*Blog on: Insurance Claim Fraud Detection Using Machine Learning*



# *Author: Priya Patidar*

# *Batch: 1843*

# *DataTrained*

Introduction:

Scam or Fraud is one of the largest and most well-known problems that insurers face in the insurance industry.

Insurance fraud is a deliberate deception perpetrated against or by an insurance company or by an agent for financial profit. Fraud may be committed at different points in the transaction by applicants, policyholders, third-party claimants, or professionals who provide services to claimants. Insurance agents and company employees may also commit insurance fraud. Common frauds include "padding," or inflating claims; misrepresenting facts on an insurance application; submitting claims for injuries or damage that never occurred; and staging accidents.

Data Science and Machine Learning, which has been very useful in many industries that have always managed to bring accuracy or detect negative incidents. in this blog, I have created a Machine Learning model to detect if the claim is fraudulent or not. Here various features have been used like, insured personal information, insured persons, personal details, and incident information. In total the dataset has 40 different features. So using all these previously acquired information and analysis done with the data I have achieved a good model that has a 92% accuracy rate. So let's see what steps are involved to attain this accuracy.

Various visualization techniques have also been used to understand the co-linearity and importance of the features.

**Software Requirements & Tools Used:**

**Software requirement**:

* Jupiter Notebook

**Libraries Used**:

* Python
* NumPy
* Pandas
* Matplotlib
* Seaborn
* Date Time
* Scikit Learn

## Problem Definition:

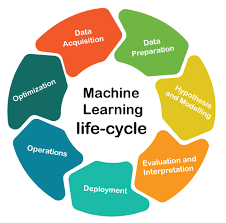
Insurance fraud is a big problem in this industry. It's difficult to identify fraud claims. Machine Learning algorithms are in a unique position to help the Auto Insurance industry with this huge problem. In this project, we are provided a dataset that has the details of the insurance policy along with the customer's details. It also has the details of the accident based on which the claims have been made.

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

In this problem, we will be looking into the insured person's details and the incidents and we will be analyzing the sample to understand if the claim is genuine or not.

**Let's deep dive step by step into the data analysis process.**

To build a Machine Learning Model, we have a Machine Learning Life Cycle that every Machine Learning algorithm has to touch upon in the life of the model. Let's a sneak peek into the model life cycle and then we will look into the actual machine learning model and understand it better along with the lifecycle.



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

## Importing the necessary Libraries:

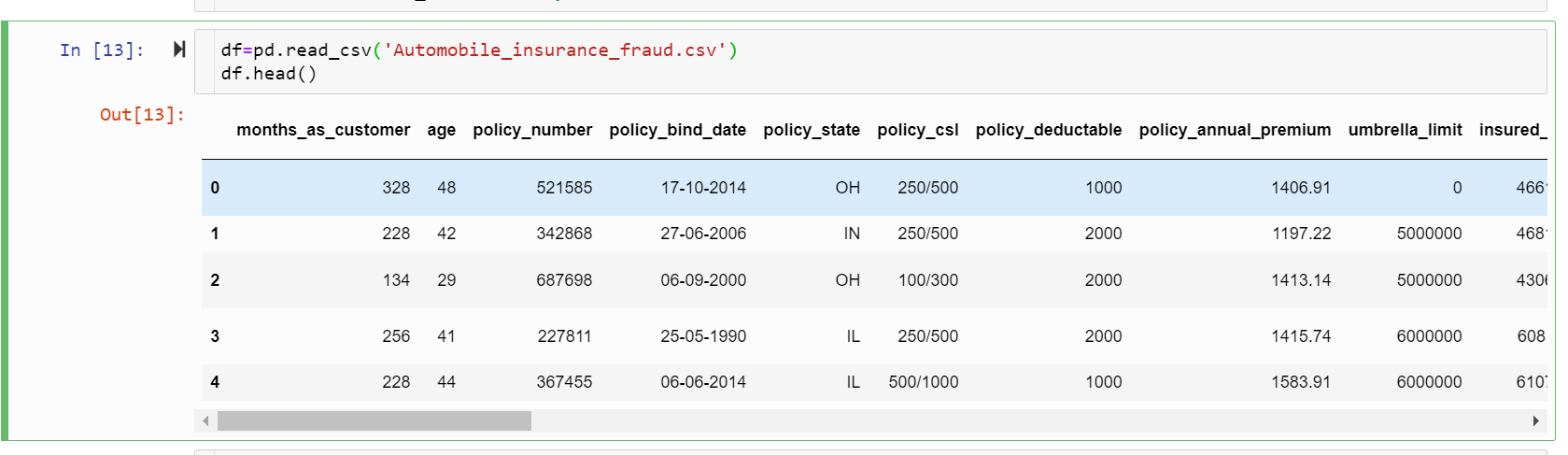
To analyze the dataset, we have imported all the necessary libraries as shown below.

* Pandas have been used to import the dataset and also in creating data frames.
* Seaborn and Matplotlib have been used for visualization
* Date Time has been used to extract day/month/date separately
* Sklearn has been used in the model building



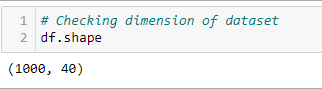
## Importing the Dataset

* Let’s import the dataset from the source.
* imported the dataset which was in “CSV" format as “df”. Below is how the dataset looks.



By observing the dataset we could understand that the dataset contains both categorical and numerical columns. Here "fraud\_reported" is our target column since it has two categories so it is called to be a "Classification Problem" where we need to predict if an insurance claim is fraudulent or not. As it is a classification problem hence we will use all the classification algorithms while building the model.

by using 'df. shape' we can find figure out how many rows and columns we have in the dataset. We have got the output that we have 1000 rows and 40 columns. PCA can be done, however, I decided not to lose any data at this time as the dataset is comparatively small in size, and the first lesson of a data scientist is 'Data is Crucial' hence proceeded will all the datasets.

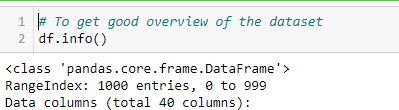


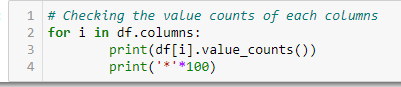
As per the lifecycle of the machine learning model, we have defined our problem and we have selected the data from the source. Now we will perform EDA, data preprocessing, and transformation, which is the most important part of any machine learning model, further data will be analyzed and cleaned for better model accuracy which we will get, or the model can remain overfitting or underfitting. We will discuss further why all the steps are used.

## Exploratory Data Analysis and Data Preparation:

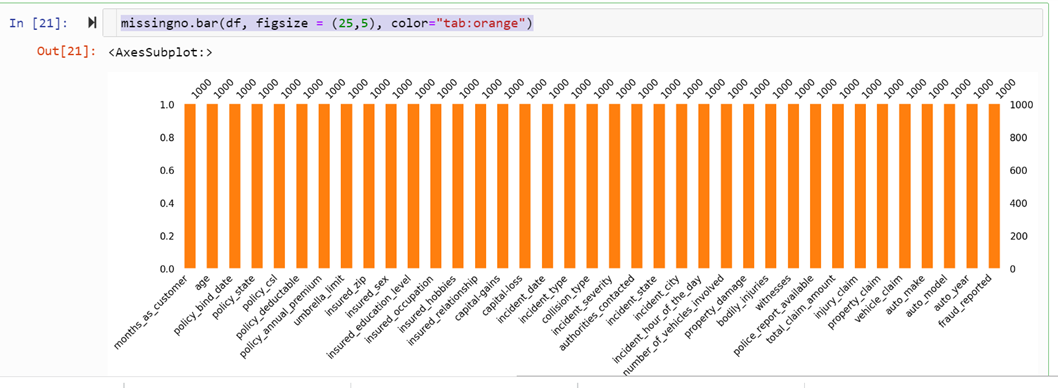
Now, we will explore the data with some basic steps and then further proceed with some crucial analysis, like feature extraction, imputing, and encoding.

* Let's start with checking shape, unique values, value counts, info, etc.
* After analyzing if we find any unnecessary columns in the datasets, we can use the drop command to drop those columns.





* After doing this basis analysis, now we are checking for the null values and further will mention all the observations.

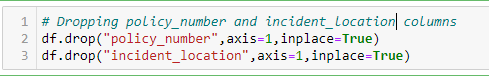


## Observations :

1. As we can see the dataset does not have any null values in it.
2. dataset contains 3 different types of data namely integer data type, float data type, and object data type.
3. after analyzing we can observe that the c\_39 column has only NaN entries. Keeping all entries NaN is useless for evaluation, hence we are dropping that column.



1. it can observe that column **policy number** and **incident location** have 1000 unique values which mean they have only one value count. So it is not required for our analysis so we can drop it.



1. by looking at the value counts of the particular column we can realize that the columns **umbrella limit, capital-gains**, and **capital-loss** contain more zero values around 79.8%, 50.8%, and 47.5%. we will be keeping the zero values in the capital gains and capital loss columns as it is. Since the umbrella limit columns have more than 70% of zero values, let's drop them as it will create no impact on the analysis.

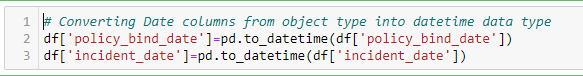


1. the column **insured zip** is the zip code assigned to each person. If we take a look at the value count and unique values of the column **insured zip,** it contains 995 unique values which means the 5 entries are repeating. Since it is giving some information about the person, either we can drop this or we can convert its data type from integer to object for better processing.

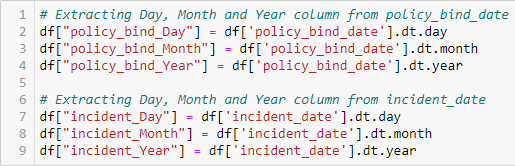


## Proceeding to Feature Extraction:

policy\_bind\_date and incident\_date have object data types that should be in the DateTime data type, which means python is not able to understand the type of this column and give the default data type. We will convert this object data type to the Date Time data type and we will extract the values from these columns.



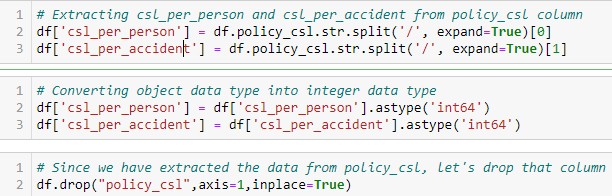
As we have converted the object data type into a DateTime data type. let's extract Day, Month, and Year from both columns.



we have extracted Day, Month, & Year columns, from the policy\_bind\_date and incident\_date columns. So we can drop these columns now.



from the features we can see that the policy\_csl column is showing as an object data type but it contains numerical data, maybe it is because of the presence of "/" in that column. So first we will extract two columns csl\_per\_person and csl\_per\_accident from policy\_csl columns and then will convert their object data type into an integer data type.

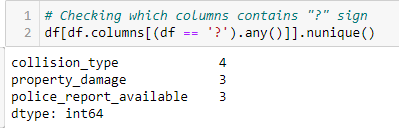


* 1. After extracting we have dropped the policy\_csl feature.
  2. we have observed that the feature 'incident-year' has one unique value throughout the column also it is not important for our prediction so we can drop this column also.



## Moving on to Imputation:

Imputation is a technique to fill null values in the dataset using mean, median, or mode. you might be thinking that we did not get any null values while checking for the null values in the dataset, however from the value counts of the columns we have observed that some columns have "?" values, they are not NAN values but we need to fill them.



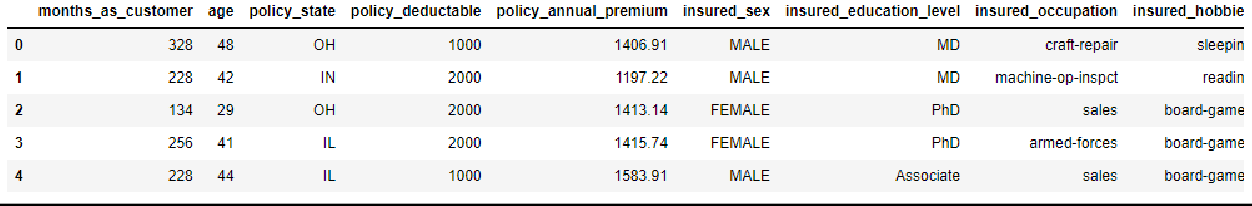
These columns contain the "?" sign. Since this column seems to be categorical so we will replace "?" values with the most frequently occurring values of the respective columns i.e. mode values.



The mode of property\_damage and police\_report\_available is "?", which means the data is almost covered by the "?" sign. So we will fill them by the second highest count of the respective column.

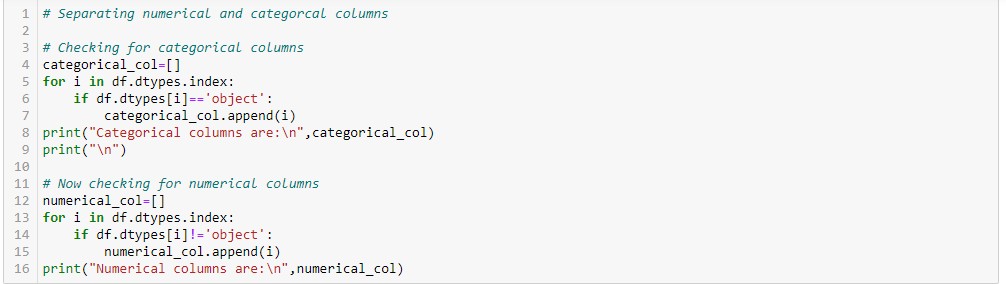


after cleaning the dataset until now, the dataset looks like this!



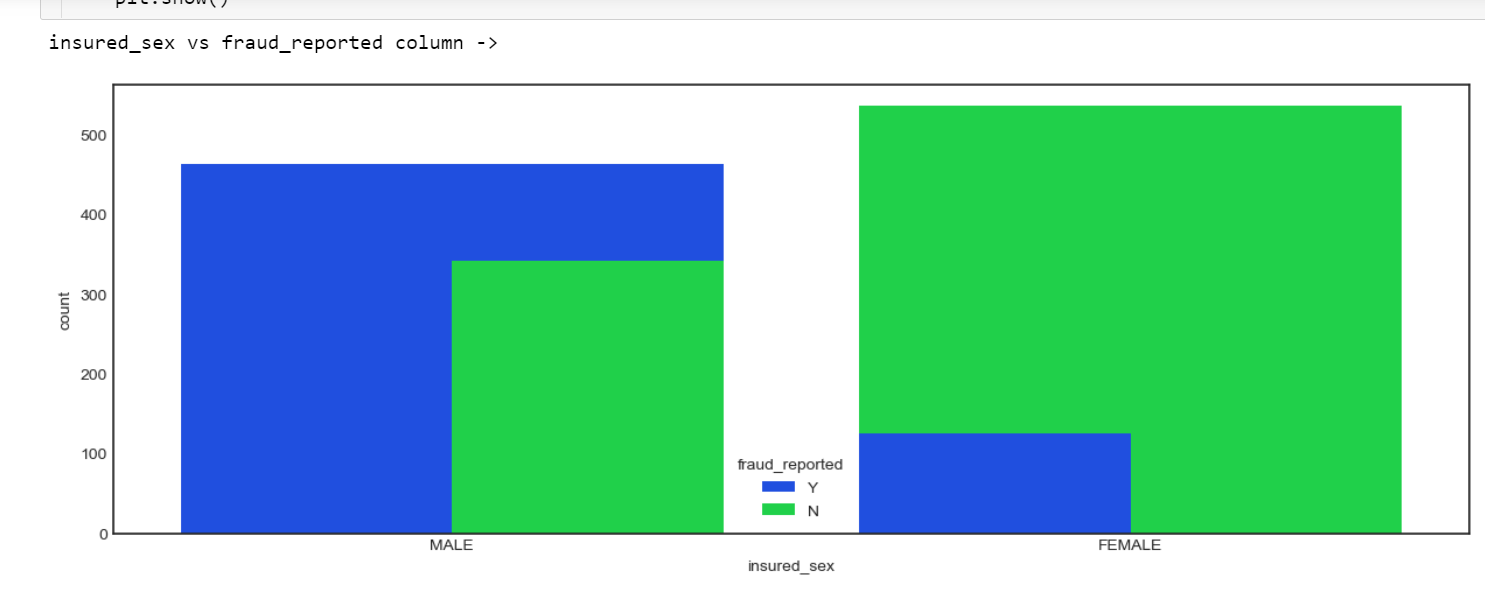
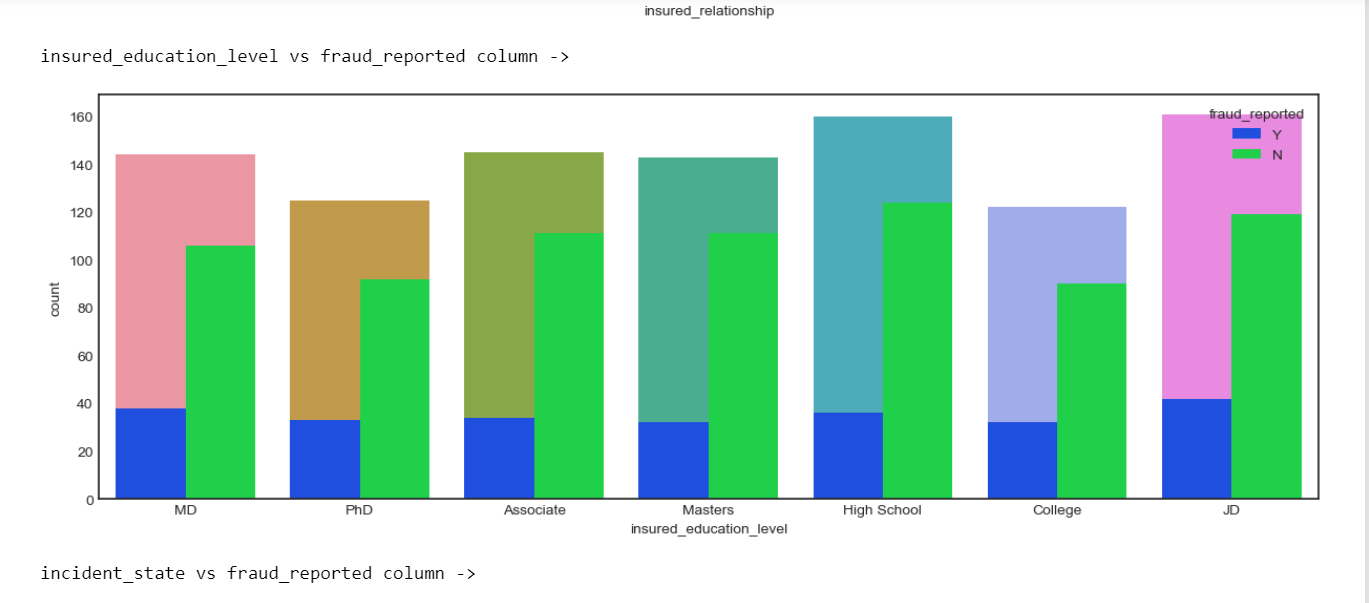
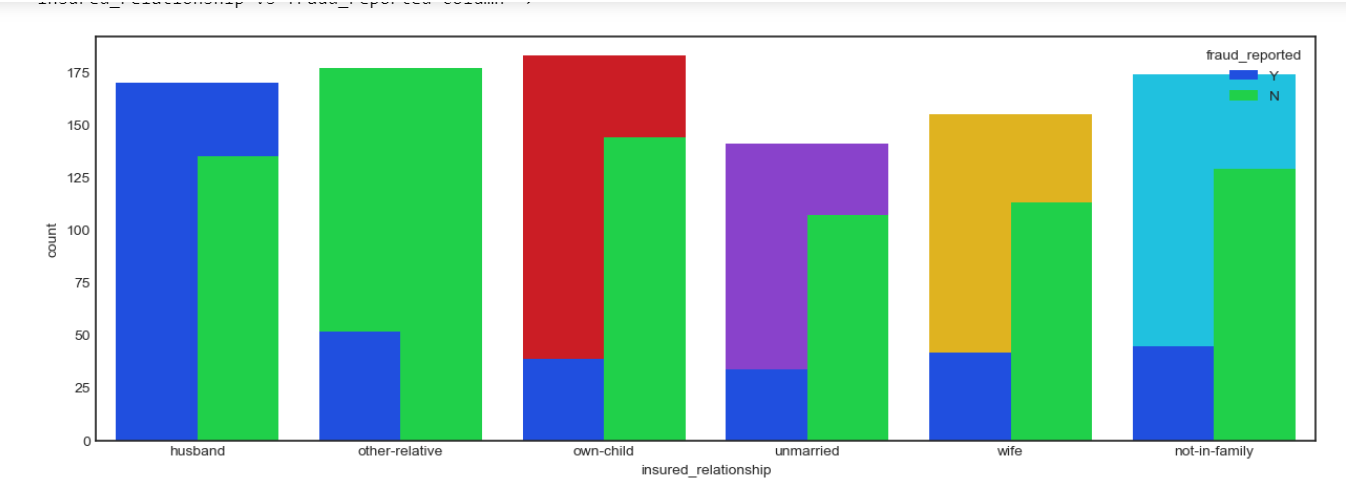
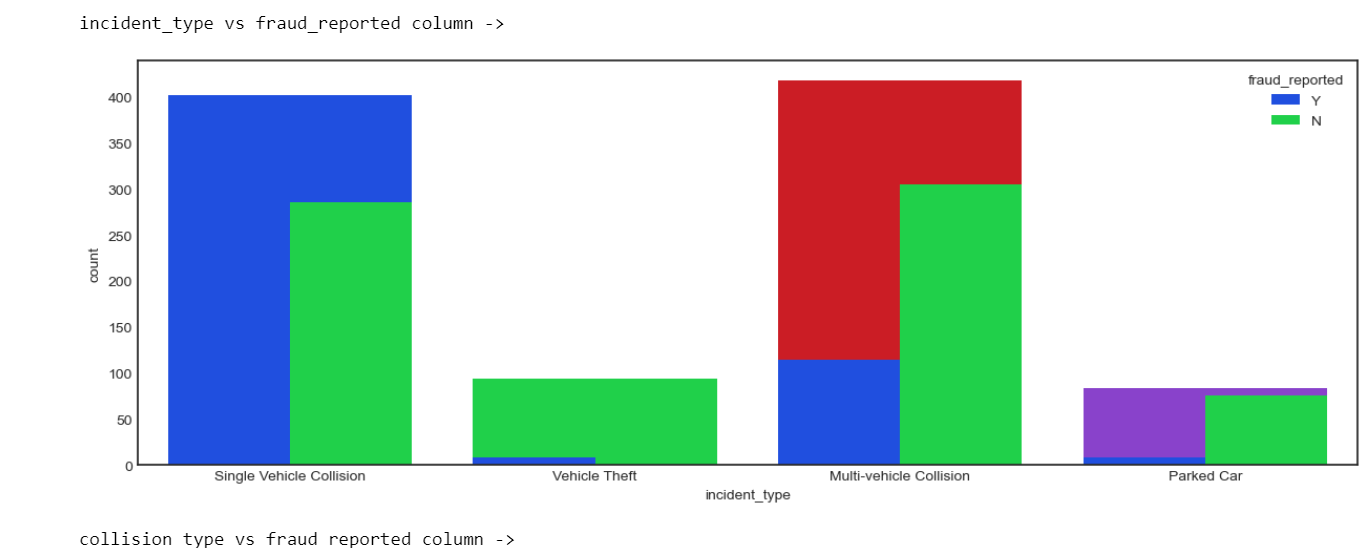
## Preparing for Visualization

we will look into the categorical and numerical columns so that we can visualize the features accordingly.



## Visualization



After looking into the plot, we can observe that the count of "N" is high compared to "Y". This means here we can assume that "Y" stands for "Yes" and that the insurance is fraudulent and "N" stands for "No" which means the insurance claim is not fraudulent. most of the insurance claims have not been reported as fraudulent. Since it is our target column, it indicates the class is imbalanced. We will balance the data using an oversampling method further.



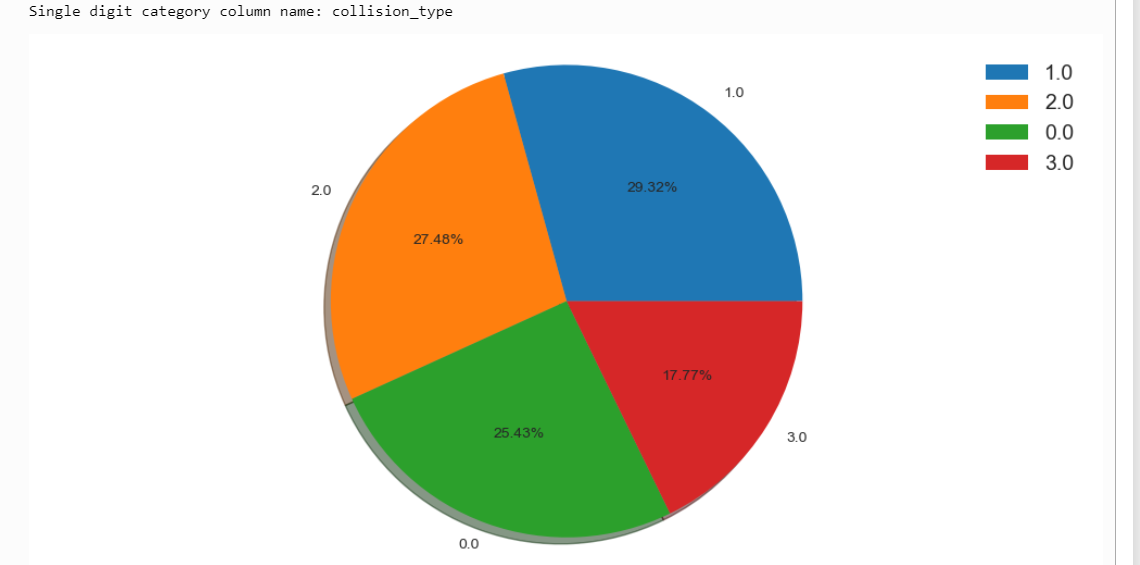
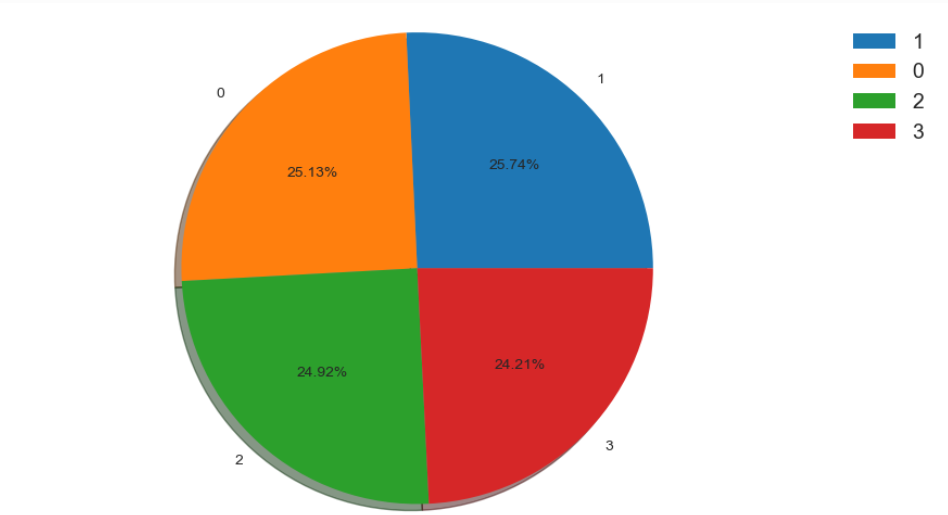
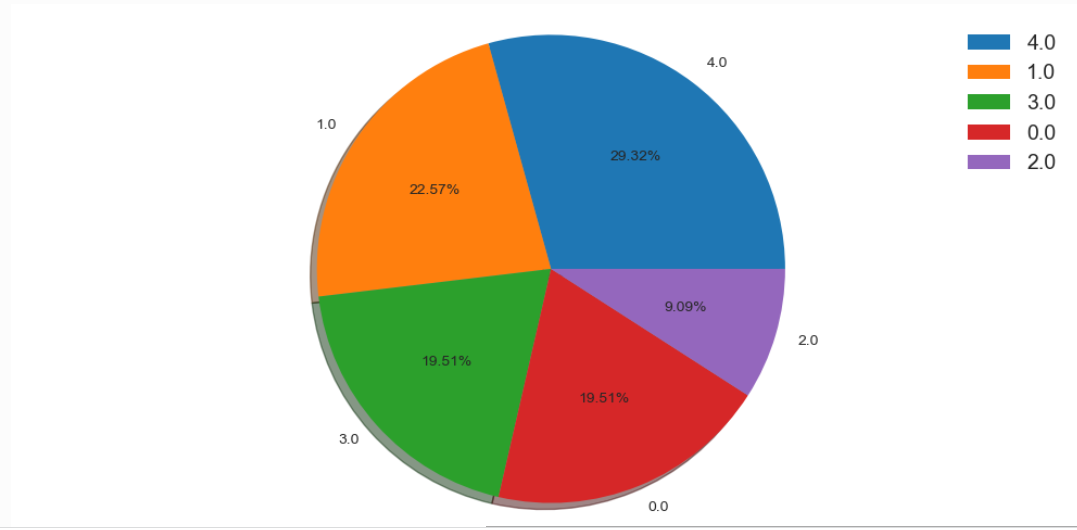
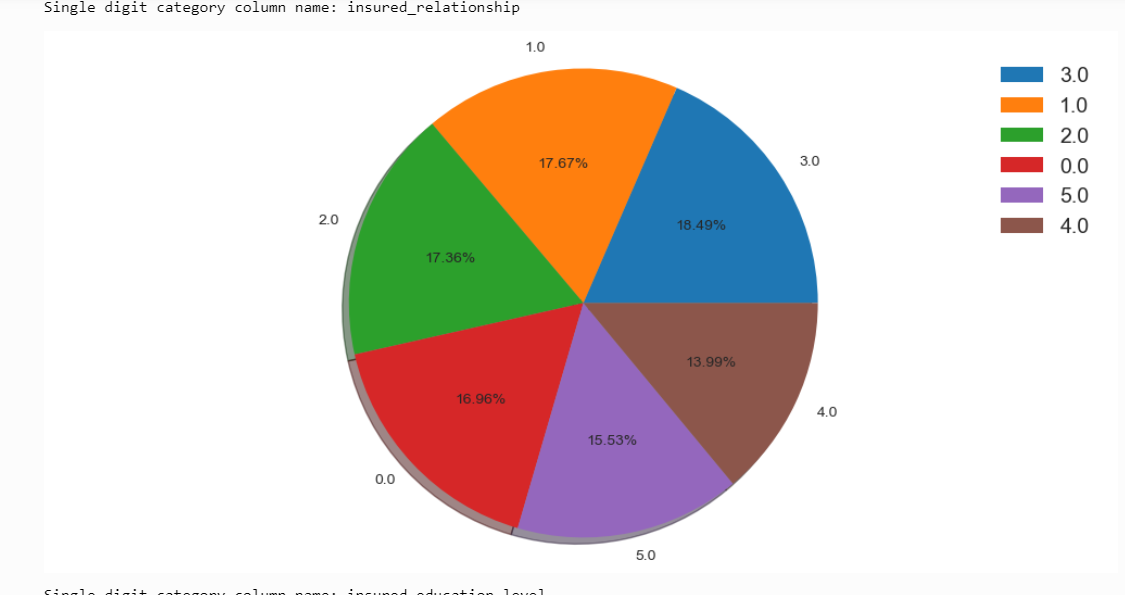
## 

## By looking into the count plots we can observe the following things:

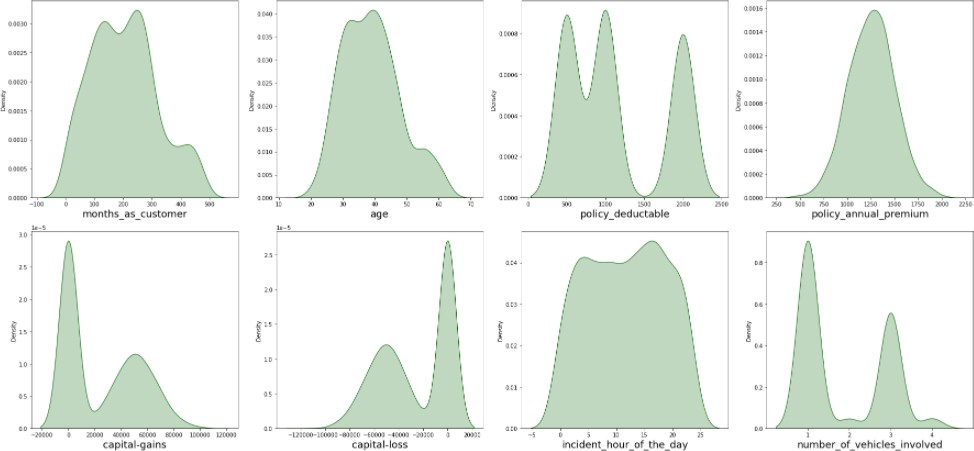
* In the insured occupation we can observe most of the data is covered by a machine operation inspector followed by a professional.
* Concerning to insured hobbies, we can notice reading covered the highest data followed by exercise. And other categories have the average counts.
* The incident severity count is high for Minor damages and trivial damage data has a very less count compared to others.
* When accidents occur most of the authorities contact the police, here the category police cover the highest data, and Fire has the second highest count. But Ambulance and Others have almost the same counts and the count is very less for none compared to all.
* Concerning to the incident state, New York, South Carolina, and West Virginia states have the highest counts. In the incident city, almost all the columns have equal counts.
* When we look at the vehicle manufactured companies, the categories Saab, Suburu, Dodge, Nissan, and Volkswagen have the highest counts.

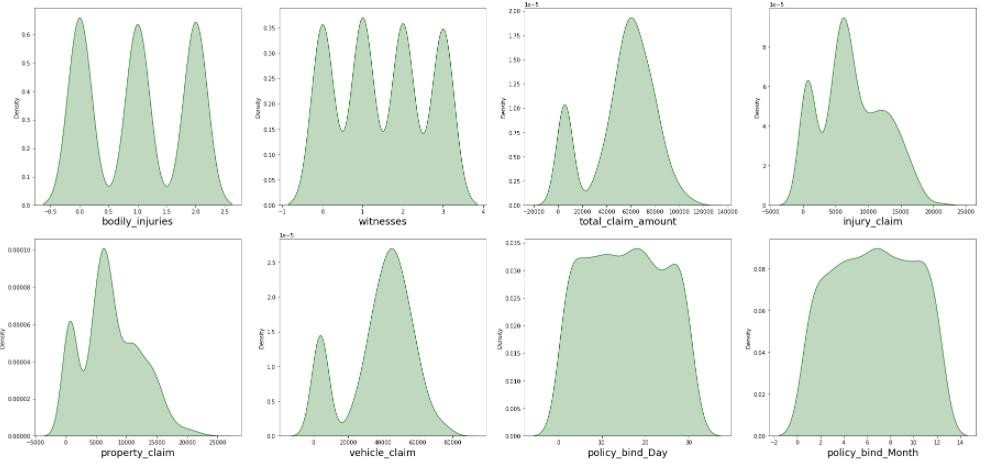
When we take a look at the vehicle models the RAM and Wrangler automobile models have the highest counts

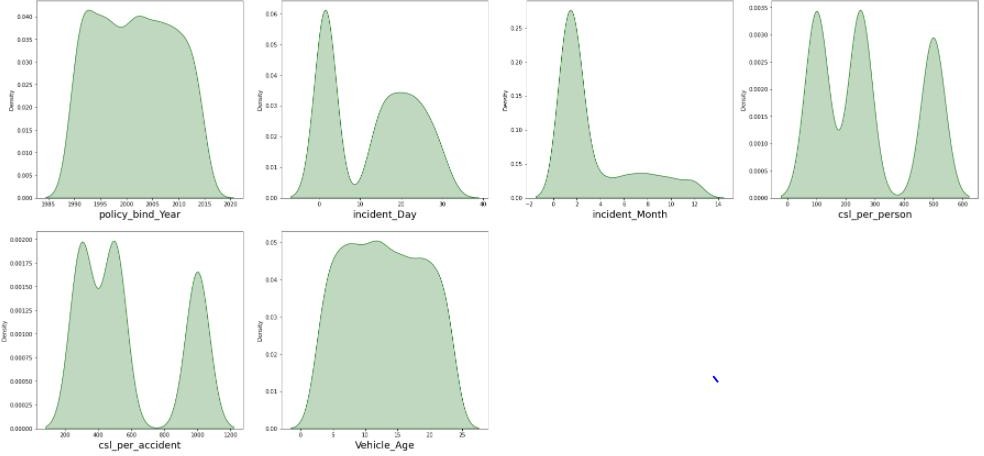


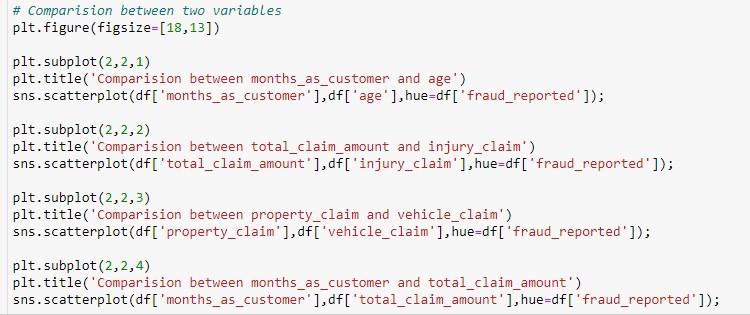


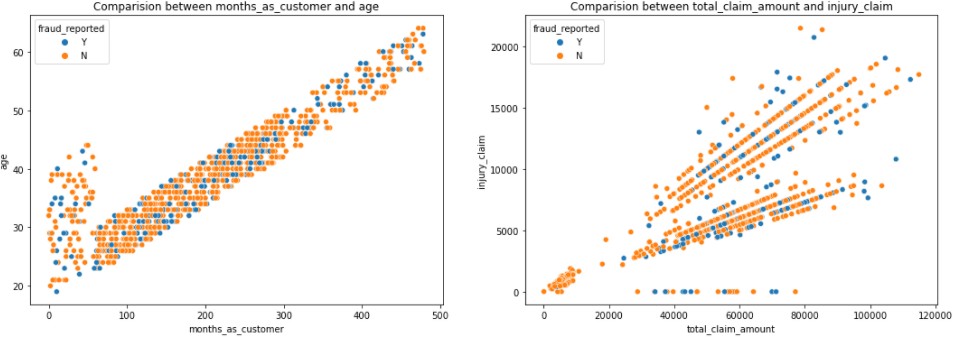


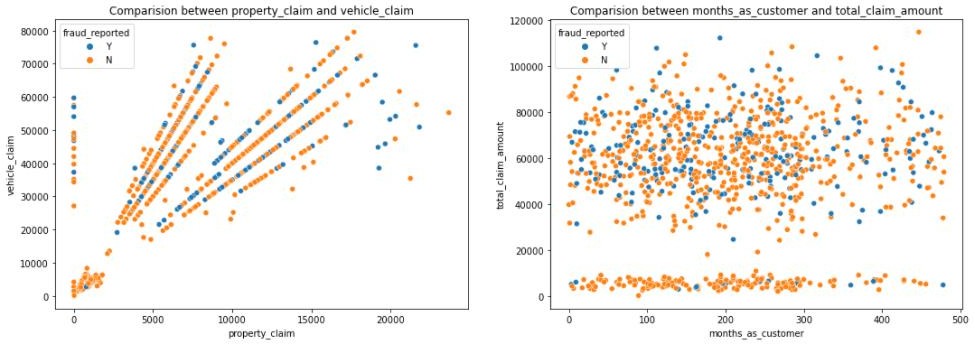




The data is normally distributed in most of the columns. Some of the columns like capital gains and incident months have a mean value greater than the median, hence they are skewed to right. The data in the column capital loss is skewed to the left since the median is greater than the mean. We will remove the skewness using appropriate methods in the later part.

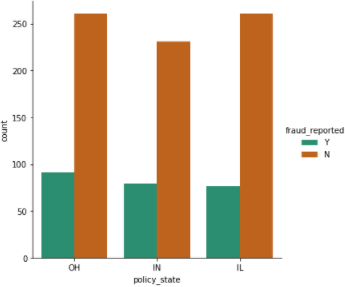


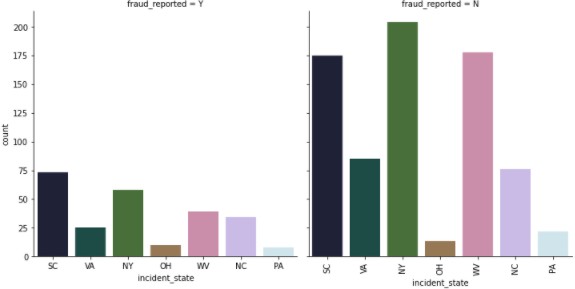
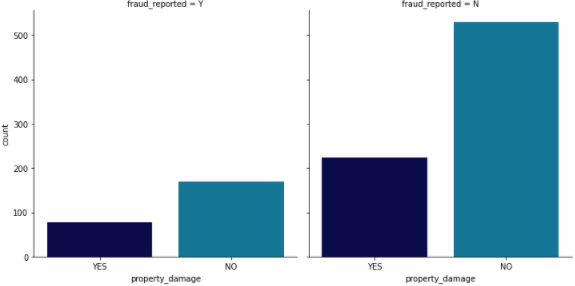




* There is a positive linear relationship between the age and month\_as\_customer column. As age increases the month\_as customers also increases, also the fraud reported is very less in this case.
* In the second graph, we can observe the positive linear relation, as the total claim amount increases, an injury claim also increases.
* The third plot is also the same as the second one as the property claim increases, vehicle claim is also increased.
* In the fourth plot, we can observe the data is scattered and there is not much relation between the features.

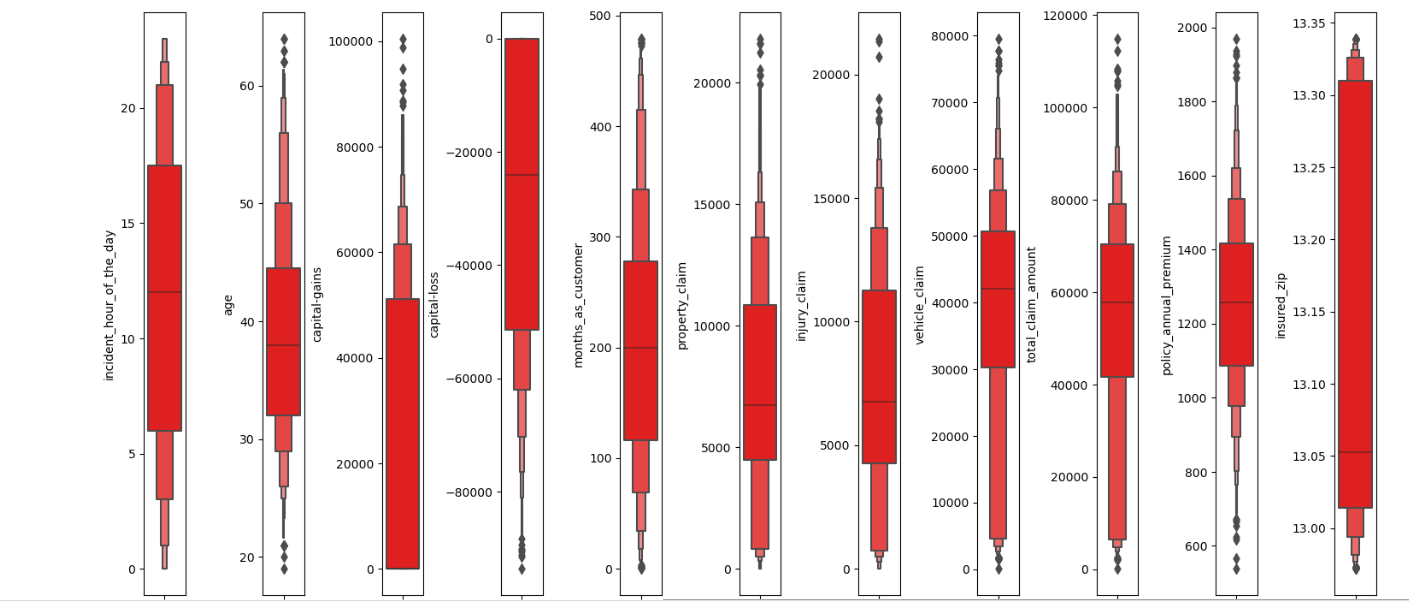
Visualization is a technique whereby comparison and plotting of the data become self-explanatory, which we have seen until now. Moving ahead with some more visualization plots before we can proceed to model building.





Now we have done the visualization to analyze and understand the data. So in this EDA part, we have looked into various aspects of the dataset, like looking for the null values and imputing, extracting date time, observing the value counts, doing the feature extraction, etc. Now we will be performing another analysis by identifying the outliers and removing them. Along with it we will also look for the skewness of the dataset and remove the skewness.

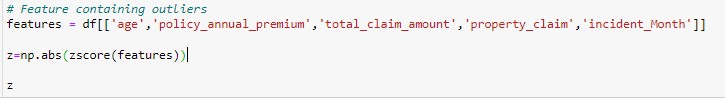
## Identifying the Outliers and Skewness



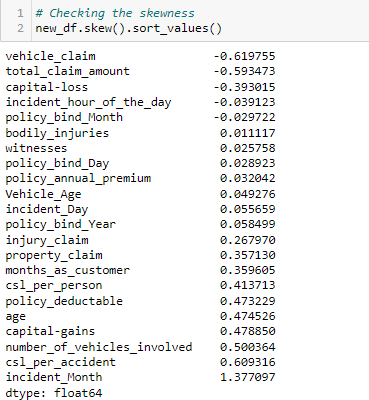
we have used a box plot to identify the outliers and we can find the outliers in the following columns:

* Age
* policy\_annual\_premium
* total\_claim\_amount
* property\_claim
* incident\_month

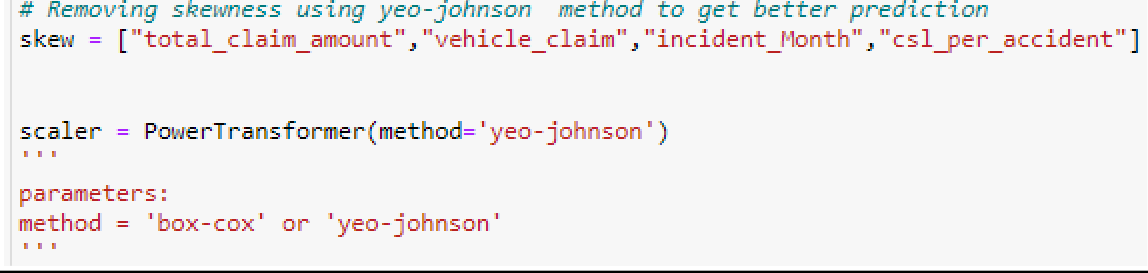
These are the numerical columns that contain outliers hence removing the outliers in these columns using the Z-score method.



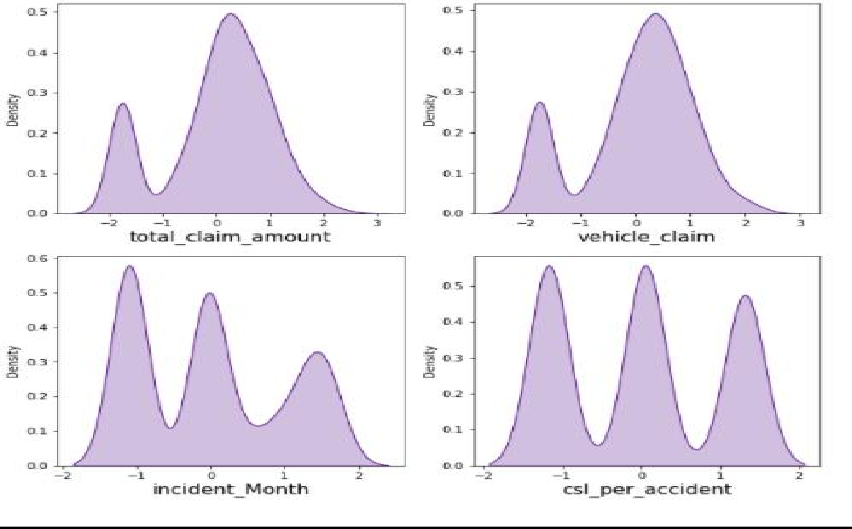
Now that we have removed the outliers, I will proceed to look into the skewness of the data and then remove it.



As we can see that skewness is present in the dataset, hence I am using the yeo-johnson method to remove the skewness.



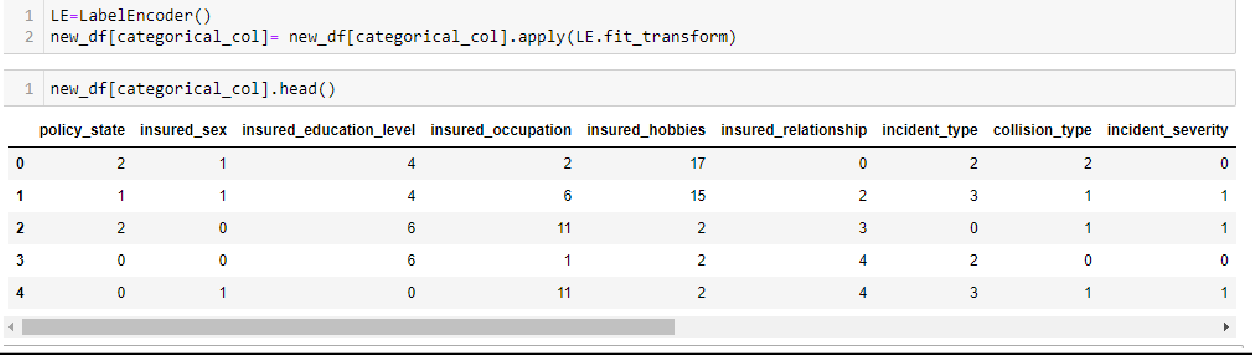
* Now we have removed the skewness and the data looks normally distributed.



Now we have completed our analysis of the dataset and also cleaned the data so that we can build a model.

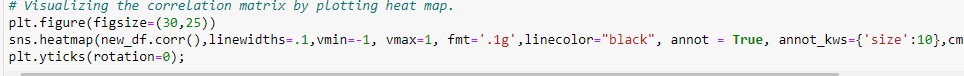
We have seen above that some problems still exist in the dataset. We have seen that the dataset has both numerical and categorical data. The model only understands numerical data; hence we will encode the data. we have seen that there can be some multi-collinearity, which we will see through a heatmap and also further try to remove it. Again we have also seen that the target variable is imbalanced, hence will fix it by oversampling. And finally, we will scale the data so that it is ready to be trained and tested.

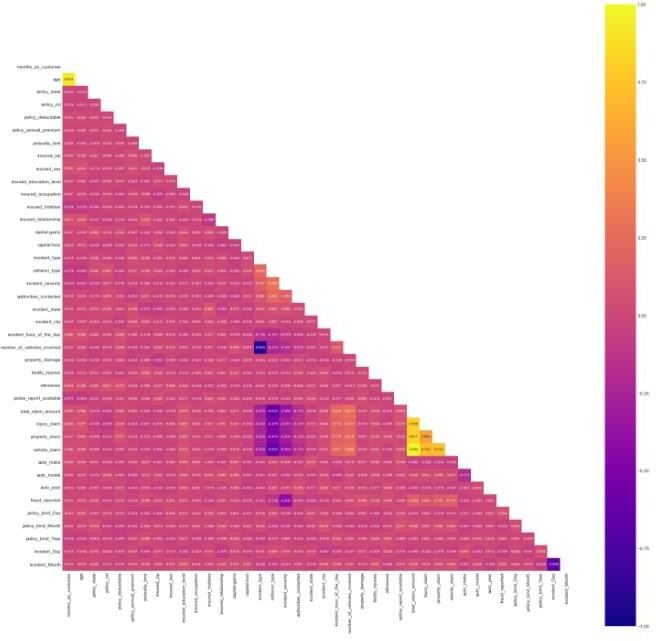
## Encoding the data



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

Moving forward, to check the co-relation between the feature and target and also the relation between the features using the heatmap.



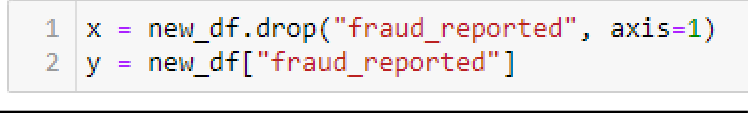


This heatmap shows the correlation matrix by visualizing the data. We can observe the relation between one feature to another.

There is very less correlation between the target and the label. We can observe that most of the columns are highly correlated with each other which leads to the multicollinearity problem. We will check the VIF value to overcome this multicollinearity problem.

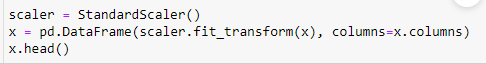
## Preprocessing Pipelines

Separating the features and label variables into x and y

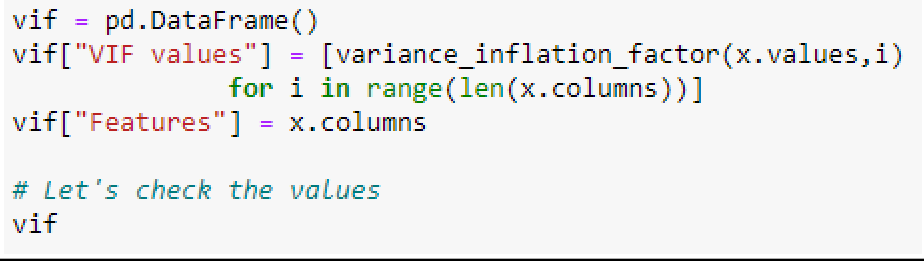


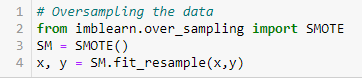
## Scaling the DataSet

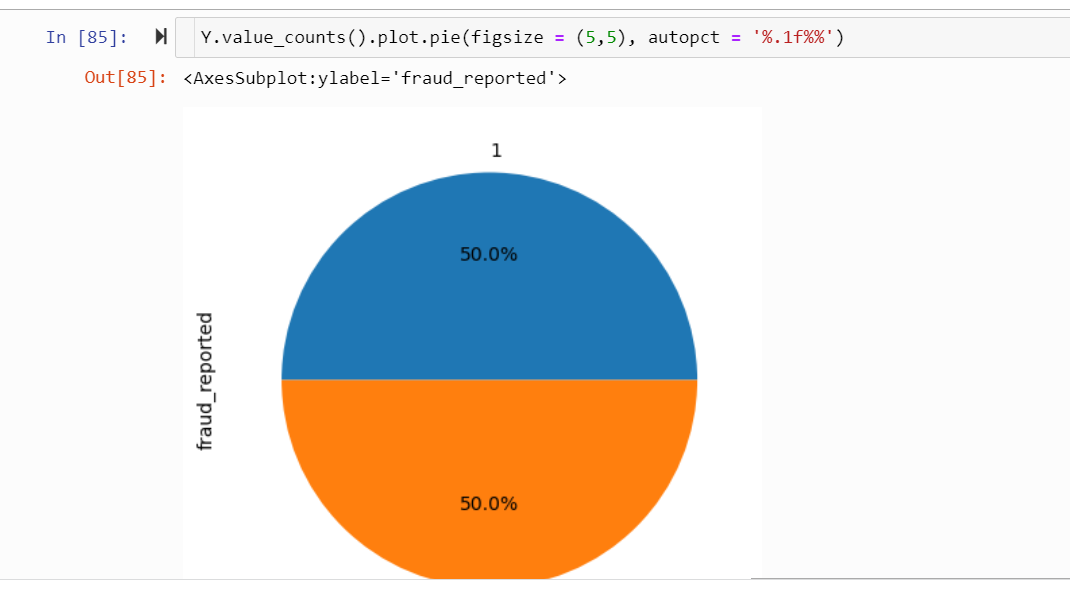
* Feature Scaling using Standard Scalarization



## Checking Multi-colinearity using VIF

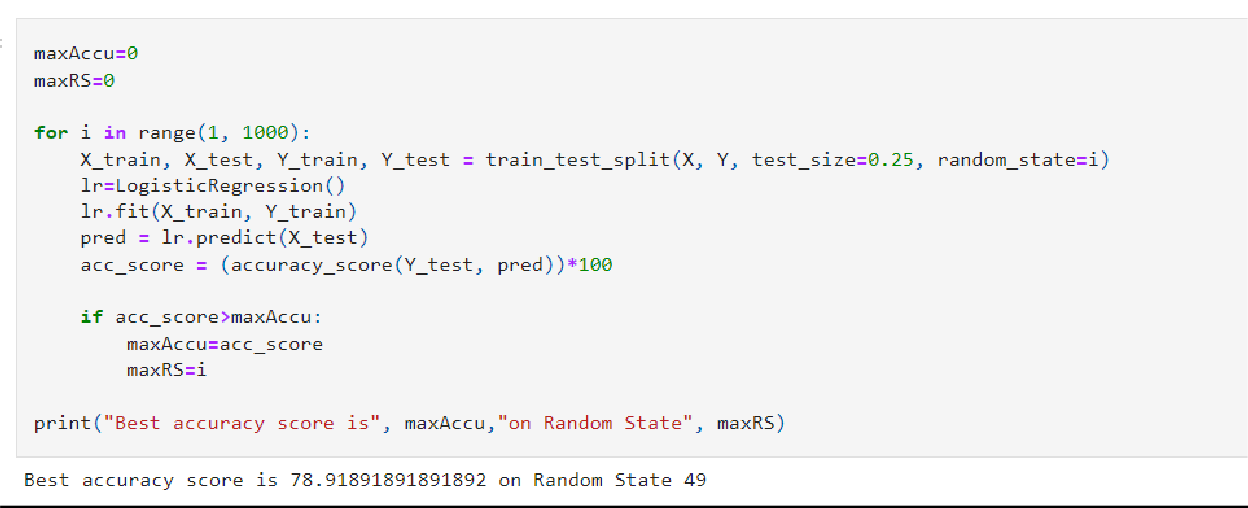


* It is observed that some columns have VIF above 10 which mean they are causing multicollinearity problem. Let's drop the feature having a high VIF value amongst all the columns.
* I have dropped the total\_claim\_amount and csl\_per\_accident features with a colinearity of more than 10, and now we have removed the problem.
* We had earlier identified another problem of imbalanced data in the target variable, let us treat it.
  + As we have treated the oversampling issue using SMOTE, now the data looks good.



* + Finally, we have got into the position where we will start building the model.
  + First, 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 uses 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.)*

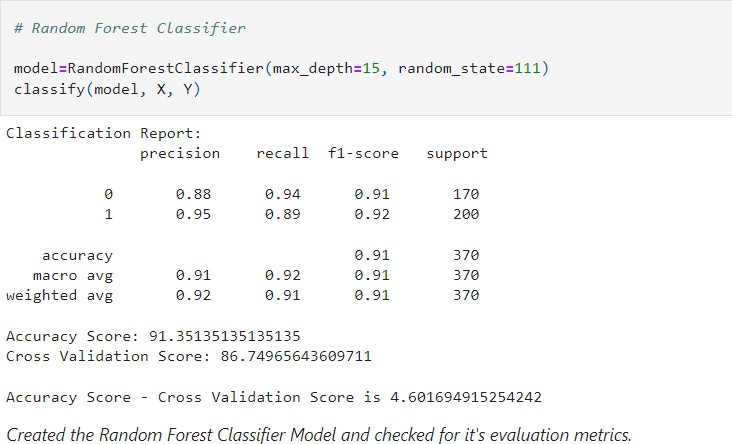


Here we have used the RandomForestClassifier to find the best random state, as we have got an accuracy score of 79% (pretty good), at the random state of 49. Let’s use this random state to build our models.

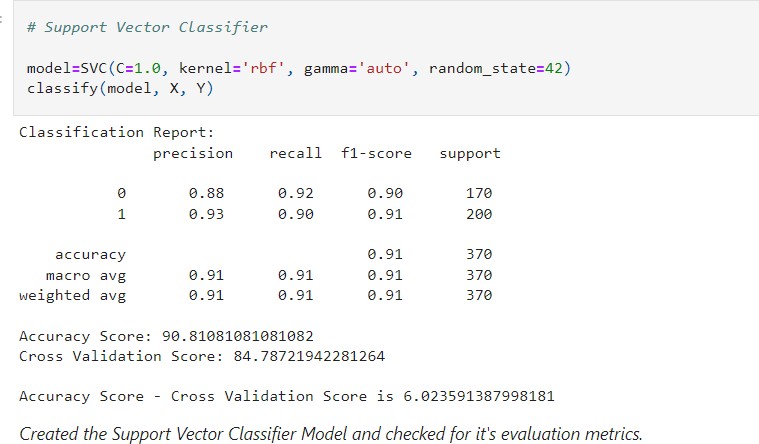
Before doing that, let us split the dataset into train and test using train\_test\_split.



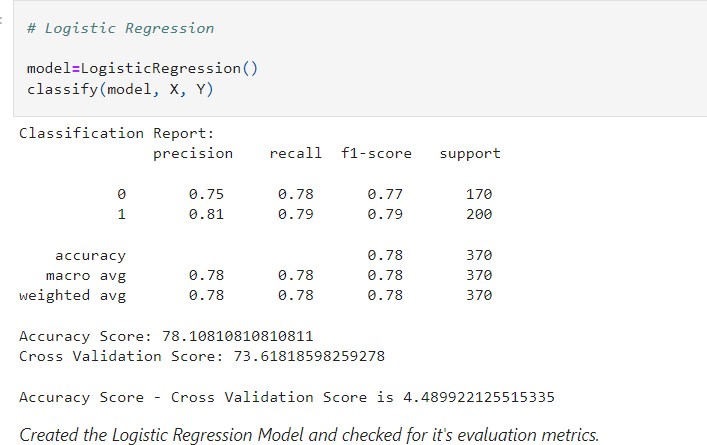
# Model Building



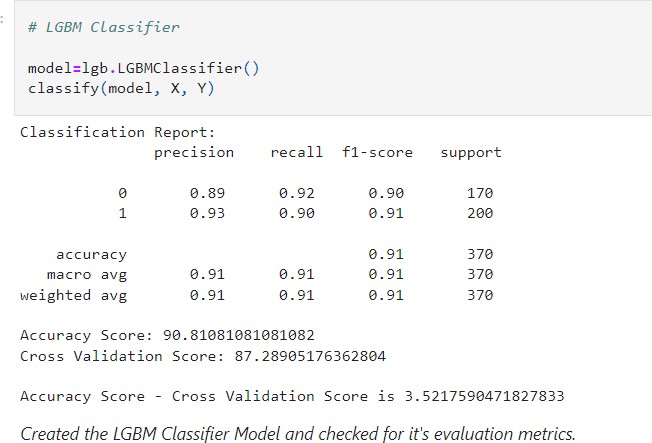
The first model was built using RandomForestClassifier, which gave an accuracy score of 91%, however, we are hungry data scientists and will not be satisfied with only one model. We will try various models and see what accuracy score we get.



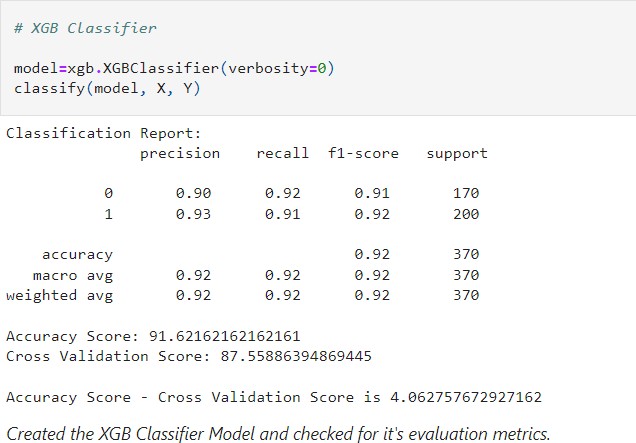
* + With Support Vector Classifier we got an accuracy score of 91%.



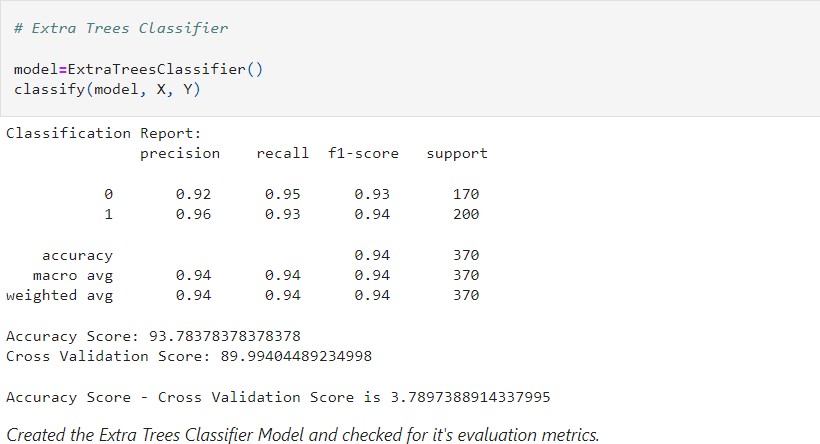
* + With logistic Regression, we got an accuracy score of 78%.



* + With LGBM we got an accuracy score of 90%.

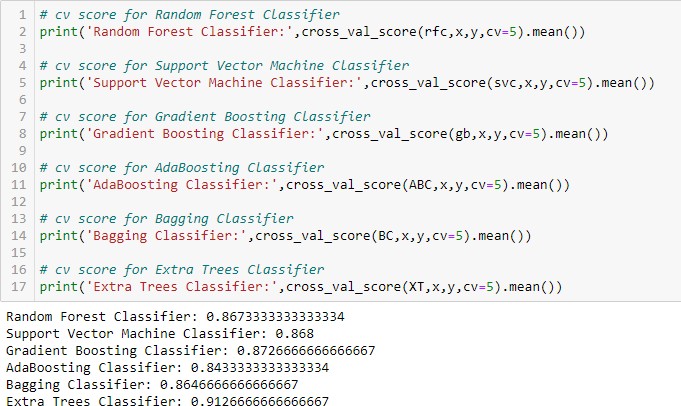


* + With XGB Classifier we got an accuracy score of 88%.

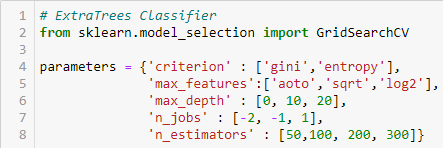


With this model, ExtraTreesClassifier we have got an accuracy score of 93%, which is better than RandomForestClassifier.

Before we can announce the best model, we always have to make sure that the model is not overfitting; hence we will perform cross-validation of all the models built.

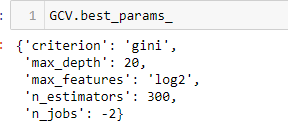


After the cross-validation, we can see that ExtraTreesClassification is the best fit model.

Now, that we have found the best fit model, let us perform some HyperParameterTuning to improve the performance of the model.

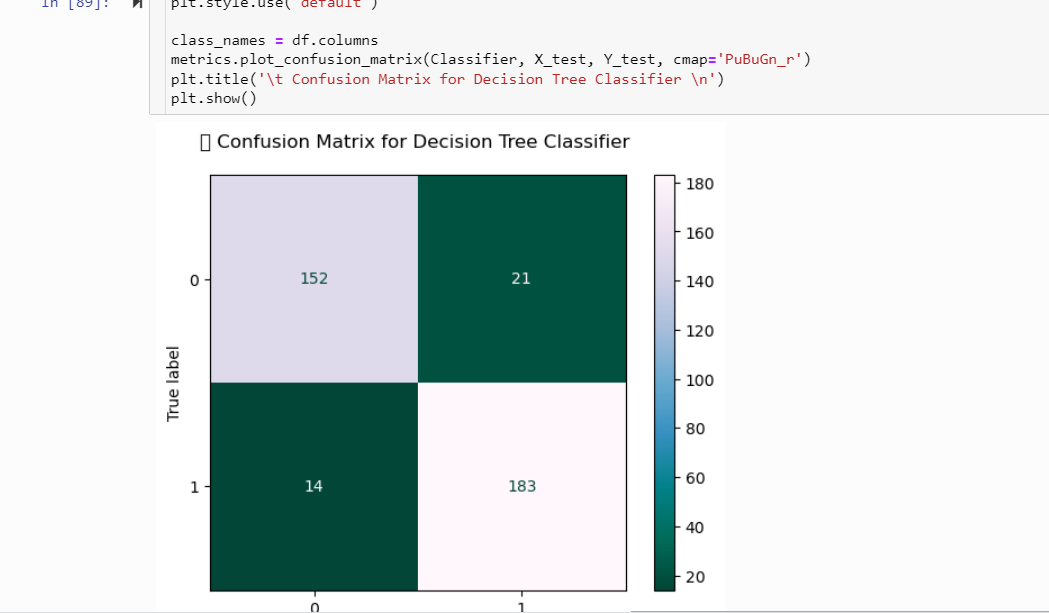




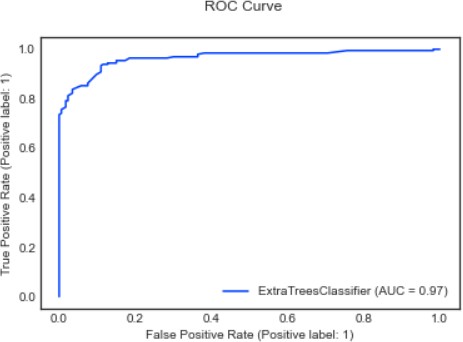


Here we have got the best parameters, and we will build our final model using these parameters.

We have built our final model and we can see that the accuracy scores have increased by 1% from the cross-validation score.

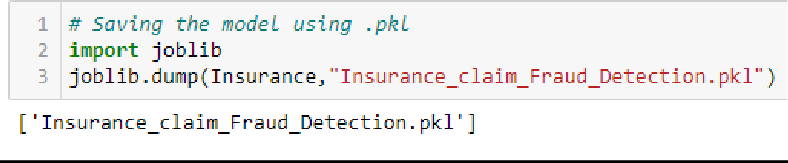


* + This is the confusion matrix for the model.
  + Plotting and AUCROC curve for the final model.

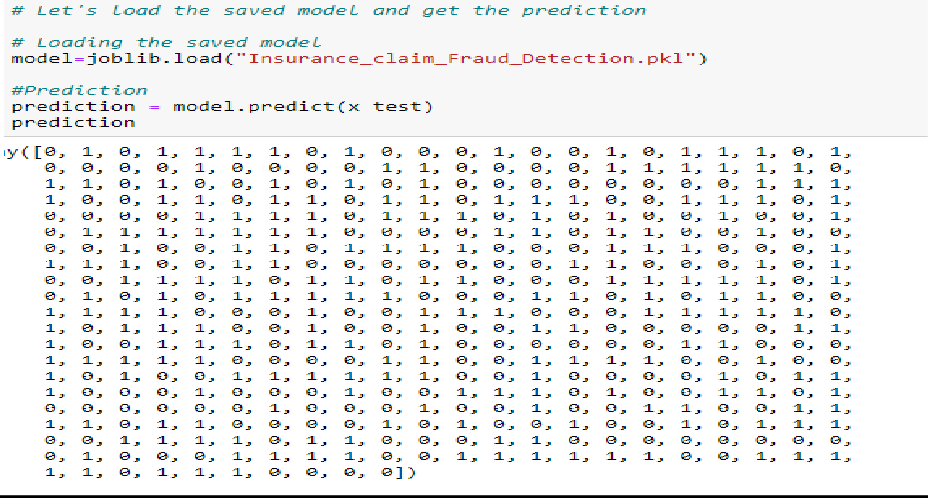


* + So here we can see that the area under the curve is quite good for this model.

**Saving the model**



**Predicting the model**



**Conclusion:-**

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

This industry area needs a good vision of 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.

Using hyperparameter tuning we can improve our model accuracy, for instance in this model the accuracy remained the same.

Using this machine Learning Model we people can easily predict whether the insurance claim is fraudulent or not and we could reject those applications which will be considered fraud claims.

## References:

* + analyticsjobs.in
  + stackoverflow.com
  + [www.iii.org](http://www.iii.org/)
  + towardsdatascience.com
  + google.com