## INSURANCE CLAIM FRAUD DETECTION

**EVALUATION PROJECT 9:-** INSURANCE CLAIM FRAUD DETECTION

**NAME –** SUBHAM  
**INTERNSHIP NO. –** 66

**BATCH –** DS2401

**1) PROBLEM DEFINITION:-**

Insurance fraud is a huge problem in the industry. It’s difficult to identify fraud claims. Some people try to claim insurance benefits which they are not entitled to . The dataset is provided which provides details of insurance policy and customer details. So this machine learning project is about detecting insurance claim fraud detection(whether an insurance claim is fraudulent or not).

The dataset used for this project consists of 1000 rows and 39 columns, containing various attributes related to insurance polices , incidents and claims. here are some key details about the dataset.

* **Attributes :-** The dataset includes attributes such as ‘month\_as\_customer’ , ‘age’ , ‘policy\_number’ , ‘policy\_bind\_date’, ‘policy\_state’, 'policy\_csl' , 'policy\_deductable', ‘policy\_ annual\_premium’ , 'umbrella\_limit', 'insured\_zip', ‘insured sex’ , ‘insured educational level’, 'insured\_occupation', 'insured\_hobbies', 'insured\_relationship', 'capital\_gains', 'capital\_loss' , ‘incident date’ , ‘incident type’ , ‘collision type’ , incident severity’, 'authorities\_contacted', 'incident\_state' , ,'incident\_city', 'incident\_location’, 'incident\_hour\_of\_the\_day’, 'number\_of\_vehicles\_involved', 'property\_damage' , 'bodily\_injuries',

'witnesses' , 'police\_report\_available' , 'total\_claim\_amount' , 'injury\_claim', 'property\_claim' , 'vehicle\_claim' , 'auto\_make' , 'auto\_model' , 'auto\_year'

* **Label / Target :-**'fraud\_reported'
* **Types of data:-**there are 2 types of data 1) categorical data and 2) numerical data

**2) DATA ANALYSIS :-**

There are 7 main things are done to the data/EDA process

**1) Handling missing values :-** there are 91 NaN values present in ‘authorities contacted’ column.

Handling method:- fill NaN values with mode

Since ‘authorities contacted’ column is of object/string datatype so it’s NaN values are filled using mode for the column(which is police).

(we can drop NaN values as well but since data is less (only 1000 rows so it’s better to avoid loss of data)

**2)Handling skewed data :-**

**Finding methods for skewed data:-**

a) df.describe()(using this method by checking the mean and median values )

b) seabprn’s distplot(plotting the graph to check the skewness in each column)

c) df.skew()(this method provides numerical values for skewness and acceptable skewness range is around +0.5 to -0.5)

Fixing skewed data:- The method used to fix skewness is cuberoot method.

Categorical data is removed as it’s data is in categories so skewness calculation and fixing the data jumbles up the data.

These are the columns having skewness :- 'umbrella\_limit'(1.80) ,

'insured\_zip' (0.81), 'vehicle\_claim'(-0.62)

Vehicle claim skewness is increasing as cuberoot method is only used for

handing right skewed data since it’s less than the other we have left the

skewness as it is.

After removing the skewness is reduced to ‘umbrella limit’(1.47) ,

‘insured zip’ (0.78)

**3)Removing outliers:-**

**Methods for outliers detection:-**

a) df.describe()(using this method by checking any jump in the values of 75% and 100%

b) seaborn’s boxplot (graphical plot where outliers can be easily detected as dots

outside the box’s lines represent the outliers.)

c) z score (z score less than 3 contains 99.7% data , any value having z score

greater than 3 represents outliers ) and numpy’s where fuction(np.where(z>3)

outliers are found to be in columns :- 'age' , 'policy\_annual\_premium' ,

'umbrella\_limit' , 'total\_claim\_amount' , 'property\_claim' ,’ fraud\_detection’

Z score of outliers have rows:- (array([229, 248, 500, 763, 807],

And columns :- array([ 7, 7, 33, 7, 16], dtype=int64))

**Method for Removing outliers :-**

Themethod used to remove outiers are z score and pandas drop function

And 5 rows are removed as they are outliers

Now the dataset have 995 rows and 39 columns

And the data lost is 0.5% . the max data that can be lost is around 10% as the data is important for model building we cannot lose too much of the data as model cannot figure out the hidden patterns in the dataset if the data is very low and more the data better the model will perform

**4)finding correlation and multicollinearity in the data :-**

Method used to find correlation:- The method used for this are pandas corr

Function(df.corr()) and also represented in the heatmap as colour coded

( cmap= ‘spectral)

The columns having high to low correlation shows blue > green > yellow > red

Gradient having blue as highest correlation and red being the lowest

correlation

some important correlations are:-

there is some correlation b/w columns which can cause multicollinearity like

'collision\_type' and 'vehicle\_claim'(0.60) ,

'collision\_type' and 'property\_claim'(0.46) ,

'collision\_type' and 'injury\_claim' (0.44),

'collision\_type' and 'total\_claim\_amount'(0.59) ,

'total\_claim\_amount'(tca) and 'vehicle\_claim'(0.98) ,

'tca' and 'injury\_claim'(0.80)

‘injury claim’ and ‘vehicle claim’(0.72)

‘property claim’ and vehicle claim’(0.73)

‘total claim amount’ and ‘property claim’(0.81)etc

'fraud\_reported' which is label/target have also low correlation to other

columns except for incident\_sensitivity(-0.41)

Finding multicollinearity in columns:- the method used to find the

Multicollinearity is variance inflation factor importing from

statsmodel.stats.outlier\_influence

Removing multicollinearity:- the higher the vif factor more the multicollinearity is present.

Most of the columns have very low vif factor around 1 except for ‘months as customer’(6.83) , ‘age’(6.85) , ‘incident type’(5.11),’no. of vehicles involved (5.12)

And total claim amount , injury claim , property claim , vehicle claim having inf vif factor

I used 2 cases where in one I didn’t remove property claim column and other I removed the property claim column and in the

2nd case :- After removal of property claim column vif values are reduced for total claim

amount( 68.29) , injury claim(4.79) , vehicle claim(50.53). I didn’t remove any other column as too much data will be lost

**5) Equaling the data:-**

Fraud reported column have unequal distribution of data

Method for equalling the data:- SMOTE fit\_resample fuction

imported from Imblearn oversampling

before equalling the data the values are yes -247 and no -753

after equalling the data the values are yes – 749 and no - 749

**6) converting string datatype values to numerical values:-**

Method used for this :- ordinal encoder fit\_transform function imported from sklearn.preprocessing

This is mainly done as after z score’s outlier removal numerical and categorical data have different number of columns and so is not accepted by model and shows error that number of columns are not same.

And problems also occurs in df.describe , boxplot , distplot etc.

So rather than that it’s better to change the object/string datatype columns

into numerical columns and don’t remove skewness and other things from it.

**7) Scaling the data :-**

Scaling is done because the whole data has different columns and all columns have different value ranges so it’s better to scale the whole data while

retaining the data’s value’s distance within the same column.

So only the value is changed to between -1 to 1 and it’s inner pattern’s and

Data itself is not changed and it will be easier for model to find meaningful

patterns as whole data’s values are b/w -1 to +1.

Method used:- It is done by Standard Scaler fit\_transform method. Imported

From sklearn.preprocessing

**3) EXPLORATORY DATA ANALYSIS(EDA) CONCLUDING REMARKS:-**

* As columns no. is high we get a lot of features to go about to build a

good model But rows are only 1000 which is less information on each

columns.

* Missing data were only in one column and can be easily handled
* After removing categorical data into consideration for skewness only a

few columns are left having high skewness

* Only five outliers are present in the dataset making it easier to handle

It.

* Multicollineary was a problem for some columns like total claim amount , injury claim ,property claim and vehicle claim having inf vif values

Which I solved by taking 2 cases 1) by not removing property claim

Column and 2) removing property claim column

* By using smote data was equalized for both categories (yes and no)
* Encoding data and scaling the data was important steps for building a

Better model.

**4) PRE-PROCESSING PIPELINE:-**

The pipeline or steps used for preprocessing is as follows:-

1) IDENTIFYING AND REMOVAL OF MISSING DATA(BY FILLNA FUNCTION)

2) CONVERTING OF STRING / OBJECT DATATYPE CATEGORICAL DATA TO

NUMERICAL DATA( BY ORDINAL ENCODER)

3) IDENTIFYING AND REMOVAL OF SKEWNESS (BY CUBEROOT METHOD)

4) IDENTIFYING AND REMOVAL OF OUTLIERS (BY Z SCORE)

5) SCALING THE DATA(BY STANDARD SCALER)

6) FINDING CORRELATION AND IDENTIFYING AND REMOVING

MULTICOLLINEARITY (BY VIF METHOD)

7) EQUALING THE DATA(BY SMOTE)

**5) BUILDING MACHINE LEARNING MODEL :-**

The data is divided into features/attributes(x)( by drop function of pandas and removing target column) and label/target(y)(by only taking target column in

Dataframe)

Then fitted into the models with training(70%) and testing (30%) using

train\_test\_split from sklearn.model\_selection having best

random state 80 for (data having property claim)(1st case) and random state 54 for (data without property claim)(2nd case)

metrics used are:-

**1) accuracy score :-** gives accuracy of model with comparing predicted test

Values with real test values( imported from sklearn.metrics)

**2) confusion matrix:-** creates a matrix having

a) true positive – predicted positive is right

b) false positive – predicted positive is wrong , real value is negative

c) true negative – predicted negative is right

d) false negative – predicted negative is wrong , real value is positive

( imported from sklearn.metrics)

**3) classification report :-** gives a report of different metrics like f1 score ,

Precision , recall etc.(imported from sklaern.metrics)

**4) cvs score:-** cross validation score , divides data into parts/subsets and taking

Different parts/subsets different combination of train and test and creates cvs score(imported from sklearn.metrics)

It also tells whether accuracy score calculated is correct or not

That’s why difference b/w accuracy score and cvs score is also of great

Importance.

Since it target/label (y) is having categories so this is a classification problem.

There are 7 models build which are :-

**1) Random Forest classifier :-** an ensemble method that combines multiple

Decision trees to improve accuracy and reduce overfitting(imported from

Sklearn.ensemble

**2) Logistic Regression :-** a simple yet effective model for binary classification

(imported from sklearn.linear\_model)

**3) support vector Classifier:-** a model that finds the optimal hyperplane for

seperating classes(imported from sklearn.svm)

**4) Gradient Boosting classifier: -** an ensemble technique that builds model

sequentially , each correcting the errors of its predecessor(import from

sklearn.ensemble)

**5) Ada Boost Classifier:-** This ensemble technique combines multiple weak

Learners(usually decision trees ) to create a strong classifier. It sequentially

Adjusts the weights of incorrectly classified instances, focuses more on difficult cases(import from sklearn.ensemble)

**6) Bagging Classifier:-** Also known as Bootstrap Aggregating , this method

Involves training multiple instances of a base model on different subsets

Of the dataset(created through bootstrapping) and then averaging their

Predictions . bagging helps reduce variance and prevent overfitting, making it

useful for models prone to high variance like decision trees (import from

sklearn.ensemble)

**7) Extra Trees Classifier**:- an ensemble learning method similar to random

Forest but differs in how it selects splits in the decision trees. Instead of

looking For the best split, extra trees splits nodes randomly , which can reduce

variance and improve model performance. This method is computationally

efficient and can provides robust predictions (import from sklearn.ensemble)

**(for case 1) (for case 2)**

**Accuracy score | cvs score | accuracy score |cvs score**

**1) random forest classifier:-** 91.33 87.45 91.55 87.85

**2) Logistic Regression**:- 76.88 74.63 78.88 73.03

**3) Support Vector Classifier**:- 88.22 86.38 88.66 86.98

**4) Gradient Boosting Classifier** 92.22 87.72 92.22 87.65

**5) Ada Boost Classifier**:- 88.88 83.99 88.66 84.25

**6) Bagging Classifier**:- 90.00 87.39 88.22 86.98

**7) Extra Trees Classifier**:- 93.11 91.26 92.44 91.05

Best model in both the cases comes out to be extra trees classifier in both the

cases and we can see that both the data is having little difference from each

other ( both cases have almost similar accuracy score and cvs score)

meaning there is little effect of property claim column removal.

Other best models after extra trees classifier in order of decreasing are

Gradient boosting classifier , random forest classifier and bagging classifier

2nd best model is gradient boosting classifier(gbc) as it's accuracy is 2nd

highest and cvs score is also 3rd highest and difference is also low. similarly 3rd best model is random forest classifier(rfc) and 4th best model is bagging

classfier(bc).

**SAVING THE MODEL:-** model is saved using joblib dump method

And then testing the model on x\_test by loading the model by joblib.load

Method result:- most of the predicted values matches the y\_test because

accuracy score and cvs scores are high and meaning that a good model is made

**6) CONCLUDING REMARKS :-**

* Model is build good , there are less errors in the dataset and less

Overfitting in the models and overfitting is also reduced by models

Like extra trees classifier , bagging classifier etc.

* Best model comes out to be extra trees classifier.
* Logistic regression is used in classification problems and is better

Suited for binary classification like this project

* For testing we have used x\_test and predicted fraud reported and we

can check also by calling y\_test(which is the real list of fraud reported)

and check y\_test and x\_test prediction are same or not. In this

case most of the predicted values are similar to y\_test thus tells us that the model is build good. And this is because model has high accuracy

and high cvs score.

* The project successfully demonstrates the use of machine learning

Techniques to detect fraudulent insurance claims. By leveraging data

Analysis , preprocessing , and model evaluation , we developed a robust

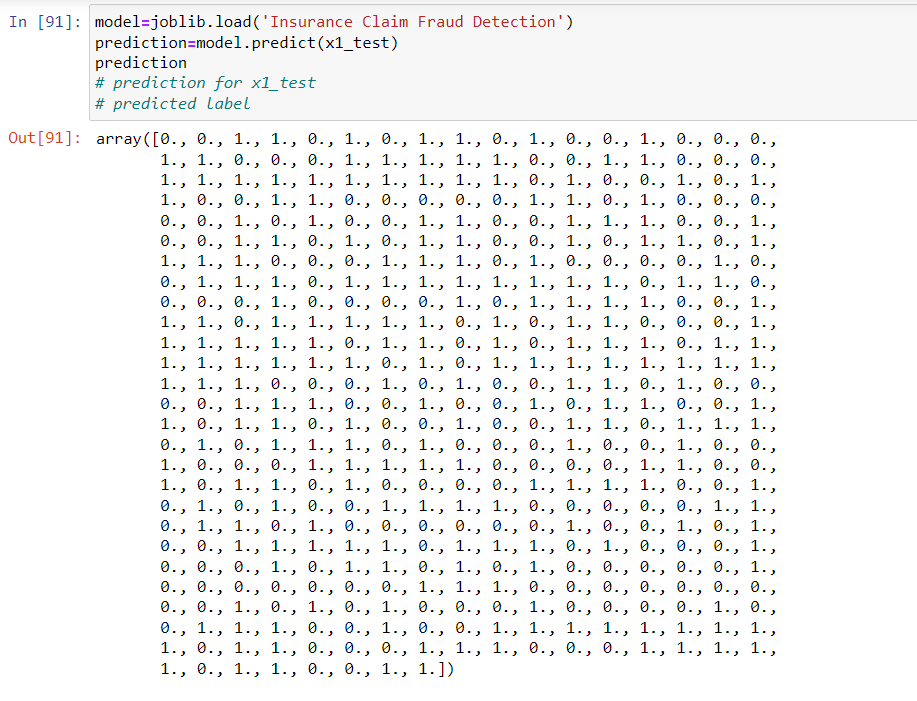
Predictive model that can help insurance companies identify potential

Frauds early , reducing financial losses and improving operational

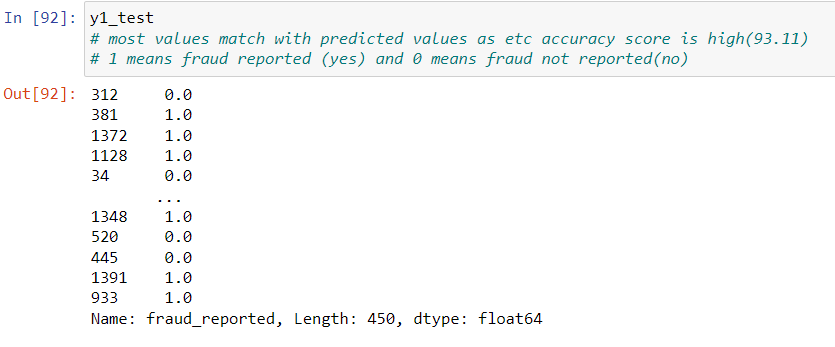
Efficiency.

* The picture shown below represents predictions and y\_test for case 1

**Predictions:-**

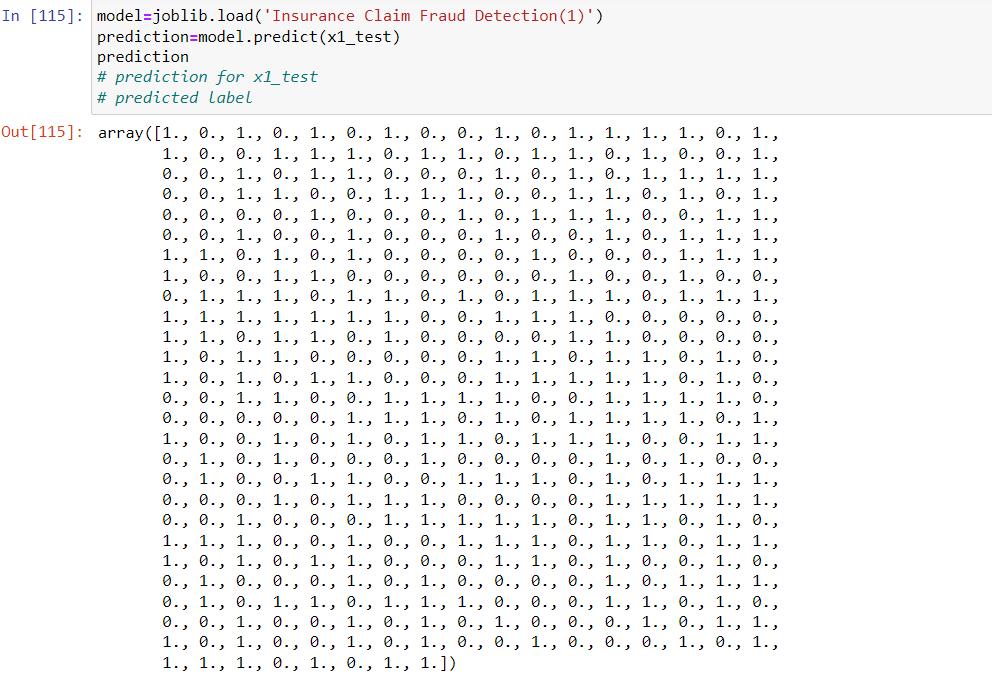


**y\_test:-**

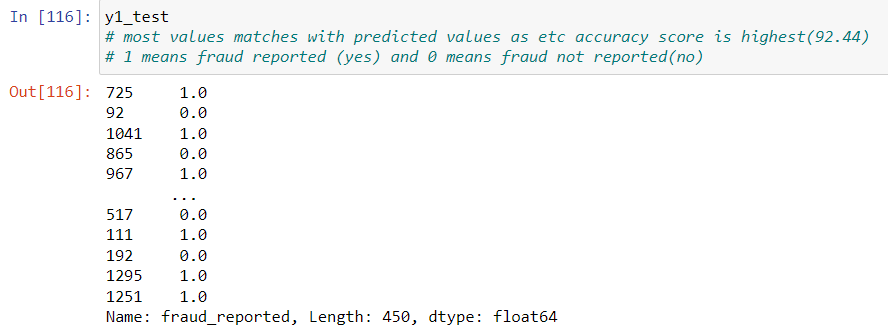


* The picture below represents prediction and y\_test for case 2

**Prediction:-**



**y\_test:-**



1 means fraud reported(yes) and 0 means fraud not reported(no)