IDENTIFYING INSURANCE CLAIM FRAUDS DETECTION USING MACHINE LEARNING TECHNIQUE

A PROJECT REPORT

Submitted by

P. GOKUL **(412321104012)**

M. GOKULSINGH **(412321104013)**

B. GUNASEKAR **(412321104015)**

B. JAYARAMAN **(412321104025)**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

SRI RAMANUJAR ENGINEERING COLLEGE

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2025

IDENTIFYING INSURANCE CLAIM FRAUDS DETECTION USING MACHINE LEARNING TECHNIQUE

A PROJECT REPORT

Submitted by

P. GOKUL **(412321104012)**

M. GOKULSINGH **(412321104013)**

B. GUNASEKAR **(412321104015)**

B. JAYARAMAN **(412321104025)**

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

SRI RAMANUJAR ENGINEERING COLLEGE

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2025

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this Project Report **"IDENTIFYING INSURANCE CLAIM FRAUDS DETECTION USING MACHINE LEARNING TECHNIQUE "** is the Bonafide work of

P. GOKUL	**(412321104012)**
M. GOKULSINGH	**(412321104013)**
B. GUNASEKAR	**(412321104015)**
B. JAYARAMAN	**(412321104025)**

Who carried out the project work under my supervision.

SIGNATURE
Mr. P. SIVAKUMAR, M.E.,
HEAD OF THE DEPARTMENT
Associate Professor,
Computer Science and Engineering,
Sri Ramanujar Engineering College,
Kolapakkam, Vandalur,
Chennai -600127.

SIGNATURE
Ms. K. LAKSHMI. M.E.,
SUPERVISOR
Assistant Professor,
Computer Science and Engineering,
Sri Ramanujar Engineering College,
Kolapakkam, Vandalur,
Chennai -600127.

Submitted the project work and viva voce held on ……
at Sri Ramanujar Engineering College, Chennai-127.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

At the juncture of presenting this project, We have a immense pleasure in expressing our heart-felt thanks to various personnel who have helped us at various levels to complete this project successfully.

At the outset, We express our respectful and sincere thanks with deep sense of gratitude to our beloved correspondent **Thiru.M.NITHYA SUNDAR, M.COM.,M.SC.,** and our beloved secretary **Thiru.G.KAMARAJ, B.A.** for providing all the necessary facility & guiding us in a right path of life with their enlightened wisdom.

Words are inadequate to express our feelings of gratitude to thank our beloved Principal **Prof.Dr.A. DHANAPAL., ME, PH.D.,** for his constant motivation and encouragement.

We feel greatly indebted to our H.O.D **Mr.P. SIVAKUMAR., ME.,** for his propellant guidance, kind advice and encouragement. We would like to express our sincere thanks to our Project Coordinators **Ms.K. LAKSHMI., ME.,** they always provide us with their scholarly ideas, guidance and suggestions for monitoring our performance throughout the Project work.

Our heartfelt thanks to Project guide **Ms.K. LAKSHMI., ME.,** for her continual guidance & support with her suggestion for the successful completion of the project work.

One personal note, We would like to use this opportunity to extend our heartiest and respectable thanks to all teaching and non-teaching faculty members of our department, friends and all well wishers, who helped us in completing this project work successfully.

Sincerely We thank our Parents who stand behind us in each and every steps with abundant blessings of the Almighty.

ABSTRACT

Insurance Company working as commercial enterprise from last few years has been experiencing fraud cases for all type of claims. Amount claimed by fraudulent is significantly huge that may cause serious problems, hence along with government, different organization also working to detect and reduce such activities. Such frauds occurred in all areas of insurance claim with high severity such as insurance claimed towards auto sector is fraud that widely claimed and prominent type, which can be done by fake accident claim. So, we aim to develop a project that work on insurance claim data set to detect fraud and fake claims amount. The project implements machine learning algorithms to build model to label and classify claim. Also, to study comparative study of all machine learning algorithms used for classification using confusion matrix in term soft accuracy, precision, recall etc. For fraudulent transaction validation, machine learning model is built using PySpark Python Library.

ABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

TERMS	EXPANSIONS
ML	**M**achine **L**earning
AI	**A**rtificial **I**ntelligence
IDE	**I**ntegrated **D**evelopment **E**nvironment
DFD	**D**ata Flow **D**iagram
RF	**R**andom **F**orest
LR	**L**ogistic **R**egression
SVM	**S**upport **V**ector **M**achine
DB	**D**atabase
API	**A**pplication **P**rogramming **I**nterface
GUI	**G**raphical **U**ser **I**nterface
CSV	**C**omma **S**eparated **V**alues
CRUD	**C**reate, Read, **U**pdate, **D**elete
HTTP	**H**ypertext **T**ransfer **P**rotocol
WSGI	**W**eb **S**erver **G**ateway **I**nterface

ORM	**O**bject **R**elational **M**apper
UI	**U**ser **I**nterface
URL	**U**niform **R**esource **L**ocator
OOP	**O**bject-**O**riented **P**rogramming
SQL	**S**tructured **Q**uery **L**anguage
PySpark	**P**ython **A**PI for **S**park
JS	**J**avaScript
JSON	**J**avaScript **O**bject **N**otation

CHAPTER 1

INTRODUCTION

The insurance industries consist of more than a thousand companies worldwide. And collect more than one trillion dollars premiums each year. Vehicle insurance fraud is the most prominent type of insurance fraud, which can be done by fake accident claim. In this project, focusing on detecting vehicle fraud by using machine learning techniques. Fraud is one of the largest and most well-known problems that insurers face. Fraudulent claims can be highly expensive for each insurer. Therefore, it is important to know which claims are correct and which are not. Ideally, an insurance agent would have the capacity to investigate each case and conclude whether it is genuine or not. However, this process is not only time consuming, but costly. Sourcing and funding the skilled labor required to review each of the thousands of claims that are filed a day is simply unfeasible. This is where machine learning comes in to save the day. Once the proper data is fed to the system it'll be very easy to find out if the claim is genuine or not.

Fraud is a criminal misrepresentation which causes hindrance for both the individual and the organization. The insurance industries are now implementing various techniques for the effective management of fraud. There are two major types of fraud, hard insurance fraud and soft insurance fraud. When a few people intentionally fake an accident then that type of fraud is known as hard insurance fraud. And when the person has an insurance claim that is valid but falsifies the part of the claim is known as soft insurance fraud.

So, the organizations should implement various techniques that are helpful for fraud detection to increase customer satisfaction. When the number of undetected fraud cases increases then the premium amount also increases to compensate the losses, which in turn affects the insured parties.

Insurance fraud detection is a difficult issue, given the assortment of misrepresentation designs and moderately small proportion of known frauds in regular examples. While building identification models, the investment funds from misfortune anticipation should be offset with cost of false alarms. Various machine learning methods consider improving prescient precision, empowering misfortune control units to accomplish higher inclusion with low false positive rates. In this paper, numerous machine learning strategies for misrepresentation location are introduced and their exhibition on different data collection sets are analyzed. The feature engineering impact, parameter tweaking, and feature selection are investigated with the objective of obtaining prevalent prescient execution. Besides, the extortion specialists may confront numerous unfavorable circumstances while detecting the vehicle protection misrepresentation cases for the most part happen because of two reasons.

Firstly, any absent or wrong case data makes the extortion identification testing challenging. Furthermore, it is additionally discovered that the quantity of noxious cases is significantly less than the absolute cases submitted. This uneven dispersion (information imbalance) prompts progressively troublesome extortion recognition. Moreover, the majority of the supervised classifiers create inefficient classification models with unequal information, since they tend to order all the information focuses as certifiable class (significant class tests) and overlook the deceitful focuses (minority class tests). Insurance frauds spread the scope of inappropriate exercises which an individual may submit to accomplish a good result from the insurance company.

Machine Learning is said to be a subset of Artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. Machine learning uses data to detect various patterns in each dataset. It can learn from past data and improve automatically.

1.1 OBJECTIVE OF THE PROJECT

1. To create a model that can accurately anticipate which transactions might be fraudulent.
2. To build a classification methodology using machine learning technique to determine whether a customer is placing a fraudulent insurance claim by using historical data.
3. To determine if a claim for insurance is legitimate or not.
4. To assess the effectiveness of the fraud detection algorithm.

1.2 PROBLEM STATEMENT

1. Inability to perceive the context-specific relationships between the parameters (geography, client section, insurance sales process) which may not mirror the typical picture.

2. Constrained to control with the restricted set of not able parameters supported heuristic knowledge – where, as being aware that a number of the opposite attributes might conjointly influence the decisions.

3. Reconstruction of the given model is that the hand operated exercise that needs to be conducted sporadically to react dynamic behavior. Also, to make sure that the model gives feedback from the examinations. The flexibility to manage this standardization is tougher.

4. The incidence of occurrence of fraud is low – generally but 1 percent of claims area unit classified.

5. Consultations with business specialists point out that there is not a typical model to determine the model exactly like the context.

CHAPTER 2

LITERATURE SURVEY

[1] Prediction of insurance fraud detection using machine learning algorithms.

Author Name: Laiqa Rukhsar; Waqas Haider Bangyal; Kashif Nisar; Sana Nisar.

DESCRIPTION

In the current era, people are influenced with various types of insurance such as health insurance, automobile insurance, property insurance and travel insurance, due to the availability of extensive knowledge related to insurance. People are trending to invest in such kinds of insurance, which helps the scam artist to cheat them. Insurance fraud is a prohibited act either by the client or vendor of the insurance contract The effectiveness of the algorithms are observed on the basis of performance metrics: Precision, Recall and F1-Score. The comparative results of classification algorithms conclude that DT gives the highest accuracy of 79% as compared to the other techniques.

[2] An intelligent unsupervised technique for fraud detection in health care systems.

Author Name: Kanksha, Bhaskar, Aman Pande, Sagar Malik, Rahul Khamparia, Aditya.

DESCRIPTION

Healthcare is an essential part of people's lives, particularly for the elderly population, and also should be economical. Medicare is one particular healthcare plan. Claims fraud is a significant contributor to increased healthcare expenses, though the effect of it could be lessened by fraud detection. In this paper, an analysis of various machine learning techniques was done to identify Medicare fraud. The isolated forest an unsupervised machine learning algorithm which improves overall performance while detecting fraud based upon outliers. The goal of this specific paper is generally to show probable dishonest providers on the ground of their allegations. Obtained results were found more promising compared to existing techniques. Around 98.76% accuracy is obtained using an isolated forest algorithm.

CHAPTER 3

SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

People are trending to invest in such kinds of insurance, which helps the scam artist to cheat them. Insurance fraud is a prohibited act either by the client or vendor of the insurance contract. Insurance fraud from the client's side is encountered in the form of overestimated claims and post-dated policies etc.

Data processing in a neighborhood of machine learning has advanced considerably within the current years. Data mining focuses on analyzing the whole data obtained.

On the contrary, within the different of getting the knowledge for world understanding is within the processing applications like machine learning, it uses the knowledge to locate patterns in information and improvise the program actions.

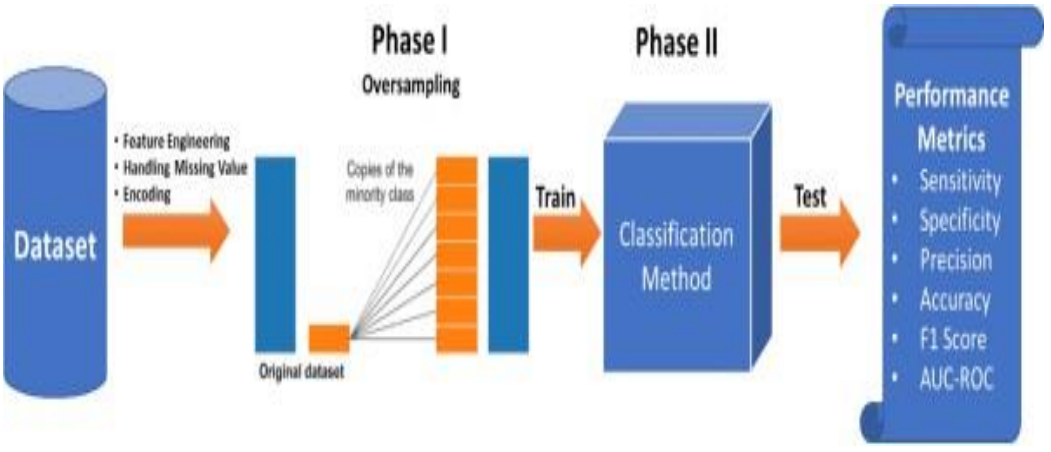


Fig 3.1 EXISTING BLOCK DIAGRAM

The proposed methodology was based on the idea of choosing the best from data partitions under-sampling. The models used for insurance fraud detection were Support Vector Machines, Decision Trees, and artificial neural networks. Experiments were carried out on a publicly available dataset containing the records of different automobile insurance claims. Results showed that the Decision Trees were the best among all the classifiers. It was demonstrated that the technique outperformed the previous work done and its accuracy was the best.

Machine learning approaches used in this research were Bayesian Networks, Decision Trees, and rule-based algorithms. This work created two Bayesian networks based on the assumptions i.e., driver is cheating, and the drivers are honest. And these two probabilities were calculated separately. And the one with the higher probability was considered as output. Decision trees were based on subtrees or labels known as classes i.e., legal and fraud in the research. Further Gini, minority, or entropy measures were used to get the impurity within a class and get the final output.

Rule-based system proceeded in with if-then rules and here conditions were driver's age, driver's rating, and auto age. Results and performance were shown in the form of confusion matric, and the accuracy was good.

3.2. PROPOSED SYSTEM

The dataset is gathered and cleaned before several models are tried on it. Several features of the model are engineered and evaluated again based on the original model performance.

Once all the options have been created, the model is created and run using entirely different entirely different values and entirely different entirely different iteration processes.

It is possible to determine whether an insurance claim is false by using a prediction model. The classification system used to identify fraudulent transactions is the subject of this report.

The effectiveness of the algorithms is observed on the basis of performance metrics: Precision, Recall and F1-Score. The comparative results of classification algorithms conclude that DT gives the highest accuracy of 79% as compared to the other techniques.

The methodological approach can be evaluated in four main steps:

- Data set collection
- Feature selection
- Classification
- Evaluation

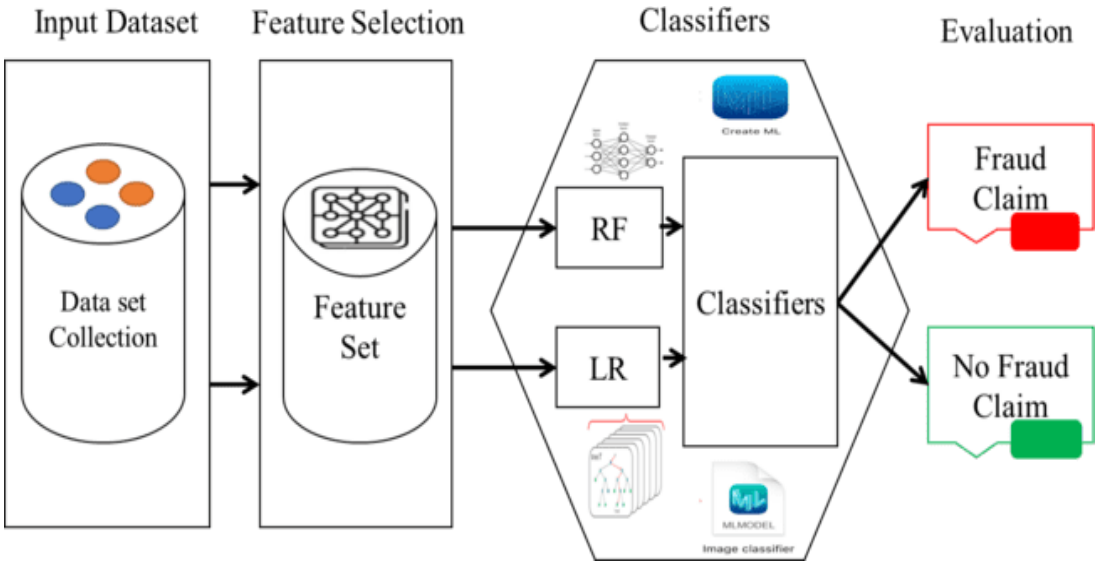


Fig 3.2 PROPOSED SYSTEM BLOCK ARCHITECTURE

Dataset collection

Before the various classification approaches to describe, it is important to introduce the data to be analyzed for predicting the fraud. This study is analyzed with an auto insurance fraud dataset. The raw dataset contains more than a thousand customers with some attributes.

Feature selection

In this section, we provide an overview of feature selection approaches. We then give a more detailed description for the specific feature selection methods used in our study. Feature selection techniques based on the selection strategy can be classified into three types: wrapper, filter and embedded. In machine learning and pattern recognition, a feature is an individual measurable property or characteristic of a phenomenon. Choosing informative, discriminating, and independent features is a crucial element of effective algorithms in pattern recognition, classification and regression.

Classification

- **Logistic Regression**

Logistic regression works with sigmoid function because the sigmoid function can be used to classify the output that is dependent feature, and it uses the probability for classification of the dependent feature. Regression helps to find out the relationship between input and target variable. Logistic regression is supervised ML algorithm that instead of classifying into different categories predicts a quantitative response within a continuous range of values, output has a constant slope. There are two types of Logistic regression Simple regression and Multivariable regression. By the term Logistic we understand that the two variables been on x and y axis are Logistic correlated. Logistic Regression has been widely used in Price prediction, Trends Prediction and Risk Management.

- **Random Forest**

The random forest randomly selects the features that are independent variables and randomly selects the rows by row sampling and the number of decision tree can be determined by using hyper parameter optimization. For classification problem statement the output is the maximum occurrence outputs from each decision tree model inside the random forest. This is one of the widely used machine learning algorithms in real word scenarios and in deployed models. And in most of the Kaggle computation challenges this algorithm is used to solve the problem statement. Leo Breiman and Adèle Cutler proposed the RF classifier in 2001. It works by utilizing the combined effect of two concepts "bagging" and "subspaces". From the training dataset, a set of decision trees are built and decision i.e. label is predicted based on votes collected from these decision tresses. RF provides high accuracy and is mainly used for classifying large datasets due to its ability to handle missing values. The application domains of RF include remote sensing, e-commerce, stock market, fraud detection, network intrusion detection, and so on.

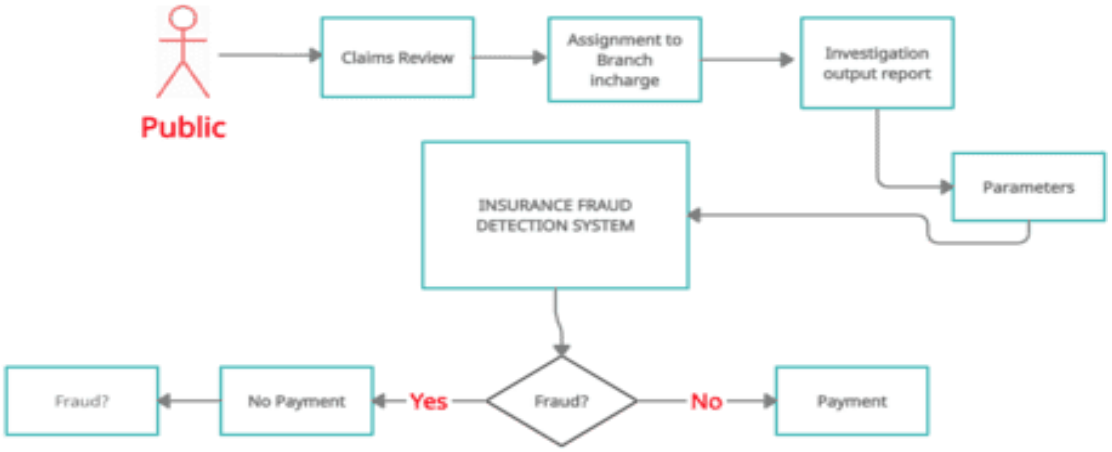


Fig 3.3 FLOW CHART

When public claims for an insurance, branch in-charge receives the claim and performs some investigation like, whether the public has claimed for an insurance before, if so what type of claim was it whether it's a fraudulent or genuine case, whether the premium has been paid properly then sends 8 parameters as input to our Insurance Fraud System. Our system performs some analysis and outputs as either fraud or genuine. If genuine then payment will be provided else case will be sent to investigation. If a customer claims for insurance, the details are initially gathered by the insurer and then the authenticity of the claim is validated using the ML technique.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

The equipment connection point might act as the reason for an agreement for the execution of the framework and ought to in this manner be a finished and predictable determination of the entire framework. They serve as the foundation for system design for software engineers. It shouldn't be about how the system works but rather what it does.

- System : Pentium i3 Processor.
- Hard Disk : 500 GB.
- Floppy Drive : 1.44 Mb.
- Monitor : 15'' LED.
- Mouse : Logitech.
- Ram : 4 GB.
- Input Devices : Keyboard, Mouse

4.2 SOFTWARE REQUIREMENTS

The system specification can be found in the software requirements document. The requirements definition and specification ought to be included. Instead describing how the system should operate, it specifies what it should do. The software requirements serve as the foundation for the software requirements specification, which can be used to estimate costs, plan team activities, carry out tasks, and monitor team progress throughout the development process.

CHAPTER 5

SYSTEM DESIGN

5.1. PROGRAMMING LANGUAGE PYTHON

Python is a popular and versatile programming language used for building websites, software, automation, and data analysis. It's known for being easy to learn and use. Python code is translated into a special kind of code called bytecode, which is then run by the computer.

5.1.1. Python Tools (IDEs):

An IDE is a software application that helps programmers write code efficiently. It includes tools for editing code, running programs, and fixing errors. IDLE is a basic IDE that comes with Python.

5.1.2. Key Programming Ideas:

- Scripting Language: A language used to automate tasks.
- Object-Oriented Programming (OOP): A way of organizing code using "objects" that contain data and actions.
- Data Types: Different kinds of data, like numbers, text (strings), and true/false values (booleans).
- Variables: Names that hold data values.
- Strings: Text in Python, enclosed in quotes.
- Data Structures: Ways to organize and store collections of data. Common ones in Python include:
  - Lists: Ordered, changeable sequences of items.
  - Tuples: Ordered, unchangeable sequences of items.
  - Dictionaries: Collections of key-value pairs.
  - Sets: Unordered collections of unique items.

- Operators: Symbols used to perform operations (like math or comparisons).
- Conditional Statements (If/Else): Used to make decisions in code based on conditions.
- Loops (For/While): Used to repeat blocks of code.
- Modules: Python files that contain reusable code.
- Packages: Collections of modules.
- Functions: Blocks of reusable code that perform specific tasks.

5.1.3. Python Libraries (for specific tasks):

- Matplotlib: For creating graphs and charts.
- Pandas: For data analysis and manipulation (using structures like Data Frames and Series).
- NumPy: For working with large arrays and mathematical functions.

5.1.4. Web Framework - Flask:

Flask is a tool that makes it easier to build web applications using Python. It handles the behind-the-scenes work so you can focus on creating your website or web app.

5.2. ALGORITHM / MODEL USED

1. Random Forest

2. Logistic Regression

5.2.1. Random Forest Classifier

The random forest randomly selects the features that is independent variables and also randomly selects the rows by row sampling and the number of decision tree can be determined by using hyper parameter optimization. For classification problem statement the output is the maximum occurrence outputs from each

decision tree models inside the random forest. This is one the widely used machine learning algorithm in real word scenarios and in deployed models. And in most of the Kaggle computation challenges this algorithm is used to solve the problem statement.

5.2.2. Logistic Regression Classifier

Logistic regression works with sigmoid function because the sigmoid function can be used to classify the output that is dependent feature, and it uses the probability for classification of the dependent feature.

5.3. Web Framework - FLASK

Flask is a web framework, it's a Python module that lets you develop web applications easily. It's has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features. It does have many cool features like url routing, template engine. It is a WSGI web app framework. A Web Application Framework or a simply a Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management, and so on.

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeg WSGI toolkit and the Jinja2 template engine.Both are Pocco projects.

WSGI

The Web Server Gateway Interface (Web Server Gateway Interface, WSGI) has been used as a standard for Python web application development. WSGI is the specification of a common interface between web servers and web applications.

Werkzeug

Werkzeug is a WSGI toolkit that implements requests, response objects, and utility functions. This enables a web frame to be built on it. The Flask framework uses Werkzeg as one of its bases.

jinja2

jinja2 is a popular template engine for Python. A web template system combines a template with a specific data source to render a dynamic web page.

Microframework

Flask is often referred to as a microframework. It is designed to keep the core of the application simple and scalable. Instead of an abstraction layer for database support, Flask supports extensions to add such capabilities to the application. Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Flash is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website. Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. In the case of Flask, its dependencies are:

CHAPTER 6

MODULES

6.1. MODULES

1) **Training Module**

   a) **Data Collection**

   b) **Data pre-processing**

   c) **Model Training**

2) **Testing Module**

   a) **Login**

   b) **Prediction**

   c) **Deployment in Cloud**

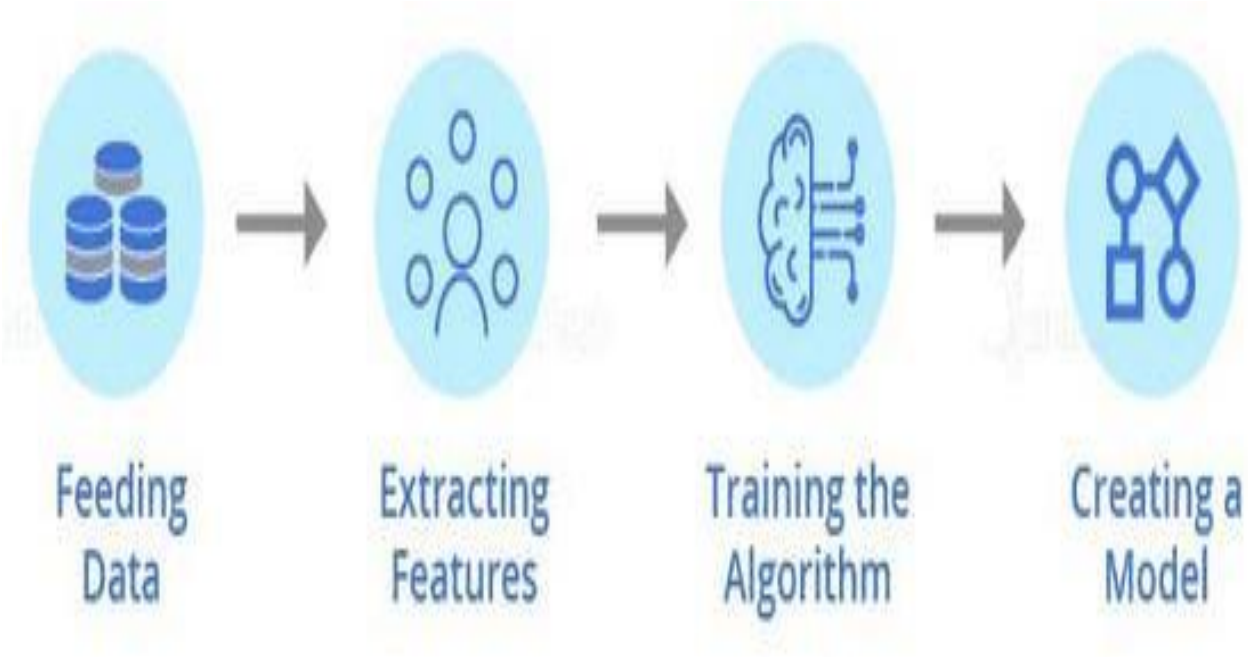


Fig 6.1 MODULES

6.2. MODULES DESCRIPTION

Vehicle insurance fraud is one of the leading categories of fraud, which will be carried out by faking accident claim. In this study, we are concentrating on tracking down auto insurance fraud by making use of Machine Learning Techniques.

MODULE 1: TRAINING MODULE: -

A training model is a dataset that is used to train an ML algorithm. It consists of sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output.

MODULE A: DATASET COLLECTION: -

Before the various classification approaches to describe, it is important to introduce the data to be analyzed for predicting the fraud. This study is analyzed with an auto insurance fraud dataset. The raw dataset contains more than a thousand customers with some attributes.

MODULE B: DATA PREPROCESSING: -

Data Pre-processing considerably inhibits Data Mining. Clean data is usually not possible, and it may contain impossible combinations, missing values, noise, inconsistencies, etc. The quality of the data is the first and foremost requirement before applying the algorithm.

MODULE C: MODEL TRAINING: -

Training a model simply means learning (determining) good values for all the weights and the bias from labeled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting

to find a model that minimizes loss; this process is called empirical risk minimization.

MODULE 2: TESTING MODULE: -

In vehicle insurance fraud detection using machine learning, model testing is referred to as the process where the performance of a fully trained model is evaluated on a testing set.

MODULE A: LOGIN: -

In computer security, logging is the process by which an individual gains access to a computer system by identifying and authenticating themselves. The user credentials are typically some forms of "Username" and a matching "password", and these credentials themselves are sometimes referred to as a login.

MODULE B: PREDICTION: -

Suppose one of the features has an outlier, then the distance will be governed by this feature. Secondly, the gradient descent converges much faster with feature scaling. Here the dataset is exposed to both insurance claims, is classified by using ML algorithms and predict the fraud claim or no fraud claim.

MODULE C: DEPLOYMENT IN CLOUD: -

Cloud deployment is the process of deploying an application through one or more hosting models. And update after classified the ML techniques the fraud is detected or not and the money is claim or no claim.

6.3. RESULTS AND DISCUSSION

As fraud poses a serious problem in current society, it has to be resolved. To resolve these problems, we can build systems which predict fraud in the data given. These systems are built using various machine learning techniques

networks. In this paper we have discussed various ML techniques and how it is implemented in the systems and how accurate it is in predicting fraud. Later these techniques are compared using two criteria from different perspectives.

DFD – Level 0

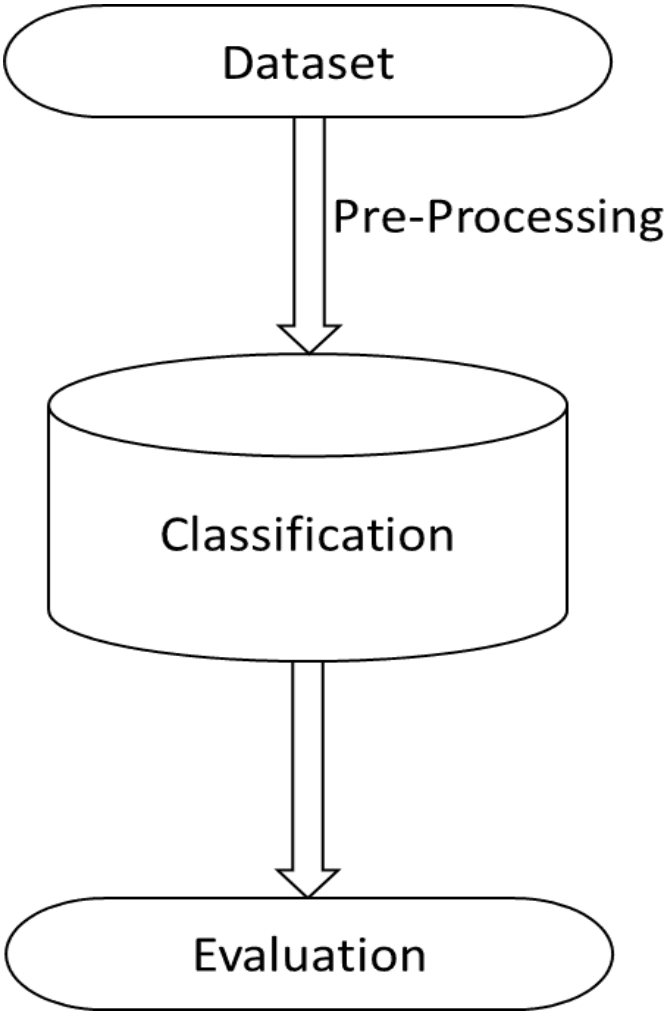


Fig 6.2 DFD – Level 0 Diagram

DFD Level -1 Diagram

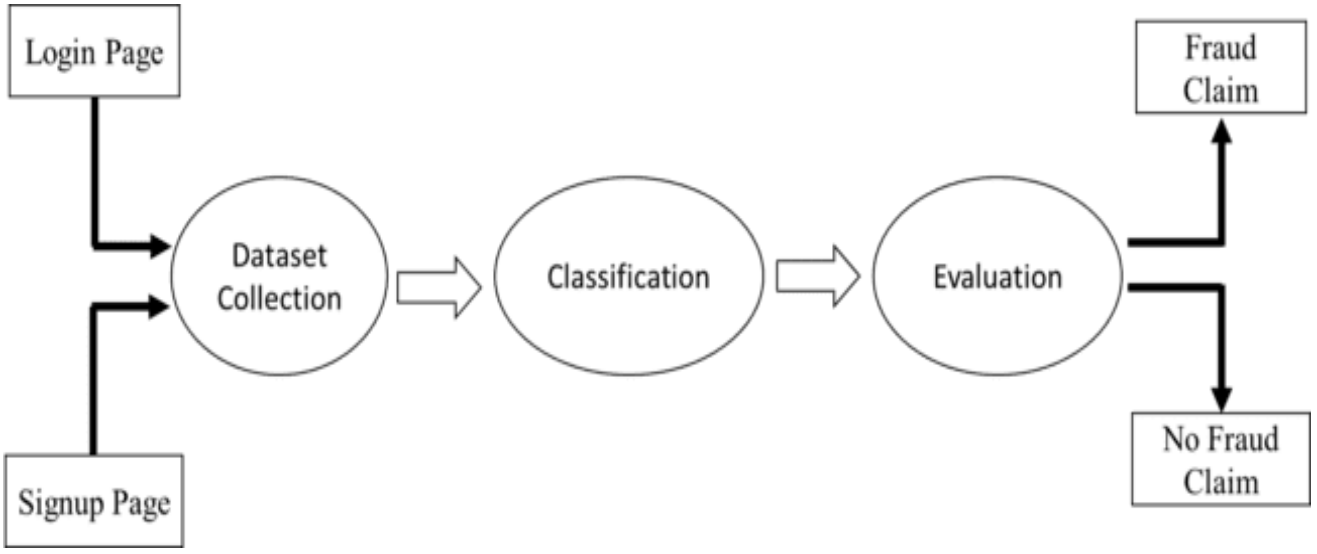


Fig 6.3 DFD – Level 1 Diagram

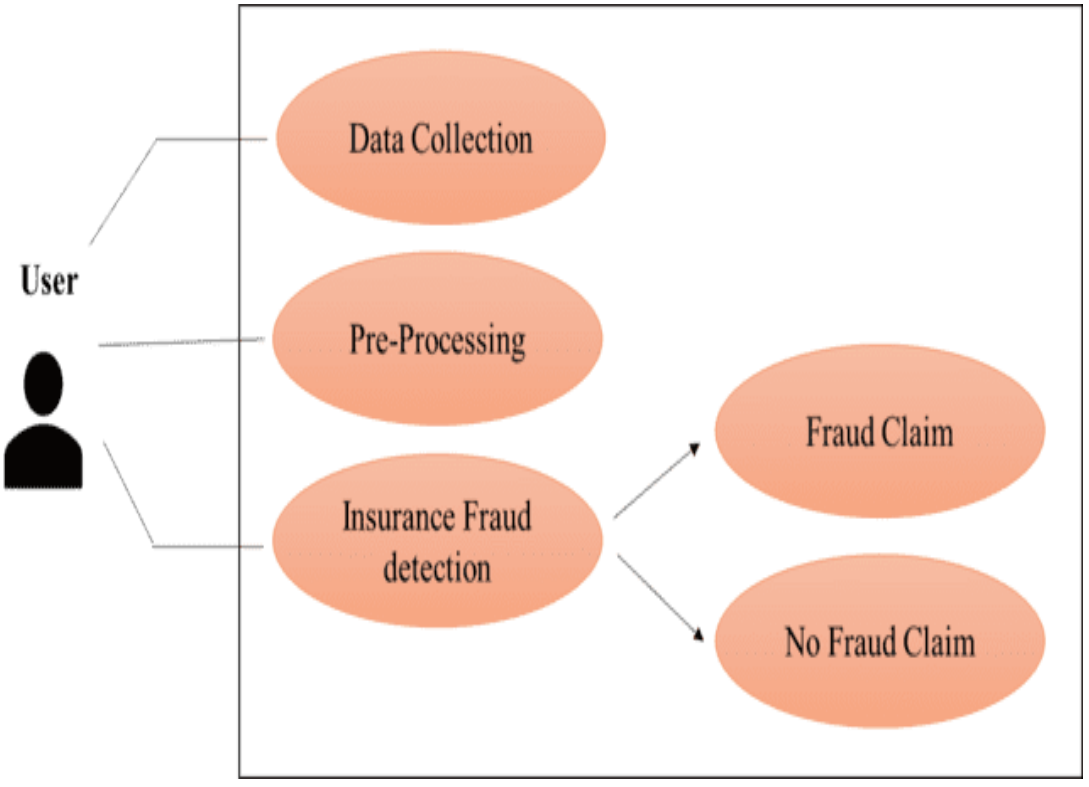


Fig 6.4 Use Case Diagram

```
cat_df.head()
```

	policy_csl	insured_sex	insured_education_level	insured_occupation	insured_relationship	incident_type	collision_type	incident_severity
0	250/500	MALE	MD	craft-repair	husband	Single Vehicle Collision	Side Collision	Major Damage
1	250/500	MALE	MD	machine-op-inspct	other-relative	Vehicle Theft	Rear Collision	Minor Damage
2	100/300	FEMALE	PhD	sales	own-child	Multi-vehicle Collision	Rear Collision	Minor Damage
3	250/500	FEMALE	PhD	armed-forces	unmarried	Single Vehicle Collision	Front Collision	Major Damage
4	500/1000	MALE	Associate	sales	unmarried	Vehicle Theft	Rear Collision	Minor Damage

authorities_contacted	property_damage	police_report_available	fraud_reported
Police	YES	YES	Y
Police	NO	NO	Y
Police	NO	NO	N
Police	NO	NO	Y
None	NO	NO	N

```
from matplotlib.pyplot import figure

figure(figsize=(8, 6), dpi=70)
data['insured_sex'].value_counts().plot(kind='bar')
plt.ylabel("Count")
```

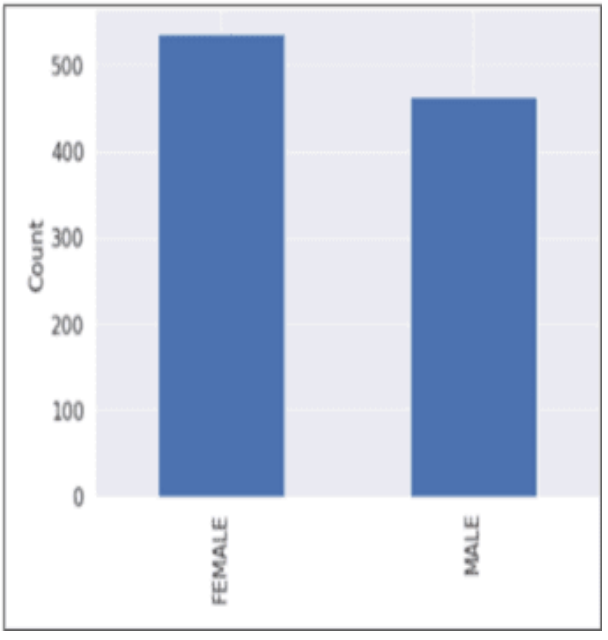


Fig 6.5 Modules Implemented- Data Collection

CONCLUSION

The problem of claim leakage deprives insurance companies of a sizeable portion of their anticipated benefit. For insurance companies, fraudulent claims are a serious and expensive problem that might cost the industry billions of dollars per year in unwarranted expenses. In conclusion, using machine learning techniques to detect insurance claim frauds provides a smart and efficient way to support insurance companies in identifying suspicious activities. By analyzing patterns in data, the model can help reduce financial losses and improve the speed and accuracy of fraud detection. This approach not only saves time and resources but also helps build trust with genuine customers. Although there are some challenges, such as data quality and model explainability, the results show that machine learning has strong potential in the field of fraud detection. With further improvements and more real-time applications, this system can become an essential tool for the insurance industry.

FUTURE ENHANCEMENTS

In the future, the insurance fraud detection system can be improved by collecting and using more real-world data to train the machine learning model, which will help increase accuracy and reliability. Additional useful features such as customer behavior patterns, claim history, and geographical details can be included to give the model a better understanding of fraud indicators. Advanced algorithms like XGBoost, LightGBM, or even deep learning techniques can be explored to improve prediction performance. The system can also be upgraded to work in real-time, so fraud can be detected as soon as a claim is submitted. To make the model more trustworthy, explainable AI tools like SHAP or LIME can be used to show why a particular claim is marked as fraudulent. Combining multiple models through ensemble methods may further enhance accuracy. Improving data cleaning processes and handling missing or incorrect information more effectively can also make a big difference. A user-friendly dashboard can be created for insurance staff to easily view and understand predictions. Adding a feedback system where users can confirm or correct predictions can help the model learn and improve continuously. Lastly, ensuring strong data privacy and security will be important to protect sensitive customer information and meet legal requirements.

APPENDICES - I

SOURCE CODE

APP.PY

```
from flask import Flask, redirect, request, render_template, flash, url_for

import os

import csv

import pymongo

import pandas as pd

import numpy as np

import pickle

s = 0

# client =
pymongo.MongoClient("mongodb+srv://Sivakumar23:siva2310@atlascluster.ti
g0e4k.mongodb.net/?retryWrites=true&w=majority")

client =
pymongo.MongoClient("mongodb+srv://gunasekar:gunasekar@shopez.x9dqx.
mongodb.net/?retryWrites=true&w=majority&appName=shopEZ")

# db = client.Farud_detection

db = client["Fraud_detection"]

c1 = db["users"]

c2 = db["files"]

flag1 = 0

database = []

app = Flask(_name_)

def newpredict():
```

```
data1=['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', 'policy_state_IL', 'policy_state_IN',
'policy_state_OH', 'policy_csl_100/300', 'policy_csl_250/500',
'policy_csl_500/1000', 'insured_sex_FEMALE', 'insured_sex_MALE',
'insured_education_level_Associate', 'insured_education_level_College',
'insured_education_level_High School', 'insured_education_level_JD',
'insured_education_level_MD', 'insured_education_level_Masters',
'insured_education_level_PhD', 'insured_occupation_adm-clerical',
'insured_occupation_armed-forces', 'insured_occupation_craft-repair',
'insured_occupation_exec-managerial', 'insured_occupation_farming-fishing',
'insured_occupation_handlers-cleaners', 'insured_occupation_machine-op-
inspct', 'insured_occupation_other-service', 'insured_occupation_priv-house-
serv', 'insured_occupation_prof-specialty', 'insured_occupation_protective-serv',
'insured_occupation_sales', 'insured_occupation_tech-support',
'insured_occupation_transport-moving', 'insured_hobbies_base-jumping',
'insured_hobbies_basketball', 'insured_hobbies_board-games',
'insured_hobbies_bungie-jumping', 'insured_hobbies_camping',
'insured_hobbies_chess', 'insured_hobbies_cross-fit', 'insured_hobbies_dancing',
'insured_hobbies_exercise', 'insured_hobbies_golf', 'insured_hobbies_hiking',
'insured_hobbies_kayaking', 'insured_hobbies_movies',
'insured_hobbies_paintball', 'insured_hobbies_polo', 'insured_hobbies_reading',
'insured_hobbies_skydiving', 'insured_hobbies_sleeping',
'insured_hobbies_video-games', 'insured_hobbies_yachting',
'insured_relationship_husband', 'insured_relationship_not-in-family',
'insured_relationship_other-relative', 'insured_relationship_own-child',
'insured_relationship_unmarried', 'insured_relationship_wife',
'incident_type_Multi-vehicle Collision', 'incident_type_Parked Car',
'incident_type_Single Vehicle Collision', 'incident_type_Vehicle Theft',
'collision_type_?', 'collision_type_Front Collision', 'collision_type_Rear
Collision', 'collision_type_Side Collision', 'incident_severity_Major Damage',
'incident_severity_Minor Damage', 'incident_severity_Total Loss',
'incident_severity_Trivial Damage', 'authorities_contacted_Ambulance',
'authorities_contacted_Fire', 'authorities_contacted_None',
'authorities_contacted_Other', 'authorities_contacted_Police',
'incident_state_NC', 'incident_state_NY', 'incident_state_OH',
'incident_state_PA', 'incident_state_SC', 'incident_state_VA',
'incident_state_WV', 'incident_city_Arlington', 'incident_city_Columbus',
'incident_city_Hillsdale', 'incident_city_Northbend', 'incident_city_Northbrook',
'incident_city_Riverwood', 'incident_city_Springfield', 'property_damage_?',
'property_damage_NO', 'property_damage_YES', 'police_report_available_?',
```

'police_report_available_NO', 'police_report_available_YES',
'auto_make_Accura', 'auto_make_Audi', 'auto_make_BMW',
'auto_make_Chevrolet', 'auto_make_Dodge', 'auto_make_Ford',
'auto_make_Honda', 'auto_make_Jeep', 'auto_make_Mercedes',
'auto_make_Nissan', 'auto_make_Saab', 'auto_make_Suburu',
'auto_make_Toyota', 'auto_make_Volkswagen', 'auto_model_3 Series',
'auto_model_92x', 'auto_model_93', 'auto_model_95', 'auto_model_A3',
'auto_model_A5', 'auto_model_Accord', 'auto_model_C300',
'auto_model_CRV', 'auto_model_Camry', 'auto_model_Civic',
'auto_model_Corolla', 'auto_model_E400', 'auto_model_Escape',
'auto_model_F150', 'auto_model_Forrestor', 'auto_model_Fusion',
'auto_model_Grand Cherokee', 'auto_model_Highlander',
'auto_model_Impreza', 'auto_model_Jetta', 'auto_model_Legacy',
'auto_model_M5', 'auto_model_MDX', 'auto_model_ML350',
'auto_model_Malibu', 'auto_model_Maxima', 'auto_model_Neon',
'auto_model_Passat', 'auto_model_Pathfinder', 'auto_model_RAM',
'auto_model_RSX', 'auto_model_Silverado', 'auto_model_TL',
'auto_model_Tahoe', 'auto_model_Ultima', 'auto_model_Wrangler',
'auto_model_X5', 'auto_model_X6']

```
    print(len(data1))

    data11 = pd.DataFrame(0,index=np.arange(1),columns=data1)

    for i in data11:

        print(i)

    data = pd.read_csv('./testing.csv')

    fdata=pd.DataFrame()

    #print(data.dtypes)

    original_data = data.copy()

    #Remove Less Correlated Columns

    deleteCols = ["policy_number", "policy_bind_date", "insured_zip",
"incident_location", "incident_date"]

    data = data.drop(deleteCols, axis=1)

    list_hot_encoded = []

    for column in data.columns:
```

```
        if(data[column].dtypes==object):
            data = pd.concat([data, pd.get_dummies(data[column], prefix=column)], axis=1)
            list_hot_encoded.append(column)
    #Drop hot-encoded columns
    data = data.drop(list_hot_encoded, axis=1)
    print(len(data.columns))
    df3 = data11.assign(**data.iloc[0])
##    df3 = df3.drop('insured_sex_Male', axis=1)
##    df3 = df3.drop('property_damage_Yes', axis=1)
##    df3 = df3.drop('police_report_available_Yes', axis=1)
    for i in df3:
        print(i)
    print(len(df3))
    model = pickle.load(open('model1.pkl', 'rb'))
    array_features = [np.array(df3.iloc[0])]
    prediction = model.predict(array_features)
    print(type(prediction[0]))
    return prediction[0]
@app.route('/')
def home_page(title='Welcome to the Login Form Demo!'):
    print('1')
    return render_template('landing.html', title=title, ok='run')
@app.route('/signin', methods=['GET', 'POST'])
def login_page(title='Login Demo'):
```

```
    global msgs, error
    error = []
    if request.method == 'GET':
        return render_template('landing.html', ok='run')
    elif request.method == 'POST':
        print('3')
        print(request.form['email'], request.form['password'])
        item_details = c1.find({"email": request.form['email']})
        for item in item_details:
            if request.form['email'] == item['email'] and request.form['password'] ==
item['password']:
                msgs = 'Login Successfuly'
                print("flash")
                return render_template('landing.html', msgs=msgs, ok='ok')
        error = 'Email or Password Incorrect'
        print(error)
    return render_template('landing.html', error=error, ok='run')
@app.route('/signup', methods=['GET', 'POST'])
def signup_page(title='Signup Page'):
    global flag1, war, msgs
    error = ''
    war = ''
    msgs = ''
    user_data = {}
    if request.method == 'GET':
```

```
        flag1 = 0
        return render_template('landing.html', error=error, title=title, ok='run')
    elif request.method == 'POST':
        if request.form['password'] != request.form['confirm_password']:
            error = 'Password do not match'
            return render_template('landing.html', error=error, ok='run')
        if request.form['email'] != '' and request.form['fname'] != '' and
request.form['lname'] != '' and request.form['password'] != '' and
request.form['confirm_password'] != '':
            user_data = dict(request.form)
            print(user_data)
            item_details = c1.find({"email": request.form['email']})
            for item in item_details:
                if request.form['email'] == item['email']:
                    war = 'Email Already Registered'
                    return render_template('landing.html', war=war, ok='run')
            if flag1 == 0:
                item_1 = {
                    "Username": user_data['fname']+user_data['lname'],
                    "email": user_data['email'],
                    "password": user_data['password'],
                    "cpassword": user_data['confirm_password']
                }
                c1.insert_one(item_1)
                msgs = 'Registration Successfully'
                return render_template('landing.html', msgs=msgs, ok='run')
```

```
@app.route('/upload', methods=['GET', 'POST'])
def upload_file():
    global coll, newc, database
    error = ''
    war = ''
    msgs = ''
    if request.method == 'GET':
        return render_template('landing.html', ok='run')
    if request.method == 'POST':
        f = request.files['file']
        filename = f.filename
        df = pd.read_csv(f)  # csv file which you want to import
        records_ = df.to_dict(orient='records')
        database = db.list_collection_names()
        print(records_)
# for s in range (0,len(database)):
# if database[s] == filename:
# war="Please Change File Name"
# return render_template('landing.html',war=war)
# else:
        collection1 = db[f.filename]
        collection1.insert_many(records_)
        msgs = 'Upload Successfully'
        return render_template('landing.html', msgs=msgs, ok='ok1',
name=f.filename)
```

```
    else:
        error = 'Invalid Request'
        return render_template('landing.html', error=error)
@app.route('/download', methods=['GET', 'POST'])
def download():
    results = []
    if request.method == 'GET':
        return render_template('landing.html', ok='run')
    if request.method == 'POST':
        filename = request.form['filename']
        # print(filename)
        strfile = str(filename)
        cursor = db[strfile].find({}, {'_id': False})
        for document in cursor:
            print(document)
            results.append(dict(document))
            fieldnames = [key for key in results[0].keys()]
        return render_template('home.html', msgs='Download Successfully',
results=results, fieldnames=fieldnames, len=len)
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    msgs = ''
    war = ''
    if request.method == 'GET':
        war = "No File Found"
```

```
        return render_template('landing.html', war=war, ok='run')
    if request.method == 'POST':
        msgs = "Predict Successfully"
        return render_template('landing.html', msgs=msgs, ok='ok')
@app.route('/claimcheck1', methods=['GET', 'POST'])
def claimcheck1():
    return render_template('reg_page.html', ok='run')
@app.route('/logout', methods=['GET', 'POST'])
def logout():
    return render_template('landing.html', ok='run')
@app.route('/result', methods=['GET', 'POST'])
def images():
    return render_template('images.html', ok='run')
@app.route('/claimcheck', methods=['GET', 'POST'])
def claimcheck():
    if request.method == 'GET':
        war = "No File Found"
        return render_template('reg_page.html', war=war, ok='run')
    if request.method == 'POST':
        item = {
            "months_as_customer"        : int(request.form['1']),
            "age"                       : int(request.form['2']),
            "policy_number"             : int(request.form['3']),
            "policy_bind_date"          : request.form['4'],
            "policy_state"              : request.form['5'],
```

```
"policy_csl"               : request.form['6'],
"policy_deductable"        : int(request.form['7']),
"policy_annual_premium"    : float(request.form['8']),
"umbrella_limit"           : int(request.form['9']),
"insured_zip"              : int(request.form['10']),
"insured_sex"              : request.form['11'],
"insured_education_level"  : request.form['12'],
"insured_occupation"       : request.form['13'],
"insured_hobbies"          : request.form['14'],
"insured_relationship"     : request.form['15'],
"capital-gains"            : int(request.form['16']),
"capital-loss"             : int(request.form['17']),
"incident_date"            : request.form['18'],
"incident_type"            : request.form['19'],
"collision_type"           : request.form['20'],
"incident_severity"        : request.form['21'],
"authorities_contacted"    : request.form['22'],
"incident_state"           : request.form['23'],
"incident_city"            : request.form['24'],
"incident_location"        : request.form['25'],
"incident_hour_of_the_day" : int(request.form['26']),
"number_of_vehicles_involved": int(request.form['27']),
"property_damage"          : request.form['28'],
"bodily_injuries"          : int(request.form['29']),
"witnesses"                : int(request.form['30']),
```

```
        'police_report_available'    : request.form['31'],
        "total_claim_amount"         : int(request.form['32']),
        "injury_claim"               : int(request.form['33']),
        "property_claim"             : int(request.form['34']),
        "vehicle_claim"              : int(request.form['35']),
        "auto_make"                  : request.form['36'],
        "auto_model"                 : request.form['37'],
        "auto_year"                  : int(request.form['38'])
    }
    print(item)
    df = pd.DataFrame(item,index=[0])
    df.to_csv('./testing.csv',index=False)
    #print(df.dtypes)
    #print(df)
    xa=newpredict()
    if xa==0:
        print('1')
        pmsgs='https://freepngimg.com/save/27880-green-tick-
clipart/1400x1600' #default msg
        Ack='Thank You !'
        msgs="ORGINAL CLAIM"
    else:
        print('2')
        pmsgs='https://www.kindpng.com/picc/m/413-4131735_transparent-
tick-mark-png-transparent-background-transparent-background.png' #default
msg
```

```
            Ack='Sorry !!!'
            msgs="FRUAD CLAIM"
    return
render_template('popup.html',msgs=msgs,pmsgs=pmsgs,Ack=Ack,ok='run')
if (_name_ == '_main_'):
    app.secret_key = "abc123"
    app.run(host='0.0.0.0', port=7000)
```

NEW MODEL FOR VIUALIZATION

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
import numpy as np
import sklearn.metrics
pd.set_option('display.max_columns',  500)
pd.set_option('display.max_rows', 500)
#load & view raw data
df = pd.read_csv('./insuranceFraud_Dataset.csv')
df.nunique()
plt.style.use('fivethirtyeight')
ax = sns.countplot(x='fraud_reported', data=df, hue='fraud_reported')
df['fraud_reported'].value_counts() # Count number of frauds vs non-frauds
df['incident_state'].value_counts()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax = df.groupby('incident_state').fraud_reported.count().plot.bar(ylim=0)
ax.set_ylabel('Fraud reported')
plt.show()
plt.style.use('fivethirtyeight')
```

```
fig = plt.figure(figsize=(18,6))
ax = df.groupby('incident_date').total_claim_amount.count().plot.bar(ylim=0)
ax.set_ylabel('Claim amount ($)')
plt.show()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax = df.groupby('policy_state').fraud_reported.count().plot.bar(ylim=0)
ax.set_ylabel('Fraud reported')
plt.show()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax = df.groupby('incident_type').fraud_reported.count().plot.bar(ylim=0)
ax.set_xticklabels(ax.get_xticklabels(), rotation=20, ha="right")
ax.set_ylabel('Fraud reported')
plt.show()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax = sns.countplot(x='incident_state', data=df)
fig = plt.figure(figsize=(10,6))
ax = sns.countplot(y = 'insured_education_level', data=df)
ax.set_ylabel('policy_annual_premium')
plt.show()
fig = plt.figure(figsize=(10,6))
ax = (df['insured_sex'].value_counts()*100.0
/len(df)).plot.pie(autopct='%.1f%%', labels = ['Male', 'Female'], fontsize=12)
```

```
ax.set_title('% Gender')

plt.show()

fig = plt.figure(figsize=(10,6))

ax = (df['insured_relationship'].value_counts()*100.0 /len(df))\

.plot.pie(autopct='%.1f%%', labels = ['husband', 'wife', 'own-child', 'unmarried', 'other-relative', 'not-in-family'],

        fontsize=12)

ax.set_title('% Relationship')

plt.show()

fig = plt.figure(figsize=(10,6))

ax = (df['incident_type'].value_counts()*100.0 /len(df))\

.plot.pie(autopct='%.1f%%', labels = ['Parked Car', 'Single Vehile Collision', 'Multi-vehicle Collision', 'Vehicle Theft'],

        fontsize=12)

fig = plt.figure(figsize=(10,6))

ax = (df['authorities_contacted'].value_counts()*100.0 /len(df))\

.plot.pie(autopct='%.1f%%', labels = ['Police', 'Fire', 'Other', 'None', 'Ambulance'],

        fontsize=12)

fig = plt.figure(figsize=(10,6))

ax = sns.countplot(x='auto_make', data=df)

ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")

plt.show()

fig = plt.figure(figsize=(10,6))

ax = (df['incident_severity'].value_counts()*100.0 /len(df))\

.plot.pie(autopct='%.1f%%', labels = ['Major Damage', 'Total Loss', 'Minor
```

```
Damage', 'Trivial Damage'],
        fontsize=12)
fig = plt.figure(figsize=(10,6))
ax = sns.countplot(x='insured_hobbies', data=df)
ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
plt.show()
df["insured_occupation"].value_counts()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax= df.groupby('auto_make').vehicle_claim.count().plot.bar(ylim=0)
ax.set_ylabel('Vehicle claim')
ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
plt.show()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax= df.groupby('insured_hobbies').total_claim_amount.count().plot.bar(ylim=0)
ax.set_ylabel('Total claim amount')
ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
plt.show()
df['fraud_reported'].replace(to_replace='Y',  value=1,  inplace=True)
df['fraud_reported'].replace(to_replace='N',  value=0, inplace=True)
df.head()
df[['insured_zip']] = df[['insured_zip']].astype(object)
df.describe()
df.auto_year.value_counts()  # check the spread of years to decide on further
```

action.

```
df['vehicle_age'] = 2018 - df['auto_year'] # Deriving the age of the vehicle based on the year value

df['vehicle_age'].head(10)

bins = [-1, 3, 6, 9, 12, 17, 20, 24]  # Factorize according to the time period of the day.

names = ["past_midnight", "early_morning", "morning", 'fore-noon', 'afternoon', 'evening', 'night']

df['incident_period_of_day'] = pd.cut(df.incident_hour_of_the_day, bins, labels=names).astype(object)

df[['incident_hour_of_the_day', 'incident_period_of_day']].head(20)

# Check on categorical variables:

df.select_dtypes(include=['object']).columns  # checking categorcial columns

df = df.drop(columns = [
    'policy_number',
    'insured_zip',
    'policy_bind_date',
    'incident_date',
    'incident_location',
    'auto_year',
    'incident_hour_of_the_day'])

df.head(2)

# identify variables with '?' values

unknowns = {}

for i in list(df.columns):
    if (df[i]).dtype == object:
```

```
        j = np.sum(df[i] == "?")
        unknowns[i] = j
unknowns = pd.DataFrame.from_dict(unknowns, orient = 'index')
df.collision_type.value_counts()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax=
df.groupby('collision_type').police_report_available.count().plot.bar(ylim=0)
ax.set_ylabel('Police report')
ax.set_xticklabels(ax.get_xticklabels(), rotation=10, ha="right")
plt.show()
df.property_damage.value_counts()
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(10,6))
ax=
df.groupby('property_damage').police_report_available.count().plot.bar(ylim=0)
ax.set_ylabel('Police report')
ax.set_xticklabels(ax.get_xticklabels(), rotation=10, ha="right")
plt.show()
df.police_report_available.value_counts()
df.columns
df._get_numeric_data().head()  # Checking numeric columns
df._get_numeric_data().columns
df.select_dtypes(include=['object']).columns  # checking categorcial columns
```

```
# #### Applying one-hot encoding to convert all categorical variables except
out target variables

#  'collision_type',

#  'property_damage',

#  'police_report_available',

#  'fraud_reported'

# In[55]:

dummies = pd.get_dummies(df[[

    'policy_state',

    'policy_csl',

    'insured_sex',

    'insured_education_level',

    'insured_occupation',

    'insured_hobbies',

    'insured_relationship',

    'incident_type',

    'incident_severity',

    'authorities_contacted',

    'incident_state',

    'incident_city',

    'auto_make',

    'auto_model',

    'incident_period_of_day']])

dummies = dummies.join(df[[

    'collision_type',
```

```
    'property_damage',
    'police_report_available',
    "fraud_reported"]])
dummies.head()
X = dummies.iloc[:, 0:-1]
y = dummies.iloc[:, -1]
len(X.columns)
from sklearn.preprocessing import LabelEncoder
X['collision_en'] = LabelEncoder().fit_transform(dummies['collision_type'])
X[['collision_type', 'collision_en']]
X['property_damage'].replace(to_replace='YES', value=1, inplace=True)
X['property_damage'].replace(to_replace='NO', value=0, inplace=True)
X['property_damage'].replace(to_replace='?', value=0, inplace=True)
X['police_report_available'].replace(to_replace='YES', value=1, inplace=True)
X['police_report_available'].replace(to_replace='NO', value=0, inplace=True)
X['police_report_available'].replace(to_replace='?', value=0, inplace=True)
X = X.drop(columns = ['collision_type'])
X = pd.concat([X, df._get_numeric_data()], axis=1)  # joining numeric columns
X = X.drop(columns = ['fraud_reported'])
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8,
random_state=1234)
print('length of X_train and X_test: ', len(X_train), len(X_test))
print('length of y_train and y_test: ', len(y_train), len(y_test))
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.metrics import accuracy_score, recall_score, classification_report,
cohen_kappa_score

from sklearn import metrics

# Baseline Random forest based Model

rfc = RandomForestClassifier(criterion = 'gini', n_estimators=1000, verbose=1,
n_jobs = -1,

                    class_weight = 'balanced', max_features = 'auto')

rfcg = rfc.fit(X_train,y_train) # fit on training data

predictions = rfcg.predict(X_test)

print('Baseline: N_features: ', len(list(X.columns)))

print('Baseline: Accuracy: ', round(accuracy_score(y_test, predictions)*100, 2))

print( 'Cohen Kappa: '+ str(np.round(cohen_kappa_score(y_test,
predictions),3)))

print('Baseline: Recall: ', round(recall_score(y_test, predictions)*100, 2))

print('\n Classification Report:\n', classification_report(y_test,predictions))

from sklearn.metrics import confusion_matrix

import itertools

#Evaluation of Model - Confusion Matrix Plot

def plot_confusion_matrix(cm, classes, title ='Confusion matrix',
normalize=False, cmap = plt.cm.Blues):

    """

    This function prints and plots the confusion matrix.

    Normalization can be applied by setting normalize=True.

    """

    print('Confusion matrix')

    print(cm)
```

```
    fig = plt.figure(figsize=(10,6))
    plt.style.use('fivethirtyeight')
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)
    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
# Compute confusion matrix
cnf_matrix = confusion_matrix(y_test, predictions)
np.set_printoptions(precision=2)
# Plot non-normalized confusion matrix
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=['Fraud
reported_Y','Fraud_reported_N'],
                      title='Random Forest-Confusion matrix')
```

With 72.5% accuracy, we take a closer look at the confusion matrix:

#

- 132 transactions were classified as valid that were actually valid

- 7 transactions were classified as fraud that were actually valid (type 1 error)

- 48 transactions were classified as valid that were fraud (type 2 error)

- 13 transactions were classified as fraud.

#

Err = ((FP+FN)/ (TP+TN+FN+FP) = {(48+7) / (132+7+48+13)}*100 = 0.275

#

So, the algorithm misclassified 27.5% fraudulent transactions. We looked at other measures too like the Cohen Kappa, Recall, and F1 score. In each case, the scores are closer to 1.

In[77]:

Generate a Histogram plot for anomaly detection

df.plot(kind='hist')

plt.show()

TESTING

```
import pandas as pd

from sklearn.ensemble import ExtraTreesRegressor

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.model_selection import train_test_split

import numpy as np

import sklearn.metrics

import pickle

#df = pd.read_csv('./insuranceFraud_Dataset.csv')

df = pd.read_csv('./testing.csv')

df.nunique()

df[['insured_zip']] = df[['insured_zip']].astype(object)

df.auto_year.value_counts()  # check the spread of years to decide on further action.

df['vehicle_age'] = 2018 - df['auto_year'] # Deriving the age of the vehicle based on the year value

df['vehicle_age'].head(10)

bins = [-1, 3, 6, 9, 12, 17, 20, 24]  # Factorize according to the time period of the day.

names = ["past_midnight", "early_morning", "morning", 'fore-noon', 'afternoon', 'evening', 'night']

df['incident_period_of_day'] = pd.cut(df.incident_hour_of_the_day, bins, labels=names).astype(object)

df[['incident_hour_of_the_day', 'incident_period_of_day']].head(20)

# Check on categorical variables:

df.select_dtypes(include=['object']).columns  # checking categorcial columns
```

```
df = df.drop(columns = [
    'policy_number',
    'insured_zip',
    'policy_bind_date',
    'incident_date',
    'incident_location',
    'auto_year',
    'incident_hour_of_the_day'])
unknowns = {}
for i in list(df.columns):
    if (df[i]).dtype == object:
        j = np.sum(df[i] == "?")
        unknowns[i] = j
unknowns = pd.DataFrame.from_dict(unknowns, orient = 'index')
df.collision_type.value_counts()

df.police_report_available.value_counts()
df.select_dtypes(include=['object']).columns  # checking categorcial columns

dummies = pd.get_dummies(df[[
    'policy_state',
    'policy_csl',
    'insured_sex',
    'insured_education_level',
    'insured_occupation',
```

```
    'insured_hobbies',
    'insured_relationship',
    'incident_type',
    'incident_severity',
    'authorities_contacted',
    'incident_state',
    'incident_city',
    'auto_make',
    'auto_model',
    'incident_period_of_day']])
dummies = dummies.join(df[[
    'collision_type',
    'property_damage',
    'police_report_available'
    ]])
X = dummies.iloc[:, 0:]
y = dummies.iloc[:, -1]
print(X)
from sklearn.preprocessing import LabelEncoder
X['collision_en'] = LabelEncoder().fit_transform(dummies['collision_type'])
X[['collision_type', 'collision_en']]
X['property_damage'].replace(to_replace='YES', value=1, inplace=True)
X['property_damage'].replace(to_replace='NO', value=0, inplace=True)
X['property_damage'].replace(to_replace='?', value=0, inplace=True)
X['police_report_available'].replace(to_replace='YES', value=1, inplace=True)
```

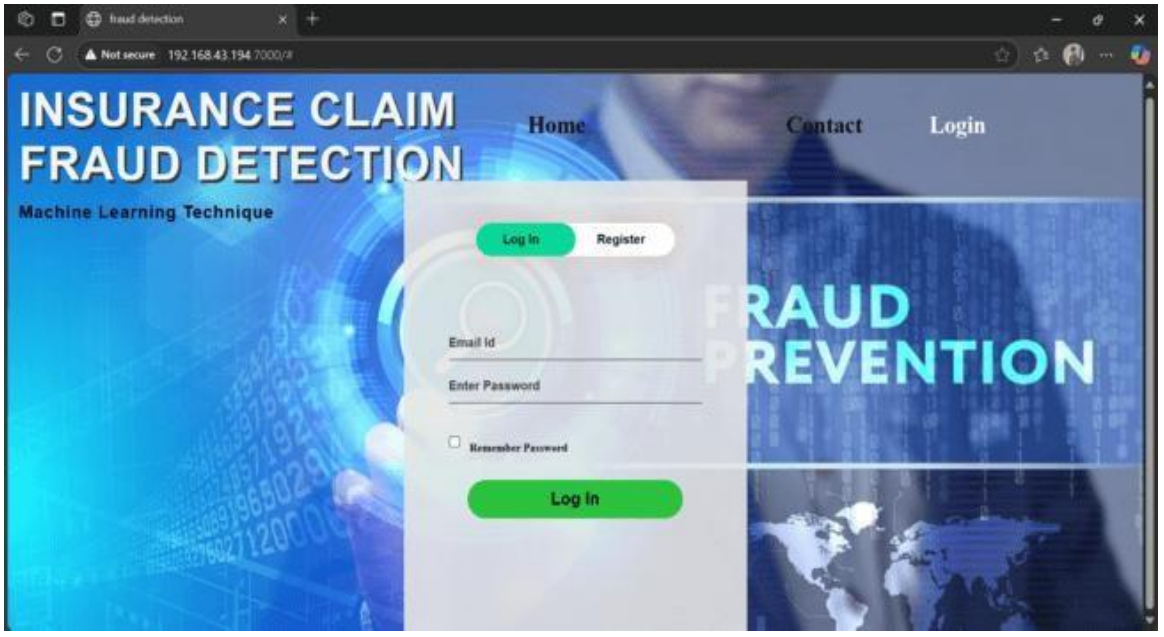
```
X['police_report_available'].replace(to_replace='NO', value=0, inplace=True)
X['police_report_available'].replace(to_replace='?', value=0, inplace=True)
X = X.drop(columns = ['collision_type'])
X = pd.concat([X, df._get_numeric_data()], axis=1) # joining numeric columns
print(X.iloc[0])


model = pickle.load(open('model1.pkl', 'rb'))
array_features = [np.array(X.iloc[0])]
prediction = model.predict(array_features)
print(prediction)
```

APPENDICES - II

SNAPSHOTS

1. LOGIN

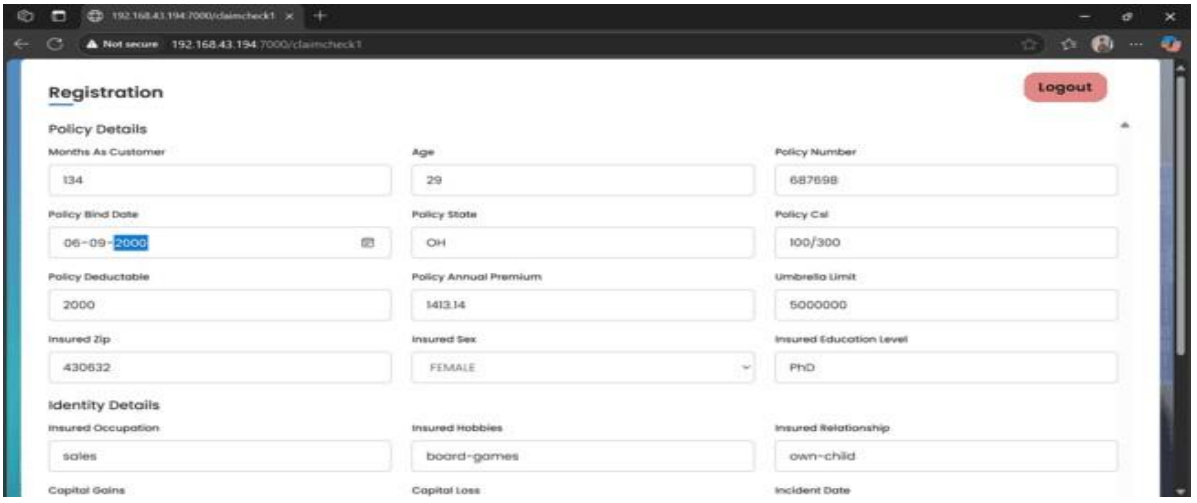


2 .REGISTER PAGE

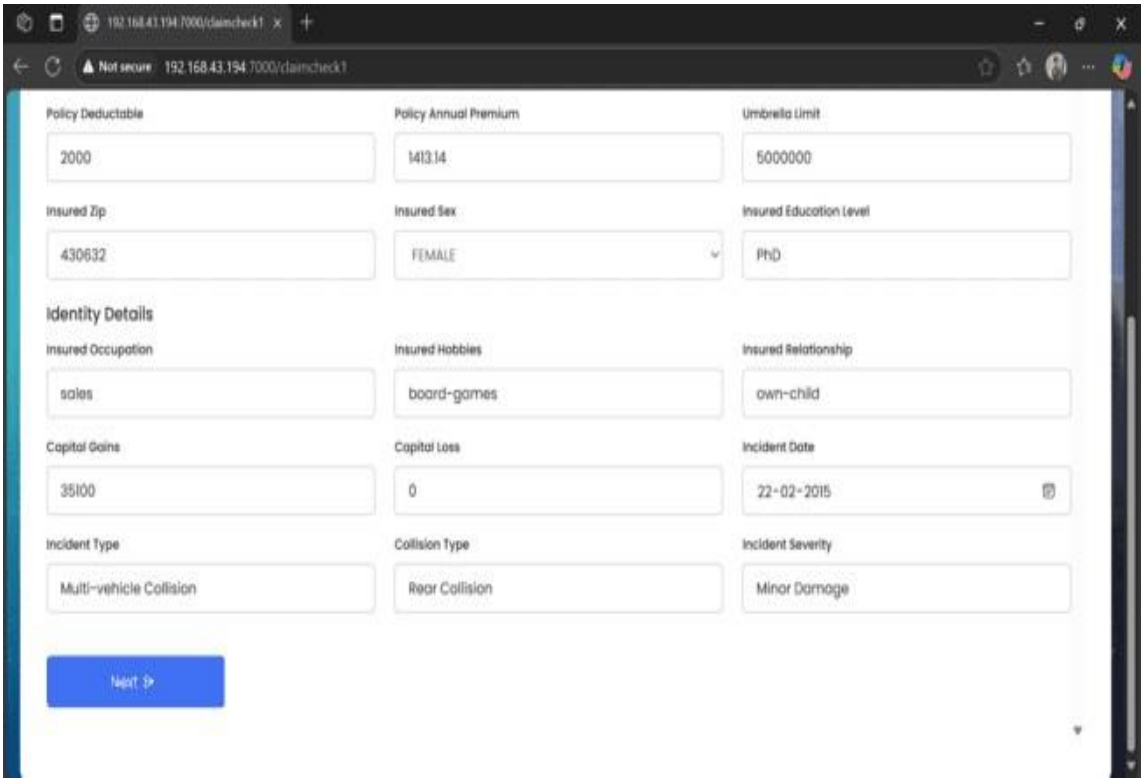
3. HOME PAGE

4 .UPLOAD PAGE

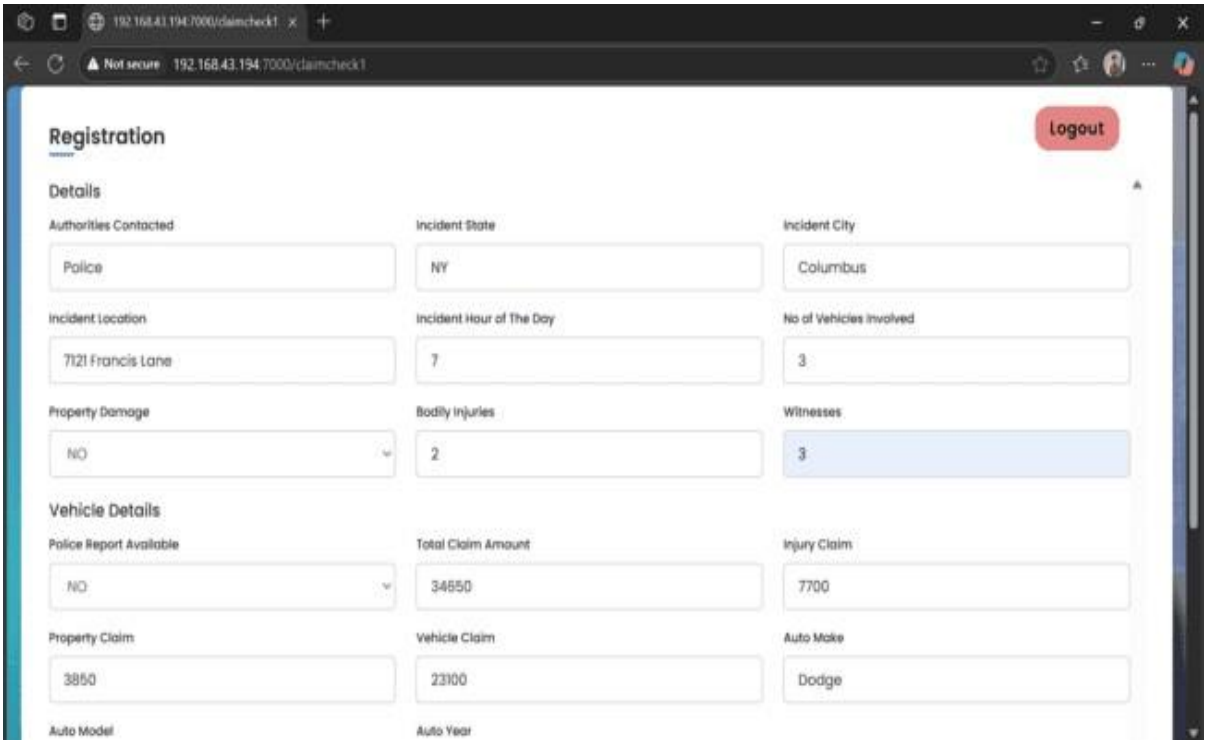
5. INSURANCE FRAUD DETECTION REGISTRATION



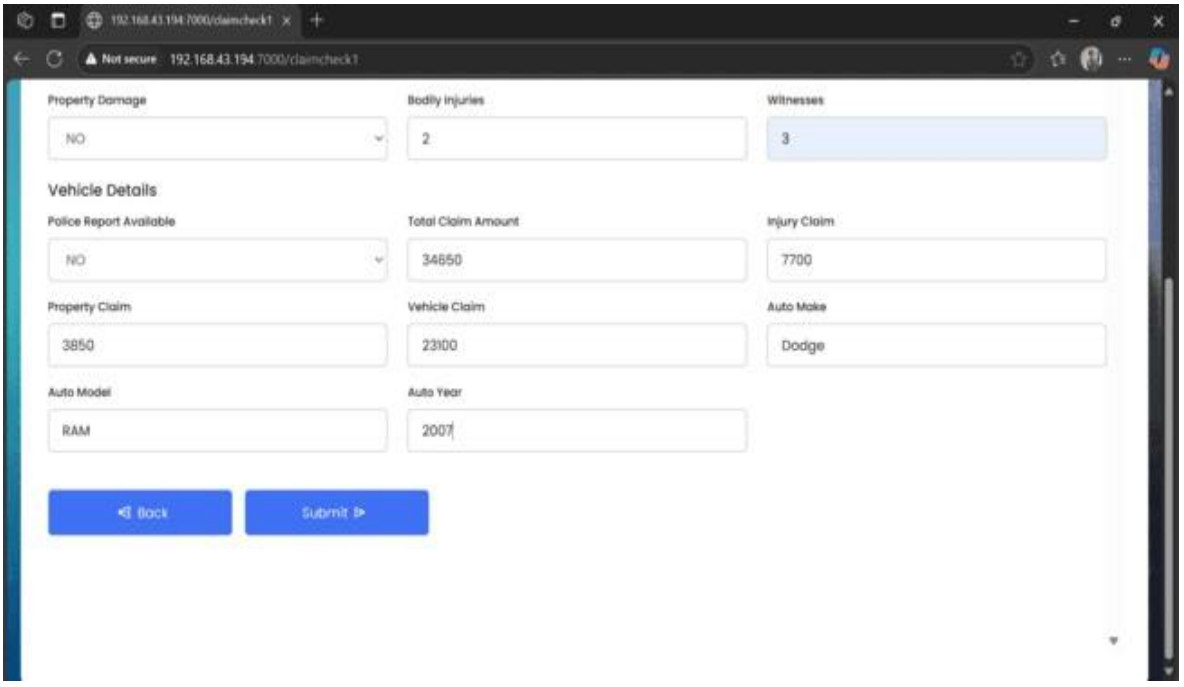
5.1 INSURANCE FRAUD DETECTION REGISTRATION NEXT PAGE



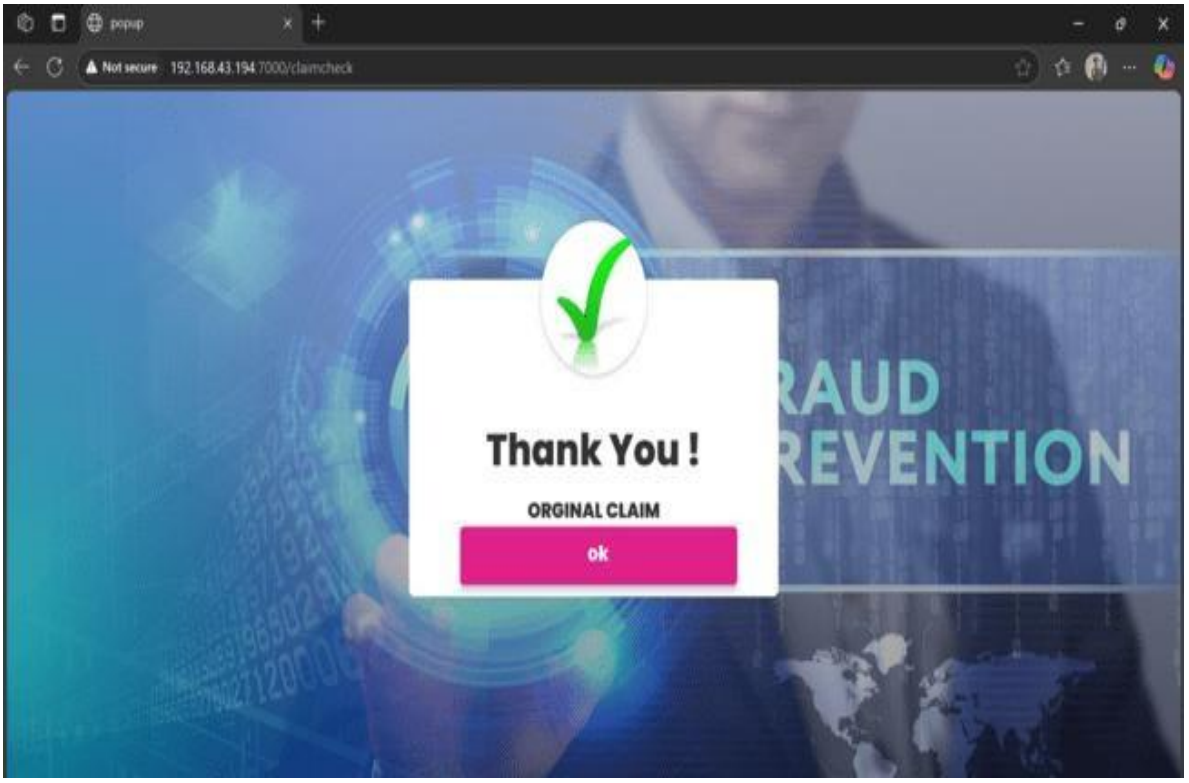
5.2 INSURANCE FRAUD DETECTION REGISTRATION DETAILS PAGE



5.3 INSURANCE FRAUD DETECTION REGISTRATION SUBMIT PAGE



6. RESULT PAGE



7 DATASETS VISUALIZATION PAGE

7.1 DATASETS VISUALIZATION PAGE

REFERENCES

[1] Bhaskar, A., Pande, S., Malik, R., & Khamparia, A. (2021). An intelligent unsupervised technique for fraud detection in health care systems. *Intelligent Decision Technologies*, 15(1), 127–139.

[2] Sathya, M., & Balakumar, B. (2022). Insurance Fraud Detection Using Novel Machine Learning Technique. *International Journal of Intelligent Systems and Applications in Engineering*, 10(3), 374–381.

[3] Rukhsar, L., Bangyal, W. H., Nisar, K., & Nisar, S. (2022). Prediction of insurance fraud detection using machine learning algorithms. *Mehran University Research Journal of Engineering & Technology*, 41(1), 33–40.

[4] Zhou, X., Cheng, S., Zhu, M., Guo, C., Zhou, S., Xu, P., Xue, Z., & Zhang, W. (2018). A state of the art survey of data mining-based fraud detection and credit scoring. In *MATEC Web of Conferences* (Vol. 189, p. 03002). EDP Sciences.

[5] Asgarian, A., Saha, R., Jakubovitz, D., & Peyre, J. (2023). AutoFraudNet: A Multimodal Network to Detect Fraud in the Auto Insurance Industry. *arXiv preprint arXiv:2301.07526*.

[6] Roy, R., & George, K. T. (2017). Detecting insurance claims fraud using machine learning techniques. In *2017 International Conference on Circuit, Power and Computing Technologies (ICCPCT)* (pp. 1–6). IEEE.

[7] Pranavi, P. S., Sheethal, H. D., Kumar, S. S., Kariappa, S., & Swathi, B. H. (2020). Analysis of Vehicle Insurance Data to Detect Fraud using Machine Learning. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, 8(7), 2033–2038.

[8] Yan, C., Li, M., Liu, W., & Qi, M. (2020). Improved adaptive genetic algorithm for the vehicle Insurance Fraud Identification Model based on a BP Neural Network. *Theoretical Computer Science*, 817, 12–23.

[9] Aslam, F., Hunjra, A. I., Ftiti, Z., Louhichi, W., & Shams, T. (2022). Insurance fraud detection: Evidence from artificial intelligence and machine learning. *Research in International Business and Finance*, 62, 101744.

[10] Narayanan, K. L., Ram, C. R., Subramanian, M., Krishnan, R. S., & Robinson, Y. H. (2021). IoT based smart accident detection & insurance claiming system. In *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 306–311). IEEE.

[11] Fernando, N., Kumarage, A., Thiyaganathan, V., Hillary, R., & Abeywardhana, L. (2022). Automated vehicle insurance claims processing using computer vision, natural language processing. In *2022 22nd International Conference on Advances in ICT for Emerging Regions (ICTer)* (pp. 124–129). IEEE.

[12] Patil, N. S., Kamanavalli, S., Hiregoudar, S., Jadhav, S., Kanakraddi, S., & Hiremath, N. D. (2021). Vehicle Insurance Fraud Detection System Using Robotic Process Automation and Machine Learning. In *2021 International Conference on Intelligent Technologies (CONIT)* (pp. 1–5). IEEE.

[13] Hanafy, M. O., & Ming, R. (2021). Using machine learning models to compare various resampling methods in predicting insurance fraud. *Journal of Theoretical and Applied Information Technology*, 99(12), 2819–2833.