Oversampling based on generative adversarial networks to overcome imbalance data in predicting fraud insurance claim

Ranu A. Nugraha[1,*], Hilman F. Pardede[1,2], Agus Subekti[1,2]
[1]*Faculty of Information Technology, Graduate School of Computer Science, Nusa Mandiri University, Jakarta, Indonesia*
[2]*National Research and Innovation Agency, Indonesia*
**Corresponding author: ranu.agastya@gmail.com*

Abstract

Fraud on health insurance impacts cost overruns and a quality decline in health services in the long term. The use of machine learning to detect fraud on health insurance is increasingly popular. However, one challenge in predicting health insurance fraud is the data imbalance. The data imbalance can cause a bias towards the majority class in many machine learning methods. Oversampling is a solution for data imbalance by augmenting new data based on the existing minority class data. Recently, there has been growing interest in employing deep learning for data augmentation. One of them is using Generative Adversarial Networks (GAN). This paper proposes using GAN as an oversampling method to generate additional data for minority classes. Since data for detecting health insurance fraud are tabular, we adopt Conditional Tabular GAN (CTGAN) architecture where the generator is conditioned to adjust the tabular data input and receive additional information to produce samples according to the specified class conditions. The new balanced data are used to train 17 classification algorithms. Our experiments showed that the proposed method performs better than other oversampling methods on several evaluation metrics, i.e., accuracy, precision score, F1-score, and ROC.

Keywords: Fraud insurance detection, generative adversarial networks, imbalance data, oversampling, tabular GAN

1. Introduction

Health insurance becomes one vulnerable target to claims from providers (Thornton *et al.*, 2013). The impact of fraud results in an increased cost of health services and a decreased quality of health services provided to recipients in the long term. Finding violation evidence of a health claim is very difficult because of the complex transaction costs associated with many parties, e.g., healthcare facility providers, treating doctors' treatment, medicine/drug-related procedures, and types of disease (Kareem *et al.*, 2018).

The Covid-19 epidemic has sparked the future trend of using AI technologies in the insurance sector (Deloitte, 2021). Data mining and machine learning have gained a role in detecting and preventing hazards (Kirlidog & Asuk, 2012; Rayan, 2019). One problem in detecting fraud using machine learning is the imbalanced distributions of data in each class (Chen & Chen, 2020), where one may have significantly more data than another. In fraud insurance detection, the fraud class is usually the minority, so the model tends to produce a bias and is not feasible to implement since the prediction is typically inaccurate for the minority class (Anbarasi & Dhivya, 2017). Oversampling is one solution to deal with imbalanced data. It works by generating "new data" for the minority class, producing a similar number of data to the majority class. Various oversampling techniques and algorithms have been applied and work well in dealing with class imbalance problems, e.g., Random Over Sampling (ROS) and Synthetic Minority Over-sampling Technique (SMOTE) (Zhu *et al.*, 2019). However, there are also disadvantages.

For instance, when using SMOTE to find the nearest neighbor and add data linearly, problems occur when faced with high-dimensional data and complex distributions (Engelmann & Lessmann, 2021).

In recent years, there has been a growing interest in applying the deep learning method, mainly when dealing with complex and nonlinear relations (Wang *et al.*, 2016). Deep learning is also suitable for big, multi-dimensional data that can be executed through a hidden network structure that supports important information for increased accuracy (Li *et al.*, 2018). This method allows simultaneous classification, recognition, and prediction (Datsi *et al.*, 2022). Recently, a new deep learning family called generative adversarial networks (GANs) was introduced. It can generate artificial data close to the original data (Goodfellow *et al.*, 2014) and be implemented for data augmentation to overcome data imbalances (Tanaka *et al.*, 2019). GAN is an alternative method for generating synthetic data on minority classes based on their complex distribution (Engelmann & Lessmann, 2021). At the algorithm level, the deep learning method deals with the class imbalances by separating the classifications, resulting in a more stable result than other methods (Johnson & Khoshgoftaar, 2019).

In this paper, we employ GAN to overcome the imbalanced data problem for fraud detection of health insurance claims by health facility providers. GANs are primarily used to process image data and have limitations regarding tabular data processing (Xu *et al.*, 2019). To overcome the limitation of GAN, we use Conditional Tabular GAN (CTGAN) instead of vanilla GAN as an oversampling method to generate new data for the minority class. When processing continuous data, such as in the real world, CTGAN employs a unique normalization mode. The generator is conditioned to adjust the tabular data input and receive additional information to generate new samples. We evaluate the proposed method by using them to train 17 different classifiers. Our experiments suggest that CTGAN is better than other reference methods for dealing with imbalanced data such as RUS, ROS, and SMOTE.

2. Related Work

The traditional approach to fraud detection is based on the comprehensive development of fraud indicators. Automotive insurance is the most established type of insurance fraud because it originates from false accident claims (Anbarasi & Dhivya, 2017). Several studies have employed machine learning methods to detect fraudulent claims by health facility providers, e.g., the decision tree, Support Vector Machine (SVM), and Back Propagation Neural Network (BPNN) (Chen & Chen, 2020), also random forest and XGBoost (Akbar *et al.*, 2020). Another study (Lavanya *et al.*, 2021) reported the Multilayer Perceptron algorithm with the best accuracy compared to five other models (i.e., Logistic Regression, Random Forest, Support Vector Machine, Adaboost, and Gradient Boosting Trees). Even though these studies achieved very high accuracy (up to 92.97%), the sensitivity and performance (AUC) values were relatively low (48.68% and 0.527, respectively), suggesting a low prediction of the minority class. In a study employing three under-sampling techniques and six classification variants of classification models (Gupta *et al.*, 2021), the Neural Network model was the best. Another study employs a hybrid of the supervised and unsupervised learning methods (Sheshasaayee & Thomas, 2018). An anomaly detection technique was used to measure the likelihood of fraudulent health insurance claims records based on historical claims (Kirlidog & Asuk, 2012). Another study (Zhang *et al.*, 2020) detected anomalies by quantifying disease–drug relationships in association scores and other features with a neural network with fully connected layers and infrequent convolutions.The study introduced focal-loss function to adapt to unbalanced data as well as relative probability scores to measure model performance. In another study (Herland *et al.*, 2019), random under sampling (RUS) and SMOTE were applied to deal with imbalance data.

Recently, interests in deep learning have multiplied. One recent architecture that attracts interest is GAN, used initially as a generative model for synthesizing data. We could also employ GAN to generate synthetic data to deal with data imbalance problems. Stochastic Gradient Descent based GAN (SDG-GAN) framework has been reported (Charitou *et al.*, 2021) to improve classification performance on the gambling fraud dataset. GAN has also been used as an alternative to generating synthetic data on minority classes based on their ability to learn complex distribution (Engelmann & Lessmann, 2021). While originally GAN was applied to generate image data, studies showed its capability of generating tabular data. The oversampling method based on the Wasserstein GAN was reported able to model

tabular data consisting of numeric and categorical data (Arjovsky *et al.*, 2017; Engelmann & Lessmann, 2021). In this study, we employ GAN in handling data imbalances for health insurance fraud detection.

3. Method

This research was carried out in several stages, i.e., data pre-processing, oversampling, and classification. We used 17 classification algorithms as prediction models, i.e., Logistic Regression (LR), Ridge Classifier (Ridge), Passive Aggressive (PsvAgr), Stochastic Gradient Descent (SGD), k-Nearest Neighbor (kNN), Decision Tree (DT), Extra Tree Classifier (ExTr), Linear Support Vector Classifier (LSVC), Support Vector Classifier (SVC), Gaussian Naïve Bayes (GNB), Adaptive Boosting Classifier (Adaboost), Bagging Classifier, Random Forest (RF), Extra Trees Classifier (ExTrees), Gradient Boosting Classifier (GBoost), Linear Discriminant Analysis (LDA), and Quadratic Discriminant Analysis (QDA). The algorithms are used through a stratified cross-validation scheme and evaluating the performance classifier model with a combination of oversampled data.

3.1 Data Preprocessing

In this study, we focus on fraud detection at the level of health care providers since this level allows a greater possibility of deviations (Thornton *et al.*, 2013). At the provider level, a provider relates to the entire claim, and the whole claim relates to the patient concerned. Provider behavior can be statistically measured using variables such as the number of transactions, average value, maximum value, minimum value, and standard deviation of a category value (Lin & Haug, 2006).

This study uses secondary data from the Kaggle data repository regarding Healthcare Provider Fraud Detection Analysis data (Gupta, 2019). The dataset consists of four files. i.e., beneficiary, inpatient, outpatient, and fraudulent provider data. The data processing was carried out in several steps. First, the data integration process of the four datasets was done to combine all of them (Figure 1). The data integration technique yielded combined table data with dimensions of 558,211 sample data and 58 features.

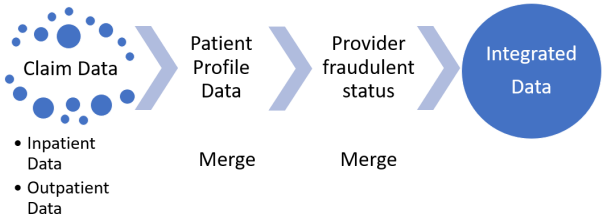

Fig. 1. Diagram of Data Integration Process.

Second, we conversed the attribute data of type date, encoded and discretized them. We transformed all categorical data types to convert them into numeric types. We applied dummyfication of gender and race data into the one-hot-encoding form. The average claim reimbursement value was calculated based on the codes from treating doctor's, diagnosis, and action for each claim data. We then calculated the unique number for categorical input. We summarized the presence of conditions from each feature that describes the diversity of demographic conditions and patient congenital diseases for each provider. Then, we summarized the total number of patients served by each provider. After applying all mentioned processes, we obtained a total of 5410 samples representing the number of health care provider IDs available in the data, with 57 features and one PotentialFraud feature as the target label. We then applied a normalization with the StandardScaler technique to ensure the data ranges from 0 to 1 without changing their distributions.

3.2 CTGAN Architecture

In this paper, we adopt the Conditional Tabular Generative Adversarial Networks (CTGAN) architecture as in (Xu *et al.*, 2019). The CTGAN architecture comprises two networks, i.e., Generator and

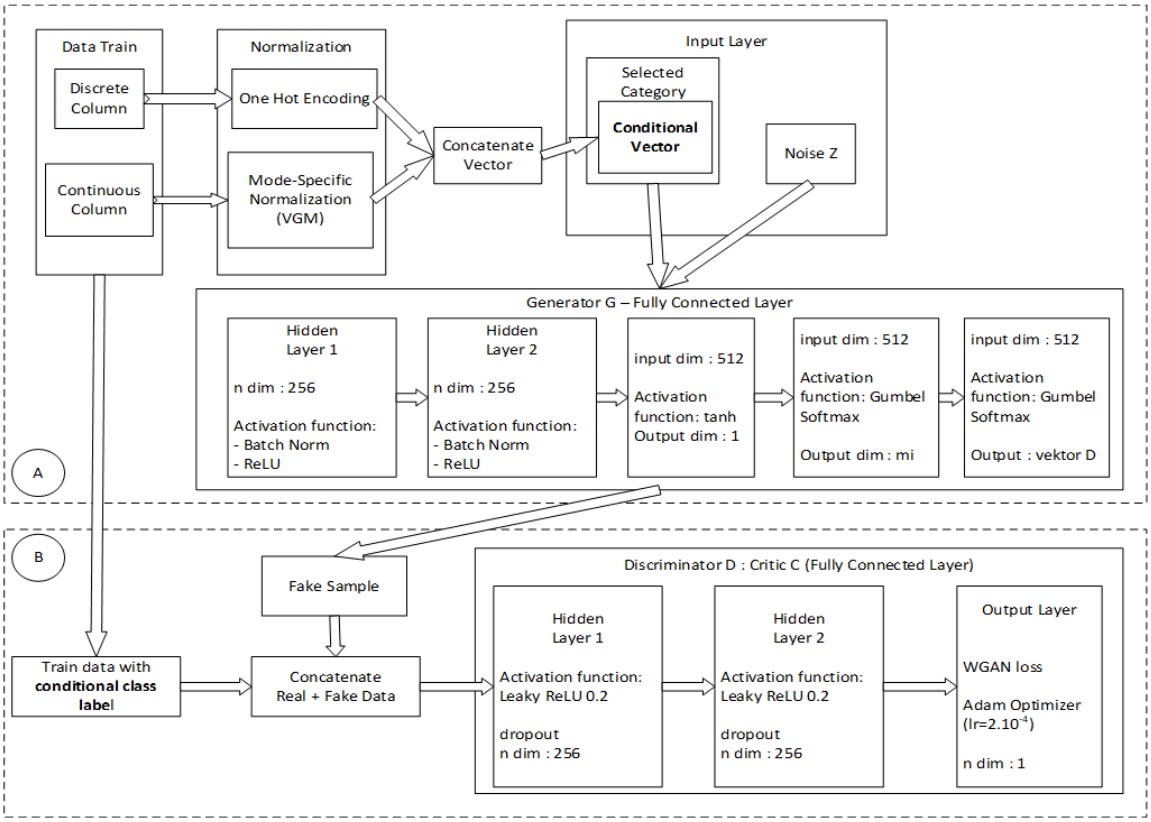

Fig. 2. Conditional Tabular GAN Architecture.

Discriminator (Figure 2). Originally for GAN, the generator inputs a random noise. We added data features from minority classes with random noise (denoted z in the figure) for data augmentation. Training data features are transformed into one-hot encoding for categorical types while a normalization with a variational Gaussian Mixture model is applied to continuous data types. Then, the resulting features are concatenated to produce a conditioned vector. The vector will be used as a feature together with randomly generated noise. The first hidden layer is a Fully Connected Layer with 256 nodes. Here, the linear transformation process of adding vector and noise is performed. We applied Batch Normalization to the ReLU activation function in this layer. The second hidden layer is a Fully Connected Layer with 512 output nodes and Batch Normalization and activation functions. The same linear transformation process is also employed here. The final layers transform neuron dimensions from 512 to 1 output neuron. Gumbel softmax with parameter 0.2 in vector alpha is used.

The discriminator ((Critic C) in part (B)), through the PacGAN framework architecture, used 10 samples in each pac, to avoid model damage. The architecture is as follows. First, 10 pacs were combined with 10 conditioned vectors as an input layer. In the first hidden layer, a linear transformation was carried out in a fully connected layer from the input dimensions to produce 256 dimensions at the output. A leaky ReLU activation function was applied with parameter 0.2, and a dropout function was performed on each input layer. In the second hidden layer, a linear transformation was also carried out in a fully connected layer from 256 dimensions of the hidden layer to produce 256 dimensions at the output. A leaky ReLU activation function was applied with parameter 0.2, and a dropout function was performed on each input layer. From the second hidden layer with 256 dimensions, a linear transformation was carried out using loss on WGAN with a penalty on the gradient and using adam optimization with a learning rate of 2e-4 to produce 1 output dimension.

4. Experiments

4.1 Setup

The experiments were carried out using the jupyter notebook tools provided by Google colab with the sklearn, imblearn libraries, and GPU access. Data oversampling with CTGAN was done by utilizing the Synthetic Data Vault (SDV) library. CTGAN was run with epochs number set to 1000. We set the number of embedding layers to 128, the generator's hidden layers are set to 256 and 256, respectively, whereas the discriminator's hidden layers are assigned to the exact dimensions. We set batch_size into 500, the learning rate to 2e-4, and the generator decay to 1e-6. For discriminator, the learning rate was set to 2e-4 and discriminator decay to 1e-6. Several oversampling methods such as ROS, RUS, SMOTE, B-SMO, and ADASYN were also applied to original datasets with default parameters. Then, 17 classification algorithms were applied to the oversampled data as predictive models above the baseline parameters. To train the classifier, we applied stratified KFold Cross Validation at k=10.

4.2 Results and Discussions

The training progress of CTGAN was up to 1000 epochs (Figure 3). The generator loss was high initially but gradually decreased as the training progressed. Meanwhile, the discriminator relatively increases as the epoch progresses. These indicate that the learned probability is getting closer to actual data. Here CTGAN is trained with data of the minority class only, i.e., the fraud class (PotentialFraud=1).

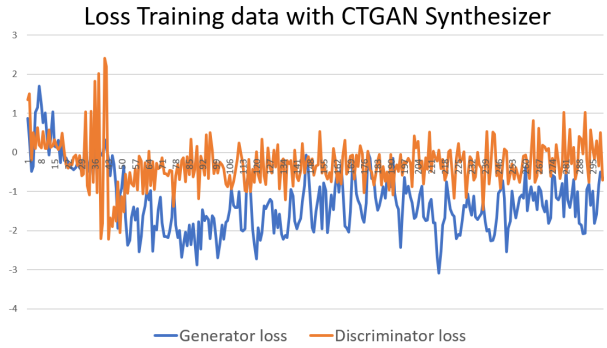

Fig. 3. Process of training data with CTGAN.

Table 1. Dimensions of the data before and after applying CTGAN on training dataset

Data Class	Without CTGAN	Oversampling with CTGAN	New Data
PotentialFraud=0	4904	0	4904
PotentialFraud=1	506	4398	4904
row count	5410	4398	9808

After training the CTGAN, it was then used to generate the minority class data to have the same number as the majority class (PotentialFraud=0). The generated data with the original data are combined, resulting in 4904 minority classes and 9808 training data (See Table 1 for details).

To compare the results of generated and real data, we plot the distributions of generated and real data in Figures 4 and 5. The sample comparison of the data distribution between the original and synthetic data is depicted in Figure 4. The log mean values and the standard deviation log values for generated and real data are located around the line. While they are not the same, they tend to be close together, indicating that the generated and real data are quite similar.

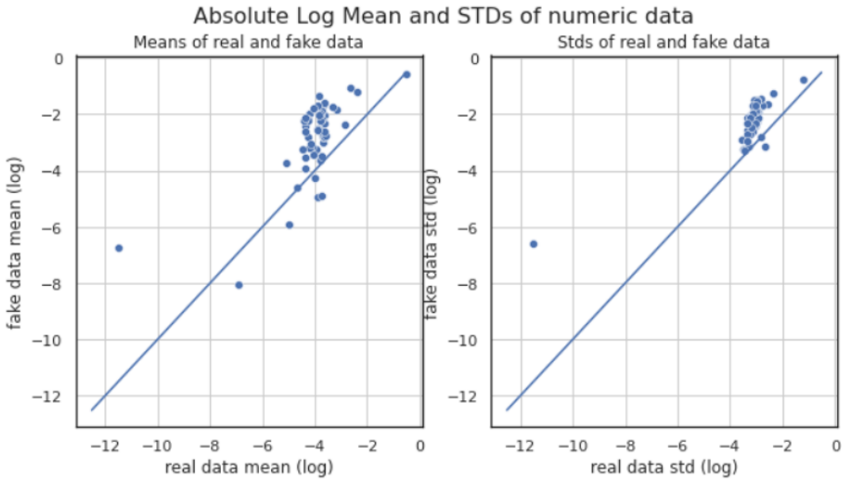

Fig. 4. The evaluation results of the log mean and log standard deviation of the sample data compared to the original data.

The density of BeneID feature (the number of participants served by the health service provider) and the distribution of the features of the original and the generated one are shown in Figure 5. The plot diagram of the cumulative numbers of data of the features appears to have the same normal distribution where the mean and variance are 0 or Gaussian (Figure 6). Similarly, real and generated data distributions are not the same. Still, the generated data have a density resembling the original data distribution. The difference density could also benefit the training of machine learning methods since the generated data are variations of the real data, improving the generalization capability of the machine learning models.

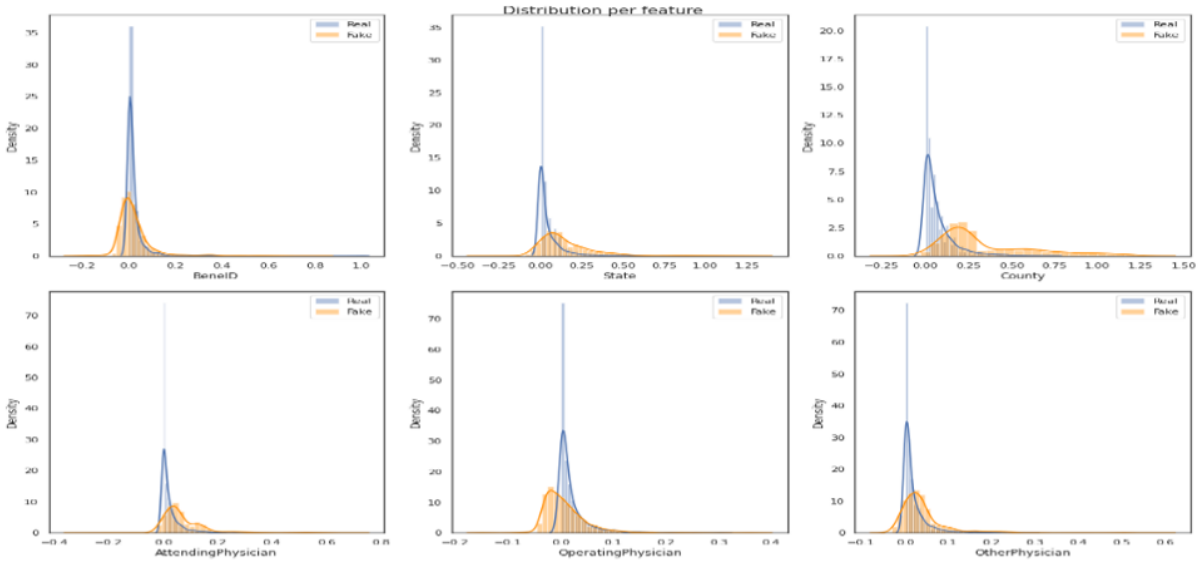

Fig. 5. Distribution Comparison of Original Data with False Data.

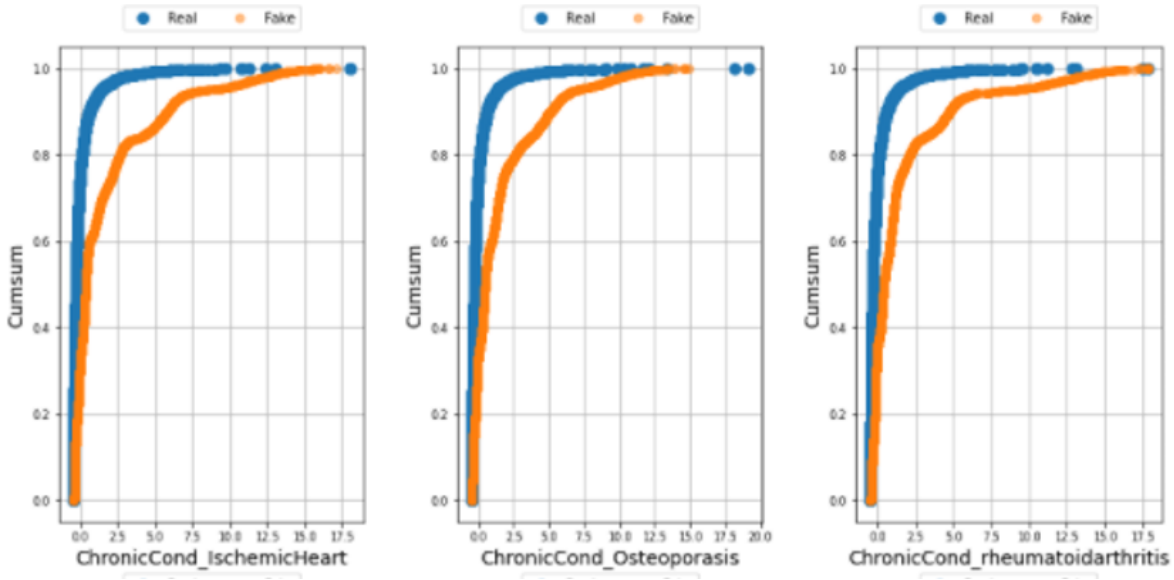

Fig. 6. Plot Diagram Comparison of the Cumulative Amount of Each Feature.

The results of 17 classification models using the new data are summarized in Tables 2 to 6. The results are compared to several resampling methods, i.e., SMOTE, RUS, ROS, B-SMO, and ADASYN.

Table 2. Comparison of Performance Results of Average Accuracy on test data

No	Model	CTGAN	SMOTE	RUS	ROS	B-SMO	ADASYN
1	LR	92.63	87.34	83.01	86.85	89.09	84.87
2	Ridge	83.54	83.96	81.53	83.5	86.51	82.07
3	SGD	92.64	86.87	81.52	86.09	88.47	84.99
4	PsvAgr	89.56	75.24	79.34	82.07	86.61	75.36
5	KNN	88.14	92.57	82.51	94.36	93.07	91.29
6	DT	94.08	93.78	79.64	97.00	93.72	92.49
7	ExTr	90.46	91.71	77.76	97.16	92.95	91.1
8	LSVC	92.45	86.88	83.31	86.27	88.82	84.71
9	SVC	93.21	87.55	83.51	87.27	91.13	86.65
10	GNB	76.4	73.61	72.34	73.95	70.62	67.33
11	AdaBoost	95.45	92.13	84.19	88.49	92.57	91.09
12	Bagging	95.56	96.17	84.09	97.92	95.70	96.08
13	RF	95.89	96.42	84.98	98.34	96.03	96.36
14	ExTrees	95.99	96.71	84.59	99.11	96.33	96.48
15	GBoost	95.96	95.04	85.77	92.18	94.84	94.49
16	LDA	83.54	83.92	81.33	83.39	86.37	82.08
17	QDA	95.59	75.3	74.72	78.12	73.39	69.63
	Max	95.99	96.71	85.77	99.11	96.33	96.48
	Average	**91.24**	87.95	81.42	88.94	89.19	86.30
	Std.dev	0.055	0.075	0.036	0.075	0.073	0.089

On average, the achieved accuracy of CTGAN (91.24%) is better than other referenced methods (Table 2) by 3.29 %, 9.89 %, 2.3 %, 2.05 %, and 4.94 % to SMOTE, RUS, ROS, B-SMO, and ADASYN, respectively. The proposed method does not consistently achieve the best accuracy for all classifiers. For instance, SMOTE is slightly better when using KNN or ensemble methods (RF, ExTrees, GBoost, etc.). This is not surprising as SMOTE employs some KNN principles when generating new data. SMOTE only performs slightly better.

Table 3. Comparison of the Average Recall on test data

No	Model	CTGAN	SMOTE	RUS	ROS	B-SMO	ADASYN
1	LR	90.11	84.48	77.69	83.52	88.93	82.55
2	Ridge	72.47	78.63	74.73	77.73	84.38	77.84
3	SGD	92.52	85.97	79.62	84.85	90.01	84.93
4	PsvAgr	87.83	78.69	79.45	81.67	89.17	78.53
5	KNN	79.06	99.73	81.44	99.84	99.31	99.64
6	DT	94.05	95.49	80.63	100	95.15	93.86
7	ExTr	89.25	94.00	79.64	100	94.98	93.88
8	LSVC	89.66	83.59	78.29	82.36	88.62	82.27
9	SVC	93.21	87.46	79.05	86.46	95.53	90.22
10	GNB	58.89	53.26	50.41	53.75	49.10	42.72
11	AdaBst	95.29	94.84	84.60	90.72	96.10	94.45
12	Bagging	94.19	97.06	83.59	99.96	96.82	97.72
13	RF	95.45	98.35	86.55	100	98.33	98.87
14	ExTrees	95.25	99.06	86.16	100	98.98	99.37
15	GBoost	95.33	95.92	87.15	95.41	96.94	95.56
16	LDA	72.47	78.49	74.33	77.61	84.07	77.82
17	QDA	93.41	61.14	59.51	65.89	58.97	52.43
	Max	95.45	99.73	87.15	100	99.31	99.64
	Average	87.55	86.25	77.81	87.05	**88.55**	84.86
	Std.dev	0.105	0.133	0.095	0.134	0.139	0.161

Table 4. Comparison of Performance of Average Precision on test data

No	Model	CTGAN	SMOTE	RUS	ROS	B-SMO	ADASYN
1	LR	94.91	89.62	86.86	89.49	89.25	86.73
2	Ridge	93.10	88.05	86.54	87.90	88.16	85.22
3	SGD	92.75	87.60	82.93	87.10	87.35	85.16
4	PsvAgr	91.30	75.92	79.51	82.66	84.98	75.96
5	KNN	96.61	87.24	83.28	90.00	88.30	85.46
6	DT	94.12	92.34	79.22	94.37	92.51	91.45
7	ExTr	91.46	89.90	76.97	94.63	91.29	89.03
8	LSVC	94.96	89.52	86.90	89.36	88.99	86.66
9	SVC	93.23	87.64	86.71	87.89	87.82	84.34
10	GNB	90.63	89.83	90.14	90.20	86.20	84.68
11	AdaBoost	95.62	89.99	84.11	86.86	89.77	88.61
12	Bagging	96.88	95.37	84.40	96.06	94.72	94.68
13	RF	96.31	94.71	83.96	96.80	94.02	94.19
14	ExTrees	96.70	94.62	83.52	98.27	94.01	93.98
15	GBoost	96.56	94.28	84.81	89.64	93.05	93.63
16	LDA	93.10	88.09	86.52	87.78	88.14	85.25
17	QDA	97.66	85.29	85.41	87.22	82.85	80.30
	Max	97.66	95.37	90.14	98.27	94.72	94.68
	Average	**94.47**	89.41	84.22	90.37	89.49	87.37
	Std.dev	0.022	0.045	0.032	0.042	0.033	0.051

Meanwhile, the recall of CTGAN is only lower than B-SMO (see Table 3). But, individually, the CTGAN method can produce the highest recall value of 95.45% with the Random Forest classification algorithm model. However, CTGAN is still better than SMOTE, RUS, ROS, and ADASYN. B-SMO has an average recall score of 87.55 %, with a standard deviation of 0.10. CTGAN is better than SMOTE, RUS, ROS, and ADASYN by 1.3 %, 9.74 %, 0.5 %, and 2.69 % respectively. In general, the proposed method only slightly improves the recall value. CTGAN achieves the highest precision score of 94.47% (Table 4). Improvements of 5.06%, 10.25, 4.10 %, 4.98 %, and 7.10 are achieved compared to SMOTE, RUS, ROS, B-SMO, and ADASYN, respectively.

Table 5. Comparison of the Performance Results of the Average AUC on test data

No	Model	CTGAN	SMOTE	RUS	ROS	B-SMO	ADASYN
1	LR	96.90	94.37	91.55	94.01	94.82	91.47
2	Ridge	94.67	92.47	89.91	92.22	93.79	89.81
3	SGD	96.38	93.42	86.91	92.54	93.89	90.63
4	PsvAgr	93.43	84.21	85.85	89.36	91.85	82.05
5	KNN	95.76	96.81	88.41	97.32	96.59	96.24
6	DT	94.08	93.78	79.64	97.00	93.72	92.48
7	ExTr	90.46	91.71	77.75	97.16	92.95	91.09
8	LSVC	96.76	94.04	90.46	93.73	94.63	91.18
9	SVC	97.96	93.88	91.84	93.54	95.55	92.45
10	GNB	91.19	87.56	85.87	88.14	88.61	83.37
11	AdaBoost	99.06	97.96	91.62	95.48	98.11	97.46
12	Bagging	98.60	99.04	90.26	99.79	98.92	99.15
13	RF	99.21	99.52	92.52	100	99.45	99.55
14	ExTrees	99.23	99.69	92.11	100	99.63	99.74
15	GBoost	99.24	98.89	92.01	97.17	98.98	98.83
16	LDA	94.67	92.49	89.49	92.20	93.81	89.80
17	QDA	98.85	86.44	86.23	88.81	86.62	81.39
	Max	99.24	99.69	92.52	100	99.63	99.74
	Average	**96.26**	93.90	88.38	94.61	94.82	92.16
	Std.dev	0.028	0.046	0.043	0.038	0.036	0.059

Table 6. Comparison of the Performance Results of the F1-Score Test Average

No	Model	CTGAN	SMOTE	RUS	ROS	B-SMO	ADASYN
1	LR	92.44	86.97	81.96	86.39	89.07	84.57
2	Ridge	81.48	83.05	80.13	82.48	86.21	81.34
3	SGD	92.63	86.75	80.94	85.90	88.62	85.04
4	PsvAgr	89.24	76.54	79.25	81.95	86.87	76.63
5	KNN	86.94	93.07	82.30	94.66	93.48	92.00
6	DT	94.08	93.89	79.85	97.10	93.81	92.63
7	ExTr	90.34	91.90	78.17	97.24	93.10	91.38
8	LSVC	92.23	86.43	82.31	85.71	88.79	84.39
9	SVC	93.21	87.54	82.60	87.16	91.51	87.17
10	GNB	71.37	66.85	64.29	67.34	62.51	56.77
11	AdaBoost	95.45	92.34	84.26	88.74	92.82	91.42
12	Bagging	95.51	96.20	83.93	97.97	95.75	96.17
13	RF	95.87	96.49	85.20	98.37	96.12	96.47
14	ExTrees	95.96	96.79	84.76	99.12	96.43	96.60
15	GBoost	95.94	95.09	85.91	92.43	94.95	94.58
16	LDA	81.48	82.99	79.89	82.36	86.05	81.35
17	QDA	95.48	71.19	69.95	75.05	68.86	63.39
	Max	95.96	96.79	85.91	99.12	96.43	96.60
	Average	**90.57**	87.30	80.34	88.23	88.53	85.40
	Std.dev	0.067	0.089	0.055	0.089	0.093	0.112

Based on the AUC scores (Table 5), CT-GAN is also better on average(96.26%). This value is better by 2.36 %, 7.88%, 1.65 %, 1.44 %, and 4.1 % to SMOTE, RUS, ROS, B-SMO, and ADASYN respectively.

CTGAN produces the highest average F1-Score value of 90.57% (Table 6). It improves SMOTE, RUS, ROS, B-SMO, and ADASYN by 3.27 %, 10.23 %, 2.34 %, 2.04 %, and 5.17 % respectively.

From the experimental result, using GAN as oversampling has several drawbacks, including the inability to find a maximum convergent point between loss D and loss G due to its adversarial characteristics. GAN has a higher cost of complexity than other oversampling approaches. However, conditionals in CTGAN merely condition the generator model to gain extra information in the form of a specified class label. As a result, the trained generator may create sample data with class criteria chosen from all

conceivable discrete values, and this can help shorten the model in training data.

5. Conclusion

In this paper, we proposed CTGAN as an oversampling method to deal with imbalanced data for predicting insurance claims fraud. Our experiments show that the proposed methods are mostly better than other resampling methods when applied to 17 classification models. Individually, the classification model combined with the CTGAN method produces the highest accuracy score from the Gradient Boosting (GBoost) model with an average accuracy of 95.96%. The average recall performance value using the CTGAN method as oversampling was obtained at 87.55%, below the B-SMO method (88.55%). However, the CTGAN method can individually produce the highest recall value of 95.45% with the Random Forest classification algorithm model.

Our evaluations of the generated data also show that CTGAN can produce similar distributions of the real data without ever seeing them. However, the dissimilarity can also be interpreted as expanding the distribution space of the actual data that may cover data variations that may not exist in the real data. However, compared to other oversampling methods, CTGAN still faces challenges, such as the complexity cost of training data.

References

Akbar, N. A., Sunyoto, A., Arief, M. R., & Caesarendra, W. (2020). Improvement of decision tree classifier accuracy for healthcare insurance fraud prediction by using Extreme Gradient Boosting algorithm, Proceedings - 2nd International Conference on Informatics, Multimedia, Cyber, and Information System, ICIMCIS 2020, pp. 110–114. doi: 10.1109/ICIMCIS51567.2020.9354286.

Anbarasi, M. S. & Dhivya, S. (2017). Fraud detection using outlier predictor in health insurance data, 2017 International Conference on Information Communication and Embedded Systems, ICICES 2017, (Icices). doi: 10.1109/ICICES.2017.8070750.

Arjovsky, M., Chintala, S. & Bottou, L. (2017). (WGAN) Wasserstein Generative Adversarial Network Junhong Huang, Icml, pp. 1–44.

Charitou, C., Dragicevic, S. & Garcez, A. d'Avila (2021). Synthetic Data Generation for Fraud Detection using GAN's. Available at: http://arxiv.org/abs/2109.12546.

Chen, L. S. & Chen, J. C. (2020). Using data mining methods to detect medical fraud, ACM International Conference Proceeding Series, pp. 89–93. doi: 10.1145/3409891.3409902.

Datsi, T., Aznag, K., Oirrak, A. E. (2022). Digit recognition using decimal coding and artificial neural network. Kuwait Journal of Science, 49(**1**), pp. 1–17. doi: 10.48129/kjs.v49i1.9556

Deloitte (2021). 2021 Global health care outlook, Deloitte, p. 25. Available at: https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Life-Sciences-Health-Care/gx-lshc-2016-health-care-outlook.pdf

Engelmann, J. & Lessmann, S. (2021). Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning, Expert Systems with Applications, 174(Ml). doi: 10.1016/j.eswa.2021.114582.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014). Generative Adversarial Networks, Communications of the ACM, 63(**11**), pp. 139–144. doi: 10.1145/3422622.

Gupta, R. A. (2019). Healthcare Provider Fraud Detection Analysis Dataset. kaggle. Available at: https://www.kaggle.com/rohitrox/healthcare-provider-fraud-detection-analysis.

Gupta, R. Y., Mudigonda, S. S. & Baruah, P. K. (2021). A comparative study of using various machine learning and deep learning-based fraud detection models for universal health coverage schemes, International Journal of Engineering Trends and Technology, 69(**3**), pp. 96–102. doi: 10.14445/22315381/IJETT-V69I3P216.

Herland, M., Bauder, R. A. & Khoshgoftaar, T. M. (2019). The effects of class rarity on the evaluation of supervised healthcare fraud detection models, Journal of Big Data, 6(**1**). doi: 10.1186/s40537-019-0181-8.

Johnson, J. M. & Khoshgoftaar, T. M. (2019) Medicare fraud detection using neural networks, Journal of Big Data. Springer International Publishing. doi: 10.1186/s40537-019-0225-0.

Kareem, S., Ahmad, R. B. & Sarlan, A. B. (2018). Framework for the identification of fraudulent health insurance claims using association rule mining, 2017 IEEE Conference on Big Data and Analytics, ICBDA 2017, 2018-Janua(October 2019), pp. 99–104. doi: 10.1109/ICBDAA.2017.8284114.

Kirlidog, M. & Asuk, C. (2012). A Fraud Detection Approach with Data Mining in Health Insurance, Procedia - Social and Behavioral Sciences, **62**, pp. 989–994. doi: 10.1016/j.sbspro.2012.09.168.

Lavanya, S., Kumar, S. M. & Kumar, P. M. (2021). Machine learning based approaches for healthcare fraud detection: A comparative analysis, Annals of the Romanian Society for Cell Biology, 25(**3**), pp. 8644–8654.

Li, Y., Lin, X., Wang, X., Shen, F. & Gong, Z. (2018). Credit Risk Assessment Algorithm Using Deep Neural Networks with Clustering and Merging, Proceedings - 13th International Conference on Computational Intelligence and Security, CIS 2017, 2018-Janua, pp. 173–176. doi: 10.1109/CIS.2017.00045.

Lin, J. H. & Haug, P. J. (2006). Data preparation framework for preprocessing clinical data in data mining., AMIA Annual Symposium proceedings / AMIA Symposium. AMIA Symposium, pp. 489–493.

Rayan, N. (2019). Framework for Analysis and Detection of Fraud in Health Insurance, in 2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS). IEEE, pp. 47–56. doi: 10.1109/CCIS48116.2019.9073700.

Sheshasaayee, A. & Thomas, S. S. (2018). A Purview of the Impact of Supervised Learning Methodologies on Health Insurance Fraud Detection. Springer Singapore. doi: 10.1007/978-981-10-7512-4.

Tanaka, F. H. K. dos S. & Aranha, C. (2019). Data Augmentation Using GANs, 2019, pp. 1–16. Available at: http://arxiv.org/abs/1904.09135.

Thornton, D., Mueller, R. M., Schoutsen, P. & Hillegersberg, J. V. (2013). Predicting Healthcare Fraud in Medicaid : A Multidimensional Data Model and Analysis Techniques for Fraud Detection, Procedia Technology, **9**, pp. 1252–1264. doi: 10.1016/j.protcy.2013.12.140.

Wang, S.,Liu, W., Wu, J., Cao, L. Meng, Q. & Kennedy, P. J. (2016). Training deep neural networks on imbalanced data sets, Proceedings of the International Joint Conference on Neural Networks, 2016-Octob(July), pp. 4368–4374. doi: 10.1109/IJCNN.2016.7727770.

Xu, L., Skoularidou, M., Cuesta-Infante, A. & Veeramachaneni, K. (2019). Modeling tabular data using conditional GAN, Advances in Neural Information Processing Systems, 32(NeurIPS).

Zhang, C., Xiao, X. & Wu, C. (2020). Medical fraud and abuse detection system based on machine learning, International Journal of Environmental Research and Public Health, 17(**19**), pp. 1–11. doi: 10.3390/ijerph17197265.

Zhu, L. Qiu, D., Ergu, D., Ying, C., Liu, K., Qiu, D., Ergu, D., Ying, L. & Kuiyi, L. (2019). A study on predicting loan default based on the random forest algorithm, Procedia Computer Science, 162(Itqm 2019), pp. 503–513. doi: 10.1016/j.procs.2019.12.017.

Submitted: 05/03/2022
Revised: 05/06/2022
Accepted: 05/06/2022
DOI: 10.48129/kjs.splml.19119