Fraudulent Claim Detection Report

Prepared by: Keshav Jadhav and Anukul Morey

---

1. Introduction

Insurance fraud is a widespread issue that continues to challenge the integrity and profitability of the insurance industry. Every year, fraudulent claims cost insurers billions of dollars globally, impacting not only the bottom line of companies but also the premiums paid by honest policyholders. Tackling this problem requires a proactive, data-driven approach that can scale with the growing number of claims being processed every day. In our project, we tackled this issue by developing a machine learning pipeline that detects fraudulent insurance claims using historical customer and policy data. This solution is not just about building a model—it's about transforming how insurers handle risk, improve efficiency, and serve their genuine customers. By using predictive analytics and automation, this system can help streamline operations, cut down costs, and boost trust. The human impact is also critical—faster approval for legitimate claims can enhance the customer experience significantly. Our goal was to deliver both technical excellence and practical value through this work

2. Problem Statement

The central business challenge was to create a binary classification system that predicts whether a given insurance claim is fraudulent or genuine. This is vital for insurance providers who want to identify high-risk claims as early as possible to reduce financial losses and investigation overheads. Fraud detection is typically done through manual review, which is slow, expensive, and inconsistent. Our aim was to automate this process by analyzing historical data patterns in customer behavior, policy details, incident information, and claim values. Using these insights, we sought to build a model that could flag suspicious claims before they are processed further. A solution like this is immensely beneficial to both the business and its customers. Not only does it save time and resources, but it also allows insurers to focus on genuinely deserving claims, ultimately increasing customer satisfaction. Therefore, this project directly aligns with the goals of efficiency, cost reduction, and improved customer service in the insurance sector.

3. Methodology

We followed a structured machine learning workflow to ensure that each phase of the project contributed effectively to the final solution:

3.1 Data Loading and Inspection

The project began with loading the dataset into a pandas DataFrame and understanding its structure. We inspected its shape, checked data types, and looked at initial statistics to identify potential inconsistencies. A thorough understanding of the data structure guided subsequent cleaning and transformation steps.

```
[319] # Load the dataset
      df = pd.read_csv('insurance_claims.csv')
```

```
[320] # Check at the first few entries
      df.head()
```

onths_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	policy_deduc
328	48	521585	2014-10-17	OH	250/500	
228	42	342868	2006-06-27	IN	250/500	
134	29	687698	2000-09-06	OH	100/300	
256	41	227811	1990-05-25	IL	250/500	
228	44	367455	2014-06-06	IL	500/1000	

s × 40 columns

```
# Inspect the shape of the dataset
print("The dataset contains", df.shape[0], "rows and", df.shape[1], "columns.")
```

```
The dataset contains 1000 rows and 40 columns.
```

```
[322] # Inspect the features in the dataset
      print("Features in the dataset:")
      print(df.columns.tolist())
```

```
Features in the dataset:
['months_as_customer', 'age', 'policy_number', 'policy_bind_date', 'policy_state', 'policy_cs
```

3.2 Data Cleaning

During cleaning, we dealt with missing values in critical columns like property_damage and police_report_available by replacing them with "Not Available". Invalid values such as negative claim amounts were removed or corrected. Columns like zip codes and customer names were dropped as they didn't offer predictive value. Data types were corrected—for instance, converting numerical fields stored as strings into integers

```
[323] # Check the number of missing values in each column
      missing_values = df.isnull().sum()
      missing_values = missing_values[missing_values > 0].sort_values(ascending=False)

      print("Missing values in each column:\n")
      print(missing_values)

Missing values in each column:

_c39                       1000
authorities_contacted        91
dtype: int64
```

2.1.2 Handle rows containing null values [1 Mark]

```
# Handle the rows containing null values
categorical_with_nulls = ['property_damage', 'police_report_available', 'collision_type']

for col in categorical_with_nulls:
    df[col].fillna('Unknown', inplace=True)

# Verify if nulls still exist
print("Remaining null values:\n", df.isnull().sum()[df.isnull().sum() > 0])

Remaining null values:
 authorities_contacted      91
_c39                     1000
dtype: int64
```

```
[326] # Identify and drop any columns that are completely empty
      empty_cols = df.columns[df.isnull().sum() == df.shape[0]]

      print("Completely empty columns:\n", list(empty_cols))

      # Drop these columns
      df.drop(columns=empty_cols, inplace=True)

Completely empty columns:
 ['_c39']
```

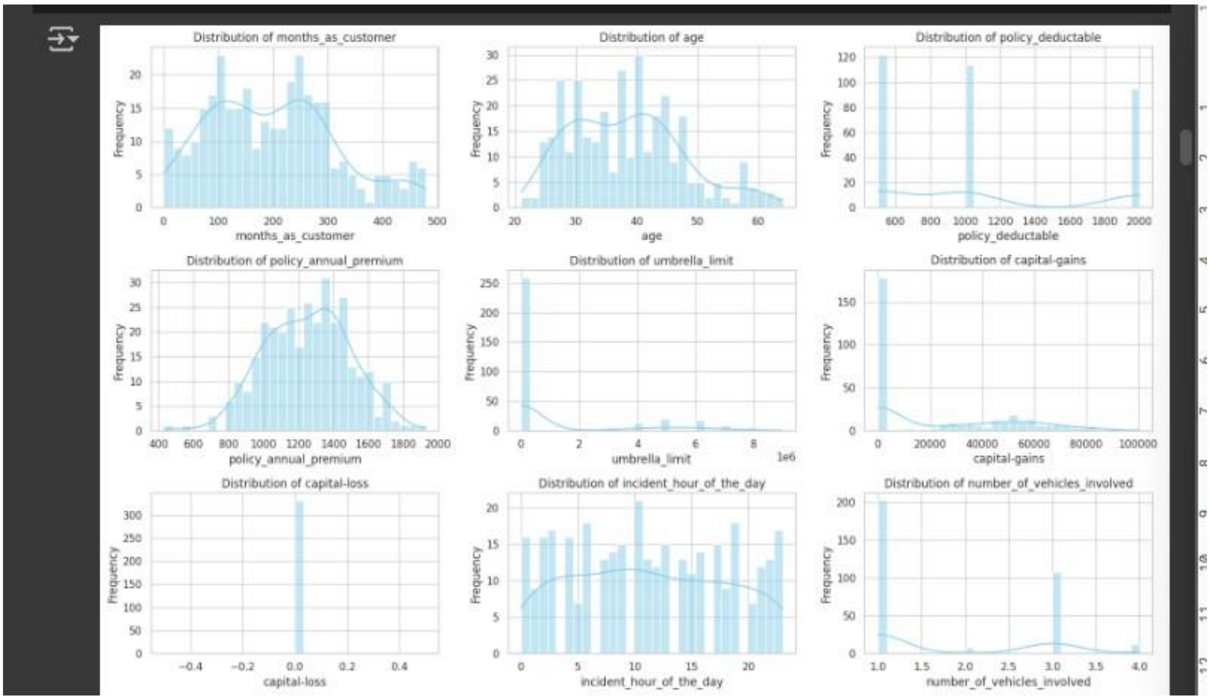
3.3 Exploratory Data Analysis (EDA)

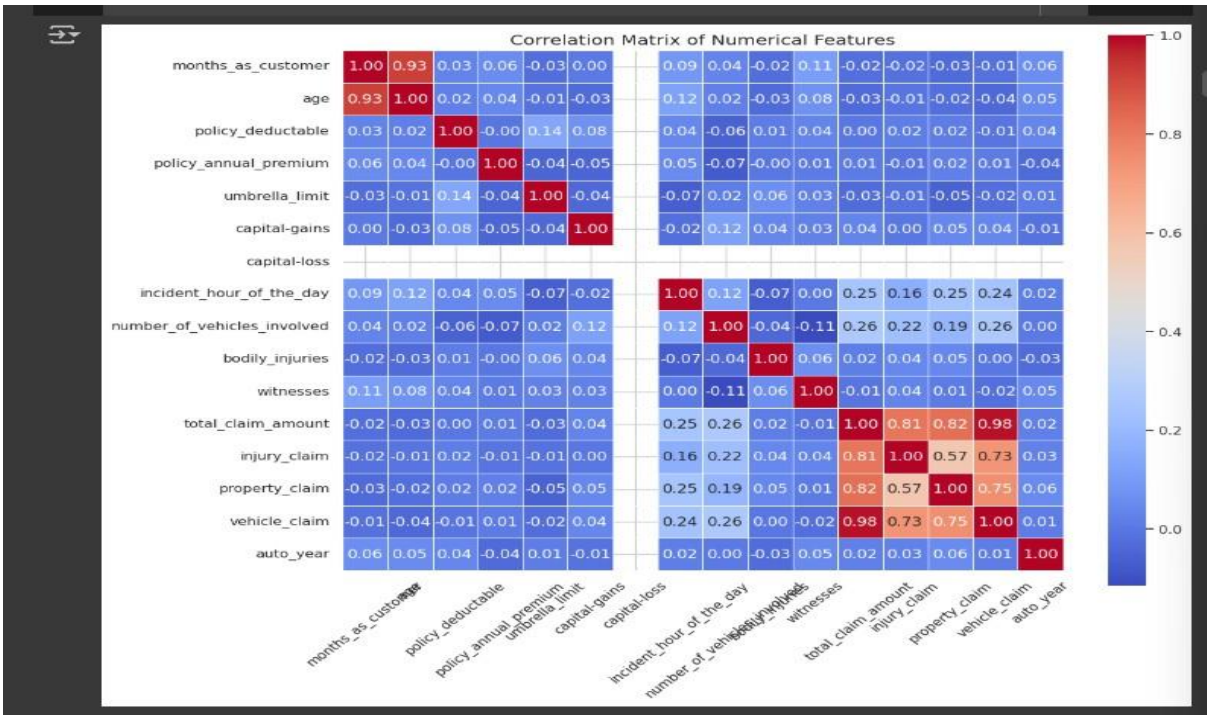
We explored both numerical and categorical features to understand their distribution and relationship with the target variable (fraud_reported). Visuals such as histograms, bar charts, and heatmaps were used to identify correlations and patterns. For instance, certain auto models and occupations showed a higher fraud likelihood.

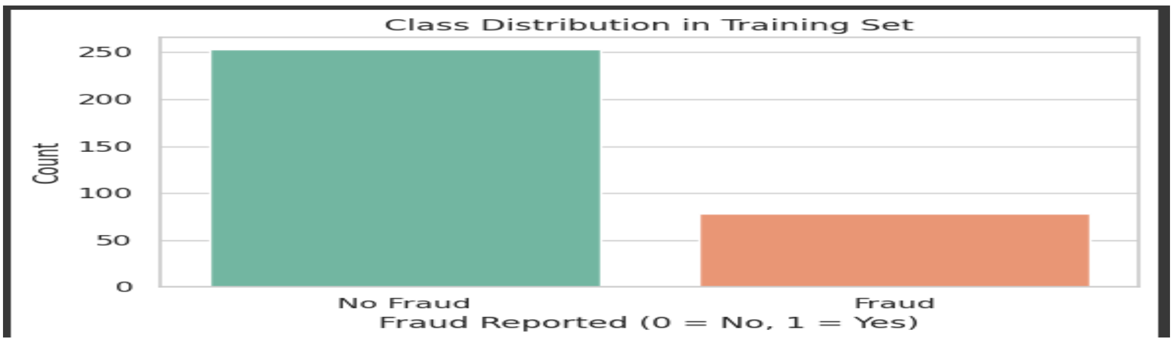
```
# Select numerical columns
numerical_cols = X_train.select_dtypes(include=[np.number]).columns.tolist()

print("🔢 Numerical columns for univariate analysis:")
print(numerical_cols)

🔢 Numerical columns for univariate analysis:
['months_as_customer', 'age', 'policy_deductable', 'policy_annual_premium', 'umbrella_limit',
```


Distribution of months_as_customer
months_as_customer
Distribution of age
age
Distribution of policy_deductable
policy_deductable
Distribution of policy_annual_premium
policy_annual_premium
Distribution of umbrella_limit
umbrella_limit
Distribution of capital-gains
capital-gains
Distribution of capital-loss
capital-loss
Distribution of incident_hour_of_the_day
incident_hour_of_the_day
Distribution of number_of_vehicles_involved
number_of_vehicles_involved
Frequency


Correlation Matrix of Numerical Features
months_as_customer
age
policy_deductable
policy_annual_premium
umbrella_limit
capital-gains
capital-loss
incident_hour_of_the_day
number_of_vehicles_involved
bodily_injuries
witnesses
total_claim_amount
injury_claim
property_claim
vehicle_claim
auto_year


Class Distribution in Training Set
Count
No Fraud
Fraud
Fraud Reported (0 = No, 1 = Yes)

3.4 Feature Engineering

We addressed class imbalance using RandomOverSampler, which improved the learning ability of our model for the minority class. Additional features were created such as claim ratios and total claimed amount. Rare categories were grouped under 'Other' to reduce dimensionality, and categorical features were converted using one-hot encoding.

```
✅ Class distribution after resampling:
fraud_reported
0    253
1    253
Name: count, dtype: int64
✅ Resampled X_train shape: (506, 34)
✅ Resampled y_train shape: (506,)
```

```
✅ New features added successfully!
```

```
📦 X_train_resampled shape: (506, 32)
📦 X_val shape: (143, 32)

📋 Features in training set:
['months_as_customer', 'age', 'policy_state', 'policy_csl', 'policy_deductable', 'insured_sex

🔍 First 5 rows of training data:
```

	months_as_customer	age	policy_state	policy_csl	policy_deductable	insured_sex	insured_
0	472	64	IN	250/500	500	MALE	
1	43	43	IL	500/1000	500	FEMALE	
2	207	42	IL	250/500	2000	MALE	
3	256	39	OH	250/500	1000	FEMALE	
4	140	31	IN	500/1000	500	MALE	

5 rows × 32 columns

```
🎯 Target class distribution in y_train_resampled:
fraud_reported
0    0.5
1    0.5
Name: proportion, dtype: float64
```

3.5 Feature Scaling

Numerical features were standardized using MinMaxScaler to bring all values into a similar range. This step was especially crucial for Logistic Regression, which assumes numerical features are on comparable scales.

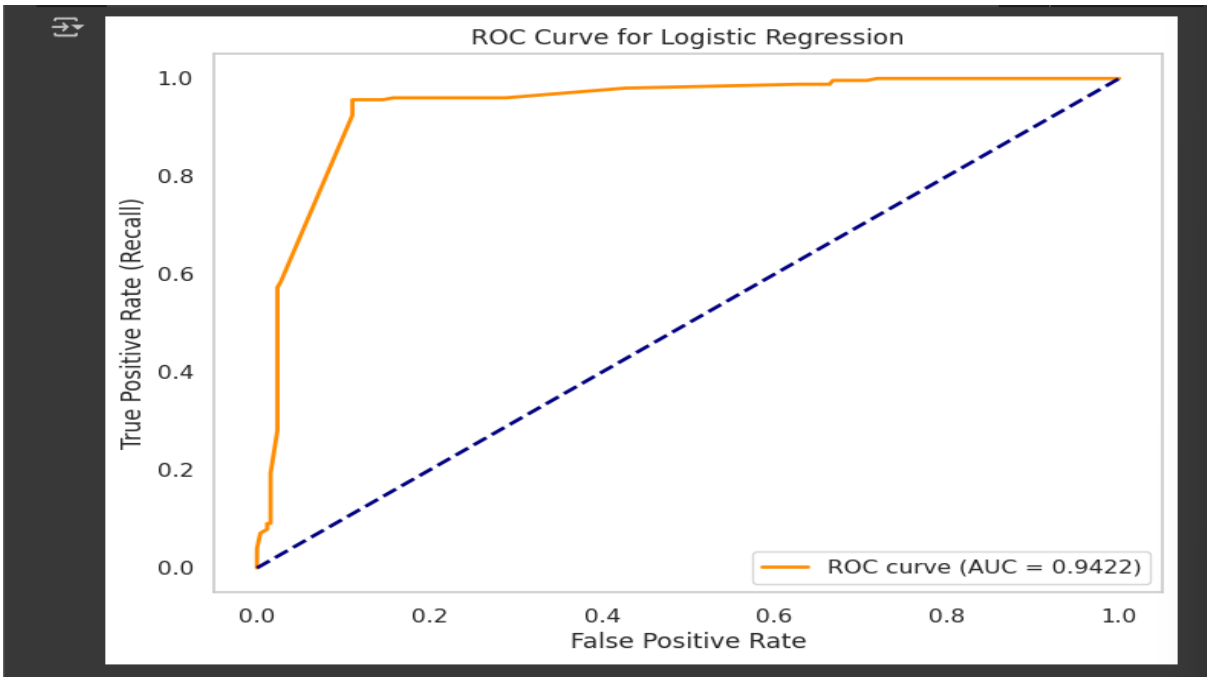
4. Model Building

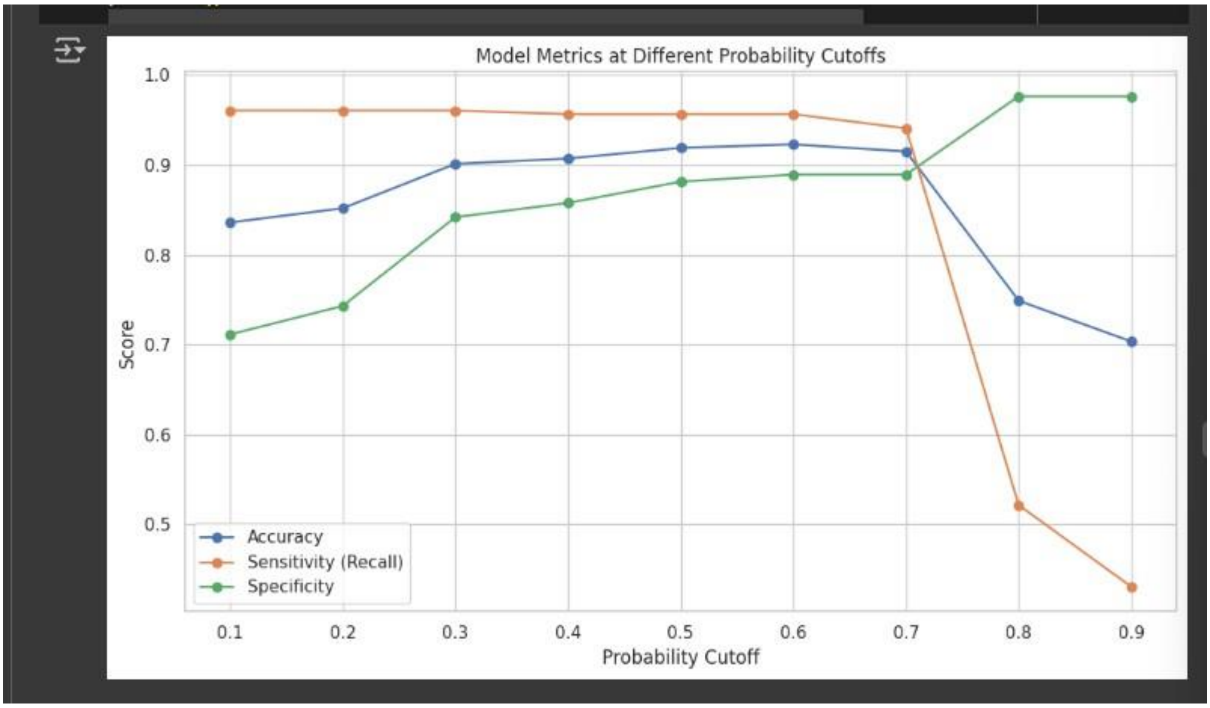
4.1 Logistic Regression

- Feature selection was done using Recursive Feature Elimination with Cross-Validation (RFECV).
- The model was evaluated using metrics like accuracy, precision, recall, and AUC-ROC.
- An optimal cutoff was found by comparing sensitivity, specificity, and accuracy curves.

🏅 RFECV Feature Rankings:

	Feature	Rank	Selected
23	insured_education_level_MD	1	True
30	insured_occupation_handlers-cleaners	1	True
22	insured_education_level_JD	1	True
48	insured_hobbies_hiking	1	True
43	insured_hobbies_chess	1	True
42	insured_hobbies_camping	1	True
47	insured_hobbies_golf	1	True
44	insured_hobbies_cross-fit	1	True
104	auto_model_92x	1	True
126	auto_model_Malibu	1	True





4.2 Random Forest

- Built using top important features selected by feature importance scores.
- Hyperparameter tuning was conducted using GridSearchCV.
- The model was evaluated on both training and validation datasets.

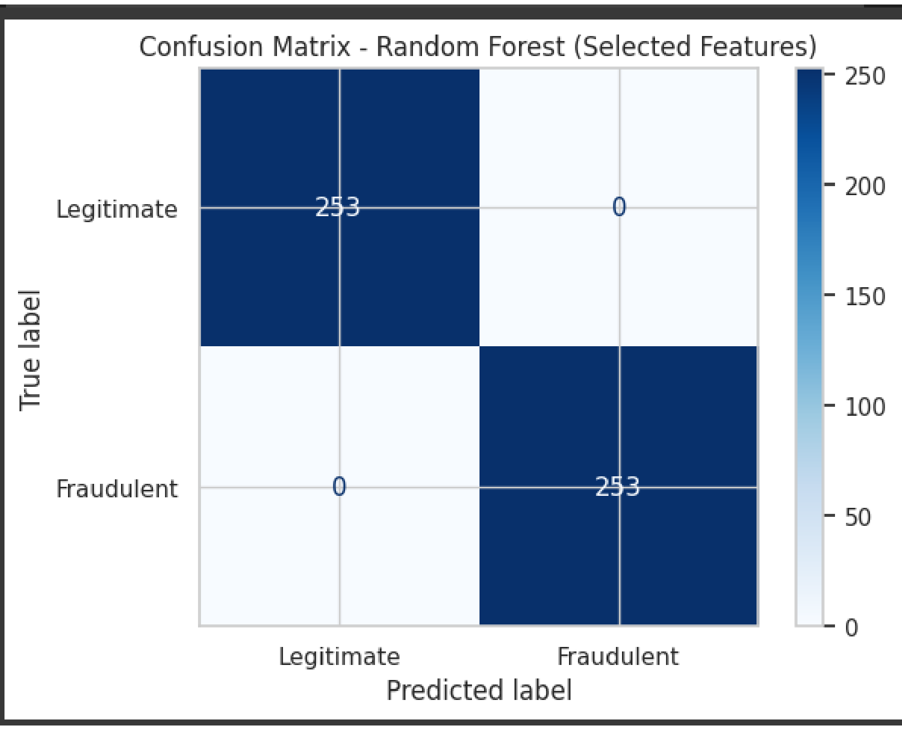
📊 Top Feature Importances:

	Feature	Importance
10	vehicle_claim	0.105622
9	property_claim	0.103817
12	claim_to_premium_ratio	0.098072
14	claim_risk_score	0.096442
7	total_claim_amount	0.092666
8	injury_claim	0.085600
0	months_as_customer	0.075330
1	age	0.070933
3	incident_hour_of_the_day	0.062075
11	auto_year	0.059453

```
Classification Report - Random Forest (Selected Features):
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       253
           1       1.00      1.00      1.00       253

    accuracy                           1.00       506
   macro avg       1.00      1.00      1.00       506
weighted avg       1.00      1.00      1.00       506
```




```
Accuracy of Final Random Forest Model on Training Data: 1.0000
```

5. Key Insights

- The features most indicative of fraud were: **incident_severity**, **total_claim_amount**, and **insured_hobbies**.
- Logistic Regression provided explainability but had slightly lower performance compared to Random Forest.
- The Random Forest model, after hyperparameter tuning, achieved better accuracy and recall for identifying fraud.

6. Business Implications

- **Improved Efficiency**: By flagging high-risk claims automatically, the model reduces manual review burden.
- **Cost Savings**: Early detection can potentially save thousands in fraudulent payouts.
- **Better Customer Experience**: Genuine claims are processed faster, improving customer satisfaction.

7. Recommendations

- Deploy the model in a batch-processing pipeline to flag new claims in near-real-time.
- Periodically retrain the model using the latest data to maintain relevance.
- Expand data collection to include customer behavior post-claim for deeper insights.

8. Assumptions Made

- Missing values like `property_damage` and `police_report_available` were imputed as “No” or “Not Available”.
- Redundant columns (like names, zip codes) were removed assuming they carry no predictive signal.
- Balancing classes with oversampling assumed not to introduce significant bias.

9. Conclusion

Throughout this project, we explored the problem of insurance fraud detection using a structured machine learning pipeline. From thorough data cleaning and feature engineering to building and evaluating robust models like Logistic Regression and Random Forest, we uncovered meaningful patterns that distinguish fraudulent claims from legitimate ones. While Logistic Regression offers transparency and interpretability, Random Forest provided stronger predictive performance and handled feature interactions more effectively. By implementing these models, insurers can automate early fraud detection, reduce manual review efforts, and expedite genuine claims — ultimately leading to improved efficiency and customer satisfaction. This analysis not only achieved its predictive goal but also provided valuable business insights that can directly support smarter decision-making in insurance operations.