Table of context

1. Problem Definition

2. Data Analysis/EDA Concluding Remarks

3. Pre-processing Pipeline

4. Building Machine Learning Models

5. Concluding Remarks

1. Problem Definition

Business case:

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

**This is a logistic regression problem**

2. Data Analysis/EDA

**Imported the libraries used:**

**import numpy as np**

**import pandas as pd**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**from scipy.stats import zscore**

**from sklearn.naive\_bayes import GaussianNB**

**from sklearn.svm import SVC**

**from sklearn.ensemble import RandomForestClassifier**

**from sklearn.linear\_model import LogisticRegression**

**from sklearn.tree import DecisionTreeClassifier**

**from sklearn.ensemble import AdaBoostClassifier**

**from sklearn.metrics import accuracy\_score,confusion\_matrix,classification\_report**

**from sklearn.preprocessing import StandardScaler**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.model\_selection import cross\_val\_score**

**from sklearn.preprocessing import LabelEncoder**

**import joblib**

**import warnings**

**warnings.filterwarnings("ignore")**

There are total of 1000 rows and 40 columns

Dropped the umbrella\_limit column as it has 798 values as "0.00"

Dropped \_c39 column as it has only nan values.

Dropped incident\_date','incident\_location' and auto\_model as this are the unneccessary details which does not effect the dataset.

There were invalid values like "?" is present in the dataset so replaced the value with nan and found that there are

178 nan values in the collision\_type which are replaced with mode

360 nan values in the property\_damage which are replaced with mode

343 nan values in the police\_report\_available which are replaced with mode

Categorical values are being encoded with label encoder

**df1.info()**

**<claspolice\_report\_availableframe.DataFrame'>**

**RangeIndex: 1000 entries, 0 to 999**

**Data columns (total 38 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 insured\_zip 1000 non-null int64**

**9 insured\_sex 1000 non-null object**

**10 insured\_education\_level 1000 non-null object**

**11 insured\_occupation 1000 non-null object**

**12 insured\_hobbies 1000 non-null object**

**13 insured\_relationship 1000 non-null object**

**14 capital-gains 1000 non-null float64**

**15 capital-loss 1000 non-null int64**

**16 incident\_date 1000 non-null object**

**17 incident\_type 1000 non-null object**

**18 collision\_type 1000 non-null object**

**19 incident\_severity 1000 non-null object**

**20 authorities\_contacted 1000 non-null object**

**21 incident\_state 1000 non-null object**

**22 incident\_city 1000 non-null object**

**23 incident\_location 1000 non-null object**

**24 incident\_hour\_of\_the\_day 1000 non-null int64**

**25 number\_of\_vehicles\_involved 1000 non-null int64**

**26 property\_damage 1000 non-null object**

**27 bodily\_injuries 1000 non-null int64**

**28 witnesses 1000 non-null int64**

**29 police\_report\_available 1000 non-null object**

**30 total\_claim\_amount 1000 non-null int64**

**31 injury\_claim 1000 non-null int64**

**32 property\_claim 1000 non-null int64**

**33 vehicle\_claim 1000 non-null int64**

**34 auto\_make 1000 non-null object**

**35 auto\_model 1000 non-null object**

**36 auto\_year 1000 non-null int64**

**37 fraud\_reported 1000 non-null object**

**dtypes: float64(2), int64(15), object(21)**

**memory usage: 297.0+ KB**

There was value as "0.00" present in XLarge Bags column and Large Bags column with huge amount which has to be replaced either by mean or median. Basically both the columns are to be replaced by median as the graph is not normally distributed. But the value XLarge Bags column was replaced by mean because the XLarge Bags column median value as coming zero.

3. Pre-processing Pipeline

**Skeweness Removal**

There is skewness present in 'Total Volume', '4046', '4225', '4770', 'Total Bags','Small Bags', 'Large Bags', 'XLarge Bags', 'year'.

**df1.skew()**

**months\_as\_customer 0.362177**

**age 0.478988**

**policy\_number 0.038991**

**policy\_state -0.026177**

**policy\_csl 0.088928**

**policy\_deductable 0.477887**

**policy\_annual\_premium 0.004402**

**insured\_zip 0.816554**

**insured\_sex 0.148630**

**insured\_education\_level -0.000148**

**insured\_occupation -0.058881**

**insured\_hobbies -0.061563**

**insured\_relationship 0.077488**

**capital-gains 1.024783**

**capital-loss -0.898987**

**incident\_type 0.101507**

**collision\_type -0.033682**

**incident\_severity 0.279016**

**authorities\_contacted -0.121744**

**incident\_state -0.148865**

**incident\_city 0.049531**

**incident\_hour\_of\_the\_day -0.035584**

**number\_of\_vehicles\_involved 0.502664**

**property\_damage 0.863806**

**bodily\_injuries 0.014777**

**witnesses 0.019636**

**police\_report\_available 0.802728**

**total\_claim\_amount -0.594582**

**injury\_claim 0.264811**

**property\_claim 0.378169**

**vehicle\_claim -0.621098**

**auto\_make -0.018797**

**auto\_year -0.048289**

**fraud\_reported 1.175051**

**months -0.029321**

**days 0.024372**

**years 0.052511**

**dtype: float64**

After removing the skewness of column "insured\_zip", "capital-gains", "capital-loss" through log transformation.

**df1.skew()**

**months\_as\_customer 0.362177**

**age 0.478988**

**policy\_number 0.038991**

**policy\_state -0.026177**

**policy\_csl 0.088928**

**policy\_deductable 0.477887**

**policy\_annual\_premium 0.004402**

**insured\_zip 0.782405**

**insured\_sex 0.148630**

**insured\_education\_level -0.000148**

**insured\_occupation -0.058881**

**insured\_hobbies -0.061563**

**insured\_relationship 0.077488**

**capital-gains -0.168017**

**capital-loss -0.537788**

**incident\_type 0.101507**

**collision\_type -0.033682**

**incident\_severity 0.279016**

**authorities\_contacted -0.121744**

**incident\_state -0.148865**

**incident\_city 0.049531**

**incident\_hour\_of\_the\_day -0.035584**

**number\_of\_vehicles\_involved 0.502664**

**property\_damage 0.863806**

**bodily\_injuries 0.014777**

**witnesses 0.019636**

**police\_report\_available 0.802728**

**total\_claim\_amount -0.594582**

**injury\_claim 0.264811**

**property\_claim 0.378169**

**vehicle\_claim -0.621098**

**auto\_make -0.018797**

**auto\_year -0.048289**

**fraud\_reported 1.175051**

**months -0.029321**

**days 0.024372**

**years 0.052511**

**dtype: float64**

**Outliers Removal**

First found the outliers present in which columns through boxplot. Then found that outliers present in policy\_annual\_premium","capital-gains","property\_claim".

Tried to remove outliers through zscore and IQR:

**--> .5%ofdata loss is there while removing outliers with zscore**

**--> 1.6%data loss is there while removing outliers with IQR**

**So we will go with less data loss i.e. ZSCORE method**

Data was scalled through StandardScaler

Applied PCA as there are 37 columns in the data set and after spplying there are only 27 columns and 995 rows left.

**4.      Building Machine Learning Models**

The training and testing data has been split:

**x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=.30,random\_state=133)**

**print(x\_train.shape)**

**print(x\_test.shape)**

**print(y\_train.shape)**

**print(y\_test.shape)**

**(696, 27)**

**(299, 27)**

**(696,)**

**(299,)**

4 models are tried to find the best accuracy. Logistic Regression, Gaussian NB, support vector classifier, random forest classifier, ada boost classifier.

**Found the best model Random forest classifier below details:**

**rf=RandomForestClassifier(n\_estimators=100)**

**rf.fit(x\_train,y\_train)**

**pred3=rf.predict(x\_test)**

**print("accuracy score:",)**

**print(accuracy\_score(y\_test,pred3))**

**print(confusion\_matrix(y\_test,pred3))**

**print(classification\_report(y\_test,pred3))**

**accuracy score:**

**0.7859531772575251**

**[[231 3]**

**[ 61 4]]**

**precision recall f1-score support**

**0 0.79 0.99 0.88 234**

**1 0.57 0.06 0.11 65**

**accuracy 0.79 299**

**macro avg 0.68 0.52 0.49 299**

**weighted avg 0.74 0.79 0.71 299**

**Cross Validation score for random forest classifier as:**

**score=cross\_val\_score(rf,x,y,cv=5)**

**print(score)**

**print(score.mean())**

**[0.7638191 0.73869347 0.75879397 0.75879397 0.75879397]**

**0.7557788944723617**

The final object was made as Insurance\_Fraud.obj

5. Concluding Remarks

Random Forest is a supervised learning algorithm. Like you can already see from it’s name, it creates a forest and makes it somehow random. The forest it builds, is an ensemble of Decision Trees, most of the time trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

One big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. With a few exceptions a random-forest classifier has all the hyperparameters of a decision-tree classifier and also all the hyperparameters of a bagging classifier, to control the ensemble itself.

The random-forest algorithm brings extra randomness into the model, when it is growing the trees. Instead of searching for the best feature while splitting a node, it searches for the best feature among a random subset of features. This process creates a wide diversity, which generally results in a better model. Therefore when you are growing a tree in random forest, only a random subset of the features is considered for splitting a node. You can even make trees more random, by using random thresholds on top of it, for each feature rather than searching for the best possible thresholds (like a normal decision tree does).

NY is the state where highest number of fraud reported

Multi vehicle collision is the highest incident type fraud reported

Most of the insured hobbies are reading

The best model is Random Forest Classifier