ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

Efficient Fraud Detection Classification: Class Imbalance and Attribute Correlations

Difei Liu[1], Ruiyi Sun[2], Haoyang Ren[3]

1 University of Warwick, Coventry, CV4 7AL, UK
2 Australian National University Canberra, ACT, 0200
3 Beijing Normal University, Zhuhai, China

ABSTRACT. *Fraud detection is a specifically important issue to protect cardholders' information from being stolen by fraudsters. By choosing proper algorithms and analyzing behavioural information of cardholders and banks, we can significantly reduce the probability of transactions being illegally manipulated. In response to possible problems in fraud analysis, this article will focus especially on tackling class imbalance problems and finding attribute correlations. Two FraudDetection datasets on Kaggle will be used to build classifiers and ananlyze the impact of different data processing techniques. Through this process, we realized recent findings of fraud detection, we got to know more about different data processing methods, and we implemented distinct types of classifiers. We confirmed the significance of class imbalance tackling and attribute correlations analyzing.*

KEYWORDS: *Fraud detection, Class imbalance, Attribute correlations, Classification algorithms*

1. Introduction

In modern society, machine learning has penetrated into every aspect of people's lives. Anti-credit card fraud is one area where it is widely used. Fraud detection refers to a series of techniques implemented to protect money or property from being stolen through pretense. Nowadays, as fraud methods of crimers is continuously updating, the requirements for the performance of anti-fraud systems are increasing. Recent fraud detection systems employ distinct kinds of algorithms to represent the performance of fraudulent data to the maximum extent, while also analyse behaviours of each customer and account. However, due to the fact that fraud detection datasets usually have few fraudulent instances and many normal samples, dealing with unbalanced dataset would be a significant issue for fraud detection problems.

In this project, we executed some data analysis work on a competition dataset on Kaggle. We made an experiment on different methods of tackling class imbalance and found the best one to process data, meanwhile we implemented behavioral analysis by studying attributes and trained classifiers with outstanding performance. We referred to two kernels on Kaggle, Credit Fraud Detector [1] and Anomaly Detection[2], which gave us some vivid thinking.

Here we will describe each part of this essay: First, we will introduce the background of fraud detection based on essays and websites published in these two years; then, we will explore into data and clean data; we will provide approaches used focusing especially on class imbalance problem; we will analyze the performance of these distinct methods and work out experimental results on the entire dataset; finally, we will calculate correlations between attributes and label and try to improve our model.

2. Background and Related Works

Fraud detection is a type of desired application of machine learning and artificial intelligence (AI). When customers receive an identification call or text, this may be associated with a series of algorithms. These security measures will maximize the protection of the user's property ownership from fraud and theft.

However, these years the fineness and complexity of fraud have been growing rapidly, urging anti-fraud techniques to update rapidly. In this section, we will discuss several related works in the recent two years detailedly.

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

2.1 Fraud Detection: Adaptive Analysis and Self-Learning Ai

These websites originated from FICO [3] talked about several key factors for fraud detection systems, which can be summarized as:

To deal with flexible and well-organized crime schemes, both supervised and unsupervised models are needed to achieve better performance. For supervised learning model, all transactions are tagged properly, the amount of clean and relevant training data is closely related to model accuracy, which may have a greater impact than the algorithm itself. Therefore, expanding the dataset would be an efficient solution for improving the model.

However, for the unsupervised learning model, few instances are labeled, so we need to implement adaptive analysis and self-learning AI to automatically enable our model to be sensitive to recently discovered fraud cases. Fig. 1 illustrates the process of implementing a hyper-focused feedback loop. When one transaction is being investigated, whether the transaction is legitimate or fraudulent is input into the adaptive model, and one adaptive score is worked out to influence the score of this event. Then, we evaluate this blended score and produce our final outcomes.

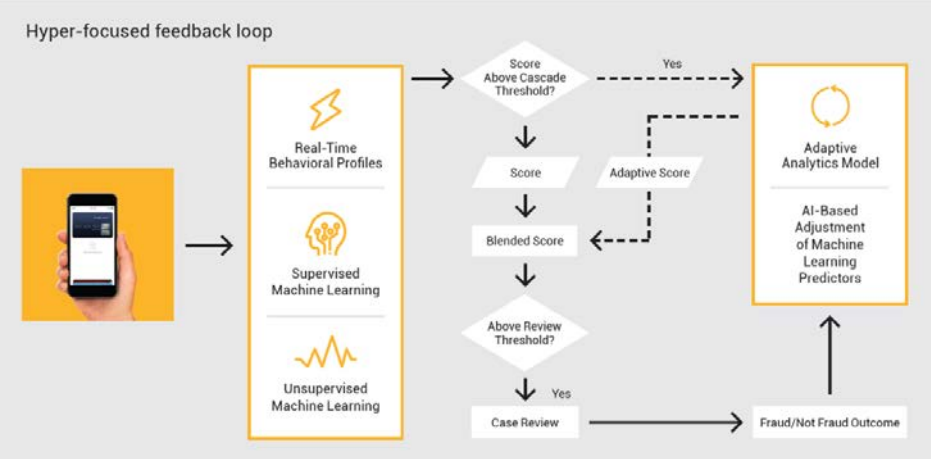

Fig.1 Working Process of Hyper-Focused Feedback Loop [3]

Another important factor of fraud detection lies in behavioral analytics. Behavioral analytics involves tracking profiles about behavioral information of each individual, account and device. Considering the fact that fraud cases may vary significantly depending on distinct people, places or card types, it would also be important to count domain knowledge during this analyzing process.

2.2 Calibrating Probability with Undersampling for Unbalanced Classification

One important issue in the fraud detection problem is that there is a large gap between the number of samples of normal and fraud transactions. A common strategy for tackling this problem is to undersample the majority class so that it can have the same size as the minority class. However, by doing this we assume that the undersampling process does not change the within-class distribution.

This essay implements Bayes Minimum Risk theory to identify the proper classification threshold and the method to readjust it after undersampling. Sample distribution bias after resampling were analyzed, and methods to minimize the risk were developed through Bayes's decision rule. By correcting the posterior probability and adjusting the threshold, calibration can be improved without losing predictive accuracy. [4]

2.3 Using Generative Adversarial Networks for Improving Classification Effectiveness in Credit Card Fraud Detection

This essay is also focused on imbalanced dataset problems. To address the imbalanced dataset issue, an augmented training set that uses GANs (Generative Adversarial Networks) to mimic the minority class was generated. A GAN consists of two feed-forward neural networks, a Generator G and a Discriminator D. In this case, G is used to produce similar data as a minority class based on machine learning models, while D is to

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

distinguish generated samples from real original samples. A dynamic game process is formed between the two networks, and in the end we can get very realistic imitation data to fill minority class. The research shows that ensemble methods perform best on the problem, which works by training several different classifiers and combining the output to produce a single decision. [5]

3. The Data

This data is the competition data of Kaggle. Vesta, a company dedicated to securing e-commerce payments, provided the raw data for this data set. This data consists of two parts (identity and transaction), both parts have the same primary key, TranscationID.

Train transaction dataset consists of a collection of information about each transaction, including transaction time, payment amount, card information, purchaser and recipient email domain as basic information of a deal. Furthermore, the information covers some other important data about safety, such as how many addresses are correlated with the payment card days since the previous transactions, whether transaction matches card details. These contents may be significant for us to perform behavioral analysis.

Identity dataset includes identity information, which is network connection information and digital signature about transactions. consists of several numerical and categorical attributes. Due to personal privacy, some features are invisible. Among this dataset, only “DeviceType”, “DeviceInfo” and “id_12”-”id_38” are categorical values, while the rest are numerical features. [6]

4. Data Cleansing

The following chapters will briefly give methods for cleaning the dataset.

4.1 Identity Dataset

The attributes in this data set are irregularly distributed, which are all multimodal properties. Many of these attributes have a large amount of NaN values, with the highest ratio of about 96% among all samples. Therefore, we used the most frequently occurring data in the dataset to supplement the null values. This will guarantee that the distribution of the data is not broken to the utmost extent.

4.2 Transaction Dataset

For the transaction dataset, the missing values in card2-card4 were deleted because the missing data in these attributes accounted for only about 1% of the total. Again, most frequently occurring number methods were implemented on most columns, while “card4” and “ProductCD” were converted from strings to numbers to enable algorithms to run in Jupyter notebook.

“Addr1” and “addr2” are both addresses. Although they always appear together in the dataset, considering that they are all expressed in numbers, separate representations will be more accurate.

“M1” - “M9” is boolean data whether the entity attribute matches the card information (i.e. names on card or address etc.). The missing values in these columns are filled with mode to ensure the distribution of these attributes is not changed.

5. Methodology

In this chapter, we will provide methodologies used in this experiment.

5.1 Encoding Attributes

Dataframes used in this project contain categorical values and cannot be used for reducing dimensionality and building classifiers. Therefore, it would be significant to encode attributes so that all of them are numerical values. Since different columns have different data distributions, a for loop was written so that each attribute should be encoded separately.

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

5.2 Dimensionality Reduction:

Here, we implemented the Principle Component Analysis (PCA) to deal with the dimensionality reduction problem. The main idea of PCA is to find a set of mutually orthogonal coordinate axes in the original space by the order of variances. For example, the selection of the first axes would be the direction with the largest variance, and the second axes with a slightly smaller variance and so on.

To find a proper dimension to maximize reducing efficiency, we selected to get PCA dimensions which have the majority of explained variance ratio (0.95 of the total), picking the first several largest orthogonal directions. We used 5 as the proper dimensionality number.

5.3 Classification

Classifiers used in this experiment are: KNearestNeighbor, LogisticRegression, GaussianNaiveBayes, SupportVectorMachine, XGBClassifier, LinearSVC.

Two functions were written in this part. The first method, Algorithm, calls indicator, training data, validation data and training labels as input and produces predicted labels as output. Based on distinct input indicators, this function will implement different kinds of algorithms. Parameters of all classifiers were adjusted to improve model performance.

The other method is about the evaluation of models by implementing performance metrics function. This function can calculate mean values of accuracy, AUC_ROC and AUC_PR to evaluate models.

5.4 Tackling Class Imbalance

As was discussed above, the most common type of approaches to tackle class imbalance problem are sampling methods. The undersampling method has a low data efficiency, and the discarded part may have important information for negative examples. However, for oversampling, since the same examples are duplicated several times, the biggest shortcoming is that our model will overfit on minority data.

Other approaches to addressing unbalanced problems include adjusting weights, changing kernel function or choosing proper models. Nevertheless, these kinds of models may be time-consuming or difficult to implement.

There are some special algorithms specifically designed for the imbalance dataset problem. EasyEnsemble and BalanceRandomForest algorithms as undersampling techniques, SMOTE (Synthetic Minority Over-sampling Technique) and ADASYN (Adaptive Synthetic) method for the use of oversampling.

EasyEnsemble: Sampling the majority class several times, and each time select the number of samples close to the number of minority classes. Then, combine this selected set with minority set and train models. Finally ensemble all models.

BalanceRandomForest: Random forest generates each tree from a bootstrap sample of the training data. However, for imbalanced data it is largely likely that a bootstrap sample contains few or even no minority class, so in each iteration, we select several samples from the minority class and the same amount from the majority class. We use these balanced samples to generate a decision tree for the CART algorithm, and ensemble them together to form a forest. [7]

SMOTE: For this algorithm, firstly we select one minority sample in our cluster, and find several nearest neighbors of it. Then, we extend our selected point in one of these nearest directions and work out a new sample. Using this method for upsampling will finally enable a balanced dataset. The following fig.2 illustrates the process of this technique. [8]

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

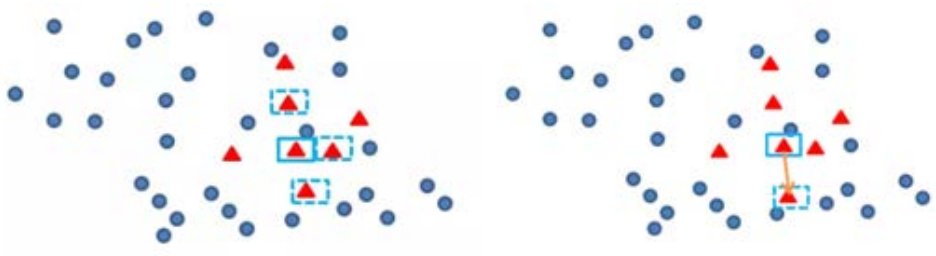

Fig.2 Process of Smote Method [8]

ADASYN: Adaptive Synthetic is an algorithm that generates synthetic data, generating more data for "harder to learn" examples.

In the first step, we calculate the ratio of minority to majority examples and calculate the number of synthetic minority data to generate. We then measure the dominance of the majority class in each specific neighborhood of minority samples. Finally, we generate data using the same technique as SMOTE (Find a point on the line segment). [9]

In this chapter, we will assess the performance of different techniques in terms of dealing with class imbalance issues. As the original dataset is too large to enable the implementation of upsampling methods, we randomly selected 1000 samples from the original dataset and implemented a series of resampling methods. The following tables tab.1 to tab.6 illustrates performance metrics of different resampling methods and different algorithms:

Table 1 Performance of Classifiers Trained on the Original Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
AUC ROC	0.5167	0.5125	0.4949	0.5837	0.5	0.5294
AUC PR	0.5322	0.4433	0.4189	0.3297	0.516	0.4293

(Data length: 1000 = 968 not fraud + 32 fraud)

Table 2 Performance of Classifiers Trained on the Upsampled Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
AUC ROC	0.9231	0.8055	0.8500	0.7226	0.9319	0.8963
AUC PR	0.9517	0.8842	0.9091	0.8167	0.9543	0.9318

(Data length: 1936 = 968 not fraud + 968 fraud)

Table 3 Performance of Classifiers Trained on the Downsampled Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
AUC ROC	0.7000	0.7071	0.6405	0.6762	0.6976	0.7643
AUC PR	0.7685	0.7587	0.7073	0.7424	0.7453	0.8253

(Data length: 64 = 32 not fraud + 32 fraud)

Table 4 Performance of Classifiers Trained on the Smote Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
AUC ROC	0.7258	0.6348	0.6946	0.4577	0.7026	0.8570
AUC PR	0.8197	0.7674	0.7813	0.6115	0.7989	0.8538

(Data length: 1936 = 968 not fraud + 968 fraud)

Table 5 Performance of Classifiers Trained on the Adasyn Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
AUC ROC	0.7479	0.7000	0.7084	0.5130	0.7418	0.8625
AUC PR	0.8287	0.7819	0.7891	0.5788	0.8183	0.8988

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

(Data length: 1932 = 968 not fraud + 964 fraud)

Table 6 Performance of Easyensembleclassifier and Balancedrandomforestclassifier

	EasyEnsembleClassifier	BalancedRandomForestClassifier
AUC ROC	0.7975	0.8302
AUC PR	0.4639	0.4822

By comparison of these different methods, it could be observed that all resampling methods have better performance comparing with the original dataset, among which upsampled dataset using the sklearn resample method has the best performance, other upsampled approaches SMOTE and ADASYN has similar performance and not bad. Generally, the performance of downsampling methods is not as good as upsampling ones. Between two kinds of special downsampling methods, BalancedRandomForest has better metrics.

6. Experiments and Results

We got a more complete dataset through the data cleaning step. Then, we will further process the data, deal with imbalance problems and train classifiers.

6.1 Class Resampling

The original dataset has an amount of 569877 not fraud data and 20663 number of fraud data, indicates a ratio of 96.5% versus 3.5%. As data length of not fraud data is too big, it would extremely computationally complex and time-consuming to implement upsampling approaches. Consequently, we used the most efficient downsampling (resampling) method.

6.2 Data Processing for Classification Methods:

In each iteration, we need to separate attributes with the label, (fraud or not fraud) and use PCA to reduce dimension. For the classification process, we used the StratifiedKfold method to split the original input dataset into five parts, and implement two pre-defined methods depending on different splits. Finally, average performance metrics of different approaches were calculated out, tab.7 illustrates the performance of all classifiers.

Table 7 Performance of Different Classifiers on the Balanced Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
Model accuracy mean	0.9149	0.8373	0.8785	0.6175	0.9124	0.8632
Model AUC ROC mean	0.9149	0.8373	0.8785	0.6175	0.9124	0.8632
Model AUC PR mean	0.9440	0.8732	0.9081	0.6957	0.9410	0.9165

All models have relatively good performance, and among them, kernelized SVM classifier with 'RBF' kernel has the highest performance metrics values. The following table, tab.8 illustrates the confusion matrix of this classifier based on distinct kinds of splits.

Table 8 Confusion Matrix of the Best Performed Ksvm Classifier

	Split 1	Split 2	Split 3	Split 4	Split 5
True positive	3741	3932	4005	4025	3785
False positive	307	116	43	23	263
False negative	2131	0	378	282	2
True negative	1917	4048	3670	3766	4046

6.3 Digging into Attribute Correlations

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

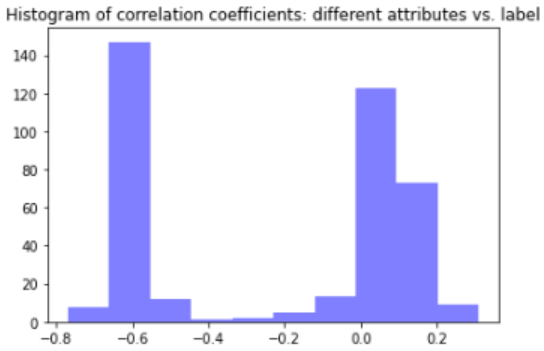

Fig.3 Histogram of Correlation Coefficients

To optimize the attribute correlations, we further dig into the correlations between different classifiers. We calculate correlation coefficients between each attribute and the label and then plot a histogram of these coefficients. The following figure fig.3 illustrates this histogram:

It could be observed that attributes who have the highest correlation with labels lie in the range of -0.8 to -0.4. We found all the attributes in this range and formed a high correlation attribute dataset with the dimension of 40480 * 167. Then the same process (encoding, reducing dimension and classification) was executed. Tab. 9 shows the result of this process:

Table 9 Performance of Different Classifiers on the Highly-Correlated Dataset

	KNN	NaiveBayes	Logreg	LSVM	KSVM	XGB
Model accuracy mean	0.8781	0.8269	0.8296	0.6388	0.9156	0.9573
Model AUC ROC mean	0.8781	0.8269	0.8296	0.6388	0.9156	0.9573
Model AUC PR mean	0.9268	0.8666	0.8774	0.7664	0.9459	0.9761

By comparison, generally highly-correlated classifiers can improve classifier performance, especially for XGB classifiers.

7. Conclusion

Fraud detection is an important issue to enable credit card payments to execute properly. In this essay, we firstly gave a background introduction about factors affecting fraud detection and some recent research on class imbalance; then we described data condition and cleaning process; in the following part we gave a detailed analysis of the methodology used, encoding, dimensionality reduction, and classification methods; we particularly focus on upsampling and downsampling techniques to deal with class imbalance issue, and tested on our dataset to find the performance of each technique; we then work out classification results on the entire dataset; finally, we explored into correlations between each attribute and improved performance a little. The experimental results well proved that tackling the class imbalance approach has a positive effect on imbalanced data sets, and digging into the relationship among attributes would also be useful.

References

[1] Bachmann, J (2019) Credit Fraud || Dealing with Imbalanced Datasets. [Online] Available at: https://www.kaggle.com/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets
[2] Sanagapti, P. (2019). Anomaly Detection – Credit Card Fraud Analysis. [Online] Kaggle.com. Available at: https://www.kaggle.com/pavansanagapati/anomaly-detection-credit-card-fraud-analysis
[3] TJ, Horan (2018) 5 Keys to Using AI and Machine Learning in Fraud Detection | FICO [Online]
[4] Available at: https://www.fico.com/blogs/5-keys-using-ai-and-machine-learning-fraud-detection
[5] Dal Pozzolo, A., Caelen, O., Johnson, R.A (2015). December. Calibrating probability with undersampling for unbalanced classification. In 2015 IEEE Symposium Series on Computational Intelligence, pp.159-166.
[6] Fiore, U., De Santis, A., Perla, F, et al (2019). Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. Information Sciences, no.479, pp.448-455.
[7] Kaggle.com (2019). IEEE-CIS Fraud Detection. [online] Available at: https://www.kaggle.com/c/ieee-fraud-detection

ISSN 2616-7433 Vol. 2, Issue 11: 96-103, DOI: 10.25236/FSST.2020.021115

[8] Chen, C., Liaw, A, Breiman, L (2015). Using random forest to learn imbalanced data. 2004. University of California, Berkeley.
[9] Chawla, N.V., Bowyer, K.W., Hall, L.O, et al (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, no.16, pp.321-357.
[10] Nian, R (2019). Fixed Imbalanced Datasets: An Introduction to ADASYN (with code!). [online] Medium. Available at: https://medium.com/@ruinian/an-introduction-to-adasyn-with-code-1383a5ece7aa