Insurance Claims- Fraud Detection Using Machine Learning

❖ Introduction:

For Auto Insurance industry, claim of fraud insurance is a huge problem. People usually give misleading information or sometime provide fake documentation to claim on their personal or commercial vehicles. It becomes very tedious task for Auto Insurance industry to identify whether claim is fraud or not. However, they tried hard to identify the genuine case and provide the benefit.

❖ Problem statement:

Insurance fraud is a huge problem in the industry. It's difficult to identify fraud claims. In this project, a dataset is provided 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.

With the help of given dataset our task is to build an appropriate model that detect auto insurance frauds and will help insurance industry for auto fraud detection.

❖ About Dataset:

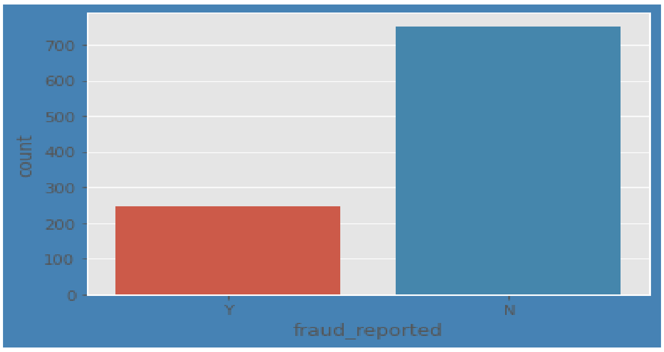
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 40 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   months_as_customer           1000 non-null   int64
 1   age                          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
```

- Dataset contains 1000 rows and 40 columns.
- 21 out of 40 columns are of object type, 17 out of 40 are of int type, 2 out of 40 are of float type.

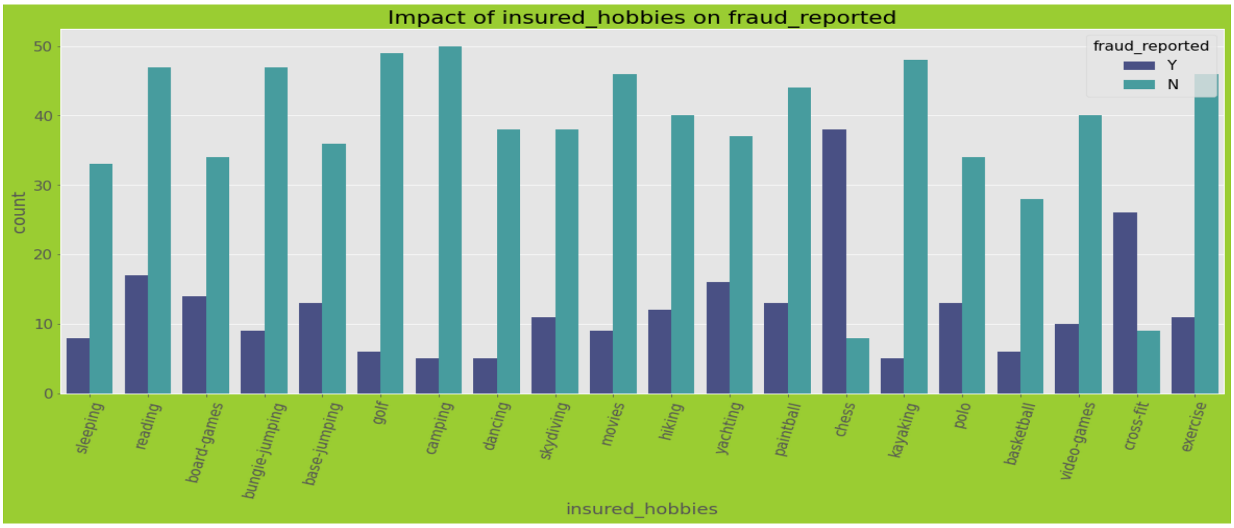
- This problem is belonging to imbalanced classification problem.
- **Imbalanced classification problem** is a classification predictive problem in which number of samples in the dataset for each class is not balanced[1].

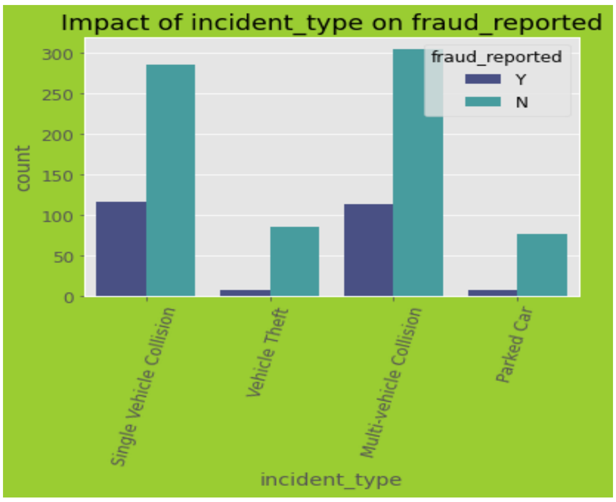
❖ Exploratory Data Analysis:

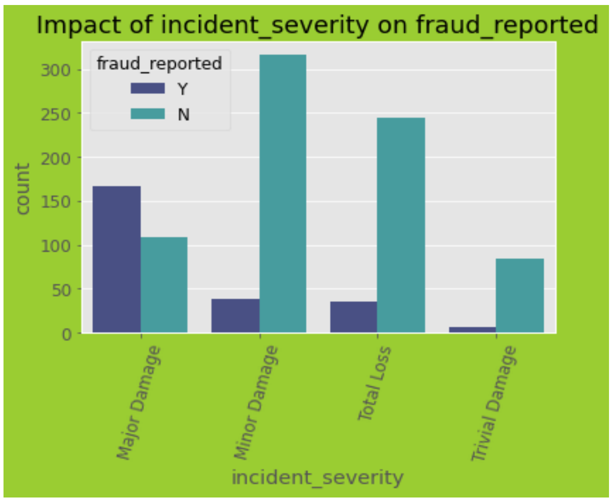
- **Dependent Variable:**

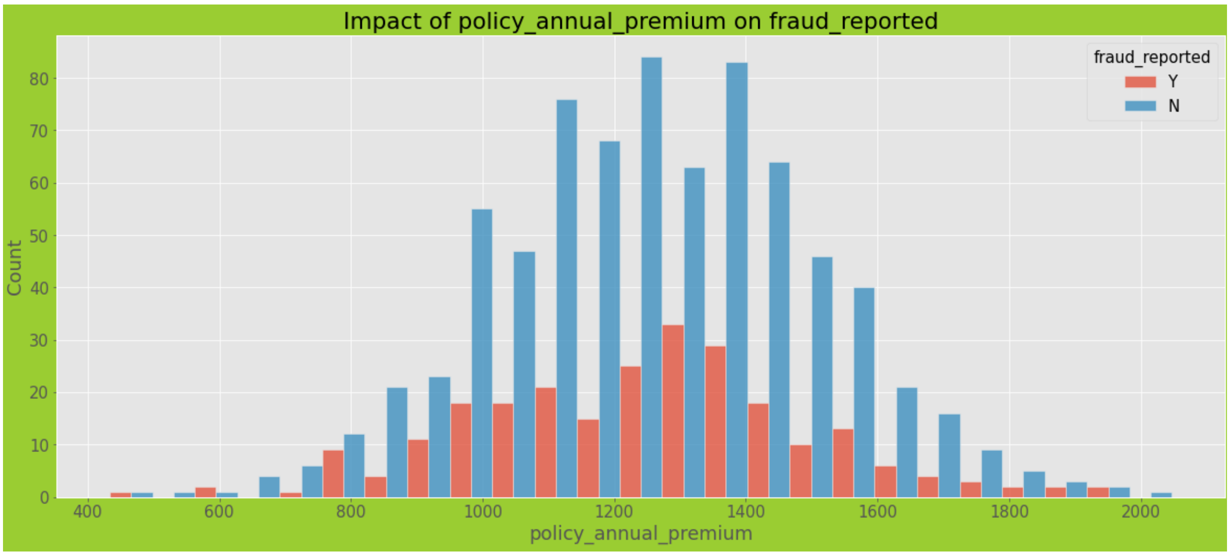


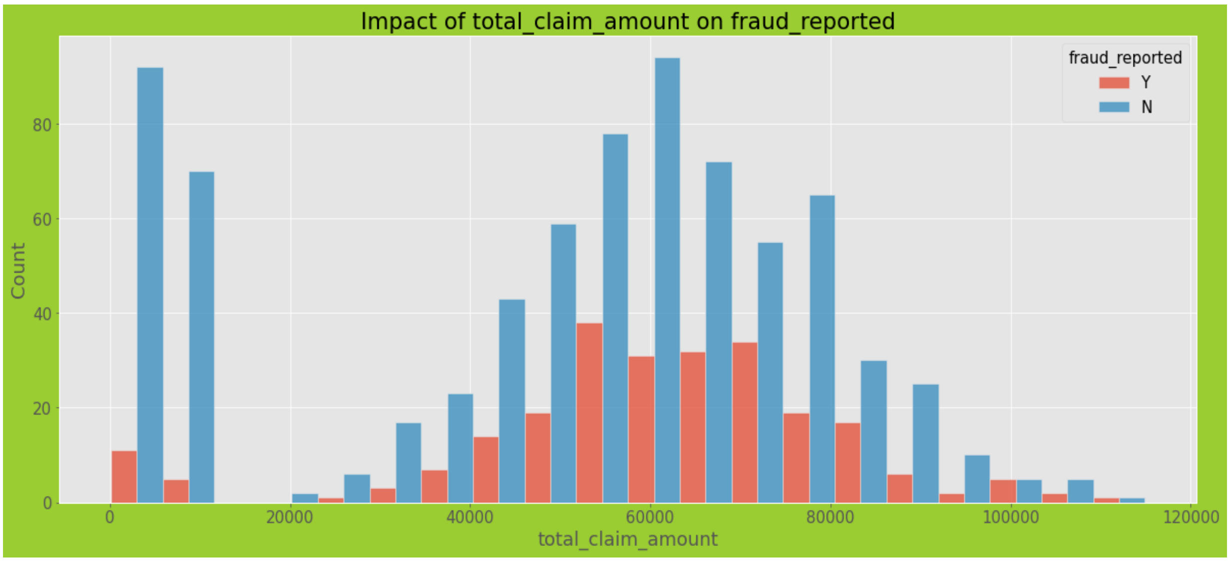
- **Data Visualization:**



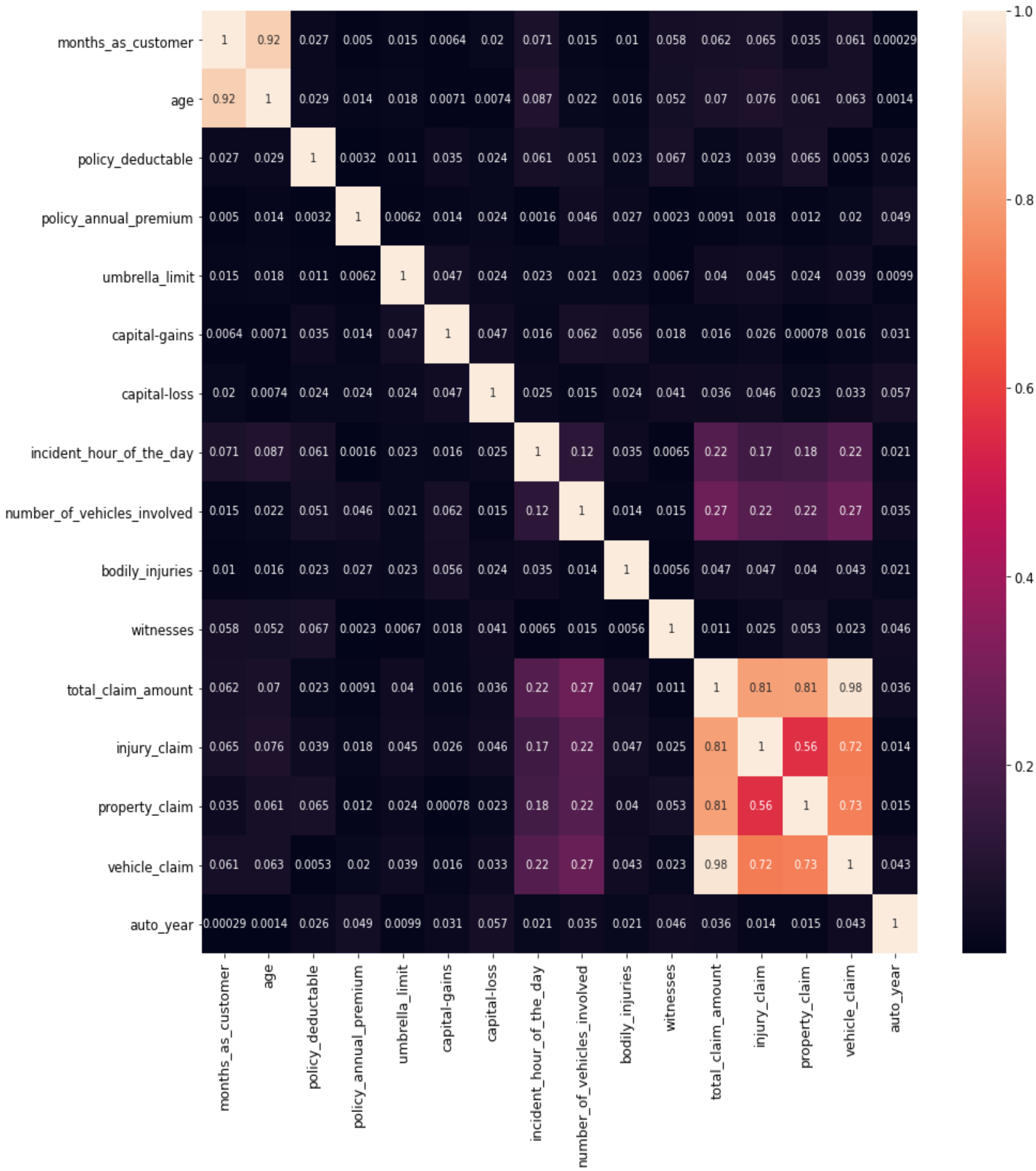





Impact of policy_annual_premium on fraud_reported
fraud_reported
Y
N
Count
0
10
20
30
40
50
60
70
80
400
600
800
1000
1200
1400
1600
1800
2000
policy_annual_premium


Impact of total_claim_amount on fraud_reported
fraud_reported
Y
N
Count
0
20
40
60
80
0
20000
40000
60000
80000
100000
120000
total_claim_amount

- ***Correlation matrix to see the coefficient of multicollinearity:***



- **Multicollinearity** occurs when two or more independent variable highly correlated with one another. It affects the performance of model. Highlighted features are highly correlated to each other are:

❖ Data Pre-processing:

- Data pre-processing includes:
  - **Treating null values:** However, now this dataset does not contain any null value. If there is null value in the dataset, null values can be treated by different technique. Few techniques are:
    - Deleting rows that contains null values.
    - Imputing missing values for continuous column by mean, median and for categorical values by mode.
    - Different Imputation methods.
    - Predicting the value for missing values.
    - Using algorithm which support missing values.
  - **Feature Extraction:** Feature extraction refers to the process of transforming the data into numerical feature. 'policy_bind_date' and 'incident_date' are those columns from which we will have to extract Day, Month and Year.
  - **Data Cleaning:** Data cleaning refers to the process of identifying incorrect, incomplete, inaccurate, irrelevant part of the data and then modifying, replacing or deleting accordingly. 'property_damage', 'police_report_available' and collision_type' contain irrelevant data as '?' so, need to replace it. One row contains negative value in 'umbrella_limit' so need to drop it.

❖ Descriptive Statistical Analysis:

	months_as_customer	age	policy_deductable	policy_annual_premium	umbrella_limit	capital-gains	capital-loss	incident_hour_of_the_day
count	999.000000	999.000000	999.000000	999.000000	9.990000e+02	999.000000	999.000000	999.000000
mean	203.873874	38.944945	1136.636637	1256.323934	1.103103e+06	25151.251251	-26820.520521	11.642643
std	115.142928	9.144354	611.839681	244.275843	2.297594e+06	27874.792269	28105.366259	6.954722
min	0.000000	19.000000	500.000000	433.330000	0.000000e+00	0.000000	-111100.000000	0.000000
25%	115.500000	32.000000	500.000000	1089.185000	0.000000e+00	0.000000	-51500.000000	6.000000
50%	199.000000	38.000000	1000.000000	1257.040000	0.000000e+00	0.000000	-24100.000000	12.000000
75%	276.000000	44.000000	2000.000000	1415.710000	0.000000e+00	51050.000000	0.000000	17.000000
max	479.000000	64.000000	2000.000000	2047.590000	1.000000e+07	100500.000000	0.000000	23.000000

8 rows × 22 columns

	number_of_vehicles_involved	bodily_injuries	...	injury_claim	property_claim	vehicle_claim	auto_year	policy_bind_Date	policy_bind_Month
count	999.000000	999.000000	...	999.000000	999.000000	999.000000	999.000000	999.000000	999.000000
mean	1.839840	0.992993	...	7432.292292	7389.839840	37898.368368	2005.112112	15.458458	6.561562
std	1.019044	0.819936	...	4883.266266	4817.316312	18870.924206	6.011966	8.848424	3.392489
min	1.000000	0.000000	...	0.000000	0.000000	70.000000	1995.000000	1.000000	1.000000
25%	1.000000	0.000000	...	4290.000000	4440.000000	30275.000000	2000.000000	8.000000	4.000000
50%	1.000000	1.000000	...	6770.000000	6750.000000	42080.000000	2005.000000	16.000000	7.000000
75%	3.000000	2.000000	...	11310.000000	10870.000000	50775.000000	2010.000000	23.000000	9.000000
max	4.000000	2.000000	...	21450.000000	23670.000000	79560.000000	2015.000000	31.000000	12.000000

	policy_bind_Year	incident_Date	incident_Month	incident_Year
count	999.000000	999.000000	999.000000	999.0
mean	2001.610611	13.068068	3.409409	2015.0
std	7.361107	10.436245	3.277046	0.0
min	1990.000000	1.000000	1.000000	2015.0
25%	1995.000000	2.000000	1.000000	2015.0
50%	2002.000000	15.000000	2.000000	2015.0
75%	2008.000000	22.000000	5.000000	2015.0
max	2015.000000	31.000000	12.000000	2015.0

- The difference between mean and 50 % shows that there is skewness in dataset.
- The huge difference between 75 % and max shows that there is outliers in dataset.

❖ Features Engineering:

- **Dealing with outliers:**

  Outliers are data that is different from others.

  - There are different techniques to deal with outliers like IQR method, zscore method.
  - In the dataset, the columns that contain outliers are: 'age', 'umbrella_limit', 'property_claim', 'total_claim_amount', 'incident_Month'.
  - To deal with these outliers z-score is used which is member of stats in scipy.

```
The shape before outlier remove is : (999, 40)
The shape after outlier remove is : (983, 40)
The loss of data in percentage is : 1.6016016016016015
```

- **Dealing with skewness:**

  Skewness refers to the measure of how much the probability distribution of a random variable deviates from the normal distribution[2]. Different type of skewness is Left, Right, Symmetric skewness.

  - There are different technique to deal with skewness:
    - Log transformation
    - Normalize
    - Cube root
    - Square root etc
  - In the dataset some skewness is present.
  - For this dataset, Skewness will be treated +/-0.5. The skewness of categorical and target variable will not be treated.
  - 'yeo-johnson' method of Power Transformer is used to deal with skewness.

- **Encoding categorical column:** As machine learning model understands numerical values or we can say that for a good model we need data in numerical form. To deal with this, machine learning has different type of encoder which converts object type of columns/variable into numeric by encoding them or we can handle it by replace/custom mapping.

  - Few encoding techniques are:
    - Label Encoding
    - Ordinal Encoding
    - One Hot Encoding
    - Dummy Encoding
    - Binary Encoding
  - Label encoder is used for encoding categorical variable.

```python
from sklearn.preprocessing import LabelEncoder
```

```python
lb = LabelEncoder()
```

```python
df_o['incident_Year'] = lb.fit_transform(df_o['incident_Year'])
df_o['incident_Month'] = lb.fit_transform(df_o['incident_Month'])
df_o['incident_Date'] = lb.fit_transform(df_o['incident_Date'])
df_o['policy_bind_Year'] = lb.fit_transform(df_o['policy_bind_Year'])
df_o['policy_bind_Month'] = lb.fit_transform(df_o['policy_bind_Month'])
df_o['policy_bind_Date'] = lb.fit_transform(df_o['policy_bind_Date'])
df_o['policy_state'] = lb.fit_transform(df_o['policy_state'])
df_o['policy_csl'] = lb.fit_transform(df_o['policy_csl'])
df_o['insured_sex'] = lb.fit_transform(df_o['insured_sex'])
df_o['insured_education_level'] = lb.fit_transform(df_o['insured_education_level'])
df_o['insured_occupation'] = lb.fit_transform(df_o['insured_occupation'])
df_o['insured_hobbies'] = lb.fit_transform(df_o['insured_hobbies'])
df_o['insured_relationship'] = lb.fit_transform(df_o['insured_relationship'])
df_o['incident_type'] = lb.fit_transform(df_o['incident_type'])
df_o['collision_type'] = lb.fit_transform(df_o['collision_type'])
df_o['incident_severity'] = lb.fit_transform(df_o['incident_severity'])
df_o['authorities_contacted'] = lb.fit_transform(df_o['authorities_contacted'])
df_o['incident_state'] = lb.fit_transform(df_o['incident_state'])
df_o['incident_city'] = lb.fit_transform(df_o['incident_city'])
df_o['property_damage'] = lb.fit_transform(df_o['property_damage'])
df_o['police_report_available'] = lb.fit_transform(df_o['police_report_available'])
df_o['auto_make'] = lb.fit_transform(df_o['auto_make'])
df_o['auto_model'] = lb.fit_transform(df_o['auto_model'])
df_o['fraud_reported'] = lb.fit_transform(df_o['fraud_reported'])
```

- **Scaling the independent variable:** Scaling is a method to normalize independent variable. In machine learning different technique are available to normalize the data.
  - Few techniques used for scaling:
    - Absolute Maximum Scaling
    - Min-Max Scaling
    - Standardization
  - After selecting X and y, Standard Scaler is used to scale independent variable.

- **Dealing with multicollinearity:**_The best way to deal with multicollinearity is VIF. VIF is variance inflation factor which is member of stats.outliers_influence of statsmodels.
  - For each feature calculate VIF score and try to drop those independent features which have highest VIF. Check VIF again.
  - The standard threshold for VIF is 5. However, sometimes it depends upon the dataset.
  - In the given dataset, total_claim_amount has highest Vif followed by vehicle_claim, injury_claim, property_claim. So, after dropping total_claim_amount there is no multicollinearity problem exist in the dataset.

- **Dealing with imbalancing:** The best way for balancing the dataset is SMOTE algorithm.
  - SMOTE algorithm works on two method oversampling or under sampling.
  - **Oversampling:** Based on given data create new entries for the minority class.
    Under sampling: Based on given data randomly delete entries for the minority class.
  - Oversampling is used for dealing this imbalancing present in target variable.

❖ Machine Learning Model:

- There are several machine learning models available in sklearn library for regression and classification problem.
- The target variable 'fraud_reported' contains two values whether the claim is fraud or not. So, the problem is binary classification problem.
- Before fit the dataset, split the dataset using Train Test Split.
- Train Test split is a mechanism by which we can split our dataset into train and test set.
- In this project dataset is split into 70:30 ratio i.e. train set contains 70% data on which model is going to be trained and test set contains 30% data on which trained model makes prediction.

```
The shape of X_train is : (1038, 37)
The shape of X_test is : (446, 37)
The shape of y_train is : (1038,)
The shape of y_test is : (446,)
```

- In this project 9 machine learning models are selected. The models are:
  - Logistic Regression
  - K-Neighbors Classifier
  - Decision Tree Classifier
  - Random Forest Classifier
  - Support Vector Machine Classifier
  - Bagging Classifier
  - Gradient Boosting Classifier
  - Ada Boost Classifier
  - XGB Classifier

- **Initiate the model:**
  - Logistic Regression:

```
Logistic Regression
----------------------------------------------------------------

The Score on train set is    : 0.7321772639691715
The Score on test set is     : 0.7645739910313901
The Accuracy on test set is : 0.7645739910313901
The Classification report is :
              precision    recall  f1-score   support

           0       0.72      0.78      0.75       200
           1       0.81      0.76      0.78       246

    accuracy                           0.76       446
   macro avg       0.76      0.77      0.76       446
weighted avg       0.77      0.76      0.77       446

The Confusion matrix is :
 [[155  45]
 [ 60 186]]

----------------------------------------------------------------
```

    - Cross Validation

```
The cross validation score is : 0.7042920847268673
```

- ## K-Neighbors Classifier:

```
K-Neighbors Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.7215799614643545
The Score on test set is     : 0.695067264573991
The Accuracy on test set is  : 0.695067264573991
The Classification report is :
              precision    recall  f1-score   support

           0       0.89      0.36      0.52       200
           1       0.65      0.96      0.78       246

    accuracy                           0.70       446
   macro avg       0.77      0.66      0.65       446
weighted avg       0.76      0.70      0.66       446

The Confusion matrix is :
 [[ 73 127]
 [  9 237]]

-------------------------------------------------------------------
```

  - ### Cross Validation

```
The cross validation score is : 0.6079199182460052
```

- ## Decision Tree Classifier:

```
Decision Tree Classifier
-------------------------------------------------------------------

The Score on train set is    : 1.0
The Score on test set is     : 0.8318385650224215
The Accuracy on test set is  : 0.8318385650224215
The Classification report is :
              precision    recall  f1-score   support

           0       0.80      0.83      0.82       200
           1       0.86      0.83      0.85       246

    accuracy                           0.83       446
   macro avg       0.83      0.83      0.83       446
weighted avg       0.83      0.83      0.83       446

The Confusion matrix is :
 [[166  34]
 [ 41 205]]

-------------------------------------------------------------------
```

  - ### Cross Validation

```
The cross validation score is : 0.7967623560014865
```

- ## Random Forest Classifier:

```
Random Forest Classifier
-------------------------------------------------------------------

The Score on train set is    : 1.0
The Score on test set is     : 0.8968609865470852
The Accuracy on test set is  : 0.8968609865470852
The Classification report is :
              precision    recall  f1-score   support

           0       0.84      0.95      0.89       200
           1       0.95      0.85      0.90       246

    accuracy                           0.90       446
   macro avg       0.90      0.90      0.90       446
weighted avg       0.90      0.90      0.90       446

The Confusion matrix is :
 [[190  10]
 [ 36 210]]

-------------------------------------------------------------------
```

  - ### Cross Validation

```
The cross validation score is : 0.8526291341508733
```

- **Support Vector Machine:**

```
Support Vector Machine Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9595375722543352
The Score on test set is     : 0.8677130044843049
The Accuracy on test set is : 0.8677130044843049
The Classification report is :
               precision    recall  f1-score   support

           0       0.83      0.89      0.86       200
           1       0.90      0.85      0.88       246

    accuracy                           0.87       446
   macro avg       0.87      0.87      0.87       446
weighted avg       0.87      0.87      0.87       446

The Confusion matrix is :
 [[177  23]
 [ 36 210]]

-------------------------------------------------------------------
```

  - Cross Validation

```
The cross validation score is : 0.8362736900780379
```

- **Bagging Classifier:**

```
Bagging Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9903660886319846
The Score on test set is     : 0.8923766816143498
The Accuracy on test set is : 0.8923766816143498
The Classification report is :
               precision    recall  f1-score   support

           0       0.85      0.92      0.88       200
           1       0.93      0.87      0.90       246

    accuracy                           0.89       446
   macro avg       0.89      0.89      0.89       446
weighted avg       0.90      0.89      0.89       446

The Confusion matrix is :
 [[184  16]
 [ 32 214]]

-------------------------------------------------------------------
```

  - Cross Validation

```
The cross validation score is : 0.846808807134894
```

- **Gradient Boosting Classifier:**

```
Gradient Boosting Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9701348747591522
The Score on test set is     : 0.9327354260089686
The Accuracy on test set is : 0.9327354260089686
The Classification report is :
               precision    recall  f1-score   support

           0       0.92      0.94      0.93       200
           1       0.95      0.93      0.94       246

    accuracy                           0.93       446
   macro avg       0.93      0.93      0.93       446
weighted avg       0.93      0.93      0.93       446

The Confusion matrix is :
 [[187  13]
 [ 17 229]]

-------------------------------------------------------------------
```

  - Cross Validation

```
The cross validation score is : 0.8584076551467856
```

- AdaBoost Classifier:

```
AdaBoost Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.8921001926782274
The Score on test set is     : 0.8834080717488789
The Accuracy on test set is : 0.8834080717488789
The Classification report is :
               precision    recall  f1-score   support

           0       0.83      0.93      0.88       200
           1       0.94      0.85      0.89       246

    accuracy                           0.88       446
   macro avg       0.88      0.89      0.88       446
weighted avg       0.89      0.88      0.88       446

The Confusion matrix is :
 [[186  14]
 [ 38 208]]

-------------------------------------------------------------------
```

  - Cross Validation

```
The cross validation score is : 0.8092902266815309
```

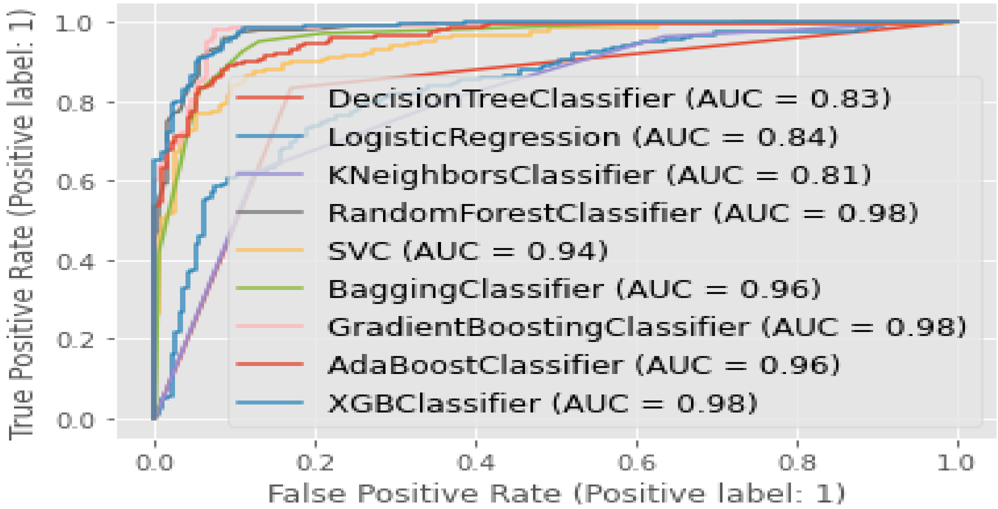
- XGB Classifier:

```
XGB Classifier
-------------------------------------------------------------------

The Score on train set is    : 1.0
The Score on test set is     : 0.9260089686098655
The Accuracy on test set is : 0.9260089686098655
The Classification report is :
               precision    recall  f1-score   support

           0       0.91      0.93      0.92       200
           1       0.94      0.93      0.93       246

    accuracy                           0.93       446
   macro avg       0.92      0.93      0.93       446
weighted avg       0.93      0.93      0.93       446

The Confusion matrix is :
 [[185  15]
 [ 18 228]]

-------------------------------------------------------------------
```

  - Cross Validation

```
The cross validation score is : 0.8641815310293571
```

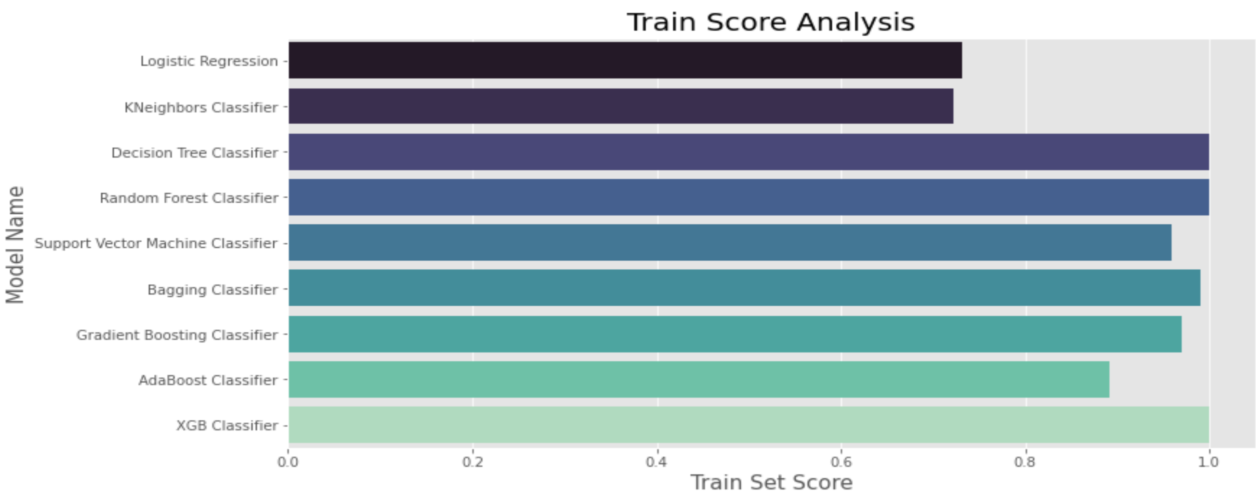
- **ROC Curve:** An ROC curve is a graph showing the performance of classification model at all classification thresholds[3].
  - The curve plot two parameter:
    - True Positive Rate
    - False Positive Rate
  - Here is the graph for all model

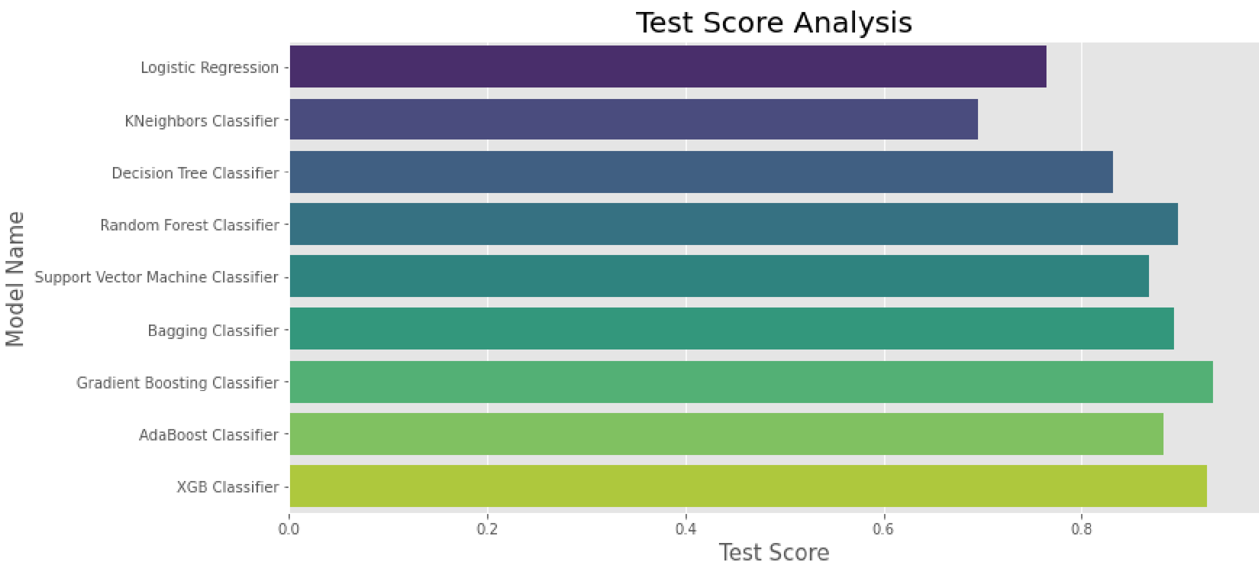


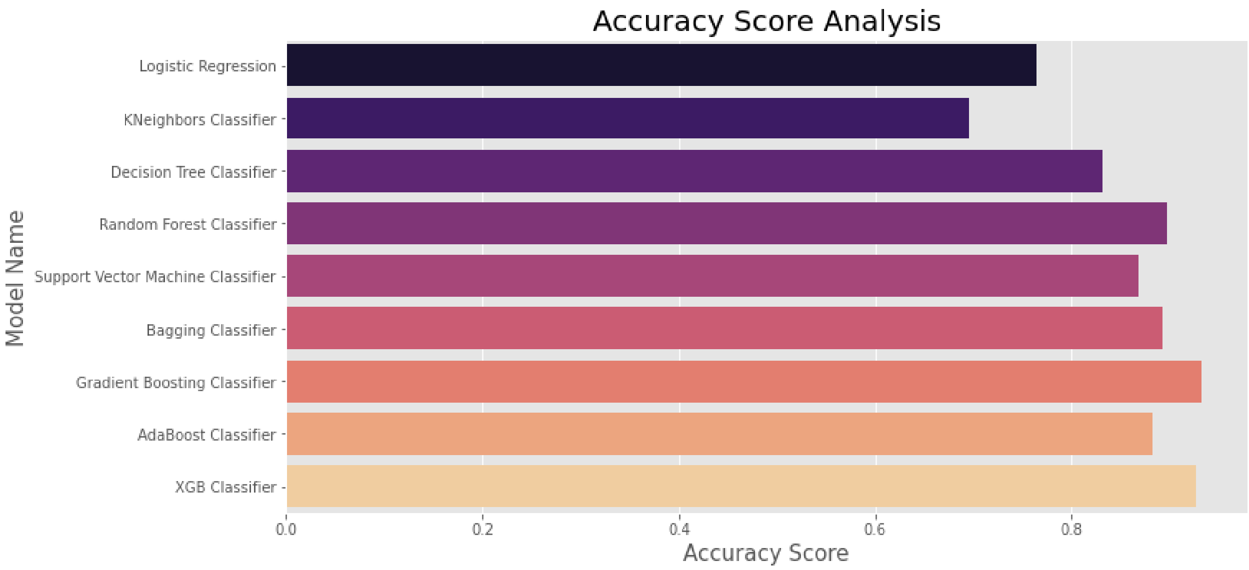
    - From above graph Random Forest Classifier, Gradient Boosting Classifier, XGB Classifier have best score.

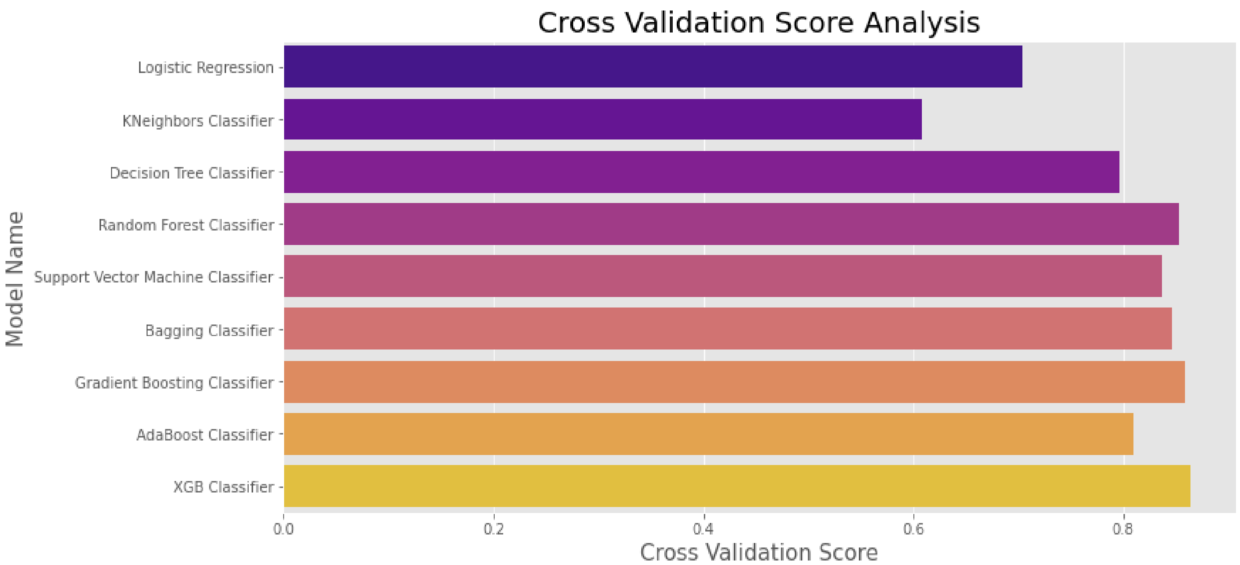
- **Observing Score of all model:**

	Model Name	Train Score	Test Score	Accuracy Score	Cross Validation Score
0	Logistic Regression	0.732177	0.764574	0.764574	0.704292
1	KNeighbors Classifier	0.721580	0.695067	0.695067	0.607920
2	Decision Tree Classifier	1.000000	0.831839	0.831839	0.796762
3	Random Forest Classifier	1.000000	0.896861	0.896861	0.852629
4	Support Vector Machine Classifier	0.959538	0.867713	0.867713	0.836274
5	Bagging Classifier	0.990366	0.892377	0.892377	0.846809
6	Gradient Boosting Classifier	0.970135	0.932735	0.932735	0.858408
7	AdaBoost Classifier	0.892100	0.883408	0.883408	0.809290
8	XGB Classifier	1.000000	0.926009	0.926009	0.864182









- Random Forest Classifier, Gradient Boosting Classifier and XGB Classifier all model are good as they gives better accuracy. The difference between accuracy and cross validation score is Random Forest Classifier which has less score. F1 score is 0.85. So, Random Forest Classifier is best model.

- **Hyperparameter Tuning:** The Hyperparameter tuning is a technique of choosing optimal hyperparameter for learning model. Hyperparameter is a parameter which value is used to control the learning process[4]. The objective of hyperparameter tuning is to improve the performance of model. It also observed the result of hyperparameter tuning is not always improving the performance of model.
  - Different technique also available for hyperparameter tuning. Few techniques are:
    - Random Search: Based on a random search on hyperparameter.
    - Grid Search: Based on an exhaustive search on hyperparameter.

  - Here Grid Search CV is used on Random forest classifier for Hyperparameter Tuning.

```
Best parameter {'criterion': 'entropy', 'max_features': 'log2', 'min_samples_split': 3, 'n_estimators': 100}

 Confusion Matrix
 [[188  12]
 [ 24 222]]
Accuracy after hyper parameter tunning 0.9192825112107623
```

    - After hyperparameter tuning, the model performance is improved.
    - Fit the model using these hyperparameter and save the model.

- **Load the model and see prediction:** Loading the saved model and observe how well this model is predicted on test set.

	0	1	2	3	4	5	6	7	8	9	...	436	437	438	439	440	441	442	443	444	445
Prediction	0	0	1	1	1	1	0	1	1	0	...	0	1	1	1	0	1	1	1	0	0
Actual	0	0	1	1	1	1	0	1	1	0	...	0	1	1	1	0	1	1	1	0	0

❖ Conclusion:

- The objective behind building this model is to detect the fraud automatically to help auto insurance industry in eliminating the frauds.
- This model will also help in economic growth of insurance industry.
- The term growth is used because if a model is able to detect auto fraud then it will help in reducing finical loss and there will be economic growth in insurance industry.
- Nine different models are used for this classification problem.
- **Data imbalancing** is most general problem found in classification dataset. To handle imbalancing, SMOTE is used by oversampling the data.
- **Standardization** and **Train Test split** are used to normalize and split the dataset respectively.

- In this project, the best performing model on given dataset is **Random Forest Classifier**.
- Accuracy of model is 89.68%, ROC AUC score is 0.98 and f1 score is 0.85. Difference between Accuracy and CV score is less as compared with other best model. All metric are good as compared with other model. Basically, good metrics are the criteria for best model.

Written by: Rohit Kachhal