CREDIT CARD FRAUD DETECTION WITH FORMULA BASED AUTHENTICATION

Submitted in partial fulfillment of the requirements for the award of

Bachelor of engineering degree in Computer Science and Engineering

By

CHAVVA NIKHITA (Reg. No. 38110102)
BHAVYA BABU (Reg. No. 38110080)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL-2022

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

(Established under Section 3 of UGC Act, 1956)
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119 www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **CHAVVA NIKHITA** (Reg.no.38110102) and **BHAVYA BABU** (Reg.no.38110080) who carried out the project entitled "**CREDIT CARD FRAUD DETECTION WITH FORMULA BASED AUTHENTICATION**" under my supervision from November 2021 to April 2022.

Internal Guide

Dr. ASHA. P, M.E.,

PH.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

I **CHAVVA NIKHITA (Reg. No. 38110080)** and **BHAVYA BABU (Reg. No. 38110080)** hereby declare the Project Report entitled "**CREDIT CARD FRAUD DETECTION WITH FORMULA BASED AUTHENTICATION"** done by me under the guidance of **Dr. ASHA. P, M.E., PH.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E.,Ph.D., Dean,** School of Computing, **Dr.S.Vigneshwari M.E.,Ph.D.** and **Dr.L.Lakshmanan M.E.,Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **Dr. ASHA. P, M.E., PH.D.,** for her valuable guidance, suggestions and constant encouragement paved the way for the successful completion of my project.

I wish to express my thanks to all teaching and non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
[DEEMED TO BE UNIVERSITY]
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

National Conference on Computational Intelligence and Communication Networks

Springer CCIS

N C C I C N

24TH – 25TH MARCH 2022

NCCICN

Certificate of Presentation

This is to certify that ***Dr./Mr./Ms. CHAVVA NIKHITA,*** *of* ***Sathyabama Institute of Science and Technology,*** *has presented a paper entitled* ***"Credit Card Fraud Detection with Formula based Authentication",*** *in the National Conference on Computational Intelligence and Communication Networks (NCCICN 2022).*

Dr. T. Sasikala
Conference Chair, Professor & Dean
School of Computing

Dr. L. Lakshmanan
Convener
Professor & Head, CSE

Dr. S. Vigneshwari
Convener
Professor & Head, CSE

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
[DEEMED TO BE UNIVERSITY]
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

National Conference on Computational Intelligence and Communication Networks

Springer CCIS

N C C I C N

24TH – 25TH MARCH 2022

NCCICN

Certificate of Presentation

This is to certify that ***Dr./Mr./Ms. BHAVYA BABU,*** *of* ***Sathyabama Institute of Science and Technology,*** *has presented a paper entitled* ***"Credit Card Fraud Detection with Formula based Authentication",*** *in the National Conference on Computational Intelligence and Communication Networks (NCCICN 2022).*

Dr. T. Sasikala
Conference Chair, Professor & Dean
School of Computing

Dr. L. Lakshmanan
Convener
Professor & Head, CSE

Dr. S. Vigneshwari
Convener
Professor & Head, CSE

ABSTRACT

Credit cards have become an important part of digital transactions. Days have come where people don’t have to carry cash in their pockets and just a small card is enough to make all the transactions. The problem with credit cards is that the password can be hacked and can easily lead to fraud. In this project, credit card fraud can be detected using Hidden Markov Model (HMM) and formula based authentication. In the Existing system, credit card transactions have become commonplace today and so are the frauds associated with it. In the proposed system, machine learning supervised and unsupervised algorithms have been applied to detect master card deception in an imbalanced dataset. In the modification process, an application is developed for a banking sector particularly for a credit or ATM card. Users can create an account and get the ATM or credit card along with a unique formula which should be used during suspicious transactions. The user behavior of every transaction is tracked by Hidden Markov Model and if there are any occurrences of suspicious transactions, then a message is sent to the user with the keys that are required to complete the formula. After the user applies the keys to the formula the solution must be entered as the password in order to complete the transaction successfully.

TABLE OF CONTENTS

	ABSTRACT	**vii**
	LIST OF FIGURES	**x**
	LIST OF ABBREVIATIONS	**xi**
CHAPTER NO	**TITLE**	**PAGE NO**
1	**INTRODUCTION**	1
	1.1 OBJECTIVE	1
	1.2 OUTLINE OF THE PROJECT	1
	1.3 ARCHITECTURE OF CREDIT CARD DETECTION AND AUTHENTICATION	2
	1.4 FLOW CHARTS	4
	1.5 APPLICATION OF CREDIT CARDS	5
2	**AIM AND SCOPE OF PROJECT**	7
	2.1 AIM OF THE PROJECT	7
	2.2 SCOPE OF THE PROJECT	7
3	**LITERATURE SURVEY OF PROJECT**	8
	3.1 LITERATURE SURVEY	8
	3.2 INFERENCES FROM LITERATURE	14
4	**EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED**	15
	4.1 SOFTWARE REQUIREMENTS	15
	4.2 HARDWARE REQUIREMENTS	15
	4.3 JAVA	15
	4.4 APPLICATION OF JAVA	16
	4.5 FEATURES OF JAVA	16
	4.6 JDK	19
	4.6.1 JDK AND JAVA COMPILER	20
	4.6.2 JDK PACKAGES	20

	4.6.3 JDK VERSIONS COMPATIBILITY	20
	4.7 TOOLS FOR INTERFACING WITH OTHER LANGUAGES	20
	4.8 ALGORITHMS STEPS	22
5	**PROJECT DESCRIPTION**	19
	5.1 EXISTING SYSTEM	19
	5.2 EXISTING SYSTEM DISADVANTAGES	19
	5.3 PROPOSED WORK	19
	5.4 ADVANTAGES OF PROPOSED WORK	20
	5.5 MODULES	20
	5.5.1 USER REGISTRATION	20
	5.5.2 BANK SERVER	20
	5.5.3 HMM MODEL	21
	5.5.4 FORMULA BASED AUTHENTICATION	21
	5.6 HIDDEN MARKOV MODEL	21
	5.7 NUMERIC NOTATION	22
	5.8 RESULTS	
6	**CONCLUSION AND FUTURE ENHANCEMENTS**	26
	6.1 CONCLUSION	26
	6.2 FUTURE ENHANCEMENTS	26
	REFERENCES	
	APPENDIX	37
	A. SOURCE CODE	37
	B. PLAGARISM REPORT	43

LIST OF FIGURES

FIGURE NO	NAME	PAGE NO
1.1	System Architecture of Credit Card Fraud Detection System	2
1.2	Flowchart representing the initial steps of the model	4
1.3	Flowchart representing the Transaction process	5
6.1	User Login Page	23
6.2	User Registration Page	23
6.3	User Amount Transaction Page	24
6.4	User transaction page with successful transaction	24
6.5	User transaction page, with unsuccessful transaction	25
6.6	Transaction History of the User	25

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
HMM	Hidden Markov Model
FDS	Fraud Detection System
FPS	Fraud Prevention Systems
OTP	One Time Password
EMI	Equated Monthly Installment
OOP	Object Oriented Programming
API	Application Programming Interface
JVM	Java Virtual Machine
JRE	Java Runtime Environment
RMI	Remote Method Innovation
EJB	Jakarta Enterprise Beans
JSE	Java Standard Edition
JEE	Java Enterprise Edition
JDK	Java Development Kit
JME	Java Mobile Edition

CHAPTER 1

INTRODUCTION

Credit cards are being used for digital transactions as a payment method by both online and offline buyers in a huge way. On the other hand, this method has few drawbacks. Criminals, hackers and perpetrators have started targeting credit card based transactions. For any transaction, only the card information has to be entered and the card need not be present physically. In most cases, a One Time-Password (OTP) authentication is used as an extra safety factor. Specifically for international transactions, a method called Card-Not-Present, is used where only the card details are required rather than the physical card for unauthorized purchases. It is very easy to get the card details using methods like shoulder surfing, buying card information, credit card stealing and web traffic sniffing.

The main victims of the credit card frauds are the card holder, the bank, and the merchant. One of the main duties of the credit card holder is to detect any suspicious activities and report fraudulent transactions to the issuing bank. The bank then takes the responsibility to investigate the problem and if any evidence for fraudulent activities are found, then the credit card transaction is reversed.

1.1 OBJECTIVE

The main objective of this project is to detect credit card frauds and authenticate using formula based authentication.

1.2 OUTLINE OF THE PROJECT

Credit card information can be fetched easily through various modernized techniques. Even though many credit card methods are emerging today, so is the fraud associated with it. Digital transactions do not require credit cards to be present physically instead authentication such as OTP methods are used. Even though there are many secured methods of transactions fraudulent activities occur frequently.

It is impossible to find out whether the credit card transaction is genuine or fraudulent. As a result, credit card fraud detection becomes more important to verify the authenticity of the transaction. To overcome this problem, user behavior is monitored using HMM model and formula based authentication is applied for security. In this process the user will be issued with a formula during credit card registration. Every transaction will be monitored by the HMM model and if any suspicious transaction is detected, the authentication key is sent to the user. The user must apply the formula with generated keys to find out and enter the correct solution as password in order to complete the transaction successfully.

1.3 ARCHITECTURE OF THE CREDIT CARD DETECTION AND AUTHENTICATION

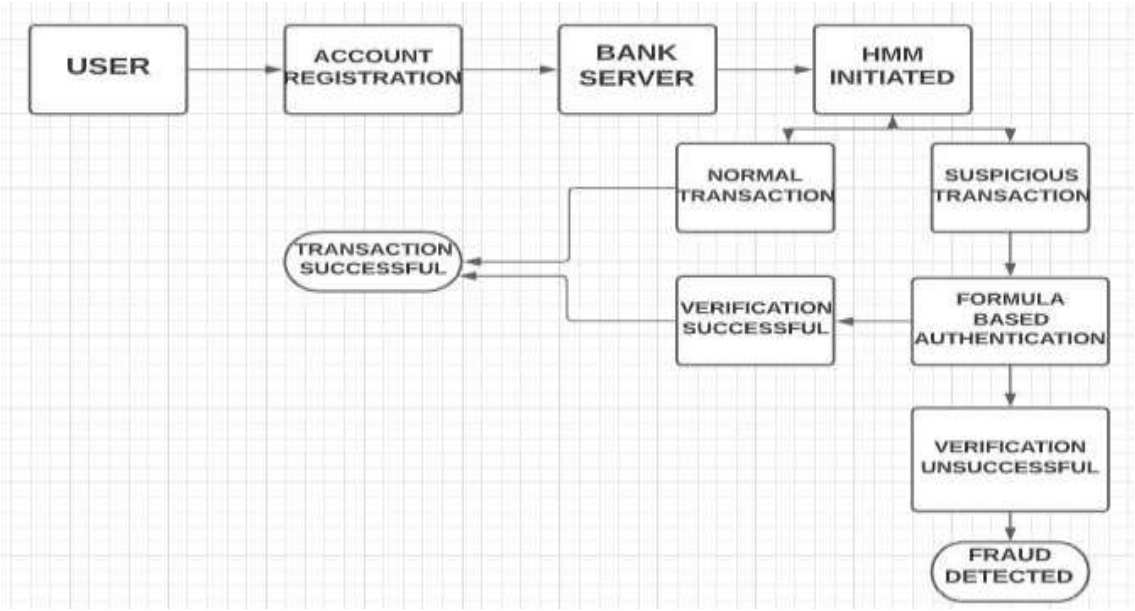


FIG 1.1 System Architecture of Credit card fraud detection system

SYSTEM ARCHITECTURE MODULES:

1. ***USER*** - User is an authorized person allowed to use a credit card. User is issued a credit card at the time of registration in a bank, along with the unique formula. Each of the transaction of the user will be tracked and monitored by the HMM model. Users will also follow the principle of formula based transaction.

2. ***INTEGRATING FORMULA BASED AUTHENTICATION*** - Formula Based Authentication is the main system used by this bank. This type of authentication allows the user to securely transact the money from the bank without having to worry about the fraudsters. This method of security prevents shoulder surfing or web tracking done by the fraudsters to get hold of the user's card information. In this verification process, the user is issued a formula at the time of registration and will use it along with a randomized key generated at the time of each suspicious transaction monitored by the HMM model.

3. ***REGISTRATION OF FORMULA*** - every user at the time of registration is given a unique formula, which is only known to the user. At a time of any suspicious transaction detected by the HMM model a randomized key for this formula will be sent to the user, which can be applied in the formula to get the solution. Once the password is applied, the transaction will be successful. This process is also called Formula Based Authentication.

4. ***SMART CARD*** - Smart card, also known as an interested circuit or chip card, is an authorization device which is used to control access to a resource since it is electronic. Smart card is basically a type or credit card embedded with an interesting circuit chip. Smart cards offer more security and confidentiality for the user. They use encryption techniques which prevent the tracking of information.

5. ***APPLICATION OF CREDIT CARD*** - Each transaction made by the user will be tracked and monitored by the HMM model. For suppose if the user transacts a certain amount of money each month, and suddenly extracts a huge sum of money, the HMM model monitoring the transaction pattern will raise a suspicion and send the user a unique key for the formula. Once the user applies the key to the formula, and answers with the correct solution, the transaction will be successful. In this way, the HMM model as well as the Formula Based Authentication will be demonstrated.

1.4 FLOW CHARTS:

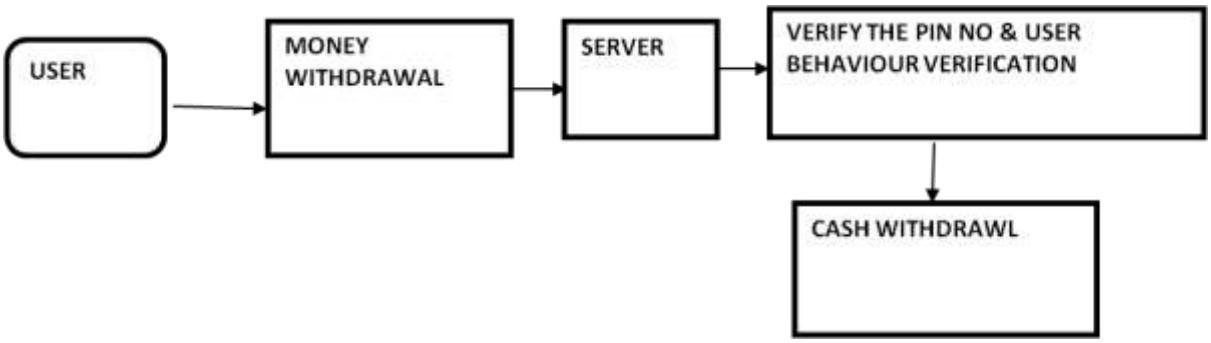


FIG 1.2 Flowchart representing the initial steps of the model

User makes a transaction in the bank. The HMM model monitors the user behavior and tracks every transaction. If the HMM model detects a suspicious activity, it immediately sends a warning to the user. This initiates the formula based authentication. The user will receive a set of keys which is applied to the formula. The solution is entered, and if it's correct, the transaction will be successful.

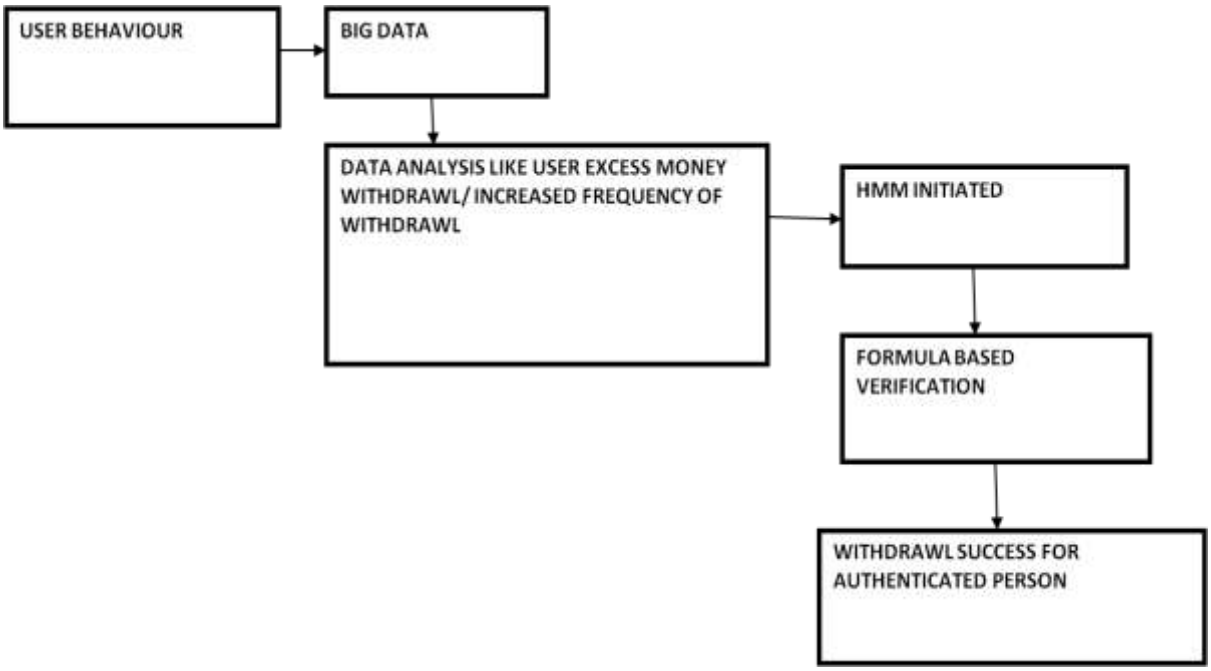


FIG 1.3 Flowchart representing the transaction process

Each transaction done by the user will be tracked, monitored by the HMM model and all these will be stored in the database. If there is any excess withdrawal or increased frequency of withdrawal, the HMM model will send a warning through formula based

authentication. The keys received by the user will be applied to the formula to receive the password. Once the correct solution is entered, the transaction will be successful.

1.5 APPLICATIONS OF CREDIT CARD

Credit cards have a broad spectrum of applications, such as shopping, dining out, travel, payment of bills, home furnishing, cab rides, movie ticket booking.

1. SHOPPING

Nowadays in markets, grocery stores, shopping malls credit card payment methods are readily available. There are rarely any shops that do not accept card payment. This type of payment can be included for items ranging from groceries to clothes. Days have come where liquid cash no longer exists and it is overtaken digitalization. A method called Auto-debit feature also gives the user the access to unlimited entertainment.

2. DINING OUT

Extinction of liquid cash has led several restaurants to accept credit card payments.

3. TRAVEL

Credit cards ensure a great deal of secure travel because liquid cash is not being used by the travelers. From booking tickets to hotel stay credit cards can be used for payment methods.

4. PAYMENT OF BILLS

Utility bills such as water, electricity, phone bills can be paid by making use of a credit card online. Since these utility bills are a necessity in our daily life the credit card method ensures that the pending bill amount is deducted automatically provided that the user has given the access and hence the user will not skip a payment.

CHAPTER 2
AIM AND SCOPE OF THE PROJECT

2.1 AIM

The main aim of the project is to secure the account by adding formula based authentication along with the user behavior being monitored by using the HMM model. User behavior is monitored based on frequency of withdrawal and the amount of money withdrawn by the user.

2.2 SCOPE

The major benefit of the project is that it cannot be easily manipulated. The formula key which is generated during the transaction cannot be manipulated because it is random every time which will lead to high authentication. Even if the key is hacked by the fraudster, they will not be able to complete the transaction due to the unique formula which is only known to the user.

In the Existing system, card withdrawals are very routine among the people and the frauds corresponding to the improvement of security are increasing. In the proposed system, ML algorithms have been applied to detect master card deception in a disproportionate dataset. In the modification process, an application is developed for a banking sector particularly for a credit or ATM card. Users can create an account and get the ATM card along with a unique formula which should be used during suspicious transactions. The user behavior of every transaction is tracked by Hidden Markov Model and if there are any occurrences of suspicious transactions, then a message is sent to the user with the keys that are required to complete the formula. After the user applies the keys to the formula the solution must be entered as the password in order to complete the transaction

CHAPTER 3
LITERATURE SURVEY

3.1 LITERATURE SURVEY

3.1.1 Andrea Dal Pozzolo, Giacomo Boracchi, Olivier Caelen, Cesare Alippi, and Gianluca Bontempi, "Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy" (2017)

A novel learning strategy using real world data containing over 75 million transactions which were authorized over a time period of three years explains how to overcome a number of challenges which includes verification latency that is few transactions are checked by investigators from time to time, class imbalance that specifies more frauds than genuine transactions and concept drift which states that the fraudsters strategies change time to time as customers evolve their habits. Many proposed machine learning algorithms depend upon assumptions due to which there is a lack of realism that leads to two major concerns: the way and timing in which the supervised data is fetched and the measures used to detect fraud-detection performance. A formalization of the fraud detection problem is proposed with the help of an industrial partner which describes the Fraud Detection System's (FDS) operations that analyze massive streams of transactions and also illustrates performance measures used for fraud detection purposes. The pros include addressing drift and verification latency and class imbalance and implementing alert feedback interaction meanwhile, the alert interactions were less precise.

PROS: Addressed class imbalance, concept drift and verification latency, and alert feedback interaction.

CONS: Less precise alerts.

3.1.2 John O. Awoyemi, Adebayo O. Adetunmbi, Samuel A. Oluwadare,"Credit card fraud detection using Machine Learning Techniques" (2017)

It explains the performance of various machine learning techniques like Naive Bayes, k-nearest neighbor and logistic regression on highly skewed dataset. The major challenges of credit card fraud detection are that the fraud transaction datasets are highly skewed and the original as well as the fraudulent behavior change constantly. The sampling approach on dataset, variables selection and detection techniques has shown great effect in the performance of fraud detection. The European cardholder's transaction dataset is considered and under-sampling and oversampling hybrid techniques are carried on the skewed data which consists of 284,407 transactions. The performance of the techniques is evaluated based on accuracy, sensitivity, specificity, precision, Matthews correlation coefficient and balanced classification rate and is implemented in Python. The accuracy for k-nearest neighbor and Naive Bayes were 97.69% and 97.92% while the logistic regression classifiers was 54.86% resulting logistic regression being a less accurate model compared to the other two models. More parameter tuning is required in order to improve the accuracy.

PROS: Highly skewed datasets are used.

CONS: More parameter tuning can be done to improve accuracy.

3.1.3 Lutao zheng, Guanjun Liu, Wenjing Luan, Zhengchuan Li, Yuwei Zhang, Chungang Yan, Changjun Jiang), "A New Credit Card Fraud Detecting Method Based on Behavior Certificate"(2018)

It explains the correlation between behavior features and special cases considered based on behavior certificate. Initially, the behavior features are extracted from the card holders historical transaction record patterns as it is an important way to detect fraud. Based on the behavior features, a behavior certificate is constructed. Using a behavior certificate the risk degree for each cardholder's incoming transaction is calculated and if the degree is more than the threshold, it is considered as fraud. In this paper to reflect the card holders transaction habits a new credit card fraud detection system (FDS) based on behavior certificate is introduced. The major asset in this paper is that it is highly effective while performing with simulated data, whereas it is difficult to learn the characteristics of fraudulent transactions.

PROS: Highly effective when performed with simulated data.

CONS: It is difficult to learn the characteristics of fraudulent transactions.

3.1.4 Abhimanyu Roy, Jingyi Sun, Robert Mahoney, Loreto Alonzi, Stephen Adams, Peter Beling, "Deep Learning Detecting Fraud in Credit Card Transactions"(2018)

It studies about a deep learning approach which provided comparable results to prevailing fraud detection methods and provided a guide to sensitivity analysis of model parameters in consideration with accuracy of fraud detection. The preceding customer data as well as the present transaction details are recorded and are used for deep learning. A number of topologies including Gradient Boosted Trees and Logistic Regression were encompassed in deep learning as well as model construction using various parameters which influenced its results. This paper evaluates topologies ranging from general artificial neural networks to topologies with built-in time and long short term memory components. From pre-labeled as fraudulent and legitimate 80 million credit card transactions, different parameters were identified to improve their effectiveness in fraud detection. Class imbalance and scalability which were the common CCF detection problems were suppressed using a high performance distributed cloud computing environment. To reduce losses by preventing fraudulent activity, a framework for parameter tuning of topologies is presented. The major benefits include a high performance computing environment and better performance than traditional algorithms whereas, the general validation performance decreases as data size is increased.

PROS: High performance computing environment that has better performance than traditional algorithms.

CONS: General validation performance decreases as data size is increased.

3.1.5 Sahil Dhankhad, Emad A. Mohammed, Behrouz Far, "Supervised Machine Learning Algorithms for Credit Card Fraudulent" (2018)

A comparative study, explains about the hidden patterns to detect the fraudulent transactions using real world dataset. The most important variables that lead to fraud transactions are identified and are used to achieve higher accuracy. The advancement in modernized technology has escalated the fraudulent activities. In order to delineate the hidden patterns the supervised machine learning algorithms were applied. Publicly available datasets were used which were highly imbalanced. The major advantages in this paper includes using various supervised machine learning algorithms to implement a super classifier using ensemble learning methods whereas, the drawback includes delay in updating the supervised model due to verification latency.

PROS: Implements super classifier using ensemble learning methods.

CONS: Delay in updating the supervised model due to verification latency.

3.1.6 Aisha Abdallah n, Mohd Aizaini Maarof, Anazida Zainal, "Survey on systematic and comprehensive overview" (2016)

This explains the collaboration of FDSs with FPSs to protect electronic commerce systems from fraudsters. FDS and FPS individually are insufficient to provide the security needed for electronic commerce systems. This paper provides the solution for the issues like large amounts of data. Real time detection, concept drift as well as skewed distribution hinders the performance of FDS and provides a systematic and comprehensive overview. The five electronic commerce systems selected for this paper are automobile insurance, credit card, online auction, telecommunication and healthcare insurance in which the prevalent fraud types are examined closely. Further, the state-of-the-art of the FDS is also systematically introduced. The major benefit is combining FDS and FPS for fraud detection whereas, this combination leads to an increase in the false prediction rate of legitimate transactions and increases investigation cost for banks.

PROS: Collaborated fraud detection and fraud prevention systems.

CONS: Leads to an increase in the false prediction rate of legitimate transactions. It also increases investigation cost for banks.

3.2 INFERENCES FROM THE LITERATURE

The major challenges were concept drift, class imbalance and verification latency. Verification latency that is few transactions are checked by investigators from time to time, class imbalance that specifies more frauds than genuine transactions and concept drift which states that the fraudsters' strategies change from time to time as customers evolve their habits. There is a lack of realism in the dataset and it was less precise with the alert interactions. As the data size increased the validation performance decreased. It is observed that there is a delay in updating the supervised model due to the verification latency. In some cases, there was a false prediction rate of legitimate transactions. Parameter tuning can be done to improve accuracy. The other major drawback was that the customers' habits evolve and the fraudsters change their strategies accordingly. There are genuine transactions along with far outnumbered frauds. Alert systems must be precise and should overcome the outliers.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS, ALGORITHMS USED

4.1 SOFTWARE REQUIREMENTS

Platform: Windows 7/8
Front End: Java-JDK1.7
Back End: MYSQL
Big Data: Apache Hadoop -2.3.1

4.2 HARDWARE REQUIREMENTS

The hardware requirements are needed to serve as the basis for implementation of the system and hence should be an absolute and coherent specification of the entire system. The software engineers use the hardware requirements as the starting point for the system design. It indicates what the system performs and what the system should execute.

PROCESSOR: Core i3/i5/i7

RAM: 2-4GB

HDD: 500 GB

4.3 JAVA

Java is a high-level programming, general-purpose, class-based, object-oriented language designed with lesser implementation dependencies. Java is a fast, secure, reliable computing platform for application development. Initially the major cause for developing java was for handheld devices, and set-top boxes but later it became popular for creating web applications, android applications and used for internet of things.

4.4 APPLICATIONS OF JAVA

The major applications of java include:

- Desktop applications
- Database connection
- Web servers and application servers
- Web applications
- Games
- Mobile applications

4.5 FEATURES OF JAVA

The important features of java are:

- SIMPLE

Java is very straightforward to understand as its syntax is not complicated, clean and easy to comprehend. According to Sun Micro system, Java is a user friendly programming language because java and C++ are relatively similar, many complex features have been eliminated while creating java, since a feature called automatic garbage collection is present in Java, no need to remove unreferenced objects.

- OBJECT ORIENTED

Java works with objects. Object oriented includes organizing the software as a combination of various types of objects that consist of data and behavior.

Java uses Object-oriented programming (OOPs) methodology that maintains using a set of rules and simplifies software development. The major concepts include encapsulation, abstraction, polymorphism, class, inheritance, abstraction.

- PLATFORM INDEPENDENT

Unlike other languages like C, C++, Java is platform independent as it can be run on multiple platforms. It is also called "code once, run anywhere." A program runs on hardware or software called a platform.

There are Software based and Hardware based platform environments in which Java provides Software based environments.

The Java platform is different from most other platforms. It is a software-based platform that runs on top of other hardware-based platforms. Java platform has two components:

1. Runtime Environment
2. API(Application Programming Interface)

Java can be run on multiple platforms. It can be compiled and converted as bytecode, which is a platform independent code.

- SECURED

Java can be used to develop various virus-free systems and is more secure because of no explicit pointer and virtual machine sandboxes are used to run the programs.

Class loader is used to load classes into Java Virtual Machine dynamically. This adds security due to the separation of classes from the local file system from those that are imported from network sources. Bytecode verification is used to check the code fragments for illegal code that violate access rights to objects. Security manages helps to determine the resources of a class which has access such as reading and writing to local disk.

- ROBUST

The meaning of Robust is strong. Java is robust because:

- Java applies well-built memory management techniques.
- Java consists of exception handling as well as type checking.

- By using Java Virtual Machine, the objects not used by the Java Application can be eliminated. Java also provides automatic garbage collection.
- Security problems can't be avoided easily due to lack of pointers.

- ARCHITECTURE NEUTRAL

No implementation dependent features make java architecture neutral. This leads to fixed size of primitive types. In java, 4 bytes are required for both 32 and 64-bit architectures whereas, in C programming, int data type occupies 2 bytes of memory for 32-bit architecture.

- PORTABLE

Java is portable as it facilitates the user to carry the Java bytecode to any platform. Java doesn't require any implementation.

- HIGH PERFORMANCE

Java is faster compared to traditional interpreted languages. It is because the byte code is “close” to native code. This is a bit slower than compiled languages like C, C++, etc.

- DISTRIBUTED

Java is distributed as it allows users to create distributed applications in Java. RMI and EJB are implemented for creating distributed applications. This feature of Java allows the users to access files by calling the methods from any machine on the internet.

- MULTI THREADED

Thread executes concurrently like a separate program. Multiple threads are used to deal with many tasks at the same time. It does not occupy memory for each thread as they share the common memory area. Threads are important for web applications, multi-media etc.

- DYNAMIC

Java is also known as a dynamic language as it supports the dynamic loading of classes. Dynamic loading means that on demand, classes are loaded. The functions from its native languages are also supported. Finally, automatic memory management as well as dynamic compilation is also supported.

4.6 JAVA DEVELOPMENT KIT

The Java Development Kit (JDK) combined with Java Virtual Machine (JVM) and Java Runtime Environment (JRE) is one of the main core technologies implemented in Java.

The difference between the three technologies is:

- The programs are executed by the Java platform component called JVM.
- JVM is created on the disk part of JRE.
- JVM and JRE executes and runs the Java programs that are created by developers given permission by JDK.

Often, Java Development Kit and Java Runtime Environment are confused by the new developers. The major difference between JDK and JRE is that JRE is a package of tools that is used to run the Java Code while JDK is a package of tools used to create the Java based software.

Since JRE is a part of JDK, it can also be implemented as a standalone component to execute and run the Java code. Developing the JDK involves JRE running the Java programs.

Below is the technical and everyday definition of JDK:

- Technical Definition: JDK is an implementation of platform specification which includes the compiler as well as class libraries.
- Daily definition: To create Java based applications, JDK is required as it is a software package.

4.6.1 THE JDK AND THE JAVA COMPILER:

Along with JRE, the environment which is implemented to execute the Java code and its applications, all JDK contains a Java compiler. The compiler is able to take the raw .Java files containing plain text and render them into executable class files.

4.6.2 JDK PACKAGES

A Java package is also needed along with the Java version. For different types of development, separate Java Development Kits called Packages are selected. Java Mobile Edition (Java ME), Java Enterprise Edition (Java EE) as well as Java Standard Edition (Java SE) are the available packages.

Beginners sometimes will be confused on choosing the package for their project. Each JDK contains Java SE. The standard version is available along with Java EE and Java ME download.

4.6.3 JDK VERSION COMPATIBILITY:

The JDK is the one which determines the type of Java Version that the user is able to code, as it is the supplier of compilers for all Java programs.

4.7 TOOLS FOR INTERFACING WITH OTHER LANGUAGES:

C LANGUAGE

INLINE-C

Inline-C applies Quasi Quotation to call the C libraries and embed the powerful Inline-C code into Haskell modules.

C-HASKELL

C-Haskell is a lightweight tool to implement access from Haskell to C Libraries.

C2HSC

For C libraries, Bindings-DSL based interface documents are created by using C2HSC.

HSFFIG

By using the Haskell FFI Binding Mosulod Generation tool, Haskell Foreign Function Interval import declarations for items are induced by the C library include file (.h).

KDIRECT

KDirect is less dominant than Haskell Direct. It is uncomplicated and more portable. KDirect is a tool to clarify the process for uniting C libraries to Haskell.

LIBFFI

From Haskell, C functions of types known at runtime with binding to LIBFFI.

JAVA

INLINE-JAVA Any JVM function is called from Haskell.

JAVA-BRIDGE a high level interface generator as well as Java Native Interface is banded by Java-Bridge.

PYTHON

CPYTHON Python is called by Haskell code, as it is given permission by Bindings.

MISSINGPY: MissingPy provides support for translating Python code by interfacing with Python or C API and handling Python objects. It is allocated to call Python from Haskell. The tools between Python objects and its Haskell equivalents are changed by using tools provided by MissingPy. Python exceptions get mapped to Haskell Dynamic Exceptions. MissingPy contains Haskell interfaces to Python Modules.

4.8 ALGORITHM STEPS:

Step 1: The client registers in a bank with new account details.

Step 2: Clients will be issued with a unique formula that will be used during a suspicious transaction detected by the HMM.

Step 3: In a certain predicament, if the client tries to take an amount that is out of the typical observable patterns, the HMM will immediately raise uncertainty.

Step 4: Formula Based Authentication will be initiated through the HMM.

Step 5: The client will receive a set of keys which are unrepeatable to his formula.

Step 6: The client will apply these keys to the formula, and once the correct password is given, the transaction will be fulfilled.

HMM provides instructions about evaluation, decrypting, and learning. Evaluation is defined as expecting the monitoring order further as well as reverse design. Decrypting specifies all unknown states order (Viterbi). Using the observed information, HMM will be created using Learning (Baum-Welch).

The HMM model works on the transaction history of the client, and after the required formulation if the current transaction is found to be suspicious, then the formula based authentication is initiated. If the user is able to authorize using the keys to their formula, then the transaction will be successful.

CHAPTER 5

PROJECT DESCRIPTION

5.1 EXISTING SYSTEM:

In the existing system whenever the user tries to make a transaction, the user receives an OTP (One Time Password), which must be entered to make successful transactions. But, if the OTP is seen by the fraudster by shoulder surfing, web trafficking or by any other means then the fraudster will be able to make transactions from the user's account which will lead to credit card fraud.

5.2 EXISTING MODEL DISADVANTAGES:

PIN, credit card account number, security code and OTP can be shoulder surfed by the fraudster and transactions can be made by them.

By using a misappropriated or stolen master card many unauthorized purchases can be made by the fraud.

A fraudster can trick the user into providing the user's card number by visiting a fraudulent website by which the fraudster may get the credit card information and make purchases.

A tiny device called a credit card skimmer is used in ATM or fuel stations which can be installed anywhere by the fraudster to get your account information making it an effective way for them to steal.

5.3 IMPLEMENTATION / PROPOSED WORK:

This application is developed for a banking sector particularly for a Credit or ATM card. Users can create an account and get the credit card along with a unique formula from the bank. User behavior is monitored using HMM model based on the user's money withdrawal sequence, which means the first condition is that every month the user will be able to withdraw a limited amount and the second condition is

that frequency of money withdrawal is monitored when using credit cards. Users can withdraw the cash as per limited money requirement. Time frequency is also monitored and recorded. It is very useful to withdraw an amount without a time delay. The user can withdraw cash from their account after successful authentication of the formula key password.

5.4 ADVANTAGES OF PROPOSED WORK:

- Hidden Markov model is used for user behavior analysis of cash withdrawal.
- Security is ensured by the implementation of formula based authentication.
- Big data is included in the system for analyzing huge volumes of data.
- Even after the fraudster gets to know the keys or the pin, the fraudster will not be able to make any transactions, as the fraudster is not aware of the unique formula.
- Due to the randomization of the generated pin, this secured pin keeps changing for every suspicious transaction.

5.5 MODULES

5.5.1 USER REGISTRATION

User initially creates the account, and allows access to the network.

To request the Service Provider for the job, the user has to login.

The Service Provider will process and respond to the user requested job based on what the user has requested.

The Service Provider database will contain all the user details.

Using Java Programming, the User interface Frame will communicate with the Server through Network Coding.

The Service Provider gives the authentication to the user to access the requested data, as the user will send a request to the Service Provider.

5.5.2 BANK SERVER

The Bank Service Provider will have a data storage which contains the information about users.
They also maintain all the user information that helps the user to authenticate

whenever they wish to access the account.
The bank server will establish connection with the client and other modules of the Company server to communicate. Hence, a User Interface Frame is connected.

5.5.3 HMM MODEL

The Hidden Markov Model is used to analyze user behavior on every transaction.
It is executed to understand the user's money withdrawal sequence, meaning that the first condition is the total amount of withdrawal each month and the second being frequency of withdrawal. The time frequency is also monitored and recorded.

5.5.4 FORMULA BASED AUTHENTICATION

Formula Based Authentication provides security by adding a formula.

This formula is unique for every user, registered at the time of creating an account.
The keys to the formula changes every time, and the user is requested to submit the answer after substitution of the corresponding keys to the formula.
This usage of formula is required only when the user tries to withdraw beyond the permitted.

5.6 HIDDEN MARKOV MODEL:

The Hidden Markov Model (HMM) is used to model sequential data in a simple way.

The Hidden Markov Model (HMM) is defined as the basic Markov Model data which is hidden or unknown. The observational data is known whereas the information about the states remains unknown which means data is produced by a specific type of model whereas the background processes producing the data are unknown. It is one of the powerful modeling statistical tool used for speech, handwriting recognition etc. HMM provides information about evaluation, decoding, and learning. Evaluation is the probability of the observation sequence (forward and backward algorithm). Decoding specifies the most probable hidden states sequence from an observation sequence (Viterbi algorithm). Learning explains the way to create HMM models from observed data (Baum-Welch).

The basic parts of HMM model include state emission probability distribution, state transition probability distribution, transition to terminal state probability distribution (in most cases excluded from model because all probabilities equal to 1 in general use),transition from initial state to initial hidden state probability distribution,

observation symbols (or states),hidden states.

The HMM model is divided into two parts which includes a hidden part that consists of unobserved hidden states. These hidden states emit observation symbols which are used to indicate their presence.

5.7 NUMERIC NOTATION

Initially the user gets a unique formula while registering for the credit card. During normal transactions the user does not have to enter the authentication formula but if the HMM model detects any particular transactions which are out of the user behavior pattern that is, if the model detects the transaction as suspicious then the user receives the keys through messages. For example: Let the unique formula be A+B-C. User will receive a message with keys as shown when suspicious transaction is found by the HMM model:

A=3 B=2 C=1 D=5 E=6 F=9 G=10 H=5 I=8 J=0 K=7 L=2 M=1 N=7 O=5 P=11 Q=4

R=3 S=5 T=6 U=1 V=7 W=6 X=0 Y=1 Z=1(which will be generated randomly)

The user must apply the formula using the keys:

=A+B-C

= 3+2-1

= 4

Hence this must be entered as the password in order to complete the transaction. Every suspicious transaction generates keys randomly therefore, even if the fraudsters hack the mobile for keys they will not be able to get the money as the

formula remains confidential with the user. In this way, the user's account will be free from fraudulent activities.

5.8 RESULTS:

FIG 6.3.1 User Login Page

FIG 6.3.2 User Registration Page

FIG 6.3.3 User Amount Transaction Page

FIG 6.3.4: Process: User amount transaction page, with formula based authentication and transaction successful page (when successful).

FIG 6.3.5: User amount transaction page, with formula based authentication and fraud detected page (when failed)

seq	ccno	month	transamt	userid
1	430223415617	JAN	1000	1
2	430223415617	JAN	1000	1
3	430223415617	JAN	1000	1
4	430223415617	JAN	1000	3
5	430223415617	JAN	1500	3
6	430223415617	JAN	2000	3
7	430223415617	JAN	1000	2
8	430223415617	FEB	1000	2
9	430223415617	FEB	2500	2
10	430223415617	FEB	3000	2

FIG 6.3.6: Sample of transaction history of the user

CHAPTER 6
CONCLUSION AND FUTURE ENHANCEMENTS

6.1 CONCLUSION

More credit card forgery is happening at a huge level these days. Even after the existence of cyber security fraudulent transactions are still active. Formula based authentication is one of the processes to perform the transactions securely. This process allows only genuine transactions to take place. HMM model is a robust technique for user behavior pattern extraction and to detect a fraudulent transaction. HMM is used to get a set of unknown variables from known variables. Pattern changes in the transaction history of the account holder can be detected by using the HMM model. If the transaction is detected to be fraudulent an authentication key is sent to the user, and the user must apply the keys to the formula in order to find the solution and enter it as password for successful transaction.

6.2 FUTURE ENHANCEMENTS

Multiple Algorithms can be deployed for effective filtering. A toggle button can be used to enable formula based detection transactions (for ex: the user can choose to enable or disable the transaction limit). Counterfactual analysis can be deployed. Hardware implementation can also be interfaced using NFC (Near Field Communication) for easy transactions.

REFERENCES

[1] The importance of credit cards: https://budgeting.thenest.com/importancecredit- cards-29514.html

[2] The chargeback process in a credit card: https://chargebacks911.com/chargeback-process/

[3] Low and Slow Is How the Credit Card Fraudsters Roll: https://www.threatmetrix.com/digital-identity-blog/fraudprevention/low-and-slow-is-how-the-credit-card-fraudsters-roll/

[4] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi, ”Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 8, pp. 3784-3797, Aug. 2018.

[5] L. Zheng, G. Liu, C. Yan and C. Jiang, ”Transaction Fraud Detection Based on Total Order Relation and Behavior Diversity,” in IEEE Transactions on Computational Social Systems, vol. 5, no. 3, pp. 796-806, Sept. 2018.

[6] Vaishali. Article: Fraud Detection in Credit Card by Clustering Approach. International Journal of Computer Applications 98(3):29-32, July 2014.

[7] J. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, ”Credit card fraud detection using machine learning techniques: A comparative analysis,” 2017 International Conference on Computing Networking and Informatics (ICCNI), Lagos, 2017, pp. 1-9.

[8] L. Zheng et al., ”A new credit card fraud detecting method based on behavior certificate,” 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), Zhuhai, 2018, pp. 1-6.

[9] SurajPatil*, VarshaNemade, PiyushKumarSoni, Predictive Modelling for Credit Card Fraud Detection Using Data Analytics, International

[10] Conference on Computational Intelligence and Data Science (ICCIDS 2018)

[11] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams and P. Beling, ”Deep /learning detecting fraud in credit card transactions,” 2018 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, 2018, pp. 129-134.

A. IMPLEMENTATION/SOURCE CODE:

```
package com.nura.entity;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "cc_trans")
public class CCTransactions implements Serializable{

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long seq;
    private String ccno;
    private String userid;
    private String transamt;
    private String month;

    public CCTransactions(){

    }


    /**
     * @return the seq
     */
    public long getSeq() {
        return seq;
    }

    /**
     * @param seq the seq to set
     */
    public void setSeq(long seq) {
        this.seq = seq;
    }

    /**
     * @return the ccno
     */
    public String getCcno() {
```

```
        return ccno;
    }

    /**
     * @param ccno the ccno to set
     */
    public void setCcno(String ccno) {
        this.ccno = ccno;
    }

    /**
     * @return the userid
     */
    public String getUserid() {
        return userid;
    }

    /**
     * @param userid the userid to set
     */
    public void setUserid(String userid) {
        this.userid = userid;
    }

    /**
     * @return the transamt
     */
    public String getTransamt() {
        return transamt;
    }

    /**
     * @param transamt the transamt to set
     */
    public void setTransamt(String transamt) {
        this.transamt = transamt;

}

    /**
     * @return the month
     */
    public String getMonth() {
        return month;
    }

    /**
     * @param month the month to set
     */
    public void setMonth(String month) {
```

```
        this.month = month;
    }
}
import com.nura.dao.impl.CCDtlsDAOImpl;
import com.nura.dao.impl.CCTransDAOImpl;
import com.nura.dao.impl.UserDetailsDAOImpl;
import com.nura.entity.CCDtls;
import com.nura.entity.CCTransactions;
import com.nura.entity.UserDetails;
import com.nura.hadoop.HadoopAnalyzer;
import com.nura.mail.SendMail;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Random;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class TransactionFrame extends javax.swing.JFrame {

    private CCTransDAOImpl _ccTransDAOImpl = new CCTransDAOImpl();
    private CCDtlsDAOImpl _ccDtlsDAOImpl = new CCDtlsDAOImpl();
    private UserDetails _usrDtls;

    /**
     * Creates new form TransactionFrame
     */
    public TransactionFrame() {

    }

    public TransactionFrame(UserDetails _usrDtls) {
        FileInputStream fis = null;
        initComponents();
        this._usrDtls = _usrDtls;
        tf_userName.setText(_usrDtls.getUserName());
        tf_userName.setEditable(false);
        // fis = new FileInputStream(constants.Constants.RFID_READER_FILE);
        // Scanner scan = new Scanner(fis);
        tf_CCno.setText(_usrDtls.getCreditCardNo());
        tf_CCno.setEditable(false);
        tf_avlBal.setEditable(false);
        setBalance();
```

```
    // fis.close();
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    cb_Month = new javax.swing.JComboBox();
    jLabel2 = new javax.swing.JLabel();
    tf_userName = new javax.swing.JTextField();
    jLabel3 = new javax.swing.JLabel();
    tf_amt = new javax.swing.JTextField();
    tf_submit = new javax.swing.JButton();
    tf_cancel = new javax.swing.JButton();
    jLabel5 = new javax.swing.JLabel();
    jLabel6 = new javax.swing.JLabel();
    tf_CCno = new javax.swing.JTextField();
    tf_avlBal = new javax.swing.JTextField();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    jLabel1.setText("Select Month:-");

    cb_Month.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    cb_Month.setModel(new javax.swing.DefaultComboBoxModel(
        constants.Constants.MONTHS));
cb_Month.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cb_MonthActionPerformed(evt);
    }
});

jLabel2.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
jLabel2.setText("User Name:-");

tf_userName.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
jLabel3.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
jLabel3.setText("Enter Amount");

tf_amt.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N

tf_submit.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
```

```
    tf_submit.setText("Submit");
    tf_submit.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            tf_submitActionPerformed(evt);
        }
    });

    tf_cancel.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    tf_cancel.setText("Cancel");
    tf_cancel.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            tf_cancelActionPerformed(evt);
        }
    });

    jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    jLabel5.setText("Credit Card Number:-");

    jLabel6.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
    jLabel6.setText("Available Balance");

    javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(138, 138, 138)
                    .addComponent(tf_submit,
javax.swing.GroupLayout.PREFERRED_SIZE, 147,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(tf_cancel,
javax.swing.GroupLayout.PREFERRED_SIZE, 147,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(53, 53, 53)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                        .addComponent(jLabel6, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                    .addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(67, 67, 67)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                    .addComponent(cb_Month, 0, 198, Short.MAX_VALUE)
                    .addComponent(tf_userName)
                    .addComponent(tf_amt)
                    .addComponent(tf_CCno)
                    .addComponent(tf_avlBal))))
            .addContainerGap(91, Short.MAX_VALUE))
    );
    jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(35, 35, 35)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(tf_userName, javax.swing.GroupLayout.DEFAULT_SIZE,
40, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE, 33,
Short.MAX_VALUE)
                .addComponent(tf_CCno))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(cb_Month)
                .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
37, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(jLabel3, javax.swing.GroupLayout.DEFAULT_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(tf_amt, javax.swing.GroupLayout.DEFAULT_SIZE, 37,
Short.MAX_VALUE))
            .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(jLabel6, javax.swing.GroupLayout.DEFAULT_SIZE, 31,
Short.MAX_VALUE)

                .addComponent(tf_avlBal))

            .addGap(73, 73, 73)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
LEADING, false)
                .addComponent(tf_submit, javax.swing.GroupLayout.DEFAULT_SIZE, 39,
Short.MAX_VALUE)
                .addComponent(tf_cancel, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    );

    pack();
    }// </editor-fold>

    private void cb_MonthActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void tf_submitActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        if (true) {
            CCTransactions ccTrans = new CCTransactions();
            ccTrans.setCcno(this._usrDtls.getCreditCardNo());
```

```
            ccTrans.setMonth(cb_Month.getSelectedItem().toString());
            ccTrans.setTransamt(tf_amt.getText());
            ccTrans.setUserid("" + this._usrDtls.getId());
            List<UserDetails> subUserList = null;
            Map<Long, CCDtls> getCCDtls = new HashMap<>();
            if (this._usrDtls.getParentAc() == 0l) {
                subUserList = new
UserDetailsDAOImpl().getUserDetalsBsdOnParentId(this._usrDtls.getId());
            }

            long maxAmt = 0l;
            long cardId = 0l;
            for (UserDetails ud : subUserList) {
                CCDtls _ccDtls = new
CCDtlsDAOImpl().getCCDtlsBsdOnUserId(ud.getId());
                getCCDtls.put(ud.getId(), _ccDtls);
                if (_ccDtls.getCcBal() > maxAmt) {
                    maxAmt = _ccDtls.getCcBal();

                    cardId = _ccDtls.getUserId();
                }
            }

            List<CCTransactions> ccTransList =
_ccTransDAOImpl.getCCTrans(this._usrDtls.getId());

            if (this._usrDtls.getParentAc() == 0l) {
                ccTransList = _ccTransDAOImpl.getCCTrans(this._usrDtls.getId());
            } else {
                ccTransList =
_ccTransDAOImpl.getCCTransForJanMonth(this._usrDtls.getId(),
cb_Month.getSelectedItem().toString());
            }

            Long totTransAmtForAMonth = 0l;
            for (CCTransactions cctrans : ccTransList) {
                totTransAmtForAMonth = totTransAmtForAMonth +
Long.parseLong(cctrans.getTransamt());
            }

            if (ccTransList.size() < 3) {
                if (Long.parseLong(tf_amt.getText()) <
Long.parseLong(tf_avlBal.getText())) {
                    ccTrans.setCcno(this._usrDtls.getCreditCardNo());
                    ccTrans.setMonth(cb_Month.getSelectedItem().toString());
                    ccTrans.setTransamt(tf_amt.getText());
                    ccTrans.setUserid("" + this._usrDtls.getId());

                    CCDtls _ccDtls =
_ccDtlsDAOImpl.getCCDtlsBsdOnUserId(this._usrDtls.getId());
```

```
                long bal = _ccDtls.getCcBal();
                long due = _ccDtls.getCcDue();
                _ccDtls.setCcBal((bal - Long.parseLong(tf_amt.getText())));
                _ccDtls.setCcDue((due + Long.parseLong(tf_amt.getText())));
                _ccDtlsDAOImpl.updateCCDtls(_ccDtls);
                if (_ccTransDAOImpl.saveCCTrans(ccTrans)) {
                    clearFields();
                    setBalance();
                    JOptionPane.showMessageDialog(this, "Transaction successful");
                } else {
                    JOptionPane.showMessageDialog(this, "Transaction failed!Contact
Admin");
                }
            } else {
                JOptionPane.showMessageDialog(this, "Transaction declined! Available
balance is less");
            }
        } else {
            try {
                //Validate from hadoop
                System.out.println("Validating the transaction from Hadoop");
                new HadoopAnalyzer().main(this._usrDtls.getId());
                File hfile = new
File(constants.Constants.FILE_HADOOP_OUT_LOCATION);
                int indicator = 0;

                FileReader fileReader = new FileReader(hfile);
                Scanner scanFile = new Scanner(fileReader);
                int freqCount = 0;
                long finalAmtAllowd = 0l;
                while (scanFile.hasNext()) {
                    String data = scanFile.nextLine();
                    String[] splitData = data.split("\t");
                    if (freqCount < Integer.parseInt(splitData[1].split("\\|")[1])) {
                        freqCount = Integer.parseInt(splitData[1].split("\\|")[1]);
                        finalAmtAllowd = Long.parseLong(splitData[1].split("\\|")[0]);
                    }
                }

                if (Long.parseLong(tf_amt.getText()) <
Long.parseLong(tf_avlBal.getText())
                        || Long.parseLong(tf_amt.getText()) < maxAmt) {
                    if ((this._usrDtls.getParentAc() == 0l &&
Long.parseLong(tf_amt.getText()) <= this._usrDtls.getCcLimit())
                            || (Long.parseLong(tf_amt.getText()) + totTransAmtForAMonth
<= this._usrDtls.getCcLimit())) {
                        CCDtls _ccDtls =
_ccDtlsDAOImpl.getCCDtlsBsdOnUserId(this._usrDtls.getId());

                        long bal = _ccDtls.getCcBal();
```

```
                    long due = _ccDtls.getCcDue();
                    _ccDtls.setCcBal((bal - Long.parseLong(tf_amt.getText())));
                    _ccDtls.setCcDue((due + Long.parseLong(tf_amt.getText())));
                    _ccDtlsDAOImpl.updateCCDtls(_ccDtls);
                    if (_ccTransDAOImpl.saveCCTrans(ccTrans)) {
                        clearFields();
                        setBalance();
                        JOptionPane.showMessageDialog(this, "Transaction
successful");
                    } else {
                        JOptionPane.showMessageDialog(this, "Transaction
failed!Contact Admin");
                    }
                } else if (this._usrDtls.getParentAc() == 0l &&
Long.parseLong(tf_amt.getText()) <= finalAmtAllowd) {
                    CCDtls _ccDtls =
_ccDtlsDAOImpl.getCCDtlsBsdOnUserId(this._usrDtls.getId());
                    long bal = _ccDtls.getCcBal();
                    long due = _ccDtls.getCcDue();
                    _ccDtls.setCcBal((bal - Long.parseLong(tf_amt.getText())));
                    _ccDtls.setCcDue((due + Long.parseLong(tf_amt.getText())));
                    _ccDtlsDAOImpl.updateCCDtls(_ccDtls);
                    if (_ccTransDAOImpl.saveCCTrans(ccTrans)) {
                        clearFields();
                        setBalance();
                        JOptionPane.showMessageDialog(this, "Transaction
successful");
                    } else {
                        JOptionPane.showMessageDialog(this, "Transaction
failed!Contact Admin");
                    }
                } else { //Allow transaction based on formula
                    Random rand = new Random();
String otp = "";
                    for (int k = 0; k < 5; k++) {
                        otp = otp + rand.nextInt(9);
                    }
                    String msgBox = "";
                    int counter = 0;
                    java.util.Map<String, Integer> formulaVal = new
java.util.HashMap<String, Integer>();
                    for (String alpha : constants.Constants.FORMULA_CHAR) {
                        if (counter == 0) {
                            //msgBox = msgBox + alpha + ":=" + "" + rand.nextInt(9) + "\t";
                        } else {
                            int val = rand.nextInt(9);
                            msgBox = msgBox + alpha + ":=" + "" + val + "\t  ";
                            formulaVal.put(alpha, val);
                        }
                        counter++;
```

```
                    }
                    String inputRec = JOptionPane.showInputDialog(this, msgBox);
                    System.out.println("Input " + inputRec);
                    String[] splitUserForm = this._usrDtls.getFormula().split("#");
                    int userVal = 0;

                    //First operator validation
                    if (splitUserForm[1].equalsIgnoreCase("+")) {
                        userVal = formulaVal.get(splitUserForm[0]) +
formulaVal.get(splitUserForm[2]);
                    } else {
                        userVal = formulaVal.get(splitUserForm[0]) -
formulaVal.get(splitUserForm[2]);
                    }

                    //Second operator validation
                    if (splitUserForm[3].equalsIgnoreCase("+")) {
                        userVal = userVal + formulaVal.get(splitUserForm[4]);
                    } else {
                        userVal = userVal - formulaVal.get(splitUserForm[4]);
                    }

                    //System.out.println("User amt " + userVal + " and rec amt " +
inputRec);
                    if (userVal == Long.parseLong(inputRec)) {
                        //Sending otp to mail id
//                          if (this._usrDtls.getParentAc() == 0l) {
//                              SendMail.main(this._usrDtls.getEmailId(), "Your OTP is =>"
+ otp, "OTP");
//                          } else {
//                              UserDetails subUser = new
UserDetailsDAOImpl().getUserDetals(this._usrDtls.getParentAc());
//                              SendMail.main(subUser.getEmailId(), "Your OTP is =>" +
otp, "OTP");
//                          }

                        //String otpEntered = JOptionPane.showInputDialog("Enter
OTP");
                        //if (otpEntered.equals(otp)) {
                            CCDtls _ccDtls =
_ccDtlsDAOImpl.getCCDtlsBsdOnUserId(this._usrDtls.getId());
                            long bal = _ccDtls.getCcBal();
                            long due = _ccDtls.getCcDue();
                            _ccDtls.setCcBal((bal - Long.parseLong(tf_amt.getText())));
                            _ccDtls.setCcDue((due + Long.parseLong(tf_amt.getText())));
                            _ccDtlsDAOImpl.updateCCDtls(_ccDtls);
                            if (_ccTransDAOImpl.saveCCTrans(ccTrans)) {
                                clearFields();
                                setBalance();
                                JOptionPane.showMessageDialog(this, "Transaction
```

```
successful");
                                } else {
                                    JOptionPane.showMessageDialog(this, "Transaction
failed!Contact Admin");
                                }
                            //  }
//                              else {
//                                  JOptionPane.showMessageDialog(this, "Invalid OTP");
//                              }
                            }
                            else {
                                JOptionPane.showMessageDialog(this, "Credit card fraud
detected");
                            }
                        }
                    } else {
                        JOptionPane.showMessageDialog(this, "Transaction declined!
Available balance is less");
                    }
                } catch (Exception ex) {

Logger.getLogger(TransactionFrame.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        } else {
            JOptionPane.showMessageDialog(this, "Invalid pin");
        }
    }

    private void setBalance() {
        CCDtls _ccDtls =
_ccDtlsDAOImpl.getCCDtlsBsdOnUserId(this._usrDtls.getId());
        tf_avlBal.setText("" + _ccDtls.getCcBal());
    }

    private void clearFields() {
        this.tf_amt.setText("");
    }

    private void tf_cancelActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        dispose();
        new UserLogin().main();
    }


    /**
     * @param args the command line arguments
     */
    public static void main(final UserDetails ud) {
```

```
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(TransactionFrame.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(TransactionFrame.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(TransactionFrame.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(TransactionFrame.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                File file = new File(constants.Constants.RFID_READER_FILE);
                FileReader fileReader = null;
                try {
                    fileReader = new FileReader(file);
                } catch (FileNotFoundException ex) {

Logger.getLogger(TransactionFrame.class.getName()).log(Level.SEVERE, null, ex);
                }
                Scanner scan = new Scanner(fileReader);
                long id = 0;
                while (scan.hasNext()) {
                    id = (Long.parseLong(scan.nextLine()));
```

```
            }
            //UserDetails _usrDB = new UserDetailsDAOImpl().getUserDetals(id);
            UserDetails _usrDB = new
UserDetailsDAOImpl().getUserDetals(ud.getId());
            new TransactionFrame(_usrDB).setVisible(true);
        }
      });
   }

   // Variables declaration - do not modify
   private javax.swing.JComboBox cb_Month;
   private javax.swing.JLabel jLabel1;
   private javax.swing.JLabel jLabel2;
   private javax.swing.JLabel jLabel3;
   private javax.swing.JLabel jLabel5;
   private javax.swing.JLabel jLabel6;
   private javax.swing.JPanel jPanel1;
   private javax.swing.JTextField tf_CCno;
   private javax.swing.JTextField tf_amt;
   private javax.swing.JTextField tf_avlBal;
   private javax.swing.JButton tf_cancel;
   private javax.swing.JButton tf_submit;
   private javax.swing.JTextField tf_userName;
   // End of variables declaration
}
```

B. PLAGARISM REPORT

CREDIT CARD FRAUD DETECTION WITH FORMULA BASED AUTHENTICATION

Abstract—**Nowadays, digital withdrawals mostly rely on cards. Days have come where people don't have to carry cash in their pockets and just a small card is enough to make all the withdrawals. Problems with cards will lead to passwords being hacked and easily lead to fraud. Credit card fraud can be identified using Hidden Markov Model (HMM) and Formula Based Authentication.**

In the Existing system, card withdrawals are very routine among the people and the frauds corresponding to the improvement of security are increasing. In the proposed system, ML algorithms have been applied to detect mastercard deception in a disproportionate dataset. In the modification process, an application is developed for a banking sector particularly for a credit or ATM card. Users can create an account and get the ATM card along with a unique formula which should be used during suspicious transactions. The user behavior of every transaction is tracked by Hidden Markov Model and if there are any occurrences of suspicious transactions, then a message is sent to the user with the keys that are required to complete the formula. After the user applies the keys to the formula the solution must be entered as the password in order to complete the transaction.

I. INTRODUCTION

Electronic and non electronic buyers use credit cards for digital withdrawals as a method of settlement in a huge way even though this method has few drawbacks. Card based activities are frequently targeted by culprits, criminals as well as perpetrators. For any transaction, only a particular attribute has to be inserted. In most cases, One Time-Password (OTP) is used as an extra safety. Specifically for international withdrawals, a method called Card-Not-Present, is used where only the details are required rather than the physical card for unauthorized payments. It is very easy to get the card details using methods like shoulder surfing, buying card details, credit card stealing as well as web traffic sniffing.

The main sufferer of the fraud will be the bank, trader and registrant. The main duties of the registrant is to detect any malicious activities and report fraudulent withdrawals. The bank then has the responsibility for analyzing all issues and if any evidence for malicious activities are identified, then the amount withdrawn is reversed.This issue can be resolved using formula based authentication.

II. LITERATURE SURVEY

Andrea [4] proposed a novel learning strategy using authentic data accommodating over abundance of negotiations which were approved over 3 years explains how to overcome a number of challenges which includes verification latency that is few transactions are checked by investigators from time to time,class imbalance which specifies more frauds than genuine actions and concept drift which states that the fraudster strategies change time to time as customers evolve their practices. Many proposed ML algorithms depend upon assumptions due to which there is a lack of realism that leads to two major concerns: the strategy and schedule in which the supervised data is fetched and the initiatives done to identify the fraud. A formalization of the malicious problem proposed with help of an industrial partner which describes the malicious systems operations which analyzes massive streams of withdrawals and illustrates performance measures used for fraudulent purposes. Advantages include addressing drift and verification latency and class imbalance and implementing alert feedback interaction meanwhile the alert interactions were less accurate.

John [5] explains the performance of various ML methods on a large crooked dataset. The major challenges of fraud detection are that the malicious datasets are highly skewed and the original as well as the fraudulent behavior change constantly. The procedure on dataset, variables chosen and diagnosed methods has shown great effect in performance of theft activities. The European cardholder's transaction dataset is considered and an undergoing sampling and hybrid techniques are implemented on the skew data which consists of 284,407 withdrawals. These methods implemented in python are evaluated based on the performance on discrete, precision, accuracy, Matthews correlation coefficient, clarity and stabilized grouping rate. The accuracy for k-nearest neighbor and naive bayes were 97.69% and 97.92% while the logistic regression classifiers was 54.86% resulting in logistic regression as a less accurate model compared to the other two models. More parameter tuning is required in order to improve the accuracy.

Lutao [6] explains the association between special cases and behavior features considered based on behavior certificates. Initially, the behavior features were extracted from the card holders historical transaction record patterns as it is an important way to detect fraud. Based on the behavior features, a behavior certificate is constructed. Using this reduces the danger rate for cardholder's revenue and if it is larger than the restriction rate, it is estimated as trickery. To reflect the applicant's transaction habits, a new Fraud Detection

System (FDS) based on behavior certificate is introduced. By implementing a Behavior Certificate, the applicant's transaction habits can be reflected using a Fraud Detection System. The major asset in this paper is that it is highly effective while performing with simulated data, whereas it is difficult to learn the characteristics of unauthorized activities.

Sahil Dhankhad[7] proposed a commensurate evaluation, explaining about the hidden ideas to detect fraudulent activities using real world dataset. The important variables that lead to fraudulent activities are identified and are used to achieve higher accuracy. The advancement in modernized technology has escalated the fraudulent activities. Inorder to delineate the hidden patterns, the supervised ML algorithms were applied. Publicly available datasets were used which were highly imbalanced. The major advantages in this paper includes using various supervised machine learning designs to administer a classifier using a combination of researching methods whereas, the drawback includes delay in updating the supervised model due to verification latency.

Aisha Abdallah n [8] explains the association of Fraud Detection System with Faster Payment System to protect computerized methodology from fraudsters. FDS and FPS individually are scarce to deliver the surety needed for computerized processes. It provides the solution for the problems like large amounts of information, real time identification, concept drift as well as skewed distribution hinders the performance of FDS and provides a systematic and comprehensive overview. The five electronic commerce systems selected for this paper are automobile insurance, credit card, online auction, telecommunication and healthcare insurance in which the prevalent fraud types are examined closely. Further, the state-of-the-art of the FDS is also systematically introduced. The increase in the false prediction rate of honest withdrawal of money is due to the combination of FDS and FPS and also increases the cost of investigation for banks.

III. INFERENCE FROM LITERATURE

The major challenges were concept drift, class imbalance and verification latency. Verification latency that means few transactions are checked by investigators from time to time, class imbalance that specifies more frauds than genuine transactions and concept drift which states that the fraudsters strategies change from time to time as customers evolve their habits. There is a lack of realism in the dataset and it was less precise with the alert interactions. As the data size increased the validation performance decreased. It is observed that there is a delay in updating the supervised model due to the verification latency. In some cases, there was a false prediction rate of honest transactions. Parameter tuning can be done to improve accuracy. The other major drawback was that the customer's habits evolve and the fraudsters change their strategies accordingly. There are genuine transactions along with far outnumbered frauds. Alert systems must be precise and should overcome the outliers.

IV. EXISTING SYSTEM

In the existing methodology whenever the user tries to make a transaction, the user receives an OTP (One Time Password), which must be entered to make successful money withdrawals. But, if the OTP is seen by the fraudster through shoulder surfing, web trafficking or any alternate means then the fraudster can steal money from the users card giving rise to credit card fraud.

V. PROPOSED WORK

This implementation is developed for the investment sector. By using Formula Based Authentication, banks can ensure reliability. Users can create an account and get the card issued along with a unique formula with the help of the bank. Every user behavior is monitored using the HMM based on the money withdrawal sequence. The client's frequency of transaction is monitored. Whenever the HMM detects any pattern out of bounds, it immediately issues a warning. This induces the formula based authentication, which sends a set of keys to the client. If the client enters the correct solution, the transaction is successful. This method assists in identifying fraudulent actions.

Modules:

1. User : A User is also called as bank depositors, since they trust the banking facilities for storage of money to receive interest. There are many types of clients in a bank, such as trustees, joint account holders, to which they have separate facilities to support their people.

2. Bank Account Registration : The client is asked to create an account under their name, along with a certain amount of deposit. They will be allowed to

withdraw, deposit as well as a loan. When an account is created, the client will be given a unique formula, which will be used by the particular client whenever they find any track of suspicious transactions. The client will be asked to enter the key in the formula to have a successful transaction.

3. Bank Server :Bank Service Provider will have a data storage which contains the information about users. They also maintain all the user information that helps the user to authenticate whenever they wish to access the account. The bank server will establish relationships with the consumer and other modules of the Company server to communicate. Hence, a User Interface Frame is connected.

4. HMM initiated using big data (user behavior) : It is operated for analyzing client actions on every transaction. It is executed for understanding retraction of money by the client, meaning the basic idea is total sum of withdrawal each month and the second being frequency of withdrawal. The time recurrence is also monitored and recorded.

5. Money Withdrawal (Malicious user behavior) : HMM tracks the user behavioral patterns. If any malicious transaction is identified, HMM will send an indication in the form of keys to the user, indicating the fraudulent withdrawal. If the client enters the correct keys to the formula, the transaction will be successful. The transactions will be blocked if the answer to the formula is incorrect.

6. Money Withdrawal (Normal user behavior): When the client withdraws a certain amount, the HMM tracks the user behavior patterns. If transactions are found to be normal, then the permission to withdraw the money is successful.
7. Formula-based Verification: Formula Based Affirmation provides safety by adding a formula. The formula is unique for every user, registered at the time of creating an account. The keys to the formula changes every time, and the user is requested to submit an answer following the substitution of corresponding keys to the formula. This usage of formula is required only when the user tries to withdraw beyond the permitted.

VI PROPOSED ALGORITHM

Step 1: The client registers in a bank with new account details.

Step 2: Clients will be issued with a unique formula that will be used during a suspicious transaction detected by the HMM.

Step 3: In a certain predicament, if the client tries to take an amount that is out of the typical observable patterns, the HMM will immediately raise uncertainty.

Step 4: Formula Based Authentication will be initiated through the HMM.

Step 5: The client will receive a set of keys which are unrepeatable to his formula.

Step 6: The client will apply these keys to the formula, and once the correct password is given, the transaction will be fulfilled.

HMM provides instructions about evaluation, decrypting, and learning. Evaluation is defined as expecting the monitoring order further as well as reverse design. Decrypting specifies all unknown states order (Viterbi). Using the observed information, HMM will be created using Learning (Baum-Welch).

The HMM model works on the transaction history of the client, and after the required formulation if the current transaction is found to be suspicious, then the formula based authentication is initiated. If the user is able to authorize using the keys to their formula, then the transaction will be successful.

Numeric Notation for formula based authentication

Initially the user gets a unique formula while registering for the credit card. During normal transactions the user does not have to enter the authentication formula but if the HMM model detects any particular deals that are not the general user behavior pattern i.e, if the model detects the money withdrawal as skeptical then the user receives the keys through messages.
Example:
Let the unique formula be A+B-C+D.
User will receive a message with keys as shown when suspicious transaction is found by the HMM model:
A=3 B=2 C=1 D=1
The user must apply the formula using the keys:
=A+B-C+1
= 3+2-1+1
= 5
Hence this must be entered as the password in order to complete the transaction. Every suspicious transaction generates keys randomly. Therefore, even if the fraudsters hack the mobile for keys they will not be able to get the money as the formula remains confidential with the user. In this way, the user's account will be free from fraudulent activities.

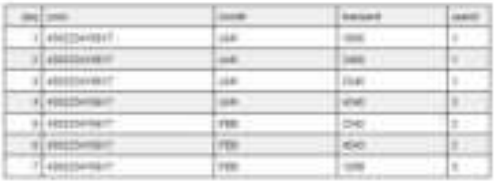
VII. RESULTS AND DISCUSSION

The main aim is to establish a systematic formula based authentication model using ML methods for predicting fraud research. In our proposed work, the

client withdrawal history acts as a dataset for the HMM model. Whenever the HMM detects a doubtful transaction the formula based verification will be activated.

When the user enters the right key, the transaction will be successful.

When the customer enters the wrong key, a suspicious transaction will be identified.



Transaction history of the customer.

VIII. CONCLUSION

More credit card forgery is happening in huge level these days. Even after the existence of cyber security, dishonest money extractions are still active. Formula based authentication is one of many processes to perform the transactions securely. This process allows only genuine deals to take place. Hidden Markov Model is a robust technique for user behavior pattern extraction and to detect deceitful behavior. HMM is used to get a set of unknown variables from known variables. Pattern changes in the transaction history of the account holder can be detected by using the HMM model. If the transaction is detected to be fraudulent an authentication key is sent to the user, and the user must apply the keys to the formula in order to find the solution and enter it as password for successful transaction.

IX FUTURE ENHANCEMENTS

Methodological filtering can be made more effective using various algorithms. Hardware implementation can be improvised using counterfactual analysis and can be interfaced using near-field communication for simple transactions. A toggle button can be used to enable formula based detection transaction (for eg: the user can choose to enable or disable the transaction limit).

REFERENCES

[1]The importance of credit cards: https://budgeting.thenest.com/importancecredit- cards-29514.html

[2]The chargeback process in a credit card: https://chargebacks911.com/chargeback-process/

[3]Low and Slow Is How the Credit Card Fraudsters Roll: https://www.threatmetrix.com/digital-identity-blog/fraudprevention/low-and-slow-is-how-the-credit-card-fraudsters-roll/

[4]Andrea Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi, "Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy," in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 8, pp. 3784-3797, Aug. 2018.

[5]John. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," 2017 International Conference on Computing Networking and Informatics (ICCNI), Lagos, 2017, pp. 1-9.

[6]Lutao Zheng et al., "A new credit card fraud detecting method based on behavior certificate," 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), Zhuhai, 2018, pp. 1-6.

[7]Sahil Dhankhad, Emad A. Mohammed, Behrouz Far, (2018) "Supervised Machine Learning Algorithms for Credit Card Fraudulent".

[8]Aisha Abdallah n , Mohd Aizaini Maarof, Anazida Zainal,(2016),"Survey on systematic and comprehensive overview"

CCFDS final draft (1).docx

ORIGINALITY REPORT

1%	1%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	e-sciencecentral.org Internet Source	<1%
2	www.ijcrt.org Internet Source	<1%
3	www.sersc.org Internet Source	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

CREDIT CARD FRAUD DETECTION WITH FORMULA BASED AUTHENTICATION

***Abstract*—Nowadays, digital withdrawals mostly rely on cards. Days have come where people don't have to carry cash in their pockets and just a small card is enough to make all the withdrawals. Problems with cards will lead to passwords being hacked and easily lead to fraud. Credit card fraud can be identified using Hidden Markov Model (HMM) and Formula Based Authentication.**

In the Existing system, card withdrawals are very routine among the people and the frauds corresponding to the improvement of security are increasing. In the proposed system, ML algorithms have been applied to detect master card deception in a disproportionate dataset. In the modification process, an application is developed for a banking sector particularly for a credit or ATM card. Users can create an account and get the ATM card along with a unique formula which should be used during suspicious transactions. The user behavior of every transaction is tracked by Hidden Markov Model and if there are any occurrences of suspicious transactions, then a message is sent to the user with the keys that are required to complete the formula. After the user applies the keys to the formula the solution must be entered as the password in order to complete the transaction.

I. INTRODUCTION

Electronic and non electronic buyers use credit cards for digital withdrawals as a method of settlement in a huge way even though this method has few drawbacks. Card based activities are frequently targeted by culprits, criminals as well as perpetrators. For any transaction, only a particular attribute has to be inserted. In most cases, One Time-Password (OTP) is used as an extra safety. Specifically for international withdrawals, a method called Card-Not-Present, is used where only the details are required rather than the physical card for unauthorized payments. It is very easy to get the card details using methods like shoulder surfing, buying card details, credit card stealing as well as web traffic sniffing.

The main sufferer of the fraud will be the bank, trader and registrant. The main duties of the registrant is to detect any malicious activities and report fraudulent withdrawals. The bank then has the responsibility for analyzing all issues and if any evidence for malicious activities are identified, then the amount withdrawn is reversed.This issue can be resolved using formula based authentication.

II. LITERATURE SURVEY

40

Andrea [4] proposed a novel learning strategy using authentic data accommodating over abundance of negotiations which were approved over 3 years explains how to overcome a number of challenges which includes verification latency that is few transactions are checked by investigators from time to time,class imbalance which specifies more frauds than genuine actions and concept drift which states that the fraudster strategies change time to time as customers evolve their practices. Many proposed ML algorithms depend upon assumptions due to which there is a lack of realism that leads to two major concerns: the strategy and schedule in which the supervised data is fetched and the initiatives done to identify the fraud. A formalization of the malicious problem proposed with help of an industrial partner which describes the malicious systems operations which analyzes massive streams of withdrawals and illustrates performance measures used for fraudulent purposes. Advantages include addressing drift and verification latency and class imbalance and implementing alert feedback interaction meanwhile the alert interactions were less accurate.

John [5] explains the performance of various ML methods on a large crooked dataset. The major challenges of fraud detection are that the malicious datasets are highly skewed and the original as well as the fraudulent behavior change constantly. The procedure on dataset, variables chosen and diagnosed methods has shown great effect in performance of theft activities. The European cardholder's transaction dataset is considered and an undergoing sampling and hybrid techniques are implemented on the skew data which consists of 284,407 withdrawals. These methods implemented in python are evaluated based on the performance on discrete, precision, accuracy, Matthews correlation coefficient, clarity and stabilized grouping rate. The accuracy for k-nearest neighbor and naive bayes were 97.69% and 97.92% while the logistic regression classifiers was 54.86% resulting in logistic regression as a less accurate model compared to the other two models. More parameter tuning is required in order to improve the accuracy.

Lutao [6] explains the association between special cases and behavior features considered based on behavior certificates. Initially, the behavior features were

extracted from the card holders historical transaction record patterns as it is an important way to detect fraud. Based on the behavior features, a behavior certificate is constructed. Using this reduces the danger rate for cardholder's revenue and if it is larger than the

restriction rate, it is estimated as trickery. To reflect the applicant's transaction habits, a new Fraud Detection System (FDS) based on behavior certificate is

introduced. By implementing a Behavior Certificate, the applicant's transaction habits can be reflected using a Fraud Detection System. The major asset in this paper is that it is highly effective while performing with simulated data, whereas it is difficult to learn the characteristics of unauthorized activities.

Sahil Dhankhad[7] proposed a commensurate evaluation, explaining about the hidden ideas to detect fraudulent activities using real world dataset. The important variables that lead to fraudulent activities are identified and are used to achieve higher accuracy. The advancement in modernized technology has escalated the fraudulent activities. Inorder to delineate the hidden patterns, the supervised ML algorithms were applied. Publicly available datasets were used which were highly imbalanced. The major advantages in this paper includes using various supervised machine learning designs to administer a classifier using a combination of researching methods whereas, the drawback includes delay in updating the supervised model due to verification latency.

Aisha Abdallah n [8] explains the association of Fraud Detection System with Faster Payment System to protect computerized methodology from fraudsters. FDS and FPS individually are scarce to deliver the surety needed for computerized processes. It provides the solution for the problems like large amounts of information, real time identification, concept drift as well as skewed distribution hinders the performance of FDS and provides a systematic and comprehensive overview. The five electronic commerce systems selected for this paper are automobile insurance, credit card, online auction, telecommunication and healthcare insurance in which the prevalent fraud types are examined closely. Further, the state-of-the-art of the FDS is also systematically introduced. The increase in the false prediction rate of honest withdrawal of money is due to the combination of FDS and FPS and also increases the cost of investigation for banks.

III. INFERENCE FROM LITERATURE

The major challenges were concept drift, class imbalance and verification latency. Verification latency that means few transactions are checked by investigators from time to time, class imbalance that specifies more frauds than genuine transactions and concept drift which states that the fraudsters strategies change from time to time as customers evolve their habits. There is a lack of realism in the dataset and it was less precise with the alert interactions. As the data size increased the validation performance decreased. It is observed that there is a delay in updating the

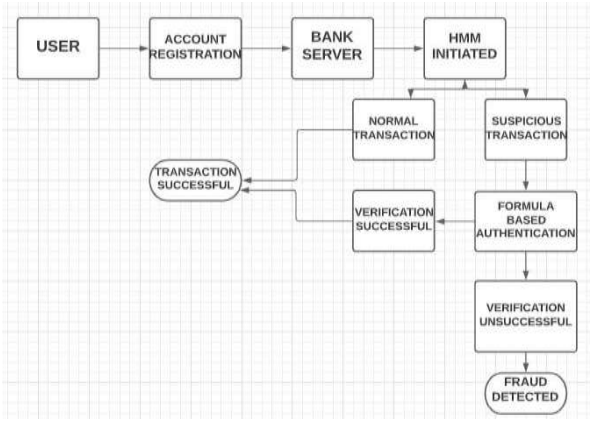
supervised model due to the verification latency. In some cases, there was a false prediction rate of honest transactions. Parameter tuning can be done to improve accuracy. The other major drawback was that the customer's habits evolve and the fraudsters change their strategies accordingly. There are genuine transactions along with far outnumbered frauds. Alert systems must be precise and should overcome the outliers.

IV. EXISTING SYSTEM

In the existing methodology whenever the user tries to make a transaction, the user receives an OTP (One Time Password), which must be entered to make successful money withdrawals. But, if the OTP is seen by the fraudster through shoulder surfing, web trafficking or any alternate means then the fraudster can steal money from the users card giving rise to credit card fraud.

V. PROPOSED WORK

This implementation is developed for the investment sector. By using Formula Based Authentication, banks can ensure reliability. Users can create an account and get the card issued along with a unique formula with the help of the bank. Every user behavior is monitored using the HMM based on the money withdrawal sequence. The client's frequency of transaction is monitored. Whenever the HMM detects any pattern out of bounds, it immediately issues a warning. This induces the formula based authentication, which sends a set of keys to the client. If the client enters the correct solution, the transaction is successful. This method assists in identifying fraudulent actions.



Modules:

1. User : A User is also called as bank depositors, since they trust the banking facilities for storage of money to receive interest. There are many types of clients in a bank, such as trustees, joint account holders, to which they have separate facilities to

support their people.

2. Bank Account Registration : The client is asked to create an account under their name, along with a certain amount of deposit. They will be allowed to withdraw, deposit as well as a loan. When an account is created, the client will be given a unique formula, which will be used by the particular client whenever they find any track of suspicious transactions. The client will be asked to enter the key in the formula to have a successful transaction.

3. Bank Server :Bank Service Provider will have a data storage which contains the information about users. They also maintain all the user information that helps the user to authenticate whenever they wish to access the account. The bank server will establish relationships with the consumer and other modules of the Company server to communicate. Hence, a User Interface Frame is connected.

4. HMM initiated using big data (user behavior) : It is operated for analyzing client actions on every transaction. It is executed for understanding retraction of money by the client, meaning the basic idea is total sum of withdrawal each month and the second being frequency of withdrawal. The time recurrence is also monitored and recorded.

5. Money Withdrawal (Malicious user behavior) : HMM tracks the user behavioral patterns. If any malicious transaction is identified, HMM will send an indication in the form of keys to the user, indicating the fraudulent withdrawal. If the client enters the correct keys to the formula, the transaction will be successful. The transactions will be blocked if the answer to the formula is incorrect.

6. Money Withdrawal (Normal user behavior): When the client withdraws a certain amount, the HMM tracks the user behavior patterns. If transactions are found to be normal, then the permission to withdraw the money is successful.
7. Formula-based Verification: Formula Based Affirmation provides safety by adding a formula. The formula is unique for every user, registered at the time of creating an account. The keys to the formula changes every time, and the user is requested to submit an answer following the substitution of corresponding keys to the formula. This usage of formula is required only when the user tries to withdraw beyond the permitted.

VI PROPOSED ALGORITHM

Step 1: The client registers in a bank with new account

details.

Step 2: Clients will be issued with a unique formula that will be used during a suspicious transaction detected by the HMM.

Step 3: In a certain predicament, if the client tries to take an amount that is out of the typical observable patterns, the HMM will immediately raise uncertainty.

Step 4: Formula Based Authentication will be initiated through the HMM.

Step 5: The client will receive a set of keys which are unrepeatable to his formula.

Step 6: The client will apply these keys to the formula, and once the correct password is given, the transaction will be fulfilled.

HMM provides instructions about evaluation, decrypting, and learning. Evaluation is defined as expecting the monitoring order further as well as reverse design. Decrypting specifies all unknown states order (Viterbi). Using the observed information, HMM will be created using Learning (Baum-Welch).

The HMM model works on the transaction history of the client, and after the required formulation if the current transaction is found to be suspicious, then the formula based authentication is initiated. If the user is able to authorize using the keys to their formula, then the transaction will be successful.

Numeric Notation for formula based authentication

Initially the user gets a unique formula while registering for the credit card. During normal transactions the user does not have to enter the authentication formula but if the HMM model detects any particular deals that are not the general user behavior pattern i.e, if the model detects the money withdrawal as skeptical then the user receives the keys through messages.

Example:

Let the unique formula be A+B-C+D.

User will receive a message with keys as shown when suspicious transaction is found by the HMM model:

A=3 B=2 C=1 D=1

The user must apply the formula using the keys:

=A+B-C+1

= 3+2-1+1

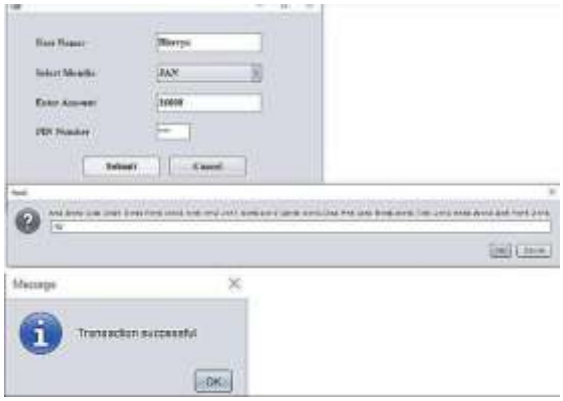
= 5

Hence this must be entered as the password in order to complete the transaction. Every suspicious transaction generates keys randomly. Therefore, even if the fraudsters hack the mobile for keys they will not be able

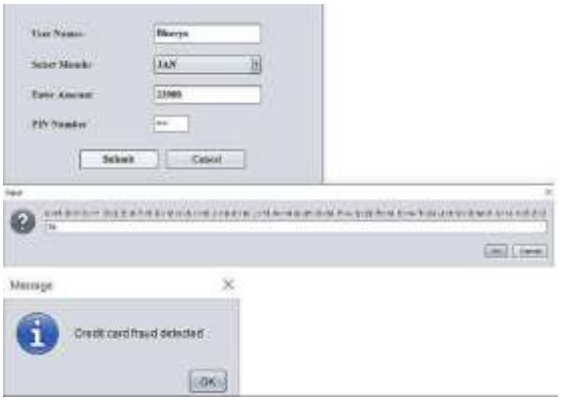
to get the money as the formula remains confidential with the user. In this way, the user's account will be free from fraudulent activities.

VII. RESULTS AND DISCUSSION

The main aim is to establish a systematic formula based authentication model using ML methods for predicting fraud research. In our proposed work, the client withdrawal history acts as a dataset for the HMM model. Whenever the HMM detects a doubtful transaction the formula based verification will be activated.



When the user enters the right key, the transaction will be successful.



When the customer enters the wrong key, a suspicious transaction will be identified.

seq	card	month	transamt	userid
1	4302234156617	JAN	1000	1
2	4302234156617	JAN	2400	1
3	4302234156617	JAN	2340	1
4	4302234156617	JAN	4340	1
5	4302234156617	FEB	2342	2
6	4302234156617	FEB	4543	2
7	4302234156617	FEB	5200	3

Transaction history of the customer.

VIII. CONCLUSION

More credit card forgery is happening in huge level these days. Even after the existence of cyber security, dishonest money extractions are still active. Formula based authentication is one of many processes to perform the transactions securely. This process allows only genuine deals to take place. Hidden Markov Model is a robust technique for user behavior pattern extraction and to detect deceitful behavior. HMM is used to get a set of unknown variables from known variables. Pattern changes in the transaction history of the account holder can be detected by using the HMM model. If the transaction is detected to be fraudulent an authentication key is sent to the user, and the user must apply the keys to the formula in order to find the solution and enter it as password for successful transaction.

IX FUTURE ENHANCEMENTS

Methodological filtering can be made more effective using various algorithms. Hardware implementation can be improvised using counterfactual analysis and can be interfaced using near-field communication for simple transactions. A toggle button can be used to enable formula based detection transaction.(for eg: the user can choose to enable or disable the transaction limit).

REFERENCES

[1]The importance of credit cards: https://budgeting.thenest.com/importancecredit- cards-29514.html

[2]The chargeback process in a credit card: https://chargebacks911.com/chargeback-process/

[3]Low and Slow Is How the Credit Card Fraudsters Roll: https://www.threatmetrix.com/digital-identity-blog/fraudprevention/low-and-slow-is-how-the-credit-card-fraudsters-roll/
[4]Andrea Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi, ”Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy,” in IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 8, pp. 3784-3797, Aug. 2018.
[5]John. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, ”Credit card fraud detection using machine learning techniques: A comparative analysis,” 2017 International Conference on Computing Networking and Informatics (ICCNI), Lagos, 2017, pp. 1-9.
[6]Lutao Zheng et al., ”A new credit card fraud detecting method based on behavior certificate,” 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), Zhuhai, 2018, pp. 1-6.
[7]Sahil Dhankhad, Emad A. Mohammed, Behrouz Far, (2018) “Supervised Machine Learning Algorithms for Credit Card Fraudulent”.
[8]Aisha Abdallah n , Mohd Aizaini Maarof, Anazida Zainal,(2016),”Survey on systematic and comprehensive overview.