**Insurance Fraud Detection**

horizontal line

# Introduction

**Problem Definition:**

Insurance fraud is a huge problem in the industry. It's difficult to identify fraud claims. Machine Learning is in a unique position to help the Auto Insurance industry with this problem. Machine Learning is a boon in helping the Auto Insurance industry with this problem.

In this project, you are provided a dataset which has the details of the insurance policy along with the customer details. It also has the details of the accident on the basis of which the claims have been made.

In this example, you will be working with some auto insurance data to demonstrate how you can create a predictive model that predicts if an insurance claim is fraudulent or not.

**Executive Summary:**

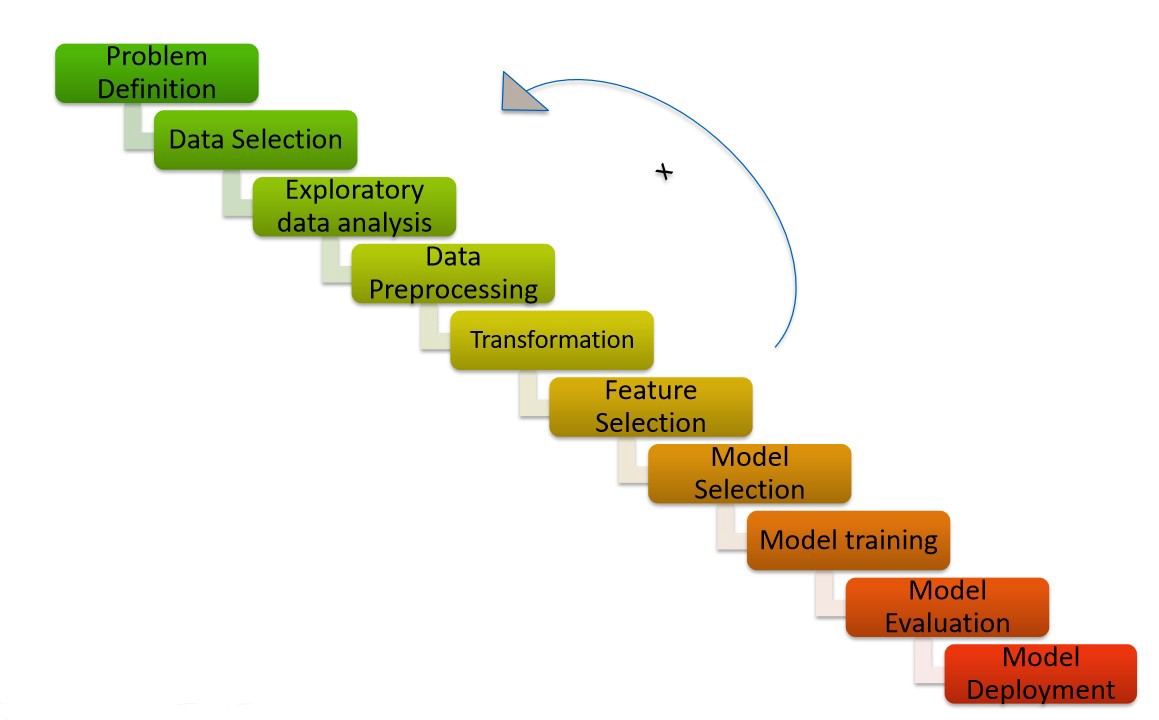
In this project, a dataset was provided with the details of the insurance policy along with the customer details, as well as details of the accident on the basis of which the claims have been made.

The Dataset was first cleaned, the various feature columns were analysed, then with feature engineering and based on strength of correlation and ANOVA f-score values, the feature columns were selected that would best predict the Target variable, to train and test machine learning models.

The auto insurance dataset was worked with to build a predictive model that best predicts if an insurance claim is fraudulent or not. Several models were trained and fitted with a part of the dataset and then tested with a different part of the dataset. The model that performed the best with the best confusion matrix performance, f1 score, ROC-AUC score and cross validation performance was then selected and tuned further with hyper parameter tuning techniques.

**Data Analysis**

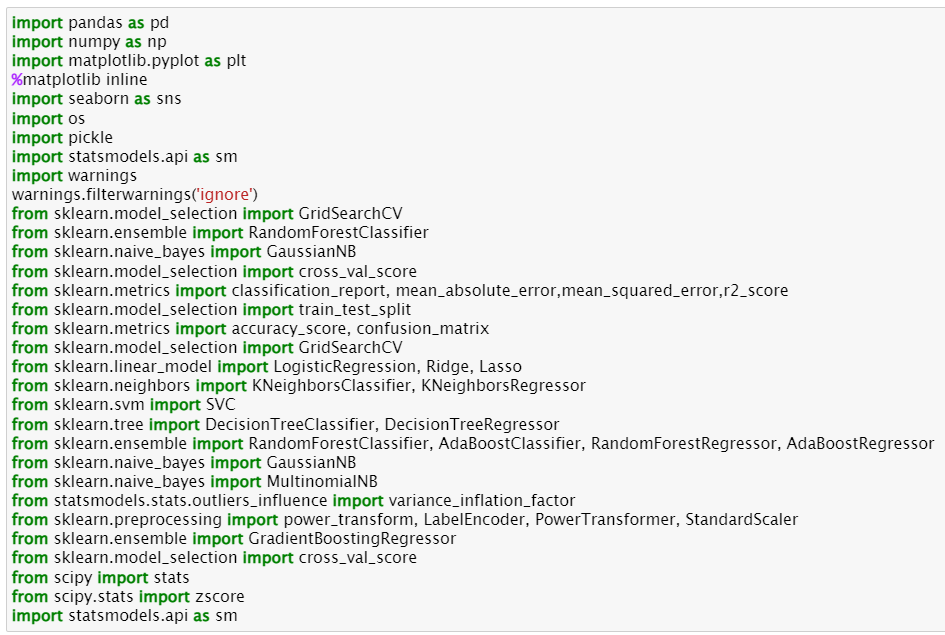
In order to build a Machine Learning Model, we have a Machine Learning Life Cycle that every Machine Learning Project has to touch upon in the life of the model. Let’s take a look at the model life cycle and then we will look into the actual machine learning model and understand it better along with the lifecycle as we move forward.

****

Now that we understand the lifecycle of a Machine Learning Model, lets import the necessary libraries and proceed further.

**Importing the necessary Libraries:**

To analyze the dataset or even to import the dataset, we have now imported all the important libraries that we will be needing during analysis.



**Importing the Dataset**

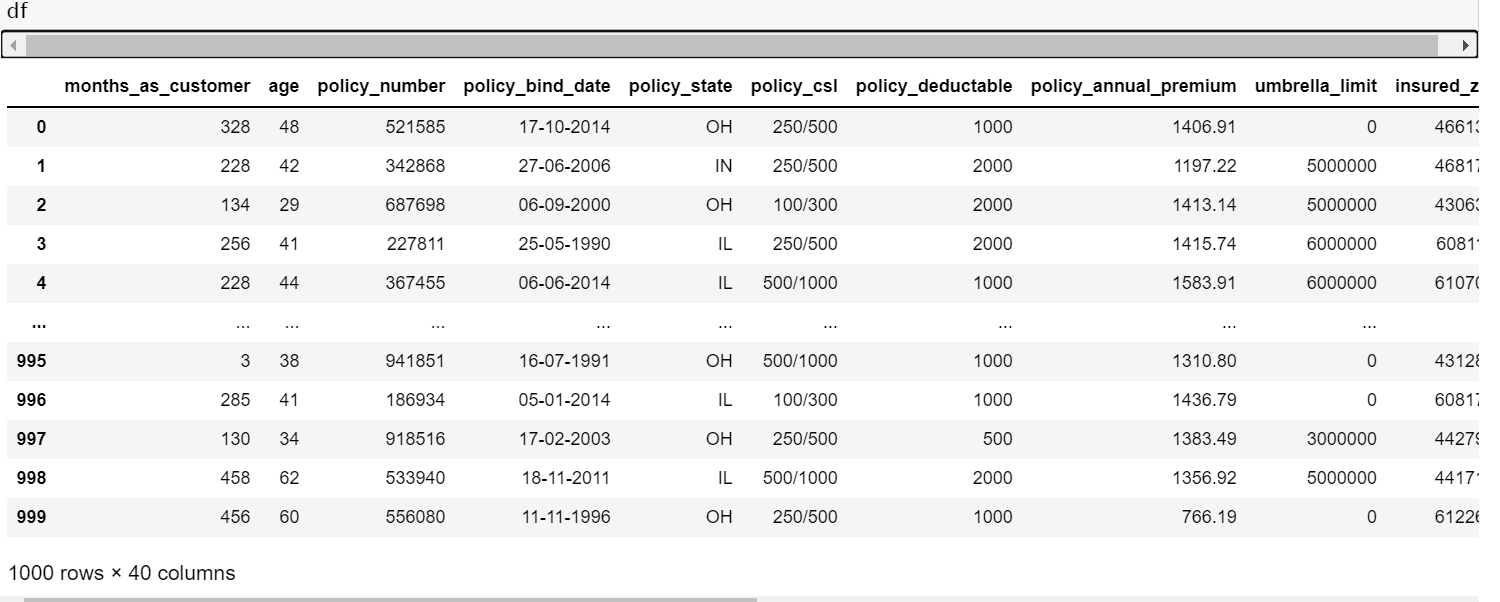
Let’s import the dataset first.



Copied the raw data and saved it as a csv file on my local computer after which I imported the entire dataset on this Jupyter Notebook with the help of pandas.

**About the Dataset:**

I have imported the dataset which was in “csv” format as “df”. Below is how the dataset looks.



The given dataset consists of 1000 rows and 40 columns.

This data set contains of Independent and Dependent (target) variables.

**Independent variable:** They are also known as Input variables. These are the input for a process that is being analyzed.

**Dependent variable:** They are also known as Output or Target variables. They are dependent on Independent variables for their outcome.

**The Independent Feature columns are:**

* **months\_as\_customer:** Number of months for which the person has been a customer
* **age:**  Age of Customer
* **policy\_number**: Identification number of policy
* **policy\_bind\_date**: Time period between effective date of coverage and policy issuance.
* **policy\_state**: State where policy is active
* **policy\_csl:**  Policy Combined single limit
* **policy\_deductable:** Amount paid before the insurance company starts paying up.
* **Policy\_annual\_premium**: The total amount of premium paid annually
* **Umbrella\_limit:** Provides excess limits and gives additional excess coverage
* **Insured\_zip:** Zip Code of the Insured address
* **insured\_sex :** Gender
* **Insured\_education\_level:** Education Background of Insured
* **Insured\_occupation:** Occupation of Insured
* **Insured\_hobbies:** Hobbies of the Insured
* **Insured\_relationship:** Relationship of the Insured
* **Capital-gains:** Capital Gains made from insurance
* **Capital-loss:** Capital Loss incurred
* **Incident\_date:** Date on which Incident Occured
* **incident\_type:** Type of Incident
* **Collision\_type:** Type of collision
* **incident\_severity:** Severity of Incident
* **Authorities\_contacted:** Whether authorities were contacted
* **Incident\_state:** State where incident occurred
* **incident\_city:** City where incident occurred
* **incident\_location:** Location of incident
* **Incident\_hour\_of\_the\_day:** Time of the day when incident occurred
* **number\_of\_vehicles\_involved:** Number of vehicles involved in incident.
* **property\_damage:** Whether there was property damage or not
* **Bodily\_injuries:** Severity of bodily injuries
* **witnesses:** Number of Witnesses
* **Police\_report\_available:** Whether police reports are available
* **Total\_claim\_amount:** Total amount of claim
* **Injury\_claim:** Injury Claim amount
* **Property\_claim:** Property Claim amount
* **vehicle\_claim:** Vehicle Claim amount
* **Auto\_make:** Make of Vehicle
* **Auto\_model:** Model of Vehicle
* **Auto\_year:** Year of Vehicle Manufacture

**The Target Variable to predict is given in the column:**

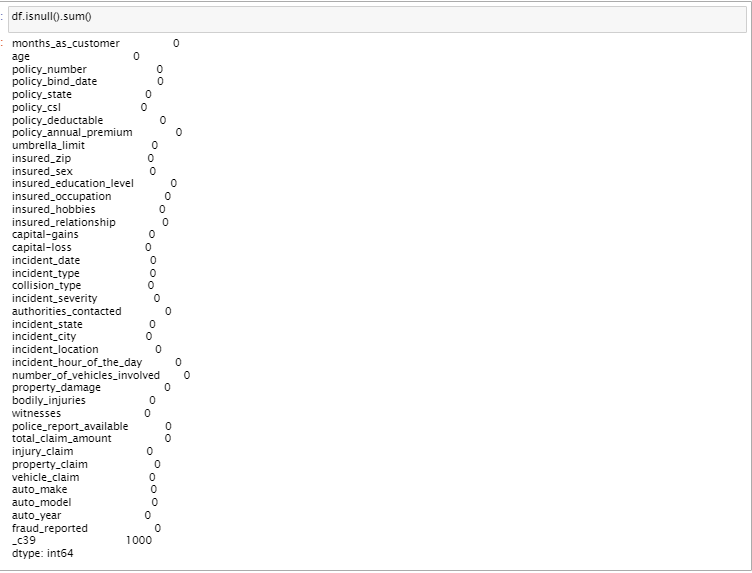
**Fraud\_reported:** Whether fraud was reported as Yes or No

**Data Analysis (Exploratory Data Analysis)**

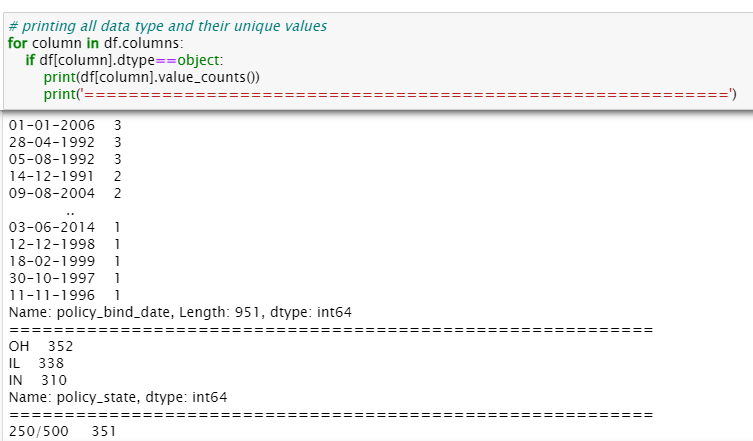
We cannot have null values in the data as this will affect the data and eventually the predicted result. Therefore, we must check for any null values in the dataset.

**Data Cleaning:**

Upon inspecting all the columns in the dataframe, it is observed that column \_c39 has no usable data present as they were all NaN values. Other columns appear to have no NaN.



**Checking for blank spaces, random characters in each column**

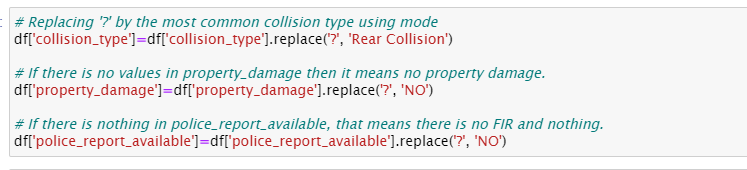
****

There are 178, 360 and 343 null values in collision\_type, property\_damage and police\_report\_available respectively.

* For collision\_type we can replace the null values with the mode of that column. The mode of the column for collision\_type is no.
* For property\_damage we can replace the null values with 'NO' because it could be that they weren't any property damage in the first place. Also, the mode of the column is 'NO'
* Similarly, for police\_report\_available we can replace the null values with 'NO' because it could be that they haven't reported. Similarly here also the mode of the column is 'NO'



**Therefore, Converting ' ?' to NaN values, which will later be imputed with values using various imputation techniques.**

****

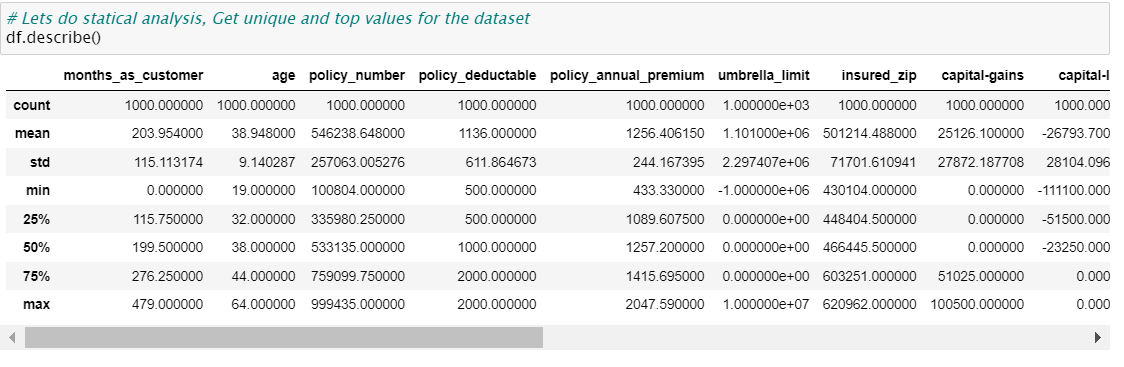
\_c39 has no usable data present. There is only one unique value, i.e: ‘nan’ which is null values. Therefore, we can eliminate this column.

****

**It is observed that no more null values remain in the dataset.**

****

**Exploratory Data Analysis**

****

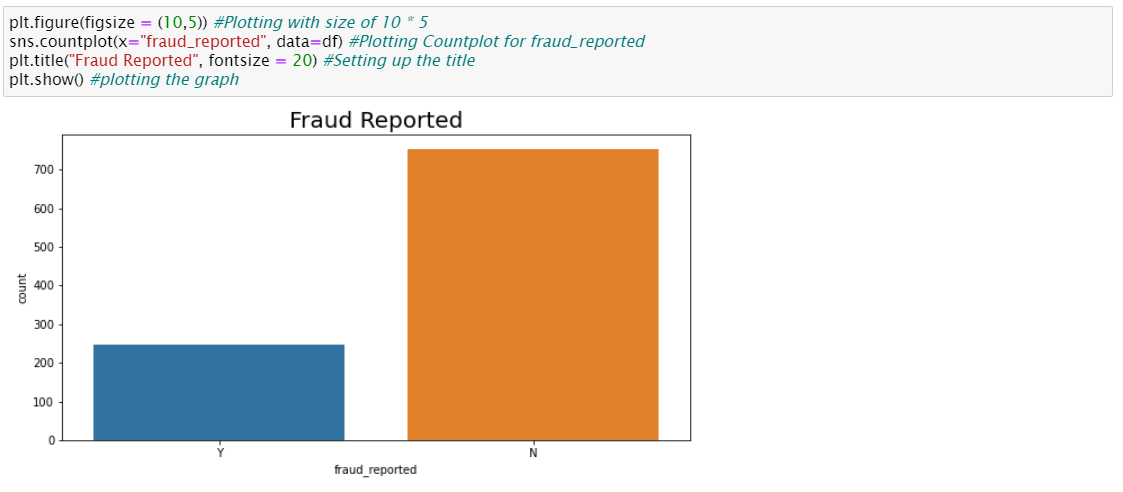
Difference in mean and 50% and considerable difference in 75% and max of columns months\_as\_customer,policy\_annual\_premium,capital-gains,total\_claim\_amount,injury\_claim and property\_claim suggests skewness in respective data distributions and presence of outliers.

### This is a Classification Problem since the Target variable / Label column ("fraud\_reported") has Categorical type of Data.

**Univariate Analysis**

**Analyzing the Target Class**

#### There are 2 unique categorical values in the Label column / target variable, viz. ‘Y’ and ‘N’.



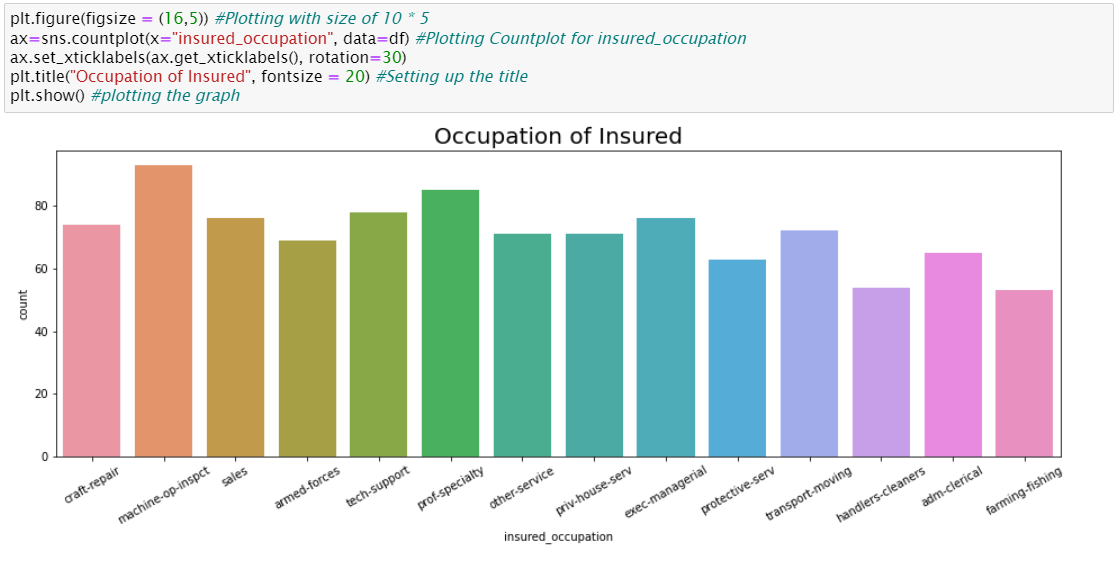
**Class**

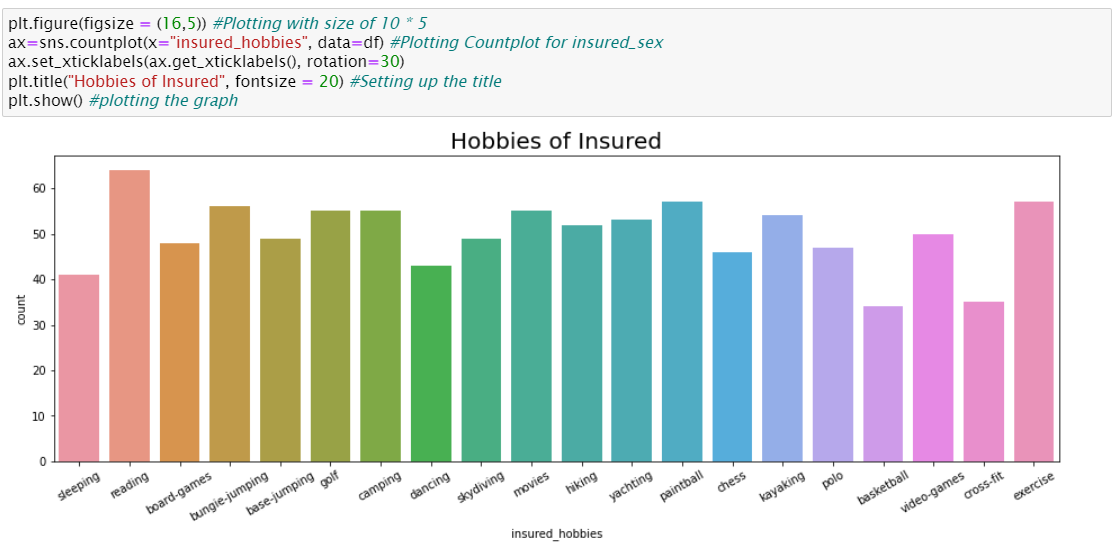
**'N' : Has 753 of total values**

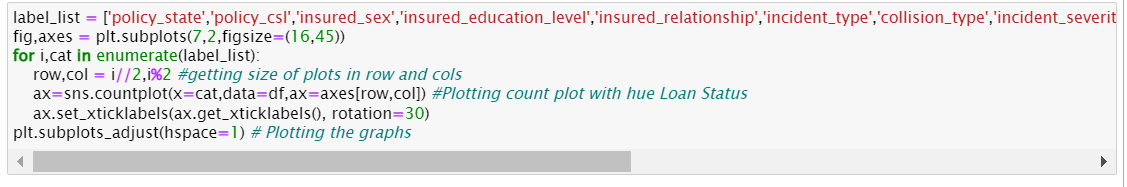
**'Y' : Has 247 of total values**

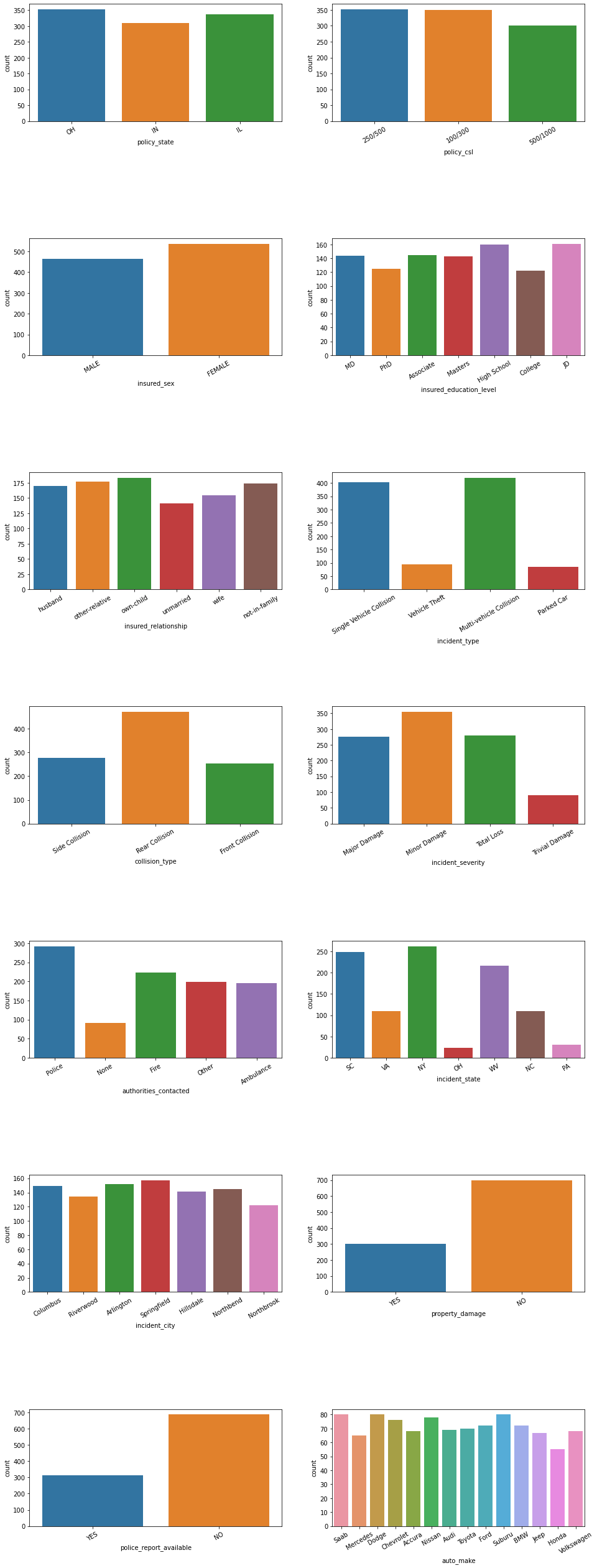
From the plot we can observe that the count of "N" is high compared to "Y". We can assume that "Y" stands for "Yes" that is the insurance is fraudulent and "N" stands for "No" means the insurance claim is not fraudulent. Here most of the insurance claims have not reported as fraudulent. Since it is our target column, it indicates the class imbalance issue. We will balance the data using oversampling method in later part.

#### **Analyzing the rest of the Feature Columns**









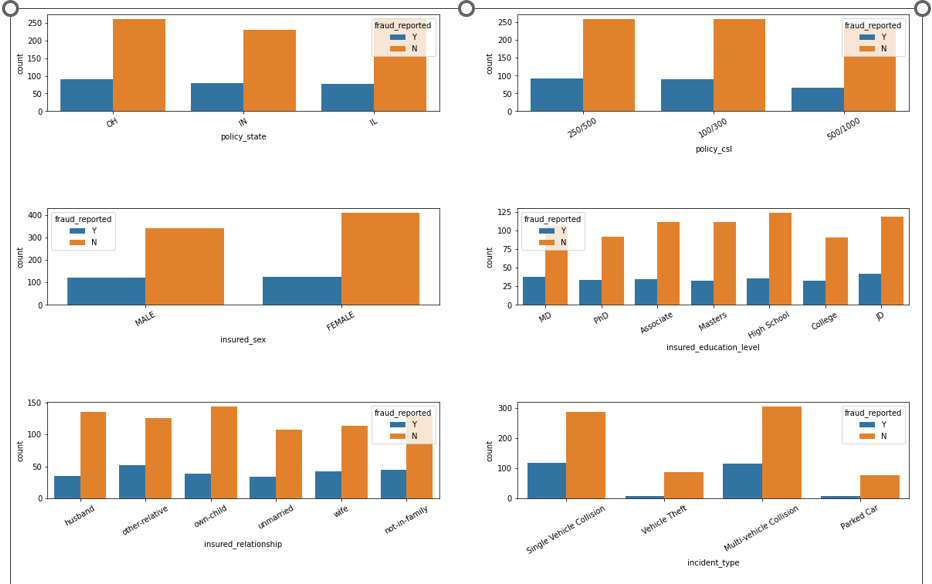
**Upon analyzing the rest of the Feature Columns, following observations are made:**

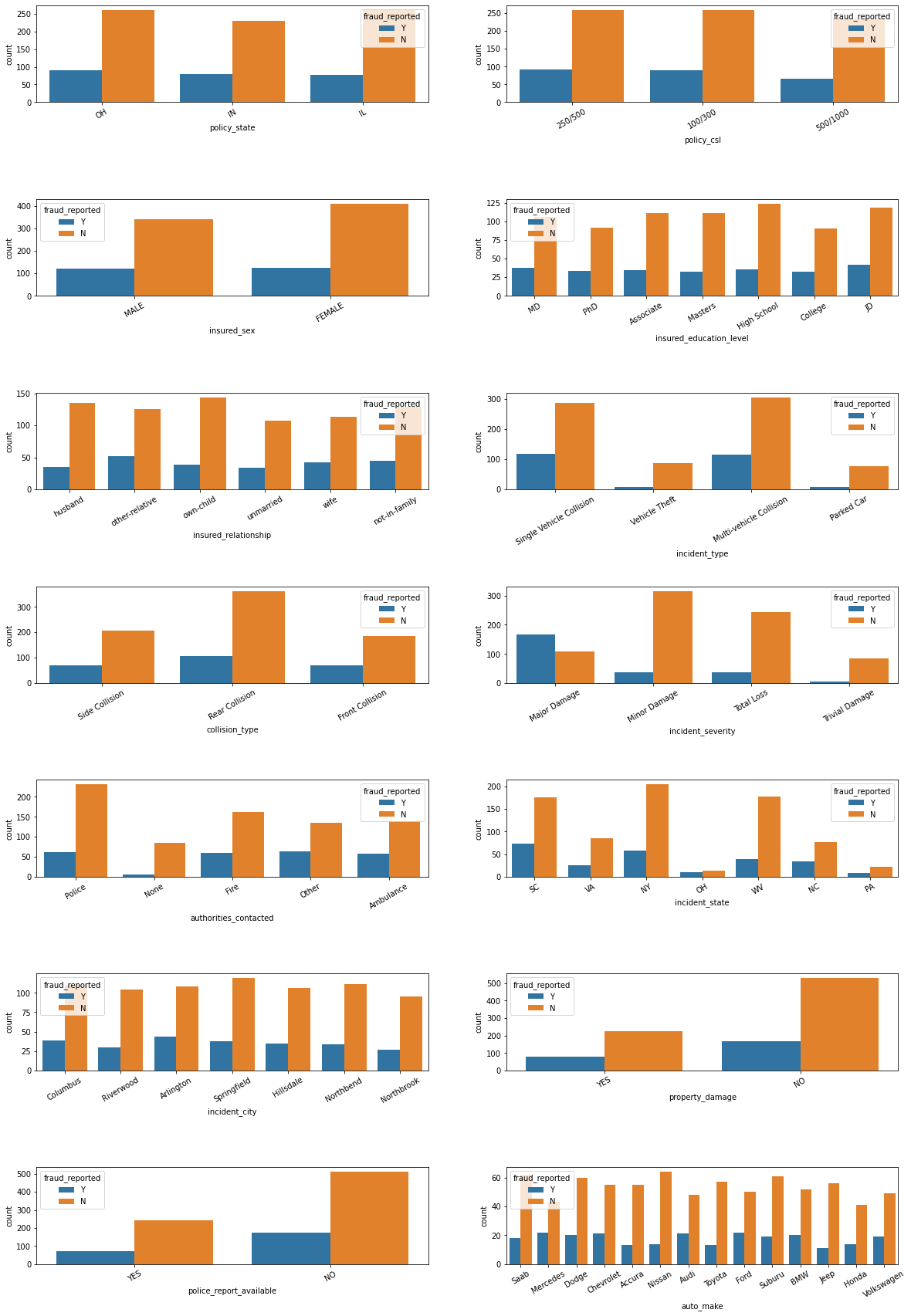
* Policy\_state is high in OH and IL where as low in IN.
* Policy\_csl is 250/500 and 100/300 is high whereas 500/1000 is less.
* Female applied for Insurance more than Male.
* Mostly insured person have education level High School and JD.
* The people who have child, those peoples did insurances more.
* Single Vehicle Collision and Multi vehicle Collision had more incident whereas Vehicle Theft and Parked car incidents are less.
* Rear Collision is more than 400 whereas Front collision is less.
* Minor Damages are happen more whereas Trival Damages are less.
* People contacted more and in very less conditions people didn't call to anyone.
* More incident happened in NY where as in OH very less incident happen.
* Almost all cities have same number of incidents.
* Property damages are less.
* Police reports are available only in 30% cases.
* All cars are insured equally near 60-80.

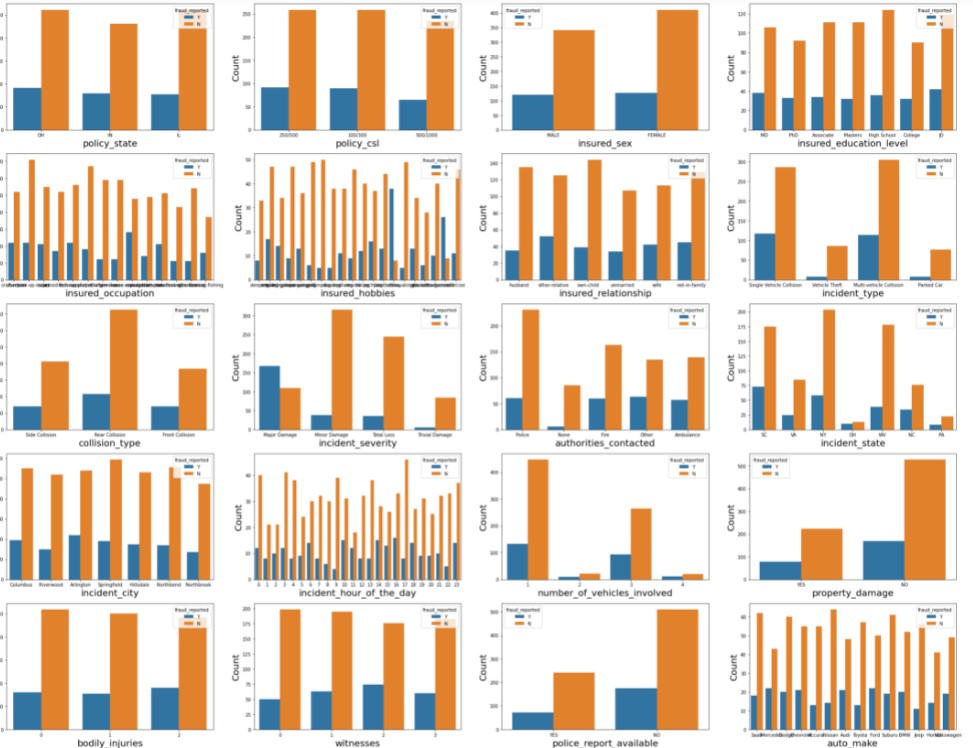
**Bivariate Analysis**

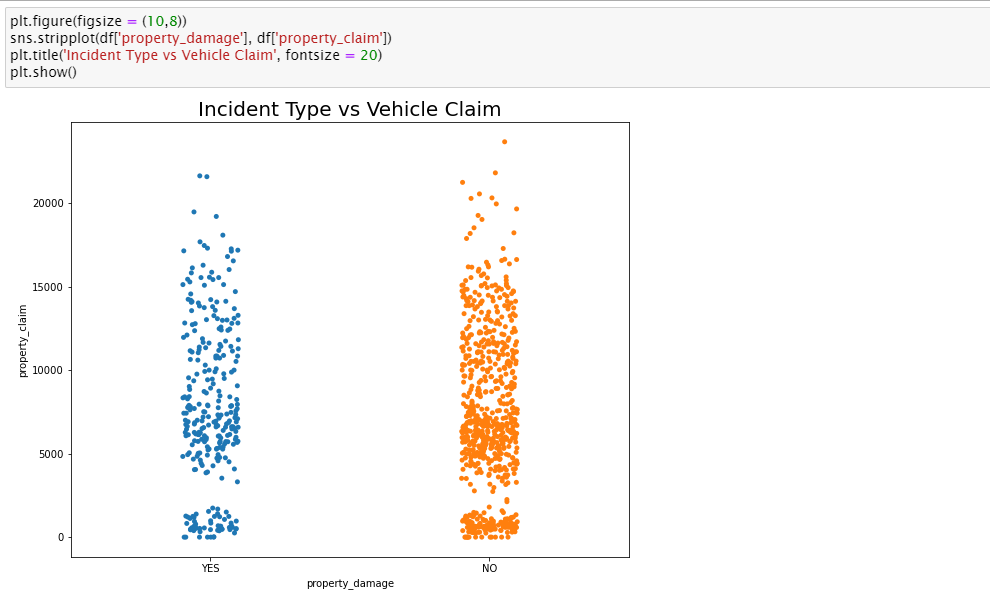
### Interpreting Relationship between Dependent Variable and Independent Variables

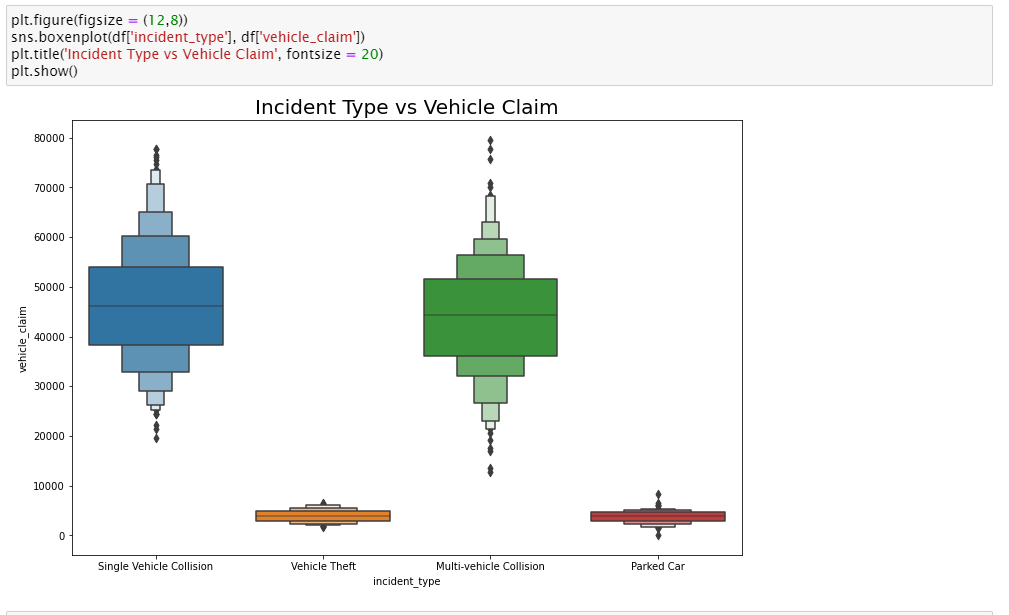


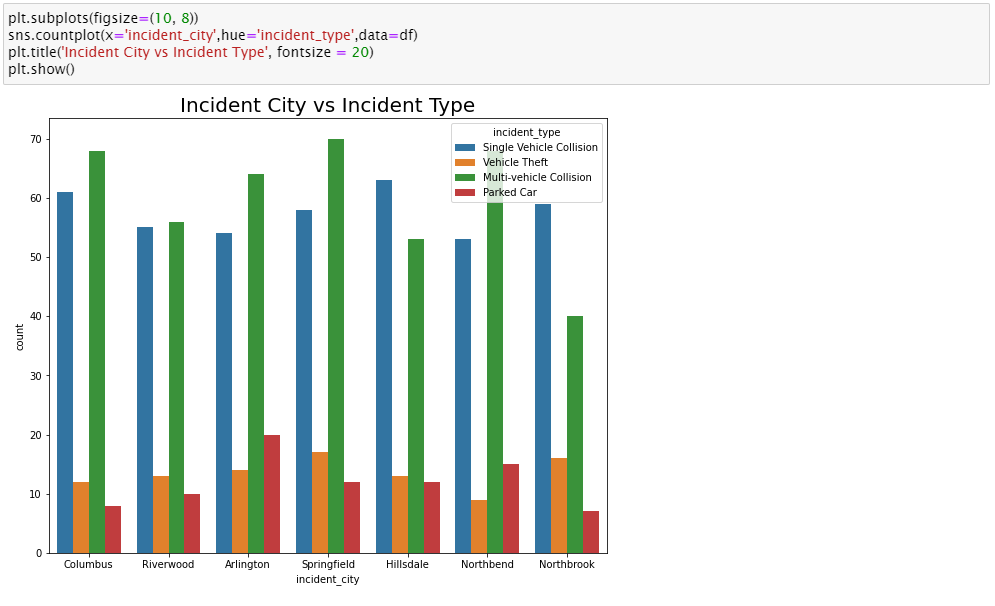


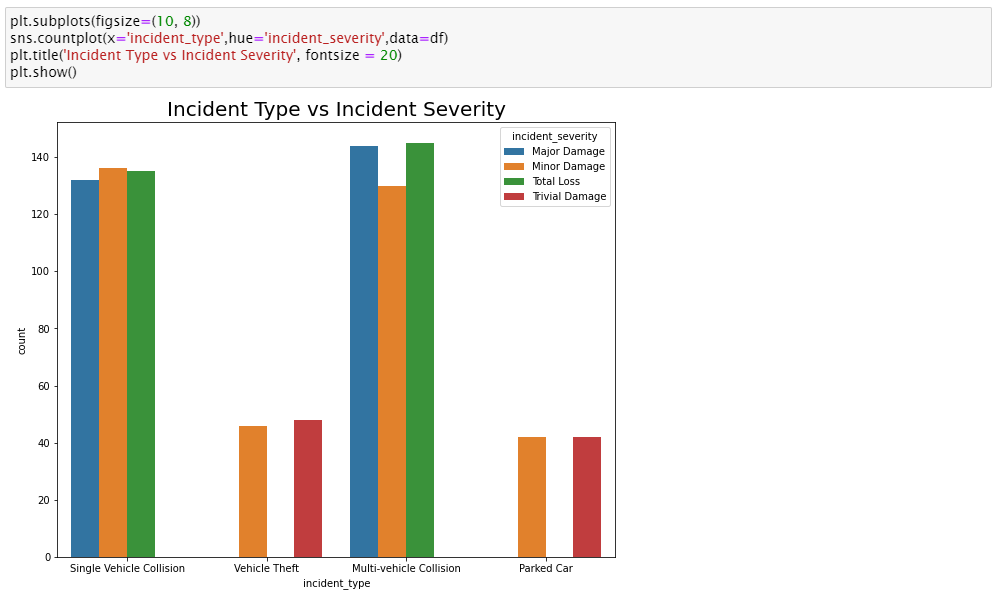




****

****

****

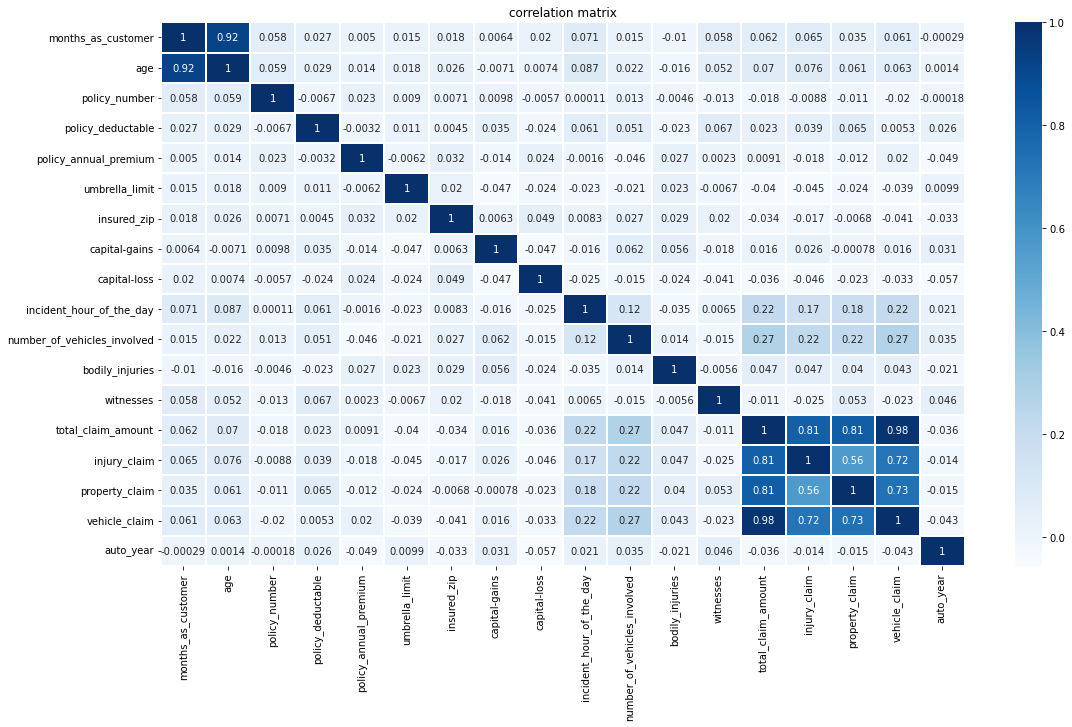
****

**Following observations can be made from above graphs:**

* In OH and IL Policy state Fraud didn't happen much when compared with IN. In IN Fraud happen more than OH and IL.
* In Policy\_csl 250/500 and 100/300 Fraud reported more than 500/1000.
* More number of Fraud case reported against Female.
* Person who has education level JD did more frauds than others.
* People who have other relatives has reported highest fraud cases.
* In case of Single Vehicle Collision and Multi vehicle Collision maximum frauds happened.
* When Collision type is Rear, Fraud reported more.
* At the time of Major Damages, Fraud is reported too high.
* When People didn't call to anyone that time Fraud happen very less. If person contacted to anyone, Fraud happened.
* In SC state Fraud Reported more than other states.
* Almost all cities have same number of Fraud Reported.
* When Property is not damages that time Fraud happed more.
* When there is no Police report available that time Fraud happen more.
* Mercedes and Ford have a greater number of Fraud Reported.
* We can clearly see that In Single Vehicle collision and Multi-vehicle collision, Major Damage, Minor Damage and Total loss happened, whereas when Vehicle is Theft or car was parked that time Minor damage and Trivial Damage happen.

**Multi-Variate Analysis**

****

****

In the above plot we can see the relation between each variable with respect to other variables.

Now we have done with the visualization in order to analyze and understand the data. So, in this EDA part, we have looked into various aspect of the dataset, like looking for the null values and imputing, extracting date time, observing the value counts and doing the feature extraction etc.

Now we will be performing Data Pre-processing by identifying the outliers and removing them. Along with it we will also look for the skewness of the dataset and remove the skewness.

**Data Visualization and EDA Concluding Remarks:**

In the given data, ‘fraud\_reported’ feature is the Target feature or variable. The unique values of this feature are only 2 i.e Y and N (Yes and No), which means it has only two classes. So, as there are only two unique values this is a ‘Classification Problem.’

The dependent or target variable has 753 nonfraudulent cases and 247 fraudulent cases which can be seen below in the form of value counts and bar chart.

**Data Pre-Processing**

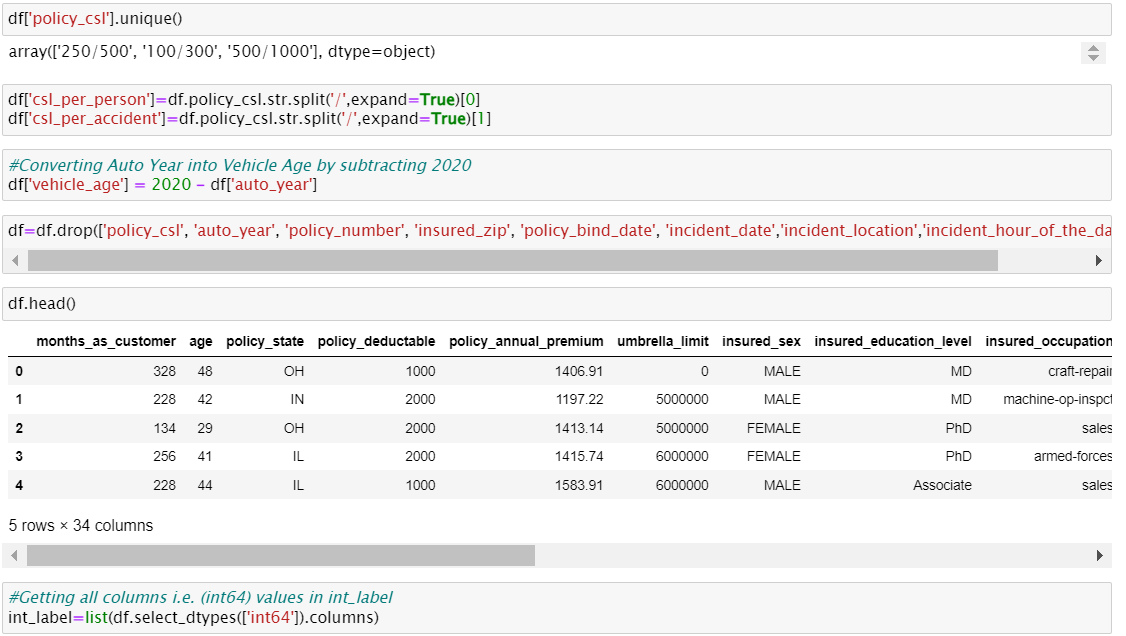
The data set has variables in both object type and numerical type (int and float)

Therefore, we have to pre-process the data to move forward.

All the float type or int type variables should be converted into the same scale since the range of values of raw data varies widely, in some machine learning algorithms, objective functions do not work correctly without normalization.

Therefore, normalization is to be performed only on the numerical type (int and float type) variables.

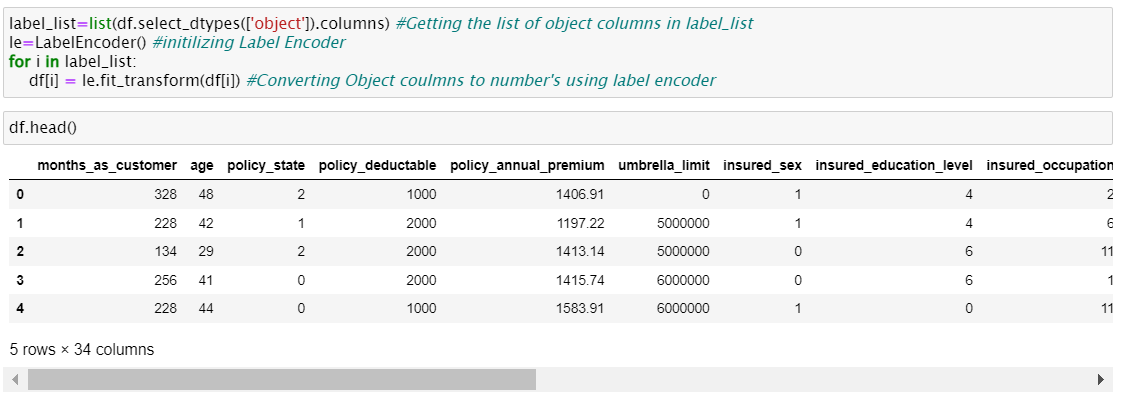
**Feature Engineering**

****

**Encoding the categorical columns using Label Encoding**

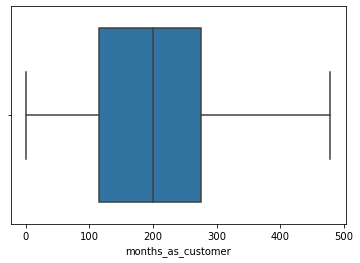
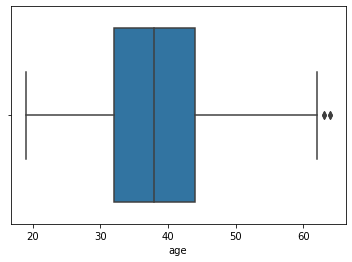
We can see that there are object-type variables too. These variables contain string data that cannot be passed into the machine learning model as it won’t be able to recognize string data type. It only recognizes numerical data.

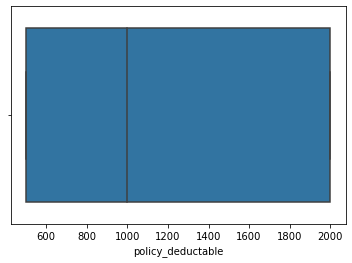
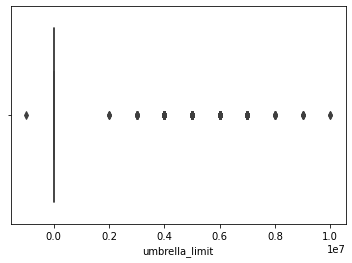
Therefore, we need to convert the string data into numerical data. This can be done by manually encoding or by using an encoder such Label Encoder, one-hot encoder etc. For example: the target variable fraud\_reported consists of only two unique values, Y & N. after encoding this will get converted to 0 and 1. Similarly, if there are three unique values then it will be converted to 0,1, and 2.

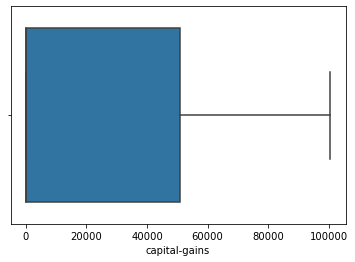
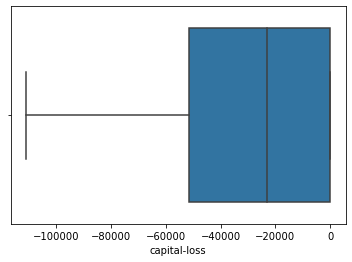
****

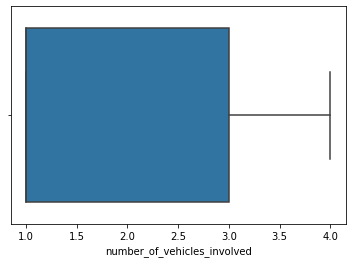
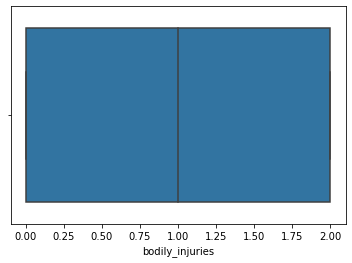
Now we have encoded the dataset using label encoder and the dataset looks like above.

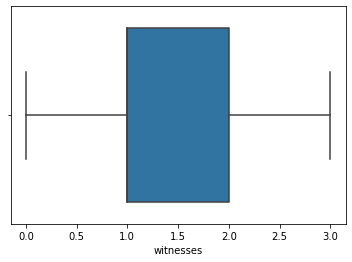
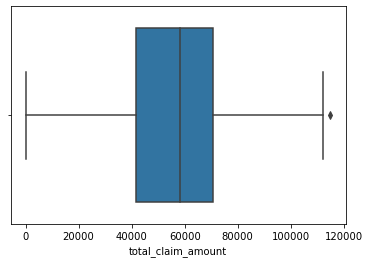
**Checking for Outliers in columns with continuous distribution**

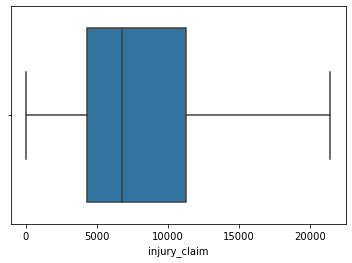
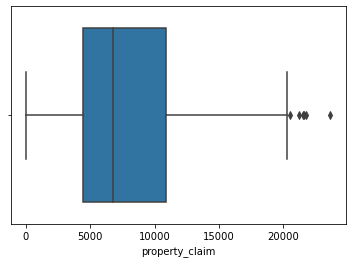
 

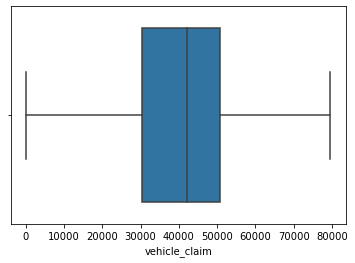
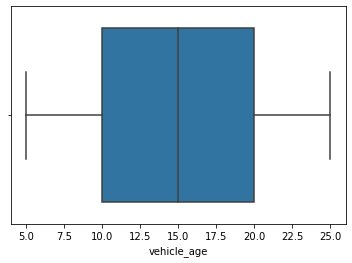
 



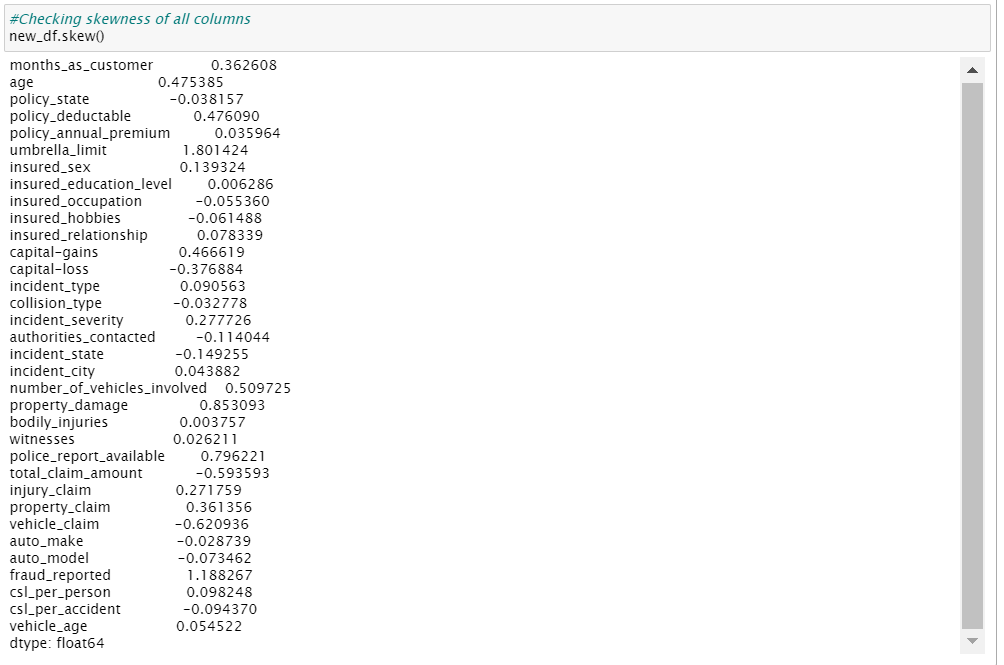
 

#### **Removing the outliers using Zscore method.**

#### 

After removing the outliers, we are checking the data loss percentage by comparing the rows in our original data set and the new data set and 2% data loss is in the acceptable range.

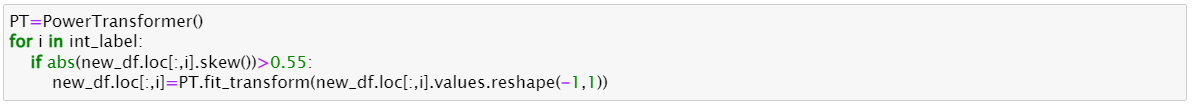
#### **Checking for Skewness in Data**

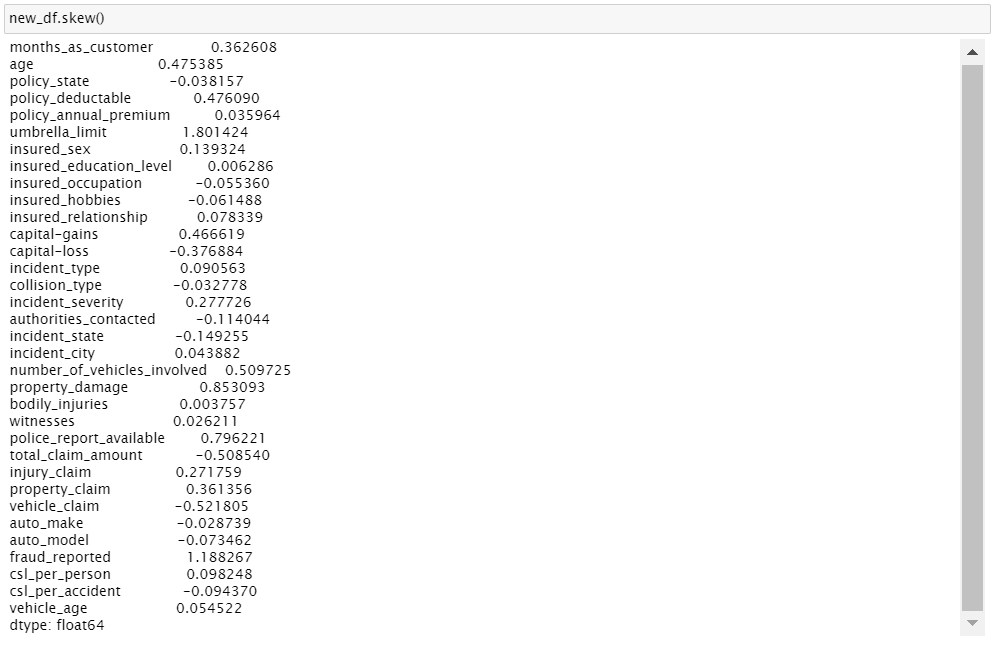


#### 

#### **Reducing skewness using PowerTransformer**

As we can see that skewness is present in the dataset, we will try to reduce it using PowerTransformer





Skewness has been greatly reduced.

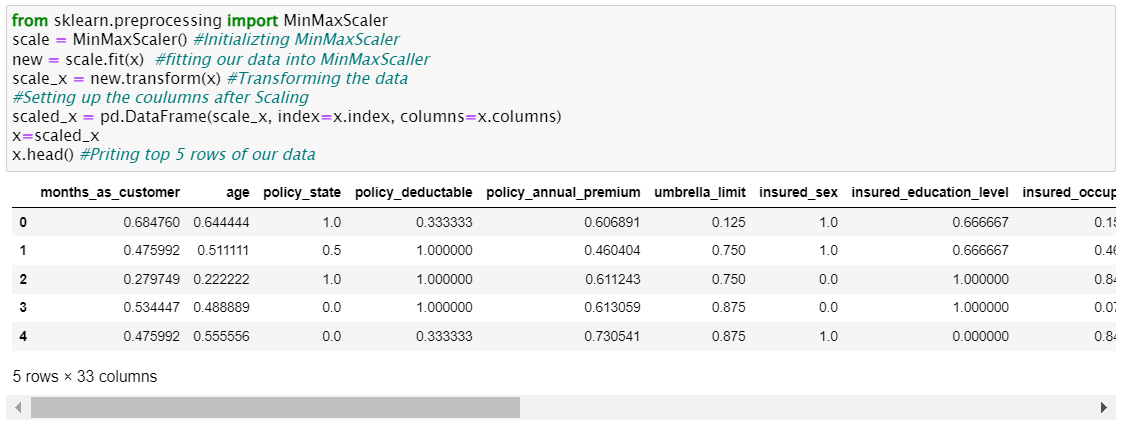
Next step is to select the best features which would build the most accurate Machine Learning Models to predict the target variable.

### Separating Input and Output Variables



**Feature Scaling**

**Scaling the values in the feature columns using MinMaxScaler.**



**Building Machine Learning Models:**

We will use a machine-learning algorithm to learn from the training set and use the model to predict the testing set and compare it with the predicted data with the target testing set to know how close the values. If the error between the predicted and target testing data is less that means the accuracy of the model is high and we can use this model to predict the result of similar datasets.

In this, we have used 7 Machine learning Algorithms

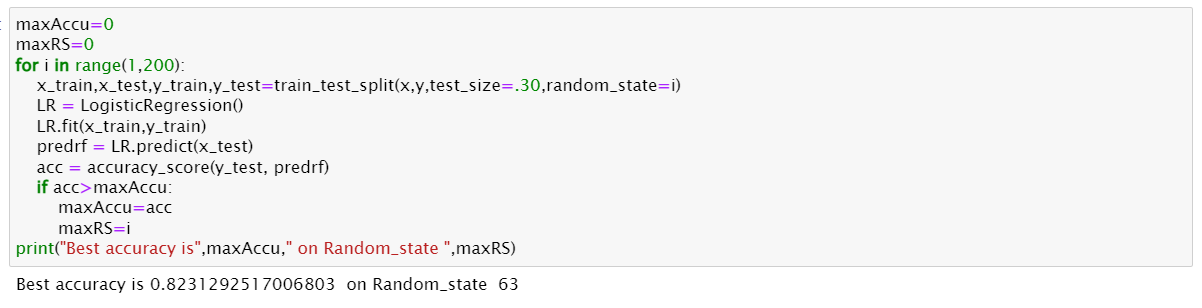
* LogisticRegression
* KneighborsClassifier
* AdaBoostClassifier
* SupportVectorClassifier (SVC)
* DecisionTreeClassifier
* RandomForestClassifier
* MLPClassifier

We can train and predict the data using the above 7 ML algorithms and save the model which has the highest frequency.

**Finding the Best Random State**

Let’s find the best random state in which we can build the model*.*

(Random state ensures that the splits that you generate are reproducible. Scikit-learn use random permutations to generate the splits. The random state that you provide is used as a seed to the random number generator. This ensures that the random numbers are generated in the same order.)

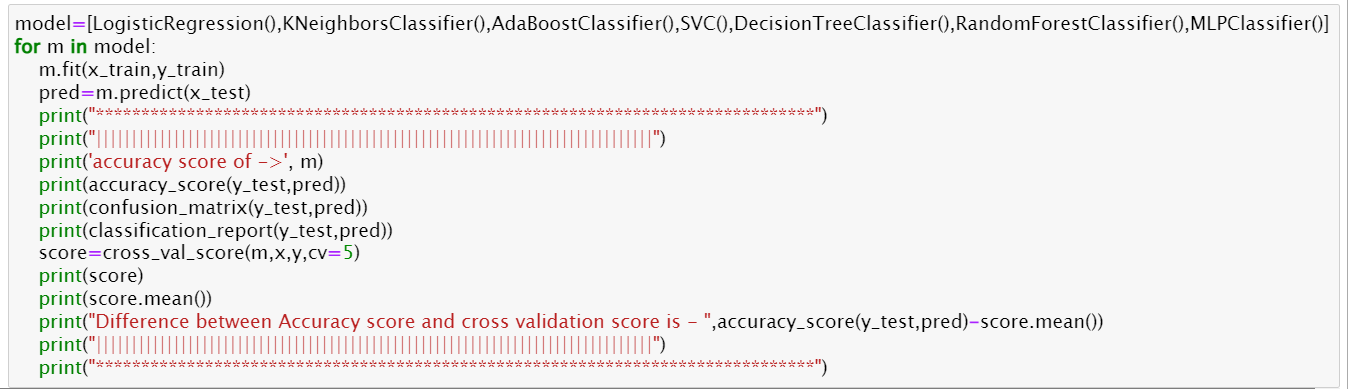


**Creating Train-Test split based on random state obtained above:**

****

### Training the Models Finding Best Algorithm

We will run various algorithms and check which has the highest Accuracy score, Cross Validation Score and (Accuracy Score - Cross Validation Score) for comparison between all the algorithms.



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> LogisticRegression()

0.8231292517006803

[[218 16]

[ 36 24]]

precision recall f1-score support

0 0.86 0.93 0.89 234

1 0.60 0.40 0.48 60

accuracy 0.82 294

macro avg 0.73 0.67 0.69 294

weighted avg 0.81 0.82 0.81 294

[0.78571429 0.75510204 0.75 0.79081633 0.79081633]

0.7744897959183674

Difference between Accuracy score and cross validation score is - 0.04863945578231288

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> KNeighborsClassifier()

0.7517006802721088

[[215 19]

[ 54 6]]

precision recall f1-score support

0 0.80 0.92 0.85 234

1 0.24 0.10 0.14 60

accuracy 0.75 294

macro avg 0.52 0.51 0.50 294

weighted avg 0.69 0.75 0.71 294

[0.72959184 0.70918367 0.69897959 0.75510204 0.71428571]

0.7214285714285713

Difference between Accuracy score and cross validation score is - 0.030272108843537482

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> AdaBoostClassifier()

0.826530612244898

[[209 25]

[ 26 34]]

precision recall f1-score support

0 0.89 0.89 0.89 234

1 0.58 0.57 0.57 60

accuracy 0.83 294

macro avg 0.73 0.73 0.73 294

weighted avg 0.83 0.83 0.83 294

[0.81122449 0.7755102 0.73979592 0.81632653 0.83673469]

0.7959183673469388

Difference between Accuracy score and cross validation score is - 0.030612244897959218

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> SVC()

0.7925170068027211

[[231 3]

[ 58 2]]

precision recall f1-score support

0 0.80 0.99 0.88 234

1 0.40 0.03 0.06 60

accuracy 0.79 294

macro avg 0.60 0.51 0.47 294

weighted avg 0.72 0.79 0.72 294

[0.75510204 0.75 0.75 0.76020408 0.76020408]

0.7551020408163265

Difference between Accuracy score and cross validation score is - 0.0374149659863946

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> DecisionTreeClassifier()

0.7891156462585034

[[197 37]

[ 25 35]]

precision recall f1-score support

0 0.89 0.84 0.86 234

1 0.49 0.58 0.53 60

accuracy 0.79 294

macro avg 0.69 0.71 0.70 294

weighted avg 0.81 0.79 0.80 294

[0.7755102 0.79591837 0.76530612 0.75 0.77040816]

0.7714285714285714

Difference between Accuracy score and cross validation score is - 0.017687074829932037

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> RandomForestClassifier()

0.7925170068027211

[[219 15]

[ 46 14]]

precision recall f1-score support

0 0.83 0.94 0.88 234

1 0.48 0.23 0.31 60

accuracy 0.79 294

macro avg 0.65 0.58 0.60 294

weighted avg 0.76 0.79 0.76 294

[0.7755102 0.79081633 0.7244898 0.76020408 0.78571429]

0.7673469387755102

Difference between Accuracy score and cross validation score is - 0.02517006802721089

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

accuracy score of -> MLPClassifier()

0.7993197278911565

[[210 24]

[ 35 25]]

precision recall f1-score support

0 0.86 0.90 0.88 234

1 0.51 0.42 0.46 60

accuracy 0.80 294

macro avg 0.68 0.66 0.67 294

weighted avg 0.79 0.80 0.79 294

[0.80612245 0.75 0.73979592 0.79591837 0.80102041]

0.7785714285714287

Difference between Accuracy score and cross validation score is - 0.02074829931972777

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

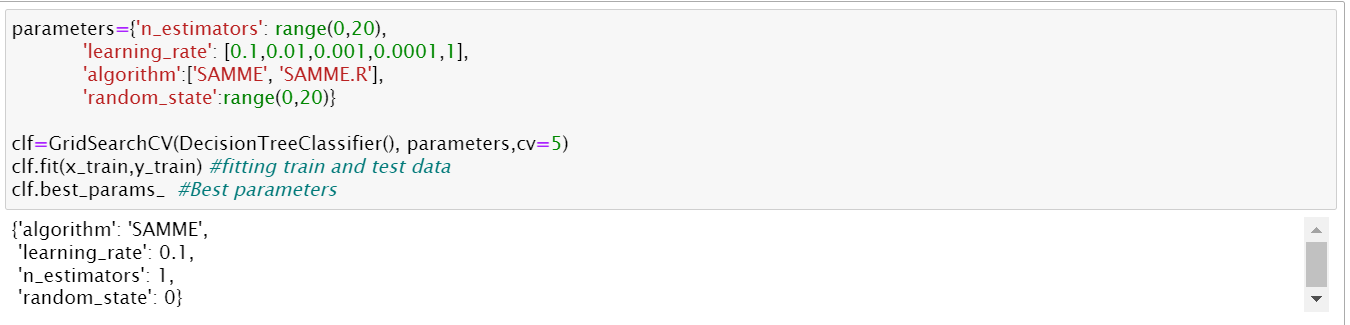
**Analyzing Model Accuracies**

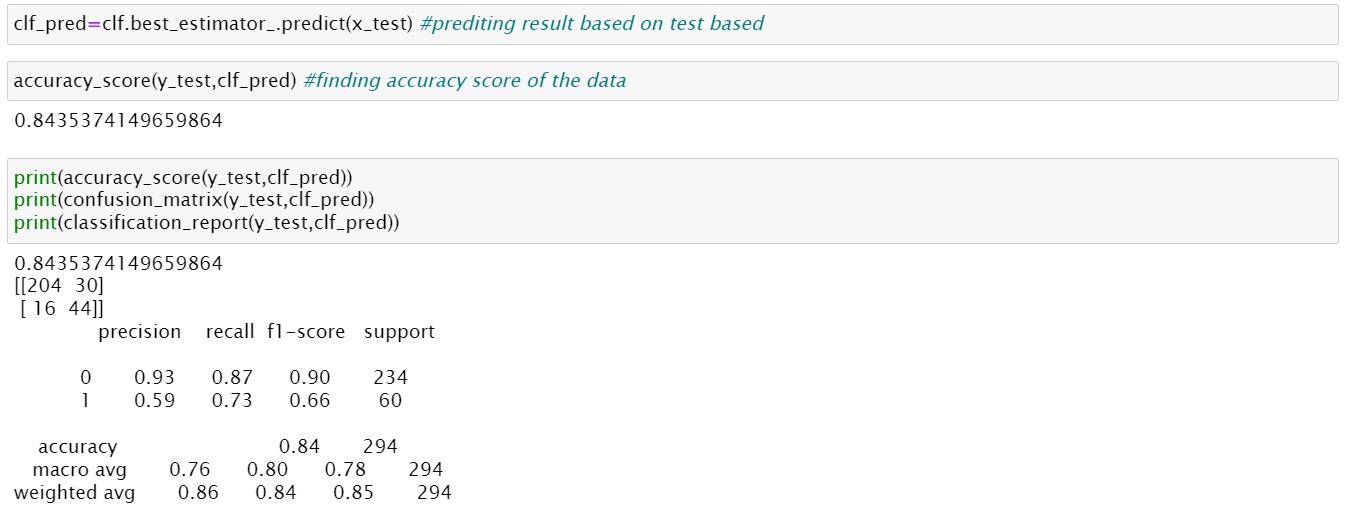
|  |  |  |  |
| --- | --- | --- | --- |
| **Models** | **Accuracy Score** | **F1 Score** | **Difference between Accuracy score and Cross validation score** |
| LogisticRegression | 0.82 | 0.89 | 0.048 |
| KNeighborsClassifier | 0.75 | 0.85 | 0.030 |
| AdaBoostClassifier | 0.82 | 0.89 | 0.030 |
| SVC | 0.79 | 0.88 | 0.037 |
| DecisionTreeClassifier | 0.78 | 0.86 | 0.017 |
| RandomForestClassifier | 0.79 | 0.88 | 0.025 |
| MLPClassifier | 0.79 | 0.88 | 0.020 |

According to Cross val score and accuracy we can see that the DecisionTreeClassifier has the least difference between Accuracy and Cross val score, therefore we select DecisionTreeClassifier model.

### Hyper Parameter Tuning

GridSearchCV was used for Hyper Parameter Tuning of the DecisionTreeClassifier model. Based on the input parameter values and after fitting the train datasets, the DecisionTreeClassifier Model was further tuned based on the parameter values yielded from GridsearchCV. Getting the list of the best parameters from Grid Search CV.





The Final Accuracy Score of the final Model displayed accuracy of 84.35%

We successfully performed the Hyper Parameter Tuning on the Final Model.

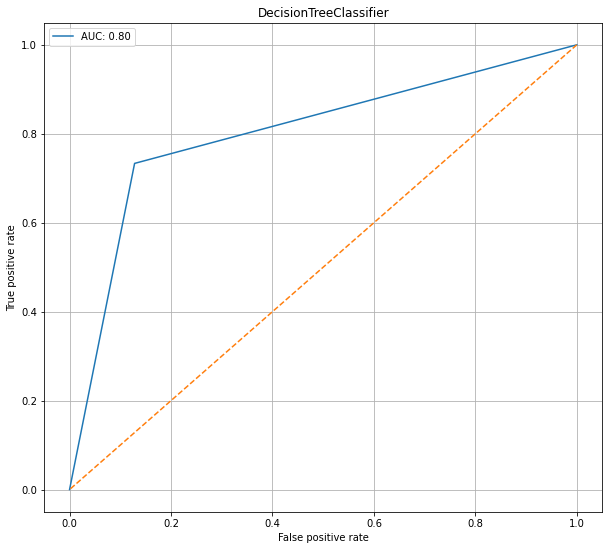
### AUC ROC curve

AUC: Area Under the curve; ROC: Receiver Operator Characteristic

The greater the ROC score the better is the model. If ROC=1, then it perfectly fits.

If the maximum of the area falls under True positive then the model is doing good.





We can see that the ROC Score is 0.80 and the area under the curve falls under True Positive Rate, Therefore, we can conclude that the model is performing well and we need to save the model in joblib file for future use

**Saving the model in joblib Format**

### 

**Prediction Conclusion**

We will predict the "fraud\_reported" target column using the final model sending the “x\_test” set for predicting the “y\_test” and then compare the original “y\_test” and the predicted “y\_test” in a dataframe.

### 

### Concluding Remarks

In the beginning of the blog, we have discussed about the lifecycle of a Machine Learning Model, you can see how we have touched based on each point and finally reached up to the model building and made the model ready for deployment.

Let’s take a quick recap on all the steps that we went through starting from understanding the Problem Definition then going through the Data Analysis and EDA processes. We went through the necessary Pre-processing Data steps before the final Building Machine Learning Models step came into picture.

The Insurance industry area needs a good vision on data, and in every model building problem Data Analysis and Feature Engineering is the most crucial part.

You can see how we have handled numerical and categorical data and also how we build different machine learning models on the same dataset. And using hyper parameter tuning we can improve our model accuracy. Hence, we ended up with a prediction accuracy of 95% after all the above steps are completed.

Using this machine Learning Model, we can predict whether the insurance claim is fraudulent or not and we could reject those application which will be considered as fraud claims. Saving the Insurance industry from possible fraudsters.