Capstone2_Final Report

Vehicle Insurance Claim Fraud Detection

Introduction

Vehicle Insurance Fraud is a major area of losses to the auto insurance companies. Study conducted by Verisk, indicated that insurance companies had to bear losses to the tune of $29 billion a year.[1] According to Progressive, auto insurance fraud is committed when someone falsely claims about an event and get monetarily compensated.[2] In this project an attempt has been made to explore major contributing factors which have led to fraud along with developing models which can distinguish fraudulent transactions. For this model data is obtained from Vehicle Insurance Claim Fraud Detection[3]

Problem Statement

How can Data Science be leveraged to detect Vehicle Insurance Fraudulent claims to reduce losses on account of exaggerated and false claims about accidents, property damage and physical injury by developing machine learning models which can improve the accuracy as measured by F1 SCORE to above 0.8 in 6 months**.**

Scope

The only available data used for analysis is a single CSV file which is used for developing models and exploratory data analysis. Additionally, the model can be used only if the new data has same feature space as original csv file.

Constraints within solution space

The major constraint is availability of only one csv file along with the fact that data is highly imbalanced consisting of mainly categorical data making it difficult to generalize the model. Further, economic background and previous offence data is not available for fraudulent claimants

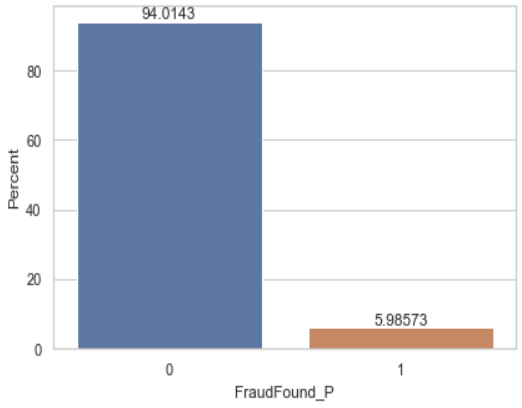
Dataset

Original dataset consists of 15420 rows and 33 features. Out of 33 features, only age in years is discrete variable whereas remaining features are categorical variable.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15420 entries, 0 to 15419
Data columns (total 33 columns):
```

Data Wrangling

During this process, data is checked for missing values, type of data and data discrepancies. The data looked clean without any missing values. The column Policy Number looked randomly distributed and the values indicated it is just a serial number. Target variable ‘FraudFound_P’ looked imbalanced as 94.01% transactions were non-fraudulent. Additionally, age column had 320 values which were 0 and most likely they represented under-age drivers. Other features did not show any major issues.




```
### Proportional distribution of age values
#vf1=vf.loc[vf['FraudFound_P']==0]
vf4=vf.loc[vf['Age']<20]
vf4['Age'].value_counts()
```

```
0     320
18     48
19     32
16      9
17      6
Name: Age, dtype: int64
```

Fig.1 Probablity distribution of target Fraud_Found column **Fig.2 Distribution of age column < 20 years**

Exploratory Data Analysis

In this step all the features were evalauted for impact of different categories on probablity of fraudulant and non-fraudulant transactions. The key observations are discussed below:

- The highest probability of fraudulent transactions is in high end cars like Acura, BMW and Mercedes. and Mercedes, most likely due to higher incentive for frauds as the cars being costlier

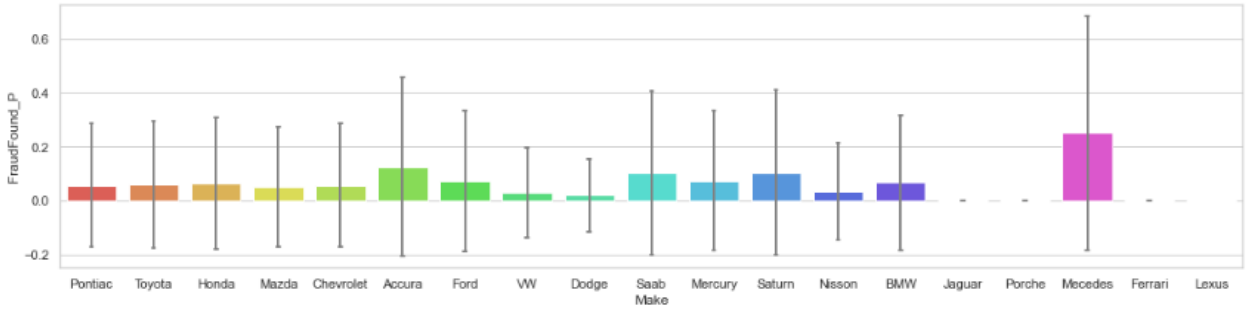


Fig.3 Bar plot of vehicle categories and fraudulent transactions

- Utility vehicles have higher probability of fraudulent transactions compared to other vehicle categories

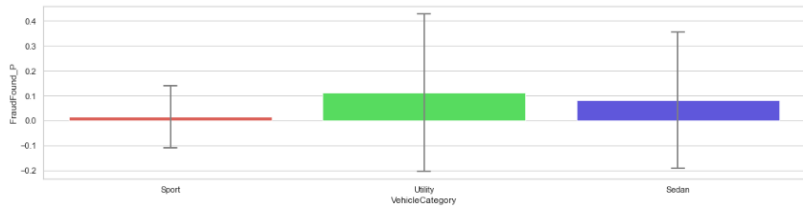


Fig.4 Bar plot of Vehicle Categories and fraudulent transactions

- Address change surprisingly showed significant relation with probability of fraud and on an average 75% of claims when address is changed within last 6 months were fraudulent.

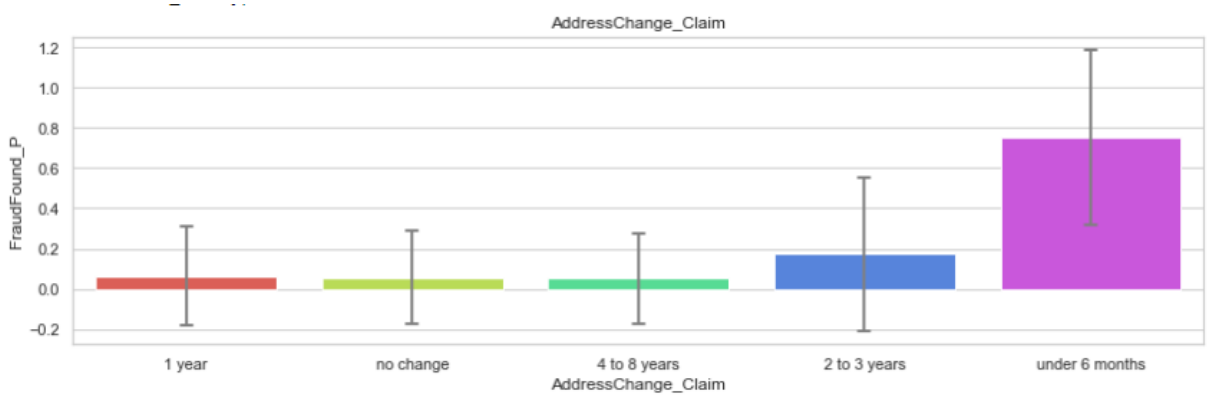


Fig.5 Bar plot of Address Change in claims and fraudulent transactions

Preprocessing and Training

The dataset consisted of only categorical features and data is highly imbalanced. Additionally, since we are interested in predicting fraudulent claims 'accuracy' measurement is meaningless. Even a base model which classifies all transactions as non-fraudulent will have accuracy of 94.1% as 94.1% of transactions are non-fraudulent. Therefore precision, recall and f1 score are chosen as metrics of interest.

Preprocessing

During preprocessing of data, 'age_bins' column is created with 4 bins from 'Age' column columns 'PolicyNumber', 'RepNumber' and 'Age' are dropped from the data set.

Training

Different modeling approaches are tried out using packages like 'PyCaret' to generate base model, utilizing different encoding and balancing techniques:

- In the first step model is developed with Label encoding and SMOTE oversampling using PyCaret library. The best model was 'Light GBM' but results were not good enough.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.9406	0.8225	0.0712	0.5642	0.1252	0.1126	0.1825	0.1010
gbc	Gradient Boosting Classifier	0.9404	0.8197	0.0373	0.6069	0.0688	0.0616	0.1327	0.9260
dummy	Dummy Classifier	0.9402	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0130
rf	Random Forest Classifier	0.9399	0.8093	0.0593	0.4988	0.1045	0.0928	0.1531	0.6490
et	Extra Trees Classifier	0.9361	0.8116	0.0763	0.3418	0.1237	0.1045	0.1375	0.4940
ada	Ada Boost Classifier	0.9058	0.7851	0.1627	0.1833	0.1708	0.1215	0.1224	0.2370

Fig.6 Label Encoding+SMOTE training results on training data set with 10-fold stratified cross validation

- In the second step dummy encoding is performed on categorical features and different balancing methods like SMOTE, ADASYN, Random Over Sampler were tried out. However, the results were not encouraging, and model did not generalize on hold out data using tuned random forest model, although it performed well on training data set. Results of Dummy encoded features with SMOTE balancing method are shown below.

Training Results

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
rf	Random Forest Classifier	0.9665	0.9697	0.9342	0.9987	0.9654	0.9330	0.9349	0.5220
et	Extra Trees Classifier	0.9656	0.9696	0.9333	0.9977	0.9644	0.9311	0.9331	0.3460
gbc	Gradient Boosting Classifier	0.9651	0.9711	0.9311	0.9989	0.9638	0.9301	0.9323	0.4500
ada	Ada Boost Classifier	0.9649	0.9702	0.9309	0.9987	0.9636	0.9297	0.9319	0.1580

Fig.7 Dummy Encoding + SMOTE training results

Validation Results on tuned Random Forest Classifier

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Random Forest Classifier	0.9403	0.4983	0.0000	0.0000	0.0000	0.0000	0.0000

Fig.8 Dummy Encoding + SMOTE validation results

- Considering suboptimal performance so far, weight of evidence encoding with average precision score as criteria are tried with PyCaret library to get base line model. Logistic Regression model is chosen for hyper parameter tuning.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	Average	TT (Sec)
knn	K Neighbors Classifier	0.7737	0.7012	0.4932	0.1308	0.2067	0.1239	0.1615	0.0951	0.6820
qda	Quadratic Discriminant Analysis	0.6778	0.7770	0.7831	0.1319	0.2257	0.1374	0.2252	0.1166	0.0200
nb	Naive Bayes	0.6675	0.7845	0.8220	0.1330	0.2288	0.1403	0.2359	0.1201	0.0160
lr	Logistic Regression	0.6401	0.8013	0.8695	0.1290	0.2247	0.1345	0.2395	0.1202	0.8020

Fig.9 Weight of Evidence Encoding results

Tuned Logistic Regression Model

```
LogisticRegression(C=9.921, class_weight='balanced', dual=False,
                   fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                   max_iter=1000, multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=1122, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

Fig.10 Tuned Logistic Regression Model for final modeling

- Above logistic regression tuned model is used with weight of evidence and cat boost encoder for class weight optimization without adjusting for imbalance. Results were similar for both methods

and f1 score for hold out data set improved to 0.21. Results for weight of evidence encoder with class weight optimization are shown below.

```
{'class_weight': {0: 0.1293467336683417, 1: 0.8706532663316583}}
```

Train

	precision	recall	f1-score	support
0	0.96	0.87	0.92	11598
1	0.20	0.50	0.28	738
accuracy			0.85	12336
macro avg	0.58	0.69	0.60	12336
weighted avg	0.92	0.85	0.88	12336

Test

	precision	recall	f1-score	support
0	0.96	0.86	0.90	2899
1	0.15	0.38	0.21	185
accuracy			0.83	3084
macro avg	0.55	0.62	0.56	3084
weighted avg	0.91	0.83	0.86	3084

Fig.11 Train and test performance for the model with weight of evidence encoding and class weight optimization

Final Model Selection

For final model selection the learnings from the training steps are leveraged and Logistic Regression with optimized parameters and class weights in conjunction with the SMOTEN balancing technique and weight of evidence encoder were used. SMOTEN is chosen as all the features in the model are categorical and is more appropriate than SMOTE for categorical features. Results were much better and F1 score improved to 0.85 for test data with precision score of 0.80 and recall of 0.91. The final model is saved using pipeline with the all the steps as a pickle object for further use.

Train

	precision	recall	f1-score	support
0	0.89	0.78	0.83	11598
1	0.81	0.91	0.85	11597
accuracy			0.84	23195
macro avg	0.85	0.84	0.84	23195
weighted avg	0.85	0.84	0.84	23195

Test

	precision	recall	f1-score	support
0	0.89	0.78	0.83	2899
1	0.80	0.91	0.85	2900
accuracy			0.84	5799
macro avg	0.85	0.84	0.84	5799
weighted avg	0.85	0.84	0.84	5799

Fig.12 **Train and test performance for the final model with SMOTEN, weight of evidence encoding & class weight optimization**

Modeling

The final model can be tweaked based on business requirement. The final model is some what balanced for precision and recall score and depending upon the cost benefit analysis the model can be tweaked by adjusting threshold levels. If the business requires all fraudulent cases are captured even if it leads to investigating non-fraudulent cases, the threshold can be lowered and vice-versa. Precision Recall Curve for the final model can guide the business decision making.

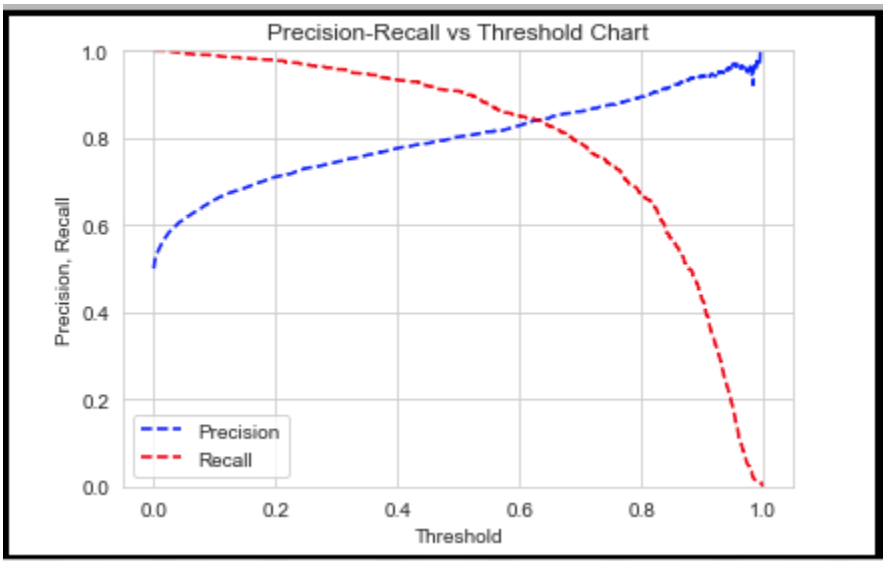


Fig.13 Precision-Recall with Threshold Chart for final model

Conclusion

- SMOTEN balancing combined with weight of evidence encoder, and tuned Logistic Regression model with class weight optimization, gave the best results and F1 score of 0.85 is achieved on minority class on validation data.
- Dummy encoding approach did not generalize well due to too many features
- Logistic regression with optimized class weight without balancing, generalized well but F1 score was only 0.22
- Utility vehicles have higher probability of fraudulent claims in vehicle categories
- Higher probability of fraud if address is changed within last 6 months
- Higher end vehicles like Mercedes, BMW and Acura are more involved in fraudulent claims
- Decision making threshold can be moved based on the business needs.

Further Work

The data set had several limitations as the data set only consisted of categorical variables and was highly imbalanced. Most of the models failed due to that. There are several other techniques to deal with categorical data **https://contrib.scikit-learn.org/category_encoders/ and imbalanced target** variable **https://imbalanced-learn.org/stable/,** which were not tried out and can be tried out to see if model performance can be improved further.

References

1. https://www.iii.org/article/background-on-insurance-fraud
2. https://www.progressive.com/answers/car-insurance-fraud/
3. Vehicle Insurance Claim Fraud Detection