Fraud Claim Detection

submitted by

Himanshu Agrawal
and
Jeevetha K C

Problem Statement:

- Global Insure company faces significant financial losses due to fraudulent insurance claims.
- The company wants to adopt a faster and data-driven approach to identify fraud early in the claims process.

Business Objective:

- The company aims to build a classification machine learning model that classifies claims as fraudulent or legitimate.
- Using historical claim details, customer profiles, and claim types, the goal is to predict fraud early, reduce financial losses, and streamline the claims approval process.

Dataset Information

- Dataset contains 1000 rows X 40 columns.
- Target variable is ‘fraud_reported’ which contains ‘Y’ as fraudulent and ‘N’ as legitimate.
- Distribution of target variable is as follows:

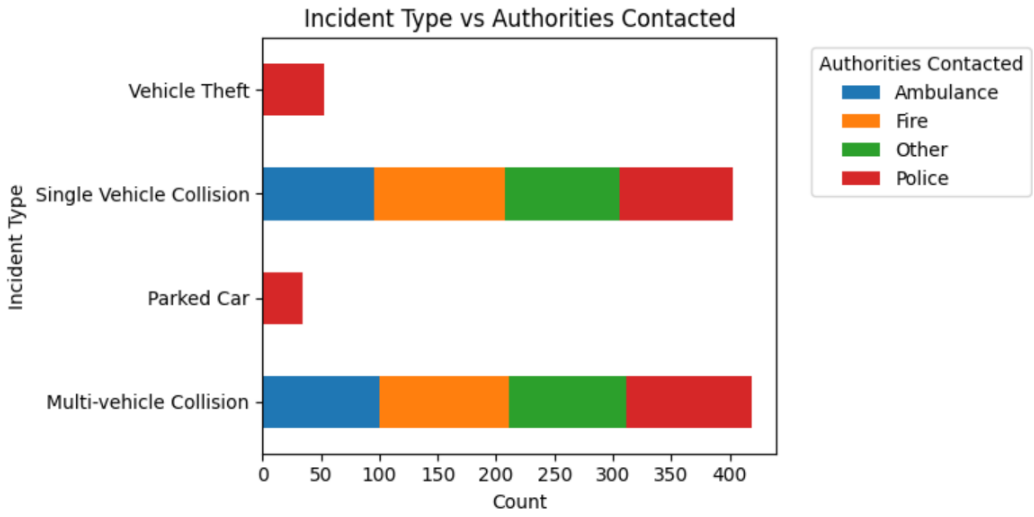
fraud_reported	count	percentage
N	753	75.3
Y	247	24.7

- Classes of target variable are somewhat imbalanced.

Overall Approach

1. Data Preparation

- Column '_c39' was empty and 'authorities_contacted' had 91 null values.
- **Null Values Treatment** - Checked effect of other column like 'incident_type' on 'authorities_contacted' to impute null values and we found that all the null values are related to 2 categories ('Vehicle_Theft' and 'Parked_Car') in 'incident_type' which are ultimately related to 'Police' only and hence imputed null values with 'Police'.




```
nulls when incident_type is Vehicle Theft = 41
nulls when incident_type is Parked Car = 50
nulls when incident_type is Vehicle Theft OR Parked Car = 91
total nulls in authorities_contacted = 91
```

2. Data Cleaning

- Removed empty column ‘c_39’.
- Converted date type columns which were objects, into datetime columns.
- Checked categorical columns and their value_counts
- Column ‘incident_location’ contained all 1000 unique values, hence dropped the column.
- Checked numerical columns and their unique values.
- One of the rows in ‘umbrella_limit’ was -ve which is not correct, hence, dropped that row.
- ‘capital-loss’ column values were <= 0 which was for representing loss, converted all of them into +ve values for simplicity.
- ‘policy_number’, ‘insured_zip’ had almost all the values unique, hence removed these columns.
- Extracted day, month, year from datetime columns.
- After extracting, dropped ‘incident_year’ as it has only 1 unique value.

3. Train Validation Split

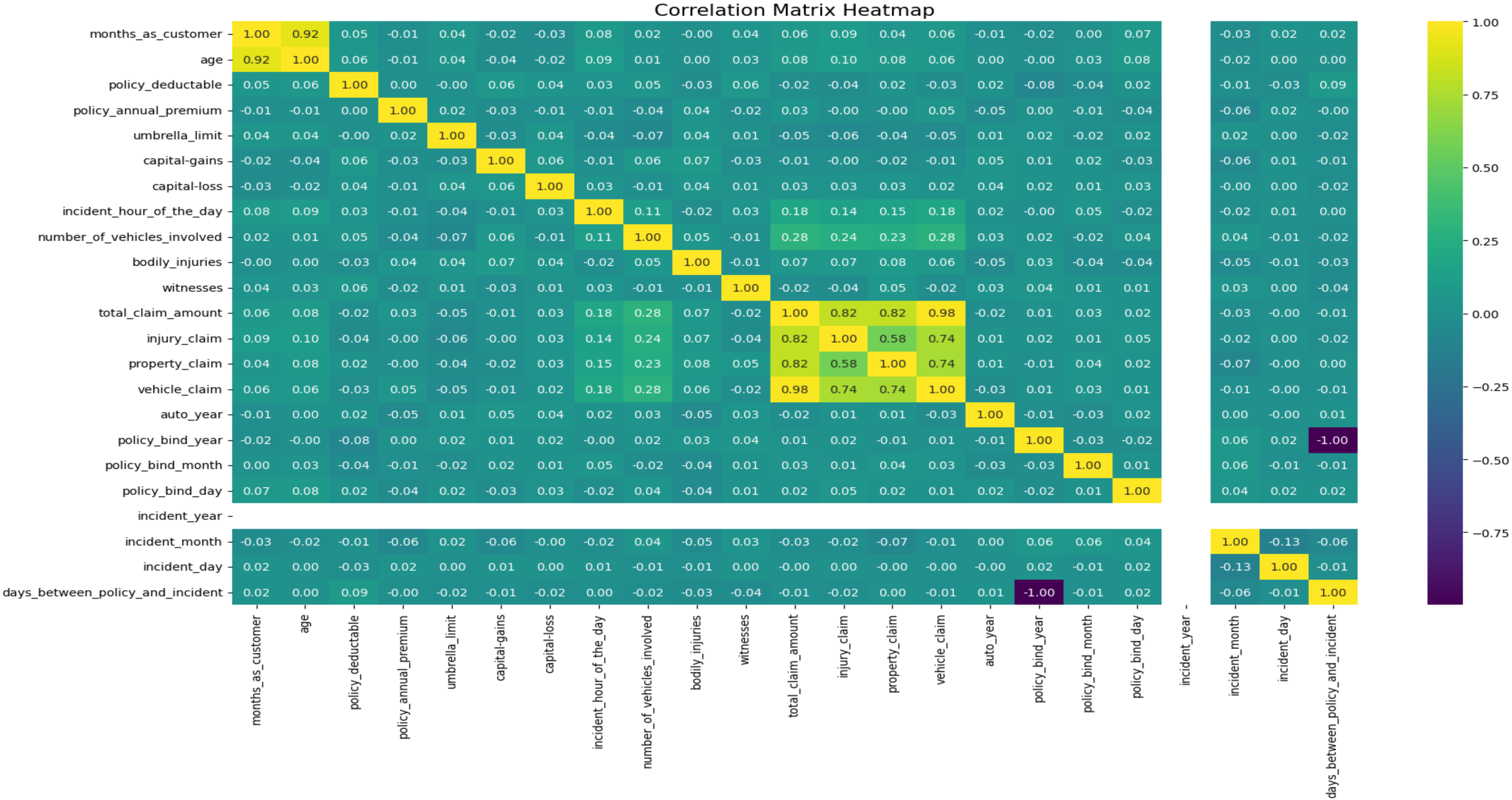
- Split the data into training set (70%) and testing set (30%) using stratified sampling which handles percentage of observations in both train and test sets with respect to classes of target variable.
- Shape of train set is (699, 41)
- Shape of test set is (300, 41)

4. EDA on Training set

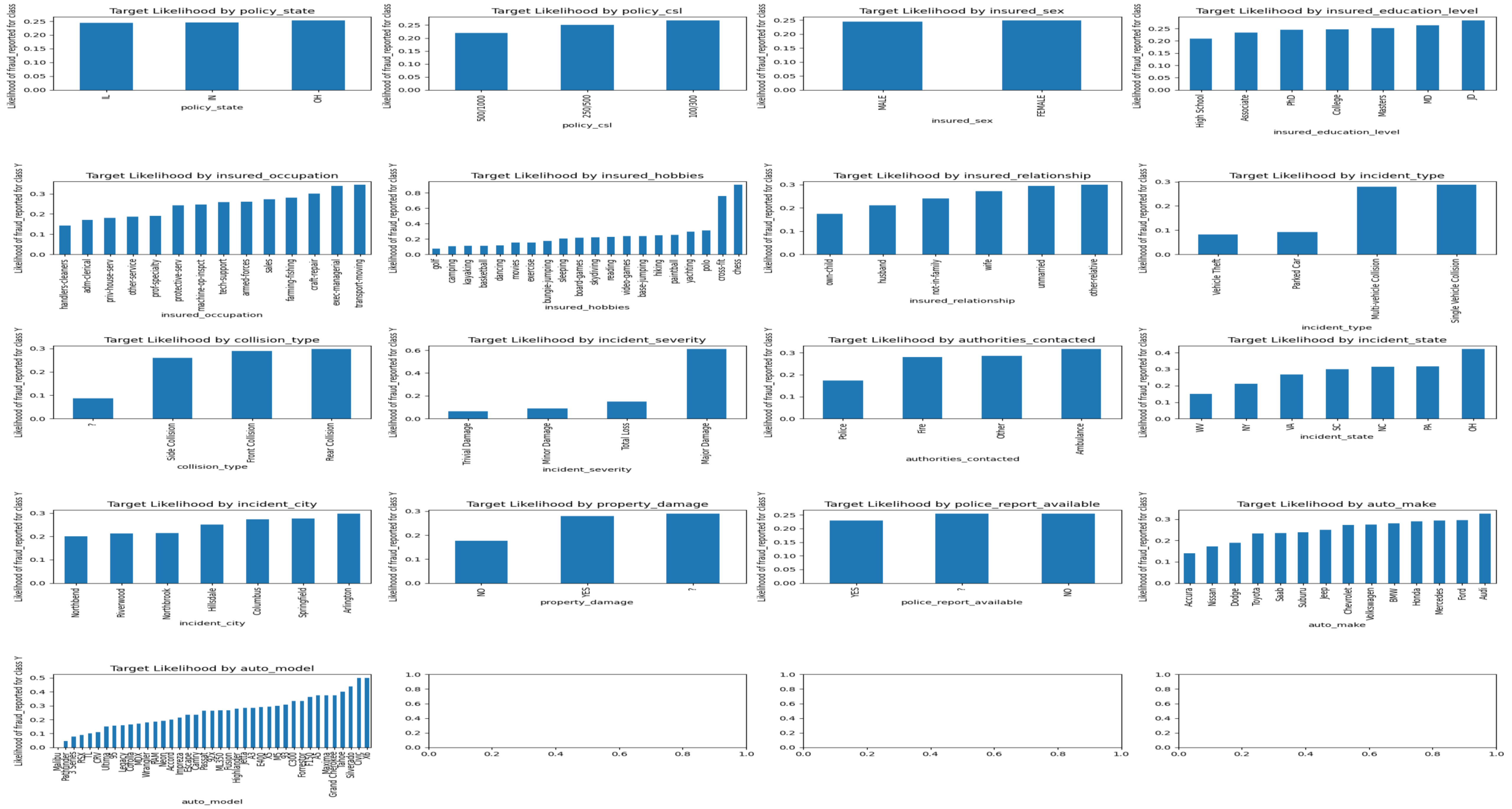
- Visualised distribution of numerical columns using univariate histograms.

- Visualised distribution of numerical columns using univariate boxplots.

- Plotted **Correlation Matrix Heat Map**.

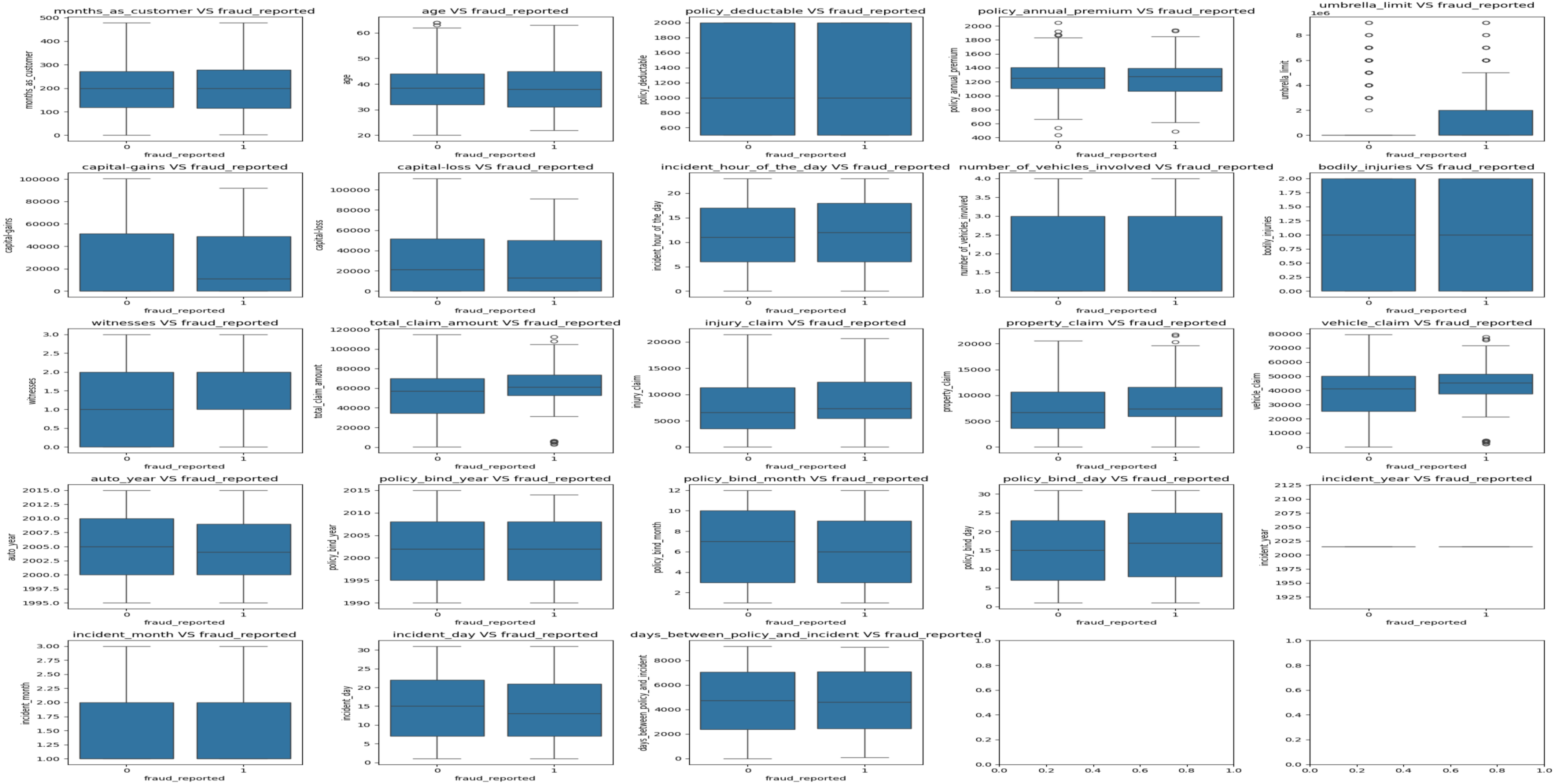


- Performed **Target Likelihood Analysis**



- In column 'insured_hobbied', persons whose hobbies are 'cross-fit' and 'chess', contain very high likelihood of being fraud which is around 80%.
- In column 'incident_severity', if it is 'Major Damage', there is around 60% chances of being fraud.
- Removed 'policy_state', ‘insured_sex’ and ‘insured_education_level’ columns as all the respective categories in these columns have almost equal contribution which makes the columns totally redundant.
- Column ‘auto_model’ alone has 39 categories. We tried to make logistic regression model with this column, but model failed to converge and made singular matrix. Hence, we dropped this column which also doesn’t look impactful.
- Correlation b/w 'age' & 'months_as_customer’ is 0.92, means both are very highly and positively correlated.
- 'policy_bind_year’ & 'days_between_policy_and_incident' are completely correlated as correation coefficient b/w them is 1.
- Hence, dropped 'months_as_customer’ and 'days_between_policy_and_incident’.

- Visualized **bivariate boxplots** for all the numerical variables vs target variable.



- Likelihood of being fraud is little bit higher if ‘total_claim_amount’ is higher.

5. Feature Engineering

- **Resampling** - Performed resampling for balancing class imbalances in target variable using RandomOverSampler.

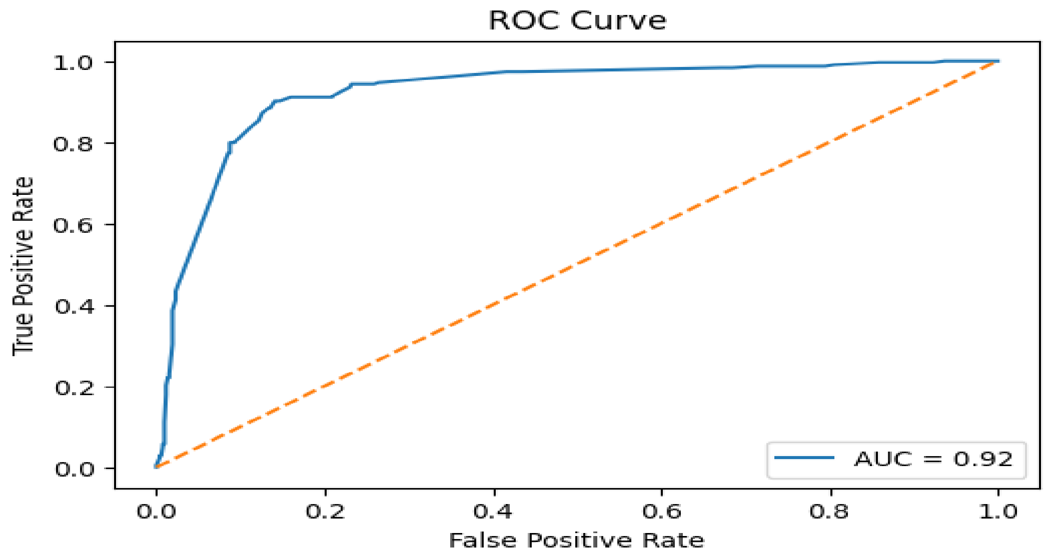
```
Imbalanced y
 fraud_reported
N     526
Y     173
Name: count, dtype: int64

Balanced y
 fraud_reported
N     526
Y     526
Name: count, dtype: int64
```

- 'total_claim_amount' was sum of 3 columns 'injury_claim', 'property_claim', 'vehicle_claim' and all are highly positively correlated with each other. Hence kept 'total_claim_amount' only and dropped other columns.
- **Dummy Variable Creation** – created dummy variables using one hot encoding and drop_first = True.
- **Scaling** – applied StandardScaler to all the numerical variables except day (1-31) and month (1-12) columns as they are cyclic in nature. Used Sin and Cos conversion and made separate Sin and Cos columns for all day, month columns and removed original ones.
- Now, shape of training data is (1052, 100)

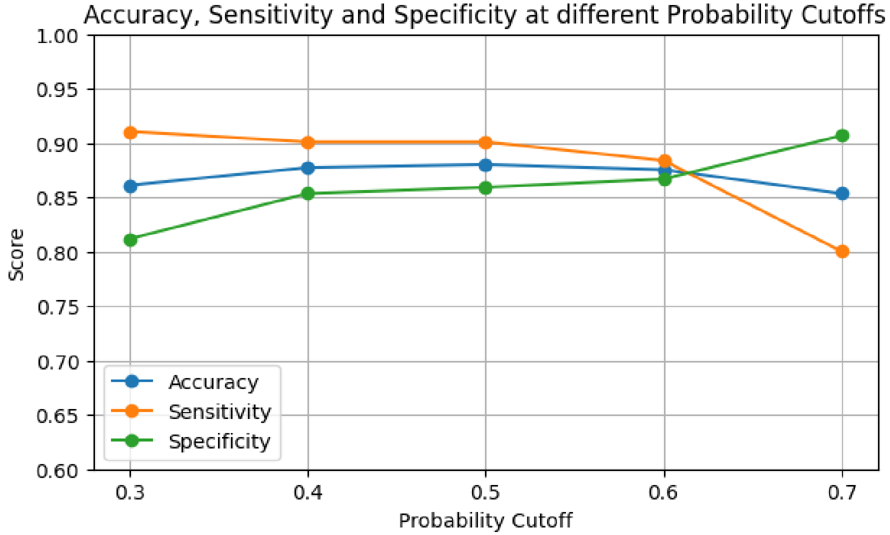
6. Model Building

6.1 Logistic Regression

- Applied **RFECV** for feature selection. 14 features were selected.
- Used Statsmodel library to make this model.
- Checked p-values and VIFs which were < 0.05 and < 5 respectively for all features. So, there is no multicollinearity and model is good.
- Predicted probabilities for training data set and set cutoff = 0.5 and Accuracy = 0.88
- Created Confusion Matrix and calculated sensitivity, specificity, recall, precision and F1-score.
- **ROC Curve** – to find optimal cutoff, plotted ROC curve and checked AUC score.



- Set different probability cutoffs = [0.3, 0.4, 0.5, 0.6, 0.7] and predicted accuracy, sensitivity and specificity at each cutoff and plotted it.



- Used Statsmodel library to make this model.
- 0.6 seems the best cutoff as all scores are nearby at 0.6.
- **Accuracy** at 0.6 cutoff = **0.8755**

```
sensitivity: 0.88
specificity: 0.87
precision: 0.87
recall: 0.88
f1-score: 0.88
```

6.2 Random Forest

- Derived feature importance using Random Forest model importance score.
- Selected 24 features where importance score > 0.01.
- Built a **simple random forest model** with above selected features.
- **Training Accuracy = 1.0**
- Sensitivity, specificity, recall, precision all were 1.0
- Used cross validation scores for 5 folds to check if it is overfitting.
- **Cross Validation Accuracy Mean = 0.9392**
- Clearly, above model is overfitting.
- **Hyperparameter Tuning** – used GridSearchCV to find best parameters which were found as {'max_depth': 10, 'min_samples_leaf': 5, 'min_samples_split': 10, 'n_estimators': 100}
- Built another RF model using above parameters.
- Now **training accuracy = 0.9477**

```
sensitivity: 0.97
specificity: 0.92
precision: 0.93
recall: 0.97
f1-score: 0.95
```

7. Prediction and Model Evaluation

- Predicted accuracy, sensitivity, specificity, recall, precision, f1-score for our validation set by the final models – **Logistic Regression (using cutoff = 0.6)** and **Random Forest (with hyperparameter tuning)**.
- The summary of all the scores is tabulated as below:

Model	Data	Accuracy	Sensitivity	Specificity	Precision	Recall	F1-Score
Logistic Regression	Train	0.8755	0.88	0.87	0.87	0.88	0.88
Logistic Regression	Test	0.8433	0.81	0.85	0.65	0.81	0.72
Random Forest	Train	0.9477	0.97	0.92	0.93	0.97	0.95
Random Forest	Test	0.8067	0.68	0.85	0.6	0.68	0.63

- **Logistic Regression model gives better prediction** and did not overfit the data. It performs consistently on both train and test datasets.
- On the other hand, Random Forest clearly overfits the data. It shows strong performance on the training data but poor generalization on the test data.

8. Conclusion

- **Logistic Regression** performed consistently on both training and test sets, showing good generalization.
- Random Forest performed extremely well on the training set but showed poor performance on the test set indicating possible overfitting.
- Features like **'insured_hobbies'**, especially **'cross-fit'** and **'chess'**, had **high predictive power** — individuals with these hobbies showed a **much higher likelihood of fraud.**
- Other important variables were 'total_claim_amount', 'incident_severity', 'policy_bind_date', 'incident_date', 'policy_annual_premium', 'age' etc.
- Logistic Regression is more stable and reliable for this fraud detection task. It shows better generalization without overfitting.

9. Recommendations

- Prefer Logistic Regression for production unless Random Forest can be further tuned. We can focus more on feature engineering or hyperparameter tuning to capture fraud patterns even better.
- Advanced resampling techniques such as SMOTE can be applied to balance class imbalances.
- Feature selection using RFECV with Random Forest model fitted can be done as we had fit the Logistic Regression model to RFECV for feature selection.
- Different training data can be fitted to Logistic Regression and Random Forest models as Random Forest can handle multicollinearity itself.

Ques 1 – How can we analyze historical claim data to detect patterns that indicate fraudulent claims?

By applying Exploratory Data Analysis (EDA) which includes null values treatment, univariate and bivariate analysis through visualization, removing redundant values, statistical analysis, and building machine learning models for classification like Logistic Regression, Naïve Byes, Decision Tree, Random Forest to identify anomalies and patterns associated with fraud.

Ques 2 – Which features are the most predictive of fraudulent behavior?

Top features include:

- Insured_hobbies (Notably cross-fit and chess)
- total_claim_amount
- incident_severity
- policy_bind_date
- incident_date
- policy_annual_premium
- age

Ques 3 – Based on past data, can we predict the likelihood of fraud for an incoming claim?

Yes, by training machine learning models like Logistic Regression or Random Forest on historical claim data, we can learn patterns associated with fraud. Logistic Regression, in particular, demonstrated consistent performance on both training and test sets, making it reliable for real-world prediction. When a new claim comes in, the model can output a probability score indicating the likelihood of fraud, helping prioritize high-risk claims.

Ques 4 – What insights can be drawn from the model that can help in improving the fraud detection process?

- **Logistic Regression** performed consistently on both training and test sets, showing good generalization.
- Random Forest performed extremely well on the training set but showed poor performance on the test set indicating possible overfitting.
- Features like **'insured_hobbies'**, especially **'cross-fit'** and **'chess'**, had **high predictive power** — individuals with these hobbies showed a **much higher likelihood of fraud.**
- Other important variables were 'total_claim_amount', 'incident_severity', 'policy_bind_date', 'incident_date', 'policy_annual_premium', 'age' etc.
- Logistic Regression is more stable and reliable for this fraud detection task. It shows better generalization without overfitting.
- Prefer Logistic Regression for production unless Random Forest can be further tuned. We can focus more on feature engineering or hyperparameter tuning to capture fraud patterns even better.
- Feature selection using RFECV with Random Forest model instead of Logistic Regression model can be checked.
- Different training data can be fit to Logistic Regression and Random Forest models as Random Forest can handle multicollinearity itself.

Thank You