Algorithmic Implementation for Insurance Fraud Detection

Yaqi Chen[1,a,#], Qianshuo Feng[2,b,#], Jiexin Zhang[3,c,#], Zheng Zhao[4,d,#]

[1]School of Computer and Network Security, Chengdu University of Technology, Chengdu, Sichuan, China
[2]SJZ NO. 24 High School, Shijiazhuang, Hebei, China
[3]Department of Calligraphy, Taiyuan Normal University, Taiyuan, Shanxi, China
[4]Computer Science and Technology, Xiamen University Malaysia, Malaysia
[a]yaqichan@foxmail.com, [b]3122088431@qq.com, [c]3063814988@qq.com, [d]3158018676@qq.com
[#]Co-first author

Abstract: *In the insurance sector, spotting insurance fraud is crucial. Insurance is vital for finance and societal security. Frequent fraud causes losses to insurers and the financial system, impacting insurance companies' functioning and trust. Insurance fraud involves policyholders giving false information or creating incidents to claim compensation. This harms insurers and raises premiums for honest policyholders. To combat frauds, insurers must use methods to detect and prevent them. This study assesses popular ML algorithms like Gradient Boosting Decision Trees and XGBoost for fraud detection efficiency and verifiability. Metrics such as efficiency, recall rate, precision F1 score, and AUC score are calculated using these methods.*

Keywords: *Insurance fraud, Machine Learning, XGBoost*

1. Introduction

In the present day, the insurance industry plays a crucial role globally by providing risk coverage and economic security to individuals and businesses. However, as the insurance business continues to expand extensively, the issue of insurance fraud has gradually emerged as a significant challenge. Insurance fraud issues exist in various insurance domains. Insurance fraud activities not only pose a serious threat to the financial health of insurance companies but also undermine fair market competition and consumer trust. Consequently, the search for effective measures to address insurance fraud has become paramount.

This paper aims to utilize algorithms to implement insurance fraud detection, aiming to efficiently identify potential fraud cases and mitigate the adverse impacts of insurance fraud. To achieve this goal, the focus is on constructing high-performance predictive models, aiming to enhance the abilities of insurance companies in recognizing and preventing fraudulent activities. Data analysis occurs within milliseconds, relieving team members from the burden of manual reviews and checks that accompany each new data acquisition [1].

In previous research, machine learning and data mining techniques have been widely applied in the financial sector, particularly in the impressive success achieved in fraud detection. Given the particularity of the data types involved in this study, Python, being a popular programming language, offers a rich set of libraries and frameworks that enable developers to efficiently build and train models, perform feature engineering, and analyses data. Nian et al. [2] proposed that the decision tree model is a promising model for analysing automobile insurance fraud.

In the practical research phase, a dataset containing a substantial amount of insurance transaction information was employed, which might include potential fraud cases. To ensure data quality and completeness, data preprocessing was conducted, encompassing steps such as data cleansing, handling missing values, and feature selection. Subsequently, feature engineering was adopted to extract fraud-related features from the raw data for the model's utilization.

Numerous researchers have endeavoured to incorporate deep networks into the domain of financial fraud detection. Aleskerov et al. [3] introduced a one-layer neural network for credit card fraud detection as early as 1997. Chouiekh et al. [4] employed deep learning techniques to detect instances of mobile communication fraud. Nonetheless, due to the constraints of deep learning techniques and the unique structure of financial data, its widespread adoption within the industry has been limited in recent years.[5]

To build predictive models, two algorithms, Gradient Boosting Decision Trees (GBDT) and XGBoost, were selected. These algorithms possess advantages in handling high-dimensional and imbalanced datasets, allowing them to capture complex relationships effectively and identify potential fraud cases.

During the experimentation phase, the models were comprehensively evaluated through cross-validation and testing datasets. The experimental results indicated that the XGBoost model exhibited higher accuracy, precision, and recall in insurance fraud detection, surpassing other models in identifying fraud cases.

However, it is acknowledged that challenges persist in the field of insurance fraud detection, such as addressing imbalanced data and enhancing model generalization. Thus, future research can further explore additional feature engineering methods and model optimization strategies to enhance model performance.

The remainder of this paper is structured as follows. Commencing with the acquisition of a substantial volume of past car insurance data, the focus shifted towards preprocessing procedures, including tasks like imputing missing values. Following this, feature engineering was executed to enhance the performance of the model. Finally, an evaluation of the two models was conducted, revealing that XGBoost exhibited superior performance.

2. Methods

This section is divided into five parts. Firstly, A substantial amount of insurance data was gathered and meticulously preprocessed. A substantial amount of insurance data was gathered and meticulously preprocessed. To further optimize the model, a grid-tuning approach was employed. Lastly, after model training and refinement, an evaluation of the model was conducted (Figure 1).

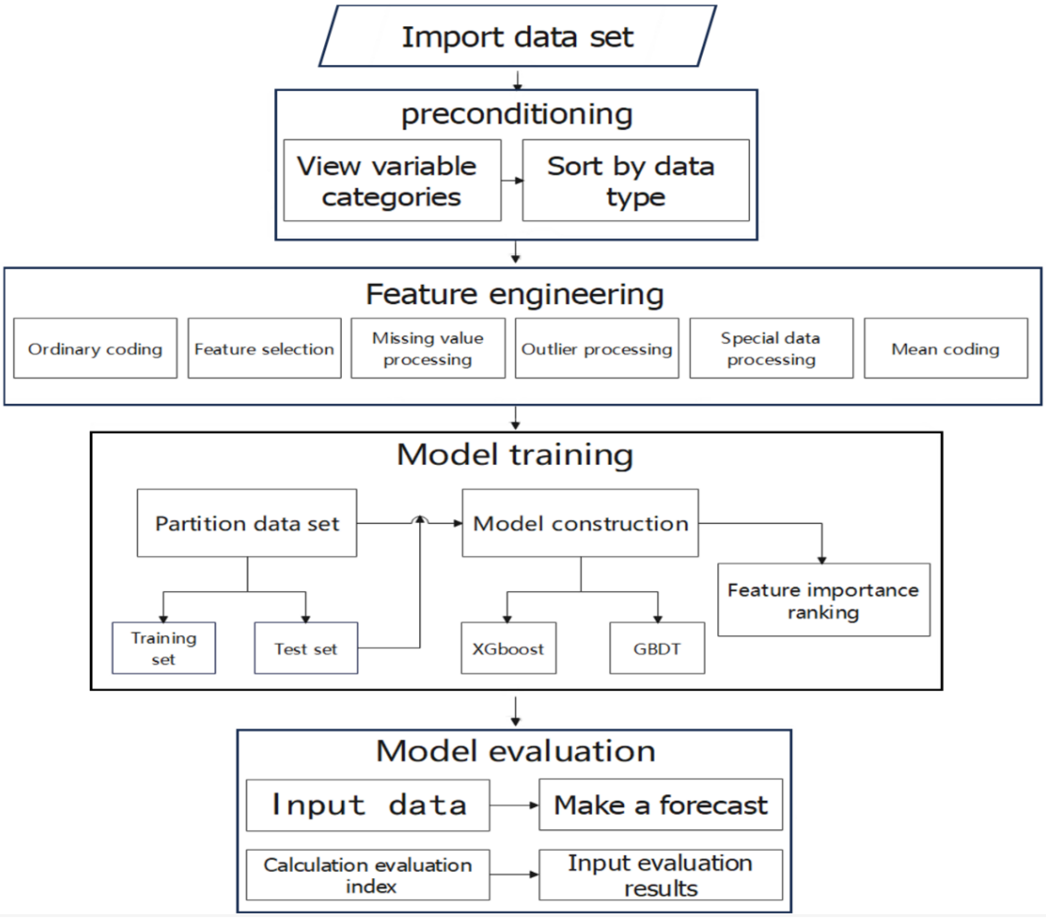


Figure 1: The procedure of the research

2.1 Data Acquisition and Understanding

A substantial amount of insurance data, including policyholder's personal information, insurance types, historical claims records, presence of fraudulent activities, and more, was initially obtained from various sources of insurance transactions. These data were collected and stored in a database, forming the foundation for subsequent analysis and modelling.

2.2 Data Preprocessing

To ensure the accuracy of the model, data preprocessing is a crucial step. The variables in the data can be categorized into several types, including int64, float64, and object. Variables of the object type require specific analysis, whereas for int64 and float64 types, methods for numerical variables analysis are commonly employed.

Numerical variables can be classified into three types: continuous variables, discrete variables, and constant variables. Distinguishing between continuous, discrete, and constant variables is highly important for data processing and analysis. During the data preprocessing stage, unorganized raw data is converted into coherent and understandable language [6]. Based on the variable type, different statistical methods, visualization techniques, and modeling approaches were chosen to better understand and analyze the data (Figure 2-3).

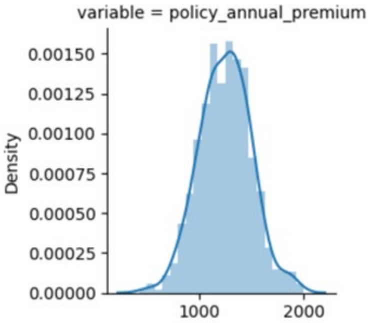


Figure 2: Visualization of annual premiums

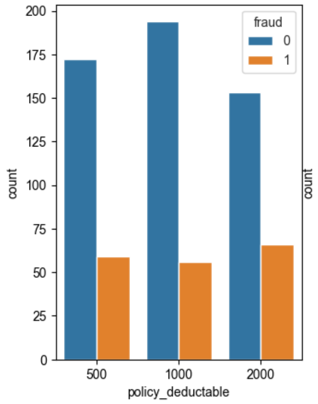
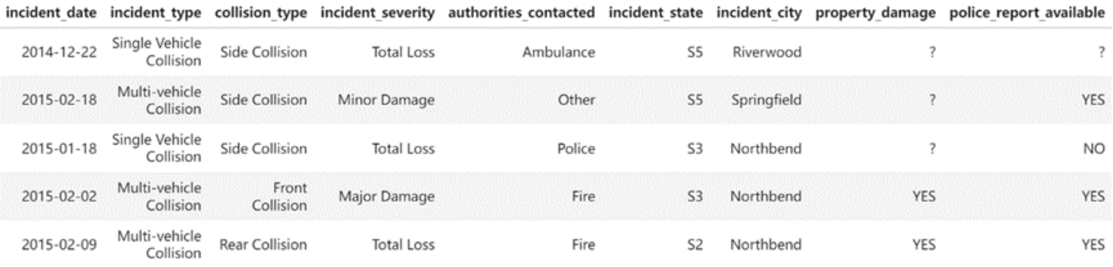


Figure 3: Visualization of policy deductible

Figure 2 and Figure 3. depict the visualizations of selected continuous and discrete variables.



incident_date	incident_type	collision_type	incident_severity	authorities_contacted	incident_state	incident_city	property_damage	police_report_available
2014-12-22	Single Vehicle Collision	Side Collision	Total Loss	Ambulance	S5	Riverwood	?	?
2015-02-18	Multi-vehicle Collision	Side Collision	Minor Damage	Other	S5	Springfield	?	YES
2015-01-18	Single Vehicle Collision	Side Collision	Total Loss	Police	S3	Northbend	?	NO
2015-02-02	Multi-vehicle Collision	Front Collision	Major Damage	Fire	S3	Northbend	YES	YES
2015-02-09	Multi-vehicle Collision	Rear Collision	Total Loss	Fire	S2	Northbend	YES	YES

Figure 4: Partial single value variables

As shown in Figure 4, the list contains missing values. Through calculations, it was found that the missing rates in the data list do not exceed 50%. Therefore, the missing values were proceeded to be filled in and treated separately as a distinct category.

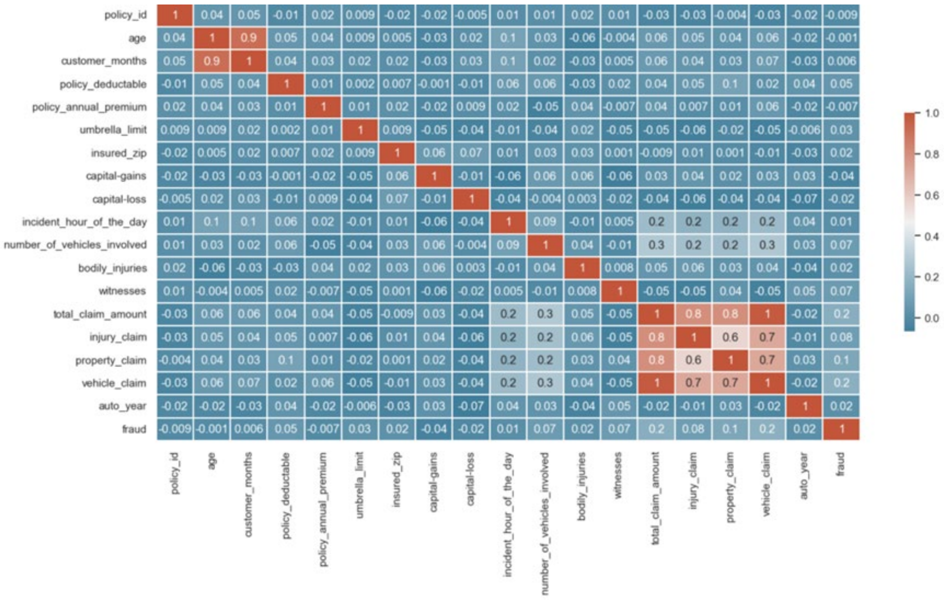


Figure 5: A heatmap displays the correlation between the data

The correlation coefficients among continuous features in the data were calculated using code and visually presented like Figure 5.

A heatmap visually represents the correlations between continuous features in the data, allowing us to easily identify strong and weak relationships between different features.

By using a heatmap, highly correlated feature pairs can be identified, which is crucial for feature selection. Highly correlated feature pairs might contain redundant information, and selecting one of them can reduce model complexity.

Heatmaps help us detect unusual correlation patterns, which can highlight potential data issues like mislabeling or data sampling problems.

2.3 Feature Engineering

Feature engineering refers to the process of modifying, transforming, and selecting raw data before constructing a machine learning model. Its aim is to extract valuable information, enhance data representation, and consequently improve the model's performance and adaptability. In machine learning, feature engineering is crucial since the model's effectiveness heavily relies on the quality of input data and the efficiency of features. Correct implementation of feature engineering can lead to more accurate and robust models. When dealing with data from specific domains, suitable feature engineering can enhance model performance and offer deeper insights for understanding and resolving problems.

2.3.1 Missing Values Processing

There are several methods for handling missing values: For discrete values, discrete values can be treated as a separate category or filled with the mode of the same column. For continuous values, continuous values can be filled with the mean or median.

2.3.2 Numerical Encoding

It was discovered that certain values within the data possess a latent logical ordering, and leaving them untreated could impact the final model's performance. Therefore, the logical order among these values was identified and they were transformed into integer values to facilitate learning and analysis by the algorithmic models. For instance, in the "incident_severity" column, there are four categories: “Total Loss”, “Major Damage”, “Minor Damage”, and “Trivial Damage”, which clearly exhibit a hierarchy of severity. The categorical labels were converted into numerical values to enhance their utility within the model.

2.3.3 Outliers Processing

While the majority of mining techniques incorporate methods to address missing or noisy data, these safeguards are often inadequate [6]. For certain outlier data, excluding them before training the model can lead to improved results. Outliers are identified using methods such as the 3σ rule or box plots.

(1) The 3σ rule, also known as the "Three Sigma Rule" or "68-95-99.7 Rule," is a statistical principle that relates to the normal distribution of data.

This rule is often used to identify outliers or unusual data points that fall significantly beyond these ranges. Data points that are more than three standard deviations away from the mean are considered potential outliers and may warrant further investigation or removal in some cases.

(2) A box plot, also known as a box-and-whisker plot, is a graphical representation used to display the distribution and spread of a dataset. It presents a visual summary of the minimum, first quartile, median (second quartile), third quartile, and maximum values of the data.

Box plots are useful for identifying the central tendency, spread, and presence of outliers in a dataset. They provide insights into the distribution of data and help in comparing distributions between different groups or categories.

2.3.4 Special Data Processing

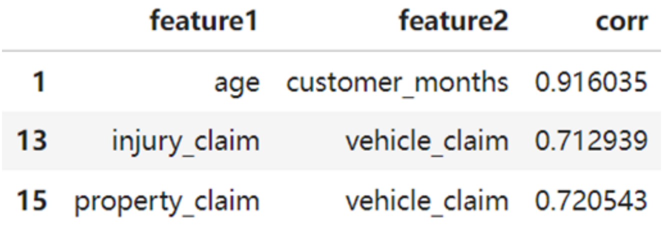
Research [7] has indicated that extracting second-level features contributes to the improvement of model performance. For instance, data types like “policy_bind_date” and “incident_date” do not hold significant meaning on their own. Hence, the difference between the incident date and the policy bind date was calculated. This difference is then extracted as a separate field to create a new feature.

Accidents might be influenced by factors like weather, so we create another new feature based on the occurrence month. However, in this case, we represent the month as a character rather than a numerical value, as we are interested in its frequency rather than its magnitude.

2.3.5 Feature Selection

Remove irrelevant fields. Eliminate fields that are evidently unrelated to the outcome, such as "id".

Explore relationships between numerical variables. A function was utilized to compute the correlation coefficients between variables and fraud. Identifying correlated variables among the numerical variables.



	feature1	feature2	corr
1	age	customer_months	0.916035
13	injury_claim	vehicle_claim	0.712939
15	property_claim	vehicle_claim	0.720543

Figure 6: Features with high correlation

As shown in Figure 6, there is a clear correlation between "age" and "customer_months". "Age" is removed (as it has low correlation with fraud), and "injury_claim" is also removed due to its low correlation with fraud.

2.3.6 Mean Encoding

For variables with multiple discrete values (more than 10), MeanEncoding is employed to encode discrete values into continuous values for easier subsequent analysis.

The Mean Encoder class implements the fundamental principles of mean encoding. It calculates the average value of the target variable corresponding to each categorical feature value, and then applies weighted averaging based on prior weights to transform categorical features into numerical features.

Basic Idea and Principle:

Mean encoding is a supervised encoding technique applicable to both classification and regression problems. For simplicity, all the following code examples are based on a classification problem.

Basic Idea of the Algorithm: Represent each k in the variable as the (estimated) probability of its corresponding target y value: $\hat{P}(target = y \mid variable = k)$

2.4 Modelling

First, the dataset is divided into a training set and a testing set. When using the function to split the data, the training set accounts for 70% of the total dataset, while the testing set accounts for 30%. The Gradient Boosting Decision Trees (GBDT) model and the XGBoost model are selected for analysis.

Gradient Boosting Decision Tree (GBDT) is a popular machine learning technique used for solving classification and regression problems. It enhances model accuracy by training a series of different decision trees in multiple rounds. In each round, it corrects errors based on the predictions of the previous round, gradually improving the model's performance. While GBDT is powerful for tackling complex problems, it's worth noting that it's sensitive to outliers and training time can be relatively long.

XGBoost is a machine learning model based on the concept of augmentation, which integrates multiple weak learners to achieve strong learning capabilities. In areas such as fraud detection, XGBoost is one of the most commonly used models, which solves the problem of class imbalance and effectively avoids the problem of overfitting in the training data [8]. This algorithm can deal with large data sets efficiently, has strong prediction performance and fast training speed. Specifically, XGBoost is an iterative computational process for decision tree classification. At step n, each learner is computed as Equation 1, where $f\ k$ is the base tree model and $x\ i$ is the input feature. Then, to measure the performance of each learner L, XGBoost computes with the loss function α and the regularization term γ. The performance was calculated by formula Equation 2.

$$\hat{y}_i = \sum_{k=1}^{n} f_k(x_i) \tag{1}$$

$$L = \sum_{i} \alpha(\hat{y}_i, y_i) + \sum\nolimits_{k} \gamma(f_k) \tag{2}$$

The regularization γ calculate using Equation 3, which aims to prevent overfitting, where T is the number of leaves in each learner, σ is the minimal loss, and w is a weight or vector score in leaves.

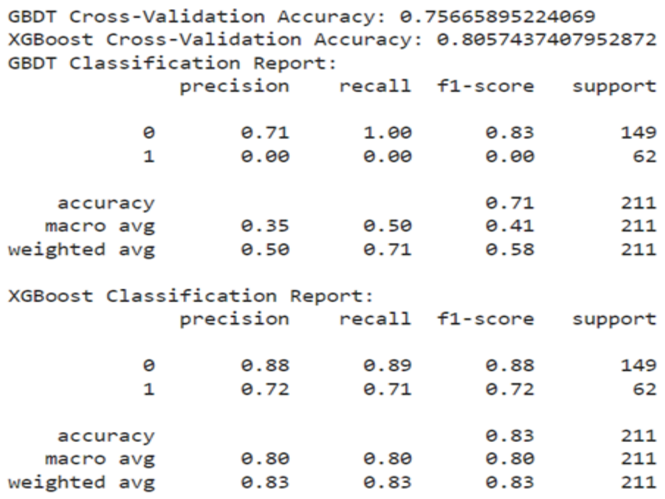
$$\gamma(f) = \sigma T + \frac{1}{2}\lambda\|w\|^2 \tag{3}$$

Hyperparaeter tuning is utilized as a method to optimize model performance. Hyperparameters are parameters that need to be manually set before model training and significantly influence the model's performance and generalization ability. By adjusting these hyperparameters, the model's performance on the validation dataset can be effectively enhanced. The most demanding aspect of building machine learning models lies in the optimization of hyperparameters [9-10]. For discovering the optimal combination of hyperparameters, methods like grid search and random search are employed. Different values for the hyperparameters are systematically tried out, and techniques like cross-validation are used to evaluate the performance of each combination. This ensures that among the numerous possible hyperparameter combinations, those values that most effectively improve model performance are identified. Hyperparameter tuning not only enhances prediction accuracy but also bolsters model stability and generalization capability, making it more suitable for new, unseen data.

3. Result and Discussion

Upon completing feature engineering and model training, a series of intriguing findings related to insurance fraud detection has been obtained by us. This section will delve into a detailed discussion of these results, analysing the model's performance, the significance of features, and potential directions for further improvement.

Model Performance Analysis: After training and cross-validation, two distinct machine learning models were employed by us: Gradient Boosting Decision Trees (GBDT) and XGBoost. Based on our experimental results, it was observed that the XGBoost model excelled in cross-validation, demonstrating higher accuracy and F1 scores that far surpassed those of the GBDT model. This might be attributed to XGBoost's superior handling of complex relationships and high-dimensional data. However, when selecting a model, other factors such as training time and resource consumption need to be taken into consideration (Figure 7).



```
GBDT Cross-Validation Accuracy: 0.75665895224069
XGBoost Cross-Validation Accuracy: 0.8057437407952872
GBDT Classification Report:
              precision    recall  f1-score   support

           0       0.71      1.00      0.83       149
           1       0.00      0.00      0.00        62

    accuracy                           0.71       211
   macro avg       0.35      0.50      0.41       211
weighted avg       0.50      0.71      0.58       211

XGBoost Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.89      0.88       149
           1       0.72      0.71      0.72        62

    accuracy                           0.83       211
   macro avg       0.80      0.80      0.80       211
weighted avg       0.83      0.83      0.83       211
```

Figure 7: Classification Report

(1) ROC (Receiver Operating Characteristic)

The ROC curve is constructed in a space where the False Positive Rate (FPR) is represented on the X-axis and the True Positive Rate (TPR) is represented on the Y-axis.

Ideal Model: An ideal model aims to maximize TPR while minimizing FPR, ensuring accurate positive predictions while minimizing misclassifications of negative samples.

(2) AUC (Area Under Curve)

By employing these evaluation criteria, the effectiveness of our model's performance in various aspects can be comprehensively assessed.

Table 1: Average AUC values for the two models

	AUC
GBDT	0.840
XGBoost	0.915

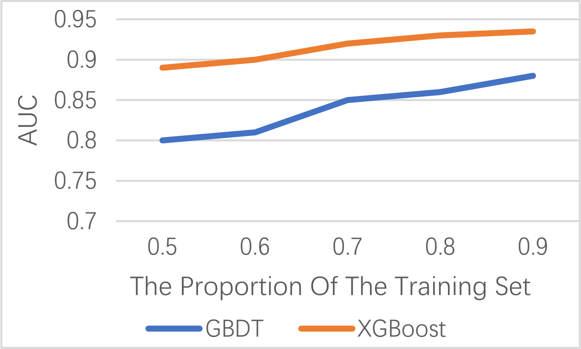


Figure 8: The variation of AUC values.

Table 1 presents the average results of multiple tests, indicating that XGBoost demonstrates higher performance in this type of problem. As shown in Figure 8, AUC values underwent changes with fluctuations in the proportion of the training set, yet the performance of XGBoost remained consistently superior to GBDT.

Feature Importance Analysis: By analyzing the feature importance within the model, it can be determined which features play a crucial role in fraud detection. It was discovered that features such as historical claims records, policyholder risk assessment, and insurance type hold higher significance in the model. This suggests that these features provide vital information about fraud risk to the model and play a pivotal role in accurate predictions. Further exploration of these features could enhance our understanding of their specific impact mechanisms in fraudulent cases.

Figure 9 represents the results calculated using functions and illustrate the importance of each feature, and such data can offer a focus direction for further research.

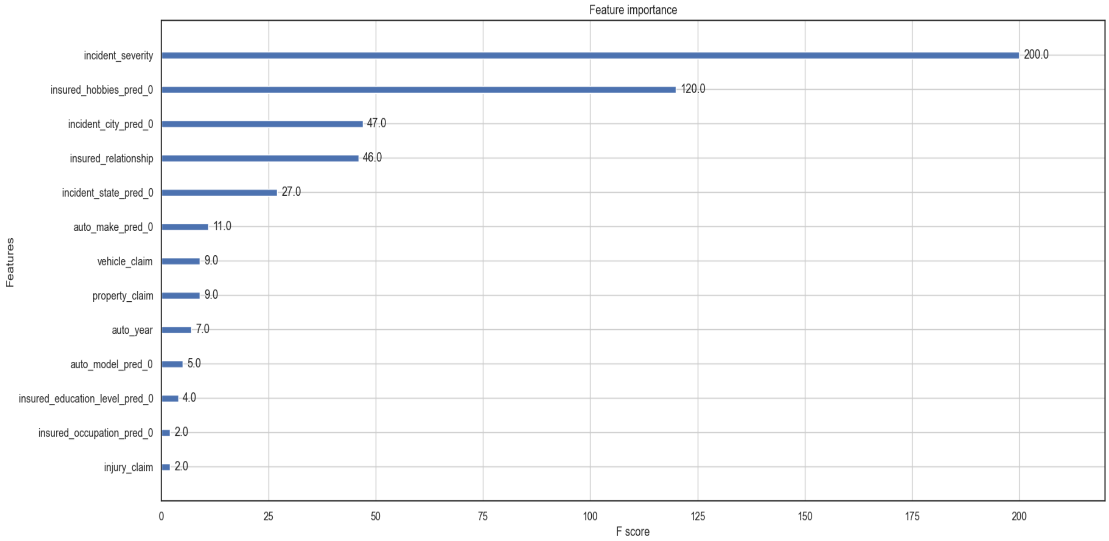


Figure 9: Feature importance

4. Conclusion

In this study, the implementation of insurance fraud detection using algorithms was delved into. Significant advancements in the field of insurance fraud detection were achieved through systematic feature engineering and machine learning model construction.

The process was initiated by extensively preprocessing the raw insurance data, encompassing data cleansing, handling missing values, and feature encoding. These steps laid a robust foundation for subsequent analysis and modelling. By gaining an in-depth understanding of the data, suitable feature engineering methods were better selected, enabling the data to be comprehended by machine learning algorithms.

Throughout the feature engineering process, crucial features related to insurance cases were extracted from the raw data, and relationships among features were explored using data visualization techniques. This aided in enhancing our comprehension of inherent data patterns and facilitated the incorporation of domain knowledge during the model training.

Through the construction of machine learning models such as Gradient Boosting Decision Trees (GBDT) and XGBoost, potential instances of insurance fraud were accurately identified by us. Through cross-validation and testing, the robustness and generalizability of the models were confirmed. The XGBoost model exhibited high accuracy, recall, and F1 scores, providing strong support for real-world applications in insurance fraud detection.

However, certain limitations in the study are also acknowledged, such as the potential for the models to exhibit bias towards dominant categories due to imbalanced datasets. Furthermore, there is room for improvement in feature engineering and model selection to further enhance model performance.

In conclusion, this research yielded significant achievements in the realm of insurance fraud detection through the utilization of Python algorithms. Our work not only enriched the field of insurance research but also holds crucial implications for practical applications. With ongoing dedication and innovation, it is confident that more precise and efficient fraud detection can be realized within the insurance industry, fostering sustainable growth for insurance companies and building customer trust.

References

[1] Dada, E. G., Bassi, J. S., Chiroma, H., Adetunmbi, A. O., & Ajibuwa, O. E. (2019). Machine learning for email spam filtering: review, approaches and open research problems. Heliyon, 5(6).
[2] Nian, K., Zhang, H., Tayal, A., Coleman, T., & Li, Y. (2016). Auto insurance fraud detection using unsupervised spectral ranking for anomaly. The Journal of Finance and Data Science, 2(1), 58-75.
[3] Aleskerov, E., Freisleben, B., & Rao, B. (1997, March). Cardwatch: A neural network based database mining system for credit card fraud detection. In Proceedings of the IEEE/IAFE 1997 computational

intelligence for financial engineering (CIFEr) (pp. 220-226). IEEE.

[4] Chouiekh, A., & Haj, E. H. I. E. (2018). Convnets for fraud detection analysis. Procedia Computer Science, 127, 133-138.

[5] Nan, Z. (2020). Deep Learning Based Approaches for Financial Fraud Detection (Master's thesis, NTNU).

[6] Dalal, S., Seth, B., Radulescu, M., Secara, C., & Tolea, C. (2022). Predicting fraud in financial payment services through optimized hyper-parameter-tuned XGBoost model. Mathematics, 10(24), 4679.

[7] Gong, J., Zhang, H., & Du, W. (2020). Research on integrated learning fraud detection method based on combination classifier fusion (THBagging): A case study on the foundational medical insurance dataset. Electronics, 9(6), 894.

[8] Lopo, J. A., & Hartomo, K. D. (2023). Evaluating Sampling Techniques for Healthcare Insurance Fraud Detection in Imbalanced Dataset. Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI), 9(2), 223-238.

[9] Rubio, J., Barucca, P., Gage, G., Arroyo, J., & Morales-Resendiz, R. (2020). Classifying payment patterns with artificial neural networks: An autoencoder approach. Latin American Journal of Central Banking, 1(1-4), 100013.

[10] Asha, R. B., & KR, S. K. (2021). Credit card fraud detection using artificial neural network. Global Transitions Proceedings, 2(1), 35-41.