P-ISSN: 2685-8738 E-ISSN: 2685-872X

Variance: Journal of Statistics and Its Applications
Volume 7 Issue 2 | October 2025
10.30598/variancevol7iss2page219-230

EVALUATING NEARMISS AND SMOTE FOR VEHICLE INSURANCE FRAUD CLAIM CLASSIFICATION WITH A RANDOM FOREST CLASSIFIER

Feby Indriana Yusuf[1*], Endang Wahyu Handamari[2]
[1,2]Departement of Mathematics, Faculty Mathematics and Natural Sciences, Universitas Brawijaya
Jalan Veteran, Malang, 65145, East Java, Indonesia

Corresponding Author's Email: *febyindrianay@ub.ac.id*

Abstract: *This study evaluates the detection of fraudulent car insurance claims in unbalanced data by comparing two resampling techniques, namely NearMiss (undersampling) and SMOTE (oversampling), combined with Random Forest. The public dataset, consisting of 1,000 observations and 40 features, was preprocessed for missing value handling, label encoding, and min–max normalization, and split into 70% training data and 30% test data. Three scenarios were evaluated: original data (unbalanced), NearMiss, and SMOTE, using accuracy, precision, sensitivity (recall), specificity, and F1-score evaluations. The analysis results show that NearMiss provides the most balanced performance for antifraud purposes, with a sensitivity of 0.865, an F1-score of 0.667, and an accuracy of 0.787. For the original unbalanced data, the model achieved a sensitivity of 0.297 and an accuracy of 0.767. SMOTE achieved the highest precision (0.567) and accuracy (0.783), but its sensitivity was lower than that of NearMiss. These findings confirm that the selection of resampling techniques must be aligned with operational objectives: NearMiss is more appropriate when the priority is to capture as many fraud cases as possible, while SMOTE is more suitable when false positive control is prioritized.*

Keywords: *fraud detection, imbalanced data, NearMiss, SMOTE, Random Forest, vehicle insurance*

1. INTRODUCTION

Fraud claims in motor vehicle insurance remain a challenge due to the extreme class imbalance between legitimate and fraudulent claims, leading models to be biased toward the majority and resulting in higher false negatives and a higher financial risk [1]. Recent research shows that ensemble algorithms, particularly Random Forest, are consistently competitive for fraud detection in insurance, credit, and online transactions due to their resilience to noise and ability to capture nonlinear interactions [2].

To address class imbalance, pre-modelling approaches such as oversampling (e.g., SMOTE) and undersampling (e.g., NearMiss) have been extensively evaluated, including hybrid combinations. They are reported to be effective in improving minority class sensitivity [3]. However, most studies focus on non-vehicle insurance domains (e.g., credit cards or healthcare) or evaluate resampling techniques without rigorous comparative reporting on relevant operational metrics (precision–recall, F1, specificity) in vehicle insurance scenarios [4].

In contrast, vehicle claim fraud typically involves heterogeneous contextual information about the driver, vehicle, policy, and crash circumstances, multi-step, sometimes delayed reporting workflows, and distinctive physical damage patterns, which together create domain-specific challenges for modelling and evaluation [5]. Evidence on head-to-head comparisons of SMOTE vs. NearMiss for vehicle fraud claim detection using Random Forests, particularly regarding false-negative suppression, remains limited and often inconsistent across studies [6].

To close this gap, this study presents a concise, measurable comparative evaluation of SMOTE and NearMiss combined with Random Forest on imbalanced vehicle insurance claim data, with an emphasis on metrics relevant to antifraud operations (sensitivity, F1-score, and precision-recall trade-off) [7].

Submitted: 2025-11-11 *Accepted: 2025-11-30*

2. METHODOLOGY

2.1. Data Resampling Technique

Resampling in statistics is the process of creating a new sample from an existing sample. In undersampling, data from the majority class is removed, potentially leading to the loss of important information. Conversely, oversampling increases the number of minority-class samples, but this can lead to overfitting because the model is repeatedly trained on the same data [8].

2.3.1. NearMiss Undersampling

The NearMiss method begins by selecting samples based on the distance between samples in the majority and minority classes in feature space, using Euclidean Distance. There are three approaches to sample selection: finding samples with the smallest average distance to the nearest sample from the minority class, selecting samples that have the smallest average distance to the furthest sample from the minority class, and keeping the nearest neighbors of each minority class sample, where samples from the majority class are selected based on the smallest average distance between those samples and their nearest neighbors [9].

2.3.2. Synthetic Minority Oversampling Technique (SMOTE)

SMOTE is an oversampling technique that creates synthetic samples for minority classes by interpolating between samples and their nearest neighbors. The SMOTE process consists of three stages, namely selecting samples randomly from the minority class, selecting samples from the nearest neighbors, and generating new samples through linear interpolation between the two samples to estimate the position of a point based on the two existing points by drawing a linear line between the points [8].

2.2. Random Forest

Random Forest is a method developed from Decision Trees that combines multiple decision trees. Each decision tree is trained on individual samples, and each attribute is split based on a random subset of selected attributes. This method performs effective feature selection, thereby improving the performance of classification models, especially on large, complex datasets [10]. Random Forest uses a supervised machine learning algorithm that repeatedly applies the Decision Tree concept to form a forest [11]. This collection of decision trees classifies data into specified classes. In addition, combining predictions from various Decision Trees to build classifier classes results in increased accuracy in generating attributes for each node, even when the process is performed randomly [12]. The Random Forest formation process is shown in Figure 1.

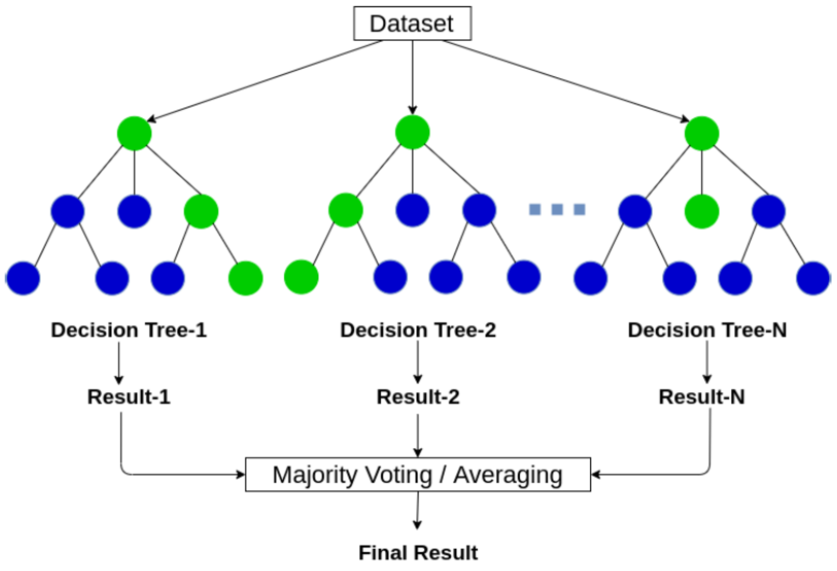

Figure 1. The Random Forest formation process

2.3. Evaluation Method

In this study, resampling is used as a pre-processing strategy to mitigate class imbalance before training the Random Forest classifier. We implement one undersampling method (NearMiss, which selectively retains majority-class observations that are closest to the minority class) and one oversampling method (SMOTE, which generates synthetic minority instances by interpolating between neighboring minority samples rather than simply duplicating existing observations) [13]. These techniques are selected because prior work has shown that appropriate resampling can substantially improve minority-class recognition in imbalanced learning problems [14]. We then compare three scenarios no resampling, NearMiss, and SMOTE using confusion-matrix-based metrics, namely accuracy, sensitivity (recall), specificity, precision, and F1-score, to assess their impact on vehicle insurance fraud claim classification [1], [3].

2.3.1. Confusion Matrix

A confusion matrix is a commonly used method for calculating accuracy [15]. In a confusion matrix, categories are used to assess whether data is True Negative (TN), True Positive (TP), False Negative (FN), or False Positive (FP).

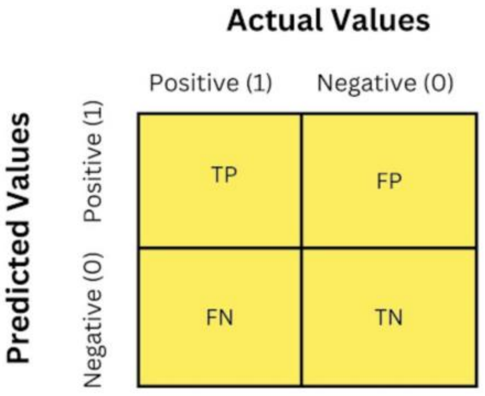

Figure 2. Confusion Matrix Model

2.3.2. Accuracy, Recall, Precision, Sensitivity, Specificity, and F1-Score

The confusion matrix values obtained will be used to evaluate the model's performance using four metrics: accuracy, recall, precision, and F1-score. Accuracy measures the closeness of the system's predictions to human predictions [16]. Recall shows the ratio of true positive predictions to total positive data, while precision compares true positive predictions to all positive prediction results [17]. Sensitivity measures the ability to detect abnormal images, while specificity measures the ability to detect normal images. F1-score calculates the weighted average of recall and precision. Thus, it can be formulated as:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$FI - Score = 2 \times \frac{(Recall \times Precision)}{(Recall + Precision)}$$

2.4. Material and Data

This study uses an insurance claim dataset available on the GitHub repository, namely Insurance Claim Fraud Detection [5]. This dataset consists of 1000 rows and 40 features covering various information related to vehicle insurance claims used to detect fraudulent claims. A detailed description of the features is shown in Table 1.

Table 1 Description of Vehicle Insurance Claim Dataset Features

Feature	Description
months_as_customer	Number of months as a policyholder.
age	Policyholder's age.
policy_number	Insurance policy number.
policy_bind_date	Policy inception/bind date.
policy_state	State where the policy was issued.
policy_csl	Policy liability limit (CSL).
policy_deductable	Policy deductible amount.
policy_annual_premium	Annual policy premium.
umbrella_limit	Umbrella coverage limit.
insured_zip	Policyholder's ZIP code.
insured_sex	Policyholder's gender.
insured_education_level	Policyholder's education level.
insured_occupation	Policyholder's occupation.
insured_hobbies	Policyholder's hobbies.
insured_relationship	Relationship of policyholder to insured.
capital-gains	Reported capital gains.
capital-loss	Reported capital losses.
incident_date	Date of the incident.
incident_type	Type of incident claimed.
incident_severity	Incident severity level.
authorities_contacted	Authorities contacted.
incident_state	State where the incident occurred.
incident_city	City where the incident occurred.
incident_location	Incident location.
incident_hour_of_the_day	Hour of day of the incident.
number_of_vehicles_involved	Number of vehicles involved.
property_damage	Property damage indicator (Y: Yes, N: No).
bodily_injuries	Number of bodily injuries.
witnesses	Number of witnesses.
police_report_available	Police report availability (Y: Yes, N: No).
total_claim_amount	Total amount of claim filed.
injury_claim	Injury claim amount.
property_claim	Property claim amount.
vehicle_claim	Vehicle claim amount.
auto_make	Vehicle make.
auto_model	Vehicle model.
auto_year	Vehicle model year.
fraud_reported	Fraud indicator (Y: Yes, N: No).

2.5. Research Method

The data will be analyzed using the Random Forest algorithm. The analysis will be carried out in accordance with the stages of machine learning, including:

1) Checking the data type and analyzing the distribution of the number of claims in the dataset.
2) Checking and handling missing values so that the dataset used is complete and accurate.
3) Converting categorical data into numerical format using label encoding techniques.
4) Performing min-max normalization on the numerical data.
5) Separating the dataset into X variables for independent features and Y variables for target features.
6) Dividing the dataset into two subsets: 70% for training data and 30% for testing data.
7) Applying the resampling method to variables X and Y.
8) Developing three modeling scenarios, namely unbalanced data, NearMiss, and SMOTE.
9) Performing classification using the Random Forest algorithm on the resampled training data.
10) Predict labels for the test data using the trained model.
11) Compile a confusion matrix to determine the accuracy of the predictions made by the trained model.
12) Calculate evaluation metrics such as accuracy, precision, recall, and F1-Score.
13) Determine the best model by comparing the performance of the two resampling techniques.

3. RESULTS AND DISCUSSION

3.1. Data Preprocessing

Preprocessing began with reviewing the data structure: of the total 40 columns, there were 19 numeric columns (17 integers and 2 floats), 21 categorical columns in text form, and one date column that was still in text form. Descriptive statistics showed significant variations in capital_gains and capital_loss, which required special attention because they potentially contained outliers; after inspecting their distributions, these extreme values were retained as they were considered to represent genuine high-gain or high-loss cases rather than data errors. After that, data quality was improved by removing column c39, which contained all missing values, and by filling in the blanks in property_damage, police_report_available, collision_type, and authorities_contacted. Numeric features were imputed with the mean, while categorical features were imputed with the mode to complete the data and improve consistency.

In order to be processed by the algorithm, all categorical columns were converted to numbers using label encoding, and the numerical columns were standardized to 0–1 using min–max normalization. The data changes can be seen from Table 2 to Table 3. After the data was cleaned and scaled, the features were defined as X (independent) and the fraud_reported label as Y (target). All records were then split into 70% training and 30% test data. This series of steps produces a clean, complete, encoded, and normalized dataset, clearly organized as X–Y with a training and testing split, ready for use in the classification modelling stage.

Table 2 Dataset Before Normalization

insured_zip	capital_gains	capital_loss	…	total_claim	injury_claim	property_claim
466132	53300	0	…	71610	6510	13020
468176	0	0	…	5070	780	780
430632	35100	0	…	34650	7700	3850
608117	48900	-62400	…	63400	6340	6340
610706	66000	-46000	…	6500	1300	650

Tabel 3 Dataset After Normalization

insured_zip	capital_gains	capital_loss	…	total_claim	injury_claim	property_claim
0.188769	0.530348	1	…	0.622801	0.303497	0.550063
0.199478	0	1	…	0.043285	0.036364	0.032953
0.002766	0.349254	1	…	0.300906	0.358974	0.162653
0.932699	0.486567	0.438344	…	0.551298	0.295571	0.267850
0.946264	0.656716	0.585959	…	0.055739	0.060606	0.027461

3.2. Applying the Resampling Method to Variables X and Y

The resampling method creates new samples from existing ones. This process aims to address data imbalance to improve the model's accuracy. The resampling techniques used in this study are undersampling and oversampling. The undersampling technique is performed using the NearMiss method, which reduces the number of samples in the majority class. Meanwhile, the oversampling technique is performed using the SMOTE method, which adds samples to the minority class. Both techniques aim to balance the distribution between the majority and minority classes.

3.3. Developing Modeling Scenarios (Unbalanced, NearMiss, and SMOTE)

In this process, the modelling scenarios are divided into three based on the previous resampling process. The three scenarios are unbalanced data, NearMiss, and SMOTE. Unbalanced data refers to the raw data before the resampling process, while the other two data sets are the data after the resampling process. This process is carried out so that the algorithm can be applied to each scenario, so that the effectiveness of the resampling method can be determined by testing the accuracy of the classification of the three scenarios.

3.4. Random Forest Classification on Training Data and Prediction on Testing Data

The classification process was carried out by comparing three data scenarios, namely unbalanced, NearMiss undersampling, and SMOTE oversampling, to assess the effectiveness of handling class imbalance in fraud claim detection. In each scenario, the same 70/30 train/test split was used, and a Random Forest model with 100 decision trees was trained on the scenario-specific training data and then evaluated on the original test data to ensure a fair comparison. NearMiss reduces the majority class samples to achieve a balanced distribution and increase sensitivity to the minority class, while SMOTE adds synthetic samples to the minority class to enrich the information without removing the majority data. All modelling and predictions were performed in Google Colab using Python. The performance of the models from each scenario was then compared to determine the most effective resampling approach for improving fraud claim detection capabilities.

3.5. Model Evaluation

3.5.1. Confusion Matrix

Figure 3 shows the confusion matrix for the original data with the following identification results:

1) True Positive (TP): the model successfully identified 22 transactions as fraudulent correctly;
2) True Negative (TN): the model correctly identified 208 transactions as not fraudulent;
3) False Positive (FP): the model incorrectly identified 18 transactions that were not actually fraudulent as fraudulent;
4) False Negative (FN): the model failed to identify 52 transactions that were actually fraudulent.

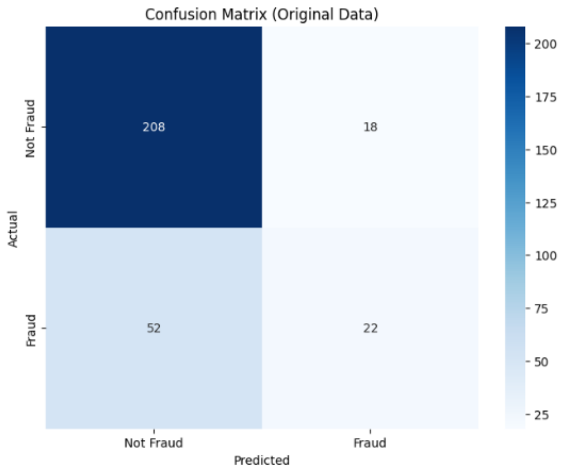

Figure 3. Confusion Matrix of Original Data

One class, namely fraud, has much less data than the other classes, which can affect the model's performance. The model tends to be more biased towards the majority class. This can be corrected by oversampling, which is adding more data to the minority class (fraud) by duplicating or generating synthetic data, and undersampling, which is reducing the amount of data in the majority class (non-fraud).

Figure 4 shows the confusion matrix resulting from model evaluation after class imbalance was handled using the NearMiss technique. This technique aims to reduce the amount of data in the majority class (non-fraud) so that the data distribution becomes more balanced.

1) True Positive (TP): the model successfully identified 64 transactions as fraud correctly;
2) True Negative (TN): the model correctly identified 172 transactions as non-fraudulent;
3) False Positive (FP): the model incorrectly identified 54 transactions that were actually non-fraudulent as fraudulent;
4) False Negative (FN): the model failed to identify 10 transactions that were actually fraudulent.

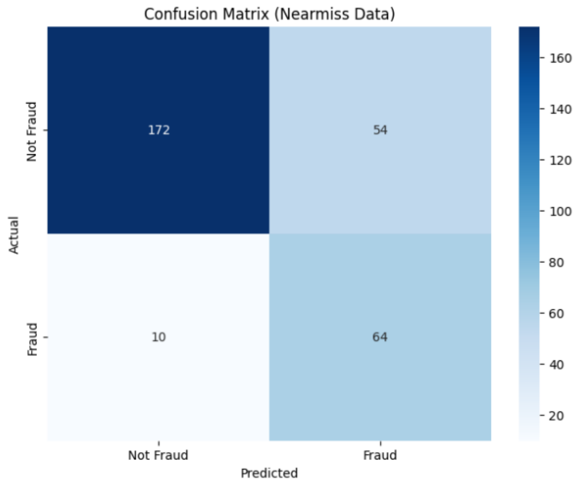

Figure 4. Confusion Matrix of NearMiss Data

When compared to the confusion matrix before class imbalance handling, there are several changes, which are:

1) Increase in true positives: the number of fraud transactions successfully identified (TP) increased significantly. This shows that the NearMiss technique successfully improved the model's ability to detect fraud transactions.

2) Increase in false positives: the number of transactions incorrectly identified as fraud (FP) also increased. This is because the NearMiss technique can reduce information in the majority class, making the model more sensitive to noise.

The confusion matrix in Figure 5 shows the results of the model evaluation after handling class imbalance using the SMOTE technique. This technique works by synthesizing new data for the minority class (fraud) so that the data distribution becomes more balanced.

1) True Positive (TP): the model successfully identified 38 transactions as fraud correctly;
2) True Negative (TN): the model correctly identified 197 transactions as not fraudulent;
3) False Positive (FP): the model incorrectly identified 29 transactions that were not actually fraudulent as fraudulent;
4) False Negative (FN): the model failed to identify 36 transactions that were actually fraudulent.

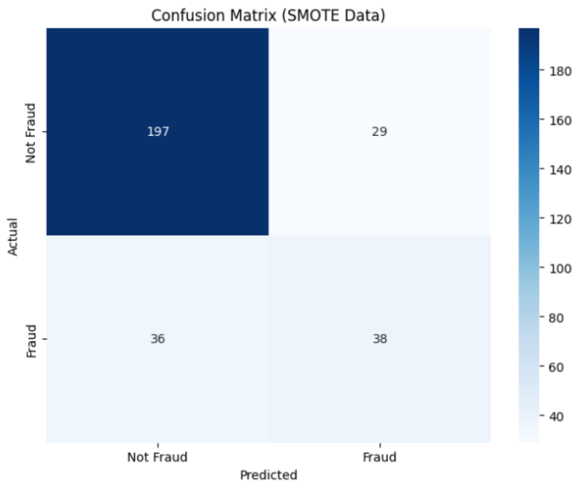

Figure 5. Confusion Matrix of SMOTE Data

When comparing the confusion matrix before class imbalance handling and after class imbalance handling using the NearMiss technique, there are several changes, namely:

1) Comparison with the confusion matrix after using the NearMiss technique, the SMOTE technique produces a slightly lower true positive value but also a lower false positive value. This shows that SMOTE is better at maintaining a balance between detection and error rates;
2) Comparison with the confusion matrix before class imbalance handling, both the NearMiss and SMOTE techniques successfully increased the true positive value, which indicates an improvement in the model's ability to detect fraudulent transactions. However, the false positive value also increased in both techniques.

3.5.2. Accuracy, Precision, Sensitivity, Specificity, and F1-score

In this study, a comparison was made between the NearMiss and SMOTE resampling methods on a dataset of car insurance fraud claims to improve classification performance with the Random Forest algorithm. Table 4 shows the results of the model performance comparison on the original data, data resampled using NearMiss, and data resampled using SMOTE.

Table 4. Results of Model Performance Comparison with Original Data, NearMiss, and SMOTE

Model	Accuracy	Precision	Sensitivity	Specificity	F1-score
Original Data	0.766667	0.550000	0.297297	0.920354	0.385965
NearMiss Data	0.786667	0.542373	0.864865	0.761062	0.666667
SMOTE Data	0.783333	0.567164	0.513514	0.871681	0.539007

1) Accuracy: all three models achieve relatively high accuracy (above 75%), indicating that, in general, they can distinguish well between legitimate and fraudulent claims;
2) Precision: the model trained on SMOTE data attains the highest precision, followed by the model on original data and then NearMiss. High precision indicates fewer errors in classifying truly non-fraudulent claims as fraud (false positives);
3) Sensitivity (Recall): the model trained with NearMiss yields the highest sensitivity, followed by SMOTE and then the original data. High sensitivity indicates better identification of truly fraudulent claims (true positives);
4) Specificity: the model trained on the original data has the highest specificity, followed by SMOTE and then NearMiss. High specificity indicates better classification of truly non-fraudulent claims as non-fraud (true negatives);
5) F1-score: the F1-score balances precision and sensitivity. The NearMiss model records the highest F1-score, indicating better overall performance in classifying both classes.

3.6. Discussion

These research findings directly address the gaps identified gaps in the detection of motor vehicle insurance fraud claims. Schrijver et al. [1] highlight that the main challenges in fraud detection include extreme class imbalance and limited empirical evidence specific to the motor vehicle insurance domain. In addition, much of the existing literature focuses on other domains such as credit cards or healthcare services. It often reports only overall accuracy, without systematically reviewing operational metrics such as sensitivity, specificity, and F1-score [3], [4]. By conducting a head-to-head evaluation of NearMiss and SMOTE on imbalanced vehicle insurance claim data using a Random Forest classifier, this study provides more targeted empirical evidence on the behavior of these two resampling techniques in an operational anti-fraud context. The results indicate that NearMiss yields the highest sensitivity and F1-score, whereas SMOTE achieves the highest precision, revealing a trade-off that, to the best of our knowledge, has not yet been described specifically for motor vehicle insurance claims.

Prior studies on resampling techniques for imbalanced data have generally used generic or non-insurance datasets and have rarely integrated an in-depth analysis of the practical consequences of resampling method selection on anti-fraud policies [6], [7], [8]. This study strengthens the literature by showing that in the context of fraud detection, operational priorities (minimizing false negatives or controlling false positives) should guide the choice of resampling approach. When the main objective is to capture as many fraudulent claims as possible, NearMiss proves to be superior because it can significantly increase sensitivity without a significant decrease in accuracy compared to the original data. Conversely, when insurance companies prioritize limiting non-fraudulent claims that are misclassified as fraud (false positives), for example, to protect customer experience and maintain investigation efficiency, SMOTE is a more suitable alternative because it delivers the highest precision. In this way, the study not only provides additional empirical evidence but also offers a more actionable framework for insurance practitioners to design machine-learning-based fraud detection systems that align with the company's operational needs.

4. CONCLUSION

Based on the comparative results between the two resampling methods evaluated, NearMiss emerges as the most effective approach for classifying automobile insurance fraud claims. NearMiss delivers superior performance in terms of sensitivity and F1-score, enabling the model to detect a larger proportion of fraudulent claims while keeping the error rate (precision) at an acceptable level. In the fraud context, this high sensitivity is significant because it improves the model's ability to identify claims that truly belong to the fraud class correctly.

By contrast, SMOTE excels in precision and is therefore more effective at reducing false positives. However, it provides a smaller improvement in sensitivity than NearMiss, suggesting that SMOTE can identify more "certain" fraud cases but may miss some claims that should be detected as fraud. Thus, although SMOTE is helpful in limiting non-fraudulent claims misclassified as fraud, NearMiss is more suitable for high-sensitivity fraud detection problems where capturing as many fraudulent claims as possible is the primary priority.

REFERENCES

[1] G. Schrijver, D. K. Sarmah, and M. El-hajj, "Automobile Insurance Fraud Detection using Data Mining: A Systematic Literature Review," *Intell. Syst. with Appl.*, vol. 21, no. February, p. 200340, 2024, doi: 10.1016/j.iswa.2024.200340.

[2] A. A. Khalil, "Enhancing Insurance Fraud Detection Accuracy with Integrated Machine Learning and Statistical Methods. 2025.," *Comput Econ*, 2025, doi: https://doi.org/10.1007/s10614-025-11074-0.

[3] P. Komsrimorakot and T. Siriborvornratanakul, "Enhancing Fraud Detection in Imbalanced Motor Insurance Datasets Using CP-SMOTE and Random Under-Sampling," *J. Big Data*, vol. 12, no. 1, 2025, doi: 10.1186/s40537-025-01217-3.

[4] A. Kushwaha, A. Yadav, A. Mishra, and P. Pandey, "Enhancing Credit Card Fraud Detection Using Deep Learning and Group Models," *Recent Trends Intell. Comput. Commun.*, no. October, pp. 788–792, 2025, doi: 10.1201/9781003593034-125.

[5] D. Mwiti, "Insurance Claim Fraud Detection Data." [Online]. Available: https://github.com/mwitiderrick/insurancedata/blob/da55cd3aed3b29377049d2543154f62858264e18/insurance_claims.csv

[6] A. Tanimoto, S. Yamada, T. Takenouchi, M. Sugiyama, and H. Kashima, "Improving Imbalanced Classification using Near-Miss Instances," *Expert Syst. Appl.*, vol. 201, no. September 2020, p. 117130, 2022, doi: 10.1016/j.eswa.2022.117130.

[7] A. Elsobky, A. Keshk, and M. Malhat, "A Comparative Study for Different Resampling Techniques for Imbalanced datasets," *IJCI. Int. J. Comput. Inf.*, vol. 10, no. 3, pp. 147–156, 2023, doi: 10.21608/ijci.2023.236287.1136.

[8] A. Nurhopipah and C. Magnolia, "Perbandingan Metode Resampling Pada Imbalanced Dataset Untuk Klasifikasi Komentar Program Mbkm," *J. Publ. Ilmu Komput. dan Multimed.*, vol. 2, no. 1, pp. 9–22, 2023, doi: 10.55606/jupikom.v2i1.862.

[9] N. M. Mqadi, N. Naicker, and T. Adeliyi, "Solving Misclassification of the Credit Card Imbalance Problem Using near Miss," *Math. Probl. Eng.*, vol. 2021, no. September 2023, 2021, doi: 10.1155/2021/7194728.

[10] Marlina Haiza, Elmayati, Zulius Antoni, and Wijaya Harma Oktafia Lingga, "Penerapan Algoritma Random Forest Dalam Klasifikasi Penjurusan Di SMA Negeri Tugumulyo," *Penerapan Kecerdasan Buatan*, vol. 4, no. 2, pp. 138–143, 2023.

[11] S. Saadah and H. Salsabila, "Prediksi Harga Bitcoin Menggunakan Metode Random Forest," *J. Komput. Terap.*, vol. 7, no. 1, pp. 24–32, 2021, doi: 10.35143/jkt.v7i1.4618.

[12] Suci Amaliah, M. Nusrang, and A. Aswi, "Penerapan Metode Random Forest Untuk Klasifikasi Varian Minuman Kopi di Kedai Kopi Konijiwa Bantaeng," *VARIANSI J. Stat. Its Appl. Teach. Res.*, vol. 4, no. 3, pp. 121–127, 2022, doi: 10.35580/variansiunm31.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique Nitesh," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[14] H. He and E. A. Garcia, "Learning from Imbalanced Data," *Knowl. Data Eng. IEEE Trans.*, vol. 21, pp. 1263–1284, Oct. 2009, doi: 10.1109/TKDE.2008.239.

[15] A. D. Adhi Putra, "Analisis Sentimen pada Ulasan pengguna Aplikasi Bibit Dan Bareksa dengan Algoritma KNN," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 8, no. 2, pp. 636–646, 2021, doi: 10.35957/jatisi.v8i2.962.

[16] M. Azhari, Z. Situmorang, and R. Rosnelly, “Perbandingan Akurasi, Recall, dan Presisi Klasifikasi pada Algoritma C4.5, Random Forest, SVM dan Naive Bayes,” *J. Media Inform. Budidarma*, vol. 5, no. 2, p. 640, 2021, doi: 10.30865/mib.v5i2.2937.

[17] A. Muhaimin, M. Amin Hariyadi, and M. I. Imamudin, “Klasifikasi Prestasi Akademik Siswa Berdasarkan Nilai Rapor dan Kedisiplinan dengan Metode K-Nearest Neighbor,” *J. Ilmu Komput. dan Sist. Inf.*, vol. 7, no. 1, pp. 193–202, 2024, doi: 10.55338/jikomsi.v7i1.2865.