CREDIT CARD FRAUD DETECTION USING GENETIC ALGORITHM AND RANDOM FOREST TECHENIQUES

[1]A. Madhu Sudhan Reddy, [2]A. Rohith Reddy, [3]T. Anand Reddy, [4]M. Sai Kumar, [5]MR. R. Azhagusundram

[1,2,3,4]Student, [5]Assistant Professor
Bharath institute of higher education and research

CHAPTER 1

***Abstract*-The advance in technologies such as e-commerce and financial technology (FinTech) applications have sparked an increase in the number of online card transactions that occur on a daily basis. As a result, there has been a spike in credit card fraud that affects card issuing companies, merchants, and banks. It is therefore essential to develop mechanisms that ensure the security and integrity of credit card transactions. In this research, we implement a machine learning (ML) based framework for credit card fraud detection using a real-world imbalanced dataset that were generated from European credit cardholders. To solve the issue of class imbalance, we re-sampled the dataset using the Synthetic Minority over-sampling Technique (SMOTE). This framework was evaluated using the following ML methods: Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), Extreme Gradient Boosting (XGBoost), Decision Tree (DT), and Extra Tree (ET). These ML algorithms were coupled with the Adaptive Boosting (AdaBoost) technique to increase their quality of classification. The models were evaluated using the accuracy, the recall, the precision, the Matthews Correlation Coefficient (MCC), and the Area Under the Curve (AUC). Moreover, the proposed framework was implemented on a highly skewed synthetic credit card fraud dataset to further validate the results that were obtained in this research. The experimental outcomes demonstrated that using the AdaBoost has a positive impact on the performance of the proposed methods. Further, the results obtained by the boosted models were superior to existing methods.**

INTRODUCTION

- In recent years there has been an increase in financial fraud due to the growth of technologies and paradigms such as the e-commerce and the financial technology (FinTech) sectors.
- The evolution of these technologies has sparked an increase in the number of credit card transactions. As a result, there has been a rapid spike in the number financial fraud cases that involved credit cards.
- Credit card Fraud occurs when an unauthorized or undesirable use of a credit card is made by a criminal. This happens when the credit card authentication details are stolen using different types of fraudulent techniques such as intercepting an e-commerce transaction or cloning an existing card.
- It is crucial to implement credit card fraud detection systems that can guarantee the integrity and security of all systems that are involved in fulfilling credit card transactions.
- In this paper, we implement machine learning (ML) algorithms for credit card fraud detection that are evaluated on a real world dataset.
- Moreover, the ML methods that were considered in this research include: Support Vector Machine (SVM), Random Forest (RF), Extra Tree (ET), Extreme Gradient Boosting (XGBoost), Logistic Regression (LR), and Decision Tree (DT).
- These ML methods were evaluated individually in terms of their effectiveness and classification quality. Additionally, the Adaptive Boosting (AdaBoost) algorithm was paired with each method to increase their robustness.

OBJECTIVES

- The objectives of the project is to implement machine learning algorithms to detect credit card fraud detection with respect to time and amount of transaction.

1.1 PROBLEM DEFINITION

The advance in technologies such as e-commerce and financial technology (FinTech) applications have sparked an increase in the number of online card transactions that occur on a daily basis. As a result, there has been a spike in credit card fraud that affects card issuing companies, merchants, and banks. It is therefore essential to develop mechanisms that ensure the security and integrity of credit card transactions. To solve the issue of class imbalance, we re-sampled the dataset using the Synthetic Minority over-sampling TEchnique (SMOTE). This framework was evaluated using the following ML methods: Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), Extreme Gradient Boosting (XGBoost), Decision Tree (DT), and Extra Tree (ET). These ML algorithms were coupled with the Adaptive Boosting (AdaBoost) technique to increase their quality of classification. The models were evaluated using the accuracy, the recall, the precision, the Matthews Correlation Coefficient (MCC), and the Area Under the Curve (AUC).

CHAPTER 2

LITERATURE REVIEW:

2.1 Statistical classification methods in consumer credit scoring

AUTHOR: D. J. Hand and W. E. Henley

Credit scoring is the term used to describe formal statistical methods used for classifying applicants for credit into 'good' and 'bad' risk classes. Such methods have become increasingly important with the dramatic growth in consumer credit in recent years. A wide range of statistical methods has been applied, though the literature available to the public is limited for reasons of commercial confidentiality. Particular problems arising in the credit scoring context are examined and the statistical methods which have been applied are reviewed.

2.2 A comparison of neural networks and linear scoring models in the credit union environment,

AUTHOR: V. S. Desai, J. N. Crook and G. A. Overstreet

The purpose of the present paper is to explore the ability of neural networks such as multilayer perceptrons and modular neural networks, and traditional techniques such as linear discriminant analysis and logistic regression, in building credit scoring models in the credit union environment. Also, since funding and small sample size often preclude the use of customized credit scoring models at small credit unions, we investigate the performance of generic models and compare them with customized models. Our results indicate that customized neural networks offer a very promising avenue if the measure of performance is percentage of bad loans correctly classified. However, if the measure of performance is percentage of good and bad loans correctly classified.

2.3 Credit Scoring Methods. Czech Journal of Economics and Finance,

AUTHOR: M.Vojtek and E. Kocenda

The paper reviews the best-developed and most frequently applied methods of credit scoring employed by commercial banks when evaluating loan applications. The authors concentrate on retail loans – applied research in this segment is limited, though there has been a sharp increase in the volume of loans to retail clients in recent years. Logit analysis is identified as the most frequent credit-scoring method used by banks. However, other nonparametric methods are widespread in terms of pattern recognition. The methods reviewed have potential for application in post-transition countries.

2.4 A survey of credit and behavioral scoring: forecasting: financial risk of lending to customers,

AUTHOR: L. C. Thomas

Credit scoring and behavioural scoring are the techniques that help organisations decide whether or not to grant credit to consumers who apply to them. This article surveys the techniques used both statistical and operational research based to support these decisions. It also discusses the need to incorporate economic conditions into the scoring systems and the way the systems could change from estimating the probability of a consumer defaulting to estimating the profit a consumer will bring to the lending organisation two of the major developments being attempted in th area. It points out how successful has been this under-researched area of forecasting financial risk.

2.5 Neural nets versus conventional techniques in credit scoring in Egyptian banking,

AUTHOR: H. Abdou, J. Pointon, and A. El-Masry

The number of Non-Performing Loans has increased in recent years, paralleling the current financial crisis, thus increasing the importance of credit scoring models. This study proposes a three stage hybrid Adaptive Neuro Fuzzy Inference System credit scoring model, which is based on statistical techniques and Neuro Fuzzy. The proposed model's performance was compared with conventional and commonly utilized models. The credit scoring models are tested using a 10-fold cross-validation process with the credit card data of an international bank operating in Turkey. Results demonstrate that the proposed model consistently performs better than the Linear Discriminant Analysis, Logistic Regression Analysis, and Artificial Neural Network (ANN) approaches, in terms of average correct classification rate and estimated misclassification cost. As with ANN, the proposed model has learning ability; unlike ANN, the model does not stay in a black box. In the proposed model, the interpretation of independent variables may provide valuable information for bankers and consumers, especially in the explanation of why credit applications are rejected.

2.6 A credit scoring model for Vietnams retail banking market

AUTHOR: T.H.T. Dinh, and S. Kleimeier

As banking markets in developing countries are maturing, banks face competition not only from other domestic banks but also from sophisticated foreign banks. Given the substantial growth of consumer credit and increased regulatory attention to risk management, the development of a well-functioning credit assessment framework is essential. As part of such a framework, we propose a credit scoring model for Vietnamese retail loans. First, we show how to identify those borrower characteristics that should be part of a credit scoring model. Second, we illustrate how such a model can be calibrated to achieve the strategic objectives of the bank. Finally, we assess the use of credit scoring models in the context of transactional versus relationship lending.

2.7 Detection A Novel and Successful Credit Card Fraud System Implemented in a Turkish Bank,

AUTHOR: Ekrem Duman,Ayse Buyukkaya,Ilker Elikucuk

We developed a credit card fraud detection solution for a major bank in Turkey. The study was completed in about three years and the developed system has been in use since February 2013. It had a great impact in the rule based fraud detection process used by the bank. Indeed, while eighty percent of the rules have been eliminated and the number of alerts has been reduced to half, a significant increase in fraud detection has been recorded. We were not allowed to give details of the model since such details will

make the business of fraudsters easier but, we tried to give the details of modeling phases that are actually more important for data mining practitioners. The final system was implemented in February 2013 and has been in use since then. The results obtained according to past data analyses are very successf ul and some very important benefits . expected/projected in future. The benefits include decrease in SMS costs and inspection officer salaries and increase in customer and employee satisfactions due to more accurate alerts.

2.8 Real-time credit card fraud detection using computational intelligence,
AUTHOR: Jon T.S. Quah *, M. Sriganesh
Online banking and e-commerce have been experiencing rapid growth over the past few years and show tremendous promise of growth even in the future. This has made it easier for fraudsters to indulge in new and abstruse ways of committing credit card fraud over the Internet. This paper focuses on real-time fraud detection and presents a new and innovative approach in understanding spending patterns to decipher potential fraud cases. It makes use of self-organization map to decipher, filter and analyze customer behavior for detection of fraud.2007 Elsevier Ltd. All rights reserved.Keywords: Self-organizing map; Unsupervised learning; Risk scoring; Transaction

2.9 Detecting credit card fraud by genetic algorithm and scatter search.
AUTHOR: Ekrem Duman a,⇑, M. Hamdi Ozcelik b
In this study we develop a method which improves a credit card fraud detection solution currently beingused in a bank.With this solution each transaction is scored and based on these scores the transactions are classified as fraudulent or legitimate. In fraud detection solutions the typical objective is to minimize the wrongly classified number of transactions. However, in reality, wrong classification of each transaction do not have the same effect in that if a card is in the hand of fraudsters its whole available limit is used up. Thus, the misclassification cost should be taken as the available limit of the card. This is what we aim at minimizing in this study. As for the solution method, we suggest a novel combination of the two well known meta-heuristic approaches, namely the genetic algorithms and the scatter search.

2.10 A cost-sensitive decision tree approach for fraud detection.
AUTHOR: Yusuf Sahin a, Serol Bulkan b, Ekrem Duman c
With the developments in the information technology, fraud is spreading all over the world, resulting in huge financial losses. Though fraud prevention mechanisms such as CHIP&PIN are developed for credit card systems, these mechanisms do not prevent the most common fraud types such as fraudulent credit card usages over virtual POS (Point Of Sale) terminals or mail orders so called online credit card fraud.As a result, fraud detection becomes the essential tool and probably the best way to stop such fraud types. In this study, a new cost-sensitive decision tree approach which minimizes the sum of misclassification costs while selecting the splitting attribute at each non-terminal node is developed and the performance of this approach is compared with the well-known traditional classification models on a real world credit card data set. In this approach, misclassification costs are taken as varying. The results show that thiscost-sensitive decision tree algorithm outperforms the existing well-known methods on the given problem set with respect to the well-known performance metrics such as accuracy and true positive rate, But also a newly defined cost-sensitive metric specific to credit card fraud detection domain. Accordingly,financial losses due to fraudulent transactions can be decreased more by the implementation of this approach in fraud detection systems.

2.11 Industry Paper: A Prototype for Credit Card Fraud Management.
AUTHOR: Alexander Artikis.Nikos Katzouris,NCSR Demokritos,AthensGreece
To prevent problems and capitalise on opportunities before they even occur, the research project SPEEDD proposed a methodology, and developed a prototype for proactive event-driven decisionmaking.We present the application of this methodology to credit card fraud management.The machine learning component of the SPEEDD prototype supports the online construction of fraud patterns, allowing it to e_ciently adapt to the continuously changing fraud types. Moreover, the user interface of the prototype enables fraud analysts to make the most out of the results of automation(complex event processing) and thus reach informed decisions.Unlike most academic research on credit card fraud management, the assessment of the prototype (components) is based on representative transaction datasets/

2.12 Trust Building with Explanation Interfaces.
AUTHOR: Pearl Pu and Li Che
Based on our recent work on the development of a trust model for recommender agents and a qualitative survey, we explore the potential of building users' trust with explanation interfaces We present the major results from the survey, which provided a roadmap identifying the most promising areas for investigating design issues for trust-inducing interfaces .We then describe a set of general principles derived from an in-depth examination of various design dimensions for constructing explanation interfaces, which mos - contribute to trust formation. We present results of a significant scale user study, which indicate that the organization-based explanation is highly effective in building users' trust in the recommendation interface, with the benefit of increasing users' intention to return to the agent and save cognitive effort.

2.13 Credit Card Fraud Detection System Using Intelligent Agents and Enhanced Security Features.
AUTHOR: Amanze, B.C., Asogwa, D.C. & Chukwuneke, C.I,

Credit card fraud can be detected using intelligent agents during transactions. Intelligent Agents aids to obtain a high fraud transaction coverage combined with low false alarm rate, thus providing a better and convenient way to detect frauds. Using intelligent agent, customers' pattern is analyzed and any deviation from the regular pattern is considered to be a fraudulent transaction. In this paper, the intelligent agent is used to detect the fraud when transaction is in progress. The existing fraud detection techniques are not capable to detect fraud at the time when transaction is in progress. As the usage of credit card has increased the credit card fraud has also increased dramatically.

CHAPTER 3
SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIRMENTS

System : Pentium IV 2.4GHZ
Hard Disk : 40 Gb
Ram : 512 Mb

3.2 SOFTWARE REQUIRMENTS

- Operating system : Window 7(32 bit)
- Coding Language : Python
- IDE : Flask Web-app /Jupiter Notebook

ALGORITHM USED

Random Forest Algorithm

- Random forest classifier, it is the most powerful and popularly used algorithm in machine learning. It consists number of individual decision trees and make decisions. It has simplicity and diversity and because of that it can performs both classification and regression tasks.

Steps

- 1. Pick N random records from the dataset.
- 2. Build a decision tree based on these n records.
- 3. Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
- 4. In case of a regression problem, for a new record, Each and every tree in the forest forecasts a value for Y (output).

3.3 LANGUAGE SPECIFICATION

SOFTWARE ENVIRONMENT

Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

```
$ python
Python2.4.3(#1,Nov112010,13:34:43)
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
Type"help","copyright","credits"or"license"for more information.
>>>
```

Type the following text at the Python prompt and press the Enter −

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result −

```
Hello, Python!
```

Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py** Type the following source code in a test.py file −

```
print"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

```
$ python test.py
```

This produces the following result −

```
Hello, Python!
```

Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods −

Sr.No	Methods & Description
1	**GET** Sends data in unencrypted form to the server. Most common method.
2	**HEAD** Same as GET, but without response body
3	**POST** Used to send HTML form data to server. Data received by POST method is not cached by server.
4	**PUT** Replaces all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

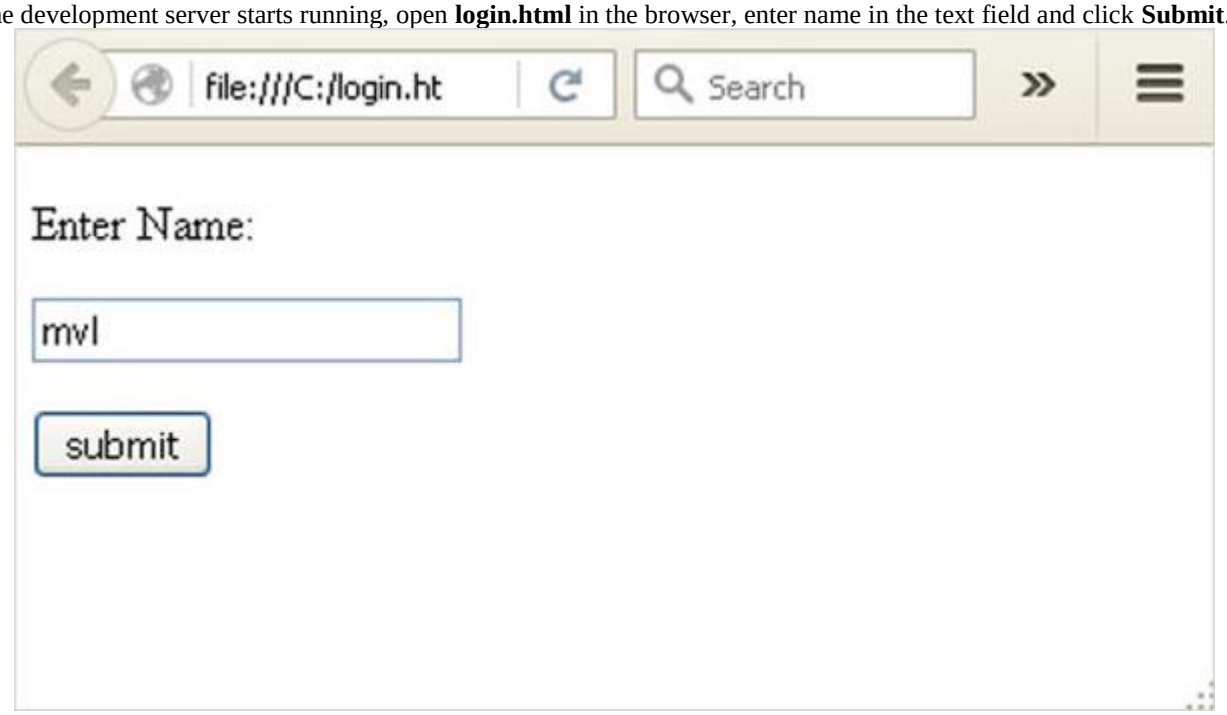
```
<html>
<body>
<formaction="http://localhost:5000/login"method="post">
<p>Enter Name:</p>
<p><inputtype="text"name="nm"/></p>
<p><inputtype="submit"value="submit"/></p>
</form>
</body>
</html>
```

Now enter the following script in Python shell.

```
from flask importFlask, redirect,url_for, request
app=Flask(__name__)
```

```
@app.route('/success/<name>')
def success(name):
return'welcome %s'% name
@app.route('/login',methods=['POST','GET'])
def login():
ifrequest.method=='POST':
user=request.form['nm']
return redirect(url_for('success',name= user))
else:
user=request.args.get('nm')
return redirect(url_for('success',name= user))
if __name__ =='__main__':
app.run(debug =True)
```

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

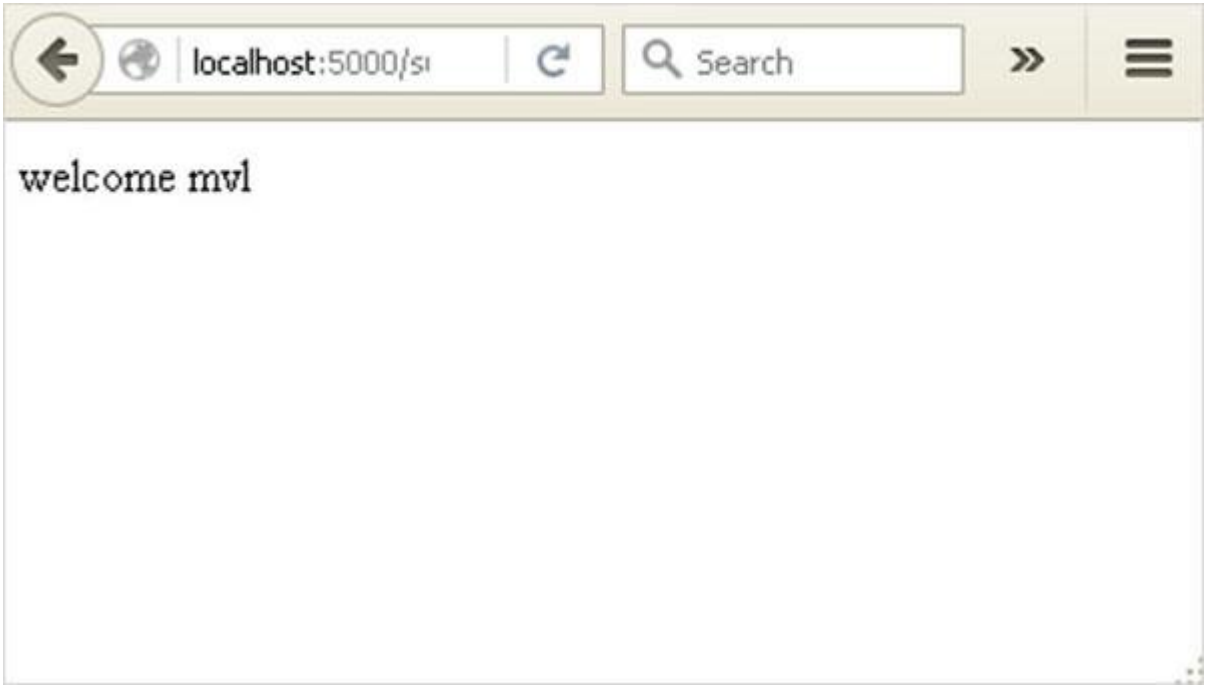


Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of 'nm' parameter obtained from the form data is obtained by −

```
user = request.form['nm']
```

It is passed to **'/success'** URL as variable part. The browser displays a **welcome** message in the window.



Change the method parameter to **'GET'** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of 'nm' parameter is now obtained by −

```
User = request.args.get('nm')
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to 'nm' parameter is passed on to '/success' URL as before.

What is Python?

Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

```
python --version
```

If you find that you do not have python installed on your computer, then you can download it for free from the following website: https://www.python.org/

Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

```
helloworld.py
print("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
C:\Users\Your Name>python helloworld.py
```

The output should read:

```
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
C:\Users\Your Name>python
```

From there you can write any python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
```

Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")
Hello, World!
```

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

Example

```
if 5 > 2:
  print("Five is greater than two!")
```

Python will give you an error if you skip the indentation:

Example

```
if 5 > 2:
print("Five is greater than two!")
```

Comments
Python has commenting capability for the purpose of in-code documentation.
Comments start with a #, and Python will render the rest of the line as a comment:
Example
Comments in Python:

```
#This is a comment.
print("Hello, World!")
```

Docstrings
Python also has extended documentation capability, called docstrings.
Docstrings can be one line, or multiline.
Python uses triple quotes at the beginning and end of the docstring:
Example
Docstrings are also comments:

```
"""This is a
multiline docstring."""
print("Hello, World!")
```

3.8 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and 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 is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

3.8.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.8.2 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. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.8.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

CHAPTER 4
SYSTEM ANALYSIS

4.1 PURPOSE

The purpose is to obtain goods without paying, or to obtain unauthorized funds from an account. Implementation of efficient fraud detection systems has become imperative for all credit card issuing banks to minimize their losses. One of the most crucial challenges in making the business is that neither the card nor the cardholder needs to be present when the purchase is being made. This makes it impossible for the merchant to verify whether the customer making a purchase is the authentic cardholder or not.

4.2 SCOPE

In this proposed project we designed a protocol or a model to detect the fraud activity in credit card transactions. This system is capable of providing most of the essential features required to detect fraudulent and legitimate transactions. As technology changes, it becomes difficult to track the behaviour and pattern of fraudulent transactions.

4.3 EXISTING SYSTEMS

- In case of credit card fraud detection, the existing system is detecting the fraud after fraud has been happen.
- Existing system maintain the large amount of data when customer comes to know about inconsistency in transaction he/she made complaint and then fraud detection system start it working.
- It first tries to detect that fraud has actually occur after that it transactions that was used to fraud detection mechanism developed by master and visa cards

- A machine learning paradigm classification, with Credit Card Fraud Detection being the base.
- Intrusion detections to track fraud location and so on in case of existing system there is no confirmation of recovery of fraud and Customer satisfaction.
- Secure electronic system used to analyse the behaviour of legitimate users.
- Data Mining mechanisms to classify and pre-process the users data.
- Genetic algorithms.

4.4 DISADVANTAGES OF EXISTING SYSTEM

PROS:

- Online alerts are available if suspicious activity is detect on your card. The issuer will notify if any unusual charges come through on the card.
- Credit scores are used in numerous ways and when a person uses a credit card responsibly and makes timely payments without over limits or late payments, their credit score rises.
- By making payments on time and keeping the balance low on the card, a person will save money on the amount of interest charged by the card issuer.
- Eco-friendly, no personal presence is required, it treats with growing technology, time saving.

CONS:

- Each payment system has its limits regarding the maximum amount in the account, the number of transactions per day and the amount of output.
- If Internet connection fails, you can not get to your online account.
- If you follow the security rules the threat is minimal. The worse situation when the system of processing company has been broken, because it leads to the leak of personal data on cards and its owners.
- The information about all the transactions, including the amount, time and recipient are stored in the database of the payment system. And it means the intelligence agency has an access to this information. Sometimes this is the path for fraudulent activities.

4.5 PROPOSED SYSTEM

Our main goal in this project is to construct models to predict whether a credit card transaction is fraudulent. We'll attempt a supervised learning approach. In Proposed system we use Random Forest Algorithm for classification and regression of dataset. First we will collect the Credit Card dataset and analysis will be done on the collected dataset. After the analysis of dataset then cleaning of dataset is required. Generally in any dataset there will be many duplicate and null values will be present, so to remove all those duplicate and null values cleaning process is required. Then we have to split the dataset into two categories as Trained dataset and Testing dataset for comparing and analyzing the dataset. After dividing the dataset we have to apply the Random Forest Algorithm where this algorithm will gives us the better accuracy about the credit card fraud transactions. By applying the Random Forest Algorithm the dataset will be classified into four categories which will be obtained in the form of confusion matrix. Based on the above classification of data performance analysis will be done. In this analysis the accuracy of credit card fraud transactions can be obtained which will be finally represented in the form of graphical representation.

4.6 ADVANTAGES OF PROPOSED SYSTEM

- Online alerts are available if suspicious activity is detecting on your card. The issuer will notify if any unusual charges come through on the card.
- Credit scores are used in numerous ways and when a person uses a credit card responsibly and makes timely payments without over limits or late payments, their credit score rises.
- By making payments on time and keeping the balance low on the card, a person will save money on the amount of interest charged by the card issuer.
- Eco friendly, no personal presence is required, it treats with growing technology, time saving.

CHAPTER 5
SYSTEM DESIGN

5.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

5.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.
The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action

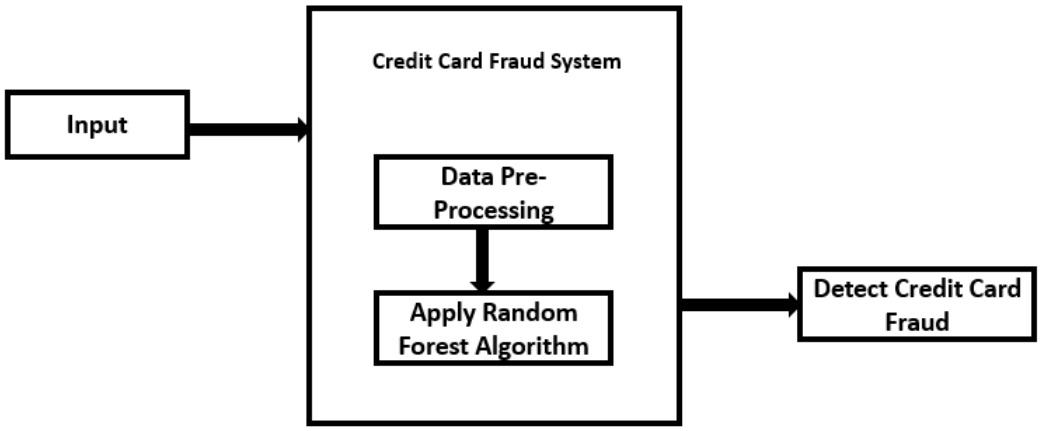
5.3 DATA FLOW DIAGRAM

Data Flow Diagram (DFD) is a two-dimensional diagram that describes how data is processed and transmitted in a system. The graphical depiction recognizes each source of data and how it interacts with other data sources to reach a mutual output. In order to draft a data flow diagram one must

- Identify external inputs and outputs
- Determine how the inputs and outputs relate to each other
- Explain with graphics how these connections relate and what they result in.

5.3.1 Role of DFD:

- It is a documentation support which is understood by both programmers and nonprogrammers. As DFD postulates only what processes are accomplished not how they are performed.
- A physical DFD postulates where the data flows and who processes the data.
- It permits analyst to isolate areas of interest in the organization and study them by examining the data that enter the process and viewing how they are altered when they leave.



UML DIAGRAMS

UML is simply anther graphical representation of a common semanticmodel. UML provides a comprehensive notation for the full lifecycle of object-oriented development.

ADVANTAGES

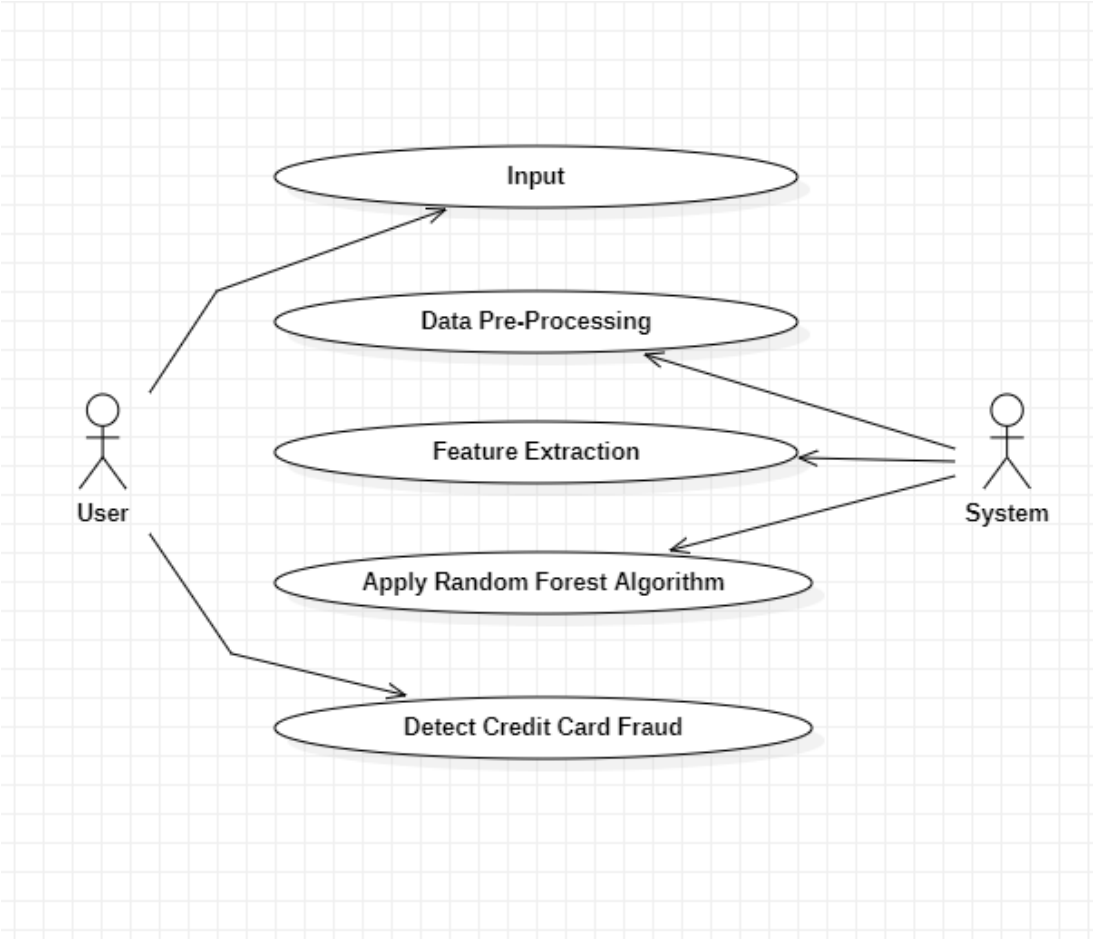
- To represent complete systems (instead of only the software portion) using object oriented concepts
- To establish an explicit coupling between concepts and executable code
- To take into account the scaling factors that are inherent to complex and critical systems
- To creating a modeling language usable by both humans and machines

UML defines several models for representing systems

- The class model captures the static structure
- The state model expresses the dynamic behavior of objects
- The use case model describes the requirements of the user
- The interaction model represents the scenarios and messages flows
- The implementation model shows the work units
- The deployment model provides details that pertain to process allocation

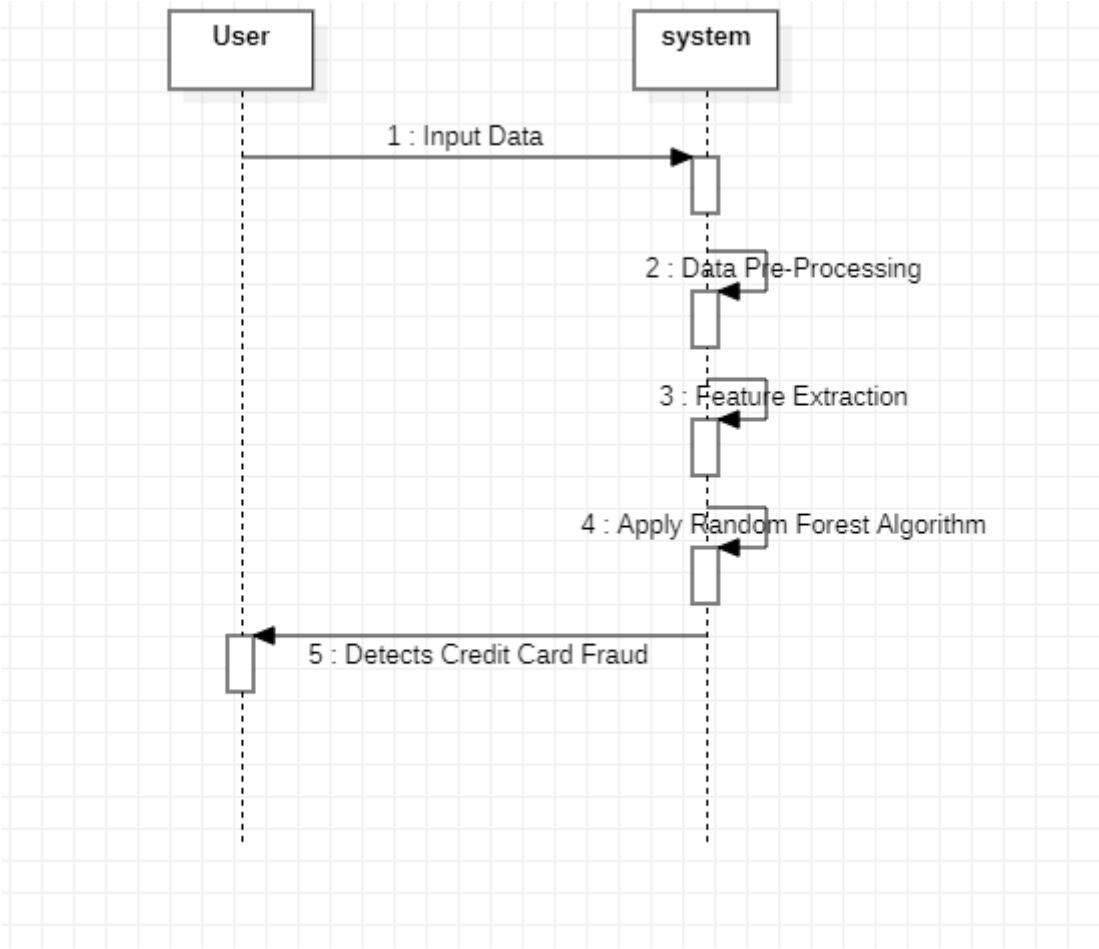
USECASE DIAGRAM

Use case diagrams overview the usage requirement for system. They are useful for presentations to management and/or project stakeholders, but for actual development you will find that use cases provide significantly more value because they describe "the meant" of the actual requirements. A use case describes a sequence of action that provides something of measurable value to an action and is drawn as a horizontal ellipse.



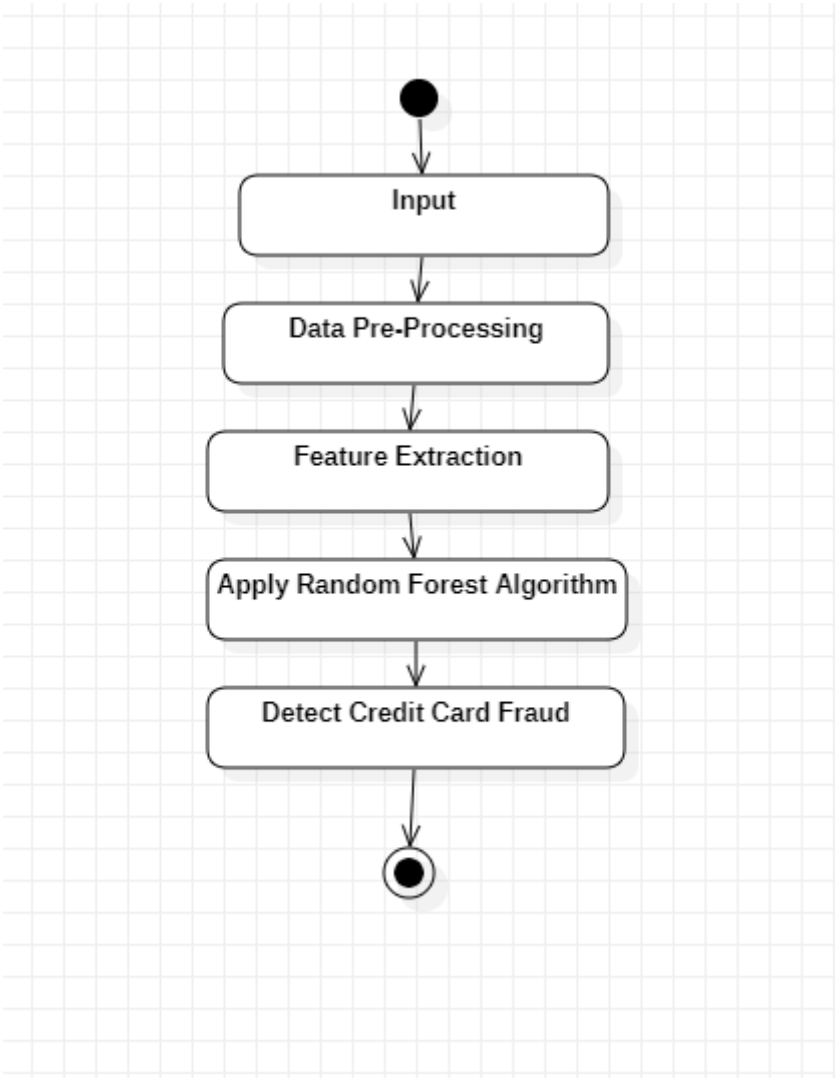
SEQUENCE DIAGRAM

Sequence diagram model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and commonly used for both analysis and design purpose. Sequence diagram are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.



ACTIVITY DIAGRAM

Activity diagram are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. The activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. Activity diagram consist of Initial node, activity final node and activities in between.



CHAPTER 6
MODULES DESCRIPTION

6.1 MODULES

- Data Collection
- Data Preparation
- Model Selection
- Fraud Detection using Random Forest Algorithm
- Accuracy on test set
- Saving the Trained Model

Data Collection Module

- This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.
- There are several techniques to collect the data, like web scraping, manual interventions and etc.

Data Preparation

- We will transform the data. By getting rid of missing data and removing some columns. First we will create a list of column names that we want to keep or retain.
- Next we drop or remove all columns except for the columns that we want to retain.
- Finally we drop or remove the rows that have missing values from the data set.

Model Selection

- While creating a machine learning model, we need two dataset, one for training and other for testing. But now we have only one. So lets split this in two with a ratio of 80:20. We will also divide the dataframe into feature column and label column.
- Here we imported train_test_split function of sklearn. Then use it to split the dataset. Also, *test_size = 0.2*, it makes the split with 80% as train dataset and 20% as test dataset.
- The *random_state* parameter seeds random number generator that helps to split the dataset.

- The function returns four datasets. Labelled them as *train_x, train_y, test_x, test_y*. If we see shape of this datasets we can see the split of dataset.
- We used RandomForestClassifier, which fits multiple decision tree to the data. Finally I train the model by passing *train_x, train_y* to the *fit* method.
- Once the model is trained, we need to Test the model. For that we will pass *test_x* to the predict method.

Fraud Detection using Random Forest Algorithm

- In this module the system must detect whether any fraud has been occurred in the transaction or not. It must also display the user about the result.

Accuracy on test set

- We got a accuracy of 0.93% on test set.

Saving the Trained Model

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or . pkl file using a library like pickle. Make Sure you have pickle installed in your environment. Next, let's import the module and dump the model into .pkl file.

CHAPTER 7
SYSTEM IMPLEMENTATION
7.1 SYSTEM ARCHITECTURE

First the credit card dataset is taken from the source and cleaning and validation is performed on the dataset which includes removal of redundancy, filling empty spaces in columns, converting necessary variable into factors or classes then data is divided into 2 part, one is training dataset and another one is test data set. Now the original sample is randomly partitioned into teat and train dataset. Finally, our models are trained using Classifier algorithm. We use classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered. The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Random forest. These algorithms are very popular in text classification tasks.

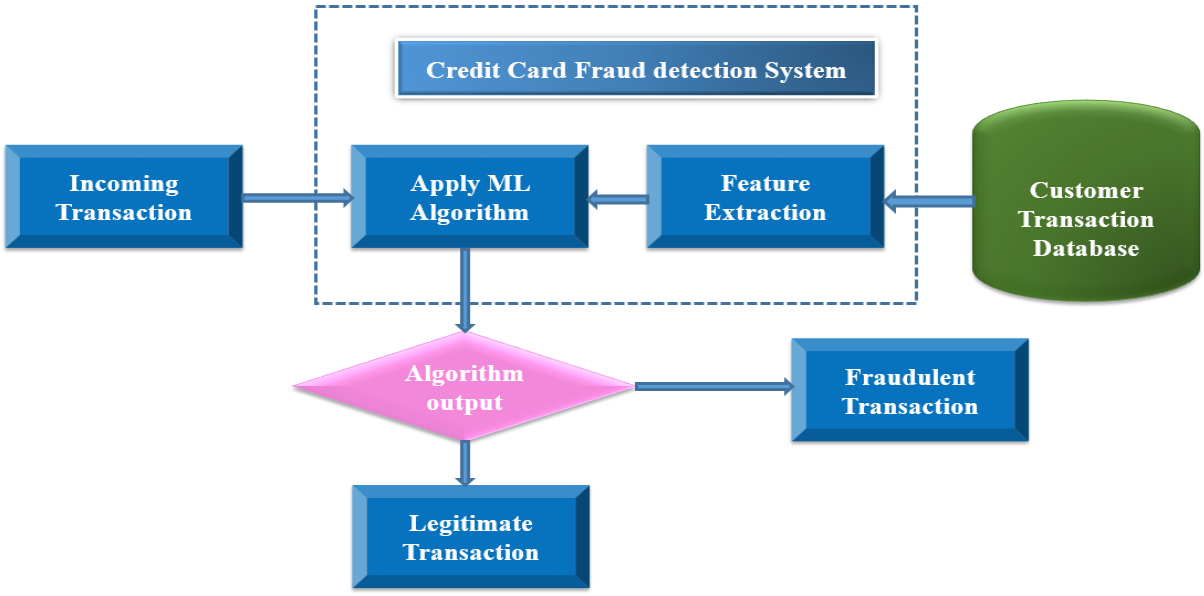


Fig. 7.1 SYSTEM ARCHITECTURE

7.2 Problem Statement:

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. This model is then used to identify whether a new transaction is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications.

CHAPTER 8
SYSTEM TESTING

8.1 Test plan

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

8.2 Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

8.3 Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

8.4 Basics of software testing

There are two basics of software testing: black box testing and white box testing.

8.5 Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

8.6 White box Testing

White box testing is a testing technique that considers the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

8.7 Types of testing: There are many types of testing like

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing

8.7.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

8.7.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

8.7.3 Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

8.7.4 System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

8.7.5 Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

8.7.6 Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

8.7.7 Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

8.7.8 Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

8.7.9 Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing

REQUIREMENT ANALYSIS

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

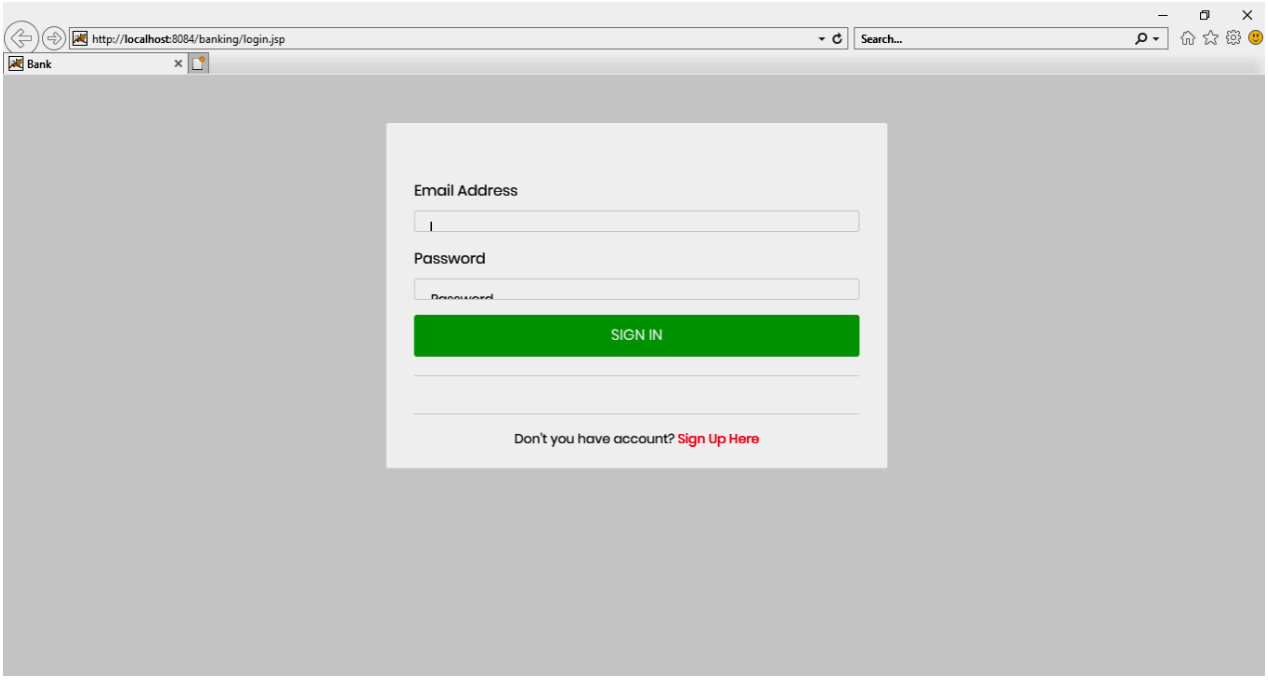
FUNCTIONAL REQUIREMENTS

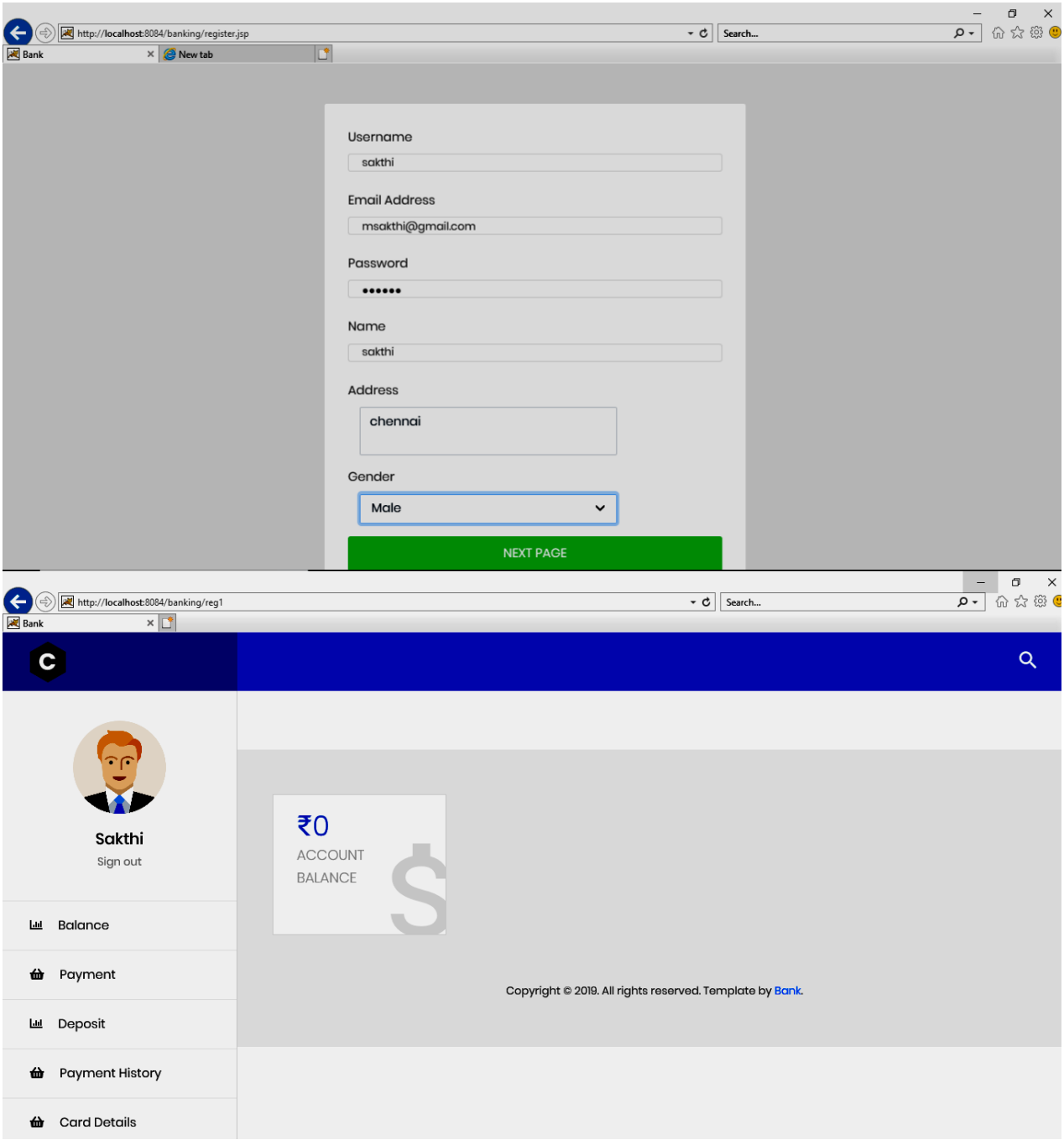
It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

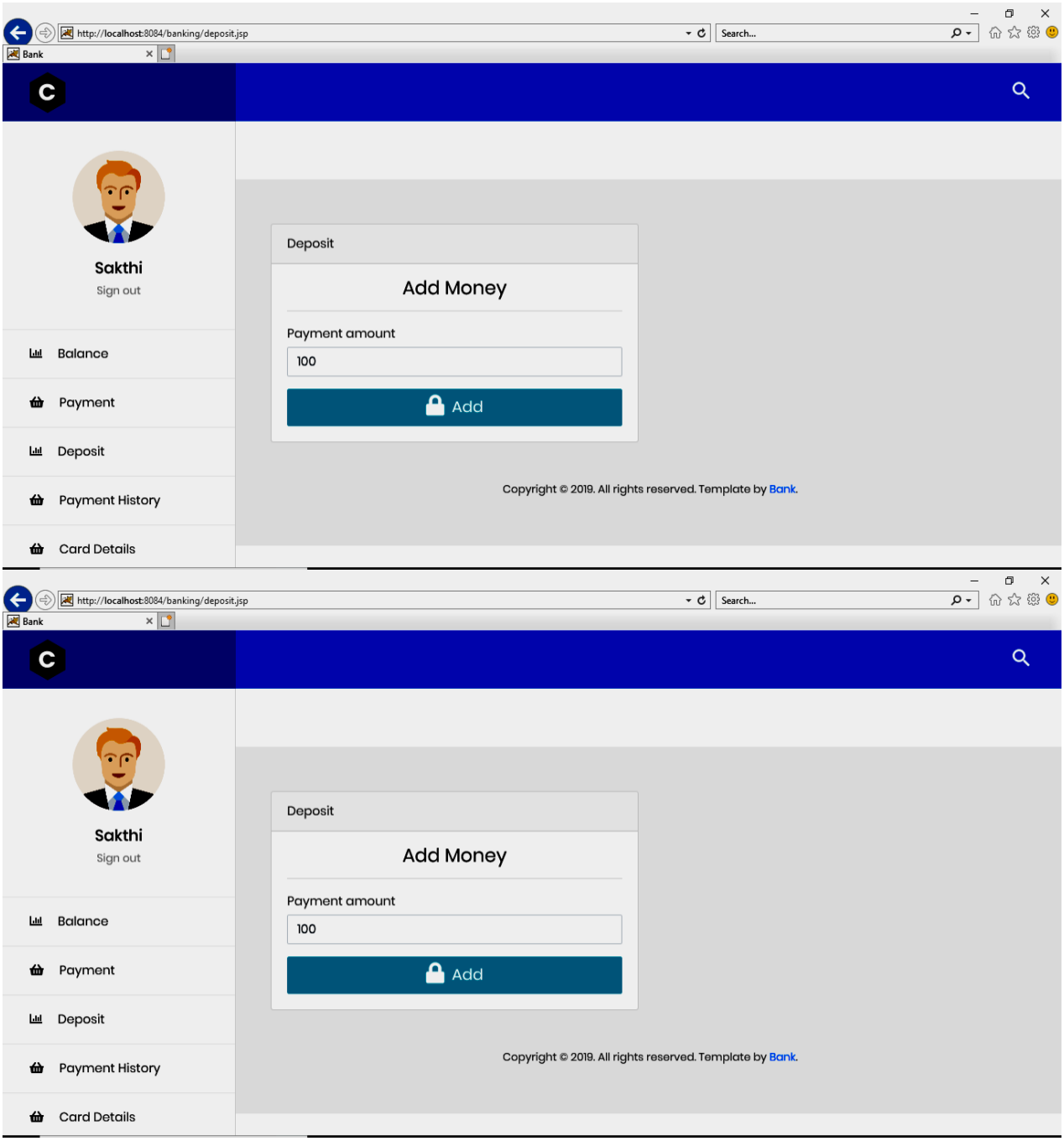
Usability
It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.
Robustness
It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.
Security
The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.
Reliability
It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.
Compatibility
It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.
Flexibility
The flexibility of the project is provided in such a way that is has the ability to run on different environments being executed by different users.
Safety
Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

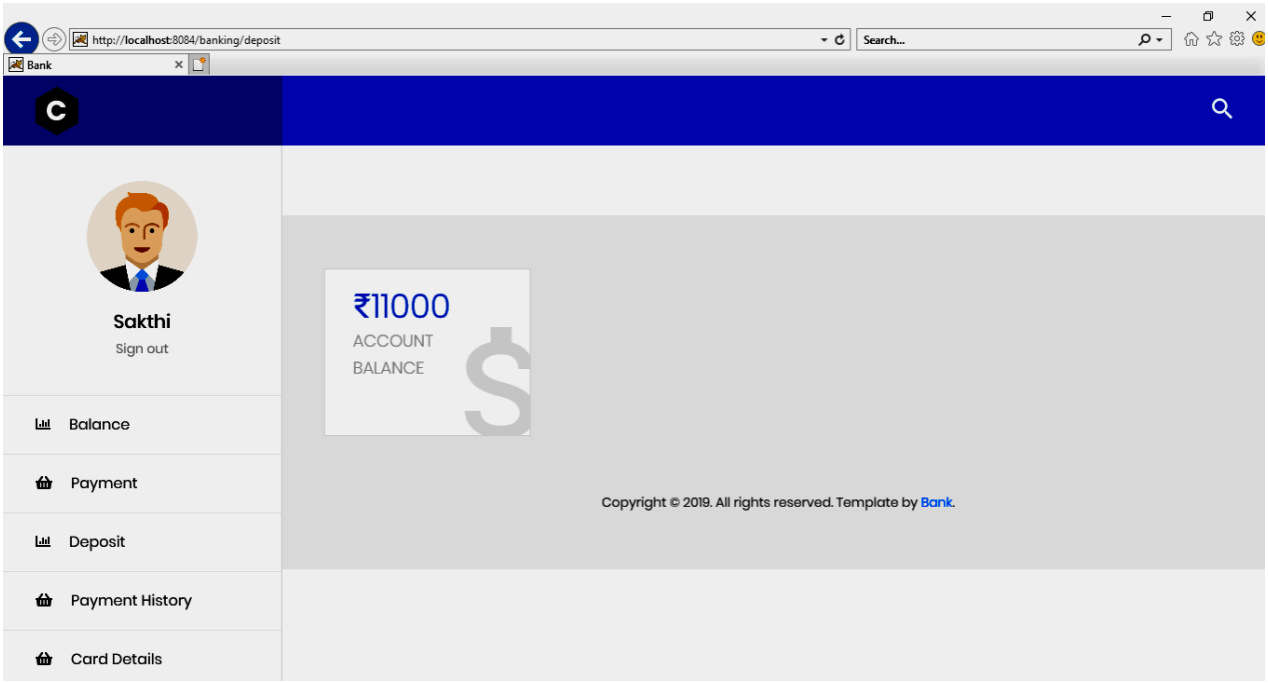
NON- FUNCTIONAL REQUIREMENTS
Portability
It is the usability of the same software in different environments. The project can be run in any operating system.
Performance
These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.
Accuracy
The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.
Maintainability
Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyse, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

IMPLEMENTATION




http://localhost:8084/banking/register.jsp
Search...
Bank
New tab
Username
sakthi
Email Address
msakthi@gmail.com
Password
Name
sakthi
Address
chennai
Gender
Male
NEXT PAGE
http://localhost:8084/banking/reg1
Search...
Bank
Sakthi
Sign out
₹0
ACCOUNT
BALANCE
Balance
Payment
Deposit
Payment History
Card Details
Copyright © 2019. All rights reserved. Template by Bank.


http://localhost:8084/banking/deposit.jsp
Search...
Bank
Sakthi
Sign out
Balance
Payment
Deposit
Payment History
Card Details
Deposit
Add Money
Payment amount
100
Add
Copyright © 2019. All rights reserved. Template by Bank.



CHAPTER 9
CONCLUSION & FUTURE ENHANCEMENT

9.1 CONCLUSION

In this paper, we have discussed that How Random Forest will be useful to detect fraudulent online transaction through credit card. The proposed Fraud Detection System is also scalable for handling vast volumes of transactions data processing. The Random Forest based credit card fraud detection system is not having complex process to perform fraud check like the existing system. Proposed Fraud detection system gives genuine and fast result than existing system. The Random Forest makes the processing of detection very easy and tries to remove the complexity.

9.2 FUTURE ENHANCEMENT

We have discussed an application of Random Forest in credit card fraud detection. The different steps in credit card transaction processing are represented as the underlying stochastic process of an Random Forest. We have study the ranges of transaction amount as the observation symbols, whereas the types of item have been considered to be states of the Random Forest. We have study the suggested a method for finding the spending profile of cardholders, as well as application of this knowledge in deciding the value of observation symbols and initial estimate of the model parameters. It has also been explained how the Random Forest can detect whether an incoming transaction is fraudulent or not. The system is also scalable for handling large volumes of transactions.

REFERENCES:

1. X. Zhang, Y. Han, W. Xu, and Q. Wang, ‘‘HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture,’’ Inf. Sci., May 2019. Accessed: Jan. 8, 2019.
2. N. Carneiro, G. Figueira, and M. Costa, ‘‘A data mining based system for credit-card fraud detection in e-tail,’’ Decis. Support Syst., vol. 95, pp. 91–101, Mar. 2017.
3. B. Lebichot, Y.-A. Le Borgne, L. He-Guelton, F. Oblé, and G. Bontempi, ‘‘Deep-learning domain adaptation techniques for credit cards fraud detection,’’ in Proc. INNS Big Data Deep Learn. Conference, Genoa, Italy, 2019, pp. 78–88.
4. H. John and S. Naaz, ‘‘Credit card fraud detection using local outlier factor and isolation forest,’’ Int. J. Comput. Sci. Eng., vol. 7, no. 4, pp. 1060–1064, Sep. 2019.
5. J. Jurgovsky, M. Granitzer, K. Ziegler, S. Calabretto, P.-E. Portier, L. He-Guelton, and O. Caelen, ‘‘Sequence classification for credit card fraud detection,’’ Expert Syst. Appl., vol. 100, pp. 234–245, Jun. 2018.
6. U. Fiore, A. D. Santis, F. Perla, P. Zanetti, and F. Palmieri, ‘‘Using generative adversarial networks for improving classification effectiveness in credit card fraud detection,’’ Inf. Sci., vol. 479, pp. 448–455, Apr. 2019.
7. Credit Card Fraud Dataset. Accessed: Sep. 4, 2019. [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud/data.
8. I. Mekterović, L. Brkić, and M. Baranovi, ‘‘A systematic review of data mining approaches to credit card fraud detection,’’ WSEAS Trans. Bus. Econ., vol. 15, p. 437, Jan. 2018.
9. M. A. Al-Shabi, ‘‘Credit card fraud detection using auto encoder model in unbalanced datasets,’’ J. Adv. Math. Comput. Sci., vol. 33, no. 5, pp. 1–16, 2019.
10. S. V. S. S. Lakshmi and S. D. Kavilla, ‘‘Machine learning for credit card fraud detection system,’’ Int. J. Appl. Eng. Res., vol. 13, no. 24, pp. 16819–16824, 2018.