Insurance Claims- Fraud Detection

Automobile Industry's Claims Fraud Increases day by day.so for Industry or Company it is necessary to predict the Claims weathers is Fraud or Not. It is not reliable for the company to check each and every Claim because it can be time Consuming and Costly.at that time Machine Learning is play very Important Role.

With the Help of Diff Diff Attributes about the claims, insured people and other Circumstances we can find weathers this Claims is Fraud or not.

Problem Definition

The main goal of this project to find the Fraud of the Claims by Machine Learning Algorithms. The main issue behind the ML is verity of the fraud and very less amount of known Fraud.

Here we have 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.

Data Analysis

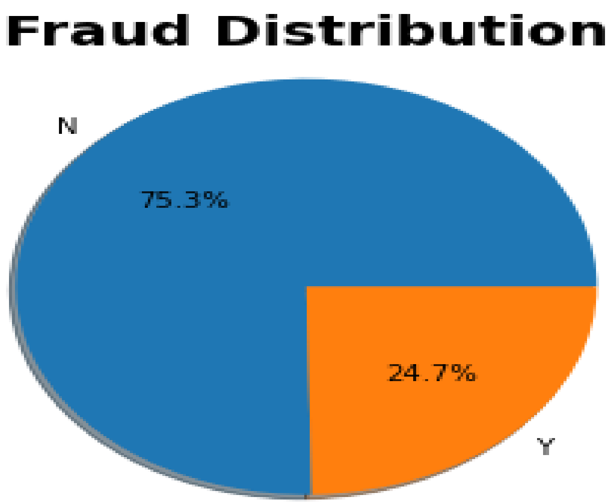
Here we have Dataset we have dataset which have insurance policy along with customers details and details of accident details. Data Contains 1000 rows and 40 columns. We have zero null data. Out of 40 features 21 features are categorical and 19 are numerical features.

Dataset contains 1000 rows and 40 features.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 40 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   months_as_customer           1000 non-null   int64
 1   age                          1000 non-null   int64
 2   policy_number                1000 non-null   int64
 3   policy_bind_date             1000 non-null   object
 4   policy_state                 1000 non-null   object
 5   policy_csl                   1000 non-null   object
 6   policy_deductable            1000 non-null   int64
 7   policy_annual_premium        1000 non-null   float64
 8   umbrella_limit               1000 non-null   int64
 9   insured_zip                  1000 non-null   int64
 10  insured_sex                  1000 non-null   object
 11  insured_education_level      1000 non-null   object
 12  insured_occupation           1000 non-null   object
 13  insured_hobbies              1000 non-null   object
 14  insured_relationship         1000 non-null   object
 15  capital-gains                1000 non-null   int64
 16  capital-loss                 1000 non-null   int64
 17  incident_date                1000 non-null   object
 18  incident_type                1000 non-null   object
 19  collision_type               1000 non-null   object
 20  incident_severity            1000 non-null   object
 21  authorities_contacted        1000 non-null   object
 22  incident_state               1000 non-null   object
 23  incident_city                1000 non-null   object
 24  incident_location            1000 non-null   object
 25  incident_hour_of_the_day     1000 non-null   int64
 26  number_of_vehicles_involved  1000 non-null   int64
 27  property_damage              1000 non-null   object
 28  bodily_injuries              1000 non-null   int64
 29  witnesses                    1000 non-null   int64
 30  police_report_available      1000 non-null   object
 31  total_claim_amount           1000 non-null   int64
 32  injury_claim                 1000 non-null   int64
 33  property_claim               1000 non-null   int64
 34  vehicle_claim                1000 non-null   int64
 35  auto_make                    1000 non-null   object
 36  auto_model                   1000 non-null   object
 37  auto_year                    1000 non-null   int64
 38  fraud_reported               1000 non-null   object
 39  _c39                         0 non-null      float64
dtypes: float64(2), int64(17), object(21)
memory usage: 312.6+ KB
```

Exploratory Data Analysis: -

Dependent Variable/Target Variable: This Data Set having Fraud Reported as target variable.in this data set having 247 as Fraud and 753 as No Fraud.so it looks like imbalanced problem. We will balance data in Data Pre-processing part.

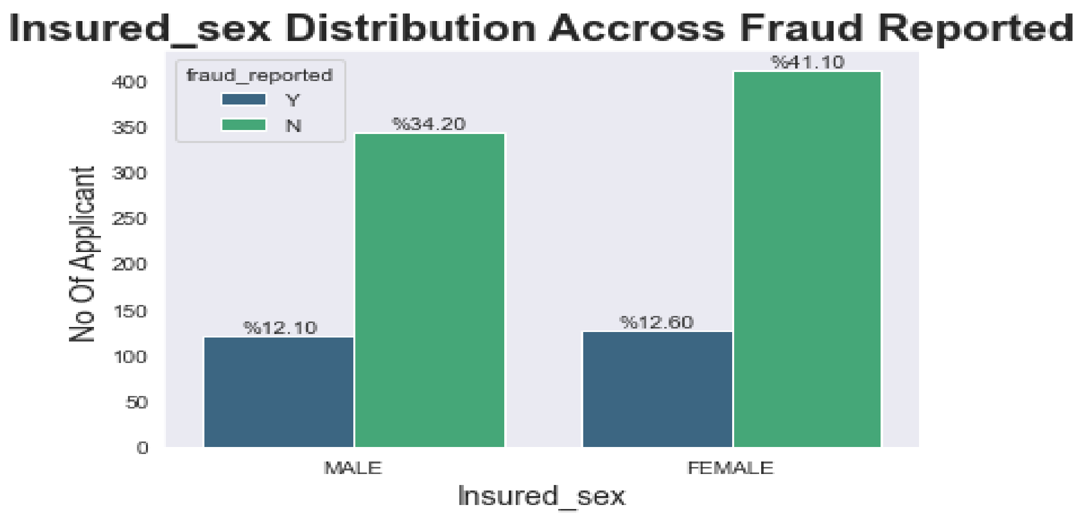


24.7% found Fraud claims which is very big size and which is concern to company.
Now we will calculate weather Company is in Loss or Profit due to Fraud claims.
We take policy Premium is Income for the Company and total claim amount is Outcome of the company. From that we can calculate Company's Profit and Loss.

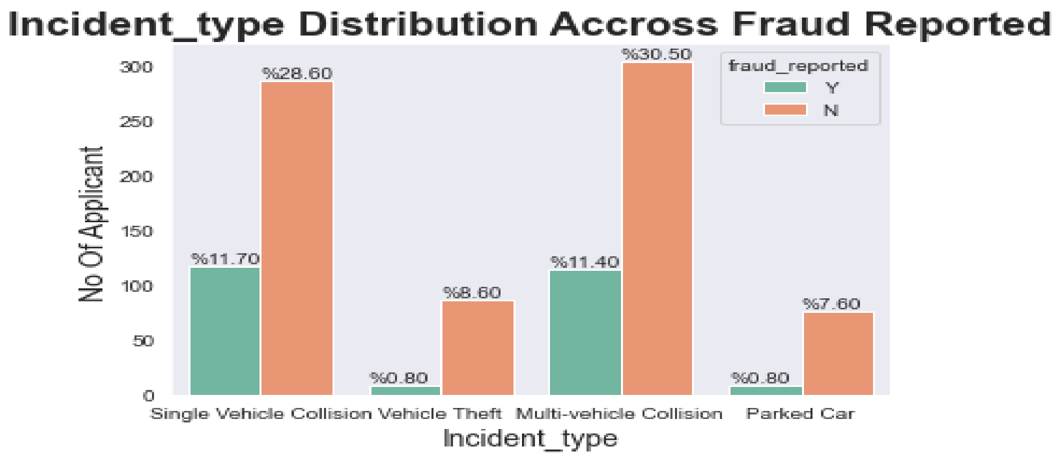
```
df['Profit/Loss']=((df['months_as_customer']/12)*(df['policy_annual_premium']))-(df['total_claim_amount'])
df['Profit/Loss'].values.sum()
```

```
-31396110.803333335
```

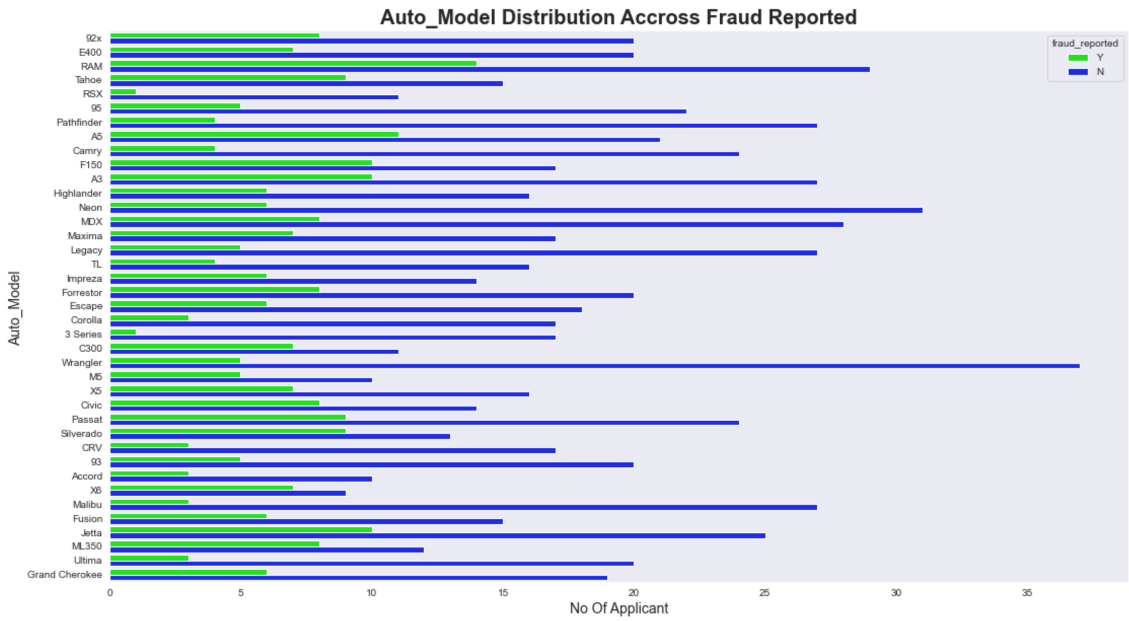
<u>Here we see that Company is in Big Loss. that's why it is very necessary to concern on this Fraud Claims.</u>



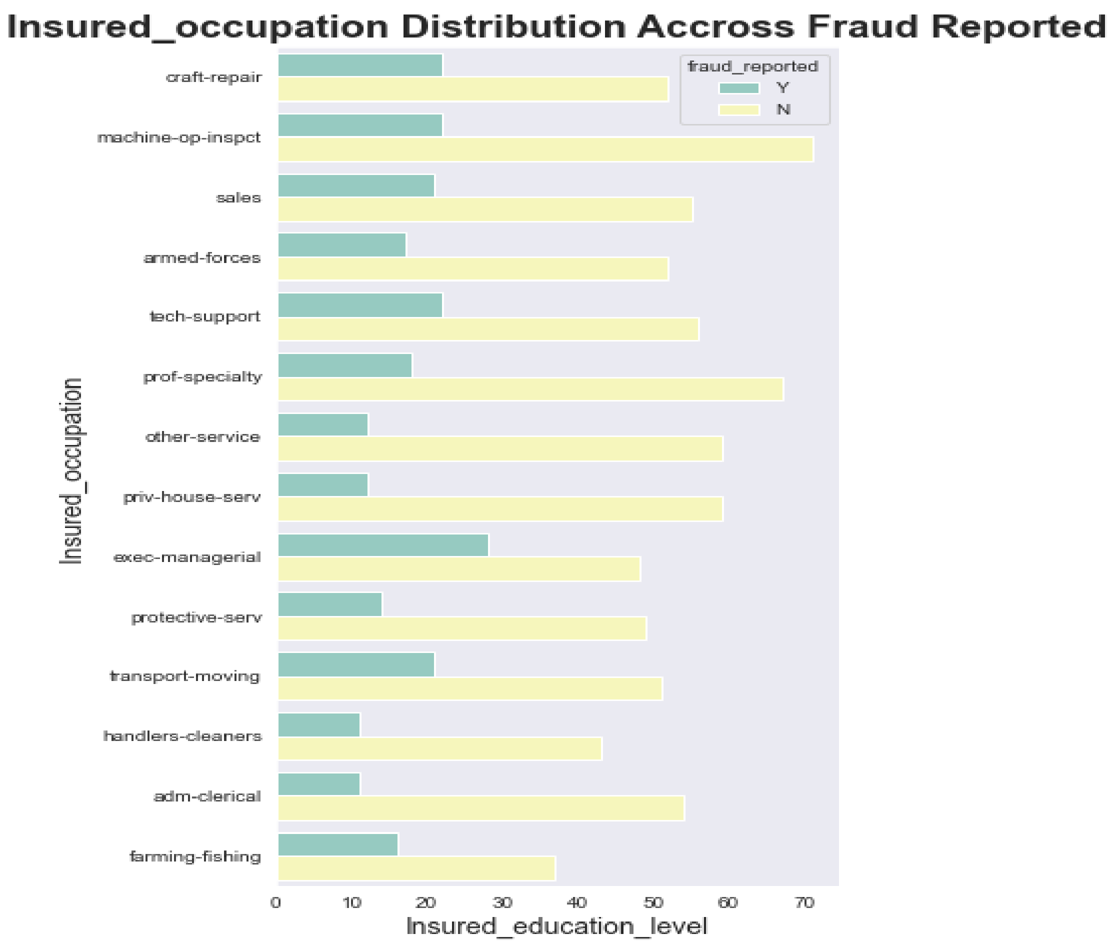
Here we see that Fraud Found in Mail Applicant is higher than Female Applicant



Single vehicle and Multi-Vehicle Collision are High in No as compare to Vehicle Thief and Parked Vehicle Accident.

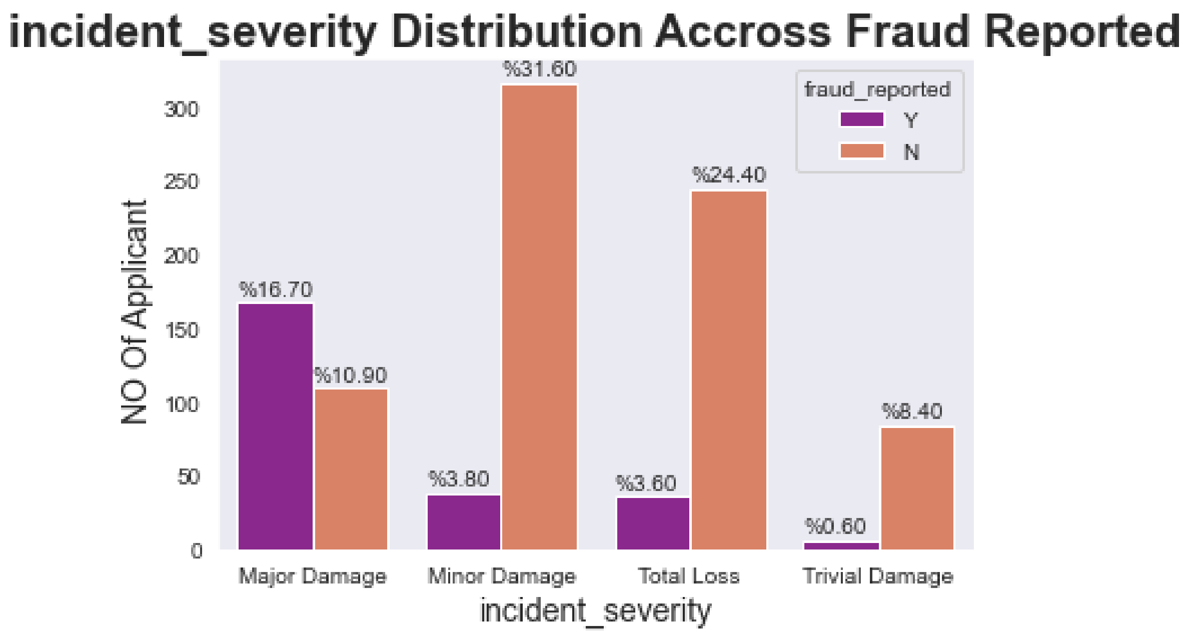


RAM, Neon and Jetta Found more No of Accident as compare to another Model. RAM model having more Fraud than RSX having less claim Fraud.

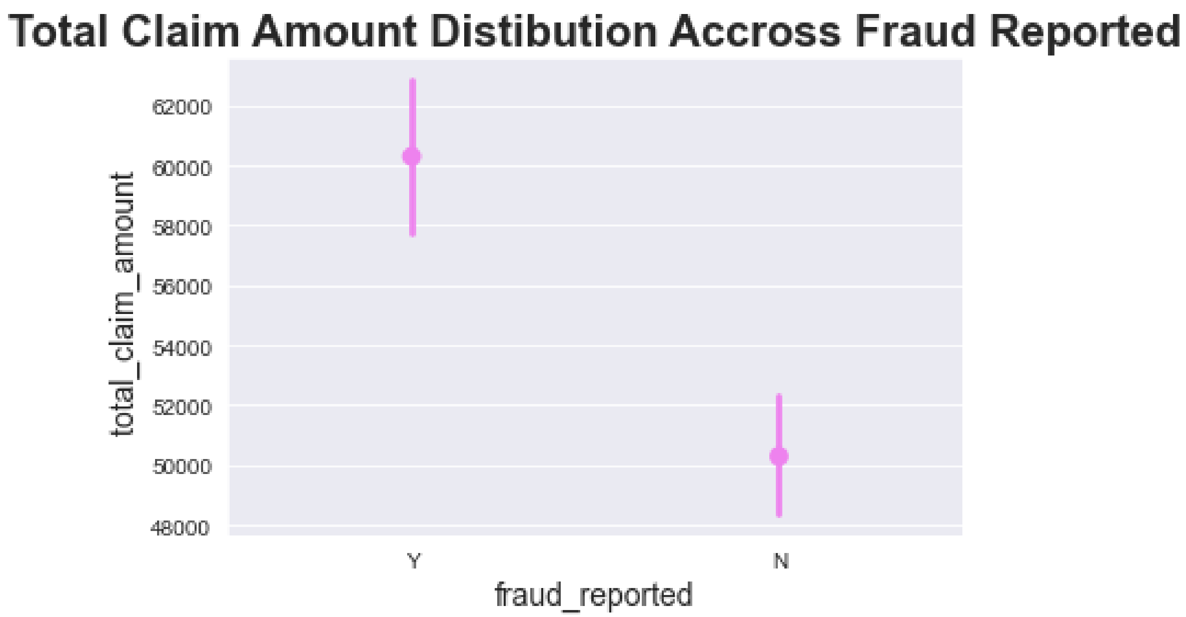


This chart indicated Ex-Managerial having higher Tendency of Fraud and handlers-cleaners having less Tendency fraud. This is very shocking Observation.

as highly educated & at higher post becomes fraud and less educated having less fraud.



From This chart we say that Major damage severity having more Fraud so that we should check across the claim amount too. There may be possible whose claim for higher amount they have fraud too.



From this visualization we clearly see that Fraud find in Higher Claim Amount and for lower amount of claim fraud is none.

Data Pre-processing Pipeline: -

Datapre-Processing is the the process of the transforming the raw data into the useful or understate Data.
In the real dataset there is lost of mixture of data like missing values, incomplete values, Noisy data and much more.so its necerессary to pre-process the data before applying into the model.

There are steps in Data Pre-Processing.

1)Data Cleaning: - Removing Outliers, Skewness and imputing Missing Values.
2) Data Transformation: - like Normalization by applying normalization we can improve the accuracy and efficiency of the models. And also reduce the errors.
3)Data Reduction: By Reducing the no of features by Feature Selection Process, PCA And VIF

In Our Project we find some'?' in data set than we first replace it with none value and after that we put most frequent.

```
df['collision_type']=df['collision_type'].replace('?',np.nan)
df['police_report_available']=df['police_report_available'].replace('?',np.nan)
df['property_damage']=df['property_damage'].replace('?',np.nan)
```

```
nan=['police_report_available','property_damage','collision_type']
```

```
for i in nan:
    df[i].fillna(df[i].mode()[0],inplace=True)
```

```
df['collision_type']=df['collision_type'].replace('?',np.nan)
df['police_report_available']=df['police_report_available'].replace('?',np.nan)
df['property_damage']=df['property_damage'].replace('?',np.nan)
```

```
df.isnull().sum()
```

```
months_as_customer              0
age                             0
policy_number                   0
policy_bind_date                0
policy_state                    0
policy_csl                      0
policy_deductable               0
policy_annual_premium           0
umbrella_limit                  0
insured_zip                     0
insured_sex                     0
insured_education_level         0
insured_occupation              0
insured_hobbies                 0
insured_relationship            0
capital-gains                   0
capital-loss                    0
incident_date                   0
incident_type                   0
collision_type                178
incident_severity               0
authorities_contacted           0
incident_state                  0
incident_city                   0
incident_location               0
incident_hour_of_the_day        0
number_of_vehicles_involved     0
property_damage               360
bodily_injuries                 0
witnesses                       0
police_report_available       343
total_claim_amount              0
injury_claim                    0
property_claim                  0
vehicle_claim                   0
auto_make                       0
auto_model                      0
auto_year                       0
fraud_reported                  0
_c39                         1000
```

Converting Categorial into Numerical: -

There are few techniques used for converting into the Numerical like OneHotEncoder and Label Encoder. In our project I used Label Encoder For converting.
Label Encoder can be import from the sklearn library.

Data Cleaning Process: -

First, we remove some Unnecessary Features like Policy no, policy_bid_data, profit/loss, incident_hour_of_the_day, number _of_vehical_involved.

Removing Outliers: -

It is defined as the points that are far away from the same points.it can be happen because of the variability of the measurements and may be some error also. If possible, outliers should be removed from the datasets. There are servals methos to remove the outliers.
1)Z score
2) Quantile Method (Capping the data)

1)Z Score: it can call from the SciPy. Stats library. And for most of the case threshold values should be used 3.

```
from scipy.stats import zscore
```

```
z=np.abs(zscore(df))
```

```
print(np.where(z>3))
```

```
(array([ 31,  48,  88, 115, 119, 229, 248, 262, 314, 430, 458, 500, 503,
       657, 700, 763, 807, 875, 922, 975], dtype=int64), array([ 8,  8,  8,  8,  8,  7,  7,  8,  8,  8,  8, 33,  8,  8,  8,  7,
16,
        8,  8,  8], dtype=int64))
```

```
df_new=df[(z<3).all(axis=1)]
```

```
df_new.shape
```

```
(980, 40)
```

```
df.shape
```

```
(1000, 40)
```

```
dataloss=(1000-979)/1000*100
dataloss
```

```
2.1
```

Here Around 2.1% id Data Loss.

2)Quantile Methods: Inter Quantile Range is used to detect or cap the outliers.
Calculate the IQR by scipy.stats.iqr
Multiply Interquartile range by 1.5
Add 1.5 x interquartile range to the third quartile. Any number greater than this is a suspected outlier.
Subtract 1.5 x interquartile range from the first quartile. Any number lesser than this is a suspected outlier.

Splitting Data Into train_test_split: -

This function is in sklearn. Model selection splitting the data array into two arrays. Train and Test with this function we don't need to splitting train and test manually.by default it make random partition and we can also set the random state.it gives four o/p like x_train, x_test, y_train, y_test.

After Doing splitting we have to balanced our data.it can be by SMOTE or oversampling methods. Like Up Sampling, down sampling.

Up sampling: -This method used to modify the unequal data into the balanced data by increases the minory class or rare class. Advantage of this method is to no loss of information but from that model can be in overfitting.

Down Sampling: like the Up sampling its also balanced data but by reducing the size of the class which is high.

SMOTE

```
from imblearn.over_sampling import SMOTE
```

```
smt=SMOTE()
trainx,trainy=smt.fit_resample(x_train,y_train)
```

```
trainy.value_counts()
```

```
1    585
0    585
Name: fraud_reported, dtype: int64
```

But if we balanced our data before train test split means we balanced from our whole data set or form x. it means at that time our test data is leak. We have to isolate our test data. Here you expose it.so our f1 or recall or precision will be good. so, our model will already know which is positive or negative. And I can also say because of that there is bias or model Overfitting.to prevent this We balanced our data

Now our data is ready to apply to the model**.**

Try Different Models….

1)Logistic Regression: -logistic regression is the supervised machine learning problem which is used for the classification problem and used to predict the probability of the classification.it is widely used for the binary classification problem. It is one od the simplest methos of Ml.

```
fun(lg)
```

```
Accuracy Score 73.46938775510205
Confusion Matrix
 [[111  44]
 [  8  33]]
Classification Report
               precision    recall  f1-score   support

           0       0.93      0.72      0.81       155
           1       0.43      0.80      0.56        41

    accuracy                           0.73       196
   macro avg       0.68      0.76      0.68       196
weighted avg       0.83      0.73      0.76       196

F1 score 55.932203389830505
```

2)Decision Tree Classifier: DTC can be used by both classification and regression both. But mostly it's used for the classification problem. Its structure is tree based. Where internal nodes represents the features of dataset and branches represents the decision rules and each leaf nodes represents the outcomes.

```
: fun(dtc)

Accuracy Score 76.0204081632653
Confusion Matrix
 [[123  32]
 [ 15  26]]
Classification Report
               precision    recall  f1-score   support

           0       0.89      0.79      0.84       155
           1       0.45      0.63      0.53        41

    accuracy                           0.76       196
   macro avg       0.67      0.71      0.68       196
weighted avg       0.80      0.76      0.77       196

F1 score 52.52525252525253
```

3)Gradient Boosting Classifier:

```
fun(gd)

Accuracy Score 82.6530612244898
Confusion Matrix
 [[132  23]
 [ 11  30]]
Classification Report
               precision    recall  f1-score   support

           0       0.92      0.85      0.89       155
           1       0.57      0.73      0.64        41

    accuracy                           0.83       196
   macro avg       0.74      0.79      0.76       196
weighted avg       0.85      0.83      0.83       196

F1 score 63.82978723404255
```

Confusion Matrix: -

We get out best score in Gradient Boosting Classifiers as Accuracy score is 82.65%. here model predict 132 as True Positive,23 False Positive, 11 False Negative and 30 True Negative.

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Now let’s understand the Recall Precision and f1-score

Accuracy: - it can be defined as the ratio of total number of correct classifications divided by total number of classifications.

Accuracy=(TP+TN)/(TP+FP+TN+FN)

Recall: -It is measure of correctly identified positive cases from all the actual positive cases.it is useful when cost of False Negative is high.

Recall=TP/(TP+FN)

Precision: - It is measure of all the positive predictions how many of them actually positive.

Precision=TP/(TP+FP)

F1-Score: - It give the combine result of Recall and Precision

F1-score=2*(Precision*Recall)/ (Precision + Recall)

Confusion Matrix: -It is the table that is used to describe the performance of classification model on set of tests data.by using different parameters.

Hyper Parameter Tunning

Hyper parameter optimisation in machine learning is used to find parameters of given machine learning algorithm that perform best as measured on validation. I used GridSearchCV for hyper tunning.

GradientBoosting Classifier

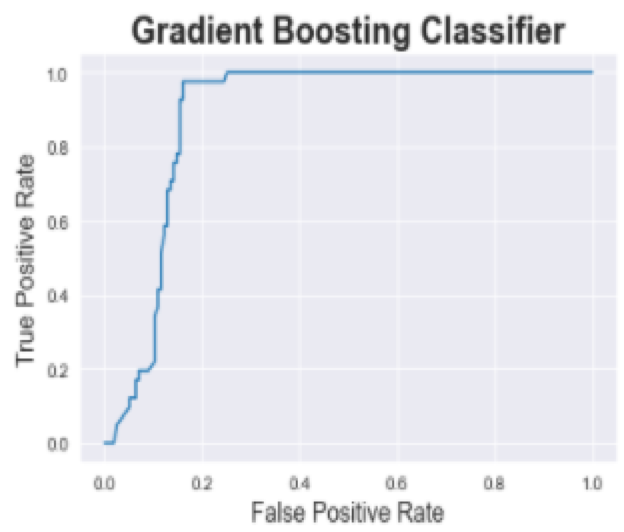
```
]: p3={'loss':['deviance','exponential'],'learning_rate':[0.1,0.01,0.001],'n_estimators':[10,100,150,250],'min_samples_leaf':[0,1,2

gd3=GridSearchCV(gd,p3)
gd3.fit(trainx,trainy)
print(gd3.best_params_)

{'learning_rate': 0.1, 'loss': 'deviance', 'max_depth': 7, 'min_samples_leaf': 2, 'n_estimators': 150}
```

```
]: gd1=GradientBoostingClassifier(learning_rate=0.01,max_depth=3,n_estimators=150,min_samples_leaf=2)
```

```
]: fun(gd1)

Accuracy Score 86.22448979591837
Confusion Matrix
 [[130  25]
 [  2  39]]
Classification Report
               precision    recall  f1-score   support

           0       0.98      0.84      0.91       155
           1       0.61      0.95      0.74        41

    accuracy                           0.86       196
   macro avg       0.80      0.89      0.82       196
weighted avg       0.91      0.86      0.87       196

F1 score 74.28571428571428
```

Score Improve After Hyper Tunning =86.22-81.63
=4.59

ROC-AUC Curve: - It is the performance measurement of the model at diff diff thresholds. ROC is the performance score and AUC is the separation score means how much mode classify 0 as 0 and 1 as 1.



Cross Validation: -

This technique is used to check weather out data set is over fitting or under fitting. If model score is high and cv score is less it means model perform well in train dataset but did not perform well in unseen or test dataset. Feature selection is the best way to overcome the overfitting problem. There are 3 ways for the validation. KFold Cross validation score, Hold Out Methods and LOOCV.

KFold: - In this technique it will rotate the data into the k-fold times. Suppose k=3

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

1st Iteration: 1-3 as Test and 4-9 Train
2nd Iteration:4-6 as Test and 1-3 & 7-9 Train
3rd Iteration:7-9 as Test and 1-6 as Train
It means all the data (9 rows) go for training.

```
for i in range(2,16):
    score=cross_val_score(gd1,trainx,trainy,cv=i)
    print("score at cv=",i,score.mean()*100)
```

```
score at cv= 2 86.4957264957265
score at cv= 3 85.81196581196582
score at cv= 4 86.15468231333864
score at cv= 5 86.23931623931624
score at cv= 6 86.4957264957265
score at cv= 7 86.24078373864516
score at cv= 8 86.24021526418787
score at cv= 9 86.4957264957265
score at cv= 10 86.23931623931625
score at cv= 11 86.24420897388629
score at cv= 12 86.41121397012414
score at cv= 13 86.58119658119661
score at cv= 14 86.49495942955495
score at cv= 15 86.58119658119661
```

LOOCV: Leave one out cross validation
It will take one row for test and remaining for training so each and every row go for test so its time-consuming processing

Concluding Remarks: -

From this model we can detect the auto insurance fraud.by using this model loss of the company can be reduced.

We used different classifiers like Logistic Regression, Decision tree Classifiers and gradient boosting classifiers. And also used the data Balanced process and also hyper parameter tunning for improving score.

We get good score in Gradient Boosting Classifiers.F1 score is 85% and Roc AUC Score is 89.the model performance is excellent. Model can distinguish correctly weather the claim is Fraud or Correctly with high accuracy.