Fraudulent Claim Detection

BY Paras Mehta
Shaik Muhammad Naim

Problem Statement

Global Insure, a leading insurance company, processes thousands of claims annually. However, a significant percentage of these claims turn out to be fraudulent, resulting in considerable financial losses. The company’s current process for identifying fraudulent claims involves manual inspections, which is time-consuming and inefficient. Fraudulent claims are often detected too late in the process, after the company has already paid out significant amounts. Global Insure wants to improve its fraud detection process using data-driven insights to classify claims as fraudulent or legitimate early in the approval process. This would minimise financial losses and optimise the overall claims handling process

Business Objective

Global Insure wants to build a model to classify insurance claims as either fraudulent or legitimate based on historical claim details and customer profiles. By using features like claim amounts, customer profiles and claim types, the company aims to predict which claims are likely to be fraudulent before they are approved.

Based on this assignment, you have to answer the following questions:

- How can we analyse historical claim data to detect patterns that indicate fraudulent claims?
- Which features are most predictive of fraudulent behaviour?
- Can we predict the likelihood of fraud for an incoming claim, based on past data?
- What insights can be drawn from the model that can help in improving the fraud detection process?

Assignment Overview

Need to perform the following steps for successfully completing this assignment:

1. Data Preparation
2. Data Cleaning
3. Train Validation Split 70-30
4. EDA on Training Data
5. EDA on Validation Data (optional)
6. Feature Engineering
7. Model Building
8. Predicting and Model Evaluation

Step to Build Models

- Data Cleaning
- Train-Validate data split
- EDA (Exploratory Data Analysis)
- Feature Creation and Scaling
- Model Building
- Model Evaluation

-

Data Cleaning

Data Cleaning

- Handle the Null value with UnKnown
- Drop the Blank column '_c39'
- Remove the invalid row from umbrella_limit.
- Remove the near to Unique column like policy_number,policy_bind_date,insured_zip,incident_date,incident_location,total_claim_amount
- Fix the datatype of categorical column
-

-

TRAIN –VALIDATION DATA SPLIT

Train and validation Data

- Define Feature and Target variable in both train and test data set
  - Target Variable – fraud_reported

- Split Train and Test data set with 70:30 means 70% train dataset and 30% test dataset

•

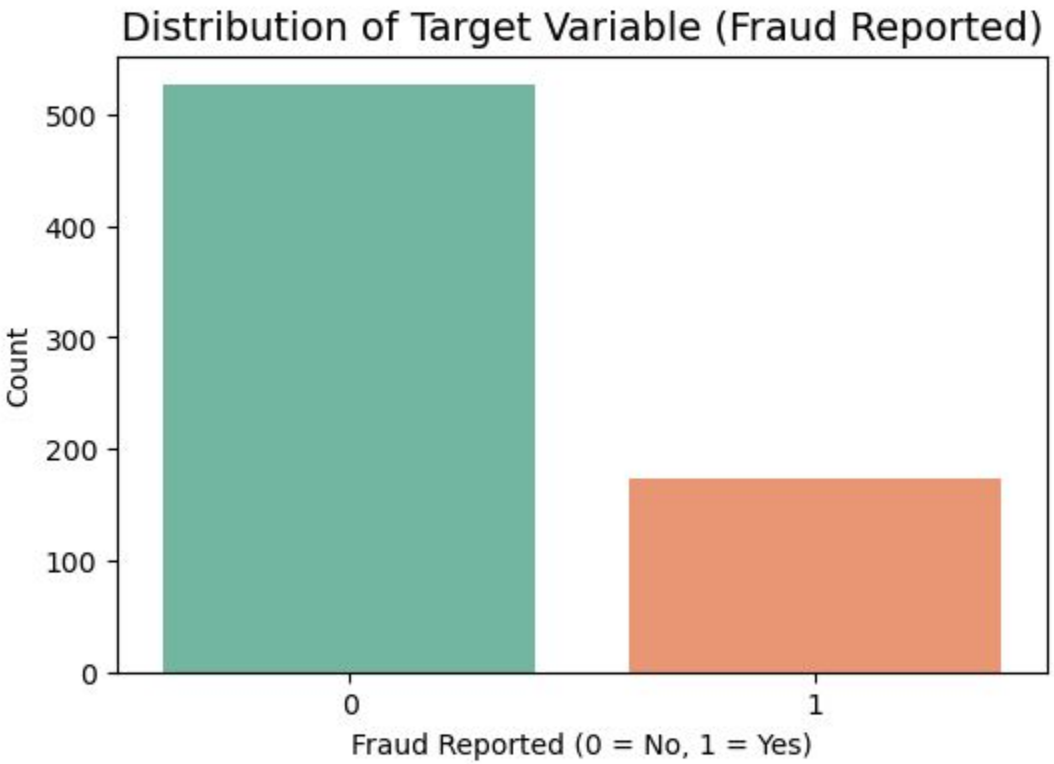
EXPLORATORY DATA ANALYSIS

EDA

- Uni Variate Analysis (Analysis of any selected feature)
- Bi Variate Analysis (Comparison of any feature with target variable, here it is fraud reported)
- Correlation Analysis – Correlation matrix of numerical columns

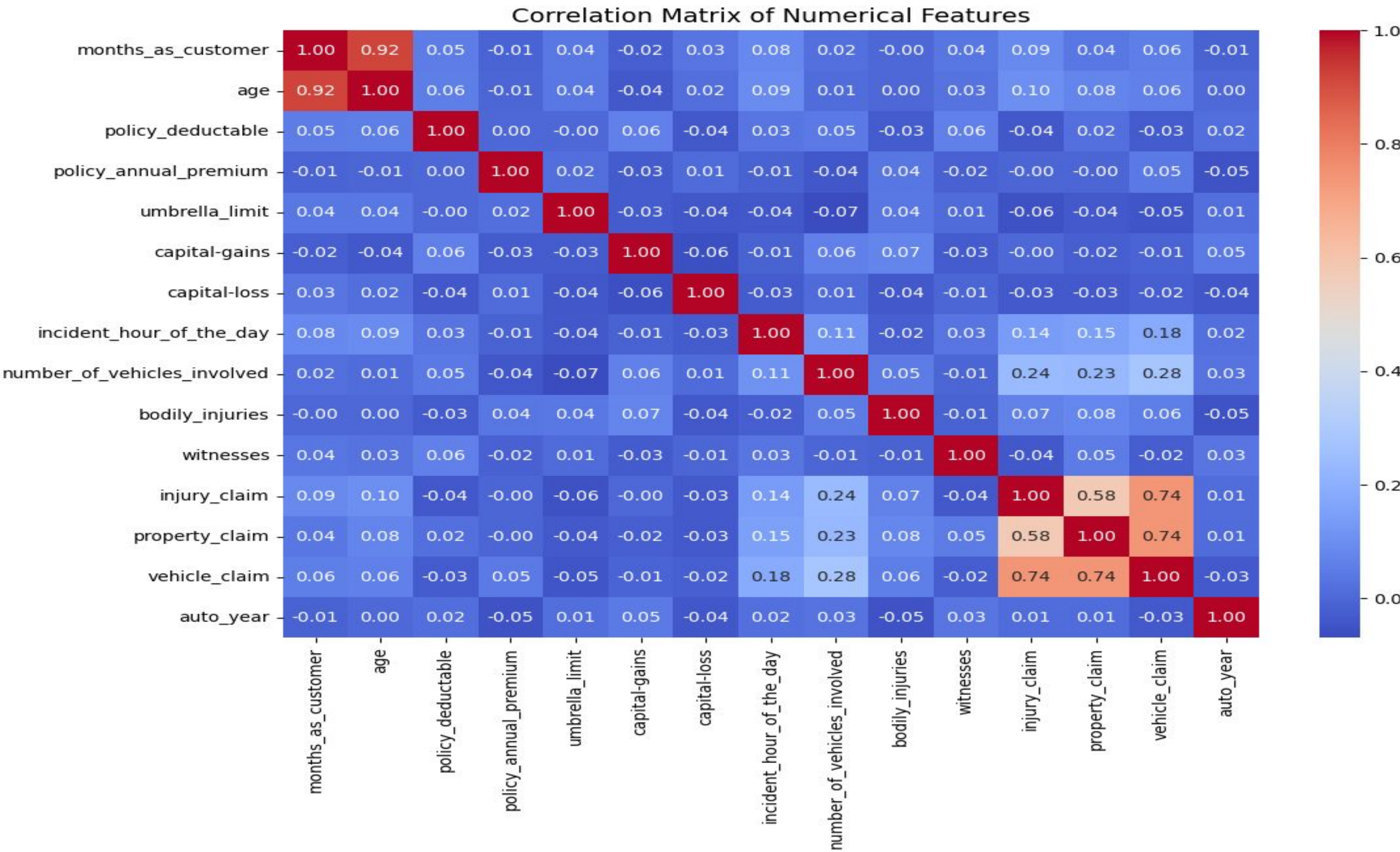
EDA – Univariate Analysis

EDA – Class Balance

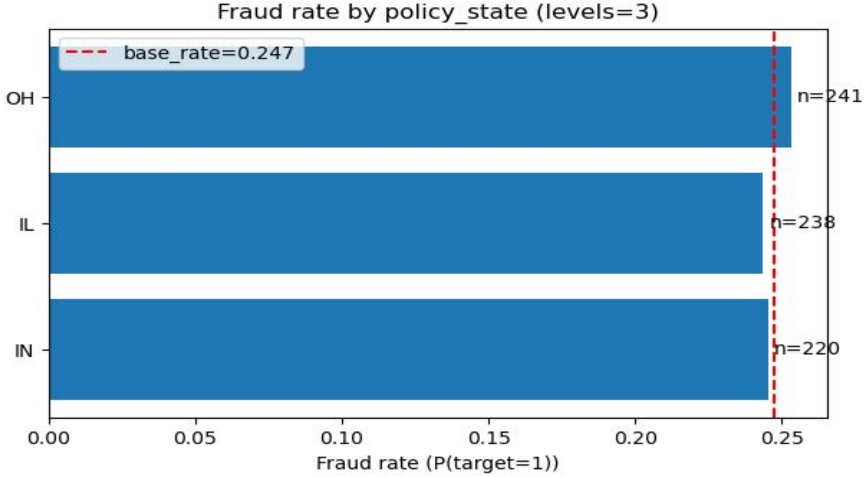


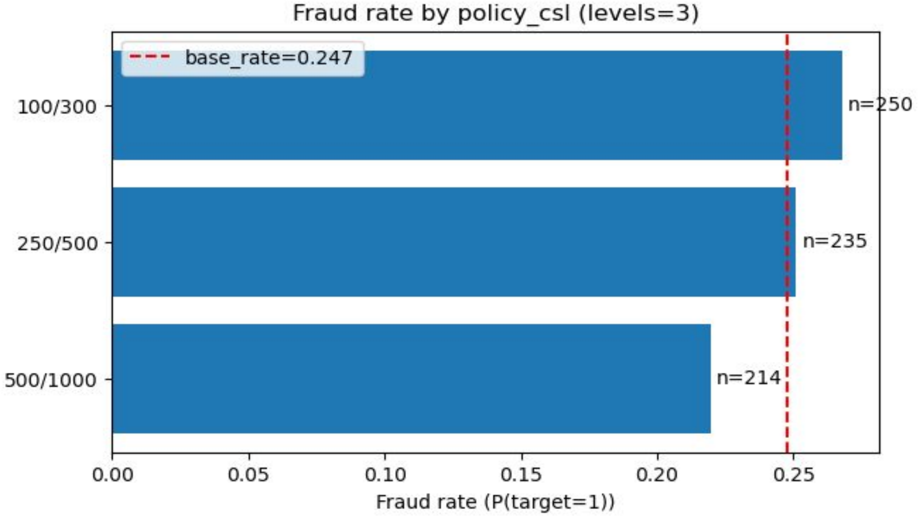
Fraud Reported percentage in Training data set nearly – 24%

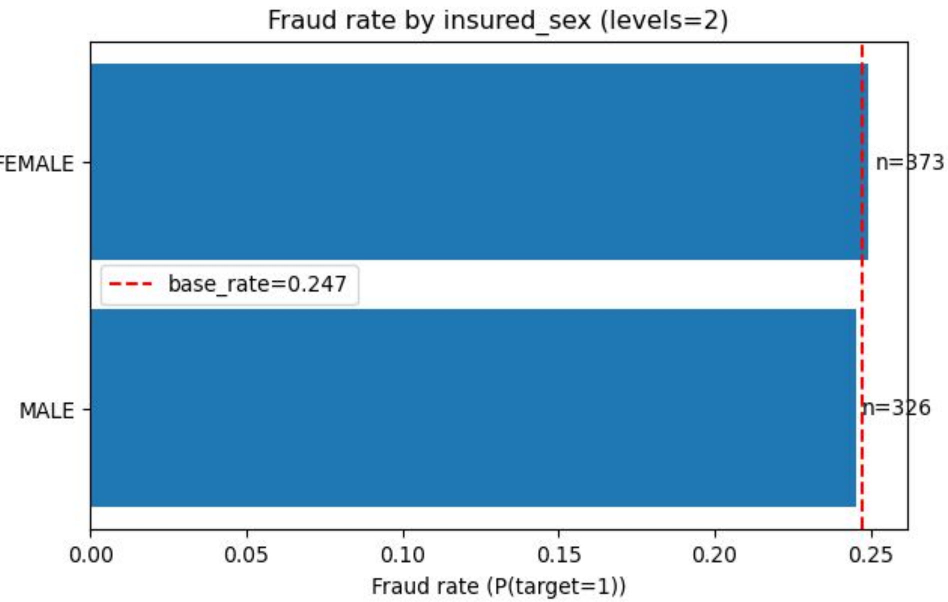
EDA – Correlation

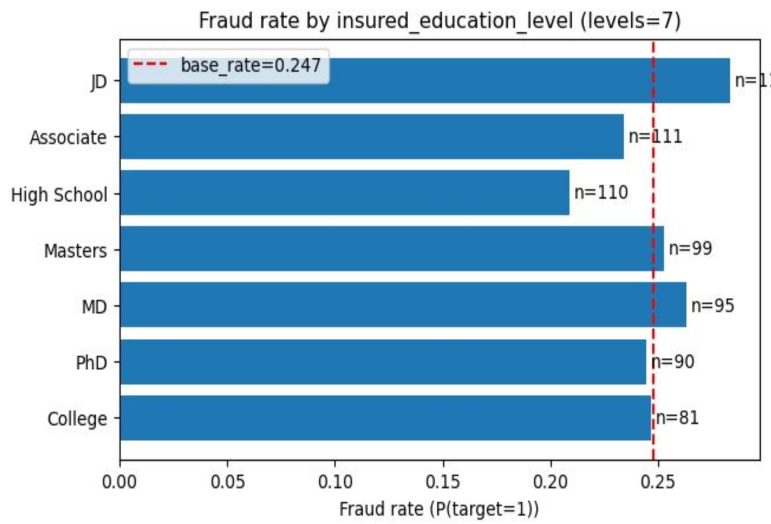


EDA – Bivariate Analisys

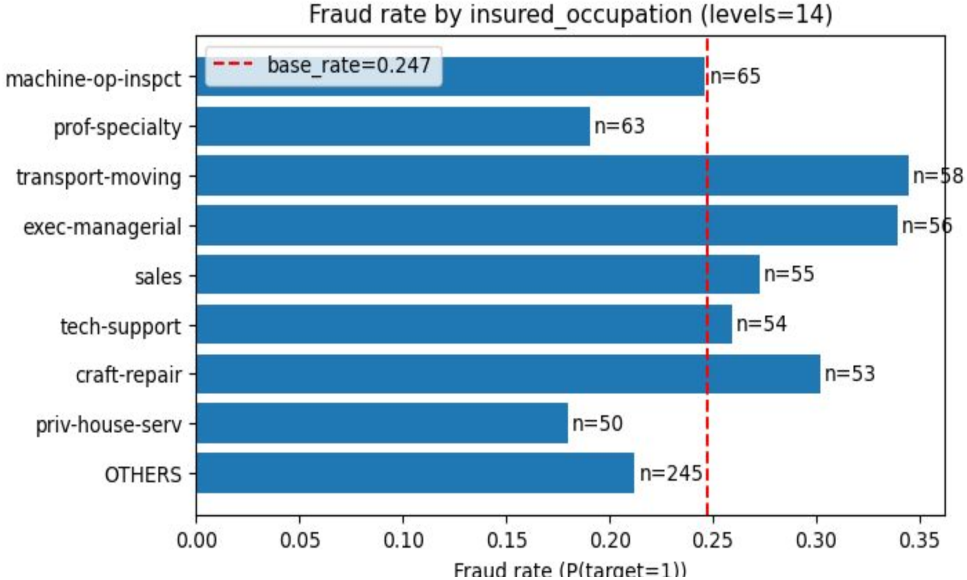


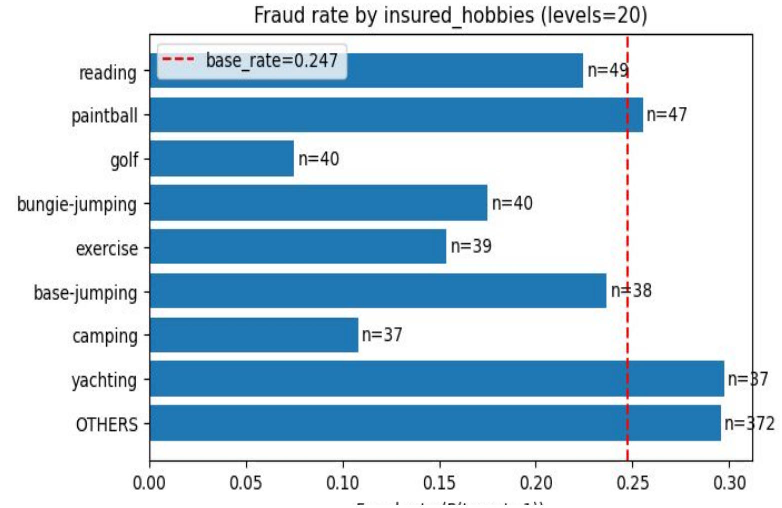


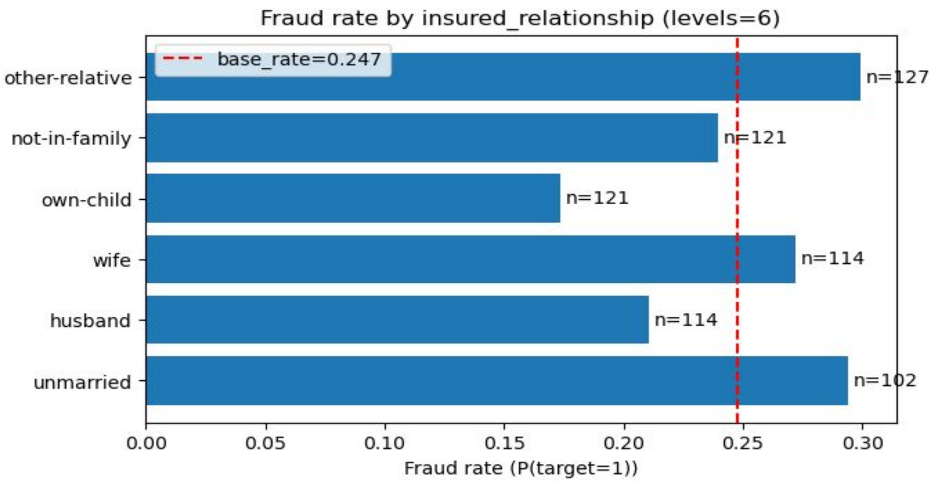


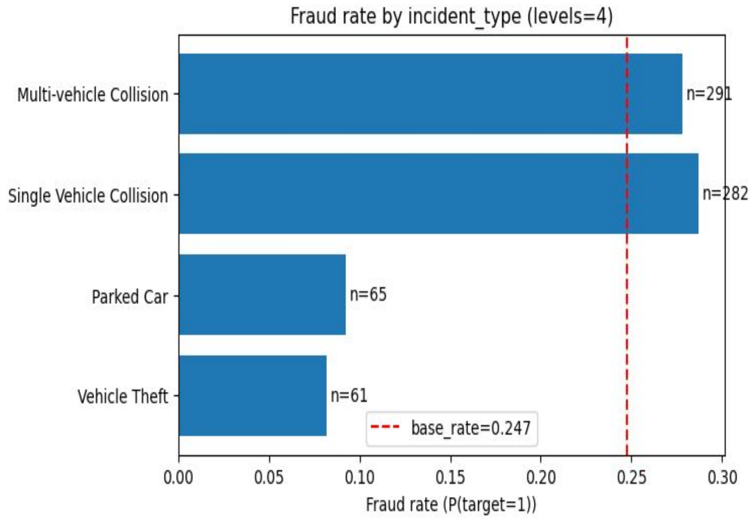


EDA – Bivariate Analisys









Feature Creation - Resampling

Feature Creation and Resampling

- Resampling technique to balance the data and handle class imbalance
- This helps prevent the model from being biased toward the majority class and improves its ability to predict the the minority class more accurately.

```
Class distribution before resampling: Counter({0: 526, 1: 173})
Class distribution after resampling: Counter({0: 526, 1: 526})
Original training shape: (699, 32) (699,)
 Resampled training shape: (1052, 32) (1052,)
```

Feature Creation and Resampling

- Derived features from existing features to give more meaningful prediction
  policy_premium_per_month, claim_component_sum, is_single_component_claim etc
- Combine values in categorical columns so that by grouping values that have low frequency or provide limited predictive information.
- Transform categorical variables into numerical representations using dummy variables
- Scale numerical features to a common range to prevent features with larger values from dominating the model

-

Model Building

Model Used

- Logistic Regression
- Random Forest

Logistic Regression Model

- Feature Selection using RFECV (Recursive Feature Elimination with Cross-Validation)

RFECV Feature Ranking:

	Feature	Rank	Selected
0	age	1	True
74	incident city Hillsdale	1	True
73	incident city Columbus	1	True
72	incident state WV	1	True
71	incident state VA	1	True
70	incident state SC	1	True
69	incident state Other	1	True
68	incident state NY	1	True
67	authorities contacted unknown	1	True
66	authorities contacted Police	1	True
65	authorities contacted Other	1	True
64	authorities contacted Fire	1	True
63	incident severity Trivial Damage	1	True
62	incident severity Total Loss	1	True
61	incident severity Minor Damage	1	True
60	collision type unknown	1	True
59	collision type Side Collision	1	True
58	collision type Rear Collision	1	True
57	incident type Vehicle Theft	1	True
56	incident_type_Single Vehicle Collision	1	True

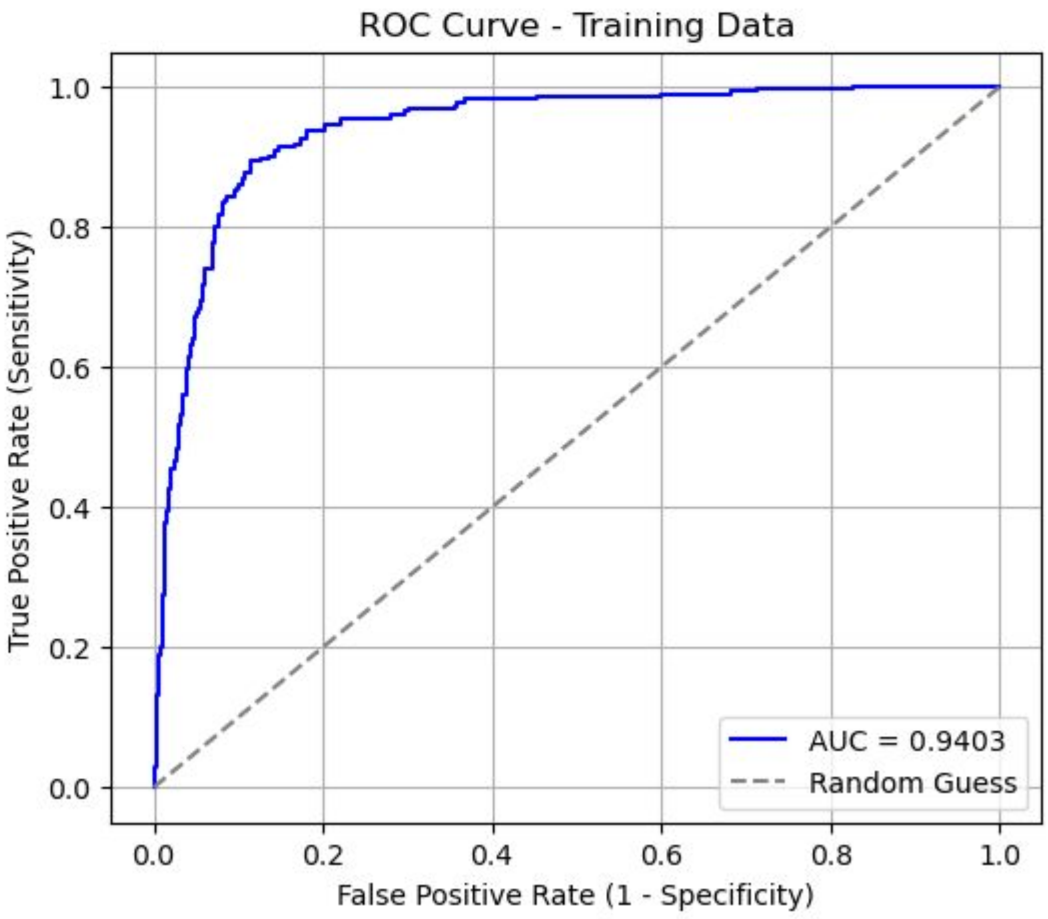
Logistic Regression Model

- Variance Inflation Factors(VIFs) of different features to assess Multi-Collinearity

	Feature	VIF
2	policy annual premium	inf
19	vehicle age	inf
10	auto year	inf
11	policy premium per month	inf
12	claim component sum	271.465101
14	avg claim component	247.817634
17	age x premium	58.554066
53	incident type Single Vehicle Collision	40.750543
0	age	38.391713
96	tenure bucket 10+yr	34.085855
18	claim per vehicle	32.938647
98	age bucket mid-age	23.825792
16	claim to premium ratio	22.257544
57	collision type unknown	20.918655
97	age bucket adult	19.238928
7	number of vehicles involved	19.226587
13	claim component nonzero	14.309691
38	insured hobbies Other	10.008777
95	tenure bucket 5-10yr	8.860079
15	is_single_component_claim	5.518891

Logistic Regression Model

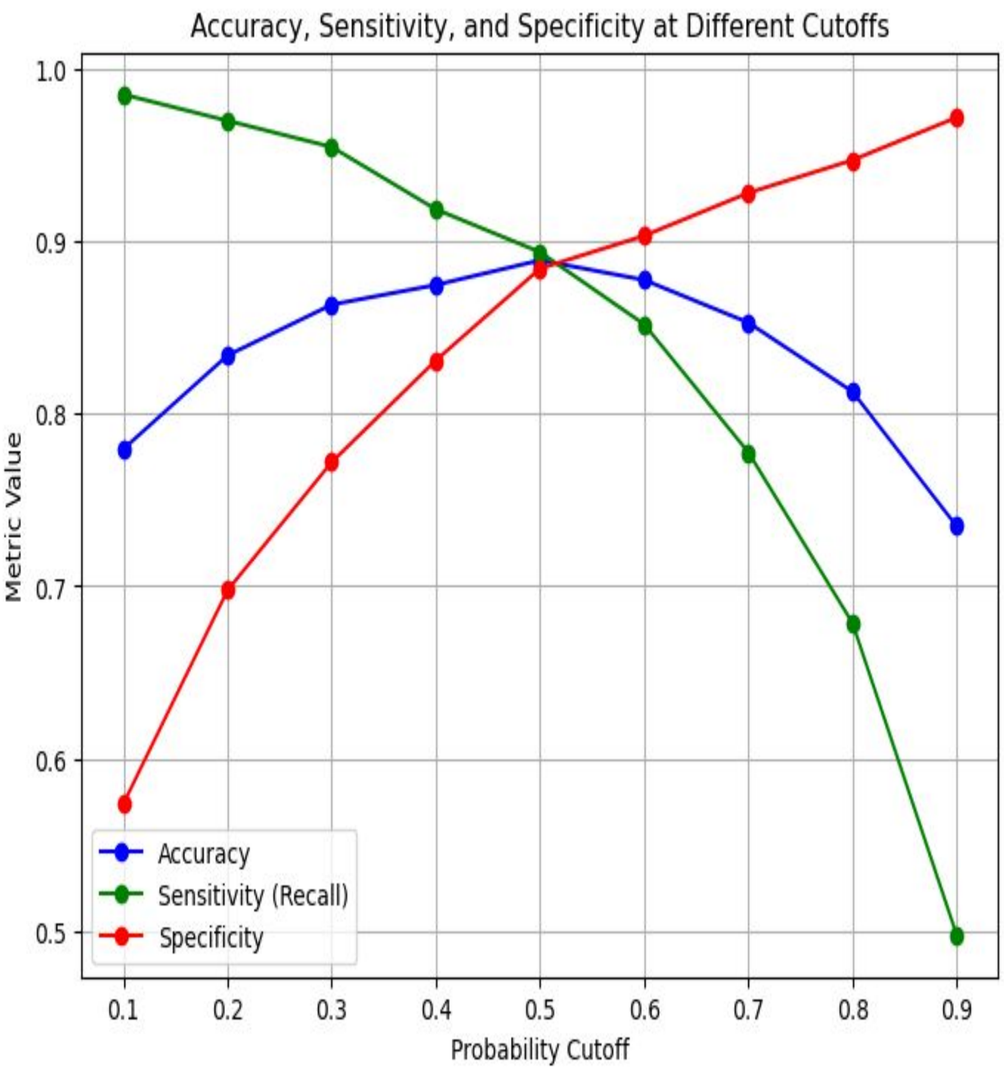
ROC Curve



AUC Curve
shows how well the model has been able to separate out the classes
ROC AUC Score of 0.94 shows the model has done a good job

Logistic Regression Model

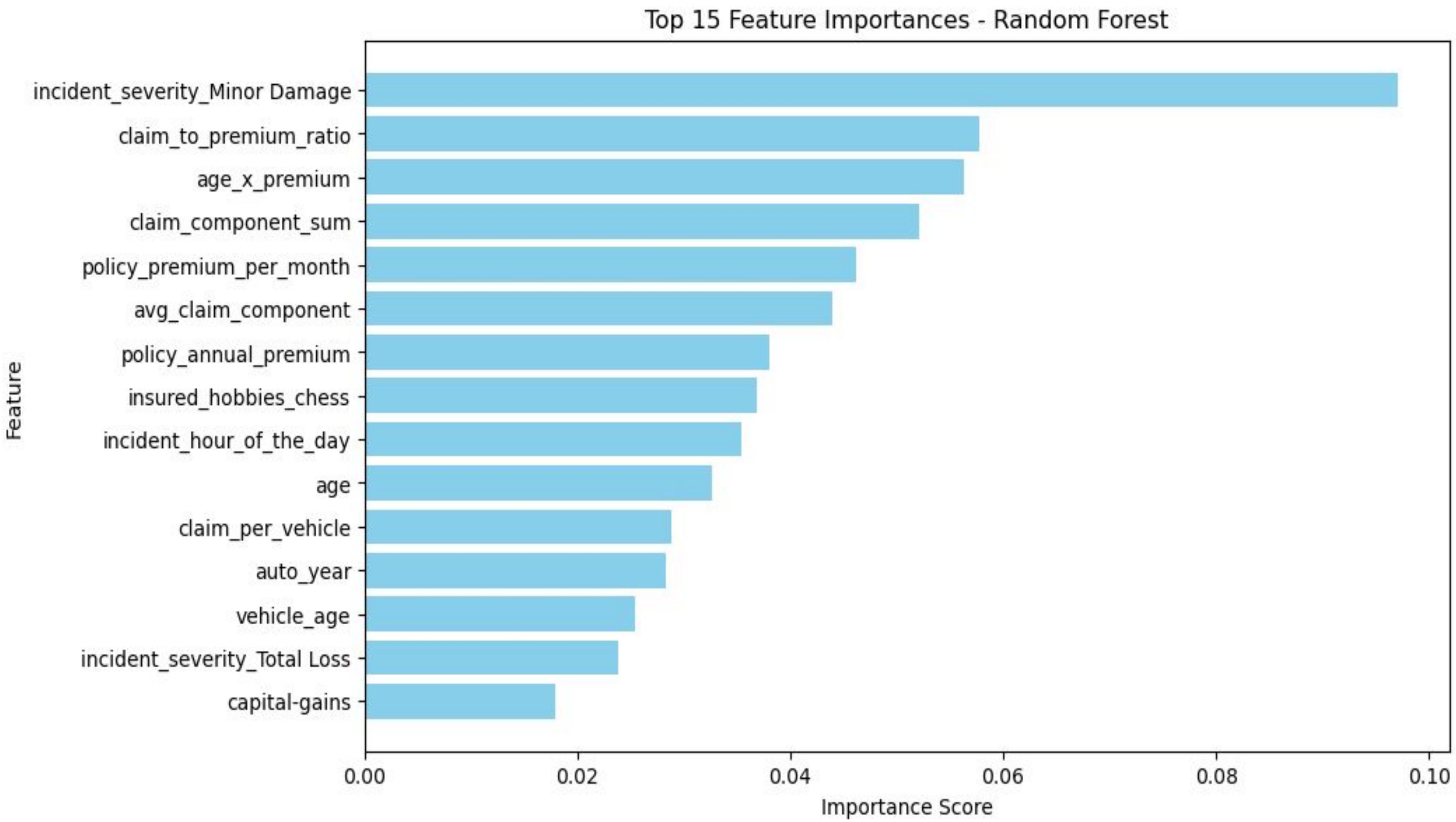
Optimal Cutoff



Optimal Cutoff 0.5

Random Forest Model

- Feature Selection



Random Forest Model

Random Forest Performance on Training Data:

Accuracy: 0.9933

Sensitivity: 0.9962

Specificity: 0.9905

Precision: 0.9905

Recall: 0.9962

F1 Score: 0.9934

After Hyper Tuning

Accuracy: 1

Sensitivity: 1

Specificity: 1

Precision: 1

Recall: 1

F1 Score:1

Model Evolution and Validation

Logistic Regression Model Evalution

Performance on Training Data:

Accuracy: 0.8888

Sensitivity: 0.8835

Specificity: 0.8840

Precision: 0.8851

Recall: 0.8935

F1 Score: 0.8893

Testing data Perfomance

Accuracy: 0.74

Sensitivity: 0.5946

Specificity: 0.7876

Precision: 0.4783

Recall: 0.5946

F1 Score:0.5301

Random Forest Model Evalution

Performance on Training Data:

Accuracy: 1

Sensitivity: 1

Specificity: 1

Precision: 1

Recall: 1

F1 Score:1

Testing data Perfomance

Accuracy: 0.77

Sensitivity: 0.4459

Specificity: 0.8761

Precision: 0.5410

Recall: 0.4459

F1 Score:0.4889

Conclusion

Logistic Regression Results

Training Performance Accuracy: 0.8888 (very strong) Confusion Matrix: [[465, 61], [56, 470]] TN = 465, FP = 61, FN = 56, TP = 470 Sensitivity (Recall): 0.8935 - model correctly catches ~89% of fraud cases. Specificity: 0.8840 - also good at correctly identifying legitimate claims. Precision: 0.8851 - 89% of predicted frauds are actual fraud. F1 Score: 0.8893 - balanced performance. Training metrics look very strong and balanced across all measures.

Validation (Test) PerformancE Accuracy: 0.7400 (drops from training → sign of overfitting / weaker generalization) Confusion Matrix: [[178, 48], [30, 44]] TN = 178, FP = 48, FN = 30, TP = 44 Sensitivity (Recall): 0.5946 - only 59% of frauds are detected (misses 41%). Specificity: 0.7876 - does better at identifying legitimate claims. Precision: 0.4783 - less than half of predicted frauds are actually fraud. F1 Score: 0.5301 - moderate balance, but weak compared to training On validation data, the model performance drops significantly, especially in Precision and Recall.

Conclusion for logistic regression Logistic Regression fits the training data very well, showing balanced and strong performance However, on validation data, it generalizes poorly: Accuracy drops to 74%. Recall falls to 59% (model misses many fraud cases). Precision is low (48%), meaning high false positives. This suggests overfitting or that linear decision boundaries are insufficient for capturing fraud patterns.

Conclusion

Random Forest Results

Training Performance (before and after tuning) Baseline RF (n_estimators=10) Accuracy: 0.9933 Confusion Matrix: [[521, 5], [2, 524]] Sensitivity (Recall): 0.9962 Specificity: 0.9905 F1 Score: 0.9934 Already extremely high performance on training set.

After GridSearchCV tuning (best params: depth=20, n_estimators=500, etc.) Training Accuracy: 1.0000 Confusion Matrix: [[526, 0], [0, 526]] Sensitivity, Specificity, Precision, Recall, F1: all = 1.0 Perfect fit on training data → a clear sign of overfitting.

Test (Validation) Performance Accuracy: 0.7400 (same as Logistic Regression test accuracy). Confusion Matrix: [[178, 48], [30, 44]] TN = 178, FP = 48, FN = 30, TP = 44 Sensitivity (Recall): 0.5946 → catches ~59% of fraud (misses 41%). Specificity: 0.7876 → correctly identifies ~79% legitimate claims. Precision: 0.4783 → less than half of flagged frauds are true fraud. F1 Score: 0.5301 → weak balance of precision & recall. Despite perfect training performance, the test metrics collapse to the same level as Logistic Regression.

Conclusion Random Forest massively overfits: It memorizes the training set (100% accuracy). But generalization to test set is poor (only 74% accuracy). Performance on test set (Accuracy = 0.74, Recall = 0.59, Precision = 0.48, F1 = 0.53) is almost identical to Logistic Regression's test performance. This means Random Forest did not improve generalization, even though it overfits more aggressively than Logistic Regression.

Conclusion

Logistic Regression vs Random Forest — Model Comparison

Model Performance

Metric	Logistic Regression	Random Forest
Training Accuracy	0.8888	1.0000 (after tuning)
Training Recall	0.8935	1.0000
Training Precision	0.8851	1.0000
Training F1 Score	0.8893	1.0000
Test Accuracy	0.7400	0.7400
Test Recall	0.5946	0.5946
Test Precision	0.4783	0.4783
Test F1 Score	0.5301	0.5301

Conclusion

1. **On training data**
   - Logistic Regression performs **very well** but not perfect.
   - Random Forest (especially after tuning) achieves **perfect fit (100%)**, which is a strong sign of **overfitting**.
2. **On validation/test data**
   - Both models perform **similarly** (Accuracy ≈ 0.74, Recall ≈ 0.59, Precision ≈ 0.48, F1 ≈ 0.53).
   - Neither model generalizes well; Random Forest's extra complexity **did not improve performance** compared to Logistic Regression.
3. **Interpretability vs Complexity**
   - Logistic Regression is **simpler, interpretable, and stable**.
   - Random Forest is more **complex and overfits easily** without improving test results.
4. **Overall**
   - Logistic Regression is a **better baseline** model here: it generalizes almost as well as Random Forest, but with less overfitting risk and easier interpretability.
   - Random Forest needs **stronger regularization/tuning** or may require **additional feature engineering / resampling techniques** to outperform Logistic Regression.