**HEALTHCARE PROVIDER FRAUD DETECTION ANALYSIS**

**Project By: Siddharth Satyakam (50293092,** [**s59@buffalo.edu**](mailto:s59@buffalo.edu)**)**

**Milestone 2:**

**Problem Statement**

The goal of this project is to " predict the potentially fraudulent providers " based on the claims filed by them along with this, we will also discover important variables helpful in detecting the behavior of potentially fraud providers. and we will study fraudulent patterns in the provider's claims to understand the future behavior of providers.

By using the dataset where we are having the sets of Beneficiary Data, Inpatient data and Outpatient Data and finally the tags relating to each beneficiary Id whether its potentially fraud or a valid case, we will focus on defining the model.

**DATA:**

We would be using the data from the source [https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis#Train\_Beneficiarydata-1542865627584.csv](https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis%23Train_Beneficiarydata-1542865627584.csv). This source has the data in the format:

* Train\_Beneficiarydata-1542865627584: The beneficiary data that would be used for training the model.
* Train\_Inpatientdata-1542865627584: The in Patient data those covered under the Medicare Part B cover
* Train\_Outpatientdata-1542865627584: The Patient data covered under the Medicare Part A cover.
* Train-1542865627584: The training tags of potential Frauds and cases.
* Test\_Beneficiarydata-1542969243754: These are the Test Beneficiary Data.
* Test\_Inpatientdata-1542969243754: These are the Test in-patient Data.
* Test\_Outpatientdata-1542969243754: These are the Test outpatient Data.
* Test-1542969243754: Final test tags.

The reason to choose these specific group of data is that this data has a very large group of collection of patients so more data points for better modelling. Moreover, I was not able to find any other collection having such combination of Beneficiary, Inpatient and outpatient data. **Lastly no other dataset in the public domain had the group of potential fraud cases dataset.**

**RESOURCES USED:**

1> Microsoft Excel(for data visualization)

2> Google COLAB PRO(25.51 GB RAM and 64GB ROM Storage)

**Method of Approach**

The procedure we will be following to reach to the goal is:

1. Data Cleaning and Discretization for analysis and insights.

2. Data Transformation (Removal of outliers, correlation within variables and definition of new variables etc.).

3. Data Standardization (Bringing all the data on the same scale for better model representation)

4. Data Balancing (Usually the problem that we would be facing in the fraud type of cases are there are very few actual cases for use in the data training as a result there a case of minority +ve cases, so we need to balance the data using various procedures)

5. Modelling of the Data.

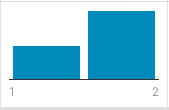
*Data Cleaning and Discretization:*

As the data cleaning was started it was seen that the data source: [https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis#Train\_Beneficiarydata-1542865627584.csv](https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis%23Train_Beneficiarydata-1542865627584.csv)

Had a lot of irregularities that caused very serious delays in all the proceedings. Firstly, the data provided doesn’t mention the country or the state the data belongs to moreover it doesn’t have any descriptions of the columns, what the values represent. As a result, there was no reference for the data cleaning and balancing, which caused very serious delays. Overall the data that we are extracting here are completely based on the assumptions that we had to infer after very long internet browsing.

The assumptions and the changes made for the data are:

1. **Country:** Now as the country was not mentioned in the dataset it was required to infer the country as the other values used in the dataset were also not clear and required country for research. So as the only available data was that the country whose data is provided is having 54 States and 5 races in majority so using the same the closest any nation was able to come close was Africa (54 Countries and 5 races accumulation.)
2. **Date-of-Birth:** Now the DOB column was not formatted correctly and was also having a lot of empty values, so we first removed the empty values then formatted it using the pandas in python 3 in total accumulating a total of 138556 data points remaining.
3. **Date-of-Death:** Here we saw that around 1438 data points in the whole dataset had DOD from the whole of 138556, others were still alive so as a result in case of dead persons the method used for Frauds done by the Providers must be different so we must separate the 2 datasets, for separate analysis. So we separated the dataset into 2 parts work\_dataset\_notDead and work\_dataset\_dead. And also formatted the DOD.
4. **Race and Gender:** In this case we didn’t need to make any changes as all the values were already there though the values were not defined.
5. **RenalDiseaseIndicator:** Here luckily all the +ve cases were marked as “Y” so we could easily convert the tags to respective values.
6. **State and Country:** Now in this case though the values are well defined and discrete but with a lagging index as which value represents which state it was impossible to deduce the respective countries and counties in Africa, hence will cause a lot of problem when we would try to deduce some insights from the given data.
7. **Noofmonths\_PartACov:** As this states the number of months in the cover in the Part A i.e. inpatient cover, as this is well discrete and values are understandable we will not make any changes.
8. **Noofmonths\_PartBCov:** As this states the number of months in the cover in the Part B i.e. outpatient cover, as this is well discrete and values are understandable we will not make any changes.
9. **ChronicCond\_Alzimers:** Now in case of the Alzheimer it was seen that every 1 out of 10 individual faces Alzheimer according to the CDC. So we are considering 1/10 of total data i.e. 138556 which is 13000 so close to it was 42026(label 1) rather than 92530(Label 2) of the data.



So we consider the 1= Yes and 2= NO.

1. **ChronicCond\_Heartfailure:** According to the CDC as the Heart Failure ios a very common disease so here we followed the same principle as the Alzheimer: 1=YES and 2=NO

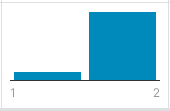


1. **ChronicCond\_KidneyDisease:** In this case, the share of adults 30 and older who have CKD is projected to rise from 13.2 percent today to 16.7 percent in 2030, according to a 2015 report in the American Journal of Kidney Diseases. So basically from the total population that we are having in our datasets of 138556, all are born on or before 1983, so above 30 easily hence our ailing population should be 15.6 to 17.7 percent of the total population of 138556(i.e. 21,000 to 23,000).



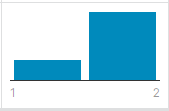
In our case the label 1 is more close to the 15-17% mark of total population than label 2 which is close to 70% of population. SO we would consider the label 1 as Yes and 2 as NO.

1. **ChronicCond\_Cancer**: As per the National Cancer Institute, it is seen that Approximately 38.4% of men and women will be diagnosed with cancer at some point during their lifetimes (based on 2013–2015 data). So basing on it similarly 38.4% of 138556 is 53,205.



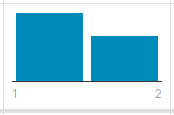
According to the figure the label 1 is more close to 38.4 percent of the data so 1 =YES and 2=NO.

1. **ChronicCond\_ObstrPulmonary:** We see according to survey conducted by Rupert Jones in European Respiratory Journal 2018 51:1702562 it is seen that around 17% people face the issue if lung obstruction in Africa primarily due to inhaling bio mass burnt gases,

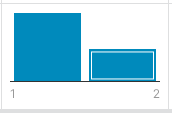


similar trend is seen that Label 1 represents Yes (i.e. 17 percent) and 2 represents NO.

1. **ChronicCond\_Depression:** We maintain the same trend as previous as this is a common disorder in Africa.
2. **ChronicCond\_Diabates**: According to WHO the rate of Diabates is 60% of total population so we would similarly consider the 1=YES and 2=NO

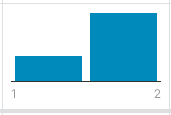


1. **ChronicCond\_ischmeicHeartDisease**: According to the article "Cardiovascular diseases in Tanzania: The burden of modifiable and intermediate risk factors" by Wilfrida P. Roman1, Haikael David Martin1, Elingarami Sauli2 it is mentioned that 81% of the population die due to cadio vascular diseases(30% by Ischemic Heart Disease in Africa) faced by the African subcontinent.



Hence as mentioned in the dataset label 1 represents the best case senario representing the same so 1=Yes and 2=No

1. **ChronicCond\_ Osteoporasis**: One of the most least cases of seen in the African Subcontinent ,hence 1= YES and 2= NO follows here.



1. **ChroniCond\_ Stroke and ChroniCond\_ rheumatoidarthritis:** As all the other columns are following the principle of 1=YES and 2-=NO, we will assume the same here\

All other columns from the Train\_Beneficiarydata-1542865627584 are maintaining the well discrete and distinct values in the various cover columns of monetary covers(**IPAnnualReimbursementAmt, IPAnnualDeductibleAmt, OPAnnualReimbursementAmt, OPAnnualDeductibleAmt.**) so we are not making any changes to the same.

**Age:** As we see that we are having the Date-Of-Birth of the respective beneficiaries and in some cases we also have the Date-Of-Death, we can calculate the age of the beneficiary, so as in order to get an idea or insight as how if the providers are using people of any particular age groups for committing fraud claims. And hence we have a new column called the **Age.**

Finally as we are seeing that the there are some cases where the fore mentioned beneficiaries in the Beneficiary data are having an additional column DOD, in the cases that the person are dead individuals. But are considerably very small in number (i.e. 63968-63394= 574), so are not helping in the model but only adding a new column i.e. DOD; so we have considered the 574 beneficiaries as outliers and are separated and have been deleted, for better modelling.

Next we will consider the data cleaning of the new dataset the more specific features of the Beneficiaries admitted the healthcare institutions/ hospitals, for recognizing the cases where the amount of monetary losses incurred are very large

the data source:

[https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis# Train\_Inpatientdata-1542865627584.csv]( https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis# Train_Inpatientdata-1542865627584.csv )

Now here from the previous dataset we already have the data about the basic conditions of our alive beneficiaries (Age, State, County, Diseases, Insurance\_Details etc.). Now we will connect these given basic data with their corresponding inpatient\_data to get a more holistic idea as how the exploiting providers are using a particular patient with given conditions with a fixed number of procedures and diagnosis codes, to successfully exploit the government into filling their pockets.

New Data:

**ClaimID:** As this is a very specific data and are not repeated over the cases, there cannot be any possible pattern that can be used by us or the fraudster providers to deceive the authorities and hence we will remove this column from our analysis.

**ClaimStartDt and ClaimEndDt:** These are the dates when the specific claim from the providers were made. And as there can be no reason why the fraud providers would make all the fraud claims at any particular timeline only or the timeline when the fraud cases were done will give any future clues, plus as large number of fraud cases at any time will only raise more eyebrows and hence increasing the chances of frauds to get caught.

Under these reasoning we have excluded these columns from our analysis datasets.

**Providers:** This particular column is included as this acts as an index to connecting the fraud cases as well as these can give model an idea pf the possible combinations of the Provider with the physicians, the various diagnosis codes, and procedures in pairs that are the red flag regions we will preserve it.

Now unfortunately there are more than 26 providers in our dataset which effectively will get converted to 26 columns across 63968 cases increasing our data by 26x63968 values, which will make the construction of model very time consuming any difficult especially as every model requires the data in numerical format for all the columns. So we have used the***LabelEncoder*** here by first collecting all the unique representations of the providers and then pandas.factorize() them, i.e. each provider now is represented by a specific number label and replaced in the main analysis/training data, thus making it more easier to analyze the data as it is very well discretized and simplified for model construction.

**InscClaimAmtReimbursed:** This is possibly the whole claim amount that has been reimbursed by the government to the providers and a thing to observe is that the mean expenditure in each of the inpatient cases is $ 12,000(***1000 + 3000 + 4000 + 7000 + 8000 + 9000 + 10000 + 11000 + 12000 + 15000 + 21000 + 26000 + 27000)***; which is equal to the minimum whole tuition fees of any student in a standard institution for a semester in United States.

Now this much of dollars being exploited by the fraudsters need to be managed or eliminated. Hence this column is included in the analysis.

**AttendingPhysician, OperatingPhysician and OtherPhysician:** These three fields are very important for our analysis as many other fields used like the ClmAdmitDiagnosisCode(Diagnosis Code for Admitting the patient), DiagnosisGroupCode, and other fields like the various diagnosis codes and procedure codes and provider can be connected using this.

Now the problem here is that in many if the cases here the attending physician was missing and hence generally we cannot admit any person, because the person wants to live in the hospital, in short we need attending or operating physicians. So here we used a more generic method to make the data complete, by using following methods:

* We first searched where there were no attending physicians or other physicians and only operating physicians and then we filled the operating physicians in the other 2 as substitute.
* The next method was we grouped the data basing on the columns:

['BeneID','Provider','DiagnosisGroupCode','AttendingPhysician','InscClaimAmtReimbursed','BeneID','Gender','age','RenalDiseaseIndicator','State','County','ChronicCond\_Alzheimer','ChronicCond\_Heartfailure','ChronicCond\_KidneyDisease','ChronicCond\_Cancer','ChronicCond\_ObstrPulmonary','ChronicCond\_ObstrPulmonary','ChronicCond\_Depression','ChronicCond\_Diabetes','ChronicCond\_IschemicHeart','ChronicCond\_Osteoporasis','ChronicCond\_rheumatoidarthritis','ChronicCond\_rheumatoidarthritis','ChronicCond\_stroke'],And model the data using regression methods of BayesianRidge, linear regression; however the size of the data and many of the features having more than thousands of values categorical, transformed the data into a matrices of sizes (40474x 140000)which with the given **Google Colab(25GB RAM and 100GB Storage)** also could not handle.

* Now under these conditions we had to resort to more rudimentary methods i.e. grouping the data into various segments groups: attendingPhy\_trainset = tr1.groupby(['State','County','Provider','DiagnosisGroupCode'])

attendingPhy\_tr1 = tr2.groupby(['State','County','Provider'])

attendingPhy\_tr2 = tr3.groupby(['State'])

and the corresponding most frequent Physician and then we replaced all the missing Attending Physicians in the data by matching their corresponding groups and filling the thus void Attending Physicians cases.

* Lastly we then collected all the cases where there was no operating physician or other physician and substituted them with the Attending Physician.

**ClmAdmitDiagnosisCode:** These were specific diagnosis codes under which each patient was admitted. This was included in the training data. This luckily didn’t have any missing values as this couldn’t have been deduced with the given features. However, this having more than 1000s of column values would have increased the size of the data after encoding for modelling hence we again used the ***LabelEncoder*** and converted this into a specific labels and substituted the column values with the respective numerical labels.

**DiagnosisGroupCode:** These were specific diagnosis codes under which each patient was generally grouped after final diagnosis. This was included in the training data. This luckily didn’t have any missing values as this couldn’t have been deduced with the given features as the feature before. This also having more than 1000s of column values would have increased the size of the data after encoding for modelling.

Hence we converted the data into numerical by replacing the OTH label in the column to ‘0’, assuming that ‘0’ code represents no diagnosis code, as was not seen in the data.

**AdmissionDt and DischargeDt:** These also representing the timeline didn’t have any specific relevance in pattern of fraud claims, so we introduced a new column called the “**Duration\_of\_stay**”(difference of the 2 columns) and deleted the2 columns.

**DeductibleAmtPaid:** This column represented the amount paid initially by the govt. for admission hence need not be paid again. This was already numerical with missing data, in specific cases where the amount had not been paid so missing data was replaced by the 0.0.

**ClmDiagnosisCode\_1-10:** These are the specific diagnosis codes under which the amount of claims were divided. As specific pairs could give a more insight into the groups of diagnosis codes that would be used by the providers to exploit the govt. So each group needed to be identified. Now to achieve the given task any model does not recognize lists or sets of data.

So to achieve the group combinations we had to convert all the 7000 codes into columns of data and then replace those columns of data with 1, where the beneficiaries were identified with and 0 in code columns where the beneficiaries were not identified with. Thus increasing the size of the data to whopping(64230x7600) matrix, which made it impossible for data modelling with limited 25.51Gb of RAM on Google Colab PRO, so 25 most frequent diagnosis codes were used for columns in the data not 7000.

**ClmProcedureCode\_1-10:** These are the specific procedures codes performed on diagnosis codes under which the amount of claims was divided. Though specific pairs could give a more insight into the groups of procedure codes for fraud cases we didn’t separate them like **ClmDiagCodes\_1-10** as our data is already extremely large for any modelling with the limited infrastructure.

So to achieve the ease of modelling we simply converted the columns into respective columns with “***labelEncoded”*** data.

Finally, we are going to deal with the dataset that has the tags, for recognizing the potential cases where the providers that have provided the care to beneficiaries are classified as Potential Frauds, and hence by connecting this dataset with the main training data created until now we will complete the data for full analysis

The data source:

[https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis#]( https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis# Train_Inpatientdata-1542865627584.csv )

This final data has 2 columns:

**PotentialFrauds:** This is the tag to associate the Providers that have been marked for potential frauds. Here this only had 2 tags “Yes” and “No”; So we very easily casted the Yes = 1(i.e. Potential Frauds) and “No” = 0 (Not Frauds). And then we converted it into int64 format.

**Provider:**

This is the column helping us tag the Providers with the Potential Fraud column. Now like previous cases we here converted the Provider to ***LabelEncoded*** data by using the same tagging methods (or tags) as used in the Provider in the Inpatient Data.

Then we finally merged this dataset with our main training data by Provider as index and thus completing the complete set.

*Data Balancing:*

Now that our data has been curated further modelling and analysis. A very common problem that we see in cases of the classification is data imbalance. In other words, we have data of a particular tag in majority causing the model to be biased with respect to the majority tag.

And hence as the majority cases are more common in the real world and our model gets biased towards them it fails to make the classification when it comes to the minority class types in those cases the accuracy of the models do not prove any relevance in the real world.

Now there are many methods to eliminate the same like Bootstraping, oversampling, undersampling, SMOTE, ensemble with the class division etc. But in all of them either we end up with reinforcement of the minority class or losing of the data. So we here are not going to go for the same.

**Here we will be using the F1-score (Harmonic Mean of Recall and Precision), as the recall will give us how efficiently the model is able to find the frauds and Precision will balance recall with how effectively does the model even identify any case (fraud or not) effectively.**

*Data Standardization:*

Now that our data is created and well modified for being put into a model for results, it is very important that we need to standardize the data to not give us very absurd patterns due to outliers or arbitrarily organized groups, so to deal with the same we will employ the preprocessing. Scaler module of the sklearn.

Now there are various scalers available namely MinMax(), MaxAbs(), Gaussian Mapper, normal Mapper, pca() etc. Now the MinMax() and MaxAbs() generally preferred to arrange the data around the mean and to reduce variance, more specifically used in the cases of sparse data. But in our case the data is not sparse after we have done so much data cleaning and transformations. So we will shift to a scaler that handles the mean and variance using the mean and quantile of the data.

We have used here RobustScaler(). As the name suggests this especially is used where we are not sure about the kind of unrequired effect the outliers or in-organized data will inflict in the model. So we used the same and transformed the data into well balanced matrix and then we split the data into 4 parts X\_train, X\_val, y\_train, y\_val.

We will be using the X\_train and X\_val for the purpose of the training the data and X\_Val and Y\_val for the validation of the data.

*Modelling:*

Now having standardized and cleaned the data we processed for the modelling of the data.

For better assessment of the models we will be using the GridSearchCV() to allow application of hyper-parameters to find the best along with application of Cross Validation to reduce the case of over fitting.

1. Logistic Regression: Here the parameters used were-

param1 = {}

param1['penalty'] = ['l2']

param1['C'] = np.logspace(-5, 5, 20)

param1['solver'] = ['sag','newton-cg','slbfgs']

param1['verbose'] = [1.0,0.5,2.0]

Here the penalty used was l2-norm for better regularization, with C in range -5 to 5 with 20 partitions for C. The cross validation of this model was kept to 10 partitions of the training data, and the scoring of the GridSearchCV() was made to be “AUC\_ROC” that will give us the rating of the model based on the F1 Score.

Output:

The models continue to run as of now.

*SUMMARY:*

Now as the datasets used were very large(974MB-1GB) the model kept running without summarize-able output so in summary, this interim report is submitted to meet the milestone deadline of the project submission, the final results thrown up by the model will be submitted shortly after completion of the respective model runs.

Furthermore, other models to be used for contrasting are SVM, Random Forest, K-means, Bayesian Classifier etc.