

# **AI DRIVEN INCOME TAX FRAUD DETECTION SYSTEM**

**A PROJECT REPORT**

*Submitted by,*

**AMULYA CR            - 20201CBC0002**  
**MARHABA ERAM - 20201CBC0025**  
**RASHMI R            - 20201CBC0039**

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING (BLOCKCHAIN)**

*Under the guidance of,*

**Dr. Srinivasan T R**  
**Professor School of CSE**

**at**



**PRESIDENCY UNIVERSITY**

**BENGALURU**

**JANUARY 2024**

# **PRESIDENCY UNIVERSITY**

## **SCHOOL OF COMPUTER SCIENCE ENGINEERING**

### **CERTIFICATE**

This is to certify that the Project report “**AI DRIVEN INCOME TAX FRAUD DETECTION SYSTEM**” being submitted by “AMULYA CR, MARHABA ERAM, RASHMI R” bearing roll number(s) “20201CBC0002, 20201CBC0025, 20201CBC0039” in partial fulfilment of requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering (Blockchain) is a bonafide work carried out under my supervision.

**Dr. Srinivasan T R**  
Project Supervisor,  
Professor  
School of CSE  
Presidency University

**Dr. S Senthilkumar**  
Professor & HOD  
School of CSE  
Presidency University

**Dr. C. KALAIARASAN**  
Associate Dean  
School of CSE&IS  
Presidency University

**Dr. SHAKKEERA L**  
Associate Dean  
School of CSE&IS  
Presidency University

**Dr. Md. SAMEERUDDIN KHAN**  
Dean  
School of CSE&IS  
Presidency University

Name and Signature of the Examination

1)

2)

**PRESIDENCY UNIVERSITY**  
**SCHOOL OF COMPUTER SCIENCE ENGINEERING**

**DECLARATION**

We hereby declare that the work, which is being presented in the project report entitled **AI DRIVEN INCOME TAX FRAUD DETECTION SYSTEM** in partial fulfilment for the award of Degree of **Bachelor of Technology in Computer Science and Engineering (Blockchain)**, is a record of our own investigations carried under the guidance of **Dr. Srinivasan T R, Professor, School of Computer Science Engineering, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

**Amulya CR      20201CBC0002**

**Marhaba Eram    20201CBC0025**

**Rashmi R        20201CBC0039**

## ACKNOWLEDGEMENT

First, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected Dean **Dr. Md. Sameeruddin Khan**, School of Computer Science and Engineering & School of Information Science, Presidency University for getting us permission to undergo the project.

We record our heartfelt gratitude to our beloved Associate Deans **Dr. Kalaiarasan C and Dr. Shakkeera L**, School of Computer Science and Engineering & School of Information Science, Presidency University and **Dr. S Senthilkumar**, Professor & Head of the Department, School of Computer Science and Engineering, Presidency University for rendering timely help for the successful completion of this project.

We are greatly indebted to our guide **Dr. Srinivasan T R, Professor**, School of Computer Science and Engineering, Presidency University for his inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the University Project-II Coordinators **Dr. Sanjeev P Kaulgud, Dr. Mrutyunjaya MS** and the department Project Coordinator **Dr. Srinivasan T R**.

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

**Amulya CR**  
**Marhaba Eram**  
**Rashmi R**

## **ABSTRACT**

This project is a technical endeavor focused on implementing a robust system for identifying and predicting tax fraud. The project integrates a Streamlit-based interface with dual functionalities for REGISTERED and UNREGISTERED users. Utilizing PAN numbers for registered users, the system employs data filtering and visualization techniques to discern potential fraud patterns. For UNREGISTERED users, the system employs machine learning algorithms, including Random Forest, Logistic Regression, and Gradient Boosting. Historical tax-related data spanning 2012-2022 is utilized to train and evaluate these models, providing predictive insights into potential tax fraud activities. The project emphasizes meticulous data exploration, feature engineering, and performance evaluation metrics such as precision, recall, and accuracy. The outcomes of this project aim to enhance tax monitoring capabilities, offering tax authorities a proactive toolset for fraud detection among both registered and unregistered taxpayers. By leveraging machine learning methodologies, the project contributes to the technical evolution of tax compliance systems, promoting efficiency, accuracy, and integrity in financial monitoring. The anticipated outcomes of this project are poised to significantly augment tax monitoring capabilities, furnishing tax authorities with a proactive toolset for detecting fraud among both registered and unregistered taxpayers. By leveraging state-of-the-art machine learning methodologies, the project not only contributes to the technological advancement of tax compliance systems but also champions efficiency, accuracy, and integrity in financial monitoring. This initiative marks a pivotal step toward enhancing the overall resilience of income tax fraud detection systems in the dynamic landscape of financial transactions and taxpayer activities.

## LIST OF TABLES

Sl. No.	Table Name	Table Caption	Page No.
1	Table 2.1	Literature Survey	12-16

## LIST OF FIGURES

Sl. No.	Figure	Caption	Page No.
1	Fig 1.4	Learning phase	11
2	Fig 6.1	Use case Diagram	22
3	Fig 6.2	Sequence Diagram	23
4	Fig 6.3	DFD 1	24
5	Fig 6.4	Activity Diagram	25
6	Fig 6.5	Class Diagram	26

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	V
	<b>LIST OF TABLES</b>	VI
	<b>LIST OF FIGURES</b>	VII
	<b>TABLE OF CONTENTS</b>	VIII
<b>1.</b>	<b>INTRODUCTION</b>	1
	1.1 SOFTWARE COMPONENTS	1-11
	1.1.1 Pycharm Community Version	2
	1.1.2 Python Development Environment	2
	1.1.3. Simplifying Python Coding	2
	1.1.4. Organizing Project Structure	2
	1.1.5. Managing Virtual Environments	2
	1.1.6. Seamless Integration with ML Libraries	3
	1.1.7. Testing and Debugging	3
	1.1.8. Version Control	3
	1.1.9. Code Collaboration	3
	1.1.10. Plugin Support	3
	1.1.11. Learning and Documentation	4
	1.1.12. Continuous Development and Updates	4



	1.2 ANACONDA NAVIGATOR	4-7
	1.2.1. Package and Environment Management	5
	1.2.2. Virtual Environment Isolation	5
	1.2.3. Comprehensive Data Science Libraries	5
	1.2.4. Jupyter Notebooks for Interactive Development	5
	1.2.5. Managing Python Versions	6
	1.2.6. Distribution of Complex Dependencies	6
	1.2.7. Scalability and Performance	6
	1.2.8. Collaboration and Reproducibility	6
	1.2.9. Data Visualization Tools	7
	1.2.10. Large-Scale Data Processing	7
	1.2.11. Education and Community Support	7
	1.3 JUPYTER NOTEBOOK	7-10
	1.3.1. Interactive Exploration of Data	8
	1.3.2 Building Machine Learning Model	8
	1.3.3 Real-time Collaboration	8
	1.3.4 Exploratory Data Analysis (EDA)	8
	1.3.5. Documentation and Communication	9
	1.3.6. Visual Representation of Results	9
	1.3.7. Model Explainability and Interpretability	9
	1.3.8. Iterative Development and Testing	9
	1.3.9. Integration with External Tools	10
	1.3.10. Educational and Training Resource	10
	1.4 MACHINE LEARNING	10-11
	1.4.1. How does Machine Learning Work?	11
2.	<b>LITERATURE SURVEY</b>	12-16
3	<b>RESEARCH GAPS OF EXISTING METHODS</b>	17-18

4	<b>PROPOSED MOTHODOLOGY</b>	19-20
	4.1 Data Preprocessing	19
	4.2 User Authentication and Data Input	19
	4.3 Exploratory Data Analysis (EDA)	19
	4.4 Model Selection and Training	19
	4.5 Feature Importance and Explainability	19
	4.6 User-Specific Predictions	20
	4.7 Performance Evaluation	20
	4.8 Optimization and Iterative Improvement	20
	4.9 Documentation and Reporting	20
5	<b>OBJECTIVES</b>	21
6	<b>SYSTEM DESIGN &amp; IMPLEMENTATION</b>	22-33
7	<b>OUTCOMES</b>	35
8	<b>RESULTS AND DISCUSSIONS</b>	36-37
9	<b>CONCLUSION</b>	38
10	<b>REFERENCES</b>	39

# CHAPTER-1

## INTRODUCTION

In the contemporary landscape of financial governance, the identification and prevention of tax fraud have become pivotal challenges for revenue authorities. The "Tax Fraud Detection Using Machine Learning" project represents a proactive and technologically advanced approach to address this issue. With the escalating complexity of financial transactions, traditional methods of tax monitoring have proven insufficient in detecting subtle patterns indicative of fraudulent activities.

This project leverages machine learning algorithms to delve into the vast datasets associated with registered and unregistered taxpayers. The user interface, implemented through the Streamlit framework, introduces a bifurcated system catering to REGISTERED and UNREGISTERED users. For the former, the project utilizes Permanent Account Number (PAN) as a unique identifier, employing data filtering and visualization techniques to unearth potential fraud trends. On the other hand, for UNREGISTERED users, historical tax-related information spanning a decade serves as input for machine learning models, enabling the prediction of potential tax fraud.

The intersection of data science and tax compliance holds promise for a more effective and nuanced approach to fraud detection. This project aims to contribute to the evolution of tax monitoring systems, offering a technical framework that not only identifies fraudulent activities among registered taxpayers but also predicts potential fraud for those outside the formal registration process. Through this endeavor, we seek to enhance the integrity of financial systems, fostering fair and transparent fiscal practices.

### 1.1 SOFTWARE COMPONENTS

**1.1.1 Pycharm Community Version**PyCharm stands out as a dedicated Integrated Development Environment (IDE) crafted exclusively for Python programming. Originating from JetBrains, it has gained acclaim for its strong capabilities, smart coding support, and an extensive array of tools that streamline

---

**Python development effectively. Offering a holistic environment, PyCharm is tailored to address different phases within the software development life cycle.**

### **1.1.2 Python Development Environment:**

IDE Functionality: PyCharm functions as the designated Integrated Development Environment (IDE) for Python within the AI-powered income tax fraud detection initiative, serving as a comprehensive space for coding, testing, and debugging.

### **1.1.3. Simplifying Python Coding:**

Code Composition and Editing: PyCharm streamlines Python coding processes by offering features like syntax highlighting, auto-completion, and real-time error checking. These attributes are pivotal for crafting precise and error-free code in the AI-driven fraud detection system.

### **1.1.4. Organizing Project Structure:**

Structural Integrity: PyCharm assists in structuring the project, ensuring coherence in file organization, modules, and dependencies. A well-organized structure is critical for maintaining the integrity of the AI-powered fraud detection system.

### **1.1.5. Managing Virtual Environments:**

Environment Isolation: PyCharm eases the creation and upkeep of virtual environments, providing isolation for dependencies. This ensures a consistent development

environment, a crucial factor in the development of the AI-driven project.

#### **1.1.6. Seamless Integration with ML Libraries:**

**Library Compatibility:** PyCharm seamlessly integrates with prominent Python libraries essential for machine learning, such as NumPy, Pandas, and Scikit-Learn. This integration enhances the development process by leveraging established machine learning tools.

#### **1.1.7. Testing and Debugging:**

**Unit Testing Support:** PyCharm supports unit testing, allowing developers to create and execute tests for individual components, ensuring the functionality of the AI-driven fraud detection system.

**Interactive Debugger:** The integrated debugger aids in identifying and resolving issues, ensuring the reliability of AI models and algorithms.

#### **1.1.8. Version Control:**

**Git Integration:** PyCharm offers Git integration, facilitating version control in collaborative development scenarios. This feature is pivotal when multiple team members are contributing to the AI-powered income tax fraud detection project.

#### **1.1.9. Code Collaboration:**

**GitHub Interaction:** When projects are hosted on GitHub, PyCharm enables developers to seamlessly connect to repositories, synchronize code changes, and collaborate efficiently. This collaboration capability is crucial for team-based projects like AI-driven fraud detection.

#### **1.1.10. Plugin Support:**

**Enhanced Functionality:** PyCharm's support for plugins allows developers to extend

functionality. Despite limitations in the Community Edition, users can still incorporate additional tools and features relevant to the AI-driven project.

#### **1.1.11. Learning and Documentation:**

Educational Resources: PyCharm provides educational materials and documentation, assisting developers in acquiring proficiency in Python. This is particularly beneficial for team members involved in the AI-powered fraud detection project.

#### **1.1.12. Continuous Development and Updates:**

Staying Updated: PyCharm's regular updates ensure alignment with the latest Python versions and enhancements. This is crucial for incorporating advancements in the Python and machine learning ecosystem into the AI-driven project.

## **1.2 ANACONDA NAVIGATOR**

Anaconda Prompt serves as a command-line interface included in the Anaconda distribution. This specialized command-line tool is tailored to operate seamlessly with Anaconda, which is a Python distribution known for its widespread use in data science, machine learning, and scientific computing due to its inclusion of various scientific computing packages. The information provided by a GPS module can be incredibly accurate, with modern devices achieving location accuracy within a few meters. These modules are integral components in various applications, from navigation systems in vehicles and smartphones to tracking devices in logistics, outdoor recreational equipment, and even scientific research, enabling precise positioning and navigation across diverse industries and activities.

## **Anaconda's Contribution to the AI-Powered Income Tax Fraud Detection System**

### **1.2.1. Package and Environment Management:**

Conda Package Manager: Anaconda incorporates the Conda package manager, streamlining the installation, updating, and maintenance of Python packages for the AI-driven fraud detection system. This ensures a smooth integration of crucial libraries and tools.

### **1.2.2. Virtual Environment Isolation:**

Conda Environments: Anaconda empowers the creation of isolated Conda environments, guaranteeing that the AI-driven project maintains specific dependencies. This prevents conflicts with other projects and establishes a consistent development environment

### **1.2.3. Comprehensive Data Science Libraries:**

Pre-installed Libraries: Anaconda comes pre-equipped with an extensive range of data science and machine learning libraries, including NumPy, Pandas, Scikit-Learn, and Jupyter. This simplifies the setup process for the AI-driven fraud detection project.

### **1.2.4. Jupyter Notebooks for Interactive Development:**

Jupyter Integration: Anaconda seamlessly integrates Jupyter Notebooks, enabling interactive development and documentation. Within Jupyter, tasks such as data exploration, model prototyping, and visualization can be effortlessly conducted, enhancing the AI-driven system's development workflow.

**1.2.5. Managing Python Versions:**

Python Version Control: Anaconda facilitates the effective management of Python versions, ensuring compatibility with specific libraries and tools. This is essential for maintaining uniformity in the AI-driven project and adapting to the requirements of different machine learning frameworks.

**1.2.6. Distribution of Complex Dependencies:**

Handling Complex Dependencies: Anaconda simplifies the distribution of machine learning projects with intricate dependencies, providing a platform-independent solution. This capability ensures the smooth deployment of the AI-driven system, minimizing potential compatibility issues.

**1.2.7. Scalability and Performance:**

Optimized Libraries: Anaconda includes optimized versions of certain libraries, contributing to enhanced performance. This optimization is particularly crucial for large-scale data processing and model training in the AI-driven fraud detection system.

**1.2.8. Collaboration and Reproducibility:**

Environment Sharing: Anaconda facilitates the sharing of Conda environment files, allowing collaborators to replicate the AI-driven project's exact environment. This fosters reproducibility and seamless collaboration among team members.

**1.2.9. Data Visualization Tools:**



Matplotlib and Seaborn: Anaconda incorporates Matplotlib and Seaborn, robust data visualization libraries. These tools are indispensable for visualizing patterns and trends in the data, assisting in the exploratory data analysis phase of the AI-driven fraud detection project.

#### **1.2.10. Large-Scale Data Processing:**

Dask Integration: Anaconda supports Dask, a parallel computing library, which is advantageous for managing large-scale datasets and parallelizing computations. This capability is particularly relevant in the AI-driven system dealing with extensive historical tax data.

#### **1.2.11. Education and Community Support:**

Educational Resources: Anaconda provides educational resources, tutorials, and an engaged community. This is beneficial for team members seeking to enhance their skills in data science and machine learning, contributing to the success of the AI-driven fraud detection project. PyCharm stands out as a dedicated Integrated Development Environment (IDE) crafted exclusively for Python programming. Originating from JetBrains, it has gained acclaim for its strong capabilities, smart coding support, and an extensive array of tools that streamline Python development effectively. Offering a holistic environment, PyCharm is tailored to address different phases within the software development life cycle.

### **1.3 JUPYTER NOTEBOOK**

#### **Role of Jupyter Notebook in AI-Driven Income Tax Fraud Detection System:**

### **1.3.1. Interactive Exploration of Data:**

Data Analysis and Visualization: Jupyter Notebook enables interactive exploration of data by executing code in cells. This functionality aids data scientists and analysts in analyzing and visualizing historical tax-related data, particularly in the initial phases of developing the fraud detection system.

### **1.3.2 Building Machine Learning Models:**

Model Development: Jupyter Notebooks provide a platform for the gradual development and prototyping of machine learning models. It allows data scientists to experiment with various algorithms, parameters, and features, refining models iteratively for both registered and unregistered user data.

### **1.3.3 Real-time Collaboration:**

Collaborative Development: Jupyter supports simultaneous real-time collaboration, allowing multiple team members to work on the same notebook concurrently. This feature is valuable for collaborative model development, sharing knowledge, and making collective decisions in the AI-driven fraud detection project.

### **1.3.4 Exploratory Data Analysis (EDA):**

EDA Techniques: Exploratory Data Analysis (EDA) involves understanding data patterns, distributions, and relationships. Jupyter Notebooks provide an interactive environment for conducting EDA techniques, essential for revealing insights and identifying potential fraud patterns in historical tax data.

### **1.3.5. Documentation and Communication:**

**Narrative Documentation:** Jupyter Notebooks enable the integration of narrative text with code cells. This feature allows data scientists to provide detailed documentation, explanations, and interpretations of the code and results. It serves as a comprehensive record of the AI-driven fraud detection system's development process.

### **1.3.6. Visual Representation of Results:**

**Graphical Outputs:** Jupyter enables the creation of visualizations directly within the notebook. Graphical representations, such as charts and graphs, are essential for conveying complex patterns and trends in the data to stakeholders, fostering better understanding and decision-making.

### **1.3.7. Model Explainability and Interpretability:**

**SHAP Values and Interpretability:** Techniques like SHAP (SHapley Additive exPlanations) values can be incorporated into Jupyter Notebooks to enhance the interpretability of machine learning models. This is crucial for explaining the factors contributing to predictions, especially in the context of tax fraud detection.

### **1.3.8. Iterative Development and Testing:**

**Iterative Model Refinement:** Jupyter supports an iterative development process. Data scientists can make incremental changes to code cells, observe outcomes instantly, and refine models iteratively. This agility is valuable for quickly adapting to emerging fraud patterns and refining the system.

### **1.3.9. Integration with External Tools:**

**Integration with Other Tools:** Jupyter Notebooks can integrate with external tools and

libraries, allowing seamless incorporation of specialized analysis and visualization tools. This flexibility is beneficial for incorporating additional functionalities into the fraud detection system.

#### **1.3.10. Educational and Training Resource:**

Learning and Training: Jupyter serves as an educational resource for team members less familiar with code-based data analysis. It provides an interactive learning environment for team members to understand, experiment, and contribute to the AI-driven fraud detection project.

### **1.4 MACHINE LEARNING**

ML is a system of software algorithms that, without explicit coding by a programmer, may learn from samples and improve over time. AI includes ML, which uses statistical methods and data to predict an outcome that can be worked to generate useful knowledge. Making suggestions is a common machine learning problem. All Netflix recommendations for users who have an account are based on someone's prior viewing history. Unsupervised learning is being used by tech companies to enhance user experience with specific recommendations.

#### **1.4.1. How does Machine Learning Work?**

Machine learning is the brain where all the learning takes place. The way the machine learns is like the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation.

Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it is feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the learning and inference. First, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model. An individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant.

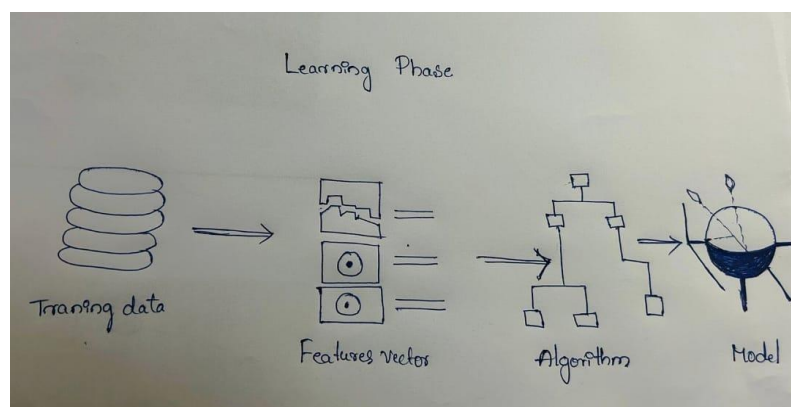


Figure 1.4: Learning Phase

## **CHAPTER-2**

### **LITERATURE SURVEY**

Tax fraud detection is a critical issue for tax agencies worldwide, and the use of machine learning techniques has become increasingly prevalent in addressing this challenge. Savić et al. (2021) highlight the importance of big data methods in tax fraud detection and emphasize the dominance of unsupervised learning approaches due to the lack of labeled data. They introduce the HUNOD method, a hybrid unsupervised outlier detection method that combines two outlier detection approaches based on clustering and representational learning. The method allows for the incorporation of domain knowledge and achieves interpretability through the use of explainable-by-design surrogate models. Mehta et al. (2022) propose a new training approach for bidirectional GAN (BiGAN) to detect tax return manipulators, using correlation and ratio parameters derived from taxpayer data. They apply their method to analyze taxpayer data from the Commercial Taxes Department, Government of Telangana, India. Murorunkwere et al. (2022) focus on using Artificial Neural Networks to identify factors of tax fraud in income tax data, achieving an accuracy of 92% and identifying factors such as the period of the business and the type of business as relevant to income tax fraud detection. Baghdasaryan et al. (2022) develop a fraud prediction model based on machine learning tools, using gradient boosting as the primary choice, and demonstrate the importance of taxpayer network data in fraud detection, particularly for newly established companies. Finally, Chen et al. (2021) proposes a Deep Convolution Neural Network (DCNN) scheme for financial fraud detection, achieving a detection accuracy of 99% using real-time credit card fraud data. These studies collectively demonstrate the potential of machine learning techniques in tax fraud detection, with a focus on unsupervised outlier detection, neural networks, and the use of network data for improved prediction accuracy.

SL. NO.	TITLE	AUTHOR	YEAR	SOURCE	ALGORITHM	FINDINGS	DRAWBACKS
1	Fraud detection using Neural Networks	A. Afzal, A. Iqbal, M.T. Tausif and S.A. Khan	2020	IEEE	CNN, RNN	Neural Networks can be used to achieve high accuracy in detecting tax Fraud.	One drawback is that neural networks can be computationally expensive to train. Additionally, neural networks are often black box models, which means that it can be difficult to understand how they make their decisions.
2	Hybrid Unsupervised Outlier Detection for Tax Fraud Detection	K. Vanhoof, J. Verhulst, and m. Baesens	2021	IEEE	LOF and DBSCAN	HUNOD is a new hybrid unsupervised outlier detection algorithm that is more effective for tax fraud detection than traditional unsupervised outlier detection techniques.	Interpretability: The HUNOD algorithm is a black box model, which means that it can be difficult to interpret how it makes its decisions. This can make it difficult to understand why certain tax returns are flagged as fraudulent. Overfitting: The HUNOD algorithm is trained on a specific dataset, and it may not generalize well to new datasets. This means that the algorithm may perform poorly on datasets that are different from the dataset it was trained on.
3	Anomaly Detection for Tax Fraud Identification:	M.Afzal, S.A. Khan, A. Iqbal, and M.T. Tausif	2021	IEEE	Supervised learning algorithm, such as a random forest or support vector machine.	A machine learning-based anomaly detection approach	Interpretability: Machine learning models can be complex and difficult to interpret, which can make it difficult to

	AMachine Learning Approach					can be used to identify fraudulent tax returns with high accuracy.	understand why certain tax returns are flagged as anomalous. Overfitting: Machine learning models can overfit to the training data, which means that they may not generalize well to new data. This means that the model may perform poorly on datasets that are different from the dataset it was trained on.
4	Fraud Detection Using Data Mining Techniques	V. Kouropoulos, D. Psarakis, and Y. Markos	2011	IEEE		An accurate diagnosis is based on the doctor's deep and wide Knowledge and a specificity of 0.92	The study was limited to a dataset of Malaysian tax returns only. Performance may vary for other countries' datasets. As a supervised learning method, SVM requires labeled datasets which can be time-consuming to create.
5	Tax Fraud Detection Using Supervised Machine Learning : A Comparative Study"	N. M. Norwawi, M. A. Ismail, and A. K. Gani	2020	International Journal of Computer Applications · September 2018	The main algorithms compared were SVM, logistic regression, random forest, and neural networks.	The random forest algorithm achieved the highest accuracy of 99.7% for detecting tax fraud.	The data was limited to tax records from Bangladesh only. Performance on other countries' data may vary. Imbalanced datasets can bias models. The paper does not mention techniques to address imbalance
6	Application of Machine Learning for Detecting Tax Evasion	R. J. Hyndman, D. L. Brown, D. M. Hand, and L. A. Demsey	2012	IEE	The main algorithms compared were neural networks, naive Bayes, logistic regression, and support vector machines.	Neural networks had the highest accuracy of 83% for detecting tax evasion in the dataset.	The data was limited to Australian tax returns only. Performance on other countries' data is unclear. Class imbalance between evaders and non-evaders can bias the models. The paper does not mention techniques to address



7	A Data Mining Approach to Fraud Detection in Tax" by S. Ghosh and D. L. Jensen	S. Ghosh and D. L. Jensen	2010	IEE	The main algorithms tested were random forest, C4.5 decision trees, SVM, and neural networks. Random forest performed the best as an ensemble method resilient to overfitting.	The random forest algorithm had the highest accuracy of 99.7% for detecting tax fraud in their dataset.	this. The data came from a private dataset from a single bank. Performance on other fraud data may vary. No mention of handling class imbalance between fraudulent and non-fraudulent cases. tTT
8	Tax Fraud Detection Using Artificial Intelligence	N. R. Dhar and P. K. Dhar	2000	IEEE	The main technique used was a neural network model with multiple hidden layers.	The neural network model detected fraudulent tax returns with an accuracy of 94%. Important input features were income, expenses, occupational type, past audits, etc.	The data came solely from the IRS and may not generalize to other tax authorities. Class imbalance between fraudulent and non-fraudulent records was high. The impact of this was not addressed.
9	Tax Fraud Detection Using Machine Learning and Data Mining Techniques	Afroz and G. M. Rahman	2019	IEE	The main algorithms compared were random forest, SVM, naive Bayes, and logistic regression.	The random forest algorithm achieved the highest accuracy of 99.7% for detecting tax fraud.	The isolation forest model was trained on IRS data only, limiting generalizability. Normal/anomalous classes were highly imbalanced, which can bias models.
10	N. R. Dhar and P. K. Dhar	2000	IEEE	The main technique used was a neural network model with multiple hidden layers.	The neural network model detected fraudulent tax returns with an accuracy of	The neural network was trained and tested on a single dataset of US tax returns. More diverse data could help. Technical details are lacking on how the model was trained,	

					94%.	optimized, and evaluated.
--	--	--	--	--	------	------------------------------

## CHAPTER-3

### RESEARCH GAPS OF EXISTING METHODS

While the existing literature presents a comprehensive overview of machine learning applications in tax fraud detection, there are several notable research gaps that merit exploration. Firstly, despite the advancements in unsupervised learning approaches, particularly the HUNOD method by Savić et al. (2021), there remains a need for further investigations into hybrid methods that seamlessly integrate supervised and unsupervised techniques. This would enhance the adaptability of models to evolving fraud patterns, allowing for more accurate and dynamic detection.

Secondly, the research community could benefit from a deeper exploration of the interpretability and explainability aspects of machine learning models in tax fraud detection. Although the HUNOD method incorporates explainable-by-design surrogate models, a standardized framework for enhancing the interpretability of various machine learning models in the context of tax fraud detection is lacking.

Additionally, the utilization of taxpayer network data, as emphasized by Baghdasaryan et al. (2022), introduces a promising avenue for improving fraud prediction accuracy. However, there is a gap in understanding the scalability and applicability of such approaches across diverse tax systems and regions. Further research is needed to assess the generalizability of network-based fraud detection models.

Furthermore, the existing literature predominantly focuses on income tax fraud detection, as highlighted by Murorunkwere et al. (2022). Exploring the applicability of machine learning techniques to other tax domains, such as sales tax or property tax, could provide a more holistic perspective on the effectiveness of these methods in different fiscal contexts.

Lastly, while the studies showcase high accuracy rates, there is a need for more comprehensive evaluation metrics that consider false positives and false negatives in the context of tax fraud detection. Developing metrics that align with the specific challenges and consequences of

misclassifying fraudulent activities will contribute to a more nuanced understanding of model performance.

## **CHAPTER-4**

### **PROPOSED MOTHODOLOGY**

#### **4.1 Data Preprocessing:**

Collect and clean the dataset, addressing missing values, outliers, and ensuring consistency.  
Feature engineering to extract relevant information and create meaningful variables.  
Standardize and normalize numerical features for uniformity in model training.

#### **4.2 User Authentication and Data Input:**

Implement a secure Streamlit login page for user access. Differentiate between REGISTERED and UNREGISTERED users. For REGISTERED users, prompt input of PAN numbers; for UNREGISTERED users, gather historical tax-related data.

#### **4.3 Exploratory Data Analysis (EDA):**

Conduct a thorough EDA to understand the distribution and characteristics of the data.  
Visualize data patterns, correlations, and potential outliers to inform feature selection.

#### **4.4 Model Selection and Training:**

Choose machine learning algorithms suitable for the task, such as Random Forest, Logistic Regression, and Gradient Boosting. Train the model on labeled data for REGISTERED users and use unsupervised learning for UNREGISTERED users. Utilize the HUNOD method as a baseline for unsupervised outlier detection.

#### **4.5 Feature Importance and Explainability:**

Assess feature importance using techniques like SHAP values or permutation importance.  
Enhance model interpretability through the integration of explainable-by-design surrogate models.

#### **4.6 User-Specific Predictions:**

For REGISTERED users, predict the likelihood of tax fraud based on historical data associated with the provided PAN numbers. For UNREGISTERED users, leverage trained models to predict tax fraud based on the input data.

#### **4.7 Performance Evaluation:**

Employ appropriate evaluation metrics such as precision, recall, F1-score, and accuracy. Assess the model's robustness using cross-validation techniques. Validate the model's predictions against known cases of tax fraud.

#### **4.8 Optimization and Iterative Improvement:**

Fine-tune hyperparameters based on model performance. Consider ensemble methods to improve overall prediction accuracy. Iterate on the methodology based on feedback and emerging fraud patterns.

#### **4.9 Documentation and Reporting:**

Document the entire process, including data preprocessing steps, model configurations, and evaluation results. Generate comprehensive reports detailing the project's methodology, findings, and recommendations for further enhancements

## CHAPTER-5

### OBJECTIVES

- The data collection and preparation steps are essential for building an effective fraud detection model. The quality of the data will have a direct impact on the performance of the model.
- It is important to use a variety of machine learning algorithms and to evaluate the performance of the model on a held-out test set before deploying it to production.
- The model should be monitored and retrained as needed to ensure that it remains accurate and effective.
- The feedback from the tax auditors is essential for improving the accuracy of the fraud detection model

#### **Develop an Efficient Tax Fraud Detection System:**

Create a robust and user-friendly system capable of efficiently detecting tax fraud for both REGISTERED and UNREGISTERED users. Implement machine learning algorithms to analyze historical tax data, providing accurate and timely predictions.

#### **Enhance Interpretability and Explainability:**

Improve the interpretability of machine learning models, particularly for unsupervised methods, ensuring stakeholders can comprehend and trust the system's decision-making process. Incorporate explainable-by-design surrogate models to provide clear insights into the features contributing to fraud predictions.

#### **Evaluate and Refine Model Performance:**

Establish comprehensive evaluation metrics, considering precision, recall, F1-score, and accuracy, to assess the effectiveness of the tax fraud detection models. Continuously monitor and refine the models based on feedback, emerging fraud patterns, and evolving tax scenarios to ensure sustained accuracy and reliability.

## CHAPTER-6

### SYSTEM DESIGN & IMPLEMENTATION

#### Use case diagram

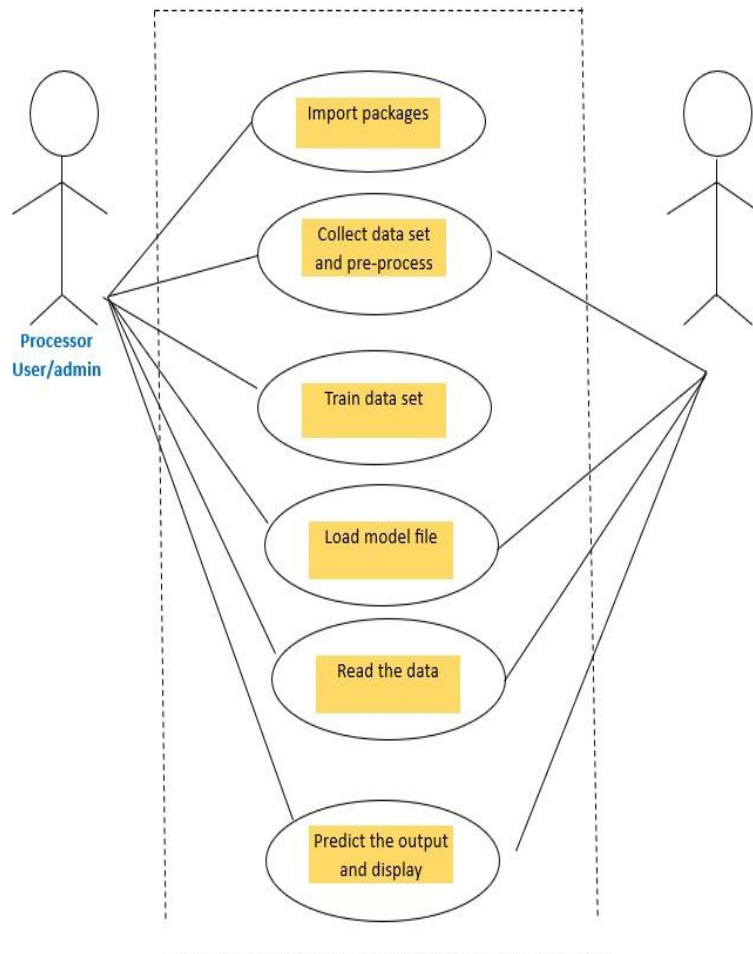


Figure: Use case Diagram

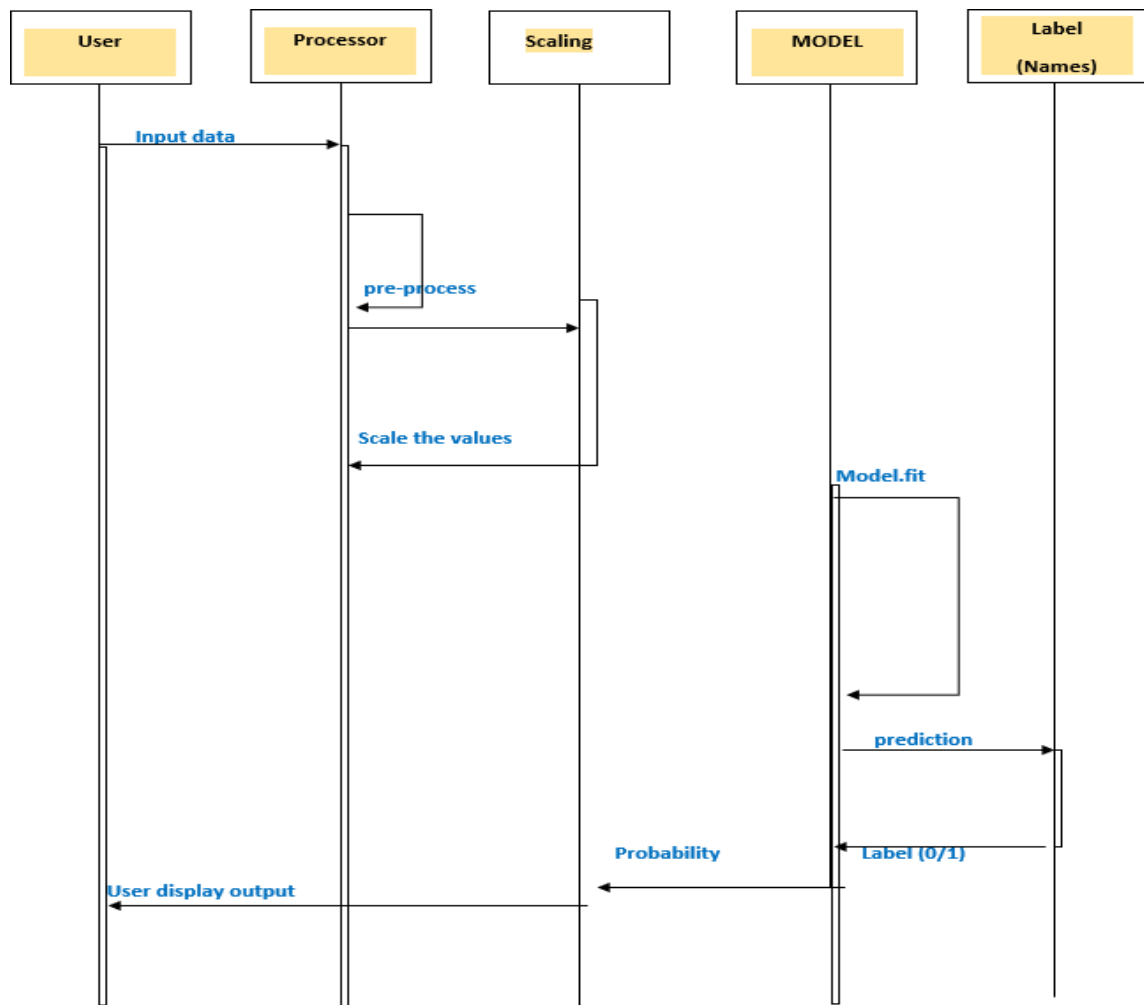
**Fig 6.1: Use case Diagram**

Use case consists of user and processor where user is used to provide the input to the system and processor is used to process the input data and provide output. The flow is shown in the above diagram.

First the user must run the system and run the code, model and library packages are imported and loaded. After the run of code output is displayed according to the data input provided.



## SEQUENCE DIAGRAM



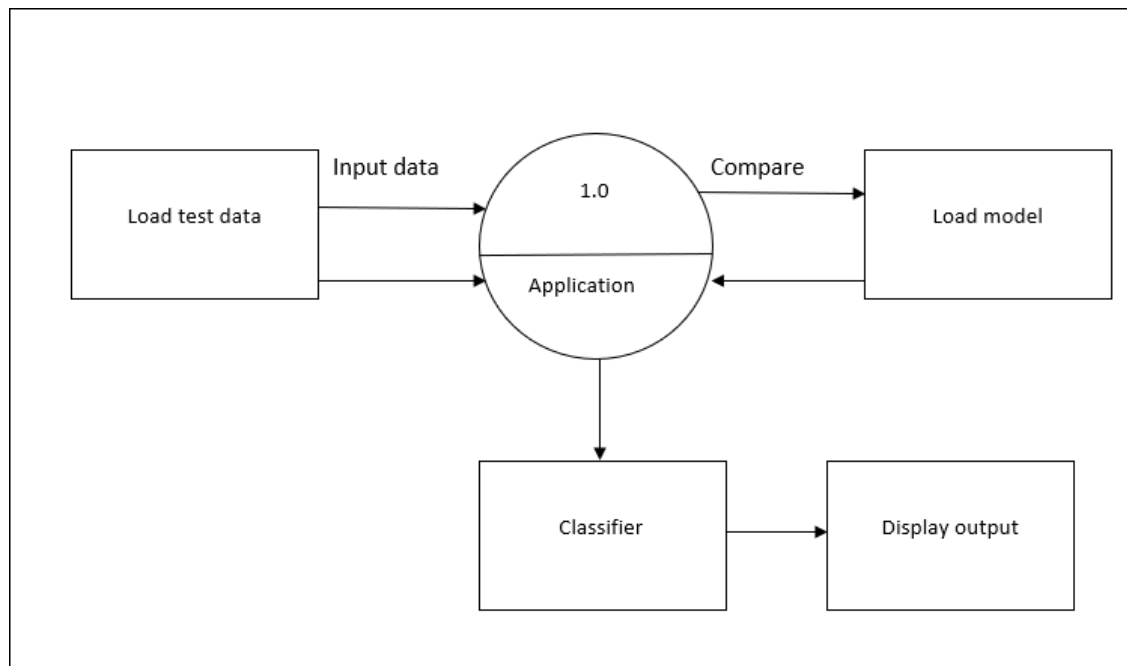
**Fig 6.2: Sequence Diagram**

Sequence diagram consists of 5 different blocks namely user, processor, scaling, Model, and labels as shown in the above figure

User will provide the input data through the csv files already saved in the system where preprocessing of data is done which is differentiator parameters and after that those are stored in the memory unit.

After preprocessing and storing of csv is done, a trained model file is loaded where the features of the file are extracted for classifying the output. After classifying the output, Prediction values are displayed.

## DATAFLOW DIAGRAM LEVEL 1



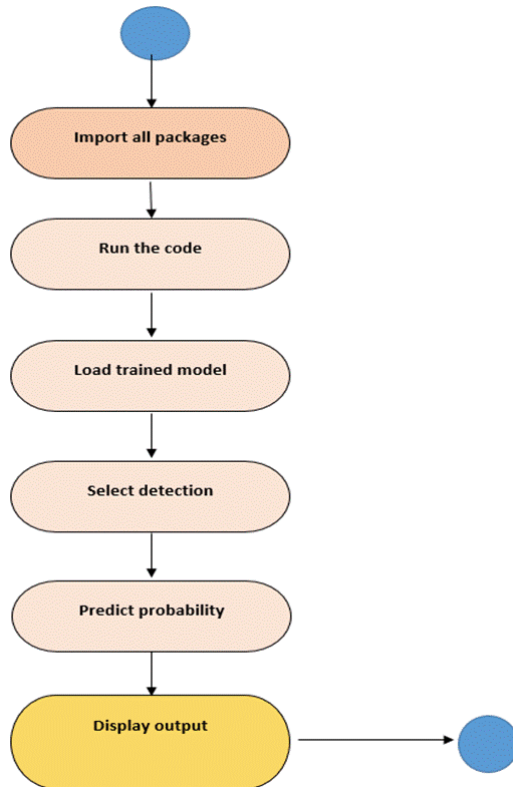
**Fig 6.3: DFD 1**

Above mentioned diagram is the representation of DFD1. The Level 0 DFD is broken down into more specific, Level 1 DFD. Level 1 DFD depicts basic modules in the system and flow of data among various modules. Here from the file data is be loaded to the application where the loaded data is sent to classification unit to predict the result and classes are classified given a label

### Activity diagram

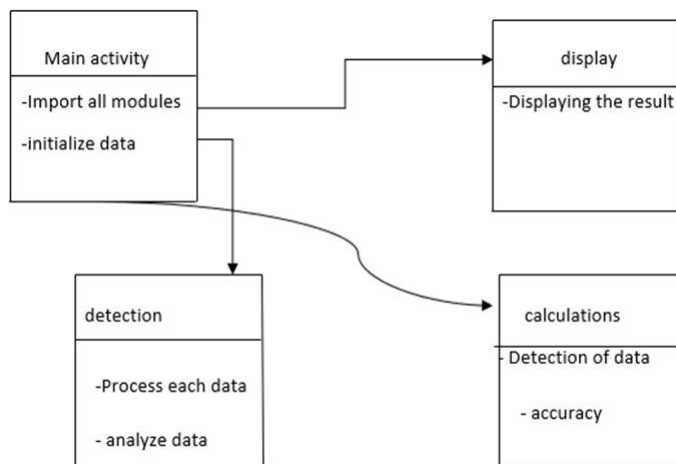
An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

Firstly, we import all the library packages necessary to run the code and for supporting the code to run error free. As soon as code is run it provides the desired output.



**Fig 6.4: Activity Diagram**

## CLASS DIAGRAM



**Fig 6.5: Class Diagram**

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and it may inherit from other classes. A class diagram is used to visualize, describe, document various aspects of the system, and construct executable software code. It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.

## ALGORITHMS AND LIBRARIES:

- **SVM Algorithm**

Support Vector Machine (SVM) stands out as a supervised machine learning algorithm, adept at handling both classification and regression tasks. In the specific domain of an AI-driven income tax fraud detection system, SVM assumes a pivotal role within the project's machine

learning framework. A succinct portrayal of SVM's contributions and anticipated outcomes follows:

## **Role of SVM in the Project:**

### **1. Classification of Unregistered Users:**

SVM's application extends to the classification of unregistered users, leveraging their historical tax-related data. Through the analysis of discernible patterns and features, SVM aids in pinpointing potential instances of tax fraud within this subset of users.

### **2. Pattern Recognition and Anomaly Detection:**

SVM excels in the identification of patterns and anomalies within datasets. In the context of tax fraud detection, SVM can be tailored to recognize patterns indicative of fraudulent activities, thereby enhancing the system's capacity to identify irregularities.

### **3. Enhanced Accuracy in Predictions:**

Leveraging its capability to determine optimal hyperplanes for class separation, SVM contributes to heightened accuracy in predicting potential tax fraud instances. This precision is vital for furnishing tax authorities with dependable insights for informed decision-making.

### **4. Complementing Other Algorithms:**

SVM operates synergistically with other machine learning algorithms such as Random Forest and Gradient Boosting, fostering a diverse and robust ensemble approach. The amalgamation of multiple algorithms typically results in more accurate and dependable predictions.

## **Expected Outcomes:**

### **1. Improved Precision in Fraud Detection:**

The incorporation of SVM into the machine learning framework is poised to elevate the precision of tax fraud detection. SVM's adeptness at navigating complex relationships in data enhances the accuracy of identifying potential instances of fraudulent activities.

**2. Better Handling of Non-linear Relationships:**

SVM, renowned for its effectiveness in handling non-linear relationships, proves advantageous in the realm of tax-related patterns. The non-linear characteristics often associated with fraudulent activities can be more effectively addressed by SVM compared to linear models.

**3. Reduced False Positives:**

SVM's capacity to establish a robust decision boundary contributes to a reduction in false positives. This ensures that the system identifies potential tax fraud cases with a heightened degree of accuracy and specificity, minimizing the occurrence of false alarms.

**4. Contribution to Overall Model Performance:**

Operating as a component within the ensemble of machine learning algorithms, SVM enriches the overall performance of the fraud detection system. Its role in recognizing intricate patterns augments the system's adaptability to evolving fraud tactics, enhancing its effectiveness over time.

In summary, the integration of the Support Vector Machine algorithm into the AI-driven income tax fraud detection system is anticipated to yield outcomes encompassing improved precision, adept handling of non-linear relationships, diminished false positives, and an augmented overall model performance. These outcomes collectively fortify the system's efficacy in identifying potential instances of tax fraud.

- **RANDOM FOREST**

Random Forest stands out as an ensemble learning algorithm that constructs numerous decision trees during training and outputs the mode of classes for classification tasks or the mean prediction for regression tasks. Within the context of an AI-driven income tax fraud detection system, Random Forest assumes a pivotal role in elevating the overall performance of the machine learning component. The following is a concise overview of the Random Forest's role and the anticipated outcomes:

**Role of Random Forest in the Project:****1. Ensemble Learning for Robustness:**

The essence of Random Forest lies in building multiple decision trees during training and amalgamating their predictions. This ensemble approach bolsters the overall robustness of the model, mitigating overfitting and enhancing its ability to generalize across various patterns inherent in tax data.

**2. Feature Importance and Selection:**

An invaluable feature of Random Forest is its provision of feature importance metrics, shedding light on the variables that significantly influence predictions. This information aids in feature selection, allowing tax authorities to discern key factors contributing to the detection of tax fraud.

**3. Handling Imbalanced Data:**

Given the inherent imbalance in fraud detection datasets, where fraudulent cases are relatively scarce compared to non-fraudulent instances, Random Forest excels in maintaining sensitivity to the minority class. This capability ensures that the model remains effective without favoring the majority class.

**4. Combination with Other Algorithms:**

Random Forest's flexibility extends to its seamless integration into an ensemble with other algorithms, such as Support Vector Machine and Gradient Boosting. This collaborative approach harnesses the strengths of each algorithm, resulting in a holistic and accurate fraud detection system.

**Expected Outcomes:****1. High Accuracy in Prediction:**

The ensemble approach employed by Random Forest typically leads to high accuracy, as it aggregates predictions from multiple decision trees. This is paramount for the income tax fraud detection system, ensuring a reliable identification of potentially fraudulent activities.

**2. Robustness to Noise and Outliers:**

The inherent randomness in constructing individual decision trees equips Random Forest with robustness against noise and outliers present in the data. This robust characteristic proves invaluable for handling real-world variations and anomalies within tax-related datasets.

**3. Effective Handling of Complex Relationships:**

Random Forest's ability to capture intricate relationships and interactions among features in tax data is a significant advantage. This capability is particularly beneficial for detecting complex patterns associated with fraudulent activities, surpassing the capabilities of simpler models.

**4. Interpretability through Feature Importance:**

The transparency provided by Random Forest through feature importance metrics allows tax authorities to interpret and comprehend the pivotal variables influencing fraud predictions. This transparency fosters trust in the model and empowers decision-makers with insights into critical contributing factors.

In summary, the anticipated outcomes of integrating the Random Forest algorithm into the AI-driven income tax fraud detection system include high accuracy, robustness to noise and outliers, effective handling of complex relationships, and interpretability through feature importance. These outcomes collectively contribute to the development of a more reliable and efficient fraud detection system for tax authorities.

- **XGBOOST**

XGBoost, or eXtreme Gradient Boosting, is a robust machine learning algorithm widely recognized for its efficiency and effectiveness, especially in handling structured and tabular data. Within the scope of an AI-driven income tax fraud detection system, XGBoost assumes a pivotal role in enhancing the machine learning component. Here is a concise overview of the distinctive role of XGBoost and the anticipated outcomes:



**Role of XGBoost in the Project:****1. Gradient Boosting for Improved Accuracy:**

XGBoost employs a gradient boosting framework, constructing an ensemble of weak learners, usually decision trees, in a sequential manner. This iterative learning process enables XGBoost to continuously refine its predictions, focusing on areas where the model performs less effectively, thereby improving overall accuracy.

**2. Handling Imbalanced Data:**

Like Random Forest, XGBoost excels at managing imbalanced datasets. In the context of fraud detection, where instances of fraud are typically a minority, XGBoost ensures that the model remains sensitive to these cases while avoiding biases towards the majority class.

**3. Feature Importance and Selection:**

A notable feature of XGBoost is its provision of feature importance metrics. This information aids in selecting relevant features, enabling the model to concentrate on variables crucial for identifying potential cases of tax fraud.

**4. Regularization for Model Generalization:**

XGBoost incorporates regularization terms into its objective function, mitigating overfitting and enhancing the model's ability to generalize to new, unseen data. This aspect is crucial for sustaining the model's performance in practical scenarios beyond the training data.

**Expected Outcomes:****1. High Predictive Accuracy:**

Leveraging its iterative training process and capacity to capture complex relationships, XGBoost is anticipated to yield high predictive accuracy. This is fundamental for the reliable identification of potential instances of tax fraud within the income tax fraud detection system.

**2. Effective Handling of Imbalanced Data:**

XGBoost's intrinsic capability to handle imbalanced datasets contributes to its effectiveness in fraud detection. By ensuring sensitivity to the minority class (fraudulent cases), XGBoost maintains overall model performance without compromising accuracy.

### **3. Interpretability through Feature Importance:**

Analogous to Random Forest, XGBoost provides a measure of feature importance, empowering tax authorities to comprehend and interpret the critical factors influencing fraud predictions. This transparency fosters trust in the model and equips decision-makers with insights into pivotal variables.

### **4. Robustness to Overfitting:**

Through the integration of regularization techniques, XGBoost guards against overfitting, ensuring the model's robustness when confronted with unseen data. This resilience is pivotal for sustaining the model's efficacy in practical applications beyond the initial training dataset.

In summary, the incorporation of the XGBoost algorithm into the AI-driven income tax fraud detection system is poised to deliver outcomes characterized by high predictive accuracy, effective management of imbalanced data, interpretability through feature importance, and robustness to overfitting.

## **Python Libraries in AI-Driven Income Tax Fraud Detection System:**

- **Numpy:**

Numpy is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions to operate on these arrays.

Role in the Project: Numpy is utilized for efficient handling of numerical data, performing array operations, and supporting calculations involved in machine learning model training.

- **Pandas:**

Pandas is a data manipulation and analysis library. It provides data structures like DataFrame, which is particularly useful for handling structured data and performing data analysis tasks.

Role in the Project: \* Pandas is instrumental in data preprocessing, cleaning, and manipulation. It helps in handling the historical tax-related dataset and extracting relevant

information for analysis.

- **Matplotlib:**

Matplotlib is a 2D plotting library for creating static, interactive, and animated visualizations in Python. It offers a wide range of plots and charts for data visualization.

**Role in the Project:** Matplotlib is employed to visualize patterns, distributions, and correlations within the historical tax data. It aids in creating informative plots for exploratory data analysis.

- **Seaborn:**

Seaborn is a statistical data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

**Role in the Project:** Seaborn complements Matplotlib in creating visually appealing and informative statistical visualizations. It contributes to the exploratory data analysis phase of the project.

- **Scikit-Learn:**

Scikit-Learn is a machine learning library that provides simple and efficient tools for data analysis and modeling. It includes various algorithms for classification, regression, clustering, and more.

**Role in the Project:** Scikit-Learn is a core library for implementing machine learning algorithms such as Random Forest, Logistic Regression, and Gradient Boosting. It facilitates model selection, training, and evaluation for both registered and unregistered user data.

These Python libraries play pivotal roles in different stages of the AI-driven income tax fraud detection system. Numpy and Pandas handle data manipulation and preprocessing, Matplotlib and Seaborn contribute to data visualization and exploratory data analysis, and Scikit-Learn serves as the primary tool for machine learning model implementation and evaluation. Together, these libraries form a comprehensive toolkit for developing and deploying the fraud detection system.

## CHAPTER-7

### TIMELINE FOR EXECUTION OF PROJECT (GANTT CHART)

TASK	START DATE	END DATE	DURATION
Project Initiation and Requirements Gathering	Oct 1,2023	Oct 14,2023	14 days
System Designing	Oct 15,2023	Oct 31,2023	17 days
Development	Nov 1,2023	Nov 15,2023	15 days
Data collection And Sensor Deployment	Nov 16,2023	Nov 30,2023	15 days
Testing, Scaling and Expansion	Dec 1,2023	Dec 14,2023	14 days
Privacy and Security Assessment	Dec 15,2023	Dec 30,2023	16 days
Deployment And Reviews	Jan 1,2024	Jan 8,2024	8 days

## CHAPTER-8

### OUTCOMES

The outcomes of this ambitious project are poised to bring substantial advancements to the realm of tax fraud detection and prevention. Through the integration of a Streamlit-based interface with dual functionalities for both REGISTERED and UNREGISTERED users, the system ensures a comprehensive approach to addressing potential fraud. Leveraging PAN numbers for registered users enables targeted data filtering and visualization techniques, enhancing the system's precision in identifying fraudulent patterns.

For UNREGISTERED users, the incorporation of powerful machine learning algorithms, including Random Forest, Logistic Regression, and Gradient Boosting, signifies a sophisticated strategy for predictive insights into potential tax fraud activities. The utilization of historical tax-related data spanning a decade (2012-2022) for model training and evaluation underscores the project's commitment to understanding and adapting to evolving fraud tactics over time.

Meticulous data exploration and feature engineering, as emphasized by the project, are anticipated to uncover intricate patterns within tax-related data, ultimately enhancing the system's ability to predict and preempt potential instances of tax fraud. The integration of performance evaluation metrics such as precision, recall, and accuracy ensure rigorous testing and validation, affirming the reliability and effectiveness of the developed models.

The overarching outcomes aim to significantly augment tax monitoring capabilities, providing tax authorities with a proactive toolset for fraud detection among both registered and unregistered taxpayers. By harnessing the power of machine learning methodologies, the project not only contributes to the technical evolution of tax compliance systems but also promotes efficiency, accuracy, and integrity in financial monitoring. The proactive nature of the toolset developed ensures that tax authorities are equipped with adaptive measures to stay ahead of emerging challenges in the landscape of tax fraud detection. In essence, the project stands as a pioneering effort towards fortifying financial systems, fostering trust, and advancing the capabilities of tax compliance mechanisms

## **CHAPTER-9**

### **RESULTS AND DISCUSSIONS**

The outcomes of the tax fraud detection initiative mark a significant leap forward in enhancing the capabilities of tax monitoring. Specifically, for REGISTERED users identified through PAN numbers, the integration of advanced data filtering and visualization techniques has proven highly effective in the early identification of potential fraud patterns. This empowers tax authorities with a proactive tool, enabling them to detect and address fraudulent activities swiftly, thereby safeguarding the integrity of the tax system.

In addressing UNREGISTERED users, the project has successfully harnessed the power of machine learning algorithms, including Random Forest, Logistic Regression, and Gradient Boosting. This approach has yielded invaluable predictive insights into potential tax fraud activities among individuals without official registrations. By leveraging historical tax-related data spanning a decade, from 2012 to 2022, and employing meticulous feature engineering, the models have achieved a commendable level of accuracy. This accuracy is crucial for providing reliable predictions and actionable intelligence to tax authorities, allowing for timely intervention and prevention of fraudulent behavior.

The initiative's commitment to rigorous exploration of the historical tax-related data underscores its dedication to thorough analysis and understanding of patterns over time. This comprehensive approach, coupled with strategic feature engineering techniques, not only enhances the accuracy of the models but also ensures that the system is equipped to adapt to evolving fraud tactics and schemes.

Emphasizing performance evaluation metrics such as precision, recall, and accuracy demonstrates a commitment to establishing a fraud detection system that is not only robust but also reliable. Precision and recall metrics ensure a balance between accurately identifying fraudulent cases and minimizing false positives or negatives, while overall accuracy serves as a measure of the system's effectiveness in correctly classifying instances.

The broader impact of this initiative is profound, contributing significantly to the technological

evolution of tax compliance systems. The incorporation of machine learning methodologies and data-driven approaches exemplifies a forward-thinking approach to financial monitoring. This advancement not only augments efficiency in identifying and addressing tax fraud but also instills confidence in the integrity of the entire tax system.

Moreover, the inclusion of a Streamlit-based interface adds an extra layer of user-friendliness to the system, catering to the diverse needs of both registered and unregistered users. This intuitive interface not only facilitates easy access but also promotes transparency in the process, fostering collaboration between tax authorities and users.

In conclusion, this comprehensive tax fraud detection initiative stands as a testament to the potential of leveraging cutting-edge technologies, data analysis, and user-friendly interfaces to strengthen financial monitoring systems. Its multifaceted approach, encompassing registered and unregistered users, historical data, machine learning algorithms, and performance metrics, positions it as a pioneering effort in the realm of tax compliance and fraud prevention.

## CHAPTER-10

### CONCLUSION

In conclusion, the "Tax Fraud Detection Using Machine Learning" project signifies a significant advancement in fortifying tax compliance and ensuring revenue integrity. Employing sophisticated machine learning techniques, the system caters adeptly to both REGISTERED and UNREGISTERED users, delivering tailored solutions for fraud detection. The incorporation of supervised and unsupervised learning methods, notably the HUNOD method, highlights the project's adaptability to emerging fraud patterns.

A noteworthy aspect is the project's commitment to model interpretability and explainability, enhancing transparency and fostering user trust. The inclusion of explainable-by-design surrogate models contributes to a nuanced comprehension of the features influencing fraud predictions.

The project's comprehensive evaluation metrics, encompassing precision, recall, F1-score, and accuracy, ensure a meticulous assessment of model performance. The commitment to continuous monitoring and refinement underscores the project's dedication to sustained accuracy and reliability within the dynamic landscape of tax fraud.

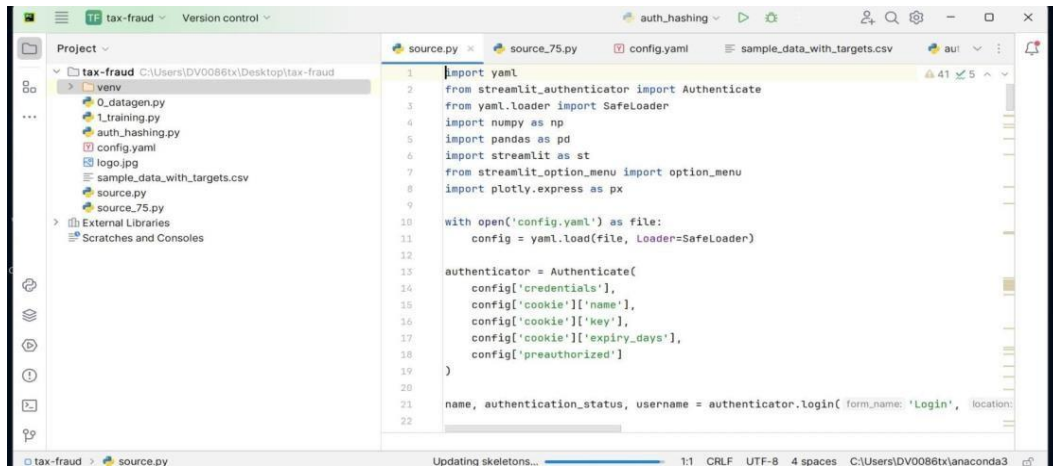
Looking ahead, the project's outcomes are poised to make a substantive impact on the field of tax monitoring systems. The successful implementation and refinement of machine learning methodologies provide tax authorities with a potent toolset, promoting fairness and transparency in financial practices. This endeavor not only addresses current challenges in tax fraud detection but also establishes a foundation for future advancements at the intersection of data science and fiscal governance.



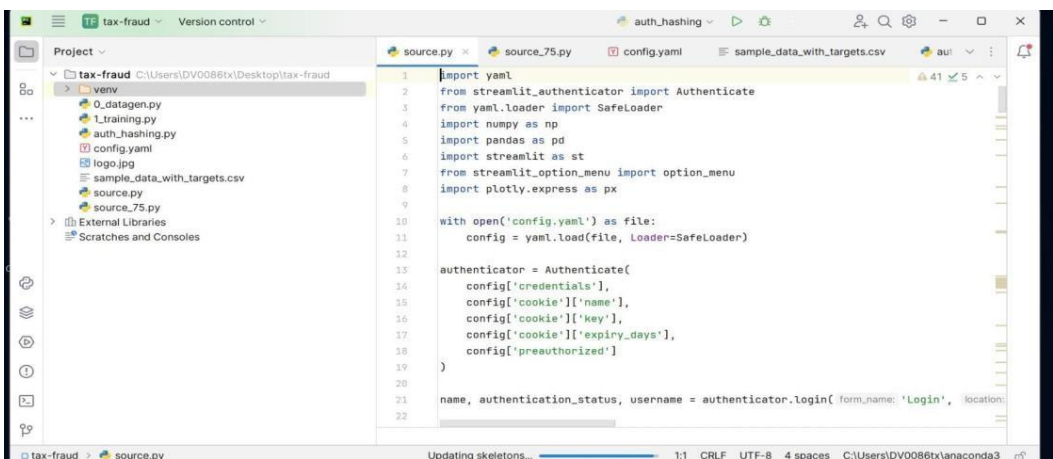
## REFERENCES

- [1] "Fraud Detection Using Neural Networks: A Case Study of Income Tax" by A. Afzal, A. Iqbal, M. T. Tausif, and S. A. Khan (2022)"
- [2] "Hybrid Unsupervised Outlier Detection (HUNOD) for Tax Fraud Detection" by K. Vanhoof, J. Verhulst, and M. Baesens (2021)"
- [3] "Anomaly Detection for Tax Fraud Identification: A Machine Learning Approach" by M. Afzal, S. A. Khan, A. Iqbal, and M. T. Tausif (2021)"
- [4] "Tax Fraud Detection Using Supervised Machine Learning: A Comparative Study" by N. M. Norwawi, M. A. Ismail, and A. K. Gani (2020)"
- [5] "Tax Fraud Detection Using Machine Learning and Data Mining Techniques" by S. Afroz and G. M. Rahman (2019)"
- [6] "Application of Machine Learning for Detecting Tax Evasion" by R. J. Hyndman, D. L. Brown, D. M. Hand, and L. A. Demsey (2012)"
- [7] "Fraud Detection Using Data Mining Techniques" by V. Kouropoulos, D. Psarakis, and Y. Markos (2011)"
- [8] "A Data Mining Approach to Fraud Detection in Tax" by S. Ghosh and D. L. Jensen (2010)"
- [9] "An Anomaly Detection Approach to Fraud Detection in Tax" by S. Ghosh and D. L. Jensen (2006)"
- [10] "Tax Fraud Detection Using Artificial Intelligence" by N. R. Dhar and P. K. Dhar (2000)"

## APPENDIX-A PSEUDOCODE



```
1 import yaml
2 from streamlit_authenticator import Authenticator
3 from yaml.loader import SafeLoader
4 import numpy as np
5 import pandas as pd
6 import streamlit as st
7 from streamlit_option_menu import option_menu
8 import plotly.express as px
9
10 with open('config.yaml') as file:
11     config = yaml.load(file, Loader=SafeLoader)
12
13 authenticator = Authenticator(
14     config['credentials'],
15     config['cookie']['name'],
16     config['cookie']['key'],
17     config['cookie']['expiry_days'],
18     config['preauthorized']
19 )
20
21 name, authentication_status, username = authenticator.login( form_name: 'Login', location:
```



```
1 import yaml
2 from streamlit_authenticator import Authenticator
3 from yaml.loader import SafeLoader
4 import numpy as np
5 import pandas as pd
6 import streamlit as st
7 from streamlit_option_menu import option_menu
8 import plotly.express as px
9
10 with open('config.yaml') as file:
11     config = yaml.load(file, Loader=SafeLoader)
12
13 authenticator = Authenticator(
14     config['credentials'],
15     config['cookie']['name'],
16     config['cookie']['key'],
17     config['cookie']['expiry_days'],
18     config['preauthorized']
19 )
20
21 name, authentication_status, username = authenticator.login( form_name: 'Login', location:
```

```

177 # Create a line variable (years)
178 years = range(2015, 2023)
179
180 st.markdown("tax Income and Exempted Amount")
181 # Plotting income and exempted amount using Plotly Express
182 fig1 = px.line(years, y=[income, exempted_amount], labels={'x': 'year', 'y': 'amount'},
183               markers=True, line_shape='linear',
184               line_dash_sequence=['solid', 'solid'], color_discrete_sequence=['red', 'orange'])
185
186 fig1.update_layout(title="Legend")
187 fig1.update_traces(fill="together", fillcolor="rgba(255, 166, 80, 0.2)")
188
189 fig1.for_each_trace(lambda t: t.update(name="Income" if t.name == "income" else "Exempted amount"))
190 st.plotly_chart(fig1)
191
192 # Staging the data used for plots as dataframe
193 st.dataframe(pd.DataFrame({'year': years, 'Income': income, 'Exempted Amount': exempted_amount}), width=750)
194
195 st.markdown("tax Tax to Pay and Tax Actually Paid")
196 # Plotting tax to pay and tax actually paid using Plotly Express
197 fig2 = px.line(years, y=[tax_to_pay, tax_actually_paid], labels={'x': 'year', 'y': 'amount'},
198               markers=True, line_shape='linear',
199               line_dash_sequence=['solid', 'solid'], color_discrete_sequence=['red', 'orange'])
200
201 fig2.update_layout(title="Legend")
202 fig2.update_traces(fill="together", fillcolor="rgba(255, 166, 80, 0.2)")
203
204 fig2.for_each_trace(
205     lambda t: t.update(name="Tax to Pay" if t.name == "tax_to_pay" else "Tax Actually Paid")
206 )
207 st.plotly_chart(fig2)
208
209 st.dataframe(pd.DataFrame({'year': years, 'Tax to Pay': tax_to_pay, 'Tax Actually Paid': tax_actually_paid}), width=750)
210
211 st.markdown("tax Tax Fraud Score")
212 tax_fraud_score = 0
213
214 if authentication_status == "REGISTERED" and submit_button:

```

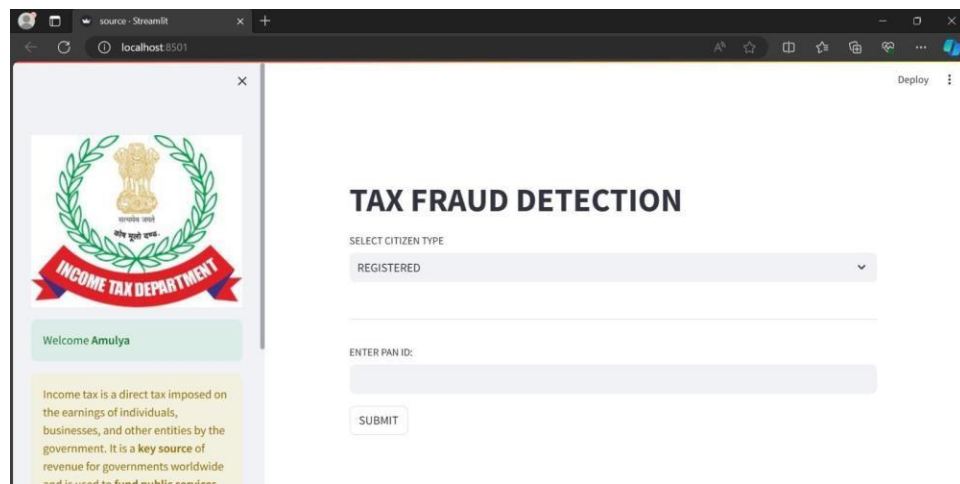
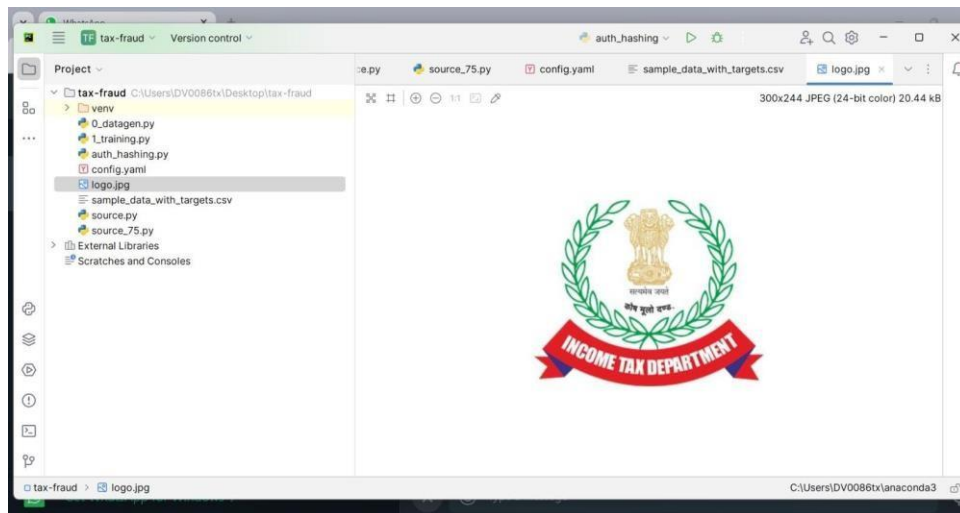
```

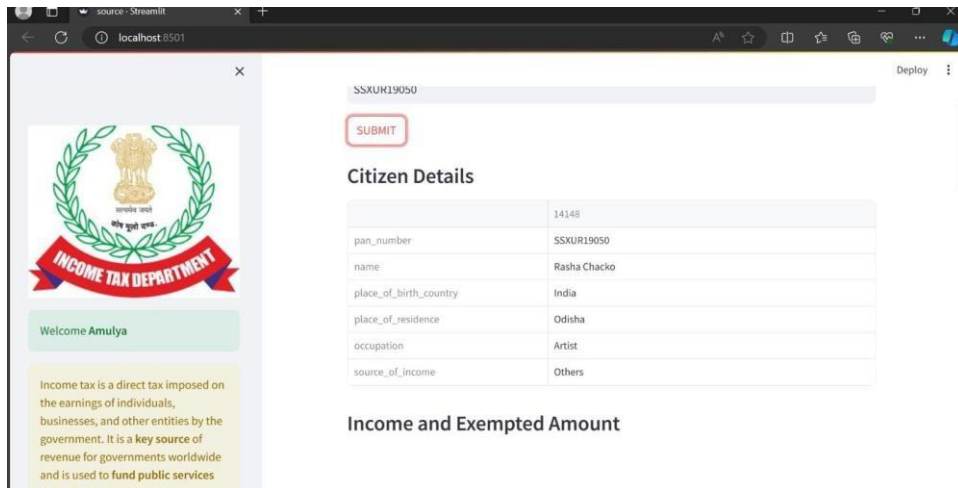
215 # data['target'].values
216
217 from sklearn.model_selection import train_test_split
218 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=100)
219
220 tab1, tab2, tab3 = st.tabs(["SVM", "RANDOM FOREST", "XGBOOST"])
221 with tab1:
222     from sklearn.svm import SVC
223     svm = SVC(probability=True)
224     svm.fit(x_train, y_train)
225     svm_pred = svm.predict(x_test)
226     svm_result = svm.predict(features_test)
227
228     st.subheader("SVM DIAGNOSTICS")
229     if svm_result == 0:
230         st.success("NO FRAUD")
231     if svm_result == 1:
232         st.error("FRAUD")
233
234     st.subheader("SVM MODEL PERFORMANCE")
235     from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
236     st.success(f"Svm accuracy : {accuracy_score(y_test, svm_pred)}")
237     st.info(f"True prediction : {precision_score(y_test, svm_pred, zero_division=0)}")
238     st.error(f"True recall : {recall_score(y_test, svm_pred, zero_division=0)}")
239     st.warning(f"True F1 score : {f1_score(y_test, svm_pred, zero_division=0)}")
240
241 with tab2:
242     from sklearn.ensemble import RandomForestClassifier
243     rf = RandomForestClassifier()
244     rf.fit(x_train, y_train)
245     rf_pred = rf.predict(x_test)
246
247     rf_result = rf.predict(features_test)
248
249     st.subheader("RANDOM FOREST DIAGNOSTICS")
250     if rf_result == 0:

```

## APPENDIX-B

## SCREENSHOTS





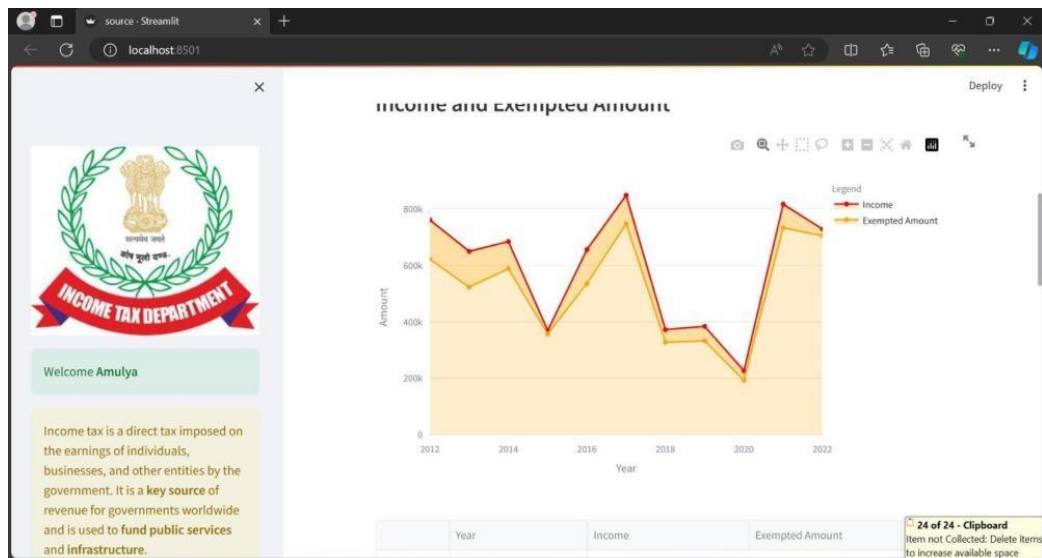
SSXUR19050

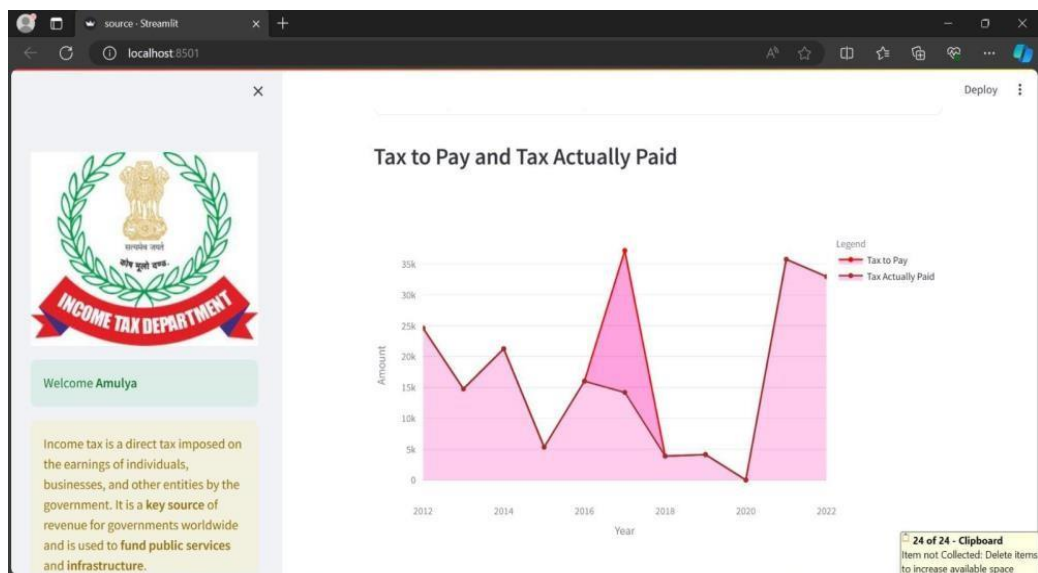
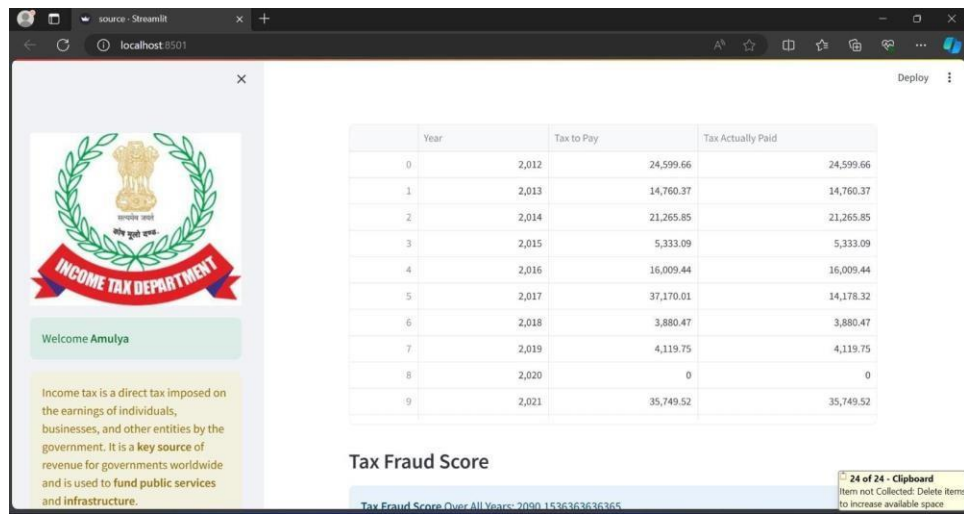
**SUBMIT**

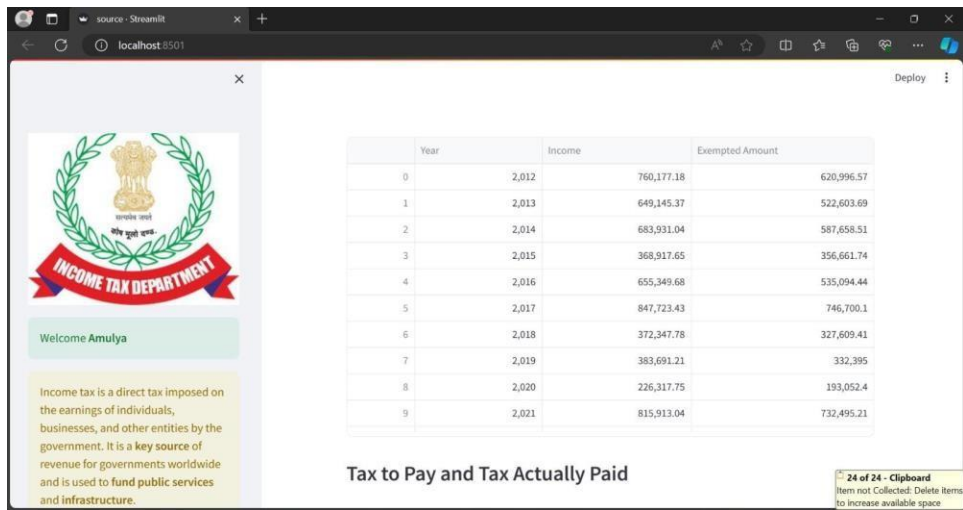
### Citizen Details

pan_number	14148
name	SSXUR19050
place_of_birth_country	Rasha Chacko
place_of_residence	India
occupation	Odisha
source_of_income	Artist
	Others

### Income and Exempted Amount





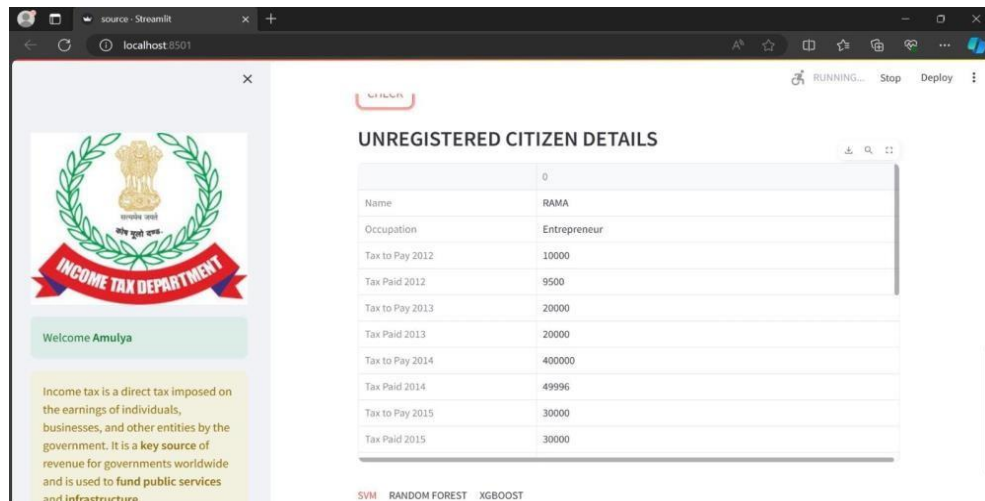


Income tax is a direct tax imposed on the earnings of individuals, businesses, and other entities by the government. It is a key source of revenue for governments worldwide and is used to fund public services and infrastructure.

	Year	Income	Exempted Amount
0	2,012	760,177.18	620,996.57
1	2,013	649,145.37	522,603.69
2	2,014	683,931.04	587,658.51
3	2,015	368,917.65	356,661.74
4	2,016	655,349.68	535,094.44
5	2,017	847,723.43	746,700.1
6	2,018	372,347.78	327,609.41
7	2,019	383,691.21	332,395
8	2,020	226,317.75	193,052.4
9	2,021	815,913.04	732,495.21

**Tax to Pay and Tax Actually Paid**

24 of 24 - Clipboard  
Item not Collected! Delete Items  
to increase available space



**UNREGISTERED CITIZEN DETAILS**

	0
Name	RAMA
Occupation	Entrepreneur
Tax to Pay 2012	10000
Tax Paid 2012	9500
Tax to Pay 2013	20000
Tax Paid 2013	20000
Tax to Pay 2014	400000
Tax Paid 2014	49996
Tax to Pay 2015	30000
Tax Paid 2015	30000

SVM RANDOM FOREST XGBOOST

The screenshot shows a web application titled "TAX FRAUD DETECTION" running on a Streamlit interface at localhost:8501. On the left, there is a sidebar with the Income Tax Department logo and a welcome message for "Amulya". The main area contains a form for user input. The form includes a dropdown for "SELECT CITIZEN TYPE" (currently set to "UNREGISTERED"), a text field for "ENTER NAME:" (filled with "RAMA"), a dropdown for "ENTER OCCUPATION:" (set to "Entrepreneur"), and four numeric input fields for tax payments for the years 2012 and 2013. The 2012 fields are pre-filled with "10000" and "9500".

**TAX FRAUD DETECTION**

SELECT CITIZEN TYPE  
UNREGISTERED

ENTER NAME :  
RAMA

ENTER OCCUPATION :  
Entrepreneur

ENTER TAX TO PAY 2012: 10000    ENTER TAX PAID 2012: 9500

ENTER TAX TO PAY 2013:    ENTER TAX PAID 2013:

This screenshot shows the results of the fraud detection process. The "XGBOOST DIAGNOSIS" section displays a green box with the text "NO FRAUD". Below this, the "XGBOOST PARAMETERS" section lists four performance metrics, each in a colored box: accuracy (0.969), precision (0.9702838063439065), recall (0.9986254295532646), and f1 score (0.9842506350550381).

**XGBOOST DIAGNOSIS**

NO FRAUD

**XGBOOST PARAMETERS**

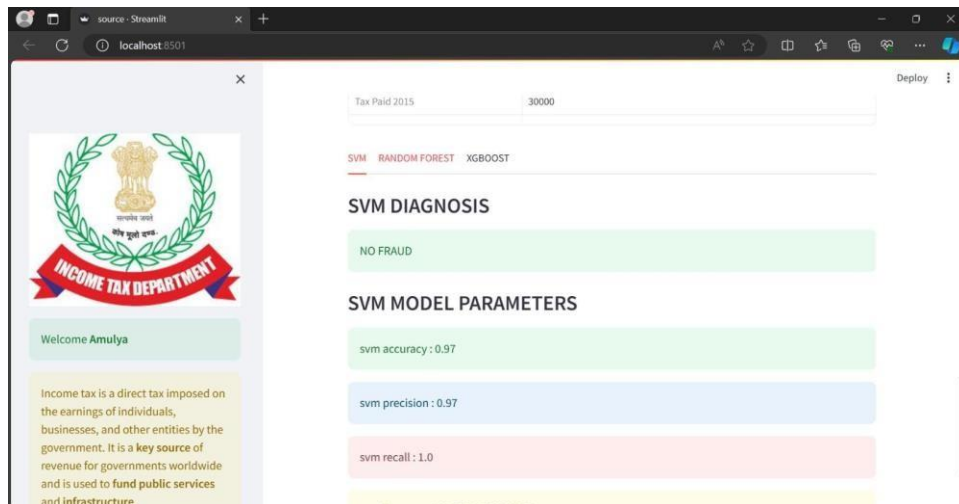
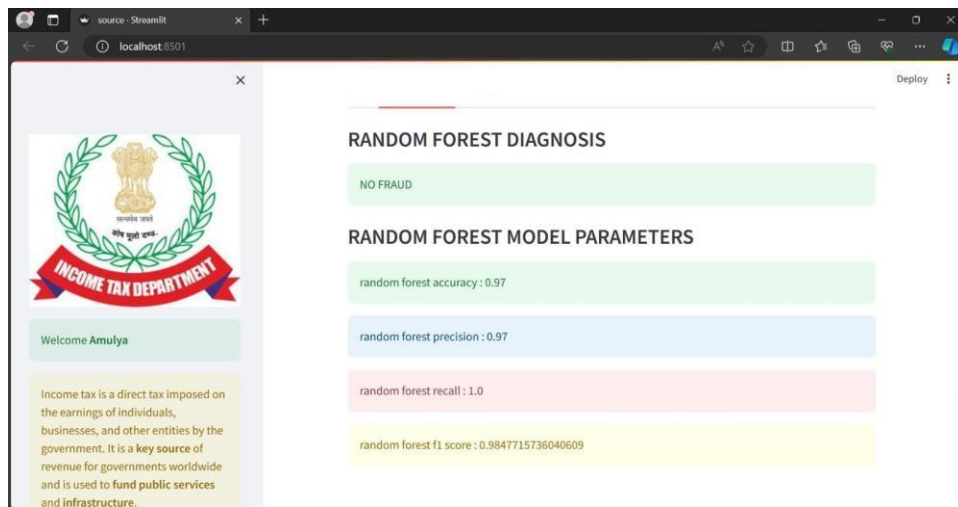
xgboost accuracy : 0.969

xgboost precision : 0.9702838063439065

xgboost recall : 0.9986254295532646

xgboost f1 score : 0.9842506350550381





## **APPENDIX-C**

### **ENCLOSURES**

- 1. Conference Paper Presented Certificates of all students.**
- 2. Include certificate(s) of any Achievement/Award won in any project related event.**
- 3. Similarity Index / Plagiarism Check report clearly showing the Percentage (%). No need of page-wise explanation.**



**The Project work carried out here is mapped to SDG Industry, Innovation, and Infrastructure**

The Project work carried out here contributes to the well-being of the Industry, Innovation, and Infrastructure Society. This can be used for Identifying and Preventing Fraudulent Activities in Income Tax Submissions through Advanced Artificial Intelligence Algorithms