisCode_6	1743.645563
26	PerProviderAvg_Age	1337.414213
25	PerProviderAvg_OPAnnualDeductibleAmt	1020.188647
94	ClmCount_Provider_ClmDiagnosisCode_7	906.017221
27	PerProviderAvg_NoOfMonths_PartACov	881.363579
107	Race_3	531.775496
24	PerProviderAvg_OPAnnualReimbursementAmt	453.083750
95	ClmCount_Provider_ClmDiagnosisCode_8	410.735192
28	PerProviderAvg_NoOfMonths_PartBCov	223.110972
108	Race_5	197.363216

Figure 4.1 20 Selected K best Features

c. Class imbalance

Because our dataset was highly imbalanced we employed a technique in Sci-Kit Learn where we combined two methods of oversampling called Synthetic Minority Oversampling technique (SMOTE) and Tomek Link Removal.

We implemented SmoteTomek in Ski-Kit Learn and instances of sample datasets increased as shown;

```
X_train without SmoteTomek (362415, 30)
X_Train with SmoteTomek (449838, 30)
```

Figure 4.1 21 Shape of Data after applying SMOTETomek

4. Data mining

a. Model Building

After preprocessing and labelling our data, we built our model using Sci-Kit Learn library in python to fit the model.

LOGISTIC REGRESSIONCV

This class implements logistic regression using lbfgs solvers that support only L2 regularization with primal formulation. Hyper parameter tuning involves using different strategies to optimize the accuracy and validity of the model.

```
LogisticRegressionCV(Cs=10, class_weight='balanced', cv=10, dual=False,
           fit_intercept=True, intercept_scaling=1.0, max_iter=100,
           multi_class='warn', n_jobs=None, penalty='l2', random_state=123,
           refit=True, scoring=None, solver='lbfgs', tol=0.0001, verbose=0)
```

Figure 4.1 22 Logistic Regression Model output in Ski-kit Learn

$$y = \frac{e^{b_0 + b_1 x}}{1 + e^{b_0 + b_1 x}}$$

Equation 5

Logit (p) =B0 + B1X1 + B2X2+ ……. + BkXk Equation 6

Where odds that y = 1 is given by p/(l-p).The log odds or logit of p equals the natural logarithm of p/(l-p). p is probability of response. B0 is constant coefficient and B 1-k is k^{th} coefficient, X is the k^{th} feature.

The probability that Y=1 is given by P(Y=1)=1/(1+e−^(b0+∑(biXi))). Equation 7

	0	1
0	InscClaimAmtReimbursed	[-4.3525103945162374e-05]
1	DeductibleAmtPaid	[-0.000115202271064028]
2	NoOfMonths_PartACov	[-3.210851745415571e-05]
3	NoOfMonths_PartBCov	[0.0007049772232203252]
4	ChronicCond_Alzheimer	[-0.001732879718672382]
5	ChronicCond_Heartfailure	[-0.027939274555389235]
6	ChronicCond_KidneyDisease	[-0.04475204412543798]
7	ChronicCond_Cancer	[-4.3525103945162374e-05]
8	ChronicCond_ObstrPulmonary	[-6.405293659166846e-06]
9	ChronicCond_Depression	[4.033844941531244e-05]
10	ChronicCond_Diabetes	[0.20078685734347299]
11	ChronicCond_IschemicHeart	[0.0029306646747652614]
12	ChronicCond_Osteoporasis	[0.0028989284567489167]
13	ChronicCond_rheumatoidarthritis	[-0.0003509681141445795]
14	ChronicCond_stroke	[0.13660233055407103]
15	IPAnnualReimbursementAmt	[0.06493872923266465]
16	IPAnnualDeductibleAmt	[0.0408043822514661]
17	OPAnnualReimbursementAmt	[0.0448788732381542]

Figure 4.1 23 Selected Logistic Regression Coefficients

The coefficients above can fit into our model when P(y=1) for all the selected variables. The logistic regression coefficients give the change in the log odds of the outcome for a one unit increase in the predictor variable.

In Figure 4.1 24 any output value that is greater than 0.5 is considered as 1 otherwise is 0. When it's 1 then it means there is fraudulent claim hence it's under review, and 0 means No fraudulent claim hence its correct claim to be paid.

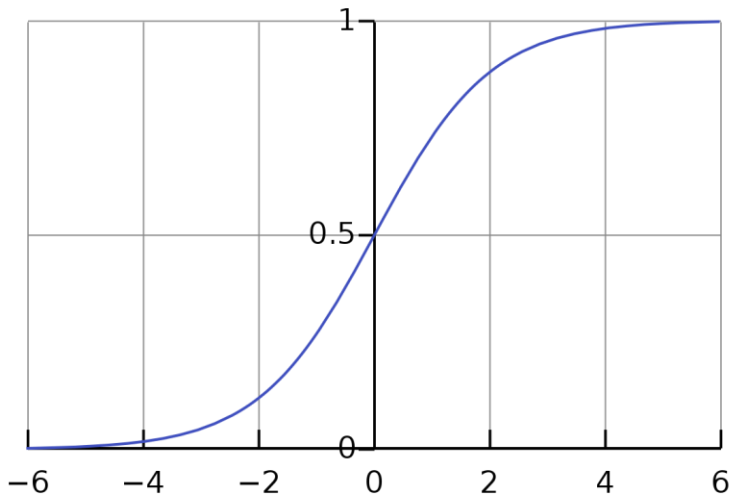


Figure 4.1 25 Logistic Regression curve

5. Evaluation

ROC curve

Receiver Operating Characteristics (ROC) is a graph showing the performance of a binary classification model. It evaluates TPR (Precision) versus FPR. Where;

$$\text{True Positive Rate} = \text{Sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{False Positive Rate} = (1 - \text{Specificity}) = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

Sensitivity and specificity are inversely related as we change the probability threshold, (i.e., when we decrease the threshold, sensitivity increases while specificity decreases and when we increase the threshold, sensitivity decreases while specificity increases). From the ROC curve, we can calculate the area under the curve which is the probability that a model will rank a randomly chosen positive instance higher than a randomly chosen negative one. (Shakya, 2018)

The ROC curve determines the sensitivity of the data and specificity of the false positives.

The cost of the false negatives is much more than the benefits of the True positive. Hence in our model the evaluation criterion that predicts correctly the negative class is more suited than the positive class.

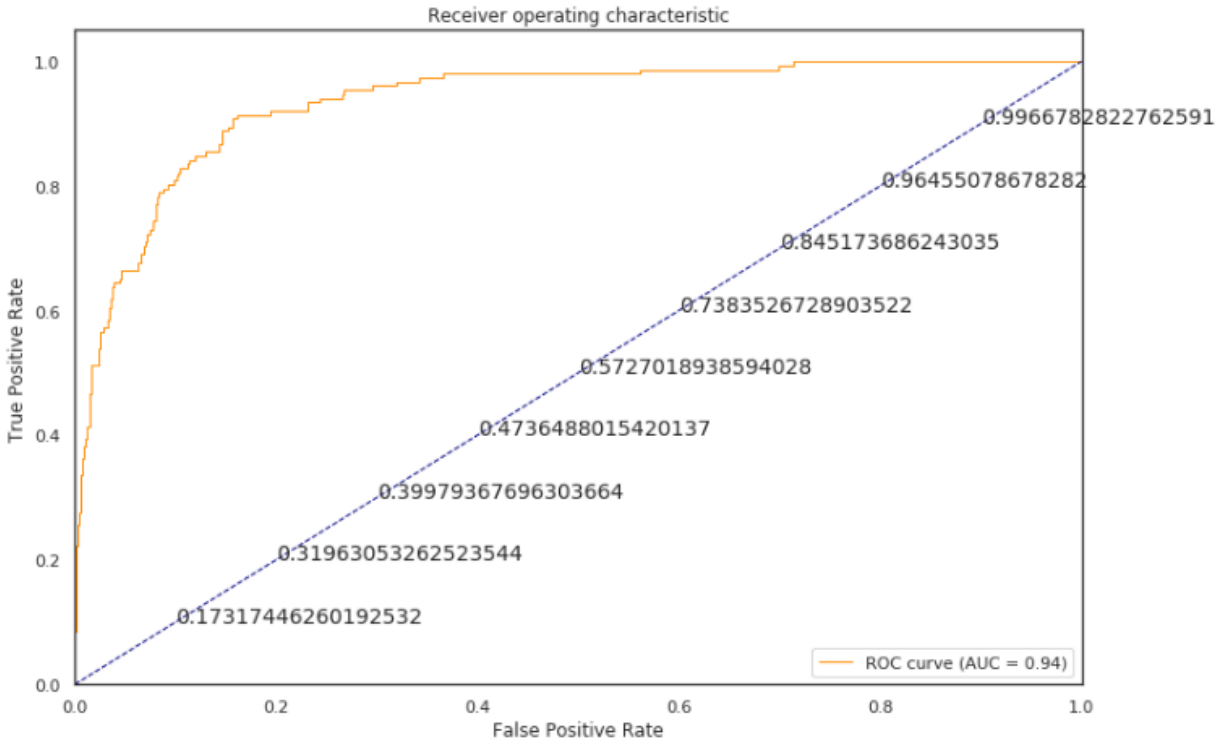


Figure 4.1 26 ROC curve

Accuracy

This is the proportion of total number of correct predictions over the total number of all predictions. For our data the accuracy is not suitable evaluation matrix.

F1 Score

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$

Equation 8

This evaluation matrix creates a balance between precision and recall. F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall AND there is an uneven class distribution (large number of Actual Negatives). (Shung, 2018). Because in our case we had uneven distribution of the actual negatives, we evaluated our model choice by looking at the F1 score.

SUMMERY EVALUATION MATRIX

```
Confusion Matrix Train :
 [[ 270   84]
 [ 210 3223]]
Confusion Matrix Val:
 [[ 103   49]
 [  93 1378]]
Accuracy Train:  0.922365988909427
Accuracy Val:  0.9125077017868145
Sensitivity Train :  0.7627118644067796
Sensitivity Val:  0.6776315789473685
Specificity Train:  0.9388290125254879
Specificity Val:  0.9367777022433719
Kappa Value : 0.5438304105142315
AUC         : 0.8072046405953702
F1-Score Train  :  0.6474820143884892
F1-Score Val  :  0.5919540229885056
```

```
Confusion Matrix Train :
 [[ 320   34]
 [ 388 3045]]
Confusion Matrix Test:
 [[ 124   28]
 [ 181 1290]]
Accuracy Train :  0.8885661473461843
Accuracy Test :  0.8712261244608749
Sensitivity :  0.8157894736842105
Specificity :  0.8769544527532291
Kappa Value : 0.47733173495472203
AUC         : 0.8463719632187199
F1-Score Train 0.6026365348399246
F1-Score Validation :  0.5426695842450766
```

Figure 4.1 27 L-R Summary Evaluation Matrix

Figure 4.1 28 shows the summary of the results from Sci-kit learn. From the first experiment we can see that from confusion matrix the correctly identified class were 270 TP and 3223 TN and the incorectly identified class were 84 FP and 210 FN. This achieved the accuracy of **0.92** and recall of **0.76.**

Table 4.1 5 Result summary

Experiment	**Accuracy**	**Recall**	**AUC**	**F-1 Score**
1 k=30	0.88	0.82	0.84	0.60
2 k=100	0.92	0.76	0.80	0.64
3 k=50	0.91	0.80	0.81	0.62
Average	**0.90**	**0.79**	**0.81**	**0.62**

We did three different experiments while setting different number of best features to be included as shown above. It shows that with more features included we get better results but the time taken to run the model increases.

Table 4.1 6 shows that the accuracy of the model is in good range with average of **0.90**. Despite our dataset being highly imbalanced, we used a SmoteTomek technique to balance our data. Accuracy is usually not suitable evaluation metric for the fraud detection models.

Recall of **0.62**, also called sensitivity is a measure of percentage of total relevant results correctly classified by the algorithm. Usually is in tandem with accuracy. The higher the accuracy and precision the lower the recall. Recall is more suited evaluation matrix for the case when there is high cost associated with false negative. For our project we were identifying the correct claims to be paid. For instance it will be costly if the model predicted that the claim was not fraudulent (correct to be paid) and then it was paid only later to realize that it was incorrectly predicted. This would mean we paid the wrong claim. Hence for our model to be effective we paid more attention to F-1 Score.

F-1 score is a combination of both precision and recall, it is a trade-off to find a working balance for both. F-1 Score is a harmonic mean of precision and recall measures. It is usually the better model performance evaluation metric for the fraud detection problems.

Also from our confusion matrix, Logistic regression had less False Negatives than SVM. Hence Logistic Regression becomes the algorithm of choice as we are trying to reduce the number of false negatives more than false positives.

4.7: Discussion of the Results

Medical claims fraud is categorized as organized crime which involves peers working in cohort to create fraudulent claims. Additional features from grouping them helped in improving accuracy of prediction and fraud pattern recognition. Aggregating and grouping numeric features to provider level helped in detecting behavior of their transactions overall.

The independent variables used in our study (patient information, provider information and claim details information) differed from (R. A. Bauder and T. M. Khoshgoftaar, 2017) as our data did not incorporate the policy information data. We deduced that policy information is the eligibility criteria and thus before any patient is given any medical service, must be eligible for such. Hence these independent variables has no direct contribution to our output variable. We dropped the policy information and achieved an **AUC** of **0.81** that was in range with the (R. A. Bauder and T. M. Khoshgoftaar, 2017) of **0.83.**

We used Logistic regression unlike (Jeni, Cohn, & and De La Torre, 2013) recommendation to use SVM when dealing with imbalanced, fraud detection data. Our data had large samples and various features. With this the running of the model becomes very slow and needs large computer power. Unless we used Linear kernel which has lower accuracy scores. Logistic Regression was algorithm of choice because it is easy to interpret and has good speed. Our study using the L-R achieved Recall of **0.79** as compared to their **0.80**

From the table of result summary we can see that our model performed well with average F-1 score of **0.62** which is a good evaluation matrix of the logistic regression model for classifying fraud detection problems. Because we had few positive class we preferred F-1 score to AUC. Compared to other studies (Shakya, 2018) Application of Machine Learning Techniques in Credit Card Fraud Detection, they got an average F-1 score of **0.59** when using logistic regression. Recall value of **0.79** shows that the positive class was correctly identified with 79% certainty.

4.8: Summery

In this project we applied machine techniques to classify medical claims, if they are either correct for payment or not. The data was collected from the public use file, on online portal for data scientist Kaggle. We applied machine learning cycle and finally fitted algorithm for model construction. We used python environment with different machine learning libraries. We used supervised learning for binary classification algorithms. The novice of our study was incorporating the hybrid class imbalance technique of SMOTE and Tomek Link Removal. The SmoteTomek was implemented in Sci-Kit Learn and oversampled the minority class while removing the excess to balance and avoid overfitting.

Applying machine learning predictive models with unbalanced input data causes bias towards majority class. This usually misrepresents the fraudulent transactions as genuine hence

increasing false positives. Because we wanted to classify claim correctly and with certainty, we had to minimize both false positives and false negatives. Our model performed well though we need to improve model performance by adding more features to our dataset. We need to improve our F-1 score and reduce the number of false negatives before the model can be deployed in real world. Healthcare is a sensitive industry and very restrictive in adopting technology, therefore the industry must be totally convinced by assuring safety and security of the patients data before adoption of technology in handling claim assessment.

The cost of misclassifying the claim is different. Suppose we classified a fraudulent claim as a correct claim then it becomes very costly for the insurer to pay for a fraudulent claim. But also if we were to deny a genuine claim for a reason that it was fraudulent then the cost associated with repairing the damage caused to the customer becomes high and we risk damaging the insurers' organization brand loyalty.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1: Introduction

In this chapter, we are self-evaluating whether we met our general objectives of the study. We will cover the conclusions of the study and some recommendations that could spur future research in this field.

5.2: Conclusions

From our study, we can draw some conclusions. The study was focused on identifying the correct claims to be paid and to review those that were classified as review or fraudulent for further audit by medical assessors. Our study was to determine the best model for medical claim assessment. Because we were dealing with fraudulent data which is usually imbalanced, we used a unique method that balances the data and simultaneously removes over fit. The SMOTETOMEK method was used to balance the data.

The study used best features with required correlation with the target variable using SelectKBest with chi-square. Using features with high chi-square scores outputs improved results. The relationship between the input variable and output variable was determined independently and the best input variable were chosen. The model was evaluated and for this particular case study, we wanted to reduce the number of false negatives as much as possible. The cost of reputational damage much outweighs the customer experience damage. Hence we would rather have more number of false positives than false negatives. Our false positives were classifying non fraudulent claims as fraud and false negatives were classifying fraudulent claims as non-fraud. Therefore for our model to be effective it has to effectively minimize the false negative.

Logistic regression is a good algorithm for fraud detection problems. It fits massive data with averagely good running time. It requires hypermeters tuning to achieve better results. The more optimization is done the more time it takes to run.

Our F-1 score is a bit lower than expected. The validity of the model is not convincing for deployment into real business enterprise. The running time for Logistic regression is slow especially when we were running the whole dataset without choosing the best features. The running time is effected mostly by the number of features. The more the features the slower the model runs. The number of samples also determine the running time of the model. Because we used SMOTE to oversample the minority class, the number of the samples of the training data increased hence contributing the slowness of the model. Using different number of features gives different evaluation results. The more the features the better the accuracy but the slower the model runs. Hence finding compromise of time and feature accuracy has to be made.

The overall importance of the study was to effectively use this formulated model to reduce operational cost of claim assessment hence reducing the healthcare cost for both the insurer and insured. Insurance industry is very sensitive especially for the data of the beneficiaries. This hinders the academic research on the industry. Because of the sensitivity of the information in the dataset we are limited to some features. This in effect reduces the accuracy and validity of the formulated model. With limited set of data we cannot effectively detect the networks of colluded beneficiaries, providers and physicians in perpetrating medical claim fraud.

5.3: Limitations of the study

The study was limited to chronic diseases and the patterns in which they are claimed. In this study we used data mostly on chronic diseases and few instances of the general diseases. The patterns in claims for this diseases could differ or not to other less chronic or general diseases. Therefore we recommend the research community to use data for less chronic diseases and evaluate the patterns in claims of these diseases.

5.4: Contributions

We excluded the policy information in our model, without policy information features the model did not suffer significance loss in accuracy. Hence policy information in model construction is negligible. According to (Ilker Kose, 2015) study, as much as the policy information is very vital in identification of the patient and the eligibility criterion, the model construction is not much based on this rules but is complimented by these rules. The study dropped this information in trying to learn the effects it will have on the performance on the model. Therefore future studies can research on specifically using policy information as control experiment and alternating other variables to evaluate the performance of the model.

Our study was unique in that we incorporated a unique minority oversampling technique that is fused with Tomek link removal. This balanced our data and avoided overfitting of the majority class. This helps solve a problem of imbalanced data especially in fraud detection problems like healthcare claims data. This has been a challenge in detecting fraud problems as postulated by (Shakya, 2018) in his study Application of Machine Learning Techniques in Credit Card Fraud Detection

Fraud detection in healthcare claims is organized crime that encompasses providers, beneficiaries and the insurers. Using machine learning and data mining techniques to detect fraud patterns is vital in reducing operational cost for the insurers and hence reducing healthcare cost for the patient. By automating the model of claim assessment process, the insurance company can pay more attention to their core business which is underwriting. This research is useful as it contributes to achieving universal healthcare for all, by reducing healthcare cost governments and individual can reduce the burden of paying colossal amount of money for healthcare that usually puts people to poverty.

The industry is limiting the research in the area as a result of unwillingness to share available data for research. This in effect has negative outcome on the improvement of health insurance sector. This study will inform the regulators and industry players in Kenya to formulate policy around availing the medical claims data to researchers for scientific research purposes. Due to sensitivity of the data, the policy should devise a coding system that preserves the security and privacy of the data availed for public research.

5.3: Recommendations

In future works researchers should examine the non-stationary nature of the organized crime in fraudulent claims. Because the nature of fraudulent claims keeps changing with time, researchers should also keep open tabs and add features that are necessary with the changing environment.

We recommend that the researchers should come up with combined hybrid algorithms that can improve F-1 score and reduce false negatives while preserving the precision.

We propose that future studies should focus on cost function for both false positive and false negative. The cost of each mistake should be calculated and examined separately depending on the real world scenarios of the problems at hand. This cost sensitive learning approach should be implemented based on business requirement.

We recommend that the healthcare regulators in conjunction with the healthcare providers to device a public data portal that the researchers can easily access the healthcare data for research purposes by partnering with academic institutions.

Bibliography

AKI. (2013). *Insurance Industry Annual Report.* Nairobi: Association of Kenya Insurers.

AKI. (2017). *Insurance Industry Annual Report.* Nairobi: Association of Kenya Insurers.

Apache. (2018, December 28). *RDD-based API.* Retrieved from Apache: Linear Methods—: https ://spark .apach e.org/docs/lates t/mllib -linear- metho ds.html

BD. (2018, June 23). Automation will cure Kenya's ailing medical insurance. *Business Daily*.

Bekkar, M., Djemaa, H. K., & and Alitouche, T. A. (2013). Evaluation measures for models assessment over imbalanced data sets. *Journal Of Information Engineering and*, 3(10).

Ben Colton. (2015). HOW TO PREVENT AND MANAGE MEDICAL CLAIM DENIALS TO INCREASE REVENUE.

Branting, L. K., Reeder, F., Gold, J., & and Champney, T. (2016). Graph analytics for healthcare fraud risk estimation. *2016 IEEE/ACM International Conference* (pp. 845–851.). IEEE.

Clara, s. (2013). Big Data in bioinformatics and health care informatics. *Conjunction with the IEEE international conference on BigData:*. IEEE.

Dr. Daniel Schlegel, A. R. (2019, january 28). *Cognitive Advantage for the Insurance and Health Insurance industry in Switzerland.* Retrieved from deloitte: https://www2.deloitte.com/ch/en/pages/innovation/articles/cognitive-advantage-for-the-insurance-and-health-insurance-industry-in-switzerland.html

Im Gee, M. B. (2014). The Financial Cost of Healthcare Fraud 2014. *BDO LLP,*.

Jeni, L. A., Cohn, J. F., & and De La Torre. (2013, july 23). Facing imbalanced data recommendations for the use of performance. *In Affective Computing and Intelligent Interaction*, pp. 245–251.

K. Shih, Y. h. (2002). *Not too hot, not too cold: The.* ICML Workshop on Text.

MOH. (2010). *The Kenya household health expenditure and utilisation survey.* Nairobi: Government of Kenya;.

Mohit Kumar, R. G.-S. (2014). Data Mining to Predict and Prevent Errors in Health. *Accenture Technology Labs*, 4-9.

NCHC. (2010). *Health care facts: Costs.* National Coalition on Health Care.

Orayo, J. (2017, october 9). Automation will cure Kenya's ailing medical insurance. *Business Daily*.

Qi Liu, M. V. (2013). Healthcare fraud detection: A survey and a clustering model for incorporating geo-location information. *29th WORLD CONTINUOUS AUDITING AND REPORTING SYMPOSIUM (29WCARS),*, (pp. 3-5). BRISBANE, AUSTRALIA.

R. A. Bauder and T. M. Khoshgoftaar. (2017). Medicare fraud detection using machine learning methods . *2017 16th IEEE International Conference* (pp. 858–865). Florida: IEEE.

Richard A. Bauder, R. C. (2018). Identifying Medicare Provider Fraud with Unsurpervised machine Learning. *2018 IEEE International Conference on Information Reuse and Integration for Data Science*. Florida: IEEE.

WHO. (2010). *Health systems financing: The path to universal coverage.* Geneva.

WHO. (2017). *World Report on Health Policy and Systems Research.* Switzerland: World Health Organisation,.

APPENDIX

BUDGET AND RESOURCES.

Table 3.2 1 Budget and Resources

Item	**Description**	**Usage**	**Cost (Kes.)**
Computer Machine	• At least 8GB RAM • 2.7GHz processor • At least 250GB HDD	• Analysis • Processing	50,000
External Storage	• At least 100GB	• Data Back up	5,000
Python Programming Software	• Software	• Programming Platform	5,000
Tableau	• Software	• Visual Analytics	10,000
Reading materials	• Academic journals • Academic papers • Forums	• Understanding subject matter	20,000
Miscellaneous	• Data collection • Interviews	• Domain expertise	10,000

PROJECT SCHEDULE

Table 3.2 2 Project Schedule

TASK	DURATION	START	FINISH
Data Acquisition	**30 Days**	15th June 2019	16th July 2019
Feature Construction	**5 Days**	17th July 2019	22nd July 2019
Model learning & Selection	**5 Days**	23rd July 2019	28th July2019
Model scoring & performance testing	**20 Days**	28th July 2019	15th August 2019
Report Writing (Methodology and Results)	**30 Days**	16th August 2019	17th Sept 2019
Discussion and Conclusion	**10 Days**	18th Sep 2019	29th Sep 2019
Thesis Defense	**5 Days**	30th Oct 2019	5th Oct 2019

Python Code

```
import numpy as np
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt
# import pandas_profiling as profile
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.model_selection import train_test_split
import pickle
from scipy import stats
from pylab import rcParams
import tensorflow as tf

LABELS = ["Normal", "Fraud"]

#Loading Data sets

Train =pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Train.csv")
Train_Ben_data=pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Train_Benefici
arydata.csv")
Train_IP_data=pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Train_Inpatient
data.csv")
Train_OP_data=pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Train_Outpatien
tdata.csv")
Test =pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Test.csv")
Test_Ben_data =pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Test_Beneficia
rydata.csv")
Test_IP_data =pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Test_Inpatientd
ata.csv")
Test_OP_data =pd.read_csv("D:\Msc\SEM5\DATA\Insurance data\kernel\Test_Outpatient
data.csv")
# print(Train_OP_data.shape)
# print(Train.Provider.value_counts(sort=True, ascending=False).head(2))

# print(outpatient.columns, outpatient.head() )
#checking missing values in datasets
#  print(Train_OP_data.isna().sum())

Train_Ben_data=Train_Ben_data.replace({'ChronicCond_Alzheimer': 2, 'ChronicCond_H
eartfailure': 2, 'ChronicCond_KidneyDisease': 2,
'ChronicCond_Cancer': 2, 'ChronicCond_ObstrPulmonary': 2, 'ChronicCond_Depression
': 2,
```

```
'ChronicCond_Diabetes': 2, 'ChronicCond_IschemicHeart': 2, 'ChronicCond_Osteopora
sis': 2,
'ChronicCond_rheumatoidarthritis': 2, 'ChronicCond_stroke': 2 },0)
Train_Ben_data=Train_Ben_data.replace({'RenalDiseaseIndicator': 'Y'}, 1)
Test_Ben_data=Test_Ben_data.replace({'ChronicCond_Alzheimer': 2, 'ChronicCond_Hea
rtfailure': 2, 'ChronicCond_KidneyDisease': 2,
'ChronicCond_Cancer': 2, 'ChronicCond_ObstrPulmonary': 2, 'ChronicCond_Depression
': 2,
'ChronicCond_Diabetes': 2, 'ChronicCond_IschemicHeart': 2, 'ChronicCond_Osteopora
sis': 2,
'ChronicCond_rheumatoidarthritis': 2, 'ChronicCond_stroke': 2 }, 0)
Test_Ben_data=Test_Ben_data.replace({'RenalDiseaseIndicator': 'Y'}, 1)
# print(Train_Ben_data.head())

# creating age column
Train_Ben_data['DOB'] = pd.to_datetime(Train_Ben_data['DOB'],format = '%Y-%m-%d')
Train_Ben_data['DOD'] = pd.to_datetime(Train_Ben_data['DOD'],format = '%Y-%m-
%d',errors='ignore')
Train_Ben_data['Age'] = round(((Train_Ben_data['DOD'] - Train_Ben_data['DOB']).dt
.days)/365)
Test_Ben_data['DOB'] = pd.to_datetime(Test_Ben_data['DOB'] , format = '%Y-%m-%d')
Test_Ben_data['DOD'] = pd.to_datetime(Test_Ben_data['DOD'],format = '%Y-%m-
%d',errors='ignore')
Test_Ben_data['Age'] = round(((Test_Ben_data['DOD'] - Test_Ben_data['DOB']).dt.da
ys)/365)

Train_Ben_data.Age.fillna(round(((pd.to_datetime('2009-12-01' , format = '%Y-%m-
%d') - Train_Ben_data['DOB']).dt.days)/365),inplace=True)
Test_Ben_data.Age.fillna(round(((pd.to_datetime('2009-12-01' , format = '%Y-%m-
%d') - Test_Ben_data['DOB']).dt.days)/365),inplace=True)

Train_Ben_data.loc[Train_Ben_data.DOD.isna(),'WhetherDead']=0
Train_Ben_data.loc[Train_Ben_data.DOD.notna(),'WhetherDead']=1
Train_Ben_data.loc[:,'WhetherDead'].head(7)


Test_Ben_data.loc[Test_Ben_data.DOD.isna(),'WhetherDead']=0
Test_Ben_data.loc[Test_Ben_data.DOD.notna(),'WhetherDead']=1
Test_Ben_data.loc[:,'WhetherDead'].head(3)
# print(Train_Ben_data.head(2))

## Merge Train_OP_data to Train_Beneficiarydata
Train_PatientDetaildata=pd.merge(Train_OP_data,Train_Ben_data,left_on='BeneID', r
ight_on='BeneID', how='inner' )
```

```
Test_PatientDetaildata=pd.merge(Test_OP_data,Test_Ben_data, left_on='BeneID', rig
ht_on='BeneID', how='inner')

# print(Train_PatientDetaildata.shape)

## Merge Train_PatientDetaildata with fraudelent providers using "Provider" as jo
ining key
Train_PatientDetail_labeldata=pd.merge(Train, Train_PatientDetaildata, on='Provid
er')
Test_PatientDetail_labeldata=pd.merge(Test, Test_PatientDetaildata, on='Provider'
)
# print(Train_PatientDetail_labeldata.shape)

####DESCRIPTIVE STATISTICS
describe=Train_PatientDetail_labeldata.describe()      #overall descriptive stats
class_counts=Train_PatientDetail_labeldata.groupby('PotentialFraud').size()   #cl
ass distribution
correlations=Train_PatientDetail_labeldata.corr(method='pearson')             #co
rrelations btn attributes
skew=Train_PatientDetail_labeldata.skew()                              #skew of univ
ariate distribution
# print(skew)

sns.set_style('white',rc={'figure.figsize':(12,8)})
count_classes = pd.value_counts(Train_PatientDetail_labeldata['PotentialFraud'],
sort = True)
print("Percent Distribution of Potential Fraud class:- \n",count_classes*100/len(
Train_PatientDetail_labeldata))
LABELS = ["Non Fraud", "Fraud"]
#Drawing a barplot
count_classes.plot(kind = 'bar', rot=0,figsize=(10,6))

#Giving titles and labels to the plot
plt.title("Potential Fraud distribution in Aggregated claim transactional data")
plt.xticks(range(2), LABELS)
plt.xlabel("Potential Fraud Class ")
plt.ylabel("Number of PotentialFraud per Class ")

plt.savefig('PotentialFraudDistributionInMergedData')

#PLotting the frequencies of Statewise beneficiaries
count_States = pd.value_counts(Train_PatientDetail_labeldata['State'], sort = Tru
e)
#print("Percent Distribution of Beneficieries per state:- \n",count_States*100/le
n(Train_Beneficiarydata))
```

```
#Drawing a barplot
(count_States*100/len(Train_PatientDetail_labeldata)).plot(kind = 'bar', rot=0,fi
gsize=(16,8),fontsize=12,legend=True)

#Giving titles and labels to the plot

# plt.annotate('Maximum Beneficiaries are from this State', xy=(0.01,8), xytext=(
8, 6.5),
#              arrowprops=dict(facecolor='black', shrink=0.05))

plt.yticks(np.arange(0,10,2), ('0 %','2 %', '4 %', '6 %', '8 %', '10%'))
plt.title("State - wise Beneficiary Distribution",fontsize=18)
plt.xlabel("State Number",fontsize=15)
plt.ylabel("Percentage of Beneficiaries "'%',fontsize=15)
plt.show()

plt.savefig('StateWiseBeneficiaryDistribution')

#PLotting the frequencies of race-wise beneficiaries
count_Race = pd.value_counts(Train_PatientDetail_labeldata['Race'], sort = True)

#Drawing a barplot
(count_Race*100/len(Train_PatientDetail_labeldata)).plot(kind = 'bar', rot=0,figs
ize=(10,6),fontsize=12)

#Giving titles and labels to the plot
plt.yticks(np.arange(0,100,20))#, ('0 %','20 %', '40 %', '60 %', '80 %', '100%'))
plt.title("Race - wise Beneficiary Distribution",fontsize=18)
plt.xlabel("Race Code",fontsize=15)
plt.ylabel("Percentage of Beneficiaries "'%',fontsize=15)

plt.show()

plt.savefig('RacewiseBeneficiaryDistribution')

sns.set(rc={'figure.figsize':(12,8)},style='white')

ax=sns.countplot(x='State',hue='PotentialFraud',data=Train_PatientDetail_labeldat
a
              ,order=Train_PatientDetail_labeldata.Race.value_counts().iloc[:5].i
ndex)

plt.title('Top-5 State invloved in Healthcare Fraud')
```

```
plt.show()

plt.savefig('TopStateinvlovedinHealthcareFraud')

sns.set(rc={'figure.figsize':(12,8)},style='white')

ax=sns.countplot(x='Provider',hue='PotentialFraud', data=Train_PatientDetail_labe
ldata
              ,order=Train_PatientDetail_labeldata.Provider.value_counts().iloc[:
10].index)

plt.title('Top-10 Provider invloved in Healthcare Fraud')

plt.show()

plt.savefig('TopProviderinvlovedinHealthcareFraud')

sns.set(rc={'figure.figsize':(12,8)},style='darkgrid')

ax=sns.countplot(x='OperatingPhysician',hue='PotentialFraud', data=Train_PatientD
etail_labeldata
              ,order=Train_PatientDetail_labeldata.OperatingPhysician.value_count
s().iloc[:10].index)

plt.title('Top-10 OperatingPhysician invloved in Healthcare Fraud')

plt.show()

plt.savefig('TopOperatingPhysicianrinvlovedinHealthcareFraud')

sns.set(rc={'figure.figsize':(10,8)},style='darkgrid')

ax=sns.countplot(x='ClmDiagnosisCode_1',hue='PotentialFraud', data=Train_PatientD
etail_labeldata
              ,order=Train_PatientDetail_labeldata.ClmDiagnosisCode_1.value_count
s().iloc[:10].index)

plt.title('Top-10 ClmDiagnosisCode_1 invloved in Healthcare Fraud')

plt.show()
plt.savefig('TopClmDiagnosisCode_1rinvlovedinHealthcareFraud')

sns.set(rc={'figure.figsize':(10,8)},style='darkgrid')
```

```
ax=sns.countplot(x='OPAnnualDeductibleAmt',hue='PotentialFraud', data=Train_Patie
ntDetail_labeldata
              ,order=Train_PatientDetail_labeldata.OPAnnualDeductibleAmt.value_co
unts().iloc[:10].index)

plt.title('Top-10 OPAnnualDeductibleAmt invloved in Healthcare Fraud')

plt.show()

plt.savefig('TopOPAnnualDeductibleAmtinvlovedinHealthcareFraud')

sns.set(rc={'figure.figsize':(10,8)},style='darkgrid')

ax=sns.countplot(x='Race',hue='PotentialFraud', data=Train_PatientDetail_labeldat
a
              ,order=Train_PatientDetail_labeldata.Race.value_counts().iloc[:5].i
ndex)

plt.title('Top-10 Race invloved in Healthcare Fraud')

plt.show()

plt.savefig('TopRaceinvlovedinHealthcareFraud')

##AVERAGE FEATURES BASED ON GROUPING VARIABLES
###Provider
Train_PatientDetail_labeldata["PerProviderAvg_DeductibleAmtPaid"]=Train_PatientDe
tail_labeldata.groupby('Provider')['DeductibleAmtPaid'].transform('mean')
Train_PatientDetail_labeldata["PerProviderAvg_IPAnnualReimbursementAmt"]=Train_Pa
tientDetail_labeldata.groupby('Provider')['IPAnnualReimbursementAmt'].transform('
mean')
Train_PatientDetail_labeldata["PerProviderAvg_IPAnnualDeductibleAmt"]=Train_Patie
ntDetail_labeldata.groupby('Provider')['IPAnnualDeductibleAmt'].transform('mean')
Train_PatientDetail_labeldata["PerProviderAvg_OPAnnualReimbursementAmt"]=Train_Pa
tientDetail_labeldata.groupby('Provider')['OPAnnualReimbursementAmt'].transform('
mean')
Train_PatientDetail_labeldata["PerProviderAvg_OPAnnualDeductibleAmt"]=Train_Patie
ntDetail_labeldata.groupby('Provider')['OPAnnualDeductibleAmt'].transform('mean')
Train_PatientDetail_labeldata["PerProviderAvg_Age"]=Train_PatientDetail_labeldata
.groupby('Provider')['Age'].transform('mean')
Train_PatientDetail_labeldata["PerProviderAvg_NoOfMonths_PartACov"]=Train_Patient
Detail_labeldata.groupby('Provider')['NoOfMonths_PartACov'].transform('mean')
Train_PatientDetail_labeldata["PerProviderAvg_NoOfMonths_PartBCov"]=Train_Patient
Detail_labeldata.groupby('Provider')['NoOfMonths_PartBCov'].transform('mean')
```

```
Test_PatientDetail_labeldata["PerProviderAvg_DeductibleAmtPaid"]=Test_PatientDeta
il_labeldata.groupby('Provider')['DeductibleAmtPaid'].transform('mean')
Test_PatientDetail_labeldata["PerProviderAvg_IPAnnualReimbursementAmt"]=Test_Pati
entDetail_labeldata.groupby('Provider')['IPAnnualReimbursementAmt'].transform('me
an')
Test_PatientDetail_labeldata["PerProviderAvg_IPAnnualDeductibleAmt"]=Test_Patient
Detail_labeldata.groupby('Provider')['IPAnnualDeductibleAmt'].transform('mean')
Test_PatientDetail_labeldata["PerProviderAvg_OPAnnualReimbursementAmt"]=Test_Pati
entDetail_labeldata.groupby('Provider')['OPAnnualReimbursementAmt'].transform('me
an')
Test_PatientDetail_labeldata["PerProviderAvg_OPAnnualDeductibleAmt"]=Test_Patient
Detail_labeldata.groupby('Provider')['OPAnnualDeductibleAmt'].transform('mean')
Test_PatientDetail_labeldata["PerProviderAvg_Age"]=Test_PatientDetail_labeldata.g
roupby('Provider')['Age'].transform('mean')
Test_PatientDetail_labeldata["PerProviderAvg_NoOfMonths_PartACov"]=Test_PatientDe
tail_labeldata.groupby('Provider')['NoOfMonths_PartACov'].transform('mean')
Test_PatientDetail_labeldata["PerProviderAvg_NoOfMonths_PartBCov"]=Test_PatientDe
tail_labeldata.groupby('Provider')['NoOfMonths_PartBCov'].transform('mean')
# print(Train_PatientDetail_labeldata.shape)

##BeneID
Train_PatientDetail_labeldata["PerBeneIDAvg_DeductibleAmtPaid"]=Train_PatientDeta
il_labeldata.groupby('BeneID')['DeductibleAmtPaid'].transform('mean')
Train_PatientDetail_labeldata["PerBeneIDAvg_IPAnnualReimbursementAmt"]=Train_Pati
entDetail_labeldata.groupby('BeneID')['IPAnnualReimbursementAmt'].transform('mean
')
Train_PatientDetail_labeldata["PerBeneIDAvg_IPAnnualDeductibleAmt"]=Train_Patient
Detail_labeldata.groupby('BeneID')['IPAnnualDeductibleAmt'].transform('mean')
Train_PatientDetail_labeldata["PerBeneIDAvg_OPAnnualReimbursementAmt"]=Train_Pati
entDetail_labeldata.groupby('BeneID')['OPAnnualReimbursementAmt'].transform('mean
')
Train_PatientDetail_labeldata["PerBeneIDAvg_OPAnnualDeductibleAmt"]=Train_Patient
Detail_labeldata.groupby('BeneID')['OPAnnualDeductibleAmt'].transform('mean')

Test_PatientDetail_labeldata["PerBeneIDAvg_DeductibleAmtPaid"]=Test_PatientDetail
_labeldata.groupby('BeneID')['DeductibleAmtPaid'].transform('mean')
Test_PatientDetail_labeldata["PerBeneIDAvg_IPAnnualReimbursementAmt"]=Test_Patien
tDetail_labeldata.groupby('BeneID')['IPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerBeneIDAvg_IPAnnualDeductibleAmt"]=Test_PatientDe
tail_labeldata.groupby('BeneID')['IPAnnualDeductibleAmt'].transform('mean')
Test_PatientDetail_labeldata["PerBeneIDAvg_OPAnnualReimbursementAmt"]=Test_Patien
tDetail_labeldata.groupby('BeneID')['OPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerBeneIDAvg_OPAnnualDeductibleAmt"]=Test_PatientDe
tail_labeldata.groupby('BeneID')['OPAnnualDeductibleAmt'].transform('mean')
# print(Test_PatientDetail_labeldata.shape)
```

```
###Operating Physician
Train_PatientDetail_labeldata["PerOperatingPhysicianAvg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('OperatingPhysician')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerOperatingPhysicianAvg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('OperatingPhysician')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerOperatingPhysicianAvg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('OperatingPhysician')['IPAnnualDeductibleAmt
'].transform('mean')
Train_PatientDetail_labeldata["PerOperatingPhysicianAvg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('OperatingPhysician')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerOperatingPhysicianAvg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('OperatingPhysician')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerOperatingPhysicianAvg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('OperatingPhysician')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerOperatingPhysicianAvg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('OperatingPhysician')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerOperatingPhysicianAvg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('OperatingPhysician')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerOperatingPhysicianAvg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('OperatingPhysician')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerOperatingPhysicianAvg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('OperatingPhysician')['OPAnnualDeductibleAmt']
.transform('mean')

###Attending Physician
Train_PatientDetail_labeldata["PerAttendingPhysicianAvg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('AttendingPhysician')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerAttendingPhysicianAvg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('AttendingPhysician')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerAttendingPhysicianAvg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('AttendingPhysician')['IPAnnualDeductibleAmt
'].transform('mean')
```

```
Train_PatientDetail_labeldata["PerAttendingPhysicianAvg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('AttendingPhysician')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerAttendingPhysicianAvg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('AttendingPhysician')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerAttendingPhysicianAvg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('AttendingPhysician')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerAttendingPhysicianAvg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('AttendingPhysician')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerAttendingPhysicianAvg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('AttendingPhysician')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerAttendingPhysicianAvg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('AttendingPhysician')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerAttendingPhysicianAvg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('AttendingPhysician')['OPAnnualDeductibleAmt']
.transform('mean')
# print(Train_PatientDetail_labeldata.columns)

###ClmAdmitDiagnosisCode
Train_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_DeductibleAmtPaid"]=Tr
ain_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['DeductibleAmtPaid']
.transform('mean')
Train_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_IPAnnualReimbursementA
mt"]=Train_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['IPAnnualReim
bursementAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_IPAnnualDeductibleAmt"
]=Train_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['IPAnnualDeducti
bleAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_OPAnnualReimbursementA
mt"]=Train_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['OPAnnualReim
bursementAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_OPAnnualDeductibleAmt"
]=Train_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['OPAnnualDeducti
bleAmt'].transform('mean')

Test_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_DeductibleAmtPaid"]=Tes
t_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['DeductibleAmtPaid'].t
ransform('mean')
```

```
Test_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_IPAnnualReimbursementAmt"]=Test_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['IPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_IPAnnualDeductibleAmt"]=Test_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['IPAnnualDeductibleAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_OPAnnualReimbursementAmt"]=Test_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['OPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmAdmitDiagnosisCodeAvg_OPAnnualDeductibleAmt"]=Test_PatientDetail_labeldata.groupby('ClmAdmitDiagnosisCode')['OPAnnualDeductibleAmt'].transform('mean')
# print(Train_PatientDetail_labeldata.shape)
# ###ClmProcedureCode_1
Train_PatientDetail_labeldata["PerClmProcedureCode_1Avg_DeductibleAmtPaid"]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['DeductibleAmtPaid'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_1Avg_IPAnnualReimbursementAmt"]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['IPAnnualReimbursementAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_1Avg_IPAnnualDeductibleAmt"]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['IPAnnualDeductibleAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_1Avg_OPAnnualReimbursementAmt"]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['OPAnnualReimbursementAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_1Avg_OPAnnualDeductibleAmt"]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['OPAnnualDeductibleAmt'].transform('mean')

Test_PatientDetail_labeldata["PerClmProcedureCode_1Avg_DeductibleAmtPaid"]=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['DeductibleAmtPaid'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_1Avg_IPAnnualReimbursementAmt"]=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['IPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_1Avg_IPAnnualDeductibleAmt"]=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['IPAnnualDeductibleAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_1Avg_OPAnnualReimbursementAmt"]=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['OPAnnualReimbursementAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_1Avg_OPAnnualDeductibleAmt"]=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_1')['OPAnnualDeductibleAmt'].transform('mean')
```

```
# print(Test_PatientDetail_labeldata.shape)
# ###ClmProcedureCode_2
# ###ClmProcedureCode_3
Train_PatientDetail_labeldata["PerClmProcedureCode_3Avg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_3Avg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_3Avg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['IPAnnualDeductibleAmt
'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_3Avg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmProcedureCode_3Avg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerClmProcedureCode_3Avg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('ClmProcedureCode_3')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_3Avg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_3Avg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_3Avg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmProcedureCode_3Avg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmProcedureCode_3')['OPAnnualDeductibleAmt']
.transform('mean')
# print(Test_PatientDetail_labeldata.shape)
# ###ClmDiagnosisCode_1
Train_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['IPAnnualDeductibleAmt
'].transform('mean')
```

```
Train_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('ClmDiagnosisCode_1')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_1Avg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_1')['OPAnnualDeductibleAmt']
.transform('mean')
# print(Train_PatientDetail_labeldata.shape)
# ###ClmDiagnosisCode_2
Train_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['IPAnnualDeductibleAmt
'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('ClmDiagnosisCode_2')['DeductibleAmtPaid'].transfo
rm('mean')
```

```
Test_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_2Avg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_2')['OPAnnualDeductibleAmt']
.transform('mean')

# ###ClmDiagnosisCode_3
Train_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['IPAnnualDeductibleAmt
'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('ClmDiagnosisCode_3')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_3Avg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_3')['OPAnnualDeductibleAmt']
.transform('mean')
```

```
# ###ClmDiagnosisCode_4
Train_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_DeductibleAmtPaid"]=Train
_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['DeductibleAmtPaid'].trans
form('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_IPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['IPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_IPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['IPAnnualDeductibleAmt
'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_OPAnnualReimbursementAmt"
]=Train_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['OPAnnualReimbursem
entAmt'].transform('mean')
Train_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_OPAnnualDeductibleAmt"]=T
rain_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['OPAnnualDeductibleAmt
'].transform('mean')

Test_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_DeductibleAmtPaid"]=Test_P
atientDetail_labeldata.groupby('ClmDiagnosisCode_4')['DeductibleAmtPaid'].transfo
rm('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_IPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['IPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_IPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['IPAnnualDeductibleAmt']
.transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_OPAnnualReimbursementAmt"]
=Test_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['OPAnnualReimbursemen
tAmt'].transform('mean')
Test_PatientDetail_labeldata["PerClmDiagnosisCode_4Avg_OPAnnualDeductibleAmt"]=Te
st_PatientDetail_labeldata.groupby('ClmDiagnosisCode_4')['OPAnnualDeductibleAmt']
.transform('mean')
# print(Train_PatientDetail_labeldata.shape)
###Combinations of different variables
# Train_PatientDetail_labeldata["ClmCount_Provider"]=Train_PatientDetail_labeldat
a.groupby(['Provider'])['ClaimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID"]=Train_PatientDetail_lab
eldata.groupby(['Provider','BeneID'])['ClaimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_AttendingPhysician"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','AttendingPhysician'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_OtherPhysician"]=Train_PatientDe
tail_labeldata.groupby(['Provider','OtherPhysician'])['ClaimID'].transform('count
')
```

```
Train_PatientDetail_labeldata["ClmCount_Provider_OperatingPhysician"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','OperatingPhysician'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmAdmitDiagnosisCode"]=Train_Pa
tientDetail_labeldata.groupby(['Provider','ClmAdmitDiagnosisCode'])['ClaimID'].tr
ansform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_1"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmProcedureCode_1'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_2"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmProcedureCode_2'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_3"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmProcedureCode_3'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_4"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmProcedureCode_4'])['ClaimID'].transfor
m('count')
# Train_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_5"]=Train_Pat
ientDetail_labeldata.groupby(['Provider','ClmProcedureCode_5'])['ClaimID'].transf
orm('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_1"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_1'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_2"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_2'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_3"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_3'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_4"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_4'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_5"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_5'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_6"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_6'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_7"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_7'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_8"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_8'])['ClaimID'].transfor
m('count')
```

```
Train_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_9"]=Train_Patie
ntDetail_labeldata.groupby(['Provider','ClmDiagnosisCode_9'])['ClaimID'].transfor
m('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician"]=Trai
n_PatientDetail_labeldata.groupby(['Provider','BeneID','AttendingPhysician'])['Cl
aimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_OtherPhysician"]=Train_Pa
tientDetail_labeldata.groupby(['Provider','BeneID','OtherPhysician'])['ClaimID'].
transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician_ClmPro
cedureCode_1"]=Train_PatientDetail_labeldata.groupby(['Provider','BeneID','Attend
ingPhysician','ClmProcedureCode_1'])['ClaimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician_ClmDia
gnosisCode_1"]=Train_PatientDetail_labeldata.groupby(['Provider','BeneID','Attend
ingPhysician','ClmDiagnosisCode_1'])['ClaimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_OperatingPhysician"]=Trai
n_PatientDetail_labeldata.groupby(['Provider','BeneID','OperatingPhysician'])['Cl
aimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmProcedureCode_1"]=Trai
n_PatientDetail_labeldata.groupby(['Provider','BeneID','ClmProcedureCode_1'])['Cl
aimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmDiagnosisCode_1"]=Trai
n_PatientDetail_labeldata.groupby(['Provider','BeneID','ClmDiagnosisCode_1'])['Cl
aimID'].transform('count')
Train_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmDiagnosisCode_1_ClmPro
cedureCode_1"]=Train_PatientDetail_labeldata.groupby(['Provider','BeneID','ClmDia
gnosisCode_1','ClmProcedureCode_1'])['ClaimID'].transform('count')

# Test_PatientDetail_labeldata["ClmCount_Provider"]=Test_PatientDetail_labeldata.
groupby(['Provider'])['ClaimID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID"]=Test_PatientDetail_label
data.groupby(['Provider','BeneID'])['ClaimID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_AttendingPhysician"]=Test_Patient
Detail_labeldata.groupby(['Provider','AttendingPhysician'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_OtherPhysician"]=Test_PatientDeta
il_labeldata.groupby(['Provider','OtherPhysician'])['ClaimID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_OperatingPhysician"]=Test_Patient
Detail_labeldata.groupby(['Provider','OperatingPhysician'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmAdmitDiagnosisCode"]=Test_Pati
entDetail_labeldata.groupby(['Provider','ClmAdmitDiagnosisCode'])['ClaimID'].tran
sform('count')
```

```
Test_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_1"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmProcedureCode_1'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_2"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmProcedureCode_2'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_3"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmProcedureCode_3'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_4"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmProcedureCode_4'])['ClaimID'].transform(
'count')
# Test_PatientDetail_labeldata["ClmCount_Provider_ClmProcedureCode_5"]=Test_Patie
ntDetail_labeldata.groupby(['Provider','ClmProcedureCode_5'])['ClaimID'].transfor
m('count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_1"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_1'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_2"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_2'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_3"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_3'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_4"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_4'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_5"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_5'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_6"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_6'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_7"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_7'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_8"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_8'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_ClmDiagnosisCode_9"]=Test_Patient
Detail_labeldata.groupby(['Provider','ClmDiagnosisCode_9'])['ClaimID'].transform(
'count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician"]=Test_
PatientDetail_labeldata.groupby(['Provider','BeneID','AttendingPhysician'])['Clai
mID'].transform('count')
```

```
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_OtherPhysician"]=Test_Pati
entDetail_labeldata.groupby(['Provider','BeneID','OtherPhysician'])['ClaimID'].tr
ansform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician_ClmProc
edureCode_1"]=Test_PatientDetail_labeldata.groupby(['Provider','BeneID','Attendin
gPhysician','ClmProcedureCode_1'])['ClaimID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_AttendingPhysician_ClmDiag
nosisCode_1"]=Test_PatientDetail_labeldata.groupby(['Provider','BeneID','Attendin
gPhysician','ClmDiagnosisCode_1'])['ClaimID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_OperatingPhysician"]=Test_
PatientDetail_labeldata.groupby(['Provider','BeneID','OperatingPhysician'])['Clai
mID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmProcedureCode_1"]=Test_
PatientDetail_labeldata.groupby(['Provider','BeneID','ClmProcedureCode_1'])['Clai
mID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmDiagnosisCode_1"]=Test_
PatientDetail_labeldata.groupby(['Provider','BeneID','ClmDiagnosisCode_1'])['Clai
mID'].transform('count')
Test_PatientDetail_labeldata["ClmCount_Provider_BeneID_ClmDiagnosisCode_1_ClmProc
edureCode_1"]=Test_PatientDetail_labeldata.groupby(['Provider','BeneID','ClmDiagn
osisCode_1','ClmProcedureCode_1'])['ClaimID'].transform('count')
# print('Test_ProviderWithPatientDetailsdata shape-
',Test_PatientDetail_labeldata.shape)
# print('Train_ProviderWithPatientDetailsdata shape-
',Train_PatientDetail_labeldata.shape)

Train_PatientDetail_labeldata.rename({'PotentialFraud':'ClaimStatus'},inplace=Tru
e)
# print(Train_PatientDetail_labeldata.columns)

###Impute numeric columns with 0
cols1 = Train_PatientDetail_labeldata.select_dtypes([np.number]).columns
cols2 = Train_PatientDetail_labeldata.select_dtypes(exclude = [np.number]).column
s

Train_PatientDetail_labeldata[cols1] = Train_PatientDetail_labeldata[cols1].filln
a(value=0)
Test_PatientDetail_labeldata[cols1]=Test_PatientDetail_labeldata[cols1].fillna(va
lue=0)

## FEATURE SELECTION
###Remove unnecessary columns
cols=Train_PatientDetail_labeldata.columns
cols[:58]
remove_these_columns=['BeneID', 'ClaimStartDt','ClaimEndDt','AttendingPhysician',
```

```
       'OperatingPhysician', 'OtherPhysician', 'ClmDiagnosisCode_1',
       'ClmDiagnosisCode_2', 'ClmDiagnosisCode_3', 'ClmDiagnosisCode_4',
       'ClmDiagnosisCode_5', 'ClmDiagnosisCode_6', 'ClmDiagnosisCode_7',
       'ClmDiagnosisCode_8', 'ClmDiagnosisCode_9', 'ClmDiagnosisCode_10',
       'ClmProcedureCode_1', 'ClmProcedureCode_2', 'ClmProcedureCode_3',
       'ClmProcedureCode_4', 'ClmProcedureCode_5', 'ClmProcedureCode_6',
       'ClmAdmitDiagnosisCode',
        'DOB', 'DOD',
        'State', 'County']
Train_data=Train_PatientDetail_labeldata.drop(axis=1, columns=remove_these_column
s)
Test_data=Test_PatientDetail_labeldata.drop(axis=1,columns=remove_these_columns)
# print('Train_data shape', Train_data.shape)
# print('Test_data shape', Test_data.shape)

##Type conversion
###Convert gender & race to categorical
Train_data.Gender=Train_data.Gender.astype('category')
Test_data.Gender=Test_data.Gender.astype('category')

Train_data.Race=Train_data.Race.astype('category')
Test_data.Race=Test_data.Race.astype('category')

###Dummification
Train_data=pd.get_dummies(Train_data,columns=['Gender','Race'],drop_first=True)
Test_data=pd.get_dummies(Test_data,columns=['Gender','Race'],drop_first=True)

###Convert Target 1(Yes) and 0(No)
Train_data.PotentialFraud.replace(['Yes','No'],['1','0'],inplace=True)
print(Train_data['PotentialFraud'].head())

###Data aggregration to providers level
Train_data_groupedbyProv_PF=Train_data.groupby(['ClaimID','PotentialFraud'],as_in
dex=False).agg('sum')
Test_data_groupedbyProv_PF=Test_data.groupby(['ClaimID'],as_index=False).agg('sum
')
# print(Train_data_groupedbyProv_PF.head())

##Train Validation Split
X=Train_data_groupedbyProv_PF.drop(axis=1,columns=['ClaimID','PotentialFraud'])
y=Train_data_groupedbyProv_PF['PotentialFraud']
# print(X.head())

##Standardization
###Apply StandardScaler() and transform values to its z form (-3 to 3)
```

```
sc=StandardScaler(copy=True, with_mean=False, with_std=False) #normalized (0,1)
# MinMaxScaler
sc.fit(X)
X_std=sc.transform(X)

X_teststd=sc.transform(Test_data_groupedbyProv_PF.iloc[:,1:])
# print(X_std)


# from sklearn.feature_selection import SelectKBest
# # from sklearn.feature_selection import f_classif
# from sklearn.feature_selection import chi2

# bestfeatures = SelectKBest(score_func=chi2, k=10)
# # bestfeatures = f_classif(X, y)
# fit = bestfeatures.fit(X_std, y)

# dfscores = pd.DataFrame(fit.scores_)
# dfcolumns = pd.DataFrame(X_std.columns)

# #concat two dataframes for better visualization
# featureScores = pd.concat([dfcolumns,dfscores],axis=1)
# featureScores.columns = ['Specs','Score']  #naming the dataframe columns
# print(featureScores.nlargest(10,'Score'))  #print 10 best features

from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
fvalue_selector = SelectKBest(f_classif, k=30)
X_kbest = fvalue_selector.fit_transform(X_std, y)


dfscores = pd.DataFrame(fvalue_selector.scores_)
dfcolumns = pd.DataFrame(X.columns)

#concat two dataframes for better visualization
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Specs','Score']  #naming the dataframe columns
# print(featureScores.nlargest(30,'Score'))
##visualize
# dset = pd.DataFrame()
# dset['attr'] = X.columns
# dset['importance'] = fvalue_selector.estimator_.feature_importances_

# dset = dset.sort_values(by='importance', ascending=False)
```

```
# plt.figure(figsize=(16, 14))
# plt.barh(y=dset['attr'], width=dset['importance'], color='#1976D2')
# plt.title('RFECV - Feature Importances', fontsize=20, fontweight='bold', pad=20
)
# plt.xlabel('Importance', fontsize=14, labelpad=20)
# plt.show()

##Split data in train and validation
### 'stratify=y' will make sure equal distribution of yes:no in both train and va
lidation
X_train,X_val,y_train,y_val = train_test_split(X_kbest,y,test_size=0.3,random_sta
te=101,stratify=y,shuffle=True)
# print(X_train)
# print(y_train)
# print(y_val)
from imblearn.combine import SMOTETomek
sm=SMOTETomek()
X_Train,y_Train=sm.fit_sample(X_train, y_train)
# print('X_train without SmoteTomek',X_train.shape)
# print('X_Train with SmoteTomek', X_Train.shape)
# print(y_Train)

##Model Building

###LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegressionCV
log=LogisticRegressionCV(Cs=10, fit_intercept=True, cv=10, dual=False, penalty='l
2', scoring=None, solver='lbfgs', tol=1e-
4, max_iter=10000, class_weight='balanced', n_jobs=None, verbose=0, refit=True, i
ntercept_scaling=1.,random_state=123)
# log = LogisticRegressionCV(cv=10,class_weight='balanced',random_state=123)
log.fit(X_Train,y_Train)

###SVM
# from sklearn.svm import LinearSVC
# log=LinearSVC(penalty='l2', loss='squared_hinge', dual=True, tol=0.0001, C=1.0,
 multi_class='ovr', fit_intercept=True, intercept_scaling=1, class_weight=None, v
erbose=0, random_state=None, max_iter=1000)
# log.fit(X_Train,y_Train)

# ### Lets predict probability of 0 and 1 for X_train and X_val
log_train_pred_probability=log._predict_proba_lr(X_Train)
log_val_pred_probability=log._predict_proba_lr(X_val)
# print(log_train_pred_probability[0:5])
```

```
# ## Lets Set probability Threshold to 0.50

log_train_pred_60=(log._predict_proba_lr(X_Train)[:,1]>0.50).astype(bool)
log_val_pred_60=(log._predict_proba_lr(X_val)[:,1]>0.50).astype(bool)   # set thr
eshold as 0.50
# print(log_train_pred_60[0:5])

##coefficients of LR
pd.DataFrame(zip(X.columns, np.transpose(log.coef_)))[0:5]

# # ROC curve
from sklearn.metrics import roc_curve, auc,precision_recall_curve
fpr, tpr, thresholds = roc_curve(y_val,log._predict_proba_lr(X_val)[:,1])
 #log_val_pred_probability[:,1])
roc_auc = auc(fpr, tpr)

plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (AUC = %0.2f)' % ro
c_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=1, linestyle='--')

for label in range(1,10,1):
    plt.text((10-label)/10,(10-
label)/10,thresholds[label*15],fontdict={'size': 14})

plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()

##Confusion matrix, Accuracy, sensitivity and specificity

from sklearn.metrics import confusion_matrix,accuracy_score,cohen_kappa_score,roc
_auc_score,f1_score,auc

cm0 = confusion_matrix(y_Train, log_train_pred_60,labels=[1,0])
print('Confusion Matrix Train : \n', cm0)

cm1 = confusion_matrix(y_val, log_val_pred_60,labels=[1,0])
print('Confusion Matrix Val: \n', cm1)

total0=sum(sum(cm0))
```

```
total1=sum(sum(cm1))
#####from confusion matrix calculate accuracy
accuracy0=(cm0[0,0]+cm0[1,1])/total0
print ('Accuracy Train: ', accuracy0)

accuracy1=(cm1[0,0]+cm1[1,1])/total1
print ('Accuracy Val: ', accuracy1)
sensitivity0 = cm0[0,0]/(cm0[0,0]+cm0[0,1])
print('Sensitivity Train : ', sensitivity0 )

sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])
print('Sensitivity Val: ', sensitivity1 )


specificity0 = cm0[1,1]/(cm0[1,0]+cm0[1,1])
print('Specificity Train: ', specificity0)

specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])
print('Specificity Val: ', specificity1)

KappaValue=cohen_kappa_score(y_val, log_val_pred_60)
print("Kappa Value :",KappaValue)
AUC=roc_auc_score(y_val, log_val_pred_60)

print("AUC:",AUC)

print("F1-Score Train: ",f1_score(y_Train, log_train_pred_60))

print("F1-Score Val: ",f1_score(y_val, log_val_pred_60))
```