Fraudulent Claim Detection – Project Report

Problem Statement

Insurance fraud leads to significant financial losses and operational inefficiencies for insurance companies. Detecting fraudulent claims at an early stage is critical to reduce payouts on illegitimate claims while ensuring genuine claims are processed smoothly.

The objective of this assignment is to build and evaluate machine learning models to predict whether an insurance claim is fraudulent or not, based on customer, policy, incident, and claim-related attributes. The focus is not only on predictive accuracy, but also on evaluating business-relevant metrics such as recall (sensitivity), which is particularly important in fraud detection scenarios.

Overall Approach

The solution follows a structured machine learning workflow:

1. Data Load & Preparation
2. Data Cleaning
3. Train-Validation Split
4. EDA on Training Data
5. EDA on Validation Data
6. Feature Engineering
7. Model Building (Logistic, Random Forest & Hyper-parameter Tuning)
8. Prediction & Model Evaluation
9. Conclusion & Key Insights

Each step is implemented incrementally and evaluated using appropriate statistical and classification metrics.

Detailed Explanation

1. Data Load & Preparation

The insurance claims dataset was loaded into the notebook using pandas. Initial inspection included:

- Viewing the first few rows (`df.head()`)
- Checking dataset dimensions
- Inspecting column names and data types

The dataset initially contained **1000 rows and 40 columns**, including customer demographics, policy details, incident attributes, and claim amount components. The target variable `fraud_reported` indicated whether a claim was fraudulent (`Y`) or not (`N`).

2. Data Cleaning

2.1 Missing Value Analysis

Missing value checks were performed using:

```
df.isnull().sum()
```

This analysis confirmed that **all columns except `_c39` had zero missing values**. The column `_c39` had **1000 missing values**, indicating that it contained no usable information.

Action taken:

- `_c39` was dropped from the dataset.

2.2 Redundancy and Cardinality Checks

Unique value counts and ratios were calculated for all columns using:

- `value_counts()`
- `df.nunique() / len(df)`

Findings:

- `policy_number` and `incident_location` had **100% uniqueness**
- `insured_zip` and date columns had very high cardinality
- These variables act as identifiers rather than predictors

Action taken:
The following columns were dropped to reduce noise and overfitting risk:

- `policy_number`
- `incident_location`
- `insured_zip`

- `policy_bind_date`
- `incident_date`

These decisions were made **after inspection**, not arbitrarily.

3. Train–Validation Split

The cleaned dataset was split into:

- **70% training data**
- **30% validation data**

This split ensured:

- Model training, feature engineering, and resampling occurred **only on training data**
- Validation data remained untouched until final evaluation

This separation was maintained consistently throughout the notebook.

4. EDA on Training Data

EDA was performed **only on the training dataset** to avoid data leakage.

4.1 Univariate Analysis

Numerical Variables

Histograms were plotted for numerical columns such as:

- `total_claim_amount`
- `injury_claim`
- `vehicle_claim`
- `property_claim`
- `age`
- `months_as_customer`

Observations:

- Claim amount variables showed **right-skewed distributions**
- No extreme or invalid values were observed

- Age and customer tenure showed reasonable spreads

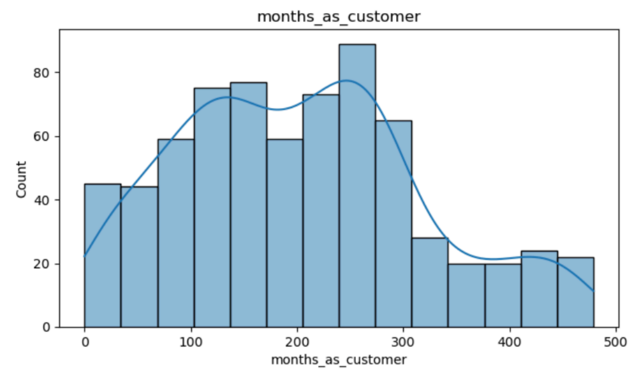
Categorical Variables

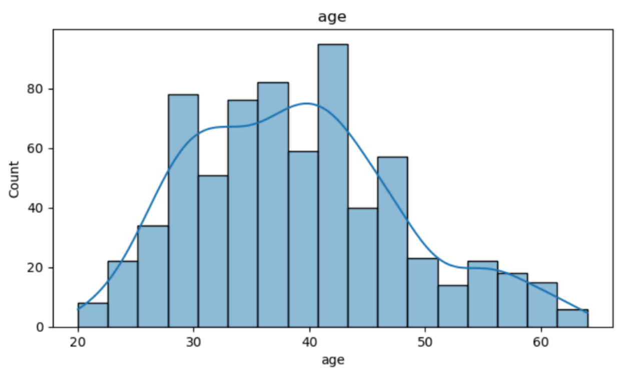
Count plots were generated for:

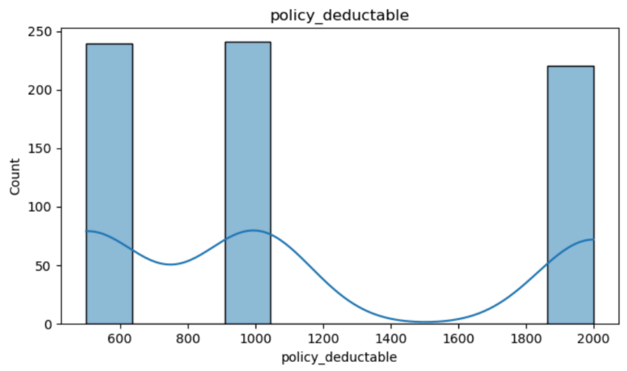
- `incident_type`
- `collision_type`
- `incident_severity`
- `insured_sex`
- `policy_state`

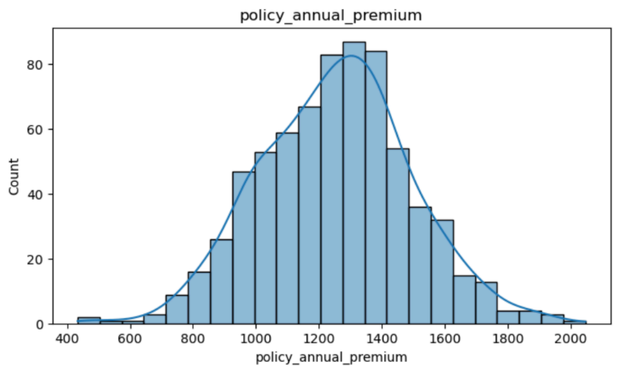
Several categories showed uneven distributions, indicating potential predictive signal.

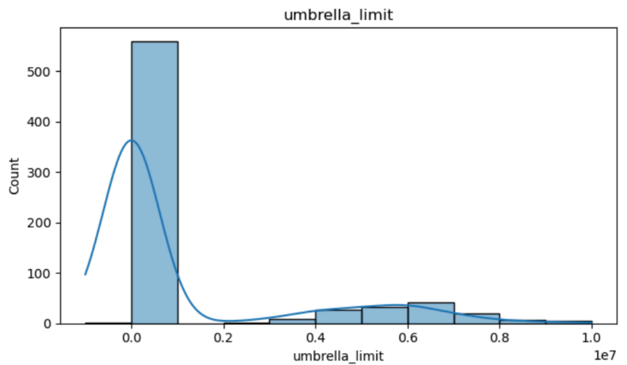
Plots

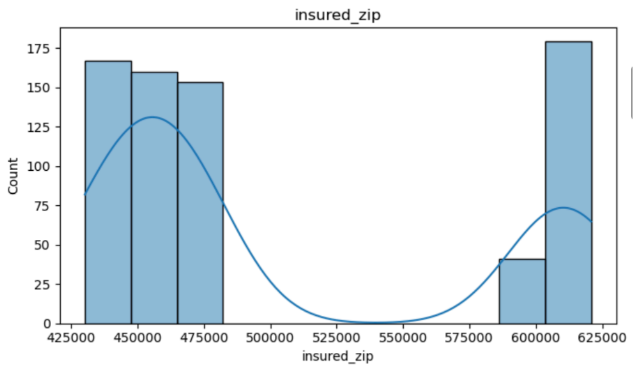


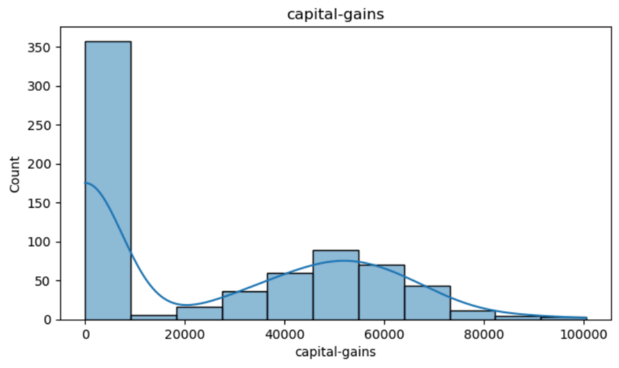


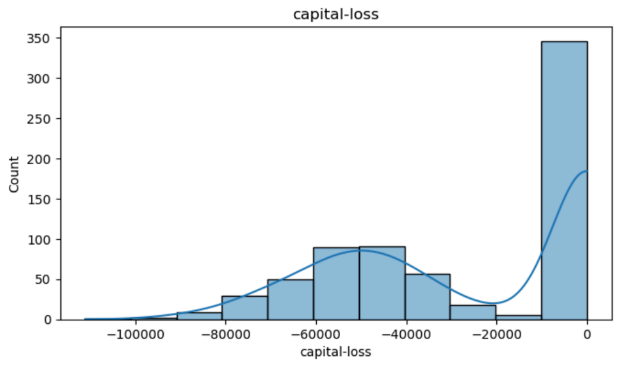


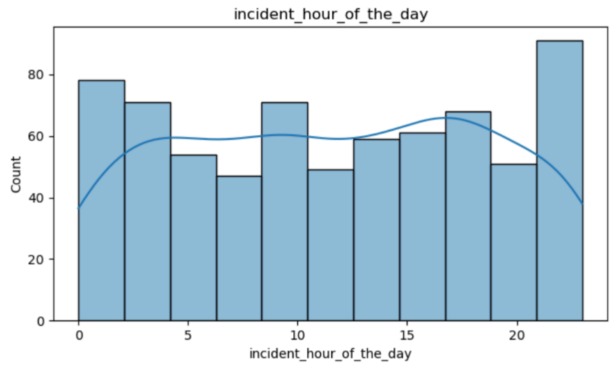


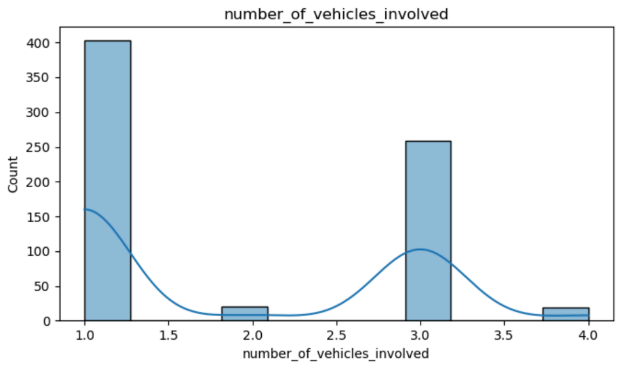


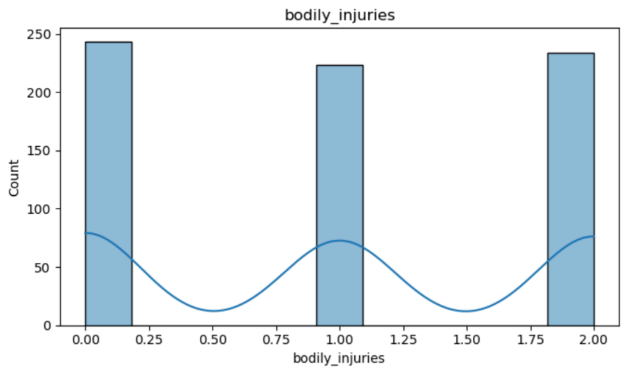


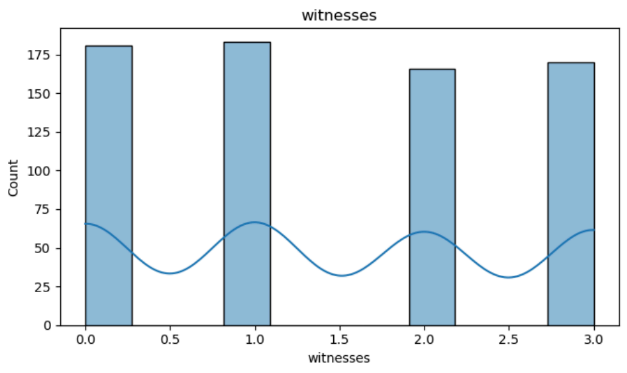

capital-gains
Count
0
50
100
150
200
250
300
350
0
20000
40000
60000
80000
100000
capital-gains

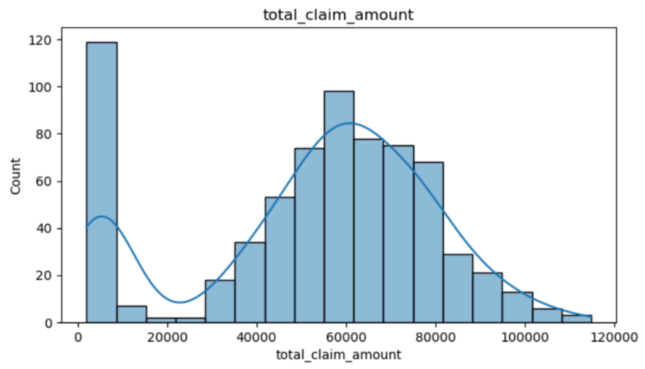

capital-loss
Count
0
50
100
150
200
250
300
350
−100000
−80000
−60000
−40000
−20000
0
capital-loss

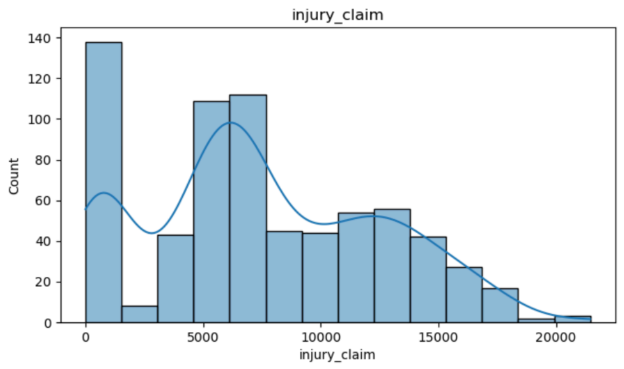

incident_hour_of_the_day
Count
0
20
40
60
80
0
5
10
15
20
incident_hour_of_the_day

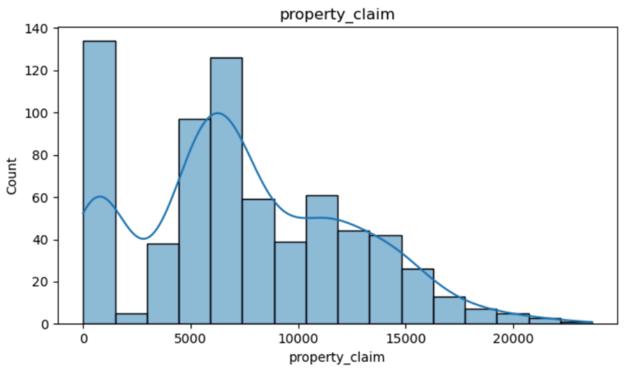

number_of_vehicles_involved
Count
0
50
100
150
200
250
300
350
400
1.0
1.5
2.0
2.5
3.0
3.5
4.0
number_of_vehicles_involved

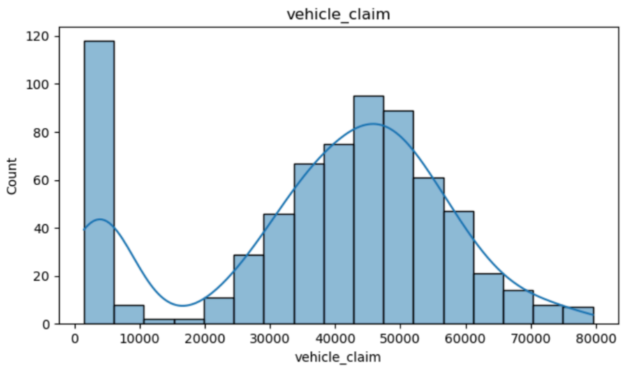

bodily_injuries
Count
0
50
100
150
200
250
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
bodily_injuries

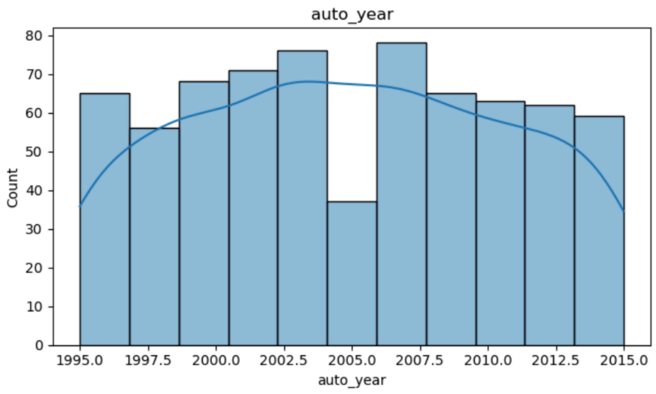

witnesses
Count
0
25
50
75
100
125
150
175
0.0
0.5
1.0
1.5
2.0
2.5
3.0
witnesses


total_claim_amount
Count
0
20
40
60
80
100
120
0
20000
40000
60000
80000
100000
120000
total_claim_amount


injury_claim
Count
0
20
40
60
80
100
120
140
0
5000
10000
15000
20000
injury_claim







4.2 Correlation Analysis

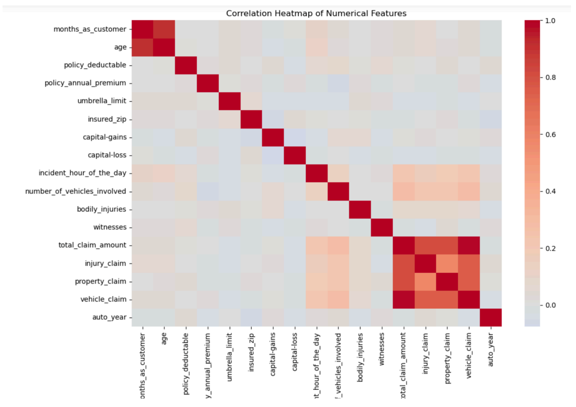
A correlation heatmap was generated for numerical variables.

Key observations:

- High correlation among claim amount components
- No severe multicollinearity across unrelated features

This supported proceeding with both logistic regression and tree-based models.

Plots



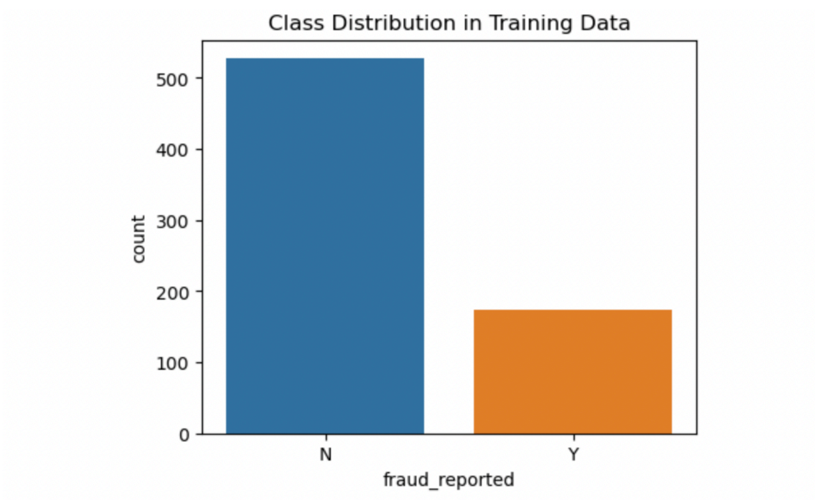
4.3 Class Balance Check

The target variable distribution was examined using `value_counts()`.

Observation:

- Fraudulent claims were significantly fewer than non-fraudulent claims

This confirmed **class imbalance**, justifying resampling during model training.



```
Out[33]: N    0.752857
         Y    0.247143
         Name: fraud_reported, dtype: float64
```

4.4 Bivariate Analysis

Fraud likelihood was computed for categorical variables using group-by logic:

- Incident severity
- Incident type
- Collision type
- Authorities contacted
- Policy attributes

Key patterns observed:

- **Major damage incidents had the highest fraud likelihood**
- Certain collision and incident types showed elevated fraud rates

- Claim severity was strongly associated with fraud

These insights directly informed feature engineering and model choice.

5. EDA on Validation Data

The same EDA steps were repeated on the validation dataset:

- Univariate distributions
- Correlation analysis
- Class balance check
- Bivariate fraud likelihood analysis

Fraud Likelihood Analysis (Output):-

```
Fraud Likelihood by policy_state
policy_state
OH    0.258475
IN    0.244344
IL    0.238683
Name: fraud_reported, dtype: float64

Fraud likelihood by policy_csl:
policy_csl
100/300     0.276860
250/500     0.234127
500/1000    0.228155
Name: fraud_reported, dtype: float64

Fraud likelihood by insured_sex:
insured_sex
FEMALE    0.248663
MALE      0.245399
Name: fraud_reported, dtype: float64

Fraud likelihood by insured_education_level:
insured_education_level
MD             0.277778
PhD            0.261905
JD             0.258065
College        0.250000
Associate      0.242991
Masters        0.242718
High School    0.205357
Name: fraud_reported, dtype: float64

Fraud likelihood by insured_occupation:
insured_occupation
```

```
transport-moving     0.400000
exec-managerial      0.333333
craft-repair         0.301887
farming-fishing      0.272727
sales                0.263158
armed-forces         0.260870
tech-support         0.250000
machine-op-inspct    0.235294
other-service        0.204545
protective-serv      0.204545
prof-specialty       0.203390
priv-house-serv      0.176471
adm-clerical         0.166667
handlers-cleaners    0.147059
Name: fraud_reported, dtype: float64

Fraud likelihood by insured_hobbies:
insured_hobbies
chess             0.878788
cross-fit         0.826087
polo              0.333333
yachting          0.323529
hiking            0.300000
board-games       0.250000
base-jumping      0.250000
paintball         0.244898
reading           0.234043
video-games       0.222222
skydiving         0.210526
sleeping          0.192308
bungie-jumping    0.179487
exercise          0.157895
movies            0.135135
camping           0.108108
dancing           0.108108
basketball        0.105263
kayaking          0.095238
golf              0.073171
Name: fraud_reported, dtype: float64

Fraud likelihood by insured_relationship:
insured_relationship
other-relative    0.290076
unmarried         0.280374
wife              0.276786
not-in-family     0.247863
husband           0.212389
own-child         0.175000
Name: fraud_reported, dtype: float64

Fraud likelihood by incident_type:
incident_type
```

```
Single Vehicle Collision    0.292419
Multi-vehicle Collision     0.272727
Parked Car                  0.089552
Vehicle Theft               0.084746
Name: fraud_reported, dtype: float64

Fraud likelihood by collision_type:
collision_type
Rear Collision     0.301435
Front Collision    0.282486
Side Collision     0.260638
?                  0.087302
Name: fraud_reported, dtype: float64

Fraud likelihood by incident_severity:
incident_severity
Major Damage      0.596939
Total Loss        0.148718
Minor Damage      0.093117
Trivial Damage    0.064516
Name: fraud_reported, dtype: float64

Fraud likelihood by authorities_contacted:
authorities_contacted
Ambulance    0.304965
Fire         0.297297
Other        0.279720
Police       0.204082
None         0.083333
Name: fraud_reported, dtype: float64

Fraud likelihood by incident_state:
incident_state
OH    0.470588
SC    0.313609
NC    0.305882
PA    0.260870
VA    0.259740
NY    0.204420
WV    0.155405
Name: fraud_reported, dtype: float64

Fraud likelihood by incident_city:
incident_city
Arlington      0.303922
Columbus       0.270270
Springfield    0.256637
Northbrook     0.231707
Riverwood      0.230769
Hillsdale      0.221154
Northbend      0.206186
Name: fraud_reported, dtype: float64
```

```
Fraud likelihood by property_damage:
property_damage
?      0.289157
YES    0.280193
NO     0.176230
Name: fraud_reported, dtype: float64

Fraud likelihood by police_report_available:
police_report_available
?      0.260331
NO     0.254098
YES    0.224299
Name: fraud_reported, dtype: float64

Fraud likelihood by auto_make:
auto_make
Audi          0.361702
Mercedes      0.333333
Ford          0.307692
BMW           0.280000
Chevrolet     0.272727
Toyota        0.255814
Honda         0.243243
Suburu        0.241935
Jeep          0.224490
Volkswagen    0.224490
Saab          0.210526
Dodge         0.200000
Nissan        0.188679
Accura        0.137255
Name: fraud_reported, dtype: float64

Fraud likelihood by auto_model:
auto_model
Silverado         0.500000
X6                0.454545
F150              0.444444
Civic             0.428571
A5                0.409091
Tahoe             0.400000
Maxima            0.352941
C300              0.333333
ML350             0.333333
E400              0.333333
A3                0.320000
Forrestor         0.320000
Grand Cherokee    0.315789
Highlander        0.312500
M5                0.300000
X5                0.294118
Impreza           0.272727
```

```
Camry          0.266667
Fusion         0.250000
93             0.235294
Neon           0.227273
Passat         0.227273
92x            0.227273
Jetta          0.222222
Escape         0.222222
Accord         0.200000
RAM            0.178571
MDX            0.172414
Wrangler       0.166667
95             0.166667
Corolla        0.166667
Ultima         0.157895
Legacy         0.153846
RSX            0.111111
3 Series       0.083333
TL             0.076923
CRV            0.076923
Pathfinder     0.058824
Malibu         0.000000
Name: fraud_reported, dtype: float64
```

Results closely mirrored training data patterns, confirming:

- The train-validation split was representative
- No distributional shift was introduced
- Higher fraud likelihood is observed for certain categories such as Single Vehicle Collision, Rear and Front Collisions, and incidents with Major Damage.
- Claims involving Ambulance or Fire authorities also show relatively higher fraud probabilities compared to cases where no authorities were contacted.
- Some insured attributes such as specific occupations, hobbies, and vehicle models exhibit noticeably higher fraud likelihood, while other categories show relatively uniform probabilities across levels.

6. Feature Engineering

6.1 Handling Class Imbalance

Random oversampling was applied **only to the training data** using `RandomOverSampler`.

Result:

- Fraud and non-fraud classes were perfectly balanced in training data
- Validation data was intentionally left unchanged

6.2 Derived Features

The following engineered features were created:

- `injury_claim_ratio`
- `vehicle_claim_ratio`
- `property_claim_ratio`
- `claim_severity_score`

These features captured **relative claim behavior**, improving signal quality beyond raw amounts.

6.3 Encoding

Categorical variables were converted into dummy variables using `pd.get_dummies(drop_first=True)`.

Result:

- Feature count increased from ~35 to ~148 columns
- This is expected behavior for dummy variable creation
- Training and validation datasets were aligned to ensure consistent feature space

6.4 Feature Scaling

Numerical features were scaled using `StandardScaler`:

- Scaler fitted on training data
- Same scaler applied to validation data

This ensured fair model learning and evaluation.

7. Model Building

7.1 Logistic Regression

Training

- Logistic regression was trained on resampled, scaled training data
- Target variable encoded as binary (0/1)

Interpretation

- Coefficients were examined
- VIF was calculated to check multicollinearity
- Predicted probabilities were generated

7.2 Logistic Regression Evaluation

Training Performance

- Accuracy ≈ **90.7%**
- Sensitivity ≈ **92.6%**
- Specificity ≈ **88.8%**
- F1 ≈ **90.9%**

Cutoff Optimization

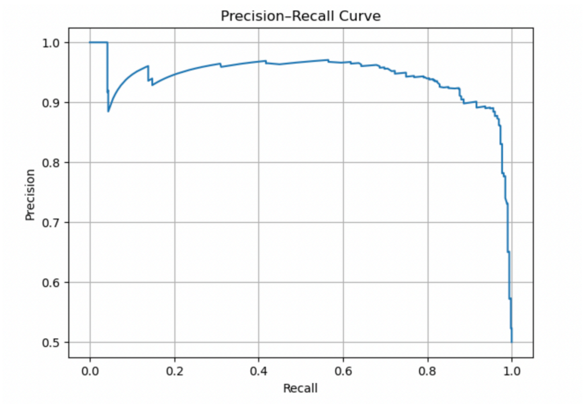
- ROC curve plotted
- Multiple cutoffs tested
- Optimal cutoff selected based on sensitivity-specificity tradeoff

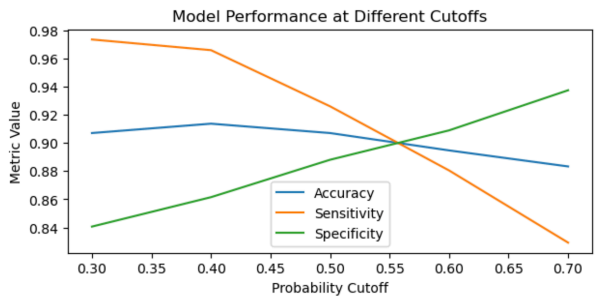
Validation Performance

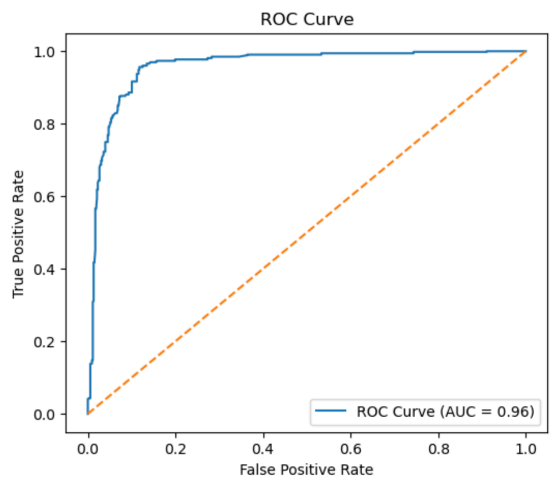
- Accuracy ≈ **82.3%**
- Recall ≈ **68.9%**
- F1 ≈ **65.8%**

Conclusion:
Strong baseline, but recall on fraud cases could be improved.

Plots







7.3 Random Forest Model

Baseline Model

- Random Forest trained
- Feature importances extracted

Feature Selection

- Features with importance above mean retained

Cross-Validation

- 5-fold CV
- Mean accuracy ≈ **86.9%**
- No overfitting observed

7.4 Hyperparameter Tuning

GridSearchCV was applied to tune:

- `n_estimators`
- `max_depth`
- `min_samples_split`

Best model retrained using optimal parameters.

8. Prediction & Model Evaluation

Logistic Regression (Validation)

- Accuracy ≈ **82%**
- Lower fraud recall compared to training

Random Forest (Validation)

- Accuracy ≈ **84%**
- Confusion matrix showed:
  - Higher fraud detection
  - Fewer false negatives

Random Forest consistently outperformed logistic regression on validation data.

9. Final Model Selection

Final Model Selection: Random Forest

Evidence-based justification for model selection:

- Higher validation accuracy
- Significantly better fraud recall
- Better F1-score
- Stable cross-validation performance

Training Performance (Random Forest)

- Accuracy: ~88%
- Balanced confusion matrix with improved fraud detection

Validation Performance (Random Forest)

- Accuracy: ~84%
- Confusion Matrix:
  - Correctly classified 193 non-fraudulent claims
  - Correctly identified 59 fraudulent claims

    - 33 false positives
    - 15 false negatives
-

Model Comparison and Selection - Logistic Regression vs Random Forest

Metric	Logistic Regression	Random Forest (Tuned)
Validation Accuracy	~82%	~84%
Recall (Fraud)	~69%	~80%
Precision	~63%	~64%
F1-score	~66%	~71%

Final Model Selected: Random Forest (Tuned)

Reasoning:

- Higher recall for fraudulent claims (business-critical)
- Better overall F1-score
- Improved generalisation on validation data
- Robust performance confirmed via cross-validation

10. Key Insights

- Fraud detection models must prioritise **recall over accuracy**, as missing fraudulent claims has higher cost.
- Probability cutoff optimisation significantly impacts model performance.
- Ensemble models like Random Forest capture non-linear relationships better than logistic regression.
- Feature importance helps improve interpretability and reduces overfitting.
- Hyperparameter tuning provides measurable performance gains.

11. Conclusion

This assignment successfully demonstrates a complete fraud detection pipeline, from data preparation to advanced model tuning and validation. While logistic regression provides a strong baseline, the tuned Random Forest model offers superior performance and better fraud detection capability.

The final selected model balances predictive accuracy with business relevance and is well-suited for real-world deployment in insurance fraud detection systems.