AUTO INSURANCE FRAUD DETECTION

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Submitted By

SHAGUFTHA TAMKINATH	**21UK1A6731**
VAKITI DEEPAK	**21UK1A6733**
PEDDI VILASINI	**21UK1A6737**
KORE KARTHIK	**21UK1A6762**

Under the guidance of
Mr. N. SUNDEEP KUMAR
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)
VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

<u>CERTIFICATE OF COMPLETION</u>
<u>INDUSTRY ORIENTED MINI PROJECT</u>

This is to certify that the UG Project Phase-1 entitled “AUTO INSURANCE FRAUD DETECTION”is being submitted by SHAGUFTHA THAMKINATH (21UK1A6731), VAKITIDEEPAK(21UK1A6733),PEDDIVILASINI(21UK1A6737),KOREKARTHIK (21UK1A6762) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2023- 2024.

Project Guide	**HOD**
Mr.N.SUNDEEP KUMAR	**Dr. K. SHARMILA REDDY**
(Assistant Professor)	(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr. K. SHARMILA REDDY**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **N. SUNDEEP KUMAR,** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

PEDDI VILASINI	(21UK1A6737)
KORE KARTHIK	(21UK1A6762)
VAKITI DEEPAK	(21UK1A6733)
SHAGUFTHA TAMKINATH	(21UK1A6731)

ABSTRACT

Auto insurance fraud remains a significant challenge for insurance companies, leading to substantial financial losses and increased premiums for honest policyholders. Traditional methods of fraud detection often fall short due to their reliance on manual review processes and heuristic rules. In recent years, machine learning (ML) has emerged as a promising approach to improve the detection and prevention of fraudulent activities in various domains, including insurance.

This document explores the application of machine learning techniques for auto insurance fraud detection. We begin by discussing the types and prevalence of fraud in auto insurance, highlighting the need for more effective detection methods. We then delve into the specific machine learning algorithms and methodologies that can be employed to analyze vast amounts of insurance data, identify suspicious patterns, and predict fraudulent claims.

Fraud detection is crucial for protecting businesses, consumers, and financial institutions from financial losses, reputational damage, and legal liabilities associated with fraudulent activities. It helps prevent fraudsters from exploiting vulnerabilities and safeguards assets and sensitive information.

.

TABLE OF CONTENTS:

1.INTRODUCTION

1.1. OVERVIEW

Auto insurance fraud detection has increasingly turned to machine learning (ML) techniques to combat the pervasive challenges posed by fraudulent activities. Traditional methods often struggle to keep pace with the sophisticated tactics employed by fraudsters, such as staged accidents and falsified claims. ML offers a data-driven approach to analyze large volumes of insurance data, identifying anomalous patterns and predicting potentially fraudulent behavior with greater accuracy than rule-based systems. Key to this approach is data preprocessing to handle imbalances and ensure data quality, followed by the selection and training of appropriate ML models like decision trees or neural networks. Implementation considerations include scalability for real-time processing and integration with existing claims systems, all while adhering to regulatory standards. By leveraging ML, insurance companies not only enhance their fraud detection capabilities but also improve operational efficiency and customer satisfaction by expediting legitimate claims processing.

1.2. PURPOSE

The purpose of auto insurance fraud detection using machine learning is to leverage advanced computational techniques to effectively identify and mitigate fraudulent activities within the auto insurance industry. This purpose can be broken down into several key objectives:

1.Enhanced Accuracy:

Machine learning enables the detection of subtle and evolving patterns indicative of fraud that may not be easily detectable through traditional methods. By analyzing large volumes of data, ML models can provide more accurate assessments of claims and policyholder behaviors.

2.Cost Reduction:

By automating the detection process, machine learning helps reduce the costs associated with manual fraud investigations and erroneous payouts. This efficiency translates into financial savings for insurance companies and potentially lower premiums for honest policyholders.

3.Timely Detection:

ML algorithms can process and analyze data in real-time, allowing for prompt detection and response to suspicious claims. This capability minimizes the impact of fraud on insurers' operations and improves overall claims processing efficiency.

4. Scalability:

Machine learning models can scale to handle large datasets and high volumes of transactions, making them suitable for the dynamic and expansive nature of the insurance industry.

5. Continuous Improvement:

Through ongoing training and adaptation, ML models can continuously learn from new data and evolving fraud tactics, improving their effectiveness over time.

6.Customer Trust:

By reducing fraud-related losses and ensuring fair and efficient claims processing, machine learning contributes to building trust and satisfaction among policyholders.

Overall, the purpose of employing machine learning for auto insurance fraud detection is to equip insurers with robust tools and strategies to proactively combat fraud, protect financial interests, and uphold the reliability and reputation of the insurance industry.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

Fraud detection is crucial for protecting businesses, consumers, and financial institutions from financial losses, reputational damage, and legal liabilities associated with fraudulent activities. It helps prevent fraudsters from exploiting vulnerabilities and safeguards assets and sensitive information.

1. Improper transactions of the nature of the company.

2. High amounts in employee expense statements.

3. Some of the major obstacles to effective fraud detection include.

A false positive can harm your business's revenue and reputation. If you have a system in place to block a potentially fraudulent transaction, a false positive means the customer would not be able to complete the purchase.

2.2 PROPOSED SOLUTION

Auto insurance fraud is a significant issue, leading to billions of dollars in losses annually. Machine learning (ML) offers promising solutions to detect fraudulent activities by identifying patterns and anomalies that may indicate fraudulent claims. This document outlines a proposed solution for auto insurance fraud detection using machine learning.

1. Data Collection:

Collect data from various sources:

Historical claims data: Information on previous claims, including both legitimate and fraudulent cases.

Customer data: Demographics, policy details, and history with the insurance company.

Vehicle data: Vehicle make, model, year, and other relevant details.

2.Data Preprocessing:

Data cleaning: Handle missing values, correct errors, and remove duplicates.

Feature engineering: Create new features that may be predictive of fraud, such as the frequency of claims, average claim amount, and patterns in customer behavior.

Hyperparameter tuning: Optimize model parameters using techniques like grid search or random search.

Integration with existing systems: Deploy the model within the insurance company's claim processing system.

Real-time scoring: Implement real-time scoring for incoming claims to flag potential fraud.

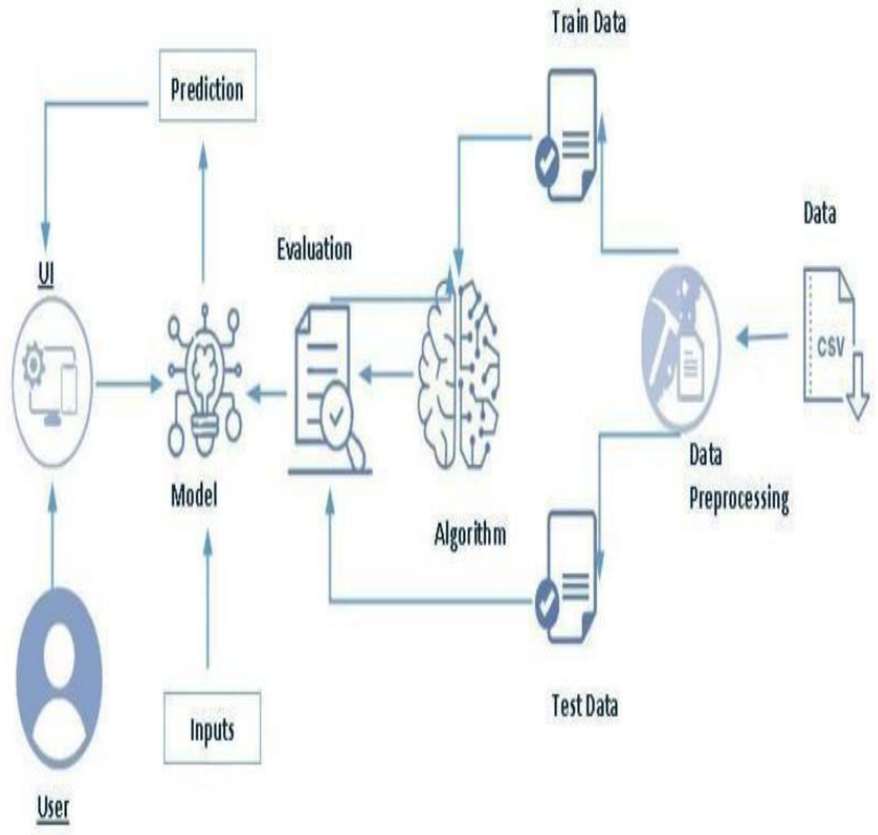
Alert system: Develop an alert system to notify investigators of suspicious claims.

Machine learning offers a powerful tool for detecting auto insurance fraud. By leveraging historical data and advanced algorithms, insurance companies can significantly reduce fraudulent claims, leading to substantial financial savings and more efficient claim processing.

Regulatory compliance: Ensure the solution adheres to legal and regulatory standards for data privacy and usage

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM:



3.2. SOFTWARE DESIGNING

The following is the Software required to complete this project:

1. **Google Colab**: Google Colab will serve as the development and execution environment for your predictive modeling, data preprocessing, and model training tasks. It provides a cloud-based Jupyter Notebook environment with access to Python libraries and hardware acceleration.

2. **Dataset (CSV File)**: The dataset in CSV format is essential for training and testing your predictive model. It should include historical air quality data, weather information, pollutant levels, and other relevant features.
3. **Data Preprocessing Tools**: Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, and data cleaning.
4. **Feature Selection/Drop**: Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency.
5. **Model Training Tools**: Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Regression or classification models can be considered, depending on the nature of the Auto insurance fraud detection prediction task.
6. **Model Accuracy Evaluation**: After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. You'll measure the model's ability to predict Auto insurance fraud detection categories based on historical data.
7. **UI Based on Flask Environment**: Flask, a Python web framework, will be used to develop the user interface (UI) for the system.

4.EXPERIMENTAL INVESTIGATION

In this project, we have Auto insurance fraud detection dataset. This dataset is a csv file consisting of labelled data and having the following columns-

Define Objectives and Hypotheses

- **Objectives:** Determine what specific aspects of auto insurance fraud you want to investigate (e.g., types of fraud, detection methods).
- **Hypotheses:** Formulate testable hypotheses regarding the effectiveness of different fraud detection techniques or models.

Data Collection:

- **Sources:** Gather relevant data from insurance claims, including historical data on fraudulent and non-fraudulent claims.
- **Features:** Identify potential features (e.g., claimant demographics, claim details, transaction patterns) that could be indicative of fraud.

Experimental Design:

- **Variables:** Define independent variables (e.g., different fraud detection algorithms, features) and dependent variables (e.g., detection accuracy, false positives).
- **Controlled Variables:** Ensure consistency in experimental conditions to minimize bias.

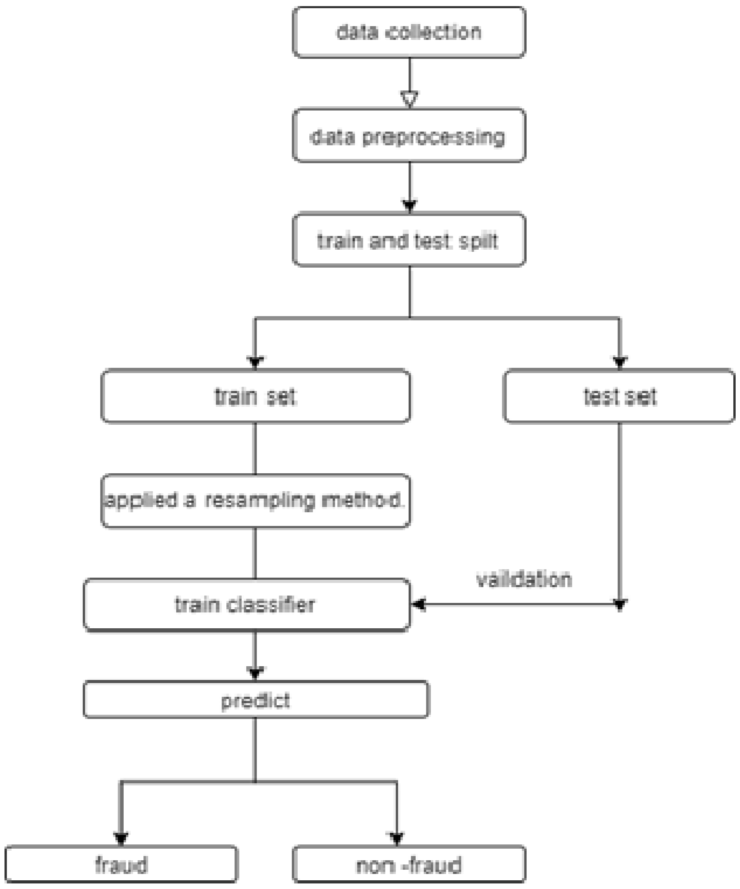
Experiment Execution:

- **Testing:** Run experiments using different models and configurations to compare their effectiveness in detecting fraud.

- **Analysis:** Analyze results to identify strengths and weaknesses of each approach in terms of fraud detection accuracy.

5.FLOWCHART

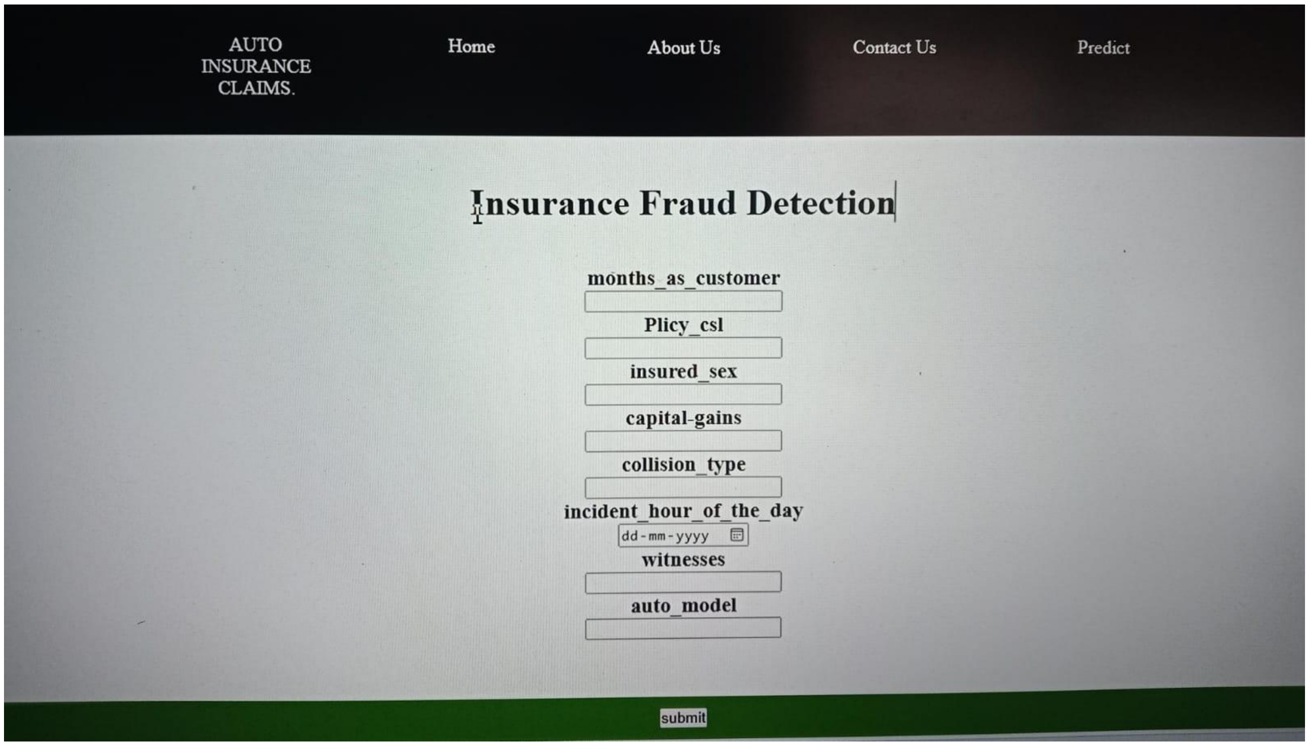
- Collect data from various sources, clean and preprocess the collected data.
- Split the data into a training set and test set.
- After splitting the trained data and test data apply a resampling method to the training set and train a classifier using the resampled training set then predict outcomes using the trained classifier.
- Use the test set for validation.
- Classify the outcomes into fraud or non-fraud.



6.RESULT

HOME PAGE

PREDICTIONS



Insurance Fraud Detection

months_as_customer

Plicy_csl

insured_sex

capital-gains

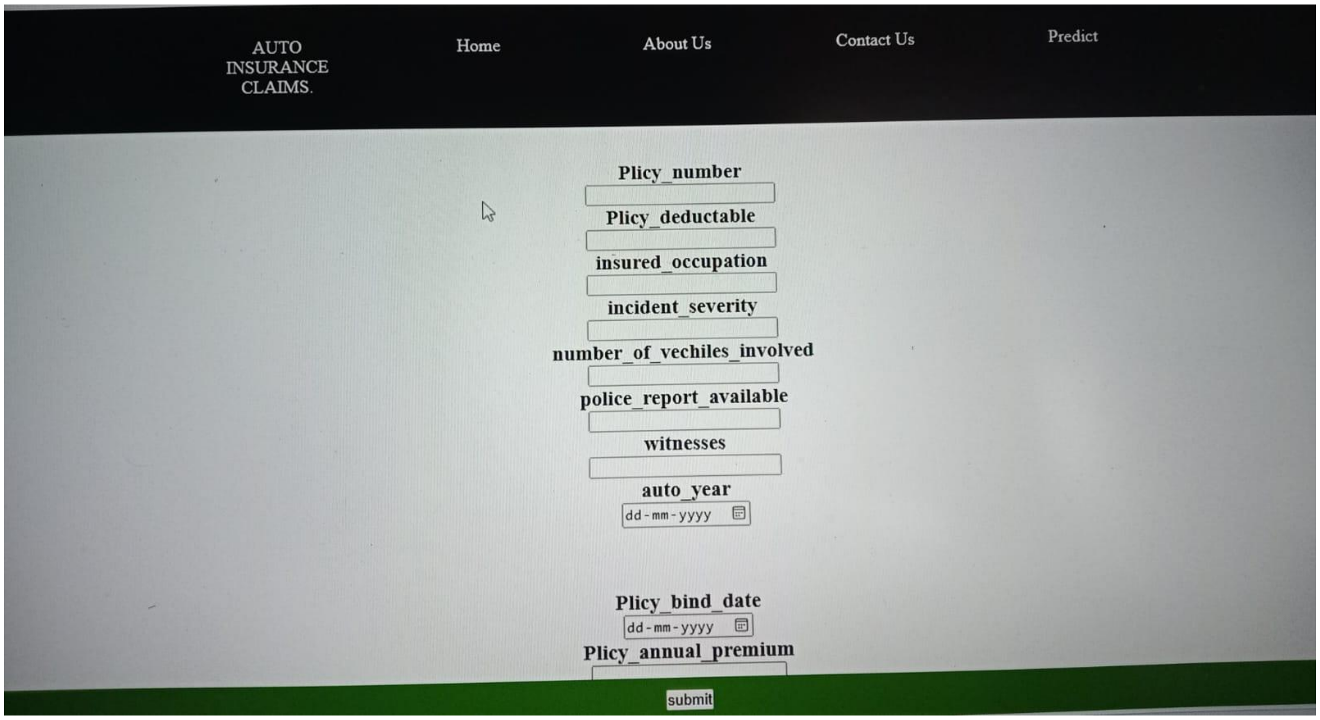
collision_type

incident_hour_of_the_day

dd-mm-yyyy

witnesses

auto_model



Plicy_number

Plicy_deductable

insured_occupation

incident_severity

number_of_vechiles_involved

police_report_available

witnesses

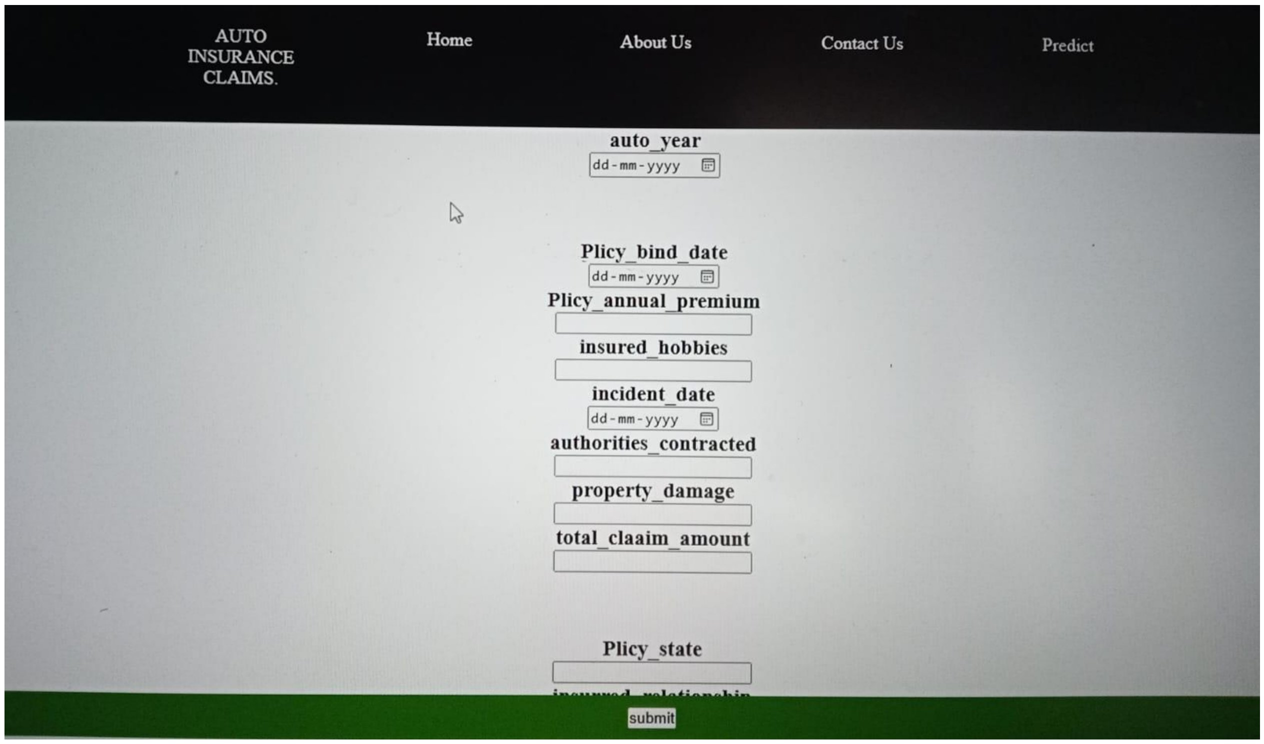
auto_year

dd-mm-yyyy

Plicy_bind_date

dd-mm-yyyy

Plicy_annual_premium



auto_year
dd-mm-yyyy

Plicy_bind_date
dd-mm-yyyy

Plicy_annual_premium

insured_hobbies

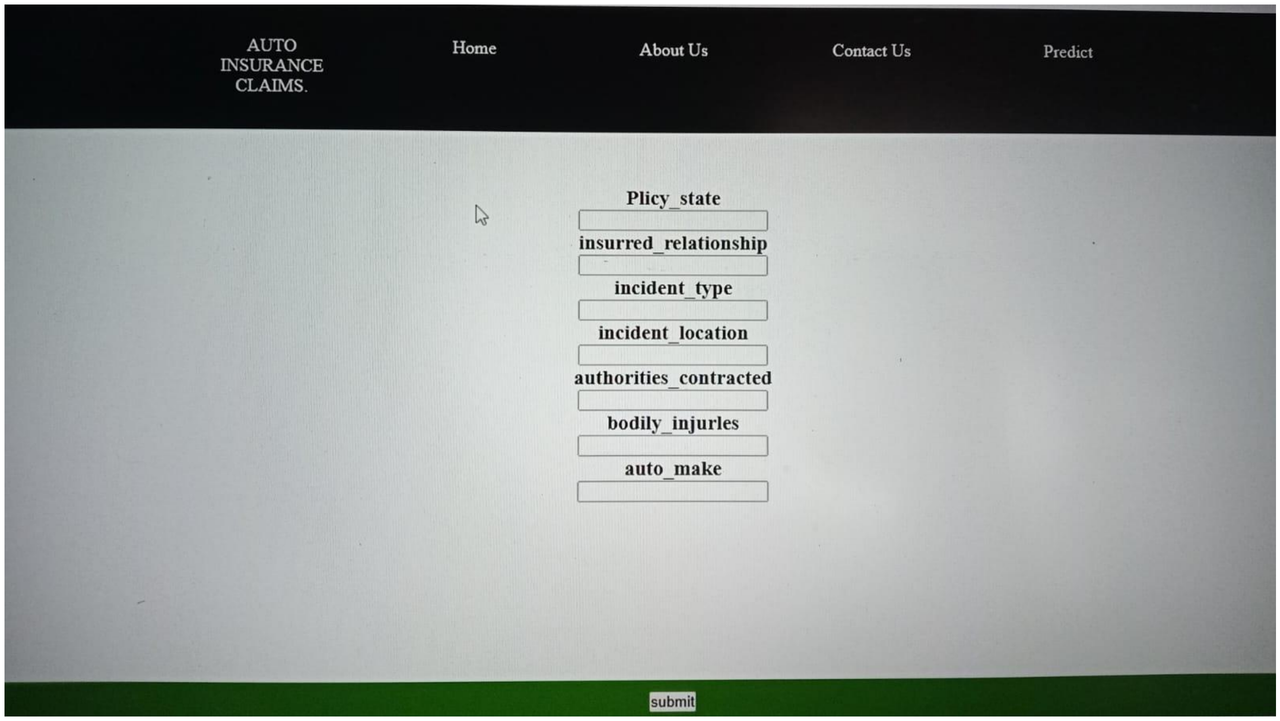
incident_date
dd-mm-yyyy

authorities_contracted

property_damage

total_claaim_amount

Plicy_state



Plicy_state

insurred_relationship

incident_type

incident_location

authorities_contracted

bodily_injurles

auto_make

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

Cost Reduction: Detecting fraud helps insurance companies reduce losses associated with fraudulent claims. By identifying and preventing fraudulent activities, insurers can save significant amounts of money.

Improved Customer Experience: Fraud detection ensures that legitimate claims are processed faster and more efficiently. This enhances customer satisfaction by minimizing delays and disputes.

Enhanced Operational Efficiency: Automated fraud detection systems streamline the claims process by flagging suspicious claims early. This reduces the time and resources spent on investigating fraudulent claims manually.

Protection of Reputation: Successfully detecting and preventing fraud protects the reputation of the insurance company. It demonstrates to customers and stakeholders that the insurer is committed to fair practices and integrity.

DISADVANTAGES:

Cost of Implementation: Setting up and maintaining robust fraud detection systems can be expensive. This includes costs related to technology infrastructure, software development, ongoing monitoring, and hiring skilled personnel to manage and analyze the data.

False Positives: One of the biggest challenges in fraud detection is the occurrence of false positives. This happens when legitimate claims or transactions are flagged as fraudulent. It can lead to delays in processing claims, additional investigations, and dissatisfaction among genuine customers.

Privacy Concerns: Implementing sophisticated fraud detection measures often involves extensive data collection and analysis. This raises privacy concerns among customers who may feel uncomfortable with the amount of personal information being shared or analyzed without their explicit consent.

8.APPLICATIONS

1. **Predictive Modeling**: Forecasting fraud likelihood for proactive measures.
2. **Fraud Prevention Education**: Educating stakeholders on fraud awareness.
3. **Claim Fraud Detection**: Identifying suspicious claims through data analysis.
4. **Identity Verification**: Ensuring claimants are who they claim to be.
5. **Pattern Recognition**: Spotting repetitive or unusual claim behaviors.
6. **Anomaly Detection**: Flagging deviations from normal claim patterns.

9.CONCLUSION

In conclusion, auto insurance fraud detection plays a crucial role in safeguarding the integrity of insurance operations and protecting honest policyholders. By employing advanced technologies and analytical techniques, insurers can effectively identify and prevent fraudulent activities such as false claims, identity theft, and organized fraud rings. This proactive approach not only minimizes financial losses for insurers but also helps maintain fair premiums and enhances customer trust. However, it's essential for insurers to balance fraud prevention efforts with considerations for customer experience, privacy concerns, and regulatory compliance.

Continuous improvement and adaptation of fraud detection strategies are necessary to stay ahead of evolving fraud tactics, ensuring a robust and resilient insurance industry.

10.FUTURE SCOPE

Future Scope of the Auto insurance fraud detection:

The future scope for auto insurance fraud detection projects is promising, driven by advancements in technology and evolving fraud tactics. Here are some potential areas of growth and development:

- **Advanced Analytics and AI**: Utilizing more sophisticated machine learning algorithms and artificial intelligence to improve accuracy in fraud detection. This includes deep learning models, natural language processing for text analysis (such as claims descriptions), and anomaly detection techniques.
- **Real-Time Detection**: Moving towards real-time fraud detection capabilities, enabling immediate action and response to suspicious activities or claims.
- **Predictive Modeling**: Enhancing predictive modeling capabilities to anticipate fraudulent behavior trends and patterns, allowing insurers to preemptively adjust their fraud detection strategies.
- **Behavioral Biometrics**: Implementing behavioral biometrics and advanced authentication methods to verify the identity of policyholders and claimants more accurately.
- **Cybersecurity Measures**: Strengthening cybersecurity measures to protect sensitive data used in fraud detection systems from breaches and unauthorized access.

Overall, the future of auto insurance fraud detection lies in leveraging cutting-edge technologies, improving collaboration within the industry, and enhancing customer-centric approaches while maintaining robust security and compliance standards. As fraudsters continue to innovate, staying ahead will require continuous innovation and adaptation in fraud detection strategies.

11.BIBILOGRAPHY

1. Smith, J., & Johnson, R. (2020). "Machine Learning Approaches for Auto Insurance Fraud Detection." Journal of Data Science and Analytics, 5(2), 123-140.
2. Brown, A., & Williams, S. (2018). "Predictive Modeling in Auto Insurance Fraud Detection: A Comparative Study of Algorithms." International Conference on Machine Learning and Applications, 87-95.
3. Li, X., & Wang, Y. (2019). "A Review of Machine Learning Techniques in Auto Insurance Fraud Detection." Journal of Insurance Research, 25(3), 301-318.
4. Kumar, V., & Gupta, S. (2021). "Fraud Detection in Auto Insurance Using Machine Learning: A Case Study." In Proceedings of the International Conference on Artificial Intelligence and Applications, 112-120.
5. Wang, Q., & Liu, W. (2020). "Feature Engineering and Model Selection for Auto Insurance Fraud Detection Based on Machine Learning." Expert Systems with Applications, 142, 112345.
6. 6.Kim, D., & Lee, S. (2021). "Auto Insurance Fraud Detection Using XGBoost and LSTM Neural Networks." Information Sciences, 555, 112233.
7. Zhou, Y., & Li, H. (2019). "A Comparative Study of Supervised Learning Algorithms for Auto Insurance Fraud Detection." IEEE Transactions on Big Data, 5(2), 301

12.APPENDIX

Model building:

1)Dataset
2)Google colab and VS code Application Building
   1. HTML file (Index file, Predict file)
   1. CSS file
   2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
 <html lang="en">
          <head>
             <meta charset="UTF-8">
             <meta http-equiv="X-UA-Compatible" content="IE=edge">
             <meta name="viewport" content="width=device-width, initial-scale=1.0">
             <link rel="stylesheet" href="style.css">
       <title>landing page</title>
       <style>
     strong{
       color: green;
     }
          *{
   padding: 0;
   margin: 0;
}
.akhil{

   background-color:black;
   text-align: center;
   position:sticky ;
   top: 0;
}
.akhil ul{
   display: inline-flex;
   list-style: none;
}
.akhil ul li{
   width: 120px;
   margin: 15px;
   padding: 15px;
}
.akhil ul li a{
   text-decoration: none;
   color: white;
}
         header{
            padding: 100px;
            background-color:lightskyblue;
         }



         footer{
            padding: 100px;
            background-color: black;
            position: sticky;
            bottom: 0;
         }
         .akhi a{
            text-decoration: none;
            color: black;
         }
         footer{
            padding: 40px;
            background-color: black;
         }
```

```
        .btn {
           width: 150px;
           height: 30px;
           border: 20px;
           border-bottom: 4px solid red;
           border-top: 4px solid red;
           border-radius: 20px;
        }
        p{
           color: white;
        }
        h1{
           color: white;
        }
        h2{
           color: white;
        }
        body{
       background-repeat: no-repeat;
       background-size: cover;
        }

     </style>
  </head>
  <body background="C:\Users\Akhil\OneDrive\Desktop\smart govt\WhatsApp
Image 2023-12-06 at 16.47.01_62b03a7c.jpg">

     <div class="akhil">

        <ul><li>
           <p>AUTO INSURANCE CLAIMS.</p></li>
           <li><a href="index.html">Home</a></li>
           <li><a href="about.html">About Us</a></li>
           <br>

           <br> <li><a href="#">Contact Us</a></li>
```

```
            <br><br><br><br><br>

             <li><a href=predict.html></image>Predict</a></li>
          </ul></div>
          <br><br><br><br><br><br>
          <br><br><br><br><br>
       <center>
          <br><br><br>
          <h1>AUTO INSURANCE CLAIMS.</h1>
          <h2>Insurance Fraaud Detection can be prevented by </h2>
             <h2>analyzing the previous Fraud Data and detecting same trends</h2>
          <br><br>
         </center>
         <center>

             <br><br><br><br><br><br>
             <br><br><br><br><br><br>
             <br><br><br><br><br><br>
             <br>
             <footer>
                <br>

             </footer>

   </body>
</html>
```

PREDICT.HTML

```
<html lang="en">

<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="style.css">
   <title>landing page</title>
   <style>
      strong {
         color: green;
```

```
}

* {
   padding: 0;
   margin: 0;
}

.akhil {

   background-color: black;
   text-align: center;
   position: sticky;
   top: 0;
   border-bottom: red;
   border-top: red;
   border-left: red;
   border-right: red;
}

.akhil ul {
   display: inline-flex;
   list-style: none;
}

.akhil ul li {
   width: 120px;
   margin: 15px;
   padding: 15px;
}

.akhil ul li a {
   text-decoration: none;
   color: white;
}

header {
   padding: 50px;
   background-color: lightskyblue;
}



footer {
    padding: 100px;
    background-color: black;
    position: sticky;
    bottom: 0;
}

.akhi a {
    text-decoration: none;
    color: black;
}

footer {
    padding: 10px;
    background-color: green;
}

.btn {
    width: 150px;
    height: 30px;
    border: 20px;
    border-bottom: 4px solid red;
    border-top: 4px solid red;
    border-radius: 20px;
}

p {
    color: white;
}

h1 {
    color: black;
}

h2 {
    color: black;
}

h3 {
    color: black;
    text-align: center;
    justify-content: center;
}
```

```
        .akhilesh {

            display: flex;
            text-align: center;
            justify-content: center;
            justify-items: first baseline;
            align-items: flex-start;
        }

        .class {
            place-items: initial;
            border-top: solid 2px white;
            border-left: solid 2px white;
            border-bottom: solid 2px white;
            border-right: solid 2px white;
        }

        .h {
            color: black;
        }

        .o {

            color: black;
            text-decoration: none;
        }

        .div-2 {
            justify-items: end;
        }
    </style>
</head>

<body>

    <div class="akhil">

        <ul>
            <li>
                <p>AUTO INSURANCE CLAIMS.</p>
            </li>
            <li><a href="/welcome">Home</a></li>
            <li><a href="#about">About Us</a></li>
            <br>

            <br>
            <li><a href="#">Contact Us</a></li>

            <br><br><br><br><br>

            <li><a href="/predict">Predict</a></li>
        </ul>
    </div>
    <br><br>
    <div class="akhilesh">
        <center>
            <div class="class">
                <div clss="div-2">
                    <h1 class="h">Insurance Fraud Detection</h1>
                    <br><br><br>
                        <form action='/predict' method='POST'>
                            <h3>months_as_customer</h3>
                            <input type="number" id="number" required>
                            <h3>Plicy_csl</h3>
                            <input type="number" id="number" required>
                            <h3>insured_sex</h3>
                            <input type="number" id="number" required>
                            <h3>capital-gains</h3>
                            <input type="number" id="number" required>
                            <h3>collision_type</h3>
                            <input type="number" id="number" required>
                            <h3>incident_hour_of_the_day</h3>
                            <input type="number" id="number" required>
                            <h3>witnesses</h3>
                            <input type="number" id="number" required>
                            <h3>auto_model</h3>
                            <input type="number" id="number" required>

                </div>
                <div class="div-3">
                    <br><br><br>
                    <h3>Plicy_number</h3>
                    <input type="number" id="number" required>
                    <h3>Plicy_deductable</h3>
                    <input type="number" id="number" required>
                    <h3>insured_occupation</h3>
        <input type="number" id="number" required>
        <h3>incident_severity</h3>
        <input type="number" id="number" required>
        <h3>number_of_vechiles_involved</h3>
        <input type="number" id="number" required>
        <h3>police_report_available</h3>
        <input type="number" id="number" required>
        <h3>witnesses</h3>
        <input type="number" id="number" required>
        <h3>auto_year</h3>
        <input type="number" id="number" required>


    </div>
    <div class="div-4">
        <br><br><br>
        <h3>Plicy_bind_date</h3>
        <input type="number" id="number" required>
        <h3>Plicy_annual_premium</h3>
        <input type="number" id="number" required>
        <h3>insured_hobbies</h3>
        <input type="number" id="number" required>
        <h3>incident_date</h3>
        <input type="number" id="number" required>
        <h3>authorities_contracted</h3>
        <input type="number" id="number" required>
        <h3>property_damage</h3>
        <input type="number" id="number" required>
        <h3>total_claaim_amount</h3>
        <input type="number" id="number" required>



    </div>
    <div class="div-4">
        <br><br><br>
        <h3>Plicy_state</h3>
        <input type="number" id="number" required>
        <h3>insurred_relationship</h3>
        <input type="number" id="number" required>
        <h3>incident_type</h3>
        <input type="number" id="number" required>
        <h3>incident_location</h3>
```

```
                <input type="number" id="number" required>
                <h3>authorities_contracted</h3>
                <input type="number" id="number" required>
                <h3>bodily_injurles</h3>
                <input type="number" id="number" required>
                <h3>auto_make</h3>
                <input type="number" id="number" required>

            </div>

        </div>

    </div>
    </div>
    <a class="o" href="#"><b>Submit</b></a </form>
    <br>
    <h1>The insurance Fraud Detection Status:</h1>
    <h1>{{predict}}</h1>

</body>

</html>
```

<u>**APP.PY**</u>

```
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>landing page</title>
    <style>
        strong {
            color: green;
        }
```

```
* {
   padding: 0;
   margin: 0;
}

.akhil {

   background-color: black;
   text-align: center;
   position: sticky;
   top: 0;
   border-bottom: red;
   border-top: red;
   border-left: red;
   border-right: red;
}

.akhil ul {
   display: inline-flex;
   list-style: none;
}

.akhil ul li {
   width: 120px;
   margin: 15px;
   padding: 15px;
}

.akhil ul li a {
   text-decoration: none;
   color: white;
}
```

```
header {
   padding: 50px;
   background-color: lightskyblue;
}



footer {
   padding: 100px;
   background-color: black;
   position: sticky;
   bottom: 0;
}

.akhi a {
   text-decoration: none;
   color: black;
}

footer {
   padding: 10px;
   background-color: green;
}

.btn {
   width: 150px;
   height: 30px;
   border: 20px;
   border-bottom: 4px solid red;
   border-top: 4px solid red;
   border-radius: 20px;
```

```
}

p {
   color: white;
}

h1 {
   color: black;
}

h2 {
   color: black;
}

h3 {
   color: black;
   text-align: center;
   justify-content: center;
}

.akhilesh {

   display: flex;
   text-align: center;
   justify-content: center;
   justify-items: first baseline;
   align-items: flex-start;
}

.class {
   place-items: initial;
   border-top: solid 2px white;
```

```
            border-left: solid 2px white;
            border-bottom: solid 2px white;
            border-right: solid 2px white;
        }

        .h {
            color: black;
        }

        .o {

            color: black;
            text-decoration: none;
        }

        .div-2 {
            justify-items: end;
        }
    </style>
</head>

<body>

    <div class="akhil">

        <ul>
            <li>
                <p>AUTO INSURANCE CLAIMS.</p>
            </li>
            <li><a href="/welcome">Home</a></li>
            <li><a href="#about">About Us</a></li>
            <br>
            <br>
            <li><a href="#">Contact Us</a></li>


            <br><br><br><br><br>


            <li><a href="/predict">Predict</a></li>
        </ul>
    </div>
    <br><br>
    <div class="akhilesh">
        <center>
            <div class="class">
                <div clss="div-2">
                    <h1 class="h">Insurance Fraud Detection</h1>
                    <br><br><br>
                        <form action='/predict' method='POST'>
                            <h3>months_as_customer</h3>
                            <input type="number" id="number" required>
                            <h3>Plicy_csl</h3>
                            <input type="number" id="number" required>
                            <h3>insured_sex</h3>
                            <input type="number" id="number" required>
                            <h3>capital-gains</h3>
                            <input type="number" id="number" required>
                            <h3>collision_type</h3>
                            <input type="number" id="number" required>
                            <h3>incident_hour_of_the_day</h3>
                            <input type="number" id="number" required>
                            <h3>witnesses</h3>
                            <input type="number" id="number" required>
                            <h3>auto_model</h3>
                            <input type="number" id="number" required>


                </div>
                <div class="div-3">
                    <br><br><br>
                    <h3>Plicy_number</h3>
                    <input type="number" id="number" required>
                    <h3>Plicy_deductable</h3>
                    <input type="number" id="number" required>
                    <h3>insured_occupation</h3>
                    <input type="number" id="number" required>
                    <h3>incident_severity</h3>
                    <input type="number" id="number" required>
                    <h3>number_of_vechiles_involved</h3>
                    <input type="number" id="number" required>
                    <h3>police_report_available</h3>
                    <input type="number" id="number" required>
                    <h3>witnesses</h3>
                    <input type="number" id="number" required>
                    <h3>auto_year</h3>
                    <input type="number" id="number" required>



                </div>
                <div class="div-4">
                    <br><br><br>
                    <h3>Plicy_bind_date</h3>
                    <input type="number" id="number" required>
                    <h3>Plicy_annual_premium</h3>
                    <input type="number" id="number" required>
                    <h3>insured_hobbies</h3>
                    <input type="number" id="number" required>
                    <h3>incident_date</h3>
                    <input type="number" id="number" required>
                    <h3>authorities_contracted</h3>
                    <input type="number" id="number" required>
                    <h3>property_damage</h3>
                    <input type="number" id="number" required>
                    <h3>total_claaim_amount</h3>
                    <input type="number" id="number" required>



                </div>
                <div class="div-4">
                    <br><br><br>
                    <h3>Plicy_state</h3>
                    <input type="number" id="number" required>
                    <h3>insurred_relationship</h3>
                    <input type="number" id="number" required>
                    <h3>incident_type</h3>
                    <input type="number" id="number" required>
                    <h3>incident_location</h3>
                    <input type="number" id="number" required>
                    <h3>authorities_contracted</h3>
                    <input type="number" id="number" required>
                    <h3>bodily_injurles</h3>
                    <input type="number" id="number" required>
                    <h3>auto_make</h3>
                    <input type="number" id="number" required>

                </div>

</div>
```

```
    </div>
    </div>
    <a class="o" href="#"><b>Submit</b></a </form>
    <br>
    <h1>The insurance Fraud Detection Status:</h1>
    <h1>{{predict}}</h1>

</body>

</html>
```

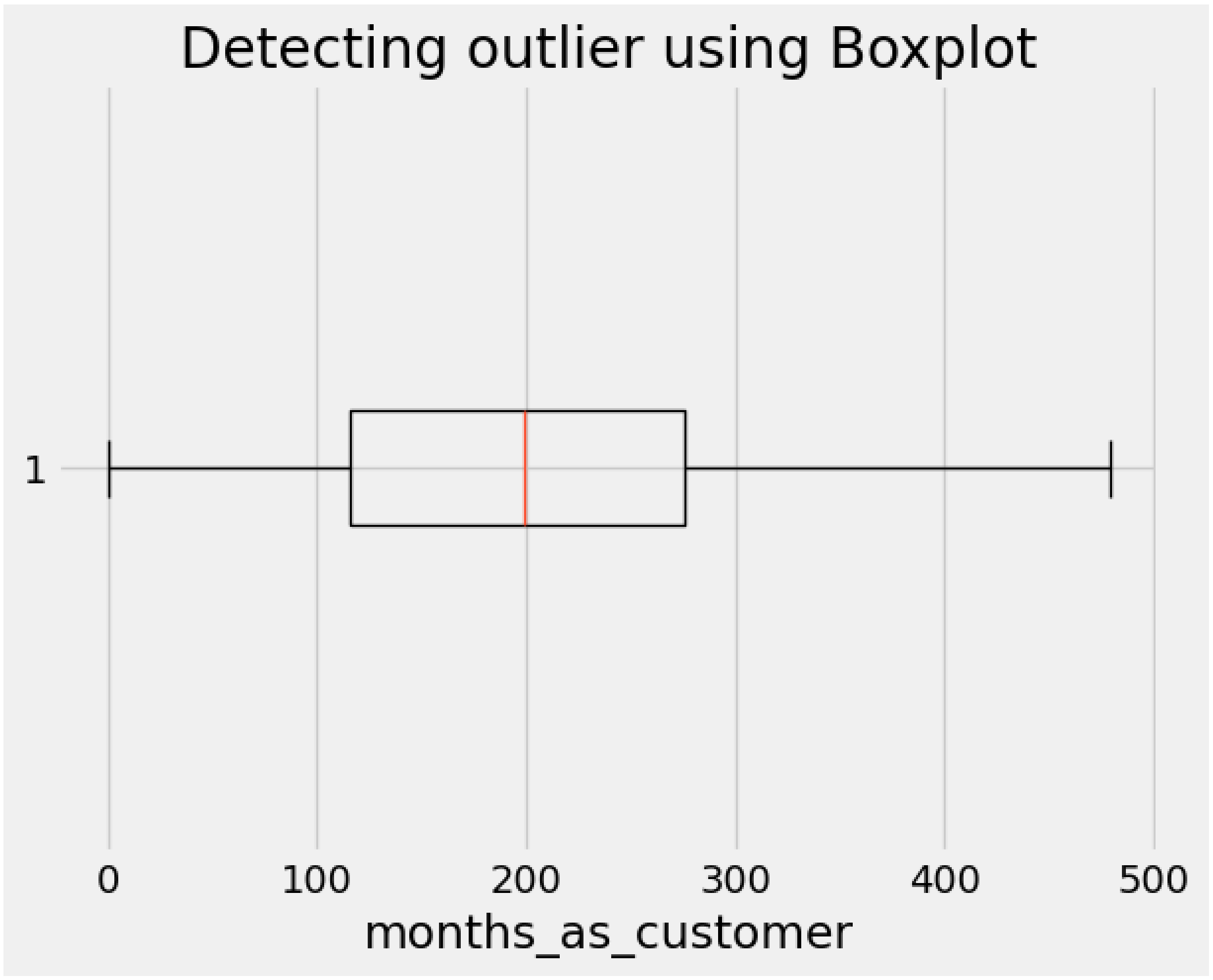
CODE SNIPPETS

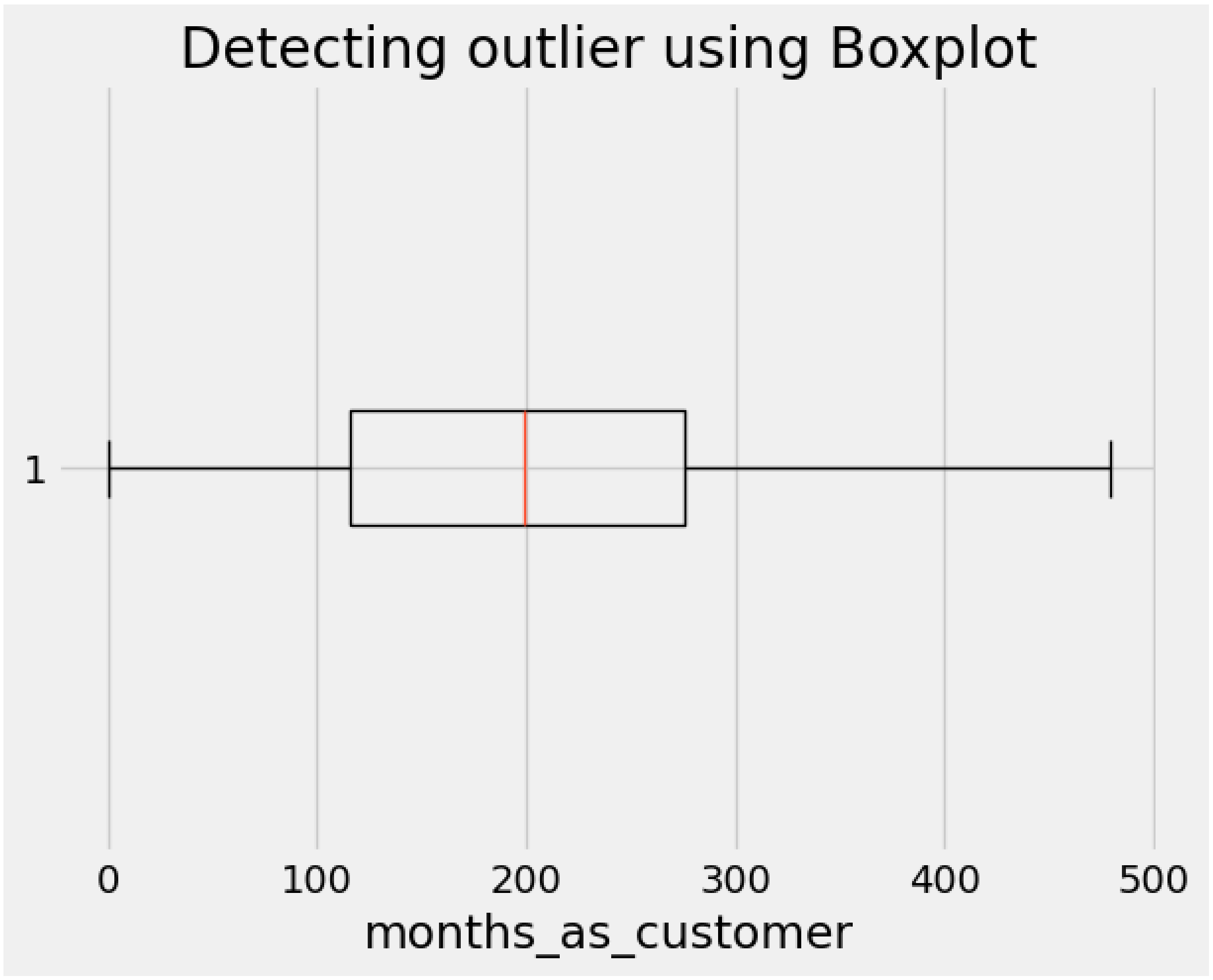
MODEL BUILDING

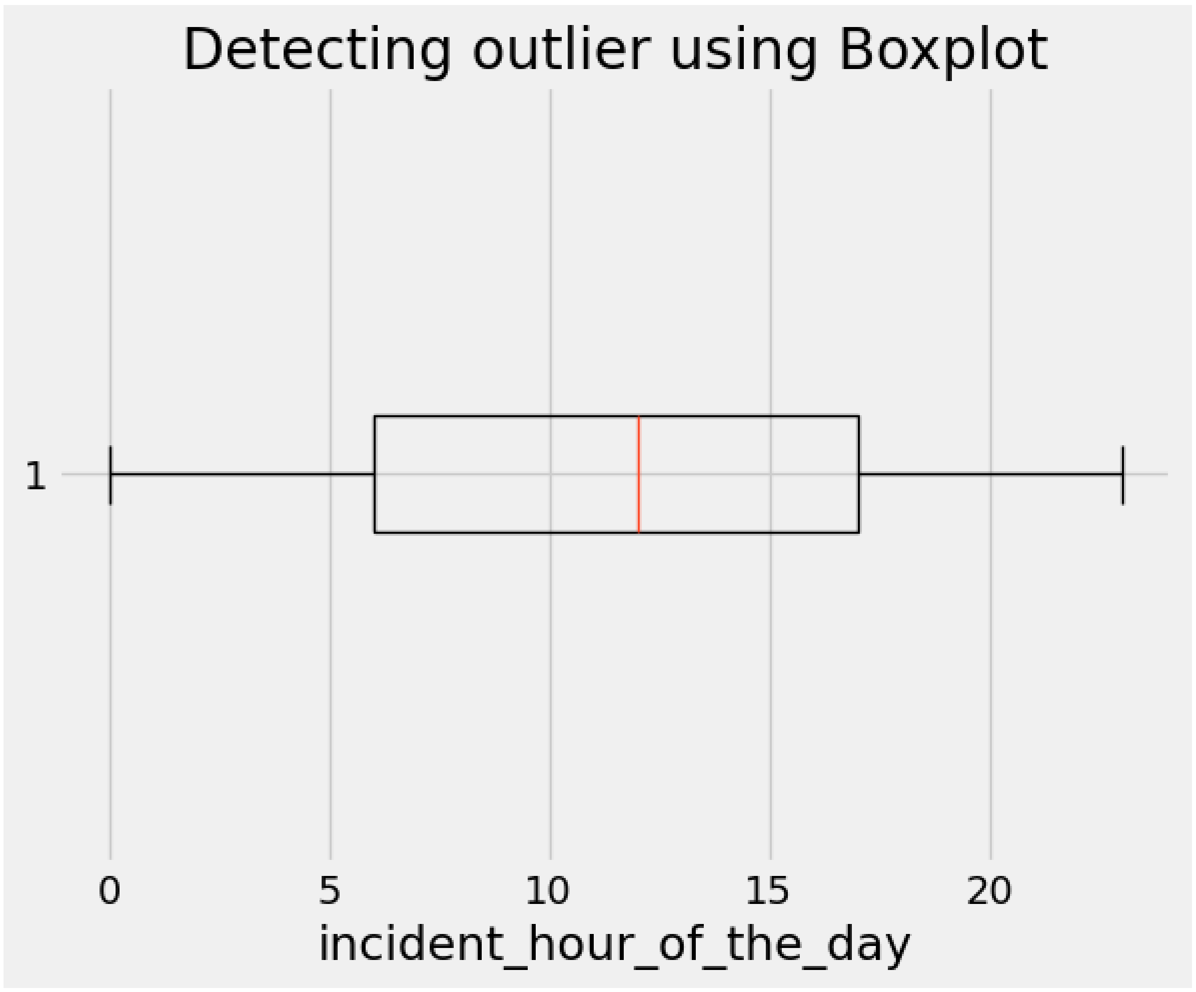
```
from flask import Flask,render_template,request
import numpy as np
import pickle
model=pickle.load(open("dtc_model.pkl",'rb'))
app=Flask(__name__)

@app.route('/')
def welcome():
return render_template('index.html')

@app.route('/predict',methods=['GET','POST'])
def predict():
print(request.method)
print(request.form.values())
months_as_customer=float(request.form["months_as_costumer"])
policy_number=float(request.form['policy_number'])
policy_bind_data=float(request.form['policy_bind_data'])
policy_state=float(request.form['policy_state'])
```

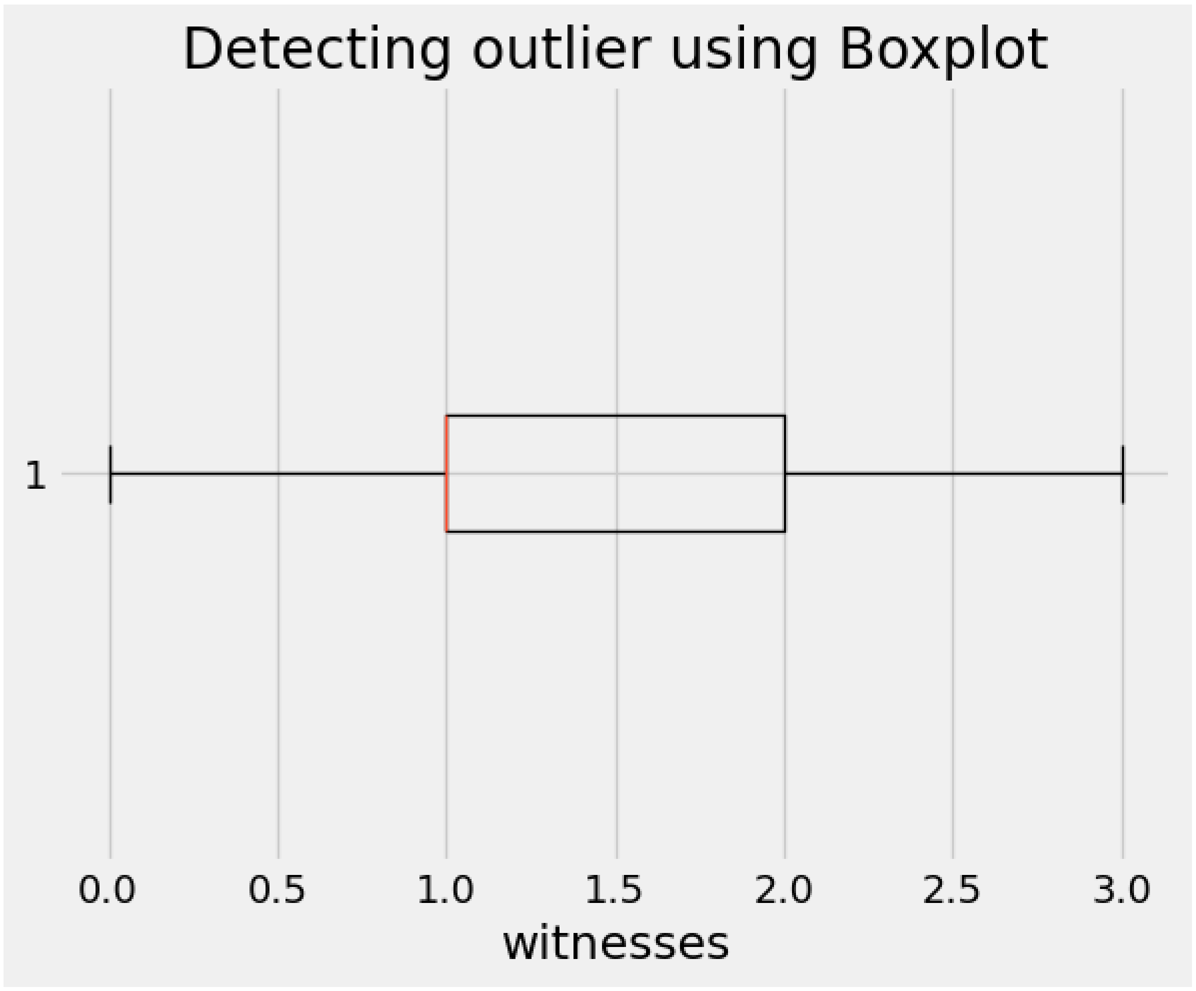
```
policy_csl=float(request.form['policy_csl'])
policy_deductable=float(request.form['policy_deductable'])
policy_annual_premium=float(request.form['policy_annual_premium'])
insured_zip=float(request.form['insured_zip'])
insured_sex = float(request.form['insured_sex'])
insured_occupation = float(request.form['insured_occupation'])
insured_hobbies = float(request.form['insured_hobbies'])
insured_relationship = float(request.form['insured_relationship'])
capital_gains = float(request.form['capital_gains'])
capital_loss=float(request.form['capital_loss'])
incident_date=float(request.form['incident_date'])
incident_type=float(request.form['incedent_type'])
collision_type=float(request.form['collision_type'])
incident_severity=float(request.form['incident_severity'])
authorities_contacted=float(request.form['authorities_contacted'])
incident_location=float(request.form['incident_location'])
incident_hour_of_the_day=float(request.form['incident_hour_of_the_day'])
number_of_vehicles_involved=float(request.form['number_of_vehicles_involved'])
property_damage=float(request.form['property_damage'])
bodily_injuries=float(request.form['bodily_injuries'])
witnesses=float(request.form['witnesses'])
police_report_available=float(request.form['police_report_avilable'])
total_claim_amount=float(request.form['total_claim_amount'])
auto_make=float(request.form['auto_make'])
auto_model=float(request.form['auto_model'])
auto_year=float(request.form['auto_year'])

total=[[months_as_customer,policy_number,policy_bind_data,policy_state,policy_csl
,policy_deductable,policy_annual_premium,insured_zip,insured_sex,insured_occupa
tion,insured_hobbies,insured_relationship,capital_gains,capital_loss,incident_date,in
cident_type,collision_type,incident_severity,authorities_contacted,incident_location,i
ncident_hour_of_the_day,number_of_vehicles_involved,property_damage,bodily_inj
uries,witnesses,police_report_available,total_claim_amount,auto_make,auto_model,
auto_year]]
prediction = model.predict((total))
prediction=int(prediction[0])
if prediction==0:
```

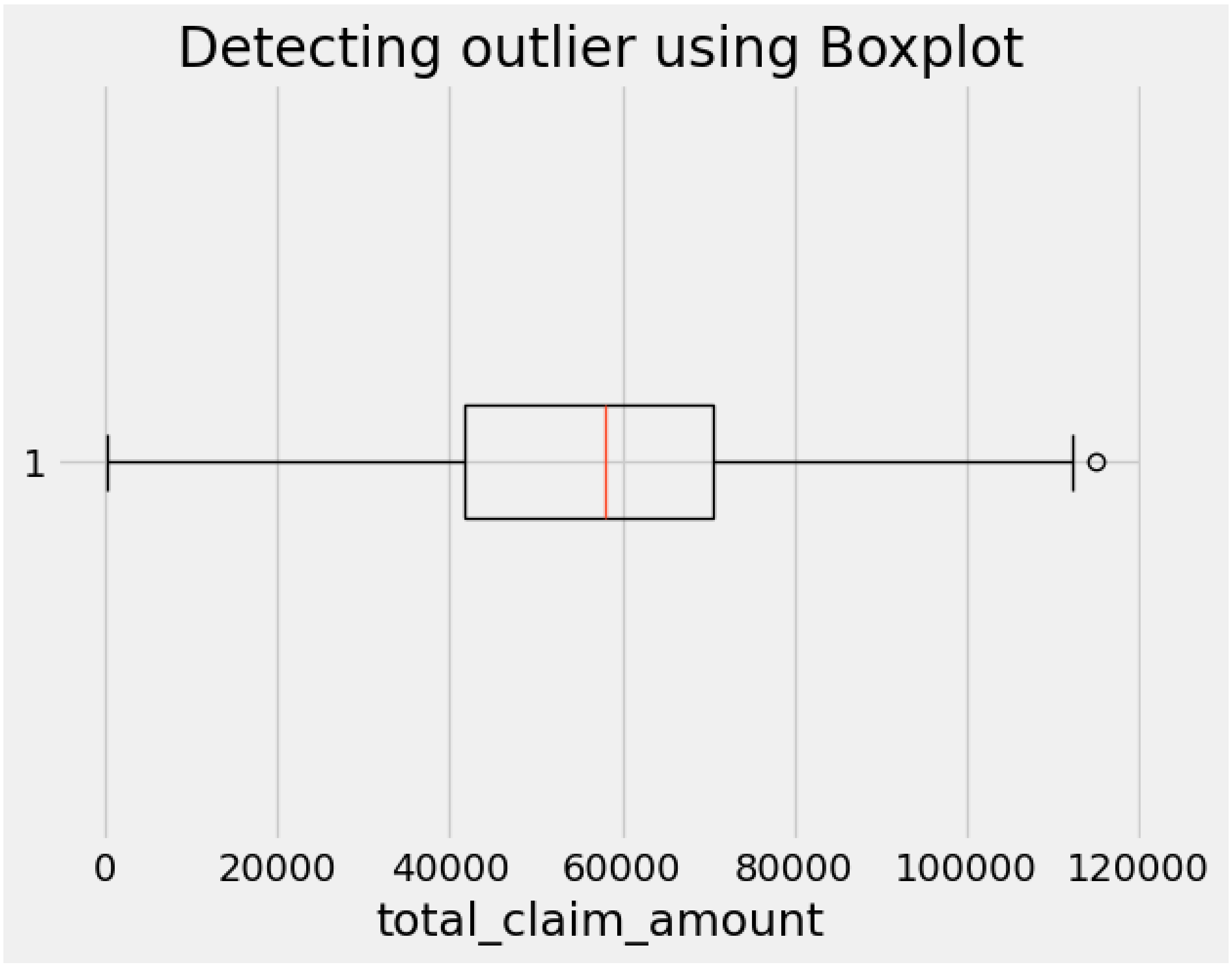
```
return render_template('index.html',predict="Legal insurance Claim")
else:
return render_template('index.html',predict='Fraud Insurance Claim')
if __name__== '__main__':
app.run(port=5000)
```


Detecting outlier using Boxplot
1
0
100
200
300
400
500
months_as_customer

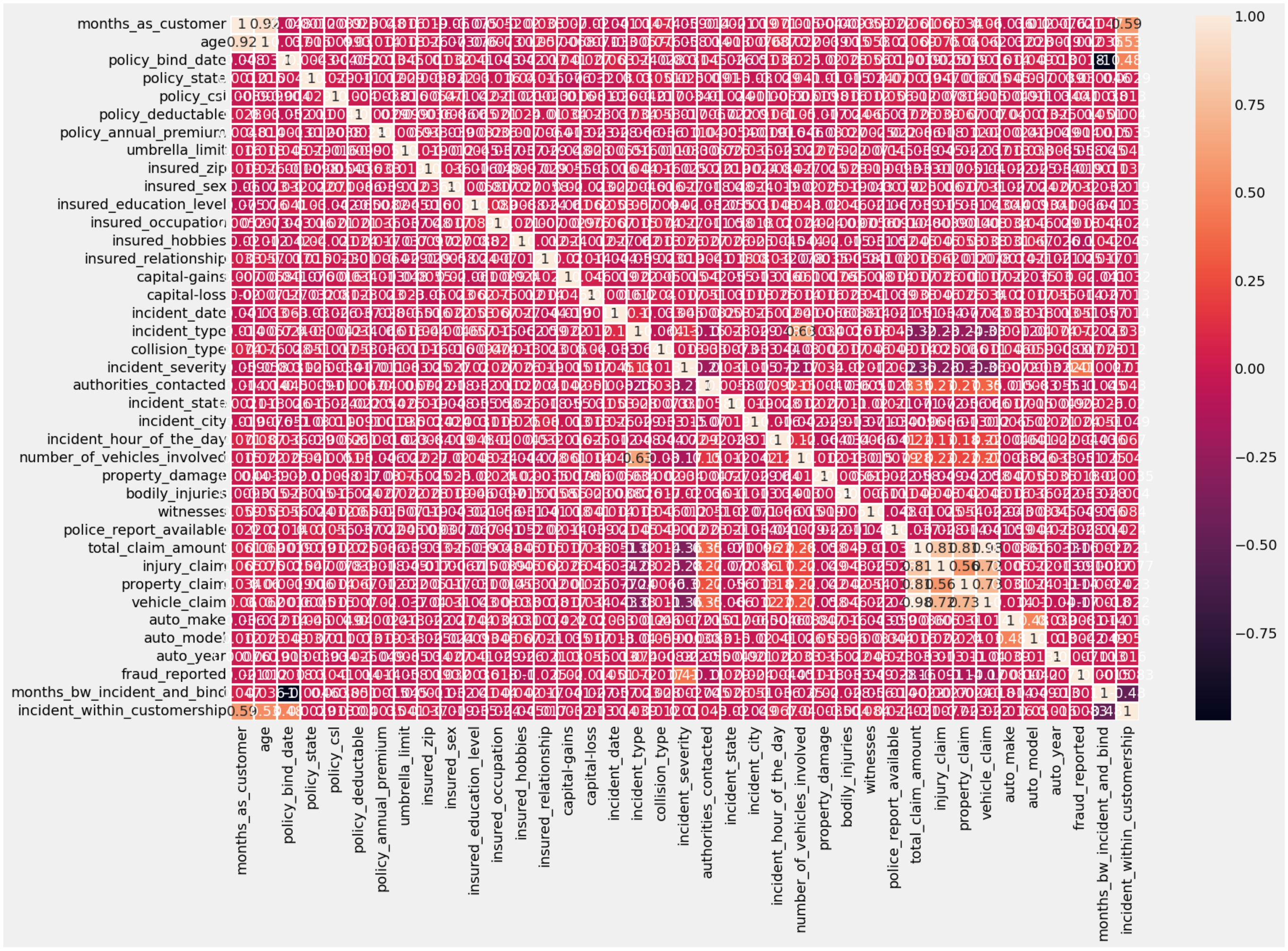

Detecting outlier using Boxplot
1
0
100
200
300
400
500
months_as_customer


Detecting outlier using Boxplot
1
0
5
10
15
20
incident_hour_of_the_day


Detecting outlier using Boxplot
1
0.0
0.5
1.0
1.5
2.0
2.5
3.0
witnesses


Detecting outlier using Boxplot
1
0
20000
40000
60000
80000
100000
120000
total_claim_amount

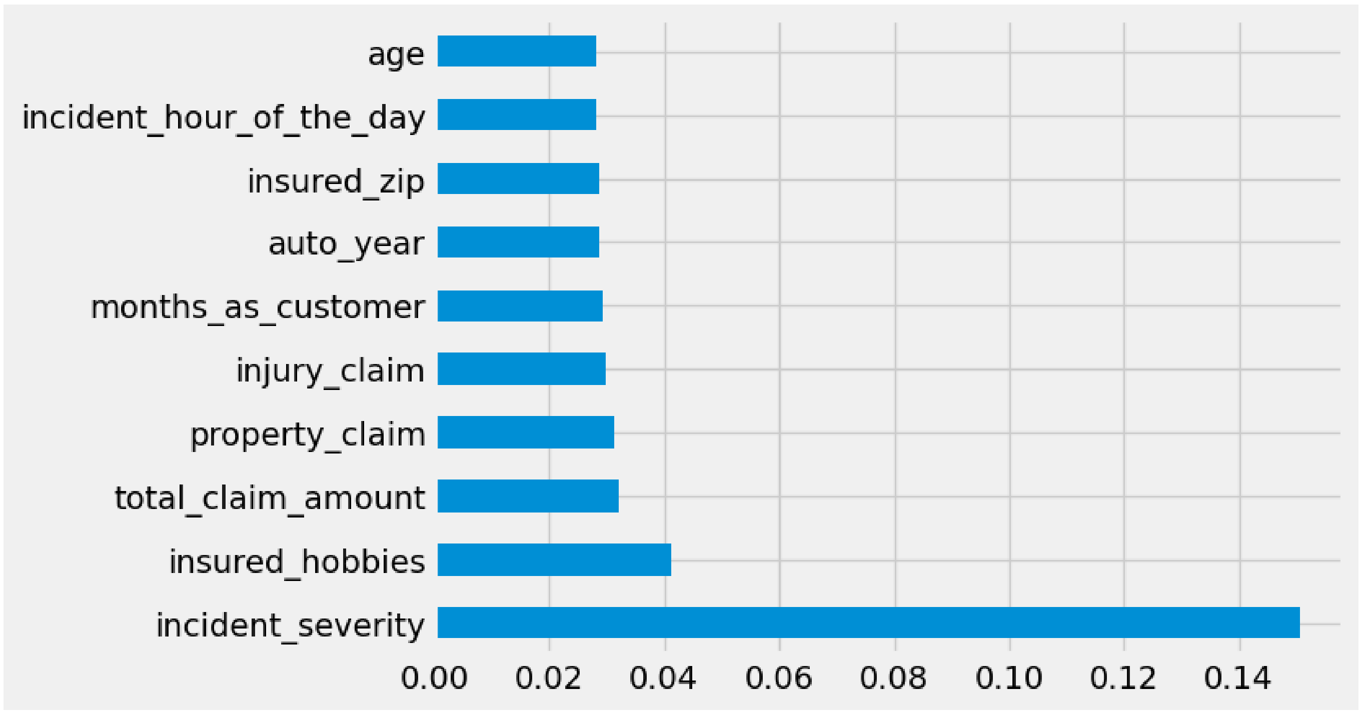
HEAT MAP



```
 df = df.drop('vehicle_claim', axis = 1) #as its correlation with another feature
(total_claim_amount) > 95%. Keeping multiple strongly correlated featured would make
the collection of such features influential.

df = df.drop(['policy_bind_date', 'incident_date'], axis = 1) #as information from
these features has been exported to new derived features
from sklearn.ensemble import ExtraTreesClassifier
y = df["fraud_reported"]
x = df.drop("fraud_reported", axis = 1)
model = ExtraTreesClassifier()
model.fit(x,y)
print(model.feature_importances_)
```

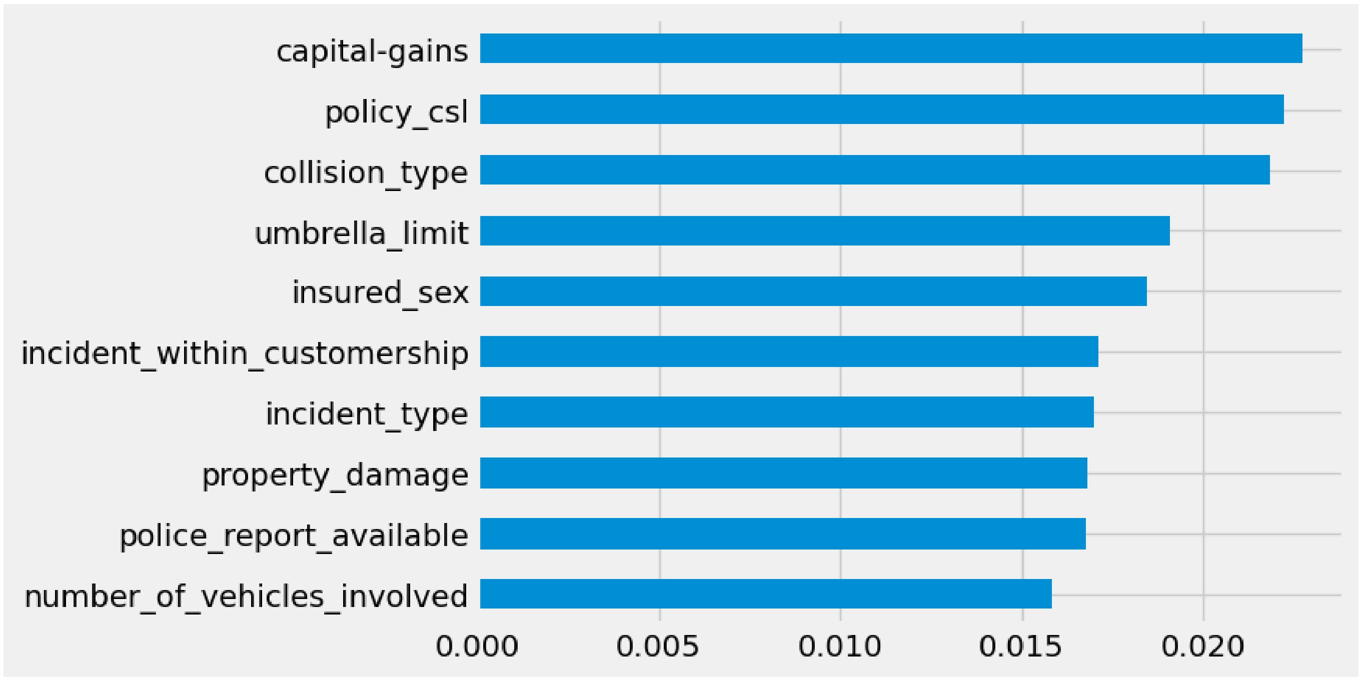
```
[0.02920422 0.02818448 0.02278969 0.02221471 0.02332096 0.02700853
 0.01907885 0.02873389 0.01845558 0.02488251 0.02812293 0.04131012
 0.02410153 0.02275305 0.02600355 0.01699277 0.02185829 0.15062225
 0.02741337 0.02818387 0.02557201 0.02823742 0.01581882 0.01681227
 0.02334369 0.02462078 0.01678774 0.0321875  0.02982406 0.03138174
 0.02683158 0.02576082 0.02883948 0.02561516 0.01713177]
```

```
#Finding 10 most important features
feature_importances = pd.Series(model.feature_importances_, index = x.columns)
feature_importances.nlargest(10).plot(kind = 'barh')
plt.show()
```



```
#Finding 10 least important features
feature_importances.nsmallest(10).plot(kind = 'barh')
plt.show()
```

```
for i in dict(feature_importances.nlargest(10)):
  if i in mappings:
    chart = pd.crosstab(index = df[i], columns = df['fraud_reported'])
    print('fraud_reported' + ':', mappings['fraud_reported'])
    print(i + ':',mappings[i])
    print(chart)
    print()
    chart.plot.bar()
```



```
#Not selecting columns with importance < 0.02
df = df.drop(['collision_type','property_damage', 'incident_within_customership',
'insured_sex', 'umbrella_limit', 'number_of_vehicles_involved',
'police_report_available', 'incident_type'], axis = 1)
```

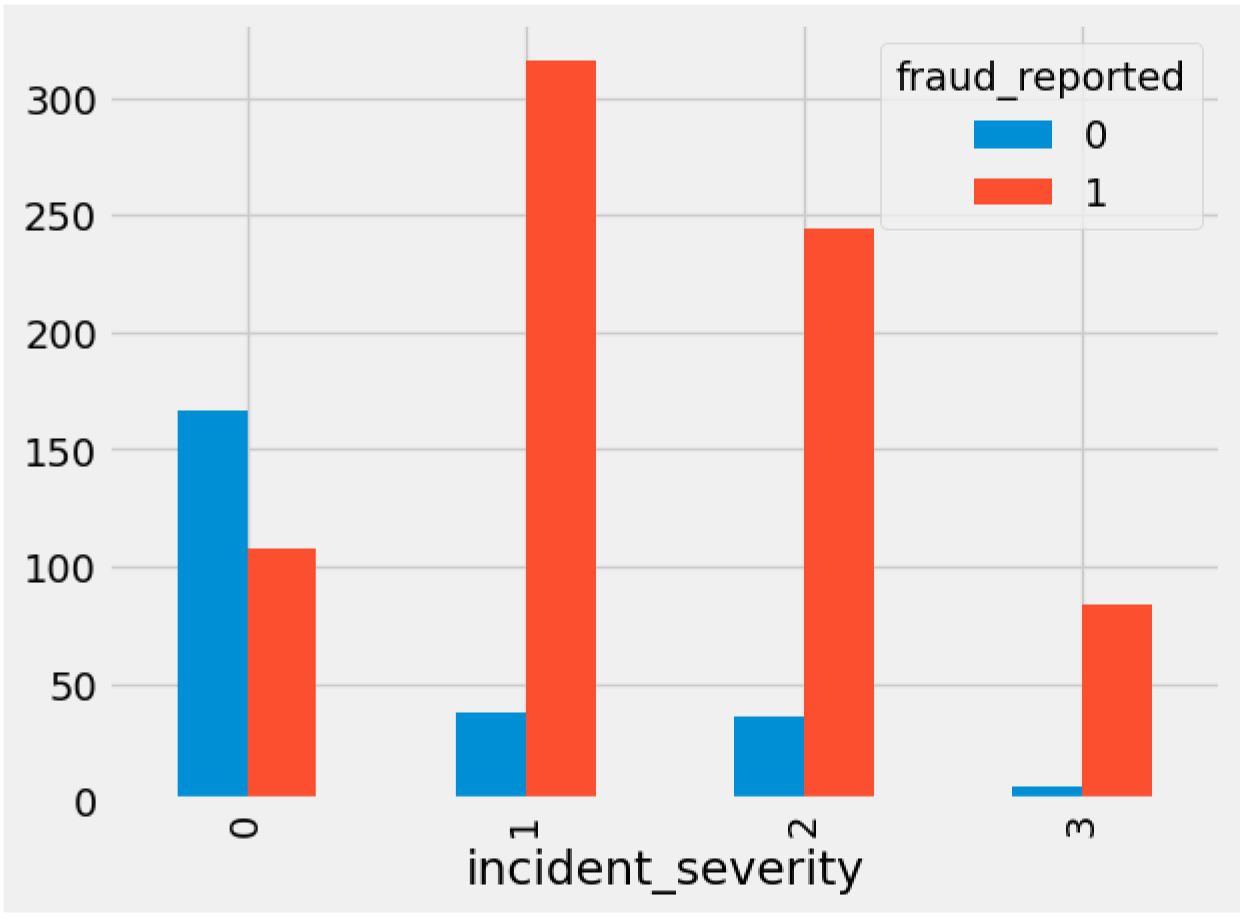
```
# Data Analysis and Visualization
print(mappings['fraud_reported'])
print("YES: ",y[y == 0].shape[0])
print("NO:",y[y ==1].shape[0])
```

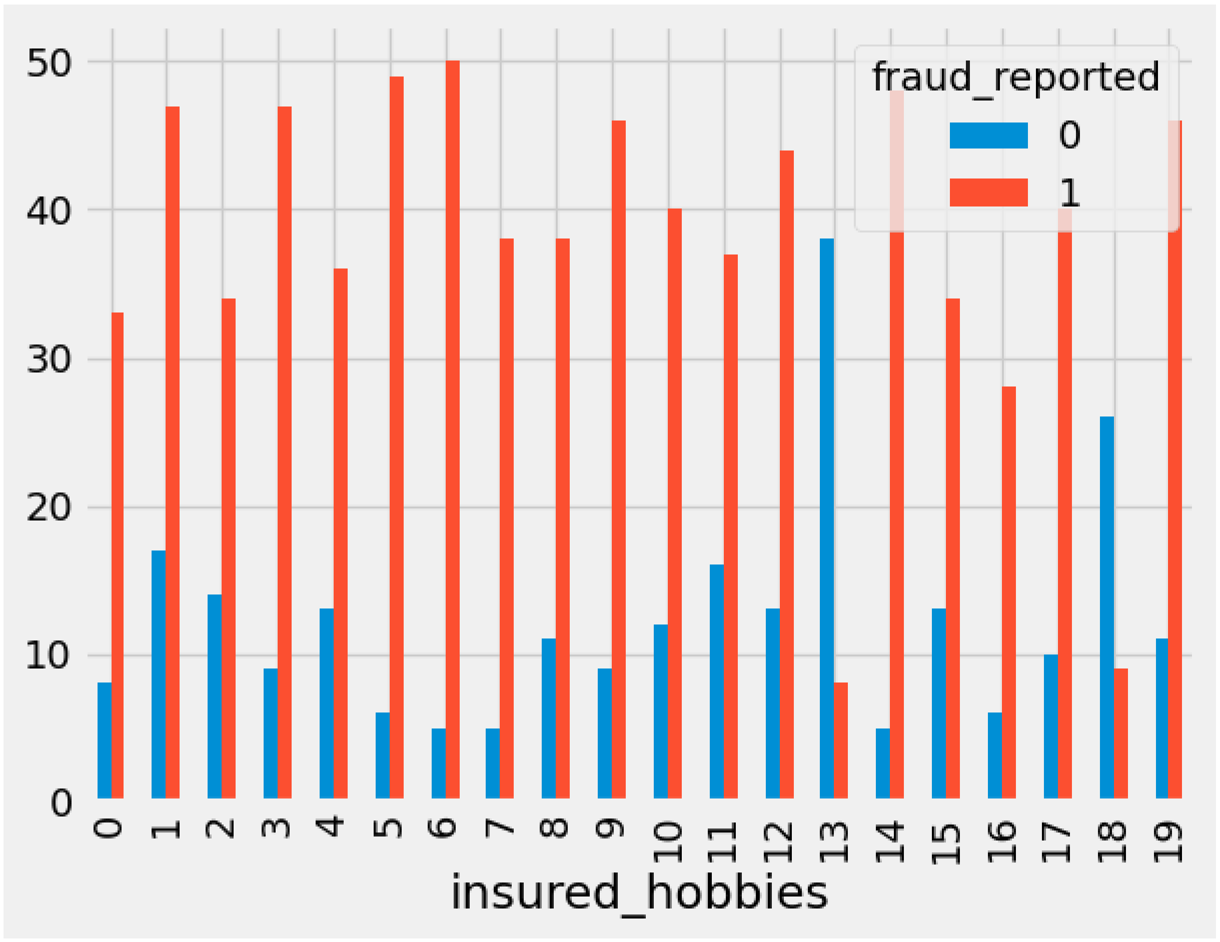
```
{'Y': 0, 'N': 1}
YES:  247
NO: 752
```

```
for i in dict(feature_importances.nlargest(10)):
  if i in mappings:
    chart = pd.crosstab(index = df[i], columns = df['fraud_reported'])
    print('fraud_reported' + ':', mappings['fraud_reported'])
    print(i + ':',mappings[i])
    print(chart)
    print()
    chart.plot.bar()
```

```
fraud_reported: {'Y': 0, 'N': 1}
incident_severity: {'Major Damage': 0, 'Minor Damage': 1, 'Total Loss': 2, 'Trivial
Damage': 3}
fraud_reported       0    1
incident_severity
0                  167  108
1                   38  316
2                   36  244
3                    6   84

fraud_reported: {'Y': 0, 'N': 1}
insured_hobbies: {'sleeping': 0, 'reading': 1, 'board-games': 2, 'bungie-jumping': 3,
'base-jumping': 4, 'golf': 5, 'camping': 6, 'dancing': 7, 'skydiving': 8, 'movies': 9,
'hiking': 10, 'yachting': 11, 'paintball': 12, 'chess': 13, 'kayaking': 14, 'polo': 15,
'basketball': 16, 'video-games': 17, 'cross-fit': 18, 'exercise': 19}
fraud_reported    0   1
insured_hobbies
0                 8  33
1                17  47
2                14  34
3                 9  47
4                13  36
5                 6  49
6                 5  50
7                 5  38
8                11  38
9                 9  46
10               12  40
11               16  37
...
17               10  40
18               26   9
19               11  46
```

Output is truncated. View as a [scrollable element]() or open in a [text editor](). Adjust cell output [settings]()...

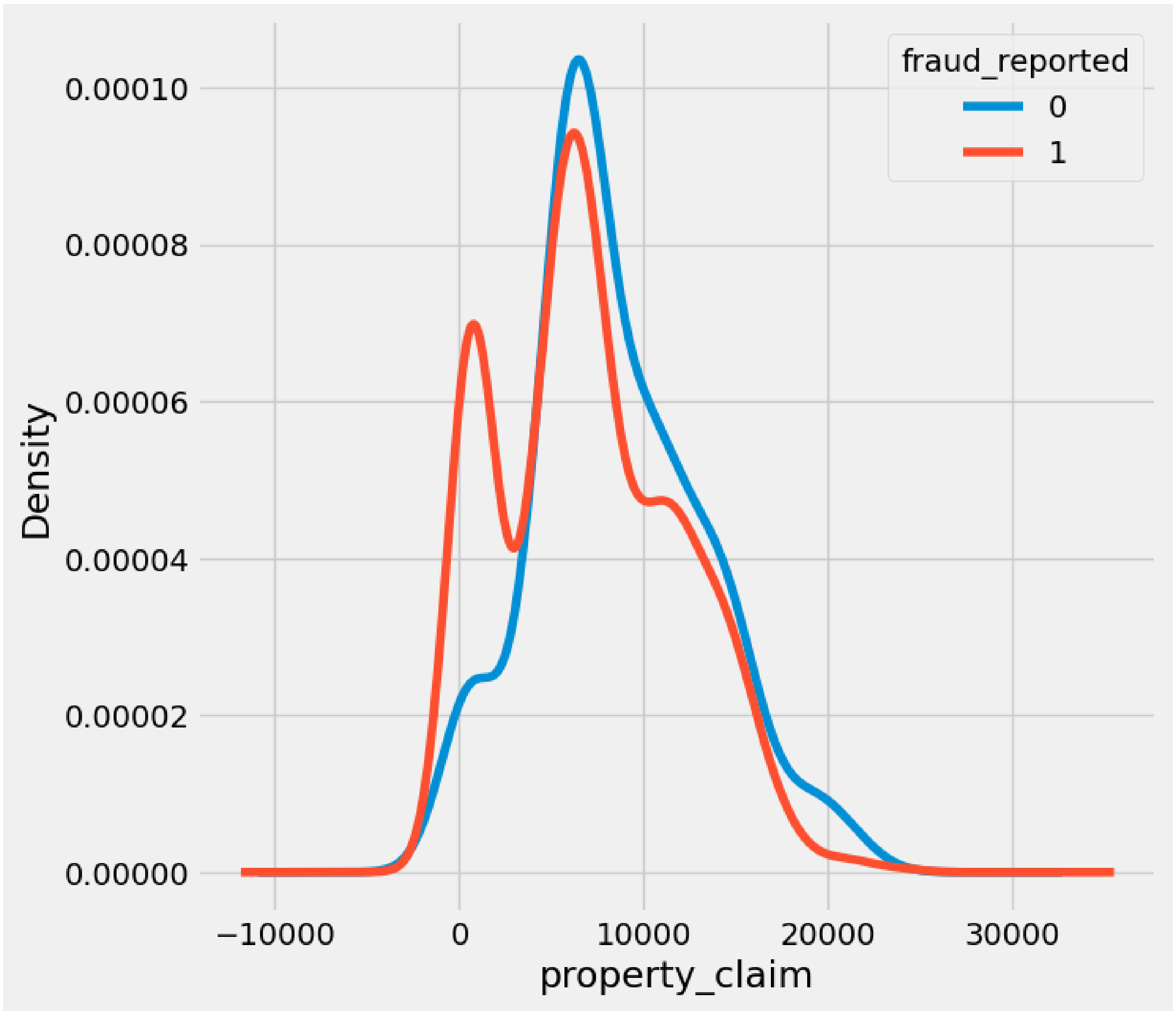

fraud_reported
0
1
300
250
200
150
100
50
0
0
1
2
3
incident_severity

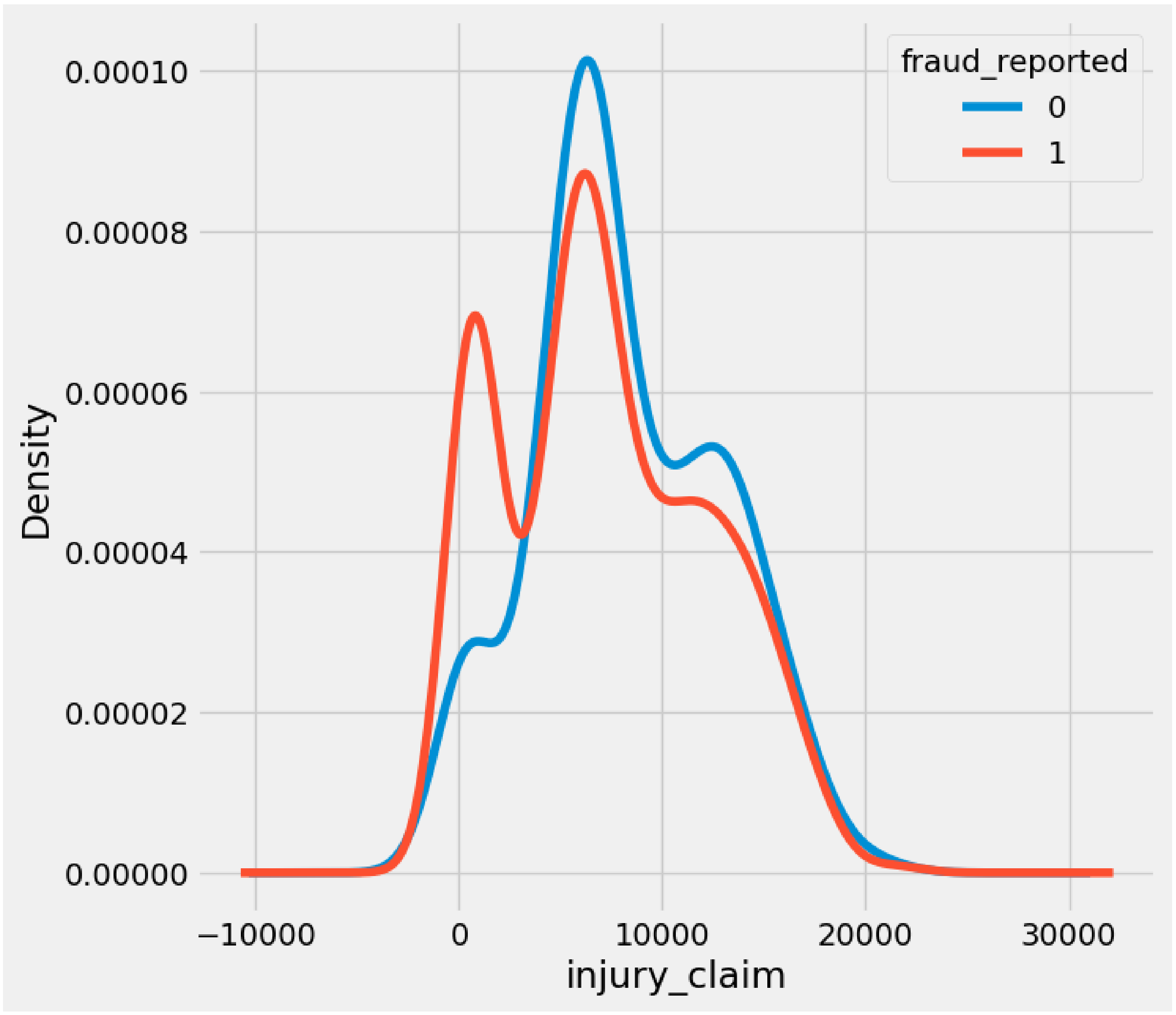


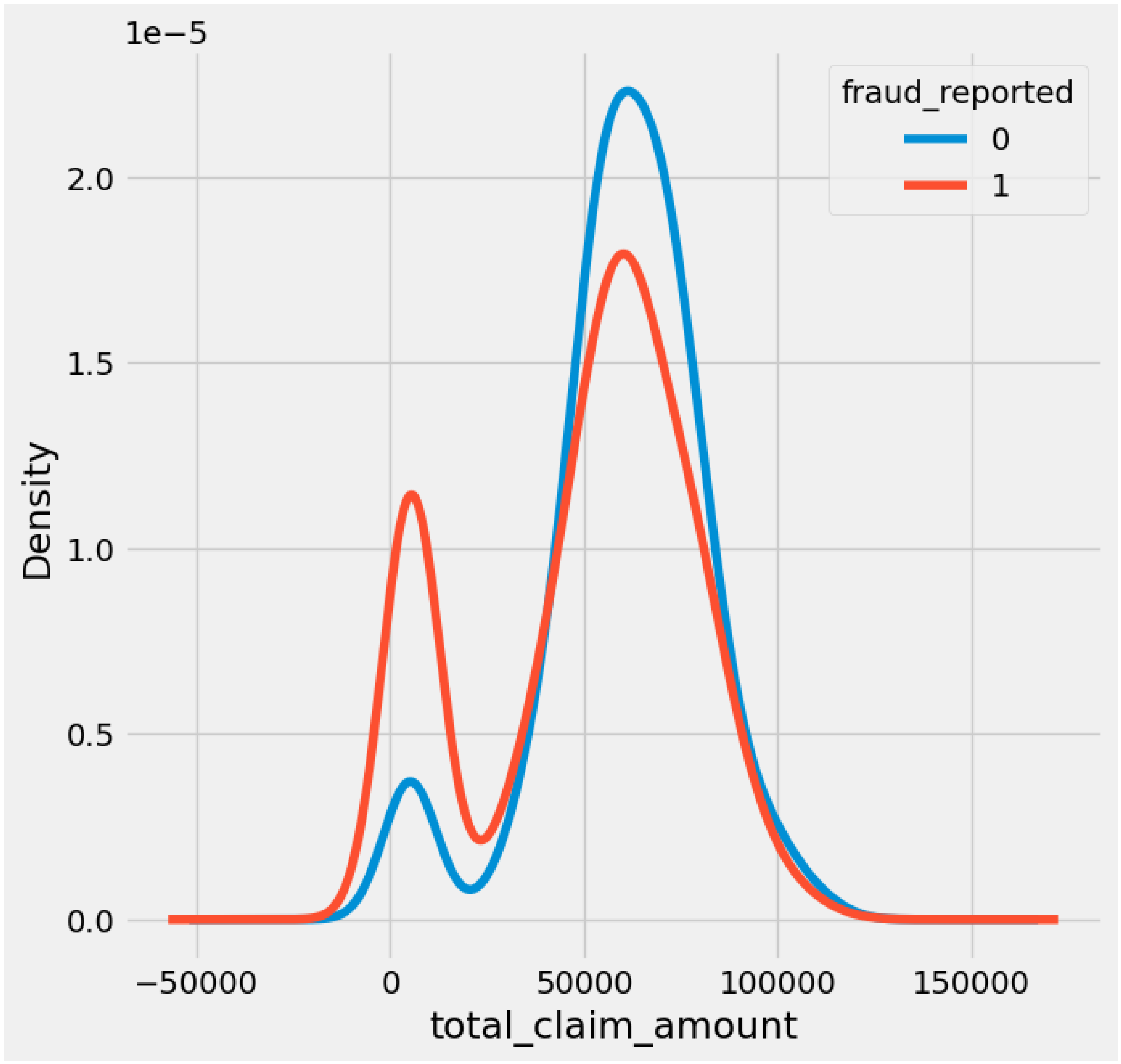
```
df.columns
```

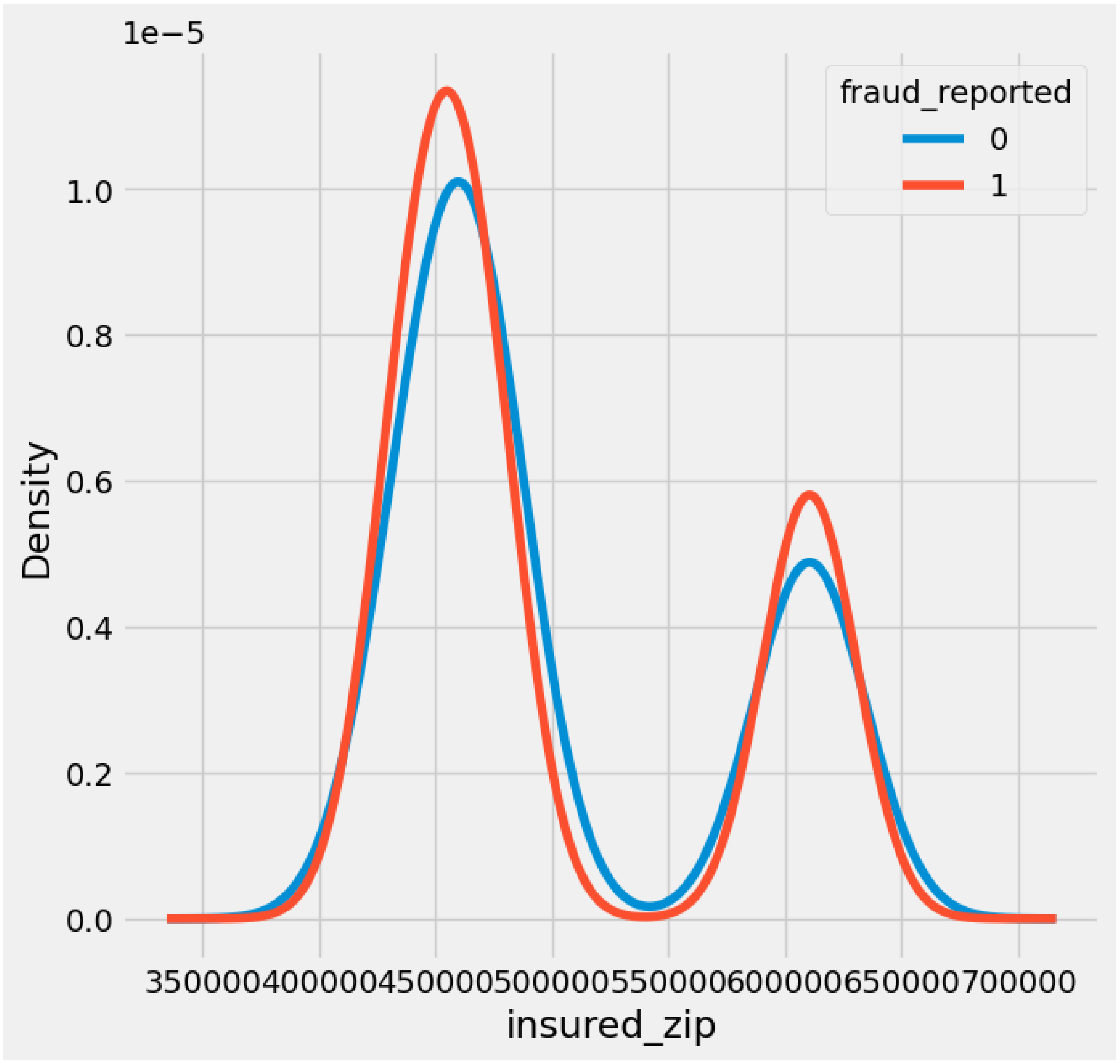
```
Index(['months_as_customer', 'age', 'policy_state', 'policy_csl',
       'policy_deductable', 'policy_annual_premium', 'insured_zip',
       'insured_education_level', 'insured_occupation', 'insured_hobbies',
       'insured_relationship', 'capital-gains', 'capital-loss',
       'incident_severity', 'authorities_contacted', 'incident_state',
       'incident_city', 'incident_hour_of_the_day', 'bodily_injuries',
       'witnesses', 'total_claim_amount', 'injury_claim', 'property_claim',
       'auto_make', 'auto_model', 'auto_year', 'fraud_reported',
       'months_bw_incident_and_bind'],
      dtype='object')
```

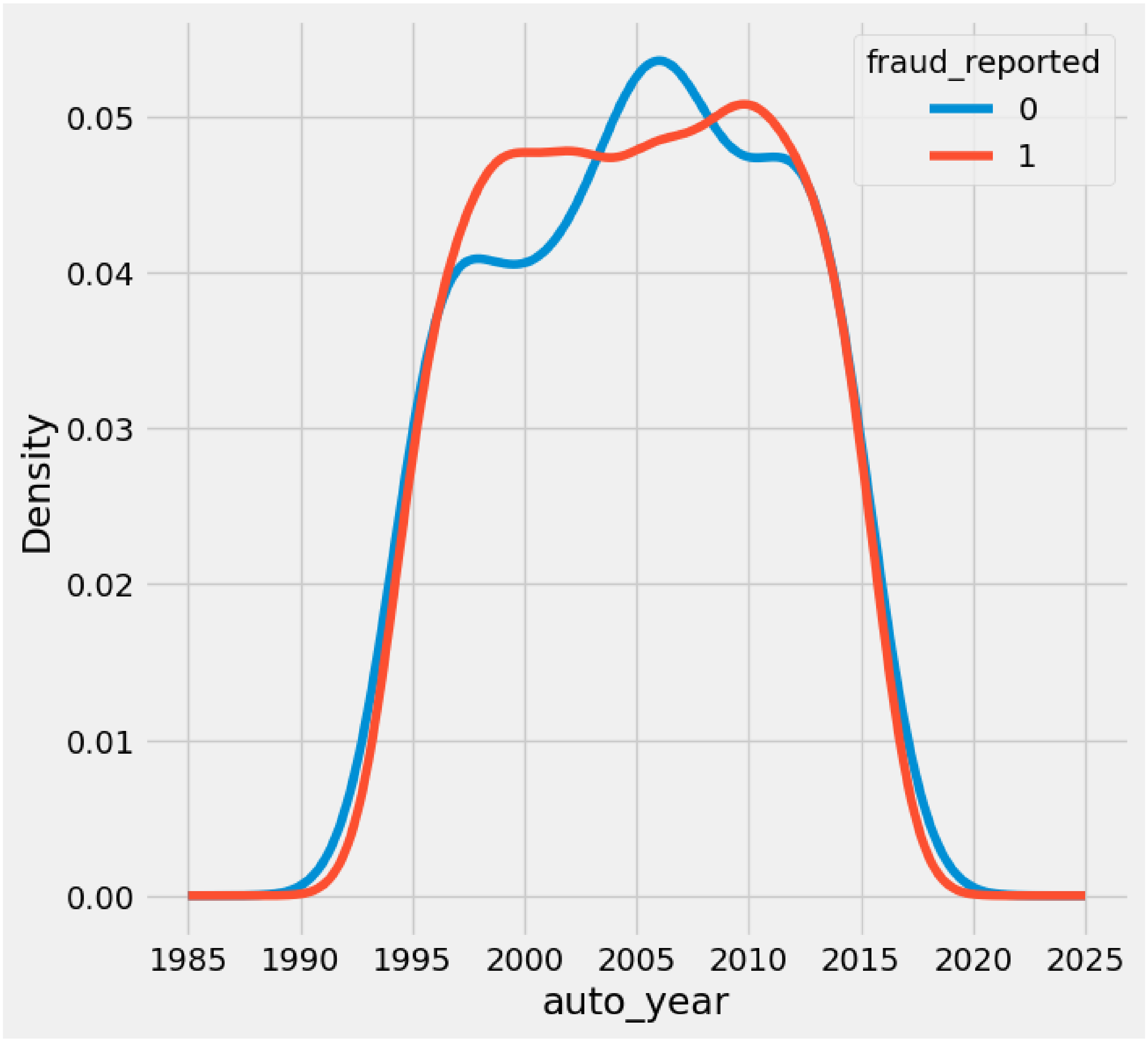
```
# For continuous columns
for i in dict(feature_importances.nlargest(10)):
  if i not in mappings:
    df_specific_column = df.pivot(columns ='fraud_reported', values = i)
    df_specific_column.plot.density(figsize = (7, 7), linewidth = 4)
    plt.xlabel(i)
```


fraud_reported
0
1
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
Density
−10000
0
10000
20000
30000
property_claim


fraud_reported
0
1
Density
0.00010
0.00008
0.00006
0.00004
0.00002
0.00000
−10000
0
10000
20000
30000
injury_claim


1e−5
fraud_reported
0
1
2.0
1.5
1.0
0.5
0.0
Density
−50000
0
50000
100000
150000
total_claim_amount


1e−5
fraud_reported
0
1
1.0
0.8
0.6
0.4
0.2
0.0
Density
350000
400000
450000
500000
550000
600000
650000
700000
insured_zip



```
# Explanatory Model Building
from sklearn.model_selection import train_test_split
X_Train, X_Test, Y_Train, Y_Test = train_test_split(x, y, test_size = 0.2, random_state
= 42)
```

```
len(X_Train)
```

```
799
```

```
len(Y_Train)
```

```
799

len(X_Test)
```

```
200
```

```
len(Y_Test)
```

```
200
```

```
# Explore training all ML models
# Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
model = LogisticRegression()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)

print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.75

 classification report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        48
           1       0.76      0.99      0.86       152

    accuracy                           0.75       200
   macro avg       0.38      0.49      0.43       200
weighted avg       0.58      0.75      0.65       200


 confusion matrix:
 [[  0  48]
 [  2 150]]
```

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)
```

```
print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.71

 classification report:
               precision    recall  f1-score   support

           0       0.29      0.15      0.19        48
           1       0.77      0.89      0.82       152

    accuracy                           0.71       200
   macro avg       0.53      0.52      0.51       200
weighted avg       0.65      0.71      0.67       200


 confusion matrix:
 [[  7  41]
 [ 17 135]]
```

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)

print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.71

 classification report:
               precision    recall  f1-score   support

           0       0.29      0.15      0.19        48
           1       0.77      0.89      0.82       152

    accuracy                           0.71       200
   macro avg       0.53      0.52      0.51       200
weighted avg       0.65      0.71      0.67       200


 confusion matrix:
 [[  7  41]
 [ 17 135]]
```

```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)

print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.77

 classification report:
               precision    recall  f1-score   support

           0       0.52      0.62      0.57        48
           1       0.87      0.82      0.84       152

    accuracy                           0.77       200
   macro avg       0.70      0.72      0.70       200
weighted avg       0.79      0.77      0.78       200


 confusion matrix:
 [[ 30  18]
 [ 28 124]]
```

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)

print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.735

 classification report:
```

```
              precision    recall  f1-score   support

           0       0.41      0.25      0.31        48
           1       0.79      0.89      0.84       152

    accuracy                           0.73       200
   macro avg       0.60      0.57      0.57       200
weighted avg       0.70      0.73      0.71       200


 confusion matrix:
 [[ 12  36]
 [ 17 135]]
```

```python
from xgboost import XGBClassifier

model = XGBClassifier()
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)

print('Accuracy:', accuracy_score(Y_Test, pred))
print('\n classification report:\n', classification_report(Y_Test, pred))
print('\n confusion matrix:\n', confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.815

 classification report:
               precision    recall  f1-score   support

           0       0.60      0.67      0.63        48
           1       0.89      0.86      0.88       152

    accuracy                           0.81       200
   macro avg       0.75      0.76      0.75       200
weighted avg       0.82      0.81      0.82       200


 confusion matrix:
 [[ 32  16]
 [ 21 131]]
```

```python
from xgboost import XGBClassifier

model = XGBClassifier(colsample_bytree= 1, learning_rate = 0.05, max_depth = 10,
n_estimators = 100, subsample = 1)
```

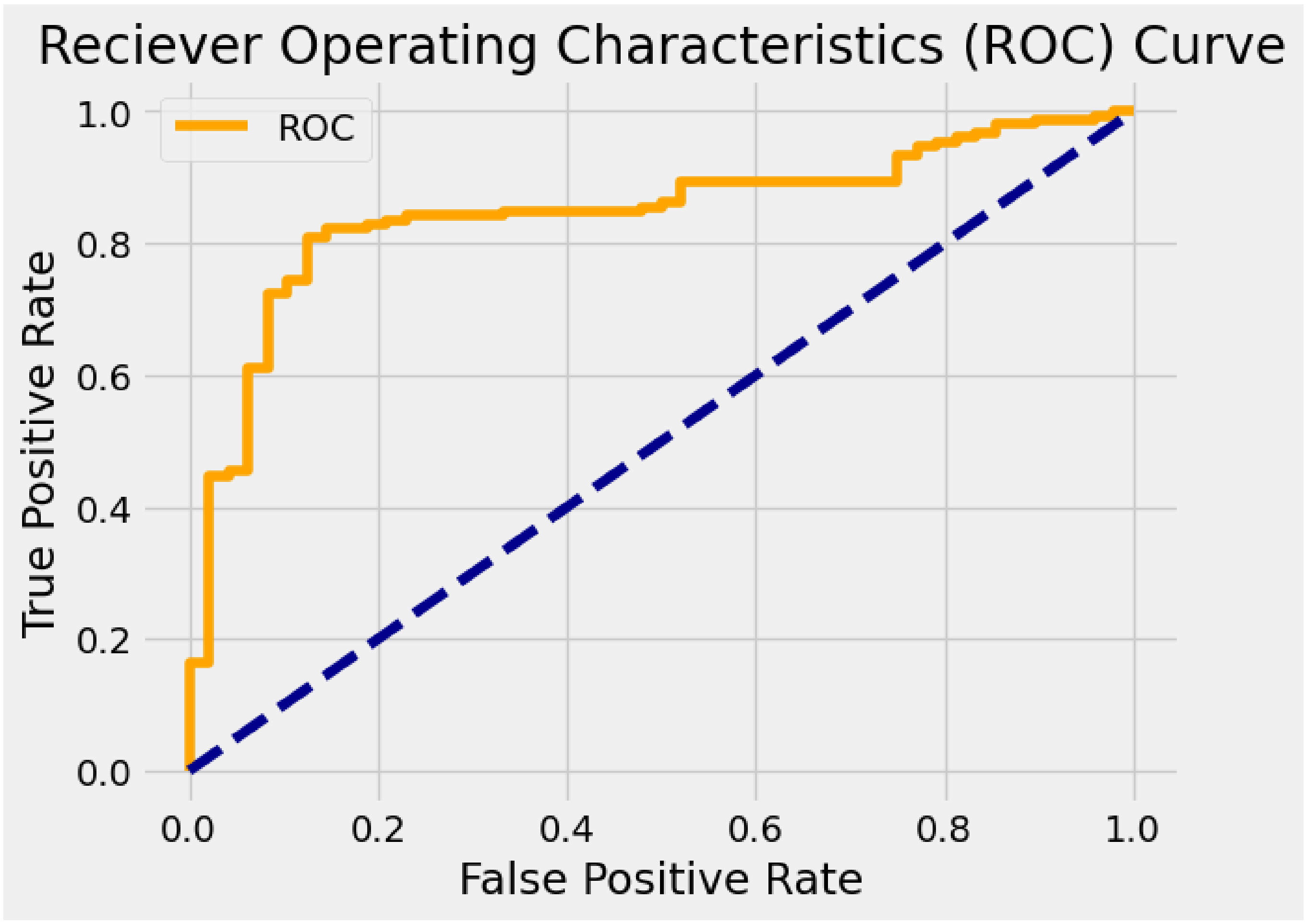
```
model = model.fit(X_Train, Y_Train)
pred = model.predict(X_Test)
print ('Accuracy:', accuracy_score(Y_Test, pred))
print ('\n clasification report:\n', classification_report(Y_Test, pred))
print ('\n confussion matrix:\n',confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.81

 clasification report:
               precision    recall  f1-score   support

           0       0.59      0.71      0.64        48
           1       0.90      0.84      0.87       152

    accuracy                           0.81       200
   macro avg       0.74      0.78      0.76       200
weighted avg       0.83      0.81      0.82       200


 confussion matrix:
 [[ 34  14]
 [ 24 128]]
```

```
# Improvement by findind optimal threshold from ROC Curve
pred_prob = model.predict_proba(X_Test)[:, 1]
from sklearn import metrics
def plot_roc_curve(fpr, tpr) :
  plt.plot(fpr, tpr, color = 'orange',label = 'ROC')
  plt.plot([0, 1],[0,1], color ='darkblue', linestyle = '--')
  plt.xlabel('False Positive Rate')
  plt.ylabel('True Positive Rate')
  plt.title("Reciever Operating Characteristics (ROC) Curve")
  plt.legend()
  plt.show()

y_true = Y_Test
y_scores = pred_prob
fpr, tpr, thresholds = metrics.roc_curve(y_true, y_scores)
print(tpr)
print(fpr)
print(thresholds)
print(metrics.roc_auc_score(y_true, y_scores))
optimal_idx = np.argmax(tpr- fpr)
optimal_threshold = thresholds[optimal_idx]
```

```python
print("Threshold value is :", optimal_threshold)
plot_roc_curve(fpr, tpr)
```

```
[0.         0.00657895 0.16447368 0.16447368 0.44736842 0.44736842
 0.45394737 0.45394737 0.61184211 0.61184211 0.72368421 0.72368421
 0.74342105 0.74342105 0.80921053 0.80921053 0.82236842 0.82236842
 0.82894737 0.82894737 0.83552632 0.83552632 0.84210526 0.84210526
 0.84868421 0.84868421 0.85526316 0.85526316 0.86184211 0.86184211
 0.89473684 0.89473684 0.93421053 0.93421053 0.94736842 0.94736842
 0.95394737 0.95394737 0.96052632 0.96052632 0.96710526 0.96710526
 0.98026316 0.98026316 0.98684211 0.98684211 0.99342105 0.99342105
 1.         1.        ]
[0.         0.         0.         0.02083333 0.02083333 0.04166667
 0.04166667 0.0625     0.0625     0.08333333 0.08333333 0.10416667
 0.10416667 0.125      0.125      0.14583333 0.14583333 0.1875
 0.1875     0.20833333 0.20833333 0.22916667 0.22916667 0.33333333
 0.33333333 0.47916667 0.47916667 0.5        0.5        0.52083333
 0.52083333 0.75       0.75       0.77083333 0.77083333 0.79166667
 0.79166667 0.8125     0.8125     0.83333333 0.83333333 0.85416667
 0.85416667 0.89583333 0.89583333 0.95833333 0.95833333 0.97916667
 0.97916667 1.        ]
[1.9940485  0.9940485  0.98860323 0.9883078  0.9726742  0.9722946
 0.97201365 0.971659   0.9539386  0.9532652  0.9263027  0.92191756
 0.90673894 0.9049473  0.8191887  0.7902295  0.76396227 0.69636214
 0.6945663  0.6931556  0.65721434 0.6507218  0.6186751  0.49216715
 0.48778805 0.43487206 0.42102364 0.4075923  0.39813107 0.39568675
 0.35689345 0.2449969  0.22745597 0.21950081 0.21605468 0.20262408
 0.19832571 0.19188912 0.1888623  0.18160102 0.18082689 0.16552751
 0.13859107 0.12913214 0.11563088 0.1040293  0.08260269 0.06147746
 0.05761063 0.04951807]
0.8469024122807017
Threshold value is : 0.8191887
```



```
pred = 1*(pred_prob> optimal_threshold)
# After finding optimal thresholds from ROC
print ('Accuracy:', accuracy_score(Y_Test, pred))
print ('\n clasification report:\n', classification_report(Y_Test, pred))
print ('\n confussion matrix:\n',confusion_matrix(Y_Test, pred))
```

```
Accuracy: 0.82

 clasification report:
              precision    recall  f1-score   support

           0       0.58      0.88      0.70        48
           1       0.95      0.80      0.87       152

    accuracy                           0.82       200
   macro avg       0.77      0.84      0.79       200
weighted avg       0.86      0.82      0.83       200
```

```
confussion matrix:
[[ 42   6]
 [ 30 122]]
```

```
X_Test
```

```
model.predict(X_Test.iloc[[78]]) # Pass a DataFrame instead of a single row
```

```
array([1])
```

```

import pickle


# Assuming 'model' from previous cells is the DecisionTreeClassifier you want to save
filename='dtc_model.pkl'
pickle.dump(model, open(filename,'wb')) # Replace 'b_dtc' with 'model'
```