8/10/2019

# CLASSIFICATION OF CLAIMS BASED ON POLICY DECLARATIONS AND CLAIM DOCUMENTS

SUMMER INTERNSHIP REPORT

SIDDHARTH SATYAKAM
UNIVERSITY AT BUFFALO (STATE UNIVERSITY OF NEW YORK)

**MENTOR: FRANCIS RODRIGUES**
**EMAIL ID: francis.rodrigues@gmail.com**

# INDEX

# ACKNOWELGEMENT

I would like to thank Mr. Francis Rodrigues, Senior Vice President, HDFC Life for his help, guidance and mentoring in conceiving and completing the project. He was also very helpful in application of the models.

I would also like to acknowledge the help and guidance extended by Mr. Sreejith Sreekumar for initially setting up of all the systems and understanding the project with help of other underwriters and in understanding the practices, insights and contacts in the HDFC company.

I would like to thank Atul Yehram, for his help in acquiring all the data available in the datalabs and also for his help in directing to the actual people and departments who can help in getting the other relevant data.

Lastly I would like to thank all other executives and staffs of the department for their help in completing the project.

## **The Company:**

HDFC Life is a subsidiary company of HDFC Ltd.; which was created as an independent entity to cater to the life insurance of the company's retail and life insurance of the independent customers and govt./private firm's employees as a whole.

*History:*

It was a joint venture between Housing Development Finance Corporation Ltd (HDFC), one of India's leading housing finance institution and Standard Life Aberdeen PLC, leading well known provider of financial savings & investments services in the United Kingdom. (On 14 August 2015 HDFC Ltd. entered into a share sale agreement with Standard Life to sell a 9.00% stake in HDFC Life to the latter. Post this HDFC was holding 61.65% stake in HDFC Life and Standard Life's stake was increased to 35.00%, with rest was held by others.). The current share holding pattern is 51.48% by Indian promoters, 23.02% by foreign promoters, 11.75% by foreign institutions, 6.28% by general public and balance by others.

Presently, it's listed on the BSE with a BSE Code of 540777 and the NSE with an NSE Code of HDFC Life. The current market capitalization stands at Rs 97,454.81 crore.

Being a leading insurance major in a diverse market like India, the vision of the company is to be:

One of the most successful and admired life insurance company, which means that they are the one of the most trusted company, the easiest to deal with, offer the best value for money and set the standards in the industry.

With each aspect of their work they are aiming to achieve:

1> Excellence
2> People Engagement
3> Integrity
4> Customer Centricity
5> Collaboration

*HDFC Life Network:*

HDFC Life has about 414 branches and presence in 980+ cities and towns in India. The company has also established a liaison office in Dubai.

HDFC Life distributes its products through a multi-channel network consisting of Insurance agents, bancassurance partners (HDFC Bank, Saraswat Bank, RBL Bank), Direct channel, Insurance Brokers & Online Insurance Platform.

The partnerships comprise 265 bancassurance partners including NBFCs (Non-Banking Financial Companies), MFIs (Micro Finance Institutions), SFBs (Small Finance Banks), etc. and 39 partnerships within non-traditional ecosystems. The Company is also strengthened by a strong base of financial consultants.

In Fiscal 2012, The Company established a wholly-owned subsidiary, HDFC Pension Management Company Ltd., to operate its pension fund business under the National Pension Scheme (NPS). And in Fiscal 2016, the Company established its first international wholly-owned subsidiary in the UAE, HDFC International Life and Re Company Ltd., to operate its reinsurance business.

Some of their achievements from their date of inception are:

1> India's Best Companies to Work 2017
HDFC Life adjudged one of India's Best Companies to Work for in The India's Best Companies to Work for study conducted by The Economic Times & Great Place to Work Institute. India's Best Companies to Work For - The India's Best Companies to Work for study is conducted by The Economic Times & Great Place To Work® Institute

2> Gold Shield in the ICAI Awards - Annual Report
HDFC Life received the ICAI Awards - Gold Shield for Excellence in Financial Reporting for the Annual Report 2015-16

3> Finnoviti 2016 Award for innovations in BFSI
HDFC Life Cancer Care was awarded the prestigious Finnoviti 2016, an award that salutes the spirit of innovation. Finnoviti 2016 Conference and Awards recognize and reward innovations in BFSI sector and honour the innovators.

4> Indian Insurance Awards 2016 - Best Product
HDFC Life received the Indian Insurance Awards for Best Product Innovation (Life Insurance) by Fintelekt.

5> SAP ACE Awards 2016 - Technology Adoption
HDFC Life won the SAP ACE Awards - Technology Adoption in Banking, Financial Services and Insurance (BFSI).

6> Data Quest Business Technology Awards - Mobility
HDFC Life received the Data Quest Business Technology Awards, Delhi Edition, for excellence in implementation & use of technology for business benefits, in the "Mobility" category.

7> Indian Insurance Awards 2015 - Life Insurance Company of the Year
HDFC Life won 5 awards in the Indian Insurance Awards, including the Life Insurance Company of the Year, awarded by Fintelekt.

8> Frost & Sullivan 2015 – Project Evaluation & Recognition Program
HDFC Life won the Frost & Sullivan 2015 in Project Evaluation & Recognition Program for the project, "Creating a Culture of Continuous Improvement", an insurance case study.

9> Recognized as Celent Model Insurer of Asia for the Year 2013
HDFC Life has been recognized as the Celent Model Insurer of Asia for 2013, the highest honor among the 17 technology initiatives which are selected as Model Insurer Component in the Asia Pacific region. The company has also received two Model Insurer Component awards namely Model Insurer Award in the area of Underwriting and the other in Distribution/New Business.

With such widespread recognition for their work in insurance, marketing and social entrepreneurship and drive to keep getting better, HDFC Life is utilizing application of the data science to predict the behavior of the customer, make internal processes more bottom-line efficient, while making life easier for customers of the organization.

Another primary reason, for the laurels of the HDFC Life in Insurance companies is their claim settlement ratio; as per latest statistics HDFC Life honored 99.03% of the individual claims, which is a benchmark in itself and is appreciated by its growing customer base.

## ONE OF THE BIGGEST CHALLENGES FOR HDFC LIFE:

Like any insurance firm, the settlement of the claims is a major area of operation at HDFC Life. With the growing number of fraud cases and insurance policies bought by furnishing incorrect information; it is a great predicament to be able to identify the customer's likelihood of either filing a genuine claims or fraudulent claims. Making fake claims could be due to being sucked up into the hands of vicious scavengers of criminal cartels, willingness to defraud the insurance provider or other such circumstances. The idea is to reduce the number of cases where the settlement was done incorrectly, as per the terms of the insurance contract.

As per the latest statistics, Insurers have identified at least 80 districts across the country which have made most fraudulent claims over the past decade. They have identified rings that operate with the efficiency of a corporation with well-trained men and women who collect data with the efficiency of a 21st century start-up.

A combination of poor due diligence in writing policies by insurance companies and the organizational efficiencies of criminals in identifying those who are on deathbed and in enlisting doctors to produce fake certificates led to frauds which are estimated to have cost over Rs. 10,000 crore annually to the industry.

Organized fraudsters identify people who are terminally ill and buy insurance on their behalf and share the booty with the family members. There is a nexus between fraudsters, doctors, lawyers and village-level administrators.

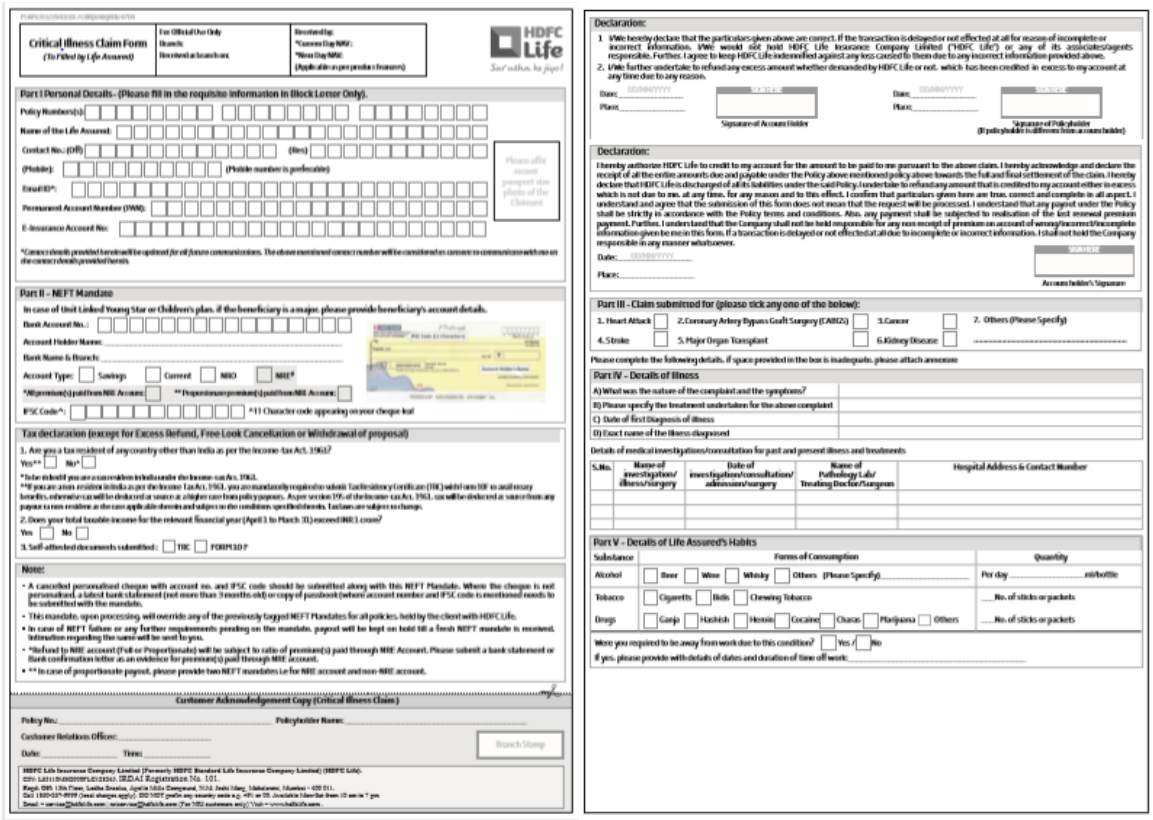
Another major observation that was made in India recently was almost 85-90% of life insurance frauds are in the sum assured bracket of Rs. 1 lakh to Rs. 10 lakh, and the bulk of it is from customers in villages. The industry has been seeing early claims under the insurance scheme Pradhan Mantri Jeevan Jyoti Yojana, where Rs 2-lakh cover is available at a small premium of Rs. 330 a year.

To be able to predict those Fraud cases in HDFC Life would be a boon to the Insurance industry.

## Process Overview of Approval an Insurance Policy in HDFC Life

The whole process of claims starts with an intimation letter or a Claim Intimation form of HDFC Life, through email, post or online page (https:/www.hdfclife.com/customer-service/claims). Each of the Claim Intimation forms basically collects all the preliminary details of the authenticity of the claimant, details of the cause of the claim, NEFT mandate to assure the accounts of the policy holder to be active, the treating doctor and other such required details and declarations made by the claimant.

A sample of the form is shown below:


Critical Illness Claim Form

Part I Personal Details- (Please fill in the requisite information in Block Lorser Only).

Part II - NEFT Mandate

Tax declaration (except for Excess Refund, Free Look Cancellation or Withdrawal of proposal)

Note:

Customer Acknowledgement Copy (Critical Illness Claim)

Declaration:

Declaration:

Part III - Claim submitted for (please tick any one of the below):

1. Heart Attack 2.Coronary Artery Bypass Graft Surgery (CABG's) 3.Cancer 7. Others (Please Specify)
4.Stroke 5.Major Organ Transplant 6.Kidney Disease

Please complete the following details, if space provided in the box is inadequate, please attach annexure

Part IV - Details of Illness

A) What was the nature of the complaint and the symptoms?
B) Please specify the treatment undertaken for the above complaint
C) Date of first Diagnosis of illness
D) Exact name of the illness diagnosed

Details of medical investigations/consultation for past and present illness and treatments

| S.No. | Name of investigation/ illness/surgery | Date of investigation/consultation/ admission/surgery | Name of Pathology Lab/ Treating Doctor/Surgeon | Hospital Address & Contact Number |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

Part V - Details of Life Assured's Habits

| Substance | Forms of Consumption | Quantity |
|---|---|---|
| Alcohol | Beer Wine Whisky Others (Please Specify) | Per day ml/bottle |
| Tobacco | Cigarette Bidi Chewing Tobacco | No. of sticks or packets |
| Drugs | Ganja Hashish Heroin Cocaine Charas Marijuana Others | No. of sticks or packets |

Were you required to be away from work due to this condition? Yes / No
If yes, please provide with details of dates and duration of time off work:

After the intimation letter has been received; a partial verification is done of the form and then a letter is sent to the person nominee of the form requesting all the documents required for the authentication of the case, and further proceedings. This letter is followed by numerous follow-ups in case of non-acknowledgements, and then on receiving the documents the process is followed

and the decision is made accordingly to the letter. In case of any discrepancy in the received document is found then the form is sent for the field verification.

In ideal case where every document completely fulfils the requirements, then the payment is made to the nominee.

In case if some error or ~~hint of~~ false declarations or causes are seen then the cases are denied and the rejection letter with proper reasons are sent to the nominee.

Now the reasons of the denial of the claim are primarily:

**1>** Non fulfillment of the terms and conditions of the Policy.
**2>** Non-Disclosures of facts that are seen in the forms.

Due to such simple procedures, and non-stringent laws upholding the out-payment, the whole system is taken for a ride every now and then; causing the payments to be made to fraudsters, while the rightful claimants go uncovered thus allowing the names of the insurance companies to be tarnished easily.

## PROJECT OBJECTIVE:

Our task at hand is to collect all the data of millions of customers that have insured themselves or their family members in any policy, in HDFC Life till today. And then build a model that is supposed to do two things:

Classify all the cases based on the information of the customer on whose name the insurance is made (like his demographics, financial documents, policy and premium details etc.) and to predict/classify the cases where the situations have been seen to end up as a fraud or wrongful decision by the company.

Presently the situations that each of the cases generally end up are:

a. Settled
b. Settled suspicious
c. Adjourned in the court
d. Rejected
e. Repudiated

And so on. So based on the information provided by the customer we would try to imitate the output of the cases, using our models.

**Data collected and its Problems:**

The data was collected in numerous fields relating to the customer to predict the possible frauds by classifying the data into the Claims; according to the data:

1> Financial Data (All the personal data of the Customer like occupation, monthly income, other income source etc.)
2> Claims Data (All the possible fields that relate to the Claims)
3> Non Claims Data
4> Magnum Data (The initial analysis data of the MAGNUM Software counterpart.)
5> Underwriter Decision Data (Data regarding the underwriter decision, corresponding decision of MAGNUM.)

Now my initial task was to collect the data from the central database of the company.

Here the problem was that as the company is now shifting from the old database, the datalabs (where all the data wrangling and modelling is done), doesn't have access to the complete database of the company. And in case of requirement they have to request the database from the Business Insights (A parallel department that is responsible for helping in business related and capital decisions using the data for same.) by raising a ticket.

The task of data collection started with consultation with the Underwriters (The people who actually make the decisions initially whether to approve the application by a customer to a profitable policy for the company or not), the Data wranglers and some of the business insights counterparts as which are possibly the fields that are going to affect the Claims.

As there was no complete database or a common path for the whole data and sometimes the data being more simplified than required; my first task was to find the data that might be required, and sometimes even go through multiple versions of the same datasets to find the exact requirement for my model.

Starting with the same, with help of some of the old school data wranglers already simplifying data, I had gone through all the databases already with the datalabs, and found the corresponding fields that might be affecting the claims

and the Underwriter decision were noted in each of the distinctions of data i.e. Medical, Financial, Magnum, Claims and Underwriter Decisions.

1. The fields were collected for classifying the **Claims** they were:

| | | | | | |
|---|---|---|---|---|---|
| POLICY NO | CUSTOMER ID | PRODUCT CODE | SUM ASSURED | PREMIUM | API |
| RISK CESSATION TERM | COMMENCEMENT DATE | PREMIUM CESSATION TERM | CHANNEL | SUB CHANNEL | PRODUCT CATEGORY |
| CLAIM CATEGORY | CLAIM STATUS | MEDICAL FLAG | SMOKER AND NON SMOKER | CLUB CHANNEL | BRANCH CITY ZONE |
| BRANCH STATE | BRANCH CITY | BRANCH CITY CLASS | CUSTOMER AGE | CUSTOMER MARITAL STATUS | CUSTOMER STATE |
| CUSTOMER ZONE | CUSTOMER INCOME | BILLING FREQUENCY | INITIATION DATE | RISK CESSATION DATE | PREMIUM CESSATION DATE |
| LAPSE-REINSTATEMENT DATE | RISK CESSATION DATE | PROPOSAL DATE | ISSUANCE DATE | PROPOSAL RECEIVED DATE | RISK CESSATION DATE |

2. Similarly, the fields noted for the **Underwriter Decision** were:

| | | |
|---|---|---|
| POLICY NO | POS_APPLICATION_NO | UNDERWRITER-Decision |
| MAGNUM-Decision | AML-DECISION | APPLICATION-STATUS |

Unlike other fields as the Underwriter Decision are generally not required for other customer behaviors being one of the most crucial task of the datalabs; the same was requested by the Datalabs to the Data Insights department.

3. Then the next data collected was the **financial** Data like the “income”, “Social tier”, “Marital Status”, “Occupation” etc.

4. Another selected dataset was that regarding the date of death, Claim intimation date, action date, ALM Decision, Cause of Claim and Claim Status.

Now having collected the data the next issue was to clean the Data and make it viable for a model.

**Data cleaning and its Problems:**

Now having collected all the required data the next issue was to clean the data; now having collected all the data of customers of HDFC Life till date; it accumulated dataset's with at least 2 91000 to 350000 rows with each spanning at maximum of 30 columns.

With data of this size a machine of 4GB RAM, accommodating machine learning algorithms and other computer services proved to be deficient in all measures, moreover the use of Anaconda during business hours through the Spyder also proved difficult because of the online Kernel being busy throughout the day. On reporting of the same to the higher authorities, and due to intervention of the Senior Vice President, handling all Data related modelling; I was given the access to the Amazon Web services facility called SageMaker, immediately.

With the availability of the AWS Sage Maker, using the notebook instance of 16GB RAM and 64 GB ROM, we were able to import the Data and code the same in the Jupyter Notebook configuration.

Now after importing the data the data cleaning was started with 3 principles:

1. Standardize the Data
2. Remove the Nan/ null values by one of the 3 mechanisms using the
   a. Mean/median.
   b. Algorithms: MissForest, Earhart
   c. Use other standards.
3. And check if the data is equally available for all the classifications.

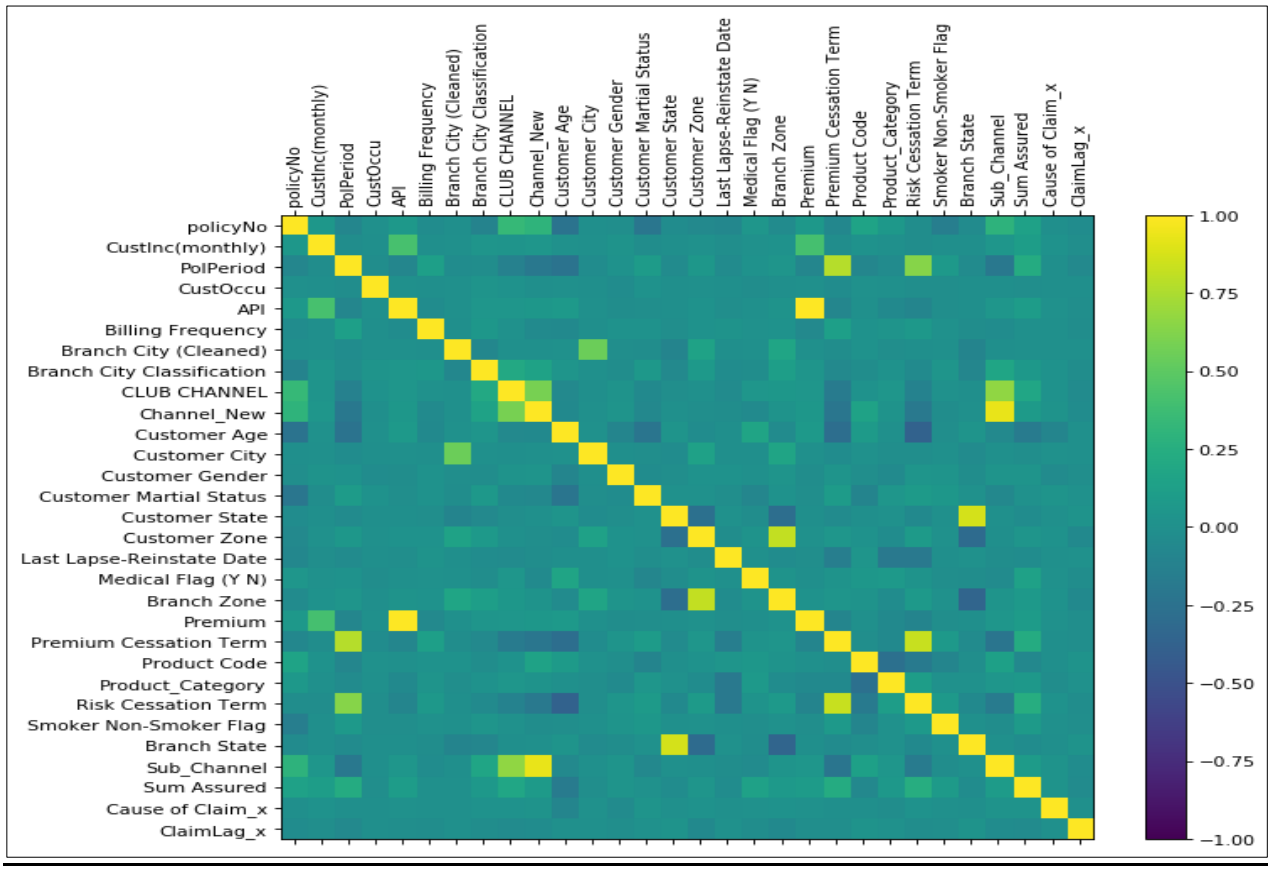
## *AMAZON SAGEMAKER:*

Amazon SageMaker is a cloud machine learning platform that was launched in November 2017.SageMaker enables developers to create, train, and deploy machine learning (ML) models in the cloud.

Amazon SageMaker provides every developer and data scientist with the ability to build, train, and deploy machine learning models quickly. Amazon SageMaker is a fully-managed service that covers the entire machine learning workflow to label and prepare your data, choose an algorithm, train the model, tune and optimize it for deployment, make predictions, and take action. Your models get to production faster with much less effort and lower cost.

SageMaker enables developers to operate at a number of levels of abstraction when training and deploying machine learning models. At its highest level of abstraction, SageMaker provides pre-trained ML models that can be deployed as-is. In addition, SageMaker provides a number of built-in ML algorithms that developers can train on their own data. Further, SageMaker provides managed instances of TensorFlow and Apache MXNet, where developers can create their own ML algorithms from scratch. Regardless of which level of abstraction is used, a developer can connect their SageMaker-enabled ML models to other AWS services for structured data storage, for offline batch processing, for real-time processing.

The deletions of any of the columns were completely based on the heat-map as shown below:



Proceeding with the data cleaning, where the data were standardized as:

- Firstly, the connector of the all the POLICY NO("policyNo") and APPLICATION NO("POS_APPLICATION_NO") were standardized i.e. they were converted to the "INT" type, followed by removal of all the cases where either of the data were missing as each of them are unique they cannot be replaced or joined by any other variable and they cannot be combined with any if the data so they were useless.
- Then the next field "CUSTOMER ID" was removed as the policy No. and POS_APPLICATION_NOS were already acting as an index so there was no requirement for an identifier moreover as approving an application by the ID and not by other factors is not a fair practice and is not followed by HDFC Life so, this column was only acting as a redundant field.
- PRODUCT CODE: Now the product code can shed some light on the possible products that are misused, so this can be a contributing factor to the Fraudulent cases. With the help of as another field that is the Product category the field, the small discrepancies in the data like the comas, and special characters were removed and the field was standardized.
- COMMENCEMENT DATE, INITIATION DATE, PREMIUM CESSATION DATE, RISK CESSATION DATE:
  Now generally the dates cannot be used in general models except for the time series however as my forecast was not regarding the time forecasting but the possible cases of Fraud with given inputs.
  So now to use the date I thought of making a separate column that would act as a representation of the duration called the "POLPERIOD". Now the cells of the same will represent the duration of the policy /or its payment period (difference of the COMMENCEMENT DATE and the PREMIUM CESSATION TERM) in general.

  Now the problem with the data was that all the data were in simple numeric terms (0312019 = 03-01-2019), so on conversion of the same to its datetime64 equivalent was giving garbage data. So to help with the same; my first job was to **convert the data to a string of same size** so that the data can be used for formatting as per requirement of the datetime64 datatype.

So I used the length of the data as my counter and for any data with length 7 was typically the data of type-(0312019) so basically it lagged from other data of length 8; as for months 1-9 (JAN-SEPT); were represented as a single digit, so to make it same I needed to insert another 0 for those 7 length equivalents after the first 2 numbers where the month are only a single digit.

Now after the conversion of the data into a similar type of length 8 strings the next idea was to insert the "-" into the string locations to convert it into the format "XX-XX-XXXX"; using the functions of the string we were able to accomplish the same. Then the dates (COMMENCEMENT DATE and PREMIUM CESSATION DATE) were converted to the corresponding type datetime64 and the difference of the both were recorded in the column "POLPERIOD".

Then having got the duration as the dates were not useful anymore they were deleted i.e. "COMMENCEMENT DATE", "INITIATION DATE", "PREMIUM CESSATION DATE", "RISK CESSATION DATE".

- In case of CHANNEL and SUB-CHANNEL; we saw that there were minimum discrepancies in the data i.e. the data was already in discrete form with representation of each column in a particular format and there were also 1-2 cases where the data was missing where using mean method i.e., what is followed by other customers in general in the location and the values were filled.

- CLAIM CATEGORY: In case of the claim category as we are trying to define the cases of frauds this is also a very significant field. And as one of the direct codependent of the Claim field so all the rows that had the same absent were removed.

- CLAIM STATUS: Now for the cases where the field were absent are useless as this being the output tag, so assuming any of the particular value can completely make the whole dataset misaligned requiring data load balancing using SMOTE (Synthetic Minority Oversampling Technique.), deleting of the already present data of majority but that would have caused the loss of Data which is a strict NO in any case.

OR adding jumbled up columns that will destroy the pattern.

The tags that were primarily used were:
Approved, settled, settled suspicious, Contract Matured, Invalid, Paid-Ex Gratia, Pending, Repudiate, Suspended

- MEDICAL FLAG: Now here the medical flag will act as the representative of all the columns of medical tests so only the ones where the data were noted are generally the cases as assumed by the Underwriters and other data users as the positive cases where they have to pay attention to the case. So only the noted cases were considered positive cases in other cases they can be considered to be N or normal, the dangerous cases are considered to be Y as medical flag raised Yes.

- SMOKER AND NON SMOKER: This also follows the same principle as in the caser of the Medical Flag so it was dealt in the same format as the above.

- LAPSE-REINSTATEMENT DATE: This is a field that represents the date where the customers had extended their policies to extended period. As we cannot consider these dates as these data/dates are never going to be repeated, so here only those cases where the extension was done were replaced by string “Y” and other cases as “N”.

- CUSTOMER CITY: This represents the city of the customer, now there are many cases where the customers having done the insurance in the place of their work which doesn’t exactly replicate the actual city of the customer, so are filled separately, now in the cases where the city is absent we assume the city of the branch where the insurance is made as the city of the customer. So here for filling in the blanks for all the column cells where there was Branch City, customer city was made same as them as a default. As well as there were also many cases where the name of the city was written in mixed cases and sometimes with different spellings which made it very difficult to make the city the same and so it took serious time to find such anomalies in a dataset of 35 lakh rows and make them standardized.

- CUSTOMER STATE: Now here fortunately in the cases where there was missing of the data; we had the corresponding customer city or customer state, so with help of extensive web browsing we were able to access the states of the respective missing data or in cases where there was missing customer city we were able to directly access the data from the branch state and copy the same here. Here we also followed another idea of using the whole previously archived city-state combinations that were available in the branch index of the whole company we used it to map most of the state.

- CUSTOMER ZONE AND BRANCH ZONE: Here for the cases of missing data as it was extremely difficult to go through all the data and fill up the zones based on the state and cities so here, using the already present data of the respective combinations, I drew out a dictionary and mapped all the zones based on the same for both customer and Branch. After that there were some cases where there were still no data as the same combinations were not present in the data, for them the old tedious method was opted of browsing the combination on Internet and adding them into the dictionary constructed earlier and then filling the data. There were some cases in reference to “Dubai” where the data was not properly arranged in terms of the zone of the country there I simply reset all the data to “Dubai”.

- CUSTOMER AGE: Now here we see that the in case of the customer age there were ages filled in correct order except for cases where the age was not filled at all, now age being a tricky feature now a-days with no restrictions on type of insurance being opted for any age, so this could not be mean or missforest; so on further analysis as such cases existed for an 300 cases it was seen as the combination of age, income and occupation when absent at the same time inferred to the Joint Family Insurances so here multiple people in the same insurance may have different occupation ,Income and age so were left blank and as such cases are generally not seen in fraud cases as there are other people in the same insurance to verify the claims so such cases were effectively deleted.

- Branch City: In case of the Branch City, we kept the same in the model

as it can basically be one of the major features; as there can be a great number of fraudulent cases taking place in a region. So this can act as a good predictor. Now here the major issue was there were a large number of null values. To solve the same, we sought of used the mean idea.
Firstly, we saw whether the respective columns had the state or zone of the branch to narrow down our chances. But on the contrary we found the cases to be unique as without the city the other 2 were also not defined. So using the same principle as used when the customer city are absent (Branch city is copied); we modified it slightly for the respective Branch city of the customer; I tried finding the possible cities the customers from the same customer city generally opt for making the insurance in the ideal cases where the complete sequence of Branch and customer detail was there, and from there I assumed the city irrespective of state and zone, with maximum frequency is the city to be Replaced in the branch city absent.
Except from those cases we also modified the cases where there were multiple entries for the same city but with varied cases or for the cases where there spelling mistakes.

- Branch Zone and state: As an extension to the same idea as used in the Branch city, once I had the same using the dictionary that I had created from the Customer city-state (keys=city; values=state) and customer state-zone (keys=state; values=zone) I simply used the same and with minor additions to the dictionaries with the newer branch indices, for some cases I had to recreate the entire the dictionaries for the Branch cases. Using both I simply mapped the State from the cities and the zones from the states respectively.
  And for other null cases the issue was of simple missing values but either having the Branch city or Branch state, the filling of the null values were quite easy, just use the same dictionaries used earlier to fill in the values (map the values).
  And as all the values were previously discrete, we didn't need to make any other changes as these could be easily mapped to the plots, after encoding to a numerical value.

- Customer Occupation: In case of customer occupation there were

initially the cases all the customer Occupation, Customer Age and Customer Salary all were blank these were the specific cases of a Joint insurance as seen during the standardization of the customer age, and they have been already deleted.

In case rest of the cases the customer occupation was initially in following formats: Business, salaried, unsalaried, self-Employed, Housewife, unemployed, service, govt., sports, staff and student. But as these tags were creating an ambiguous condition, like a person in service (not military) can also be seen as a salaried and a person in sports can be seen as a self-employed person; this ambiguity needed to be reduced.

Now to solve this ambiguity, the old archived database was searched extensively and then, we were able to get across the older dataset of the same where these tags already there, were a derived column from the already present well distinguished occupations; so to solve the same we just filtered these older policy No to Occupation data frame and then used the same to map the occupation in the older well distinguished form to the respective policy Nos. and lastly we deleted the older tags in the occupations to reduce the errors.

For other cases where there were still null cases there the customer income was also null; hence in those cases the policies were given without the requirement of these fields, so we simply could simply assume them to be not required in those cases. Hence filled “NotReqd”.

- <u>RISK CESSATION DATE, PROPOSAL DATE, PROPOSAL RECEIVED DATE, ISSUANCE DATE:</u>
  In these case we saw that these dates did not in any way infer to the claim, maybe these can infer to the cancellation or other such customer behavior but these in no way inferred to the claim as can be seen from the Heat Map at the start so we were, so these were deleted.

- <u>CUSTOMER INCOME:</u> In the case of the customer income we saw that the except for the specific cases of the joint family insurances which were deleted while cleaning the customer age. Here the major issue as faced was that the income was not standardized to annual or monthly, or any other time period. So to deal with the same we needed a counter of salary, however as different people opted for different jobs with very

varied pay-scales, it was extremely difficult to be able to extract such index.

So we went for a very conservative approach now the premiums are generally in payment cycles (monthly, yearly, half-yearly, quarterly) but over a year they collect the same premiums for the respective customer. So I first made another small set where the all the premiums using the billing frequency i.e. the payment cycle were made yearly and then I started comparing all the incomes with the same, the cases where the income was less than the yearly premiums they were tagged and converted from the monthly to its yearly equivalent by multiplying with 12. If more i.e. if the incomes were already more than the yearly premiums they were maintained to be the same, i.e. the yearly salary.

Unfortunately, the whole salary calculation was based on the idea that the Customer Income(salary) filled is either monthly or annual and accordingly mapped.

- Branch City Classification: Here we see that the in branch city classifications, this can be a very good index of which are the cases where there is a claim/fraud and can be used to classify the same.

  Here following the old trusted method of mapping these values from the Branch city using another dictionary created from the earlier cases. The new dictionary mapped the city to the zones respectively. So by simply using the dictionary and with slight modifications and additions we mapped the columns correctly.

With these data some more new columns were also added into the mixture from the last dataset as noted in the **Data collected** section above, due to joining of the new data it was seen that the new dataset with the data as mentioned below was having data for approximately 95000 which was very less compared to the 350000 data of claims that had been already cleaned this was because the cause of the claim and other date were missing most of the cases except for the cleaning as mentioned below. So these resulted to reduction of the amount of data for training by a large margin. The filtering of the data was followed by:

- ALM Decision: Though this was a rough indication of the authenticity of the claim, but unfortunately this only had very few columns filled, which as a whole did not make any sense; so unable to fill the null

values as it could not be derived from any of the given columns, it was deleted from the dataset.

- Cause of Claims: Now the cause of the claims was also having a large number of null values, now these values also could not be derived as each Product Category (Health, Pension, Children Investment, Annuity etc.) can have multiple sub categories of claim like Claim for a Health Insurance can be easily Cancer which again has many sub-types like the type of cancer in picture, so assuming anything from the Product category is not yielding any well based result. SO not being able to assume, draw mean and as many insurance claim had very similar values for other columns the Missforest and other algorithms would also not fetch any reasonable results.

  Under these conditions, the only result was to delete the cells with unknown values of the data.

- Action Date: Here as the action date was how long the initiation of any action relating to any Claim is done following the Claim intimation date, so this being completely done inside the company this in no way could be affected by the False Claiming party. So this has no relevance to the false claims. Hence it was also deleted.

- Date of Death and Claim Intimation Date: In these columns we also deleted all the entries where no values were given as they cannot be assumed from the given columns as there are a number of unknown factors.

  Of the rest data as the date will never repeat itself and we were not solving any time forecasting model, we found the difference between the both and added a new column called the "Claim_lag" in the dataset.

- Claim Type: This had the same data as our column Claim Status already standardized in the dataset, so we removed the same, from the dataset.

**Problems that were primarily faced:**

1> Domain knowledge plays a very important role in the data cleaning, especially in the medical data the point that most of the tests have various sub categories and so on makes, it really very difficult to access the data completely.

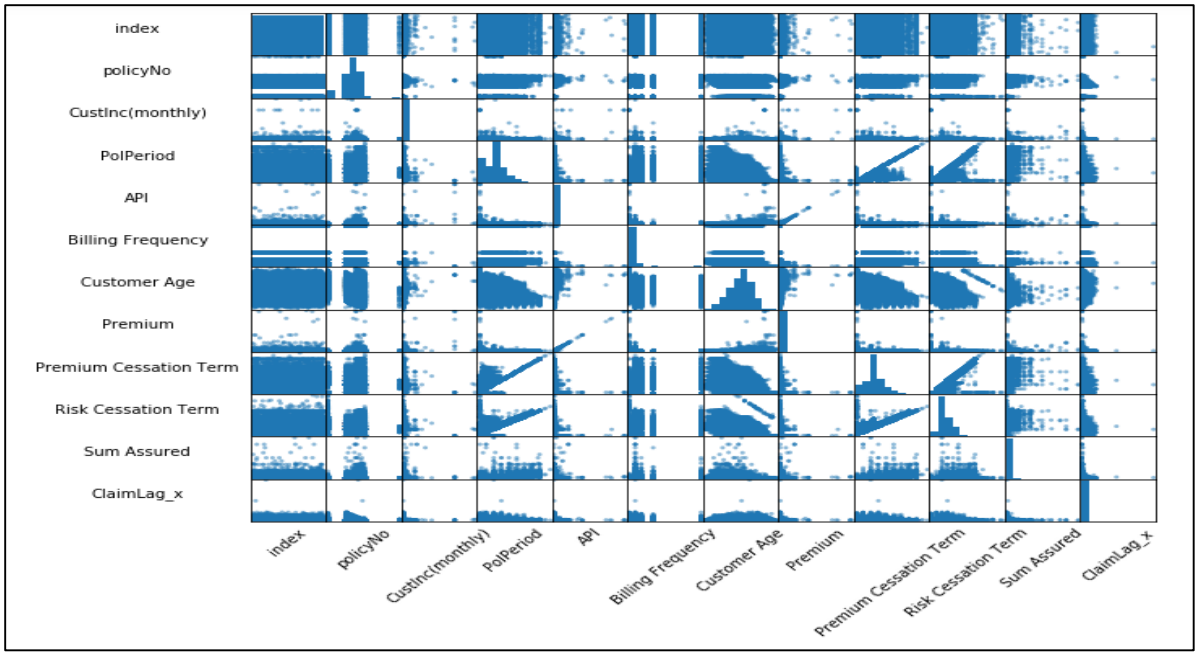
2> Practical application was that the data are stored in unstructured manner i.e. the same data can be present under multiple names and in many different databases which makes it a very important to be cautious about the data to choose and very adventurous to keep trying to find more and more similar data to be able to land with the best state of data.

3> There was no single database for all data and there were cases where data redundancy was seen to outplay the whole model, i.e. there were occasions where after complete data cleaning and modelling newer and more discrete alternatives for the same data were seen which repeatedly compelled to clean the same data in light of the newer data recovered.

4> Moreover, some of the data though existing with the respective departments are not informed inter departmentally, resulting to cases where the data though were found in other departments were not accessible timely.

Now having the data completely ready with all the standardizations completed, and it was seen that from the data heat map that there were no inter-relations between the features in the model so, none of feature could be deleted or required any further feature engineering with respect to each other.

Thus flagging for the start of modelling of the data.




**SCATTER PLOT OF THE PREDICTORS**

## Modeling:

***Starting with the modeling, Claim_Status = Target; Rest = Predictors***

After cleaning of the data, the first approach was to use the PCA (Principal Component analysis)/MCA to reduce the number of predictors or columns, however the same was unrequited as the we only had an accumulative total of 30-35 columns which were generally categorical.

Now for the modeling we opted for the Classification as here most of the data were categorical so they won't exactly give a very vivid pattern as at best they could be only encoded into a specific index, where the values encoded to them won't represent their exact value. The same reason could be seen for the not opting LDA QDA or logistic regression as they work on pattern observed, which I feared would not be represented by the encoded values.

Under the above presumptions I opted for KNN, Random Forest and SVM for the model.

Before diving into the depths of model tuning and cross validations and so on so forth; I had to convert the categorical variables as encoded values.

Now the categorical values, being simple strings rather, than the numerical values face a very rudimentary problem when it comes to the modelling, as the modelling is completely based on the statistics which is drawn on plots, the categorical string variables cannot be directly plotted.

So to avoid the same problem we need to plot the outputs on the graphs to achieve which we opt for the encoding. Now there were 2 options of the same:

1> Label Encoding: These are the encoders where the categorical features are converted to a numerical equivalent based on some index (generally it was range (0, *maxval* (*len* of categorical variable set)). Now all the categorical features can be plotted according to it.

2> One-Hot Encoders: Now on contrary to the logical beliefs that the label encoding solves all the problem of categorical variables as we have

*LABEL ENCODERS:*

*Sklearn provides a very efficient tool for encoding the levels of categorical features into numeric values. LabelEncoder encode labels with a value between 0 and n_classes-1 where n is the number of distinct labels. If a label repeats it assigns the same value to as assigned earlier.*

*ONE-HOT ENCODERS:*

*A one hot encoding is a representation of categorical variables as binary vectors. This first requires that the categorical values be mapped to integer values. Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1.*

encoded the variables to a particular number and it is unique to that vale in the same column, however we are just seeing it from the perspective of the single column ,but when plotting all the features on a single plot the whole concept of the numbers on axes changes ,in most cases or features these numbers are actually representing a value, and not just a tag. And same tag can imply multiple values when many categorical features are taken into consideration.

So here the one hot encoders play a very important role they convert each categorical feature into a binary equivalent (generally derived from

the by splitting a column into a large number of columns), so there is no chance of many features getting mixed up or the vales being messed up.

Now starting with the encoding, all the categorical features were grouped together into a list and then we started with the Label Encoding, this is because the reason that the one-hot encoders find it very difficult to directly convert the values to their corresponding one-hot equivalent from their string representation.

Then the values were converted to one hot equivalent and were stored in the same index.

Then the whole dataset was standardized using the StandardScaler of the preprocessing package of sklearn, and the whole data was split to the predictors and the target.

In short the Claim_Status was separated as the Output(Target) and the rest of the predictors were kept to be the data, with finally approaching to the modelling.

**Application of Models:**

Starting with the modelling, the first approach was to opt for the

**RANDOM FOREST** as it generally gives a very good and user friendly outputs.

To have a better tuned output, I took the help of GRIDSEARCH of the sklearn package, this is very helpful if we want to run the model across multiple hyper-parameters, and want to find the best parameters and algorithm to reach the same. Now using these best parameters, I had run the model and found the best result the random forest can provide model can provide.

```
gdx1= {
  'max_depth': range (3,7),
  'n_estimators': (10, 50, 100, 1000),
  }

# perform grid-search
gsc = gridsearchcv (estimator=randomforestregressor (),param_grid = gdx1,
scoring='neg_mean_squared_error', verbose=0, n_jobs=-1)
grid_result = gsc.fit (x_train, y_train)
best_params = grid_result. best_params_

rfr = randomforestregressor(max_depth=best_params["max_depth"],
n_estimators=best_params["n_estimators"],random_state=false,
verbose=false)
rfr.fit (x_train, y_train)

rfr. score (x_test, y_test)
```

Output result:

```
0.2050284440626485
```

Now this result is only equal to 20%, far from the actual required accuracy.

So to improve the same I opted for the cross validation i.e. retraining the model across multiple sets so choosing the number of cross validation to the minimum of 3-fold cross validation as more number of cross validations are computationally

```
with cross val:
# cross validation measure:
output1 = claims_data_full["claims_status"].copy()
claims_data_full = claims_data_full.drop(columns = {"claims_status"})
claims_data_full = claims_data_full.drop(columns = {"claim type","claim
intimation date", "date of death"}, axis = 1)
claims_data_full = claims_data_full. drop(columns = {"cause of claim",
claimlag"}, axis = 1)

rf_preds=cross_v(estimator = rfr, x = claims_data_full, y = output1)

rf_preds
#sm. accuracy_score (y_test, rf_preds, normalize=true, sample_weight=none)
```

very expensive.

So the result after the cross-validation:

Output result:

```
{'fit_time': array([9.51169586, 9.53314877, 9.58456206]),
 'score_time': array([0.05999899, 0.05969501, 0.06083846]),
 'test_score': array([0.21457208, 0.20197308, 0.20320625]),
 'train_score': array ([0.21287607, 0.2184379, 0.2182952])}
```

So even with the use of the cross validation the accuracy of the model only improved by hardly 1%; giving an output of 21.28%.
So use of Random Forest did not yield any result in our model, and with our data.

**K-Nearest Neighbors** as it doesn't use complex algorithms and is based on simple Euclidean distance:

Here the application of the KNN was somewhat successful, unlike other model the KNN purely relies on the Euclidean distance between the points to see them as same type of elements.

```
#define the model and parameters
knn = kneighborsclassifier()
parameters = {'n_neighbors':[7,10,40,50],
        'leaf_size':[1,3,5,2],
        'algorithm':['auto', 'kd_tree'],
        'n_jobs':[-1]}

#fit the model
modelgc = gridsearchcv(knn, param_grid=parameters,cv=5)
mc1 = modelgc.fit(x_train,y_train)
bestparam = mc1.best_params_

knn.fit(x_train, y_train)
kneighborsclassifier(algorithm=bestparam['algorithm'],
            leaf_size=bestparam['leaf_size'],
            metric='minkowski',
            metric_params=none,
            n_jobs=bestparam['n_jobs'],
            n_neighbors=bestparam['n_neighbors'],
            p=2,
            weights='uniform')
print("predictions form the classifier:")
print(knn.predict(x_test))
print("target values:")
print(y_test)

knn_preds=cross_v(estimator = knn, x = claims_data_full_inonehot, y =
output_inonehot)

#sm.accuracy_score(y_test, knn.predict(x_test), normalize=true,
sample_weight=none)
```

So using the K-Nearest Neighbors it yields an admissible increase in the accuracy of 60 %.

So we tried Cross Validation but we still didn't see any significant growth in the accuracy of our model.

```
In [117],
sm. accuracy_score (y_test, knn. predict(X_test), normalize=True, sample_weight=None)

Out [117]:
0.5924373103338125
```

**Naïve Bayesian Classifier:** This works on the principle of Bayes Probability principle

Here we tried another perspective of the possibility of any probability based relationship, that might exist between the features and the target, as here the all features are considered independent of each other and the conditional probability is found between the features and the target.

As seen in the scatter matrix and Heat-Map (in the Data Cleaning section.), there were no such well-defined relations, so the chances of this was seen to be minute, but under the condition of 60% accuracy, the Naïve Bayesian was also

```
model = gaussiannb()

# fit the model with the training data
model.fit(x_train1,y_train1)

# predict the target on the train dataset
predict_train = model.predict(x_train1)
# accuray score on train dataset
accuracy_train = sm.accuracy_score(y_train1,predict_train,normalize=true)
print('accuracy_score on train dataset : ', accuracy_train)

# predict the target on the test dataset
predict_test = model.predict(x_test1)
# accuracy score on test dataset
accuracy_test = sm.accuracy_score(y_test1,predict_test,normalize=true)
print('accuracy_score on test dataset : ', accuracy_test)
```

tried:
Output Results were:

```
Target on train data [7 1 8 ... 7 1 7]
accuracy_score on train dataset :  0.13373834174724053
Target on test data [1 7 1 ... 4 7 7]
accuracy_score on test dataset :  0.1351221849544801
S,
```

This was even worse than the other 2 models; as the accuracy was only 13%, in both the cases of training and test sets.

**Model Tuning and Accuracy Improvement Measures:**

Now having such reduced efficiency of the model, and the same accuracy or lower observed in every model; were compelled to look back into the data as even after cross validation in each set we were hardly reaching to the efficiency of 55-60%, so there many modifications done later on for better accuracy.

Starting with modifications:

1> The first modification was that the Customer Income was having many values ranging from -1 to 100 which is not a possible monthly salary of any life policy holder, so to represent it as a separate group of people where the income is not required for policy, we replaced all the values with -1.

2> The next modification was on the target variable, we simplified the target as 2 labels True and False, depending on the labels:

- All the labels like “Court Adjourned”, “Invalid”, “Settled-Suspicious”, “Paid Ex-gratia”, “Repudiated” etc. into “FALSE”
- All the labels like “Approved”, “Settled”, “Settle”, “Contract Matured” etc. into “TRUE”

3> Now the major change that was done was it was seen that the data was not balanced i.e. the data had 68000 cases of FALSE while only 8000 cases of True claims that causes the model to be very ill prepared to deal with the False claims.

So to deal with the same I first separated the False claims and then appended them into the same Data-frame ,6 times to reinforce the False claims and also increase the data by oversampling.

4> Lastly I increased the efficiency of the KNN model by using the n-neighbors for the single cluster to 300 by hit and trail.

Due to these changes we were able to get an accuracy of 75%, which is a good increase in the accuracy of any model.

## CONCLUSION:

The aim of the project was to be able to model the data, so as to reduce the false positive (False Claims but the settlement paid). Using the Classification model of KNN we were able to reach the classification up to 75%, which is considerable, due to many reasons like redundancy of data across multiple databases and no firm column available for the distinction of Claims into genuine and False (**A temporary make shift changes were made as shown in the model tuning and changes section above**).

Some of the methods that could lead to improving the model are:

The models can be further improved by adding newer columns to the dataset namely:

1. Claimant
2. Doctors who has approved the death certificate.
3. The Hospital from which the death was certified.
4. Account Number to which the claims sum will be credited
5. Relationship of the claimant to the insured.
6. Claimant contact details like the address, phone number etc.
7. A column that could distinguish between the claims more distinctly

As these can be very similar or same as being used by the insurance manipulators across the India.

With consultation with superior, it was also seen that new model can use the Cost-sensitive learning more commonly used in the case of fraud detection that doesn't exactly point out the false claims but focuses on more effectively filtering the genuine claims by increasing the True Positive cases (Genuine Claims settled) and True Negative Cases (Genuine Claims Not settled).

Or using a multi model portal, basing on classes separated by sum insured i.e. the possible loss to company in case of false claims.

**Resources and Technologies Specifications:**

The resources and technologies that need to be used for the purpose will be:

1. Due to huge volume of the data, we need to use the **Amazon AWS EC2 SageMaker notebook Instance (16GB RAM)** machine learning platform for all the programming.
2. Use of **Python 3.7** for having latest libraries related to the machine learning algorithms for data cleaning, classification and predictive analysis.
3. **Microsoft Excel** for the data visualization and cleaning when the data could not be accessed by the Python due to “NaN” values.