Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol. 14, Issue 14s (2025)
https://www.jneonatalsurg.com

OPEN ACCESS

Improving Income Tax Fraud Detection Accuracy with Logistic Regression

P Sudha[1], Rafeeda Fatima[2], Meena Kumari KS[3], Vinutha CB[4]

[1]Department of Computer Science and Engineering, Presidency University, Bangalore, Karnataka, India
Email ID: sudha.p@presidencyuniversity.in
[2]Department of Computer Science and Engineering, Presidency University, Bangalore, Karnataka, India
Email ID: rafeeda28@gmail.com
[3]Department of Computer Science and Engineering, Presidency University, Bangalore, Karnataka, India
Email ID: meenakumari.ks@presidencyuniversity.in
[4]Department of Electronics and Communication Engineering, Presidency University, Bangalore, Karnataka, India
Email ID: vinutha.cb@presidencyuniversity.in

Cite this paper as: P Sudha, Rafeeda Fatima, Meena Kumari KS, Vinutha CB, (2025) Improving Income Tax Fraud Detection Accuracy with Logistic Regression. *Journal of Neonatal Surgery*, 14 (14s), 461-474.

ABSTRACT
Another critical source of revenue for governments involves taxation on income for people and companies, which must be paid under law. Probably the biggest challenge is tax fraud, which is the act of intentionally machining the declaration to avoid taxation. In conjunction with this, our project aims to analyze the financial data of taxpayers and establish a strong machine learning model for flagging such errant behavior. The authors compared six machine learning algorithms, the six types being a Feedforward Neural Network, k nearest Neighbors, Random Forest, Naive Bayes, Decision Tree, and Logistic Regression, for the detection and classification of tax fraud. Logistic regression performed best in the detection of tax fraud. The proposed framework is found to be better, whereby it is able to identify complex patterns better than the previously existing methods since both linear and non-linear correlations are considered in between variables. We trained and ran our model on the OpenML dataset. The results seem promising. Our model also assures that, besides being financially viable, it promotes fruitfulness in policy design with much less tax revenue loss. Using Tensor Flow for deployment of the model on Android Studio also enhanced accessibility. It has also led to the build-up of a simple prediction tool that helped people understand the risks involved in tax fraud.

Keywords *Income Tax Fraud Detection, Logistic Regression, Decision Tree, Random Forest, Naive Bayes, K-Nearest Neighbors, Feed Forward Neural Network, Android Studio*.

1. INTRODUCTION

Income tax is crucial for the functioning of our society, as it provides countries with the necessary revenue to make vital investments in infrastructure, health, and education. However, despite its importance, many people are averse to paying taxes, and make the government lose millions of dollars every year. There are various strategies to evade taxes, such as underreporting income, which reduces the tax liability. Criminals who commit fraud are becoming increasingly sophisticated in their methods, making it difficult to identify them. In many cases, they try to blend in with their environment, much like military units that use camouflage or chameleons that use their colouring to hide from predators. These tactics are not random, but rather carefully planned and executed. As a result, new techniques are needed to detect and address patterns that appear to be normal but are actually part of fraudulent activities. Tax authorities are given the task of finding these fraudsters and usually rely on experts' intuition. Random auditing is a way of discouraging tax frauds. The way tax audits are conducted are undergoing a huge shift with the introduction of artificial intelligence (AI) and machine learning (ML). Conventional modes are either painfully slow, resource-consuming, or simply leave out vital information. However, artificial intelligence can include examining vast databases in rapid succession across extensive ranges that go unnoticed by a human. This can enhance precision while saving considerable time during an audit.

ML revolves around a domain of artificial intelligence within which computer systems use statistical models and algorithms to learn from data and improve performance concerning specific tasks. In brief, machine learning algorithms give computer systems the ability to learn and make decisions based on pattern and trend recognition from vast volumes of data. Figure-1 gives an overview of the classification of ML into three basic categories: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is a procedure whereby a machine is trained using labeled data to predict or classify something based

upon that input data. Unsupervised learning enables the machine to process unlabeled data, thus enabling it to analyze the data and determine any inherent patterns and relationships within without any human help. Reinforcement learning is a machine learning approach whereby the machine learns through the actions it takes in an environment based on specific response feedback provided in the form of rewards and punishments, determined by the results of such actions.

Machine learning has shown great potential in the domain of health management, finance, marketing, and cybersecurity. Its ability to learn from data and adjust itself when things change, without being told how, has established it as one of the strongest tools in modern data analytics.

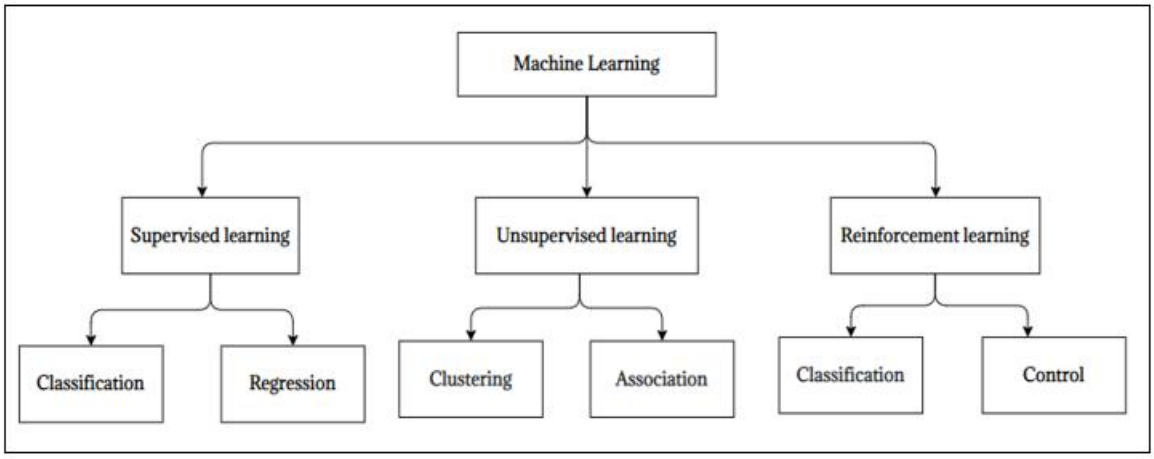

Figure-1. Classification of Machine Learning.

Android Studio is an IDE specifically designed for Android application development. The platform enables developers to design, program, debug, and deploy their apps using its plethora of features and tools. Built on top of IntelliJ IDEA, Android Studio comes bundled with the Android SDK, which is fairly rich and has all the references for development and testing of Android applications. This development platform supports several programming languages, including Java, Kotlin, and C++. Android Studio also sports advanced features such as code completion, debugging, profiling, and testing. Developers can create visually pleasing user interfaces, have their apps interact with backend services, and utilize machine learning platforms like TensorFlow to create data-driven applications; this has changed the course of Android app creation, simplifying the once tedious task for developers to create great and efficient apps targeted for Android. Millions of Android devices worldwide are compatible with the apps built with Android Studio.

Motivation and Contribution

This article consists of cutting-edge methodology that identifies and prevents income tax fraud using machine learning algorithms. The study is based on decision tree, random forest, naive Bayes, k-nearest neighbors, feed-forward neural network, and logistic regression models. Thus, all these algorithms combine to suspicious activities related to income tax fraud. The methodology designed to be addressing the OpenML Income Dataset investigates behavior and demographic characteristics within model logistic regression. Extensive training and testing activities have taken into consideration in calculating the effectiveness of these approaches, which consequently highlights considerable improvement on detection and prevention of fraud. The evaluation is being enhanced with the use of the Android application developed within Android Studio IDE based on collecting all necessary inputs considering the effective model. Hence, the new idea introduced stands out from other kinds of solutions dealing with the recurring issue of income tax fraud, equal to that shown in many other countries.

Organization of the paper

The rest of the paper will flow in this sequence: Section 2-The Literature Survey, which covers existing research related to this field; Section 3-The proposed methods and methodology for the various studies; Section 4-The Results and discussion covering the results obtained and their implications; and finally, in Section 5- Concluding Remarks, the summary of the major achievements is presented in conclusion.

2. LITERATURE REVIEW

Neural networks are widely recognized as a well-known and efficient technique of machine learning for fraud detection. Provides case study of income tax fraud detection and boasts its accuracy. The problem statement has limited data and there is a concern regarding the data privacy since this method requires access to sensitive personal data [1]. The neural network's enormous potential in machine learning makes them practically unique in being able to discover complex

patterns in huge and intricate datasets, obtaining accreditations of accuracy higher than 95%. Instances of tax evasion moreover present some difficulties due to a lack of transparency, erroneous suit findings, and a general lack of data [2]. Finally, different approaches based on data mining techniques promise to increase the automation of analysis and generate high-risk taxpayer list from voluminous data, thus reducing the need for extensive manual examination. This one is efficient and less time-consuming compared to any other traditional way of analysis and provides increasing accuracy and flexibility. Such techniques have been successfully applied to diverse data sources including financial transactions, tax returns, and other relevant data and can easily accommodate large datasets. However, it is noteworthy to mention that the success of data mining techniques for tax fraud detection is dependent upon data quality, the choice of algorithm, interpretation of the model, and privacy concerns [3]. Major improvements have been accomplished by the improvement of the particle swarm optimization algorithm to facilitate tax evasion detection and reach an accuracy of 95%. It saves time and money, but it must be tested against a larger dataset before it can be accepted with confidence [4].

The proposed technique is based on clustering to separate the taxpayers in groups with similar income profiles and find out which ones are reporting income significantly lower. It relies very much on the works of one type and requires conducting a manual investigation to confirm the suspected tax fraud [5]. Milos Savić has created a hybrid unsupervised outlier detection approach to detect risks of tax evasion. This method fuses the strength of unsupervised and supervised techniques in order to increase the accuracy of the detection procedure. However, while this method can identify tax evasion in a specific case involving a grocery shop owner, it has limited applicability and fails to consider the ethical and legal issues of tax evasion detection. As such, it becomes critical to assess the method for its reliability and effectiveness when applied over different datasets and scenarios [6]. The article by González and Velásquez titled "Characterization and Detection of Taxpayers Engaging in Tax Evasion through False Invoices: A Data Mining Approach" highlights the identification and profiling of individuals using false invoices to evade taxes within the Colombian context. The authors used different data mining techniques, including decision trees, neural networks, and logistic regression, to abstract patterns within tax data that can be helpful in recognizing fraudulent activities. The results of this study present practical implications for policymakers and tax authorities wishing to enhance tax compliance and curtail instances of tax evasion [7]. A neural network classifies credit card fraud. Neural networks are so complex that they require a lot of information for training, which renders them inefficient for smaller datasets. Neural networks are very expensive to train and deploy and do not address any issues relating to data privacy and security. Data protection and ethical use need to be ensured [8].

AI and ML algorithms have demonstrated their worth in detecting fraudulent tax returns in throughout income tax audits. Their application shows successful outcomes employed in Taiwan for both profit-seeking enterprise income tax as well as for individual income tax as presented in the study. This research sheds significant insight into the major causes of tax fraud, consequently requests good practice for more effective tax regulations and policies. However, the attributes studied and findings of this research may exist only within the context of Taiwan's tax system. The study leaves room for further investigations regarding approaches for handling missing or unreliable data regarding the system [9]. It includes many techniques like descriptive, predictive, and social network analyses. It presents case studies and examples of fraud detection across various industries while supporting the concept that fraud detection could be accomplished using data mining tools. It gives a broad approach to fraud detection while exhibiting a lack of relatedness to tax fraud, less concern in regulatory observance hence making it quite dependent on the availability of data. The success of any of the aforementioned techniques would probably depend on the availability and quality of the data; therefore, this could limit the modeling itself to be used [10]. This research article presents comparative analysis of supervised and unsupervised neural networks of 1,700 Korean firms over a period of 10 years where financial ratios and non-financial factors were taken as input variables for analysis. The study arrived at the conclusion that bankruptcy prediction can be done satisfactorily by both supervised and unsupervised neural networks. This highlights the usefulness of machine learning techniques in financial analyses. However, these systems raise certain legal and ethical issues [11].

Detection of financial fraud is a very great challenge that is characterized by the frequent occurrence of fraudulent practices. To support this statement, carrying a detailed analysis on 32 documents published from 2015 to 2020 has been done with respect to improvements in NN algorithms for fraud detection by a team of researchers. The study emphasized on DNN, CNN, SMOTE integrated NN and other complementary methods within ANN. The experiments with these were primarily focused on credit card fraud detection and indirectly making online transactions secure. The comparative study showed that these were convolutional ANN with functional sequencing, ANN integrated with Gradient Boosting Decision Tree (XGBoost) and ANN using automatic ontology learning that catered to theoretical foundation, mathematical momentum, experimental inquiry and result accuracy in equal measure. However, it must be pointed out that, in future research, if the aspects of epidemic, financial and characterizing data concerning neural network training are considered, such aspects would have an enormous impact on the efficiency of these algorithms [12]. Artificial neural networks are cheap and simple in the means they get the analysis done by avoiding statistical assumptions of matrix homogeneity, normality, and more data processing. These are self-tuning models, which are fault-tolerant. They can include all the available variables in model estimation and allow for speedy revisions. It was determined in the author's research that multilayer perceptron feeds neurons on fraudulent taxpayers and assesses the proportion of tax evaders with an efficiency of 84.3%. The analysis of sensitivity based on the ROC curve proved the model's good capability of separating fraudulent

from non-fraudulent taxpayers. The Multilayer Perceptron network offers a very effective opportunity to classify taxpayers, and the study's results provide avenues to enhance tax fraud detection algorithms to predict fraud tendencies using sensitivity analysis. Such concept exploration in relevant future taxes would be interesting [13]. "Online Virtual Classroom Application in Android Studio" focuses on how to develop an Android-based virtual classroom application with an emphasis on integrating various features and functionalities and creating efficient online learning. The actual implementation phases done in Android Studio are explained by the authors, and the need for such applications in support of remote education is justified. The article adds insightful contributions that help weigh the pros and cons of the online virtual classroom and is thus a valuable addition to the existing literature [14]. "Android Application: Skin Abnormality Analysis Based on Edge Detection Technique" addresses the application of edge detection for skin abnormality analysis on an Android-based system. The article evaluates merits and demerits of the mode of expression and emphasizes the prospect for detachment and diagnosis at the primary level [15].

3. PROPOSED METHOD

Here we give the methodology used by the project in which six different models are employed for the processing of research objectives. The models selected are Logistic Regression, Decision Tree, Random Forest, Naive Bayes, k-Nearest Neighbors, and Feed Forward Neural Network. This is important since each of them is key to digging deep into the dataset and getting the insights needed.

3.1 Logistic Regression (LR)

One of the frequently applied methods in machine learning deals with all forms of binary classification problems. The logistic function is used to calculate the probability of a particular outcome and, thus, shows the relationship between the input features and the output variable.

The logistic function can be expressed as:

$$P(y = 1|x) = \frac{1}{1+e^{-(b_0+b_1x_1+b_2x_2+\ldots.+b_nx_n)}} \quad (1)$$

Where:

$x_1, x_2, \ldots., x_n$ are the input features.

$b_0, b_1, b_2, \ldots., b_n$ are the co-efficient of the input features.

The natural logarithm base e is used in the expression.

An optimization procedure such as gradient descent or Newton-Raphson is utilized for calculating the coefficients; navigate through this one carefully. Usually, when the coefficients are known, the logistic function may be used to predict the probability that the outcome would apply for new observations. This is shown in Figure-2: logistic curve of probabilities as a function of features.

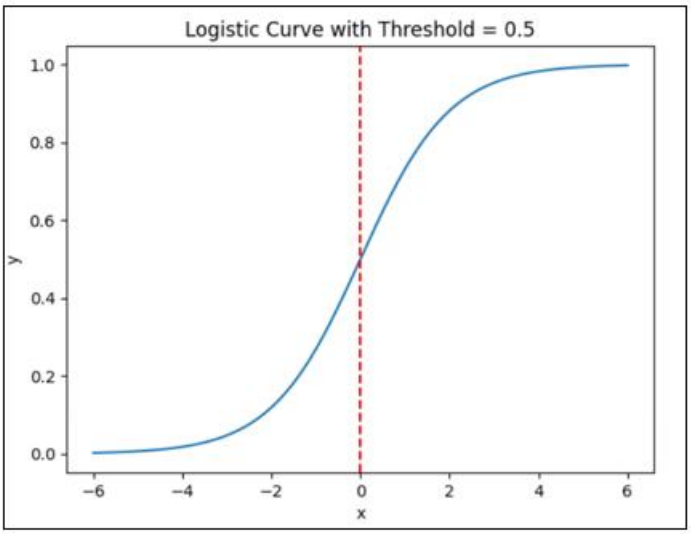

Figure-2. Logistic Curve.

The threshold for the assignment of an observation to either of its binary outcomes can be set to 0.5 (i.e., in this case, if P(y=1|x) is more than 0.5, then the observation is referred to as positive, while if it's less than 0.5, it is regarded as negative).

3.2 Decision Tree (DT)

The algorithm DT is widely used in machine learning for classification and regression problems. It has a unique structure with features used for internal nodes; branches are associated with decisions pertaining to features and leaf nodes are assigned classes or values. The process is iterative, dividing data sets into subsets to utilize features that are the most informative until a stopping criterion has been reached, like maximum depth or minimum samples per leaf.

To mathematically represent the decision tree, we can use the equation:

$$f(x) = \sum y_i \cdot I(x_i \in R_i) \quad (2)$$

Where:

$f(x)$denotes the predicted output for a new input x .

y_i represents the output value for the i^{th}leaf node.

R_i is the region of the i^{th} leaf node defined by the decision tree.

$I(x_i \in R_i)$ is an indicator function that returns 1 if the input x_i belongs to the region R_i, and 0 otherwise.

In other words, this equation calculates the predicted output by adding the output values of the leaf nodes whose regions the new input belongs to. The decision tree algorithm learns to partition the input space into regions where the output values are similar and assigns a unique output value to each region.

3.3 Random Forest (RT)

There is a type of machine learning algorithm that possesses the ability to handle both classification and regression tasks. This particular algorithm falls within the ensemble technique category, which involves combining multiple models to enhance the accuracy of predictions.

The functioning of the algorithm involves generating a set of decision trees using random subsets of the training data and features. Each tree within this collection is trained on a unique subset of the data. The final forecast is obtained by aggregating the predictions from all the trees, employing a majority voting process.

The RF algorithm

Randomly select 'n' samples from the training data set.

For each sample, randomly select k features from the total m features.

Build a decision tree using the selected samples and features.

Repeat steps 1-3 for T times to create T decision trees.

To make a prediction for a new data point, pass it through all T trees and calculate the average prediction from the majority votes.

The RF algorithm can be represented mathematically as follows:

Suppose we have a training dataset consisting of N observations with p input features and a corresponding set of target values $Y = \{y_1, y_2, \ldots, y_n\}$.

For each tree $t = 1, 2, \ldots, T$ in the forest, the algorithm selects a random subset of the training data D_t of size n (where $n < N$), and a random subset of the features F_t of size k (where $k < p$).

The decision tree is built, and a weight wi is assigned to the tree based on its performance on the out-of-the-bag samples (that is, the samples not used in building the tree).

To predict the new data point x, it is passed through each of the n trees, and their averaged prediction is given by:

$$Y(x) = \left(\frac{1}{T}\right) \cdot \sum_{t=1}^{T} w_t \cdot h_t(x) \quad (3)$$

Where:

$h_t(x)$ is tree t's prediction for input x

w_t is the weight given to tree t .

The Random Forest classifier handles high-dimensional datasets very well. It is robust to noise and outliers. Additionally, it is capable of revealing very useful insights, which is another rationale for feature selection and allow one to interpret the model.

3.4 Naive Bayes (NB)

The Naive Bayes algorithm is based on Bayes Theorem, a statistical principle that estimates how likely a hypothesis is given certain knowledge regarding circumstances related to the hypothesis. It is a simple yet so influential Naive Bayes classification rule.

Basically, based on a number of input features, compute the probability of each class. In Naive, the features are customarily taken to be independent of each other, which is not going to be so probable in practice. And yet, in actual life deals, the algorithm usually works rather nicely.

The following formula represents Naive Bayes:

$$P(y|x_1, x_2, \ldots, x_n) = \frac{P(y) \cdot P(x_1|y) \cdot P(x_2|y) \cdot \ldots \cdot P(x_n|y)}{P(x_1, x_2, \ldots, x_n)} \quad (4)$$

In this formula:

$P(y|x_1, x_2, \ldots, x_n)$ denotes the posterior probability of class y given the input features $x_1, x_2, \ldots, x_n$.

$P(y)$ represents the prior probability of class y .

$P(x_i|y)$ denotes the probability of feature x_i given class y

$P(x_1, x_2, \ldots, x_n)$ represents the probability of the input features.

The Naive Bayes algorithm

Calculate the prior probability $P(y)$ for each class given a collection of training data with labelled classes.

For each feature x_i calculate the conditional probability $P(x_i|y)$ for each class y .

For a new input instance with features $x_1, x_2, \ldots, x_n$ calculate the posterior probability $P(x_1, x_2, \ldots, x_n)$ for each class y using the above equation.

Allocate the input instance to the class whose posterior probability is the highest.

Verify that the denominator acts as a normalization constant in order that the probabilities sum one. It can be determined by adding the numerator across all potential classes:

$$P(x_1, x_2, \ldots, x_n) = \sum P(y) \cdot P(x_1|y) \cdot P(x_2|y) \cdot \ldots \cdot P(x_n|y) \; for \; all \; classes \; y.$$

3.5 K-Nearest Neighbors (K-NN)

The K-NN algorithm is a supervised learning algorithm that can be applied to classification and regression problems. The main goal of the K-NN algorithm is to label the test data point or value by looking for its k nearest data points in the training set.

Different metrics may be used to measure the distance in different ways between two data points, such as Manhattan distance, cosine similarity, or Euclidean distance. The algorithm may therefore predict the label or value of the test data point, based on the values or labels assigned by the k-nearest neighbors.

The equation for the K-NN algorithm can be expressed as follows:

For classification:

Let D represent the training dataset.

Let x be the test data point.

The number of neighbors to take consideration is k .

The distance metric between test point x & any point y in the dataset D is $dist(x, y)$.

Let neighbors (x) be the set of k nearest neighbors to x in D .

Let class (y) be the class label of y .

The following is the predicted class label for x :

predicted_class(x) = argmax(class(y)) for y in neighbors(x) (5)

In this equation, **argmax** returns the class label that occurs most frequently among the k nearest neighbors.

For regression:

Let D represent the training dataset.

Let x be the test data point.

The number of neighbors to take consideration is k .

The distance metric between test point x & any point y in the dataset D is $dist(x, y)$.

Let neighbors (x) be the set of k nearest neighbors to x in D .

Let class (y) be the class label of y .

Then, the predicted value for x is:

predicted_value(x) = mean(value(y)) for y in neighbors(x) (6)

In this equation, **mean** returns the average value of the k nearest neighbors.

3.6 Feed Forward Neural Network (FFNN)

An FFNN is an example of an artificial neural network structured to pass information through a set of hidden layers in a unidirectional manner from the input layer to the output layer. Feedback loops are absent in the FFNN, with the input being forwarded through the network until it comes to the output layer.

The FFNN is structured around its fundamental units, called neurons or perceptrons. The units linearly transform the input signals, apply an activation function, and pass the output to the next layer. The neurons within the different layers of the FFNN are interconnected with neurons of the preceding and succeeding layers.

That first layer, which carries raw input data, henceforth is designated as the input layer; this forwards signals to the topmost hidden layer. Signals are propagated to the hidden layers, which perform a linear transformation, applied activation function, and signal are forwarded to the last layer. The last layer, however, for the entire system, generates the output.

The equation for the output of a single neuron In an FFNN is as follows:

$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$ (7)

Where:

x_i is the input to the neuron from the previous layer or the input layer.

w_i is the weight of the connection between the input x_i and the neuron.

b is the bias term, which is added to shift the output of the neuron.

$\sum_{i=1}^{n} w_i x_i + b$ represents a weighted sum of the inputs and bias.

f is the activation function, which introduces non-linearity into the output of the neuron.

The feedforward neural network (FFNN) activation function introduces non-linearity to the network. Most common activation functions include ReLU, sigmoid, softmax, etc. The choice of an activation function is based on the problem and the type of output required.

The output of an FFNN could then be derived by summing the equations related to all neurons available. During training, the neuron weights and biases are learned via an optimizing algorithm, such as backpropagation. The functional model of the proposed system is shown in Figure -3.

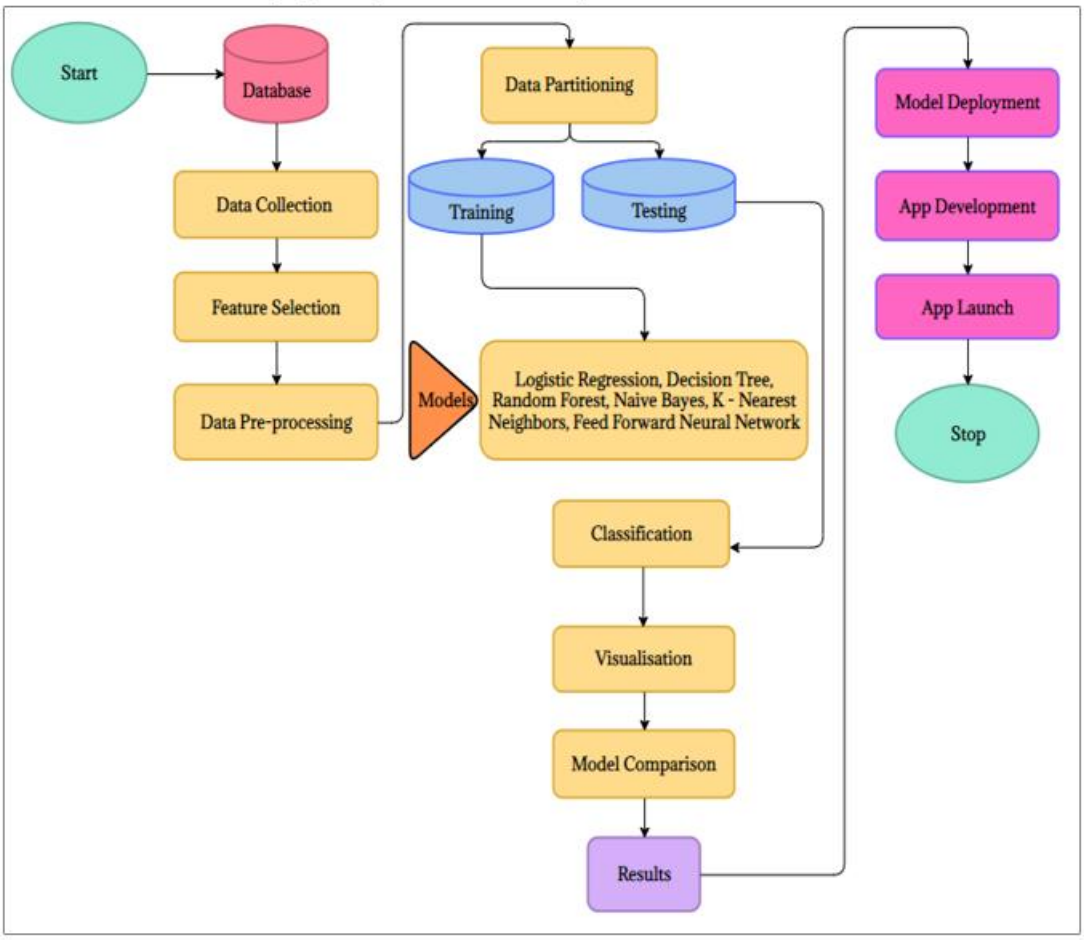

Figure-3. Architecture of the Proposed Model.

Data Collection: Models developed to detect income tax fraud were interpreted on the basis of data input for the construction of indicators of parameters such as age, workClass, fnlwgt, education, educationNum, maritalStatus, occupation, relationship, race, sex, capitalGain, capitalLoss, hoursPerWeek, nativeCountry, and income.

Feature Selection Various methodologies like correlational studies, feature importance, or domain knowledge were made to ensure the selection of important variables for income tax fraud detection. Selected features would be in accordance with the presented study in such a way as to involve demographics: age, workClass, race, sex, nativeCountry, income; and factors related to behavior: hoursPerWeek, education, maritalStatus, occupation, relationship- would constitute other features.

Data Pre-processing: The data was pre-processed by removing missing values, outliers, and irrelevant variables. Additionally, feature scaling, normalization, and encoding categorical variables were performed.

Data Partitioning: The dataset was separated into training and testing sets to train the models and test their performance.

Model Training: Six different models, including Logistic Regression, Decision Tree, Random Forest, Naive Bayes, k-Nearest Neighbors, and Feed Forward Neural Network, were trained on the selected features using the training data. These models learned to predict the likelihood of income tax fraud based on the input variables.

Model Evaluation: The performance of the tested models on the test data was done primarily by other metrics in exploiting the full potential of accuracy and further based on precision, recall, and f1 score to examine how satisfactorily well that particular model is working in detecting income tax fraud.

Visualization: Visual resulted plots are here aiming at a comparison of different algorithms in use. Bar plots serve to compare accuracy scores of various models, ROC graphical representations show which model would work better with an area under the curve (AUC), while combinations of precision-recall curves of models merely compare precision against recall.

Model Comparison: The performance of each algorithm was compared to determine which algorithm performs the best.

Model Deployment: The best model is implemented in Android Studio using the TensorFlow library.

Application Development: The user interface is built using App view. The implementation that we have implies its building on the three XML layout files that define the user interface. In the interaction with this corresponds to its design. The app controller also serves as an intermediate unit between Model and View elements, controlling the flow of data and event handling. In addition, might say some points on that thing. Maybe this point of you should emphasize more points on that issue.

App Launch: Starting this application means compilation of sources with resources into APK. Starting it after installation allocates an instance of the main activity on the screen with an Android way.

4. RESULTS AND DISCUSSION

The results from the computation performed to detect fraud have been discussed in detail here. The input data used here was downloaded from OpenML-a common platform to share datasets. It consists of 48842 entries with 15 columns. To check the correlation between the variables and visualize the distributions of variables, exploratory data analysis was performed. Some of the representative plots included a scatterplot matrix of six rows and six columns to depict the relationship between two variables in each scatterplot. The plots on the diagonal showed the distribution of each variable, while the plots above the diagonal were the same as those below the diagonal. The hue value was used to represent data points based on income level, which allowed for identification of any patterns or differences between the two classes. The resulting diagram is represented Figure-4 below.

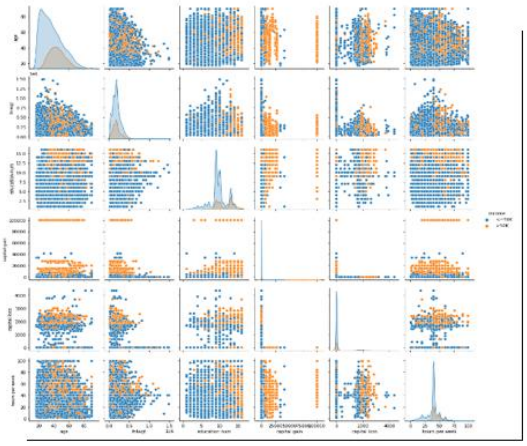

Figure-4. Exploratory Data Analysis.

This involved numerous preprocessing steps like duplicate removal, missing value treatment, categorical variables encoding, and continuous variables scaling. After that, six machine learning models were trained on the training data and evaluated according to various metrics, such as accuracy, precision, recall, and F1-score, which lend themselves to a fair model comparison. The following visualizations were then created:

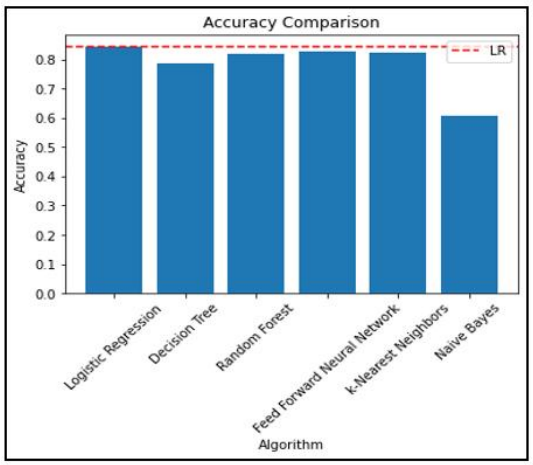

Figure-5. Accuracy Bar Plot.

Bar Plot: A comparison of the accuracy of each model was made, and the results are visualized in Figure-5. The accuracy

of the logistic regression model was represented by a red horizontal line, which served as a reference point for comparison.

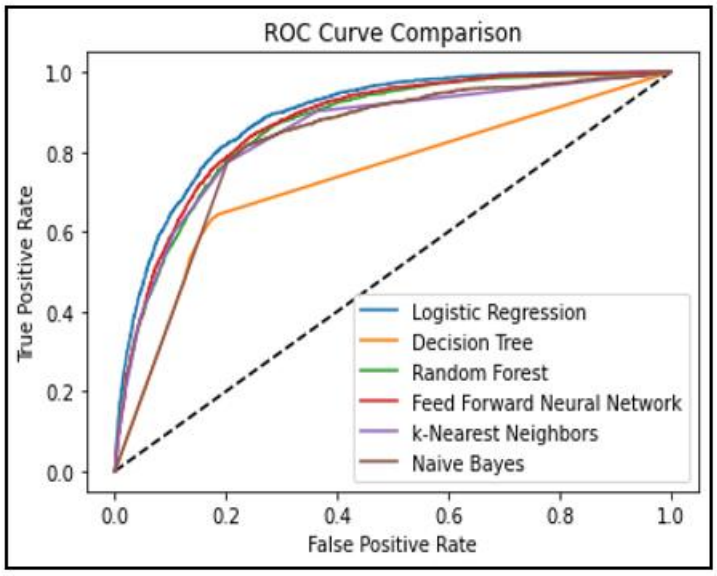

Figure-6. ROC Curve.

A plot of the **Receiver Operating Characteristic (ROC) Curve** was made for each algorithm using different colors for each model, as shown in Figure-6. This displays the tradeoff between sensitivity and specificity for each method, allowing them to be compared.

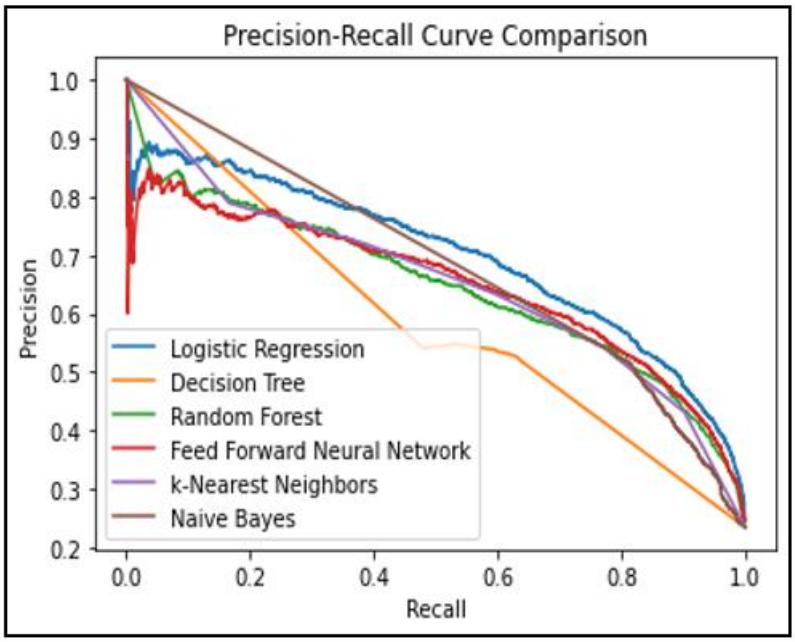

Figure-7. Precision-Recall Curve.

Precision-Recall Curve: A plot of the precision-recall curves was made for each algorithm, see Figure-7. This graph depicts the link between precision and recall for each method.

After comparing the performance of various algorithms, cross-validation was performed to reduce the impact of chance and sampling bias that may occur in a single train-test split. The output displayed in Table-1, presents the name of each model and its mean cross-validation score over 10 folds, which provides a measure of the generalization performance of each model. It indicates the amount of performance of every model likely on new and unseen data.

Table-1. Cross Validation Table.

Models	**Mean cross–validation score**
Logistic Regression	0.8346172945234638
Decision Tree	0.7825865720373295
Random Forest	0.817162880366683
Naive Bayes	0.605046495559648
K-Nearest Neighbors	0.8182374679046269
Feed Forword Neural Network	0.81933381089868617

After that, using the GridSearchCV method from scikit-learn, we optimize the performance of the best performing model with hyperparameters. The hyperparameters being tuned are the regularization strength parameter "C" of the logistic regression algorithm. After the hyperparameters are tuned, the best estimator model is selected based on the average score obtained across all folds of the cross-validation. The logistic regression model's prediction probabilities were used to identify suspicious fraudsters based on a threshold value of 0.35, and 2769 records were recognized as suspicious of committing income tax fraud.

We compare the performance of all models using evaluation metrics, and the results are summarized in Table-2.

Table-2. Model Comparison Table.

Models	**Evaluation metrics**			
	Accuracy	Precision	Recall	F1-Score
Logistic Regression	**0.832973**	**0.703883**	**0.569869**	**0.629826**
Decision Tree	**0.787184**	**0.547077**	**0.535371**	**0.541161**
Random Forest	**0.820248**	**0.629990**	**0.565066**	**0.595764**
Naive Bayes	**0.608660**	**0.366626**	**0.920087**	**0.524325**
K-Nearest Neighbors	**0.824445**	**0.632917**	**0.597817**	**0.614866**
Feed Forword Neural Network	**0.825366**	**0.627178**	**0.628821**	**0.62799**

After completing the analysis, the best performing model, which is logistic regression model was converted into a TensorFlow lite format and deployed into an android studio to build a prediction app using the same inputs as the dataset used in ML models building.

The research aims to develop a mobile phone application for income tax fraud detection, considering the widespread usage of Android and iOS as mobile operating system platforms. Given that Android powers a significant number of affordable smartphones utilized in rural areas, the study focuses on utilizing the Android operating system as the framework for application development.

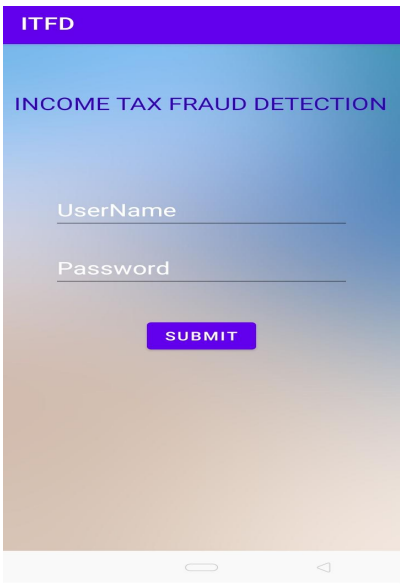

Figure-8a

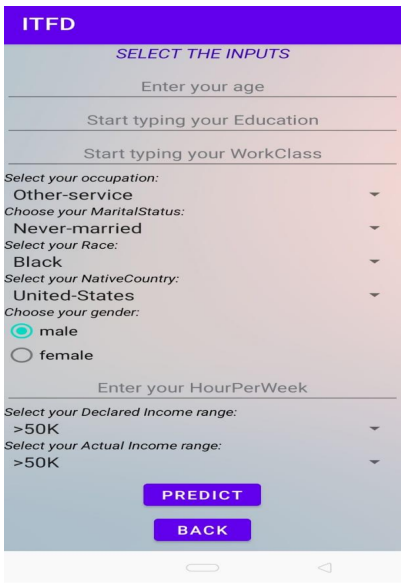

Figure-8b

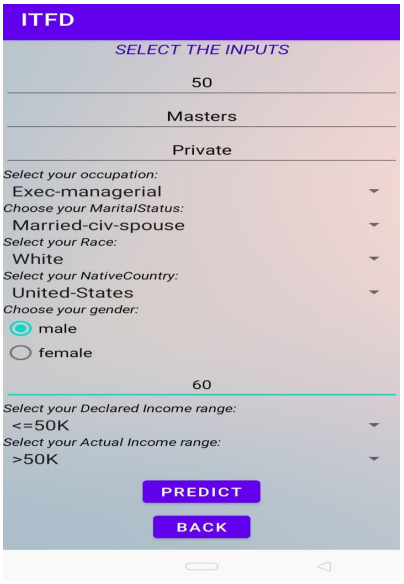

Figure-8c

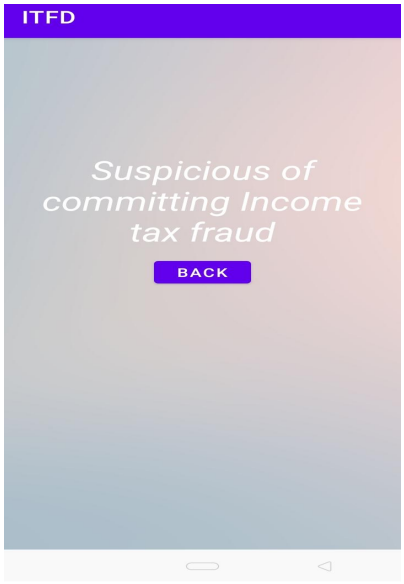

Figure-8d

Figure-8a. Login Screen. **Figure-8b. Input Screen Before Selecting the Inputs.**

Figure-8c. Input Screen After Selecting the Inputs. **Figure-8d. Display Screen.**

Figure-8a. shows the Login Screen, which is the first interface of our app. The users must provide their name and password to enter the second interface. The input screen is shown in Figure-8b. and it requests the user to enter 11 inputs, including their age, education, work class, occupation, marital status, race, gender, native country, hour per week, declared and actual income by using the edit text, auto complete text view, radio button and spinner control and the Figure-8c. represent the screen after selecting the inputs. The app controller will then analyze the inputs and provide a predicted result. The third interface is displayed in Figure-8d. retrieves the predicted outcome from the inputs interface and displays it in text view.

5. CONCLUSION

The best-performing model among those considered was the logistic regression model with hyperparameters optimized using grid search. This model achieved an accuracy of 0.8429, precision of 0.7038, recall of 0.5698, and F1-score of 0.5698. The ROC and Precision-Recall curves show that the logistic regression model offers an optimal trade-off between true positive rates and false positive rates, along with precision and recall, when compared to other models. Cross-validation finds that these models manifest consistent performance across folds, with logistic regression obtaining the highest mean score for cross-validation. Conclusively, our project lays a bedrock for building and comparing different machine learning models aimed at detecting income tax fraud. The logistic regression model with optimal hyperparameter, in fact, emerged as the best performer. The created application compares the declared income and the actual income, then uses the deployed model to make predictions and shows its results. As a result, we have developed an accurate and efficient way to find a person suspicious of committing income tax fraud, which can aid in financial planning, market research, and other areas.

REFERENCES

[1] Murorunkwere, B.F., Tuyishimire, O., Haughton, D., Nzabanita, J., "Fraud Detection Using Neural Networks: A Case Study of Income Tax", Future Internet 2022, 14, 168.

[2] Pérez López, C., Delgado Rodríguez, M.J., de Lucas Santos, S. "Tax Fraud Detection through Neural Networks: An Application Using a Sample of Personal Income Taxpayers", Future Internet 2019, 11, 86.

[3] M. S. Rad and A. Shahbahrami, "Detecting high risk taxpayers using data mining techniques", 2016 2nd International Conference of Signal Processing and Intelligent Systems (ICSPIS), Tehran, Iran, 2016, pp. 1-5.

[4] Mojahedi, Houri & Babazadeh sangar, Amin & Masdari, Mohammad. (2022). "Towards Tax Evasion Detection Using Improved Particle Swarm Optimization Algorithm", Mathematical Problems in Engineering. 2022. 1-17.

[5] de Roux, D., Perez, B., Moreno, A., Villamil, M.D.P., Figueroa, C. "Tax fraud detection for under-reporting declarations using an unsupervised machine learning approach", In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 215–222

[6] Miloš Savić, Jasna Atanasijević, Dušan Jakovetić, Nataša Krejić, "Tax evasion risk management using a Hybrid Unsupervised Outlier Detection method", Expert Systems with Applications, Volume 193, 2022, 116409, ISSN 0957-4174.

[7] González, P.C., Velásquez, J.D, "Characterization and detection of taxpayers with false invoices using data mining techniques", Expert Syst. Appl. 2013, 40, 1427–1436.

[8] Ghosh, S., Douglas, L.R, "Credit card fraud detection with a neural-network", In Proceedings of the Twenty-Seventh Hawaii International Conference, Wailea, HI, USA, 4–7 January 1994.

[9] Chi-Hung Lin, I-Chun Lin, Ching-Huei Wu, Ya-Ching Yang & Jinsheng Roan (2012) "The application of decision tree and artificial neural network to income tax audit: the examples of profit-seeking enterprise income tax and individual income tax in Taiwan", Journal of the Chinese Institute of Engineers, 35:4, 401-41.

[10] B. Baesens, V. Vlasselaer, W. Verbeke. "Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques", Wiley, 2015.

[11] Lee, Kidong, David E. Booth and Pervaiz Alam. "A comparison of supervised and unsupervised neural networks in predicting bankruptcy of Korean firms", Expert Syst. Appl. 29 (2005): 1-16.

[12] Clavería Navarrete, A. y Carrasco Gallego, A. (2021). "Neural network algorithms for fraud detection: a comparison of the complementary techniques in the last five years", Journal of Management Information and Decision Sciences, 24 (special 1), 1-16.

[13] Pérez López, César, María Jesús Delgado Rodríguez, and Sonia de Lucas Santos. 2019. "Tax Fraud Detection through Neural Networks: An Application Using a Sample of Personal Income Taxpayers", Future Internet 11, no. 4: 86.

[14] D. Buddhi, R. Singh and A. Gehlot, "Online Virtual Classroom Application In Android Studio," 2022

International Interdisciplinary Humanitarian Conference for Sustainability (IIHC), Bengaluru, India, 2022, pp. 960-963.

[15] Z. Zulfikar, Z. Zulhelmi, T. Y. Arif, A. Afdhal and P. N. Syawaldi, “Android Application: Skin Abnormality Analysis based on Edge Detection Technique,” 2018 International Conference on Electrical Engineering and Informatics (ICELTICs), Banda Aceh, Indonesia, 2018, pp. 89-94.

..