### 

### **ARTICLE**

**DATATRAINED ACADEMY**



**Insurance Claim Fraud Detection Project**

Submitted by: V SAMYUKTA

Batch number: 1839

**INTRODUCTION**

**INSURANCE FRAUD:**

Insurance fraud is a deliberate deception perpetrated against or by an insurance company or agent for the purpose of financial gain. 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.

The total cost of an insurance fraud is estimated to be more than forty billion dollars. So, detection of an insurance fraud is a challenging problem for the insurance industry. The traditional approach for fraud detection is based on developing heuristics around fraud indicator. The auto\vehicle insurance fraud is the most prominent type of insurance fraud, which can be done by fake accident claim.

In this article, we will focus on detecting the auto\vehicle fraud by using, machine learning technique. Also, the performance will be compared by calculation of confusion matrix. This can help to calculate accuracy, precision, and recall.

**PROBLEM STATEMENT :**

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.

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

Let’s take step by step approach to make a supervised machine learning model.

**Data Analysis:**

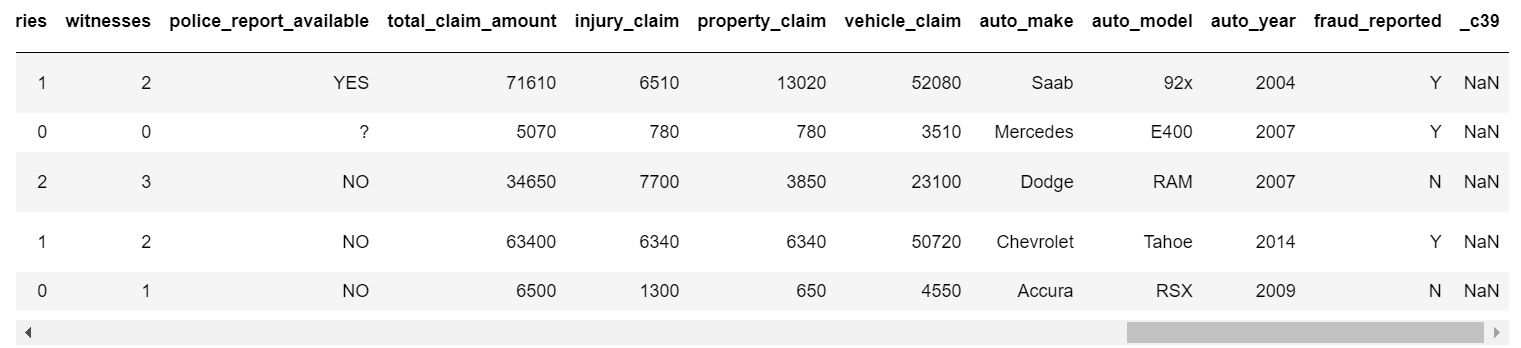
Firstly, we will import the dataset. The dataset has 40 attributes(columns)and 1000 rows. The target column is ‘fraud\_reported’ and it has categorical data with object datatype.

Since, our target is a categorical column with string entries, we will treat this problem is as a **Classification problem** and apply classification algorithms to build the model.

*# To import dataset*

df **=** pd**.**read\_csv('https://raw.githubusercontent.com/dsrscientist/Data-Science-ML-Capstone-Projects/master/Automobile\_insurance\_fraud.csv')

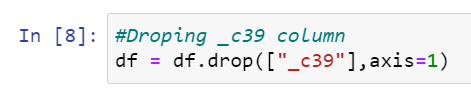
df

****

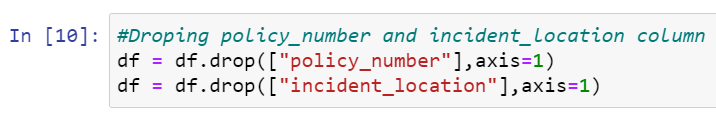
In dataset we have both numerical and categorical columns with some unnecessary entries. So, we have to clean the data.

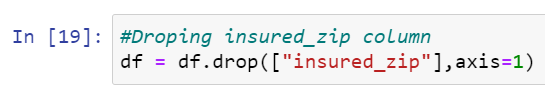
**Data cleaning and preparation:**

* To prepare the data for analysis firstly we have to check for data shape, null values, nunique values, value counts, info etc.
* There are no null values in the dataset.
* By checking the value counts we can find lot of information about each column and will be able to identify unnecessary columns in the dataset that we can be dropped.
* I found \_c39 column has all null values. Keeping all entries NaN is of no use. So, let’s drop that column.

****

* Value counts of policy\_number and incident\_location shows that these two columns have all unique values which will not help us in model building so I have dropped them.
* Insured zip is the zip-ID given to insurance person which are unique values and this also will not help us in model building so I’m going to drop this column too.
* The value counts of umbrella\_limit column shows that there are 80% zero’s in this column so this column will create some skewness in data so better let’s drop this.

****

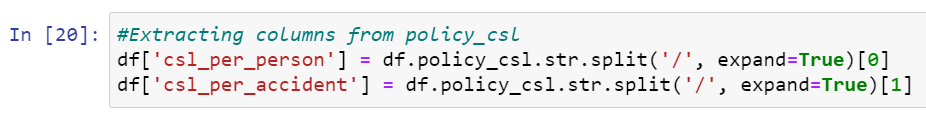
****

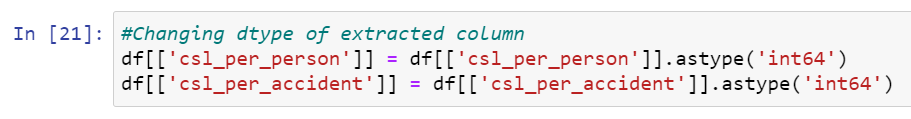
****

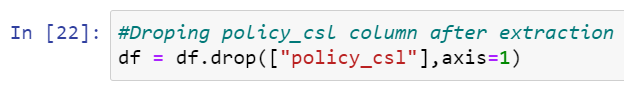
**Feature Extraction:**

* Extracted two columns csl\_per\_person and csl\_per\_accident from policy\_csl. Before policy\_csl was object type data since it was in the format (A/B), now I have extracted A and B separately (Where A and B are integers).

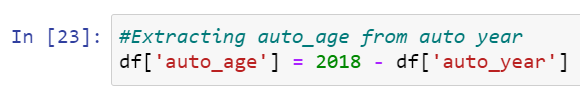
After extraction I have changed the data type from object to integer and dropped policy\_csl column to avoid multicolinearity.

****

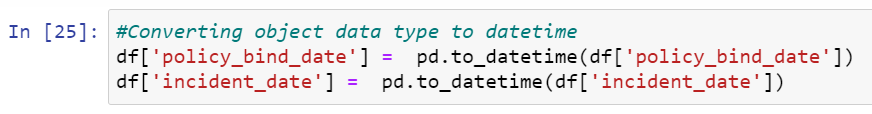
****

****

* Extracted auto\_age from auto\_year by taking difference of insurance climbed year to auto year. I felt auto age may help us more in prediction than auto year.

****

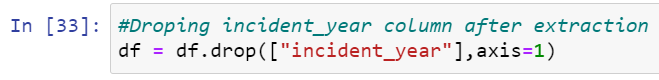
* Changed policy\_bind\_year and incident\_date datatype from object to datetime and extracted day, month and year from these columns.

****

****

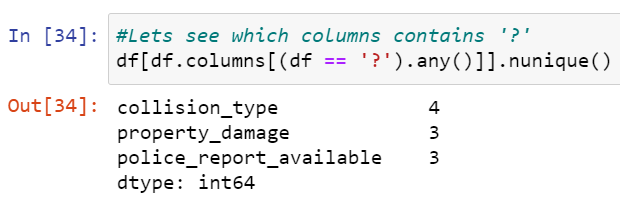
****

* The value count of extracted columns has single unique entries in incident\_year column which means all the entries in this column are same so I have dropped it.
* After extracting all the necessary columns, let’s drop old columns. If we don’t drop those columns they will behave as duplicate columns and create multicollinearity issue.

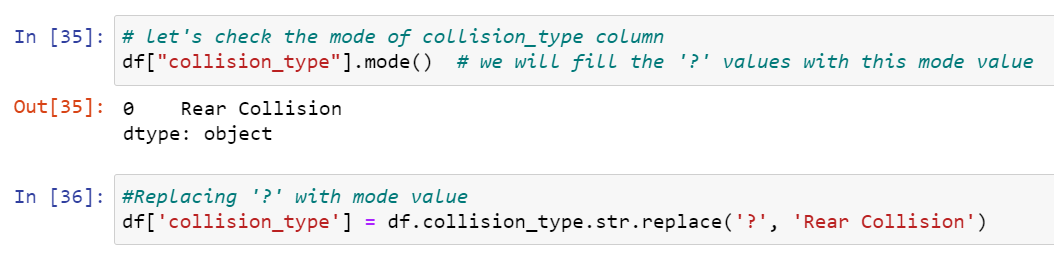
****

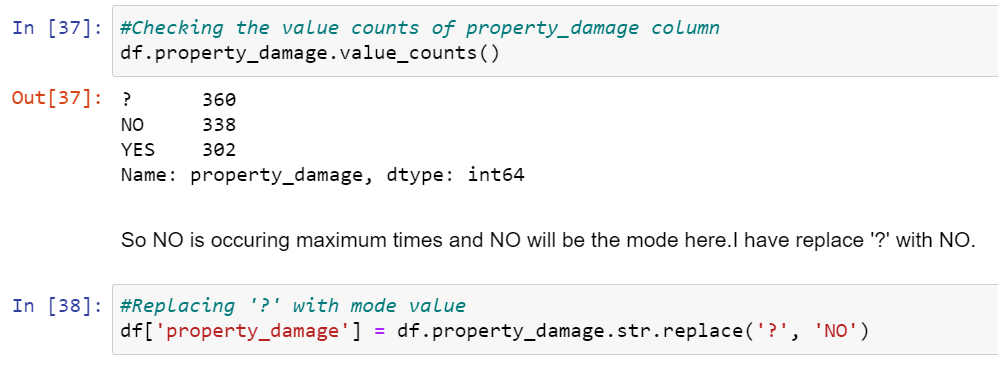
**To replace ‘ ? ’ :**

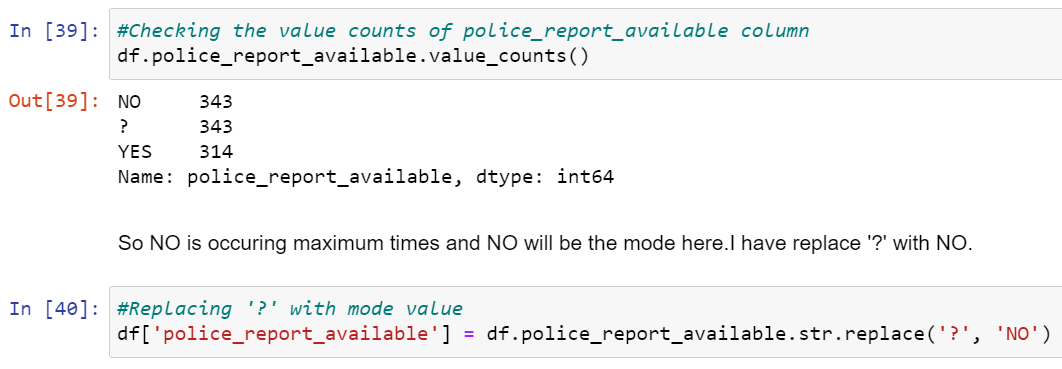
While checking value counts I have noticed some unnecessary entries in the dataset like ‘ ? ’. It may be because of some typing errors or some techinal error. So, let’s replace those unwanted entries.



* Checking for ‘ ? ’ entries in all the columns. I found these entries in 3 columns.
* As collision\_type is a categorical type column so replaced the ‘ ? ’ values with it’s mode

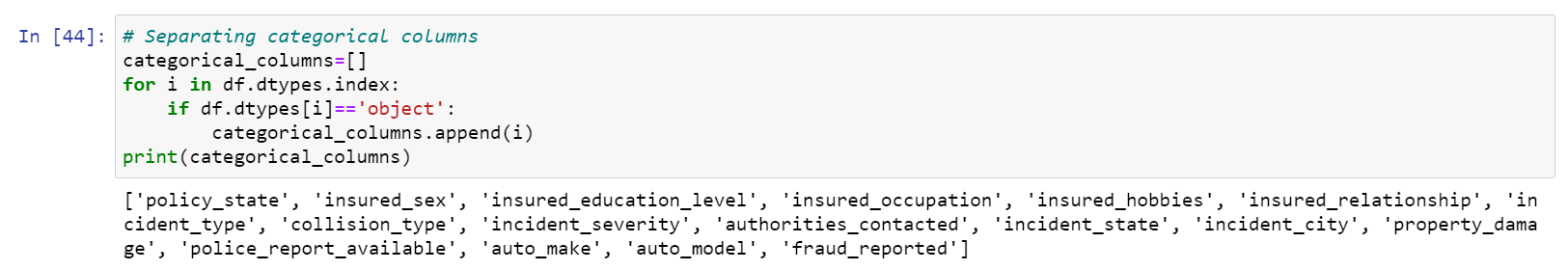


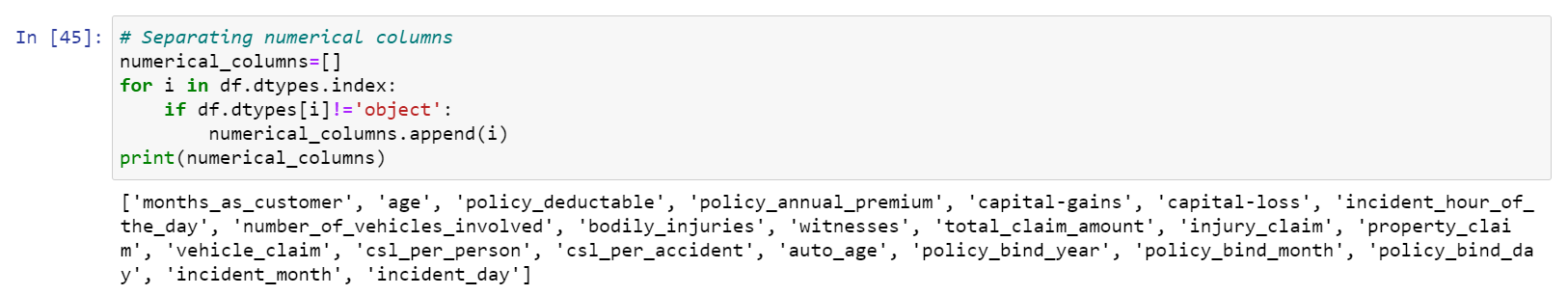




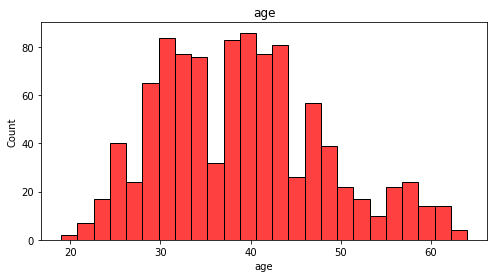
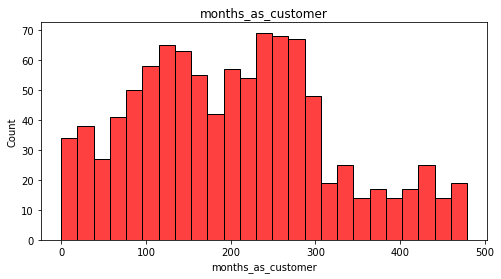
**Visualization:**

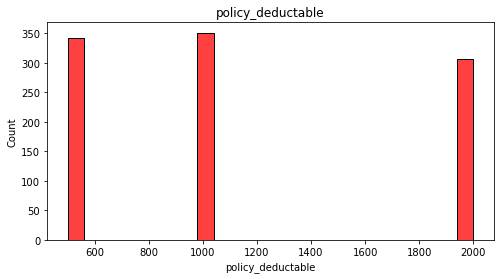
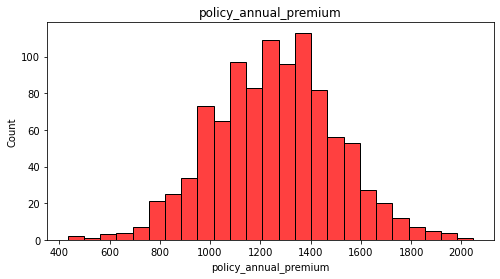
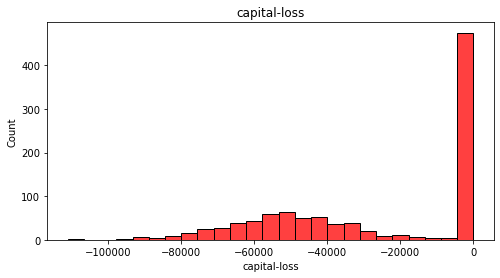
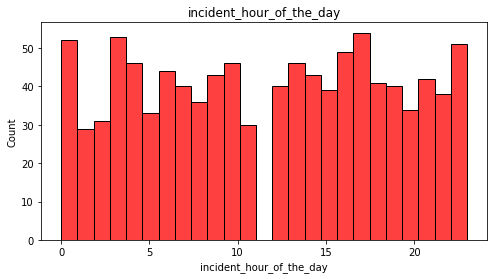
* Separate all numerical and categorical columns for visualization

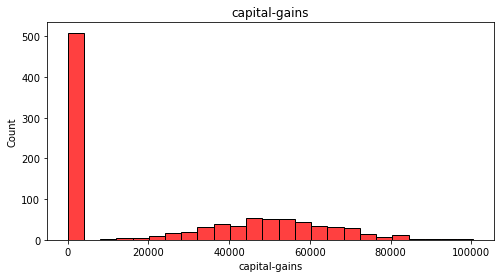
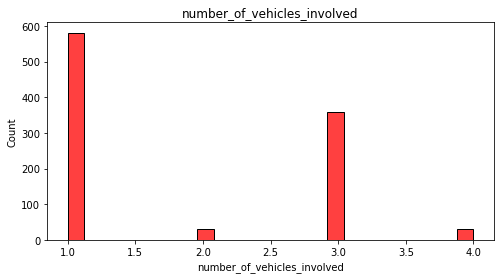
****

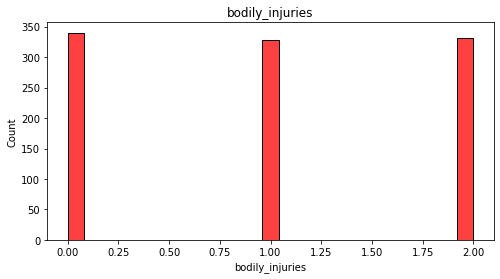
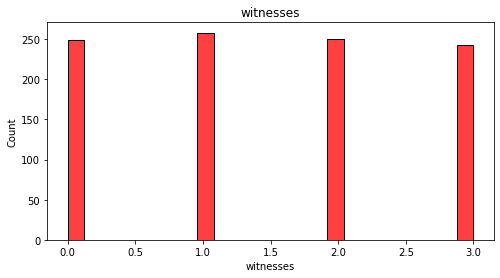
****

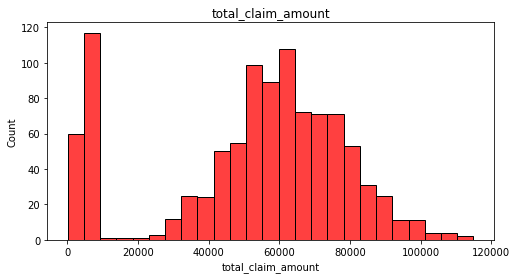
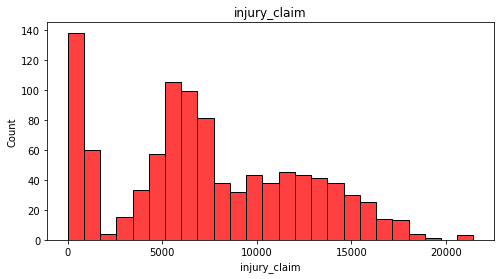
* Used histogram to visualize all the numerical columns.

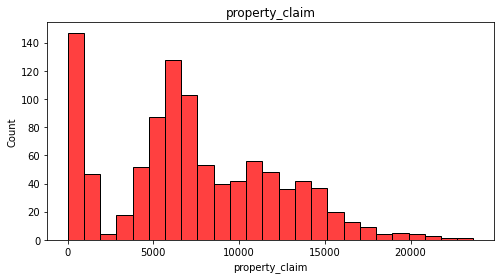
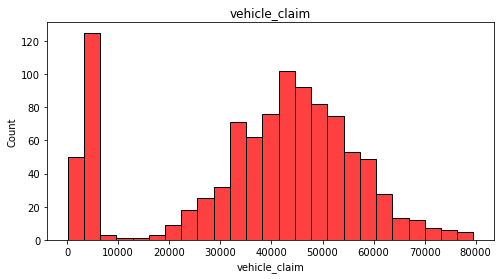
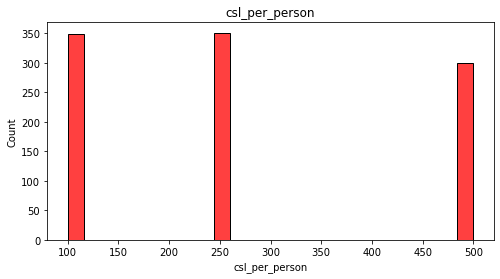
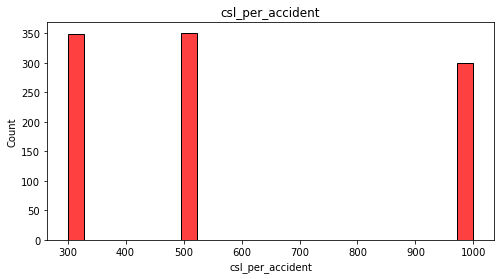


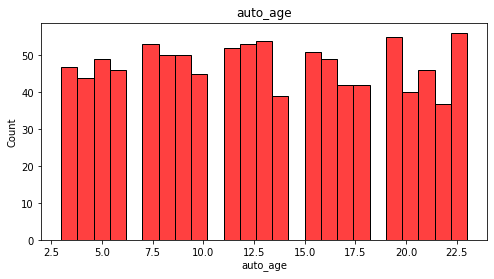
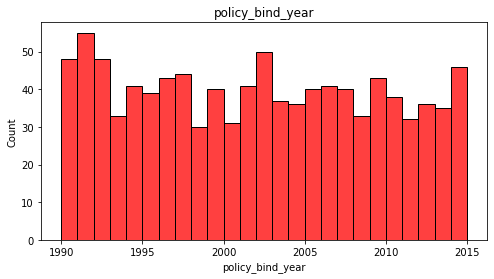
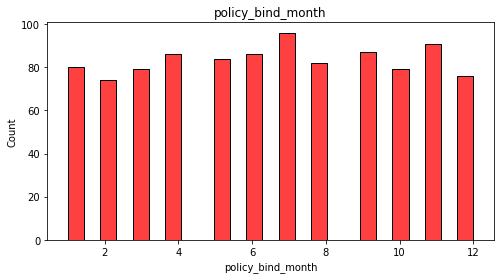
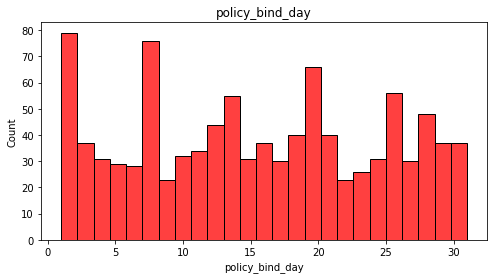
  

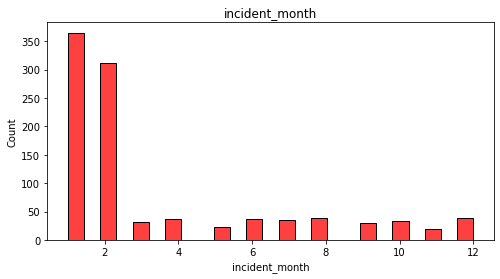
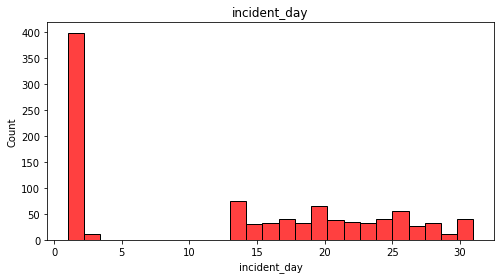
 

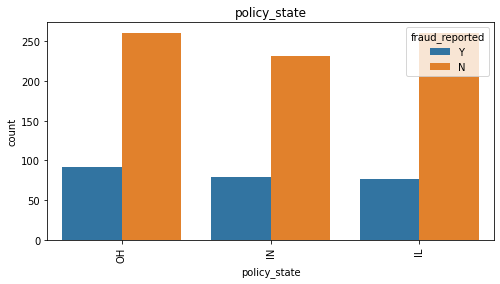
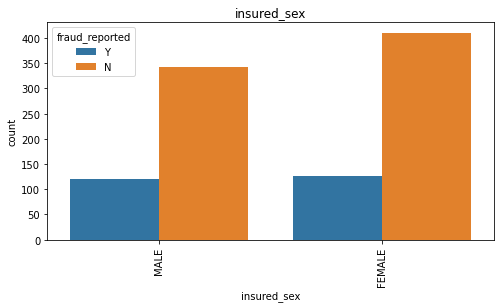
   

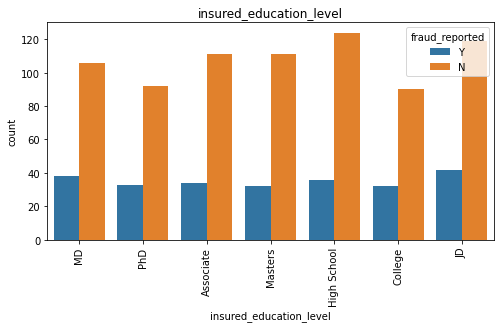
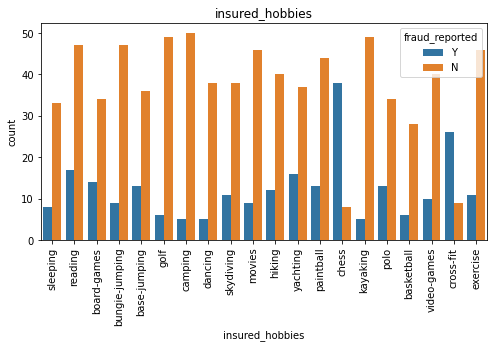
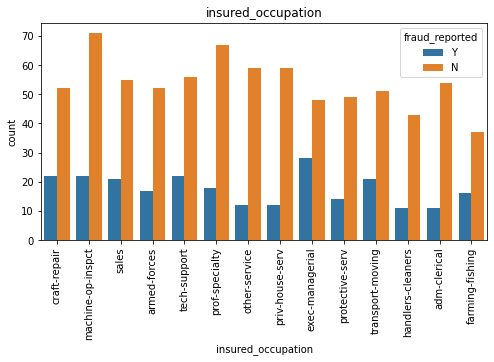
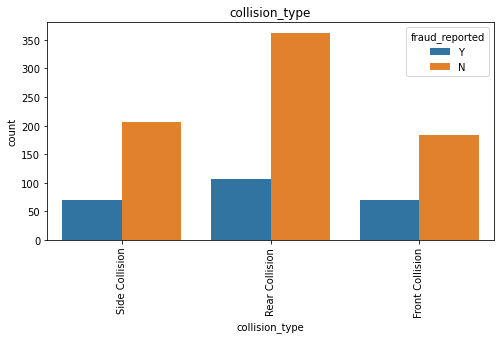
  

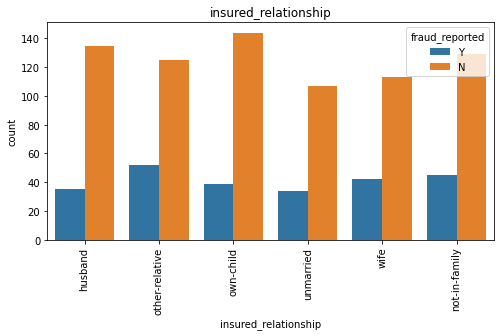
 

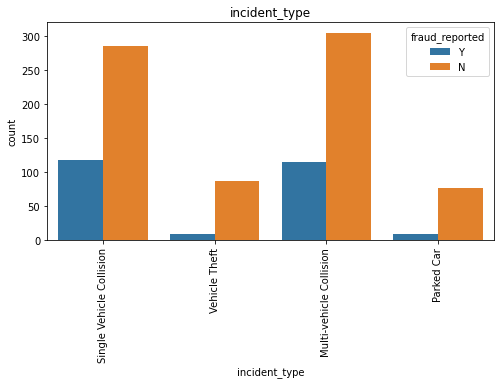
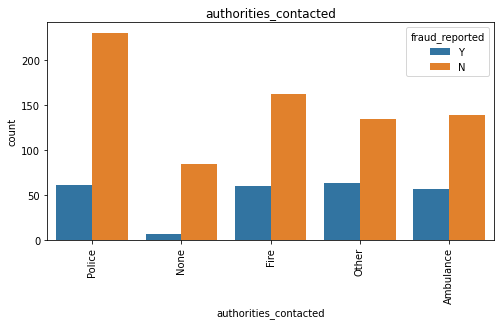
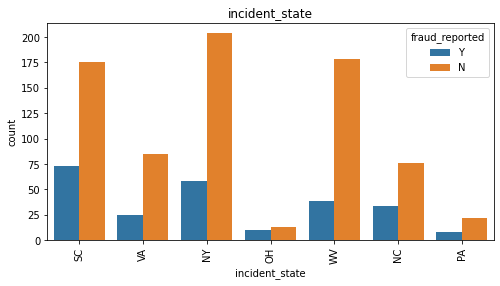
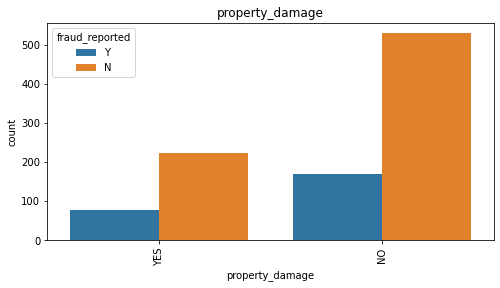
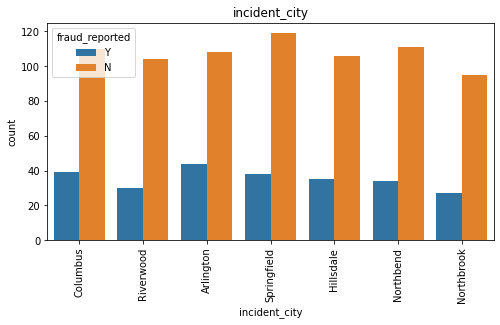
policy\_deductable, capital-gains, capital-loss, number\_of\_vehicles\_involved, bodily\_injuries, witness, total\_claim\_amount, injury\_claim, property\_claim, vehicle\_claim, incident\_month and incident\_day has some skewness which need to be removed.

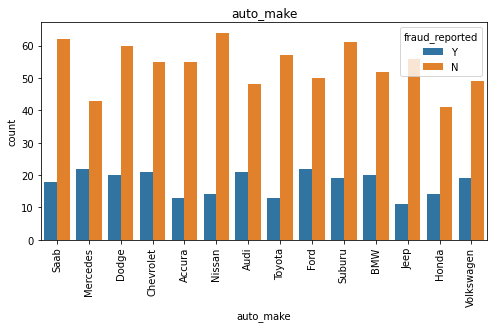
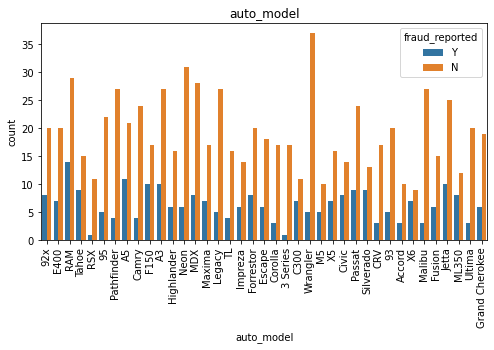
* Count plot is used to visualize categorical columns

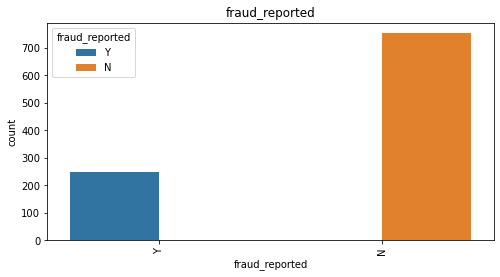
 



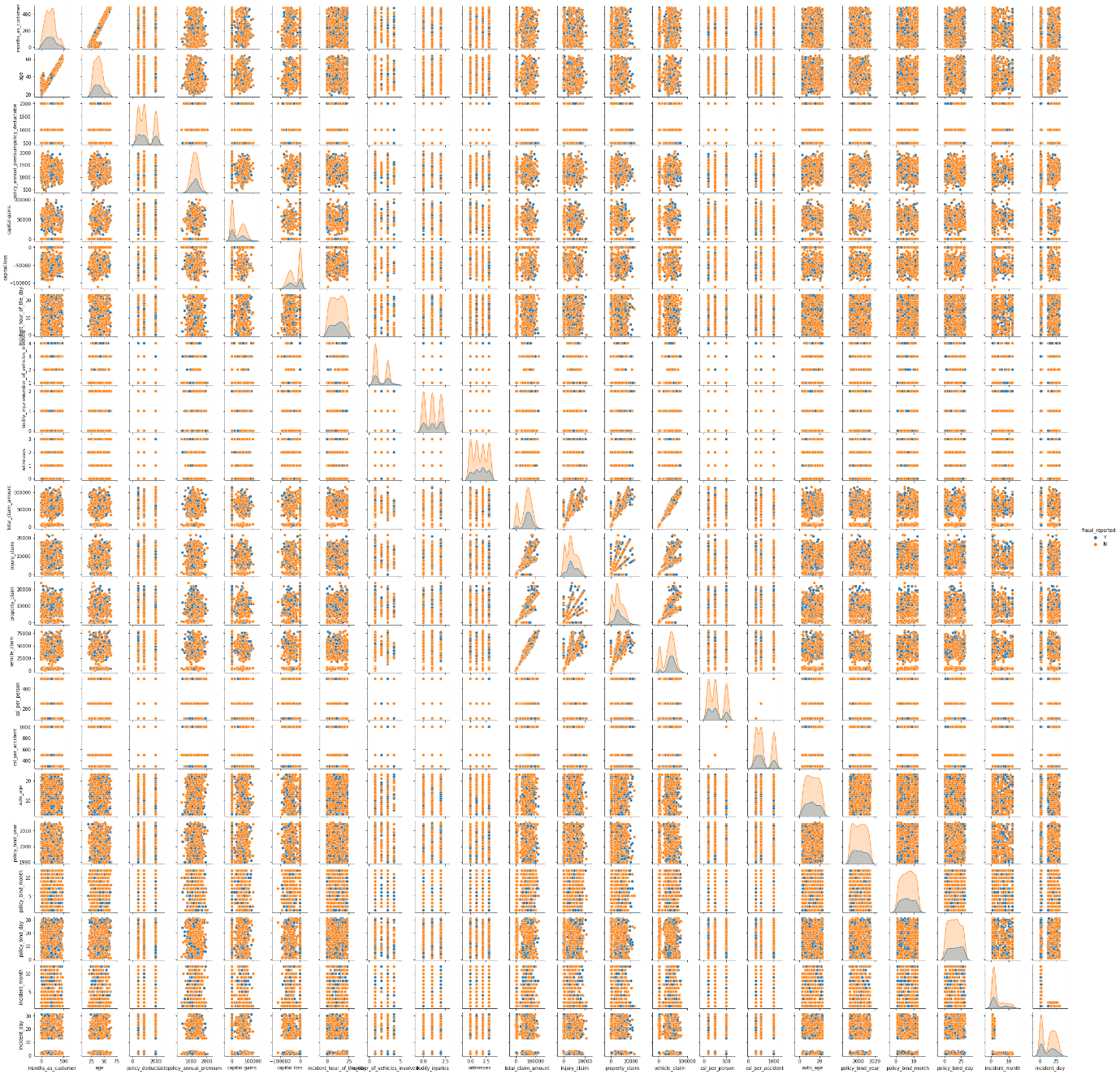
  



By looking into the count plots below are my observations.

* OH policy\_state count is high and high number of fraud are also reported here.
* Female insured are more in number however almost equal number of frauds are reported for both male and female.
* JD High school qualified insured are more in number.
* Machine-op-inspct are high in number however exec-managerial reported more number of frauds.
* Most of the insured persons are having reading as their hobby.People with chess as hobby reported very high number of frauds.
* own-child has maximum count in insured\_relationship.
* Multi-vehicle-collision and single-vehicle-collision are more in number and both reported almost equal number of frauds.
* Rather than front and side collision Rear Collision has maximum count.
* Minor damage has maximum count however major damage reported more number of frauds.
* Police authority is mostly contacted.
* In NY state most of the cases are registered and PA registered least number of cases.
* In all the cities the insuered climb are almost similar.
* There is very less count for property damage.
* Very less cases have police\_report\_available.
* Nissan has high number of cases compare to other car brands.
* Since the problem is of classification. And the target column fraud\_reported is imbalanced. The count of Y is less compared to N. We have to balance these counts to get a good model.
* Pair plot is used for better understanding of correlation -

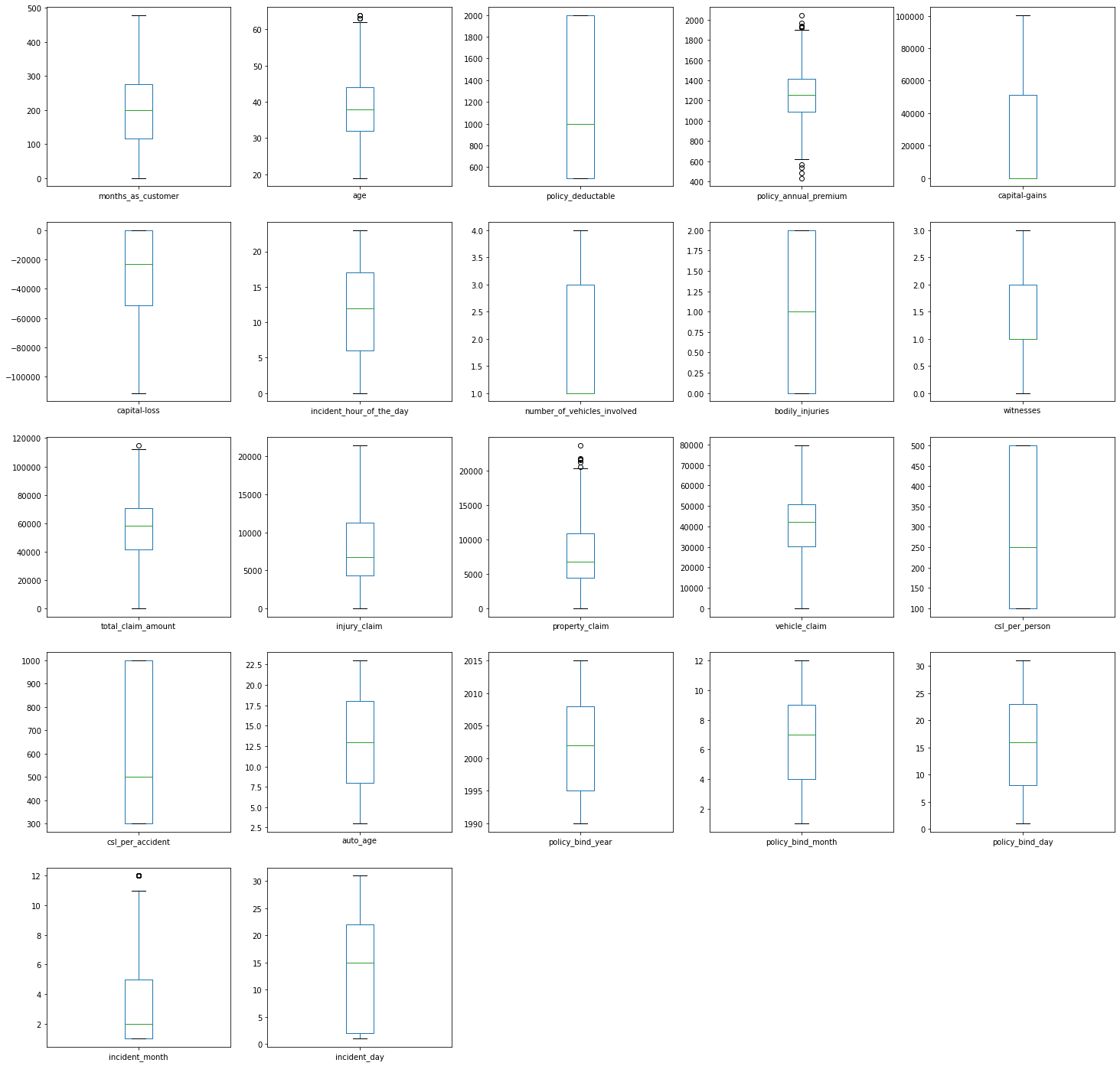


**EDA Concluding Remark:**

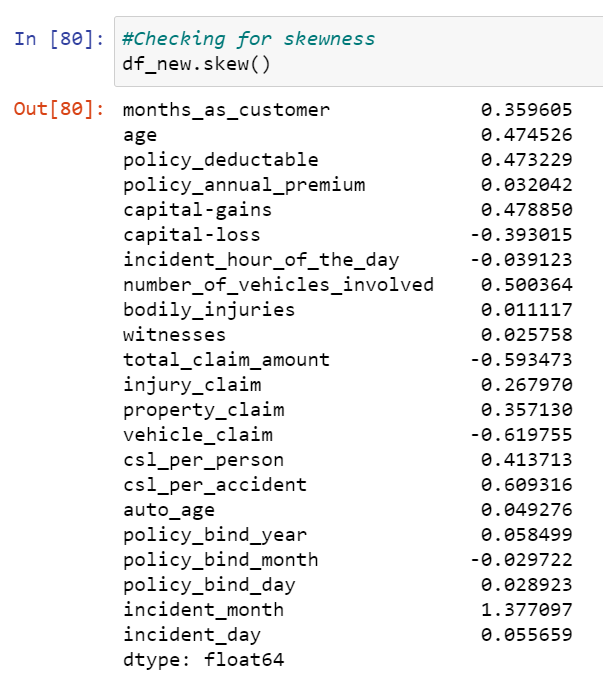
* Checked for NaN values and there are no missing values in the dataset.
* Extracted the necessary features from existing features to get better accuracy and dropped the old columns to avoid multicollinearity.
* Dropped the unnecessary columns and replaced the ‘?’ entries with their suitable values.
* Used both matplotlib and seaborn to visualize the data.
* Used distplot, barplot, scatterplot and boxplot to get better insight on the features. Since most of my columns were categorical, I have used all categorical plots. For numerical columns I have used numerical plotting but I did not get any good pattern with numerical columns.

**Checking for Outliers and Skewness:**

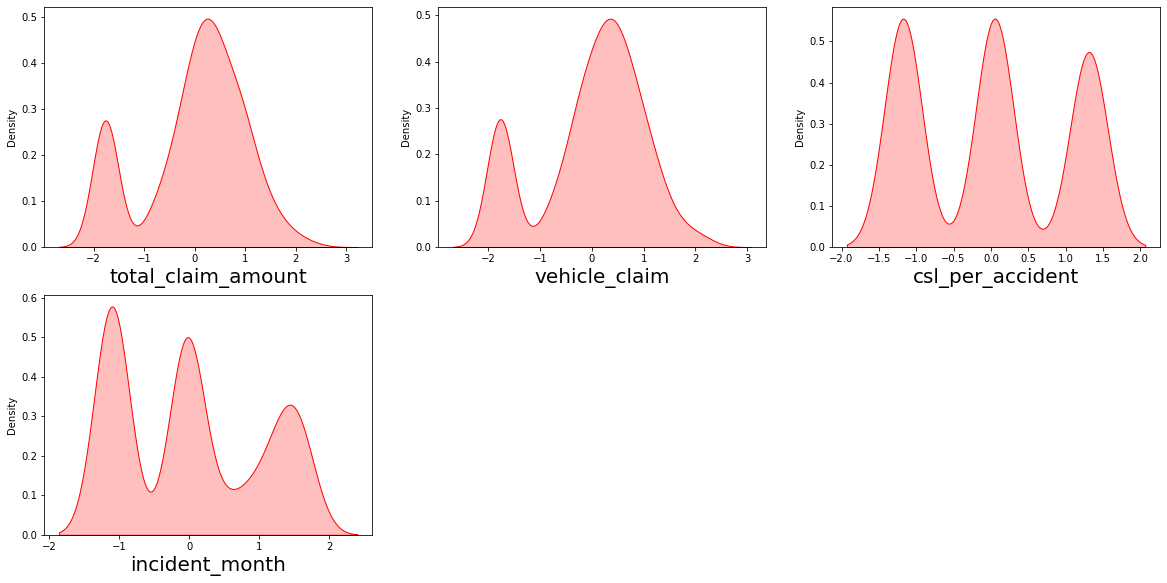
Used box to check outliers in the dataset-



* Used box plot to check outliers and found outliers in age, policy\_annual\_premium, total\_claim\_amount, property\_claim and incident\_month. Now I have to remove outliers in these columns.
* zscore method with 0.4% data loss is used to remove outliers, after removing the outliers using zscore I have saved the dataset as df\_new.



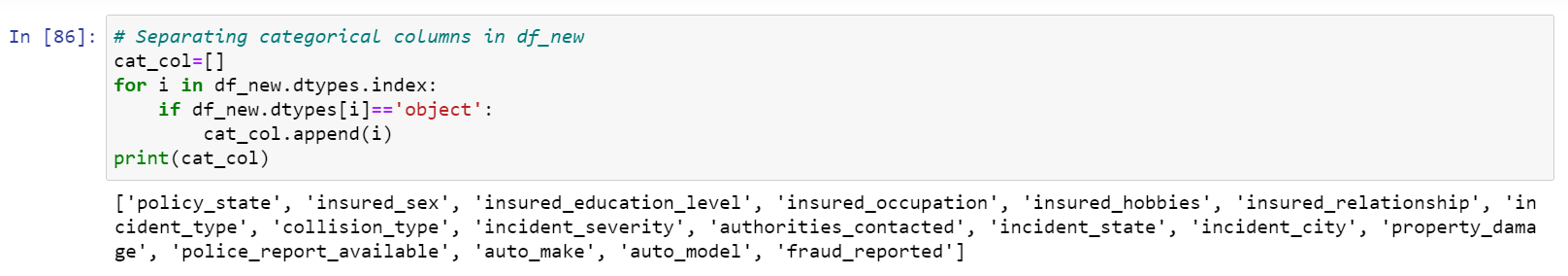
* Skewness is present in total\_claim\_amount, vehicle\_claim, csl\_per\_accident and incident month.
* To remove skewness Yeo-johson method is used. After removing the skewness the distplot of skewed columns is shown below.

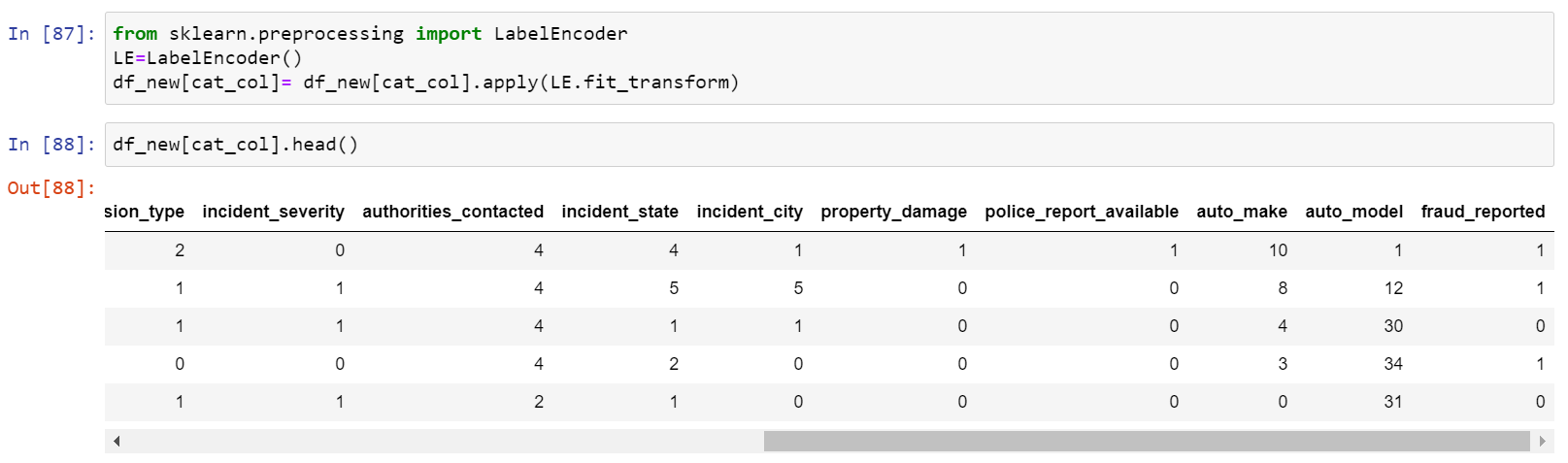


* skewness has reduced in almost all the columns. It looks good to proceed now.

**Label encoding:**

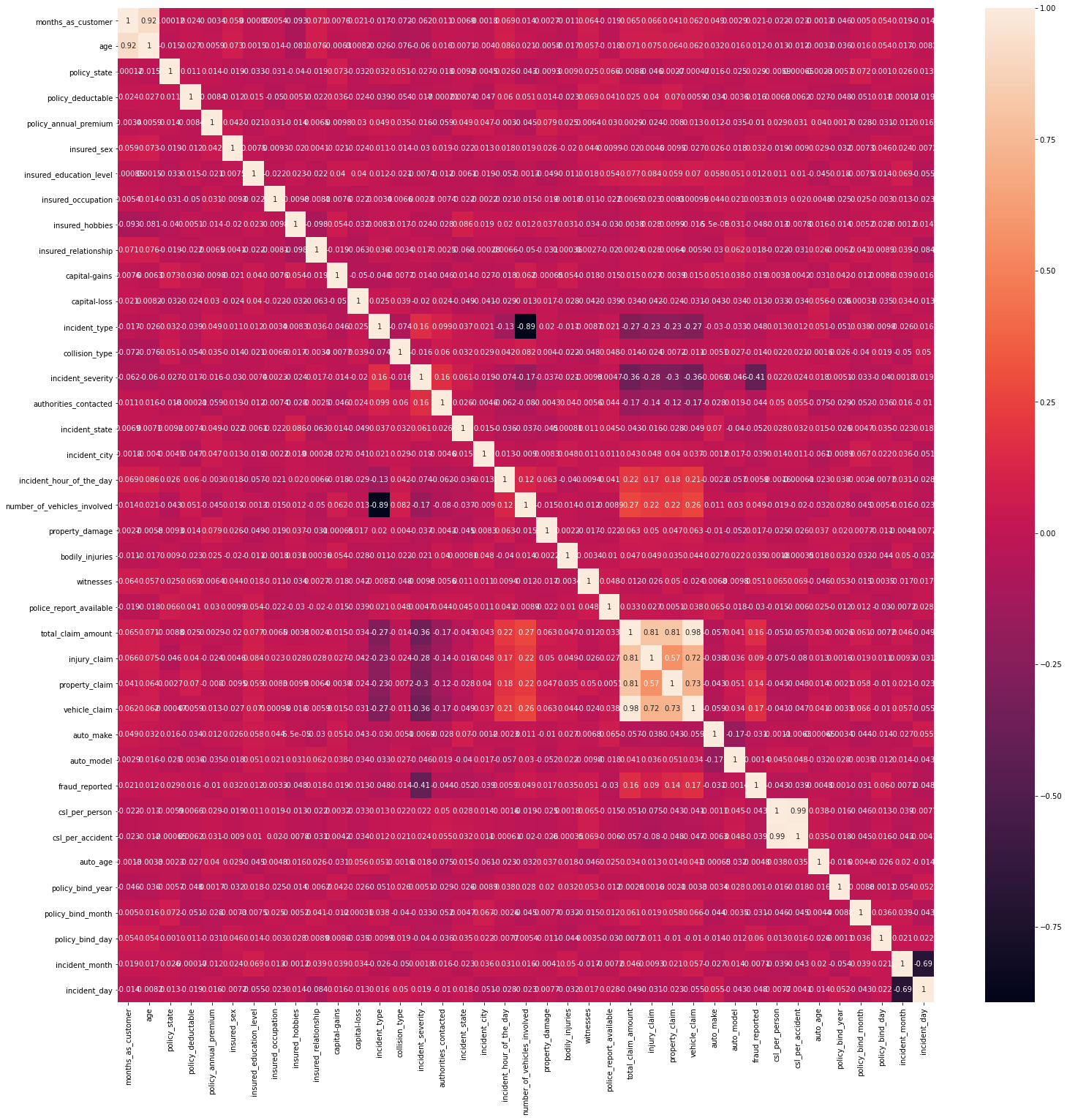
To encode categorical data in to numerical data using label encoding





**Checking for correlation using heat map:**

After checking the correlation, to get better insight on the corr values I have plotted heat map.



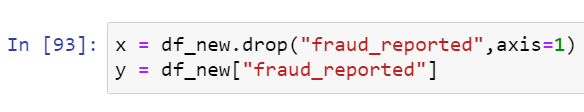
Looking into the heat map I can say that there is multicollinearity issue.

multicollinearity can be seen in following columns-

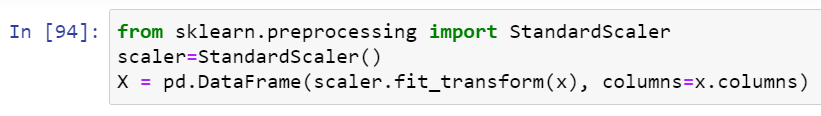
* incident type and number\_of\_vehicles\_invovled
* incident\_month and incident\_day
* total\_claim\_amount and injury\_claim
* total\_claim\_amount and property\_claim
* injury\_claim and vehicle\_claim
* total\_claim\_amount and vehicle\_claim
* vehicle\_claim and property\_claim
* total\_claim\_amount and property\_claim

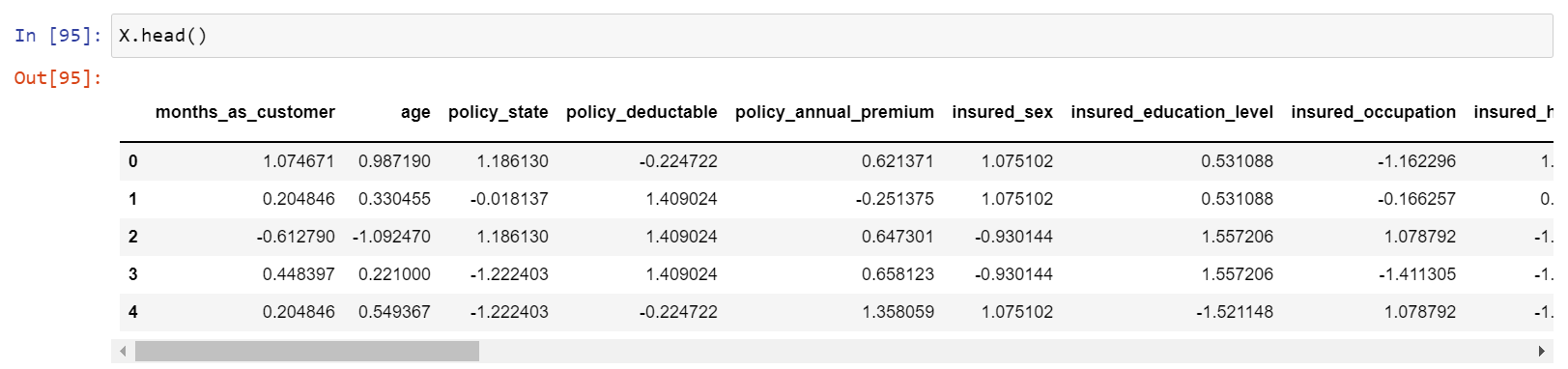
**Pre-processing Pipeline:**

* As a first step to make model, separate the dependent and independent features.

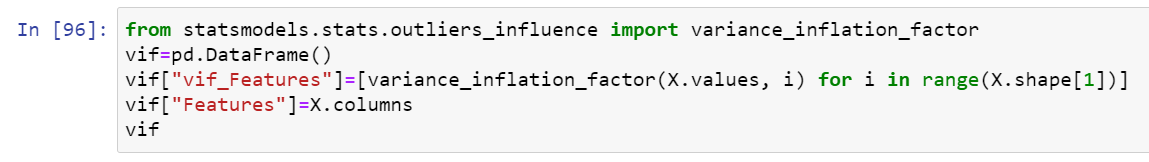


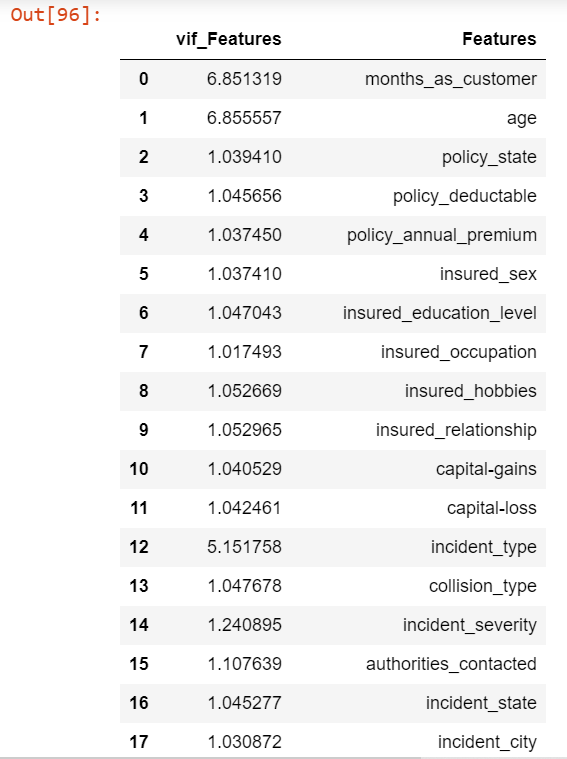
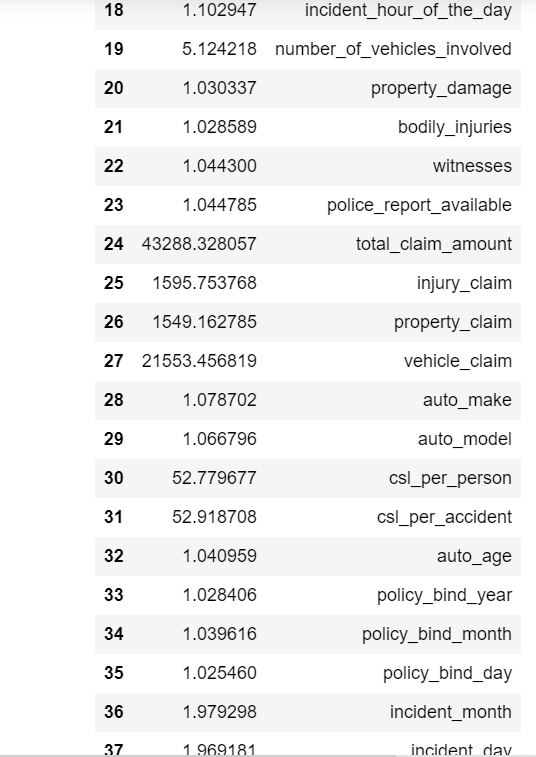
* Taken x as all independent features and y as dependent/target feature.
* Scaled independent features to get the same range in all the columns. If independent columns are not scaled then there is a chance that, model may get baised.





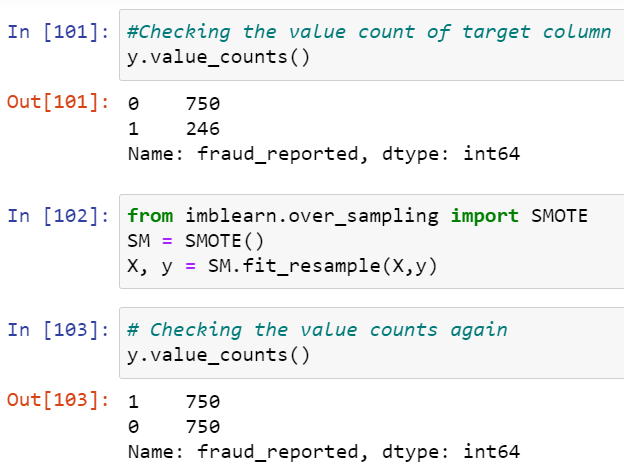
* Now scaling is done. But have to check out for multicollinearity using VIF (variance inflation factor).



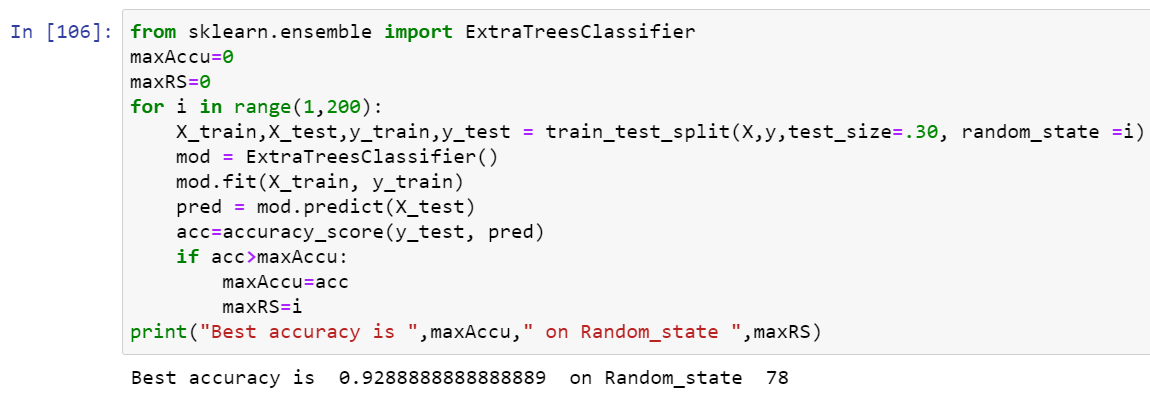
* Noticed a high VIF for total\_claim\_amount, so dropped that column and again checked for VIF and got the highest value for csl\_per\_accident so dropped that column too. Then multicollinearity issue has been solved.

**Data Balancing:**

* As observed before target column is imbalanced which need to balance by using over sampling. I can also use under sampling but I haven’t because of data loss.
* Now the balance issue is solved
* Data is all set for model building. Let’s go ahead with classification algorithms as we are dealing with Classification Problem.

**Building Machine Learning Models:**

**Finding best random state and accuracy:**



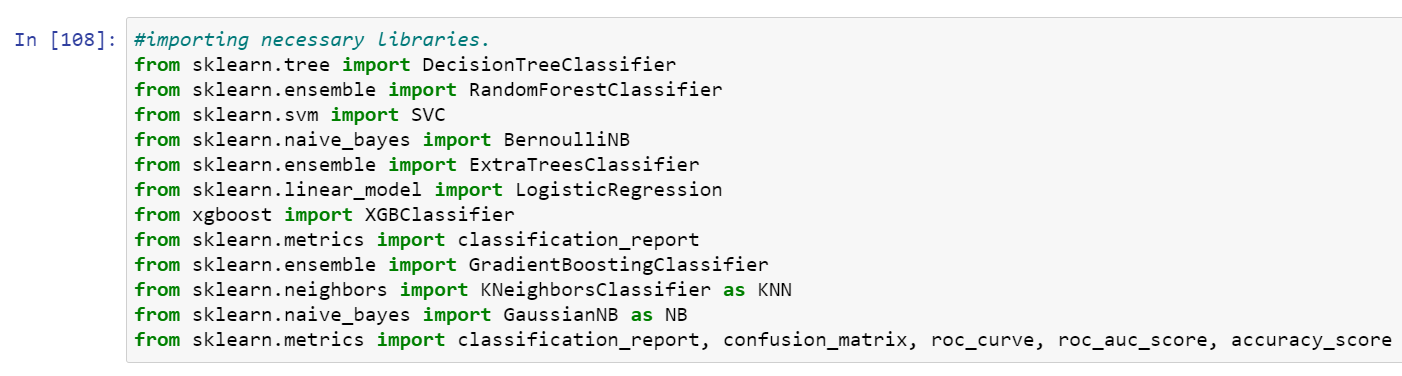
* I got best random state = 93 and accuracy = 92.89%. Now the task is to find the best fitting model.



* Created train and test data as X\_train, X\_test and y\_train, y\_test.

**Classification Algorithms:**

Import all necessary libraries



* Used Cross validation as model evaluation metrics for all the algorithms and accuracy\_score, Confusion metrics as metrics in model building.

**i) RandomForestClassifier:**

RFC**=**RandomForestClassifier()

RFC**.**fit(X\_train,y\_train)

predrf**=**RFC**.**predict(X\_test)

print('Accuracy Score:',accuracy\_score(y\_test, predrf))

print('Confusion Matrix:',confusion\_matrix(y\_test, predrf))

print(classification\_report(y\_test,predrf))

Accuracy Score: 0.9155555555555556

Confusion Matrix: [[195 16]

[ 22 217]]

precision recall f1-score support

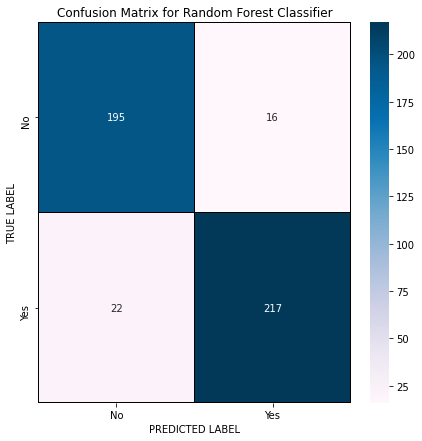
0 0.90 0.92 0.91 211

1 0.93 0.91 0.92 239

accuracy 0.92 450

macro avg 0.91 0.92 0.92 450

weighted avg 0.92 0.92 0.92 450

****

* Random Forest Classifier model is giving me 91.5% accuracy\_score and the cross validation is 86.6%. RFC is working good .

**ii) ExtraTreeClassifier:**

In [103]:

ETC**=**ExtraTreesClassifier()

ETC**.**fit(X\_train,y\_train)

predet**=**ETC**.**predict(X\_test)

print('Accuracy Score:',accuracy\_score(y\_test, predet))

print('Confusion Matrix:',confusion\_matrix(y\_test, predet))

print(classification\_report(y\_test,predet))

Accuracy Score: 0.94

Confusion Matrix: [[194 17]

[ 10 229]]

precision recall f1-score support

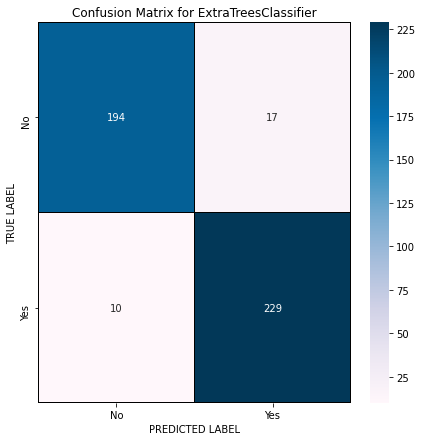
0 0.95 0.92 0.93 211

1 0.93 0.96 0.94 239

accuracy 0.94 450

macro avg 0.94 0.94 0.94 450

weighted avg 0.94 0.94 0.94 450



* Extra Trees Classifier model is giving me 94% accuracy\_score and the cross validation is 91.1%.

**iii) Gradient Boosting Classifier:**

In [105]:

GBC**=**GradientBoostingClassifier()

GBC**.**fit(X\_train,y\_train)

predgb**=**GBC**.**predict(X\_test)

print('Accuracy Score:',accuracy\_score(y\_test, predgb))

print('Confusion Matrix:',confusion\_matrix(y\_test, predgb))

print(classification\_report(y\_test,predgb))

Accuracy Score: 0.9222222222222223

Confusion Matrix: [[191 20]

[ 15 224]]

precision recall f1-score support

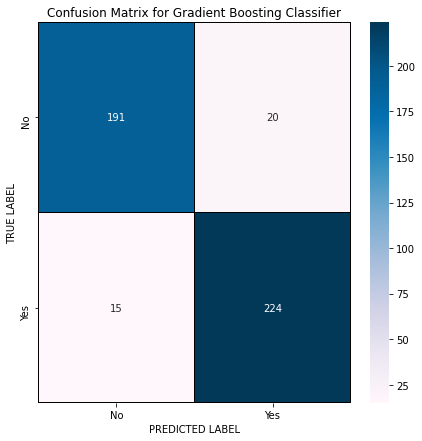
0 0.93 0.91 0.92 211

1 0.92 0.94 0.93 239

accuracy 0.92 450

macro avg 0.92 0.92 0.92 450

weighted avg 0.92 0.92 0.92 450



* Gradient Boosting Classifier model is giving me 92% accuracy\_score and the cross validation is 86.6%.

**iv) SupportVectorClassifier:**

In [107]:

SV**=**SVC()

SV**.**fit(X\_train,y\_train)

predsv**=**SV**.**predict(X\_test)

print('Accuracy Score:',accuracy\_score(y\_test, predsv))

print('Confusion Matrix:',confusion\_matrix(y\_test, predsv))

print(classification\_report(y\_test,predsv))

Accuracy Score: 0.8866666666666667

Confusion Matrix: [[185 26]

[ 25 214]]

precision recall f1-score support

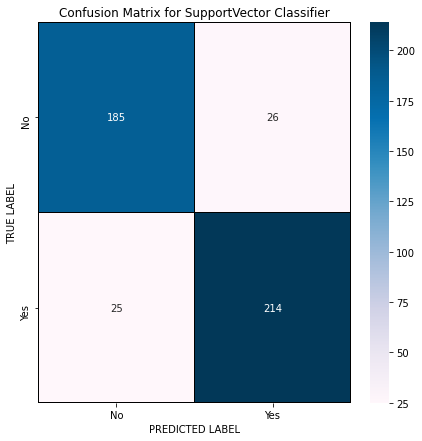
0 0.88 0.88 0.88 211

1 0.89 0.90 0.89 239

accuracy 0.89 450

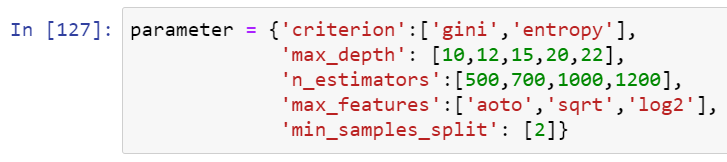
macro avg 0.89 0.89 0.89 450

weighted avg 0.89 0.89 0.89 450

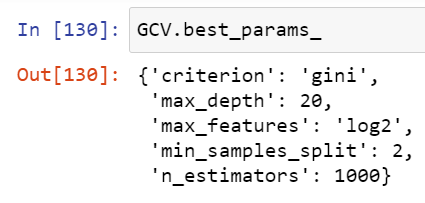
****

* Support Vector Classifier model is giving me 88.6% accuracy\_score and the cross validation is 86.5%

**Hyper Parameter Tuning:**



* Using the above parameters list let’s tune my best model i.e., Extra Trees Classifier and build the best model.



* After knowing the above best parameters, run the model again for improved accuracy.

Final\_mod**=**ExtraTreesClassifier(criterion**=**'gini', max\_depth**=**20,max\_features**=**'log2', min\_samples\_split**=**2, n\_estimators**=**1000)

Final\_mod**.**fit(X\_train,y\_train)

pred**=**Final\_mod**.**predict(X\_test)

acc**=**accuracy\_score(y\_test, pred)

print('Accuracy Score:',(accuracy\_score(y\_test,pred)**\***100))

print('Confusion matrix:',confusion\_matrix(y\_test,pred))

print(classification\_report(y\_test,pred))

Accuracy Score: 93.55555555555556

Confusion matrix: [[196 15]

[ 14 225]]

precision recall f1-score support

0 0.93 0.93 0.93 211

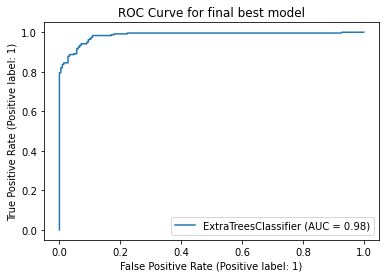
1 0.94 0.94 0.94 239

accuracy 0.94 450

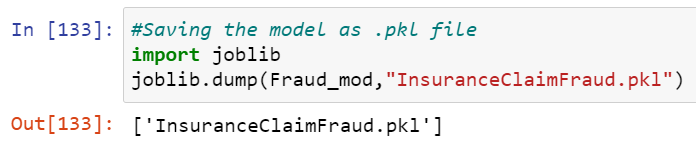
macro avg 0.94 0.94 0.94 450

weighted avg 0.94 0.94 0.94 450

**Best accuracy 93.55**

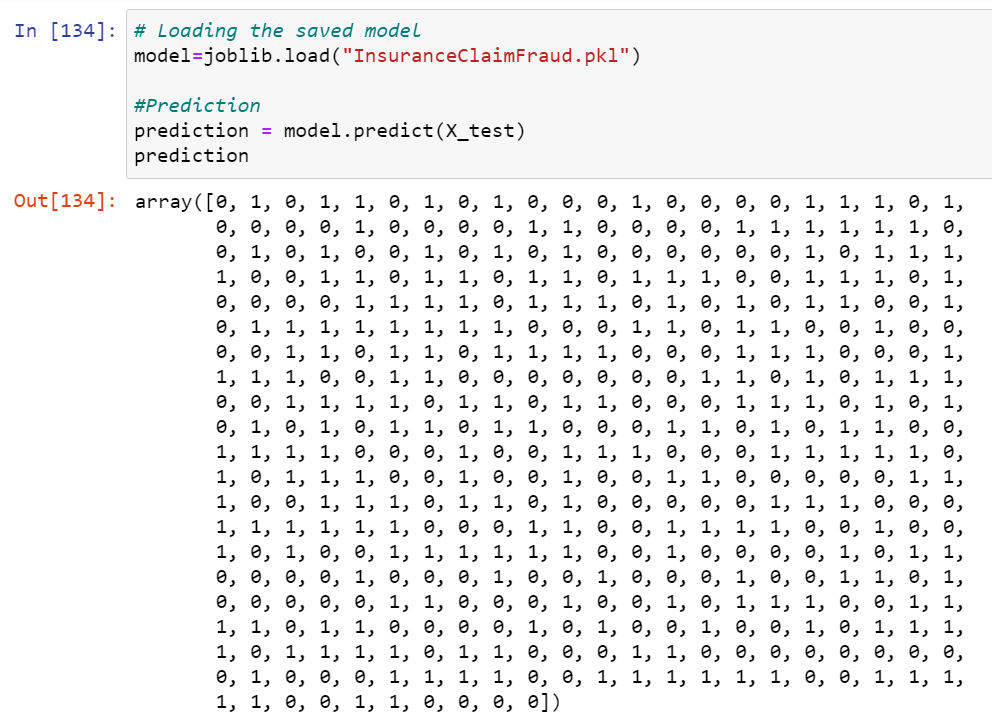
****

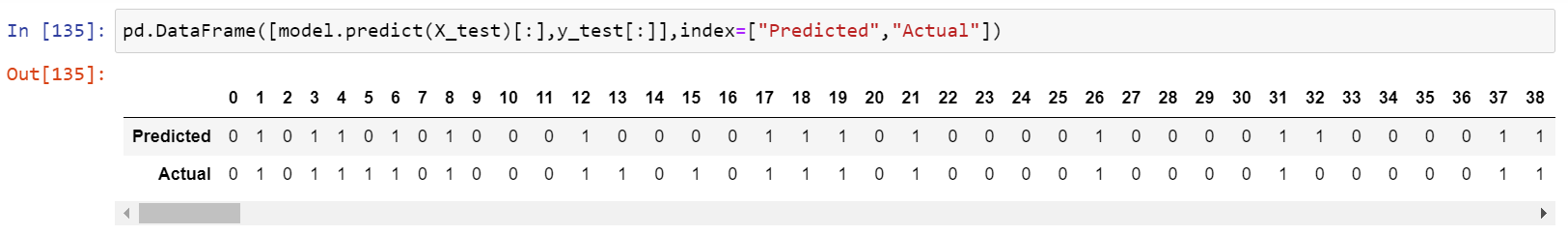
* Above is the ROC curve for best model. And the AUC value also remained same.
* After getting this best model I have saved it using .pkl. As Insurance\_claim\_fraud.



**Predictions:**

* Now using the saved model, we can predict whether the insurance claim is fraudulent or not.





**Concluding Remarks:**

* Insurance claim fraud detection project needs a good visualization on data. Feature Engineering is one of the most crucial aspects of this project.
* Need to handled numerical and categorical data separately and carefully to build different machine learning models on the same dataset and identify which model is giving best accuracy.
* Use of hyper parameter tunning to improve best model accuracy.
* Using this machine Learning Model, we can predict if the insurance claim is fraudulent or not.