Insurance Fraud Detection using Machine Learning

Manasa Madiraju
Rohit Roy Chowdhury

Abstract

Insurance fraud is a significant issue that leads to substantial financial losses for insurance companies. The objective of this project is to explore the application of machine learning techniques, specifically XGBoost and Random Forest, to detect fraudulent claims in insurance datasets. By leveraging these models, we aim to identify patterns and anomalies that indicate fraudulent activities, thus helping insurers mitigate risks and reduce fraudulent claims. This report covers the theoretical background, experimental setup, results, and conclusions drawn from the study.

Introduction

The insurance industry is vast and complex, offering diverse policies like car, rental, health, and life insurance. Sadly, insurance fraud is widespread, occurring throughout the process – from policy sales to claiming. In response, insurance companies need sophisticated tools, including data analytics and machine learning, to detect and deter these fraudulent activities. Fraud can be perpetrated by individuals, groups, policyholders, insurers, or third parties. Due to its scope and complexity, robust solutions are essential. Insurance fraud costs both insurers and policyholders billions annually, taking forms like:

- False claims
- Fake policies
- Identity theft

This project aims to develop a machine learning model that analyzes historical claims and policyholder data to detect insurance fraud. Our goal is to uncover patterns and trends that signal fraudulent behavior.

Machine learning offers a promising solution by automating and enhancing the detection process through sophisticated algorithms that can analyze large datasets and identify fraudulent patterns. This report focuses on two powerful machine learning algorithms: XGBoost and Random Forest, both of which are known for their robustness and efficiency in handling classification problems. The following sections will delve into the theoretical background, experimental setup, results, and conclusions of our study on using these models for insurance fraud detection.

Types of Insurance Fraud

1. **Hard Fraud**: Involves deliberate and planned actions to fabricate or cause a loss in order to claim insurance benefits. Examples include staging accidents, arson for profit, and faking deaths.
   - **Staging Accidents**: Individuals deliberately cause accidents and then file insurance claims for damages and injuries. This often involves multiple conspirators who act as witnesses or involved parties.
   - **Arson for Profit**: Policyholders intentionally set fire to insured property, such as homes or businesses, to claim the insurance payout. This type of fraud is particularly dangerous and costly.
   - **Faking Deaths**: In extreme cases, individuals may fake their own death or the death of a family member to claim life insurance benefits.

2. **Soft Fraud**: Occurs when a policyholder exaggerates a legitimate claim or provides false information to receive a higher payout. Examples include inflating the value of stolen items or claiming for damages that were pre-existing.
   - **Inflating Claim Amounts**: Claimants might inflate the value of items lost or damaged in an insured event to receive a higher payout than they are entitled to.
   - **Pre-existing Damages**: Claimants may include damages that occurred before the insured event in their claims to receive compensation for old damages.

Machine Learning Approaches

We have chosen two popular supervised machine learning algorithms to build our model for insurance fraud detection.

Random Forest

Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It reduces overfitting by averaging multiple trees, thus improving accuracy and robustness.

Advantages

- **High Accuracy**: Generally provides high accuracy due to the ensemble approach, which reduces the variance of the predictions.
- **Feature Importance**: Provides insights into which features are most important for predictions, helping to understand the factors influencing fraudulent behavior.
- **Robustness to Overfitting**: Averaging over many trees reduces the risk of overfitting, especially when using a large number of trees. Random Forest is also less sensitive to noisy data.

Disadvantages

- **Complexity**: Can be computationally intensive and require significant resources for training, particularly with large datasets and a high number of trees.
- **Interpretability**: While individual trees are interpretable, the ensemble of trees can be less so. The model's complexity can make it challenging to understand the decision-making process.

XGBoost

XGBoost (Extreme Gradient Boosting) is a scalable and efficient implementation of gradient boosting framework by Friedman et al. It uses a more regularized model formalization to control overfitting, which provides better performance. XGBoost is known for its speed and performance in machine learning competitions.

Advantages

- **Efficiency**: Highly optimized for speed and performance, capable of handling large datasets efficiently.
- **Handling of Missing Values**: Automatically learns the best imputation strategy from the data, making it robust to incomplete datasets.
- **Regularization**: Includes L1 and L2 regularization to control overfitting, enhancing the model's generalization capabilities.

Disadvantages

- **Complexity**: Requires careful tuning of hyperparameters to achieve optimal performance. The model's complexity can be a barrier to understanding and implementation.
- **Resource Intensive**: Can be computationally demanding, especially with large datasets. The training process may require significant memory and processing power.

Evaluation Metrics

To evaluate the performance of the machine learning models, several metrics are used:

- **Accuracy**: The proportion of correctly classified instances among the total instances. Accuracy is a straightforward metric but can be misleading in the case of imbalanced datasets, where the number of legitimate claims vastly outnumbers fraudulent claims.
- **Precision**: The proportion of positive identifications that were actually correct. Precision is important in fraud detection to minimize the number of false positives, which can be costly and time-consuming to investigate.
- **Recall**: The proportion of actual positives that were correctly identified. Recall is crucial in ensuring that a high number of fraudulent claims are detected.
- **F1 Score**: The harmonic mean of precision and recall, providing a balance between the two. F1 Score is particularly useful when dealing with imbalanced datasets, as it considers both false positives and false negatives.
- **ROC-AUC**: Area under the Receiver Operating Characteristic curve, which evaluates the trade-off between true positive rate and false positive rate. A higher AUC indicates a better performing model.

Experimental Steps

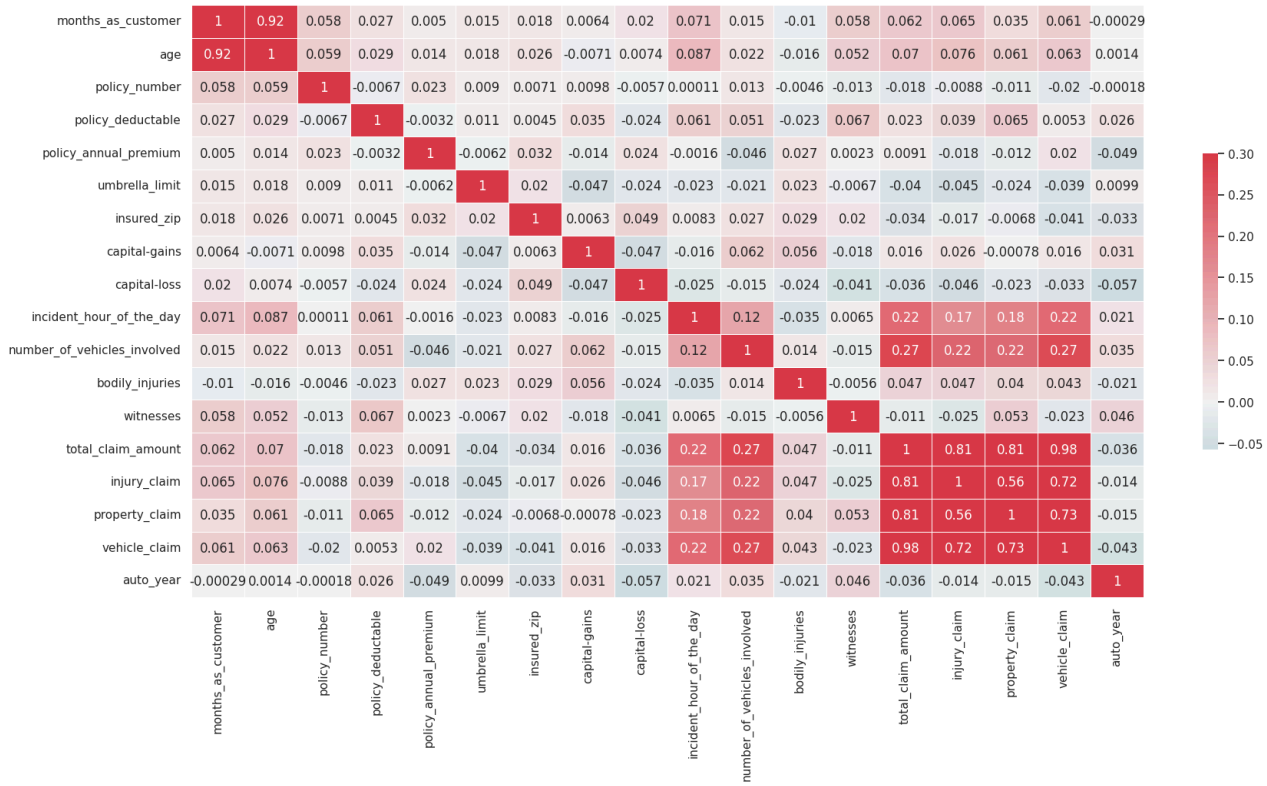
Data Collection

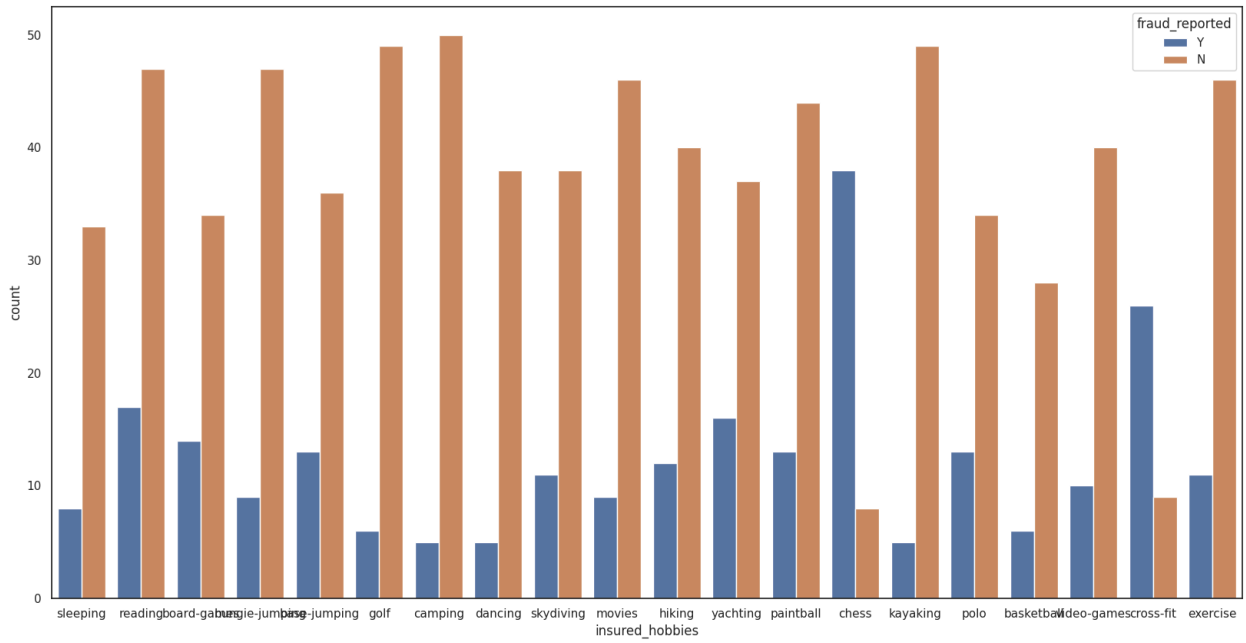
The dataset used for this project includes historical insurance claim data, which is publicly available or obtained from an insurance company. The dataset typically contains features such as:

- *Claim Amount*: The monetary value of the claim. Higher claim amounts might be indicative of fraudulent activity, especially if they exceed a certain threshold.
- *Policy Details*: Information about the insurance policy, such as policy type, coverage limits, and duration. Certain policy types may be more prone to fraud.
- *Claimant Information*: Demographic and other details about the claimant, including age, gender, and past claim history. Repeat claimants or those with a history of fraud may be flagged.
- *Incident Details*: Information about the incident leading to the claim, such as date, location, and description. Patterns in incident details can help identify fraudulent claims.
- *Label*: Indicating whether the claim was fraudulent or legitimate. This is the target variable used for training supervised learning models.

Data Preprocessing

The initial data preparation involved loading a dataset containing various details about insurance policies, claims, and reported fraud. The dataset was loaded from a CSV file using Pandas. The first step was to inspect the dataset for missing values and understand the data types of each feature. Missing values were handled by either dropping columns with a significant amount of missing data or replacing placeholders such as *'?'* with a more appropriate category like '*Unknown'*. For instance, columns such as *police_report_available*, *property_damage*, and *authorities_contacted* were cleaned by replacing '*?*' and *pd.NA* with *'Unknown'* and *'None'*, respectively. For feature engineering, we identified redundant features using a correlation matrix, consolidated dominant categories (for instance. *Claimant Hobbies*), and checked for class imbalance to detect bias.


months_as_customer 1 0.92 0.058 0.027 0.005 0.015 0.018 0.0064 0.02 0.071 0.015 -0.01 0.058 0.062 0.065 0.035 0.061 -0.00029
age 0.92 1 0.059 0.029 0.014 0.018 0.026 -0.0071 0.0074 0.087 0.022 -0.016 0.052 0.07 0.076 0.061 0.063 0.0014
policy_number 0.058 0.059 1 -0.0067 0.023 0.009 0.0071 0.0098 -0.0057 0.00011 0.013 -0.0046 -0.013 -0.018 -0.0088 -0.011 -0.02 -0.00018
policy_deductable 0.027 0.029 -0.0067 1 -0.0032 0.011 0.0045 0.035 -0.024 0.061 0.051 -0.023 0.067 0.023 0.039 0.065 0.0053 0.026
policy_annual_premium 0.005 0.014 0.023 -0.0032 1 -0.0062 0.032 -0.014 0.024 -0.0016 -0.046 0.027 0.0023 0.0091 -0.018 -0.012 0.02 -0.049
umbrella_limit 0.015 0.018 0.009 0.011 -0.0062 1 0.02 -0.047 -0.024 -0.023 -0.021 0.023 -0.0067 -0.04 -0.045 -0.024 -0.039 0.0099
insured_zip 0.018 0.026 0.0071 0.0045 0.032 0.02 1 0.0063 0.049 0.0083 0.027 0.029 0.02 -0.034 -0.017 -0.0068 -0.041 -0.033
capital-gains 0.0064 -0.0071 0.0098 0.035 -0.014 -0.047 0.0063 1 -0.047 -0.016 0.062 0.056 -0.018 0.016 0.026 -0.00078 0.016 0.031
capital-loss 0.02 0.0074 -0.0057 -0.024 0.024 -0.024 0.049 -0.047 1 -0.025 -0.015 -0.024 -0.041 -0.036 -0.046 -0.023 -0.033 -0.057
incident_hour_of_the_day 0.071 0.087 0.00011 0.061 -0.0016 -0.023 0.0083 -0.016 -0.025 1 0.12 -0.035 0.0065 0.22 0.17 0.18 0.22 0.021
number_of_vehicles_involved 0.015 0.022 0.013 0.051 -0.046 -0.021 0.027 0.062 -0.015 0.12 1 0.014 -0.015 0.27 0.22 0.22 0.27 0.035
bodily_injuries -0.01 -0.016 -0.0046 -0.023 0.027 0.023 0.029 0.056 -0.024 -0.035 0.014 1 -0.0056 0.047 0.047 0.04 0.043 -0.021
witnesses 0.058 0.052 -0.013 0.067 0.0023 -0.0067 0.02 -0.018 -0.041 0.0065 -0.015 -0.0056 1 -0.011 -0.025 0.053 -0.023 0.046
total_claim_amount 0.062 0.07 -0.018 0.023 0.0091 -0.04 -0.034 0.016 -0.036 0.22 0.27 0.047 -0.011 1 0.81 0.81 0.98 -0.036
injury_claim 0.065 0.076 -0.0088 0.039 -0.018 -0.045 -0.017 0.026 -0.046 0.17 0.22 0.047 -0.025 0.81 1 0.56 0.72 -0.014
property_claim 0.035 0.061 -0.011 0.065 -0.012 -0.024 -0.0068 -0.00078 -0.023 0.18 0.22 0.04 0.053 0.81 0.56 1 0.73 -0.015
vehicle_claim 0.061 0.063 -0.02 0.0053 0.02 -0.039 -0.041 0.016 -0.033 0.22 0.27 0.043 -0.023 0.98 0.72 0.73 1 -0.043
auto_year -0.00029 0.0014 -0.00018 0.026 -0.049 0.0099 -0.033 0.031 -0.057 0.021 0.035 -0.021 0.046 -0.036 -0.014 -0.015 -0.043 1
months_as_customer age policy_number policy_deductable policy_annual_premium umbrella_limit insured_zip capital-gains capital-loss incident_hour_of_the_day number_of_vehicles_involved bodily_injuries witnesses total_claim_amount injury_claim property_claim vehicle_claim auto_year
0.30 0.25 0.20 0.15 0.10 0.05 0.00 −0.05


fraud_reported
Y
N
count
0 10 20 30 40 50
sleeping reading board-games bungie-jumping base-jumping golf camping dancing skydiving movies hiking yachting paintball chess kayaking polo basketball video-games cross-fit exercise
insured_hobbies

Data Processing

Once the initial cleaning was complete, the data processing steps included encoding categorical variables into numerical formats suitable for machine learning models. Categorical responses were converted using Label Binarizer, and categorical features were transformed using One Hot Encoding. This process expanded the dataset to 93 columns, ensuring all categorical data was properly represented numerically. This prepared dataset was then exported for model building.

Model Implementation

Random Forest

Hyperparameter tuning: In the Random Forest model, the following hyperparameters were used for tuning –

- *n_estimators*: This parameter specifies the number of trees in the forest. More trees usually improve the model's performance but also increase the computation time.
- *max_features*: This parameter determines the number of features to consider when looking for the best split. The options included:
    - *'sqrt'*: The square root of the total number of features.
    - *'log2'*: The base-2 logarithm of the total number of features.
    - *None*: The total number of features.
- *max_depth*: This parameter sets the maximum depth of the trees. Limiting the depth of the tree helps prevent overfitting. The values used were 3, 5, and 7.
- *max_leaf_nodes*: This parameter restricts the number of leaf nodes in the tree. Fewer leaf nodes reduce model complexity and help in avoiding overfitting. The values tested were 3, 6, and 9.
- *min_samples_split*: This parameter specifies the minimum number of samples required to split an internal node. The values used were 2, 4, and 6.

The optimization of hyperparameters was performed using Grid Search with Cross-Validation. Grid Search exhaustively tests all possible combinations of the

specified hyperparameter values to find the best combination that maximizes the model's performance

Training: The model is trained on the training set with cross-validation to tune hyperparameters. Random Forest's inherent feature importance helps in understanding which features are most influential in predicting fraud.

Cross-Validation: k-fold cross-validation is used to ensure the model's generalizability and to fine-tune hyperparameters. This involves dividing the dataset into k subsets, training the model on k-1 subsets, and validating it on the remaining subset. This process is repeated k times.

Evaluation: The trained model is evaluated on the test set using the defined metrics. The evaluation metrics provide insights into the model's performance and its ability to generalize to new data.

XGBoost

Hyperparameter tuning: In the XGBoost model, the following hyperparameters were used for tuning:

- *n_estimators*: Number of boosting rounds.
- *learning_rate*: Controls the weight of each boosting round.
- *max_depth*: Maximum depth of each tree.
- *min_child_weight*: Minimum sum of instance weight (hessian) needed in a child.
- *subsample*: Fraction of samples used for fitting individual trees.
- *colsample_bytree*: Fraction of features used for fitting individual trees.
- *gamma*: Minimum loss reduction required to make a split.
- *reg_alpha*: L1 regularization term on weights.
- *reg_lambda*: L2 regularization term on weights.
- *max_leaves*: Maximum number of leaves in a tree.
- *max_bin*: Maximum number of bins for discretizing continuous features.
- *scale_pos_weight*: Controls the balance of positive and negative weights.

Training: The final model, trained with the optimized hyperparameters, demonstrated improved performance metrics, providing a robust and accurate classification solution for the insurance fraud detection dataset.

Evaluation: The trained model is evaluated on the test set using the defined metrics. The evaluation metrics provide a comprehensive understanding of the model's strengths and weaknesses.

Training and Validation

- Train-Test Split: The dataset is split into training and testing sets, typically in an 80:20 ratio. This ensures that the model is trained on a subset of the data and validated on a separate subset to assess its performance.
- Cross-Validation: k-fold cross-validation is used to ensure the model's generalizability and to fine-tune hyperparameters. This technique helps prevent overfitting and ensures that the model performs well on unseen data.

Results

Random Forest

- *Accuracy*: 83%
- *Precision*: Positive class - 67%, Negative class - 91%
- *Recall*: Positive class - 76%, Negative class - 86%
- *F1 Score*: Positive class - 81%, Negative class - 88%
- *ROC-AUC*: 0.81

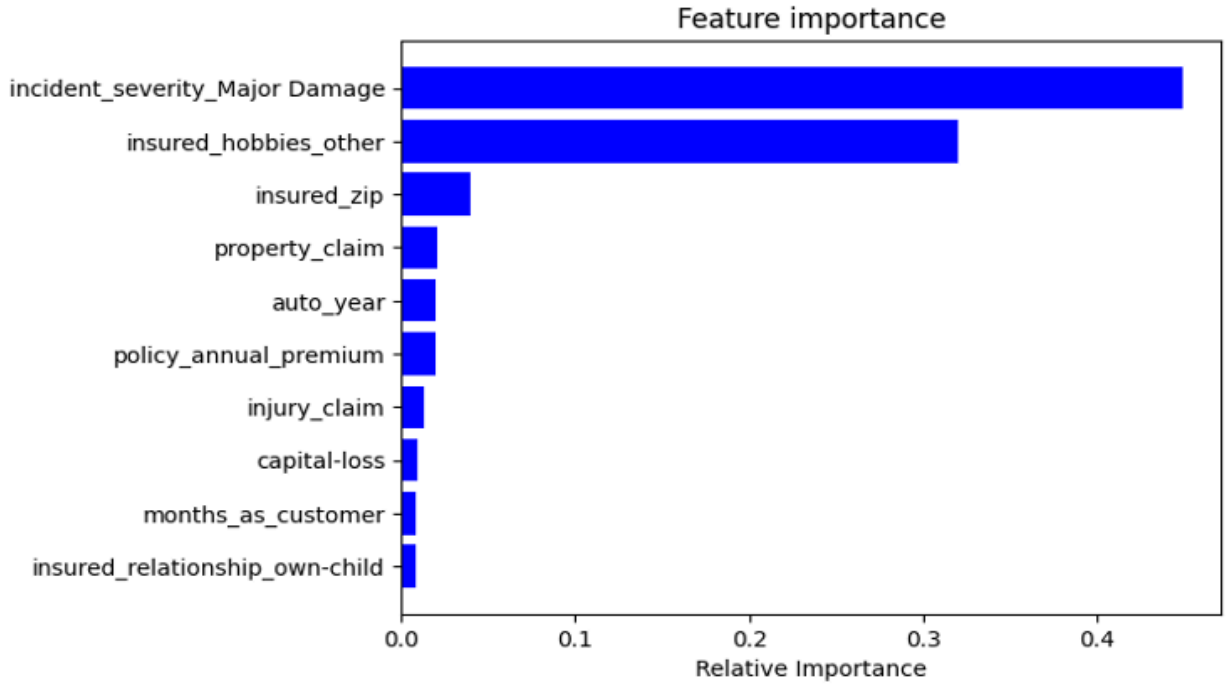
The Random Forest model showed a good balance between precision and recall, indicating its ability to effectively identify fraudulent claims while minimizing false positives. The feature importance provided by Random Forest highlighted that the most significant features were the claim amount, policy details, and claimant's history of claims.

Classification Report

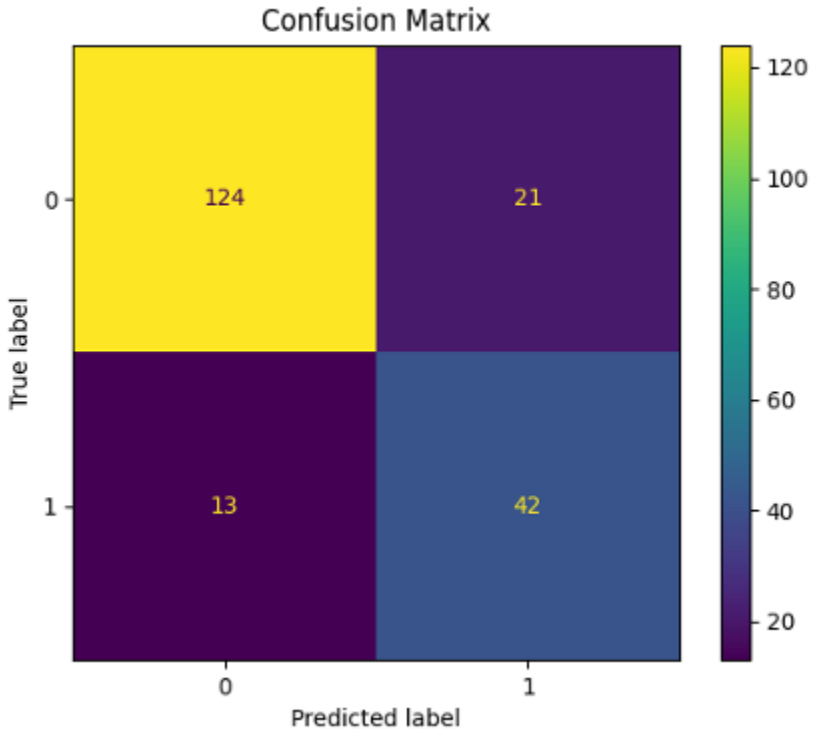
Classfication Report

	precision	recall	f1-score	support
0	0.91	0.86	0.88	145.00
1	0.67	0.76	0.71	55.00
accuracy	0.83	0.83	0.83	0.83
macro avg	0.79	0.81	0.80	200.00
weighted avg	0.84	0.83	0.83	200.00

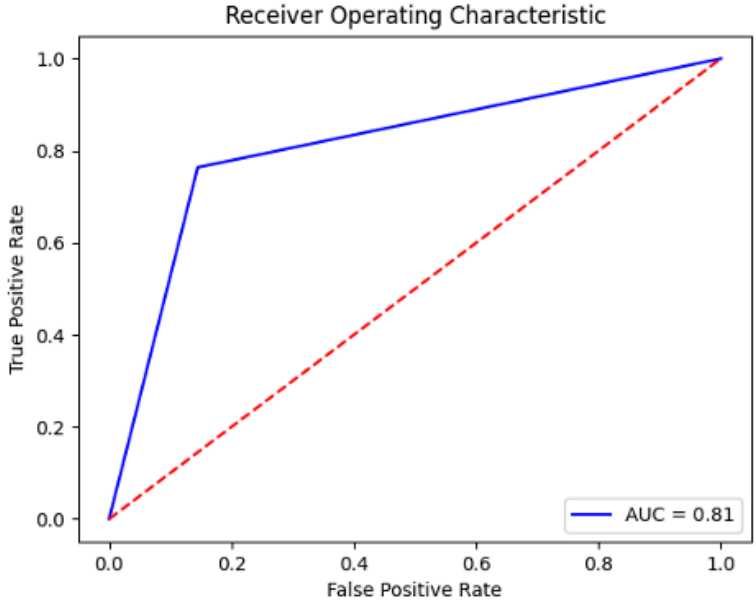
Feature Importance



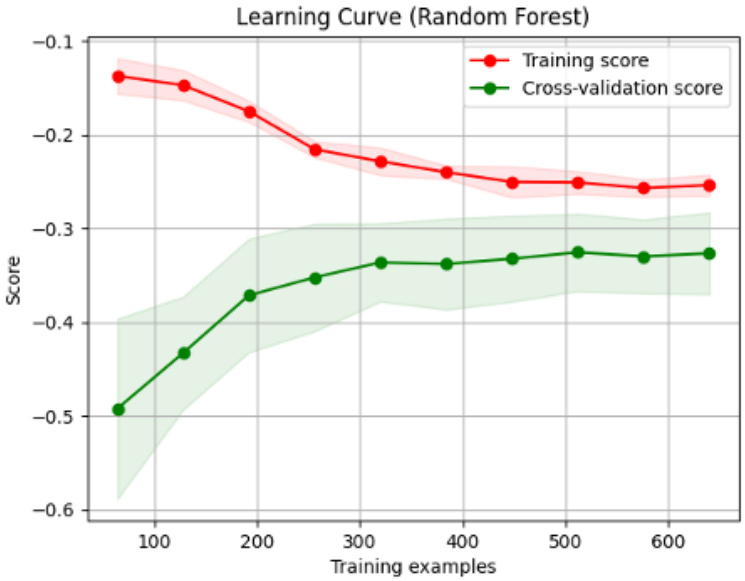
Confusion Matrix



Receiver Operating Characteristic (ROC)



Learning Curve



XGBoost

- *Accuracy*: 83.5%
- *Precision*: Positive class - 65%, Negative class - 94%
- *Recall*: Positive class - 87%, Negative class - 82%
- *F1 Score*: Positive class - 74%, Negative class - 88%
- *ROC-AUC*: 0.85

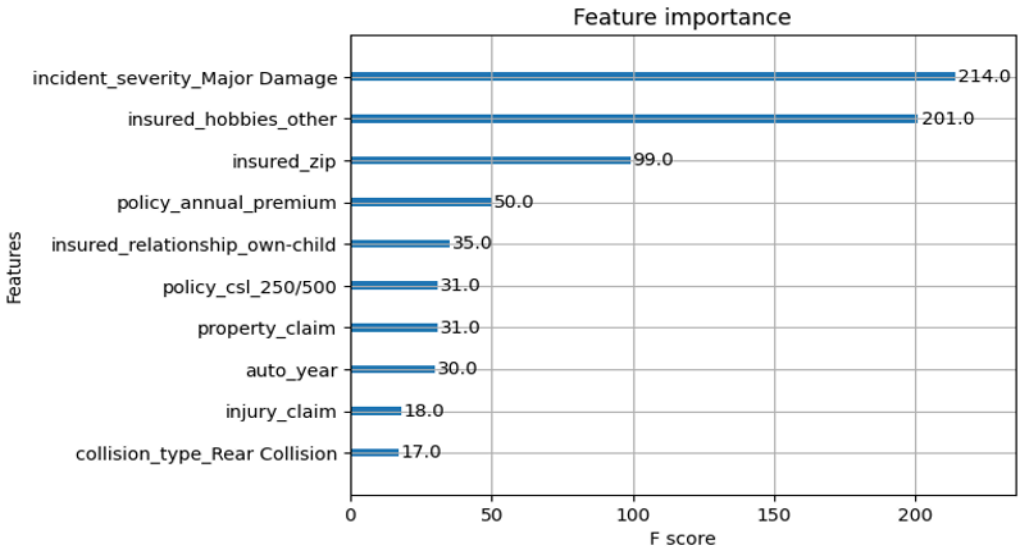
The XGBoost model outperformed the Random Forest model in terms of accuracy, precision, recall, and F1 score. Its ability to handle missing values and incorporate regularization contributed to its superior performance. The feature importance analysis from XGBoost also pointed out similar key features as Random Forest but with more refined granularity.

Classification Report

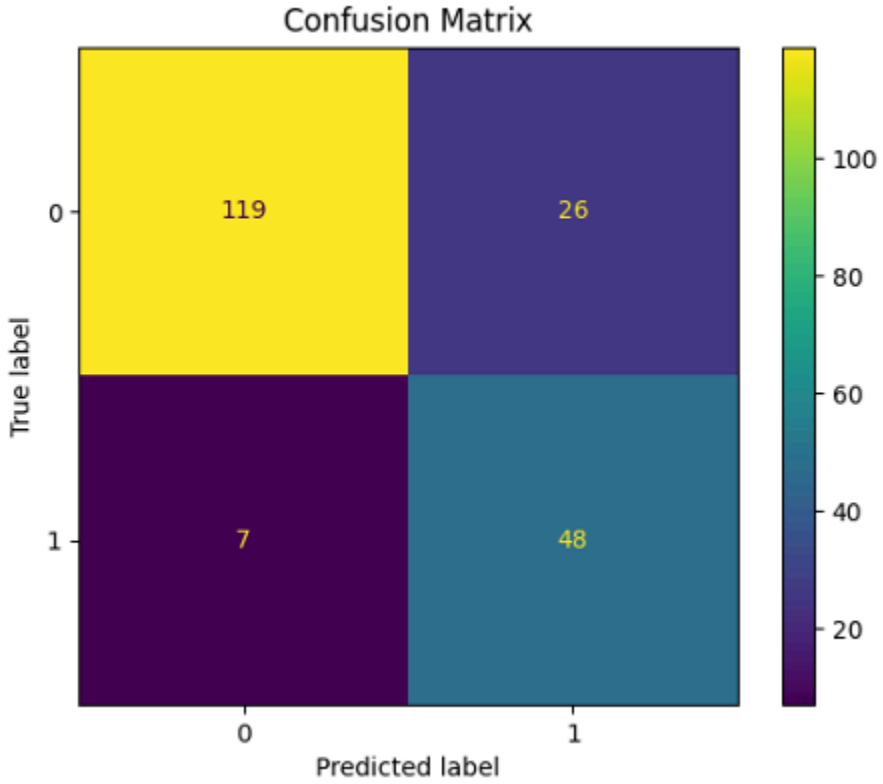
Classfication Report

	precision	recall	f1-score	support
0	0.94	0.82	0.88	145.00
1	0.65	0.87	0.74	55.00
accuracy	0.83	0.83	0.83	0.83
macro avg	0.80	0.85	0.81	200.00
weighted avg	0.86	0.83	0.84	200.00

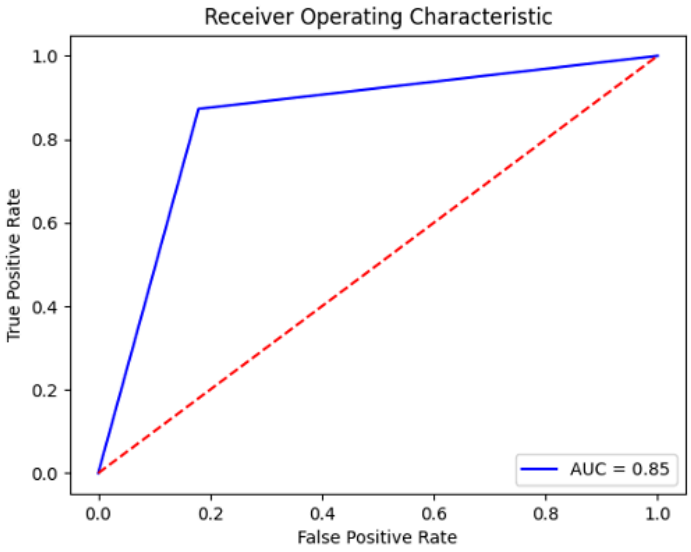
Feature Importance



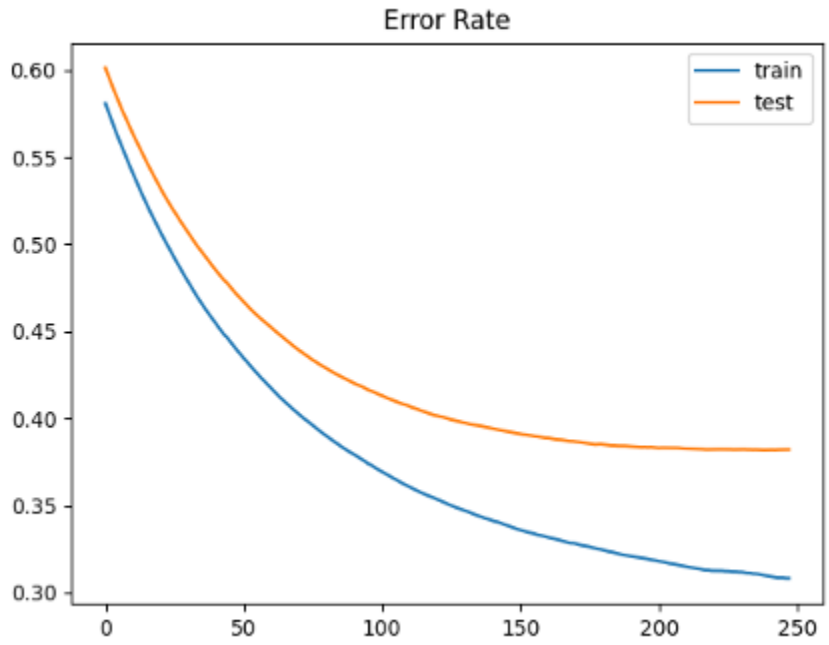
Confusion Matrix



Receiver Operating Characteristic (ROC)



Error Rate



Comparative Analysis

- **Precision-Recall Trade-offs**: XGBoost showed a higher precision and recall balance compared to Random Forest, indicating better performance in detecting fraudulent claims while minimizing false positives.
- **Feature Importance**: Both models highlighted similar key features influencing fraud detection, such as claim amount, policy details, and claimant history. However, XGBoost provided more detailed insights into feature interactions and their impact on predictions.
- **Model Complexity**: While Random Forest is easier to implement and interpret, XGBoost's complexity requires more careful tuning but offers superior performance when optimized correctly.

Conclusion

The application of machine learning in insurance fraud detection demonstrates significant potential in enhancing the accuracy and efficiency of identifying fraudulent claims. XGBoost outperformed Random Forest in terms of accuracy and overall robustness. The results indicate that machine learning models, particularly XGBoost, can effectively identify patterns and anomalies associated with fraudulent claims, providing a valuable tool for insurance companies to mitigate risks.

Future work could explore the integration of additional data sources, such as social media data or other external databases, to further improve the detection accuracy. Additionally, implementing real-time detection systems and exploring advanced techniques like deep learning and reinforcement learning could offer further improvements in fraud detection capabilities. Collaboration with domain experts in insurance can also help in refining feature engineering and model interpretation, ensuring that the models are aligned with practical needs and constraints.

References

1. Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., & Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decision Support Systems, 50(3), 559-569.
2. Phua, C., Lee, V., Smith, K., & Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. arXiv preprint arXiv:1009.6119.
3. Bauder, R. A., & Khoshgoftaar, T. M. (2018). A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Access, 6, 19201-19223.
4. Sahoo, S., Singh, A. K., & Mukhopadhyay, S. (2018). A survey on different approaches for detecting fraud in insurance. Journal of King Saud University-Computer and Information Sciences, 30(4), 419-432.
5. Ghosh, S., & Reilly, D. L. (1994). Credit card fraud detection with a neural-network. In System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on (Vol. 3, pp. 621-630). IEEE.