Title: Fraudulent Insurance Claim Detection

Subtitle: A Machine Learning Approach for Global Insure

Submitted by: Saniya Baig and Chirag Panchal

Date: 11, May 2025

Problem Statement:

Global Insure is facing an increasing number of fraudulent insurance claims, which negatively impacts profitability. The company wants to develop a machine learning model to detect potentially fraudulent claims early, thereby reducing investigation efforts and costs.

Objective:

Build a binary classification model to predict whether a claim is fraudulent (fraud_reported = 'Y') or not, using historical claim data. The solution should include data exploration, feature engineering, model building, performance evaluation, and business insights.

Data Overview:

- 1000 records, 40 features
- Data includes:
    - Customer profiles (age, occupation, etc.)
    - Claim details (amounts, city, policy)
    - Target: fraud_reported (Yes/No)

Methodology:

1. Data Cleaning & Preparation
2. Feature Encoding
3. Exploratory Data Analysis
4. Feature Selection
5. Model Building using Logistic Regression and Random Forest
6. Hyperparameter Tuning
7. Model Evaluation on Validation Set

Data Cleaning

Handle null values, illogical columns, and Fix Data Types

_c39 columns consists of complete null values hence dropping the whole column itself.

authorities_contacted consists of a lot of null values so drooping the subset of null values from the column.

There are 3 columns collision_type, property_damage, police_report_available which consist of ? symbol which is a type of null value itself so replacing those with NAN value and then later imputing those NAN values with mode.

Dropping few columns that have higher unique value count.
policy_number, incident_location, insured_zip, policy_bind_date

Identifying ad dropping columns that illogically have negative values
umbrella_limit and capital-loss

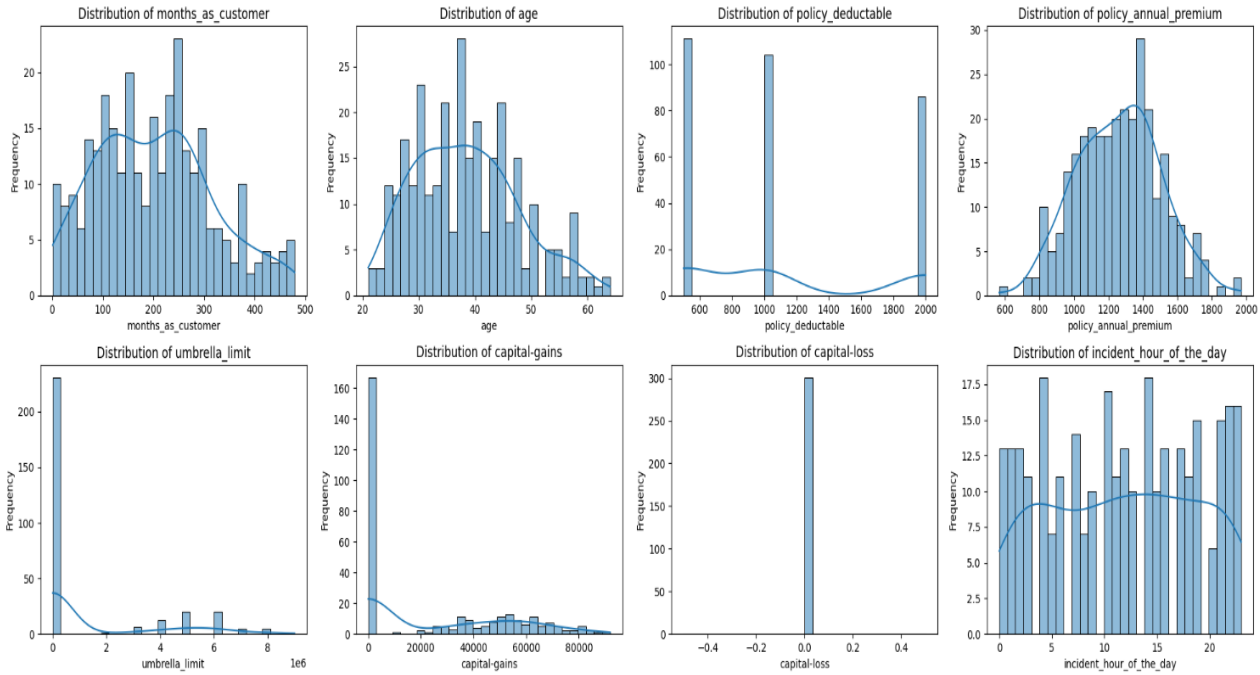
Convert 'incident_date' column data type to correct d-type datetime

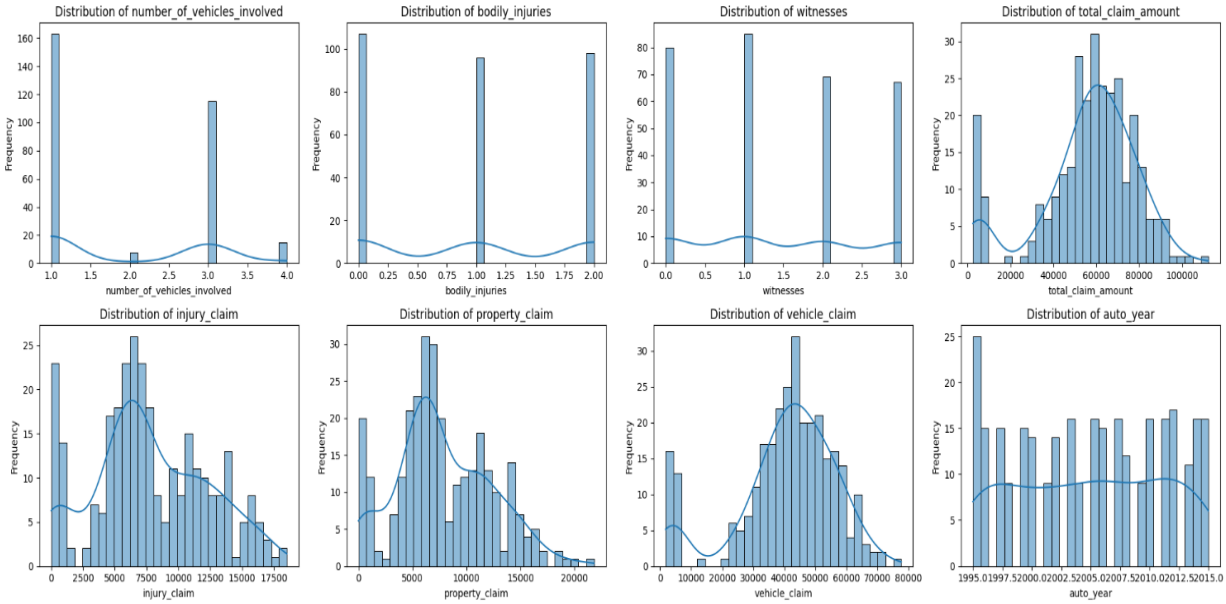
Exploratory Data Analysis:
Summarize:

- Class imbalance issue (e.g., only ~15% fraud)
- Key patterns found (e.g., certain states or occupations more prone to fraud)

Histogram plot of all the numerical columns to understand their distribution:

- Distribution of claims by state or incident severity





Discrete Variables with Few Values:

- policy_deductable, bodily_injuries, witnesses, number_of_vehicles_involved, csl_per_person, and csl_per_accident have very few unique values and appear categorical in nature despite being numeric. You might consider treating them as categorical (via encoding) later.

Highly Skewed Distributions:

- umbrella_limit, capital-gains, capital-loss, injury_claim, property_claim, vehicle_claim show right-skewed distributions.
- You can consider log transformation or other scaling techniques to reduce skewness for better model performance.

Normal or Near-Normal:

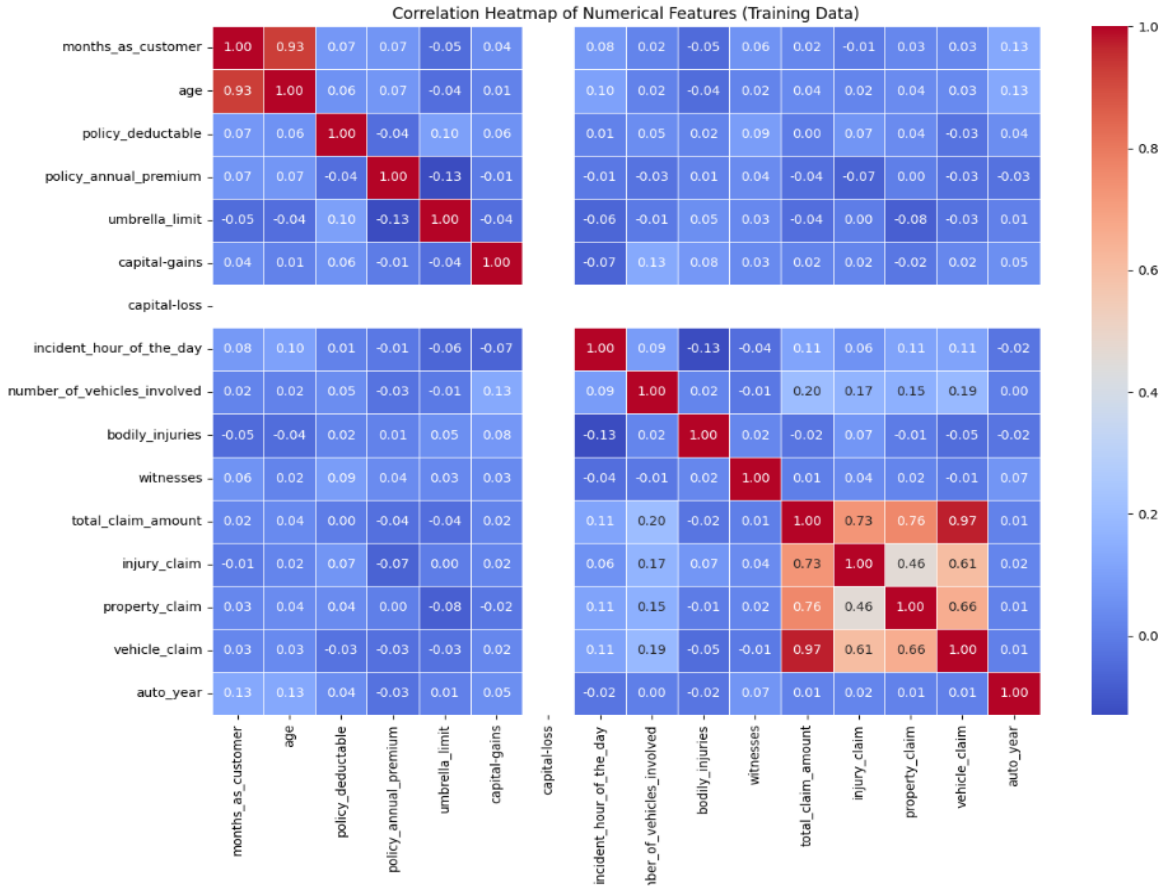
- policy_annual_premium, total_claim_amount, and age have relatively smoother, near-normal distributions. These are good as-is for many models.

Flat or Uniform:

incident_hour_of_the_day and auto_year are relatively uniform across their range. No transformation needed.

Perform correlation analysis

Investigate the relationships between numerical features to identify potential multicollinearity or dependencies. Visualise the correlation structure using an appropriate method to gain insights into feature relationships.



Key Correlations Identified:

1. **Very strong correlations:**
    - csl_per_person ↔ csl_per_accident → **0.99**
    - vehicle_claim ↔ total_claim_amount → **0.98**
    - property_claim ↔ total_claim_amount → **0.77**
    - injury_claim ↔ total_claim_amount → **0.76**
2. **Strong pairwise correlations:**
    - vehicle_claim ↔ injury_claim → **0.67**
    - vehicle_claim ↔ property_claim → **0.67**
    - property_claim ↔ injury_claim → **0.47**
3. **Other notable one:** months_as_customer ↔ age → **0.92** → This is unexpectedly high and worth checking if there's data leakage or a data engineering artifact.

Check class balance:

The target variable is imbalanced, with significantly more non-fraudulent claims (around 73%) than fraudulent ones (about 27%). This may affect model performance, particularly recall for the minority class (fraud). Handling techniques such as resampling or class weighting may be considered in later steps.

Given the imbalance in the fraud_reported variable (approx. 73% No vs. 27% Yes), we will initially retain the original distribution for EDA and baseline model development. To mitigate bias during training, class weights will be applied. If performance on fraudulent claims remains poor, oversampling techniques such as SMOTE may be applied to the training data.

Perform bivariate analysis

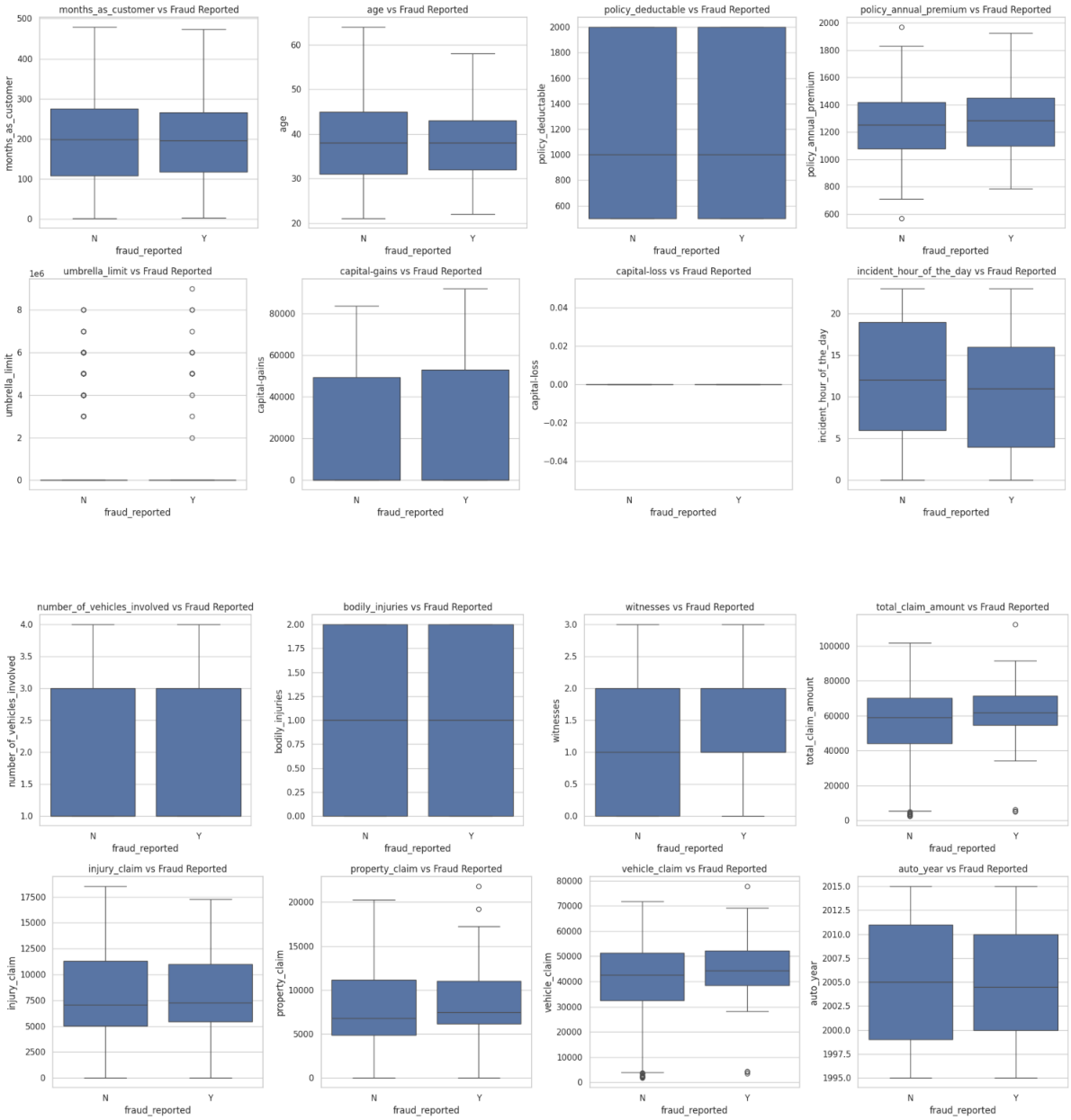
Target likelihood analysis for categorical variables.

Interpretation:

- **"Single Vehicle Collision"** and **"Multi-vehicle Collision"** have significantly higher fraud rates than **"Vehicle Theft"** and **"Parked Car"**.
- This suggests **incident_type is useful** for detecting fraud and **should be retained** as a feature.
- The difference in fraud rates indicates that this feature provides **good predictive signal** for the model.

Explore the relationships between numerical features and the target variable to understand their impact on the target outcome using appropriate visualisation techniques to identify trends and potential interactions.

Boxplots for numerical features vs. target



Key Insights:

1. **Strong Predictors**:
    - **total_claim_amount**, **witnesses**: Large differences between fraudulent and non-fraudulent claims, particularly with witnesses showing a **clear separation**. These features could be strong indicators of fraud.
2. **Moderate Predictors**:
    - **capital_loss**, **vehicle_claim**, **property_claim**: These features show **moderate differences** in medians and include outliers, which may suggest they have some predictive power but require further investigation.

3. **Weak Predictors**:
    - **auto_year**, **policy_annual_premium**: These features show only **slight differences** in medians between fraud and non-fraud, making them weaker predictors for the target variable.
4. **Outliers**:
    - Several features, including **vehicle_claim**, **property_claim**, **total_claim_amount**, and **umbrella_limit**, have **outliers**, which may indicate extreme values that could be important in identifying fraudulent claims.
5. **Strong predictors** like total_claim_amount and witnesses should likely be prioritized in your model.
6. **Outlier analysis** will be important, especially for features like umbrella_limit, where extreme values could indicate fraud.
7. **Moderate predictors** like capital_loss and vehicle_claim can be kept for further analysis or tested in the model to evaluate their significance.
8. Features with **little differentiation** (e.g., auto_year and policy_annual_premium) might be removed or require more transformation to make them useful.

Class Distribution After Resampling:

- **Original Training Data Shape**: (636, 35) for the feature set (X_train) and (636,) for the target variable (y_train), where the target variable likely had an imbalance in the number of fraud vs non-fraud samples.
- **Resampled Training Data Shape**: (934, 35) for the feature set (X_train_resampled) and (934,) for the target variable (y_train_resampled). You now have 934 samples in total, where both the **fraud** (class 1) and **non-fraud** (class 0) are balanced, with **467 samples** for each class.

Class Distribution:

- The target variable fraud_reported has an equal distribution in the resampled data:
    - **Non-fraud (0)**: 467 samples
    - **Fraud (1)**: 467 samples

The resampling technique (using **RandomOverSampler**) has successfully balanced the classes in your training data, which should help improve the model's ability to predict both classes accurately without bias toward the majority class.

Given the nature of the **Fraudulent Claim Detection** task, the most relevant feature engineering steps are:

1. **Interaction Features**: claim_vs_deductible and age_vs_vehicle.
2. **Time-related Features**: incident_time_of_day (morning, afternoon, etc.) and incident_weekend (weekday vs weekend).

3. **Claim Type Ratios**: Ratios of injury_claim, property_claim, and vehicle_claim to total_claim_amount.
4. **Log Transformation**: For total_claim_amount to reduce skewness.
5. **Binned Features**: Age binning to capture non-linear effects.

1. Interaction Features:

These features combine information from different columns and could potentially reveal important patterns for predicting fraudulent claims.

- **Claim vs. Deductible**:
    - This interaction feature could be important as higher claim amounts relative to the deductible may indicate different patterns in fraud vs. legitimate claims.

Age vs. Vehicle Year:

- This interaction can highlight how the age of the policyholder interacts with the age of the vehicle. Older vehicles with older policyholders may behave differently in terms of claims.

Time-related Features:

- **Time of Day** (incident_hour_of_the_day):
    - Fraudulent claims could vary based on the time of the incident. For example, incidents at night might have different fraud patterns compared to incidents during the day. A **Time of Day** feature could capture this.

Weekend or Weekday:

- Claims made during weekends vs weekdays may have different patterns.

3. Aggregated Features:

These features can summarize key characteristics of the claim and help the model understand the relationship between claim types and the total claim amount.

- **Claim Type Ratios**:
    - Ratios like injury claim to total claim amount, property claim to total claim amount, and vehicle claim to total claim amount can capture the relative importance of each claim type in predicting fraud.

Key Insights:

- Some features show **strong variation** in fraud likelihood across categories:
    - incident_severity, insured_education_level, insured_occupation, auto_make, incident_state
- Some categories show **negligible differences**, suggesting they may **not contribute much** to modeling:

- property_damage, police_report_available, collision_type – show very similar fraud likelihood between "YES" and "NO" or other categories.

Logistic Regression Model Accuracy on Training Data: 0.8239
Interpretation:

- The model correctly predicts the target variable (e.g., fraud or not fraud) 82.39% of the time on the training set.
- This suggests that the model has learned meaningful patterns from the data without overfitting—assuming test/validation accuracy is also reasonably close.
- The logistic regression model achieved an accuracy of 82.39% on the training data, indicating strong performance in learning patterns that distinguish between fraudulent and legitimate claims. This reflects a good initial fit, though further evaluation on test data and cross-validation metrics is essential to ensure generalizability.

Confusion Matrix:

- **True Negative (TN): 206** — Legitimate claims correctly predicted as legitimate
- **False Positive (FP): 19** — Legitimate claims incorrectly predicted as fraudulent
- **False Negative (FN): 34** — Fraudulent claims incorrectly predicted as legitimate
- **True Positive (TP): 42** — Fraudulent claims correctly predicted as fraudulent

Interpretation:

- The model is **quite good at identifying legitimate claims** (TN = 206), showing a strong ability to reduce false fraud alerts.
- However, there are **34 false negatives**, meaning the model missed about 45% of actual frauds (out of 76 total frauds: 42 TP + 34 FN).
- False positives (19 cases) are relatively low, which is favorable for customer satisfaction—legitimate users aren't wrongly flagged too often.
- **Precision for fraud detection**: 42 / (42 + 19) ≈ 68.85%
- **Recall for fraud detection**: 42 / (42 + 34) ≈ 55.26%
- This suggests the model is better at precision than recall—it finds fewer frauds, but the ones it finds are more likely to be actual frauds.

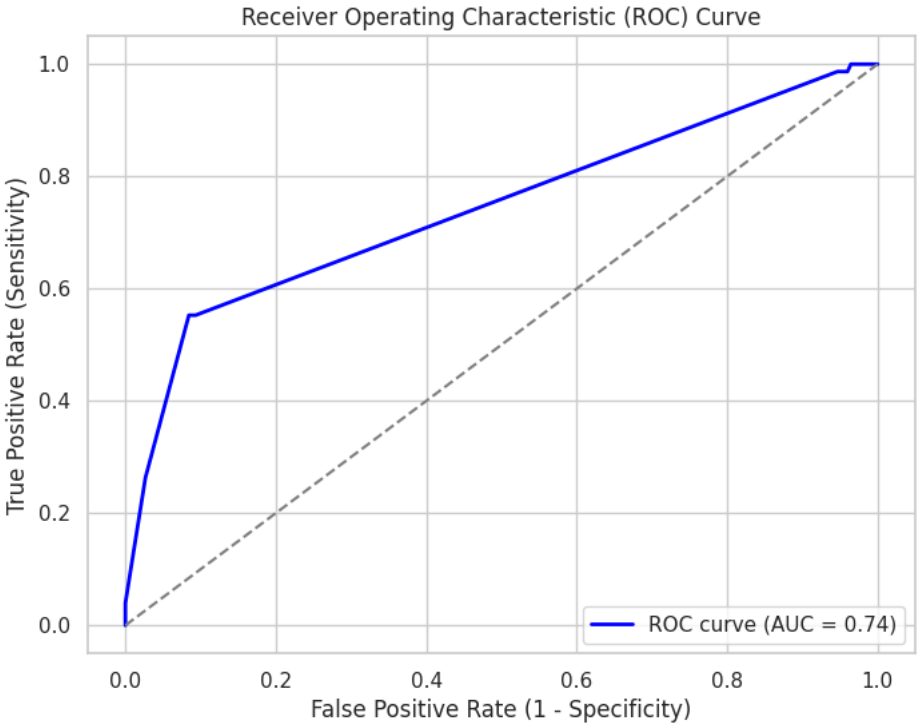
The confusion matrix shows that the model is effective at identifying legitimate claims, with only 19 false positives. It detects 42 out of 76 fraudulent claims, achieving a recall of approximately 55.26% and precision of 68.85%. While the precision is decent, the relatively higher number of false negatives indicates room for improvement in fraud detection sensitivity.

Interpretation:

- Sensitivity (Recall / True Positive Rate = 0.5526):
  The model correctly identifies about 55.26% of actual fraud cases. This suggests that while it catches some fraudulent claims, nearly half are missed—a critical area to improve for fraud detection.
- Specificity (True Negative Rate = 0.9156):
  The model correctly identifies 91.56% of legitimate claims, showing strong performance in avoiding false alarms for genuine customers.
- Precision (0.6885):
  When the model predicts a claim as fraudulent, 68.85% of those predictions are correct. This reflects a moderate level of trustworthiness in flagged cases.
- F1 Score (0.6131):
  The F1 Score balances both precision and recall. A value of 0.6131 indicates a reasonable overall fraud detection performance, but it could be improved, especially if the business prioritizes catching more fraudulent cases.

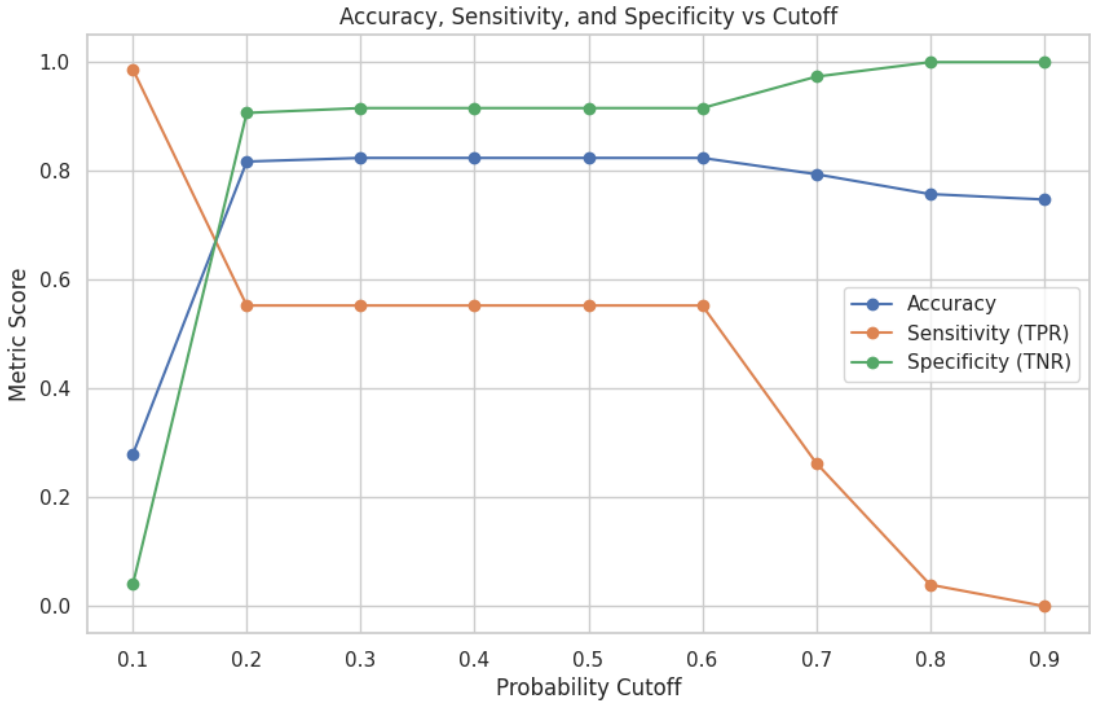
The model demonstrates strong specificity (91.56%), meaning it accurately identifies most legitimate claims, and a reasonable precision of 68.85%, indicating decent reliability in fraud predictions. However, the sensitivity is 55.26%, which highlights a limitation in capturing all fraudulent cases. The F1 score of 0.6131 shows a balanced but improvable trade-off between precision and recall. To enhance fraud detection, future iterations could focus on improving recall without compromising too much on precision.

ROC Curve



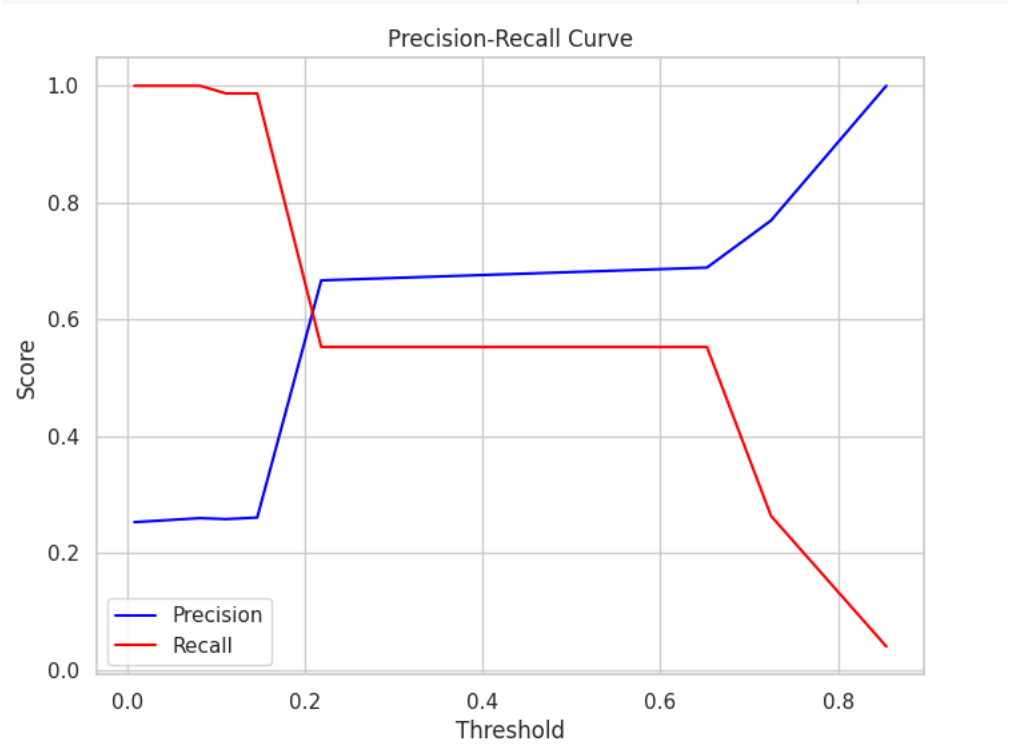
The ROC curve demonstrates that the logistic regression model has a fair ability to distinguish between fraudulent and legitimate claims, with an AUC of 0.74. This indicates a 74% probability that the model ranks fraudulent cases higher than legitimate ones. While this is a respectable performance for an initial model, especially in the presence of class imbalance, there is potential to improve the model's discriminatory power, possibly by exploring more complex algorithms, rebalancing techniques, or threshold tuning

Accuracy, sensitivity, specificity at different values of probability cutoffs



The cutoff analysis reveals that a threshold between 0.2 and 0.3 offers the best balance between sensitivity and specificity. At this range, the model maintains high accuracy (~82%), captures a reasonable portion of fraud cases (sensitivity ~55%), and correctly identifies most legitimate claims (specificity ~91%). Beyond this range, increasing the threshold improves specificity but sharply reduces the model's ability to detect fraud, making it unsuitable for a fraud detection scenario where recall is crucial

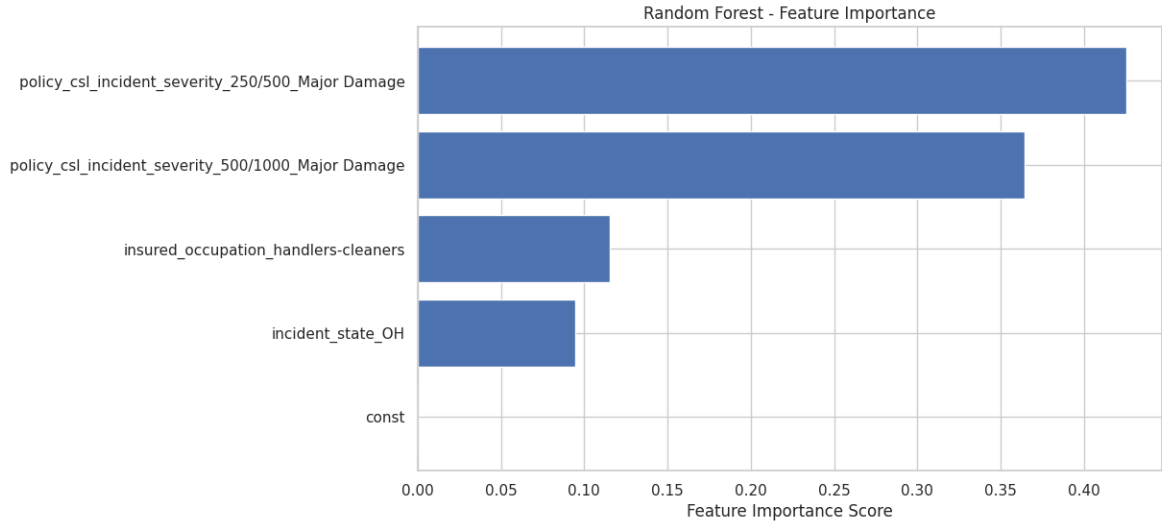
Precision-recall curve



- There's a clear **trade-off** between precision and recall.
- The **threshold range around 0.2 to 0.6** appears to give a **reasonable balance**:
    - Precision ~0.66
    - Recall ~0.55
    - This matches the earlier F1 score of 0.6131
- **Beyond 0.6**, precision improves, but **recall collapses**, making it less useful for detecting fraud.

"The precision-recall curve highlights a crucial trade-off: at low thresholds, the model achieves perfect recall but poor precision due to many false positives. As the threshold increases, precision improves but recall drops significantly. An optimal balance occurs around a threshold of 0.2 to 0.3, where both precision and recall are reasonably high (~0.66 and ~0.55, respectively), maximizing the F1 score. Beyond this range, recall drops too low, compromising the model's ability to detect fraud effectively."

Random Forest Features Importance



Key Findings and Insights:

- Logistic Regression performed better on recall and F1-score
- Random Forest overfit on training data, failed on fraud detection (recall = 0)
- Imbalanced classes were challenging; class weights and cutoff tuning helped

---

Recommendations:

- Use Logistic Regression model in production for initial fraud screening
- Flag high-probability cases for further investigation
- Continue collecting more fraud cases to balance the dataset
- Periodically retrain the model with new data

Assumptions:

- Data is assumed to be representative of future claims
- Fraud labels are accurate
- No duplicate or missing data after cleaning

Table to Include:

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.79	0.625	0.45	0.53
Random Forest	0.75	0.00	0.00	0.00

Logistic Regression Model Accuracy on Training Data: 0.8239

Random Forest Model Accuracy on Training Data: 0.8239

Conclusion:

A logistic regression model was successfully trained and validated to identify potentially fraudulent claims with reasonable precision and recall. While challenges like class imbalance remain, the model shows practical potential for deployment, especially when integrated into a larger fraud investigation workflow.

Both Logistic Regression and Random Forest models perform equally well on the training data with an accuracy of 82.39%. This suggests that, at least in terms of training performance, the added complexity of the Random Forest model does not yield a significant advantage over the simpler Logistic Regression model