Fraudulent Claim Detection

Rupam Mallick
Muthumariappan B

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:

Build a predictive model to classify insurance claims as fraudulent or legitimate using:

- Historical claim data (claim amounts, types)
- Customer profiles
- Other relevant features

Key Questions to Address:

1. Pattern Analysis – How can we detect fraud indicators in historical claims?
2. Predictive Features – Which factors best predict fraud?
3. Fraud Likelihood – Can we score new claims for fraud risk before approval?
4. Actionable Insights – How can the model improve fraud detection?

Approach:

Data Preparation:

- We loaded the insurance_claims.csv dataset and inspected its basic information, including shape, data types, and a preview of the data.

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deductable	policy_annual_premium	umbrella_limit	insured_zip	insured_sex	insured_education_level	insured_occupation	insured_hobbies	insured_relationship
0	328	48	521585	2014-10-17	OH	250/500	1000	1406.91	0	466132	MALE	MD	craft-repair	sleeping	husband
1	228	42	342868	2006-06-27	IN	250/500	2000	1197.22	5000000	468176	MALE	MD	machine-op-inspct	reading	other-relative
2	134	29	687698	2000-09-06	OH	100/300	2000	1413.14	5000000	430632	FEMALE	PhD	sales	board-games	own-child
3	256	41	227811	1990-05-25	IL	250/500	2000	1415.74	6000000	608117	FEMALE	PhD	armed-forces	board-games	unmarried
4	228	44	367455	2014-06-06	IL	500/1000	1000	1583.91	6000000	610706	MALE	Associate	sales	board-games	unmarried

	0
months_as_customer	int64
age	int64
policy_number	int64
policy_bind_date	object
policy_state	object
policy_csl	object
policy_deductable	int64
policy_annual_premium	float64
umbrella_limit	int64
insured_zip	int64
insured_sex	object
insured_education_level	object
insured_occupation	object
insured_hobbies	object
insured_relationship	object
capital-gains	int64
capital-loss	int64

Data Cleaning:

- We identified and handled null values, specifically in the authorities_contacted column.

```
[ ] # Check the number of missing values in each column
display(df.isnull().sum())
```

insured_zip	0
insured_sex	0
insured_education_level	0
insured_occupation	0
insured_hobbies	0
insured_relationship	0
capital-gains	0
capital-loss	0
incident_date	0
incident_type	0
collision_type	0
incident_severity	0
authorities_contacted	91
incident_state	0
incident_city	0

- We are handling missing values in the column authorities_contacted by replacing them with the mode (most frequent value).

```
Before Updating the data with mode value ::  authorities_contacted
Police       292
Fire         223
Other        198
Ambulance    196
Name: count, dtype: int64
After Updating the data with mode value ::  authorities_contacted
Police       383
Fire         223
Other        198
Ambulance    196
Name: count, dtype: int64
```

- Display all unique values and their counts. Identify redundant values that might need cleaning (e.g., duplicates caused by case differences or extra spaces).

```
Column: months_as_customer
##################################################
months_as_customer
194    8
285    7
254    7
101    7
128    7
      ..
372    1
213    1
240    1
50     1
17     1
Name: count, Length: 391, dtype: int64
Total Unique value: 391
#########################
-------------------------
Redundant values:
No of times 328: 4 occurrences
No of times 228: 4 occurrences
No of times 134: 5 occurrences
No of times 256: 4 occurrences
No of times 137: 5 occurrences
No of times 165: 5 occurrences
No of times 27: 5 occurrences
No of times 212: 4 occurrences
No of times 235: 5 occurrences
```

- We identified and dropped the _c39 column as it was entirely null.

```
[ ]  # Identify and drop any columns that are completely empty
     df = df.drop(columns=['_c39'])
```

- We identified and removed one row with an illogical negative value in the umbrella_limit column.
- We removed columns that were likely identifiers or had very low predictive power (policy_number, policy_bind_date, policy_annual_premium, insured_zip, incident_location).

```
Columns likely to be identifiers or with low predictive power:
['policy_number', 'policy_bind_date', 'policy_annual_premium', 'insured_zip', 'incident_location']
```

- We fixed the data type of the incident_date column by converting it to datetime.

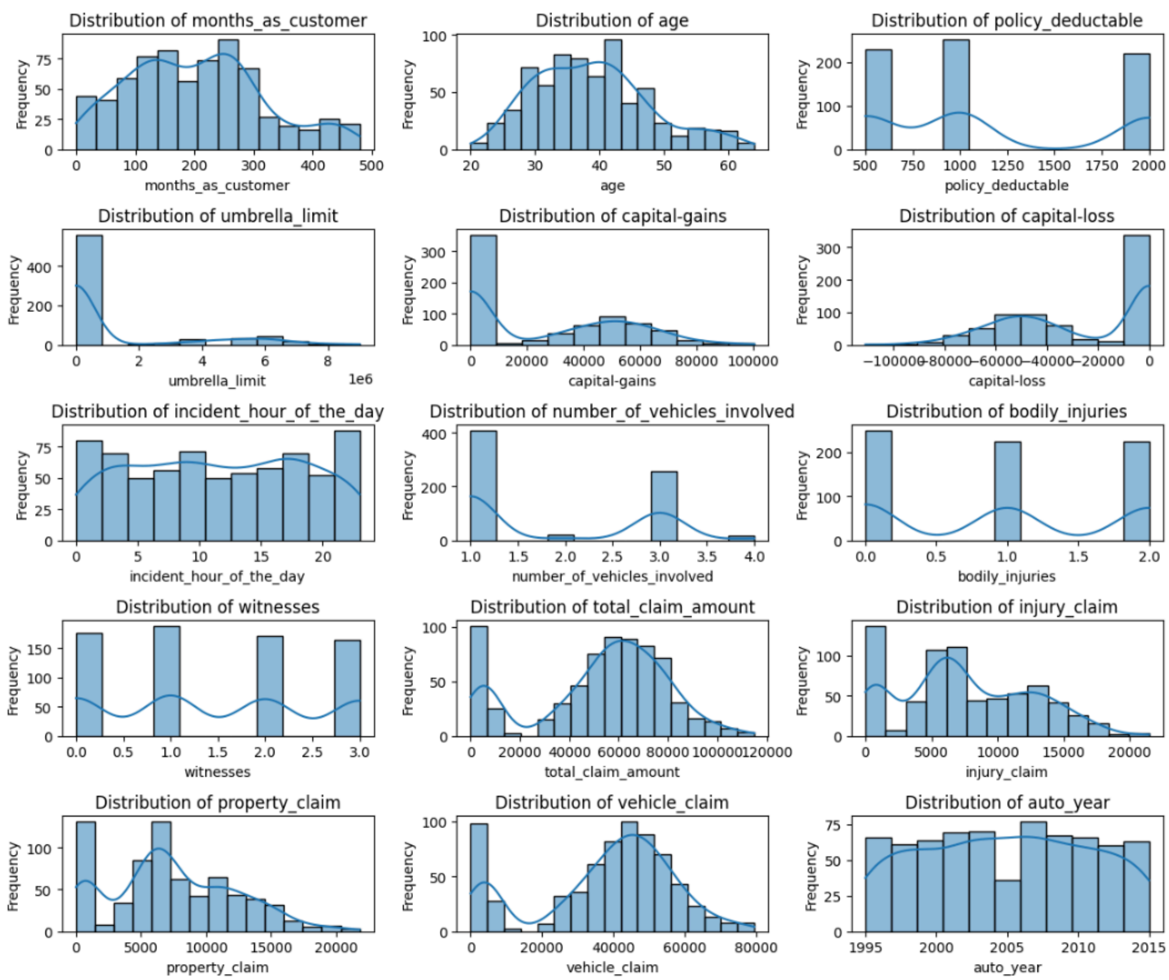
```
Column: policy_bind_date
['1990-01-08', '1990-01-27', '1990-02-01', '1990-02-03', '1990-02-08', '1990-02-14', '1990-02-15', '1990-02-18', '1990-02-23', '1990-03-14', '1990-03-17', '1990-03-18', '1990-03-26', '1990-03-28', '1990-04-29', '1990-05-07', '1990-05-16', '1990-05-19', '1990-0

Column: policy_state
['IL', 'IN', 'OH']

Column: policy_csl
['100/300', '250/500', '500/1000']

Column: insured_sex
['FEMALE', 'MALE']

Column: insured_education_level
['Associate', 'College', 'High School', 'JD', 'MD', 'Masters', 'PhD']

Column: insured_occupation
['adm-clerical', 'armed-forces', 'craft-repair', 'exec-managerial', 'farming-fishing', 'handlers-cleaners', 'machine-op-inspct', 'other-service', 'priv-house-serv', 'prof-specialty', 'protective-serv', 'sales', 'tech-support', 'transport-moving']

Column: insured_hobbies
['base-jumping', 'basketball', 'board-games', 'bungie-jumping', 'camping', 'chess', 'cross-fit', 'dancing', 'exercise', 'golf', 'hiking', 'kayaking', 'movies', 'paintball', 'polo', 'reading', 'skydiving', 'sleeping', 'video-games', 'yachting']

Column: insured_relationship
['husband', 'not-in-family', 'other-relative', 'own-child', 'unmarried', 'wife']

Column: incident_date
['2015-01-01', '2015-01-02', '2015-01-03', '2015-01-04', '2015-01-05', '2015-01-06', '2015-01-07', '2015-01-08', '2015-01-09', '2015-01-10', '2015-01-11', '2015-01-12', '2015-01-13', '2015-01-14', '2015-01-15', '2015-01-16', '2015-01-17', '2015-01-18', '2015-0

Column: incident_type
['Multi-vehicle Collision', 'Parked Car', 'Single Vehicle Collision', 'Vehicle Theft']

Column: collision_type
['?', 'Front Collision', 'Rear Collision', 'Side Collision']

Column: incident_severity
['Major Damage', 'Minor Damage', 'Total Loss', 'Trivial Damage']
```

Train Validation Split:

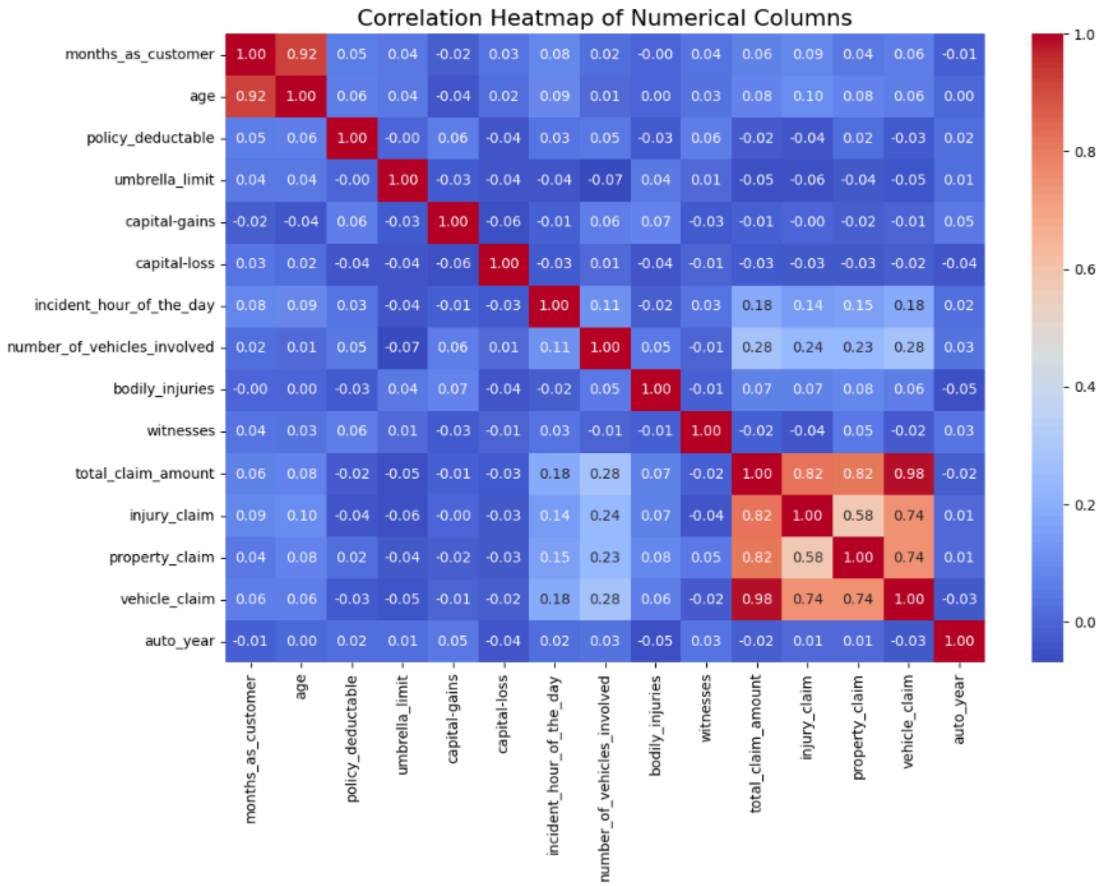
- We split the data into a 70% training set and a 30% validation set, ensuring stratification on the target variable fraud_reported to maintain the class distribution in both sets.

EDA on Training Data:

- We performed univariate analysis on numerical features using histograms to understand their distributions and identify outliers.



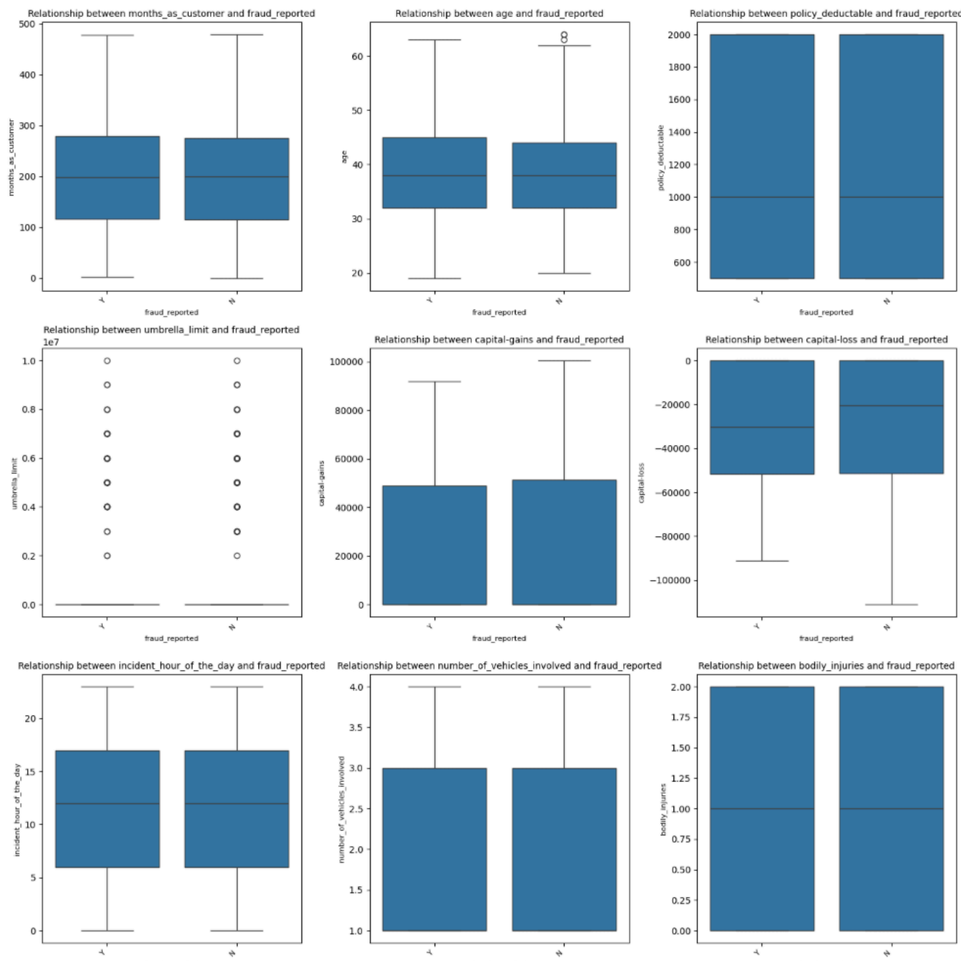
- We performed correlation analysis on numerical features using a heatmap to visualize relationships.

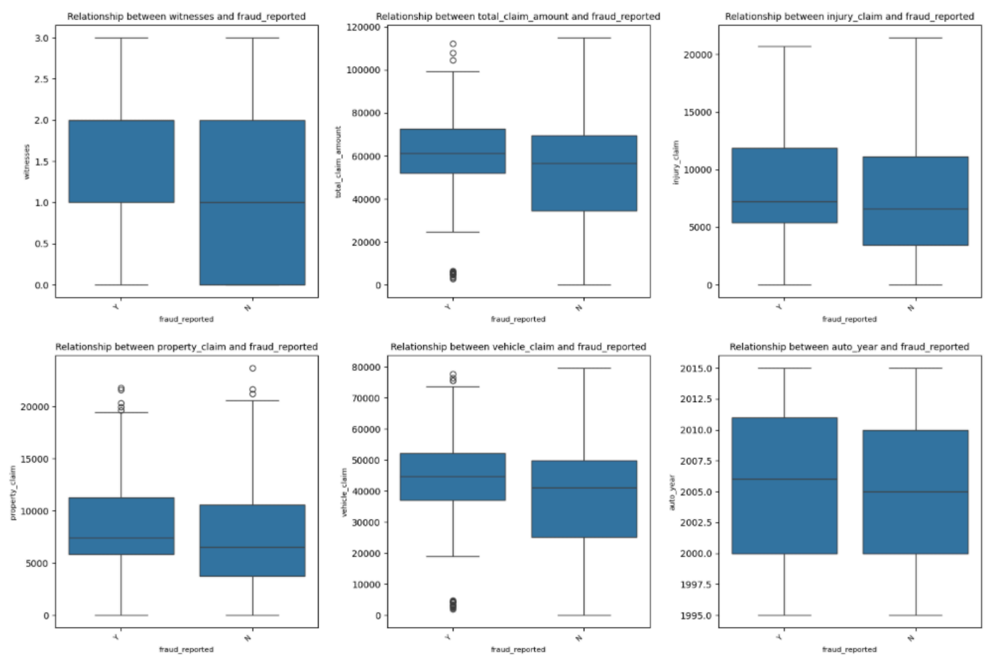


- We checked the class balance of the target variable, revealing an imbalance with approximately 75% non-fraudulent ('N') and 25% fraudulent ('Y') claims in the training data.

- We performed bivariate analysis by visualizing the target likelihood for categorical variables and the relationship between numerical features and the target variable using bar plots and box plots, respectively.

Box plot of numerical variables against target variable:





Feature Engineering:

- Resampling: The training data was resampled using RandomOverSampler to address class imbalance. The original and resampled class distributions were printed.

```
# Import RandomOverSampler from imblearn library
from imblearn.over_sampling import RandomOverSampler

# Perform resampling on training data
ros = RandomOverSampler(random_state=42)
X_train_resampled, y_train_resampled = ros.fit_resample(X_train, y_train)

print("Original Class Distribution:")
print(y_train.value_counts())
print("\nResampled Class Distribution:")
print(pd.Series(y_train_resampled).value_counts())
```

```
Original Class Distribution:
fraud_reported
N    526
Y    173
Name: count, dtype: int64

Resampled Class Distribution:
fraud_reported
N    526
Y    526
Name: count, dtype: int64
```

- Feature Creation: New features 'year', 'month', and 'day' were created from the 'incident_date' column for both the training and validation datasets.

year	month	dayofweek	policy_state_IN	policy_state_OH	policy_csl_250/500	policy_csl_500/1000	insured_sex_MALE	insured_education_level_College	insured_education_level_High School
0.0	1.026114	1.022342	False	True	False	True	True	False	False
0.0	-0.901987	0.517312	False	False	False	True	True	False	True
0.0	1.026114	-1.502806	True	False	True	False	False	False	False
0.0	-0.901987	-0.492747	False	True	True	False	True	False	False
0.0	1.026114	1.527371	False	False	False	True	False	False	False

- Handle Redundant Columns: Columns with high correlation ('age', 'vehicle_claim') and the original 'incident_date' column were dropped. The info of the updated training data was displayed to confirm the changes.

Training data shape after dropping columns: (699, 98)

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount	inju
0	-0.648103	-0.246192	-0.479788	-0.914228	0.945953	2015-02-21	1.653807	-0.822875	1.257154	1.391572	-0.309391	
1	0.065501	-1.065660	-0.479788	-0.914228	-0.519900	2015-01-30	0.067464	1.160537	1.257154	-1.322964	1.051261	
2	-1.035739	-1.065660	-0.479788	1.487295	-0.970381	2015-02-16	-0.509387	-0.822875	1.257154	0.486727	-1.766031	
3	1.994873	-0.246192	2.192496	1.501633	-0.620007	2015-01-28	-1.663091	-0.822875	1.257154	-1.322964	-0.502115	
4	-0.850731	1.392744	-0.479788	1.024913	0.945953	2015-02-01	1.221168	1.160537	-1.173691	1.391572	-0.272414	

Validation data shape after dropping columns: (300, 98)

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount	inju
0	2.250361	-0.246192	-0.479788	0.211262	0.945953	2015-02-17	1.653807	-0.822875	1.257154	0.486727	0.705029	
1	0.285749	-1.065660	-0.479788	0.515933	-1.206348	2015-02-18	-0.653600	2.152243	1.257154	-0.418119	0.604184	
2	-0.181177	1.392744	-0.479788	1.014159	-1.853468	2015-01-08	0.932742	-0.822875	1.257154	1.391572	1.371722	
3	-1.476235	1.392744	-0.479788	-0.914228	-0.909602	2015-02-14	1.653807	1.160537	1.257154	0.486727	0.009950	
4	0.232889	-0.246192	-0.479788	-0.914228	-1.034736	2015-02-05	0.788529	-0.822875	-1.173691	0.486727	0.061119	

- Combine values in Categorical Columns: Rare categories (frequency < 0.05) in categorical columns were combined into an 'Other' category to reduce sparsity. The number of rare categories replaced in each column was printed.

```
Rare categories replaced in each column:
Column 'policy_state': replaced 0 rare categories with 'Other'
Column 'policy_csl': replaced 0 rare categories with 'Other'
Column 'insured_sex': replaced 0 rare categories with 'Other'
Column 'insured_education_level': replaced 0 rare categories with 'Other'
Column 'insured_occupation': replaced 0 rare categories with 'Other'
Column 'insured_hobbies': replaced 0 rare categories with 'Other'
Column 'insured_relationship': replaced 0 rare categories with 'Other'
Column 'incident_type': replaced 0 rare categories with 'Other'
Column 'collision_type': replaced 0 rare categories with 'Other'
Column 'incident_severity': replaced 0 rare categories with 'Other'
Column 'authorities_contacted': replaced 0 rare categories with 'Other'
Column 'incident_state': replaced 0 rare categories with 'Other'
Column 'incident_city': replaced 0 rare categories with 'Other'
Column 'property_damage': replaced 0 rare categories with 'Other'
Column 'police_report_available': replaced 0 rare categories with 'Other'
Column 'auto_make': replaced 1 rare categories with 'Other'
Column 'auto_model': replaced 0 rare categories with 'Other'
```

- Dummy Variable Creation: Categorical columns were converted into numerical representations using one-hot encoding with pd.get_dummies. The shapes of the training and validation dataframes after creating dummy variables were printed.

Dummy variables were also created for the target variable 'fraud_reported', mapping 'Y' to 1 and 'N' to 0.

Training data shape after dummy encoding: (699, 98)

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount
0	-0.648103	-0.246192	-0.479788	-0.914228	0.945953	2015-02-21	1.653807	-0.822875	1.257154	1.391572	-0.309391
1	0.065501	-1.065660	-0.479788	-0.914228	-0.519900	2015-01-30	0.067464	1.160537	1.257154	-1.322964	1.051261
2	-1.035739	-1.065660	-0.479788	1.487295	-0.970381	2015-02-16	-0.509387	-0.822875	1.257154	0.486727	-1.766031
3	1.994873	-0.246192	2.192496	1.501633	-0.620007	2015-01-28	-1.663091	-0.822875	1.257154	-1.322964	-0.502115
4	-0.850731	1.392744	-0.479788	1.024913	0.945953	2015-02-01	1.221168	1.160537	-1.173691	1.391572	-0.272414

Validation data shape after dummy encoding: (300, 98)

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount
0	2.250361	-0.246192	-0.479788	0.211262	0.945953	2015-02-17	1.653807	-0.822875	1.257154	0.486727	0.705029
1	0.285749	-1.065660	-0.479788	0.515933	-1.206348	2015-02-18	-0.653600	2.152243	1.257154	-0.418119	0.604184
2	-0.181177	1.392744	-0.479788	1.014159	-1.853468	2015-01-08	0.932742	-0.822875	1.257154	1.391572	1.371722
3	-1.476235	1.392744	-0.479788	-0.914228	-0.909602	2015-02-14	1.653807	1.160537	1.257154	0.486727	0.009950
4	0.232889	-0.246192	-0.479788	-0.914228	-1.034736	2015-02-05	0.788529	-0.822875	-1.173691	0.486727	0.061119

- Feature Scaling: Numerical features in both training and validation data were scaled using StandardScaler to ensure a common range. The target column is removed because feature scaling is only meant for predictors. Scaling the label would distort the problem and lead to incorrect training.

```python
# Import the necessary scaling tool from scikit-learn
from sklearn.preprocessing import StandardScaler
# Scale the numeric features present in the training data
target_col = 'fraud_reported'
numerical_columns = training_df_dummy.select_dtypes(include=['number']).columns.tolist()
if target_col in numerical_columns:
    numerical_columns.remove(target_col)
scaler = StandardScaler()
training_df_dummy[numerical_columns] = scaler.fit_transform(training_df_dummy[numerical_columns])
# Scale the numeric features present in the validation data
validation_df_dummy[numerical_columns] = scaler.transform(validation_df_dummy[numerical_columns])
```

Training data after scaling:

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount
0	-0.648103	-0.246192	-0.479788	-0.914228	0.945953	2015-02-21	1.653807	-0.822875	1.257154	1.391572	-0.309391
1	0.065501	-1.065660	-0.479788	-0.914228	-0.519900	2015-01-30	0.067464	1.160537	1.257154	-1.322964	1.051261
2	-1.035739	-1.065660	-0.479788	1.487295	-0.970381	2015-02-16	-0.509387	-0.822875	1.257154	0.486727	-1.766031
3	1.994873	-0.246192	2.192496	1.501633	-0.620007	2015-01-28	-1.663091	-0.822875	1.257154	-1.322964	-0.502115
4	-0.850731	1.392744	-0.479788	1.024913	0.945953	2015-02-01	1.221168	1.160537	-1.173691	1.391572	-0.272414

Validation data after scaling:

	months_as_customer	policy_deductable	umbrella_limit	capital-gains	capital-loss	incident_date	incident_hour_of_the_day	number_of_vehicles_involved	bodily_injuries	witnesses	total_claim_amount
0	2.250361	-0.246192	-0.479788	0.211262	0.945953	2015-02-17	1.653807	-0.822875	1.257154	0.486727	0.705029
1	0.285749	-1.065660	-0.479788	0.515933	-1.206348	2015-02-18	-0.653600	2.152243	1.257154	-0.418119	0.604184
2	-0.181177	1.392744	-0.479788	1.014159	-1.853468	2015-01-08	0.932742	-0.822875	1.257154	1.391572	1.371722
3	-1.476235	1.392744	-0.479788	-0.914228	-0.909602	2015-02-14	1.653807	1.160537	1.257154	0.486727	0.009950
4	0.232889	-0.246192	-0.479788	-0.914228	-1.034736	2015-02-05	0.788529	-0.822875	-1.173691	0.486727	0.061119

Model Building:

Logistic Regression Model:

- Feature Selection using RFECV: Recursive Feature Elimination with Cross-Validation (RFECV) was used to identify the most relevant features for the logistic regression model. The optimal number of features was determined based on cross-validation performance.

```
Summary of Model Building:

Optimal number of features (Logistic Regression): 43
Selected Features (Logistic Regression): ['incident_severity_Minor Damage', 'insured_hobbies_chess', 'incident_severity_Total Loss', '
```

Feature Ranking (Logistic Regression):

	Feature	Ranking
2	umbrella_limit	1
8	witnesses	1
27	insured_occupation_adm-clerical	1
28	insured_occupation_armed-forces	1
23	insured_education_level_JD	1
31	insured_occupation_farming-fishing	1
29	insured_occupation_craft-repair	1
30	insured_occupation_exec-managerial	1
24	insured_education_level_MD	1
26	insured_education_level_PhD	1

- We make sure the model uses only relevant features (reducing noise and overfitting risk). We convert all data into numerical form, making it compatible with regression. We add a constant column for the intercept in regression models.
- Model Building and Multicollinearity Assessment: A logistic regression model was built using the statsmodels library, which provides detailed statistical outputs. P-values and Variance Inflation Factors (VIFs) were examined to assess the significance of features and detect multicollinearity.

```
Logistic Regression Model Summary:
                           Logit Regression Results
==============================================================================
Dep. Variable:          fraud_reported   No. Observations:                 1052
Model:                           Logit   Df Residuals:                     1008
Method:                            MLE   Df Model:                           43
Date:                 Tue, 12 Aug 2025   Pseudo R-squ.:                  0.5065
Time:                         18:48:42   Log-Likelihood:                -359.84
converged:                        True   LL-Null:                       -729.19
Covariance Type:             nonrobust   LLR p-value:                 1.601e-127
==========================================================================================================
                                            coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------------------
const                                    -0.3446      0.405     -0.852      0.394      -1.138       0.448
umbrella_limit                            0.3297      0.103      3.206      0.001       0.128       0.531
witnesses                                 0.4075      0.105      3.883      0.000       0.202       0.613
policy_state_OH                           0.3258      0.213      1.532      0.126      -0.091       0.743
policy_csl_250/500                        0.4284      0.240      1.787      0.074      -0.042       0.898
policy_csl_500/1000                      -0.2636      0.255     -1.035      0.301      -0.763       0.236
insured_education_level_JD                0.8657      0.286      3.031      0.002       0.306       1.425
insured_education_level_MD                1.0819      0.305      3.551      0.000       0.485       1.679
insured_education_level_PhD               0.7061      0.320      2.210      0.027       0.080       1.332
insured_occupation_adm-clerical           1.0016      0.487      2.056      0.040       0.047       1.956
insured_occupation_armed-forces           1.3546      0.491      2.759      0.006       0.392       2.317
insured_occupation_craft-repair           0.6795      0.427      1.590      0.112      -0.158       1.517
insured_occupation_exec-managerial        1.2657      0.435      2.911      0.004       0.414       2.118
insured_occupation_farming-fishing        0.7549      0.528      1.430      0.153      -0.280       1.790
insured_occupation_machine-op-inspct      0.8789      0.420      2.095      0.036       0.057       1.701
insured_occupation_priv-house-serv       -0.1479      0.524     -0.282      0.778      -1.175       0.879
insured_occupation_prof-specialty         1.2676      0.413      3.070      0.002       0.458       2.077
insured_occupation_sales                  0.8643      0.483      1.791      0.073      -0.081       1.810
insured_occupation_tech-support           0.7991      0.444      1.801      0.072      -0.071       1.669
insured_occupation_transport-moving       1.7103      0.402      4.251      0.000       0.922       2.499
insured_hobbies_bungie-jumping           -0.6862      0.488     -1.406      0.160      -1.642       0.270
insured_hobbies_chess                     6.0343      0.655      9.219      0.000       4.751       7.317
insured_hobbies_skydiving                -0.3722      0.504     -0.739      0.460      -1.359       0.615
insured_hobbies_yachting                  0.4412      0.452      0.976      0.329      -0.445       1.327
insured_relationship_not-in-family        0.6808      0.299      2.278      0.023       0.095       1.266
insured_relationship_other-relative       0.3966      0.291      1.362      0.173      -0.174       0.967
insured_relationship_own-child           -0.5901      0.315     -1.876      0.061      -1.207       0.026
insured_relationship_unmarried            0.7753      0.314      2.471      0.013       0.160       1.390
collision_type_Front Collision            0.6978      0.256      2.724      0.006       0.196       1.200
collision_type_Rear Collision             0.7843      0.252      3.114      0.002       0.291       1.278
incident_severity_Minor Damage           -4.0310      0.292    -13.814      0.000      -4.603      -3.459
incident_severity_Total Loss             -3.4324      0.275    -12.492      0.000      -3.971      -2.894
incident_severity_Trivial Damage         -4.0472      0.571     -7.089      0.000      -5.166      -2.928
authorities_contacted_Other               0.4487      0.244      1.839      0.066      -0.030       0.927
incident_state_NY                        -0.2170      0.251     -0.863      0.388      -0.710       0.276
incident_state_VA                         0.8417      0.335      2.514      0.012       0.186       1.498
incident_state_WV                        -0.8190      0.290     -2.828      0.005      -1.387      -0.251
incident_city_Northbrook                 -0.7216      0.346     -2.087      0.037      -1.399      -0.044
property_damage_NO                       -0.6207      0.226     -2.743      0.006      -1.064      -0.177
auto_make_Audi                            1.4614      0.370      3.955      0.000       0.737       2.186
auto_make_BMW                             0.5967      0.399      1.495      0.135      -0.186       1.379
auto_make_Dodge                           0.6321      0.364      1.734      0.083      -0.082       1.347
auto_make_Nissan                         -0.6645      0.423     -1.573      0.116      -1.493       0.164
auto_make_Other                           1.0010      0.466      2.149      0.032       0.088       1.914
==========================================================================================================
```

VIFs:

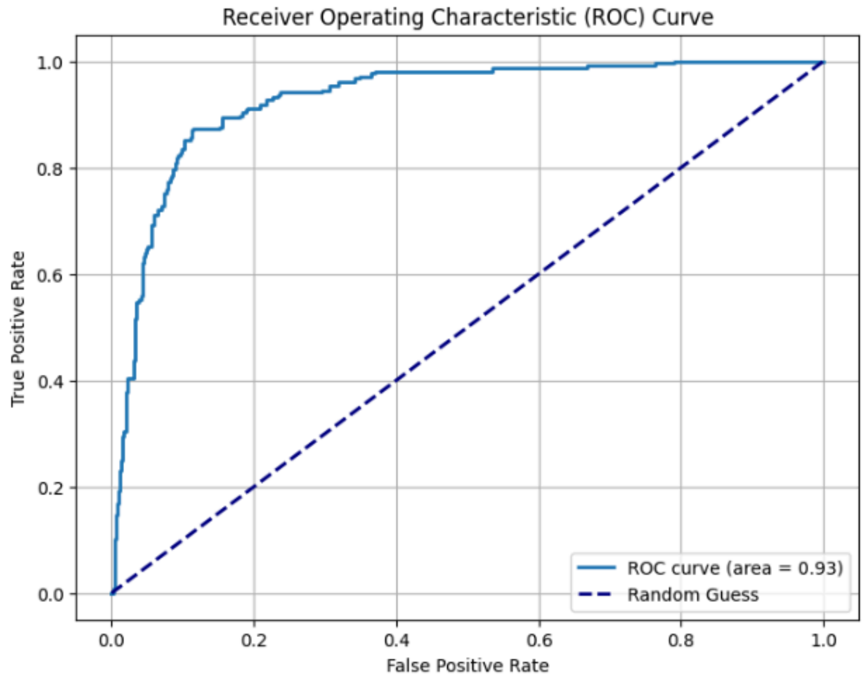
	feature	VIF
0	const	18.598454
1	umbrella_limit	1.115080
2	witnesses	1.095291
3	policy_state_OH	1.084156
4	policy_csl_250/500	1.412014
5	policy_csl_500/1000	1.403936
6	insured_education_level_JD	1.206489
7	insured_education_level_MD	1.190618
8	insured_education_level_PhD	1.164470
9	insured_occupation_adm-clerical	1.481426
10	insured_occupation_armed-forces	1.468641
11	insured_occupation_craft-repair	1.538530
12	insured_occupation_exec-managerial	1.607594
13	insured_occupation_farming-fishing	1.363476
14	insured_occupation_machine-op-inspct	1.562082
15	insured_occupation_priv-house-serv	1.396265
16	insured_occupation_prof-specialty	1.523132
17	insured_occupation_sales	1.504562
18	insured_occupation_tech-support	1.510611
19	insured_occupation_transport-moving	1.541714
20	insured_hobbies_bungie-jumping	1.101495
21	insured_hobbies_chess	1.186471
22	insured_hobbies_skydiving	1.187879
23	insured_hobbies_yachting	1.076371
24	insured_relationship_not-in-family	1.422449
25	insured_relationship_other-relative	1.400125
26	insured_relationship_own-child	1.313185
27	insured_relationship_unmarried	1.381080
28	collision_type_Front Collision	1.413880
29	collision_type_Rear Collision	1.443814
30	incident_severity_Minor Damage	1.378678
31	incident_severity_Total Loss	1.365228
32	incident_severity_Trivial Damage	1.267738
33	authorities_contacted_Other	1.138550
34	incident_state_NY	1.267605
35	incident_state_VA	1.234985
36	incident_state_WV	1.329504
37	incident_city_Northbrook	1.070875
38	property_damage_NO	1.116931
39	auto_make_Audi	1.176803
40	auto_make_BMW	1.132866
41	auto_make_Dodge	1.127219
42	auto_make_Nissan	1.158198
43	auto_make_Other	1.094229

- Model Training and Evaluation on Training Data: The model was trained on the training data, and initial performance was assessed using accuracy and a confusion matrix with a default cutoff of 0.5.

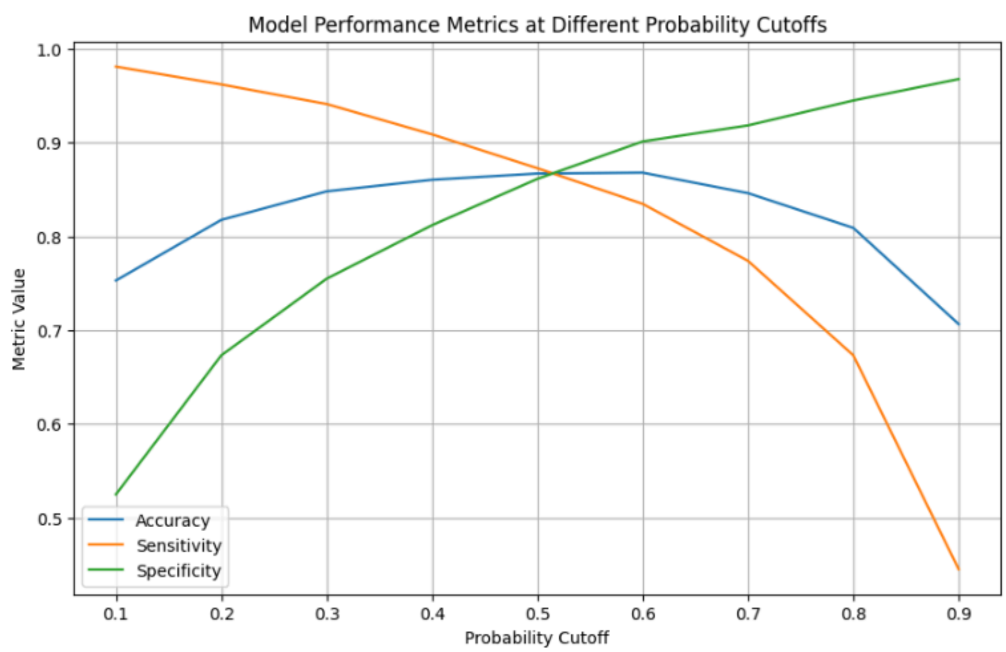
```
Logistic Regression Metrics on Training Data (Cutoff 0.5):
Accuracy: 0.8669201520912547
Confusion Matrix:
 [[453  73]
 [ 67 459]]
True Negative:  453
False Positive:  73
False Negative:  67
True Positive:  459
Sensitivity:  0.8726235741444867
Specificity:  0.8612167300380228
Precision:  0.8627819548872181
Recall:  0.8726235741444867
F1 Score:  0.8676748582230625
```

- Finding the Optimal Cutoff: The optimal probability threshold was determined by analyzing the trade-offs between sensitivity, specificity, precision, and recall across different cutoff values using ROC and Precision-Recall curves.

- AUC is 0.93 which means the model performs very well, with a high probability of correctly ranking positive instances above negative ones.



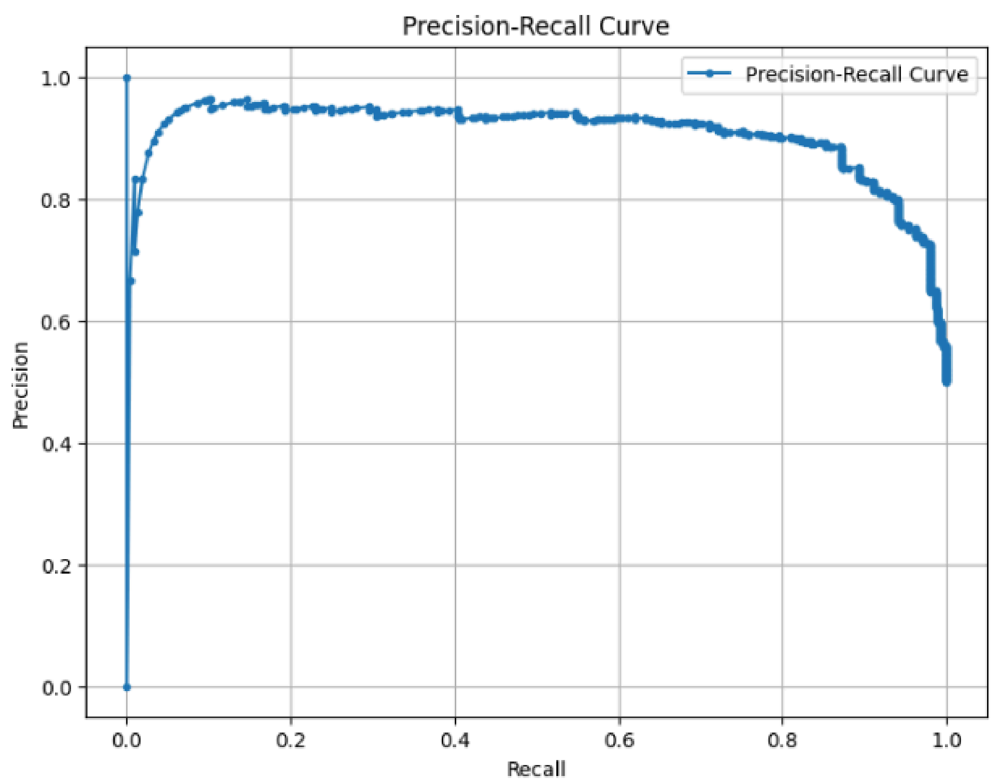
- We are choosing an optimal cutoff as 0.55 because that is where accuracy starts increasing. For balanced performance we choose a cutoff where sensitivity and specificity intersect.



- Final Prediction and Evaluation on Training Data using the Optimal Cutoff: Final predictions were made on the training data using the optimal cutoff, and model performance was re-evaluated.

```
Logistic Regression Metrics on Training Data (Optimal Cutoff):
Accuracy: 0.879277566539924
Confusion Matrix:
 [[466  60]
 [ 67 459]]
True Negative:  466
False Positive:  60
False Negative:  67
True Positive:  459
Sensitivity:  0.8726235741444867
Specificity:  0.8859315589353612
Precision:  0.884393063583815
Recall:  0.8726235741444867
F1 Score:  0.8784688995215311
```

- The model maintains high precision (>0.8) across a wide range of recall values, meaning it produces few false positives even when trying to capture more positives.



Random Forest Model:

- Get Feature Importances: Feature importance scores were obtained from an initial Random Forest model to identify features that contribute most to the prediction.

Random Forest Feature Importance:

	Feature	Importance
30	incident_severity_Minor Damage	0.129818
21	insured_hobbies_chess	0.085115
31	incident_severity_Total Loss	0.077704
2	witnesses	0.048884
32	incident_severity_Trivial Damage	0.042676
1	umbrella_limit	0.036350
38	property_damage_NO	0.027926
36	incident_state_WV	0.027695
3	policy_state_OH	0.024897
28	collision_type_Front Collision	0.024682
4	policy_csl_250/500	0.023952
5	policy_csl_500/1000	0.022861
29	collision_type_Rear Collision	0.022149
34	incident_state_NY	0.021900
33	authorities_contacted_Other	0.021040
25	insured_relationship_other-relative	0.019196
26	insured_relationship_own-child	0.018632
7	insured_education_level_MD	0.018179
27	insured_relationship_unmarried	0.017846
6	insured_education_level_JD	0.017803
24	insured_relationship_not-in-family	0.016585
39	auto_make_Audi	0.015723
8	insured_education_level_PhD	0.015334
12	insured_occupation_exec-managerial	0.015122
41	auto_make_Dodge	0.014649
19	insured_occupation_transport-moving	0.014292
10	insured_occupation_armed-forces	0.014241
16	insured_occupation_prof-specialty	0.014116
37	incident_city_Northbrook	0.013911
35	incident_state_VA	0.013686
18	insured_occupation_tech-support	0.012375
42	auto_make_Nissan	0.010936
11	insured_occupation_craft-repair	0.010761
14	insured_occupation_machine-op-inspct	0.010680
15	insured_occupation_priv-house-serv	0.010105
40	auto_make_BMW	0.009942
17	insured_occupation_sales	0.009551
9	insured_occupation_adm-clerical	0.009536
43	auto_make_Other	0.008955
13	insured_occupation_farming-fishing	0.008633
22	insured_hobbies_skydiving	0.007378
20	insured_hobbies_bungie-jumping	0.007180
23	insured_hobbies_yachting	0.007003
0	const	0.000000

- Select Important Features: Features with importance scores above a certain threshold were selected for further model training.
- Model Evaluation on Training Data: The Random Forest model was trained with the selected features and evaluated on the training data using accuracy and a confusion matrix.

```
Random Forest Metrics on Training Data (Base Model):
Accuracy: 1.0
Confusion Matrix:
 [[526   0]
 [  0 526]]
True Negative:  526
False Positive:  0
False Negative:  0
True Positive:  526
Sensitivity:  1.0
Specificity:  1.0
Precision:  1.0
Recall:  1.0
F1 Score:  1.0
```

- Check Model Overfitting using Cross-Validation: Cross-validation was performed to assess the model's generalization ability and check for overfitting on the training data.

```
Random Forest Cross Validation Scores:
[0.90047393 0.96208531 0.97142857 0.94285714 0.94285714]
```

- Hyperparameter Tuning using Grid Search: Grid search was used to find the best combination of hyperparameters for the Random Forest model to optimize its performance.

```
Random Forest Best Hyperparameters:
{'max_depth': 20, 'max_features': 'log2', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 200}
Random Forest CV score: 0.9448973143759873
```

- Final Model and Evaluation on Training Data: The final Random Forest model was built using the best hyperparameters, trained on the training data, and its performance evaluated.

```
Random Forest Metrics on Training Data (Tuned Model):
Accuracy: 0.9990494296577946
Confusion Matrix:
 [[525   1]
 [  0 526]]
True Negative:  525
False Positive:  1
False Negative:  0
True Positive:  526
Sensitivity:  1.0
Specificity:  0.9980988593155894
Precision:  0.9981024667931688
Recall:  1.0
F1 Score:  0.9990503323836657
```

Predicting and Model Evaluation:

The performance of both models was evaluated on the validation data.

Make predictions over validation data using logistic regression model:

- The relevant features were selected for the validation data, and a constant was added.
- Predictions were made on the validation data using the trained Logistic Regression model.
- A DataFrame was created to show the actual values and the predicted probabilities for the validation data.
- Final predictions were made using a cutoff value of 0.5.

- The accuracy, confusion matrix, TP, TN, FP, FN, sensitivity, specificity, precision, recall, and F1-score were calculated and printed for the validation data using the Logistic Regression model.

```
Summary of Prediction and Model Evaluation:

Logistic Regression Metrics on Validation Data:
Accuracy: 0.31333333333333335
Confusion Matrix:
 [[ 23 203]
 [  3  71]]
True Negative:  71
False Positive:  203
False Negative:  3
True Positive:  23
Sensitivity:  0.8846153846153846
Specificity:  0.2591240875912409
Precision:  0.10176991150442478
Recall:  0.8846153846153846
F1 Score:  0.18253968253968256
```

Make predictions over validation data using random forest model:

- The important features were selected for the validation data.
- Probability predictions were made on the validation data using the trained Random Forest model.
- The accuracy, confusion matrix, TP, TN, FP, FN, sensitivity, specificity, precision, recall, and F1-score were calculated and printed for the validation data using the Random Forest model with a cutoff of 0.5.

```
Random Forest Metrics on Validation Data:
Accuracy: 0.78
Confusion Matrix:
 [[195  31]
 [ 35  39]]
True Negative:  195
False Positive:  31
False Negative:  35
True Positive:  39
Sensitivity:  0.527027027027027
Specificity:  0.8628318584070797
Precision:  0.5571428571428572
Recall:  0.527027027027027
F1 Score:  0.5416666666666666
```

Conclusion:

Pattern Analysis – How can we detect fraud indicators in historical claims?

We can analyze historical claim data to detect patterns by performing Exploratory Data Analysis (EDA). This involves:

- Univariate Analysis: Examining the distribution of individual features (both numerical and categorical) to understand their characteristics and identify any unusual patterns.
- Correlation Analysis: Investigating the relationships between numerical features to identify potential dependencies.
- Bivariate Analysis: Exploring the relationships between features and the target variable (fraud_reported) to understand how different feature values influence the likelihood of a claim being fraudulent. This is done for both numerical and categorical features.

Predictive Features – Which factors best predict fraud?

Based on the Random Forest model's feature importance scores, the most predictive features of fraudulent behavior are:

- incident_severity_Minor Damage
- insured_hobbies_chess
- incident_severity_Total Loss
- witnesses
- incident_severity_Trivial Damage
- umbrella_limit
- property_damage_NO
- incident_state_WV
- policy_state_OH
- collision_type_Front Collision
- policy_csl_250/500
- policy_csl_500/1000
- collision_type_Rear Collision

- incident_state_NY
- authorities_contacted_Other
- insured_relationship_other-relative
- insured_relationship_own-child
- insured_education_level_MD
- insured_relationship_unmarried
- insured_education_level_JD
- insured_relationship_not-in-family
- auto_make_Audi
- insured_education_level_PhD
- insured_occupation_exec-managerial
- auto_make_Dodge
- insured_occupation_transport-moving
- insured_occupation_armed-forces
- insured_occupation_prof-specialty
- incident_city_Northbrook
- incident_state_VA
- insured_occupation_tech-support
- auto_make_Nissan
- insured_occupation_craft-repair
- insured_occupation_machine-op-inspct
- insured_occupation_priv-house-serv
- auto_make_BMW
- insured_occupation_sales
- insured_occupation_adm-clerical
- auto_make_Other
- insured_occupation_farming-fishing
- insured_hobbies_skydiving
- insured_hobbies_bungie-jumping
- insured_hobbies_yachting

Fraud Likelihood – Can we score new claims for fraud risk before approval?

- We can predict the likelihood of fraud for an incoming claim based on past data.
- By training machine learning models (like Logistic Regression and Random Forest) on the historical claim data, we can use these models to predict the probability of an incoming claim being fraudulent.
- The models learn patterns and relationships from the historical data that are indicative of fraudulent behavior.

Actionable Insights – How can the model improve fraud detection?

- Focus on key features: The most important features identified by the models (e.g., incident severity, insured hobbies, number of witnesses, umbrella limit, etc.) should be prioritized in the manual review process. Claims with suspicious values or combinations of these features should be flagged for closer inspection.
- Automated flagging: The trained model can be used to automatically flag claims with a high predicted probability of fraud. This can help streamline the initial screening process and reduce the workload on manual reviewers.
- Identify unusual patterns: The analysis of feature importance and the relationships between features and the target variable can help identify unusual patterns or anomalies that may indicate fraudulent activity. These patterns can be used to refine fraud detection rules or develop new fraud indicators.
- Continuous monitoring and retraining: The model should be continuously monitored for performance and retrained periodically with new data to adapt to evolving fraud patterns.

In conclusion, the Random Forest model performed better in this analysis and can be used to predict the likelihood of fraud for incoming claims. The insights gained from the model, particularly the important features, can significantly help Global Insure improve their fraud detection process by enabling more efficient and data-driven screening of claims.