# Detecting Insurance Fraud with Machine Learning

Insurance Fraud has been around since the beginning of insurance organizations. These are unnoticed crimes which cost the insurance industry billions a year. Insurance fraud is a grave problem with all insurance service providers. Even a small fraud can be extremely expensive. As a result, businesses must be careful while analyzing every claim during the insurance claims settlement process. And while it is not possible to verify manually every single claim, so the claim settlement process in insurance can be made easier through Artificial Intelligence.

Manual Investigation: an insurance agent would have the capability to investigate each case and conclude whether it is genuine or not. However, this process is very time-consuming and costly.

AI-based Investigation: We will train the machine to identify whether the case is fraud or not. Many past data have been provided to Machine Learning on which basis AI helps to find the pattern and decide upon the case.

## **Machine Learning: A Big Step in Fraud Detection**

Machine learning is a part of Artificial Intelligence (AI). The idea behind AI is to create a computerized system that can engage in complex analysis and not only replace human input but improve upon it.

Machine learning gives systems the ability to learn and improve from experience, with no extra programming. To do this, the system analyzes large, labelled datasets. Labelled dataset means, with some input, features it also concludes output features. By comparing those inputs to outputs ML finds the pattern and do more complex analysis.



This article contains the following sub-topics

1. Problem Definition.

2. Data Analysis.

3. EDA Concluding Remark.

4. Pre-Processing Pipeline.

5. Building Machine Learning Models.

6. Concluding Remarks.

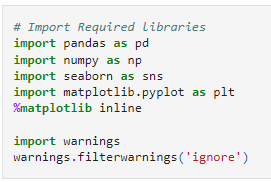
**Problem Definition:**

The goal of this project is to build a model that can detect auto insurance fraud. The challenge behind fraud detection in machine learning is that frauds are less common as compared to legit insurance claims. However, frauds are unethical and are a loss to the company. Building a model that can classify auto insurance fraud which can help to cut losses for the insurance company.

**Data Analysis:**

About the dataset

Data Source: <https://raw.githubusercontent.com/mohittomar2008/Insurance-Claims--Fraud-Detection/main/Automobile_insurance_fraud.csv>



Load the dataset to your environment for analysis. In this case, we are using, Jupyter Notebook.



The inspiration for this project was to perform classification on imbalance class data sets, in particular fraud. (Imbalanced class datasets where one type of data (target) is much more than the other)

The current data set was labelled with n=1000 records.



And 38 different features on which basis it decided about the 39th feature (Target feature). This dataset was found on kaggle.com, however, it is not mentioned if this data is from multiple insurance companies or just one company.

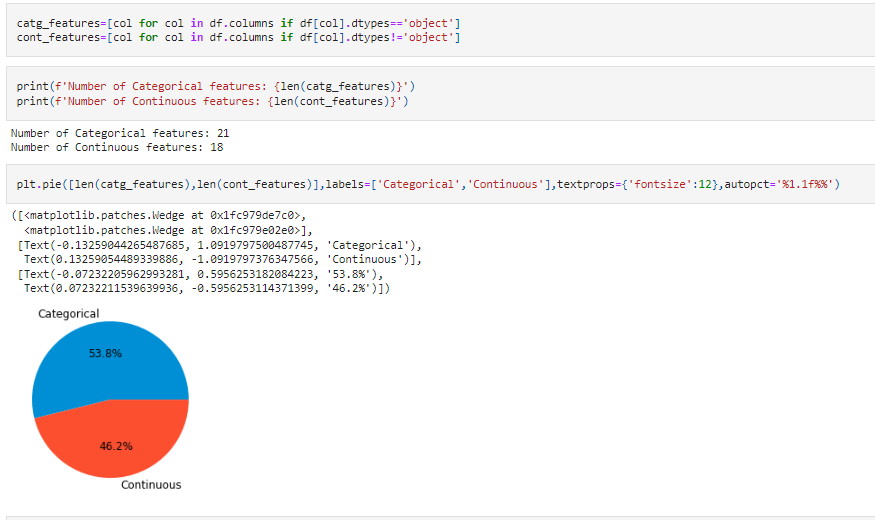
Explore the features on which basis Machine was able to predict about the label.

We have 17 integer features, 21 Object features, 1 float feature

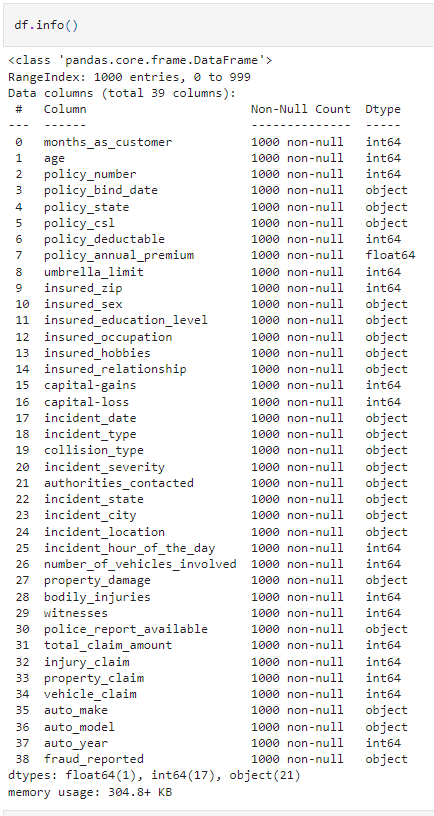
Integer features are those who has values in integer like 1, 2, 3,….

Float features are those who has values in decimal like 1.2, 2.3,…

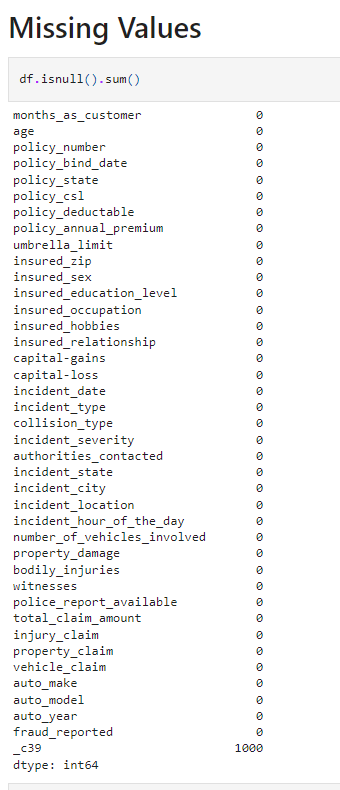
Object features are categorical in nature or in string format like Yes, No



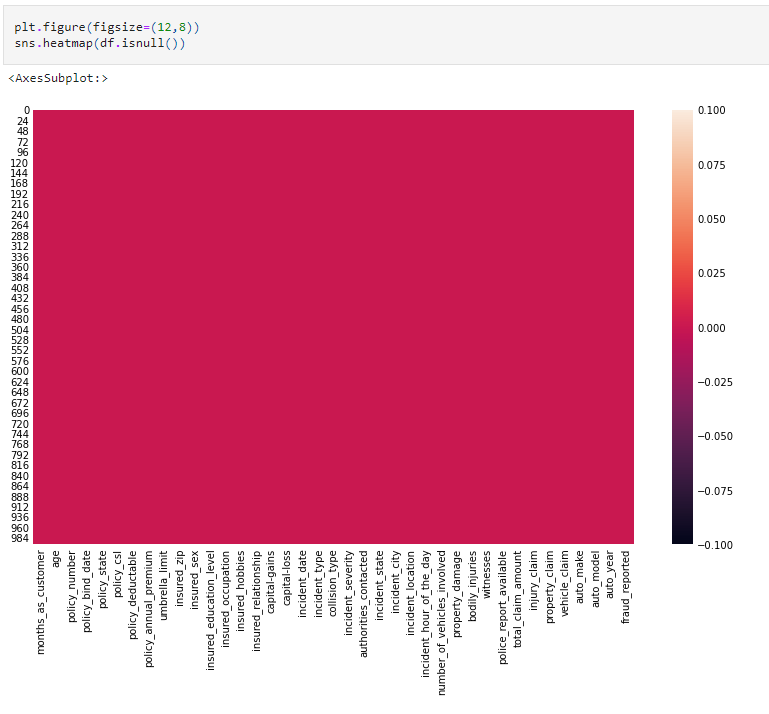
In our dataset, we have 53.8% Categorical features and 46.2% are continuous features.



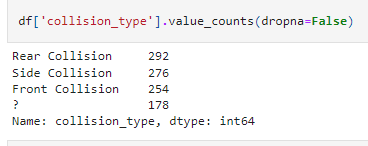
First of all, to know the authenticity of our dataset, we check for the Null values (empty entry)



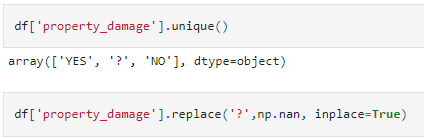
We can also visualize any null values



We don’t have any Null values in our dataset.Upon further investigation, we come to know that this dataset have ‘?’ instead of NaN values.



Check every Object type feature and replace ‘?’ into Nan values so that can be handled easily.



we had to impute those Null values with the meaningful entry by various methods like

Mean, Median, Mode replacement

Random Sample imputation

Capturing NaN values with a new feature

End of distribution imputation

Arbitrary Imputation

Frequent categories imputation

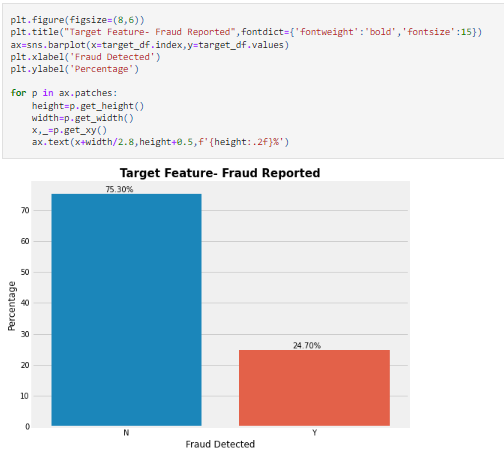
**EDA Concluding Remark.**

Check my GitHub for more detailed EDA

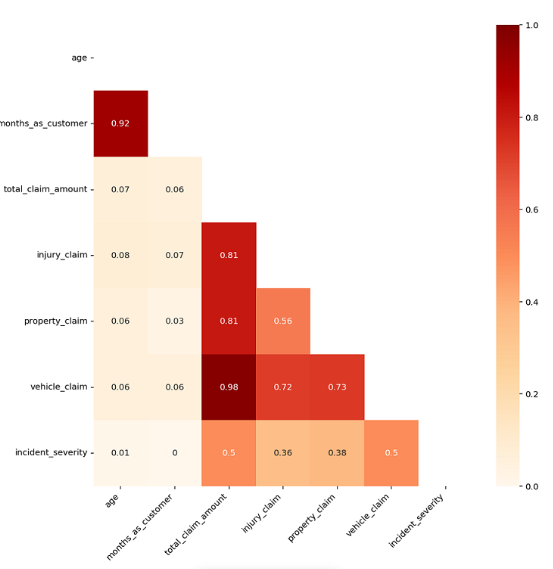
<https://github.com/mohittomar2008/Insurance-Claims--Fraud-Detection>

Dependent Variable (Target Feature)

EDA started with the dependent variable, fraud\_reported. There were 247 frauds detected and 753 were genuine cases. In percentage, 24.7% of the data were with fraud details while 75.3% were with genuine case



Correlation Amount variables:



Correlation is the quantitative method to find the relationship between independent features with Target feature as well within an independent feature. If you are a beginner, let me revise that all the input features are also known as independent features and the target feature is also known as dependent features as it depends on the input variables.

1. Month\_as\_customer and age had.92 correlation
2. Total\_claim\_amount has good correlation with other claim features

Total\_claim\_amount – injury\_claim =.81

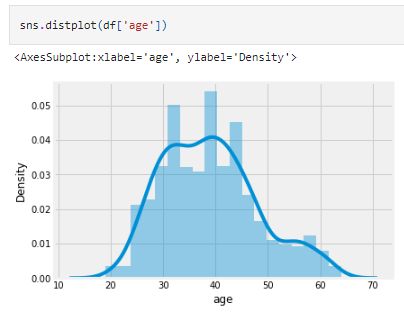
Total\_claim\_amount – property\_claim = .81

Total\_claim\_amount – vehicle\_claim = .98

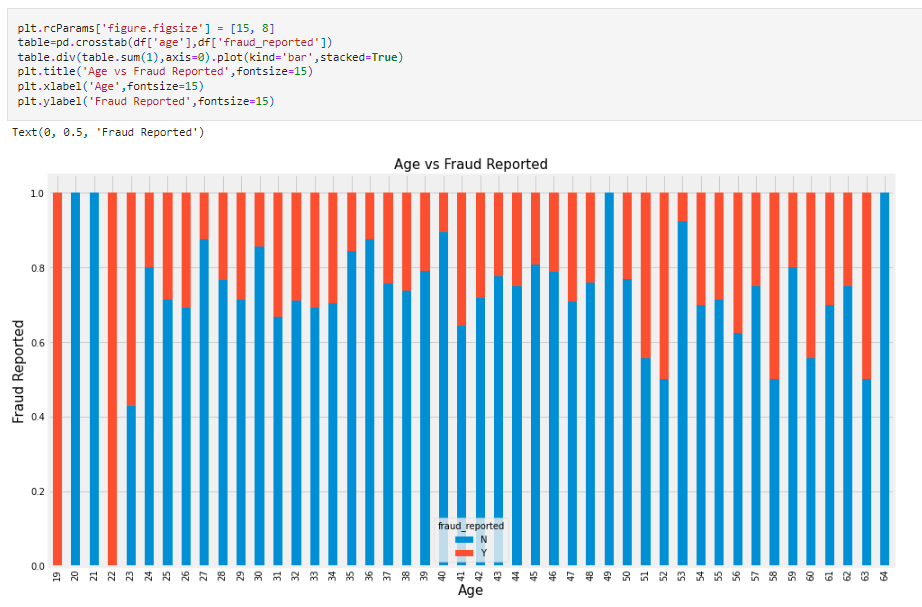
EDA on Independent features:

This will help to understand which features are more responsible for output prediction.

**Age**



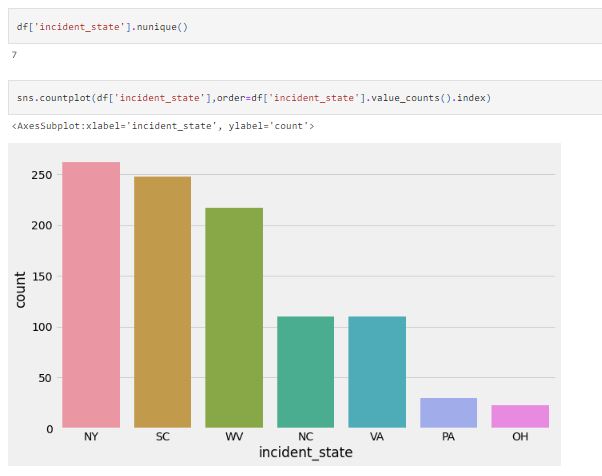
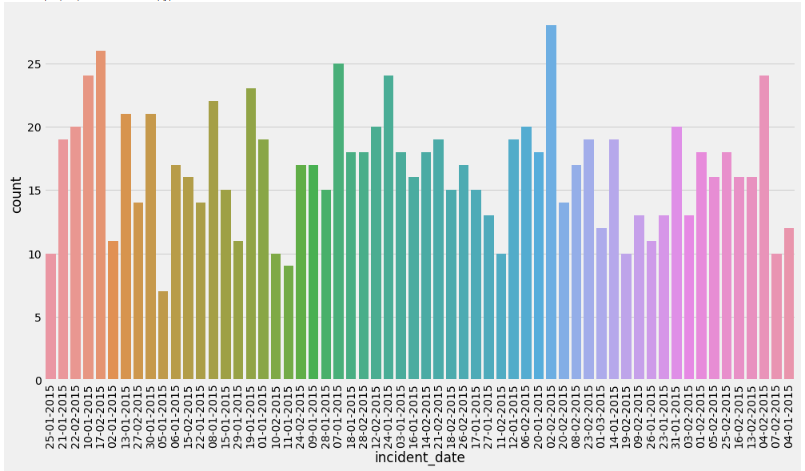
The distribution of Age data is not normally distributed, somehow right-skewed. This is the Age of Insurer.



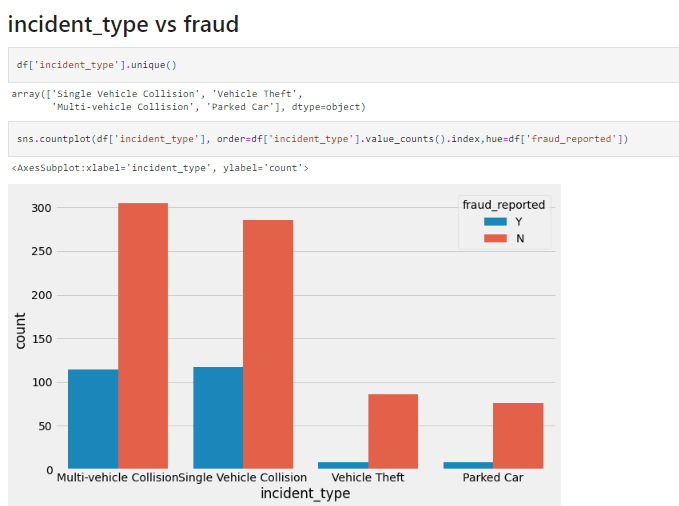
This visualization says that most cases where an insurer is 19 and 22 years old are totally fraud. Age has a very significant impact on target feature.

**Incident\_state**

According to his data, Most of the incident happened in NY state, after SC, WV.

**Date** : For this data records, all incidents happened in January and February of 2015



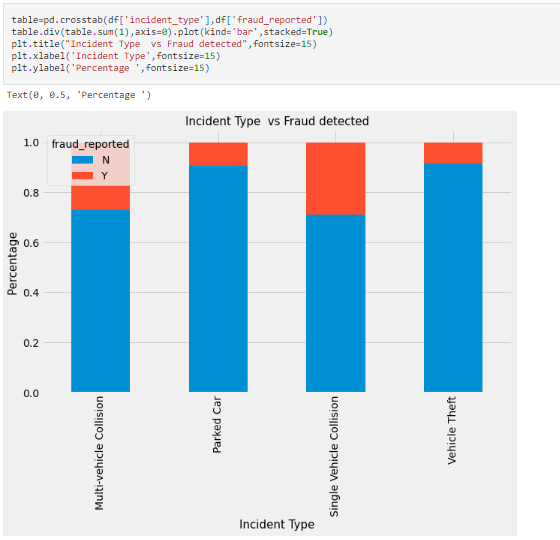
4 types of Incident file has been came for insurance.

Multi-Vehicle Collision (Most cases)

Single-Vehicle Collision

Vehicle-Theft

Parked Car

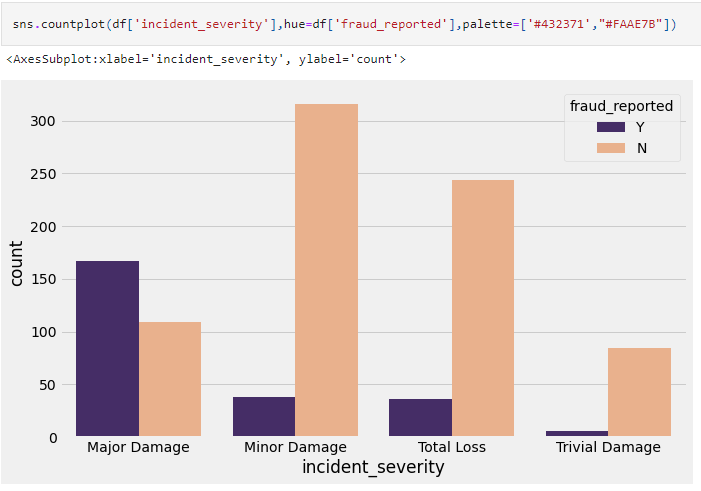


Here, we can notice, most fraud cases were found with incident type multi-Vehicle Collision and Single Vehicle collision.

This could be the case in a parked car, parking in charge has to take care of the parked car along with CCTV security. Fraud cases would be difficult to generate from here.

Vehicle theft involves police FIR, which can lead to serious investigation so fraud cases are less in these incidents.

**Incident Severity vs fraud\_reported:**



Most of the cases came with Minor damage, then total loss and major damage. This visualization shows major damage have more fraud\_cases.

# authorities\_contacted vs fraud\_reported

# As per accident severity, Police has been contacted most of the time after accident. There are equal chances of fraud when the authorities contacted after accident are Police, Fire, Other , Ambulance however, no one contacted in most of genuine cases.



# number\_of\_vehicles\_involved

# As we have investigated already, most of the reported accidents involved 1 or 3 vehicle. So accordingly, fraud cases are more with accident 1 or 3 vehicle.



**Witnesses**

Fraud are more , when witnesses are 1 or 2.



# police\_report\_available

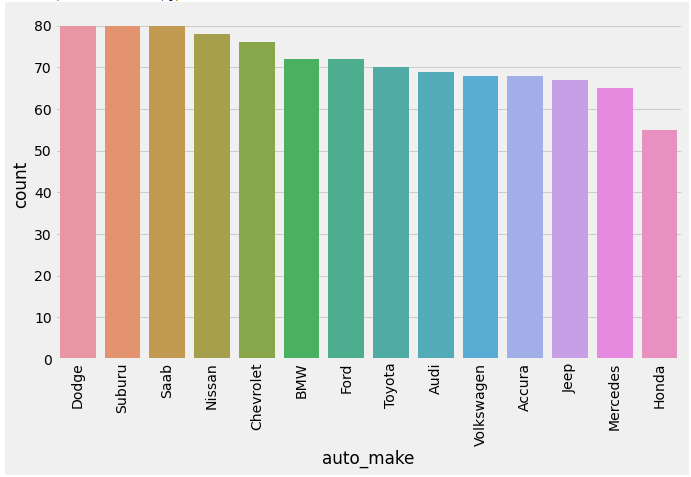


Fraud cases are more when police report is not available. Obviously, guilty person will not contact police for fraud cases.

Let’s check some Vehicle details which was included in accidents.

# auto\_make

This is the Vehicle made Company.



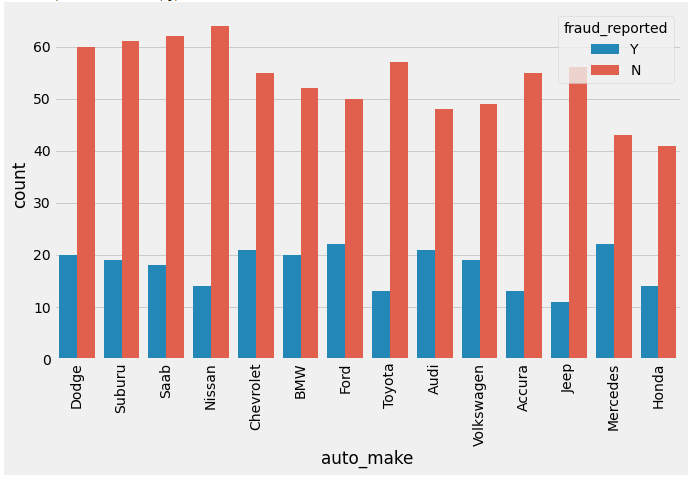
Most accidents happened with Dodge, Suburu and Saab, Nisaan.

Lets check which vehicle was guide in fraud cases.

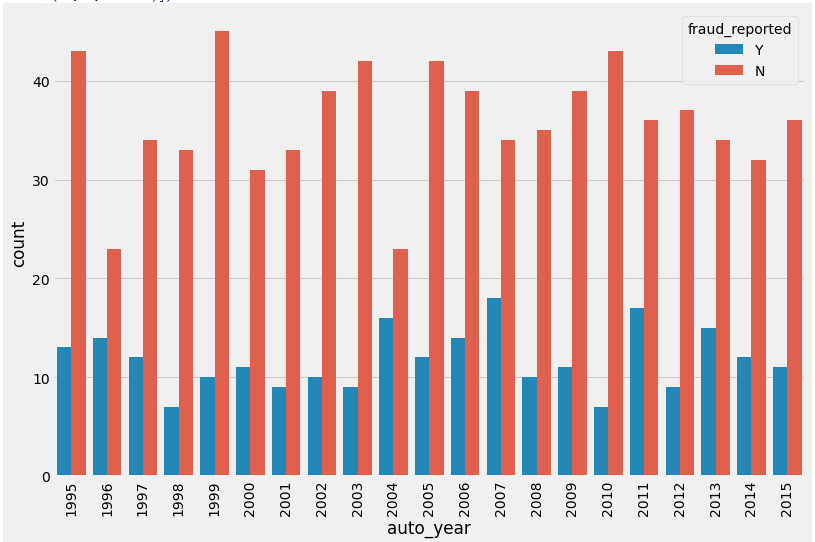
As per below visualization,

Fraud cases were more with Ford car and Mercedes car,

Audi , Chevrolet, BMW, Dodge, Suburu are also had more fraud cases. These are expensive car’s therefore vehicle owner did fraud to claim/bear expenses for their vehicle.

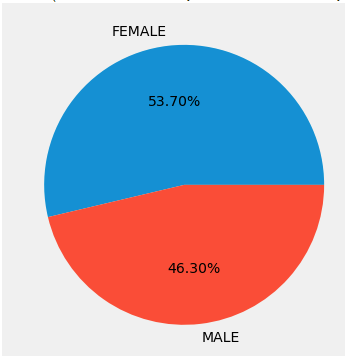


**Auto\_year vs fraud**



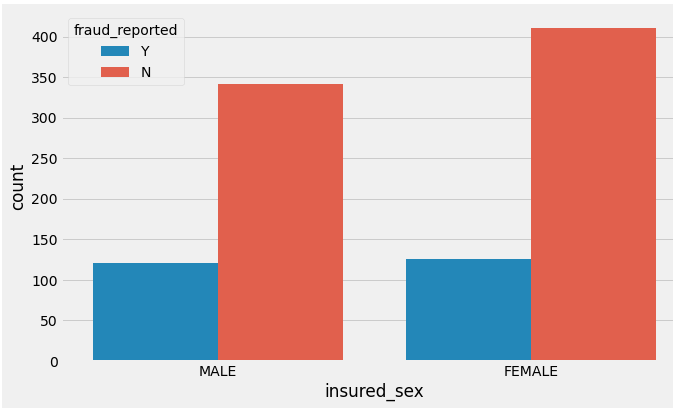
Since 1995 to 2015 registered vehicles data we have among which most fraud cases came from 2004, 2007 and 2011 registered vehicles.

Let’s do some EDA on Insured person:



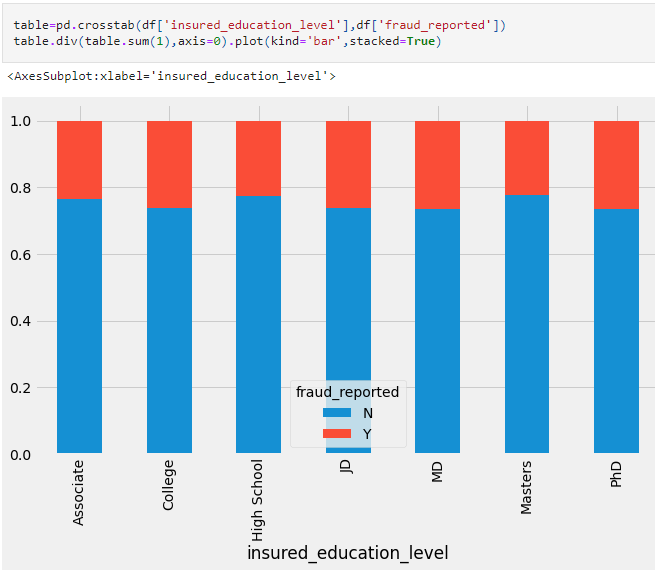
As per our data, we had almost equal cases of Male and female insurer.

53.7% are Male while 46.3% are female insurer.



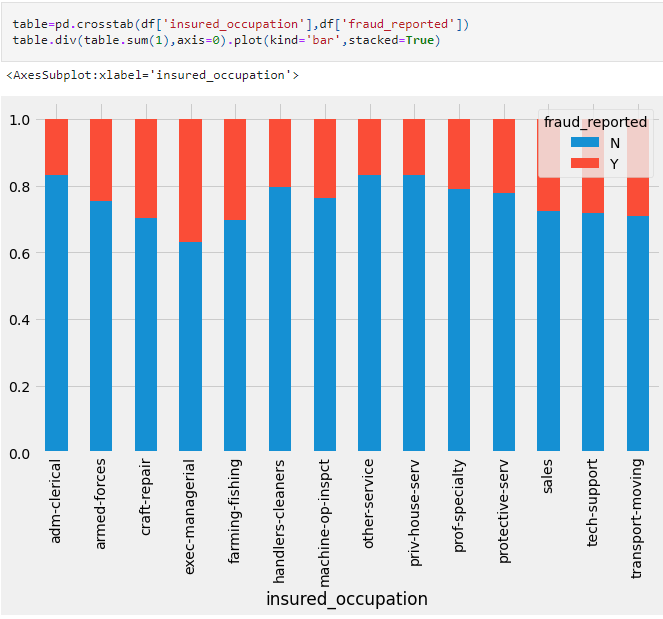
Among which chances are same for fraud case. Means fraud happening doesn’t depends on insurer sex.

**Insured\_education\_level**



Almost chances are same for all educated insured. But specifically, College, JD, MD, PhD have done more frauds

If we go through, Insured occupation. Executive Manager have done most fraud. Insurer involved in the below occupation has done more frauds Craft-repair , farming-fishing, sales, tech-support, transport-moving



# insured\_hobbies vs fraud

# Peoples having hobbies Chess and cross-fit are more crucial for insurance company because in mostly guilt cases insurer had the hobbies like chess and cross-fit.

# Below visualization clears that hobbies can tell the intension and intelligence level of the insurer.

# 

# Policy\_annual\_premium

# Policy annual premium seems to be having Normally distributed data. Little bit skewed on both sides.

# 

# Month\_as\_customer vs Age

# 

# Month\_as\_customer has very good positive correlation with the age of insurer. As the Age increases, months\_as\_customer increases with the company.

# 

# 

# Total\_claim\_amount is highly good correlated with other claims like injury\_claim, property\_claim and vehicle\_claim.

# Upon investigation, we come to know that total\_claim is the total of all 3 other claims. So later on we can drop this total\_claim feature because its information has been covered by other 3 features.

# Distribution of Numerical features can be draw at once. Please visit GitHub for complete python code: [Click Here](https://github.com/mohittomar2008/Insurance-Claims--Fraud-Detection/blob/main/Insurance%20Claims-%20Fraud%20Detection.ipynb)

# 

# Missing Values

# After replacing ‘?’ into Nan values now our 3 categorical features have some missing / NaN values

# Collision\_type – 17.8% missing values

# Property\_damage – 36% missing values

# Police\_report\_available - available – 34.3% missing values

# 

# After analyse, we decided to fill NaN values with most frequent category within feature.

# Wheresoever is null values, fill with Mode. This will fill all of your null values

# 

# Feature Selection

Irrelevant columns:

* policy\_number is not required as it is no help in prediction fraud case
* policy\_bind\_date is not required as we have months\_as\_customer, how old is policy.
* insured\_zip is not required as we have policy\_state and many more details for insured like sex, education, hobby, occupation, relationship

**Feature Engineering**

# Policy\_csl

# 

# We have done feature extraction here from policy\_csl, we have created 2 new features csl\_per\_person and csl\_per\_sccident

# Incident\_hour\_of\_the\_day

# 

# We have converted incident hours into time of the day, like in early morning, morning, afternoon, evening, night

# Auto\_year

# 

# Auto\_year is a important feature, we have extracted the total\_age of vehicle

# Drop the features to avoid garbage information to the ML

# 

# Outliers

# With the help of box plot, we can visualize if any outliers are present in our feature. Outliers are unusual values in data which are far from reality.

# We have separated our available features into category and continuous according to feature types.

# 

# Draw Boxplot with continuous features:

# Step 1: Draw box plot for all continuous datatype features available

# Step 2: check skewness in the features

# Step 3: point out those features having outliers

# Step 4: With IQR method, we can handle outliers in dataset

# 

# Feature Selection – Multi-Collinearity

# Multi-Collinearity is unavoidable issue with data. Machine Learning Algorithms assumes that all independent features are correlated with target variable only. There is no relation between independent features but in reality this is not true. Somehow independent features are also correlated within independents features. Various techniques are available to find that correlation within independent features or feature importance accordingly to some extent we can handle multi collinearity.

# Techniques like:

# VIF

# Constant Features

# Mutual Info Gain

# In this case, we are using VIF (Variance Inflation Factor) to find multicollinearity within our independent features only.

# VIF works for continuous features only, also we will not include target variable because we want to find out the multicollinearity within independent features.

# So, go ahead with continuous features with target feature.

# VIF is 1/(1-R2)

# The Variance Inflation Factor is a measure of collinearity among predictors variables within a multiple regression or

# In simple words, this matric tells you how other variables are explaining your 1 variable. If VIF is large for 1 features means that can be very well explained by other features in your dataset. We don’t require that VIF with large value.

# 

# VIF required Scaled data, so we have used StandardScaler to bring all continuous features to scaled then VIF calculated.

# VIF of total\_claim\_amount is very very large 47858, means this particular features can be explained by other variables and we also know that total\_claim\_amount is the total of all 3 claims. We can drop this features and calculate VIF again for the remaining features.

# 

# Normal accepted values for VIF is -+5, Here, age and months\_as\_customer is more than 6. As pwe our analysis, we know age and month\_as\_customer feature are highly correlated. However for insurance purposes we require the details of how old the customer is so we can drop age features here.

# Again, calculate VIF for remaining features.

# Now, after dropping 2 variables VIF are in range for all features. Along with skewness are also in control.

# Skewness accepted range is -+ .5

# 

# For complete Python file: [Click Here](https://github.com/mohittomar2008/Insurance-Claims--Fraud-Detection/blob/main/Insurance%20Claims-%20Fraud%20Detection.ipynb)

# Transformation and Standardization

# Data transformation is the process of converting raw data into a format or structure that would be more suitable for model building and also data discovery in general.

# Data standardization is the process of rescaling the attributes so that they have mean as 0 and variance as 1

# 

# Power\_transform will transform the data and StandardScaler will scale down the data

# Here, we have fixed the numerical features.

# Encoding

# Machine Learning Algorithms are trained to understand digits only, ML Algos can’t work on strings or categorical data so we have to convert categorical data into numerical.

# Categorical data are of 2 types:

# Nominal 2. Ordinal

# Nominal data can be converted into numerical by get\_dummy method or one-hot encoding while

# Ordinal data have order within feature according to their weightage.

# 

# Remaining categorical are Nominal

# 

# X= independent features

# Y= dependent feature

# Imbalance data- SMOTE

# Our data is imbalance as our target feature have 1 type of data is more than the other. Here if we process imbalanced data to the machine algorithm, it will be learning more for 1 type of data that will create bias in the target prediction.

# 

# As of now, our data have 75.3% cases of Genuine and 24.7% cases of fraud so ML will learn more about genuine cases.

# Various Techniques can be used to balance the data, here we will use SMOTE or oversampling. It will create more records to balance the target feature as per their classification.

# 

# Data is balanced now, we are ready to feed data into the machine algorithm

# Building Machine Learning Models

# Import required libraries for machine Learning

# 

# Train\_test\_split is used to split the complete dataset into the train and test portions. Train data will be used to train the Model then with test data we will compare the accuracy.

# We will check the accuracy of the model through metrics like accuracy\_score,

# confusion\_matrix,

# classification\_report

# f1 score

# 

# Data split into x\_train,x\_test,y\_train ,y\_test

# Modeling

# Five different classifiers were used in this Project:

# -logistics regression

# -Ridge Classifier

# -Decision Tree Classifier

# -SVC

# -K-nearest neighbors

# -Random Forest

# -XGBoost

# -SGD Classifier

# -BaggingClassifier

# -Adaboost classifier

# -Gradient Boosting

# 

# I have taken most of the Algos, so you can find the difference.

# Now, though the below code, fit the data with every ML one by one and calculate accuracy of the model and f1 score

# 

# 

# Cross Validation

# The goal of cross validation is to test the model’s ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or section bias and to give an insight on how the model will generalize to an independent dataset.

# Like an unknown dataset, for instance from a real problem

# 

# From this code, we will get the accuracy score of all models with the train dataset along with cross validation with complete dataset.

# 

# For a generalized model, we select the model with minimum difference between accuracy of train data and the accuracy score of the complete dataset.

# As per our requirement and based on analysis, we will decide the model to go with.

# Here, we are going further with GradientBoostingClassifier

# HyperTuning

# Basically, Models work on defaults parameters, so if we can change the parameters upon our requirement we can also improve the performance of the model

# For hyper tuning, we can use RandomSearchCV and GridSearchCV

# RandomSearchCV will select few parameters combinations from the options while GridSearchCV will try all parameters combinations.

# 

# We have passed 3 parameters options in the dictionary.

# CV=5 , it will cross validate 5 times

# N\_jobs= -1 will use every core for this computation

# Fit with train data

# 

# GCV.best\_estimator\_ and GCV.best\_params provides the best estimator for this cross-validation

# This is not necessary that hyper tuning will always work better, sometimes default parameters provides the best results.

# Default parameters accuracy is 89.15

# Hypertuned accuracy is 88.49

# Other Metrics to evaluate

# Confusion Matrix

# The number of cases for each class of the test set is shown in the confusion matrix below.

# The y-axis shows the actual classes while the x-axis shows the predicted classes. Percentage out of the total sample size of the test set is printed on each quadrant.

# 

# ROC AUC Curve

# The ROC curve below summarizes how well our model is at balancing between the true positive rate (sensitivity) and the false positive rate(1-specificity). Ideally, we want to have a 100% true positive rate of predicting fraud and a 100% true negative rate of predicting non-frauds (or a 0% false-positive which is 100% — 100% true negative rate). This means we have a perfect prediction for both classes. However, in imbalance class problems, this is extremely hard to achieve in the real world. On top of that, there is a trade between the true positive rate and the true negative rate and conversely the false positive rate.

# This graph summarizes how well we can distinguish between two classes at each threshold of the true positive and false positive rate. The area under curve is used as a summary percentage of this metric. In sum, the model has outperformed the baseline ROC AUC scores by a huge margin.

# Although our model performed better in predicting non-fraud cases, the model has performed very well on fraud cases as well. We have a higher false alarm than frauds escaping the detection. It is better in our case to identify more frauds than to let fraud cases escape detection. Thus, this model has succeeded in its purpose to detect fraud claims. Unlike the baseline model that sacrifices too much resources into investigations and hinder customer experience, we are also able to balance this out in this model. We can detect more fraud and we are able to balance this with correct prediction of non-fraud cases

# 

# ROC AUC= 97%

# Concluding Remarks

# This project has built a model that can detect auto insurance fraud. In doing so, the model can reduces loses for insurance companies. The challenge behind fraud detection in machine learning is that frauds are far less common as compared to legit insurance claims.

# In conclusion, the model was able to correctly distinguish between fraud claims and legit claims with high accuracy.

# The Study is not without limitation:

# This study is restricted by its small sample size. Statistical models are more stable when data sets are larger.

# It also generalize better as it takes a bigger portion of the actual population.

# We are also restricted to incidents between 2 months which may not be an accurate picture of the year.

# Further studies may investigate acquiring a larger data set with multiple years. However, due to sensitive nature of fraud and confidential information tagged to such data, this may remain a challenge.