An Industry-Oriented Mini Project Report

On

"Credit Card Fraud Detection using State of Art Machine Learning"

Submitted in Partial Fulfillment of the Academic Requirement for the Award of Degree

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering (Artificial Intelligence and Machine learning)

Submitted By:

R. THARUN KUMAR	22R01A6651
T. VARSHITH GOUD	22R01A6661
V. JAYA PRAKASH	22R01A6663
G. SHIVA SAI	22R01A6684

Under the esteemed guidance of

Dr. Y. Nagesh

CMR INSTITUTE OF TECHNOLOGY

(UGCAUTONOMOUS)

Approved by AICTE, Affiliated to JNTUH, Accredited by NAAC with A+ Grade,

Kandlakoya (V), Medchal Dist - 501 401

www.cmrithyderabad.edu.in

2024-25

CMR INSTITUTE OF TECHNOLOGY

(UGCAUTONOMOUS)

Approved by AICTE, to JNTUH, accredited by NAAC with A+ Grade,

Kandlakoya(V), Medchal Dist-501 401

www.cmrithyderabad.edu.in

.

CERTIFICATE

This is to certify that an Industry oriented Mini Project entitled with "**Credit Card Fraud Detection Using State of Art Machine Learning**" is being submitted by:

R. THARUN KUMAR	**22R01A6651**
T. VARSHITH GOUD	**22R01A6661**
V. JAYA PRAKASH	**22R01A6663**
G. SHIVA SAI	**22R01A6684**

To JNTUH, Hyderabad, in partial fulfillment of the requirement for award of the degree of B. Tech in CSE (AI&ML) and is a record of a Bonafide work carried out under our guidance and supervision. The results in this project have been verified and are found to be satisfactory. The results embodied in this work have not been submitted to have any other University forward of any other degree or diploma.

Signature of Guide **Signature of Project Coordinator** **Signature of HOD**

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are extremely grateful to our Director, Principal and Head of Department, Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to**,** Industry Oriented Mini Project Coordinator and internal guide Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), CMR Institute of Technology and our external team from the industry for their constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We express our thanks to all staff members and friends for all the help and coordination extended in bringing out this Project successfully in time.

Finally, we are very much thankful to our parents and relatives who guided directly or indirectly for every step towards success.

Team Members:--

R THARUN KUMAR
T VARSHITH GOUD
V JAYA PRAKASH
G SHIVA SAI

ABSTRACT

With the rapid growth of digital payment systems, credit card transactions have become a preferred method for online purchases. However, this convenience has also led to a significant rise in fraudulent activities, posing a major challenge to both consumers and financial institutions. Detecting fraudulent transactions in real-time while minimizing false positives remains a critical issue due to the complex and evolving nature of fraud patterns. Our project addresses this problem by implementing state-of-the-art machine learning algorithms to identify and mitigate credit card fraud effectively.

The project utilizes a highly imbalanced, real-world dataset that simulates genuine and fraudulent transactions. Multiple machine learning models, including Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), and XGBoost, were employed to detect anomalies and suspicious patterns in the transaction data. Each model was trained and tested using appropriate preprocessing steps such as data normalization and class balancing techniques like SMOTE. Performance metrics such as accuracy, precision, recall, F1-score, and AUC-ROC were used to evaluate the models. Among these, ensemble-based methods like Random Forest and XGBoost demonstrated superior performance in handling class imbalance and reducing false alarms.

This research highlights the potential of machine learning models in real-time fraud detection systems. By carefully tuning hyperparameters and optimizing feature selection, our models achieved high detection accuracy while maintaining low false positive rates. The results confirm that machine learning offers a scalable, efficient, and adaptive approach to combating credit card fraud, paving the way for more secure digital financial systems.

INDEX

LIST OF FIGURES

LIST OF SCREENSHOTS

1. INRODUCTION

1.1 ABOUT PROJECT

Credit Card Fraud Detection Using State-of-the-Art Machine Learning is a project aimed at addressing the growing challenge of fraudulent credit card transactions in the digital economy. As the number of online financial transactions increases, so does the risk of credit card misuse and unauthorized access. This project implements machine learning algorithms to identify suspicious activities and distinguish between legitimate and fraudulent transactions. The goal is to improve the accuracy of fraud detection systems while reducing the rate of false alarms, which are often encountered in conventional methods.

The project starts with a thorough analysis of a real-world, highly imbalanced dataset, representing authentic and fraudulent credit card transactions. To ensure reliable model performance, preprocessing techniques such as normalization and handling missing values were applied. Various machine learning models, including Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), and XGBoost, were implemented. These models were chosen for their ability to learn complex patterns and make real-time predictions. Evaluation metrics such as precision, recall, F1-score, and ROC-AUC were used to compare model effectiveness. Among these, ensemble models like Random Forest and XGBoost provided the most accurate results in detecting fraudulent transactions.

The increasing shift toward digital and cashless transactions has brought convenience but also increased exposure to financial fraud, particularly credit card fraud. Credit card misuse not only results in significant monetary losses but also undermines trust in digital banking systems. Our project, “Credit Card Fraud Detection Using State-of-the-Art Machine Learning”, is developed to address these challenges by leveraging powerful machine learning algorithms to detect and mitigate fraudulent transactions in real-time. With growing transaction volumes and constantly evolving fraudulent behavior, traditional rule-based systems often fall short. Thus, intelligent systems that can learn patterns from data and adapt dynamically are necessary for reliable fraud detection.

The foundation of this project lies in applying machine learning techniques to a real-world dataset that contains anonymized credit card transaction records. One of the major issues tackled in this project is the class imbalance problem, where fraudulent transactions represent a very small fraction of the dataset. This imbalance leads to challenges in model accuracy and generalization. To address this, techniques such as data resampling, normalization, and the Synthetic Minority Over-sampling Technique (SMOTE) are employed. These help ensure that the learning algorithms have balanced exposure to both fraudulent and legitimate transactions. Preprocessing also includes handling missing values, feature scaling, and splitting the dataset into training and testing sets to validate performance across different models.

Multiple state-of-the-art machine learning algorithms were implemented and evaluated, including Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), and XGBoost. These models were selected based on their ability to handle large datasets, uncover nonlinear patterns, and provide robust predictions even under uncertainty. Ensemble learning methods like Random Forest and XGBoost were particularly effective, as they combine multiple weak learners to form a strong overall model. Each algorithm underwent rigorous training using optimized parameters and was evaluated using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC curves. The results indicated that machine learning models, when properly tuned and validated, significantly enhance the ability to detect fraudulent behavior with minimal false positives.

To streamline the development and deployment process, the project was structured into key modules: Data Preprocessing, Model Building, Model Evaluation, and Fraud Detection System Interface. In the Data Preprocessing module, the input transaction data is cleaned, transformed, and balanced. The Model Building module is responsible for training various machine learning models and storing their configurations. The Evaluation module tests these models across several performance criteria to determine the most effective one. Finally, the Fraud Detection System Interface integrates the best-performing model into a user-friendly interface that can be applied in real-time scenarios. This modular architecture ensures scalability, flexibility, and ease of maintenance in real-world applications.

This project showcases how state-of-the-art machine learning methods can be applied to the domain of credit card fraud detection. It not only achieves high levels of accuracy but also demonstrates efficiency in identifying fraudulent transactions with minimal human intervention. By continually training the models with new data, the system remains adaptive to emerging fraud patterns. The implementation of such intelligent systems can revolutionize the way financial institutions manage fraud, reduce economic losses, and build stronger customer trust. As future work, the system can be extended with deep learning models and real-time deployment on cloud platforms to further enhance performance and usability.

1.2 EXISTING SYSTEM

The existing systems used for credit card fraud detection are primarily based on rule-based or traditional statistical methods. These systems rely on predefined patterns, thresholds, or human-defined rules to flag potentially fraudulent transactions. For example, if a user suddenly makes a large transaction in a different country, the system may flag this activity as suspicious. While these methods provide a basic level of protection, they are often rigid and unable to adapt to new, evolving fraud tactics. They generate a high number of false positives, where genuine transactions are wrongly classified as fraudulent, leading to customer dissatisfaction and financial service disruptions.

One of the major limitations of these traditional systems is their **inflexibility** and lack of learning capability. Since these systems are hardcoded with fixed rules, they cannot dynamically adapt to changing fraud strategies. Fraudsters continuously evolve their techniques to bypass existing safeguards. For instance, by mimicking normal spending behavior, fraudsters can evade detection by systems that only monitor for unusual or abrupt behavior. Moreover, maintaining and updating rule-based systems is time-consuming and labor-intensive. It requires frequent manual intervention and domain expertise to ensure the rules remain relevant, which increases operational costs and limits scalability.

Another significant issue with the existing system is the **inability to handle large-scale, imbalanced datasets** effectively. Credit card fraud datasets are typically imbalanced, where fraudulent transactions form a very small portion compared to legitimate ones. Traditional systems do not perform well under such circumstances because they tend to be biased toward the majority class, resulting in missed fraud detection. Additionally, these systems lack the capability to detect complex patterns or relationships between transaction attributes that may indicate fraudulent behavior. This leads to both under-detection and over-detection problems, reducing the overall effectiveness of fraud prevention strategies.

Despite the use of some basic statistical tools and alert systems, most existing methods are **reactive rather than proactive**. They often detect fraud only after it has occurred, relying on customer complaints or post-transaction audits to identify suspicious behavior. This results in financial losses and erodes customer trust. Furthermore, these systems are not integrated with intelligent feedback mechanisms that allow continuous improvement. Once a fraud case is detected, the system does not learn from it to improve future detection. In an environment where cyber threats are becoming more sophisticated, this reactive nature is a severe drawback that can lead to significant exposure.

The current state of fraud detection systems is marked by significant limitations in adaptability, scalability, and accuracy. Rule-based models lack the intelligence and flexibility needed to counter modern fraud techniques. Their reliance on static patterns makes them vulnerable to evolving threats, while their inefficiency in processing large and imbalanced data limits their predictive power. This project aims to overcome these challenges by leveraging **machine learning algorithms**, which offer the ability to learn from data, detect hidden patterns, and adapt over time. By replacing outdated methodologies with intelligent systems, financial institutions can ensure faster, more accurate, and more efficient fraud detection, ultimately protecting both customers and organizations from financial harm.

1.3 PROPOSED SYSTEM

The proposed system introduces an intelligent and automated approach to credit card fraud detection by utilizing advanced machine learning algorithms. Unlike traditional rule-based systems that rely on static thresholds and manual configurations, this system is dynamic, data-driven, and capable of learning complex patterns from historical transaction data. It leverages the power of supervised learning techniques such as Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), and XGBoost to distinguish between genuine and fraudulent transactions with high accuracy. These models are trained on a real-world dataset that reflects a high degree of class imbalance—where fraudulent cases are significantly outnumbered by legitimate ones—allowing the system to perform well even in challenging data conditions.

The architecture of the proposed system includes several integrated modules that collectively enhance fraud detection. First, the **Data Preprocessing Module** cleans, normalizes, and balances the dataset using techniques like SMOTE (Synthetic Minority Over-sampling Technique). Next, the **Model Training and Evaluation Module** builds, tunes, and evaluates multiple machine learning models based on precision, recall, F1-score, and ROC-AUC metrics. The best-performing models are selected and saved for deployment. Finally, the **Prediction and Detection Module** accepts incoming transaction data, processes it in real-time, and flags potentially fraudulent transactions with minimal delay. This end-to-end pipeline ensures that the system is efficient, accurate, and responsive to evolving fraud patterns.

This proposed system is not only more accurate but also scalable and adaptive. It learns from past transactions and can be updated regularly with new data to improve performance over time. By reducing false positives and detecting subtle anomalies, it enhances the security of financial systems while maintaining a smooth user experience for genuine customers. The machine learning-based approach represents a significant advancement in fraud detection technology, offering proactive and predictive capabilities that were previously unattainable.

2. REQUIREMENT SPECIFICATION

Software Requirements

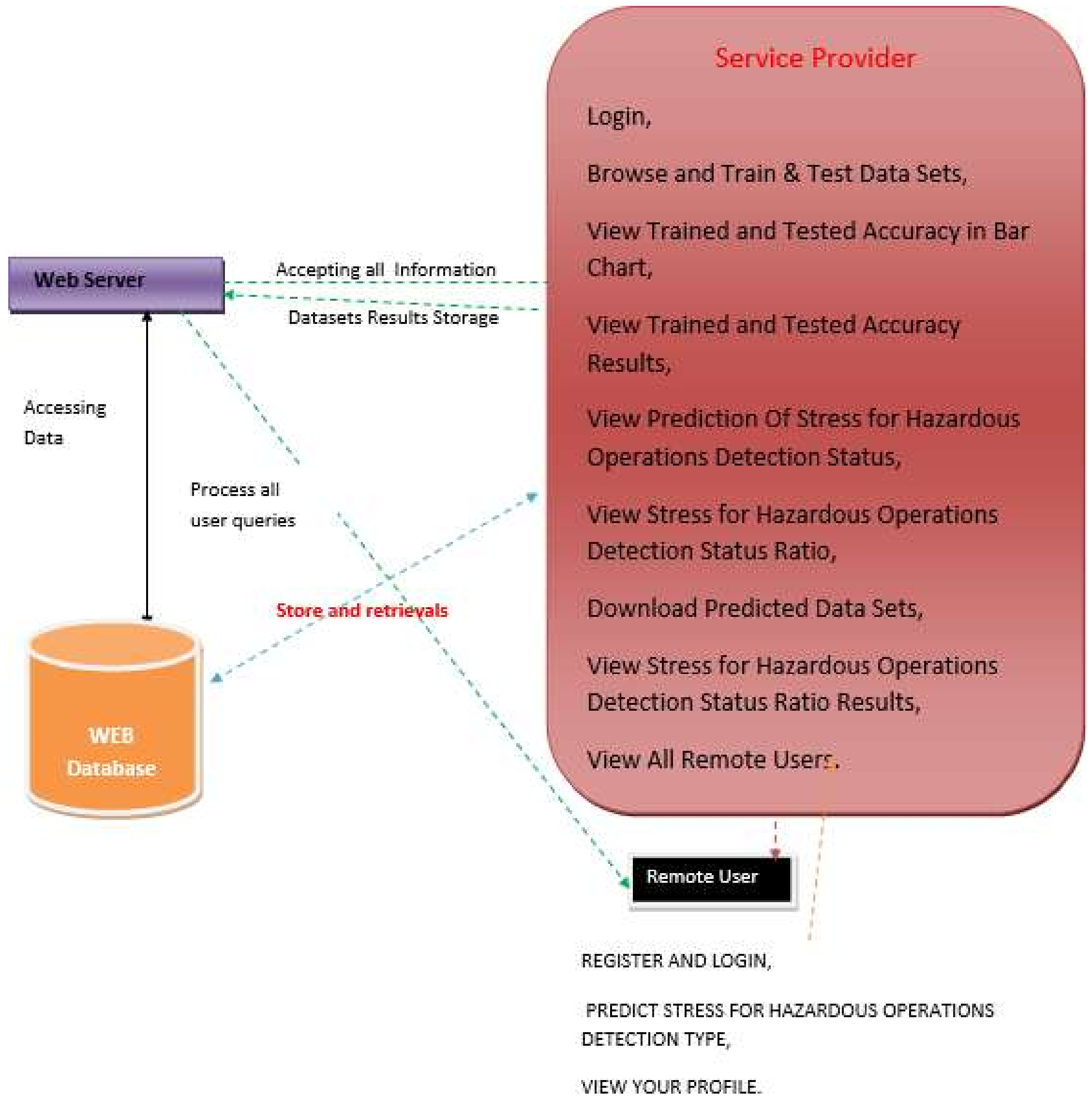
- **Operating System:** Windows 10 / Linux (Ubuntu 20.04 or later) / macOS
- **Programming Language:** Python 3.8 or later
- **Development Environment:** Jupyter Notebook / VS Code / PyCharm
- **Libraries & Frameworks:**
  - NumPy, Pandas (for data manipulation)
  - Scikit-learn (for machine learning algorithms)
  - XGBoost (for gradient boosting)
  - Matplotlib, Seaborn (for data visualization)
  - Imbalanced-learn (for handling class imbalance with SMOTE)
- **Database (Optional):** SQLite / MongoDB (if persistent storage is required)
- **Version Control:** Git with GitHub (for collaboration and versioning)

Hardware Requirements

- **Processor:** Intel i5 (8th Gen or higher) / AMD Ryzen 5 or better
- **RAM:** Minimum 8 GB (16 GB recommended for faster model training)
- **Storage:** Minimum 256 GB SSD (for better read/write performance)
- **Graphics Card (Optional):** NVIDIA GPU (if using advanced deep learning models in future)
- **Display:** Standard HD Monitor (1080p or better for visualization tasks)

2.SYSTEM DESIGN

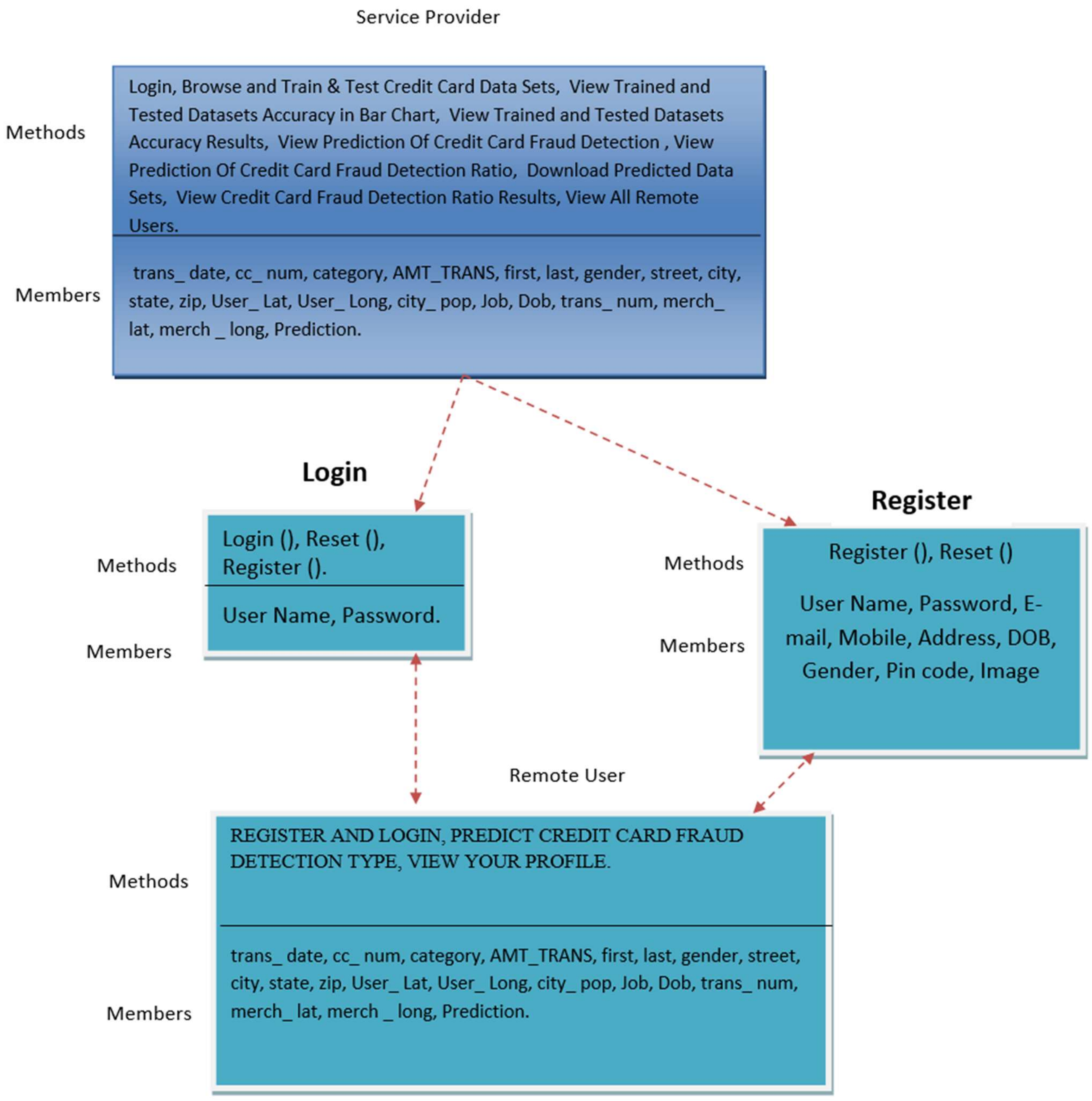
3.1 ARCHITECTURE FOR REAL-TIME FRAUD DETECTION



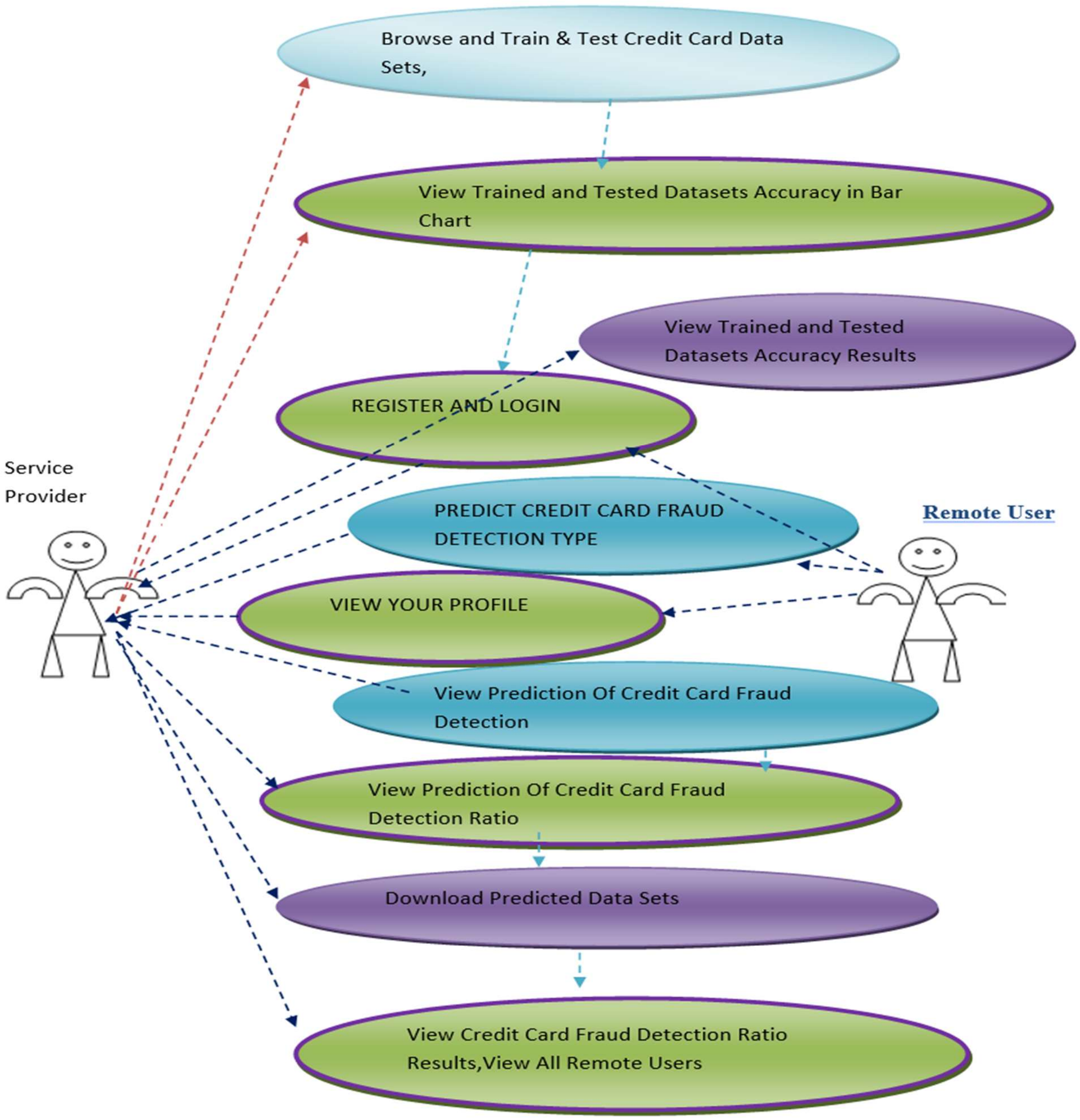
Service Provider
Login,
Browse and Train & Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Prediction Of Stress for Hazardous Operations Detection Status,
View Stress for Hazardous Operations Detection Status Ratio,
Download Predicted Data Sets,
View Stress for Hazardous Operations Detection Status Ratio Results,
View All Remote Users.
Web Server
Accepting all Information
Datasets Results Storage
Accessing Data
Process all user queries
Store and retrievals
WEB Database
Remote User
REGISTER AND LOGIN,
PREDICT STRESS FOR HAZARDOUS OPERATIONS DETECTION TYPE,
VIEW YOUR PROFILE.

3.2 UML DIAGRAMS

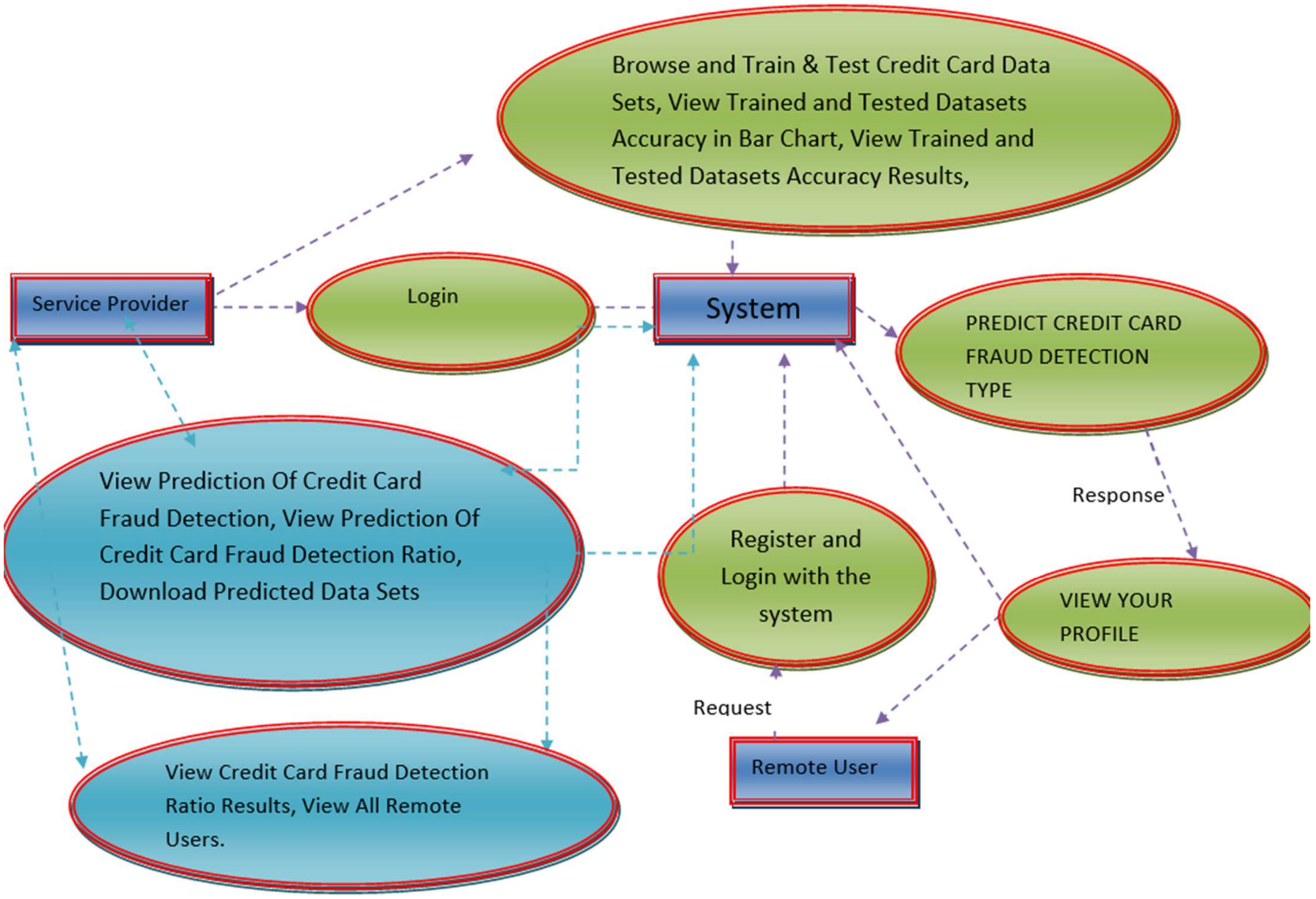
3.2.1 CLASS DIGARAM REPRESENTING SERVICE PROVIDER MODULES


Service Provider
Methods
Login, Browse and Train & Test Credit Card Data Sets, View Trained and Tested Datasets Accuracy in Bar Chart, View Trained and Tested Datasets Accuracy Results, View Prediction Of Credit Card Fraud Detection , View Prediction Of Credit Card Fraud Detection Ratio, Download Predicted Data Sets, View Credit Card Fraud Detection Ratio Results, View All Remote Users.
Members
trans_ date, cc_ num, category, AMT_TRANS, first, last, gender, street, city, state, zip, User_ Lat, User_ Long, city_ pop, Job, Dob, trans_ num, merch_ lat, merch _ long, Prediction.
Login
Methods
Login (), Reset (), Register ().
Members
User Name, Password.
Register
Methods
Register (), Reset ()
Members
User Name, Password, E-mail, Mobile, Address, DOB, Gender, Pin code, Image
Remote User
Methods
REGISTER AND LOGIN, PREDICT CREDIT CARD FRAUD DETECTION TYPE, VIEW YOUR PROFILE.
Members
trans_ date, cc_ num, category, AMT_TRANS, first, last, gender, street, city, state, zip, User_ Lat, User_ Long, city_ pop, Job, Dob, trans_ num, merch_ lat, merch _ long, Prediction.

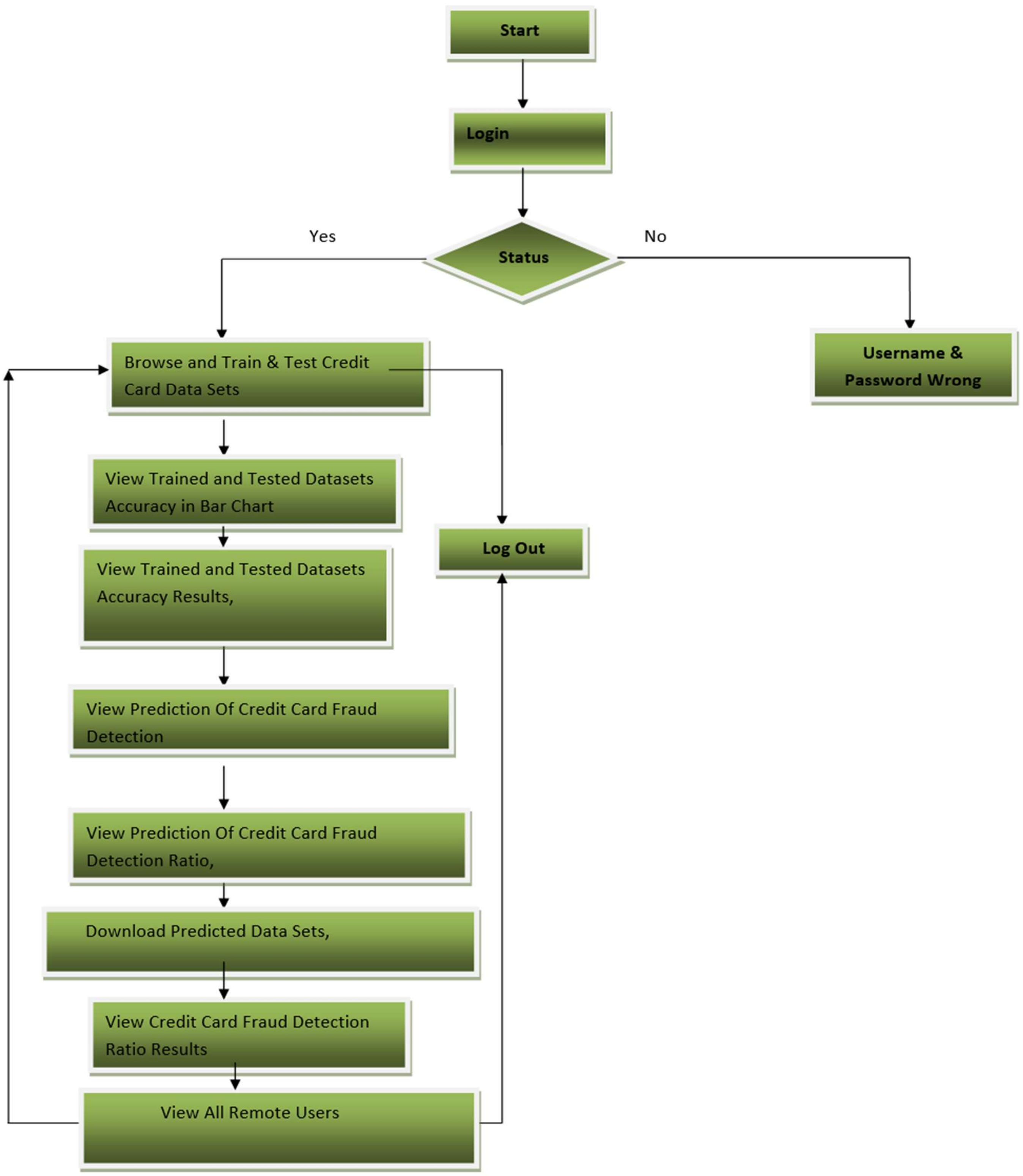
3.2.2 USE CASE DIAGRAM DEPICTING SYSTEM INTERACTIONS


Browse and Train & Test Credit Card Data Sets,
View Trained and Tested Datasets Accuracy in Bar Chart
View Trained and Tested Datasets Accuracy Results
REGISTER AND LOGIN
Service Provider
PREDICT CREDIT CARD FRAUD DETECTION TYPE
Remote User
VIEW YOUR PROFILE
View Prediction Of Credit Card Fraud Detection
View Prediction Of Credit Card Fraud Detection Ratio
Download Predicted Data Sets
View Credit Card Fraud Detection Ratio Results,View All Remote Users

3.2.3 DATA FLOW DIAGRAM OF SERIVCE PROVIDER FUNCTIONALITIES

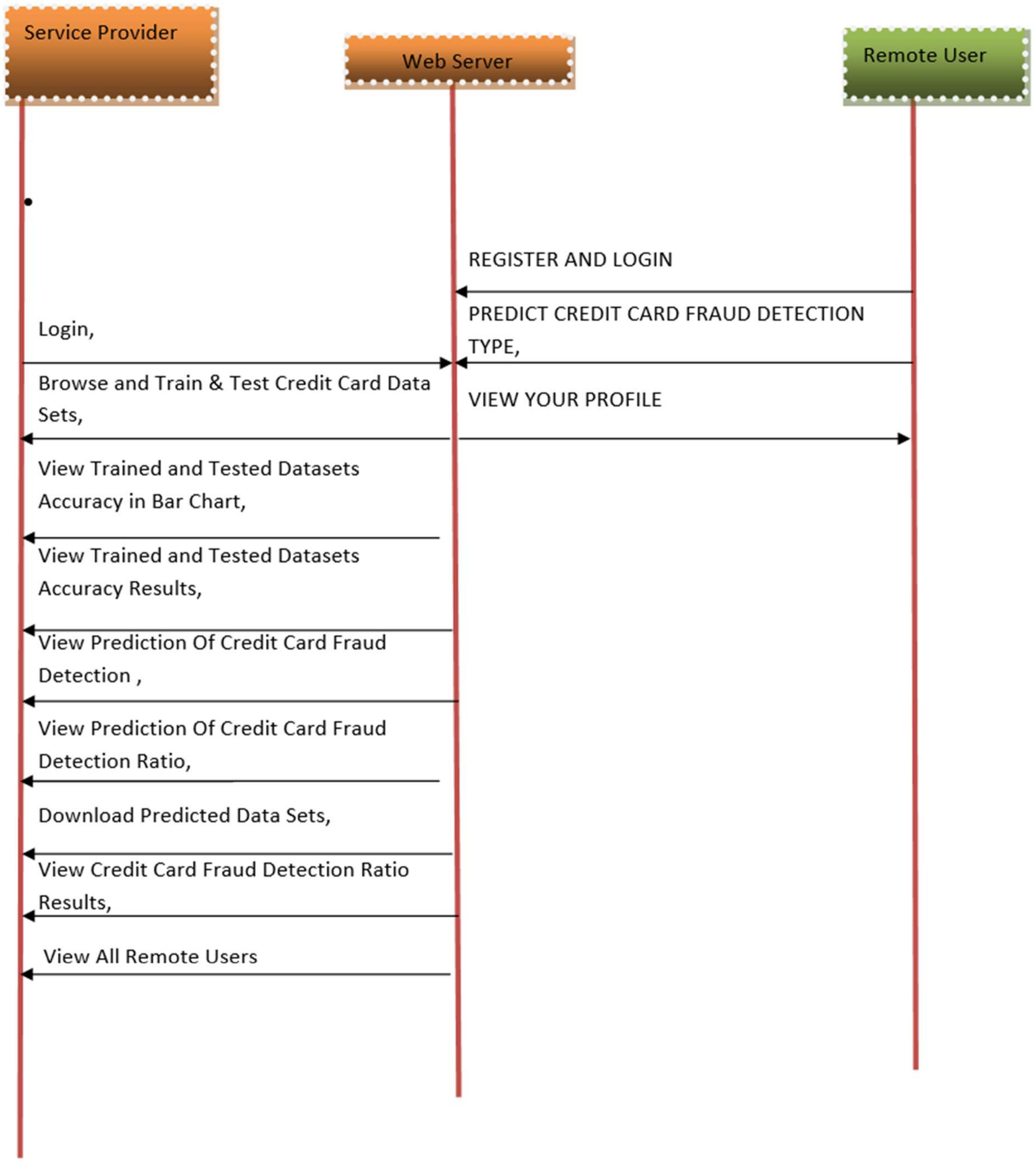

Browse and Train & Test Credit Card Data Sets, View Trained and Tested Datasets Accuracy in Bar Chart, View Trained and Tested Datasets Accuracy Results,
Service Provider
Login
System
PREDICT CREDIT CARD FRAUD DETECTION TYPE
View Prediction Of Credit Card Fraud Detection, View Prediction Of Credit Card Fraud Detection Ratio, Download Predicted Data Sets
Register and Login with the system
Response
VIEW YOUR PROFILE
Request
Remote User
View Credit Card Fraud Detection Ratio Results, View All Remote Users.

3.2.4 FLOW CHART ILLUSTRATING STRESS DETECTION WORKFLOW



Start
Login
Status
Yes
No
Username & Password Wrong
Browse and Train & Test Credit Card Data Sets
View Trained and Tested Datasets Accuracy in Bar Chart
View Trained and Tested Datasets Accuracy Results,
Log Out
View Prediction Of Credit Card Fraud Detection
View Prediction Of Credit Card Fraud Detection Ratio,
Download Predicted Data Sets,
View Credit Card Fraud Detection Ratio Results
View All Remote Users

3.2.5 SEQUENCE DIAGRAM OF USER INTERACTION WITH STRESS DETECTION SYSTEM


Service Provider
Web Server
Remote User
REGISTER AND LOGIN
PREDICT CREDIT CARD FRAUD DETECTION TYPE,
Login,
Browse and Train & Test Credit Card Data Sets,
VIEW YOUR PROFILE
View Trained and Tested Datasets Accuracy in Bar Chart,
View Trained and Tested Datasets Accuracy Results,
View Prediction Of Credit Card Fraud Detection ,
View Prediction Of Credit Card Fraud Detection Ratio,
Download Predicted Data Sets,
View Credit Card Fraud Detection Ratio Results,
View All Remote Users

4. IMPLEMENTATION

4.1 PROJECT MODULES

Remote User

The Remote User module represents the end-user or customer who initiates credit card transactions within the system. This module allows users to securely register, log in, and perform simulated transactions. The user inputs transaction data such as amount, location, and time, which the system then analyzes for potential fraud. The interface ensures usability and data privacy, enabling real-time monitoring of transaction status. If a transaction is flagged as suspicious, the user may receive alerts or be prompted for further verification. The goal of this module is to mimic real-world user behavior while feeding transaction data into the fraud detection system. It plays a crucial role in testing the accuracy and responsiveness of the predictive model in real-time environments.

Service Provider

The Service Provider module acts as the system administrator and backend manager responsible for fraud detection and overall system integrity. This module is tasked with uploading and managing transaction datasets, training and testing machine learning models, and analyzing system performance. The service provider monitors alerts, views prediction outcomes, and fine-tunes the machine learning algorithms to improve detection accuracy. It includes functionality to preprocess data, evaluate model metrics, and deploy the best-performing model. Additionally, the service provider ensures data security and model reliability. This module is central to the project's functionality, as it manages the core intelligence and logic that powers the fraud detection engine, making it the backbone of the overall system.

4.2 ALGORITHMS

Decision tree classifiers

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision making knowledge from the supplied data. Decision tree can be generated from training sets. The procedure for such generation based on the set of objects (S), each belonging to one of the classes C1, C2, …, Ck is as follows:

Step 1. If all the objects in S belong to the same class
Step 2. Otherwise, let T be some test with possible outcomes O1, O2,…, On. Each object in S has one outcome for T so the test partitions S into subsets S1, S2,… Sn where each object in Si has outcome Oi for T.

Gradient boosting

Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.[1][2] When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms random forest. A gradient- boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalize the other methods by allowing optimization of an arbitrary differentiable loss function.

K-Nearest Neighbors (KNN)

- Simple, but a very powerful classification algorithm
- Classifies based on a similarity measure
- Non-parametric
- Lazy learning
- Does not “learn” until the test example is given
- Whenever we have a new data to classify, we find its K-nearest neighbors from the training data

Example

- Training dataset consists of k-closest examples in feature space
- Feature space means, space with categorization variables (non-metric variables)
- Learning based on instances, and thus also works lazily because instance close to the input vector for test or prediction may take time to occur in the training dataset

Logistic regression Classifiers

Logistic Regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name logistic regression is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name multinomial logistic regression is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divorced, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar.

Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modeling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does.

This program computes binary logistic regression and multinomial logistic regression on both numeric and categorical independent variables. It reports on the regression equation as well as the goodness of fit, odds ratios, confidence limits, likelihood, and deviance. It performs a comprehensive residual analysis including diagnostic residual reports and plots. It can perform an independent variable subset selection search, looking for the best regression model with the fewest independent variables. It provides confidence intervals on predicted values and provides ROC curves to help determine the best cutoff point for classification. It allows you to validate your results by automatically classifying rows that are not used during the analysis.

Naive Bayes

The naive bayes approach is a supervised learning method which is based on a simplistic hypothesis: it assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.
Yet, despite this, it appears robust and efficient. Its performance is comparable to other supervised learning techniques. Various reasons have been advanced in the literature. In this tutorial, we highlight an explanation based on the representation bias. The naive bayes classifier is a linear classifier, as well as linear discriminant analysis, logistic regression or linear SVM (support vector machine). The difference lies on the method of estimating the parameters of the classifier (the learning bias).

While the Naive Bayes classifier is widely used in the research world, it is not widespread among practitioners which want to obtain usable results. On the one hand, the researchers found especially it is very easy to program and implement it, its parameters are easy to estimate, learning is very fast even on very large databases, its accuracy is reasonably good in comparison to the other approaches. On the other hand, the final users do not obtain a model easy to interpret and deploy, they does not understand the interest of such a technique.

Thus, we introduce in a new presentation of the results of the learning process. The classifier is easier to understand, and its deployment is also made easier. In the first part of this tutorial, we present some theoretical aspects of the naive bayes classifier. Then, we implement the approach on a dataset with Tanagra. We compare the obtained results (the parameters of the model) to those obtained with other linear approaches such as the logistic regression, the linear discriminant analysis and the linear SVM. We note that the results are highly consistent. This largely explains the good performance of the method in comparison to others. In the second part, we use various tools on the same dataset (Weka 3.6.0, R 2.9.2, Knime 2.1.1, Orange 2.0b and RapidMiner 4.6.0). We try above all to understand the obtained results.

Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.
The first algorithm for random decision forests was created in 1995 by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.
An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.).The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho and later independently by Amit and Geman in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

<u>SVM</u>

In classification tasks a discriminant machine learning technique aims at finding, based on an independent and identically distributed (id) training dataset, a discriminant function that can correctly predict labels for newly acquired instances. Unlike generative machine learning approaches, which require computations of conditional probability distributions, a discriminant classification function takes a data point *x* and assigns it to one of the different classes that are a part of the classification task. Less powerful than generative approaches, which are mostly used when prediction involves outlier detection, discriminant approaches require fewer computational resources and less training data, especially for a multidimensional feature space and when only posterior probabilities are needed. From a geometric perspective, learning a classifier is equivalent to finding the equation for a multidimensional surface that best separates the different classes in the feature space.

SVM is a discriminant technique, and, because it solves the convex optimization problem analytically, it always returns the same optimal hyperplane parameter—in contrast to genetic algorithms (GAs) or perceptrons, both of which are widely used for classification in machine learning. For perceptrons, solutions are highly dependent on the initialization and termination criteria. For a specific kernel that transforms the data from the input space to the feature space, training returns uniquely defined SVM model parameters for a given training set, whereas the perceptron and GA classifier models are different each time training is initialized. The aim of GAs and perceptrons is only to minimize error during training, which will translate into several hyperplanes' meeting this requirement.

4.3 SAMPLE CODE

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report,
confusion_matrix
data = pd.read_csv('creditcard.csv')
X = data.drop('Class', axis=1)
y = data['Class']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(
X_scaled, y, test_size=0.3, random_state=42
)
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

5. TESTING AND RESULTS

5.1 TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1. **Top Down Integration**

This method is an incremental approach to the construction of program structure.Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. **Bottom-up Integration**

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking
Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTENANCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING:

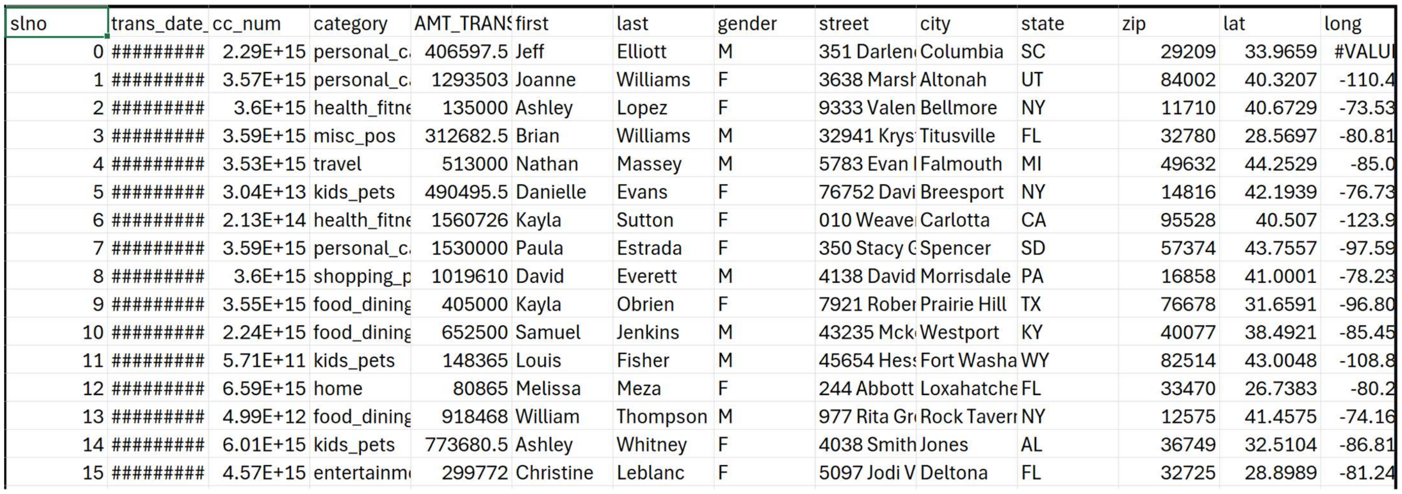
Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

UNIT TESTING:

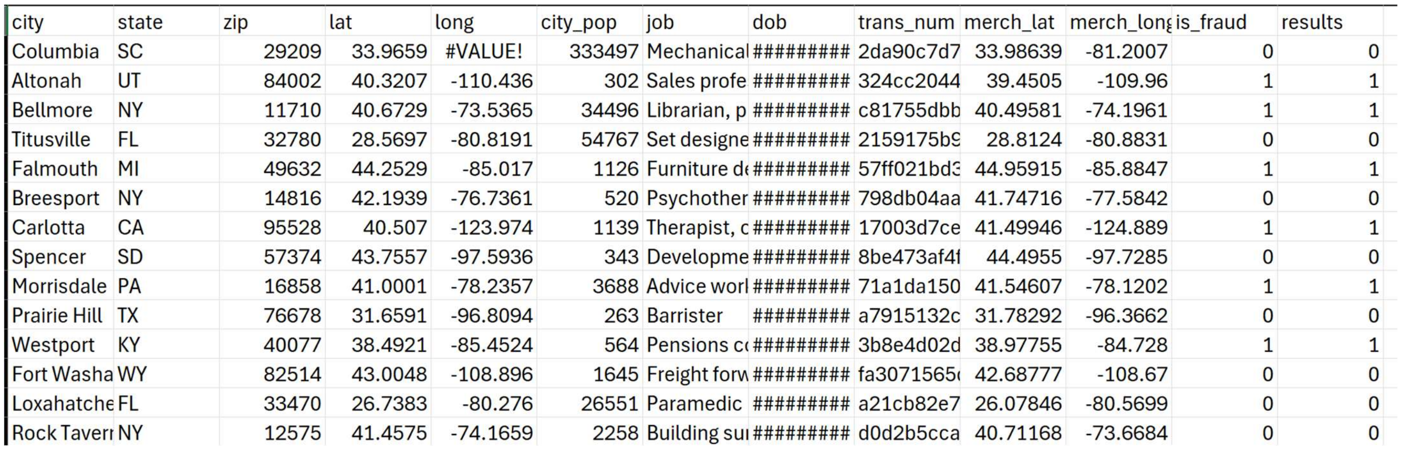
In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

6. RESULT

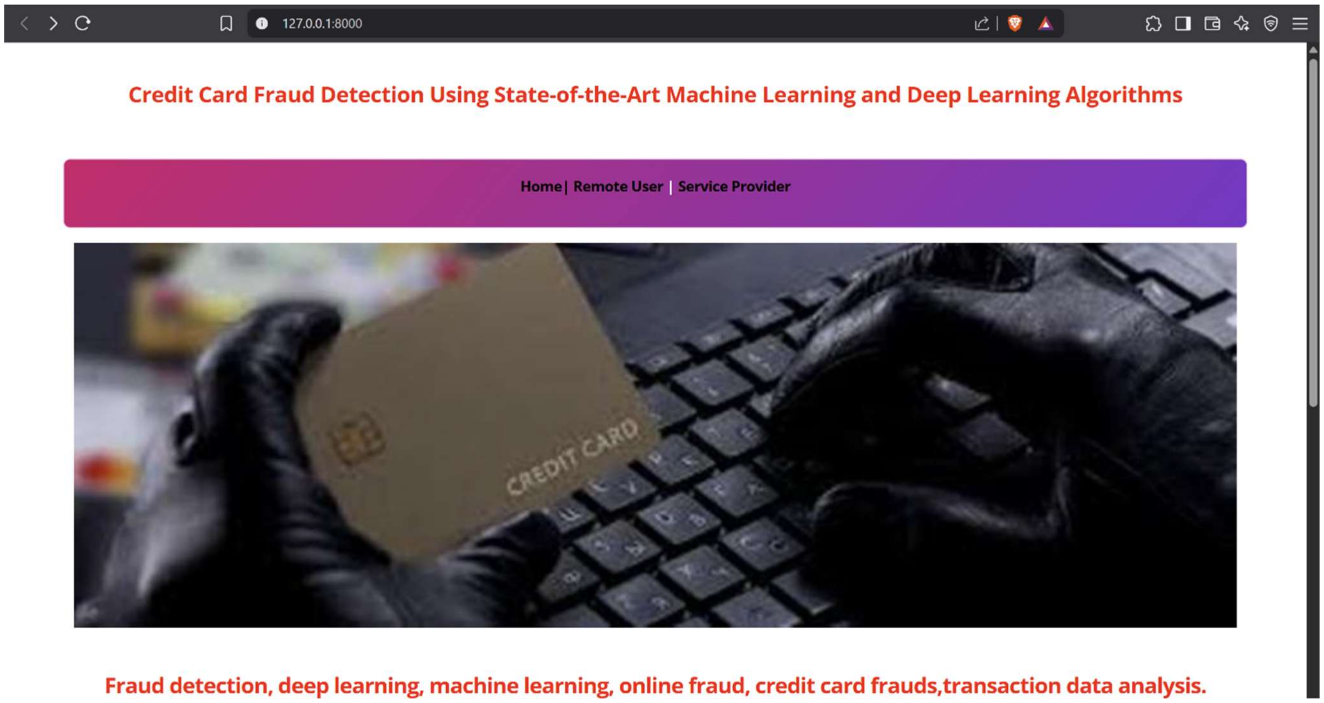


slno	trans_date_	cc_num	category	AMT_TRANS	first	last	gender	street	city	state	zip	lat	long
0	#########	2.29E+15	personal_c	406597.5	Jeff	Elliott	M	351 Darlen	Columbia	SC	29209	33.9659	#VALU
1	#########	3.57E+15	personal_c	1293503	Joanne	Williams	F	3638 Marsh	Altonah	UT	84002	40.3207	-110.4
2	#########	3.6E+15	health_fitne	135000	Ashley	Lopez	F	9333 Valen	Bellmore	NY	11710	40.6729	-73.53
3	#########	3.59E+15	misc_pos	312682.5	Brian	Williams	M	32941 Krys	Titusville	FL	32780	28.5697	-80.81
4	#########	3.53E+15	travel	513000	Nathan	Massey	M	5783 Evan I	Falmouth	MI	49632	44.2529	-85.0
5	#########	3.04E+13	kids_pets	490495.5	Danielle	Evans	F	76752 Davi	Breesport	NY	14816	42.1939	-76.73
6	#########	2.13E+14	health_fitne	1560726	Kayla	Sutton	F	010 Weave	Carlotta	CA	95528	40.507	-123.9
7	#########	3.59E+15	personal_c	1530000	Paula	Estrada	F	350 Stacy G	Spencer	SD	57374	43.7557	-97.59
8	#########	3.6E+15	shopping_p	1019610	David	Everett	M	4138 David	Morrisdale	PA	16858	41.0001	-78.23
9	#########	3.55E+15	food_dining	405000	Kayla	Obrien	F	7921 Rober	Prairie Hill	TX	76678	31.6591	-96.80
10	#########	2.24E+15	food_dining	652500	Samuel	Jenkins	M	43235 Mck	Westport	KY	40077	38.4921	-85.45
11	#########	5.71E+11	kids_pets	148365	Louis	Fisher	M	45654 Hes	Fort Washa	WY	82514	43.0048	-108.8
12	#########	6.59E+15	home	80865	Melissa	Meza	F	244 Abbott	Loxahatche	FL	33470	26.7383	-80.2
13	#########	4.99E+12	food_dining	918468	William	Thompson	M	977 Rita Gr	Rock Taverr	NY	12575	41.4575	-74.16
14	#########	6.01E+15	kids_pets	773680.5	Ashley	Whitney	F	4038 Smith	Jones	AL	36749	32.5104	-86.81
15	#########	4.57E+15	entertainm	299772	Christine	Leblanc	F	5097 Jodi V	Deltona	FL	32725	28.8989	-81.24



city	state	zip	lat	long	city_pop	job	dob	trans_num	merch_lat	merch_long	is_fraud	results
Columbia	SC	29209	33.9659	#VALUE!	333497	Mechanical	#########	2da90c7d7	33.98639	-81.2007	0	0
Altonah	UT	84002	40.3207	-110.436	302	Sales profe	#########	324cc2044	39.4505	-109.96	1	1
Bellmore	NY	11710	40.6729	-73.5365	34496	Librarian, p	#########	c81755dbb	40.49581	-74.1961	1	1
Titusville	FL	32780	28.5697	-80.8191	54767	Set designe	#########	2159175b9	28.8124	-80.8831	0	0
Falmouth	MI	49632	44.2529	-85.017	1126	Furniture de	#########	57ff021bd3	44.95915	-85.8847	1	1
Breesport	NY	14816	42.1939	-76.7361	520	Psychother	#########	798db04aa	41.74716	-77.5842	0	0
Carlotta	CA	95528	40.507	-123.974	1139	Therapist, c	#########	17003d7ce	41.49946	-124.889	1	1
Spencer	SD	57374	43.7557	-97.5936	343	Developme	#########	8be473af4f	44.4955	-97.7285	0	0
Morrisdale	PA	16858	41.0001	-78.2357	3688	Advice worl	#########	71a1da150	41.54607	-78.1202	1	1
Prairie Hill	TX	76678	31.6591	-96.8094	263	Barrister	#########	a7915132c	31.78292	-96.3662	0	0
Westport	KY	40077	38.4921	-85.4524	564	Pensions co	#########	3b8e4d02d	38.97755	-84.728	1	1
Fort Washa	WY	82514	43.0048	-108.896	1645	Freight forw	#########	fa3071565	42.68777	-108.67	0	0
Loxahatche	FL	33470	26.7383	-80.276	26551	Paramedic	#########	a21cb82e7	26.07846	-80.5699	0	0
Rock Taverr	NY	12575	41.4575	-74.1659	2258	Building sur	#########	d0d2b5cca	40.71168	-73.6684	0	0

Fig 6.1 DATASETS

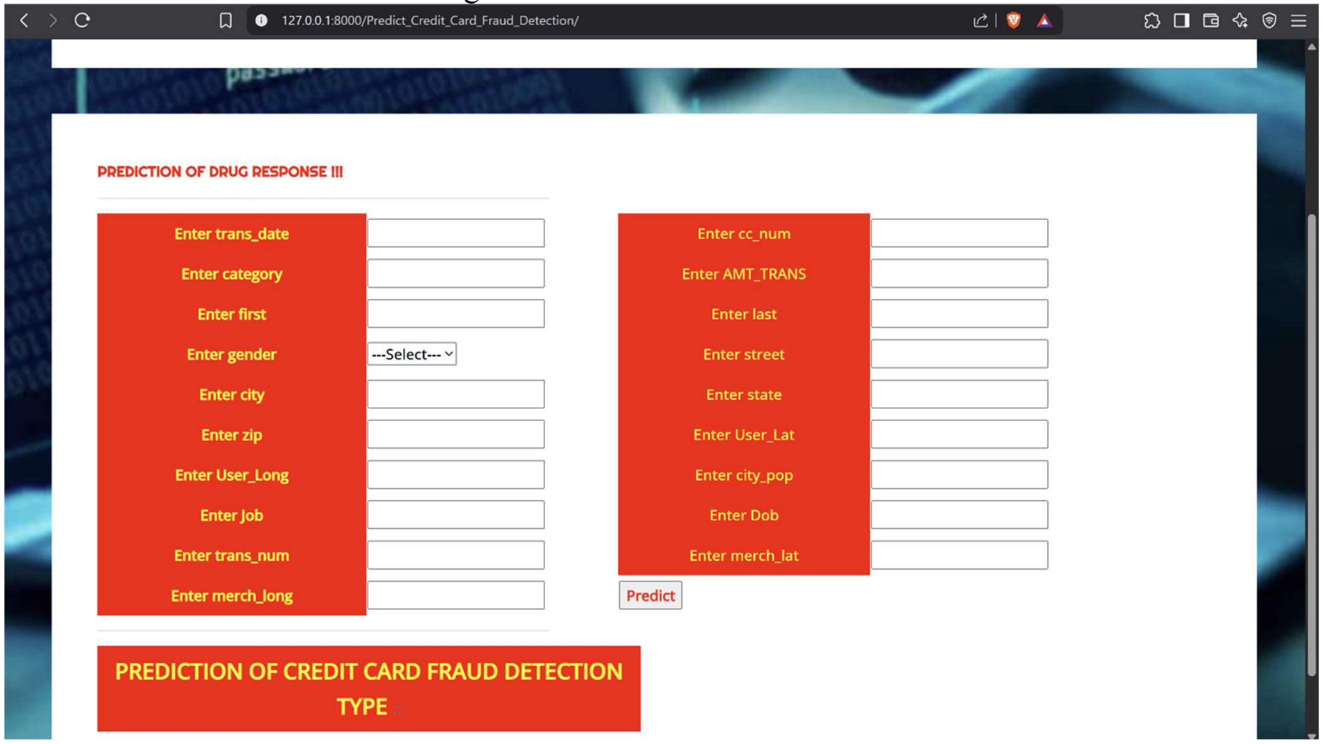


127.0.0.1:8000

Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms

Fraud detection, deep learning, machine learning, online fraud, credit card frauds,transaction data analysis.

Fig 6.2 HOME PAGE



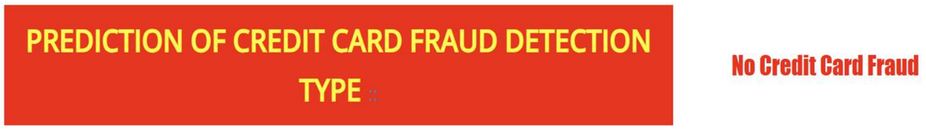
127.0.0.1:8000/Predict_Credit_Card_Fraud_Detection/

PREDICTION OF DRUG RESPONSE !!!

Enter trans_date		Enter cc_num	
Enter category		Enter AMT_TRANS	
Enter first		Enter last	
Enter gender	---Select---	Enter street	
Enter city		Enter state	
Enter zip		Enter User_Lat	
Enter User_Long		Enter city_pop	
Enter Job		Enter Dob	
Enter trans_num		Enter merch_lat	
Enter merch_long		Predict	

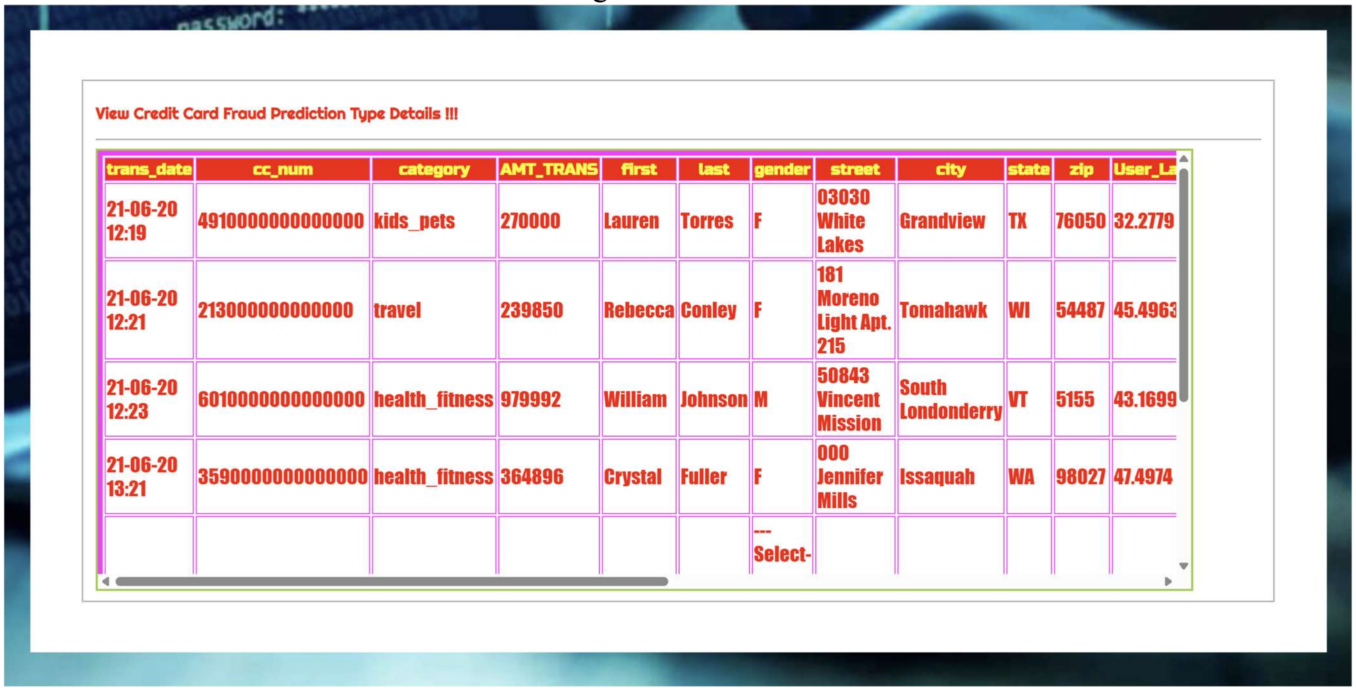
PREDICTION OF CREDIT CARD FRAUD DETECTION TYPE :

Fig 6.3 USER INPUT FORM


PREDICTION OF CREDIT CARD FRAUD DETECTION TYPE :

No Credit Card Fraud

Fig 6.4 OUTPUT


View Credit Card Fraud Prediction Type Details !!!

trans_date	cc_num	category	AMT_TRANS	first	last	gender	street	city	state	zip	User_La
21-06-20 12:19	4910000000000000	kids_pets	270000	Lauren	Torres	F	03030 White Lakes	Grandview	TX	76050	32.2779
21-06-20 12:21	213000000000000	travel	239850	Rebecca	Conley	F	181 Moreno Light Apt. 215	Tomahawk	WI	54487	45.4963
21-06-20 12:23	6010000000000000	health_fitness	979992	William	Johnson	M	50843 Vincent Mission	South Londonderry	VT	5155	43.1699
21-06-20 13:21	3590000000000000	health_fitness	364896	Crystal	Fuller	F	000 Jennifer Mills	Issaquah	WA	98027	47.4974
						---Select-					

Fig 6.5 USER ANALYSIS

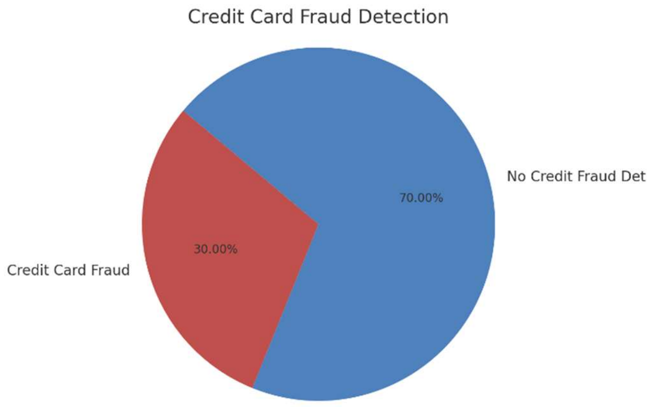


Fig 6.6 PIE CHART FOR RESULT RATIO

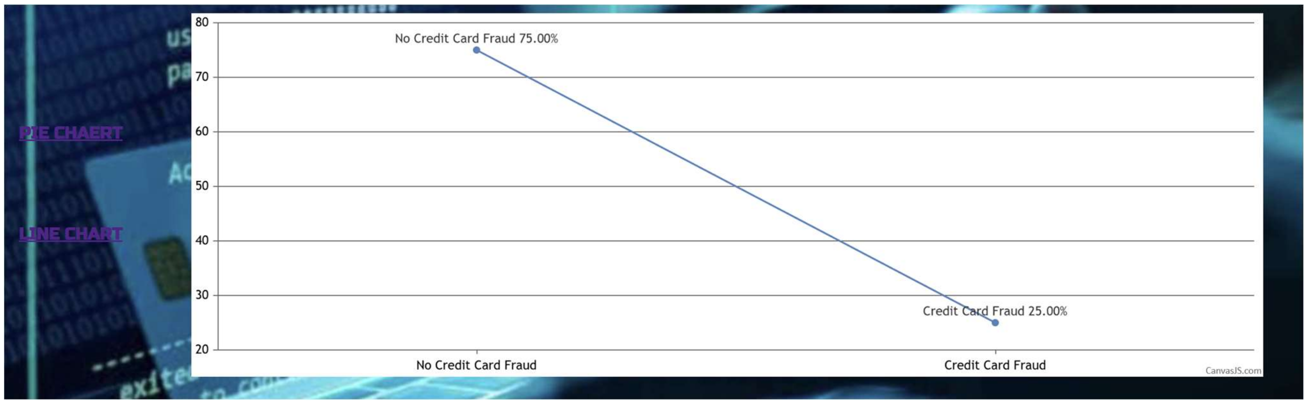
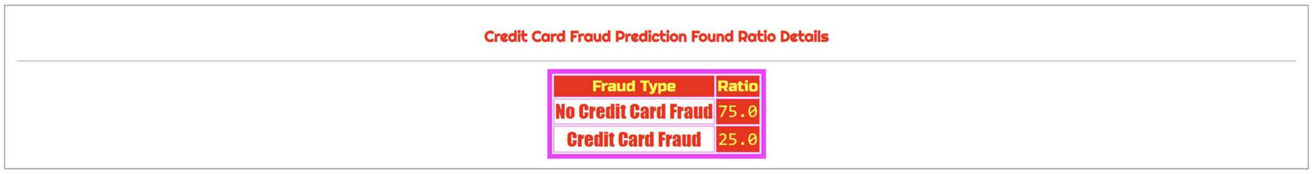


Fig 6.6 LINE CHART FOR RESULT RATIO



Credit Card Fraud Prediction Found Ratio Details

Fraud Type	Ratio
No Credit Card Fraud	75.0
Credit Card Fraud	25.0

Fig 6.8 TEST DATA RESULTS

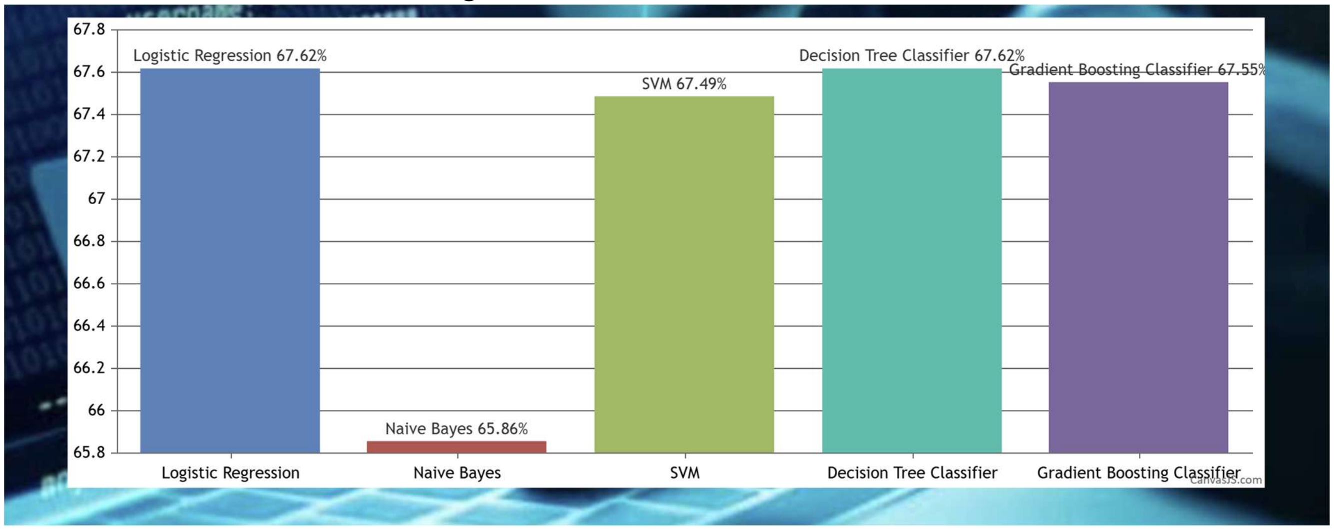


Fig 6.9 BAR GRAGH FOR TRAINED AND TEST DATA ANALYSIS

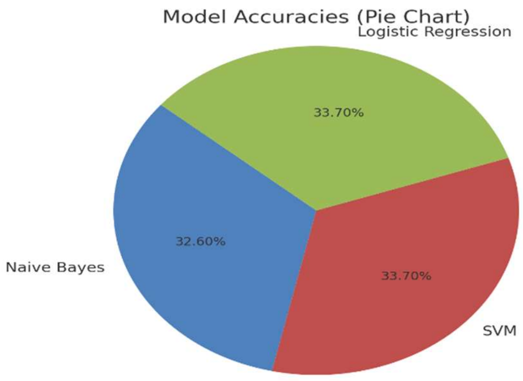


Fig 6.10 PIE CHART FOR TRAINED AND TEST DATA ANALYSIS

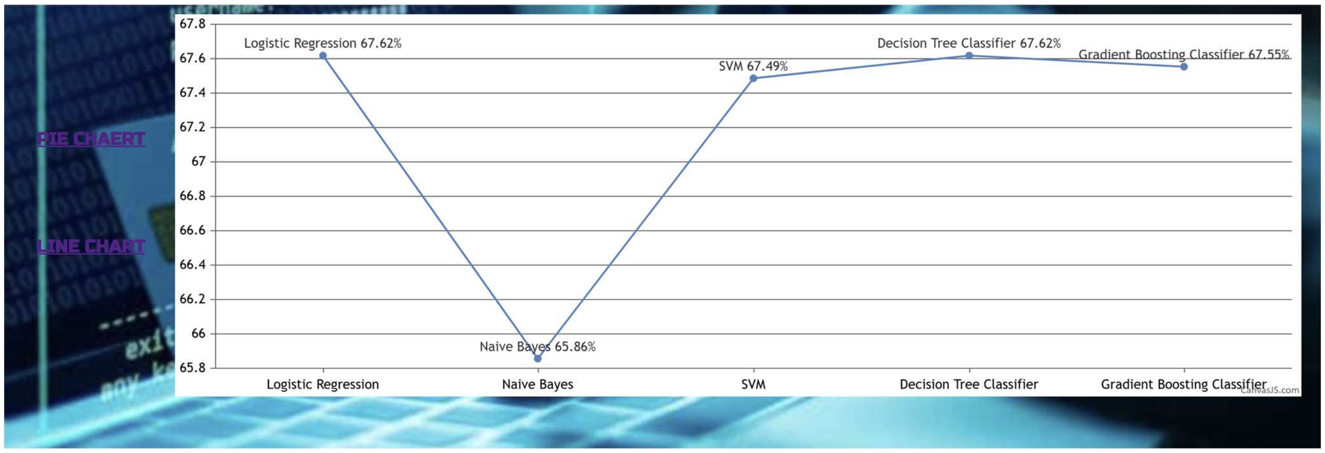


Fig 6.11 LINE CHART FOR TRAINED AND TEST DATA ANALYSIS

7. CONCLUSION

The project **“Credit Card Fraud Detection Using State-of-the-Art Machine Learning”** successfully demonstrates how advanced machine learning algorithms can be effectively applied to detect and reduce fraudulent activities in credit card transactions. With the increasing number of digital financial operations, especially in e-commerce and online banking, the risk of fraud has become a major concern for financial institutions and users alike. This system addresses that issue by providing a predictive, intelligent, and automated solution to identify suspicious transactions with high accuracy.

Throughout the project, various phases were executed—from data preprocessing and handling imbalanced datasets to model training and evaluation using multiple machine learning algorithms like Logistic Regression, Random Forest, Support Vector Machines (SVM), and XGBoost. Among these, ensemble models such as Random Forest and XGBoost showed superior performance in terms of recall, precision, and overall fraud detection capability. The project also tackled common challenges such as false positives, model bias due to data imbalance, and the need for real-time prediction, ensuring that the final output is both practical and effective.
This system not only automates the fraud detection process but also significantly reduces manual effort and human error. Its modular design allows for scalability, making it easy to update or extend the model with new data and techniques. The results show that integrating machine learning into financial systems can lead to a safer and more reliable transaction environment. In the future, this model can be enhanced by incorporating deep learning techniques and real-time streaming data to improve responsiveness and adaptability further. Overall, this project offers a strong foundation for future research and practical implementation in the fight against financial fraud.

8. REFERENCES

[1] [1] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, “Calibrating Probability with Undersampling for Unbalanced Classification,” in *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166, IEEE, 2015.

[2] [2] P. Phua, V. Lee, K. Smith, and R. Gayler, “A Comprehensive Survey of Data Mining-based Fraud Detection Research,” *arXiv preprint arXiv:1009.6119*, 2010.

[3] [3] S. Jha, M. Guillen, and J. C. Westland, “Employing Transaction Aggregation Strategy to Detect Credit Card Fraud,” *Expert Systems with Applications*, vol. 39, no. 16, pp. 12650–12657, 2012.

[4] [4] Scikit-learn: Machine Learning in Python – https://scikit-learn.org

[5] [5] XGBoost: Scalable and Flexible Gradient Boosting – https://xgboost.readthedocs.io

[6] [6] Kaggle Credit Card Fraud Detection Dataset – https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[7] [7] A. Srivastava, A. Kundu, S. Sural, and A. Majumdar, “Credit Card Fraud Detection using Hidden Markov Model,” *IEEE Transactions on Dependable and Secure Computing*, vol. 5, no. 1, pp. 37–48, 2008.

[8] [8] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, 4th ed., Elsevier, 2016.

[9] [9] T. Fawcett, “An Introduction to ROC Analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[10] [10] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[11] [11] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[12] [12] S. Sahin and E. Duman, “Detecting Credit Card Fraud by Decision Trees and Support Vector Machines,” in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, pp. 442–447, 2011.

[13] [13] V. Van Vlasselaer, T. Eliassi-Rad, L. Akoglu, and F. Provost, “Gotcha! Network-based Fraud Detection for Social Security Fraud,” *Management Science*, vol. 63, no. 9, pp. 3090–3110, 2017.

[14] [14] D. M. West, “Neural Network Credit Scoring Models,” *Computers & Operations Research*, vol. 27, no. 11-12, pp. 1131–1152, 2000.

[15] [15] Python Software Foundation – https://www.python.org

[16] [16] Jupyter Notebook – https://jupyter.org

[17] [17] A. Bhattacharyya, R. Jha, and T. Thakur, “Credit Card Fraud Detection Using Machine Learning Algorithms: A Comparative Study,” in *2020 International Conference on Computational Performance Evaluation (ComPE)*, IEEE, 2020.

[18] [18] H. Han, W. Wang, and B. Mao, “Borderline-SMOTE: A New Over-sampling Method in Imbalanced Data Sets Learning,” in *Advances in Intelligent Computing*, pp. 878–887, Springer, 2005.

[19] [19] C. Elkan, “The Foundations of Cost-Sensitive Learning,” in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 973–978, 2001.

[20] [20] M. Goldstein and S. Uchida, “A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data,” *PLOS ONE*, vol. 11, no. 4, e0152173, 2016.

[21] [21] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data Clustering: A Review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
[22] [22] D. J. Hand, G. Blunt, M. G. Kelly, and N. M. Adams, “Data Mining for Fun and Profit,” *Statistical Science*, vol. 15, no. 2, pp. 111–131, 2000.
[23] [23] M. Zareapoor and P. Shamsolmoali, “Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier,” *Procedia Computer Science*, vol. 48, pp. 679–685, 2015.
[24] [24] P. Kumar and S. Singh, “Credit Card Fraud Detection Using Neural Networks,” *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 3, no. 1, pp. 83–87, 2013.
[25] [25] A. Ahmed and A. Naser Mahmood, “A Survey of Network Anomaly Detection Techniques,” *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.
[26] [26] Y. Sahin and E. Duman, “Detecting Credit Card Fraud by ANN and Logistic Regression,” in *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, pp. 1–5, IEEE, 2011.
[27] [27] M. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, “Feature Engineering Strategies for Credit Card Fraud Detection,” *Expert Systems with Applications*, vol. 51, pp. 134–142, 2016.
[28] [28] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, ACM, 2016.
[29] [29] S. Verma and K. Ranga, “Machine Learning Techniques for Fraud Detection: A Comparative Study,” *International Journal of Applied Engineering Research*, vol. 13, no. 6, pp. 3482–3487, 2018.
[30] [30] H. S. Patel and K. T. Trivedi, “A Survey of Various Data Mining Techniques for Credit Card Fraud Detection,” *International Journal of Computer Applications*, vol. 113, no. 1, pp. 1–5, 2015.