Vehicle Insurance Claim Fraud Detection Analysis using Machine Learning

Maryam Jafari Mesrdashti, DePaul University, School of Computing, MJAFARIM@depaul.edu
Hema Sri Harsha Thota, DePaul University, School of Computing, hthota@depaul.edu
Lakshmi Sowjanya Gangumolu, DePaul University, School of Computing, lgangumo@depaul.edu
Shaarif Anas Mohammed, DePaul University, School of Computing, SMOHAM87@depaul.edu

ABSTRACT

This study explores machine learning techniques for detecting fraudulent vehicle insurance claims using Kaggle's "Vehicle Insurance Claim Fraud Detection" dataset. By analyzing attributes related to policies, vehicles, incidents, and claims, we addressed class imbalance through oversampling methods like SMOTE and ADASYN. Models such as logistic regression, decision trees, and ensemble classifiers were assessed, revealing the potential of machine learning to improve fraud detection, operational efficiency, and financial stability.

KEYWORDS: Insurance fraud, Vehicle claims, Machine learning, Fraud detection, Imbalanced Data

INTRODUCTION

Insurance fraud causes billions of dollars in losses each year, leading to higher premiums for customers and financial strain on insurers. In vehicle insurance, fraud often involves false claims or exaggerated damages, which increase claim payouts and complicate risk management. Detecting fraud early is critical to reducing costs, maintaining fair premiums, and improving resource allocation.

Machine learning offers a powerful approach to fraud detection by analyzing complex patterns in large datasets. Unlike traditional rule-based systems, machine learning models adapt to changing fraud schemes, providing a more efficient and scalable solution. This study uses a Kaggle dataset containing detailed information about policies, vehicles, driver history, and incidents to build models that detect fraudulent claims.

Significance of the Study

Fraudulent claims hurt both insurers and customers, making effective fraud detection essential. This study explores how machine learning can improve fraud detection, reduce costs, and support fairer insurance practices.

Research Questions and Hypotheses

Key questions include:

1. How effective are machine learning models for detecting fraudulent claims?
2. How do oversampling techniques like SMOTE and ADASYN help address data imbalance?
3. Which models offer the best performance based on accuracy, precision, recall, and F1 score?

We hypothesize that:

1. Machine learning models, especially ensemble methods, will outperform traditional approaches.

2. Oversampling techniques will improve model performance by addressing class imbalance.
3. Using detailed features like incident severity and driver history will enhance fraud detection.

This study evaluates various models to identify the best approaches for detecting fraudulent claims effectively and efficiently.

LITERATURE REVIEW

Fraud Detection in Insurance

Machine learning has transformed fraud detection by replacing manual inspections and basic statistical techniques.

- Baran and Rola [1] focused on predicting motor insurance claims but did not directly tackle fraud detection, missing the challenge of identifying fraudulent claims.
- Muranda, Ali, and Shongwe [2] and Gangadhar et al. [4] used advanced methods like variational autoencoders for fraud detection in imbalanced datasets. While innovative, these approaches are computationally intensive and less feasible for smaller organizations.
- Duval et al. [3] used anomaly detection to extract features, but their approach was better suited for detecting unusual patterns rather than specifically identifying fraud.

Handling Imbalanced Data

Managing imbalanced datasets is crucial for fraud detection:

- Muranda et al. [2] and Urunkar et al. [6] applied traditional oversampling and undersampling techniques but did not leverage advanced algorithms like LightGBM and CatBoost, which better handle imbalance.
- Popular methods like SMOTE and ADASYN have not been fully explored in combination with modern machine learning models. This study compares these techniques with various algorithms to improve fraud detection.

Machine Learning Models in Fraud Detection

Tree-based models, like random forests and gradient boosting, are widely used for their effectiveness and interpretability.

- Zhang [7] and Owolabi et al. [5] highlighted their strengths but lacked insights into their adaptability across different datasets.
- Deep learning models, such as variational autoencoders used by Gangadhar et al. [4], are powerful but prone to overfitting and require significant computational resources.
- CatBoost, designed for handling categorical data, remains underused in fraud detection. This study evaluates its potential alongside LightGBM for identifying fraudulent claims.

Gaps Addressed in This Study

This study addresses several gaps:

1. Comprehensive Model Evaluation: By comparing logistic regression with advanced methods like LightGBM, CatBoost, and gradient boosting, this study offers a balanced perspective on both simplicity and complexity in fraud detection.
2. Focus on Imbalanced Data: By combining SMOTE, ADASYN, and undersampling with modern algorithms, this research explores new ways to tackle class imbalance in fraud detection.

METHODS

Data Overview:

The dataset comprises 15,420 records and 33 attributes covering:

- **Policy Details:** Policy number, premium, tenure.
- **Vehicle Information:** Age, type, and fuel type.
- **Driver Information:** License details, age, experience.
- **Incident Information:** Incident date, location, and cause.
- **Claim Information**: Date, amount, and a binary fraud indicator.

Data preprocessing:

<u>1. Data Cleaning</u>

- Handling Missing Values:
    - Checked for missing values in the dataset.
    - Found that there were no missing values in the dataset after performing this check.
    - For the 'Age' column, observed that some entries had a value of 0, which is unrealistic. These entries were replaced with NaN values using data['Age'] = data['Age'].replace(0, float('nan')).
    - The missing values in the 'Age' column were filled with the median value for non-fraudulent cases (fraud = 0) to ensure the integrity of the dataset.
    - Used data['Age'] = data['Age'].fillna(median_age_non_fraud) to fill the NaN values in the 'Age' column.

- Removing Unnecessary Columns: Dropped columns that were not relevant for the analysis such as PolicyNumber and RepNumber.

- Removing Duplicates: Checked for and removed duplicate rows to ensure the dataset contains only unique rows.

<u>2. Encoding Categorical Variables</u>

- **Label Encoding**: Categorical variables were encoded into numerical format for machine learning models using the LabelEncoder from sklearn.preprocessing. The following columns were encoded: Make, MaritalStatus, VehicleCategory, and BasePolicy.

- **Handling Numerical Columns**: Numerical columns (such as Age, DriverRating, and Deductible) were retained as is, with no further transformations.

Model Training and Validation

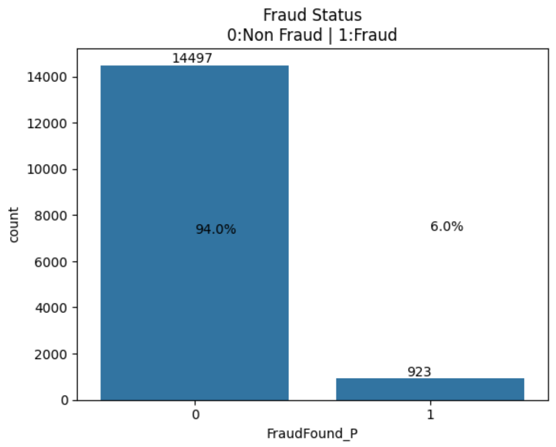
1. **Training and Testing Split:** The data was split into 80% training and 20% testing sets using stratified sampling to preserve the distribution of the target variable (fraud vs. non-fraud).
2. **Model Selection:** Multiple models were trained and evaluated, including:
   - Logistic Regression, Decision Trees, Random Forest, Gradient Boosting, Neural Networks, CatBoost, and a Voting Classifier (ensemble method).
3. **Evaluation Metrics:**
   - The models were evaluated using Accuracy, Precision, Recall, F1-Score, and Mean Average Precision (MAP) to balance performance across all classes, with special emphasis on recall due to the importance of detecting fraud.
   - Confusion Matrix and Classification Report were also generated for deeper insights.

Resampling and Evaluation

As we can see from the figure 1, there is a class the class imbalance, to mitigate this various resampling techniques were applied:

1. **SMOTE (Synthetic Minority Over-sampling Technique):** SMOTE was used to generate synthetic instances of the minority class (fraudulent claims) by interpolating between existing fraudulent samples.

Figure1. Distribution of Target variable



2. **ADASYN (Adaptive Synthetic Sampling):** ADASYN was used to focus on harder-to-classify instances within the minority class, generating synthetic samples accordingly.
3. **Combined SMOTE + ADASYN:** This method combined SMOTE and ADASYN techniques to generate synthetic samples, aiming to further improve model performance.
4. **Random Under-sampling:** This method involved reducing the number of non-fraudulent cases to balance the dataset and ensure more effective training on the minority class.

Each resampling method was applied to the training set, and the models were evaluated on the resampled data to compare performance.

Model Performance Evaluation

A function for model evaluation was implemented to generate detailed performance metrics and confusion matrices for each trained model.

RESULTS

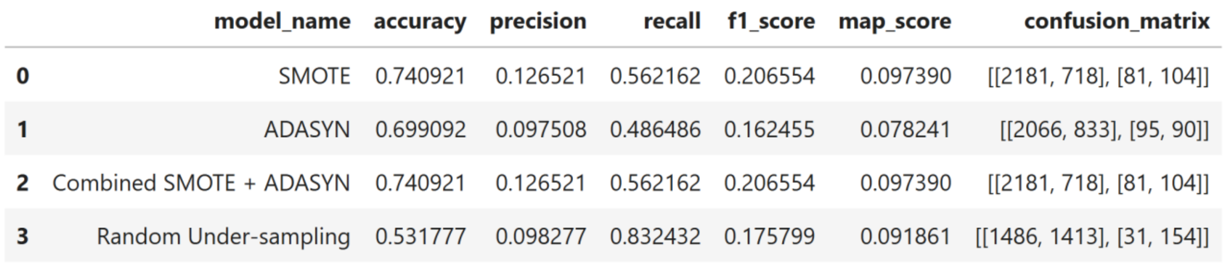
The models' performance was evaluated using key metrics: accuracy, precision, recall, F1-score, mean average precision (MAP), and confusion matrix. These metrics provide a comprehensive assessment, particularly important in fraud detection, where the primary goal is to capture the minority class (fraudulent claims) accurately. Below, we summarize the results for different approaches.

Using Oversampling and Under sampling Techniques:

1. <u>Logistic Regression model:</u>

In Logistic Regression, SMOTE and Combined SMOTE + ADASYN improved recall for fraud detection but resulted in low precision. Random Under-sampling achieved high recall (83%) but significantly reduced accuracy and precision.

Table 1. Logistic Regression model performance

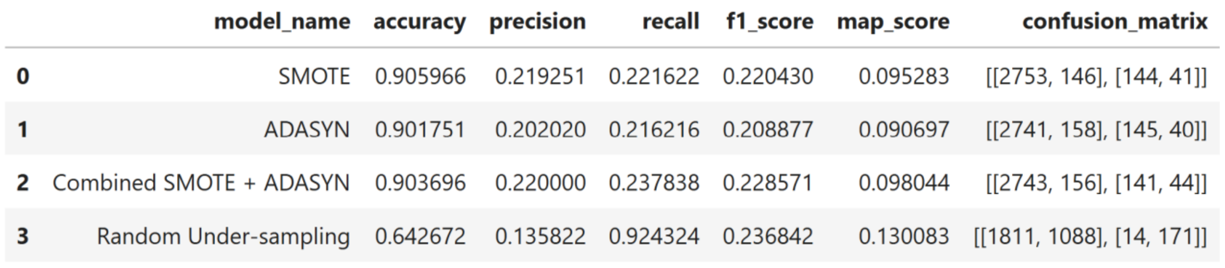


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.740921	0.126521	0.562162	0.206554	0.097390	[[2181, 718], [81, 104]]
1	ADASYN	0.699092	0.097508	0.486486	0.162455	0.078241	[[2066, 833], [95, 90]]
2	Combined SMOTE + ADASYN	0.740921	0.126521	0.562162	0.206554	0.097390	[[2181, 718], [81, 104]]
3	Random Under-sampling	0.531777	0.098277	0.832432	0.175799	0.091861	[[1486, 1413], [31, 154]]

2. <u>Random forest model:</u>

   For Random Forest, SMOTE and Combined SMOTE + ADASYN improved recall and maintained high accuracy (~90%), while Random Under-sampling achieved high recall (92%) but significantly reduced accuracy and precision.

Table 2. Random forest model performance

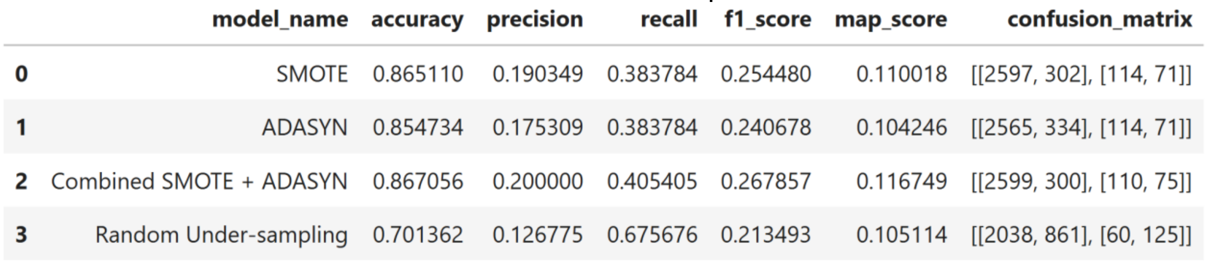


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.905966	0.219251	0.221622	0.220430	0.095283	[[2753, 146], [144, 41]]
1	ADASYN	0.901751	0.202020	0.216216	0.208877	0.090697	[[2741, 158], [145, 40]]
2	Combined SMOTE + ADASYN	0.903696	0.220000	0.237838	0.228571	0.098044	[[2743, 156], [141, 44]]
3	Random Under-sampling	0.642672	0.135822	0.924324	0.236842	0.130083	[[1811, 1088], [14, 171]]

3. Decision Tree model:

The Decision Tree model with SMOTE shows an accuracy of 0.8651 and a recall of 0.3838, which indicates a better performance in detecting fraudulent cases compared to other techniques, but with room for improvement in precision (0.1903) and F1-score (0.2545). Random under-sampling leads to a significant drop in accuracy but achieves a high recall of 0.6757, highlighting its ability to identify most fraudulent cases at the cost of precision.

Table 3. Decision tree model performance

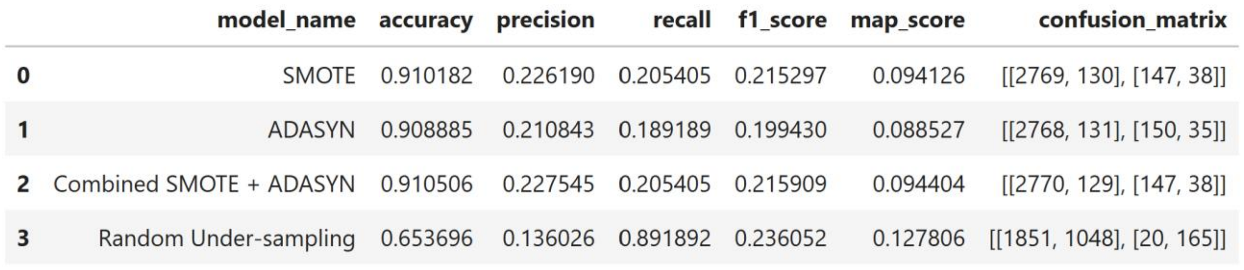


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.865110	0.190349	0.383784	0.254480	0.110018	[[2597, 302], [114, 71]]
1	ADASYN	0.854734	0.175309	0.383784	0.240678	0.104246	[[2565, 334], [114, 71]]
2	Combined SMOTE + ADASYN	0.867056	0.200000	0.405405	0.267857	0.116749	[[2599, 300], [110, 75]]
3	Random Under-sampling	0.701362	0.126775	0.675676	0.213493	0.105114	[[2038, 861], [60, 125]]

4. LightGBM model:

The **Extra Trees** model with **SMOTE** achieves high accuracy (0.9102), but precision (0.2262) and recall (0.2054) indicate a modest ability to detect fraud. Random under-sampling, while increasing recall to 0.8919, significantly lowers accuracy (0.6537) and precision, showing that it identifies most fraudulent cases at the cost of overall model performance. The combination of SMOTE and ADASYN marginally improves the model's performance without significant gains over SMOTE alone.

Table 4. LightGBM model performance

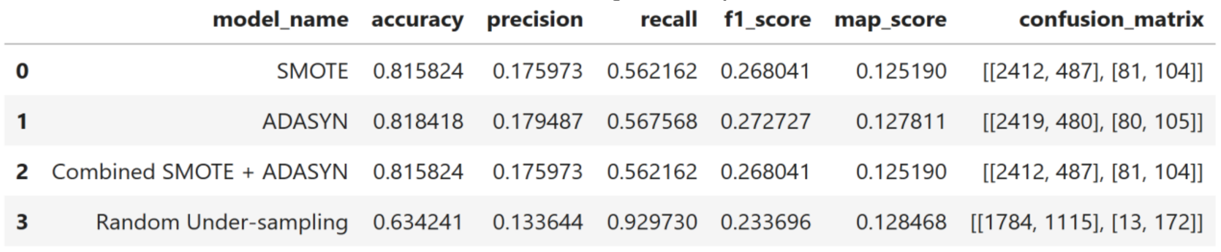


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.910182	0.226190	0.205405	0.215297	0.094126	[[2769, 130], [147, 38]]
1	ADASYN	0.908885	0.210843	0.189189	0.199430	0.088527	[[2768, 131], [150, 35]]
2	Combined SMOTE + ADASYN	0.910506	0.227545	0.205405	0.215909	0.094404	[[2770, 129], [147, 38]]
3	Random Under-sampling	0.653696	0.136026	0.891892	0.236052	0.127806	[[1851, 1048], [20, 165]]

5. Gradient Boosting model:

The Gradient Boosting model with SMOTE achieves an accuracy of 81.6%, with a recall of 56.2% for detecting fraud, but low precision (17.6%). Using ADASYN slightly improves recall to 56.8%, while random under-sampling increases recall to 92.97%, but lowers accuracy to 63.4% and precision, showing a trade-off between detecting fraud and model performance.

Table 5. Gradient Boosting model performance

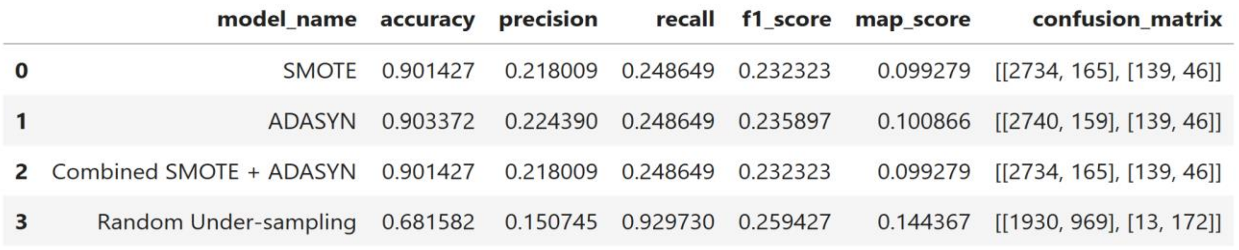


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.815824	0.175973	0.562162	0.268041	0.125190	[[2412, 487], [81, 104]]
1	ADASYN	0.818418	0.179487	0.567568	0.272727	0.127811	[[2419, 480], [80, 105]]
2	Combined SMOTE + ADASYN	0.815824	0.175973	0.562162	0.268041	0.125190	[[2412, 487], [81, 104]]
3	Random Under-sampling	0.634241	0.133644	0.929730	0.233696	0.128468	[[1784, 1115], [13, 172]]

6. CATBOOST

The CatBoost model with SMOTE achieves an accuracy of 90.1%, but its recall for fraud detection is relatively low at 24.9%. Using ADASYN shows a slight improvement in recall to 24.9%, but precision remains low (22.4%). Random under-sampling leads to a significant drop in accuracy (68.2%) but boosts recall to 92.97%, indicating a better detection of fraud at the cost of overall model performance.

Table 6. CATBOOST model performance

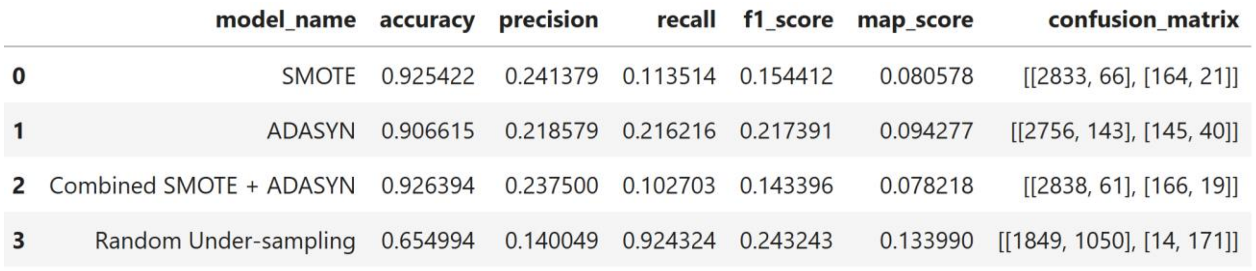


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.901427	0.218009	0.248649	0.232323	0.099279	[[2734, 165], [139, 46]]
1	ADASYN	0.903372	0.224390	0.248649	0.235897	0.100866	[[2740, 159], [139, 46]]
2	Combined SMOTE + ADASYN	0.901427	0.218009	0.248649	0.232323	0.099279	[[2734, 165], [139, 46]]
3	Random Under-sampling	0.681582	0.150745	0.929730	0.259427	0.144367	[[1930, 969], [13, 172]]

7. Stacking ensemble model

The Stacking Ensemble model with SMOTE has an accuracy of 92.5%, but recall for fraud detection is low at 11.4%. The ADASYN method improves recall slightly to 21.6%, while maintaining high accuracy (90.7%). Combined SMOTE + ADASYN shows a small accuracy increase (92.6%) but drops recall to 10.3%. Random under-sampling significantly reduces accuracy (65.5%) while improving recall to 92.4%.

Table 7. Stacking ensemble performance

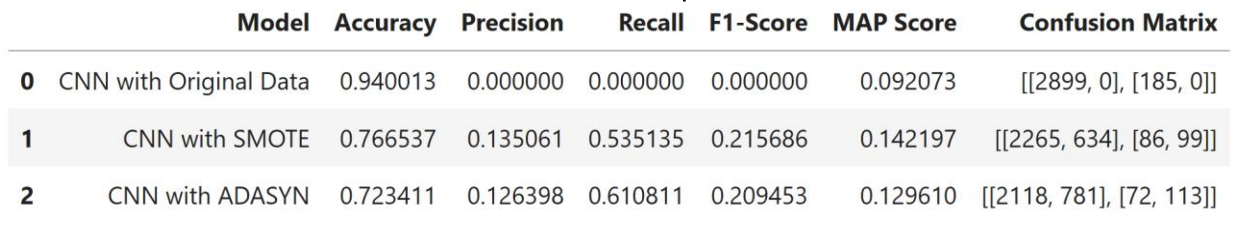


	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	SMOTE	0.925422	0.241379	0.113514	0.154412	0.080578	[[2833, 66], [164, 21]]
1	ADASYN	0.906615	0.218579	0.216216	0.217391	0.094277	[[2756, 143], [145, 40]]
2	Combined SMOTE + ADASYN	0.926394	0.237500	0.102703	0.143396	0.078218	[[2838, 61], [166, 19]]
3	Random Under-sampling	0.654994	0.140049	0.924324	0.243243	0.133990	[[1849, 1050], [14, 171]]

8. CNN model:

The CNN models with SMOTE and ADASYN improved fraud detection by increasing recall, but accuracy and precision decreased. The original model had high accuracy but failed to detect fraud. SMOTE boosted recall to 53%, and ADASYN increased it to 61%, though both had lower accuracy and precision.

Table 8. CNN model performance



	Model	Accuracy	Precision	Recall	F1-Score	MAP Score	Confusion Matrix
0	CNN with Original Data	0.940013	0.000000	0.000000	0.000000	0.092073	[[2899, 0], [185, 0]]
1	CNN with SMOTE	0.766537	0.135061	0.535135	0.215686	0.142197	[[2265, 634], [86, 99]]
2	CNN with ADASYN	0.723411	0.126398	0.610811	0.209453	0.129610	[[2118, 781], [72, 113]]

Conclusion:

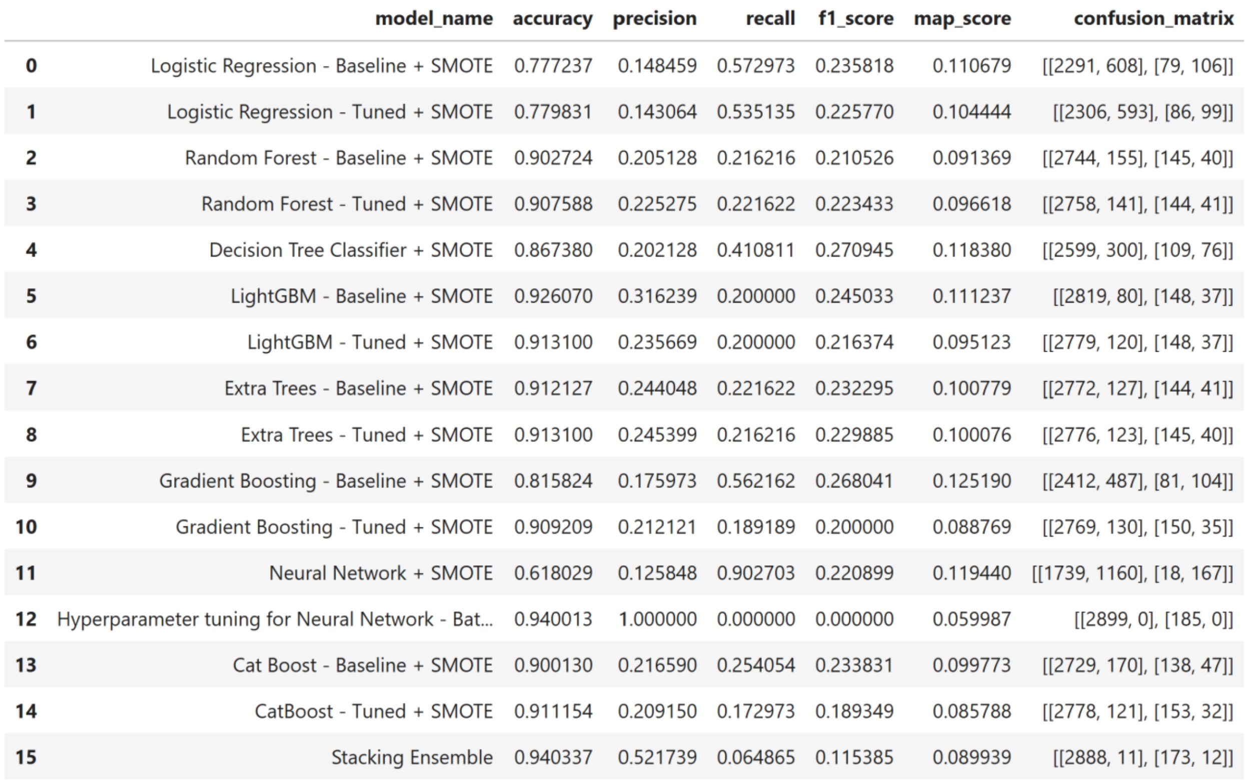
Oversampling techniques like SMOTE and ADASYN improved fraud detection by increasing recall for the minority class (e.g., Logistic Regression's recall reached 56%), but reduced precision due to more false positives. Random Under-sampling improved recall (over 80%) but lowered accuracy and precision. The combined SMOTE + ADASYN approach provided small improvements but didn't outperform individual methods. Ensemble models like Random Forest and Extra Trees handled oversampled data well, maintaining high accuracy (~90%) with modest recall gains. Overall, while oversampling helped detect fraud better, it lowered precision and accuracy, requiring careful selection of the sampling technique based on goals.

Using Only SMOTE balanced data with tuned models:

Using only SMOTE-balanced data with tuned models (Table 9), we observed significant improvements in recall for fraud detection across various models, with several tuned versions outperforming their baseline counterparts. Logistic Regression, Decision Tree, and Random Forest models showed strong performance, achieving around 90% accuracy with moderate recall. LightGBM and Extra Trees also demonstrated good results, with LightGBM performing the best at 92.61%. While the neural network model had high recall, it struggled with low accuracy and precision. The Stacking Ensemble model achieved the highest accuracy at 94.03%, but its recall and precision were less impressive. Hyperparameter tuning for Neural Networks resulted in perfect precision but failed to detect any fraudulent cases, yielding 0 recall.

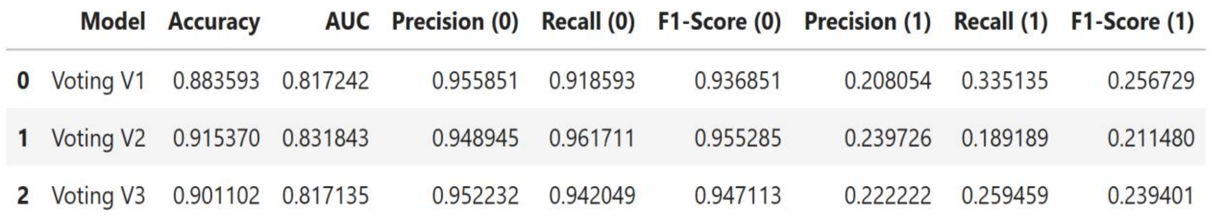
In the case of Voting Classifiers with SMOTE (Table 10), Voting V2 (Gradient Boosting, Random Forest, CatBoost) delivered the highest accuracy at 91.54% and an AUC of 0.8318, outperforming other models. Voting V1 (Logistic Regression, Random Forest, Extra Trees) performed well in terms of precision for non-fraudulent cases, though it had lower precision and recall for fraud detection. Voting V3 (Logistic Regression, LightGBM, Extra Trees) achieved a more balanced performance between precision and recall but with slightly lower overall metrics compared to V2. These results demonstrate the power of ensemble methods in improving model performance, particularly in addressing the class imbalance in fraud detection tasks.

Table 9. Models performance with SMOTE balanced data



	model_name	accuracy	precision	recall	f1_score	map_score	confusion_matrix
0	Logistic Regression - Baseline + SMOTE	0.777237	0.148459	0.572973	0.235818	0.110679	[[2291, 608], [79, 106]]
1	Logistic Regression - Tuned + SMOTE	0.779831	0.143064	0.535135	0.225770	0.104444	[[2306, 593], [86, 99]]
2	Random Forest - Baseline + SMOTE	0.902724	0.205128	0.216216	0.210526	0.091369	[[2744, 155], [145, 40]]
3	Random Forest - Tuned + SMOTE	0.907588	0.225275	0.221622	0.223433	0.096618	[[2758, 141], [144, 41]]
4	Decision Tree Classifier + SMOTE	0.867380	0.202128	0.410811	0.270945	0.118380	[[2599, 300], [109, 76]]
5	LightGBM - Baseline + SMOTE	0.926070	0.316239	0.200000	0.245033	0.111237	[[2819, 80], [148, 37]]
6	LightGBM - Tuned + SMOTE	0.913100	0.235669	0.200000	0.216374	0.095123	[[2779, 120], [148, 37]]
7	Extra Trees - Baseline + SMOTE	0.912127	0.244048	0.221622	0.232295	0.100779	[[2772, 127], [144, 41]]
8	Extra Trees - Tuned + SMOTE	0.913100	0.245399	0.216216	0.229885	0.100076	[[2776, 123], [145, 40]]
9	Gradient Boosting - Baseline + SMOTE	0.815824	0.175973	0.562162	0.268041	0.125190	[[2412, 487], [81, 104]]
10	Gradient Boosting - Tuned + SMOTE	0.909209	0.212121	0.189189	0.200000	0.088769	[[2769, 130], [150, 35]]
11	Neural Network + SMOTE	0.618029	0.125848	0.902703	0.220899	0.119440	[[1739, 1160], [18, 167]]
12	Hyperparameter tuning for Neural Network - Bat...	0.940013	1.000000	0.000000	0.000000	0.059987	[[2899, 0], [185, 0]]
13	Cat Boost - Baseline + SMOTE	0.900130	0.216590	0.254054	0.233831	0.099773	[[2729, 170], [138, 47]]
14	CatBoost - Tuned + SMOTE	0.911154	0.209150	0.172973	0.189349	0.085788	[[2778, 121], [153, 32]]
15	Stacking Ensemble	0.940337	0.521739	0.064865	0.115385	0.089939	[[2888, 11], [173, 12]]

Table 10. Voting Classifier models performance comparison



	Model	Accuracy	AUC	Precision (0)	Recall (0)	F1-Score (0)	Precision (1)	Recall (1)	F1-Score (1)
0	Voting V1	0.883593	0.817242	0.955851	0.918593	0.936851	0.208054	0.335135	0.256729
1	Voting V2	0.915370	0.831843	0.948945	0.961711	0.955285	0.239726	0.189189	0.211480
2	Voting V3	0.901102	0.817135	0.952232	0.942049	0.947113	0.222222	0.259459	0.239401

CONCLUSION

In conclusion, this study highlights the impact of different oversampling techniques and model tuning on fraud detection performance. SMOTE consistently improved recall across models, while ensemble methods like Voting V2(Gradient Boosting, Random Forest, CatBoost) demonstrated the best balance of accuracy and recall, making them well-suited for handling class imbalance. Although neural networks achieved high recall, they struggled with precision and overall accuracy. These findings underscore the importance of balancing metrics to ensure effective fraud detection. Future work will focus on refining ensemble strategies, improving sampling methods, and leveraging domain-specific insights for further performance enhancement.

AUTHOR CONTRIBUTIONS

Maryam Jafari Mesrdashti: preprocessing and modeling,Class Imbalance Techniques, documentation.

Hema Sri Harsha Thota: Modeling and Hyperparameter Tuning,Class Imbalance Techniques, and presented uptdate1.
Lakshmi Sowjanya Gangumolu: Modeling and Hyperparameter Tuning,Class Imbalance Techniques, handled documentation.
Shaarif Anas Mohammed: modeling, Class Imbalance Techniques, and presented uptdate2,3.

REFERENCES

1. Baran, S., & Rola, P., "Prediction of motor insurance claims occurrence as an imbalanced machine learning problem," arXiv preprint arXiv:2204.06109. doi: https://arxiv.org/abs/2204.06109.

2. C. Muranda, A. Ali and T. Shongwe, "Deep Learning Method for Detecting Fraudulent Motor Insurance Claims Using Unbalanced Data," *2021 62nd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, Riga, Latvia, 2021, pp. 1-5, doi: 10.1109/ITMS52826.2021.9615264.

3. Duval, F., Boucher, J.-P., & Pigeon, M., "Enhancing Claim Classification with Feature Extraction from Anomaly-Detection-Derived Routine and Peculiarity Profiles," arXiv preprint arXiv:2209.11763. doi: https://arxiv.org/abs/2209.11763.

4. Gangadhar, K. S. N. V. K., Kumar, B. A., Vivek, Y., & Ravi, V., "Chaotic Variational Auto Encoder based One Class Classifier for Insurance Fraud Detection," arXiv preprint arXiv:2212.07802. doi: https://arxiv.org/abs/2212.07802.

5. Owolabi, T., Shahra, E. Q., & Basurra, S., "Auto-Insurance Fraud Detection Using Machine Learning Classification Models," Proceedings of the Eighth International Congress on Information and Communication Technology (ICICT 2023), pp. 503–513, Springer, 2023. doi: https://link.springer.com/chapter/10.1007/978-981-99-3043-2_39.

6. Urunkar, A. Khot, R. Bhat and N. Mudegol, "Fraud Detection and Analysis for Insurance Claim using Machine Learning," 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), THIRUVANANTHAPURAM, India, 2022, pp. 406-411, doi: 10.1109/SPICES52834.2022.9774071.

7. Zhang, Z., "Fraud Detection in Vehicle Insurance Claims using Machine Learning," University of California eScholarship, 2022. Available at: https://escholarship.org/uc/item/0jx1h48j.