



PRESIDENCY COLLEGE
(AUTONOMOUS)
AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

CREDIT CARD FRAUD DETECTION



PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA

RE-ACCREDITED BY NAAC WITH 'A+' GRADE

A Project Report on

“Credit Card Fraud Detection”

Submitted in Partial Fulfillment of the Requirements For the award of the degree

Master of Computer Applications

SUBMITTED BY

Akshata B Nayak

22P01101

PRESIDENCY COLLEGE

Kempapura, Hebbal Bengaluru – 24

Re-accredited by NAAC with 'A+' Grade

DEPARTMENT OF COMPUTER APPLICATIONS



PRESIDENCY COLLEGE
(AUTONOMOUS)
AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

CREDIT CARD FRAUD DETECTION



PRESIDENCY COLLEGE
(AUTONOMOUS)

AFFILIATED TO BENGALURU CITY UNIVERSITY, APPROVED BY AICTE, DELHI & RECOGNISED BY THE GOVT. OF KARNATAKA
RE-ACCREDITED BY NAAC WITH 'A+' GRADE

CERTIFICATE

This is to certify that **AKSHATA B NAYAK** with Register No. **22P01101** has satisfactorily completed the fourth semester MCA Project titled “**CREDIT CARD FRAUD DETECTION**“, as a partial fulfillment of the requirements for the award of the Degree in **Master of Computer Applications**, awarded by **Bengaluru City University**, during the Academic Year **2024**.

Ms, Veena S Badiger
Project Guide :

Dr. Alli
Head of Department
(Department of computer Application)

Examiners

1. -----
2. -----

Reg No: -----

Examination Center: -----

Date of the exam: -----



Declaration

The project titled “**CREDIT CARD FRAUD DETECTION**” developed by me in the partial fulfillment for the award of Master of Computer Application. It is a systematic work carried by us under the guidance of MS, VEENA S BADIGER, Assistant professor, Department of Computer Applications.

I, declare that this same project has not been submitted to any degree or diploma to the Bengaluru City University or any other Universities.

Name of the student: -

Date:-

Signature

Acknowledgement

The development of software is generally bit complex and time consuming task. The goal of developing the project “**CREDIT CARD FRAUD DETECTION**” could not be archived without the encouragements of kindly helpful and supportive people. Here by we convey our sincere thanks for all of them.

I take this opportunity to express my gratitude to people who had been instrumental in the successful completion of this project.

I am thankful to our management trustee for providing us an opportunity to work and complete the project successfully.

I wish to express my thanks to our **Principal Dr. Suchithra R Nair** for her support to the project work. I would like to acknowledge my gratitude to our HOD of Master of Computer Applications. **Dr. Alli** for her encouragement and support. Without their encouragement and guidance this project would not have materialized.

The guidance and support received from our Internal Guide **Ms. Veena S Badiger** who contributed to this project, was vital for the success of the project. We are grateful for his constant support and help.

INDEX

SL.NO	DESCRIPTION	PAGE NO
01.	INTRODUCTION	1-4
02.	REQUIREMENT ANALYSIS	5-9
2.1	LITERATURE SURVEY	9-11
2.2	PROPOSED SYSTEM	11-13
03.	SYSTEM SPECIFICATIONS	14-15
04.	H/W and S/W CONFIGURATIONS	16-17
05.	SOFTWARE PROFILE	18-36
06.	SYSTEM DESIGN	37-39
6.1	DESIGN DIAGRAMS – CLASS, USE-CASE, DFD, ERD, SEQUENCE, SCHEMA, ACTIVITY, DEPLOYMENT	40-44
07.	SYSTEM TESTING	45-47
7.1	SAMPLE TEST CASES	48-51
08.	UI DESIGN	52-57
09.	SYSTEM IMPLEMENTATION – CODING	58-76
10.	CONCLUSION and FUTURE ENHANCEMENTS	77-79
11.	BIBLIOGRAPHY	80-83



Introduction

1. INTRODUCTION

1.1 Overview

Credit card is the most popular mode of payment. As the number of credit card users is rising world-wide, the identity theft is increased, and frauds are also increasing. In the virtual card purchase, only the card information is required such as card number, expiration date, secure code, etc. Such purchases are normally done on the Internet or over telephone. To commit fraud in these types of purchases, a person simply needs to know the card details. The mode of payment for online purchase is mostly done by credit card. The details of credit card should be kept private. To secure credit card privacy, the details should not be leaked. Different ways to steal credit card details are phishing websites, steal/lost credit cards, counterfeit credit cards, theft of card details, intercepted cards etc. For security purpose, the above things should be avoided. In online fraud, the transaction is made remotely and only the card's details are needed. The simple way to detect this type of fraud is to analyze the spending patterns on every card and to figure out any variation to the "usual" spending patterns. Fraud detection by analyzing the existing data purchase of cardholder is the best way to reduce the rate of successful credit card frauds. As the data sets are not available and also the results are not disclosed to the public. The fraud cases should be detected from the available data sets known as the logged data and user behavior. At present, fraud detection has been implemented by a number of methods such as data mining, statistics, and artificial intelligence.

1.2 Problem Statement

The card holder faced a lot of trouble before the investigation finish. And also, as all the transaction is maintained in a log, we need to maintain huge data, and also now a day's lot of online purchase are made so we don't know the person how is using the card online, we just capture the ip address for verification purpose. So there need a help from the cyber- crime to investigate the fraud.

1.3 Significance and Relevance of Work

Relevance of work includes consideration of all the possible ways to provide a solution to given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can easily done based on the future upcoming requirements like Machine learning techniques.

There are two important categories of machine learning techniques to identify the frauds in credit card transactions: supervised and unsupervised learning model. In supervised approach, early transactions of credit card are labelled as genuine or frauds.

Then, the scheme identifies the fraud transaction with credit card data.

1.4 Objectives

Features Extractions from recognized facial information then data will be normalized for extracting features of good Objective of the project is to predict the fraud and fraud less transaction with respect to the time and amount of the transaction using classification machine learning algorithms such as SVM, Random Forest, Decision tree and confusion matrix in building of the complex machine learning models.

1.5 Methodology

First the Dataset is read. Exploratory Data Analysis is performed on the dataset to clearly understand the statistics of the data, Feature selection is used, A machine learning model is developed. Train and test the model and analysis the performance of the model using certain evaluation techniques such as accuracy, confusion matrix, precision etc.



Software Requirement Analysis

2. SOFTWARE REQUIREMENT ANALYSIS

System Requirement Specification:

System Requirement Specification (SRS) is a fundamental document, which forms the foundation of the software development process. The System Requirements Specification (SRS) document describes all data, functional and behavioral requirements of the software under production or development. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time. The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and non-functional requirements only. It doesn't offer design suggestions, possible solutions to technology or business issues, or any other information other than what the development team understands the customer's system requirements.

Hardware specification

- RAM: 8GB
- Processor: i5 or above
- Hard Disk: 10GB

Software specification

- CLIENT SIDE TECHNOLOGIES : HTML, CSS, JavaScript
- SERVER SIDE TECHNOLOGIES : PYTHON
- DATABASE : DATABASE
- APP SERVER : XAMPP
- IDE : JUPYTER NOTEBOOK
- TESTING & BUILD TOOLS : VSCODE

Functional Requirements:

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:

- Collect the Datasets.
- Train the Model.
- Predict the results

Non-Functional Requirements :

- The system should be easy to maintain.
- The system should be compatible with different platforms.
- The system should be fast as customers always need speed.
- The system should be accessible to online users.
- The system should be easy to learn by both sophisticated and novice users.
- The system should provide easy, navigable and user-friendly interfaces.
- The system should produce reports in different forms such as tables and graphs for easy visualization by management.
- The system should have a standard graphical user interface that allows for the online

Performance Requirement:

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

SYSTEM ANALYSIS :

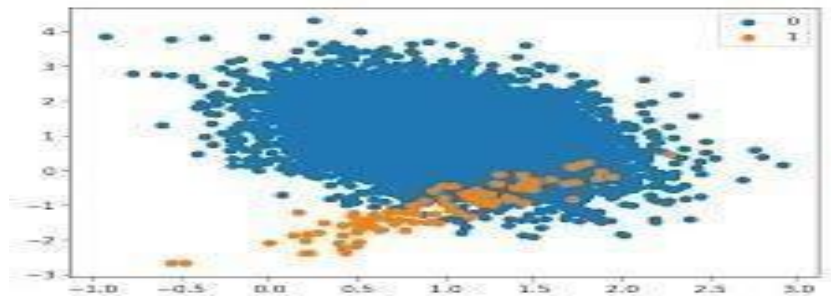
Systems analysis is the process by which an individual studies a system such that an information system can be analyzed, modeled, and a logical alternative can be chosen. Systems analysis projects are initiated for three reasons: problems, opportunities, and directives

Existing System :

- Since the credit card fraud detection system is a highly researched field, there are many different algorithms and techniques for performing the credit card fraud detection system.
- One of the earliest systems is CCFD system using Markov model. Some other various existing algorithms used in the credit cards fraud detection system includes Cost sensitive decision tree (CSDT).
- credit card fraud detection (CCFD) is also proposed by using neural networks. The existing credit card fraud detection system using neural

network follows the whale swarm optimization algorithm to obtain an incentive value.

- It uses BP network to rectify the values which are found error



fraud and Non Fraud Representation

Limitations :

If the time interval is too short, then Markov models are inappropriate because the individual displacements are not random, but rather are deterministically related in time. This example suggests that Markov models are generally inappropriate over sufficiently short time intervals.

2.1 LITERATURE SURVEY :

Credit Card Fraud Detection Techniques : Data and Technique Oriented Perspective

In this paper, after investigating difficulties of credit card fraud detection, we seek to review the state of the art in credit card fraud detection techniques, datasets and evaluation criteria.

Disadvantages :

- Lack of standard metrics

Detection of credit card fraud: State of art :

In this paper, we propose a state of the art on various techniques of credit card fraud detection. The purpose of this study is to give a review of implemented techniques for credit card fraud detection, analyses their incomes and limitless, and synthesize the finding in order to identify the techniques and methods that give the best results so far.

Disadvantages :

- Lack of adaptability

Credit card fraud detection using machine learning algorithm

The main aim of the paper is to design and develop a novel fraud detection method for Streaming Transaction Data, with an objective, to analyze the past transaction details of the customers and extract the behavioral patterns.

Disadvantages :

- Imbalanced Data

Fraudulent Transaction Detection in Credit Card by Applying Ensemble Machine Learning techniques

In this study, the application of various classification models is proposed by implementing machine learning techniques to find out the accuracy and other performance parameters to identify the fraudulent transaction.

Disadvantages :

- Overlapping data.

Detection of Credit Card Fraud Transactions using Machine Learning Algorithms and Neural Networks

Authors: Deepti Dighe, Sneha Patil, Shrikant Kokate

Credit card fraud resulting from misuse of the system is defined as theft or misuse of one's credit card information which is used for personal gains without the permission of

the card holder. To detect such frauds, it is important to check the usage patterns of a user over the past transactions. Comparing the usage pattern and current transaction, we can classify it as either fraud or a legitimate transaction.

Disadvantages :

- Different misclassification importance

Credit card fraud detection using machine learning algorithms and cyber security

Authors: Jiatong Shen

As they have the same accuracy the time factor is considered to choose the best algorithm. By considering the time factor they concluded that the Ada boost algorithm works well to detect credit card fraud.

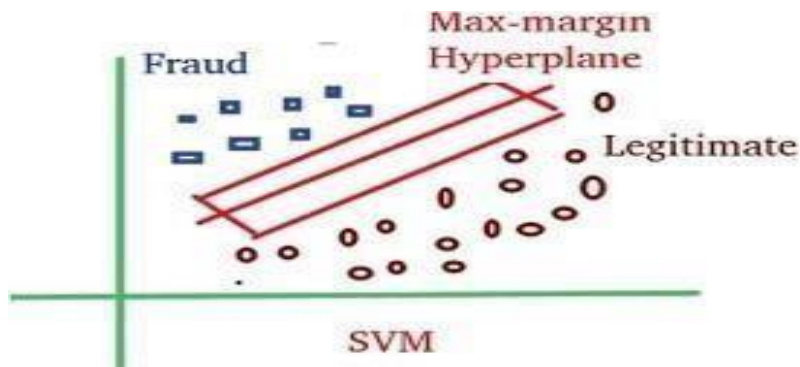
Disadvantages:

- Accuracy is not getting perfectly

2.2 Proposed System:

Support Vector Machine:

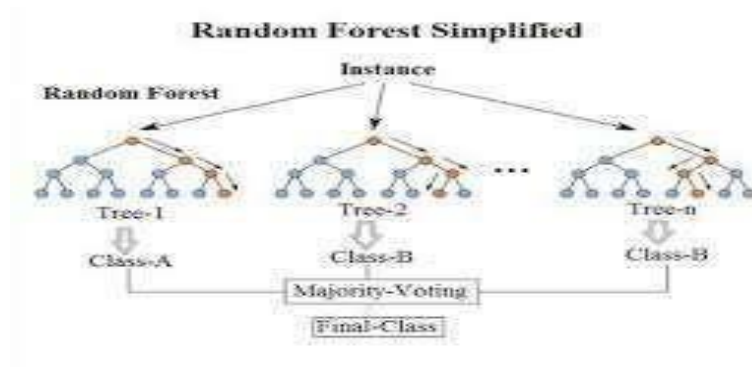
SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data are transformed in such a way that the separator could be drawn as a hyperplane Training regression model and finding out the best one.



SVM Representation

Random Forest Classifier:

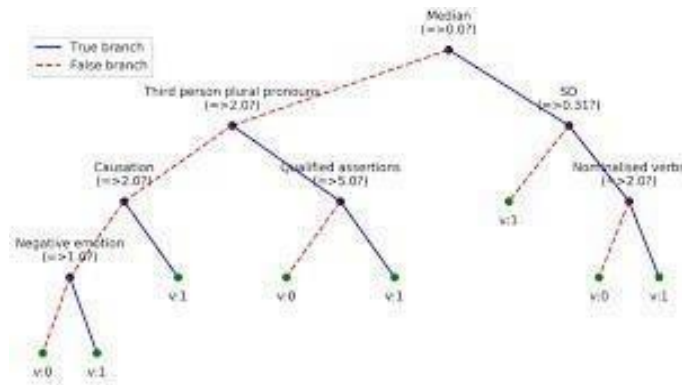
Features are cheekbone to jaw width, width to upper facial height ratio, perimeter to area ratio, eye size, lower face to face height ratio, face width to lower face height ratio and mean of eyebrow height. The extracted features are normalized and finally subjected to support regression.



Simplified Random Forest algorithm

Decision Tree :

A decision tree is a type of supervised machine learning used to categorize or make predictions based on how a previous set of questions were answered. The model is a form of supervised learning, meaning that the model is trained and tested on a set of data that contains the desired categorization.



Decision tree Algorithm

Advantages :

- Support vector machine works comparably well when there is an understandable margin of dissociation between classes.
- SVM is effective in instances where the number of dimensions is larger than the number of specimens.
- Simple to understand and to interpret.
- Requires little data preparation.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of datapoints used to train the tree.
- Able to handle both numerical and categorical data.
- Random forest classifier can be used to solve for regression or classification problems.
- The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample.



Software Requirement Specifications

3. SOFTWARE REQUIREMENT SPECIFICATION

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform.

An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations. Parameters such as operating speed, response time, availability, portability, maintainability, footprint, security and speed of recovery from adverse events are evaluated.

The requirements of this web application (Credit Card Fraud Detection) developed by us is very well understood and documented initially when the process started, accordingly the hardware and software requirements are chosen. The same can be referred in the next section of this report. This web application runs in the Apache Tomcat Server, which can be accessed by web clients through the Internet from anywhere in the world. The SRS parameters are adopted in this system, with the detailed feasibility study. While developing, this system tested as the localhost, through the Mozilla fire fox browser. Also, few other popular web browsers are technology



H/W and S/W Configurations

S



4.SOFTWARE AND HARDWARE CONFIGURATION

HARDWARE	
Processor	i5 or above
RAM	8GB
HDD	10GB
SOFTWARE	
CLIENT SIDE TECHNOLOGIES	HTML, CSS, JavaScript
SERVER SIDE TECHNOLOGIES	PYTHON
DATABASE	MYSQL
APP SERVER	XAMPP
IDE	JUPYTER NOTEBOOK
TESTING & BUILD TOOLS	VSCODE



Software Profile

5. SOFTWARE PROFILE

ABOUT HTML/XHTML

HTML (Hyper Text Markup Language) :

HTML was created by Tim Berners-Lee at European Laboratory for Particle Physics (CERN) in late 1980's. Developed by the World Wide Web Consortium; HTML or the Hyper Text Markup Language, as its name suggests, is a markup language for Web pages. Today, the most important component of any web page is the text-based information that it contains. The markup tags of HTML define the structure of the text-based information of a web page. HTML tags are used to denote various text-based information of a webpage as paragraphs, headings, links, bullet points etc. Various HTML tags can also be used to supplement the text with images, forms and other objects. HTML tags are for browsing; they are meant for interactions between humans and computers.

Hardware Interface:

Client side:

- ⊙ Processor: Intel Core i3.
- ⊙ RAM: 2GB.
- ⊙ Network Interface.

Server Side:

- ⊙ Processor: Intel Core i3.
- ⊙ RAM: 2GB.
- ⊙ Disk space: 2GB.

Software Interface

Client side:

- ⊙ Windows XP/ Vista/ Win 7.
- ⊙ Internet Explorer 6.0 or above
- ⊙ Network Interface.

Server Side:

- ⊙ Apache Tomcat Web Server 7.0.1.
- ⊙ Oracle 10g Database as Back End.
- ⊙ J2EE Framework.

HTML draft version timeline

October 1991

HTML Tags an informal CERN document listing eighteen HTML tags, was first mentioned in public.

June 1992

First informal draft of the HTML DTD with seven subsequent revisions (July 15, August 6, August 18, November 17, November 19, November 20, November 22)

November 1992

HTML DTD 1.1 (the first with a version number, based on RCS revisions, which start with 1.1 rather than 1.0), an informal draft

June 1993

Hypertext Markup Language- was published by the IETF IIR Working Group as an Internet-Draft (a rough proposal for a standard). It was replaced by a second version- one month later, followed by six further drafts published by IETF itself- that finally led to HTML 2.0 in RFC1866

November 1993

HTML+ was published by the IETF as an Internet-Draft and was a competing proposal to the Hypertext Markup Language draft. It expired in May 1994.

April 1995 (authored March 1995)

HTML 3.0- was proposed as a standard to the IETF, but the proposal expired five months later without further action. It included many of the capabilities that were in Raggett's HTML+ proposal, such as support for tables, text flow around figures and the display of

January 2008

HTML5 was published as a Working Draft by the W3C.

Although its syntax closely resembles that of SGML, HTML5 has abandoned any attempt to be an SGML application and has explicitly defined its own "html" serialization, in addition to an alternative XML-based XHTML5 serialization.

Characteristics: -

- It is the language which can be easily understood and can be modified.
- Effective presentations can be made with the HTML with the help of its all formatting tags.



- It provides the more flexible way to design web pages along with the text.
- Links can also be added to the web pages so it helps the readers to browse the information of their interest.
- You can display HTML documents on any platforms such as Macintosh, Windows and Linux etc.
- Graphics, videos and sounds can also be added to the web pages which give an extra attractive look to your web pages. Allows database integration with wide variety of applications.
- Additional internet capabilities.

XHTML Versions: -

XHTML is a separate language that began as a reformulation of HTML 4.01 using XML 1.0. It continues to be developed:

- XHTML 1.0, published January 26, 2000, as a W3C Recommendation, later revised and republished August 1, 2002. It offers the same three variations as HTML 4.0 and 4.01, reformulated in XML, with minor restrictions.
- XHTML 1.1 published May 31, 2001, as a W3C Recommendation. It is based on XHTML 1.0 Strict, but includes minor changes, can be customized, is reformulated using modules from Modularization of XHTML, which was published April 10, 2001, as a W3C Recommendation.
- XHTML 2.0 There is no XHTML 2.0 standard. XHTML 2.0 is only a draft document and it is inappropriate to cite this document as other than work in progress. XHTML 2.0 is incompatible with XHTML 1.x and, therefore, would be more accurately characterized as an XHTML-inspired new language than an update to XHTML 1.x.
- XHTML5, which is an update to XHTML 1.x, is being defined alongside HTML5 in the HTML5 draft.

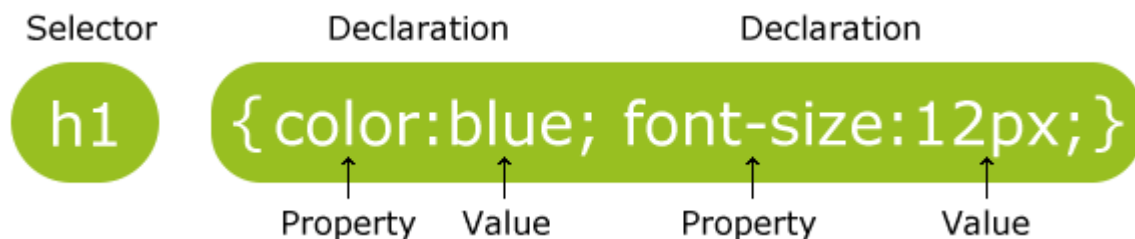
CASCADING STYLE SHEET



Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation semantics (the look and formatting) of a document written in a markup language. It's most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL.

CSS defines HOW HTML elements are to be displayed. Styles are normally saved in external .css files. External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single file! The CSS files referenced in the HTML page.

A CSS rule has two main parts: a selector, and one or more declarations:



The selector is normally the HTML element you want to style. Each declaration consists of a property and a value. The property is the style attribute you want to change. Each property has a value.

Following example formats a paragraph in an HTML document

```
p
{
    Color : red;
    text-align : center;
}
```

Python:

Why Python for Machine Learning

- **Readability and Simplicity:** Python's syntax is clean and easy to understand, making it accessible for beginners and allowing for rapid development.
- **Community Support:** A large and active community contributes to a wealth of resources, tutorials, and documentation.
- **Rich Ecosystem:** Python has a plethora of libraries and frameworks specifically designed for data manipulation, numerical computation, and machine learning.

Key Libraries and Frameworks

- **NumPy:** A fundamental library for numerical computing in Python. It provides support for arrays and matrices, along with a collection of mathematical functions to operate on them.
- **Pandas:** A library for data manipulation and analysis, providing data structures like Data Frames that make it easy to handle structured data.
- **Matplotlib and Seaborn:** Libraries for data visualization. Matplotlib provides a foundation for creating static plots, while Seaborn offers a higher-level interface for drawing attractive statistical graphics.
- **Scikit-learn:** A robust library for traditional machine learning. It offers simple and efficient tools for data mining and data analysis, including a variety of algorithms for classification, regression, clustering, and more.
- **TensorFlow:** An open-source framework developed by Google for building and deploying deep learning models. It provides a flexible architecture for designing complex neural networks.
- **Keras:** An API for building and training deep learning models. It runs on top of TensorFlow and simplifies the process of creating neural networks.
- **PyTorch:** A flexible and powerful deep learning framework developed by Facebook. It's particularly popular in research due to its dynamic computation graph and intuitive design.
- **XGBoost:** A library designed for speed and performance in gradient boosting, often used in machine learning competitions for its accuracy and efficiency.

Basic Steps in Machine Learning Using Python

The machine learning workflow typically follows these steps:

a. Data Collection

- Data can be collected from various sources, such as databases, APIs, or CSV files.

b. Data Preprocessing

- **Cleaning:** Handling missing values, removing duplicates, and correcting inconsistencies.
- **Transformation:** Normalizing or scaling data, encoding categorical variables, and feature engineering.

c. Exploratory Data Analysis (EDA)

- Using libraries like Pandas and Matplotlib to visualize data distributions, correlations, and trends to gain insights.

d. Model Selection

- Choosing the appropriate machine learning algorithm based on the problem (classification, regression, clustering, etc.).

e. Model Training

- Splitting the dataset into training and testing sets, then training the model using the training data.

f. Model Evaluation

- Assessing the model's performance using metrics like accuracy, precision, recall, F1 score, or mean squared error, depending on the task.

g. Hyperparameter Tuning

- Fine-tuning the model's parameters to improve performance, often using techniques like Grid Search or Random Search.

h. Deployment

- Integrating the model into an application or making it available via an API for use in production.

Challenges and Considerations

- **Data Quality:** The performance of ML models heavily relies on the quality of data. Poor-quality data can lead to inaccurate models.
- **Overfitting:** Models can become too complex and perform well on training data but poorly on unseen data. Regularization techniques and cross-validation can help mitigate this.
- **Interpretability:** Some models, especially complex ones like neural networks, can be hard to interpret. Techniques like SHAP or LIME can help explain model predictions.

Future of Python in Machine Learning

- The use of Python in machine learning is expected to grow as new libraries and frameworks continue to emerge, and as industries increasingly adopt AI technologies. Python's integration with big data tools, cloud computing, and its support for advanced algorithms will keep it at the forefront of the machine learning field.

Advanced Machine Learning Concepts

a. Feature Engineering

- **Definition:** The process of selecting, modifying, or creating new features to improve model performance.
- **Techniques:**
 - **Binning:** Converting continuous variables into categorical bins.
 - **Polynomial Features:** Creating new features by raising existing features to a power.
 - **Interaction Features:** Combining two or more features to capture interactions between them.

b. Model Evaluation Techniques

- **Cross-Validation:** A technique to assess how the results of a statistical analysis will generalize to an independent dataset. Common methods include:
 - **K-Fold Cross-Validation:** The dataset is divided into K subsets, and the model is trained K times, each time using a different subset as the test set.
 - **Stratified K-Fold:** Ensures that each fold is a good representative of the whole dataset, particularly for imbalanced classes.
- **Confusion Matrix:** A tool for visualizing the performance of a classification algorithm. It shows true positives, false positives, true negatives, and false negatives.

c. Regularization Techniques

- **L1 Regularization (Lasso):** Adds the absolute value of the magnitude of coefficients as a penalty term to the loss function, which can lead to sparse models.
- **L2 Regularization (Ridge):** Adds the squared magnitude of coefficients as a penalty term to the loss function, helping to reduce overfitting.

2. Deep Learning with Python

a. Neural Networks

- **Structure:** Composed of layers (input, hidden, and output), where each layer consists of interconnected nodes (neurons).

- **Activation Functions:** Functions that determine the output of a neural network node. Common ones include:
- **Re LU (Rectified Linear Unit):** Introduces non-linearity and helps with the vanishing gradient problem
- **Sigmoid:** Used for binary classification; outputs values between 0 and 1.
- **Soft max:** Used in multi-class classification to represent probabilities.

b. Training Neural Networks

- **Backpropagation:** A method used to update the weights of the network by minimizing the loss function through gradient descent.
- **Optimizers:** Algorithms to adjust the learning rate and update weights. Common optimizers include:
 - **SGD (Stochastic Gradient Descent):** Basic optimization technique.
 - **Adam:** Combines the advantages of Ada Grad and RMS Prop; widely used for its efficiency.

3. Real-World Applications of Machine Learning with Python

a. Natural Language Processing (NLP)

- **Libraries:** NLTK, Spa Cy, and Transformers (Hugging Face) for text processing and model building.
- **Applications:** Sentiment analysis, chatbots, language translation, and text summarization.

b. Computer Vision

- **Libraries:** OpenCV for image processing, and TensorFlow/PyTorch for building CNNs (Convolutional Neural Networks).
- **Applications:** Object detection, facial recognition, image classification, and autonomous vehicles.

c. Recommender Systems

- **Techniques:** Collaborative filtering, content-based filtering, and hybrid methods.
- **Use Cases:** E-commerce (Amazon), streaming services (Netflix), and social media platforms (YouTube).

4. Emerging Trends in Python and Machine Learning

a. Auto ML (Automated Machine Learning)

- Tools like H2O.ai, Google Cloud Auto ML, and TPOT automate the process of model selection, hyperparameter tuning, and feature engineering, making machine learning more accessible.

b. Explainable AI (XAI)

- The need for transparency in AI models is growing. Techniques such as LIME and SHAP are being used to interpret complex models and build trust with end-users.

c. Federated Learning

- A decentralized approach where models are trained across multiple devices or servers holding local data samples without exchanging them, enhancing privacy.

5. Best Practices for Using Python in Machine Learning

- **Version Control:** Use tools like Git to keep track of code changes and collaborate with others.
- **Environment Management:** Use virtual environments (via venv or conda) to manage dependencies and avoid conflicts between projects.
- **Documentation:** Document your code and workflows to ensure reproducibility and make it easier for others (and yourself) to understand your work in the future.
- **Code Quality:** Utilize linters and formatters (like black and flake8) to maintain code quality and readability.

Conclusion

Python continues to be a dominant language in the machine learning landscape due to its versatility, ease of use, and the powerful libraries available. As the field evolves, staying updated with the latest tools, techniques, and best practices is essential for anyone interested in machine learning.

APACHE TOMCAT:



Apache Tomcat (called "Tomcat" for short) is a [free and open-source](#) implementation of the [Java Servlet](#), [Java Server Pages](#), [Java Expression Language](#) and [WebSocket](#) technologies.^[3] Tomcat provides a "pure Java" [HTTP web server](#) environment in which [Java](#) code can run.

Tomcat is developed and maintained by an open community of developers under the auspices of the [Apache Software Foundation](#), released under the [Apache License](#) 2.0 license.

Components

Tomcat 4.x was released with Catalina (a servlet container), Coyote (an HTTP connector) and Jasper (a [JSP engine](#)).

Catalina James

Catalina is Tomcat's [servlet container](#). Catalina implements [Sun Microsystems'](#) specifications for [servlet](#) and Java Server Pages (JSP). In Tomcat, a Realm element represents a "database" of usernames, passwords, and roles (similar to [Unix](#) groups) assigned to those users. Different implementations of Realm allow Catalina to be integrated into environments where such authentication information is already being created and maintained, and then use that information to implement Container Managed Security as described in the Servlet Specification.^[4]

Coyote

Coyote is a Connector component for Tomcat that supports the HTTP 1.1 and 2 protocol as a web server. This allows Catalina, nominally a Java Servlet or JSP container, to also act as a plain web server that serves local files as HTTP documents.^[5] Coyote listens for incoming connections to the server on a specific [TCP](#) port and forwards the request to the Tomcat Engine to process the request and send back a response to the requesting client. Another Coyote Connector, Coyote JK, listens similarly but instead forwards its requests to another web server, such as Apache, using the [JK Protocol](#).^[6] This usually offers better performance.^[citation needed]

Jasper

Jasper is Tomcat's JSP Engine. Jasper [parses JSP](#) files to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.

As of version 5, Tomcat uses Jasper 2, which is an implementation of the Sun Microsystems' JSP 2.0 specification. From Jasper to Jasper 2, important features were added:

- JSP Tag library pooling – Each tag markup in JSP file is handled by a tag handler class. Tag handler class objects can be pooled and reused in the whole JSP servlet.
- Background JSP compilation – While recompiling modified JSP Java code, the older version is still available for server requests. The older JSP servlet is deleted once the new JSP servlet has finished being recompiled.
- Recompile JSP when included page changes – pages can be inserted and included into a JSP at runtime. The JSP will not only be recompiled with JSP file changes but also with included page changes.
- JDT Java compiler – Jasper 2 can use the Eclipse JDT (Java Development Tools) Java compiler instead of [Ant](#) and [java c](#).

Three new components were added with the release of Tomcat 7:

Cluster

This component has been added to manage large applications. It is used for [load balancing](#) that can be achieved through many techniques. Clustering support currently requires the JDK version 1.5 or higher.

High availability

A high-availability feature has been added to facilitate the scheduling of system upgrades (e.g. new releases, change requests) without affecting the live environment. This is done by dispatching live traffic requests to a temporary server on a different port while the main server is upgraded on the main port. It is very useful in handling user requests on high-traffic web applications.^[7]

Web application

It has also added user- as well as system-based web applications enhancement to add support for deployment across the variety of environments. It also tries to manage sessions as well as applications across the network.

Tomcat is building additional components. A number of additional components may be used with Apache Tomcat. These components may be built by users should they need them or they can be downloaded from one of the mirrors.^[8]

Features

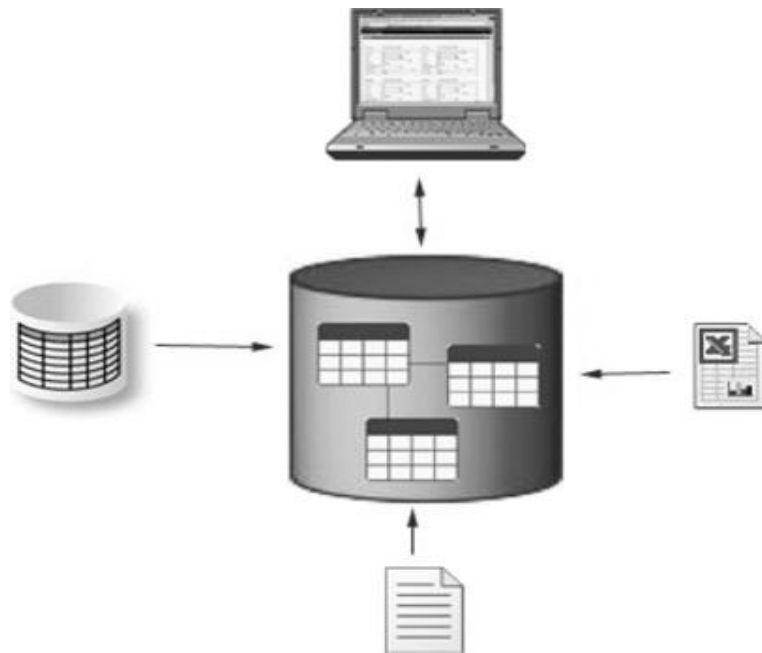
Tomcat 8.x implements the Servlet 3.1 and JSP 2.3 Specifications.^[9] Apache Tomcat 8.5.x is intended to replace 8.0.x and includes new features pulled forward from Tomcat 9.0.x. The minimum Java version and implemented specification versions remain unchanged.^[10]

The Key Features of NetBeans 8.2

- ECMAScript 6 support
- NodeJS enhancements
- Oracle JET support enhancements
- PHP7 support
- Docker support
- New editor multi caret features
- New pin able watches feature
- SQL profiling

C/C++ enhancements

DATABASE MANAGEMENT SYSTEM



A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. This is a software system that allows access to the data contained in the database. The primary goal of a DBMS is to provide an environment that is both convenient and efficient to use in storing and retrieving database information.

Functions of DBMS:-

DBMS is a general purpose software system that performs the following functions:-

- Defining a database.
- Constructing the database.
- Manipulating the database.
- Sharing database among various users.
- Protecting the database.
- Maintaining a database.

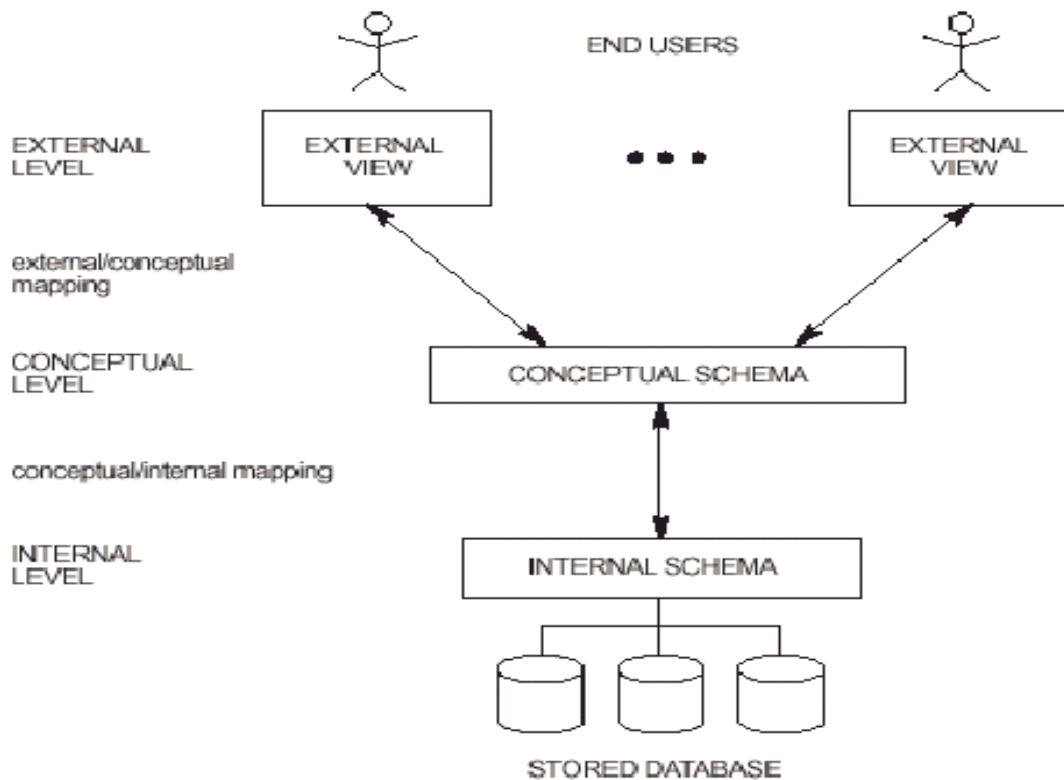
DBMS Architecture:-

DBMS architecture consists of three levels known as Three Schema Architecture. It is convenient tool with which the user can visualize the schema levels in a database system. It contains of the following three schemas:-

1. ***The Internal level:*** - This contains of an internal schema, which describes the physical storage structure of the database. It is the lowest level of abstraction. It does not hide

the storage details. It contains the definition of the stored record, the method of representing the data fields and the access aids used. This internal schema uses a physical data model and describes the complete details of data storage and access paths for the database. It is also called the physical schema.

2. ***The Conceptual level:*** - This has a conceptual schema, which describes the structures of a database for a group of users. This schema hides the storage details from the user and it includes description of entities, data types, relationships, user operations and constraints. The description of data at this level is in a format independent of its physical representation.
3. ***The External level:*** - This has a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides all the other details from this group. This is at a highest level of database absorption.



Three schema DBMS architecture

Client-Server Architecture: -

The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers and other equipment are connected via a network. There are two client/server architecture:-

- Two-tier
- Three-tier
- N-tier

Two-Tier Client/Server Architecture: -

In two tier architecture, the software components are distributed over two systems: the client and the server. This architecture has two forms as: logical two-tier and physical two-tier.

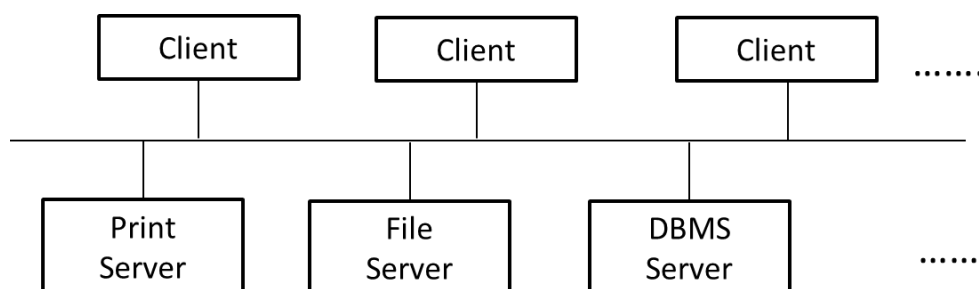


Fig: - Logical two-tier client/server architecture

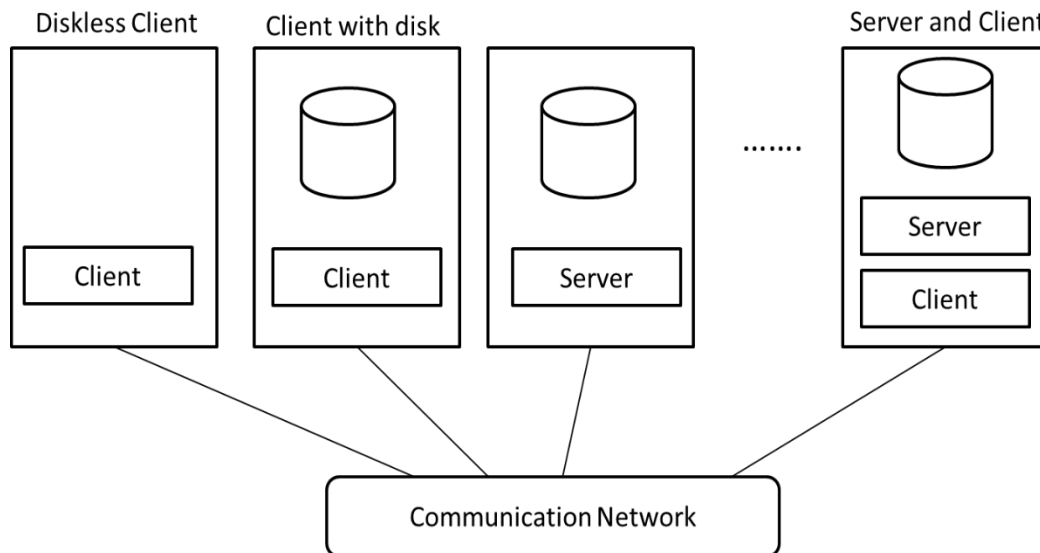


Fig: - Physical two-tier client/server architecture

The above figure shows the physical client/server architecture. Some machines like diskless workstations, or workstations/PCs with disks that have only client software installed would be only client sites. Other machines would be dedicated servers. Some other machines would have both client and server functionality. In relational DBMS, user interface and application programs can run at client side. The query and transactions functionality are included on the server side. A client program can send query and transaction requests using the ODBC API which are then processed at the server site. The query results are sending back to the client program which can process or display the results needed.

Three-Tier Client/Server Architecture: -

With the emergence of World Wide Web, many web applications use the three-tier architecture. There is an intermediate layer between the client and the database server called the application server or the web server. The web server plays the intermediary role by storing business rules that are used to access data from the database server. It checks the client's credentials before forwarding a request to the database server. The intermediate server accepts requests from the client, process the request and sends database commands to the database server, and then acts as a conduit for passing processed data from the database server to the clients, where it may be processed further.



Fig: - Logical three-tier client/server architecture

N-Tier Architecture: -

In N-Tier architecture, the middle tier is allowed to have multiple application objects rather than a single application. Each of these application objects must have a well-defined interface which allows them to contact and communication with one another. An interface actually brings an idea of contract. That is, each object states through its interface that it will accept certain parameters and return a specific set of results.

Application objects use their interfaces to do business processing. With and N-Tier architecture, one can have multiple applications using a common set of business objects across an organization. This promotes the standardization of business practices by creating a single set of business functions for the entire organization to access. If a particular business rule changes, then changes have to be made to only the business object and if need, to its interface also.

Normalization: -

Normalization is a process during which unsatisfactory relation schemas are decomposed by breaking up their attributes into smaller relation schemas that possess desirable properties.

Normalization of data can be looked upon as a process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the desirable properties of minimizing redundancy, insertion, deletion and update anomalies.

The different Normal Forms present in DBMS are: -

- ❖ **First Normal Form: -** It states that the domains of attributes must include only atomic values and that the value of any attribute in a tuple must be single value from the domain of that attribute. It disallows a set of values, a tuple of values, or a combination of both as an attribute value for a single tuple.
- ❖ **Second Normal Form: -** A relation is said to be in Second Normal Form if it is in 1NF and non-key attributes are fully functionally dependent on the key attribute(s). If the key has more than one attribute (composite key) then no non-key attributes should be functionally dependent upon a part of the key attributes.



- ❖ **Third Normal Form:** - A relation is said to be in Third Normal Form if it is in 2NF and no non-prime attributes of relation R is transitively dependent on the primary key.
- ❖ **Fourth Normal Form:** - Under this, a record type should not contain two or more independent multi-valued facts about an entity. In addition, the record must satisfy third normal form.
- ❖ **Fifth Normal Form:** - It also deals with multi-valued facts. Here, the record must satisfy the fourth normal form.
- ❖ **Boyce Codd Normal Form:** - It is a [normal form](#) used in [database normalization](#). It is a slightly stronger version of the [third normal form](#) (3NF). A table is in Boyce–Codd normal form [if and only if](#) for every one of its [nontrivial dependencies](#) $X \twoheadrightarrow Y$, X is a [super key](#)—that is, X is either a [candidate key](#) or a superset thereof



System Design

6. SYSTEM DESIGN

Project Modules

Entire project is divided into 3 modules as follows:

Data Gathering and pre processing

Training the model using following Machine Learning algorithms

- i. SVM
- ii. Random Forest Classifier
- iii. Decision Tree

Module 1: Data Gathering and Data Pre processing

- a. A proper dataset is searched among various available ones and finalized with the dataset.
- b. The dataset must be preprocessed to train the model.
- c. In the preprocessing phase, the dataset is cleaned and any redundant values, noisy data and null values are removed.
- d. The Preprocessed data is provided as input to the module.

Module 2: Training the model

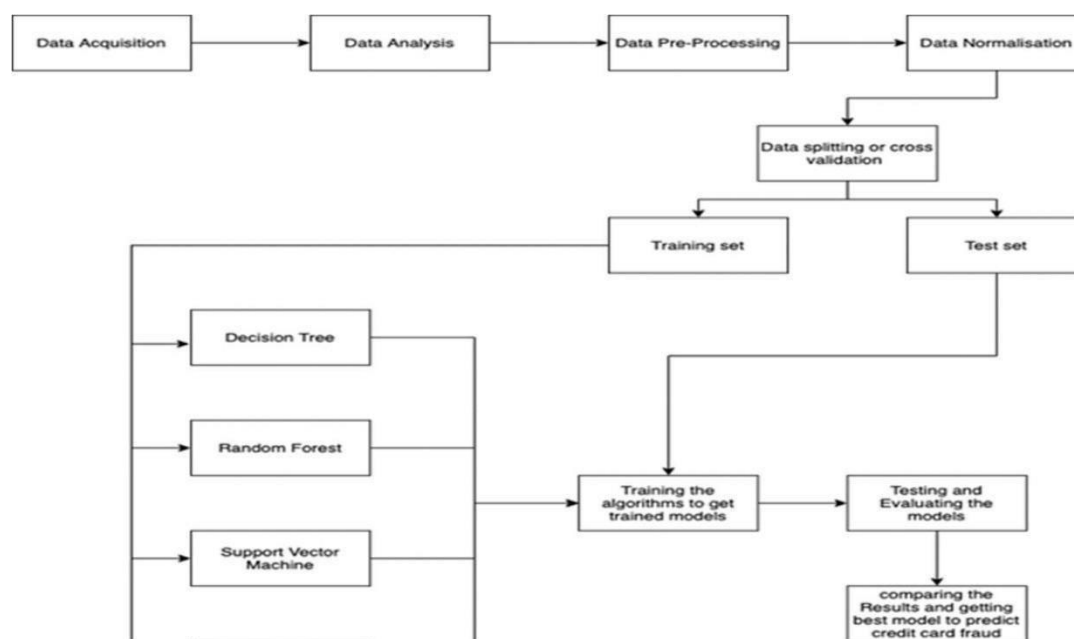
- a. The Preprocessed data is split into training and testing datasets in the 80:20 ratio to avoid the problems of over-fitting and under-fitting.
- b. A model is trained using the training dataset with the following algorithms
SVM, Random Forest Classifier and Decision Tree
- c. The trained models are trained with the testing data and results are visualized using bar graphs, scatter plots.
- d. The accuracy rates of each algorithm are calculated using different params like F1 score, Precision, Recall. The results are then displayed using various data visualization tools for analysis purpose.
- e. The algorithm which has provided the better accuracy rate compared to remaining algorithms is taken as final prediction model.

Module 3: Final Prediction model integrated with front end

- a. The algorithm which has provided better accuracy rate has considered as the final prediction model.
- b. The model thus made is integrated with front end.
- c. Database is connected to the front end to store the user information who are using it.

6.1 SYSTEM ARCHITECTURE

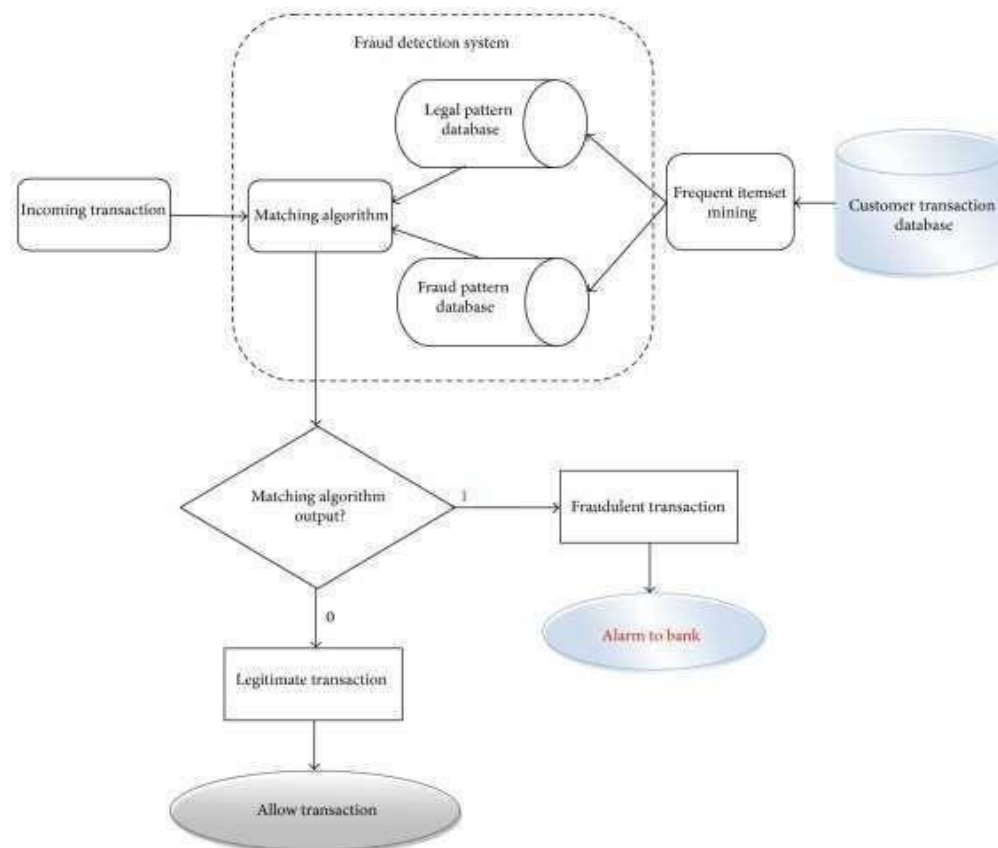
Our Project main purpose is to making Credit Card Fraud Detection aware to people from credit card online frauds. the main point of credit card fraud detection system is necessary to safe our transactions & security. With this system, fraudsters don't have the chance to make multiple transactions on a stolen or counterfeit card before the cardholder is aware of the fraudulent activity. 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.



System Architecture

Activity diagram

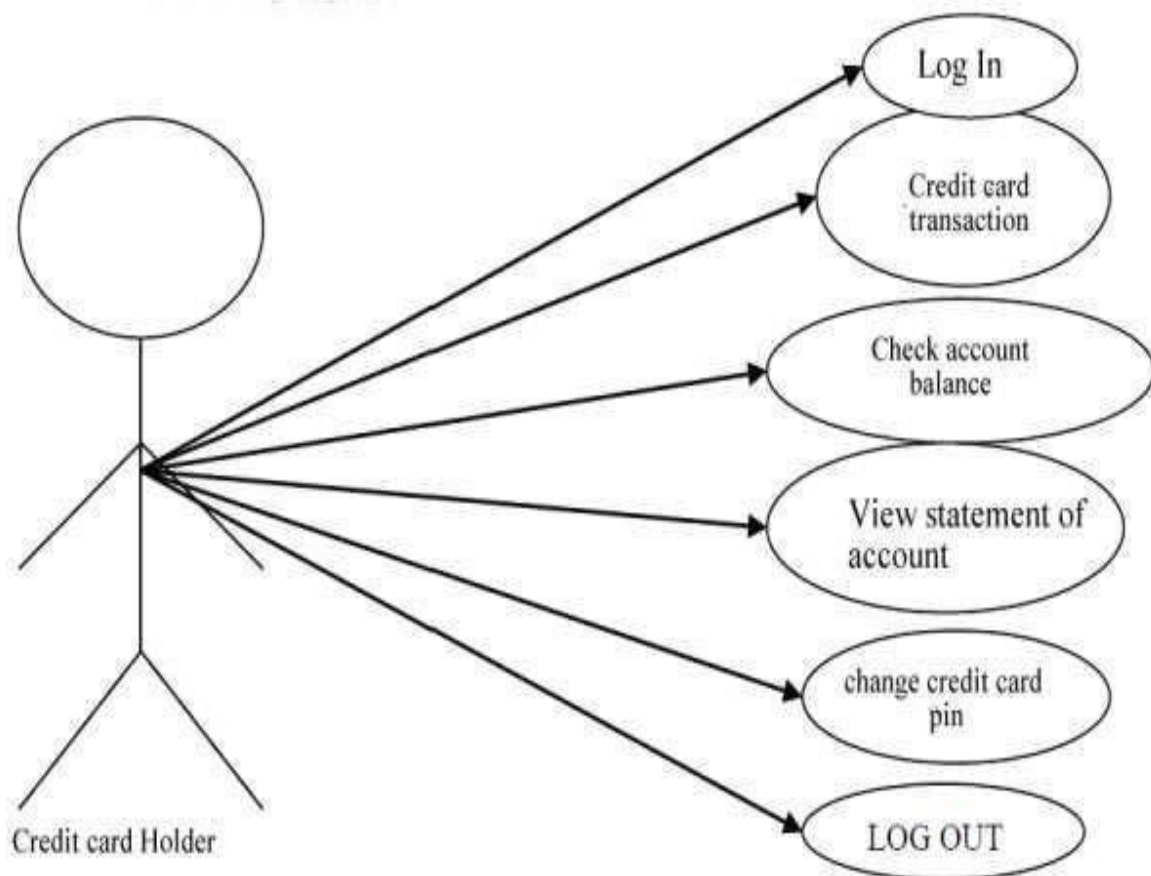
Activity diagram is an important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. The basic purposes of activity diagram are it captures the dynamic behavior of the system. Activity diagram is used to show message flow from one activity to another Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.



Activity Diagram

Use case diagram

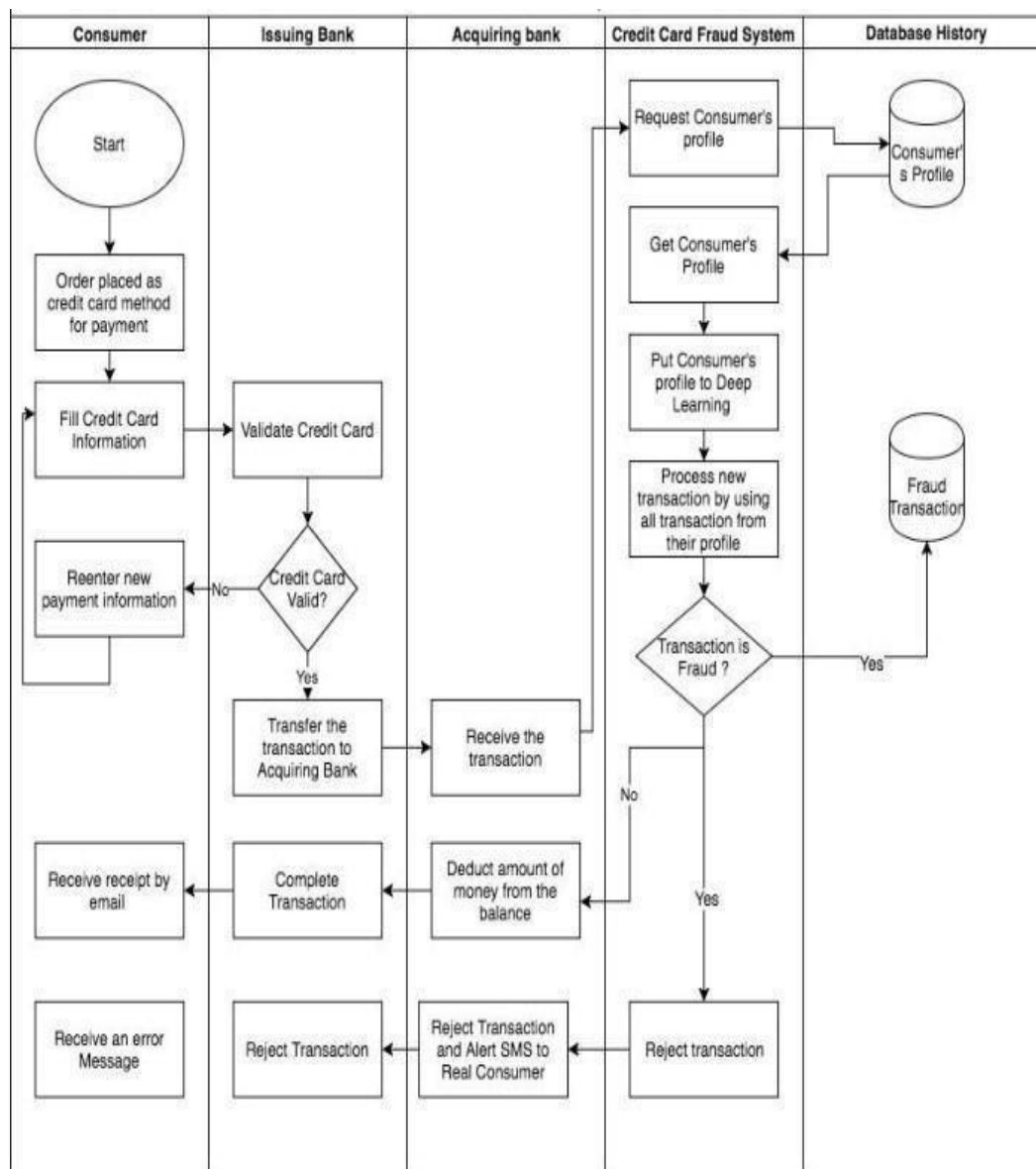
In UML, use-case diagrams model the behavior of a system and help to capture the requirements of the system. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally. Use-case diagrams illustrate and define the context and requirements of either an entire system or the important parts of the system. You can model a complex system with a single use-case diagram, or create many use-case diagrams to model the components of the system. You would typically develop use-case diagrams in the early phases of a project and refer to them throughout the development process.



Use case Diagram

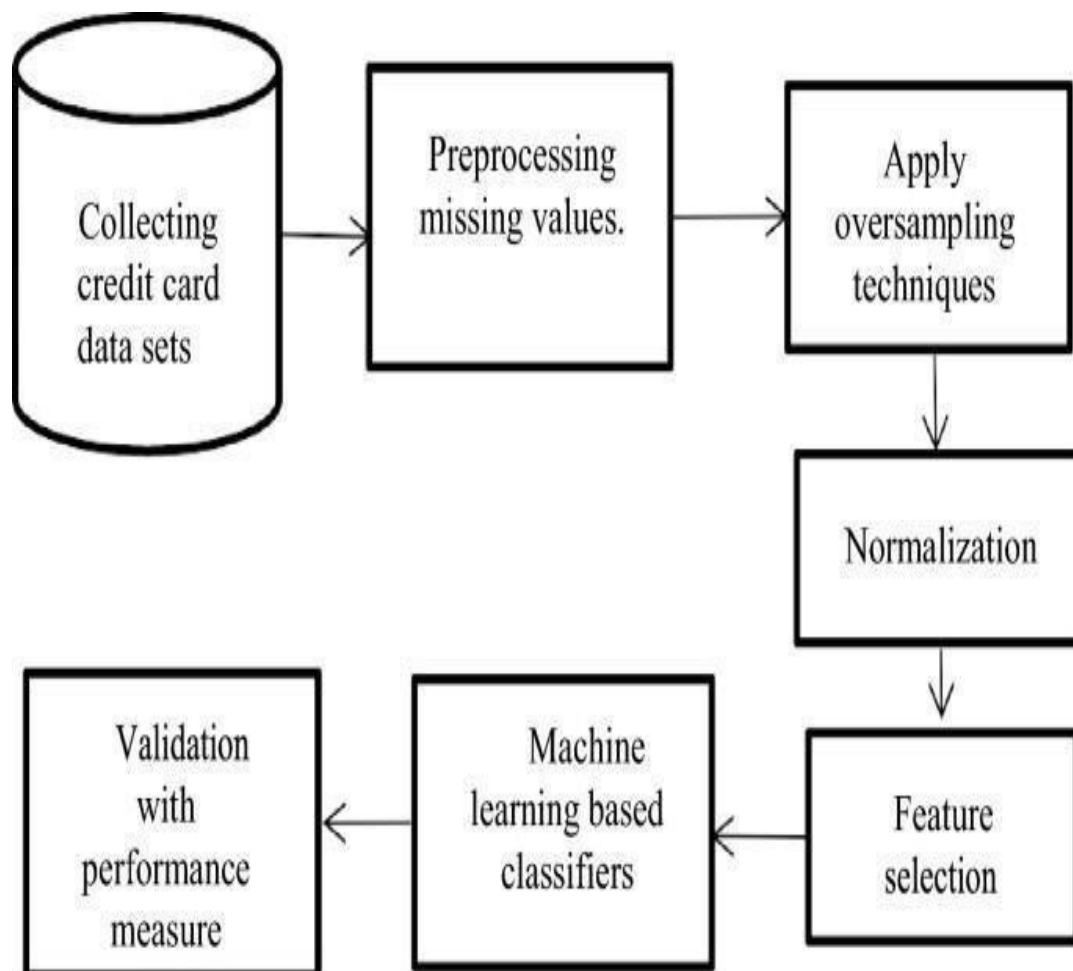
Sequence Diagram

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.



Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored. The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.



Data Flow diagram



System Testing

7. TESTING

Testing is a process of executing a program with intent of finding an error. Testing presents an interesting anomaly for the software engineering. The goal of the software testing is to convince system developer and customers that the software is good enough for operational use. Testing is a process intended to build confidence in the software. Testing is a set of activities that can be planned in advance and conducted systematically. Software testing is often referred to as verification & validation.

Unit Testing

In this testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In this testing step each module is found to working satisfactorily as regard to the expected output from the module. There are some validation checks for fields also. It is very easy to find error debut in the system.

Validation Testing

At the culmination of the black box testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests. Asking the user about the format required by system tests the output displayed or generated by the system under consideration. Here the output format is considered the of screen display. The output format on the screen is found to be correct as the format was designed in the system phase according to the user need. For the hard copy also, the output comes out as specified by the user. Hence the output testing does not result in any correction in the system.

Functional Testing

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. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

Integration Testing

Data can be lost across an interface; one module can have an adverse effect on the other sub functions when combined may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The Developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

User 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. Some of my friends who tested this module suggested that this was really a user-friendly application and giving good processing speed.

7.1 SAMPLE TEST CASES DONE:

Test Case: Successful Transaction

- **Objective:** Verify that a legitimate transaction is approved.
- **Preconditions:** A valid customer account with an active credit card.
- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$50.00
 - Location: "New York, NY"
- **Expected Result:** Transaction status should be "Approved".

2. Test Case: Transaction with Insufficient Funds

- **Objective:** Verify that a transaction is declined due to insufficient funds.
- **Preconditions:** A valid customer account with a credit limit of \$30.
- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$50.00
- **Expected Result:** Transaction status should be "Declined".

3. Test Case: Suspicious Transaction Based on Location

- **Objective:** Verify that a transaction from an unusual location is flagged as suspicious.
- **Preconditions:** Customer regularly transacts in New York.
- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$100.00
 - Location: "Tokyo, Japan"
- **Expected Result:** Transaction status should be "Suspicious", and a fraud alert should be generated.

4. Test Case: Multiple Transactions in Quick Succession

- **Objective:** Verify that multiple transactions in a short time frame are flagged as potentially fraudulent.
- **Preconditions:** Customer usually makes one transaction per day.
- **Input:**
 - First Transaction: Amount: \$20.00, Time: 10:00 AM
 - Second Transaction: Amount: \$25.00, Time: 10:05 AM
- **Expected Result:** The second transaction should be flagged as "Suspicious", and a fraud alert should be generated.

5. Test Case: High-Value Transaction

- **Objective:** Verify that a high-value transaction is flagged for review.
- **Preconditions:** Customer usually transacts amounts below \$200.

- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$2,000.00
- **Expected Result:** Transaction status should be "Suspicious", and a fraud alert should be generated.

6. Test Case: Blocked Card Usage

- **Objective:** Verify that a blocked card cannot be used for transactions.
- **Preconditions:** Customer's card has been reported lost or stolen.
- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$30.00
- **Expected Result:** Transaction status should be "Declined" with a message indicating the card is blocked.

7. Test Case: Customer Response to Alert

- **Objective:** Verify that a customer can respond to a fraud alert.
- **Preconditions:** A fraud alert has been generated for a suspicious transaction.
- **Input:**
 - Alert ID: 1
 - Response: "This transaction was not authorized."
- **Expected Result:** The alert response should be recorded, and the fraud investigation should be initiated.

8. Test Case: Fraud Detection Algorithm Performance

- **Objective:** Verify that the fraud detection algorithm correctly identifies known fraudulent patterns.
- **Preconditions:** Database contains historical transaction data with known fraud cases.
- **Input:**
 - Test data containing patterns similar to historical fraud cases.
- **Expected Result:** The system should correctly flag known fraudulent transactions as "Fraudulent".

9. Test Case: Transaction Approval after Alert Response

- **Objective:** Verify that legitimate transactions are approved after the customer confirms fraud alerts.
- **Preconditions:** Customer has confirmed that suspicious transactions are valid.
- **Input:**
 - Card Number: "1234-5678-9012-3456"
 - Amount: \$75.00
- **Expected Result:** Transaction status should be "Approved".

PERFORMANCE ANALYSIS

Performance metrics:

The basic performance measures derived from the confusion matrix. The confusion matrix is a 2 by 2 matrix table contains four outcomes produced by the binary classifier. Various measures such as sensitivity, specificity, accuracy and error rate are derived from the confusion matrix.

Accuracy: Accuracy is calculated as the total number of two correct predictions(A+B) divided by the total number of the dataset(C+D). It is calculated as (1-error rate).

$$\text{Accuracy} = \frac{A+B}{C+D}$$

Whereas,

$$A = \text{True Positive} \quad B = \text{True Negative}$$

$$C = \text{Positive} \quad D = \text{Negative}$$

Error rate:

Error rate is calculated as the total number of two incorrect predictions(F+E) divided by the total number of the dataset(C+D).

$$\text{Error rate} = \frac{F+E}{C+D}$$

Whereas,

$$E = \text{False Positive}$$

$$F = \text{False Negative}$$

$$C = \text{Positive}$$

$$D = \text{Negative}$$

Sensitivity:

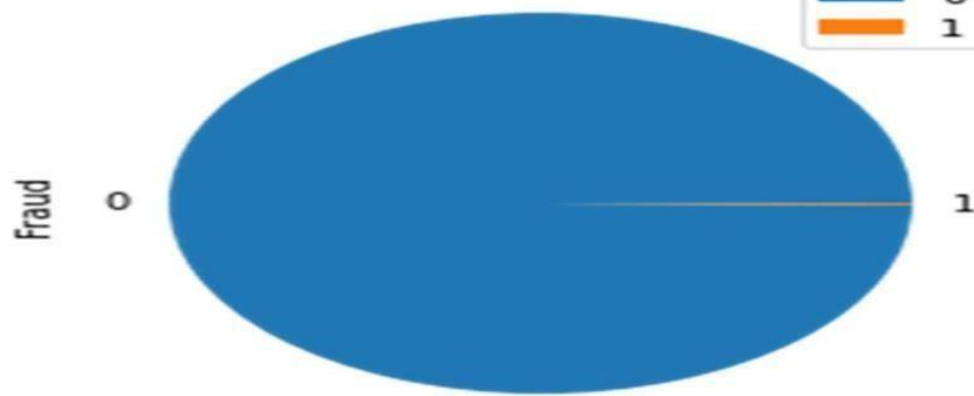
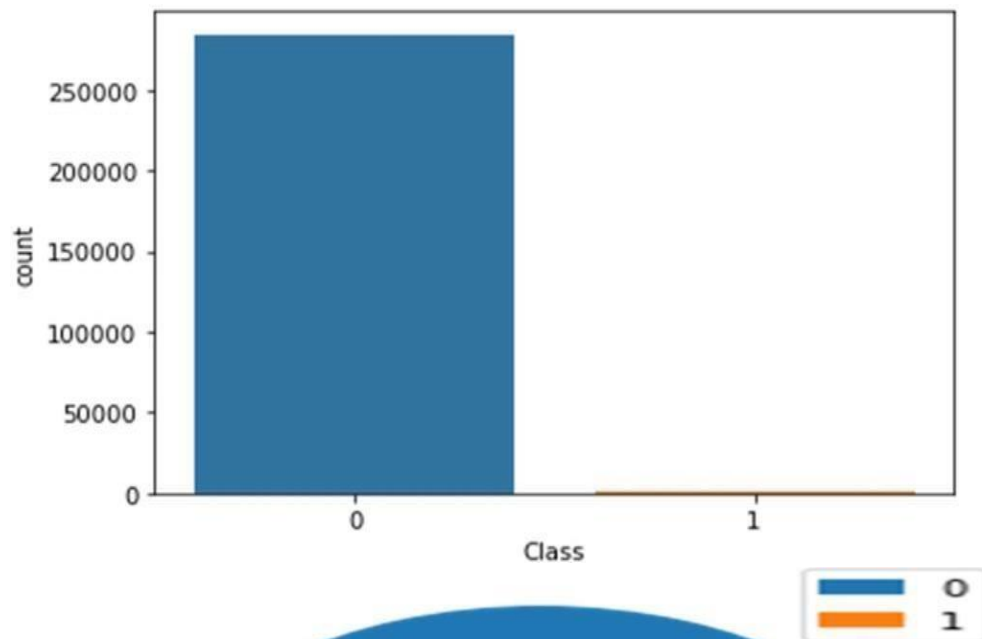
Sensitivity is calculated as the number of correct positive predictions(A) divided by the total number of positives(C).

$$\text{Sensitivity} = \frac{A}{C}$$

Specificity: Specificity is calculated as the number of correct negative predictions(B) divided by the total number of negatives(D).

$$\text{Specificity} = \frac{B}{D}$$

DATA ANALYSIS

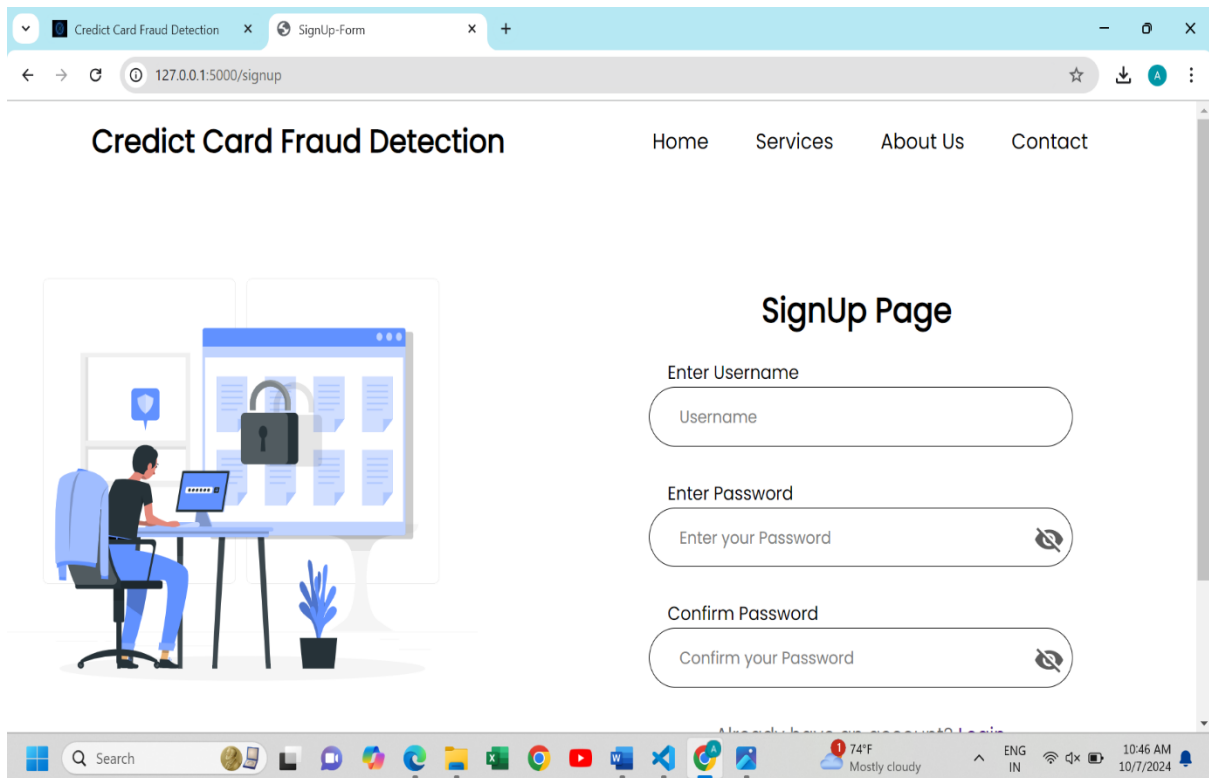


Dataset analysis



Screenshots

8. SCREENS



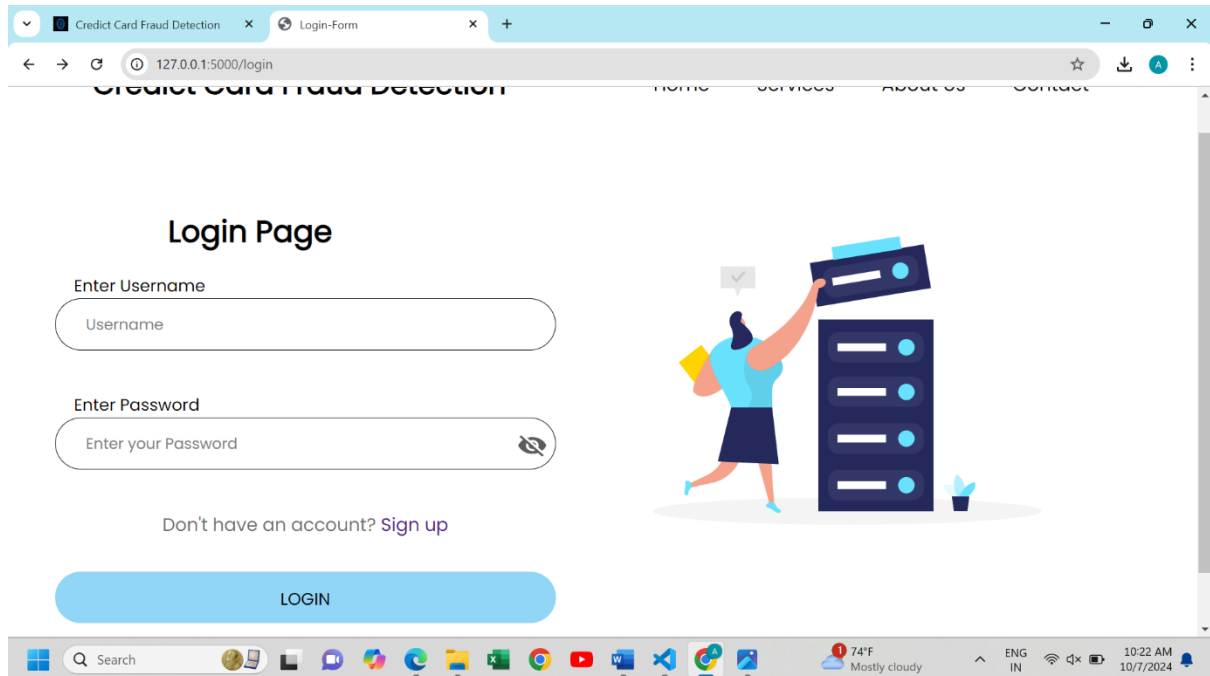
The screenshot displays a web browser window with two tabs: "Credit Card Fraud Detection" and "SignUp-Form". The address bar shows the URL "127.0.0.1:5000/signup". The website header includes the title "Credit Card Fraud Detection" and navigation links for "Home", "Services", "About Us", and "Contact".

The main content area is titled "SignUp Page" and contains the following form fields:

- Enter Username:** A text input field with the placeholder text "Username".
- Enter Password:** A password input field with the placeholder text "Enter your Password" and a toggle icon for visibility.
- Confirm Password:** A password input field with the placeholder text "Confirm your Password" and a toggle icon for visibility.

Below the form fields, there is a link that reads "Already have an account? Login".

The browser's taskbar at the bottom shows various application icons, a search bar, and system information including the temperature (74°F), weather (Mostly cloudy), language (ENG IN), and the date/time (10:46 AM, 10/7/2024).



The screenshot shows a web browser window with two tabs: 'Credit Card Fraud Detection' and 'Login-Form'. The address bar displays '127.0.0.1:5000/login'. The page title is 'Credit Card Fraud Detection'. The main content area features a 'Login Page' with the following elements:

- Enter Username:** A text input field with the placeholder text 'Username'.
- Enter Password:** A text input field with the placeholder text 'Enter your Password' and a toggle icon for password visibility.
- Don't have an account? [Sign up](#)**
- LOGIN** button

To the right of the login form is an illustration of a person in a blue shirt and dark skirt holding a yellow folder, standing next to a server rack and placing a document on top. A small potted plant is at the base of the server rack.

The Windows taskbar at the bottom shows the Start button, a search bar, and various application icons. The system tray on the right displays the weather (74°F, Mostly cloudy), language (ENG, IN), and the date/time (10:22 AM, 10/7/2024).

Credit Card Fraud Detection ML API 127.0.0.1:5000/login Logout

Welcome akshatanayak@987

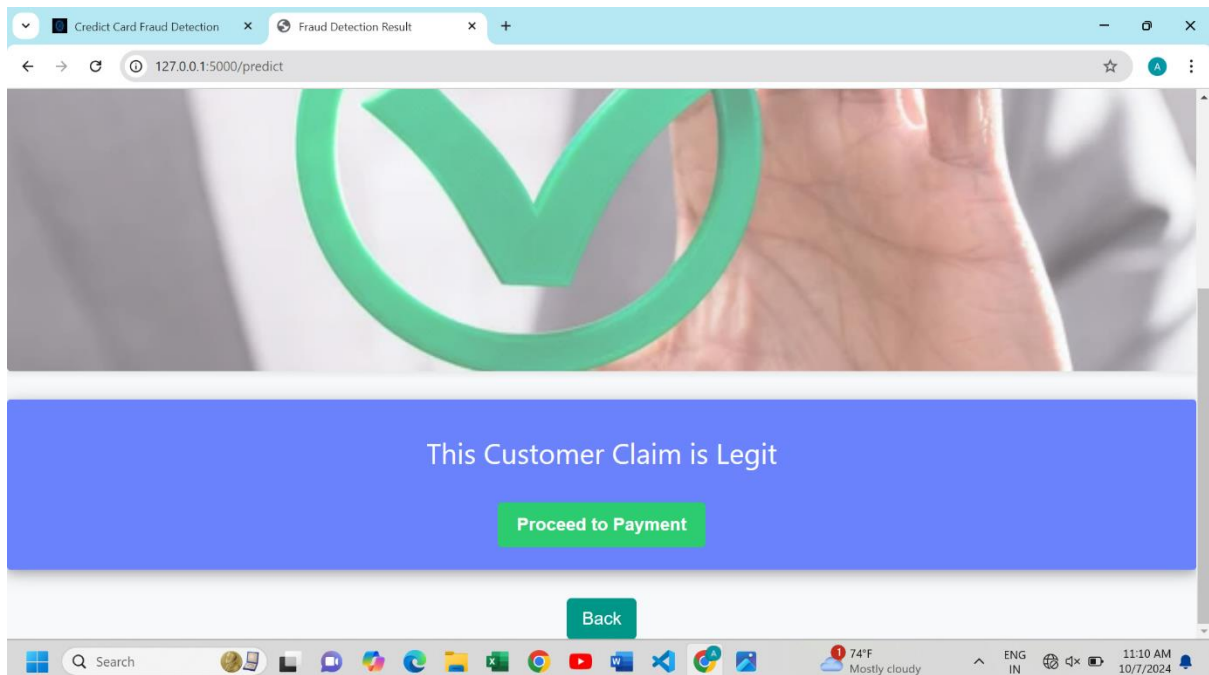
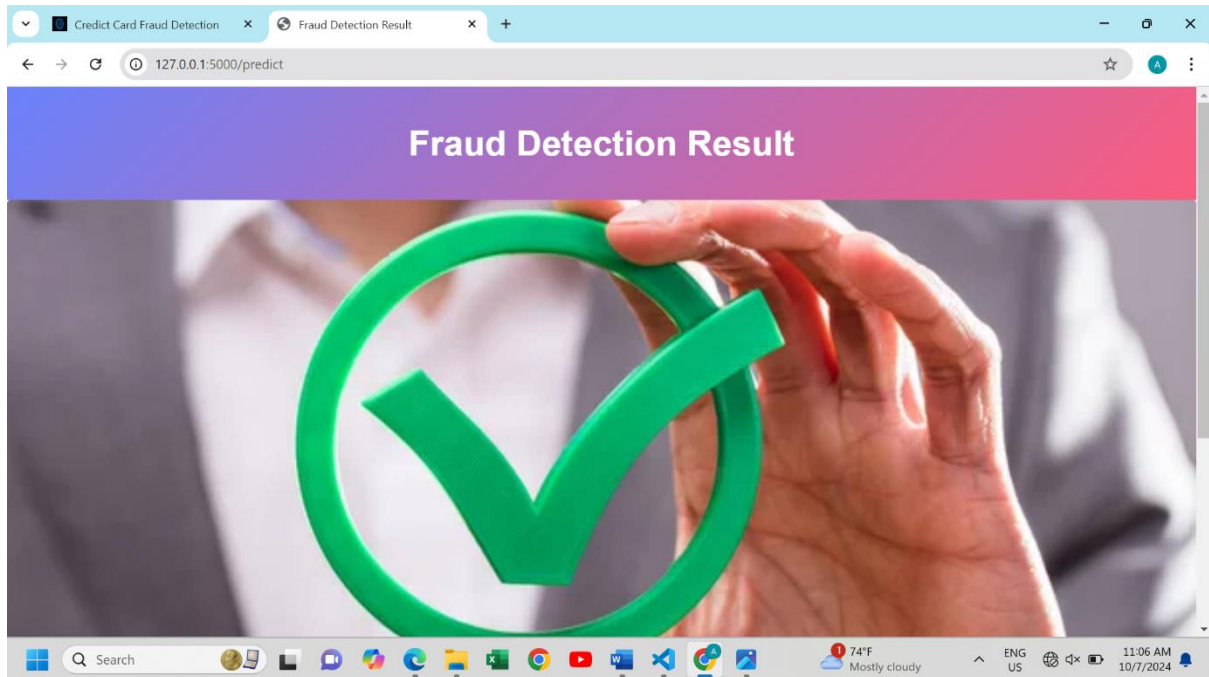
Credit Card Fraud Detection

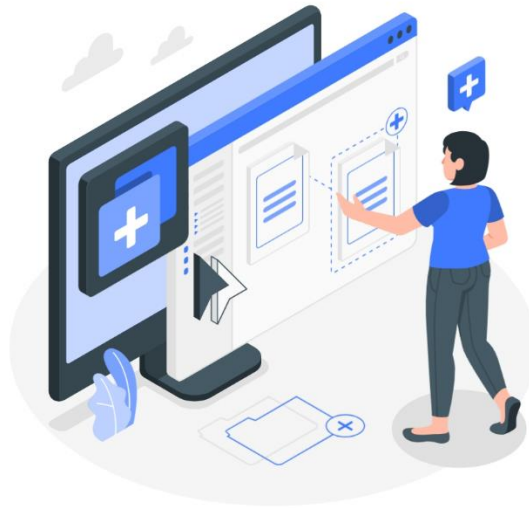
Time	V1	V2	V3
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
V4	V5	V6	V7
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
V8	V9	V10	V11
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Credit Card Fraud Detection ML API 127.0.0.1:5000/login

V16	V17	V18	V19
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
V20	V21	V22	V23
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
V24	V25	V26	V27
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
V28	Amount		
<input type="text"/>	<input type="text"/>		

Predict







Coding

PYTHON

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

model = pickle.load(open('creditp.pkl', 'rb'))
import imutils
import sklearn
from flask import Flask, render_template, request, redirect, url_for, session,
send_from_directory
from flask_sqlalchemy import SQLAlchemy
import re
from flask import Flask, request, render_template
import numpy as np
import pandas as pd
from sklearn import metrics
import warnings
import pickle
from sklearn.metrics import confusion_matrix
warnings.filterwarnings('ignore')

from sqlalchemy import func
from sqlalchemy import or_

# Configuring Flask
UPLOAD_FOLDER = 'static/uploads'
ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])

app = Flask(__name__)
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.secret_key = "secret key"
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///users.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db = SQLAlchemy(app)

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    username = db.Column(db.String(50), unique=True, nullable=False)
```



```
password = db.Column(db.String(255), nullable=False)

@app.before_request
def create_tables():
    db.create_all()

def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1] in ALLOWED_EXTENSIONS

@app.route('/')
def index():
    return render_template('home.html')

@app.route('/form')
def home():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/contact')
def contact():
    return render_template('contact.html')

@app.route('/services')
def services():
    return render_template('services.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    error=""
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        session['logged_in'] = True
        session['username'] = username
        global user
        user = User.query.filter_by(username=username).first()
        if not user or user.password != password:
            error = 'Invalid username or password.'
            return render_template('login.html', error=error)
        #return redirect(url_for('upload'))
        return render_template('index.html')
    return render_template('login.html')
```

```
@app.route('/signup', methods=['GET', 'POST'])
def signup():
    msg = ""
    c=0
    if request.method == 'POST' and 'username' in request.form and 'password' in request.form
    and 'cpassword' in request.form:
        username = request.form['username']
        password = request.form['password']
        confirm_password = request.form['cpassword']
        account = User.query.filter_by(username=username).first()
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'Username must not contain any special characters!'
        elif not username or not password or not confirm_password:
            msg = 'Please fill out the form !'
        elif password != confirm_password:
            msg = 'Passwords do not match.'
        else:
            new_user = User(username=username, password=password)
            db.session.add(new_user)
            db.session.commit()
            c=1
            msg = 'You have successfully registered!'
            #return render_template('signup.html', msg=msg)
        elif request.method == 'POST':
            msg = 'Please fill out the form !'
            return redirect(url_for('signup'))
        if c==1:
            return render_template('login.html')
        return render_template('signup.html',error=msg)
```

```
@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('username', None)
    return render_template("service.html")
```

```
@app.route('/process')
def process_file():

    # Render the template with the entire DataFrame
    return redirect('https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud')
```

```
@app.route('/predict',methods=['POST'])  
def predict():
```

```
    time=request.form['time']  
    time=float(time)
```

```
    v1=request.form['v1']  
    v1=float(v1)
```

```
    v2=request.form['v2']  
    v2=float(v2)
```

```
    v3=request.form['v3']  
    v3=float(v3)
```

```
    v4=request.form['v4']  
    v4=float(v4)
```

```
    v5=request.form['v5']  
    v5=float(v5)
```

```
    v6=request.form['v6']  
    v6=float(v6)
```

```
    v7=request.form['v7']  
    v7=float(v7)
```

```
    v8=request.form['v8']  
    v8=float(v8)
```

```
    v9=request.form['v9']  
    v9=float(v9)
```

```
    v10=request.form['v10']  
    v10=float(v10)
```

```
    v11=request.form['v11']  
    v11=float(v11)
```

```
    v12=request.form['v12']  
    v12=float(v12)
```

```
    v13=request.form['v13']  
    v13=float(v13)
```

```
    v14=request.form['v14']  
    v14=float(v14)
```

```
v15=request.form['v15']  
v15=float(v15)
```

```
v16=request.form['v16']  
v16=float(v16)
```

```
v17=request.form['v17']  
v17=float(v17)
```

```
v18=request.form['v18']  
v18=float(v18)
```

```
v19=request.form['v19']  
v19=float(v19)
```

```
v20=request.form['v20']  
v20=float(v20)
```

```
v21=request.form['v21']  
v21=float(v21)
```

```
v22=request.form['v22']  
v22=float(v22)
```

```
v23=request.form['v23']  
v23=float(v23)
```

```
v24=request.form['v24']  
v24=float(v24)
```

```
v25=request.form['v25']  
v25=float(v25)
```

```
v26=request.form['v26']  
v26=float(v26)
```

```
v27=request.form['v27']  
v27=float(v27)
```

```
v28=request.form['v28']  
v28=float(v28)
```

```
amount=request.form['amount']  
amount=float(amount)
```

SIGNUP PAGE

```
body
{
  background-image: url(/static/assets/css/images/test4.jpg);
  background-size: cover;
  background-repeat: none;
}
```

```
.logo
{
  position: absolute;
  width: 50px;
  height: 60px;
  left: 10px;
  top: 10px;
}
```

```
.logo_text
{
  font-family: Montserrat, sans-serif;
  font-size: 30px;
  position: absolute;
  left: 70px;
  top: -2px;
  color: #43a3c1;
}
```

```
.abc
{
  font-family: Montserrat, sans-serif;
  font-size: 30px;
  left: 640px;
  top: 160px;
  position: absolute;
  font-style: normal;
  font-weight: normal;
  display: inline;
  color: #FFFFFF;
}
```

```
.error
{
  position: absolute;
```

left: 632px;

top: 482px;

font-family: Montserrat;

font-style: normal;

font-weight: normal;

font-size: 18px;

line-height: 20px;

color: red;

}

.user-text

{

width: 300.1px;

height: 59.74px;

border-radius: 30px;

position: absolute;

font-family: Montserrat, sans-serif;

font-style: normal;

font-weight: normal;

font-size: 18px;

line-height: 27px;

left: 600px;

top: 240px;

border: none;

padding-left: 20px;

}

.password-text

{

width: 300.1px;

height: 59.74px;

border-radius: 30px;

position: absolute;

font-family: Montserrat, sans-serif;;

font-style: normal;

font-weight: normal;

font-size: 18px;

line-height: 27px;

left: 600px;

top: 320px;

border: none;

padding-left: 20px;

}

.confirm-password-text

{

width: 300.1px;

```
height: 59.74px;
border-radius: 30px;
position: absolute;
font-family: Montserrat, sans-serif;;
font-style: normal;
font-weight: normal;
font-size: 18px;
line-height: 27px;
left: 600px;
top: 400px;
border: none;
padding-left: 20px;
}
```

```
::placeholder {
  position: relative;
  left: 0px;
}
```

```
.l_button{
  position: absolute;
  width: 328px;
  height: 59.74px;
  left: 600px;
  top: 480px;

  background: #43a3c1;
  border-radius: 30px;
  font-family: Montserrat, sans-serif;
  font-style: normal;
  font-weight: normal;
  font-size: 18px;
  line-height: 27px;
  color: #FFFFFFF;
  border: none;
  cursor: pointer;
}
```

```
.l_button:active {
  background-color: #43a3c1;
  transform: translateY(2px);
}
```

```
.acc{
  position: absolute;
  width: 330.1px;
  height: 19.91px;
```

```
left: 635px;
top: 545px;
color: #FFFFFF;
font-family: Montserrat;
font-style: normal;
font-weight: normal;
font-size: 15px;
line-height: 20px;
}

a:link
{
    color: #FFFFFF;
    background-color: transparent;
    text-decoration: none;
}

a:visited
{
    color: #FFFFFF;
    background-color: transparent;
    text-decoration: none;
}

a:hover
{
    color: #43a3c1;
    background-color: transparent;
    text-decoration: underline;
}

a:active
{
    color: #FFFFFF;
    background-color: transparent;
    text-decoration: underline;
}

.error{
    font-family: Montserrat, sans-serif;
    font-size: 20px;
    color: red;
    position: relative;
    left: 585px;
    top: 110px;
    width: 330px;
    text-align: center;
}
```


LOGIN PAGE

```
body
{
  background-image: url(/static/assets/css/images/test2.jpg);
  background-size: cover;
}

.logo
{
  position: absolute;
  width: 50px;
  height: 60px;
  left: 10px;
  top: 10px;
}

.logo_text
{
  font-family: Montserrat, sans-serif;
  font-size: 30px;
  position: absolute;
  left: 70px;
  top: -2px;
  color: #43a3c1;
}

.abc
{
  font-family: Montserrat, sans-serif;
  font-size: 30px;
  left: 600px;
  top: 160px;
  position: absolute;
  font-style: normal;
  font-weight: normal;
  display: inline;
  color: #FFFFFF;
}

.user-text
{
  width: 300.1px;
  height: 59.74px;
```



```
border-radius: 30px;
position: absolute;
font-family: Montserrat, sans-serif;
font-style: normal;
font-weight: normal;
font-size: 18px;
line-height: 27px;
left: 600px;
top: 240px;
border: none;
padding-left: 20px;
}

.password-text
{
width: 300.1px;
height: 59.74px;
border-radius: 30px;
position: absolute;
font-family: Montserrat, sans-serif;;
font-style: normal;
font-weight: normal;
font-size: 18px;
line-height: 27px;
left: 600px;
top: 320px;
border: none;
padding-left: 20px;
}

::placeholder {
position: relative;
left: 0px;
}

.l_button{
position: absolute;
width: 328px;
height: 59.74px;
left: 600px;
top: 400px;

background: #43a3c1;
border-radius: 30px;
font-family: Montserrat, sans-serif;
font-style: normal;
font-weight: normal;
```

```
font-size: 18px;
line-height: 27px;
color: #FFFFFF;
border: none;
cursor: pointer;
}

.l_button:active {
  background-color: #43a3c1;
  transform: translateY(2px);
}

#txt{
  position: absolute;
  /* width: 131.1px;
  height: 19.91px; */
  left: 650px;
  top: 452px;

  font-family: Montserrat;
  font-style: normal;
  font-weight: normal;
  font-size: 15px;
  line-height: 20px;
  color: #FFFFFF;
}

#acc1{
  position: absolute;
  /* width: 131.1px;
  height: 19.91px; */
  left: 830px;
  top: 467px;

  font-family: Montserrat;
  font-style: normal;
  font-weight: normal;
  font-size: 15px;
  line-height: 20px;
  text-decoration: underline;
}

#acc2{
  position: absolute;
  width: 131.1px;
  height: 19.91px;
  left: 780px;
  top: 452px;
```



```
font-family: Montserrat;
font-style: normal;
font-weight: normal;
font-size: 15px;
line-height: 20px;
}

.error
{
    position: absolute;
    left: 632px;
    top: 482px;

    font-family: Montserrat;
    font-style: normal;
    font-weight: normal;
    font-size: 18px;
    line-height: 20px;
    color: red;
}

a:link
{
    color: #FFFFFFF;
    background-color: transparent;
    text-decoration: none;
}

a:visited
{
    color: #FFFFFFF;
    background-color: transparent;
    text-decoration: none;
}

a:hover
{
    color: #43a3c1;
    background-color: transparent;
    text-decoration: underline;
}

a:active
{
    color: #FFFFFFF;
    background-color: transparent;
    text-decoration: underline;
}
```

}

INDEX

```
const toggle = document.getElementById('toggle');
const header = document.getElementById('header');
const navbar = document.getElementById('navbar');
const container = document.getElementById('home');
const activePage = window.location.pathname;
const links = document.querySelectorAll('.nav-bar ul li a');

toggle.onclick = function(active){
  toggle.classList.toggle('active');
  navbar.classList.toggle('active');
  container.classList.toggle('active');
}
document.onclick = function(e){
  if(e.target.id !== 'navbar' && e.target.id !== 'toggle' && e.target.id !== 'home'){
    toggle.classList.remove('active');
    navbar.classList.remove('active');
    container.classList.remove('active');

  }
}

if (links.length) {
  links.forEach((link) => {
    link.addEventListener('click', (e) => {
      links.forEach((link) => {
        link.classList.remove('active');
      });
      link.classList.add('active');
    });
  });
}
```



MAIN

```
/**
 * Template Name: PhotoFolio
 * Updated: Mar 10 2023 with Bootstrap v5.2.3
 * Template URL: https://bootstrapmade.com/photofolio-bootstrap-photography-website-
template/
 * Author: BootstrapMade.com
 * License: https://bootstrapmade.com/license/
 */
document.addEventListener('DOMContentLoaded', () => {
  "use strict";

  /**
   * Preloader
   */
  const preloader = document.querySelector('#preloader');
  if (preloader) {
    window.addEventListener('load', () => {
      setTimeout(() => {
        preloader.classList.add('loaded');
      }, 1000);
      setTimeout(() => {
        preloader.remove();
      }, 2000);
    });
  }

  /**
   * Mobile nav toggle
   */
  const mobileNavShow = document.querySelector('.mobile-nav-show');
  const mobileNavHide = document.querySelector('.mobile-nav-hide');

  document.querySelectorAll('.mobile-nav-toggle').forEach(el => {
    el.addEventListener('click', function(event) {
      event.preventDefault();
      mobileNavToggle();
    });
  });

  function mobileNavToggle() {
    document.querySelector('body').classList.toggle('mobile-nav-active');
    mobileNavShow.classList.toggle('d-none');
    mobileNavHide.classList.toggle('d-none');
  }
}
```

```

/**
 * Hide mobile nav on same-page/hash links
 */
document.querySelectorAll('#navbar a').forEach(navbarlink => {

  if (!navbarlink.hash) return;

  let section = document.querySelector(navbarlink.hash);
  if (!section) return;

  navbarlink.addEventListener('click', () => {
    if (document.querySelector('.mobile-nav-active')) {
      mobileNavToggle();
    }
  });

});

/**
 * Toggle mobile nav dropdowns
 */
const navDropdowns = document.querySelectorAll('.navbar .dropdown > a');

navDropdowns.forEach(el => {
  el.addEventListener('click', function(event) {
    if (document.querySelector('.mobile-nav-active')) {
      event.preventDefault();
      this.classList.toggle('active');
      this.nextElementSibling.classList.toggle('dropdown-active');

      let dropDownIndicator = this.querySelector('.dropdown-indicator');
      dropDownIndicator.classList.toggle('bi-chevron-up');
      dropDownIndicator.classList.toggle('bi-chevron-down');
    }
  });
});

/**
 * Scroll top button
 */
const scrollTop = document.querySelector('.scroll-top');
if (scrollTop) {
  const togglescrollTop = function() {
    window.scrollY > 100 ? scrollTop.classList.add('active') :
scrollTop.classList.remove('active');
  }
  window.addEventListener('load', togglescrollTop);
}

```

```
document.addEventListener('scroll', togglescrollTop);
scrollTop.addEventListener('click', window.scrollTo({
  top: 0,
  behavior: 'smooth'
}));
}
```

```
/**
```

```
 * Initiate glightbox
```

```
 */
```

```
const glightbox = GLightbox({
  selector: '.glightbox'
});
```

```
/**
```

```
 * Init swiper slider with 1 slide at once in desktop view
```

```
 */
```

```
new Swiper('.slides-1', {
  speed: 600,
  loop: true,
  autoplay: {
    delay: 5000,
    disableOnInteraction: false
  },
  slidesPerView: 'auto',
  pagination: {
    el: '.swiper-pagination',
    type: 'bullets',
    clickable: true
  },
  navigation: {
    nextEl: '.swiper-button-next',
    prevEl: '.swiper-button-prev',
  }
});
```

```
/**
```

```
 * Init swiper slider with 3 slides at once in desktop view
```

```
 */
```

```
new Swiper('.slides-3', {
  speed: 600,
  loop: true,
  autoplay: {
    delay: 5000,
    disableOnInteraction: false
  },
  slidesPerView: 'auto',
  pagination: {
```



```
el: '.swiper-pagination',
type: 'bullets',
clickable: true
},
navigation: {
  nextEl: '.swiper-button-next',
  prevEl: '.swiper-button-prev',
},
breakpoints: {
  320: {
    slidesPerView: 1,
    spaceBetween: 40
  },

  1200: {
    slidesPerView: 3,
  }
}
});

/**
 * Animation on scroll function and init
 */
function aos_init() {
  AOS.init({
    duration: 1000,
    easing: 'ease-in-out',
    once: true,
    mirror: false
  });
}
window.addEventListener('load', () => {
  aos_init();
});

});
```



Conclusion and Future Enhancements

10. CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION:

Nowadays, in the global computing environment, online payments are important, because online payments use only the credential information from the credit card to fulfill an application and then deduct money. Due to this reason, it is important to find the best solution to detect the maximum number of frauds in online systems.

Accuracy, Error-rate, Sensitivity and Specificity are used to report the performance of the system to detect the fraud in the credit card. In this paper, three machine learning algorithms are developed to detect the fraud in credit card system. To evaluate the algorithms, 80% of the dataset is used for training and 20% is used for testing and validation. Accuracy, error rate, sensitivity and specificity are used to evaluate for different variables for three algorithms. The accuracy result is shown for SVM; Decision tree and random forest classifier are 99.94, 99.92, and 99.95 respectively. The comparative results show that the Random Forest performs better than the SVM and decision tree techniques.

FUTURE ENHANCEMENTS:

Detection, we did end up creating a system that can, with enough time and data, get very close to that goal. As with any such project, there is some room for improvement here. The very nature of this project allows for multiple algorithms to be integrated together as modules and their results can be combined to increase the accuracy of the final result. This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. This provides a great degree of modularity and versatility to the project. More room for improvement can be found in the dataset. As demonstrated before, the precision of the algorithms increases when the size of dataset is increased. Hence, more data will surely make the model more accurate in detecting frauds and reduce the number of false positives. However, this requires official support from the banks themselves.



Bibliography

11. BIBLIOGRAPHY

1) Books Referred:

- ⦿ Software Engineering by Roger Pressman
- ⦿ Beginning JSP – Wrox Publication
- ⦿ Database Systems by Abraham Silberschatz
- ⦿ Oracle 10g by Gary Cornell and Horstman.

2) Websites referred:

- ⦿ www.google.com
- ⦿ www.w3schools.com/html
- ⦿ www.htmlcodetutorial.com/
- ⦿ <http://en.wikipedia.org>

Books

1. **Bhattacharyya, Susmita, et al.** *Data Mining for Credit Card Fraud Detection*. Springer, 2011.
 - A comprehensive guide to techniques and methodologies for detecting credit card fraud using data mining approaches.
2. **Jha, S., & Bhattacharyya, D.** *Fraud Detection: A Data Mining Approach*. Springer, 2018.
 - This book explores various data mining techniques applied specifically to fraud detection.

Journal Articles

3. **Carcillo, F., & Cavanillas, J.** "A survey of fraud detection in electronic payments." *Computers & Security*, vol. 77, 2018, pp. 386-405.
 - A detailed survey of the techniques and challenges in electronic payment fraud detection.
4. **Fahad, A. et al.** "A survey of credit card fraud detection techniques." *Journal of Systems and Software*, vol. 91, 2014, pp. 55-70.
 - An overview of various methods used in credit card fraud detection, including machine learning techniques.
5. **Zhang, Y., & Wang, Y.** "Credit card fraud detection: A novel approach based on neural networks." *Journal of Applied Statistics*, vol. 45, no. 4, 2018, pp. 731-743.
 - Discusses a neural network-based method for improving credit card fraud detection.

Conference Papers

6. **Hodge, V. J., & Austin, J.** "A survey of outlier detection methodologies." *Artificial Intelligence Review*, vol. 22, 2004, pp. 85-126.
 - A broad survey covering outlier detection, which is essential in identifying fraudulent transactions.
7. **Bhatia, M., & Kaur, R.** "Credit card fraud detection using machine learning techniques." *Proceedings of the International Conference on Machine Learning*, 2020, pp. 185-194.
 - This paper evaluates various machine learning techniques for detecting credit card fraud.

Theses and Dissertations

8. **Vishwakarma, P.** "Credit Card Fraud Detection Using Machine Learning Techniques." Master's Thesis, University of XYZ, 2019.
 - This thesis explores various machine learning models for credit card fraud detection, comparing their effectiveness.

Online Resources

9. **UCI Machine Learning Repository.** "Credit Card Fraud Detection Data Set." [Link to Dataset](#)
 - A widely used dataset for testing fraud detection algorithms.

10. **Kaggle.** "Credit Card Fraud Detection." Kaggle Dataset
 - A popular dataset for data science projects focused on credit card fraud detection.

Reports

11. **The Nilson Report.** "Global Card Fraud Losses 2020." Nilson Report, 2020.
 - An industry report that provides statistics and insights into global credit card fraud losses.

These resources should give you a solid foundation for understanding the various techniques and challenges in credit card fraud detection.