INSURANCE CLAIMS FRAUD DETECTION USING MACHINE LEARNING TECHNIQUES

MSCP34 – Mini Project

PROJECT REPORT

Submitted By,
Linet M Shaji
P191314

Guided By,
Dr. Martin

MASTER OF SCIENCE
In
COMPUTER SCIENCE

CENTRAL UNIVERSITY OF TAMIL NADU
THIRUVARUR - 610 005
November 2020

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **INSURANCE CLAIMS FRAUD DETECTION USING MACHINE LEARNING TECHNIQUES"** is the bonafide work of **Linet M Shaji (P191314)**who carried out the project under my supervision for the course title **MSCP34 – Mini Project.**

Dr. A. MARTIN
Department of Computer Science
Central University of Tamil Nadu
Thiruvarur – 610 005

DECLARATION

I hereby declare that this project titled **INSURANCE CLAIMS FRAUD DETECTION USING MACHINE LEARNING TECHNIQUES"** submitted for the course MSCP34 – Mini Project, Department of Computer Science, School of Mathematics and Computer Science, Central University of Tamil Nadu, Thiruvarur – 610 005, is a record of bonafide project work carried out by us under the guidance and supervision of **Dr. A. Martin** , Department of Computer Science, Central University of Tamil Nadu, Thiruvarur – 610 005. This work is original and has not been submitted, in part or full to this or any other University / Institution.

Place: Thiruvarur

Date: 09-11-2020

Linet M Shaji (P191314)

ABSTRACT

Vehicle insurance is a contract between you and the insurance company that protects you against financial loss in the case of an accident or theft.If a person or entity make false information to obtain compensation or benefits of this kind of insurance is called fraud insurance claim.The detection of an insurance fraud is a challenging problem for the insurance industry.Since there is no perfect system to prevent this kind of fraudulent activities,so it has become a challenging task to make a secure system for authentication and preventing frauds.

This project is about predicting the insurance claim as fraud or not using machine learning algorithms. This project uses two models such as decision tree algorithm and gradient boosting algorithm for classification.

TABLE OF CONTENTS

1.INTRODUCTION

Vehicle insurance is a contract between you and the insurance company that protects you against financial loss in the case of an accident or theft.If a person or entity make false information to obtain compensation or benefits of this kind of insurance is called fraud insurance claim.The detection of an insurance fraud is a challenging problem for the insurance industry.Since there is no perfect system to prevent this kind of fraudulent activities,so it has become a challenging task to make a secure system for authentication and preventing frauds.

This project is about predicting the insurance claim as fraud or not using machine learning algorithms. This project uses two models such as decision tree algorithm and gradient boosting algorithm for classification.

2.PROBLEM DEFINITION

2.1 PROBLEM STATEMENT

The objective of this project is to build a model using machine learning to predict whether vehicle insurance claim is genuine or fraud.We can achieve this by performing different kind of analysis on the data set, creating subset using essential features and by spliting and building model.

The challenge behind fraud detection in machine learning is that frauds are less common as compared to legit(government) insurance claims. This type of problems is known as imbalanced class classification.

About the dataset

https://www.kaggle.com/roshansharma/insurance-claim

The dataset is U.S based insurance claim dataset,it contains of 1000 insurance claims (rows) and 39 features (columns). It contains features such as policy_number, insured_zip, property_claim, bodily_injury, incident_state, incident_severity, incident_hour_of_the_day etc.The given dataset is labeled. It contains a column called fraud_reported which enables us to classify the claim as fraud or not fraud.

The aim of this project is to use machine learning algorithm for classification. Here the dataset is labeled so we can use supervised machine learning algorithm.Supervised learning is a machine learning techniques in which machines are trained using labeled data (some input data that is already tagged with the output) and it will predict the output based on the analysis of data if some new data are given.In the real-world, supervised learning is used for Image classification, Fraud Detection, spam filtering, etc.

3.ALGORITHM DESIGN

In this project we used two models such as decision tree algorithm and gradient boosting algorithm for classifying claim as fraud or not fraud.

3.1DECISION TREE ALGORITHM

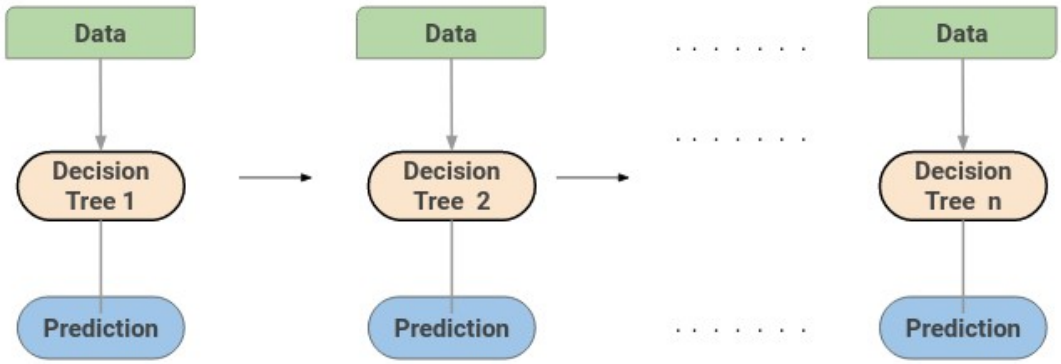
A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome.

Important Terminology related to Decision Trees

- **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
- **Splitting:** It is a process of dividing a node into two or more sub-nodes.
- **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
- **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.
- **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
- **Branch / Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.
- **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

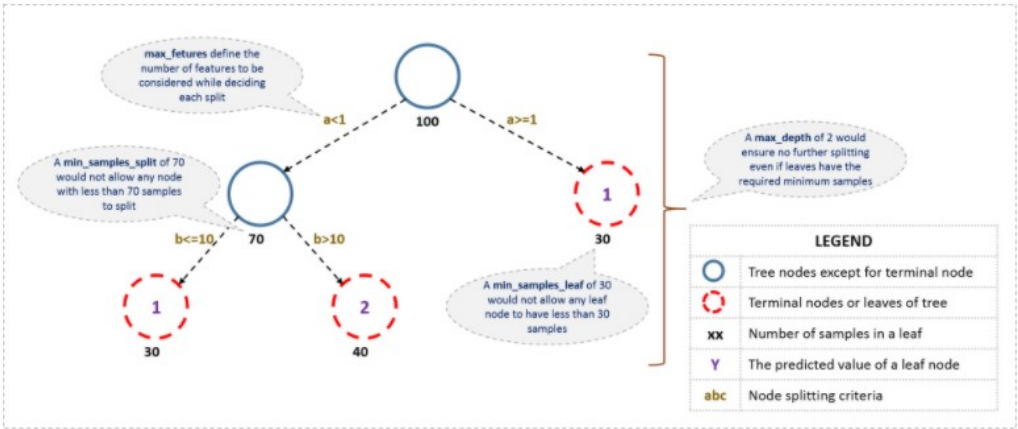
3.2GRADIENT BOOSTING ALGORITHM

A Gradient Boosting Machine or GBM combines the predictions fro m multiple decision trees to generate the final predictions. Keep in mind that all the weak learners in a gradient boosting machine are decision trees.



The overall parameters of this ensemble model can be divided into 3 categories:

1. **Tree-Specific Parameters:** These affect each individual tree in the model.
2. **Boosting Parameters:** These affect the boosting operation in the model.
3. **Miscellaneous Parameters:** Other parameters for overall functioning.



min_samples_split

- Defines the minimum number of samples (or observations) which are required in a node to be considered for splitting.

min_samples_leaf

- Defines the minimum samples (or observations) required in a terminal node or leaf.

min_weight_fraction_leaf

- Similar to min_samples_leaf but defined as a fraction of the total number of observations instead of an integer.

max_depth

- The maximum depth of a tree.

max_leaf_nodes

- The maximum number of terminal nodes or leaves in a tree.

max_features

- The number of features to consider while searching for a best split. These will be randomly selected.

learning_rate

- This determines the impact of each tree on the final outcome.

n_estimators

- The number of sequential trees to be modeled.

subsample

- The fraction of observations to be selected for each tree. Selection is done by random sampling.

CONFUSION MATRIX

Confusion matrix is one of the easiest and most intuitive metrics us ed for finding the accuracy of a classification model, where the out put can be of two or more categories

True Positive

True positive is nothing but the case where the actual value as well as the predicted value are true that is the actual value of claim is not fraud, and the model also predicted as not fraud.

True Negative

This is the case where the actual value is false and the predicted value is also false. In other words,that is the actual value of claim is not fraud, and the model also predicted as fraud.

False Negative

In false negative, the actual value is true, but the predicted value is false, which means that the actual value of claim is fraud, and the model also predicted as not fraud.

False Positive

This is the case where the predicted value is true, but the actual value is false. Here, the model predicted as fraud, but the actual value is fraud.

4.IMPLEMENTATION & SCREENSHOTS

Import libraries and load dataset

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt # visualization by plotting
import seaborn as sns #heat map
%matplotlib inline
```

Exploratory Data Analysis

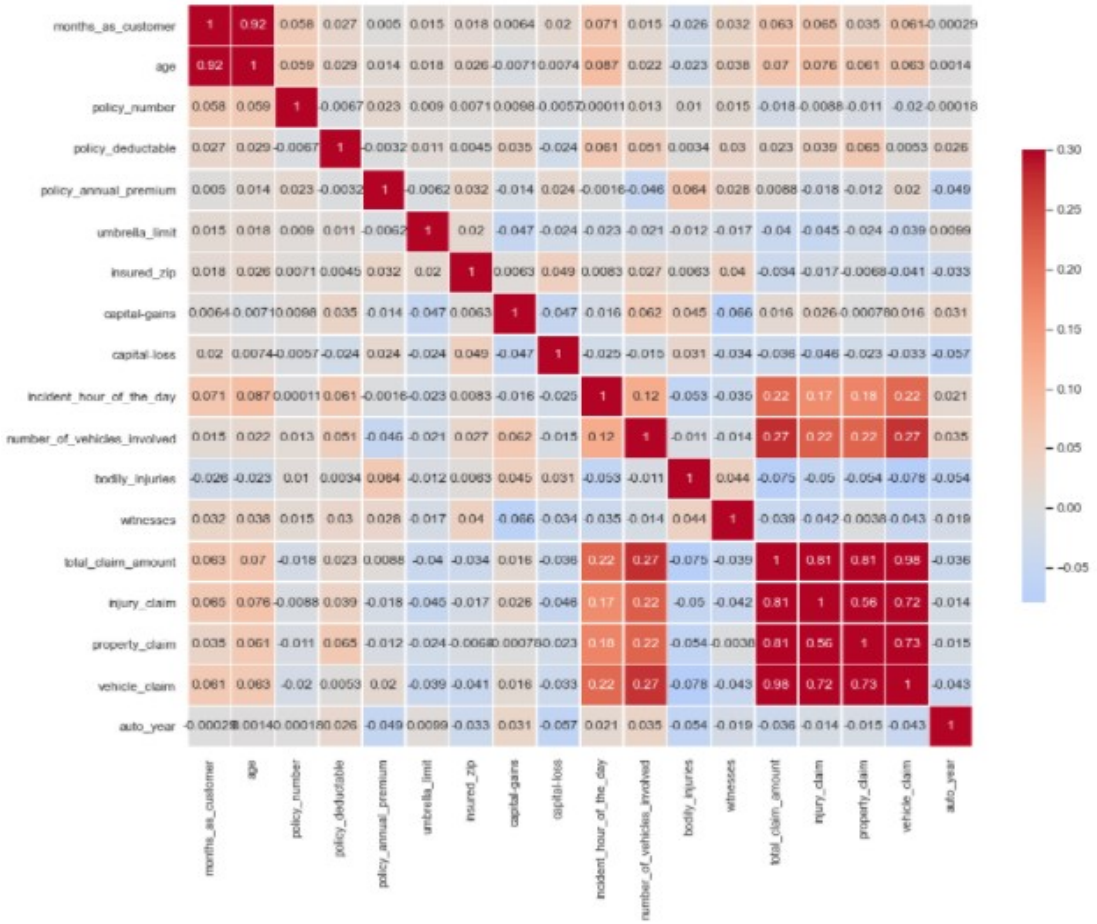
Exploratory data analysis was conducted started with the dependent variable, Fraud_reported. There were 247 frauds and 753 non-frauds. 24.7% of the data were frauds while 75.3% were non-fraudulent claims.

```
f, ax = plt.subplots(figsize=(10, 5))
sns.countplot(x='fraud_reported',data=data,palette='Set2')
```

Correlations among variables

Next, correlations among continuous variables were plotted. Heatmap was plotted for variables with at least 0.3 Pearson's correlation coefficient , including the DV.

```
sns.set(style="white")
# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(15, 15))
sns.heatmap(data.corr(),cmap='coolwarm',vmax=.3,center=0,annot=True,square=True, linewidths=.5, cbar_kws={"shrink": .5})
```



Remove features

Remove features based on heat map and plotting.

```
data.drop(['incident_location','policy_bind_date','incident_date','auto_model','insured_occupation','policy_number','policy_deductable','policy_annual_premium','umbrella_limit','insured_zip','insured_sex','capital-gains','capital-loss','insured_education_level', 'insured_occupation', 'insured_hobbies'],axis=1,inplace=True)
```

Label encoding to make more features

The target value of the dataset is string datatype.So we need to convert it into binary values as 0,1 for training and modeling.

```
le = LabelEncoder()
le_count = 0
for col in data:
    if data[col].dtype == 'object':
        # If 2 or fewer unique categories
        if len(list(data[col].unique())) <= 2:
            # Train on the training data
            le.fit(data[col])
            # Transform both training and testing data
            data[col] = le.transform(data[col])
            # Keep track of how many columns were label encoded
            le_count += 1
print('%d columns were label encoded.' % le_count)
```

Split dataset into training and training

Create dummy values

```
data = pd.get_dummies(data)
print('Training Features shape: ', data.shape)
```

Selects features and target value for modeling.

Chosen fraud_reported as target value for classification.

y=data['fraud_reported']

Chosen all values except fraud_reported as features

X= data.drop('fraud_reported',axis=1)

Import train_test_split library for splitting dataset into tarining and testing data.

from sklearn.model_selection import train_test_split

Split dataset into training(70%) and testing(30%)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

Modelling

Two different classifiers were used in this project:

- Decision Tree

A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome.

from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier

from sklearn import metrics

Create Decision Tree classifer

clf = DecisionTreeClassifier()

Train Decision Tree Classifer

clf = clf.fit(X_train,y_train)

Predict the response for test dataset

y_pred = clf.predict(X_test)

print(y_pred)

Model Accuracy, how often is the classifier correct?

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

predictors=list(X_train)

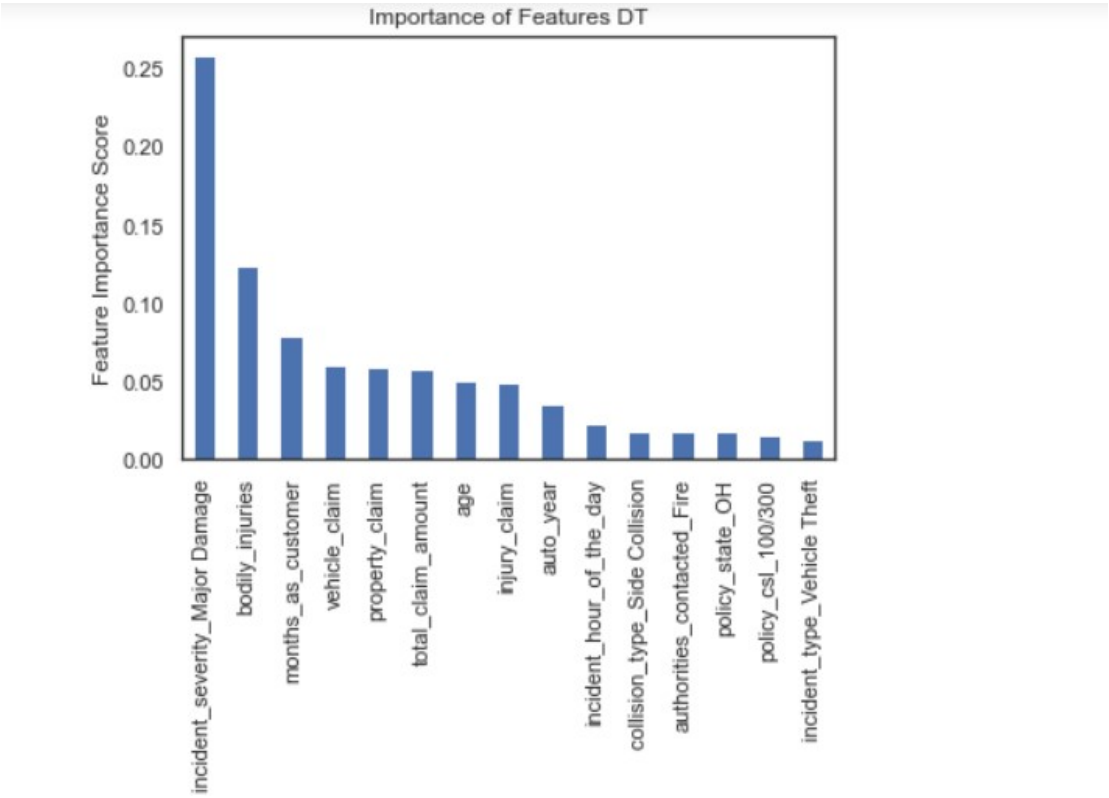
Plots feature important used by the decision tree

feat_imp = pd.Series(clf.feature_importances_, predictors).sort_values(ascending=False)

feature = feat_imp.head(15)

feature.plot(kind='bar', title='Importance of Features')

plt.ylabel('Feature Importance Score')



Import classification_report,confusion_matrix to check accuracy and classification report.

```
from sklearn.metrics import classification_report,confusion_matrix

print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.83	0.88	0.86	223
1	0.59	0.48	0.53	77
accuracy			0.78	300
macro avg	0.71	0.68	0.69	300
weighted avg	0.77	0.78	0.77	300

Confusion matrix

```
print(confusion_matrix(y_test, y_pred))

[[197  26]
 [ 40  37]]
```

The actual value of claim is not fraud, and the model also predicted as not fraud

True positive 192

Model predicted as fraud, but the actual value is fraud

False positive 26

The actual value of claim is fraud, and the model also predicted as not fraud.

False negative 37

The actual value of claim is not fraud, and the model also predicted as fraud.

True negative 40

- Gradient Boosting

Gradient Boosting Algorithm is a Boosting Algorithm.Gradient Boosting Algorithm is a sequential process.Based on the number of tree we have been given it will create many decision trees.Each decision tree take uses previous tree's output to create new decision tree and combines previous decision trees and new decision trees output to calculate values to create next decision tree.

Import GradientBoostingClassifier library for using GBM model

```
from sklearn.ensemble import GradientBoostingClassifier
```

Create GDM classifer

```
model1= GradientBoostingClassifier(n_estimators=20, learning_rate=0.5, max_depth=2, random_state=10)
```

```
model1.fit(X_train,y_train)
```

```
predictors=list(X_train)
```

Plot important features used by the GBM model to predict output

```
feat_imp = pd.Series(model1.feature_importances_, predictors).sort_values(ascending=False)
```

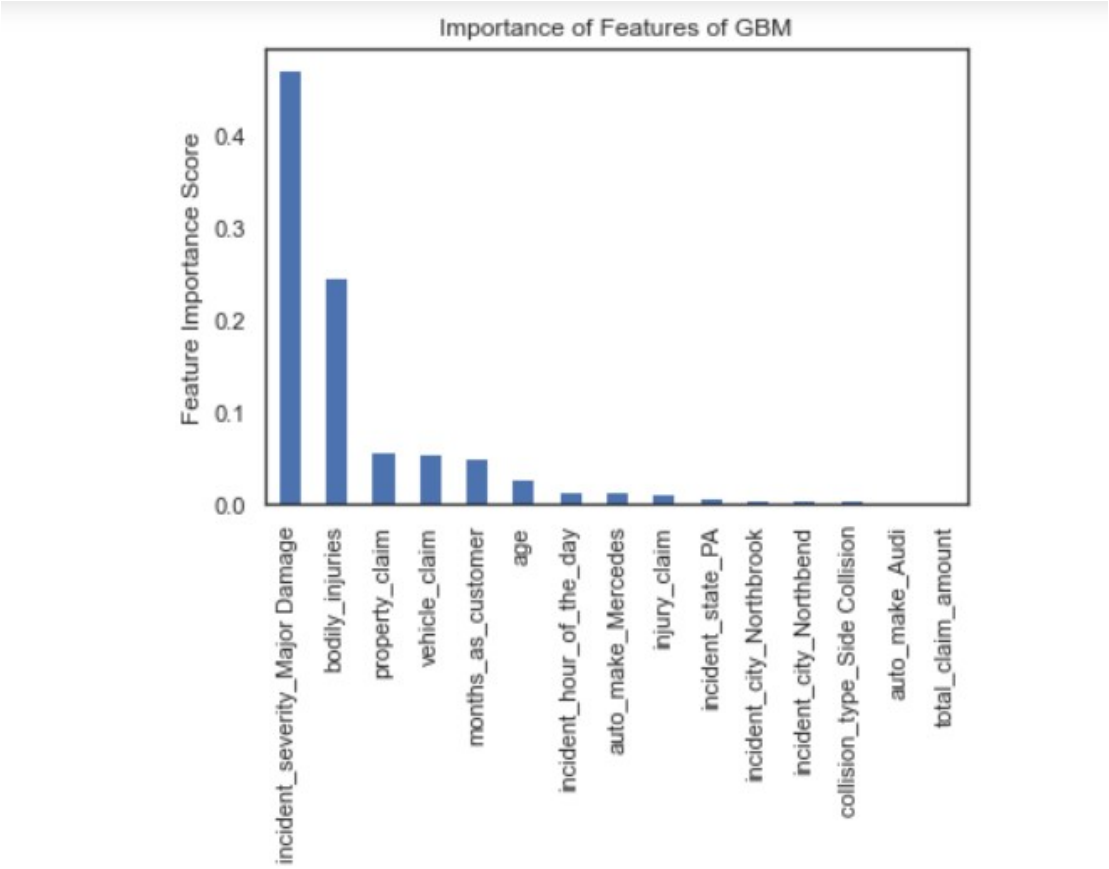
```
feature = feat_imp.head(15)
```

```
feature.plot(kind='bar', title='Importance of Features')
```

```
plt.ylabel('Feature Importance Score')
```

print('Accuracy of the GBM on test set: {:.3f}'.format(model1.score (X_test, y_test)))

Accuracy of the GBM on test set: 0.810



from sklearn.metrics import classification_report,confusion_matrix

print(classification_report(y_test, pred))

	precision	recall	f1-score	support
0	0.84	0.91	0.88	223
1	0.67	0.51	0.58	77
accuracy			0.81	300
macro avg	0.76	0.71	0.73	300
weighted avg	0.80	0.81	0.80	300

Confusion Matrix

print(confusion_matrix(y_test, pred))

[[204 19]
[38 39]]

The actual value of claim is not fraud, and the model also predicte d as not fraud

True positive 204

Model predicted as fraud, but the actual value is fraud

False positive 19

The actual value of claim is fraud, and the model also predicted a s not fraud.

False positive 38

The actual value of claim is not fraud, and the model also predicted as fraud.

False negative 39

5.CONCLUSSION

This project help us to classify insurance claim as fraud or not fraud using machine learning algorithms such as decision tree and gradient boosting algorithm.

Perfomance of Algorithms

	Algorithm	Accuracy	Precision	Recall	F1-score
0 (not fraud)	Decision Tree		0.86	0.82	0.84
1 (fraud)	Decision Tree		0.56	0.63	0.60
	Decision Tree	0.78			
0 (not fraud)	GBM		0.88	0.93	0.9i
1 (fraud)	GBM		0.73	0.59	0.65
	GBM	0.81			

The gradient boosting algorithm gave high accuracy compared to decision tree algorithm.

REFERENCE

- https://campus.datacamp.com/courses/machine-learning-with-tree-based-models-in-python/boosting?ex=6
- https://www.datacamp.com/community/tutorials/decision-tree-classification-python
- https://www.researchgate.net/publication/337508754