Table of Contents

[ACKNOWLEDGEMENT iii](#_Toc80180563)

[ABSTRACT viii](#_Toc80180564)

[CHAPTER-1 1](#_Toc80180565)

[INTRODUCTION 1](#_Toc80180566)

[1.1 INTRODUCTION 1](#_Toc80180567)

[CHAPTER- 2 5](#_Toc80180568)

[LITERATURE SURVEY 5](#_Toc80180569)

[2.1 LITERATURE REVIEW 5](#_Toc80180570)

[CHAPTER-3 7](#_Toc80180571)

[THEORETICAL BACKGROUND 7](#_Toc80180572)

[3.1 INTRODUCTION: 7](#_Toc80180573)

[3.2 INTRODUCTION TO PYTHON 12](#_Toc80180574)

[3.3 BENFITS OF PYTHON 21](#_Toc80180575)

[CHAPTER-4 30](#_Toc80180576)

[SYSTEM ANALYSIS 30](#_Toc80180577)

[4.1 EXISTING SYSTEM: 30](#_Toc80180578)

[4.1.1 DISADVANTAGES OF EXISTING SYSTEM: 30](#_Toc80180579)

[4.2 PROPOSED SYSTEM: 30](#_Toc80180580)

[4.2.1 ADVANTAGES OF PROPOSED SYSTEM: 31](#_Toc80180581)

[CHAPTER- 5 32](#_Toc80180582)

[SYSTEM DESIGN 32](#_Toc80180583)

[5.1 INTRODUCTION 32](#_Toc80180584)

[5.2 MODULES 32](#_Toc80180585)

[5.2.1 DATASET: 32](#_Toc80180586)

[5.2.2 PREPROCESSING: 32](#_Toc80180587)

[5.2.3 GRAPHS: 32](#_Toc80180588)

[5.2.4 PREDICTION: 32](#_Toc80180589)

[5.3 SYSTEM ARCHITECTURE 33](#_Toc80180590)

[5.4 UML DAIGRAMS 34](#_Toc80180591)

[5.4.1 CONSTRUCTION OF USE CASE DIAGRAMS: 37](#_Toc80180592)

[5.4.2 SEQUENCE DIAGRAMS: 40](#_Toc80180593)

[5.4.3 CLASS DIAGRAM: 42](#_Toc80180594)

[5.4.4 ACTIVITY DIAGRAM: 42](#_Toc80180595)

[CHAPTER-6 44](#_Toc80180596)

[SYSTEM REQUIREMENTS 44](#_Toc80180597)

[6.1 SYSTEM REQUIREMENTS 44](#_Toc80180598)

[6.1.1 HARDWARE REQUIREMENTS: 44](#_Toc80180599)

[6.1.2 SOFTWARE REQUIREMENTS: 44](#_Toc80180600)

[CHAPTER-7 45](#_Toc80180601)

[SYSTEM IMPLEMENTATION 45](#_Toc80180602)

[7.1 INPUT AND OUTPUT DESIGNS 45](#_Toc80180603)

[7.1.1 LOGICAL DESIGN 45](#_Toc80180604)

[7.1.2 PHYSICAL DESIGN 45](#_Toc80180605)

[7.2 INPUT & OUTPUT REPRESENTATION 46](#_Toc80180606)

[7.2.1 INPUT DESIGN 46](#_Toc80180607)

[7.2.2 OBJECTIVES 47](#_Toc80180608)

[7.2.3 OUTPUT DESIGN 47](#_Toc80180609)

[CHAPTER-8 69](#_Toc80180610)

[SYSTEM TESTING 69](#_Toc80180611)

[8.1 INTRODUCTION: 69](#_Toc80180612)

[8.2 LEVELS OF TESTING 69](#_Toc80180613)

[8.2.1 BLACK BOX TESTING 70](#_Toc80180614)

[8.2.2 WHITE BOX TESTING 72](#_Toc80180615)

[CHAPTER-9 74](#_Toc80180616)

[OUTPUT SCREENS 74](#_Toc80180617)

[CONCLUSION 75](#_Toc80180618)

[REFERENCES 76](#_Toc80180619)

**List of Figures**

**Name of the figure Pg.no**

[Figure 3. 1 Structure of cloud computing 7](#_Toc80179715)

[Figure 3. 2 Characteristics of cloud computing 9](#_Toc80179716)

[Figure 3. 3 Structure of service models 10](#_Toc80179717)

[Figure 5. 1 System Architecture 33](#_Toc80179735)

[Figure 5. 2 Use Case Diagram 39](#_Toc80179736)

[Figure 5. 3 Sequence diagram 41](#_Toc80179737)

[Figure 5. 4 Class Diagram 42](#_Toc80179738)

[Figure 5. 5 Activity Diagram 43](#_Toc80179739)

# ABSTRACT

Now a day the usage of UPI has dramatically increased. As UPI becomes the most popular mode of payment for both online as well as regular purchase, cases of fraud associated with it are also rising. In this project, we model the sequence of operations in UPI transaction processing using a Convolutional Neural Network (CNN) and show how it can be used for the detection of frauds. An CNN is initially trained with the normal behavior of a cardholder. If an incoming UPI transaction is not accepted by the trained CNN with sufficiently high probability, it is considered to be fraudulent. At the same time, we try to ensure that genuine transactions are not rejected. We present detailed experimental results to show the effectiveness of our approach and compare it with other techniques available in the literature.

.

.

.

# CHAPTER-1

# INTRODUCTION

## Introduction

The popularity of online shopping is growing day by day. According to an ACNielsen study conducted in 2005, one-tenth of the world’s population is shopping online. Germany and Great Britain have the largest number of online shoppers, and UPI is the most popular mode of payment (59 percent). About 350 million transactions per year were reportedly carried out by Barclaycard, the largest UPI company in the United Kingdom, toward the end of the last century. Retailers like Wal-Mart typically handle much larger number of UPI transactions including online and regular purchases. As the number of UPI users rises world-wide, the opportunities for attackers to steal UPI details and, subsequently, commit fraud are also increasing. The total UPI fraud in the United States itself is reported to be $2.7 billion in 2005 and estimated to be $3.0 billion in 2006, out of which $1.6 billion and $1.7 billion, respectively, are the estimates of online fraud.

Credit-card-based purchases can be categorized into two types:

1) Physical card and

2) Virtual card.

In a physical-card-based purchase, the cardholder presents his card physically to a merchant for making a payment. To carry out fraudulent transactions in this kind of purchase, an attacker has to steal the UPI. If the cardholder does not realize the loss of card, it can lead to a substantial financial loss to the UPI company. In these cond. kind of purchase, only some important information about a card (card number, expiration date, secure code) is required to make the payment. Such purchases are normally done on the Internet or over the telephone. To commit fraud in these types of purchases, a fraudster simply needs to know the card details. Most of the time, the genuine cardholder is not aware that someone else has seen or stolen his card information. The only way to detect this kind of fraud is to analyze the spending patterns on every card and to figure out any inconsistency with respect to the “usual” spending patterns. Fraud detection based on the analysis of existing purchase data of cardholder is a promising way to reduce the rate of successful UPI frauds. Since humans tend to exhibit specific behaviouristic profiles, every cardholder can be represented by a set of patterns containing information about the typical purchase category, the time since the last purchase, the amount of money spent, etc. Deviation from such patterns is a potential threat to the system. Several techniques for the detection of UPI fraud have been proposed in the last few years.

. **Algorithm**

**KNN Classifier Algorithm:**

K-nearest neighbor method can be used for both regression and classification predictive problems. This method helps in interpret output, calculate time and predictive power. The Machine learning techniques are used in various fields. KNN is also one of the machine learning method. This is also called as method of sample-based learning. This will contain the data of past datasets and can be used while predicting the new datasets. This will apply function called as distance function like Manhattan or Euclidean distance. This can be used to compute distance from samples to all other training samples. It calculates the target value for new samples. The target vale will be the weighted sum of target values of the k nearest neighbours. The valve of K can be directly proportional to the prediction. Whenever the valve of K is small this indicates there is high variance and there is low bias. If the valve of the K is larger than this indicates that there is low variance and high bias. The main advantage of this KNN is it does not require any training or the optimization. This KNN uses data samples when predicting the new datasets. Hence it is having higher complexity and also more time consumption.

This work represents a review of K-NN technique for the early prediction of food recommendation. K-NN analysis is used for predicting the unknown parameter from the known parameters. In this work we are considering vitamins as input parameters which are the main parameters to be considered for a good food recommendation, although there are many other factors that can be considered [17]. The unknown value of vitamis can be predicted from the nearest known values of the nearest neighbors by calculation of Euclidean distance between them. Then we would be able to predict type of food for given vitamin parameters. To measure the distance between points in a feature space, various distance functions can be used, in which the Euclidean distance function is the most widely used one[18]. Let p and q are represented as feature vectors. To calculate the distance between p and q, the Euclidean metric is generally used by if a=(a1, a2) and b=(b1,b2) then the distance is given

**Classifications Algorithms**

Onto the part you’ve probably been waiting for all this time: training machine learning algorithms. To be able to test the performance of our algorithms, I first performed an

80/20 train-test split, splitting our balanced data set into two pieces. To avoid overfitting, I used the very common resampling technique of k-fold cross-validation. This simply means that you separate your training data into k parts (folds) and then fit your model on k-1 folds before making predictions for the kth hold-out fold. You then repeat this process for every single fold and average the resulting predictions.

To get a better feeling of which algorithm would perform best on our data, let’s quickly spot-check some of the most popular classification algorithms:

* Logistic Regression
* Linear Discriminant Analysis
* K Nearest Neighbors (KNN)
* Classification Trees
* Support Vector Classifier
* Random Forest Classifier

**Logistic regression**: This is a classification function that uses class for building and uses a single multinomial logistic regression model with a single estimator. Logistic regression usually states where the boundary between the classes exists, also states the class probabilities depend on distance from the boundary, in a specific approach. This moves towards the extremes (0 and 1) more rapidly when data set is larger. These statements about probabilities which make logistic regression more than just a classifier. It makes stronger, more detailed predictions, and can be fit in a different way; but those strong predictions could be wrong. Logistic regression is an approach to prediction, like Ordinary Least Squares (OLS) regression. However, with logistic regression, prediction results in a dichotomous outcome [13]. Logistic regression is one of the most commonly used tools for applied statistics and discrete data analysis. Logistic regression is linear interpolation

# CHAPTER- 2

# literature survey

## 2.1 literature review

**UPI Fraud Detection:-**

UPI fraud detection has drawn a lot of research interest and a number of techniques, with special emphasis on data mining and neural networks, have been suggested. Ghosh and Reilly have proposed UPI fraud detection with a neural network. They have built a detection system, which is trained on a large sample of labelled UPI account transactions. These transactions contain exam-ple fraud cases due to lost cards, stolen cards, application fraud, counterfeit fraud, mail-order fraud, and nonreceived issue (NRI) fraud. Recently, Syeda et al. have used parallel granular neural networks (PGNNs) for improving the speed of data mining and knowledge discovery process in UPI fraud detection. A complete system has been implemented for this purpose. Stolfo et al. suggest a UPI fraud detection system (FDS) using Metalearning techniques to learn models of fraudulent UPI transactions. Metalearning is a general strategy that provides a means for combining and integrating a number of separately built classifiers or models. A metaclassifier is thus trained on the correlation of the predictions of the base classifiers. The same group has also worked on a cost-based model for fraud and intrusion detection. They use Python agents for Metalearning (JAM), which is a distributed data mining system for UPI fraud detection. A number of important performance metrics like True Positive—False Positive (TP-FP) spread and accuracy have been defined by them. Aleskerov et al. present CARDWATCH, a database mining system used for UPI fraud detection. The system, based on a neural learning module, provides an interface to a variety of commercial databases. Kim and Kim have identified skewed distribution of data and mix of legitimate and fraudulent transactions as the two main reasons for the complexity of UPI fraud detection. Based on this observation, they use fraud density of real transaction data as a confidence value and generate the weighted fraud score to reduce the number of misdetections. Fan et al. suggest the application of distributed data mining in UPI fraud detection. Brause et al. have developed an approach that involves advanced data mining techniques and neural network algorithms to obtain high fraud coverage. Chiu and Tsai have proposed Web services and data mining techniques to establish a collaborative scheme for fraud detection in the banking industry. With this scheme, participating banks share knowledge about the fraud patterns in a heterogeneous and distributed environment. To establish a smooth channel of data exchange, Web services techniques such as XML, SOAP, and WSDL are used. Phua et al. have done an extensive survey of existing data-mining-based FDSs and published a comprehensive report. Prodromidis and Stolfo use an agent-based approach with distributed learning for detect-ing frauds in UPI transactions. It is based on artificial intelligence and combines inductive learning algorithms and Metalearning methods for achieving higher accuracy. Phua et al. suggest the use of metaclassifier similar to in fraud detection problems. They consider naive Bayesian C4.5, and Back Propagation neural networks as the base classifiers. A metaclassifier is used to determine which classifier should be considered based on skewness of data. Although they do not directly use UPI fraud detection as the target application, their approach is quite generic. Vatsa et al. have recently proposed a game-theoretic approach to UPI fraud detection. They model the interaction between an attacker and an FDS as a multistage game between two players, each trying to maximize his payoff. The problem with most of the abovementioned approaches is that they require labelled data for both genuine, as well as fraudulent transactions, to train the classifiers. Getting real-world fraud data is one of the biggest problems associated with UPI fraud detection. Also, these approaches cannot detect new kinds of frauds for which labelled data is not available. In contrast, we present a Hidden Markov Model (AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING)-based UPI FDS, which does not require fraud signatures and yet is able to detect frauds by considering a cardholder’s spending habit. We model a UPI transaction processing sequence by the stochastic process of an AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING. The details of items purchased in individual transactions are usually not known to an FDS running at the bank that issues UPI to the cardholders. This can be represented as the underlying finite Markov chain, which is not observable. The transactions can only be observed through the other stochastic process that produces the sequence of the amount of money spent in each transaction. Hence, we feel that AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING is an ideal choice for addressing this problem. Another important advantage of the AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING-based approach is a drastic reduction in the number of False Positives (FPs)—transactions identified as malicious by an FDS although they are actually genuine. Since the number of genuine transactions is a few orders of magnitude higher than the number of malicious transactions, an FDS should be designed in such a way that the number of FPs is as low as possible. Otherwise, due to the “base rate fallacy” effect, bank administrators may tend to ignore the alarms. To the best of our knowledge, there is no other published literature on the application of AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING for UPI fraud detection.

**AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING Background:-**

An AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING is a double embedded stochastic process with two hierarchy levels. It can be used to model much more complicated stochastic processes as compared to a traditional Markov model. An AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING has a finite set of states governed by a set of transition probabilities. In a particular state, an outcome or observation can be generated according to an associated probability distribution. It is only the outcome and not the state that is visible to an external observer. AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING-based applications are common in various areas such as speech recognition, bioinformatics, and genomics. In recent years, Joshi and Phoba have investigated the capabilities of AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING in anomaly detection.

**Title: BLAST-SSAHA Hybridization for UPI Fraud Detection Author:** Amlan Kundu, Suvasini Panigrahi, Shamik Sural and Arun K. Majumdar **Description:**

A phenomenal growth in the number of UPI transactions, especially for online purchases, has recently led to a substantial rise in fraudulent activities. Implementation of efficient fraud detection systems has thus become imperative for all UPI issuing banks to minimize their losses. In real life, fraudulent transactions are interspersed with genuine transactions and simple pattern matching is not often sufficient to detect them accurately. Thus, there is a need for combining both anomaly detection as well as misuse detection techniques. In this paper, we propose to use two-stage sequence alignment in which a profile analyzer (PA) first determines the similarity of an incoming sequence of transactions on a given UPI with the genuine cardholder's past spending sequences. The unusual transactions traced by the profile analyzer are next passed on to a deviation analyzer (DA) for possible alignment with past fraudulent behavior. The final decision about the nature of a transaction is taken on the basis of the observations by these two analyzers. In order to achieve online response time for both PA and DA, we suggest a new approach for combining two sequence alignment algorithms BLAST and SSAHA

**TITLE: Fast algorithms for mining association rules in large databases AUTHOR:** R. AGRAWAL AND R. SRIKANT.

## DESCRIPTION:

The major consequences are loss of billions of dollars each year, investor confidence or corporate reputation. A study area called Financial Fraud Detection (FFD) is obligatory, in order to prevent the destructive results caused by financial fraud. In this study, we propose a new method based on Grammar-based Genetic Programming (GBGP), multi-objectives optimization and ensemble learning for solving FFD problems. We comprehensively compare the proposed method with Logistic Regression (LR), Neural Networks (NNs), Support Vector Machine (SVM), Bayesian Networks (BNs), Decision Trees (DTs), AdaBoost, Bagging and LogitBoost on four FFD datasets. The experimental results showed the effectiveness of the new approach in the given FFD problems including two real-life problems. The major implications and significances of the study can concretely generalize for two points. First, it evaluates a number of data mining techniques by the given real-life classification problems. Second, it suggests a new method based

# Chapter-3

# Theoretical background

## 3.1 Introduction:

**What is Machine Learning?**

Tom Mitchell states machine learning as “A computer program is said to learn from experience and from some tasks and some performance on, as measured by, improves with experience”. Machine Learning is combination of correlations and relationships, most machine learning algorithms in existence are concerned with finding and/or exploiting relationship between datasets. Once Machine Learning Algorithms can pinpoint on certain correlations, the model can either use these relationships to predict future observations or generalize the data to reveal interesting patterns. In Machine Learning there are various types of algorithms such as Regression, Linear Regression, Logistic Regression, Naive Bayes Classifier, Bayes theorem, KNN (K-Nearest Neighbor Classifier), Decision Tress, Entropy, ID3, SVM (Support Vector Machines), K-means Algorithm, Random Forest and etc.,

The name machine learning was coined in 1959 by Arthur Samuel. Machine learning explores the study and construction of algorithms that can learn from and make predictions on data Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter subfield focuses more on exploratory data analysis and is known as unsupervised learning.

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends in the data.

Machine learning tasks Machine learning tasks are typically classified into several broad categories:

Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs. As special cases, the input signal can be only partially available, or restricted to special feedback.

Semi-supervised learning: The computer is given only an incomplete training signal: a training set with some (often many) of the target outputs missing.

Active learning: The computer can only obtain training labels for a limited set of instances (based on a budget), and also has to optimize its choice of objects to acquire labels for. When used interactively, these can be presented to the user for labelling.

Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).

Reinforcement learning: Data (in form of rewards and punishments) are given only as feedback to the program's actions in a dynamic environment, such as driving a vehicle or playing a game against an opponent.

**Benefits of Machine Learning:**

1. **Achieve economies of scale** – increase volume output or productivity with fewer people. Your cost per unit, project or product plummets.
2. **Reduce spending on technology infrastructure.** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
3. **Globalize your workforce on the cheap.** People worldwide can access the cloud, provided they have an Internet connection.
4. **Streamline processes.** Get more work done in less time with less people.
5. **Reduce capital costs.** There’s no need to spend big money on hardware, software or licensing fees.
6. **Improve accessibility.** You have access anytime, anywhere, making your life so much easier!
7. **Monitor projects more effectively.** Stay within budget and ahead of completion cycle times.
8. **Less personnel training is needed.** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
9. **Minimize licensing new software.** Stretch and grow without the need to buy expensive software licenses or programs.
10. **Improve flexibility.** You can change direction without serious “people” or “financial” issues at stake.

**Advantages:**

1. **Price:** Pay for only the resources used.
2. **Security**: Cloud instances are isolated in the network from other instances for improved security.
3. **Performance:** Instances can be added instantly for improved performance. Clients have access to the total resources of the Cloud’s core hardware.
4. **Scalability:** Auto-deploy cloud instances when needed.
5. **Uptime:** Uses multiple servers for maximum redundancies. In case of server failure, instances can be automatically created on another server.
6. **Control:** Able to login from any location. Server snapshot and a software library lets you deploy custom instances.
7. **Traffic:** Deals with spike in traffic with quick deployment of additional instances to handle the load.

## 3.2 Introduction to PYTHON

**Python Technology**

Python technology is both a programming language and a platform.

**The python Programming Language**

THE PYTHON PROGRAMMING LANGUAGE IS A HIGH-LEVEL LANGUAGE THAT CAN BE CHARACTERIZED BY ALL OF THE FOLLOWING BUZZWORDS:

* + - Simple
    - Architecture neutral
    - Object oriented
    - Portable
    - Distributed
    - High performance
    - Interpreted
    - Multithreaded
    - Robust
    - Dynamic
    - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Python programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Python byte codes —the platform-independent codes interpreted by the interpreter on the Python platform. The interpreter parses and runs each Python byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

FEATURES OF MACHINE LEARNING

• It is nothing but automating the Automation.

• Getting computers to program themselves.

• Writing Software is bottleneck.

• Machine leaning models involves machines learning from data without the help of humans or any kind of human intervention.

• Machine Learning is the science of making of making the computers learn and act like humans by feeding data and information without being explicitly programmed.

• Machine Learning is totally different from traditionally programming, here data and output is given to the computer and in return it gives us the program which provides solution to the various problems. Below is the figure.

**Traditional Programming vs Machine Learning**

• Machine Learning is a combination of Algorithms, Datasets, and Programs.

• There are Many Algorithms in Machine Learning through which we will provide us the exact solution in predicting the disease of the patients.

• How Does Machine Learning Works?

• Solution to the above question is Machine learning works by taking in data, finding relationships within that data and then giving the output.

**Machine Learning Model**

• There are various applications in which machine learning is implemented such as Web search, computing biology, finance, e-commerce, space exploration, robotics, social networks, debugging and much more.

• There are 3 types of machine learning supervised, unsupervised, and reinforcement.

**BENEFITS OF PYTHON**

• Presence of Third-Party Modules

• Extensive Support Libraries

• Open Source and Community Development

• Learning Ease and Support Available

• User-friendly Data Structures

• Productivity and Speed

• Highly Extensible and Easily Readable Language.

**Python**

Python is high level language and it is also integrated version of the program. Python is an object-oriented approach and its main aim to help programmers to write the code clearly, logical code for small and large scale of project.

Pytrhon is dynamically typed and garbage collected it also support multiple programming and it is both procedure and object oriented and also functional programming. And structural programming also supported. It has many built in function it also supports filter, map and reduce function. All the machine learning algorithm and the libraries are being supported by the python programming language. Python also support list, dict, sets and other generators. Python code can be run in different platform such as anaconda, PyCharm etc.

The main goal of this programing language is as follows:

• Python is simple, object-oriented programming language.

• The language and implementation should provide support for software engineering principles such as strong type library preset for different machine learning algorithm, and all other algorithm in simple manner.

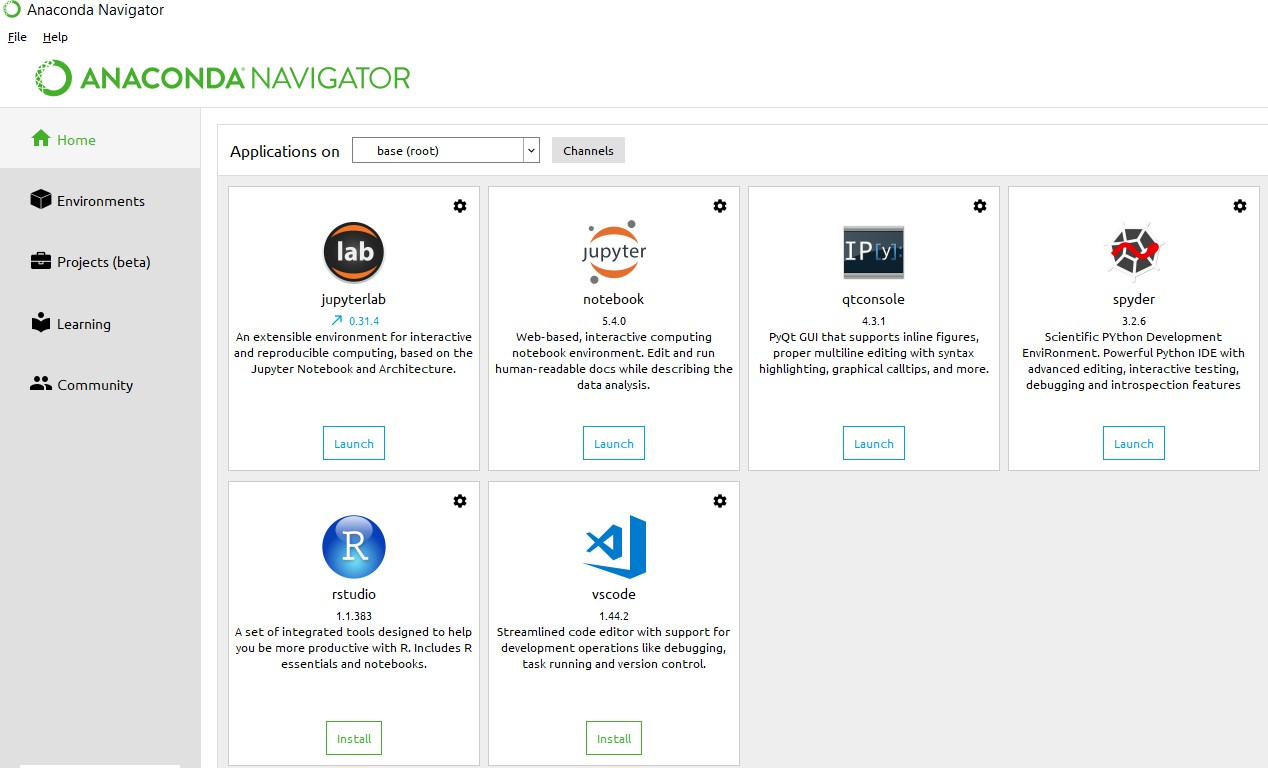
• Coding will be smooth in python and the data analysis can be easily done in python.

This is so much so to the point where we now have modules and APIs at our disposal, and you can engage in machine learning very easily without almost any knowledge at all of how it works. With the defaults from Scikit-learn, you can get 90-95% accuracy on many tasks right out of the gate. Machine learning is a lot like a car, you do not need to know much about how it works in order to get an incredible amount of utility from it.

Despite the apparent age and maturity of machine learning, I would say there's no better time than now to learn it, since you can actually use it. Machines are quite powerful, the one you are working on can probably do most of this series quickly. Data is also very plentiful lately.

**Anaconda**

Anaconda is free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine Learning applications, Large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. It is developed and maintained by Anaconda, Inc. The distribution incudes data-science packages suitable for Windows, Linux, and macOS. Packaged versions are required and are managed by the package management system anaconda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depends on, and a small number of other packages.



**Anaconda Console**

**Jupyter notebook**

Jupiter Notebook or so called IPython Notebook is an interactive web based computational mean for starting with Jupiter Notebook documents. The term notebook itself is a huge entity to represent the integration with different entity sets. JSON is the main document form from the same for the execution which follows the brief on the schema and the input and output means. It has high integration with several language set and has various flexibilities with the choices. The extension used for the same is “.ipynb” which runs in this platform. It’s an open-source software package with interactive communication means. It has it’s open standards for the same. It’s an open community best for budding programmers . The flexibility of the same is phenomenon and splendidly done the configuration and integration of the same is simplest and easy on hold so that no prior distortion is generated and the efficiency of the same is measured through out any system of choice.

It’s the best software sets that been used across cross for designing and developing of the products and support wide help support. Not only to that, it provides scalability in the code and the deployment of the same. Various Language can be changed and the project can be undertaken on the same. The created notebook files can be shared and stored in various means for further utilization. It supports cultivated and interactive output sets. Easily crossed over for graphing, plotting and visualizing of the elements. Data Integration of the same is to it’s best. The integration of big data and it can process chunks of values in an approx. time which gives a better performance and the higher computational means. Various works on data like cleaning, cleansing, transforming modeling and visualizing can be done by the same

Machine learning is the ability that gives the computer to learn without being explicitly programmed. There are two types of machine learning:

Supervised Learning: supervised learning is the learning of the labelled data. It is the types of machine learning that maps the input and output based on the examples input-output pairs. In supervised learning each training data having pairs of input and desired outputs values. Supervised learning algorithm analyzes the training data and produces a function which can be used for mapping of new data.

Fig 2.1 Supervised Learning The output to solve the supervised learning algorithm are as:

• Determine the types of data, before doing anything else the user should understand which types of data set is to be used for training the data.

• Gathered the training data sets either in form of human experts or from measurements.

• Determine the feature of inputs from the learned data and depends on the inputs it changed into feature vector; number of features should not be large but should contains enough information to accurately predict the outputs.

• Check the learned function and the learned algorithm for example we use support vector machines or decisions tree.

• Complete the design and run the trained data sets.

• Analyzed the output and verify the data sets to get the accurate outputs.

Unsupervised Learning:

Unsupervised learning is a type of machine learning that helps in finding the previously unknown patterns in the data set without any known labels. It is known as self- organization and allows modelling probability densities of given inputs.

Fig 2.2 unsupervised Learning Some of the algorithm used in unsupervised learning are:

• Clustering

• Anomaly detection

• Neural networks

• Approach for learning latent variable models

• Non labelled data

Semi Supervised Machine Learning algorithm: It’s like the middle man which have some labeled data and some unlabeled which can be prosed by the both the structured and unsupervised learning.

The algorithms have been compared based upon the parameters: Size of the dataset and Number of technical indicators used. Accuracy and F-measure values have been computed for each algorithm. Long term model has been used to compute the accuracy and F-measure.

Reinforcement Learning: This type of learning is used to reinforce or strengthen the network based on critic information. That is, a network being trained under reinforcement learning, receives some feedback from the environment. However, the feedback is evaluative and not instructive as in the case of supervised learning. Based on this feedback, the network performs the adjustments of the weights to obtain better critic information in future.

This learning process is similar to supervised learning but we might have very less information. The following figure gives the block diagram of reinforcement learning:

**import numpy as np**

* NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
* At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:
  + NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
  + The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
  + NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
  + A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

**import pandas as pd**

* Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.
* Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

**import matplotlib**

* Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.
* One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

**import seaborn as sns**

* Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with panda’s data structures.
* Seaborn helps you explore and understand your data. Its plotting functions operate on data frames and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them

**import sklearn**

* In general, a learning problem considers a set of n samples of data and then tries to predict properties of unknown data. If each sample is more than a single number and, for instance, a multi-dimensional entry (aka multivariate data), it is said to have several attributes or features.
* Learning problems fall into a few categories:
  + supervised learning, in which the data comes with additional attributes that we want to predict (Click here to go to the scikit-learn supervised learning page).This problem can be either:
* classification: samples belong to two or more classes and we want to learn from already labeled data how to predict the class of unlabeled data. An example of a classification problem would be handwritten digit recognition, in which the aim is to assign each input vector to one of a finite number of discrete categories. Another way to think of classification is as a discrete (as opposed to continuous) form of supervised learning where one has a limited number of categories and for each of the n samples provided, one is to try to label them with the correct category or class.
* regression: if the desired output consists of one or more continuous variables, then the task is called regression. An example of a regression problem would be the prediction of the length of a salmon as a function of its age and weight.
  + unsupervised learning, in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space, known as density estimation, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization (Click here to go to the Scikit-Learn unsupervised learning page).

**import imblearn**

* In machine learning, while building a classification model we sometimes come to situations where we do not have an equal proportion of classes. That means when we have class imbalance issues for example, we have 500 records of 0 class and only 200 records of 1 class. This is called a class imbalance. All machine learning models are designed in such a way that they should attain maximum accuracy but in these types of situations, the model gets biased towards the majority class and will, at last, reflect on precision and recall. So how to build a model on these types of data set in a manner that the model should correctly classify the respective class and does not get biased.
* To get rid of these imbalance class issues few techniques are used called as Imblearn Technique that is mainly used in these types of situations. Imblearn techniques help to either up sample the minority class or down sample the majority class to match the equal proportion. Through this article, we will discuss imblearn techniques and how we can use them to do up sampling and down sampling. For this experiment, we are using Pima Indian Diabetes data since it is an imbalance class data set. The data is available on Kaggle for downloading.

**import sklearn.metrics as m**

There are 3 different APIs for evaluating the quality of a model’s predictions:

• Estimator score method: Estimators have a score method providing a default evaluation criterion for the problem they are designed to solve. This is not discussed on this page, but in each estimator’s documentation.

• Scoring parameter: Model-evaluation tools using cross-validation (such as model\_selection.cross\_val\_score and model\_selection.GridSearchCV) rely on an internal scoring strategy. This is discussed in the section The scoring parameter: defining model evaluation rules.

• Metric functions: The metrics module implements functions assessing prediction error for specific purposes. These metrics are detailed in sections on Classification metrics, Multilabel ranking metrics, Regression metrics and Clustering metrics.

Finally, Dummy estimators are useful to get a baseline value of those metrics for random predictions.

**from sklearn.cluster import KMeans**

* Clustering of unlabelled data can be performed with the module sklearn.cluster.
* Each clustering algorithm comes in two variants: a class, that implements the fit method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the labels\_ attribute.

**from imblearn.under\_sampling import RandomUnderSampler**

Imbalanced datasets are those where there is a severe skew in the class distribution, such as 1:100 or 1:1000 examples in the minority class to the majority class.

This bias in the training dataset can influence many machine learning algorithms, leading some to ignore the minority class entirely. This is a problem as it is typically the minority class on which predictions are most important.

One approach to addressing the problem of class imbalance is to randomly resample the training dataset. The two main approaches to randomly resampling an imbalanced dataset are to delete examples from the majority class, called undersampling, and to duplicate examples from the minority class, called oversampling.

In this tutorial, you will discover random oversampling and undersampling for imbalanced classification

After completing this tutorial, you will know:

• Random resampling provides a naive technique for rebalancing the class distribution for an imbalanced dataset.

• Random oversampling duplicates examples from the minority class in the training dataset and can result in overfitting for some models.

• Random undersampling deletes examples from the majority class and can result in losing information invaluable to a model.

**from sklearn.preprocessing import StandardScaler**

* Standardize features by removing the mean and scaling to unit variance
* The standard score of a sample x is calculated as:
* z = (x - u) / s
* where u is the mean of the training samples or zero if with\_mean=False, and s is the standard deviation of the training samples or one if with\_std=False.
* Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using transform.
* Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance).
* For instance many elements used in the objective function of a learning algorithm (such as the RBF kernel of Support Vector Machines or the L1 and L2 regularizers of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger that others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

**from sklearn.model\_selection import train\_test\_split**

* Before discussing train\_test\_split, you should know about Sklearn (or Scikit-learn). It is a Python library that offers various features for data processing that can be used for classification, clustering, and model selection.
* Model\_selection is a method for setting a blueprint to analyze data and then using it to measure new data. Selecting a proper model allows you to generate accurate results when making a prediction.
* To do that, you need to train your model by using a specific dataset. Then, you test the model against another dataset.
* If you have one dataset, you'll need to split it by using the Sklearn train\_test\_split function first.
* train\_test\_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually.
* By default, Sklearn train\_test\_split will make random partitions for the two subsets. However, you can also specify a random state for the operation.

**import seaborn as sns**

* Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
* For a brief introduction to the ideas behind the library, you can read the introductory notes. Visit the installation page to see how you can download the package and get started with it. You can browse the example gallery to see what you can do with seaborn, and then check out the tutorial and API reference to find out how.
* To see the code or report a bug, please visit the GitHub repository. General support questions are most at home on stackoverflow or discourse, which have dedicated channels for seaborn.

**from sklearn.model\_selection import cross\_val\_score**

* Learning the parameters of a prediction function and testing it on the same data is a methodological mistake: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called overfitting. To avoid it, it is common practice when performing a (supervised) machine learning experiment to hold out part of the available data as a test set X\_test, y\_test. Note that the word “experiment” is not intended to denote academic use only, because even in commercial settings machine learning usually starts out experimentally. Here is a flowchart of typical cross validation workflow in model training. The best parameters can be determined by grid search techniques.
* **import tensorflow as tf**

To do machine learning TensorFlow, you are likely to need to define, save, and restore a model.

A model is, abstractly:

• A function that computes something on tensors (a forward pass)

• Some variables that can be updated in response to training

In this guide, you will go below the surface of Keras to see how TensorFlow models are defined. This looks at how TensorFlow collects variables and models, as well as how they are saved and restored.



**from tensorflow.keras import layers, models, optimizers, callbacks, regularizers, metrics**

When to use a Sequential model

A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

A Sequential model is not appropriate when:

• Your model has multiple inputs or multiple outputs

• Any of your layers has multiple inputs or multiple outputs

• You need to do layer sharing

• You want non-linear topology (e.g. a residual connection, a multi-branch model)

* **from sklearn import metrics**

Accuracy classification score.

In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y\_true.

# Chapter-4

# System analysis

## 4.1 EXISTING SYSTEM:

All the existing method to detect the UPI was on the mode like the detection occurs only after the complaint of the card holder about fraud done. It is not a convenient way to avoid the loss happens to the cad holder. After getting the complaint they detected the fraud on the basis of the IP address. For this they need the help of the Cyber crime to detect the fraud and make action on it. It takes so much man power.

***Disadvantages***

1. The main disadvantage of the existing system is the detection occurs only after gets a written complaint.
2. In the existing system there is physical inconvenience exists.
3. The period occurs to detect the fraud will cause so many losses to the card holder.
4. There is no particular security system in the existing so a hacker can easily access others card.

## 4.2 PROPOSED SYSTEM:

Here we are introducing a project for the UPI fraud detection using Convolutional Neural Network (CNN). It is done on the basis of the spending profile of the card holder. The usual spending of the cardholder is being checked by the FDS (Fraud Detection system) in the bank .The system checks all the spending of the user. When it turns unusual the method blocks the transaction on the card. And it alerts the bank. It occurs automatically. It doesn’t need any man power.

.

### 4.2.1 ADVANTAGES OF PROPOSED SYSTEM:

 1) The main advantage is that the detection occurs much faster than any other method.

2) In all the existing systems the real card holder should checked for the Fraud detection. But in our method there is no need of the physical inconveniences of the card holder. All the checking and the detection occur automatically.

3) This project needs no man power for the detection.

4) This project provides most accurate method in UPI fraud detection.

.

# CHAPTER- 5

# SYSTEM design

## 5.1 introduction

System Design Introduction:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

## 5.2 modules

* Register.
* Sign in.
* Security.
* User side.

Purchase

### 5.2.1 Service Provider:

* **1. Login**
* The Login module gives a login form to the user with a user name and password. The user can access the special features only when they enters correct user name and password
* **2. Register**
* In this module the hard holder registers new card. For this they are gives their personal details, UPI details. In this module the user also can fix security questions and answers for security purpose

### 5.2.2 Purchase:

* In this module all the transaction process like purchase with the UPI occurs. The user submits the total amount to be credited after the completion of purchase. The transaction occurs only when the total amount is below the spending limit. If it exceeds the limit security questions are asked. The user can proceed only when the answers are correct. Otherwise the card will be blocked.

.. **3. User side**

This module is for the user to view the home page, purchase things or view reports. This report deals with what the user did with the UPI like purchase, transactions etc.

### 5.2.3 Methodology:

As seen from the above figure, we can see how the data is divided into different sets and then trained for different models. • The dataset was first divided into training set (80%) and pre-training set(20%). • The pre-training set was divided into pre-train(80%) and pre-test(20%) • Now, the training set is further is divided into train(80%) and validation set(20%). This train set is again divided into train(80%) and test set(20%). So, now I have train validation and test sets separate which are nonoverlapping. • The pretrain set was used to find the best models for the given dataset. I took best 4 models using pretest set. Their performance was compared based on their mean absolute errors. • Once the best 4 models were obtained, hyperparameters for these models were tuned and the best parameter was selected.

**MODULES:**

**Data collection:**

* we will take UPI Fraud set from Kaggle which has features as tweet data and labels as transaction and value.

**Data preprocessing:**

* Features are extracted from data set and stored in variable as xtrain variable and labels are stored in y train variable. Data is preprocessing by standard scalar function and new features and labels are generated.

**Testing training:**

* In this stage data is sent to testing and training function and divided in to four parts x test train, and y test train. Train variables are used for passing to algorithm where as test are used for calculating accuracy of the algorithm.

**Initializing Multiple Algorithms and training with Logistic regression:**

* In this stage machine learning algorithms are initialized and train values are given to algorithm by this information algorithm will know what are features and what are labels. Then data is modeled and stored as pickle file in the system which can be used for prediction.
* Data set is trained with multiple algorithms and accuracy of each model is calculated and best model is used for prediction

**Predict data:**

* In this stage new data is taken as input and trained models are loaded using pickle and then values are preprocessed and passed to predict function to find out result which is showed on web application.

## 5.3 system architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system. Organized in a way that supports reasoning about the structures and behaviors of the system.

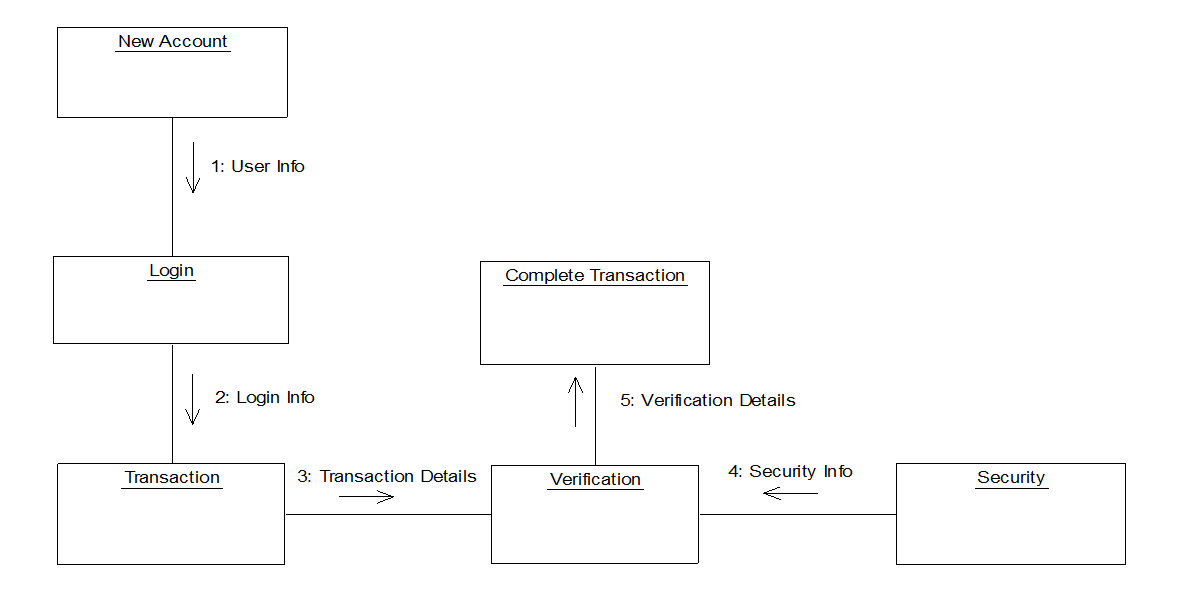


Figure 5. 1 System Architecture

3-Tier Architecture:

The three-tier software architecture (a three-layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems.

**Advantages of Three-Tier:**

* Separates functionality from presentation.
* Clear separation – better understanding.
* Changes limited to well define components.
* Can be running on WWW.
* Effective network performance.

## 5.4 UML DAIGRAMS

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



<<Actor name>>

An actor is someone or something that:

Interacts with or uses the system.

* Provides input to and receives information from the system.
* Is external to the system and has no control over the use cases.

Actors are discovered by examining:

* Who directly uses the system?
* Who is responsible for maintaining the system?
* External hardware used by the system.
* Other systems that need to interact with the system.

Questions to identify actors:

* + Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
  + Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
  + Which external hardware or systems (if any) use the system to perform tasks?
  + What problems does this application solve (that is, for whom)?
  + And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

1. System Administrator
2. Customer
3. Customer Care

Identification of use cases:

Use case: A use case can be described as a specific way of using the system from a user’s (actor’s) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

* Pattern of behavior the system exhibits
* A sequence of related transactions performed by an actor and the system
* Delivering something of value to the actor

Use cases provide a means to:

* capture system requirements
* communicate with the end users and domain experts
* test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

* For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
* Name the use cases.
* Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

* What are the tasks of each actor?
* Will any actor create, store, change, remove or read information in the system?
* What use case will store, change, remove or read this information?
* Will any actor need to inform the system about sudden external changes?
* Does any actor need to inform about certain occurrences in the system?
* What usecases will support and maintains the system?

**1.2 Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

* When and how the use case starts and ends
* Use case/actor interactions
* Data needed by the use case
* Normal sequence of events for the use case
* Alternate or exceptional flows

### 5.4.1 Construction of Use case diagrams:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



Figure 5. 2 Use Case Diagram

### 5.4.2 SEQUENCE DIAGRAMS:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Figure 5. 3 Sequence diagram

### 5.4.3 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



Figure 5. 4 Class Diagram

### 5.4.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

****

Figure 5. 5 Activity Diagram

# CHAPTER-6

# system requirements

## 6.1 SYSTEM REQUIREMENTS

### 6.1.1 HARDWARE REQUIREMENTS:

* System : Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz
* Hard Disk : 1 TB.
* Input Devices : Keyboard, Mouse
* Ram : 4 GB.

### 6.1.2 SOFTWARE REQUIREMENTS:

* Operating system : Windows XP/7/10.
* Coding Language : Python
* Tool : Anaconda
* Interface : OPENCV

## PRELIMINARY INVESTIGATION

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine ,address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, ie. preliminary investigation begins. The activity has three parts:

* **Request Clarification**
* **Feasibility Study**
* **Request Approval**

**REQUEST CLARIFICATION**

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network(LAN). In today’s busy schedule man need everything should be provided in a readymade manner. So taking into consideration of the vastly use of the net in day to day life, the corresponding development of the portal came into existence.

**FEASIBILITY ANALYSIS**

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analyzed are

* **Operational Feasibility**
* **Economic Feasibility**
* **Technical Feasibility**

###### Operational Feasibility

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the Admin and helps him in effectively tracking the project progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

**Economic Feasibility**

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization. So the project is economically feasible.

###### Technical Feasibility

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet. The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and WebLogic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

**4.3.3 REQUEST APPROVAL**

Not all request projects are desirable or feasible. Some organization receives so many project requests from client users that only few of them are pursued. However, those projects that are both feasible and desirable should be put into schedule. After a project request is approved, it cost, priority, completion time and personnel requirement is estimated and used to determine where to add it to any project list. Truly speaking, the approval of those above factors, development works can be launched.

**SYSTEM DESIGN AND DEVELOPMENT**

**INPUT DESIGN**

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

OUTPUT DESIGN

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

# Chapter-7

# System implementation

To conduct studies and analyses of an operational and technological nature, and To promote the exchange and development of methods and tools for operational analysis as applied to defense problems.

## 7.1 input and output designs

### 7.1.1 Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems design are included. Logical design includes ER Diagrams i.e. Entity Relationship Diagrams

### 7.1.2 Physical design

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified / authenticated, how it is processed, and how it is displayed as output. In Physical design, following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

## 7.2 INPUT & OUTPUT REPRESENTATION

### 7.2.1 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.

### 7.2.2 Objectives

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

### 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.

* 1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
  2. Select methods for presenting information.
  3. Create document, report, or other formats that contain information produced by the system.

**Code**

import numpy as **np**

import **pandas** as **pd**

from **sklearn**.**preprocessing** import **StandardScaler**

import tensorflow as **tf**

from flask import Flask, request, render\_template

dataset = **pd**.**read\_csv**('dataset/upi\_fraud\_dataset.csv', index\_col=0)

x = dataset.iloc[:, : 10].values

y = dataset.iloc[:, 10].values

scaler = **StandardScaler**()

scaler.**fit\_transform**(x)

model = **tf**.keras.models.load\_model('model/project\_model1.h5')

app = Flask(\_\_name\_\_)

**@app.route**('/')

**@app.route**('/first')

def **first**():

    return render\_template('first.html')

**@app.route**('/login')

def **login**():

    return render\_template('login.html')

def **home**():

    return render\_template('home.html')

**@app.route**('/upload')

def **upload**():

    return render\_template('upload.html')

**@app.route**('/preview',methods=["POST"])

def **preview**():

    if request.method == 'POST':

        dataset = request.files['datasetfile']

        df = **pd**.**read\_csv**(dataset,encoding = 'unicode\_escape')

        df.**set\_index**('Id', inplace=True)

        return render\_template("preview.html",df\_view = df)

**@app.route**('/prediction1', methods=['GET'])

def **prediction1**():

    return render\_template('index.html')

**@app.route**('/chart')

def **chart**():

    return render\_template('chart.html')

**@app.route**('/detect', methods=['POST'])

def **detect**():

    trans\_datetime = **pd**.**to\_datetime**(request.form.get("trans\_datetime"))

    v1 = trans\_datetime.hour

    v2 = trans\_datetime.day

    v3 = trans\_datetime.month

    v4 = trans\_datetime.year

    v5 = **int**(request.form.get("category"))

    v6 = **float**(request.form.get("card\_number"))

    dob = **pd**.**to\_datetime**(request.form.get("dob"))

    v7 = **np**.round((trans\_datetime - dob) // **np**.timedelta64(1, 'Y'))

    v8 = **float**(request.form.get("trans\_amount"))

    v9 = **int**(request.form.get("state"))

    v10 = **int**(request.form.get("zip"))

    x\_test = **np**.array([v1, v2, v3, v4, v5, v6, v7, v8, v9, v10])

    y\_pred = model.predict(scaler.**transform**([x\_test]))

    if y\_pred[0][0] <= 0.5:

        result = "VALID TRANSACTION"

    else:

        result = "FRAUD TRANSACTION"

    return render\_template('result.html', OUTPUT='{}'.**format**(result))

if \_\_name\_\_ == "\_\_main\_\_":

    app.run()

# Chapter-8

# System testing

## 8.1 INTRODUCTION:

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

## 8.2 Levels of Testing

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit 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 the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

### 8.2.1 BLACK BOX TESTING

What is Black Box Testing?

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

* Functional testing – This black box testing type is related to functional requirements of a system; it is done by software testers.
* Non-functional testing – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* Regression testing – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

### 8.2.2 WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

WHAT DO YOU VERIFY IN WHITE BOX TESTING?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

HOW DO YOU PERFORM WHITE BOX TESTING?

  To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This  method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

Unit testing:

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­1 |
| Name of Test: ­ | Data Set Feature Label Extraction |
| Items being tested: ­ | Features and Labels are extracted or not |
| Sample Input: ­ | Dataset values form csv file |
| Expected output: ­ | Features copied to x labels to y |
| Actual output: ­ | Data with features and labels are displayed. |
| **Remarks: ­** | **Pass.** |

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­2 |
| Name of Test: ­ | Register login |
| Items being tested: ­ | User registration details stored in Database and login success or Fail |
| Sample Input: ­ | Username, password, email, phone number, address |
| Expected output: ­ | Registration successful, login success |
| Actual output: ­ | Details stored in database and verification is success for correct details |
| Remarks: ­ | pass |

**Integration Testing:**

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. It occurs after unit testing and before validation testing. Integration testing can be done in two ways: Bottom­up integration testing and Top­down integration testing.

* + 1. **Bottom­up Integration**

This testing begins with unit testing, followed by tests of progressively higher­level combinations of units called modules or builds.

* + 1. **Top­down Integration**

In this testing, the highest­level modules are tested first and progressively, lower­level modules are tested thereafter.

In a comprehensive software development environment, bottom­up testing is usually done first, followed by top­down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations. Table 6.5 shows the test cases for integration testing and their results

|  |  |
| --- | --- |
| Sl # Test Case : ­ | ITC­1 |
| Name of Test: ­ | Input UPI Frud data details values |
| Item being tested: ­ | UPI Frud data is predicted or not |
| Sample Input: ­ | UPI Frud |
| Expected output: ­ | Given UPI Frud or not |
| Actual output: ­ | UPI Frud data is detected showed on webpage |
| Remarks: ­ | Pass. |

|  |  |
| --- | --- |
| Sl # Test Case : ­ | ITC­2 |
| Name of Test: ­ | Predict UPI Frud |
| Item being tested: ­ | Prediction is success or not for UPI Frud values |
| Sample Input: ­ | UPI Frud predicted values |
| Expected output: ­ | UPI Frud details are stored in database |
| Actual output: ­ | UPI Frud information is displayed to user |
| Remarks: ­ | Pass. |

**System testing**:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black­box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is important because of the following reasons:

System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.

The application is tested thoroughly to verify that it meets the functional and technical specifications.

The application is tested in an environment that is very close to the production environment where the application will be deployed.

System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

System Testing is shown in below tables

|  |  |
| --- | --- |
| Sl # Test Case : ­ | STC­1 |
| Name of Test: ­ | System testing in various versions of OS |
| Item being tested: ­ | OS compatibility. |
| Sample Input: ­ | Execute the program in windows XP/ Windows­7/8 |
| Expected output: ­ | Performance is better in windows­7 |
| Actual output: ­ | Same as expected output, performance is better in windows­7 |
| Remarks: ­ | Pass |

# CHAPTER-9

# Output Screens

## 9.1 Main Page

**9.2 Login page**

**User page**

**9.3 Dataset values**

# CONCLUSION

In this project, we have proposed an application of AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING in UPI Transaction fraud detection. The different steps in UPI Transaction transaction processing are represented as the underlying stochastic process of an AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING. We have used the ranges of transaction amount as the observation symbols, whereas the types of item have been considered to be states of the AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING. We have 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 AUTO ENCODER, LOCAL OUTLIER FACTOR, KMEANS CLUSTERING can detect whether an incoming transaction is fraudulent or not. Experimental results show the performance and effectiveness of our system and demonstrate the usefulness of learning the spending profile of the cardholders. Comparative studies reveal that the Accuracy of the system is close to 80 percent over a wide variation in the input data. The system is also scalable for handling large volumes of transactions.

Don’t allow your business to fall victim to fraudulent UPI Transaction charges. Our fully customizable Fraud Detection Suite (FDS) helps you to find and eliminate harmful transactions. When suspicious activity is detected, you have the control over whether to approve, decline or hold it for further review. The E-Complish Fraud Detection Suite easily integrates with other solutions for check and UPI Transaction payments using the VirtualPay system, our all-in-one online payments console.

**FUTURE SCOPE :-**

* Detailed reporting on suspicious activity
* Option to hold questionable transactions for review
* Ability to block transactions from historically fraudulent IPs
* Robust searching feature allows you to quickly locate transactions
* Choose from pre-drafted customer responses, or customize your own
* Receive instant notification of questionable activity directly to your inbox
* Customizable filters for easy identification of fraudulent charges, including but not limited to:
  + Set minimum and maximum amounts
  + Monitor amount of charges applied to a card per hour
  + Scan for address discrepancies
  + Locate abnormal activity from a single IP address, and block IPs known to be fraudulent

## BENEFITS:-

* Decreased fees and other costs associated with UPI Transaction fraud
* Increased confidence in legitimacy of transactions
* Settings tailored to your business necessities
* Advanced technology to take your UPI Transaction security to the next level
* We based intuitive design is quick to learn and easy to use
* Seamlessly integrates with other E-Complish solutions using the online payments system.

.

.

# REFERENCES

[1] “Global Consumer Attitude Towards On-Line Shopping,”

<http://www2.acnielsen.com/reports/documents/2005_cc_online> shopping.pdf, Mar. 2007.

[2] D.J. Hand, G. Blunt, M.G. Kelly, and N.M. Adams, “Data Mining for Fun and Profit,” Statistical Science, vol. 15, no. 2, pp. 111-131, 2000.

[3] “Statistics for General and On-Line Card Fraud,” <http://www>. epaynews.com/statistics/fraud.html, Mar. 2007.

[4] S. Ghosh and D.L. Reilly, “UPI Transaction Fraud Detection with a Neural-Network,” Proc. 27th Hawaii Int’l Conf. System Sciences: Information Systems: Decision Support and Knowledge-Based Systems, vol. 3, pp. 621-630, 1994.

[5] M. Syeda, Y.Q. Zhang, and Y. Pan, “Parallel Granular Networks for Fast UPI Transaction Fraud Detection,” Proc. IEEE Int’l Conf. Fuzzy Systems, pp. 572-577, 2002.

[6] S.J. Stolfo, D.W. Fan, W. Lee, A.L. Prodromidis, and P.K. Chan, “UPI Transaction Fraud Detection Using Meta-Learning: Issues and Initial Results,” Proc. AAAI Workshop AI Methods in Fraud and Risk Management, pp. 83-90, 1997.

[7] S.J. Stolfo, D.W. Fan, W. Lee, A. Prodromidis, and P.K. Chan, “Cost-Based Modeling for Fraud and Intrusion Detection: Results from the JAM Project,” Proc. DARPA Information Survivability Conf. and Exposition, vol. 2, pp. 130-144, 2000.

[8] E. Aleskerov, B. Freisleben, and B. Rao, “CARDWATCH: A Neural Network Based Database Mining System for UPI Transaction Fraud Detection,” Proc. IEEE/IAFE: Computational Intelligence for Financial Eng., pp. 220-226, 1997.

[9] M.J. Kim and T.S. Kim, “A Neural Classifier with Fraud Density Map for Effective UPI Transaction Fraud Detection,” Proc. Int’l Conf. Intelligent Data Eng. and Automated Learning, pp. 378-383, 2002.

[10] C. Chiu and C. Tsai, “A Web Services-Based Collaborative Scheme for UPI Transaction Fraud Detection,” Proc. IEEE Int’l Conf. e-Technology, e-Commerce and e-Service, pp. 177-181, 2004.

[11] V. Vatsa, S. Sural, and A.K. Majumdar, “A Game-theoretic Approach to UPI Transaction Fraud Detection,” Proc. First Int’l Conf. Information Systems Security, pp. 263-276, 2005.

[12] S.S. Joshi and V.V. Phoha, “Investigating Hidden Markov Models Capabilities in Anomaly Detection,” Proc. 43rd ACM Ann. Southeast Regional Conf., vol. 1, pp. 98-103, 2005.

[13] S.B. Cho and H.J. Park, “Efficient Anomaly Detection by Modeling Privilege Flows Using Hidden Markov Model,” Computer and Security, vol. 22, no. 1, pp. 45-55, 2003.

[14] D. Ourston, S. Matzner, W. Stump, and B. Hopkins, “Applications of Hidden Markov Models to Detecting Multi-Stage Network Attacks,” Proc. 36th Ann. Hawaii Int’l Conf. System Sciences, vol. 9, pp. 334-344, 2003.

[15] T. Lane, “Hidden Markov Models for Human/Computer Interface Modeling,” Proc. Int’l Joint Conf. Aicial Intelligence, Workshop Learning about Users, pp. 35-44, 1999.