Fortifying Financial Security: Unveiling Advanced Anti-Fraud Systems for Robust Safety Nets

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

S.N.V.S.SAI NAIDU (Reg.No - 40111143)
SAPPASATYAHARSHA (Reg.No - 40111142)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
CATEGORY-1 UNIVERSITY BY UGC
Accredited with Grade "A++" by NAAC |12B Status by UGC |Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119

APRIL - 2024

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Category - I University by UGC
Accredited "A++" by NAAC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work **SAPPA SATYA HARSHA (40111142) and S.N.V.S.SAI NAIDU (40111143)** who carried out the Project entitled **"Fortifying Financial Security: Unveiling Advanced Anti-Fraud Systems for Robust Safety Nets"** under my supervision from November 2023 to April 2024.

Internal Guide

Dr.A.Deepa, M.Tech., Ph.D.

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Viva voce Examination held on_____

Internal Examiner **External Examiner**

DECLARATION

I, **S.N.V.S.SAI NAIDU(40111143),** hereby declare that the Project Report entitled "**Fortifying Financial Security: Unveiling Advanced Anti-Fraud Systems for Robust Safety Nets"** done by me under the guidance of **Dr.A.Deepa, M.Tech., Ph.D,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE:

PLACE: Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of Sathyabama Institute of Science and Technology** 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, and **Dr. L. Lakshmanan M.E., Ph.D.,** Head 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. A. Deepa M.Tech., Ph.D.,** for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

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.

ABSTRACT

Credit card fraud events occur frequently and then lead to huge financial losses. It can happen in both online and offline. But in today's world online fraud transaction activities are increasing day by day. Criminals can use some technologies like Trojan or Phishing to steal the knowledge of other people's credit cards. Therefore, an effective fraud detection method is very important since it can identify a fraud in time when a criminal uses a stolen card to consume. This project aims to focus mainly on machine learning algorithms. The algorithms used are random forest algorithm. Initially we give the Credit card transactions dataset to the system with the help of Machine learning algorithms it will identifies whether the transaction is normal or fraudulent. The results of the algorithms are based on accuracy, precision, recall. We make a comparison with other machine learning algorithms to select the best algorithm based on a particular property perspective and to analyze their performance on credit fraud detection. In addition to the core components of the project outlined previously, it is crucial to address the challenge of imbalanced data by employing techniques like oversampling or under sampling. Furthermore, optimizing model performance through hyperparameter tuning and exploring ensemble methods such as bagging and boosting can significantly enhance fraud detection accuracy. Anomaly detection techniques like isolation forest and one-class SVM offer alternative approaches worth exploring. Real-time deployment and scalability considerations are essential for practical implementation, while navigating ethical and privacy concerns surrounding customer data usage is paramount. Case studies from various industries can provide valuable real-world insights, and interactive visualization tools can aid in better understanding and exploring data and model predictions. By integrating these elements, the project can offer a comprehensive analysis of credit card fraud detection using machine learning algorithms, addressing both technical challenges and ethical considerations.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 INTRODUCTION TO FRAUD DETECTION

Credit card generally refers to a card that is assigned to the customer (cardholder), usually allowing them to purchase goods and services within credit limit or withdraw cash in advance. Credit card provides the cardholder an advantage of the time, i.e., it provides time for their customers to repay later in a prescribed time, by carrying it to the next billing cycle. Credit card frauds are easy targets. Without any risks, a significant amount can be withdrawn without the owner's knowledge, in a short period. Fraudsters always try to make every fraudulent transaction legitimate, which makes fraud detection very challenging and difficult task to detect.

In 2017, there were 1,579 data breaches and nearly 179 million records among which Credit card frauds were the most common form with 133,015 reports, then employment or tax-related frauds with 82,051 reports, phone frauds with 55,045 reports followed by bank frauds with 50,517 reports from the statics released by FTC.

With different frauds mostly credit card frauds, often in the news for the past few years, frauds are in the top of mind for most the world's population. Credit card dataset is highly imbalanced because there will be more legitimate transaction when compared with a fraudulent one. According to 2017, the US Payments Forum report, criminals have shifted their focus on activities related to CF transactions as the security of chip cards were increased. Fig 1.1, shows the number of CF frauds

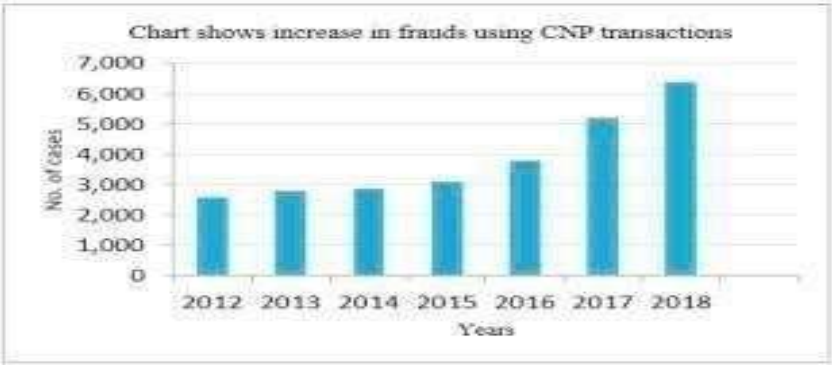


Fig:1.1 Number of CF Cases

Even then there are chances for thieves to misuse the credit cards. There are many machine learning techniques to overcome this problem. Financial fraud is an intentional crime in which a fraudster benefits himself/herself by denying a right to a victim or by obtaining financial gain. As credit card transaction is the most common method of payment in the recent years, the fraud activities have increased rapidly. Enterprises and public institutions are facing a massive problem as huge amount of financial loss are caused by fraud activities. Payments using credit cards have increased in recent years. It may be used in online or in regular shopping. Now-a-days credit card payments are necessary and convenient to use. Due to the increase of fraudulent transactions, there is a need to find the efficient fraud detection model. Fraud is increasing dramatically with the expansion of modern technology. The amount of information in the modern world is also explosive. The promising way to detect the fraud is to analyze the spending behavior of the cardholder. Detecting the fraud means identifying the suspicious one. If any abnormality arises in the spending behavior, then it is considered as suspicious and taken for further consideration. In this behavior-based approach using Random Forest algorithm is applied for fraud detection. Random Forest is an active research area and successfully solves classification problems in noisy and complex domains. Random Forest played a major role in machine learning due to its excellent generalization performance in a wide range of learning problems, The Problem of its excellent generalization performance in a wide range, The Problem of over fitting is very less in Random Forest applications.

1.2 DIFFERENT TYPES OF FRAUD

Misrepresentation discovery frameworks come into situation when the fraudsters surpass the extortion aversion frameworks and begin false exchanges. Alongside the advancements in the Data Innovation and upgrades in the correspondence channels, misrepresentation is spreading everywhere throughout the world with aftereffects of vast measure of false misfortune. Anderson (2007) has recognized and depicted the distinctive sorts of extortion. Charge card cheats can continue in a wide range of

courses, for example, basic burglary, fake cards, Never Got Issue (NRI), application misrepresentation and on the web/Electronic extortion. Master card misrepresentation recognition is appallingly troublesome, yet in addition regular issue for arrangement.

1.2.1 CARD NOT PRESENT TRANSACTION (CNP)

In a Card Not Present (CNP) transaction, the card is not physically present when a customer makes a purchase. The merchant must rely on the cardholder, or someone claiming to be so, to provide card information indirectly, whether through mail, phone, or over the Internet. This type of transaction poses increased risk for fraudulent activity, as the absence of physical verification makes it easier for unauthorized individuals to use stolen card details.

1.2.2 IDENTIFY THEFT

The Identity Theft is divided into two classifications:

Application fraud : Application fraud happens when a man utilizes stolen or counterfeit archives to open a record in someone else's name. Culprits may take records, for example, service bills and bank explanations to develop helpful individual data

Account takeover : This theft happens when an illegal poses as an intelligent customer, gains control of an account and then makes unofficial transactions

1.2.3 SKIMMING

Skimming is an electronic tactic employed by identity thieves to capture a victim's personal information. This technique involves using a skimmer, a small device that scans a credit card and saves the data encoded in the magnetic strip. Skimming can occur during a legitimate transaction at a business establishment. It is often carried out surreptitiously, with criminals discreetly installing skimming devices on legitimate card readers or ATMs. It can also be equipped with Bluetooth or Wi-Fi capabilities, allowing thieves to remotely collect the stolen data without needing physical access to the skimmer itself.

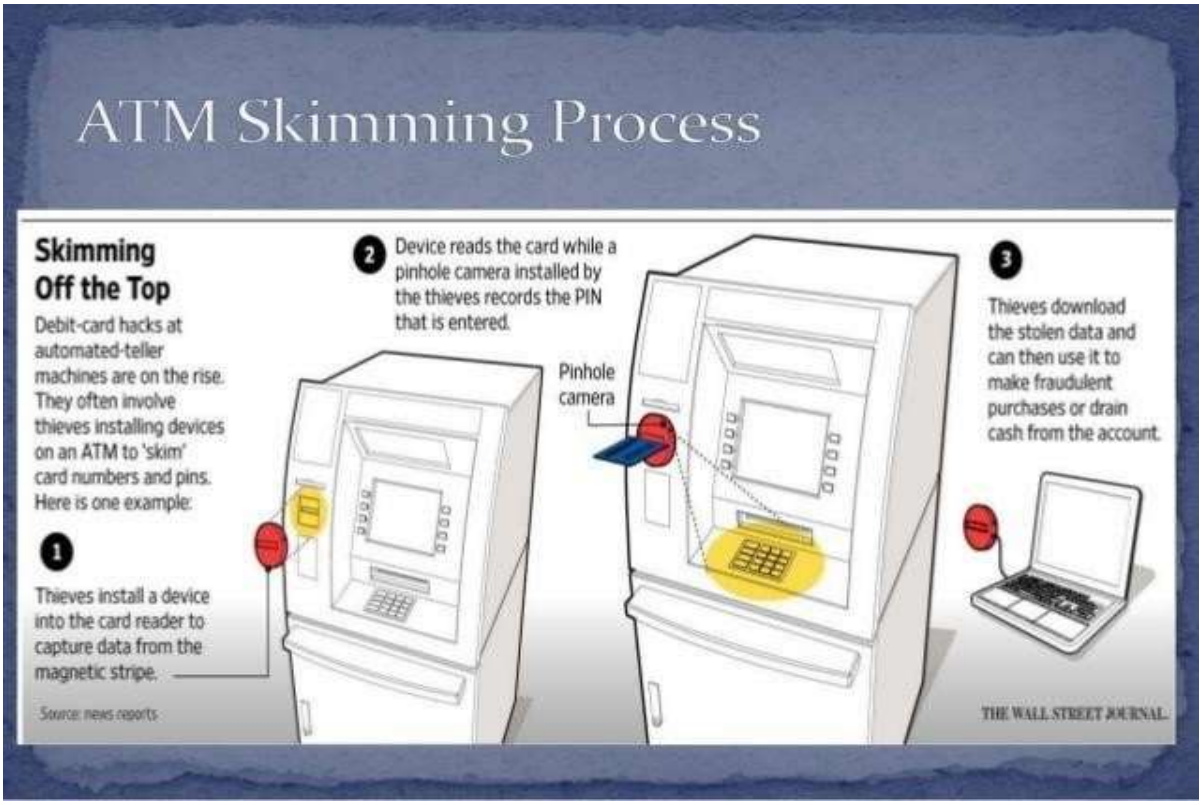


FIG:1.2.3 ATM SKIMMING PROCESS

1.2.4 PHISHING

Phishing email messages, sites, and telephone calls are intended to take cash. Cybercriminals can do this by introducing pernicious programming on your PC or taking individual data off your PC. Phishing is a sort of social designing assault regularly used to take client information, including login certifications and Master card numbers. It happens when an assailant, taking on the appearance of a confided in element, hoodwinks a casualty into opening an email, text, or instant message. The beneficiary is then deceived into clicking a pernicious connection, which can prompt the establishment of malware, the solidifying of the framework as a component of a ransom ware assault or the noteworthy of delicate data. An assault can have pulverizing comes about. For people, these incorporate unapproved buys, the taking of assets, or distinguish robbery. Credit card fraud is increasing rapidly, there are various techniques of detecting the credit card frauds

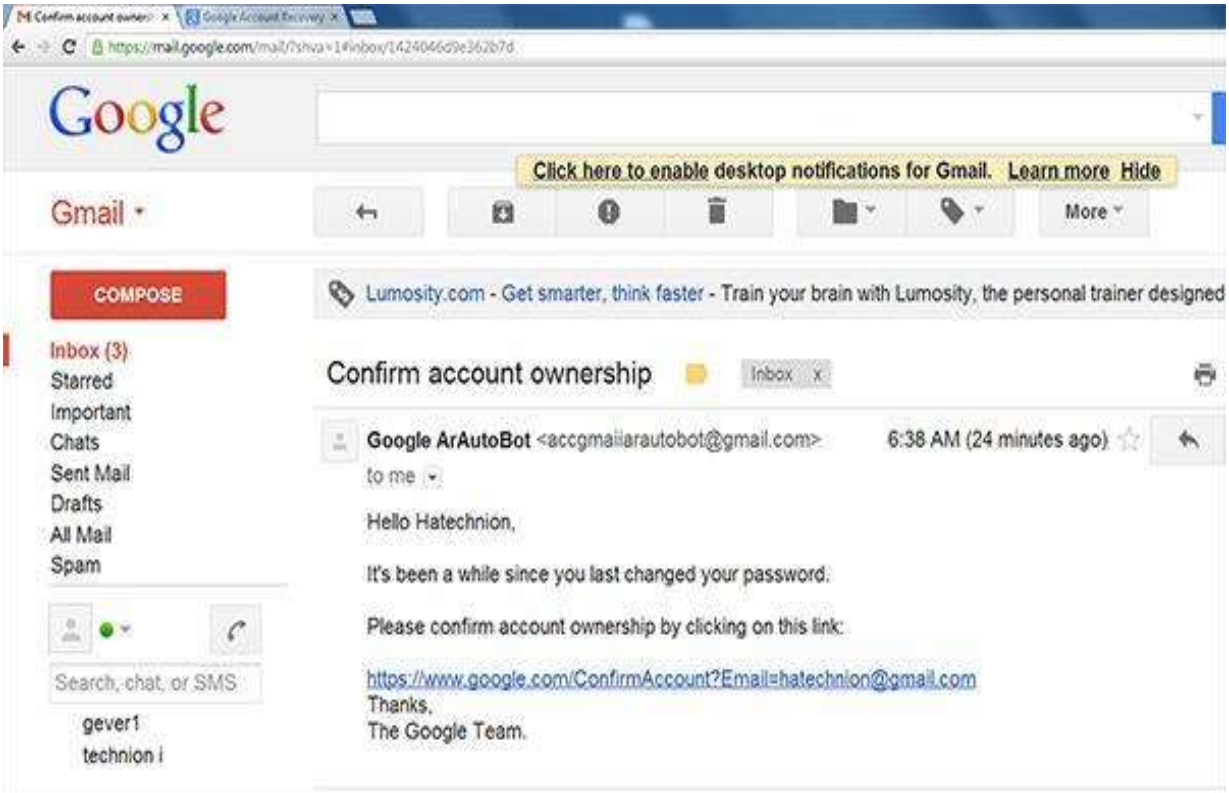


FIG:1.2.4: Real time example of Phishing attack

1.3 MOTIVATION

At the current state of the world, financial organizations expand the availability of financial facilities by employing innovative services such as credit cards, Automated Teller Machines (ATM), internet and mobile banking services. Besides, along with the rapid advances of ecommerce, the use of credit cards has become a convenient and necessary part of financial life. Credit card is a payment card supplied to customers as a system of payment. There are lots of advantages in using credit cards such as :

Ease of purchase : Credit cards can make life easier. They allow customers to purchase on credit in arbitrary time, location, and amount, without carrying the cash. Provide a convenient payment method for purchases made on the internet, over the telephone, through ATMs, etc.

Keep customer credit history : Having a good credit history is often important in the detecting loyal customers. This history is valuable not only for credit cards, but also for other financial services like loans, rental applications, or even some jobs. Lenders and issuers of credit mortgage companies, credit card companies, retail stores, and utility companies can review customer credit.

Protection of Purchases : Credit cards may also offer customers additional protection if the purchased merchandise becomes lost, damaged, or stolen. Both the buyer's credit card statement and the company can confirm that the customer has bought if the original receipt is lost or stolen. In addition, some credit card companies provide insurance for large purchases.

Despite all mentioned advantages, the problem of fraud is a serious issue in-banking services that threaten credit card transactions especially. Fraud is an intentional deception with the purpose of obtaining financial gain or causing loss by implicit or explicit trick. Fraud is a public law violation in which the fraudster gains an unlawful advantage or causes unlawful damage.

The estimation of amount of damage made by fraud activities indicates that fraud costs a very considerable sum of money. Credit card fraud is increasing significantly with the development of modern technology resulting in the loss of billions of dollars worldwide each year. Statistics from the Internet Crime Complaint Center show that there has been a significant rising in reported fraud in last decade. Financial losses caused due to online fraud only in the US, was reported to be $3.4 billion in 2011. Fraud detection involves identifying scarce fraud activities among numerous legitimate transactions as quickly as possible. Fraud detection methods are developing rapidly to adapt with new incoming fraudulent strategies across the world. But, development of new fraud detection techniques becomes more difficult due to the severe limitation of the ideas exchanged in fraud detection. On the other hand, fraud detection is essentially a rare event problem, which has been variously called outlier analysis, anomaly detection, exception mining, mining rare classes, mining imbalanced data etc.

1.4 PROBLEM STATEMENT

The Credit Card Fraud Detection Problem involves creating a model based on historical credit card transactions, particularly those identified as fraudulent, to predict the likelihood of fraud in new transactions. The objective is to achieve perfect detection of fraudulent transactions while minimizing false positives. This entails developing a highly accurate model that identifies all instances of fraud while reducing the number of legitimate transactions incorrectly classified as fraudulent. Achieving this goal requires a balance between sensitivity (detecting fraud) and specificity (minimizing false alarms), ensuring robust performance in real-world scenarios where the consequences of missing fraudulent transactions or wrongly flagging legitimate ones can be significant. In addition to developing a highly accurate model for fraud detection, it is crucial to continuously monitor and update the model to adapt to evolving fraud patterns and tactics. This involves implementing robust monitoring systems that detect shifts in transaction behavior and promptly retrain the model to maintain its effectiveness. Furthermore, deploying advanced anomaly detection techniques, such as unsupervised learning algorithms, can complement traditional supervised approaches by identifying previously unseen patterns indicative of fraud. Additionally, employing ensemble learning methods, where multiple models are combined to make predictions, can enhance the overall performance and resilience of the fraud detection system against sophisticated fraud attempts. Regular evaluation of the model's performance metrics, such as precision, recall, and F1 score, ensures ongoing optimization and refinement of the fraud detection system to meet the evolving challenges of fraud prevention in the financial sector. Continuous monitoring and adaptation are essential in the Credit Card Fraud Detection Problem. Besides developing accurate models, robust monitoring systems must detect shifts in transaction behavior, prompting timely model updates. Advanced anomaly detection methods, like unsupervised learning, complement traditional techniques by uncovering new fraud patterns. Ensemble learning, combining multiple models, bolsters system resilience against sophisticated fraud attempts. Regular evaluation using performance metrics ensures ongoing optimization, crucial for

meeting evolving fraud challenges in the financial sector.

1.5 PROPOSED SYSTEM

When fraud detection is taken into consideration, there is a plethora of factors that influence the system's accuracy. It is extremely important to apply various Machine learning techniques to standardize the transactions in this detection phenomena. We aimed to present the use of Random Forest for the high accuracy detection of Fraud Transactions using Dataset. Publicly available Credit Card Transactions consists of 2,84,000 were used in these phenomena, which involved the training of Random Forest and machine learning classifiers. Transactions data were considered separately in the experiments to evaluate the performances of models. In the proposed system for fraud detection, various machine learning techniques are applied to standardize transactions, aiming for high accuracy. Random Forest is highlighted as the primary method for detecting fraudulent transactions using a publicly available dataset comprising 284,000 credit card transactions. The approach involves training Random Forest and other machine learning classifiers on transaction data separately to assess model performance. Random Forest stands out as a widely used algorithm, leveraging dataset analysis, and splitting into training and testing data for evaluation. With an impressive 98% accuracy score, the results are presented through a confusion matrix, showcasing the algorithm's effectiveness in detecting fraud.

- This algorithm is one of the widely used machine learning algorithm.
- It analyses the dataset and splits into training and testing data.
- It has 98% of accuracy score and results are represented in confusion matrix.

2 LITERATURE SURVEY

2.1 INFERENCES FROM LITERATURE SURVEY

1) Comparative Analysis of Various Credit Card Fraud Detection Techniques

AUTHORS: Yashvi Jain, Namrata Tiwari, Shripriya Dubey, Sarika Jain

Authors have researched various techniques [1] for credit cards fraud detection such as support vector machines (SVM), artificial neural networks (ANN), Bayesian Networks, Hidden Markov Model, K-Nearest Neighbors (KNN) Fuzzy Logic system and Decision Trees. In their paper, they have observed that the algorithms k-nearest neighbor, decision trees, and the SVM give a medium level accuracy. The Fuzzy Logic and Logistic Regression give the lowest accuracy among all the other algorithms. Neural Networks, naive bayes, fuzzy systems, and KNN offer a high detention rate. The Logistic Regression, SVM, decision trees offer a high detection rate at the medium level. There are two algorithms namely ANN and the Naïve Bayesian Networks which perform better at all parameters. These are very much expensive to train.

There is a major drawback in all the algorithms. The drawback is that these algorithms don't give the same result in all types of environments. They give better results with one type of datasets and poor results with another type of dataset. Algorithms like KNN and SVM give excellent results with small datasets and algorithms like logistic regression and fuzzy logic systems give good accuracy with raw.

2) Credit card Fraud Detection based on Machine Learning Algorithms

AUTHORS: Heta Naik, Prashasti Kanikar

Heta Naik, Prashasti Kanikar, has done their research on various algorithms [2] like

Naïve Bayes, Logistic Regression, J48, and Adaboost. Naïve Bayes on among the classification algorithm. This algorithm depends upon Bayes theorem. Bayes's theorem finds the probability of an event that is occurring is given. The Logistic regression algorithm is like the linear regression algorithm. The linear regression is used for the prediction or forecasting the values. The logistic regression is mostly used for the classification task. The J48 algorithm is used to generate a decision tree and is used for the classification problem. The J48 is the extension of the ID3 (Iterative Dichotomieser). J48 is one of the most widely used and extensively analyzed areas in Machine Learning. This algorithm mainly works on constant and categorical variables. Adaboost is one of the most widely used machine learning algorithms and is mainly developed for binary classification. The algorithm is mainly used to boost the performance of the decision tree. This is also mainly used for the classification of the regression. The Adaboost algorithm is fraud cases to classify the transactions which are fraud and non-fraud. From their work they have concluded that the highest accuracy is obtained for both the Adaboost and Logistic Regression. As they have the same accuracy the time factor is considered to choose the best algorithm. By considering the time factor they concluded that the Adaboost algorithm works well to detect credit card fraud.

3) Fraud Detection System using ST and Whale Optimization Algorithm (WOA)

AUTHORS: Sahayasakila V, Kavya Monisha, Aishwarya, Sikhakolli Venkatav salakshiswshai Yasaswi

Authors have explained the Twain important algorithmic techniques [3] which are the Whale Optimization Techniques (WOA) and SMOTE (Synthetic Minority Oversampling Techniques). They mainly aimed to improve the convergence speed and to solve the data imbalance problem. The class imbalance problem is overcome using the SMOTE technique and the WOA technique. The SMOTE technique discriminates all the transactions which are synthesized are again re-sampled to check the data

accuracy and are optimized using the WOA technique. The algorithm also improves the convergence speed, reliability, and efficiency of the system.

4) **Credit Card Fraud Detection Using Machine Learning Models and Collating Machine Learning Models:**

AUTHORS: Navanushu Khare, Saad Yunus Sait

They have explained their work [4] on decision trees, random forest, SVM, and logistic regression. They have taken the highly skewed dataset and worked on such type of dataset. The performance evaluation is based on accuracy, sensitivity, specificity, and precision. The results indicate that the accuracy for the Logistic Regression is 97.7%, for Decision Trees is 95.5%, for Random Forest is 98.6%, for SVM classifier is 97.5%. They have concluded that the Random Forest algorithm has the highest accuracy among the other algorithms and is considered as the best algorithm to detect the fraud. They also concluded that the SVM algorithm has a data imbalance problem and does not give better results to detect credit card fraud.

5) **European credit card fraud data set:**

A study conducted by dataset in [5] presented a fraud detection method by using European credit card fraud data set where past transaction details of the customers are analyzed to extract the behavioral patterns. They have tried to overcome concept drift problems which causes high imbalance data set; The data set was highly imbalanced where there is total 284,807 transactions among which there are 492 i.e., 0.172% transactions are fraudulent transactions. Those transactions were made by a cardholder in a duration in 2 days. Sliding-Window method was used to extract features to find card holder's behavioral patterns. Clustering method was used to divide the cardholders into different clusters/groups based on their transaction amount. In this way a profile is built for every card holder. Then different classifiers are applied on three different groups (high, medium, and low using range partitioning); later rating scores are generated for every type of classifier

3 SYSTEM ANALYSIS

3.1 SYSTEM REQUIREMENTS

Hardware requirements

- Graphics Processing Unit (GPU).
- Intel Core i3 processor or above Software requirements:
- Windows 7 or above / Linux.
- Python 2.7 or above.
- Anaconda – Jupyter Notebook

3.2 FUNCTIONAL REQUIREMENTS:

The functions of software systems are defined in functional requirements and the behavior of the system is evaluated when presented with specific inputs or conditions which may include calculations, data manipulation and processing and other specific functionality.

Our system should be able to read the items data and pre-process the data.

- It should be able to analyses the items dataset.
- It should be able to group data based on hidden patterns.
- It should be able to assign a label based on its data groups.
- It should be able to split data into train set and test set.
- It should be able to train model using train set.
- It must validate trained model using test set.
- It should be able to classify the items dataset.

3.3 NON-FUNCTIONAL REQUIREMENTS:

Non-functional requirements, also referred to as quality attributes, are imperative in defining the overall quality and behavior of a system. Unlike functional requirements that specify what the system should do, non-functional requirements delineate how the system should perform and set constraints on its functionality. These attributes, such as performance, security, usability, and compatibility, are not features of the system but rather essential characteristics that determine its efficacy. They emerge as "developing" properties from the interaction and arrangement of various system components, influencing design and implementation choices. While functional requirements can be directly implemented through code, non-functional requirements cannot be explicitly programmed but must be considered throughout the development process. Derived from the needs and expectations of customers or stakeholders, non-functional requirements are delineated in the project specification to ensure that the final system aligns with desired quality standards. It's paramount to include only relevant non-functional requirements, as unnecessary additions can lead to system bloat and unnecessary complexity.

***3.3.1 Trade-offs and Prioritization*:** Non-functional requirements often involve trade-offs, where improving one aspect may come at the expense of another. For instance, enhancing system security might lead to a decrease in performance. Therefore, it's essential to prioritize non-functional requirements based on their importance to the stakeholders and the project's objectives.

***3.3.2 Measurability and Validation*:** Unlike functional requirements that can be easily validated through testing specific functionalities, non-functional requirements often require more sophisticated methods for validation. Metrics and benchmarks must be established to measure and validate adherence to these requirements. This ensures that the system meets the desired quality standards and provides a basis for continuous improvement throughout the development lifecycle.

3.4 PERFORMANCE REQUIREMENTS:

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements must be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use. Performance requirements are critical for assessing the effectiveness of a software application, focusing on its output. Requirement specification holds paramount importance in system analysis and design, as it sets the stage for creating a solution that seamlessly integrates into its intended environment. Users of the existing system are pivotal in providing these specifications, given their firsthand experience and ultimate utilization of the new system. Early identification of requirements is stressed, as it facilitates the design process and mitigates the challenge of modifying a system post-design. The necessity of aligning the system with user needs is emphasized, as designing a system that fails to do so serves little purpose. The outlined requirements encompass interfacing with existing systems, ensuring accuracy, and surpassing the capabilities of the current system, forming the foundational elements for effective system development.

3.4.1 Quantifiable Metrics: Performance requirements should include quantifiable metrics that allow for objective evaluation of the system's performance. These metrics might include response times, throughput, latency, and error rates, providing clear benchmarks for assessing performance.

3.4.2 Continuous Improvement: Performance requirements should not be static but should allow for continuous improvement over time. This involves monitoring the system's performance, identifying areas for enhancement, and implementing iterative

improvements to ensure that the system remains efficient and effective in meeting user needs.

3.5 PYTHON

Python is a general purpose, dynamic, high level, and interpreted programming language It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures. Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for Application Development. Python's syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development. Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles. Python is not intended to work in a particular area, such as web programming. That is why it is known as multipurpose programming language because it can be used with web, enterprise, 3D CAD, etc. We do not need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable. Python makes the development and debugging fast because there is no compilation step included in Python.

3.5.1 PYTHON'S FEATURE SET:

The factors that played an imp role in molding the final form of the language and are :

- Easy to code: Python is very easy to code. Compared to other popular languages like Java and C++, it is easier to code in Python.
- Easy to read: Being a high-level language, Python code is quite like English. Looking at it, you can tell what the code is supposed to do. Also, since it is dynamically-typed, it mandates indentation.
- Expressive: Suppose we have two languages A and B, and all programs that can be made in A can be made in B using local transformations. However, there are some programs that can be made in B, but not in A, using local transformations. Then, B is said to be more expressive than A. Python provides us with a myriad of constructs that help us focus on the solution rather than on the syntax.

- Free and Open-Source: Firstly, Python is freely available. You can download it from the link. Secondly, it is open source. This means that its source code is available to the public. You can download it, change it, use it, and distribute it. This is called FLOSS (Free/Libre and Open-Source Software.
- High- Level: This means that as programmers, we do not need to remember the system architecture. Nor do we need to manage the memory. This makes it more programmer- friendly and is one of the key python features.
- Portable: We can take one code and run it on any machine, there is no need to write different code for different machines. This makes Python a portable language.
- Interpreted: If you are any familiar with languages like C++ or Java, you must first compile it, and then run it. But in Python, there is no need to compile it. Internally, its source code is converted into an immediate form called bytecode. So, all you need to do is to run your Python code without worrying about linking to libraries, and a few other things. By interpreted, we mean the source code is executed line by line, and not all at once. Because of this, it is easier to debug your code. Also, interpreting makes it just slightly slower than Java, but that does not matter compared to the benefits it has to offer.
- Object-Oriented: A programming language that can model the real world is said to be object- oriented. It focuses on objects and combines data and functions. Contrarily, a procedure-oriented language revolves around functions, which are code that can be reused. Python supports both procedure-oriented and object-oriented programming which is one of the key python features. It also supports multiple inheritance, unlike Java. A class is a blueprint for such an object. It is an abstract.
- Extensible: If needed, you can write some of your Python code in other languages like C++. This makes Python an extensible language, meaning that it can be extended to other languages.
- Embeddable: We just saw that we can put code in other languages in our Python source code.

However, it is also possible to put our Python code in a source code in a different

language like C++ Sothis allows us to integrate scripting capabilities into us of the other language.

- Large Standard Library: Python downloads with a large library that you can use so you do not have documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and a lot of other functionality.
- GUI Programming: It also supports graphical user interface. Dynamically Typed Python is dynamically-typed. This means that the type for a value is decided at runtime, not in advance. This is why we don't need to specify the type of data while declaring it. Jupyter

  The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Language of choice

It supports more than 40 programming languages including Python, Spyder etc.

Share Notebooks

Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer.

Interactive Output

Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types.

3.5.2 MODULES USED IN PROJECT:

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

NumPy

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using NumPy which allows NumPy to integrate with a wide variety of databases seamlessly and speedily.

Seaborn

Seaborn is one of an amazing library for visualization of the graphical statistical plotting in Python. Seaborn provides many color palettes and defaults beautiful styles to make the creation of many statistical plots in Python more attractive.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics,

Statistics, analytics, etc. Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and I Python shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery. For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with I Python. For the power user, you have full control of line styles, font properties, axes properties, etc., via an object-oriented interface or via a set of functions familiar to MATLAB users.

Objective of Python Seaborn library

Seaborn library aims to make a more attractive visualization of the central part of understanding and exploring data. It is built on the core of the matplotlib. library and also provides dataset-oriented APIs.

Seaborn is also closely integrated with the Panda's data structures, and with this, we can easily jump between the various visual representations for a given variable to better understand the provided dataset.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

Categories of Plots in Python's seaborn library

Plots are generally used to make visualization of the relationships between the given variables. These variables can either be a category like a group, division, or class or can be completely numerical variables.

There are various categories of plots that we can create using the seaborn library. In

the seaborn library, the plot that we create is divided into the following various categories:

- Distribution plots: This type of plot is used for examining both types of distributions, i.e., univariate, and bivariate distribution.

- Relational plots: This type of plot is used to understand the relation between the two given variables.
- Regression plots: Regression plots in the seaborn library are primarily intended to add an additional visual guide that will help to emphasize dataset patterns during the analysis of exploratory data.
- Categorical plots: The categorical plots are used to deals with categories of variables and how we can visualize them.
- Multi-plot grids: The multi-plot grids are also a type of plot that is a useful approach is to draw multiple instances for the same plot with different subsets of a single dataset.
- Matrix plots: The matrix plots are a type of arrays of the scatterplots.

Required dependencies or prerequisites for the seaborn library:

We must have,

- Python installed with the latest version (3.6+).
- NumPy must be installed with version 1.13.3 or higher.
- SciPy must be installed with 1.0.1 or higher versions.
- Must have panda library with 0.22.0 or higher versions.
- stats model library must be installed with version 0.8.0 or higher.
- And should have matplotlib installed with 2.1.2 or higher versions

4 DESCRIPTION OF PROPOSED SYSTEM

4.1 FRAUD DETECTION PROBLEM

Fraud is defined as criminal deception. The purpose of fraud may be to obtain goods without paying or to obtain unauthorized funds from an account. Fraud can either be prevented or detected. In prevention, precaution activities are made to reduce the fraud. If the fraud prevention fails the problem of detection is taken for consideration. Fraud detection is identifying wrongful, suspect and illegitimate behavior. There are a lot of procedures used for fraud detection. The main target is defending the transactions from illegal use and maximizes the correct predictions. The credit card fraud can be of two types: offline fraud and online fraud. The fraud begins either with the theft of the credit card or the compromise of data associated with the account, including the card account number or other information that would routinely and necessarily be available to a merchant during a legitimate transaction. The offline fraud associated with the theft of the credit card. The physical card is stolen by the unauthorized person. Then purchases made by him by using the stolen card. He may use it to purchase until the usage of card is cancelled. If the user does not know about the theft, then it is a great loss to him and to the corresponding financial institutions. In online fraud, the physical card is not needed only the information about the card is enough to purchase. Thus, here the fraudster simply needs some important card details. Stealing the information from the user is called Identity Theft. In this type fraud the transactions are done through phone or internet. Generally, the genuine cardholder is not responsive if someone else has seen or stolen his card information.

4.1.1 FRAUD DETECTION PROCESS

The massive stream of payment request data set is quickly scanned by automatic tools named Fraud Detection System that determines which transactions to authorize. Here, Machine learning algorithms are employed to analyses all the authorized transactions and report the suspicious ones. The system then alerts the bank and generate report.

These reports are investigated by professionals who contact the cardholders to confirm if the transaction was genuine or fraudulent. The investigators provide feedback to the automated system which is used to train and update the algorithm to eventually improve the fraud-detection performance over time.

4.1.2 CREDIT CARD FRAUD DETECTION TECHNIQUES

Credit card fraud detection is designed to prevent any unauthorized credit card transactions from fraudsters and to recover losses and credibility for customers and businesses. Although there are better financial mechanisms, the fraudster is continually updating his techniques. Also, it makes the anti- credit card fraud techniques very challenging; the standard anti-credit card fraud methods available in the market today are listed below.

- Validation method through merchant trade :

The merchants often require a complete list of receipts to identify the user and have added tokenization techniques to protect credit card information by using the referenced card number instead of the current card number. It can make sure that they offer additional information like a PIN, zip code or card security code. Also, they may be requested to show them during the merchant transaction, and they are currently used by merchants to combat fraud (Contributors 2020).

- Geolocation of transactions by IP address :

Geolocation technology provides an absolute geographic location through the IP address of the computer where the order placed in a real-time e-commerce transaction which can identify areas with a high potential for fraud. It might allow merchants to attach authentication acne to transaction applications that vary widely in realistic examples to protect them from credit card fraud (FTC.gov 2012).

- Detect IP address countries and whether they are high-risk areas :

Detection system makes sure that the IP address country is the same as the billing address country. By using a fraud prevention service, the service can detect the IP address country for the customer placing the order. If the customer's billing and shipping address are in the UK (Duman et al. 2013) but the person placing the order

logged in from a Russian IP address, a more rigorous review is required, and anti-fraud precautions are often triggered. It is also always needed that orders shipped to international addresses scrutinized if the card or shipping address is in an area prone to credit card fraud.

- Detecting the use of anonymous mailboxes and proxies :

Many legitimate customers use free email addresses because they are convenient and economical.

Indeed, most fraudsters use free email addresses to remain anonymous. Detecting new domain registrations for email addresses is one of the most important ways to do a better job of fighting fraud (Bhatla et al. 2003). Secondly, anonymous proxy servers allow Internet users to hide their actual IP addresses. The primary purpose of using a proxy server is to remain anonymous or to avoid detection so people need to save the list of proxies as a web service to prevent credit card fraud.

- Using Neural Networks to Detect Credit Card Payment Fraud :

Most of the existing techniques based on deep learning and oversampling algorithms for credit card fraud detection. The Long Short Term Memory Networks (LSTM) fraud detection model for serial classification of transaction data and integration of synthetic minority class oversampling. The Smote and the k-Nearest Neighbor (KNN) classification algorithm design and build a KNN-model, STM based fraud detection network model which can Improve fraud detection performance by continuously filtering out security-generating samples through KNN discriminant classifiers (Mae's et al. 2002).

- Machine learning detection :

They are using Machine Learning Classification Algorithms to Detect Credit Card Fraud. Machine learning is a very effective way to detect fraudulent transactions if his performance is good enough because he determined by choice of features, the training of the data drink testing, and the classification methods of machine learning. All these factors contribute to different generation rates. Many studies have shown that using machine learning classification algorithms to detect credit card fraud has resulted in better accuracy. They have also compared the results of different algorithms and other

studies and agreed that machine learning detection is the right choice.

4.1.3 CHALLENGING ISSUES IN FRAUD DETECTION

The datasets are extreme imbalance and highly skewed. The genuine transactions dominate than fraudulent transactions. The fraudulent events occur rarely. So, it is difficult to find the fraudulent. If the fraudulent transaction is considered as legal then it will cause great loss. The huge number of datasets and the dimensionality is very high. It is not an easy process to handle the massive amount of data efficiently. The scalable machine learning system is needed to process the large amount of data. The real data is not shared for the number of reasons such as to maintain the privacy of the user. Generally, the misclassification cost is high for these detections. Efficient measure should take to reduce the misclassification cost. Fraud detection is a multifaceted endeavor fraught with several daunting challenges. One of the primary hurdles lies in the ever-evolving sophistication of fraudulent techniques. Fraudsters adeptly leverage advanced technologies such as artificial intelligence and machine learning, perpetually outpacing conventional detection methods. Additionally, the integrity and scale of data pose significant obstacles. While effective detection necessitates extensive data from diverse sources, ensuring its quality, accuracy, and integration remains a formidable task. Compounded by the prevalence of imbalanced datasets, where genuine transactions far outnumber fraudulent ones, the efficacy of detection algorithms becomes compromised. Real-time detection, imperative for timely intervention, presents its own set of challenges. Balancing the need for rapid analysis with low latency in processing voluminous data without compromising accuracy is an intricate balancing act. Moreover, fraudsters' adaptive behavior, privacy concerns surrounding sensitive customer data, and compliance with stringent regulatory standards further exacerbate the complexity of fraud detection. Addressing these challenges demands a holistic approach, integrating advanced technologies, robust data management practices, and continuous monitoring while fostering collaboration across industries to stay ahead of evolving fraud schemes and safeguard financial systems.

4.2 DOMAIN INTRODUCTION

4.2.1 MACHINE LEARNING

In general context, machine learning can be defined as a field in artificial intelligence that provides the system the capability to learn from the experience automatically without the human intervention and aims to predict the future outcomes as accurate as possible utilizing various algorithmic models. Machine Learning is very different than the conventional computation approaches, where systems are explicitly programmed to calculate or solve a problem. Machine learning deals with the input data that are used to train a model where the model learns different patterns in the input data and uses that knowledge to predict unknown results. The application of machine learning is incredibly vast. It is used in various applications like the spam filter, weather prediction, stock market prediction, medical diagnosis, fraud detection, autopilot, house price prediction, face detection, and many more. Typically, machine learning has three categories: supervised, unsupervised and reinforcement learning. This thesis deals with supervised learning and we will discuss it in the next section. For now, we can define supervised learning as the approach where the model is trained with both input and output labels. In contrast, unsupervised learning is where the dataset has input labels, (i.e., a model is trained with unlabeled data), from which it learns different patterns and structures. Usually, it is implemented in applications like visual recognition, robotics, speech recognition and so on. Reinforcement learning deals with learning how to obtain a complex goal by maximizing along a specific dimension step by step.

4.2.2 SUPERVISED LEARNING

Supervised learning can be defined as a machine learning approach in which both input and output labels are provided to the model to train. The supervised model uses the input and output labeled data for training, and it extracts the patterns from the input data. These extracted patterns are used to support future judgments. Supervised learning can be formally represented as follows: 7 Y = f(x) where x represents the input

variables, Y denotes an output variable and f(X) is a mapping function. The goal is to approximate mapping function such that when an unseen input is given to the mapping function, it can predict the output variable (Y) correctly. Furthermore, supervised learning has two sub-categories: classification and regression. In a classification problem, the output variable is a category, (e.g., fraud or genuine, rainy or sunny, etc.). In a regression problem, the output variable is

4.2.3 METHODOLOGY

The approach uses the latest machine learning algorithms to detect anomalous activities, called outliers. The basic rough architecture diagram can be represented with the following. When looked at in detail on a larger scale along with real life elements, the full architecture diagram can be represented as follows:

First, we obtained our dataset from Kaggle, a data analysis website which provides datasets. Inside this dataset, there are 31 columns out of which 28 are named as v1-v28 to protect sensitive data.

The other columns represent Time, Amount and Class. Time shows the time gap between the first transaction and the following one. Amount is the amount of money transacted. Class 0 represents a valid transaction and 1 represents a fraudulent one.

We plot different graphs to check for inconsistencies in the dataset and to visually comprehend it: This shows that the number of fraudulent transactions is much lower than the legitimate ones. this shows the times at which transactions were done within two days. The least number of transactions were made during night time and highest during the days. This graph represents the amount that was transacted. Most transactions are relatively small.

After checking this data set, we plot a histogram for every column. This is done to get a graphical representation of the data set which can be used to verify that there are no missing any values in the data set. This is done to ensure that we don’t require any missing value imputation and the machine learning algorithms can process the dataset smoothly.

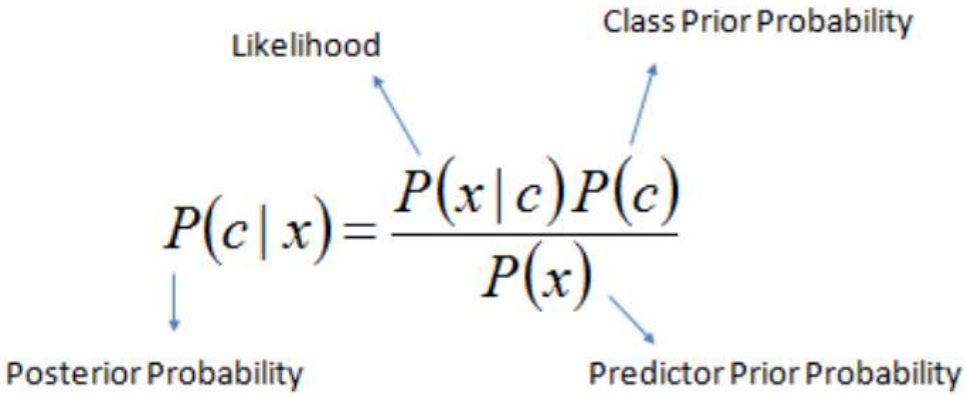
After this analysis, we plot a heat map to get a colored representation of the data and to study the correlation between out predicting variables and the class variable. This heat map is shown below: The data set is now formatted and processed. The time and amount column are standardized and the Class column is removed to ensure fairness of evaluation. The data is processed by a set of algorithms from modules. The following module diagram explains how these algorithms work together. This data is fit into a model and the following outlier detection modules are applied on it:

- Random Forest Algorithm
- Ada-Boost Classifier

4.3 RANDOM FOREST ALGORITHM

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. This model is basically an ensemble classifier, i.e., combining classifiers that use and combines many decision tree classifiers. The main agenda behind using multiple trees is to be able to train the trees enough, such that contribution from each of them comes in the form of a model. After the generation of the tree, the output is combined through majority. It uses multiple decision trees so that the dependence of each of them is on a particular data set possessing similar distribution throughout the tree . This model has the quality of

efficiently balancing errors in a class population of unbalanced data sets. It can be used to solve both classification as well as regression problems. In simple language, the random forest builds multiple decision trees and combines them to improve the performance of the model. Moreover, the random forest uses random subsets of features. For example, if there are 50 features in the data, a random forest will only choose a certain number of them, let us say 10, to train on each tree. Thus, each tree will have 10 random features that will be used for training including finding the best split of each node of the tree. Once we have the collection of decision trees, the results of each tree will be aggregated to get the result (vote). The model trained in such a way will ensure generalization since not one, but multiple decision trees are used for making the decision, and moreover, each tree is trained with different subsections of data. This algorithm is widely used in E-commerce, banking, medicine, the stock market, etc. For example: In the Banking industry it can be used to find which customer will default on the loan. It is highly stable as the average answers given by many trees are taken. It maintains diversity as all the attributes are not considered while making each decision tree though it is not true in all cases. The Random Forest algorithm has been found to provides a good estimate of the generalization error and to be resistant to overfitting.



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

FIG:4.3.1 Posterior Probability

Random forest ranks the importance of variables in a regression or classification problem in a way

Steps involved in random forest algorithm:

- In Random Forest n number of random records are taken from the data set having k number of records.
- Individual decision trees are constructed for each sample.
- Each decision tree will generate an output.
- Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

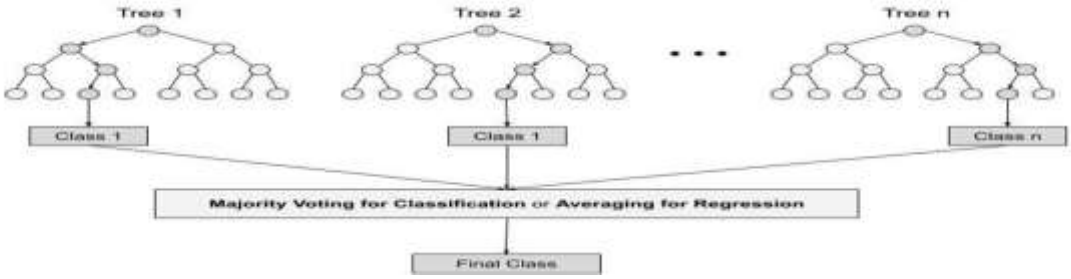


FIG:4.3.2 Random Forest Algorithm

4.4 Ada-Boost Classifier

AdaBoost, short for Adaptive Boosting, stands out as a robust ensemble learning technique widely employed in classification tasks. Its strength lies in its ability to iteratively combine multiple weak learners, typically shallow decision trees known as "stumps," to construct a powerful classifier. Initially, each data point in the training set is assigned equal weight. The algorithm then proceeds by training a weak learner on the data and calculating the error rate, comparing its predictions against the actual labels. The weight of each weak learner is subsequently determined based on its error rate, with lower errors receiving higher weights. AdaBoost then adjusts the weights of misclassified samples, enhancing their significance in subsequent iterations. This iterative process continues, with each subsequent learner focusing more on the misclassified samples from the

previous iterations. Ultimately, AdaBoost combines all weak learners through weighted majority voting, where each learner's weight corresponds to its accuracy. The resulting classifier effectively harnesses the collective predictive power of the weak learners, offering robust performance even in complex classification scenarios. Despite its effectiveness, AdaBoost may exhibit sensitivity to noisy data and outliers, warranting careful preprocessing steps to ensure optimal performance.

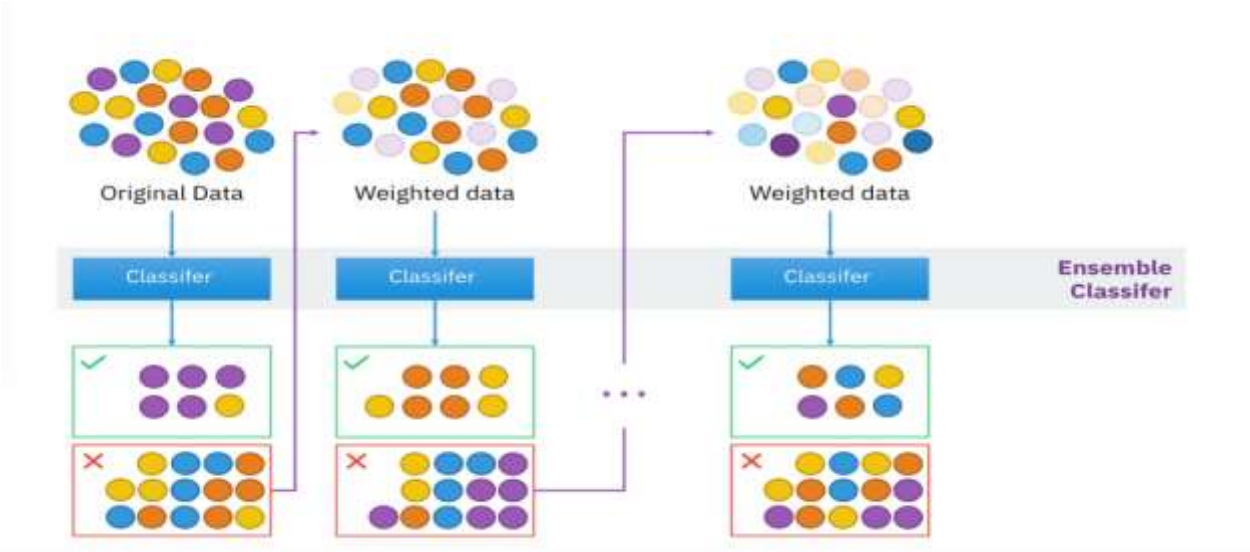


FIG: 4.4.1 Ada-Boost Classifier

Steps involved in Ada-Boost Classifier:

- Initialize Sample Weights: Initially, each sample in the training dataset is assigned an equal weight.
- Train Weak Learner: A weak learner (often a decision tree stump) is trained on the data. It aims to minimize the classification error but doesn't have to be particularly accurate.
- Compute Error: After training each weak learner, AdaBoost computes the error rate by comparing the predicted labels with the actual labels of the training data.

- Compute Learner Weight: The weight of the weak learner is calculated based on its error rate. Lower error rates result in higher weights assigned to the weak learner.
- Update Sample Weights: AdaBoost increases the weights of misclassified samples, making them more important for subsequent learners to focus on.
- Repeat: Steps 2-5 are repeated iteratively, with each subsequent learner focusing more on the misclassified samples from the previous iterations.
- Combine Learners: Finally, AdaBoost combines all the weak learners through weighted majority voting, where the weight of each learner depends on its accuracy.

These steps are repeated for a predefined number of iterations or until a certain level of accuracy is achieved. The resulting ensemble model effectively combines the predictions of multiple weak classifiers, resulting in a strong classifier with improved performance.

5 IMPLEMENTATION

5.1 SYSTEM ARCHITECTURE

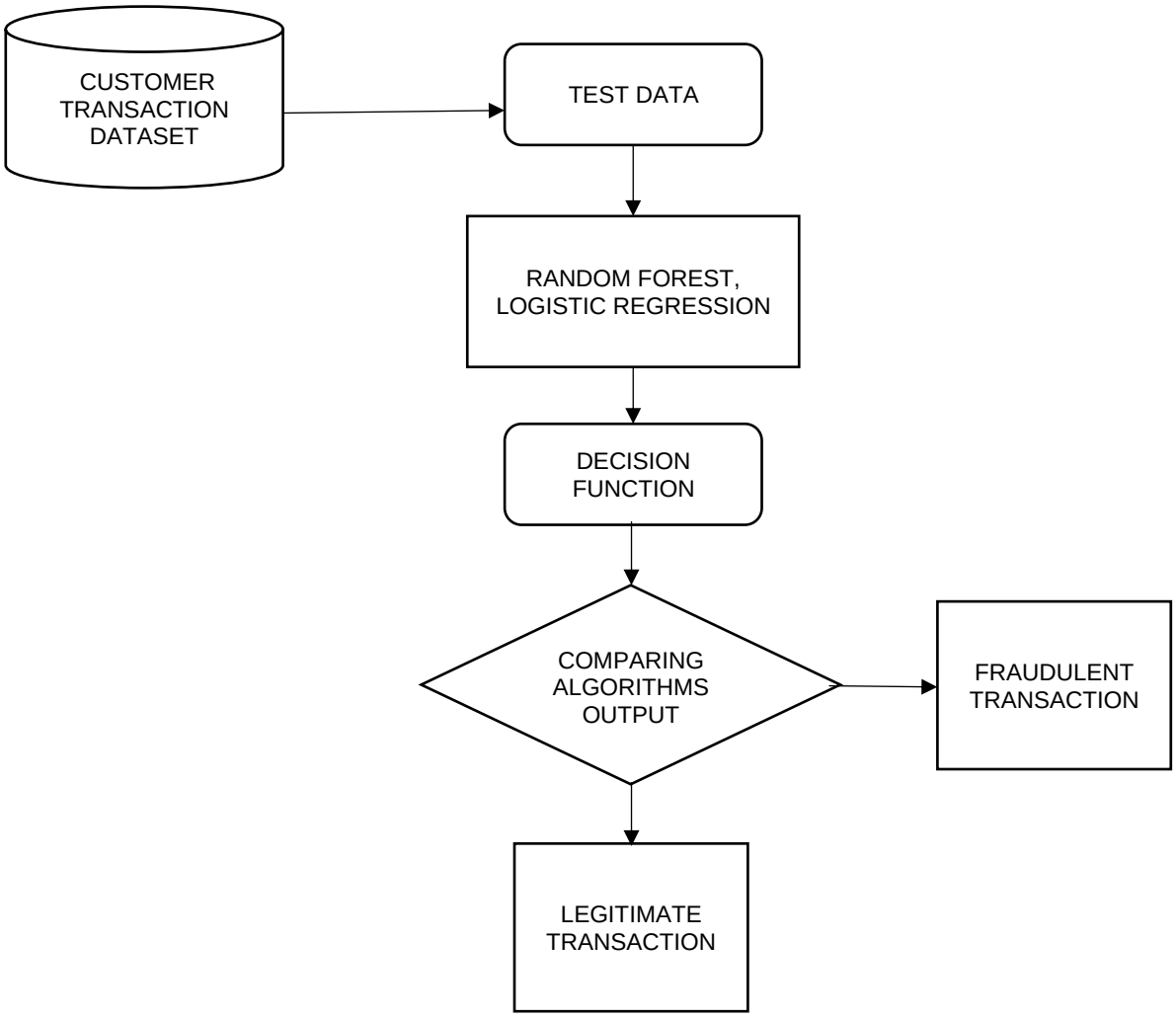


FIG: 5.1.1 ARCHITECTURE DIAGRAM

- Data collection: Customer transaction data is collected, which could include information such as the amount of the transaction, the time and location of the transaction, and the customer's past transaction history.

- Test data: The customer transaction data is then split into two sets: a training set and a test set. The training set is used to train the machine learning models, while the test set is used to evaluate the performance of the models.
- Feature selection: A subset of features is selected from the customer transaction data. These features are the ones that are most likely to be relevant to predicting whether a transaction is fraudulent.
- Model training: Three machine learning models are trained on the training data: a random forest, a logistic regression, and an unspecified algorithm. These models learn to identify patterns in the data that are associated with fraudulent transactions.
- Model comparison: The performance of the three models is compared on the test data. The model that performs the best is then used to make predictions about whether new transactions are fraudulent.
- Decision making: When a new transaction occurs, the system uses the best machine learning model to predict whether the transaction is fraudulent. If the transaction is predicted to be fraudulent, the system may take some action, such as blocking the transaction or contacting the customer for verification.

5.2 UML DIAGRAMS

UML is an acronym that stands for Unified Modelling Language. Simply put, UML is a modern approach to modelling and documenting software. In fact, it is one of the most popular businesses process modelling techniques.

It is based on diagrammatic representations of software components. As the old

proverb says: “a picture is worth a thousand words”. By using visual representations, we can better understand possible flaws or errors in software or business processes.
UML was created because of the chaos revolving around software development and documentation. In the 1990s, there were several different ways to represent and document software systems. The need arose for a more unified way to visually represent those systems and as a result, in 1994-1996, the UML was developed by three software engineers working at Rational Software. It was later adopted as the standard in 1997 and has remained the standard ever since, receiving only a few updates.

5.2.1 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a pivotal tool derived from use-case analysis, serving to visually depict the interactions between actors and the system. It encapsulates the system's functionality by delineating actors, their respective goals (as use cases), and any interdependencies between these functionalities. The primary aim of a use case diagram is to provide a clear overview of the system's functions vis-a-vis its actors, elucidating which tasks the system performs for each actor involved. Additionally, these diagrams can effectively portray the roles assumed by various actors within the system, enhancing comprehension of system behavior and functionality. Through this visual representation, stakeholders gain insights into the system's operational dynamics, facilitating communication and aiding in system design and analysis processes. A well-designed use case diagram not only outlines the interactions between actors and the system but also serves as a foundation for further system development and refinement. By visually representing the system's functionalities and the actors' roles, use case diagrams facilitate effective communication among stakeholders, including developers, designers, testers, and end-users. Moreover, these diagrams play a crucial role in requirements elicitation and validation processes, ensuring that the system meets stakeholders' needs and expectations. Additionally, use case diagrams can evolve alongside the system, accommodating changes in requirements, user roles, or system functionality.

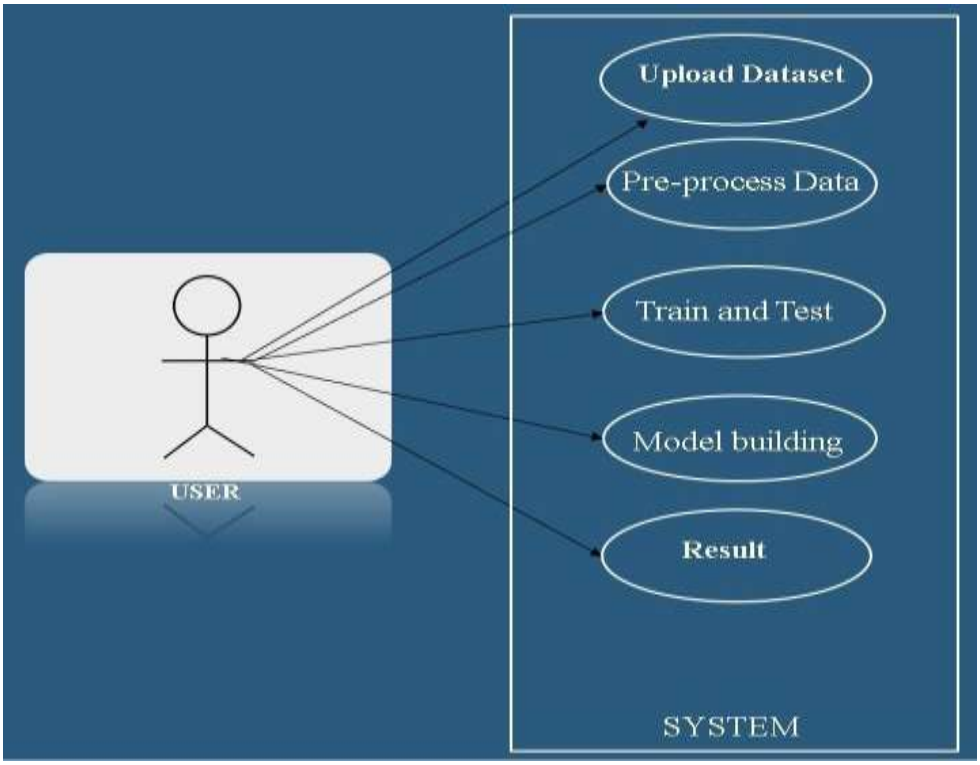


FIG: 5.2.1 Use Case Diagram for Credit Card Fraud Detection

5.2.2 CLASS DIAGRAM

The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. A class with three sections, in the diagram, classes is represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake.

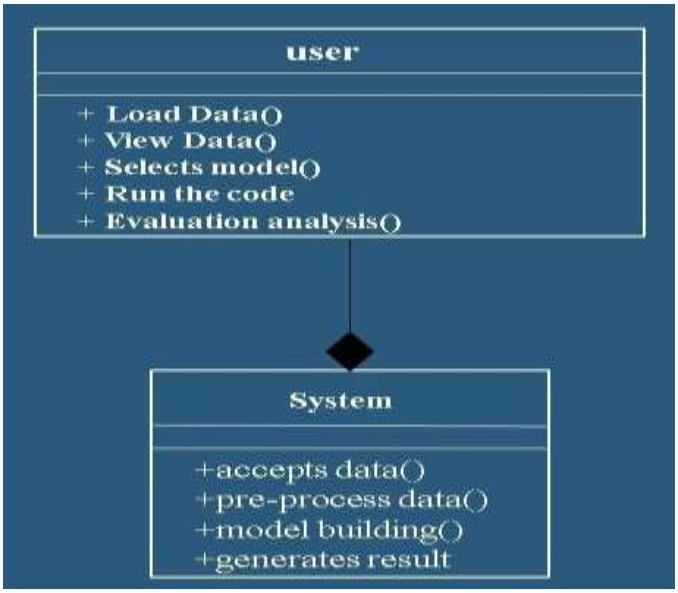


FIG: 5.2.2 Class Diagram for Fraud Detection

5.2.3 DATA FLOW DIAGRAM:

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation. Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes flow of data through a system to perform certain functionality of a business. The physical data flow diagram describes the implementation of the logical data flow.

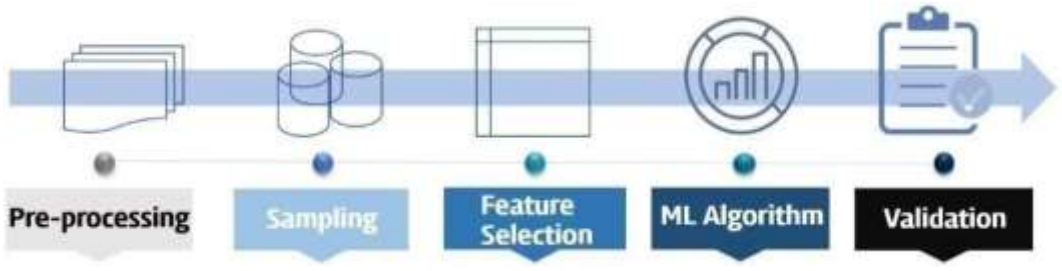


FIG: 5.2.3 Data Flow Diagram for Fraud Detection

6 RESULT AND DISCUSSION

6.1 DATA COLLECTION AND EXPLORATION

- Dataset plays a vital role in any software related projects.
- This dataset consists of credit card transaction records.
- It is collected from the Kaggle community having approximately twenty-eight lakh transactions in it.
- The dataset consists of 31 parameters. Due to confidentiality issues, 28 of the features are the result of the PCA transformation. “Time’ and “Amount” are the only aspects that were not modified with PCA. • There are a total of 284,807 transactions with only 492 of them being fraud. So, the label distribution suffers from imbalance issues.

We can explore the data and its features through visualization by importing necessary packages

Figure 6.1 Transaction Class Distribution

All the necessary libraries and packages are installed through Conda. Libraries and packages like: NumPy, Pandas, Matplotlib, Sckit-Learn, TensorFlow, keras. We imported Pandas NumPy and pandas to our Jupiter notebook and feed our csv data into pandas Data frame through the function called read_csv.Matplotlib library is used to plot 2D figures in a variety of formats. It makes it easy to visualize data cause a picture worth a thousand words. It's a very powerful plotting library useful for peoples using python and NumPy. Pyplot is the most used module of matplotlib. It provides an interface like MATLAB but instead it uses Python.

6.2 DATA PRE-PROCESSING

Data pre-processing is pivotal in optimizing the performance of machine learning and data mining algorithms, involving the transformation of raw data into a machine-readable format. Raw data typically exhibit issues like incompleteness, noise, and inconsistencies. In the context of the Credit Card Dataset, common pre-processing steps include cleaning to handle missing values, errors, and outliers, ensuring data quality, and sampling techniques to address class imbalance problems, enhancing model performance. Moreover, for categorical data, label encoding serves as an efficient tool to convert categorical values into numeric format, facilitating model training and analysis.

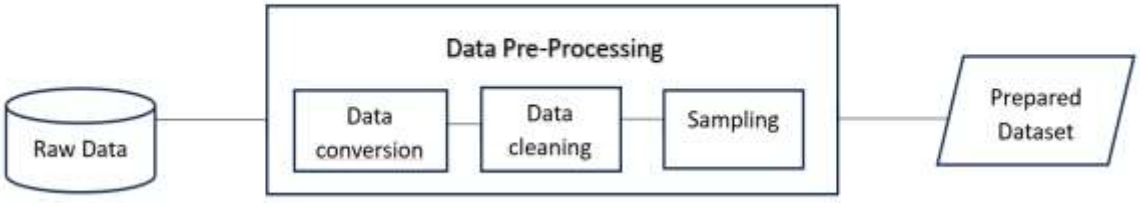


FIG: 6.2.1 Data Pre-Processing

The dataset being utilized in this project comprises credit card transaction records obtained from the Kaggle community, containing around twenty-eight lakh transactions. The dataset encompasses 31 parameters, with 28 of them subjected to PCA transformation due to confidentiality concerns, while "Time" and "Amount" remain unaltered. Among the 284,807 transactions, merely 492 are identified as fraud, resulting in an imbalanced label distribution. To explore the data and its features visually, essential packages such as NumPy, Pandas, Matplotlib, Scikit-Learn, TensorFlow, and Keras have been installed via Conda. Specifically, Pandas and NumPy are employed to handle data within a Jupiter notebook environment, while Matplotlib facilitates the visualization of 2D figures, offering a powerful tool for data analysis and interpretation.

6.3 TRAINING AND TESTING

The train/test split it is an important step in machine learning process. Where the model is split into two sets: training and testing set. Usually, the major part of the dataset is reserved for the training and a small part for testing. The train data will be used to create the machine learning model, and the test data is used to check the accuracy of the model.

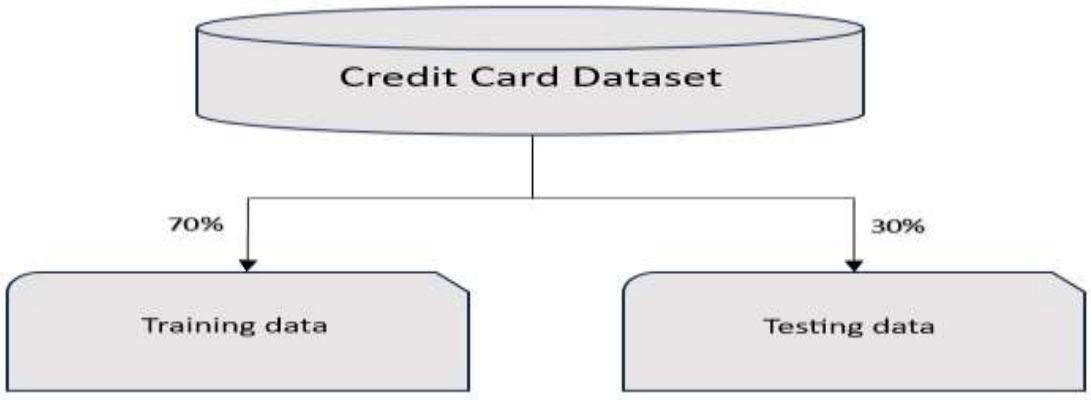


FIG:6.3.1 Training and Testing

6.4 MODEL BUILDING

from sklearn import ensemble

Model Building is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. Model building is the vital part of the detection process. The train and test data undergo through the appropriate machine learning algorithm to build a efficient model. Here the trained data go through the two machine learning algorithms such as Random Forest and adaboost algorithms which is represented in the form of confusion matrix consists of accuracy score, precision,F-1 score and recall. The best model is chosen after the comparison of two performance metrics.

6.5 RESULT ANALYSIS

from sklearn. metrics import accuracy score, classification report, confusion matrix

It is the final stage of the process in which we compare and analyses the model values to select the best one. In the report below, we've some metrics to check the model performance, let us make a brief explanation about how to understand those values, and evaluate the machine learning model. Before explaining the mathematical formulas, we will explain some terms and what it does represent.

- **TN — True Negative:** when a case was negative and predicted negative;
- **TP — True Positive:** when a case was positive and predicted positive;
- **FN — False Negative:** when a case was positive but predicted negative;
- **FP — False Positive:** when a case was negative but predicted positive.

Accuracy — What percent of predictions the model did correctly?

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$$

Precision — What percent of the predictions were correct?

$$Precision = \frac{TP}{TP + FP}$$

Recall — What percent of the positive cases the model catches?

$$Recall = \frac{TP}{TP + FN}$$

F1-Score — What percent of positive predictions were correct?

$$F1 - Score = \frac{2(Recall * Precision)}{Recall + Precision}$$

METRICS	ADABOOST	RANDOM FOREST
ACCURACY	0.991	0.999
PRECISION SCORE	0.82	0.93
F1-SCORE	0.77	0.88
RECALL	0.72	0.83

FIG: 6.5.1 RESULT ANALYSIS

7 CONCLUSION

7.1 CONCLUSION

While Random Forest is indeed a powerful algorithm and often yields excellent results in various machine learning tasks, including fraud detection, it's essential to approach such conclusions with caution. Firstly, stating that Random Forest "works better than any other machine learning algorithms" may be an overgeneralization. The performance of algorithms can vary depending on the specific dataset, its characteristics, and the problem domain. What works best for one dataset or problem may not necessarily be the best for another. Additionally, achieving 100% satisfactory results in fraud detection is incredibly challenging, if not impossible. Fraudsters continually adapt and evolve their tactics, making it a cat-and-mouse game between fraud detection systems and perpetrators. While machine learning algorithms can significantly enhance fraud detection capabilities, there will always be instances where fraud goes undetected. Moreover, while increasing the size of the training dataset and applying various preprocessing techniques can improve model performance, there are still limitations. The quality and representativeness of the data, feature engineering, model tuning, and other factors play crucial roles in determining the effectiveness of the model. Random Forest is indeed known for its accuracy, scalability, and ability to handle high-dimensional data. However, it's just one of many tools in the machine learning toolbox. Depending solely on one algorithm may limit your ability to explore alternative approaches that could potentially yield better results in specific contexts.In summary, while Random Forest can be highly effective in credit card fraud detection, it's essential to approach algorithm selection and model evaluation with a nuanced understanding of the problem domain, data characteristics, and the limitations of the techniques employed. Additionally, continuous improvement and adaptation of fraud detection systems are necessary to stay ahead of evolving fraud tactics. In creating a framework for credit card fraud detection, meticulous attention is given to data quality

and feature engineering during collection and preprocessing. Model selection involves evaluating diverse algorithms and optimizing performance through techniques like cross-validation. Ensemble methods are explored to leverage the strengths of multiple models, while continuous improvement ensures adaptability to emerging fraud tactics. Deployment emphasizes rigorous monitoring and adherence to regulations, supported by transparent documentation and communication to foster accountability and collaboration. This systematic approach enables organizations to develop robust fraud detection systems capable of effectively combating evolving fraudulent activities.

7.2 FUTURE WORK

In the realm of credit card fraud detection, future endeavors could focus on advancing machine learning techniques, enabling real-time detection systems, and integrating behavioral analysis methods. Exploring unsupervised learning and graph-based approaches could uncover novel fraud patterns and networks, while prioritizing model interpretability and robustness against adversarial attacks. Additionally, fostering cross-industry collaboration and upholding ethical standards are paramount for developing effective and responsible fraud detection solutions. By pursuing these avenues, the field can stay ahead of evolving fraud tactics and better protect consumers and businesses from financial harm. In addition to advancing machine learning techniques and real-time detection systems, attention should be given to fostering collaboration among industry stakeholders, leveraging geospatial analysis, and integrating blockchain technology for enhanced transaction transparency. Behavioral biometrics can provide an extra layer of security, while regulatory compliance and customer education are essential for maintaining trust and protecting sensitive data. By combining AI with human expertise and addressing these additional aspects, the field of credit card fraud detection can evolve to meet the challenges of tomorrow's financial landscape.

REFERENCE

[1] Yashvi Jain, Namrata Tiwari, Shripriya, Dubey, Sarika Jain: A Comparative Analysis of Various Credit Card Fraud Detection Techniques, International Journal of Recent Technology and Engineering (IJRTE) 2019 ISSN: 2277-3878, Volume-7 Issue-5S2, January.

[2]. Heta Naik, Prashasti Kanikar: Credit card Fraud Detection based on Machine Learning Algorithms, International Journal of Computer Applications (0975 – 8887) Volume 182 – No. 44, March 2019.

[3]. Sahayasakila.V, D. Kavya Monisha, Aishwarya, Sikha kolli Venkatavisalakshi Sesh Sai Yasaswi: Credit Card Fraud Detection System using Smote Technique and Whale Optimization Algorithm, International Journal of Engineering and Advanced Technology (IJEAT), June 2019 , ISSN: 2249-8958, Volume8 Issue-5.

[4]. Navan Shu Khare, Saad Yunus Sait: Credit Card Fraud Detection Using Machine Learning Models and Collating Machine Learning Models, International Journal of Pure and Applied Mathematics, 2018, Volume 118 No. 20, 825-838 ISSN: 1314-3395. [5] https://www.kaggle.com/mlg-ulb/creditcardfraud

[5]. E Ileberi, Y Sun, Z Wang ; A machine learning based credit card fraud detection using the GA algorithm for feature selection, Journal of Big Data (2022) 9:24 https://doi.org/10.1186/s40537-022-00573-8

[6]. A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection , A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection, Received January 14, 2022, accepted January 27, 2022,

date of publication January 31, 2022, date of current version February 15, 2022

[7]. D Ge, J Gu, S Chang, JH Cai : Credit card fraud detection using lightgbm model , 2020 International Conference on E-Commerce and Internet Technology (ECIT) , Year: 2020, Pages: 232-236

[8]. Naresh Kumar Trivedi1 , Sarita Simaiya2 ,*Umesh Kumar Lilhore3 , Sanjeev Kumar Sharma: An Efficient Credit Card Fraud Detection Model Based on Machine Learning Methods. International Journal of Advanced Science and Technology Vol. 29, No. 5, (2020), pp. 3414 - 3424

[9]. Fawaz Khaled Alarfaj; Iqra Malik; Hikmat Ullah Khan; Naif Almusallam: Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms, Received March 20, 2022, accepted April 8, 2022, date of publication April 12, 2022, date of current version April 18, 2022

[10] R Sailusha, V Gnaneswar, R Ramesh : Credit card fraud detection using machine learning , 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)

APPENDIX

A. SOURCE CODE

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(2)
from sklearn.metrics import confusion_matrix, classification_report,
roc_auc_score, precision_recall_curve, roc_curve, auc,
average_precision_score

data=pd.read_csv('creditcard.csv')

data. head ()
data. tail ()
data.info ()
data. shape
count_classes = pd.value_counts(data['Class'], sort = True)
count classes. plot (kind = 'bar', rot=0)
plt. title ("Transaction Class Distribution")
plt. xticks (range (2))
plt. xlabel("Class")
plt. ylabel("Frequency")
fraud = data[data['Class’] ==1]
normal = data[data['Class’] ==0]
print (fraud. shape, normal. shape)
f, (ax1, ax2) = plt. subplots (2, 1, sharex=True)
f. suptitle ('Amount per transaction by class')
bins = 50
ax1.hist(fraud. Amount, bins = bins)
```

```
ax1.set_title('Fraud')
ax2.hist(normal. Amount, bins = bins)
ax2.set_title('Normal')
plt. xlabel ('Amount ($)')
plt. ylabel ('Number of Transactions')’
plt. xlim ((0, 20000))
plt. yscale('log')
plt. show ();

data. is null (). values. Any ()
fraud. Amount. Describe ()
normal. Amount. Describe ()
data. group by('Class'). mean ()

normal_sample = normal.sample(n=492)
new_dataset = pd. concat ([normal_sample, fraud], axis=0)
new_dataset. head ()
new_dataset. tail ()
new_dataset['Class'].value_counts ()
new_dataset. group by('Class'). mean ()
X = new_dataset. drop (columns='Class', axis=1)
Y = new_dataset['Class']
print(X)
print(Y)

from sklearn. model selection import train_test_split
X = data. iloc[:,:30]
Y = data. iloc[:,30]
print (X. shape)
print (Y. shape)
```

```
X_train, X_test, Y_train, Y_test = train_test_split (X, Y, test_size=0.25,
random_state=2)
print (X_train. shape)
print (X_test. shape)
print (Y_train. shape)
print (Y_test. shape)

# Creating classifier Object
ada = ensemble. AdaBoostClassifier ()
#Fitting the classifier to training data
ada.fit (X_train, Y_train)
ada_pred = ada. predict(X_test)
print ("Training Score: %f"%ada. score (X_train, Y_train))
print ("Testing Score: %f"%ada. score(X_test,Y_test))
from sklearn.metrics import accuracy_score, classification_report,confusion_matrix
print (accuracy score (Y_test, ada_pred))
print(classification_report(Y_test,ada_pred))
print(confusion_matrix(Y_test,ada_pred))
from sklearn. ensemble import RandomForestClassifier
# Random forest model creation
rfc = RandomForestClassifier ()
rfc.fit (X_train, Y_train)
# Predictions r
fc_pred = rfc. predict (Test)
print ("Training Score: %f"%rfc. score (X_train, Y_train))
print ("Testing Score: %f"%rfc. score (X_test, Y_test))

print (accuracy score (Y_test, rfc_pred))
print (classification report (Y_test, rfc_pred))
print (confusion matrix (Y_test, rfc_pred))
```

```
#Evaluating the classifier
from sklearn. metrics import classification report, accuracy score, precision score,
recall score, f1_score, roc_curve, roc_auc_score
from sklearn. metrics import confusion matrix
n_outliers = len(fraud)
errors = (rfc_pred! = Y_test). sum ()
print ("The model used is Random Forest classifier")
acc= accuracy score (Y_test, rfc_pred)
print ("The accuracy is {}". format(acc))
prec= precision score (Y_test, rfc_pred)
print ("The precision is {}". format(prec))
rec= recall score (Y_test, rfc_pred)
print ("The recall is {}". format(rec))
f1= f1_score (Y_test, rfc_pred)
print ("The F1-Score is {}". format(f1))
LABELS = ['Normal', 'fraud']
conf_matrix = confusion matrix (Y_test, rfc_pred)
plt. figure (fig size= (9, 9))
sns. heatmap (conf_matrix, xticklabels=LABELS, yticklabels=LABELS, annot=True,
fmt="d"); plt. title ("Confusion matrix")
plt. ylabel ('True class')
plt. xlabel ('Predicted class')
plt. show ()
plt. figure (fig size= (9, 7))
print ('{}: {}'. format ("Random Forest", errors))
print (accuracy score (Y_test, rfc_pred))
print (classification report (Y_test, rfc_pred))
false_positive_rate, true_positive_rate, thresholds = roc_curve (Y_test, rfc_pred)
roc_auc_rf = auc (false_positive_rate, true_positive_rate)
print ("roc_auc", roc_auc_rf)
```

B. SCREENSHOTS

```
In [1]:
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(2)
from sklearn.metrics import confusion_matrix, classification_report, roc_auc_
```

FIG: B.1 IMPORTING PACKAGES AND DATASET

```
In [2]:
data=pd.read_csv('creditcard.csv')
```

```
In [3]:
data
```

Out[3]:

	Time	V1	V2	V3	V4	V5	V6	V7
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941
...	...	...	...	...	...	...	...	...
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006

284807 rows × 31 columns

FIG: B.2 DATASET

In [4]:
```
data.head()
```

Out[4]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533

5 rows × 31 columns

In [5]:
```
data.tail()
```

Out[5]:

	Time	V1	V2	V3	V4	V5	V6	V7
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006

5 rows × 31 columns

FIG: B.3 DATASET EXPLORATION

```
In [6]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
```

FIG: B.4 DATASET INFORMATION

```
In [8]: count_classes = pd.value_counts(data['Class'], sort = True)
        count_classes.plot(kind = 'bar', rot=0)
        plt.title ("Transaction Class Distribution")
        plt.xticks (range (2))
        plt.xlabel("Class")
        plt.ylabel("Frequency")

Out[8]: Text(0, 0.5, 'Frequency')
```

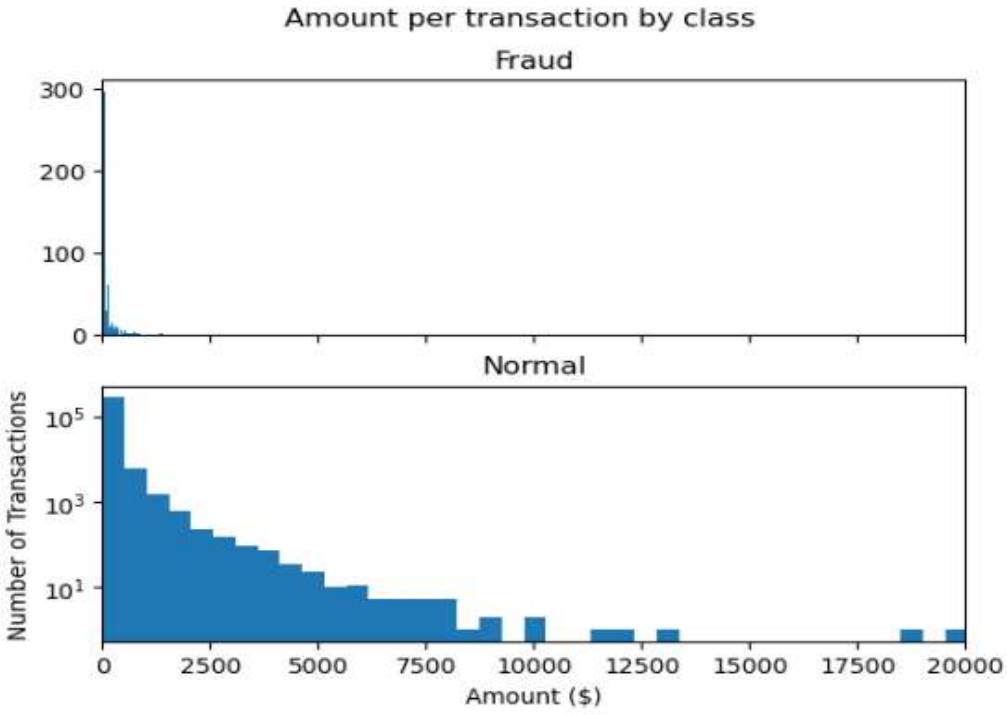


FIG: B.5 DATASET VISUALIZATION

```
In [10]: data.isnull().values.any()

Out[10]: False

In [11]: fraud.Amount.describe()

Out[11]: count     492.000000
         mean      122.211321
         std       256.683288
         min         0.000000
         25%         1.000000
         50%         9.250000
         75%       105.890000
         max      2125.870000
         Name: Amount, dtype: float64

In [12]: normal.Amount.describe()

Out[12]: count    284315.000000
         mean         88.291022
         std         250.105092
         min           0.000000
         25%           5.650000
         50%          22.000000
         75%          77.050000
         max       25691.160000
         Name: Amount, dtype: float64
```

FIG: B.6 DATA PRE-PROCESSING

```
In [13]: normal_sample=normal.sample(n=492)

In [14]: new_dataset = pd.concat([normal_sample, fraud], axis=0)

In [15]: new_dataset.head()
```

Out[15]:

	Time	V1	V2	V3	V4	V5	V6	V7
188671	128077.0	-3.288944	-4.809848	-1.231524	6.041078	-1.948400	1.772856	2.177367
190546	128878.0	2.020493	-1.000658	-1.039495	-0.453275	-0.737034	-0.433760	-0.587955
46318	42728.0	0.883694	-0.761362	0.928801	1.389779	-0.730351	1.228938	-0.771123
267636	162855.0	-0.072377	0.735400	-2.211240	-2.153156	3.556343	2.781633	1.142571
189610	128481.0	-0.264285	0.990040	-0.643148	-0.984799	0.813840	0.033159	0.536661

5 rows × 31 columns

```
In [16]: new_dataset.tail()
```

Out[16]:

	Time	V1	V2	V3	V4	V5	V6	V7
279863	169142.0	-1.927883	1.125653	-4.518331	1.749293	-1.566487	-2.010494	-0.882850
280143	169347.0	1.378559	1.289381	-5.004247	1.411850	0.442581	-1.326536	-1.413170
280149	169351.0	-0.676143	1.126366	-2.213700	0.468308	-1.120541	-0.003346	-2.234739
281144	169966.0	-3.113832	0.585864	-5.399730	1.817092	-0.840618	-2.943548	-2.208002
281674	170348.0	1.991976	0.158476	-2.583441	0.408670	1.151147	-0.096695	0.223050

5 rows × 31 columns

FIG: B.7 UNDER-SAMPLING

```
In [17]: new_dataset.groupby('Class').mean()
```

Out[17]:

Class	Time	V1	V2	V3	V4	V5	V6	V7
0	95169.597561	-0.019460	0.014204	-0.073895	0.051097	0.102674	-0.022626	0.014979
1	80746.806911	-4.771948	3.623778	-7.033281	4.542029	-3.151225	-1.397737	-5.568731

2 rows × 30 columns

```
In [18]: X = new_dataset. drop (columns='Class', axis=1)

In [19]: Y = new_dataset['Class']
```

FIG: B.8 GROUPING THE DATASET

```
In [20]: from sklearn. model_selection import train_test_split
         X = data.iloc[:,:30]
         Y = data.iloc[:,30]
         print (X.shape)
         print (Y.shape)

         (284807, 30)
         (284807,)

In [21]: X_train, X_test, Y_train, Y_test = train_test_split (X, Y, test_size=0.25, ra
         print (X_train.shape)
         print (X_test.shape)
         print (Y_train.shape)
         print (Y_test.shape)

         (213605, 30)
         (71202, 30)
         (213605,)
         (71202,)
```

FIG:B.9 TRAINING AND TESTING

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(X_train,Y_train)
rfc_pred = rfc.predict(X_test)
print("Training Score: %f"%rfc.score(X_train, Y_train))
print("Testing Score: %f"%rfc.score(X_test, Y_test))
```

```
Training Score: 1.000000
Testing Score: 0.999649
```

In [25]:
```
from sklearn.metrics import accuracy_score, classification_report,confusion_
print (accuracy_score (Y_test,rfc_pred))
print(classification_report(Y_test,rfc_pred))
print(confusion_matrix(Y_test,rfc_pred))
```

```
0.9996488862672397
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     71095
           1       0.93      0.83      0.88       107

    accuracy                           1.00     71202
   macro avg       0.96      0.92      0.94     71202
weighted avg       1.00      1.00      1.00     71202

[[71088     7]
 [   18    89]]
```

FIG: B.10 ADA-BOOST MODEL

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(X_train,Y_train)
rfc_pred = rfc.predict(X_test)
print("Training Score: %f"%rfc.score(X_train, Y_train))
print("Testing Score: %f"%rfc.score(X_test, Y_test))
```

```
Training Score: 1.000000
Testing Score: 0.999649
```

In [25]:
```
from sklearn.metrics import accuracy_score, classification_report,confusion_matrix
print (accuracy_score (Y_test,rfc_pred))
print(classification_report(Y_test,rfc_pred))
print(confusion_matrix(Y_test,rfc_pred))
```

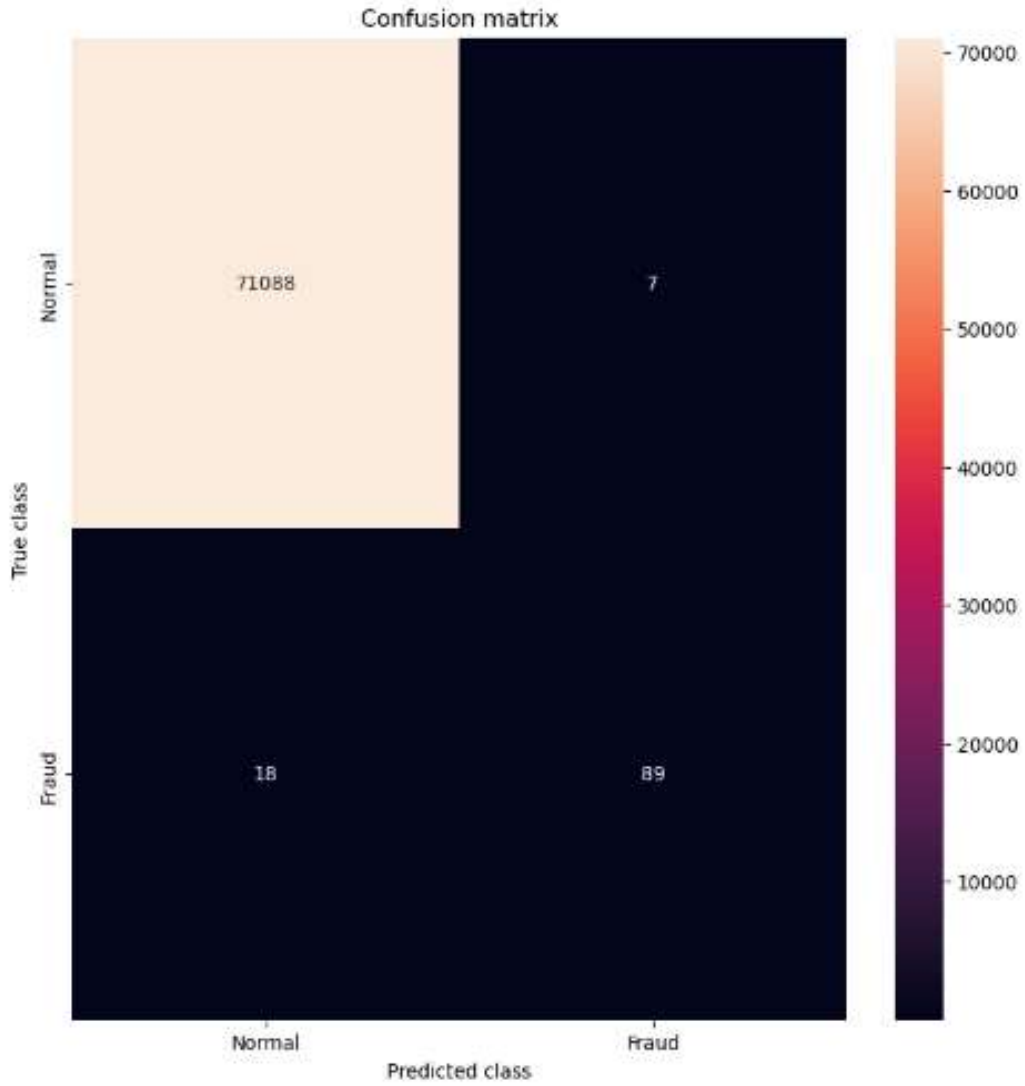
```
0.9996488862672397
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     71095
           1       0.93      0.83      0.88       107

    accuracy                           1.00     71202
   macro avg       0.96      0.92      0.94     71202
weighted avg       1.00      1.00      1.00     71202

[[71088     7]
 [   18    89]]
```

FIG: B.11 RANDOM FOREST MODEL



	precision	recall	f1-score	support
0	1.00	1.00	1.00	71095
1	0.93	0.83	0.88	107
accuracy			1.00	71202
macro avg	0.96	0.92	0.94	71202
weighted avg	1.00	1.00	1.00	71202

ROC AUC: 0.9158386205636397

FIG: B.12 CONFUSION-MATRIX

C. RESEARCH PAPER

Fortifying Financial Security: Unveiling Advanced Anti-Fraud Systems for Robust Safety Nets

S.N.V.S.Sai Naidu[1, a)], Sappa Satya Harsha[2, b)] and Dr.A.Deepa[3, c) (*)]

[1,2,3]*Department of Computer Science and engineering, Sathyabama Institute of Science and Technology, Chennai,*
Tamil Nadu, India.

[a)] sainaidusaragadam@gmail.com
[b)] sappasatyaharsha7281@gmail.com
[c)] *Corresponding author:* adeepa.cse@sathyabama.ac.in

Abstract: In this credit card detection project, the primary emphasis is on harnessing the power of machine learning algorithms, with specific focus on the random forest algorithm, to effectively discern and prevent fraudulent transactions. Recognizing the escalating prevalence of online fraud, the project aims to fulfill the critical need for robust fraud detection methods capable of swiftly distinguishing between normal and fraudulent credit card transactions. The initial phase involves feeding a credit card transactions dataset into the system, where machine learning algorithms, particularly the random forest algorithm, are deployed to securitize the characteristics of each transaction. The assessment of algorithmic outcomes is based on key metrics such as precision, accuracy, and recall. To determine the optimal algorithm, a comparative analysis is undertaken, pitting the random forest against other machine learning algorithms, considering diverse perspectives and performance criteria in the context of credit fraud detection. The project encompasses pivotal stages including data preprocessing, feature engineering, Model selection, data partitioning, training evaluation, and fine-tuning. The overarching objective is to systematically evaluate the performance of machine learning algorithms, ultimately pinpointing the most effective approach for credit card fraud detection. These endeavors aimed at curtailing financial losses arising from fraudulent activities

KEYWORDS-Random Forest algorithm, Class distribution, KNN, Training models.

INTRODUCTION

This project addresses the escalating issue of credit card fraud, with a specific focus on leveraging machine learning algorithms, notably the Random Forest algorithm, for effective fraud detection. In the contemporary landscape, the surge in online fraud activities, facilitated by technologies like Trojan and Phishing, emphasizes the critical need for robust fraud detection methods. By inputting credit card transactions datasets into the system, the Random Forest algorithm is employed to discern between normal and fraudulent transactions. The project aims to mitigate financial losses by timely identifying and preventing unauthorized credit card usage. Evaluation of the algorithm's performance is conducted use key metrics such as acuracy, precision, and recall, with a comparative analysis against other ML algorithms commonly used in creadit card fraud deteiction. overarching objective is to provide insights into the strengths and weaknesses of various algorithms, ultimately selecting the best-performing one based on specific evaluation criteria. This research contributes to the ongoing efforts in developing accurate and efficient fraud detection systems, crucial for safeguarding individuals and financial institutions from the detrimental impacts of credit card fraud in an increasingly digitized world.

LITERATURE SURVEY

The challenge in ensuring security in digital era has been extensively analyzed by researchers in recent decades. In case of credit cards safety, automotive companies bid to create a solution, and all their solutions vary with respect to technology. In case of analyzing the gestures, various methods of hand sign recognition are extensively used. The research in creditcard fraudlent detection has explored variety of methods, with a particular emphasis neural networks, distributed data mining and data mining. This research incorporates ML and DL techniques to effectively identify and combat fraudulent activities within vast transaction datasets. The complexity of fraud detection is further compounded by the challenge of imbalanced data, where legitimate transactions significantly outnumber fraudulent ones. To address this, researchers have investigated various techniques, including classification methods, sampling methods, and resampling techniques. Classification methods aim to enhance the performance of models, especially for the minority class representing fraudulent transactions. Sampling methods involve strategies such as over-sampling the under-sampling the majority class, with resampling techniques create synthetic instances to balance the dataset.

In the reality of ML algorithms for credit card detection of fraud, diverse approaches which are employed. SVM, Logistic Regression, , Decision Trees, Gradient Boosting, and KNN are among the key algorithms utilized. SVM, a supervised learning algorithm, is adept at classification and regression tasks. Decision Trees employ tree-like structures for classification and regression, while Logistic Regression is a regression analysis method focused on predicting binary outcomes. Gradient Boosting combines predictions from weak models to enhance overall performance, and KNN is a straightforward yet effective classification algorithm based on the proximity to neighbors. Overall, credit card fraud detection remains a dynamic field where researchers continually innovate to adapt to evolving fraud patterns and improve the efficacy of detection methodologies. Data pre-processing which is pivotal in enhancing the generalization performance of ML and data mining algorithms. This phase involves transforming raw data into format that is easily interpretable by these algo's. Raw data often presents challenges such as incompleteness, noise, and inconsistency. Incomplete data may lack attribute values, and dealing with missing data is addressed through techniques like imputation or removal of instances with missing values. Noisy data, containing errors or outliers, requires techniques like smoothing or outlier detection for effective handling. Inconsistent data, marked by discrepancies or coding errors, necessitates standardization and error correction. Upon importing libraries and inspecting raw data, checking for missing values is a crucial initial step. The subsequent process involves addressing categorical data, where label encoding is highlighted as an efficient tool for converting categorical labels into numeric values. The paper [1] notes that KNN, SVM and decision trees, achieve medium level accuracy.LR and FL show the slow accuracy. ANN, fuzzy systems, and KNN have more detection rate, while SVM, LR , and DT provide detection rate high at a medium level. emphasizes the ongoing need for research and development in the field of Indian Sign Language recognition. In the paper [2] we researched algorithms such as Naïve Bayes and Logistic Regression. Naïve Bayes relies on Bayes' theorem for classification, while Logistic Regression is used for predicting values, like linear regression. In the paper [3], They've described the important Train algorithmic techniques which are the Synthetic Minority Oversampling Techniques (SMOTE) and Whale Optimization Techniques. They mostly focus to enhance the performance to solve. In the paper [4] evaluates different machine learning strategies for fraud detection, with random forest showing superior results in terms of precision, recall, F1-score, accuracy, and false rate. The method is recommended for testing on larger real-time datasets with diverse machine learning methods in future work. Through paper [5], Credit card fraud is a rising threat and adapting to evolving methods is crucial. Machine learning's performance depends on factors like input data type. Deep learning, particularly CNNs with 20 layers, outperforms traditional algorithms, achieving a top accuracy of 99.72% in fraud detection. This [6] project focuses mostly credit card fraud spotting using ML algorithms, specifically RF and Adaboost. The goal is to combat the rising issue of credit card fraud in online transactions and e-commerce. The project evaluates the performance of both algorithms based on validity, recall, and F1-score. The algorithms with the best overall metrics is considered the most effective for fraud detection. The research paper[7] introduces a method using Genetic Algorithm (GA) for featured selection in combination with machine learning classifiers for credit card fraud detection. Results on the European dataset show high accuracy, with GA-RF achieving 99.98%. The proposed framework is also validated on a synthetic dataset, where GA-DT and GA-ANN achieve perfect accuracy. The study aims to expand validation using more datasets in future work. The paper[8] addresses credit card fraud detection, tackling class imbalance by using the SMOTEENN technique for data balancing. It proposes a powerful approach with a deep learning ensemble, leveraging the LSTM neural network in the AdaBoosht framework. Experimental results on a creadit card fraudlent datasets demonstrate

superior performance with high sensitivity (0.99), specificity (0.98), and AUC (0.90) compared to benchmark's algorithm. The combination of SMOTE-ENN and boosted LSTM proves effective, and future research may explore more resamipling technique and improv feature selecthion for enhance classifications.

EXISTING WORK

For Credit Card Fraud detection, we have many existing algorithms for detection. Among them we have several algorithms such as SVM, Logistic Regression which are widely used in different scenarios. In this type of technique, this algorithm requires greater detection potential in Large Datasets. In these algorithms we have mainly three basic problems. They have variable in nature, less precision, accuracy, and F-1 scores. They cannot deal with large amount of Data.

S.NO	AUTHORS	ADVANTAGE	DISADVANTAGES
1	Yashvi Jain, Namrata Tiwari	ANN, fuzzy systems, and KNN have a high detection rate, while Logistic Regression, SVM, and decision trees provide a high detection rate at a medium level.	k-nearest neighbor, decision trees, and SVM achieve medium accuracy. Fuzzy Logic and Logistic Regression show the lowest accuracy.
2	Heta Naik, Prashasti Kanikar	Predictive models not used before classifications	In algorithm KNN accuracy depend on peer distances
3	V Sahaysakila, D Aishwaryaikhakolli, V Yasaswi	explained the Twain important algorithmic techniques which are the WOA and SMOTE	More resources required for training and inference in both WOA and SMOKE
4	Naresh Kumar Trivedi , Sarita Simaiya,Umesh Kumar Lilhore , Sanjeev Kumar Sharma	Algorithms has high accuracy , adaptability , real-time detection, and analysis	SVM algorithm faces difficulty in imbalanced data , result in poor detection
5	Fawaz Khaled Alarfaj; Iqra Malik; Hikmat Ullah Khan; Naif Almusallam; Muhamma d Ramzan; Muzamil Ahmed	ML performance depends on factors like input data type. Deep learning, particularly CNNs with 20 layers, outperforms traditional algorithms, achieving a top accuracy	Deep learning models are susceptible of overfitting and failed to generalize new , unseen data.This leads in poor performance and upgrades false positives in real-world scenarios.
6	Ruttala sailusha; V.Gnaneswar; R. Ramesh	The project evaluates the performance of both algorithms based on accuracy, precision, recall, and F1-score.	Over time the patterns of fraudulent activities may, change leading to concept drift .
7	Emmanuel Ileberi1*, Yanxia Sun1 and Zenghui Wang2	Introduces a method using Genetic Algorithm (GA) for feature selection in combination with machine learning classifiers for credit card fraud detection. Results on the European dataset show high accuracy, with GA-RF achieving 99.98%.	GA provide an approach to feature selection , they do not guarantee finding the global optimal feature subsets

8	Delamaire, L; Abdou HAH; Pointon J	It addresses credit card fraud detection, tackling class imbalance by using the SMOTEENN technique for data balancing. It proposes a powerful approach with a deep learning ensemble, leveraging the LSTM neural network in the AdaBoost framework.	Benchmark algorithms performs with less sensitivity, specificity , Auc compared to these techniques
9	Dingling Ge; Jianyang Gu; Shunyu Chang	It addresses online credit card fraud using a LightGBM-based algorithm on the IEEE-CIS Fraud Detection dataset	It criticizes previous studies for limitations in data and models.
10	R Sailusha, V Gnaneswar, R Ramesh	It works great in Metrix like accuracy and scalability	Precision is based on neighbor data's

PROPOSED METHOD

The proposed approach for credit fraud detection employs fine art ML algorithms to identify anomalous activities, often referred to as outliers. The preliminary architecture diagram illustrates the system's structure, with a more detailed representation reflecting real-life elements. The dataset, sourced from Kaggle, encompasses 31 columns, with 28 anonymized as v1-v28 to safeguard sensitive information. Key columns include Time, representing the time gap between transactions, Amount indicating the transacted sum, and Class with zero denoting a true transaction and 1 signifying a fraud transaction.

Graphical analysis is performed to uncover patterns and inconsistencies in the dataset. Notably, the distribution reveals a significant imbalance between legitimate and fraudulent transactions, and a temporal analysis exposes transaction peaks during daytime.

A histogram is generated for each column to visually inspect the dataset for potential missing values. This step ensures data completeness, facilitating smooth processing by machine learning algorithms.

Post-data analysis, a heatmap is constructed to visualize the correlation in the middle of predictor adjustables and the target classes variable. Then dataset is then formatted and processed, involving standardization of the Time and Amount columns and the removal of the Class column to ensure fair evaluation.

The machine learning pipeline involves fitting the processed data into a model, with subsequent application of outlier detection modules. Among these modules is the Ada-Boost Classifier, a powerful algorithm designed to enhance classification accuracy by combining weak classifiers. This methodology aims to provide a robust fraud detection system that can effectively identify outliers within credit card transactions, contributing to the ongoing efforts in ensuring the security of electronic payment systems.

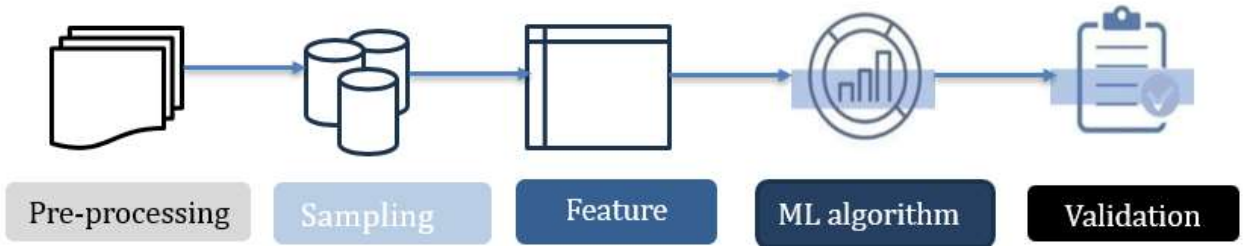


Figure:1- Data flow diagram

The above flow chart explains the steps involved in getting a proper result in finding fraudulent transactions in present model of the project

MODEL TRAINING

The model development process is a pivotal step in constructing effective machine learning models, encompassing the judicious selection of suitable algorithms and a meticulous evaluation of their performance. This intricate journey begins with the segregation of data into train and test sets, where the former make easier the model's learning process, and the latter work for as a litmus test for its ability to generalize to unseen data. In the method delineated, two potent ensemble learning algorithms, Random Forest and AdaBoost, take center stage during the training phase. The resulting models are subjected to evaluation through the construction of a confusion matrix, shedding light on TP, TN, FP and FN Critical perform metrics, include precision, recall, and F-1 score, are subsequently computed from this confusion matrix, provides a nuanced understanding of the model's efficacy. This ensuing comparison of these metrics between the Random Forest and AdaBoost models becomes the compass for selecting the superior model, ensuring an optimal representation of the underlying data, and anticipating its potential performance in future scenarios.

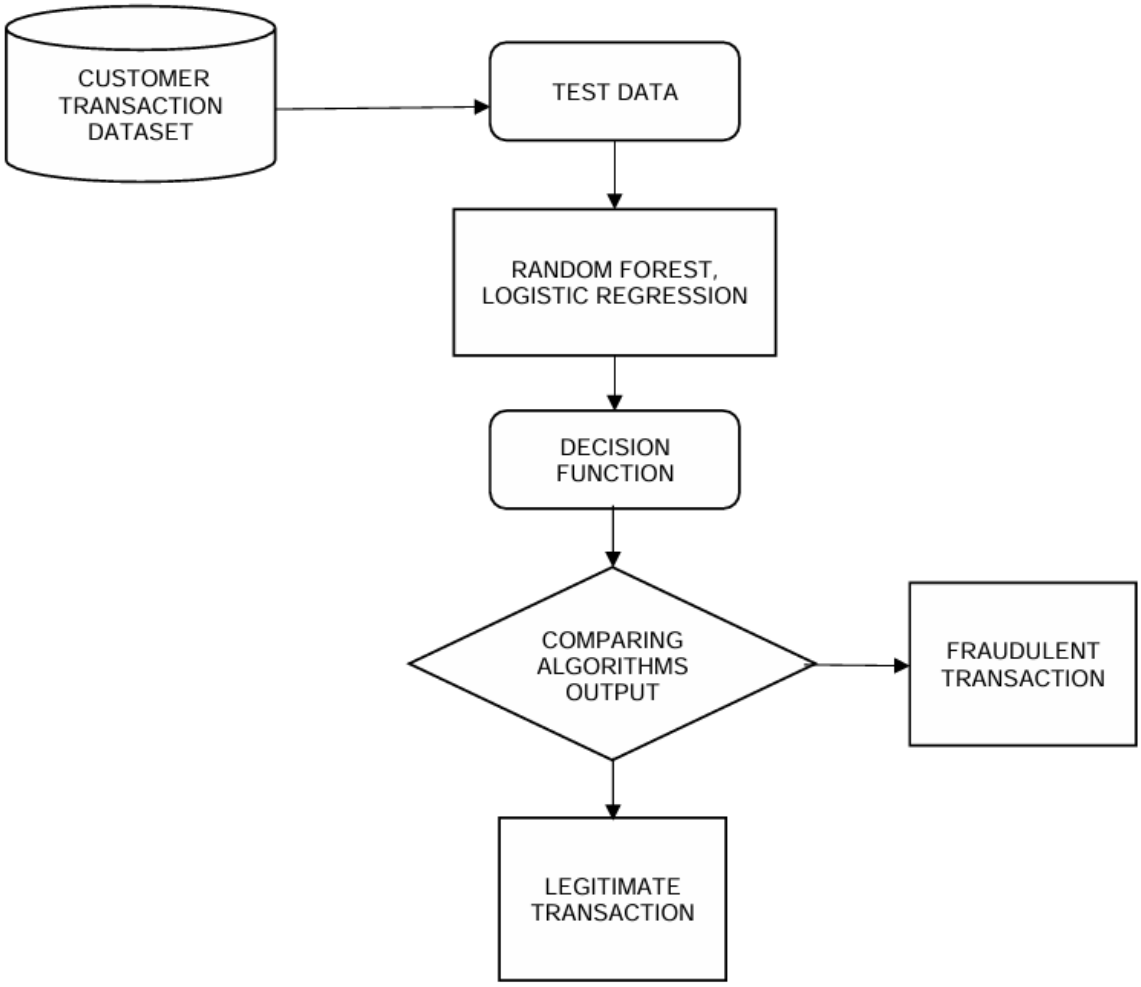


Figure:2- Architecture Diagram

The above figure explains the architecture and frame work of the project and Initial step involves in storing the data of a transaction in the Dataset and then involves in the algorithmic outputs and finds between fraudulent and

legitimate transaction in each data set

Data collection: Customer transaction data is collected, which could include details such as the size of the transaction, the time stamp and location of the withdrawn, and the customer's past transaction history. Test data: The customer transaction data is then divided into two parts: a train set and a test set. The train set is used for training the machine learning models, while the test set is use to assess the performance of models. Featured select: A subsets of features is selecting from the customer transaction data. These features are the ones that are most likely to be relevant to predicts if a transaction is fraud . Model training: Three machine learning models are trained on the training data: a logistic regression, random forest, and an unspecified algorithm. These models learn to identify' s patterns in these data that are associated with fraudulent transactions. Model comparison: The performance of the three models is compared on the test data. The model that performs the best is then used to make predictions about whether new transactions are fraudulent. Decision making: When a new transaction occurs, the system use best ML models to forecast if the transaction was fraud or legit. If the transaction forecast as fraud, the system should take some action, such as contacting the customer for verification or blocking the card or transaction.

Pandas stands out as an important role for open-source Python library, giving some powerful data manipulation and analyzing the tools through its robust data structures And giving Prior to the Pandas, the role of Python in data preprocessing it is primarily revolving around munging and preparation giving some gap in the data analysis capabilities. Pandas will adapt the addresses the problem providing functionalities to seamlessly execute these seven core steps in the data preprocessing and analysis : preparing, modeling analyzing data, loading, manipulating, retrieving and implementing without depending on source. Its versality that will finds in extensive app in the various fields, including commerce and academia, spanning finance ,economics statistics and analytics. And also Matplotlib will emerges as the cornerstone in the Machine learning within the python libraries. And the Python 2D plotting library, Matplotlib will gives in publication across some hardcopy formats and these interactive environments , making in data exploration and presenting the Pandas. It is also easy to use facilitates the generation of the plots and some of the histograms, power spectras with the minimal codes.

TensorFlow and Numpy are important components in the tool for data scientists and machine learning practitioners. TensorFlow, is an open-source library which was developed by the Google , which is a cornerside for dataflow for diffentiable programming, that giving a technical suite of tools to build and deploying the machine learning models. Its versatility which is spans the researchers to productioh environments, empowering the users with symbolic and with the help of some math operations and some neural network. Numpy gives as a fundamental package for some scientific computing in the python, delivering robust multidimensional array for capable and the array for essential functions in some numerical operations. And also its beyond core array for processing features, Numpy facilitates look like integration with some other languages like C or C++ ,Fortan and also offering functionalities in linear algebra, Fourier transforms and generating the random number. And also it adaptes as a container in diverse data types and it makes it as choice to the broad spectrum for scientific applications, providing efficiency and fast in data manipulation activities. And these all combine TensorFlow and NumPy gives a formidable foundation in tackling the difficulties in the machine learning.

Seaborn will gives the exceptional library in the graphical plotting for the python libraries. It is known for the capabilities in the visualizing the data with neat and clarity, Seaborn gives a plethora of color palattes and it will gives some good looking defaults producing the appeal to the statistical plots that are generated by the python. And it is also user-friendly interface and it will gives rich set of functions that makes a preferred choice for the scientists and the data analysts looking to create visually stunning and beautiful visualizations effortlessly. By without integrating the pandas data structures and the Matplotlib plotting functionalities , Seaborn will process the generating a wide range of the statistical plots, from the simple distributions to complex visualizations. while exploring the relationships between these visualizing distributions, variables and highlighting patterns in the data, Seaborn Finally gives users to convey their objectives fastly, giving heights the standard of these data visualizations in python.

Detecting the credit card payment fraud using the neural networks, like particularly deep learning techniques such as Long Short=Term Memory Networks(LSTM), has gained the recognition in the present years. And Many

methods will depend on the combination of these deep learning architectures and the oversampling algorithms to find the fraud detection accurately. The LSTM fraud detection model operates by sequentially dividing the transaction data, demonstrating the networks ability to capture the temporary dependence of the data. Integration of these synthetic minority class in the oversampling techniques, such as SMOTE(Synthetic Minority Over-sampling Technique),helps address the issue of these imbalanced datasets that are commonly viewed in the fraud detection tasks. And there are some new approaches such as combining the SMOTE with the k-Nearest Neighbour(KNN)classification algorithm these have been proposed for the further to improve the fraud detection performance. By the way the designing and building these KNN-SMOTE-LSTM-based fraud detection network model, this approach main goal is to produce the output of the robustness and accuracy of the fraud detection systems. The incorporation of KNN discriminant that classifiers into the model facilitates the continous searching of these security-generating samples, that are contributing for more effective and more ease to enhance the fraud detection process. Hence Leveraging the neural networks, oversampling techniques and classification algorithms that will holds the promise for the field of the credit card fraud detection and producing security measures in the financial transactions.

A. DATA COLLECION AND EXPLORATION

The dataset containing credit card transaction records is crucial for understanding fraudulent activities. It comprises approximately twenty-eight lakh transactions collected from the Kaggle community. The dataset encompasses 31 parameters, with 28 of them being the result of PCA transformation for confidentiality reasons. Only "Time" and "Amount" remain unchanged. Among the transactions, 492 are labeled as fraud, indicating a significant class imbalance issue. To gain insights into the data and its features, visualization techniques can be employed after importing necessary packages. This exploration aims to understand patterns and relationships within the data, especially concerning fraudulent transactions.

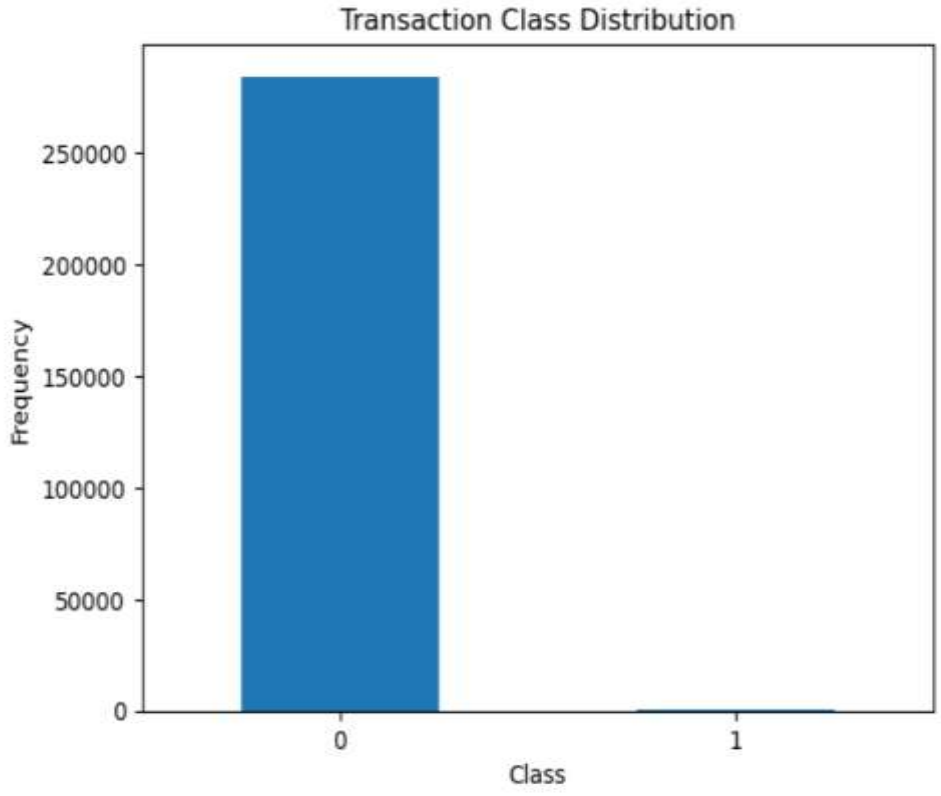


Figure:3 - Transaction Class Distribution

The above graph contails x-axis – class , y-axis – Frequency. All necessary libraries like NumPy, Pandas, Matplotlib, Scikit-Learn, TensorFlow, and Keras are installed via Conda. In our Jupyter notebook, we import Pandas and NumPy to handle data and use the `read_csv` function to load CSV data into a Pandas DataFrame. Matplotlib is employed for plotting 2D figures in various formats, facilitating data visualization. Pyplot, the primary module of Matplotlib, offers a MATLAB-like interface for plotting in Python, enhancing data exploration and analysis.

B. DATA PREROCESSING

Data preprocessing is vital for preparing raw data for machine learning and data mining. It involves transforming data to a usable format, handling missing values, noise, and inconsistencies. Label encoding is often used to convert categorical data into numerical values. For credit card datasets, common preprocessing steps include cleaning to address data quality issues and sampling to manage imbalanced data distributions.

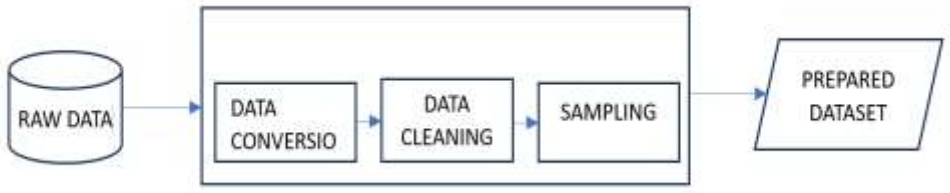


Figure :4 - DATA PRE-PROCESSIG

C. TRAINING and TESTING

The train/test split is a critical step in machine learning where the dataset is divided into two sets: training and testing. The training set is used to build the model, while the testing set evaluates its performance. This helps ensure the model can generalize well to new data.

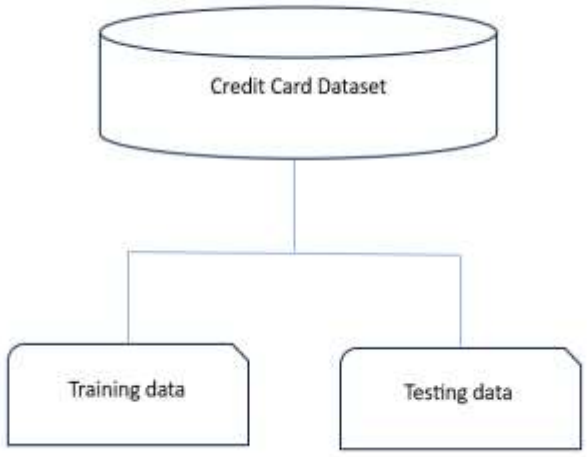


Figure:5 - Training and Testing

A. MODEL BUILDING

Model building is a crucial step in developing a machine learning system. It involves training the data with different algorithms, such as Random Forest and AdaBoost, to find the best model. The performance of each model is evaluated using metrics like accuracy, precision, F-1 score, and recall, presented in a confusion matrix. The model with the highest performance metrics is selected as the best choice for further use.

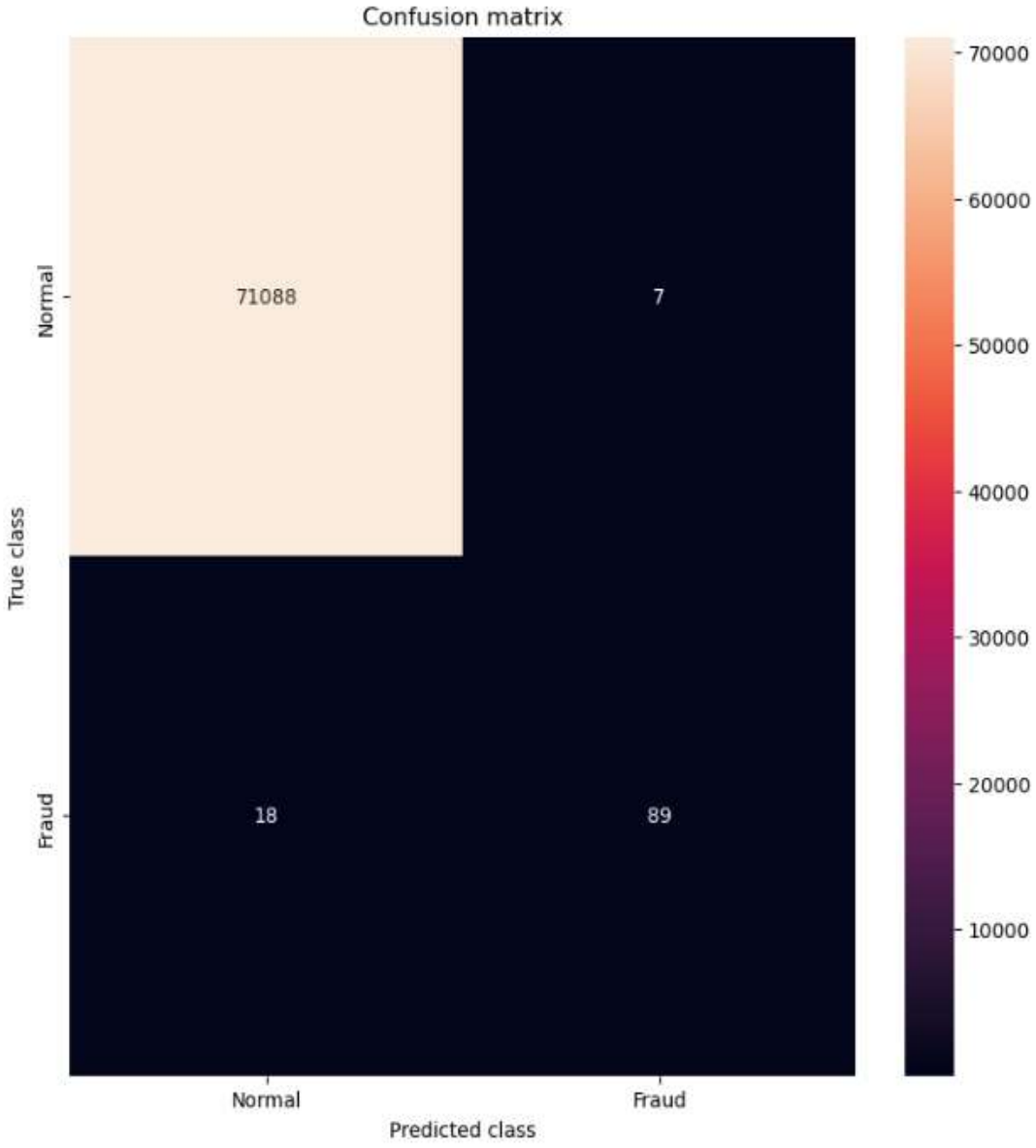


Figure:6 – Confusion Matrix

OUTPUT:

In evaluating machine learning models, several metrics provide insights into their performance. True Negative (TN) and True Positive (TP) represent correct predictions, while False Negative (FN) and False Positive (FP) denote incorrect predictions. Accuracy measures overall correctness, Precision assesses positive prediction accuracy, and Recall evaluates the model's ability to capture all positive instances. F1 Score balances Precision and Recall. Together, these metrics offer a comprehensive understanding of the model's strengths and weaknesses, aiding informed decision-making. Our project employs diverse training data for accurate translation of hand signs.

Result Analysis

It is the final stage of the process in which we compare and analyses the model values to select the best one. In the report below, we've some metrics to check the model performance, let's make a brief explanation about how to understand those values, and evaluate the machine learning model. Before explaining the mathematical formulas, we will explain some terms and what it does represent.

TN — True Negative: when a case was negative and predicted negative;
TP — True Positive: when a case was positive and predicted positive;
FN — False Negative: when a case was positive but predicted negative;
FP — False Positive: when a case was negative but predicted positive.

Accuracy - What percent of predictions the model did correctly?

$$\text{Accuracy} = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions}$$

Precision — What percent of the predictions were correct?

$$Precision = \frac{TP}{TP + FP}$$

Recall — What percent of the positive cases the model catches?

$$Recall = \frac{TP}{TP + FN}$$

F1-Score — What percent of positive predictions were correct?

$$F1 - Score = \frac{2(Recall * Precision)}{Recall + precision}$$

CONCLUSION

Exploring the dataset through visualization tools provides essential insights into its characteristics. The machine learning process involves a crucial train/test split, where most of the dataset is allocated for training the data and a small portion of data for testing. This facilitates the creation of an efficient machine learning model and subsequent accuracy assessment. Model building, a pivotal phase in development, incorporates Random Forest and Adaboost algorithms. The comparison of performance metrics, includes score of accuracy, F-1 score, recall, through a confusion matrix, guides the select of the most successful model's for credit card trust transaction detection. This comprehensive approach ensures the project's success in addressing the complexities of fraud detection while considering the unique challenges posed by the dataset.

Predicted class

	precision	recall	f1-score	support
0	1.00	1.00	1.00	71095
1	0.93	0.83	0.88	107
accuracy			1.00	71202
macro avg	0.96	0.92	0.94	71202
weighted avg	1.00	1.00	1.00	71202

ROC AUC: 0.9158386205636397

REFERENCES

[1] Yashvi Jain, Namrata Tiwari, Shripriya, Dubey, Sarika Jain: A Comparative Analysis of Various Credit Card Fraud Detection Techniques, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-7 Issue-5S2, January 2019.

[2]. Heta , Kanikar: Credit card Fraud Detection on Machine Learning Algorithms, International Journal of Computer Applications (0975 – 8887) Volume 182 – No. 44, March 2019.

[3]. Sahayasakila.V, D. Kavya Monisha, Aishwarya, Sikha kolli Venkatavisalakshi Sesh Sai Yasaswi: Credit Card Fraud Detection System using Smote Technique and Whale Optimization Algorithm, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958, Volume8 Issue-5, June 2019.

[4]. Navan Shu Khare, Saad Yunus Sait: Credit Card Fraud Detection Using Machine Learning Models and Collating Machine Learning Models, International Journal of Pure and Applied Mathematics Volume 118 No. 20 2018, 825-838 ISSN: 1314-3395.

[5]. E Ileberi, Y Sun, Z Wang ; A machine learning based credit card fraud detection using the GA algorithm for feature selection, Journal of Big Data (2022) 9:24 https://doi.org/10.1186/s40537-022-00573-8

[6]. A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection , A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection, Received January 14, 2022, accepted January 27, 2022, date of publication January 31, 2022, date of current version February 15, 2022

[7]. D Ge, J Gu, S Chang, JH Cai : Credit card fraud detection using lightgbm model , *2020 International Conference on E-Commerce and Internet Technology (ECIT)* , Year: 2020, Pages: 232-236

[8]. Naresh Kumar Trivedi1 , Sarita Simaiya2 ,*Umesh Kumar Lilhore3 , Sanjeev Kumar Sharma: An Efficient Credit Card Fraud Detection Model Based on Machine Learning Methods
International Journal of Advanced Science and Technology Vol. 29, No. 5, (2020), pp. 3414 - 3424

[9]. Fawaz Khaled Alarfaj; Iqra Malik; Hikmat Ullah Khan; Naif Almusallam: Credit Card Fraud Detection Using State-of-the-Art Machine Learning and Deep Learning Algorithms, Received March 20, 2022.

[10] R Sailusha, V Gnaneswar, R Ramesh : Credit card fraud detection using machine learning , 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)

D. CERTIFICATE

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
CATEGORY-1 UNIVERSITY BY UGC
Accredited "A++" Grade by NAAC | 2B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

SCHOOL OF COMPUTING

Department of Computer Science and Engineering

Certificate of Presentation

This is to certify that

SAPPA SATVAHARSHA

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

has presented a paper

FORTIFYING FINANCIAL SECURITY : UNVEILING ADVANCED ANTI-FRAUD SYSTEMS FOR ROBUST SAFETY NETS

in the

International Conference on Cognitive Computing and Artificial Intelligence (ICCCAI-2024) in association with Taylor's University, Malaysia held on **7th & 8th March 2024.**

Dr. Sayan Kumar Ray
Head, School of Computer Science
Taylor's University

Dr. L. Lakshmanan
HOD, CSE Dept
Sathyabama Institute of Science and Technology

Dr. T. Sasikala
Dean, SOC
Sathyabama Institute of Science and Technology