INSURANCE CLAIM FRAUD DETECTION WORKSHEET

INTRODUCTION TO THE DATASET

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.

This dataset provides 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.

Here in this worksheet(Insurance\_claim\_fraud\_detection.ipynb) I have worked with some auto insurance data and created a predictive model that predicts if an insurance claim is fraudulent or not.

The dataset contains a total of 1000 rows and 40 columns out of which 39 are predictor features while one(**fraud\_reported) is a predicted feature.**

**The predicted fearure is of categorical type with ‘Y’(yes) and’N’(no) as the unique values i.e the fact that if a fraud was reported or not for each specific row is explained by a ‘YES’ or a ‘NO’.**

DATA ANALYSIS

On analysing the dataset it was found that:

1)the dataset has 21 features with object datatype,17 with integer datatype and one with float datatype.

2)the feature named ‘\_c39’ had all its values missing.It was discarded from the dataset to be used for further analysis as it would have not contributed anything to our goal of predicting the target variable.

3)there were no missing values in any column other than the ‘\_c39’feature that we discarded.

The features with object datatype were separated from the ones with integer and float datatype with the names object\_col and cont\_col respectively for a more organised and specific analysis.

EXPLORATORY DATA ANALYSIS

The columns with object datatype were analysed first using a countplot with the hue as the target variable(fraud\_reported).

The detected patterns and findings of the EDA on the object columns are:

1)The policy\_state feature(OH,IN,N) had the fraud reporting in almost the same propotion i.e the Policy\_state did not seem to have any noticiable insight regarding the fraud claims.

2)The policy\_csl feature too had all its three aspects in somewhat similar propotions ,hence it too did not provide any significant insight.

3)The female customers had a comparatively lesser propotion of fraud\_claims than the male customers.The difference though was not very huge.

4)The propotion of fraud claims was higher in the customers with an education level equivalent to JD.(junior division) with almost one fraud report in every 4(JD) customers.(source feature-‘insured\_education\_level’).

5) The customers in the exec-managerial occupation had the most cases and the highest propotion of fraud reports with almost 7 fraud claims in every 19 from the same occupation.

6) The customers in the priv-house-serv and other-services had the least cases aswell as the lowest propotion of fraud reports.

7) The customers having chess as a hobby had a significantly high rate of fraud reports with almost 19 fraud claims out of every 23 chess players.

8) Customers having cross-fit as a hobby too had a considerably high rate of fraud reports succeeding the chess players.(4:1) 4 fraud claims in every 5.

9) Insured with relationship "own-child" and " husband" had lesser rate of fraud cases although no significant difference was observed.

10)The incident\_type ‘single vehicle collision’ had the highest propotion of fraud cases 3 fraud claims in every 10 of its kind.It is succeeded by the incident\_type ‘multi-vehicle collision’.

11)The collision\_type ‘rear-collision had the highest propotion of fraud cases.

12) The incidents claiming Major Damage had a significantly high rate of fraud cases(11 fraud cases in every 18 approx.)  whereas The incidents claiming minor damage had the lowest rate of detected fraud cases reported.

13)The cases in which the police were contacted had the lowest propotion of fraud reports as compared to the other cases in which some other or no authorities were contacted.

14)Out of the different states in which the incidents occurred OH had the highest propotion of detected fraud cases(though the total number of cases from OH is significantly lower as compared to the others.

15) Out of the different states in which the incidents occurred SC had the most number of reported fraud cases.

16)The customers claiming no property damage had the lowest propotion of fraud reports.

17) The insured with an auto-make of 'Mercedes' and 'Audi' had a higher propotion of fraud cases whereas the Insured 'Nissan' and 'jeep' had the lowest rate of fraud cases.

18) The insured with models 'ML\_350','Silverado','X6','Tahoe' had a relatively high propotion of detected fraud cases followed by ‘C300’ and ‘F150’.The insured with model RAM had the highest number of fraud claims.

19) The insured with models 'Wrangler','Neon','Malibu' had a relatively low propotion of reported fraud cases.

20) The insured owning :2004 ,1996,2007 AND 2011 made automobiles had the highest propotion of fraud cases..

The features with integer and float datatype(cont\_col) were analysed The detected patterns and findings of the EDA on the contionous columns are:

1)The insured of age 31,33,34,41 had a higher propotion of fraud claims than the others.(Considering only the age categories of which atleast 20 cases are present in the dataset).

2)customers with policy deductables of 1000 had a relatively lower propotion of fraud reports.

3)Most of the insured had an umbrella\_limit of 0.

4) The insured who reported the hour of the incident occurance as 2,10,11,14,15 and 18 had a a relatively higher propotion of fraud reports .

5) ) The insured who reported the hour of the incident occurance as 3,8,9,17,22 had a relatively lower propotion of fraud cases.

6)Most reported incidents had 1 or 3 vehicles involved.

7)The incidents involving 1 vehicle had the highest number of fraud claims.

8)The incidents reporting bodily injuries as 2 had a slightly higher propotion of fraud cases.

9)The incidents involving 2 witnesses had a higher propotion of fraud reports whereas the incidents involving 0 witnesses had the lowest propotion of fraud reports.

The continous type columns were then checked for outliers using a boxplot but no significant outliers were detected except for the umbrella column which basically had a value of 0 for 90% of its rows.

The continous datatypes were further analysed with the help of a heatmap and it was obsereved that The features -total\_claim\_amount,injury\_claim,property\_claim,vehicle\_claim show considerably high correlation amongst themselves i.e total\_claim\_amount feature is basically the sum of -injury\_claim,property\_claim,vehicle\_claim . Hence it was decided that there is no need to provide the similar information multiple times as it will only increase the size of the dataset and thus, hinder the model learning process.(CURSE OF DIMENSIONALITY).

PREPROCESSING

The first thing to be done is removing the columns(injury\_claim,property\_claim,vehicle\_claim) as they are already being represented by the total\_claim\_amount column.

For this,a list was created by the name claim\_list which consisted of all the three columns to be dropped and those columns were removed from the dataframe by iterating through the list using a for loop.

LABEL ENCODING:

There was the ordinal –‘insured\_education\_level’ feature for which OneHotEncoding was done by creating dummies and adding(concatenating) them to the original dataframe and later the original- ‘insured\_education\_level’ feature was dropped as it had no more use in further operations.

The rest of the data appeared to be of nominal type.Hence,LabelEncoding was done on them using the LabelEncoder instance from the sklearn.preprocessing

The ‘policy\_bind\_date’ feature was removed from the dataset as it had unique values for almost every row and was unlikely to provide with a pattern or any insight.Also,converting it into datetime format would only have added to the problem(curse of dimensionality) rather than helping solve it.Moreover,fitting the models with datetime dtypes would also have been complicated.

Further,

The target(fraud\_reported) feature was separated from rest of the features for proceeding towards the model\_selection phase.

The target variable(y) was then analysed for its value\_counts and it was observed that the data was imbalanced i.e there were 753 instances of one class and only 247 of the other(y).This would have lead to poor model performance as the model would not have been trained for the (Y) class of the target variable leading to poor model performance(the model would’nt have properly identified the (Y) fraud\_reports).Hence,balancing the data was required.

For balancing the dataset ‘pip install -U imbalanced-learn’ command was executed and the imbalanced-learn library was installed from the google.

As i proceeded to over sampling the data, SMOTE was imported from ‘imblearn.over\_sampling’.

The fit\_resample method was used and the resultant dataset had 753 rows for each Y and N classes.

## MODEL SELECTION

The method train\_test\_split was imported from the sklearn.linear\_model and x and y were passed into it for splitting them into testing and training data according to the size specified(0.33).

CLASSIFICATION METRICS:

The metrics: accuracy\_score,confusion\_matrix,classification\_report,roc\_auc\_curve were imported from sklearn.metrics for evaluation of the model’s performance on training and testing data simultaneously.

BUILING MACHINE LEARNING MODELS

LOGISTIC REGRESSION

LogisticRegression was imported from sklearn.linear\_model class and fit and predict method were used on the training and testing data using its instance(lr).

RESULTS:

The accuracy\_score of the model on the test data was : 0.5653923541247485

The confusion\_matrix figures were: [140 96]

[120 141]

The classification\_report was:

precision recall f1-score support

0 0.54 0.59 0.56 236

1 0.59 0.54 0.57 261

accuracy 0.57 497

macro avg 0.57 0.57 0.57 497

weighted avg 0.57 0.57 0.57 497

REMARKS: THE LOGISTIC REGRESSION MODEL PERFORMED POORLY IN ALL THREE METRICS USED.

DECISIONTREECLASSIFIER

DecisionTreeClassifier was imported from sklearn.tree class an class and fit and predict method were used on the training and testing data using its instance(dtr) and criterion as ‘gini’.

RESULTS:

The accuracy\_score of the model on the test data was : 0.8309859154929577

The confusion\_matrix had figures: [186 50]

[ 34 227]

The classification report was:

precision recall f1-score support

0 0.85 0.79 0.82 236

1 0.82 0.87 0.84 261

accuracy 0.83 497

macro avg 0.83 0.83 0.83 497

weighted avg 0.83 0.83 0.83 497

REMARKS:The DecisionTreeClassifier performed well in all three metrics with a good f1 score of : 0.82 and 0.84 respectively for both N and Y classes.

RANDOM FOREST CLASSIFIER

RandomForestClassifier was imported from the sklearn library under the class ensemble.Fit and predict methods were used on training and testing data using its instance(rfr)

With parameters: n\_estimators=100,criterion=’gini’,random\_state=42

RESULTS:

The accuracy\_score of the model on the test data was : 0.8832997987927566

The confusion\_matrix had figures: [203 33]

[ 25 236]

The classification report was:

precision recall f1-score support

0 0.89 0.86 0.88 236

1 0.88 0.90 0.89 261

accuracy 0.88 497

macro avg 0.88 0.88 0.88 497

weighted avg 0.88 0.88 0.88 497

REMARKS:The RandomForestClassifier performed well in all the three metrics with a f1 score of : 0.88 and 0.89 for both N and Y classes.

KNEIGHBORS CLASSIFIER

The KNeighborsClassifier was imported from sklearn library under the class neighbors and the fit and predict methods were used on its instance(knr) with parameter(n\_neighbors=5).

RESULTS:

The accuracy score was: 0.6861167002012073

The confusion\_matrix looked like: [129 107]

[ 49 212]

The classification\_report was:

precision recall f1-score support

0 0.72 0.55 0.62 236

1 0.66 0.81 0.73 261

accuracy 0.69 497

macro avg 0.69 0.68 0.68 497

weighted avg 0.69 0.69 0.68 497

REMARKS:The KNeighborsClassifier’s performance was average (better than LogisticRegression and worse than RandomForestClassifier aswell as DecisionTreeClassifier) with an f1 score of: 0.62 and 0.73 for N and Y classes.

ADABOOSTCLASSIFIER

The AdaBoostClassifier was imported from the sklearn library under the class ensemble.Fit and predict methods were used on training and testing data using its instance(abc) with parameters: base\_estimator=rfc(RandomForestClassifier),n\_estimators=100.

RESULTS:

The accuracy\_score was: 0.8832997987927566

The confusion\_matrix had figures: [208 28]

[ 30 231]

The classification\_report was:

precision recall f1-score support

0 0.87 0.88 0.88 236

1 0.89 0.89 0.89 261

accuracy 0.88 497

macro avg 0.88 0.88 0.88 497

weighted avg 0.88 0.88 0.88 497

REMARKS: The AdaBoostClassifier performed well on the test data with an f1 score of 0.88 and 0.89

For both N and Y cases respectively.

GRADIENTBOOSTINGCLASSIFIER

**The GradientBoostingClassifier was imported from the sklearn library under the class ensemble.Fit and predict methods were used using its instance(gbc) .**

**RESULTS:**

**The accuracy\_score was :** 0.8792756539235412

**The confusion\_matrix had figures:** [204 32]

[ 28 233]

**The classification\_report was:**

precision recall f1-score support

0 0.88 0.86 0.87 236

1 0.88 0.89 0.89 261

accuracy 0.88 497

macro avg 0.88 0.88 0.88 497

weighted avg 0.88 0.88 0.88 497

REMARKS: GradientBoostingClassifier performed well in all the three metrics and had an f1 score of 0.87 and 0.89 For both N and Y cases respectively.

VOTINGCLASSIFIER

The VotingClassifer **was imported from the sklearn library under the class ensemble.Fit and predict methods were used using its instance(vot\_hard) .The parameters were: estimators: (('decision tree',DecisionTreeClassifier(criterion='gini')),** **(('random frest',RandomForestClassifier(criterion='gini')),** **('Gradient Boosting',GradientBoostingClassifier())), voting=’hard’)**

**RESULTS:**

**The accuracy\_score was:** 0.8933601609657947

**The confusion\_matrix displayed:** [202 34]

[ 19 242]

**The classification\_report was:**

precision recall f1-score support

0 0.91 0.86 0.88 236

1 0.88 0.93 0.90 261

accuracy 0.89 497

macro avg 0.90 0.89 0.89 497

weighted avg 0.89 0.89 0.89 497

REMARKS:The VotingClassifier performed best in all the three metrics with an f1 score of 0.88 and 0.89 For both N and Y cases respectively.

FINAL REMARKS:

THE VOTING CLASSIFIER MODEL PERFORMED BEST ON THE TEST DATA IN ALL THE THREE METRICS but this in itself will not be sufficient to choose it as our best model because RandomForestClassifier,AdaboostClassifier and GradientBoostingClassifier too had very close performance.

CROSS-VALIDATION

The cross\_val score was imported from the sklearn.model\_selection in order to compare the skills of the models on unseen data.

The performance of the models by their mean scores and cross\_val\_scores in a 10-fold cross validation are:

LOGISTIC REGRESSION

mean score: 0.752

cross val score: [0.75 0.75 0.75 0.75 0.74 0.75 0.75 0.76 0.76 0.76]

RANDOM FOREST CLASSIFIER

mean score: 0.776

cross val score: [0.75 0.82 0.77 0.82 0.7 0.76 0.73 0.8 0.83 0.78]

KNEIGBORS CLASSIFIER

mean score: 0.7100000000000001

cross val score: [0.72 0.71 0.68 0.72 0.71 0.71 0.68 0.73 0.7 0.74]

DECISION TREE CLASSIFIER

mean score: 0.798

cross val score: [0.8 0.81 0.75 0.8 0.73 0.8 0.8 0.84 0.79 0.86]

ADABOOST CLASSIFIER

mean score: 0.776

cross val score: [0.76 0.83 0.73 0.74 0.74 0.75 0.77 0.83 0.8 0.81]

VOTING CLASSIFIER

mean score: 0.8220000000000001

cross val score: [0.79 0.82 0.76 0.85 0.78 0.8 0.83 0.86 0.89 0.84]

GRADIENT BOOSTING CLASSIFIER

mean score: 0.8239999999999998

cross val score: [0.82 0.8 0.78 0.83 0.8 0.76 0.86 0.87 0.89 0.83]

FINAL REMARKS:THE GRADIENT BOOSTING CLASSIFIER HAD THE BEST CROSS-VALIDATION SCORE FOLLOWED BY VOTING CLASSIFIER.

**ROC\_AUC\_CURVE**

REMARKS:

The GradientBoostingClassifier was plotted with the RandomForestClassifier and the curve indicated that both of them did a good job in identifying the positive class(N) in the fraud report though,the RandomForestClassifier did a slightly better job.

The gradient boosting classifier and Voting Classifier too had almost the same performaces in the roc\_curve job in identifying the positive class(N) in the fraud report



SINCE,

THE RANDOM FOREST CLASSIFIER AND THE GRADIENT BOOSTING CLASSIFIER BOTH PERFORMED WELL,

HYPERPARAMETER TUNING WAS DONE FOR BOTH OF THEM SIMULTANEOUSLY IN ORDER TO MAKE THE FINAL PICK.

GRID SEARCH

THE RESULT OF THE GRIDSEARCHCV WAS:

RANDOM FOREST CLASSIFER

0.8731737352839761

200

{'criterion': 'entropy', 'n\_estimators': 200, 'random\_state': 42}

GRADIENT BOOSTING CLASSIFIER

0.8811093049603468

300

{'criterion': 'friedman\_mse', 'n\_estimators': 300, 'random\_state': 42}

REAMRKS:

The GradientBoostingClassifier has a better overall performance(considering accuracy\_score,cross\_val\_score,f1\_score and the best\_score derived from the grid search as well as the roc\_auc\_curve).Hence, the GradientBoostingClassifier model was tuned with the best parameters from the GridSearchCV and chosen as the best model.

SAVING THE MODEL

The GradientBoostingClassifier model was saved as ‘insurance\_claim\_fraud\_detection.sav’ using pickle.dump method from pickle class that was imported(SERIALIZATION).

RELOADING AND TESTING THE MODEL

The model was loaded back into the system using load method from the pickle class(DESERIALIZATION) and was tested with the training and testing data simultaneously.

A sample of 100 rows from the training data was tested against the model and the model gave a score of 1.0.

When tested against a 100 rows of test data the model gave an accuracy\_score of 0.91.

CONCLUSION:

The goal of this project was to successfully detect the fradudelent claims and here in this worksheet(insurance\_claim\_fraud\_detection.ipynb) i have worked with some auto insurance data and created a predictive model that predicts if an insurance claim is fraudulent or not.

The dataset had about 40 columns containing both categorical and continous class of features ,so I had to be extra careful while going on with the feature engineering part.

Also,since in today’s world with the insurance industry on the boom there are quite less number of fraudulent claims as compared to the legitimate ones.This also creates the problem of an imbalanced dataset with more cases of the legitimate type and fewer of the fraudulent ones which can hinder the model learning processes.Hence, one has to properly analyse the dataset type and use over\_sampling and under\_sampling accordingly.

The EDA of the data provided some useful insights that would not have been smelled out of the surroundings.For example - The customers having chess as a hobby had a significantly high rate of fraud reports with almost 19 fraud claims out of every 23 chess players’

AND

The customers in the exec-managerial occupation had the most cases and the highest propotion of fraud reports with almost 7 fraud claims in every 19 from the same occupation.

A lot of such insights have already been mentioned in the EDA part of this report.

Finally,

I was able to get a model which can predict and detect almost 90% of the fraululent claims beforehand just by using the details of the insured and the claims itself.

I hope you like my work.

Thank you,

Regards,

Prakhar Prakash