# “ML for Insurance Claims Fraud Detection”

***Course: Data Science for Business***

***Study Program: Business Consulting Master***

***WS 24-25***

**Subhashri Ravichandran**

**Prof. Dr. Vinay Singh**

# Agenda

1. Problem Statement
2. Available Techniques
3. Data Source and Preprocessing
4. Supervised Model
5. Summary

# 1. Problem Statement

- In the past, fraud detection was left to insurance fraud investigators.
- The situation improved when rule-based systems appeared. It operates on a set of “rules”, that warn about potential fraud once it’s detected.

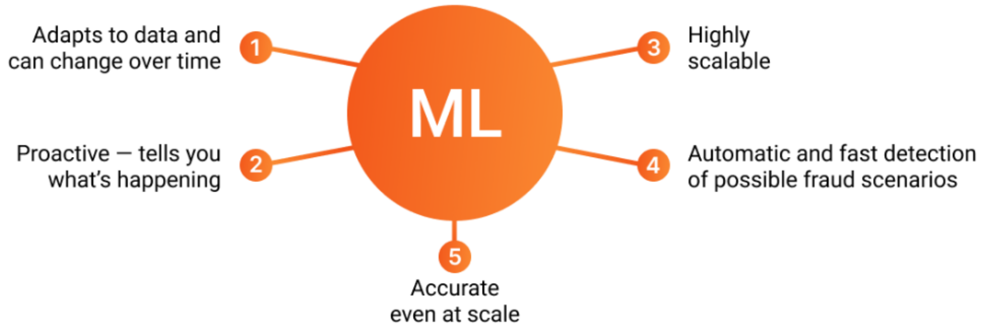
## Drawbacks of rule based systems

Blind spots

False positives

Effective only for simple cases

## Advantages with machine learning



# 2. Available Techniques

### Supervised

*Predicts fraudulent claims using labeled data (eg. claim amount, frequency)*

### Natural Language Processing

*Detects inconsistencies in supporting documents*

### Unsupervised

*Identifies anomalies and unknown fraud patterns* (e.g., unusually high repair costs)

### Graph based techniques

*Identifies connections between seemingly unrelated claims*

### Combined

*Ensemble methods combine anomaly detection and classification for enhanced fraud detection*

### Explainable AI

*Provides reasons for why a claim is fraudulent, thus builds trust*

# 3. Data Source and Preprocessing (1/3)

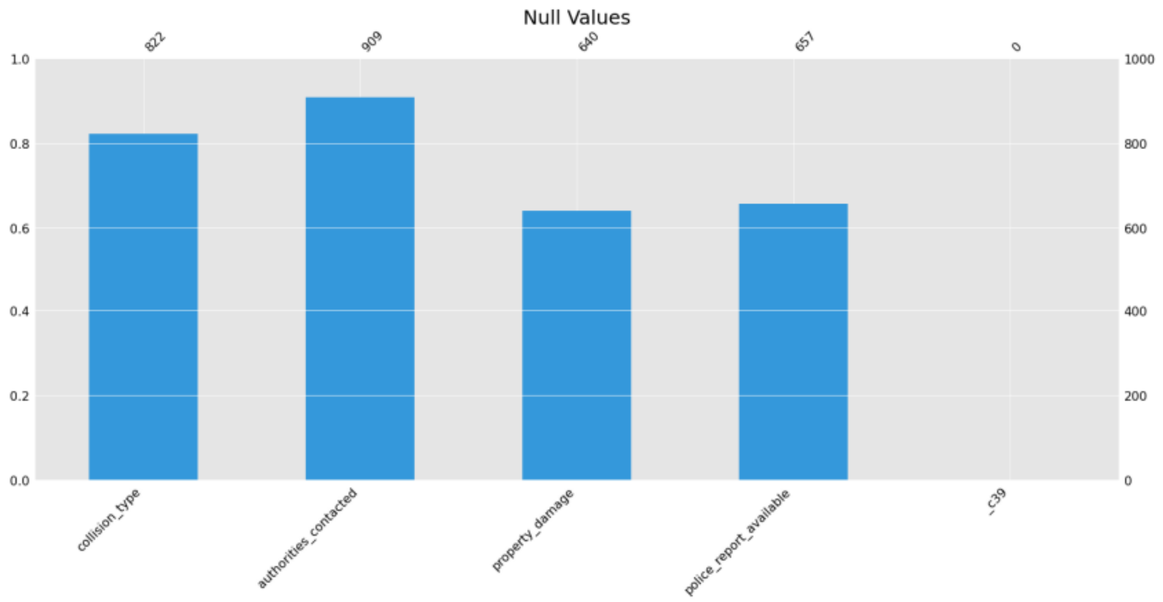
## Data Source

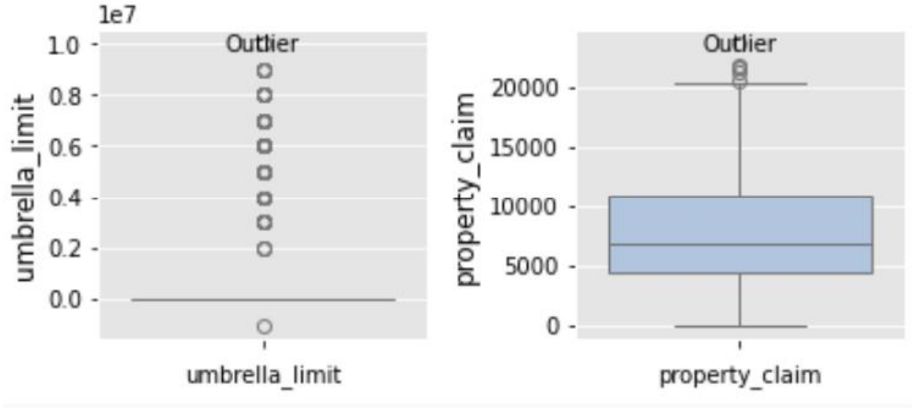
https://www.kaggle.com/datasets/buntyshah/auto-insurance-claims-data/data

## Data Set Overview:

| | | | | | |
|---|---|---|---|---|---|
| 1. | months_as_customer | : int64 | 21. | incident_severity | : object |
| 2. | age | : int64 | 22. | authorities_contacted | : object |
| 3. | policy_number | : int64 | 23. | incident_state | : object |
| 4. | policy_bind_date | : object | 24. | incident_city | : object |
| 5. | policy_state | : object | 25. | incident_location | : object |
| 6. | policy_csl | : object | 26. | incident_hour_of_the_day | : int64 |
| 7. | policy_deductable | : int64 | 27. | number_of_vehicles_involved | : int64 |
| 8. | policy_annual_premium | : float64 | 28. | property_damage | : object |
| 9. | umbrella_limit | : int64 | 29. | bodily_injuries | : int64 |
| 10. | insured_zip | : int64 | 30. | witnesses | : int64 |
| 11. | insured_sex | : object | 31. | police_report_available | : object |
| 12. | insured_education_level | : object | 32. | total_claim_amount | : int64 |
| 13. | insured_occupation | : object | 33. | injury_claim | : int64 |
| 14. | insured_hobbies | : object | 34. | property_claim | : int64 |
| 15. | insured_relationship | : object | 35. | vehicle_claim | : int64 |
| 16. | capital-gains | : int64 | 36. | auto_make | : object |
| 17. | capital-loss | : int64 | 37. | auto_model | : object |
| 18. | incident_date | : object | 38. | auto_year | : int64 |
| 19. | incident_type | : object | 39. | fraud_reported | : object |
| 20. | collision_type | : object | 40. | _c39 | : float64 |

# 3. Data Source and Preprocessing (2/3)




## Preprocessing:

- ✓ Replaced all occurences of '?' with NaN
- ✓ Filled null values with most frequent value

  collision_type, property_damage, police_report_available

- ✓ Dropped unwanted columns

  policy_number, policy_bind_date, policy_state and so on

- ✓ Encoded categorical columns

  incident_type, property_damage, police_report_available and so on

- ✓ Scaled numerical columns
- ✓ Balanced class distribution with SMOTE

# 3. Data Source and Preprocessing (3/3)

## Feature Selection:

### Features

```
In [90]: X_train.columns
Out[90]:
Index(['months_as_customer', 'policy_csl', 'policy_deductable',
       'policy_annual_premium', 'umbrella_limit', 'insured_sex',
       'insured_education_level', 'insured_occupation', 'insured_relationship',
       'capital-gains', 'capital-loss', 'incident_type', 'collision_type',
       'incident_severity', 'authorities_contacted',
       'incident_hour_of_the_day', 'number_of_vehicles_involved',
       'property_damage', 'bodily_injuries', 'witnesses',
       'police_report_available', 'injury_claim', 'property_claim',
       'vehicle_claim'],
      dtype='object')
```

### Label

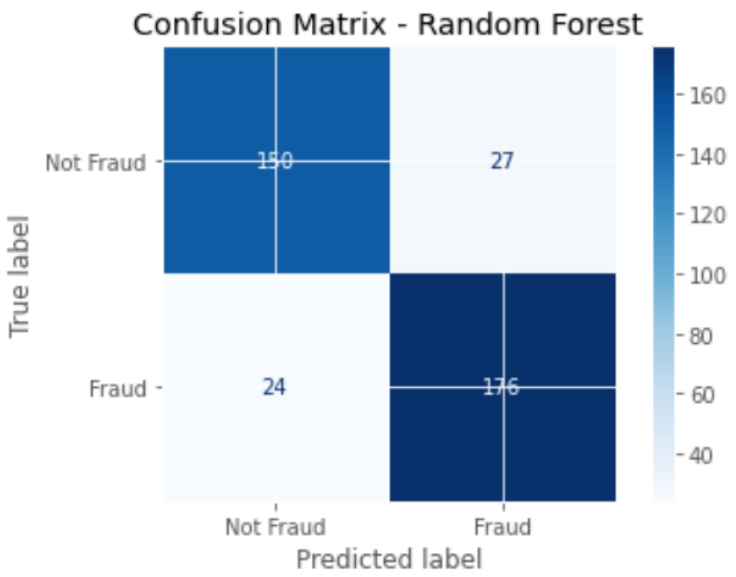
'fraud_reported'

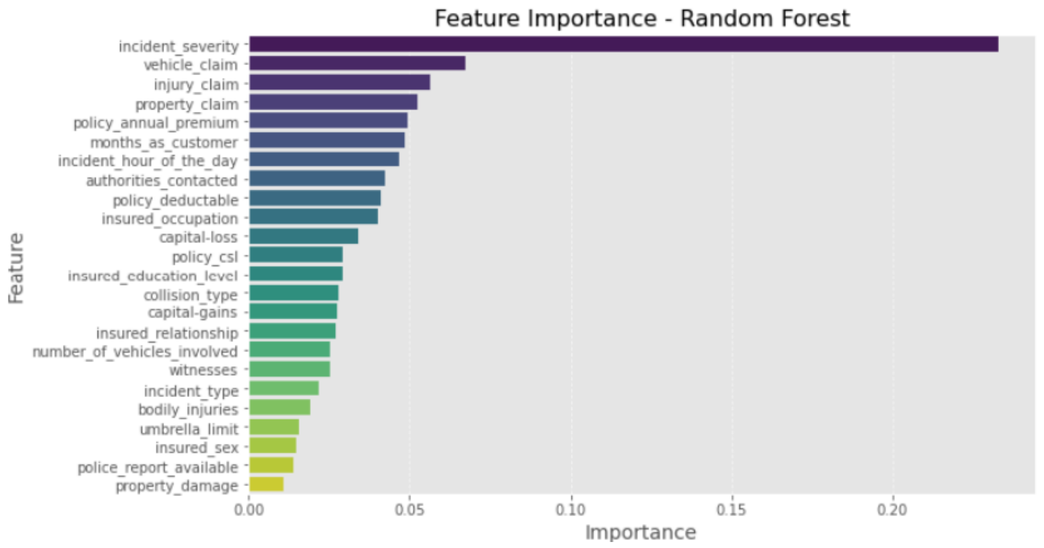
# 4. Supervised Model (1/2)

**Model:**
Random Forest Classifier

Train-test Split (75:25)

GridSearchCV (5-fold CV) to optimize key parameters based on ROC-AUC score



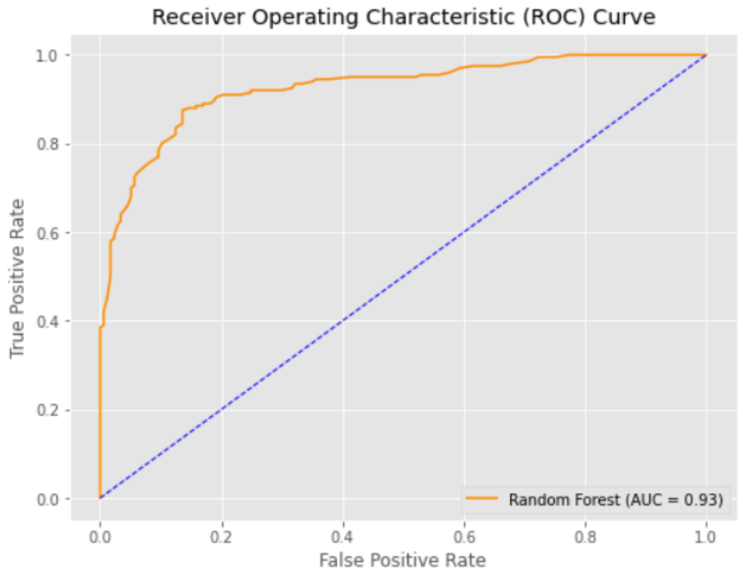

**Out of 200 Fraud Cases**
176 correctly predicted as fraud, 24 incorrectly predicted as non-fraud

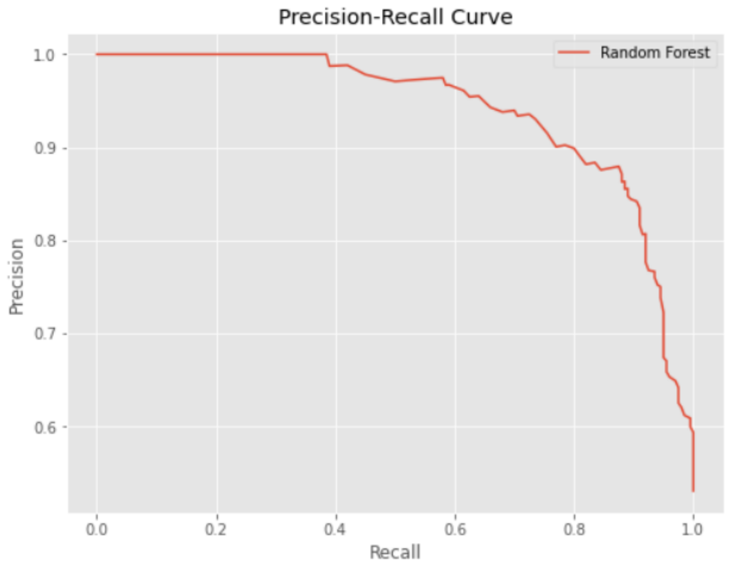
# 4. Supervised Model (2/2)

**Model:**
Random Forest Classifier

Train-test Split (75:25)

**Accuracy:** 86%




Classification Report (Default Threshold):

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.86 | 0.85 | 0.85 | 177 |
| 1 | 0.87 | 0.88 | 0.87 | 200 |
| accuracy | | | 0.86 | 377 |
| macro avg | 0.86 | 0.86 | 0.86 | 377 |
| weighted avg | 0.86 | 0.86 | 0.86 | 377 |

# 5. Summary

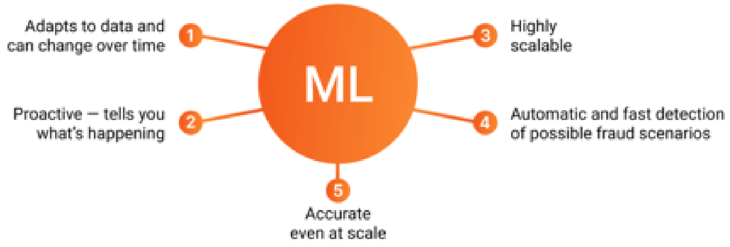
### Drawbacks of rule based systems

Blind spots

False positives

Effective only for simple cases

### Advantages with machine learning




**Supervised**
*Predicts fraudulent claims using labeled data (eg. claim amount, frequency)*

**Unsupervised**
*Identifies anomalies and unknown fraud patterns* (e.g., unusually high repair costs)

**Combined**
*Ensemble methods combine anomaly detection and classification for enhanced fraud detection*

**Natural Language Processing**
*Detects inconsistencies in supporting documents*

**Graph based techniques**
*Identifies connections between seemingly unrelated claims*

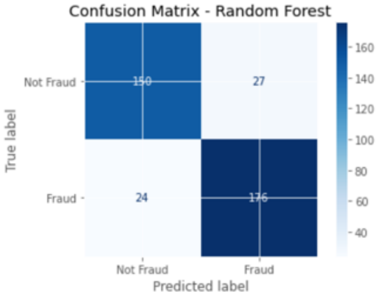
**Explainable AI**
*Provides reasons for why a claim is fraudulent, thus builds trust*

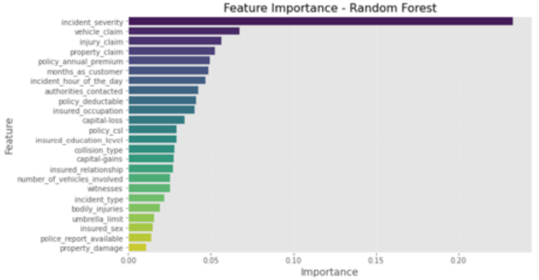
**Model:**
Random Forest Classifier

**Validation Technique:**

Test-train Split

**Accuracy:** 86%




**Out of 200 Fraud Cases**
176 correctly predicted as fraud, 24 incorrectly predicted as non-fraud

*Incorporating additional fraud indicators or exploring other models, could enhance accuracy and minimize errors in fraud detection*