A

Major Project

On

CREDIT CARD FRAUD DETECTION USING STATE OF THE ART USING MACHINE LEARNING AND DEEP LEARNING ALGORITHM

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

M. RISHITHA REDDY (207R1A0595)

GUGLAVATH SRISHANTH (207R1A0580)

MARIPEDDA PRAVEEN (217R5A0508)

Under the Guidance of

Ms. Saba Sultana

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"CREDIT CARD FRAUD DETECTION USING STATE OF THE ART USING MACHINE LEARNING AND DEEP LEARNING ALGORITHM"** being submitted by **M. RISHITHA REDDY (207R1A0595), GUGLAVATH SRISHANTH (207R1A0580) & MARIPEDDA PRAVEEN (217R5A0508)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Ms. Saba Sultana
(Assistant Professor)
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on _____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard tomy guide **Ms. Saba Sultana,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G.Vinesh Shanker, Dr. J. Narasimharao , Ms. Shilpa,& Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

M. RISHITHA REDDY (207R1A0595)
GUGLAVATH SRISHANTH (207R1A0580)
MARIPEDDA PRAVEEN (217R5A0508)

ABSTRACT

Credit cards have become an integral part of modern financial transactions, providing users with a convenient and widely accepted method of payment. The widespread use of credit cards, both online and offline, has, however, given rise to the escalating threat of credit card fraud. Fraudulent activities such as unauthorized transactions, identity theft, and skimming pose substantial risks to financial institutions and cardholders. With the increasing prevalence of online transactions, the risk of credit card fraud has become a major concern for financial institutions and users alike. This study proposes an advanced Credit Card Fraud Detection system leveraging state-of-the-art machine learning and deep learning algorithms to enhance the accuracy and efficiency of fraud detection.

The financial ecosystem's reliance on credit cards necessitates the development of robust and sophisticated fraud detection systems to safeguard the integrity of transactions and protect consumers from financial losses. Traditional methods of fraud detection, such as rule-based systems and signature verification, have become increasingly inadequate in the face of evolving and sophisticated fraudulent techniques. As a response to the growing challenges, the integration of advanced technologies, particularly machine learning and deep learning, has emerged as a promising avenue for improving the accuracy and efficiency of credit card fraud detection. These technologies leverage complex algorithms and data patterns to identify abnormal behavior, detect anomalies, and adapt to the dynamic nature of fraudulent activities. This project aims to explore and implement state-of-the-art machine learning and deep learning algorithms for credit card fraud detection. By harnessing the power of these advanced techniques, the research seeks to enhance the capability of financial institutions to detect and prevent fraudulent transactions in real-time, providing a more secure environment for credit card users.

LIST OF FIGURES/TABLES

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

The project, "Credit Card Fraud Detection using State of the Art usingMachine Learning and Deep Learning Algorithm," involves implementing cutting-edgemachine learning and deep learning algorithms for Credit Card Fraud Detection. The study encompasses the exploration and utilization of diverse datasets containing both legitimate and fraudulent transactions. Various state-of-the-art algorithms, including machine learning models such as logistic regression, decision trees, random forests, support vector machines, and deep learning models such as neural networks and convolutional neural networks, will be employed. The project will assess the performance of each algorithm and explore ensemble methods to enhance the overall fraud detection system. Performance evaluation metrics, including precision, recall, F1 score, and the area under the Receiver Operating Characteristic curve, will be used to measure the effectiveness of the proposed models. The ultimate goal is to develop a robust and adaptive system that can accurately detect credit card fraud in real-time, contributing to the ongoing efforts in securing financial transactions against evolving fraudulent techniques.

1.2 PROJECT PURPOSE

The purpose of this project is to address the escalating threat of credit card fraud through the implementation of cutting-edge machine learning and deep learning algorithms. As the reliance on credit cards for financial transactions grows, so does the risk of fraudulent activities. The project aims to develop a sophisticated fraud detection system capable of identifying and preventing unauthorized transactions in real-time. By leveraging state-of-the-art techniques, including machine learning algorithms such as logistic regression, decision trees, and deep learning models like neural networks, the project seeks to enhance the accuracy and efficiency of fraud detection.

The ultimate goal is to provide financial institutions with a robust and adaptive tool to safeguard users from the evolving nature of credit card fraud in the digital era. This project seeks to leverage the capabilities of cutting-edge machine learning and deep learning algorithms to enhance the accuracy, sensitivity, and efficiency of fraud detection systems. By harnessing the power of these state-of-the-art technologies, the project aims to contribute to the development of a robust and proactive solution that can quickly and accurately identify fraudulent activities, thereby safeguarding financial institutions and protecting the interests of credit card users in the rapidly evolving digital ecosystem.

1.3 PROJECT FEATURES

This project encompasses a range of features that leverage cutting-edge machine learning and deep learning algorithms to create an advanced and robust system. Firstly, the project incorporates a comprehensive dataset, comprising both legitimate and fraudulent transactions, to facilitate effective model training and evaluation. The utilization of state-of-the-art machine learning algorithms, such as logistic regression, decision trees, random forests, and support vector machines, forms the foundation for establishing a baseline performance in fraud detection. The integration of deep learning models, including neural networks and convolutional neural networks (CNNs), takes the project to a higher level by capturing intricate patterns and non-linear relationships within the data. The adoption of these advanced algorithms aims to enhance the system's accuracy and efficiency, particularly in identifying subtle and evolving fraudulent patterns that may escape traditional detection methods.

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

System Analysis for Credit Card Fraud Detection using state-of-the-art machine learning and deep learning algorithms involves a comprehensive examination of the various components and processes within the proposed framework. At the core of this analysis is the exploration of diverse machine learning and deep learning models, each designed to uncover intricate patterns and anomalous behavior indicative of credit card fraud. The initial phase entails the acquisition and preprocessing of a comprehensive dataset, comprising both legitimate and fraudulent transactions, to facilitate effective model training.

2.1 PROBLEM DEFINITION

The problem addressed is the increasing prevalence of credit card fraud poses a significant challenge for financial institutions and users alike. Traditional methods of fraud detection are proving insufficient in addressing the sophisticated techniques employed by fraudsters. This study aims to address this problem by leveraging state-of-the-art machine learning and deep learning algorithms for credit card fraud detection. The primary objective is to enhance the accuracy and efficiency of fraud detection systems, providing financial institutions with a more robust tool to identify and prevent fraudulent transactions in real-time. The research aims to contribute to the ongoing efforts in developing advanced solutions to safeguard financial transactions and protect users from the evolving landscape of credit card fraud.

2.2 EXISTING SYSTEM

ML has many branches, and each branch can deal with different learning tasks. However, ML learning has different framework types. The ML approach provides a solution for CCF, such as random forest (RF). The ensemble of the decision tree is the random forest. Most researchers use the RF approach. To combine the model, we can use (RF) along with network analysis. This method is called APATE. Researchers can use different ML techniques, such as supervised learning and unsupervised techniques. ML algorithms, such as LR, ANN, DT, SVM and NB, are commonly used for CCF detection. The researcher can combine these techniques with ensemble techniques to construct solid detection classifiers. The linking of multiple neurons and nodes is known as an artificial neural network. A feed-forward perceptron multilayer is built up of numerous layers: an input layer, an output layer and one or more hidden layers. For the representation of the exploratory variables, the first layer contains the input nodes. With a precise weight, these input layers are multiplied, and each of the hidden layer nodes is transferred with a certain bias, and they are added together.

An activation function is then applied to create the output of each neuron for this summation, which is then transferred to the next layer. Finally, the algorithm's reply is provided by the output layer. The first set randomly used weights and formerly used the training set to minimise the error. All these weights were adjusted by detailed algorithms such as back propagation. The graphic model for contingency relationships between a set of variables is called the Bayesian belief network. The independence assumption in naïve Bayes is that it was developed to relax and allow for dependencies among variables.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- The system is not implemented Classification on Imbalanced Data.
- The system is not implemented CONVOLUTIONAL NEURAL NETWORK (CNN) for test and train the datasets.

2.3 PROPOSED SYSTEM

In this project, Feature selection algorithms are used to rank the top features from the CCF transaction dataset, which help in class label predictions. The deep learning model is proposed by adding a number of additional layers that are then used to extract the features and classification from the credit card farad detection dataset. To analyse the performance CNN model, apply different architecture of CNN layers. To perform a comparative analysis between ML with DL algorithms and proposed CNN with baseline model, the results prove that the proposed approach outperforms existing approaches. To assess the accuracy of the classifiers, performance evaluation measures, accuracy, precision, and recall are used. Experiments are performed on the latest credit cards dataset. It integrates cutting-edge machine learning and deep learning algorithms to enhance the accuracy and efficiency of fraud detection in credit card transactions.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- The proposed system uses SUPERVISED MACHINE LEARNING APPROACHES which are effective for testing and training datasets.
- The proposed system implemented CNN is to minimise processing without losing key features by reducing the image to make predictions

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This isto ensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on a project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminaryinvestigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication that the system is economically possible for development.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 BEHAVIORAL FEASIBILITY

This includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface betweenthe software product and the hardware components of the system. The followingare some hardware requirements.

- Processor : Pentium-IV
- Hard disk : 20 GB
- RAM : 4 GB (min)
- Key Board : Standard Windows Keyboard
- Mouse : Two or Three Button Mouse
- Monitor : SVGA

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

- Operating system : Windows 7 Ultimate
- Coding Language : Python
- Front-End : Python
- Back-End : Django-ORM
- Designing : HTML, CSS, Javascript
- Data Base : MySQL (WAMP Server)

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

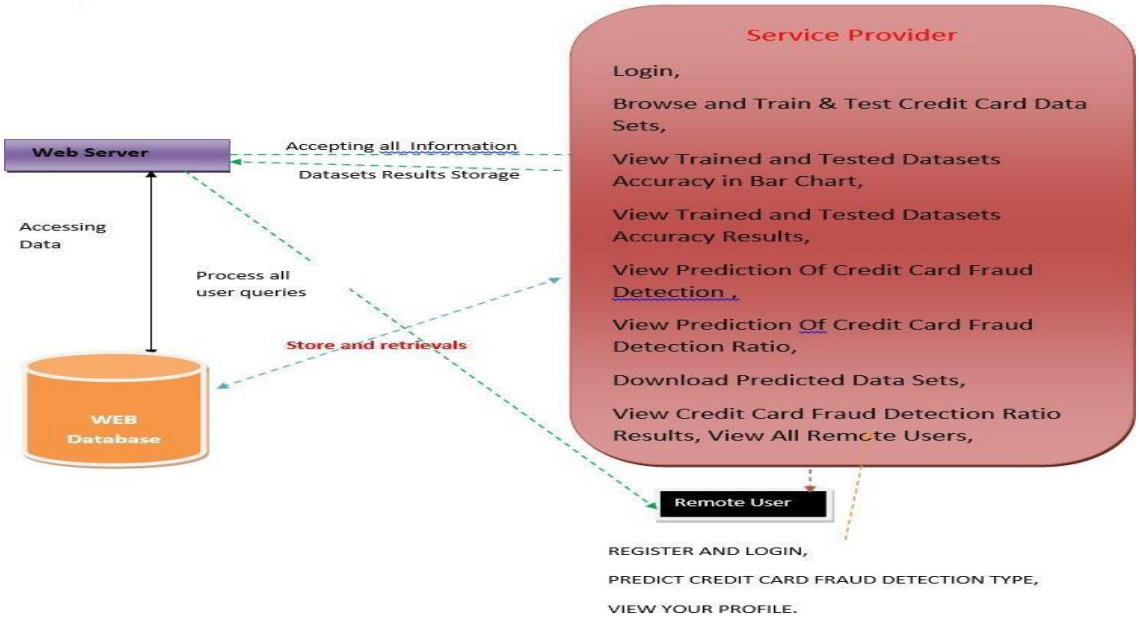


Figure 3.1: Project Architecture for Credit Card Fraud Detection using Stateof Art Machine Learning and Deep Learning Algorithms

3.2 DESCRIPTION

This project involves the integration of feature selection algorithms, machine learning (ML), and deep learning (DL) techniques for fraud detection in credit card transactions. Feature selection algorithms are employed to rank the top features from the CCF (Credit Card Fraud) transaction dataset. Different architectures of Convolutional Neural Networks (CNN) layers are applied to analyze the performance of the CNN model. Experiments are conducted on the latest credit card dataset, suggesting that the project incorporates up-to-date data for training and evaluation.

3.3 USE CASE DIAGRAM

This use case diagram represents how a system can be used to detect credit card fraud. This use case diagram provides a high-level overview of the functionalities and interactions involved in the credit card fraud detection project, emphasizing the user's role in initiating and overseeing various tasks within the system.

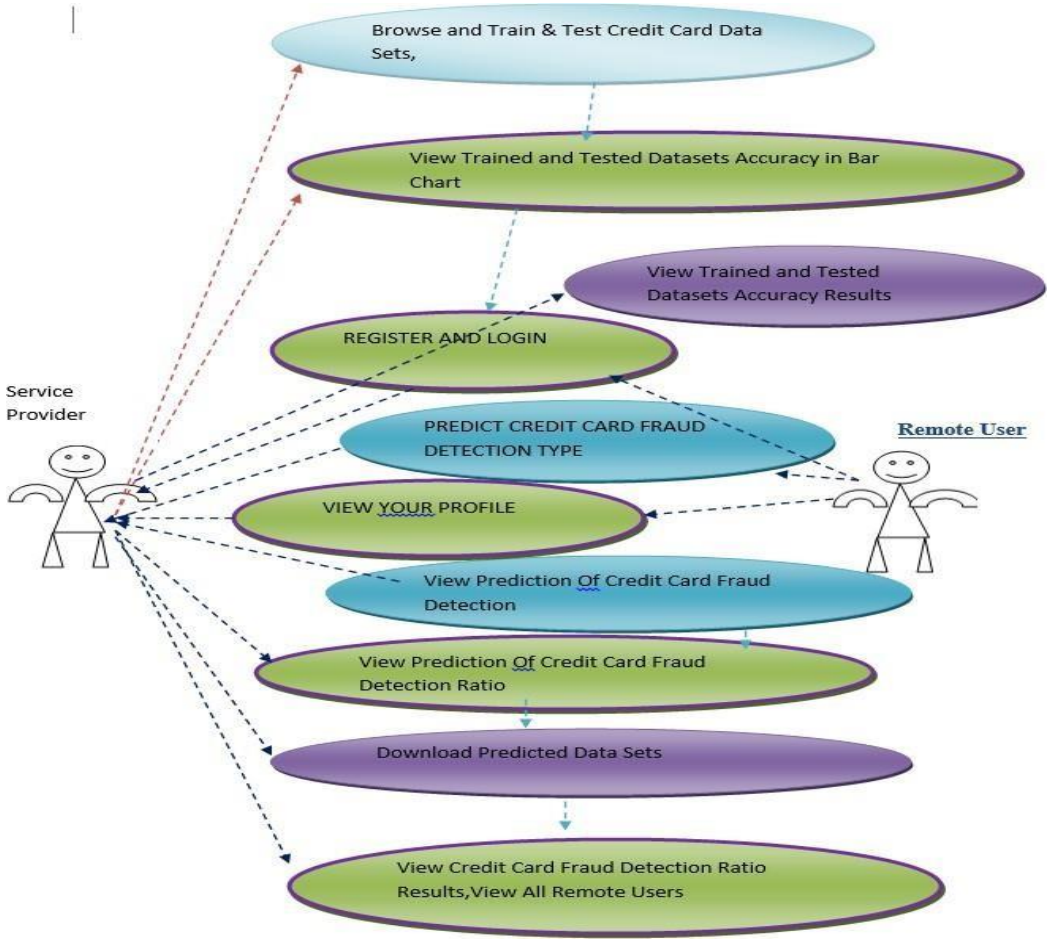


Figure 3.2: Use Case Diagram for Credit Card Fraud Detectionusing StateofArt Machine Learningand Deep Learning Algorithms

3.4 CLASS DIAGRAM

A class diagram is a static structure diagram that illustrates a system's structure by presenting its classes, attributes, methods (operations), and object relationships.Creating a class diagram based on the abstract you provided involves identifyingkey entities, their attributes, and relationships. Class diagrams are essential for visualizing the structure of a system and understanding how different classes collaborateto achieve specific functionalities.

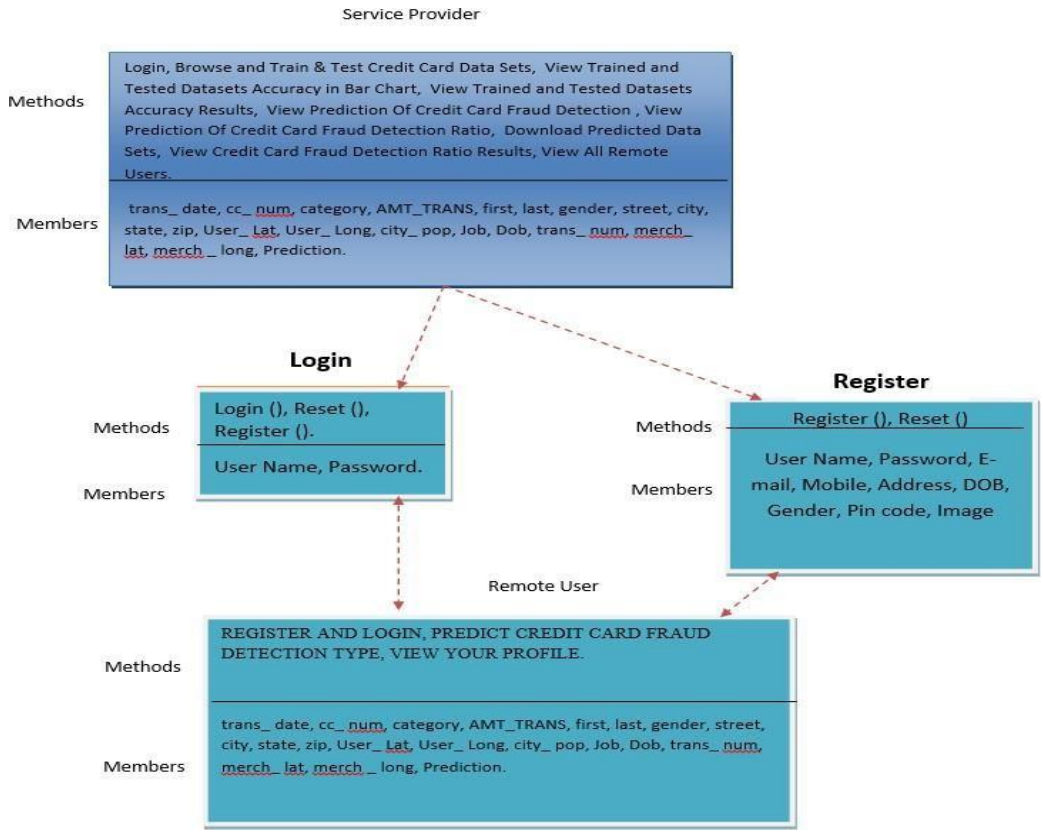


Figure 3.3: Class Diagram for Credit Card Fraud Detectionusing StateofArt Machine Learningand Deep Learning Algorithms

3.5 SEQUENCE DIAGRAM

A sequence diagram visually depicts object interactions in chronological order, showcasing the involved objects and their message exchanges to execute a scenario. These diagrams are often linked to use case realizations in the system's logical development view. Sequence diagrams are valuable for visualizing the flow of control and collaboration among objects in a system, especially in scenarios where the order of interactions is crucial.

In a sequence diagram, objects are represented by lifelines, and the interactions between these objects are visualized through messages exchanged along the lifelines. Each message between objects is annotated with the specific operation or communication occurring, helping to depict the flow of control and collaboration in the system.

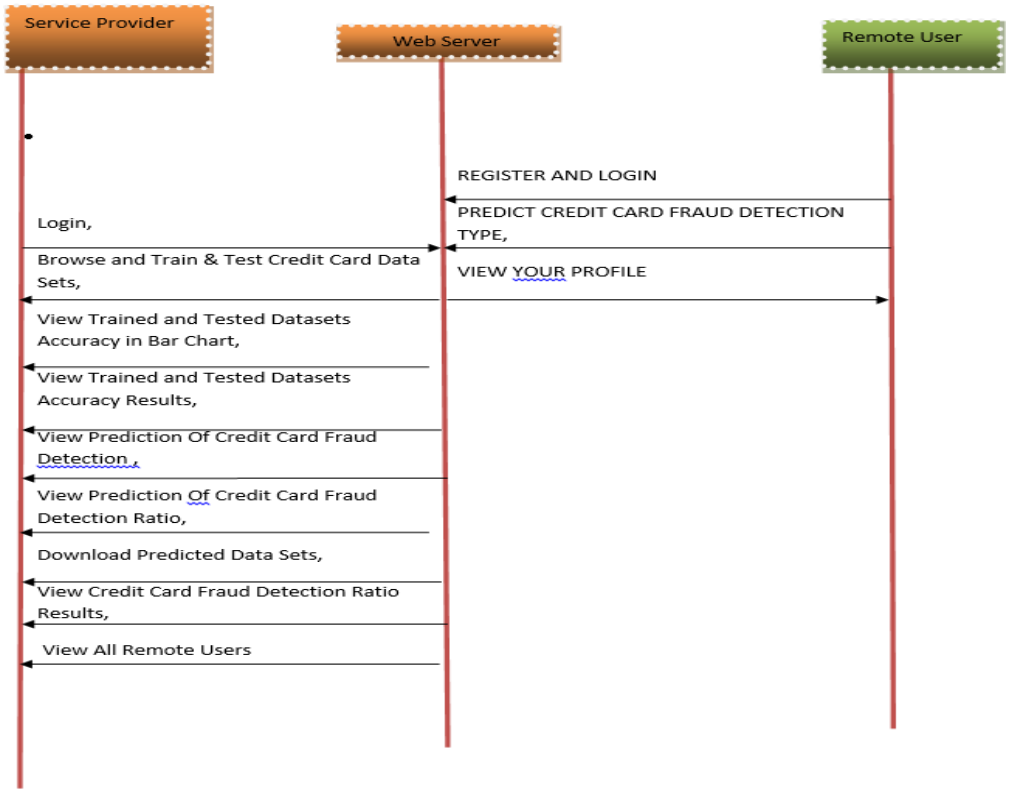


Figure 3.4: Sequence Diagram for Credit Card Fraud Detection using StateofArt Machine Learningand Deep Learning Algorithms

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores. It provides a high-level view of the dynamic aspects of a system, emphasizing the sequence of actions and the decisions made during the execution of a particular process. In an activity diagram, nodes represent activities, which can range from simple operations to complex processes, and arrows depict the flow of control between these activities. Decision points and branches are articulated through conditional and looping constructs, enhancing the diagram's ability to model complex logic and parallel activities.

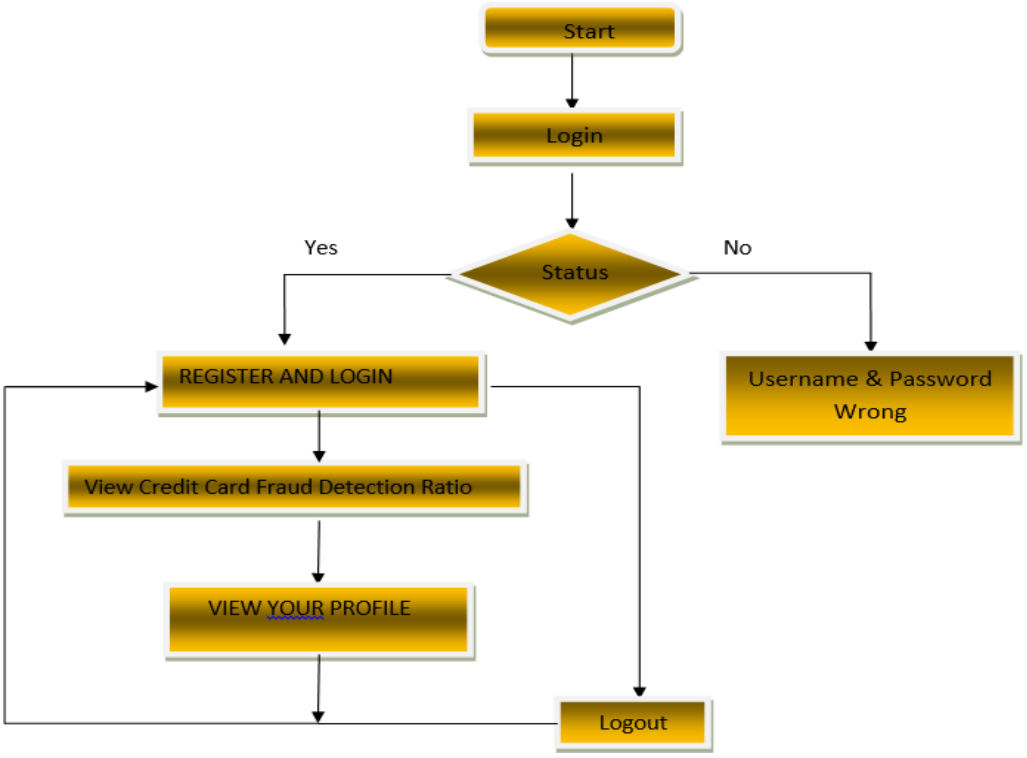


Figure 3.5: Activity Diagram for Credit Card Fraud Detection using State of Art Machine Learning and Deep Learning Algorithms

4.IMPLEMENTATION

4.1 SAMPLE CODE

```
from django.db.models import Count, Avg
from django.shortcuts import render, redirect
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse


import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier

# Create your views here.
from Remote_User.models import
ClientRegister_Model,cc_fraud_detection_type,detection_ratio,detection_accuracy


def serviceproviderlogin(request):
    if request.method  == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password =="Admin":
            detection_accuracy.objects.all().delete()
            return redirect('View_Remote_Users')
return render(request,'SProvider/serviceproviderlogin.html')
def View_CC_Fraud_Detection_Ratio(request):
```

```
    detection_ratio.objects.all().delete()
    ratio = ""
    kword = 'No Credit Card Fraud'
    print(kword)
    obj = cc_fraud_detection_type.objects.all().filter(Q(Prediction=kword))
    obj1 = cc_fraud_detection_type.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)

    ratio12 = ""
    kword12 = 'Credit Card Fraud'
    print(kword12)
    obj12 = cc_fraud_detection_type.objects.all().filter(Q(Prediction=kword12))



    obj112 = cc_fraud_detection_type.objects.all()
    count12 = obj12.count();
    count112 = obj112.count();
    ratio12 = (count12 / count112) * 100
    if ratio12 != 0:
        detection_ratio.objects.create(names=kword12, ratio=ratio12)

    obj = detection_ratio.objects.all()
    return render(request, 'SProvider/View_CC_Fraud_Detection_Ratio.html', {'objs': obj})


def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})
chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})
```

```
def charts1(request,chart_type):
    chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1, 'chart_type':chart_type})


def View_Prediction_Of_CC_Fraud_Detection(request):
    obj =cc_fraud_detection_type.objects.all()
    return render(request, 'SProvider/View_Prediction_Of_CC_Fraud_Detection.html',
{'list_objects': obj})


def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})

def Download_Predicted_DataSets(request):
    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="Predicted_Datasets.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj = cc_fraud_detection_type.objects.all()
```

```
data = obj  # dummy method to fetch data.

        for my_row in data:
      row_num = row_num + 1


      ws.write(row_num, 0, my_row.trans_date, font_style)
      ws.write(row_num, 1, my_row.cc_num, font_style)
      ws.write(row_num, 2, my_row.category, font_style)
      ws.write(row_num, 3, my_row.AMT_TRANS, font_style)
      ws.write(row_num, 4, my_row.first, font_style)
      ws.write(row_num, 5, my_row.last, font_style)
      ws.write(row_num, 6, my_row.gender, font_style)
      ws.write(row_num, 7, my_row.street, font_style)
      ws.write(row_num, 8, my_row.city, font_style)
      ws.write(row_num, 9, my_row.state, font_style)
      ws.write(row_num, 10, my_row.zip, font_style)
      ws.write(row_num, 11, my_row.User_Lat, font_style)
      ws.write(row_num, 12, my_row.User_Long, font_style)
      ws.write(row_num, 13, my_row.city_pop, font_style)
      ws.write(row_num, 14, my_row.Job, font_style)
      ws.write(row_num, 15, my_row.Dob, font_style)
      ws.write(row_num, 16, my_row.trans_num, font_style)
      ws.write(row_num, 17, my_row.merch_lat, font_style)
      ws.write(row_num, 18, my_row.merch_long, font_style)
      ws.write(row_num, 19, my_row.Prediction, font_style)
      wb.save(response)return response


  def train_model(request):
   detection_accuracy.objects.all().delete()
```

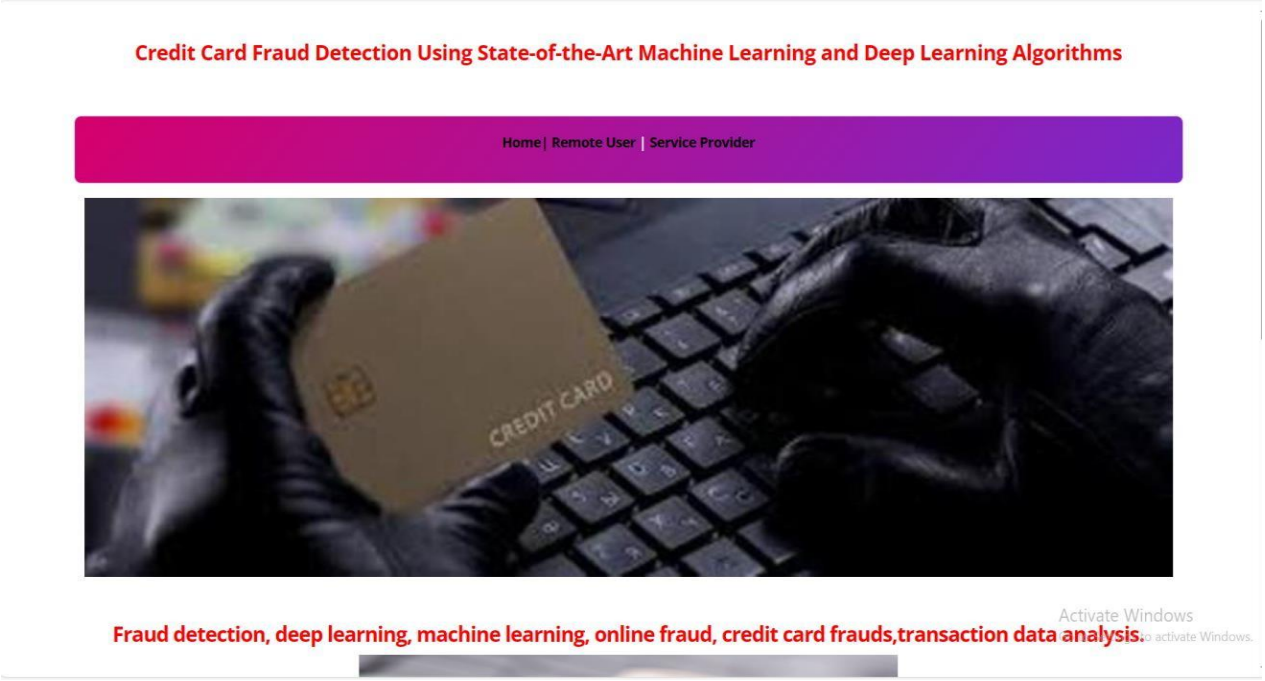
```
df = pd.read_csv('CC_Datasets.csv')


 def apply_response(label):
     if (label== 0):
        return 0  # No Fraud
     elif (label==1):
        return 1 # Fraud


  df['results'] = df['is_fraud'].apply(apply_response)

  cv = CountVectorizer()
  X = df['street']
  y = df['results']

  print("Transaction Number")
  print(X)
  print("Results")
  print(y)

  X = cv.fit_transform(X)

  models = []
  from sklearn.model_selection import train_test_split
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
  X_train.shape, X_test.shape, y_train.shape
  print(X_test)

  print("Naive Bayes")
```

```
from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)

predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100

print(naivebayes)
print(confusion_matrix(y_test, predict_nb))
print(classification_report(y_test, predict_nb))
models.append(('naive_bayes', NB))
detection_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)

# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
```

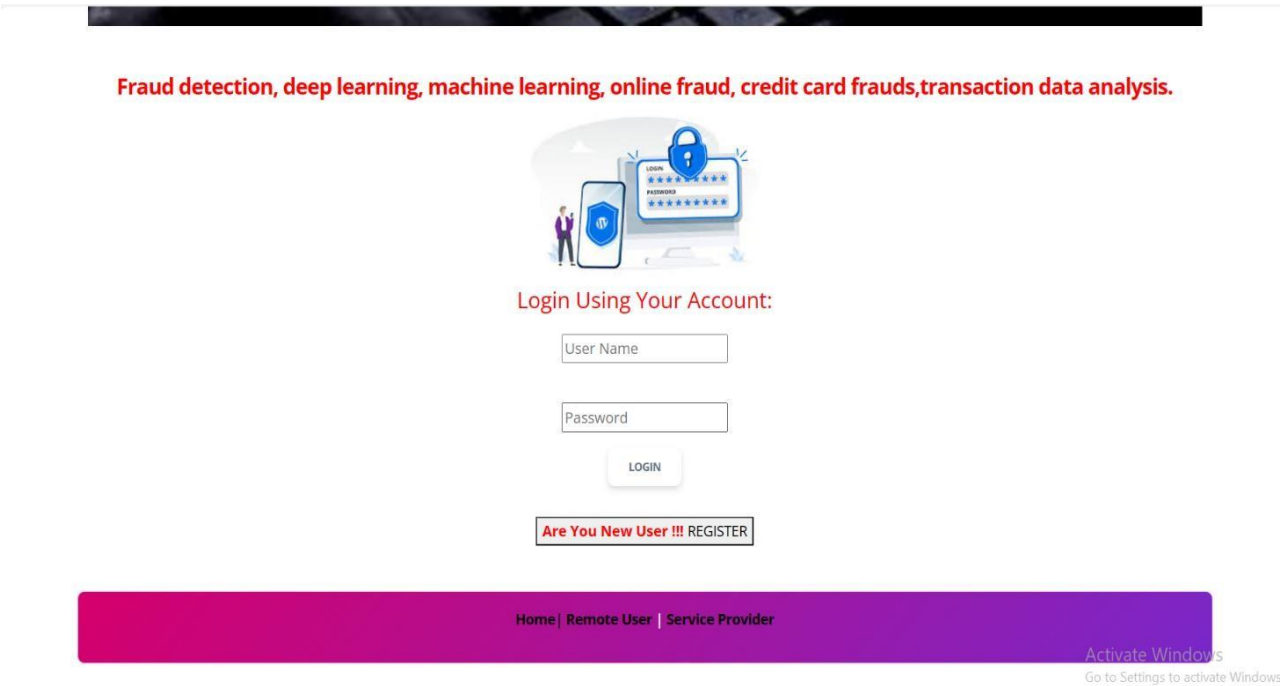
```
print(accuracy_score(y_test, y_pred) * 100)
 print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))



print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
models.append(('logistic', reg))
detection_accuracy.objects.create(names="Logistic Regression",
      ratio=accuracy_score(y_test,y_pred) * 100)
print("Decision Tree
Classifier")
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict = dtc.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, dtcpredict) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, dtcpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, dtcpredict))
models.append(('DecisionTreeClassifier', dtc))
detection_accuracy.objects.create(names="Decision Tree Classifier",
ratio=accuracy_score(y_test, dtcpredict) *

100)print("Gradient Boosting Classifier")

from sklearn.ensemble import GradientBoostingClassifier



clf = GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=1,
random_state=0).fit(y_train)
```

```
clfpredict = clf.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, clfpredict) * 100)

print("CLASSIFICATION REPORT")
print(classification_report(y_test, clfpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, clfpredict))
models.append(('GradientBoostingClassifier', clf))
detection_accuracy.objects.create(names="Gradient Boosting Classifier",


ratio=accuracy_score(y_test, clfpredict) * 100)
csv_format = 'Results.csv'
df.to_csv(csv_format, index=False)
df.to_markdown

obj = detection_accuracy.objects.all()
return render(request,'SProvider/train_model.html', {'objs': obj})
```

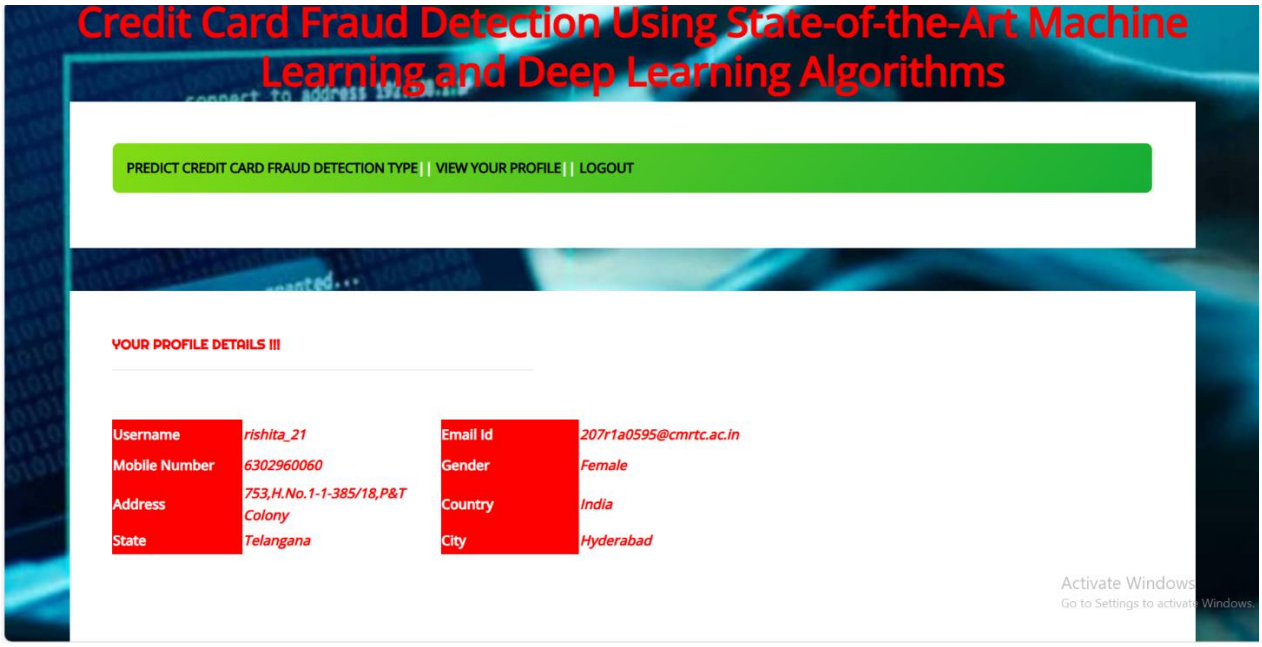
5.SCREENSHOTS



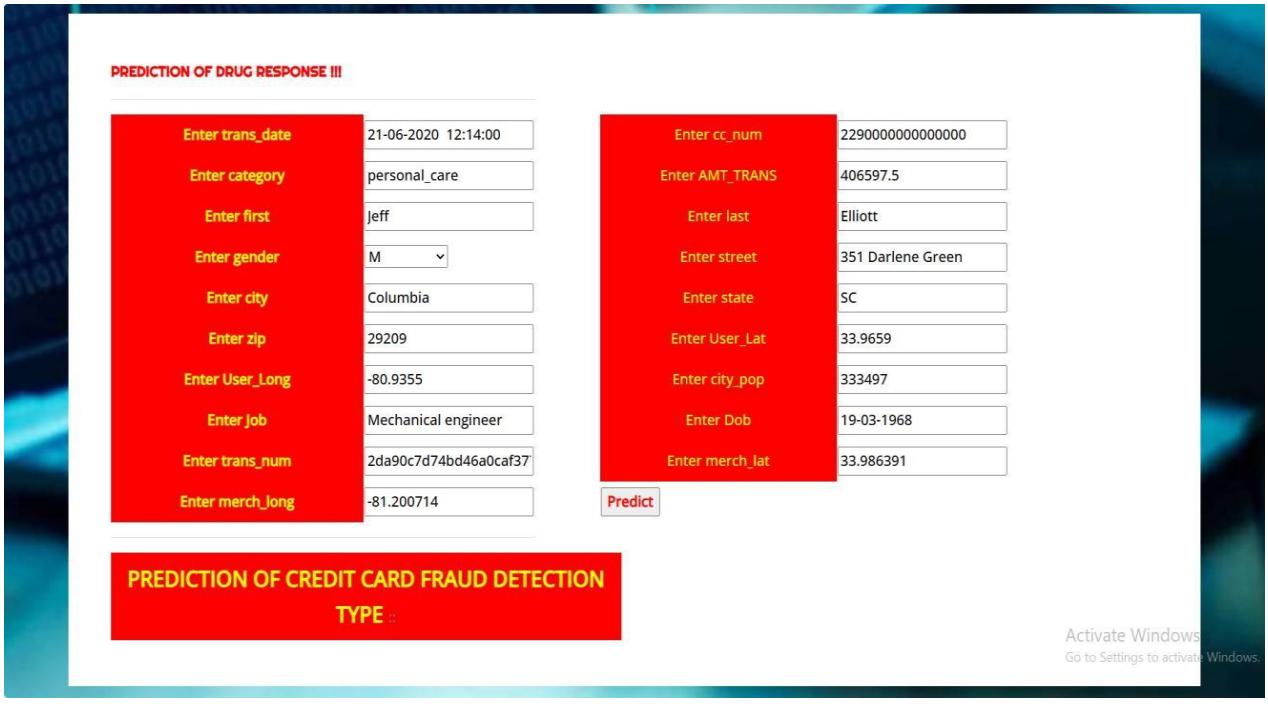
Screenshot 5.1: Home Page



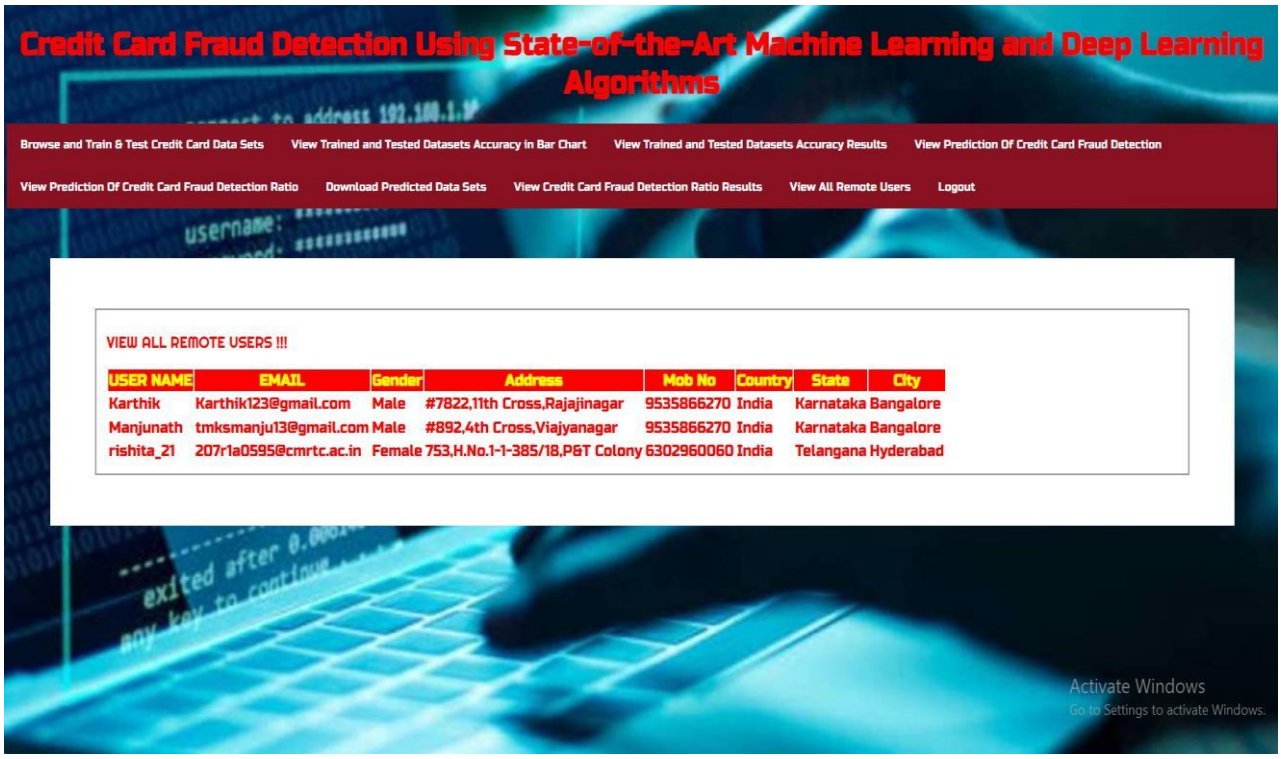
Screenshot 5.2: Login Page



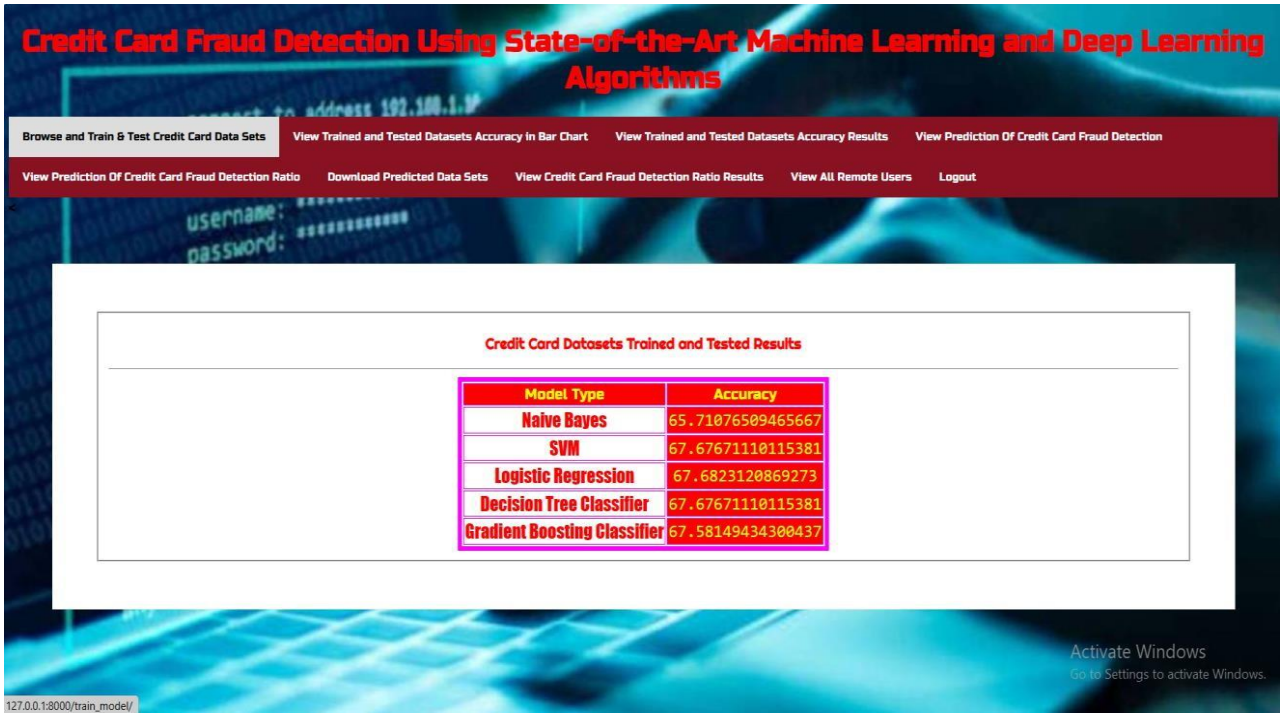
Screenshot 5.3: Profile Details



Screenshot 5.4 : Prediction of credit card fraud detection type

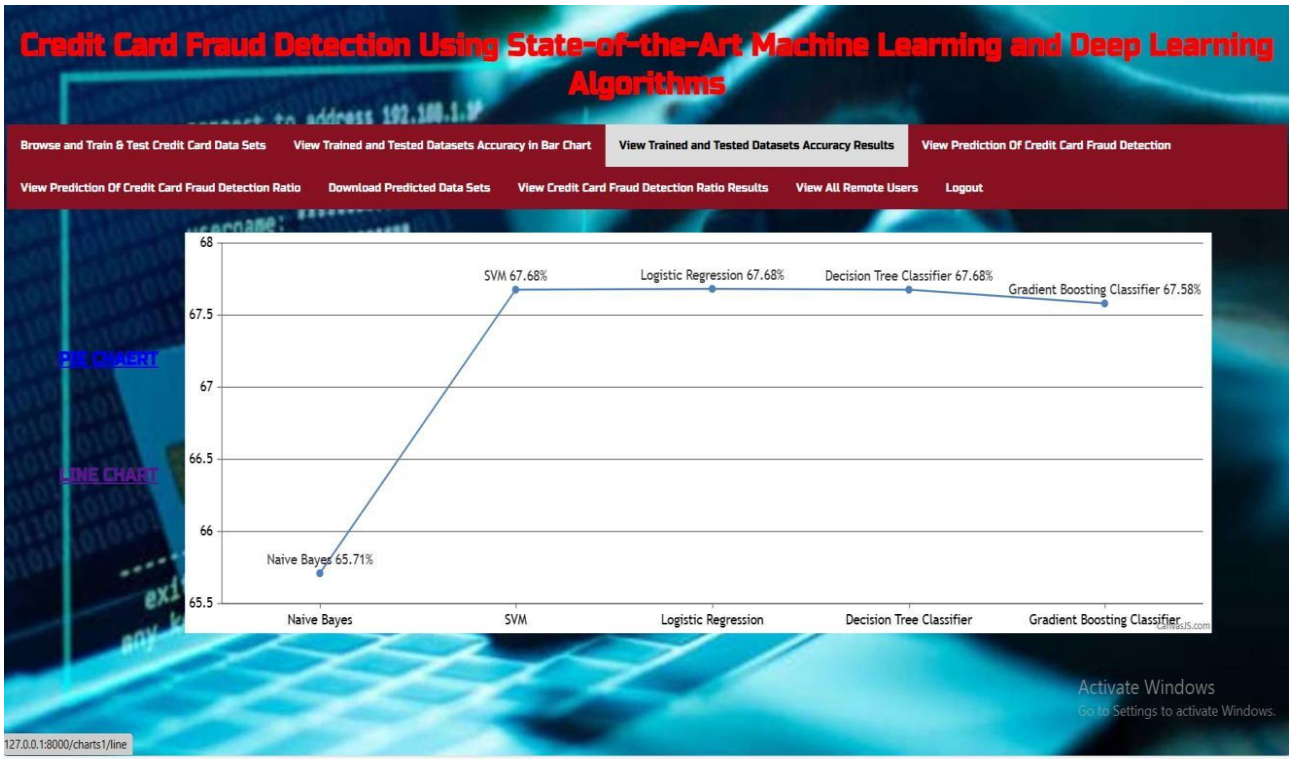


Screenshot 5.5: Viewing the remote users



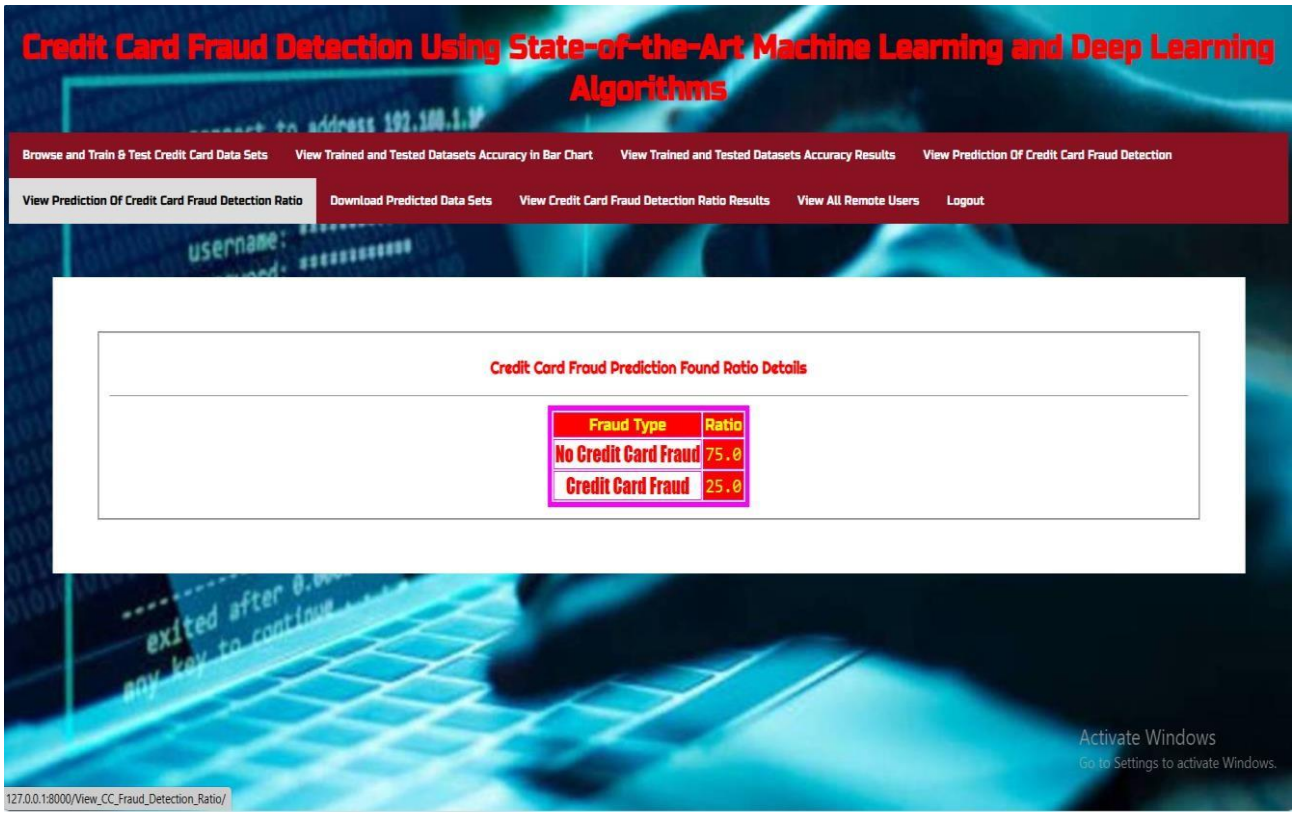
Screenshot 5.6: Datasets trained and tested results

Screenshot 5.7: Trained and tested datasets accuracy in a bar chart

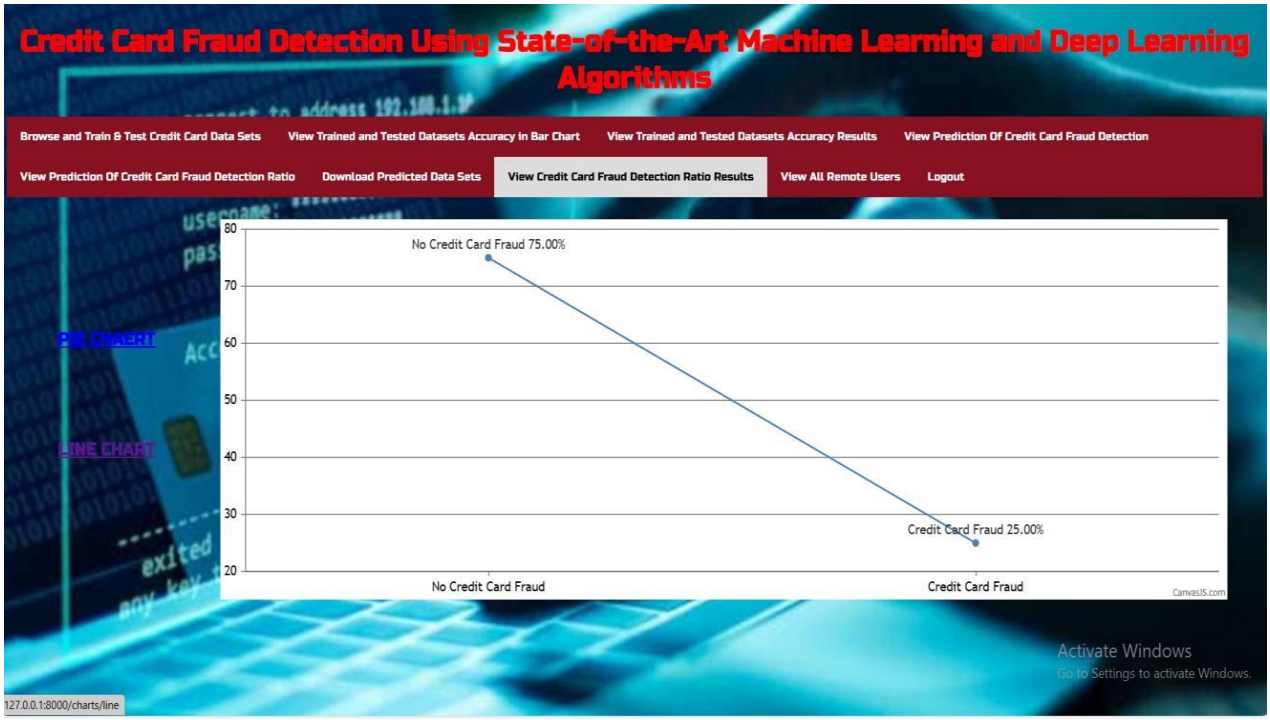


Screenshot 5.8: Trained and tested datasets accuracy in a graph

Screenshot 5.9: credit card fraud detection type details



Screenshot 5.10 : Prediction of credit card fraud detection ratio



Screenshot 5.11: Credit card fraud detection ratio results

6.TESTING

6.1. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2. TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the innerworkings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification .

3.1 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

3.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

3.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.
Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3. 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 isintegrated with a main module and tested for functionality.

6.4. OTHER TESTING METHODOLOGIES

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

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

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 primarilydesigned. 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.

MAINTAINENCE

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 plannedseries 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 leveltests 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.

6.5. TEST CASES

Test case ID	Test case name	Purpose	Input	Output
1	Credit Card fraud detection type	To detect credit card fraud detection type	The user gives Details about transaction and the user	An output is displayed showing the type
2	View profile details	To view the profile	The user gives the required login credentials	An output is details about the user.
3	Service Provider	To provide details about the trained datasets	The user gives the required login credentials	An output is Providing the required results in bar graphs and charts.

7.CONCLUSION

7.1. CONCLUSION

In conclusion, credit card fraud detection using machine learning and deep learning algorithms has proven to be a powerful and effective approach to safeguarding financial transactions. The combination of feature engineering, anomaly detection, and predictive modeling has significantly enhanced the security of credit card transactions, providing both financial institutions and customers with greater peace of mind. However, it's important to remain vigilant in continuously updating and improving these algorithms to stay one step ahead of increasingly sophisticated fraudsters in the ever-evolving landscape of cybercrime.

Ensemble methods, combining multiple algorithms, can enhance overall performance. Feature engineering is crucial for extracting meaningful information, and model interpretability is essential for understanding decision-making processes. Continuous monitoring and adaptation of models are necessary due to evolving fraud patterns. Striking a balance between accuracy and computational efficiency is vital for real- time fraud detection in the dynamic landscape of financial transactions. Regular updates and collaboration with domain experts help ensure robust and adaptive fraud detection systems.

7.2. FUTURE SCOPE

The future scope for this project involves further advancements in fraud detection by exploring novel feature selection techniques tailored to the nuances of credit card transaction data. Continuously refining deep learning architectures, including convolutional neural networks (CNNs), by experimenting with diverse layers and configurations, can unlock even greater predictive power. Additionally, future endeavors could delve into ensemble methods to combine the strengths of multiple models for more robust and accurate fraud detection systems. Further research may also focus on enhancing interpretability and explainability of these models, ensuring transparency and trustworthiness in decision-making processes. Collaboration with financial institutions and regulatory bodies can facilitate the integration of these cutting-edge techniques into real-world applications, ultimately bolstering the security and efficiency of credit card transactions on a broader scale.

7. BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1. REFERENCES

[1] Y. Abakarim, M. Lahby, and A. Attioui, ``An ef_cient real time model for credit card fraud detection based on deep learning," in *Proc. 12th Int. Conf. Intell. Systems: Theories Appl.*, Oct. 2018, pp. 1_7, doi: 10.1145/3289402.3289530.

[2] H. Abdi and L. J. Williams, ``Principal component analysis," *Wiley Inter-discipl. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433_459, Jul. 2010, doi: 10.1002/wics.101.

[3] V. Arora, R. S. Leekha, K. Lee, and A. Kataria, ``Facilitating user authorization from imbalanced data logs of credit cards using arti_cial intelligence," *Mobile Inf. Syst.*, vol. 2020, pp. 1_13, Oct. 2020, doi: 10.1155/2020/8885269.

[4] A. O. Balogun, S. Basri, S. J. Abdulkadir, and A. S. Hashim, ``Performance analysis of feature selection methods in software defect prediction: A search method approach," *Appl. Sci.*, vol. 9, no. 13, p. 2764, Jul. 2019, doi: 10.3390/app9132764.

[5] B. Bandaranayake, ``Fraud and corruption control at education system level: A case study of the Victorian department of education and early childhood development in Australia," *J. Cases Educ. Leadership*, vol. 17, no. 4, pp. 34_53, Dec. 2014, doi: 10.1177/1555458914549669.

[6] J. Bªaszczy«ski, A. T. de Almeida Filho, A. Matuszyk, M. Szelg_, and R. Sªowi«ski, ``Auto loan fraud detection using dominance-based rough set approach versus machine learning methods," *Expert Syst. Appl.*, vol. 163, Jan. 2021, Art. no. 113740, doi: 10.1016/j.eswa.2020.113740.

[7] B. Branco, P. Abreu, A. S. Gomes, M. S. C. Almeida, J. T. Ascensão, and P. Bizarro, ``Interleaved sequence RNNs for fraud detection," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2020, pp. 3101_3109, doi: 10.1145/3394486.3403361.

[8] F. Cartella, O. Anunciacao, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht, ``Adversarial attacks for tabular data: Application to fraud detection and imbalanced data," 2021, *arXiv:2101.08030*.

[9] S. S. Lad, I. Dept. of CSERajarambapu Institute of TechnologyRajaramnagarSangliMaharashtra, and A. C. Adamuthe, ``Malware classi _cation with improved convolutional neural network model," *Int. J. Comput. Netw. Inf. Secur.*, vol. 12, no. 6, pp. 30_43, Dec. 2021, doi: 10.5815/ijcnis.2020.06.03.

[10] V. N. Dornadula and S. Geetha, ``Credit card fraud detection using machine learning algorithms," *Proc. Comput. Sci.*, vol. 165, pp. 631_641, Jan. 2019, doi: 10.1016/j.procs.2020.01.057.

[11] I. Benchaji, S. Douzi, and B. E. Ouahidi, ``Credit card fraud detection model based on LSTM recurrent neural networks," *J. Adv. Inf. Technol.*, vol. 12, no. 2, pp. 113_118, 2021, doi: 10.12720/jait.12.2.113-118.

[12] Y. Fang, Y. Zhang, and C. Huang, ``Credit card fraud detection based on machine learning," *Comput., Mater. Continua*, vol. 61, no. 1, pp. 185_195, 2019, doi: 10.32604/cmc.2019.06144.

[13] J. Forough and S. Momtazi, ``Ensemble of deep sequential models for credit card fraud detection," *Appl. Soft Comput.*, vol. 99, Feb. 2021, Art. no. 106883, doi: 10.1016/j.asoc.2020.106883.

[14] K. He, X. Zhang, S. Ren, and J. Sun, ``Deep residual learning for image recognition," 2015, *arXiv:1512.03385*.

[15] X. Hu, H. Chen, and R. Zhang, ``Short paper: Credit card fraud detection using LightGBM with asymmetric error control," in *Proc. 2nd Int. Conf. Artif. Intell. for Industries (AII)*, Sep. 2019, pp. 91_94, doi: 10.1109/AI4I46381.2019.00030.

[16] J. Kim, H.-J. Kim, and H. Kim, ``Fraud detection for job placement using hierarchical clusters-based deep neural networks," *Int. J. Speech Technol.*, vol. 49, no. 8, pp. 2842_2861, Aug. 2019, doi: 10.1007/s10489-019-01419-2.

[17] M.-J. Kim and T.-S. Kim, ``A neural classi_er with fraud density map for effective credit card fraud detection," in *Intelligent Data Engineering and Automated Learning*, vol. 2412, H. Yin, N. Allinson, R. Freeman, J. Keane, and S. Hubbard, Eds. Berlin, Germany: Springer, 2002, pp. 378_383, doi: 10.1007/3-540-45675-9_56.

[18] N. Kousika, G. Vishali, S. Sunandhana, and M. A. Vijay, ``Machine learning based fraud analysis and detection system," *J. Phys., Conf.*, vol. 1916, no. 1, May 2021, Art. no. 012115, doi: 10.1088/1742-6596/1916/1/012115.

[19] R. F. Lima and A. Pereira, ``Feature selection approaches to fraud detection in e-payment systems," in *E-Commerce and Web Technologies*, vol. 278, D. Bridge and H. Stuckenschmidt, Eds. Springer, 2017, pp. 111_126, doi: 10.1007/978-3-319-53676-7_9.

[20] Y. Lucas and J. Jurgovsky, ``Credit card fraud detection using machine learning: A survey," 2020, *arXiv:2010.06479*.

[21] H. Zhou, H.-F. Chai, and M.-L. Qiu, ``Fraud detection within bankcard enrollment on mobile device based payment using machine learning," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 12, pp. 1537_1545, Dec. 2018, doi: 10.1631/FITEE.1800580.

[22] S. Makki, Z. Assaghir,Y. Taher, R. Haque, M.-S. Hacid, and H. Zeineddine, ``An experimental study with imbalanced classi_cation approaches for credit card fraud detection," *IEEE Access*, vol. 7, pp. 93010_93022, 2019, doi: 10.1109/ACCESS.2019.2927266.

GITHUB LINK:

github.com/praveen7036/Credit-Card-Fraud-Detection-Using-State-of-theArt-Machine-Learning-and-Deep-Learning-Algorithms