Fraudulent Claim Detection Report

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 Tasks

Data Dictionary

1. Data Preparation

1.1 Load the Data

2. Data Cleaning

2.1 Handle null values

2.1.1 Examine the columns to determine if any value or column needs to be treated

1. `authorities_contacted`

2. `_c39`

```
These columns have null value
```

2.1.2 Handle rows containing null values

1. _c39 all values in this column is null so this column is dropped
2. df['authorities_contacted'] = df['authorities_contacted'].fillna('Unknown')

   filled with a value named ‘Unknown’

2.2 Identify and handle redundant values and columns

2.2.1 Examine the columns to determine if any value or column needs to be treated

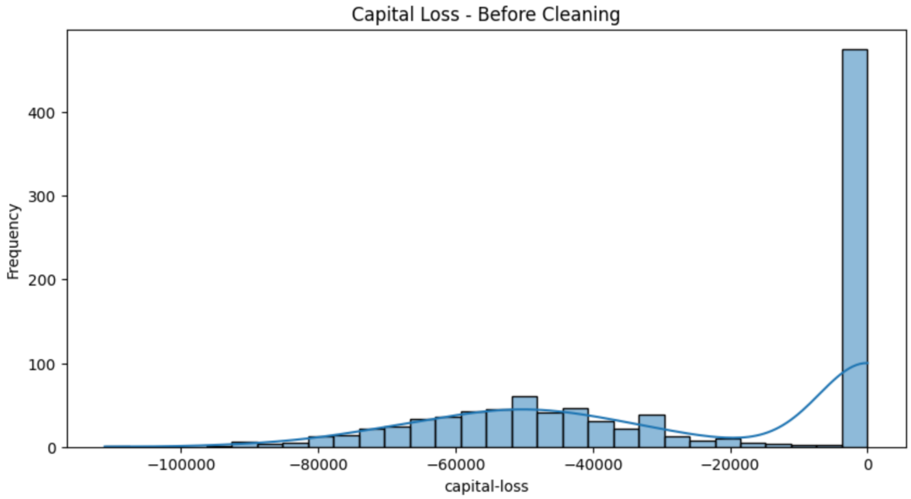
1. 'months_as_customer', 'age', 'capital-gains', 'capital-loss',
2. 'number_of_vehicles_involved', 'witnesses',
3. 'total_claim_amount', 'injury_claim',
4. 'property_claim', 'vehicle_claim'

   All these columns must have positive values but capital loss is negative

.2.2 Identify and drop any columns that are completely empty

- _c39 is a impact less column so its dropped

2.2.3 Identify and drop rows where features have illogical or invalid values, such as negative values for features that should only have positive



The capital loss column mostly has negative values so they can be the same.

2.2.4 Identify and remove columns where a large proportion of the values are unique or near unique, as these columns are likely to be identifiers or have very limited predictive power.

2.3 Fix Data Types

Fixed the data types of time series data and numerical columns.

3. Train-Validation Split

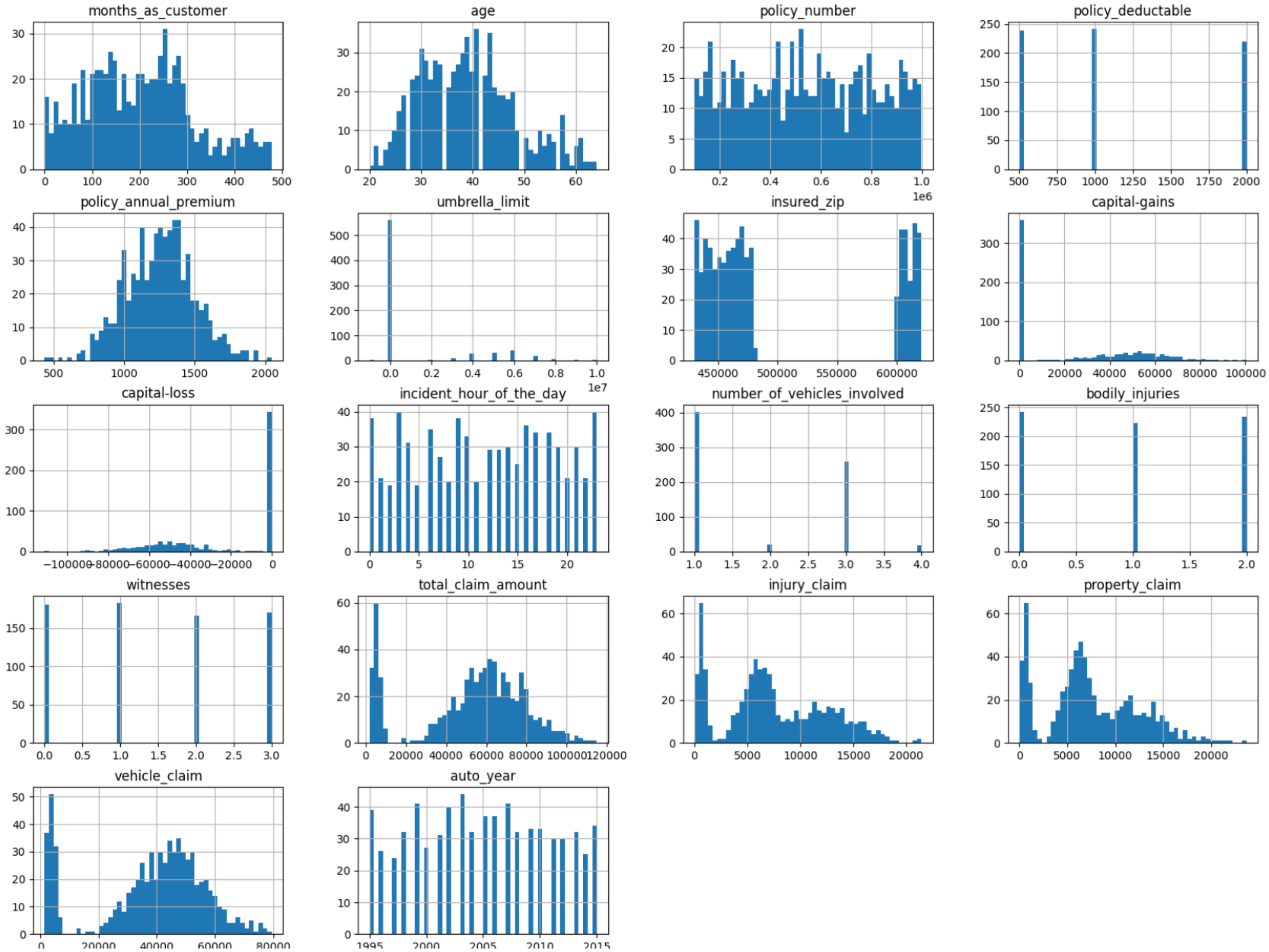
4. EDA on training data

4.1 Perform univariate analysis

4.1.1 Identify and select numerical columns from training data for univariate analysis

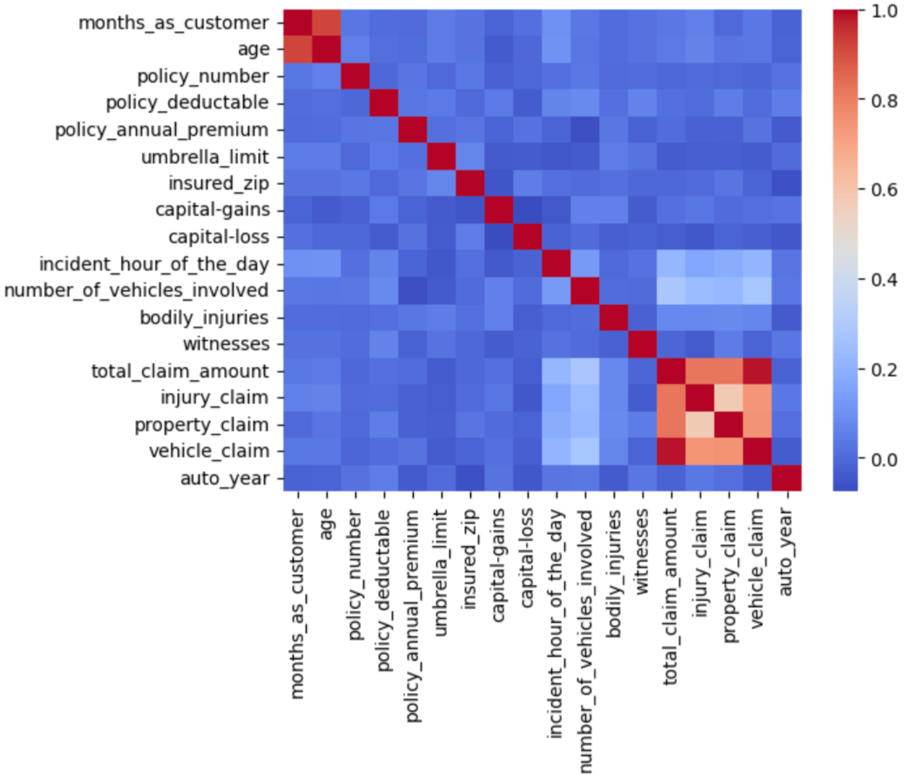
4.1.2 Visualise the distribution of selected numerical features using appropriate plots to understand their characteristics

```
plt.show()
```

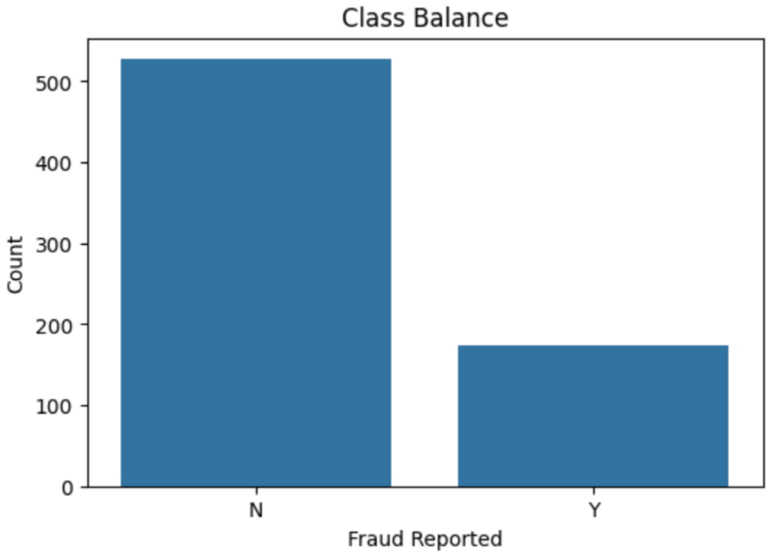

months_as_customer
0 10 20 30
0 100 200 300 400 500
age
0 10 20 30
20 30 40 50 60
policy_number
0 5 10 15 20
0.2 0.4 0.6 0.8 1.0
1e6
policy_deductable
0 50 100 150 200 250
500 750 1000 1250 1500 1750 2000
policy_annual_premium
0 10 20 30 40
500 1000 1500 2000
umbrella_limit
0 100 200 300 400 500
0.0 0.2 0.4 0.6 0.8 1.0
1e7
insured_zip
0 10 20 30 40
450000 500000 550000 600000
capital-gains
0 100 200 300
0 20000 40000 60000 80000 100000
capital-loss
0 100 200 300
−100000−80000−60000−40000−20000 0
incident_hour_of_the_day
0 10 20 30 40
0 5 10 15 20
number_of_vehicles_involved
0 100 200 300 400
1.0 1.5 2.0 2.5 3.0 3.5 4.0
bodily_injuries
0 50 100 150 200 250
0.0 0.5 1.0 1.5 2.0
witnesses
0 50 100 150
0.0 0.5 1.0 1.5 2.0 2.5 3.0
total_claim_amount
0 20 40 60
0 20000 40000 60000 80000 100000120000
injury_claim
0 20 40 60
0 5000 10000 15000 20000
property_claim
0 20 40 60
0 5000 10000 15000 20000
vehicle_claim
0 10 20 30 40 50
0 20000 40000 60000 80000
auto_year
0 10 20 30 40
1995 2000 2005 2010 2015

4.2 Perform correlation analysis

```
<Axes: >
```

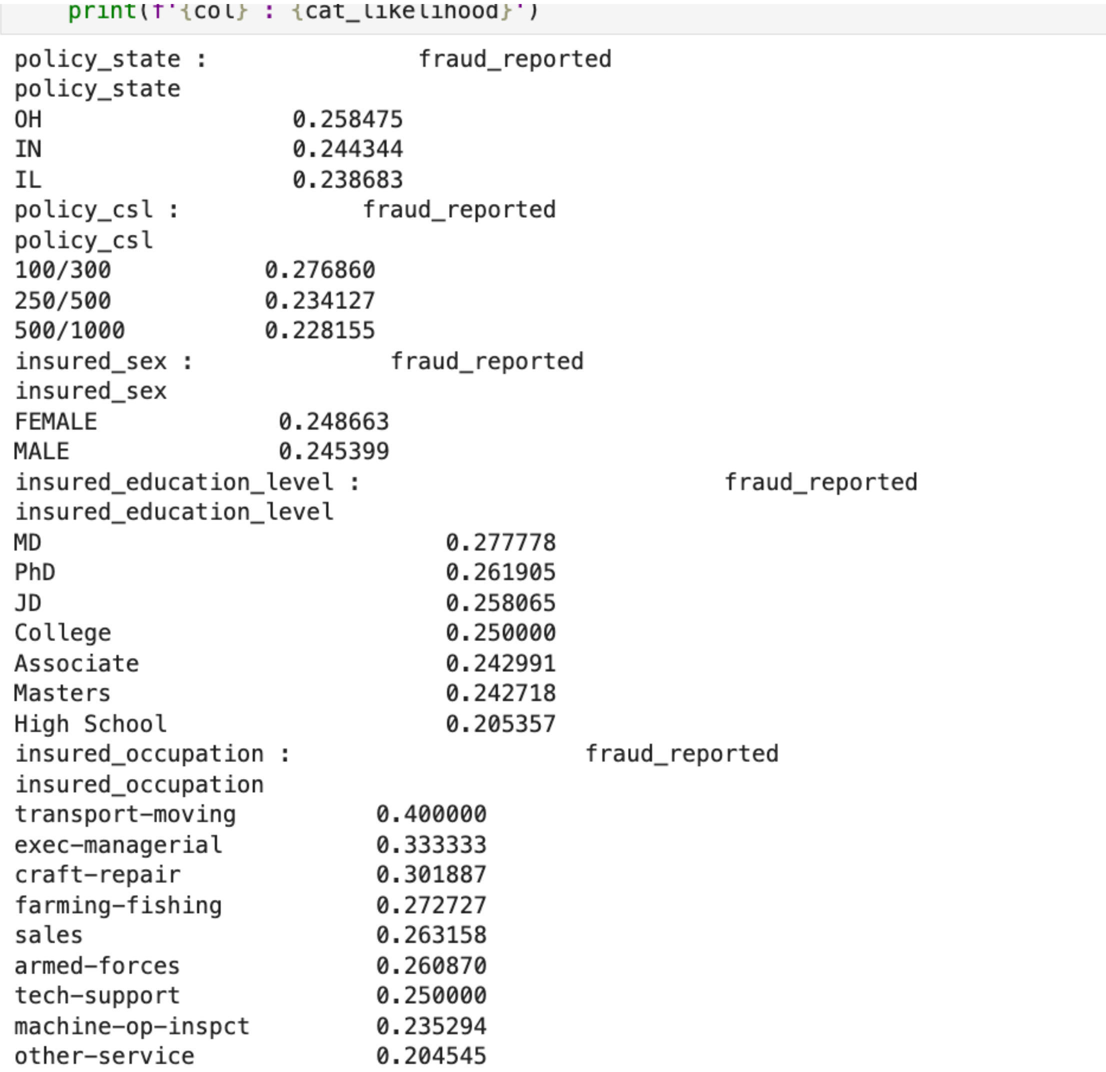


4.3 Check class balance



4.4 Perform bivariate analysis

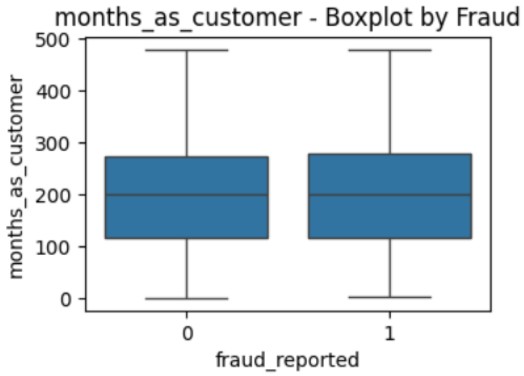
4.4.1Target likelihood analysis for categorical variables.

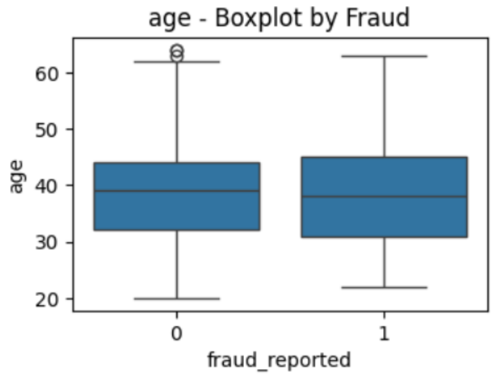


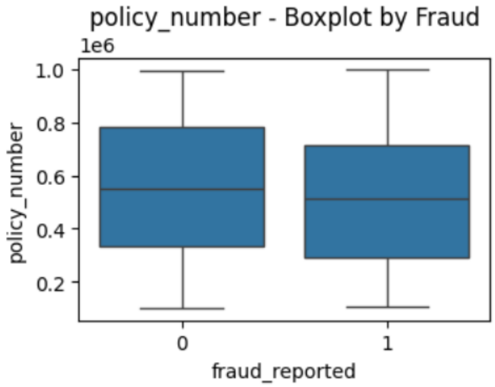
```
    print(f'{col} : {cat_likelihood}')
```

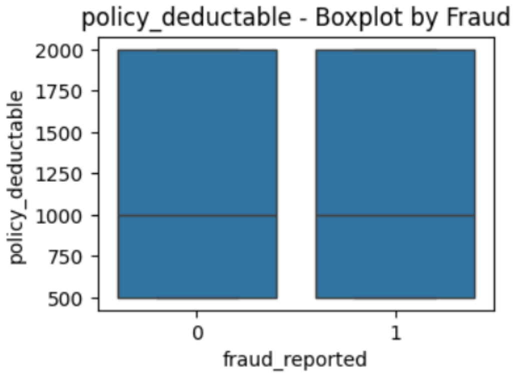
```
policy_state :               fraud_reported
policy_state
OH                  0.258475
IN                  0.244344
IL                  0.238683
policy_csl :               fraud_reported
policy_csl
100/300           0.276860
250/500           0.234127
500/1000          0.228155
insured_sex :               fraud_reported
insured_sex
FEMALE              0.248663
MALE                0.245399
insured_education_level :                          fraud_reported
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
insured_occupation :                         fraud_reported
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
```

4.4.2 Explore the relationships between numerical features and the target variable to understand their impact on the target outcome using appropriate visualisation techniques









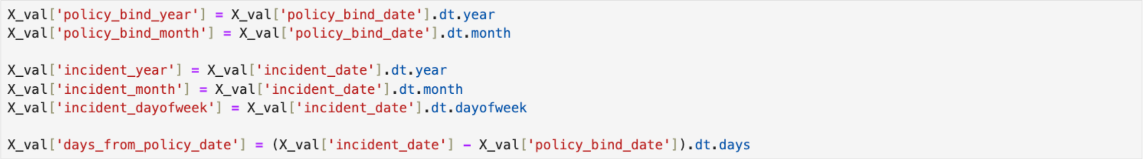
6. Feature Engineering

6.1 Perform resampling

Random Over Sampler technique to balance the data and handle class imbalance. This method increases the number of samples in the minority class by randomly duplicating them, creating synthetic data points with similar characteristics. This helps prevent the model from being biased toward the majority class and improves its ability to predict the minority class more accurately.

6.2 Feature Creation

Extracted meaningful features from time series columns and other meaning full features from other columns



```
X_val['policy_bind_year'] = X_val['policy_bind_date'].dt.year
X_val['policy_bind_month'] = X_val['policy_bind_date'].dt.month

X_val['incident_year'] = X_val['incident_date'].dt.year
X_val['incident_month'] = X_val['incident_date'].dt.month
X_val['incident_dayofweek'] = X_val['incident_date'].dt.dayofweek

X_val['days_from_policy_date'] = (X_val['incident_date'] - X_val['policy_bind_date']).dt.days
```

6.3 Handle redundant columns

- Features that don't strongly influence the prediction, which you may have observed during the EDA phase.

- Categorical columns with low value counts for some levels can be remapped to reduce number of unique levels, and features with very high counts for just one level may be removed, as they resemble unique identifier columns and do not provide substantial predictive value.

- Additionally, eliminate any columns from which the necessary features have already been extracted in the preceding step.

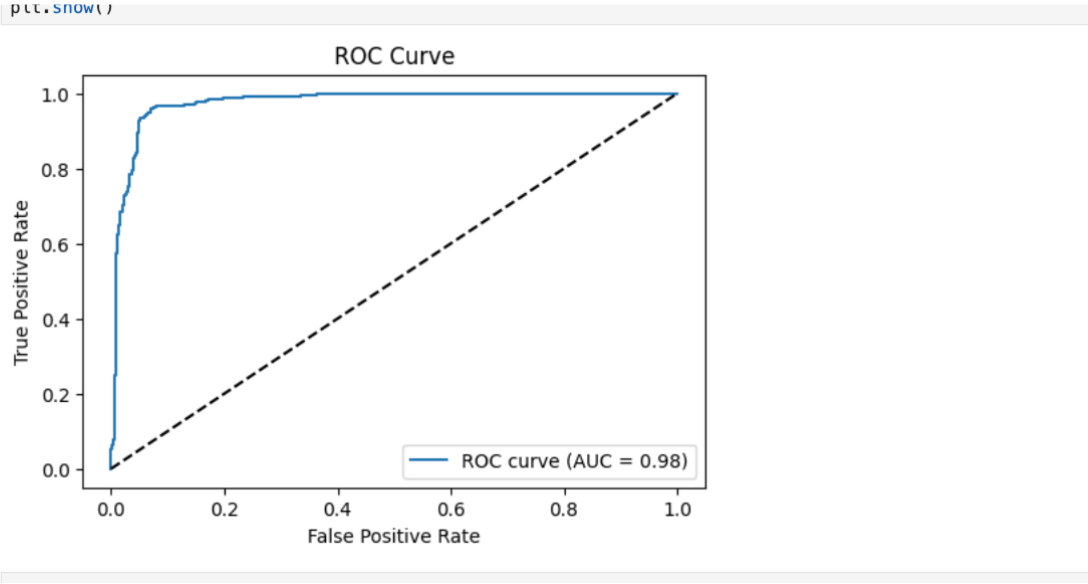
6.4 Combine values in Categorical Columns

During the EDA process, categorical columns with multiple unique values may be identified. To enhance model performance, it is essential to refine these categorical features by grouping values that have low frequency or provide limited predictive information.

Combine categories that occur infrequently or exhibit similar behaviour to reduce sparsity and improve model generalisation.

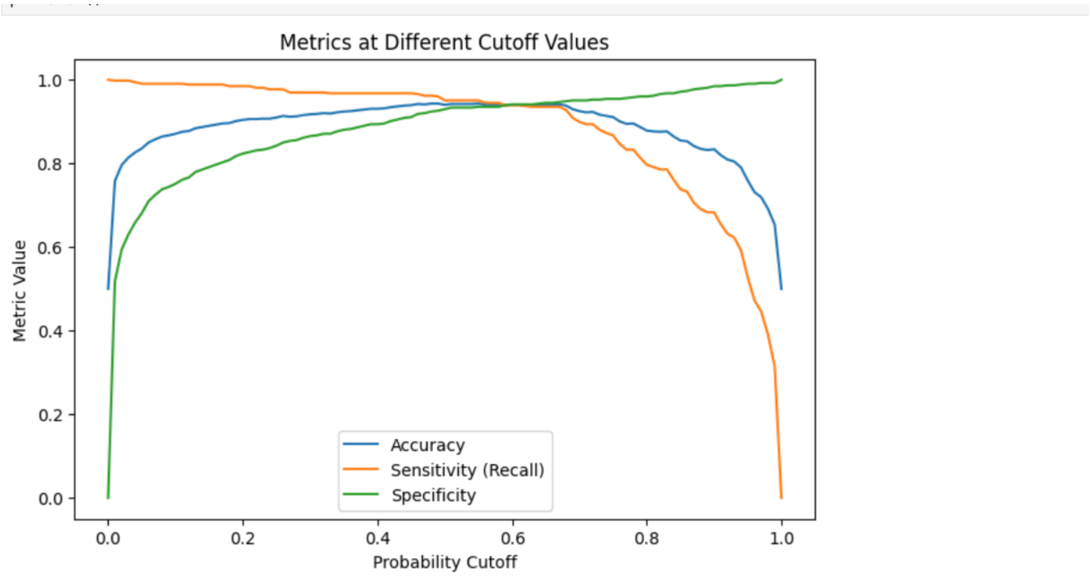
7. Model Building

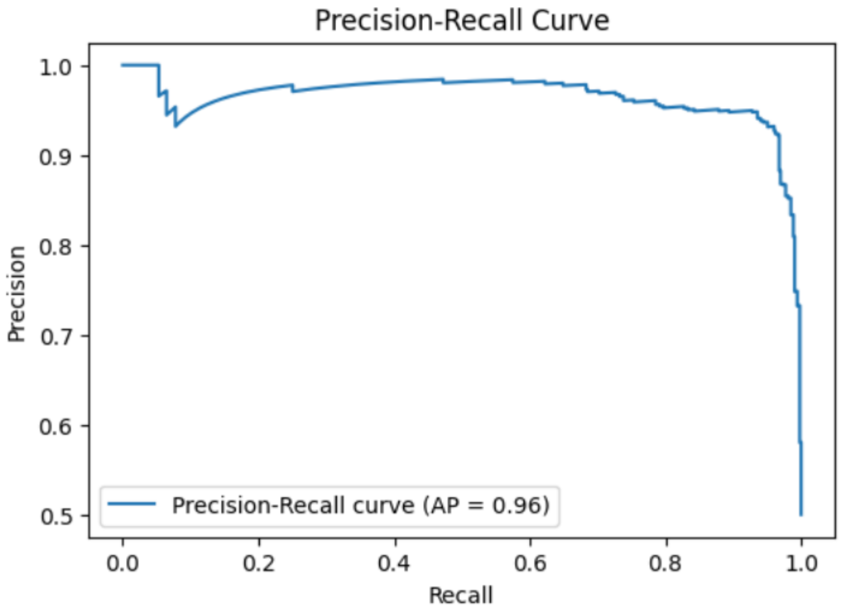
OC Curve to visualize the trade-off between true positive rate and false positive rate across different classification thresholds.



```
plt.show()
```

accuracy, sensitivity, specificity at different values of probability cutoffs





The Random Forest model developed for fraudulent insurance claim detection demonstrates **strong overall performance**, with a validation accuracy of **0.82**. This means the model correctly classifies about 82% of cases, indicating robust learning from the training data and an ability to generalize to unseen data.

Confusion Matrix Analysis:

- **True Positives (TP):** 43
- **True Negatives (TN):** 203
- **False Positives (FP):** 23
- **False Negatives (FN):** 31

This distribution reveals that the model is **slightly conservative**, favoring true negatives but still capturing a significant number of fraudulent cases.

Metric Scores:

- **Sensitivity (Recall):** 0.581
  - The model correctly identifies ~58% of all actual fraud cases.
- **Specificity:** 0.898

    - ~90% of genuine (non-fraud) cases are correctly identified, minimizing disruption to legitimate claimants.

- **Precision:** 0.652
    - Of all claims flagged as fraud, ~65% are truly fraudulent, limiting unnecessary investigations.

- **F1 Score:** 0.614
    - The balance between precision and recall is moderate, reflecting the inherent trade-off in fraud detection tasks.

Model Strengths:

- **High Specificity:** The model is excellent at minimizing false positives, which is crucial for customer satisfaction and reducing unnecessary costs.
- **Good Overall Accuracy:** At 82%, the accuracy suggests that the model is suitable for deployment in a production environment as an automated screening tool.
- **Reasonable Precision:** The model does a good job of ensuring that a significant fraction of cases flagged as fraud truly are fraudulent.