BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

A PROJECT REPORT

Submitted by

GEETH AKSHAY KUMAR M	**113319104022**
SANJAY M	**113319104073**
SHAIK ABDUL ALEEM	**113319104078**
THIYAGARAJAN P G	**113319104093**

in partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

VELAMMAL INSTITUTE OF TECHNOLOGY – CHENNAI

ANNA UNIVERSITY

MAY 2023

BONAFIDE CERTIFICATE

Certified that this project report **"BITCOIN ATTACK PREDICTION USING MACHINE LEARNING"** is the bonafide work of **"Geeth Akshay Kumar M 113319104022, Shaik Abdul Aleem 113319104078, Sanjay M 113319104073, Thiyagarajan P G 113319104093"** who carried out the project work under my supervision.

SIGNATURE

Dr.V.P.GLADIS PUSPARATHI, M.Tech., Ph.D.
HEAD OF THE DEPARTMENT
Computer Science and Engineering,
Velammal Institute of Technology,
Velammal Gardens, Panchetti,
Chennai – 601 204.

SIGNATURE

Mr.A.ANBUMANI, M.Tech.
ASSISTANT PROFESSOR
Computer Science and Engineering,
Velammal Institute of Technology,
Velammal Gardens, Panchetti,
Chennai – 601 204.

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

VIVA-VOCE EXAMINATION

The viva-voice examination of this project work was done as a part Computer science and Engineering held on _____

GEETH AKSHAY KUMAR M	**113319104022**
SANJAY M	**113319104073**
SHAIK ABDUL ALEEM	**113319104078**
THIYAGARAJAN P G	**113319104093**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are personally indebted to many who had helped us during the course of this project work. Our deepest gratitude to the God Almighty. We are greatly and profoundly thankful to our beloved Chairman **Thiru. M.V. Muthuramalingam** for facilitating us with this opportunity. Our sincere thanks to our respected Director **Thiru. M. V. M. Sasi Kumar** who took keen interest on us. We are also thankful to our Advisors **Prof. K. Razak, Shri. M. Vasu** and our Principal **Dr. N.Balaji** for their never-ending encouragement which accelerates us towards innovation. We are extremely thankful to our Head of the Department **Dr. V.P.Gladis Pushparathi**, Project Coordinator **Mr.A.Manickavasagan**, Assistant Professor, Computer Science and Engineering for their valuable teachings and suggestions. From the bottom of our heart, we would like to thank our guide **Assistant Prof.A.Anbumani** who has been the pillar of this project without whom we would not have been able to complete the project successfully. The Acknowledgment would be incomplete if we would not mention word of thanks to our Parents, Teaching and Non-Teaching Staffs, Administrative Staffs and Friends who had motivated and lend their support throughout the project.

Thank you one and all.

ABSTRACT

Ransomware attacks are emerging as a major source of malware intrusion in recent times. While so far ransomware has affected general-purpose adequately resourceful computing systems. Many ransomware prediction techniques are proposed but there is a need for more suitable ransomware prediction techniques for machine learning techniques. Tracing cryptocurrencies payments due to malicious activity and criminal transactions is a complicated process. Therefore, the need to identify these transactions and label them is crucial to categorize them as legitimate digital currency trade and exchange or malicious activity operations. Machine learning techniques are utilized to train the machine to recognize specific transactions and trace them back to malicious transactions.

This paper presents an attack of ransomware prediction technique that uses for extracting information features in Artificial Intelligence and Machine Learning algorithms for predicting ransomware attacks. The application of the data science process is applied for getting a better model for predicting the outcome. Recently, the development of ransomware strains as well as changes in the marketplace for malware have greatly reduced the entry barrier for attackers to conduct large-scale ransomware attacks.

Variable identification and data understanding is the main process of building a successful model. Different machine learning algorithms are applied to the pre-processed data and the accuracy is compared to see which algorithm performed better other performance metrics like precision, recall, f1-score are also taken in consideration for evaluating the model. The machine learning model is used to predict the ransomware attack outcome.

ஆய்வுசுருக்கம்

சுவரொட்டு தாக்குதல்கள் இறங்குவது இறுதியாக மல்வேர் அகலத்திற்கான முக்கியமான மூலமாக வளர்ந்துவருகின்றன. முதன்முதலில் சுவரொட்டு பொறியியல் போன்ற பொதுக்களத்தில் அபிராப்பமாக வளர்ந்துவரும். பல சுவரொட்டு முன்னணி மற்றும் பயன்பாட்டு இயந்திர மாணவியிடம் பேராசையை வளர்ந்துவருகின்ற போது, இவ்வாறான சுவரொட்டு வகைப்பாட்டு முனைபாடுகள் தேவைப்படுகின்றன. தீமையான செயல்முறைகளுக்கான பாதுகாப்பு தேவையும் சுவரொட்டுகளை குறிப்பிட வேண்டும். இதனால், மேலும் படிப்புள்ள தரவுவிளைவுகளைப் பெற தரவு அறிவியல் பூர்வப் போக்குக்கு பயன்படுத்தப்படுகின்றது. சுவரொட்டு தாக்குதல் செயல்முறையை முன்னிலைப்படுத்த கையாளும் மேற்கோள்களைப் பார்க்கப்பட்டுள்ள முதன்முதலியான இயந்திர மாணவி முறைகள் விண்ணப்பத்தை பயன்படுத்தி மாற்றப்படும் இந்த ஆய்வுக்குப் பின்னர், விண்ணப்பப் போக்குகளின் வரிசைப்படுத்தல் மற்றும் தரவு புரிந்துகொள்ள முனைபாடு வழங்கப்படுகின்றது. சுவரொட்டு வகைப்பாடுகளின் மேல் நடவடிக்கைகள் முழுவதும் குறிப்பிடப்பட்டுள்ள முனைபாடுகள் போன்ற பெறுமதிகளையும் நியமித்துக்கொண்டு மற்ற முனைபாடுகளையும் ஒருங்கிணைத்து பார்க்கப்படுகின்றது. சுவரொட்டு தாக்குதல் தொழில்நுட்ப மாதிரிக்குள் நிரம்புவதற்கான முக்கிய அமைப்புகளைத் திருப்பியதும் இதனை வழங்குகிறது.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

API	APPLICATION PROGRAMMING INTERFACE
NB	NAIVE BAYES
IOT	INTERNET OF THINGS
ML	MACHINE LEARNING
DM	DATA MINING
UML	UNIFIED MODELING LANGUAGE

CHAPTER 1

INTRODUCTION

Cryptocurrencies, such as Bitcoin, are a form of digital currency designed to work outside of the traditional banking ecosystem. Cryptocurrencies are decentralized currencies that use blockchain technology to record transactions. Cryptocurrency transactions, aka the buying and selling of digital currency, are typically handled using a crypto-exchange platform. Unlike other threats, crypto-ransomware is neither subtle or hidden. Instead, it prominently displays lurid messages to call attention to itself, and explicitly uses shock and fear to pressure you into paying the ransom. A few so-called crypto-ransomware do not perform the encryption at all, and just use the threat of doing so to extort money or other properties form the victim.

These transactions often involve large sums of cryptocurrency, typically anonymized utilizing the blockchain, hence attracting cyber criminals. Like any system, cryptocurrency platforms and exchange mechanisms are vulnerable to cyber-attacks. Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python. Process of training and prediction involves use of specialized algorithms. It feed the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated in to three categories.

There are supervised learning, unsupervised learning and reinforcement learning. Supervised learning program is both given the input data and the corresponding labelling to learn data has to be labelled by a human being beforehand. Unsupervised learning is no labels. It provided to the learning algorithm. This algorithm has to figure out the clustering of the input data. Finally, Reinforcement learning dynamically interacts with its environment and it receives positive or negative feedback to improve its performance.

CHAPTER 2

LITERATURE SURVEY

1.Title **:** Bitcoin Heist : Topological Data Analysis for Ransomware Detection on the Bitcoin Blockchain

Author **:** Yitao Li , Cuneyt Gurcan Akcora , Yulia R. Gel, Murat Kantarcioglu

Year : 2019

Ransomware is a type of malware that infects a victim's data and resources, and demands ransom to release them. In two main types, ransomware can lock access to resources or encrypt their content. In addition to computer systems, ransomware can also infect IoT and mobile devices [23]. Ransomware can be delivered via email attachments or web-based vulnerabilities. More recently, ransomware have been delivered via mass exploits. For example, CryptoLocker used Gameover ZeuS botnet to spread through spam emails. Once the ransomware is installed, it communicates with a command-and-control center. Although earlier ransomware used hard-coded IPs and domain names, newer variants may use anonymity networks, such as TOR, to reach a hidden command and control server Once resources are locked or encrypted, the ransomware displays a message that asks a certain number of bitcoins to be sent to a bitcoin address. This amount may depend on the number and size of the encrypted resources. After payment, a decryption tool is delivered to the victim. However, in some cases, such as with WannaCry, the ransomware contained a bug that made it impossible to identify who paid a ransomware amount.

Proliferation of cryptocurrencies (e.g., Bitcoin) that allow pseudo-anonymous transactions, has made it easier for ransomware developers to demand ransom by encrypting sensitive user data. The recently revealed strikes of ransomware attacks have already resulted in significant economic losses and societal harm across different sectors, ranging from local governments to health care. Most modern ransomware use Bitcoin for payments. However, although Bitcoin transactions are permanently recorded and publicly available, current approaches for detecting ransomware depend only on a couple of heuristics and/or tedious information gathering steps (e.g., running ransomware to collect ransomware related Bitcoin addresses). To our knowledge, none of the previous approaches have employed advanced data analytics techniques to automatically detect ransomware related transactions and malicious Bitcoin addresses.

2.Title **:** The Bitcoin Heist : Classifications Of Ransomware Crime Families

Author **:** Micheline Al Harrack

Year : 2021

Tracing cryptocurrencies payments due to malicious activity and criminal transactions is a complicated process. Therefore, the need to identify these transactions and label them is crucial to categorize them as legitimate digital currency trade and exchange or malicious activity operations. Machine learning techniques are utilized to train the machine to recognize specific transactions and trace them back to malicious transactions or benign ones. I propose to work on the Bitcoin Heist data set to classify the different malicious transactions. The different transactions features are analyzed to predict a classifier label among the classifiers that have been identified as ransomware or associated with malicious activity. I use decision tree classifiers and ensemble learning to implement a random forest classifier, the results were promising to correctly classify a ransomware family based on transactional features and path activities combined with the length. This model is both simple in design and powerful in predicting ransomware families in supervised context.

To seek high confidence, a transaction classified by both models as belonging to the same ransomware family would justify further investigation and tracing of the individuals and entities behind the identified addresses. This study was limited to years 2014-2017 and analyzed specific ransomware as identified by Padua, Princeton, and Montreal. All white labels were excluded. Possible future work can include covering the transactions from 2011 until 2018 with all classes of ransomware to be tested and classified. Another stage would incorporate white labels to test against an existing class of ransomware or possibly different classes of unidentified ransomware based on transactional features, length, and path activities. An approach to apply is clustering the BitcoinHeist transactions to identify any possible commonalities among families of ransomwares.

The Bitcoin transactions are permanently recorded and publicly available. Based on the collected ransomware related Bitcoin addresses, the researchers have employed advanced data analytics techniques to detect ransomware related transactions and malicious Bitcoin addresses automatically. The machine learning approaches are evaluated based on Bit-Heist Ransomware dataset which is publicly available. Various approaches are used for it. The machine learning approaches are evaluated based on the patterns differentiating such cybercrime operations from normal bit-coin transactions in order to identify and report attacks.

3.Title : A Survey on Detection and Classification of Ransomware Bitcoin Transactions

Author : Sabira Karim, Shemitha PA

Year : 2021

Bitcoin might be a suburbanized form of payment system wherever the general public ledger is correctly supported in a very distributed manner. The unknown anonymous members referred to as miners, capital punishment a protocol that maintains and extends a distributed public ledger that records bitcoin transactions is termed a block chain. Block chain is enforced as a series of blocks. Bitcoin is that the known crypto-currency business. The transactions of bitcoin area unit utterly digital and unknown to a good extent. This case has crystal rectifier several cyber-crime perpetuators to use bitcoin as a secure haven for misbr transactions like Ransomware payments. Ransomware is malicious code that affects the payments entry reciprocally of ransom that should be paid. Machine Learning approaches could also be utilized to pour over the previous transactions as coaching information inorder to properly predict the people or teams to whom Ransomware payments area unit being created. This paper tries to explore the efficaciousness of various machine learning approaches in police work such payments. Ransomware may be a form of malware that infects a victim's information and resources, and demands ransom to unleash them.

Ransomware attacks have been among the major threats that target a wide range of Internet and mobile users throughout the world, especially critical cyber physical systems. Due to its unique characteristics, ransomware has attracted the attention of security professionals and researchers toward achieving safer and higher assurance systems that can effectively detect and prevent such attacks. The state-of-the-art crypto ransomware early detection models rely on specific data acquired during the runtime of an attack's lifecycle. However, the evasive mechanisms that these attacks employ to avoid detection often nullify the solutions that are currently in place.

More effort is needed to keep up with an attacks' momentum to take the current security defenses to the next level. This survey is devoted to exploring and analyzing the state-of-the-art in ransomware attack detection toward facilitating the research community that endeavors to disrupt this very critical and escalating ransomware problem. The focus is on crypto ransomware as the most prevalent, destructive, and challenging variation. The approaches and open issues pertaining to ransomware detection modeling are reviewed to establish recommendations for future research directions and scope.

4.Title **:** Predictive Analysis of Ransomware Attacks using Context-aware AI in IoT Systems.

Author **:** Vytarani Mathane, P.V. Lakshmi

Year : 2021

Among all those ransomware attacks could be more impacting owing to attack methodology where victim systems become unusable until a ransom is paid, typically have attacker-defined timelines to respond, and can cause more monetary loss. Ransomware attacks, one of the malware attacks affect all types of security issues availability which causes monetary losses, and sensitive information loss. Crypto ransomware, locker ransomware, and hybrid ransomware are common types of ransomwares. In crypto-ransomware attacks, data files are encrypted and the decryption key is provided only after paying the ransom. In locker ransomware attacks, the resources are blocked and are released only after paying the ransom. In hybrid ransomware attacks, both concepts of crypto ransomware and locker ransomware are used.

Due to the rising dependency on digital technology, cybersecurity has emerged as a more prominent field of research and application that typically focuses on securing devices, networks, systems, data and other resources from various cyber-attacks, threats, risks, damages, or unauthorized access. Artificial intelligence, also referred to as a crucial technology of the current Fourth Industrial Revolution, could be the key to intelligently dealing with these cyber issues. However, the dynamic nature and complexity of real-world situations and data gathered from various cyber sources make it challenging nowadays to build an effective AI-based security model.

While so far ransomware has affected general-purpose adequately resourceful computing systems, there is a visible shift towards low-cost Internet of Things systems which tend to manage critical endpoints in industrial systems. Many ransomware prediction techniques are proposed but there is a need for more suitable ransomware prediction techniques for constrained heterogeneous IoT systems. Using attack context information profiles reduces the use of resources required by resource-constrained IoT systems. This paper presents a context-aware ransomware prediction technique that uses context ontology for extracting information features like connection requests, software updates, Artificial Intelligence, Machine Learning algorithms for predicting ransomware. The proposed techniques focus and rely on early prediction and detection of ransomware penetration attempts to resource-constrained IoT systems. There is an increase of 60 % of reduction in time taken when using context-aware dataset over the non-context aware data.

5.Title **:** An economic analysis of ransomware and its welfare consequences

Author **:** J. Hernandez-Castro , A. Cartwright, E. Cartwright

Year : 2021

The case of the ransomware is particularly interesting in this regard, in that we can see a criminal experimenting and making multiple mistakes and numerous failed attempts before 'getting it right in the end' [5]. Crypto locker, discovered in the wild in 2013, was one of the first, if not the first, to implement a scheme close to the Young and Yung protocol in a technically sound way, from its conception [6]. Crypto locker demonstrated the potential to extract large amounts of money through a crypto virus (see below). Since then, there has been an explosion in the number of different ransomware families and variants found building up an industry estimated to be worth up to $1 billion a year [1,7–9]. The crucial point that motivates the current paper is that we would expect the criminals are refining their techniques, not only in terms of the malware component technology, but also regarding the economic tools they use to extract money from victims. In this regard, Crypto locker was relatively unsophisticated when compared with some modern strands.

Recently, the development of ransomware strains as well as changes in the marketplace for malware have greatly reduced the entry barrier for attackers to conduct large-scale ransomware attacks. In this paper, we examine how this mode of cyberattack impacts software vendors and consumer behavior. When victims face an added option to mitigate losses via a ransom payment, both the equilibrium market size and the vendor's profit under optimal pricing can actually increase in the ransom demand. Profit can also increase in the scale of residual losses following a ransom payment.

The vendor restricts software adoption by substantially hiking up price. Social welfare is higher under ransomware compared to the benchmark in both sufficiently low and high-risk settings. However, for intermediate risk, it is better from a social standpoint if consumers do not have an option to pay ransom. We also show that the expected ransom paid is non-monotone in risk, increasing when risk is moderate in spite of a decreasing ransom-paying population. For ransomware attacks on other vectors there can still be an incentive to hike price. However, market size and profits instead weakly decrease in the ransom amount. When studying a generalized model that includes both traditional and ransomware attacks, our results remain robust to a wide range of scenarios, including threat landscapes where ransomware has only a small presence.

CHAPTER 3
SYSTEM ANALYSIS

System analysis is the act, process, or profession of studying an activity (as a procedure, a business, or a physiological function) typically by mathematical means in order to define its goals or purposes and to discover operations and procedure for accomplishing them most efficiently. It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

System analysis is a review of a technological system, like a software package, for troubleshooting, development or improvement purposes. Through in-depth analysis, analysts can uncover errors in code, accessibility issues for end-users or design incompatibilities. Performing an effective systems analysis often requires experts to have knowledge of a software product's or package's requirements so that they can approach their analysis effectively.

Unlike systems administrators, who focus on day-to-day system maintenance, systems analysts consider the viability and effectiveness of a product overall. This allows them to suggest changes or make fixes that improve the system. during analysis, considering the goals of the system is important for solving problems and creating efficiencies. From there, dividing a system into components can make it easier to perform individual analyses that influence the complete system.

A system analysis is a method for identifying and solving problems that looks at each component in the overall system for the purpose of achieving specific goals. The systematic analysis will answer the question, "What does the integrated whole look like and how is it functioning?" Based on this system analysis definition, the overall framework of steps will provide knowledge of interdependencies between people, structure, objects, tasks, and goals. This information can inform the decision-making process or enhance efficient use of time, money, and resources

3.1 EXISTING SYSTEM

Ransomware attacks are among the most disruptive cyber threats, causing significant financial losses while impacting productivity, accessibility, and reputation. Despite their end goals of encryption or locking, ransomware is often designed to evade detection by executing a series of pre-attack API calls, namely “paranoia” activities, for determining a suitable execution environment. In this work, we present a first-of-a-kind effort to utilize such paranoia activities for characterizing ransomware distinguishable behaviors. To this end, we draw-upon more than 3K samples from recent/prominent ransomware families to fingerprint their uniquely leveraged paranoia activities. In this work, we propose a dynamic analysis approach for attributing ransomware samples based on their pre-attack paranoia activities. We execute more than 3,000 ransomware samples that belong to 5 predominant families in a sandboxing environment to collect their behavioral characteristics/features in terms of 23 selected pre-attack evasion API calls that are associated with sensing the execution environment.

LIMITATIONS OF THE EXISTING SYSTEM

- They did not mention what kind of ransomware attacks they are predicting.
- Voting Classifier is not implemented.
- Deployment is not done.

3.2 PROBLEM STATEMENT

- Propose to work on the Bitcoin Heist data set to classify the different malicious transactions.
- The different transactions features are analyzed to predict a classifier label among the classifiers that have been identified as ransomware or associated with malicious activity.
- To use decision tree classifiers and ensemble learning to implement a random forest classifier.

The Results are assessed to evaluate accuracy, precision, and recall.

3.3 PROPOSED SYSTEM

The proposed system is to build a model able to predict the types of ransomware attacks. The process starts with variable identification like dependent and independent variables where we find the target column. Then the pre-processing techniques are applied to deal with missing values the pre-processed data is then used to build a model by dividing the dataset into 7:3 ratios were 70% of the data is used for training purposes that are model learns the pattern and the remaining 30% testing data is used to test the performance of our project. The classification model can be used to predict the bitcoin heist ransomware attack types.

The goal is to develop a machine learning model for Bitcoin heist ransomware Prediction, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithm. Exploration data analysis of variable identification by loading the given dataset, import required libraries packages, analyze the general properties and find duplicate and missing values, checking unique and count values.

ADVANTAGES

- We are implementing particularly on bitcoin ransomware attacks.
- We are implementing the voting classifier.
- Deployment can be done.

CHAPTER 4
REQUIREMENT SPECIFICATIONS

4.1. INTRODUCTION

Requirement analysis determines the requirements of a new system. This project analyses on product and resource requirement, which is required for this successful system. The product requirement includes input and output requirements it give the wants in term of input to produce the required output. The resource requirements give in brief about the software and hardware that are needed to achieve the required functionality.

OVERVIEW OF THE SYSTEM

When you send someone any amount of Bitcoin the senders and receivers bitcoin addresses are saved in a public ledger with the amount transferred.

These Ledgers or records are called as blockchain. Each transaction is encrypted with public key cryptography and a copy of single transaction is saved in all the systems which are processing bitcoins.

SCOPE

I propose to work on the Bitcoin Heist data set to classify the different malicious transactions. The different transactions features are analyzed to predict a classifier label among the classifiers that have been identified as ransomware or associated with malicious activity. I use decision tree classifiers and ensemble learning to implement a random forest classifier. Results are assessed to evaluate accuracy, precision, and recall. I limit the study design to known ransomware identified previously and made available under the Bitcoin transaction graph from January 2009 to December 2018.

FEASIBILITY STUDY

DATA WRANGLING

This section of the report will load in the data, check for cleanliness, and then trim and clean given dataset for analysis. Make sure that the document steps carefully and justify for cleaning decisions.

DATA COLLECTION

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Random Forest, logistic, Decision tree algorithms, Support vector classifier (SVC), Multilayer Perceptron are applied on the Training set and based on the test result accuracy, Test set prediction is done.

PREPROCESSING

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion need to be done.

BUILDING THE CLASSIFICATION MODEL

The prediction of Stellar classification, a high accuracy prediction model is effective because of the following reasons: It provides better results in classification problem.

- It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables.
- It produces out of bag estimate error which has proven to be unbiased in many tests and it is relatively easy to tune with.

CONSTRUCTION OF A PREDICTIVE MODEL

Machine learning needs data gathering have lot of past data. Data gathering have sufficient historical data and raw data. Before data pre-processing, raw data can't be used directly. It's used to pre-process then, what kind of algorithm with model. Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.

HARDWARE AND SOFTWARE SPECIFICATION

4.2 HARDWAREREQUIREMENTS

Hard Disk	80GB and above
RAM	2GB and above
Processor	Intel i3

Table 4.2 Hardware Requirements

4.3 SOFTWARE REQUIREMENTS

Tools	Anaconda with jupyter notebook
Operating System	Windows

Table 4.3 Software Requirements

4.4 TECHNOLOGIES USED

FLASK (WEB FRAMEWORK)

Flask was designed to be easy to use and extend. The idea behind Flask is to build a solid foundation for web applications of different complexity. From then on you are free to plug in any extensions you think you need. Also, you are free to build your own modules. Flask is great for all kinds of projects. It's especially good for prototyping. Framework Flask is a web framework from Python language. Flask provides a library and a collection of codes that can be used to build websites, without the need to do everything from scratch.

CHAPTER 5
SYSTEM DESIGN

System design is the process architecture, components, modules, interfaces and data for a system to satisfy specified requirements. System design could be seen as the application of systems theory to product development. System design is the process of defining the elements of a system such as architecture, modules, and components, the different interfaces of those components and the data go through that system.

A systemic approach is required for a coherent and well-running system. Bottom-Up or Top-Down approach is required to take into account all related variables of the system. A designer uses the modelling languages to express the information and knowledge in a structure of system that is defined by a consistent set of rules and definitions. The designs can be defined in graphical or textual modelling languages.

System design in machine learning involves the process of designing and architecting the components, infrastructure, and workflows required to develop, train, deploy, and maintain machine learning models within a larger system or application. It encompasses various considerations such as data management, model selection, scalability, performance, and integration with existing systems.

Here are some key aspects of system design in machine learning:

1. **Data Acquisition and Storage**: Designing an effective data pipeline to acquire and store the required data for training and inference is crucial. This may involve data collection, preprocessing, cleaning, and integration from various sources. Choosing appropriate data storage systems and formats (e.g., databases, data lakes) to efficiently handle large volumes of data is important.
2. **Feature Engineering**: Designing the feature engineering pipeline involves selecting relevant features, transforming and preprocessing the data, and creating a feature representation that captures the underlying patterns and relationships in the data. This step is critical for model performance and may involve techniques like dimensionality reduction, normalization, and feature selection.
3. **Model Selection and Architecture**: Determining the appropriate machine learning model(s) and architecture(s) based on the problem domain, available data, and desired outcomes. This includes selecting algorithms (e.g., decision trees, neural networks) and

defining model architectures (e.g., deep learning networks) that can effectively learn from the data and make accurate predictions or classifications.

4. **Training and Evaluation**: Designing a robust training pipeline involves splitting data into training, validation, and testing sets, defining appropriate loss functions, selecting optimization algorithms, and setting up training hyperparameters. Regular evaluation of the model's performance is necessary to assess its accuracy, generalization, and potential biases. Techniques like cross-validation and hyperparameter tuning can be employed during this stage.
5. **Deployment and Integration**: Designing the deployment infrastructure and integrating machine learning models into the larger system or application. This may involve creating APIs or microservices for inference, setting up scalable and reliable infrastructure (e.g., cloud platforms, containers), and establishing proper version control and monitoring mechanisms. It is essential to ensure smooth integration with existing systems and maintain compatibility with evolving requirements.
6. **Scalability and Performance**: Designing the system to handle large-scale datasets, high-throughput processing, and real-time or near real-time predictions. Considerations like distributed computing, parallelization, and optimizing model inference are crucial for achieving scalability and low-latency performance.
7. **Monitoring and Maintenance**: Establishing mechanisms for monitoring model performance, detecting and handling drift or concept shift in data, and handling model updates and retraining. Regular maintenance is required to address model performance degradation, handle changing data distributions, and incorporate new features or improvements.

System design in machine learning is an iterative process that requires a deep understanding of the problem domain, the available data, and the system's requirements. It involves making architectural decisions, selecting appropriate tools and technologies, and integrating machine learning components seamlessly into the larger system to deliver reliable and accurate predictions or classifications.

5.1 ARCHITECTURE DIAGRAM

Software environments are complex, and they aren't static. New features are frequently added to accommodate growing customer needs and demands. Your team, even those team members who aren't immersed in the code every day, needs to understand your organization's software architecture so it can scale seamlessly.

This is where software architecture diagrams come in. They give the entire development team a visual overview making it easier to communicate ideas and key concepts in terms everyone can understand.

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries.

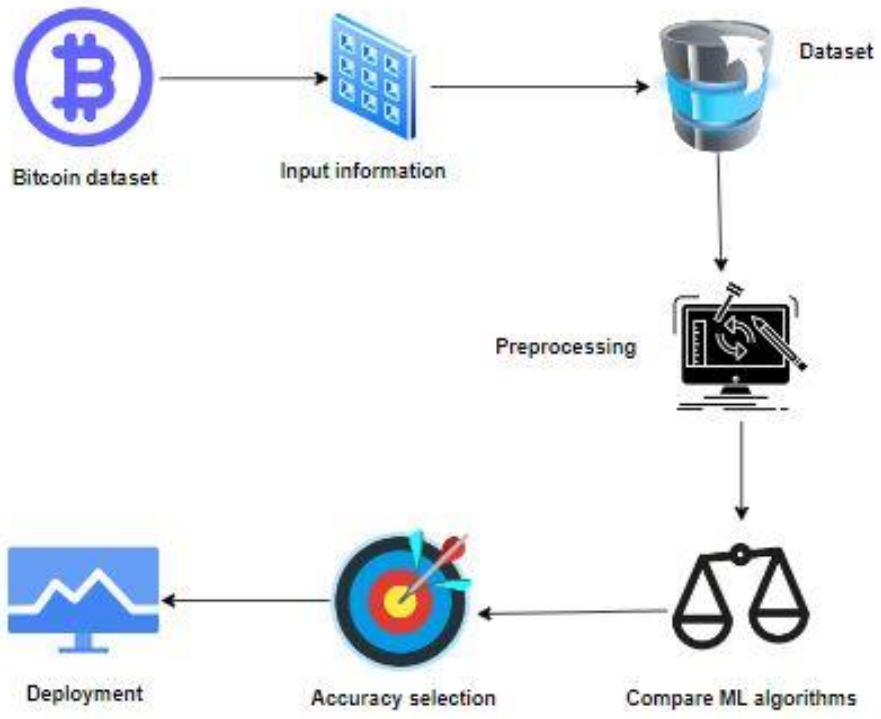


Fig:5.1.1 Architecture Diagram

5.2 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams and timing diagram.

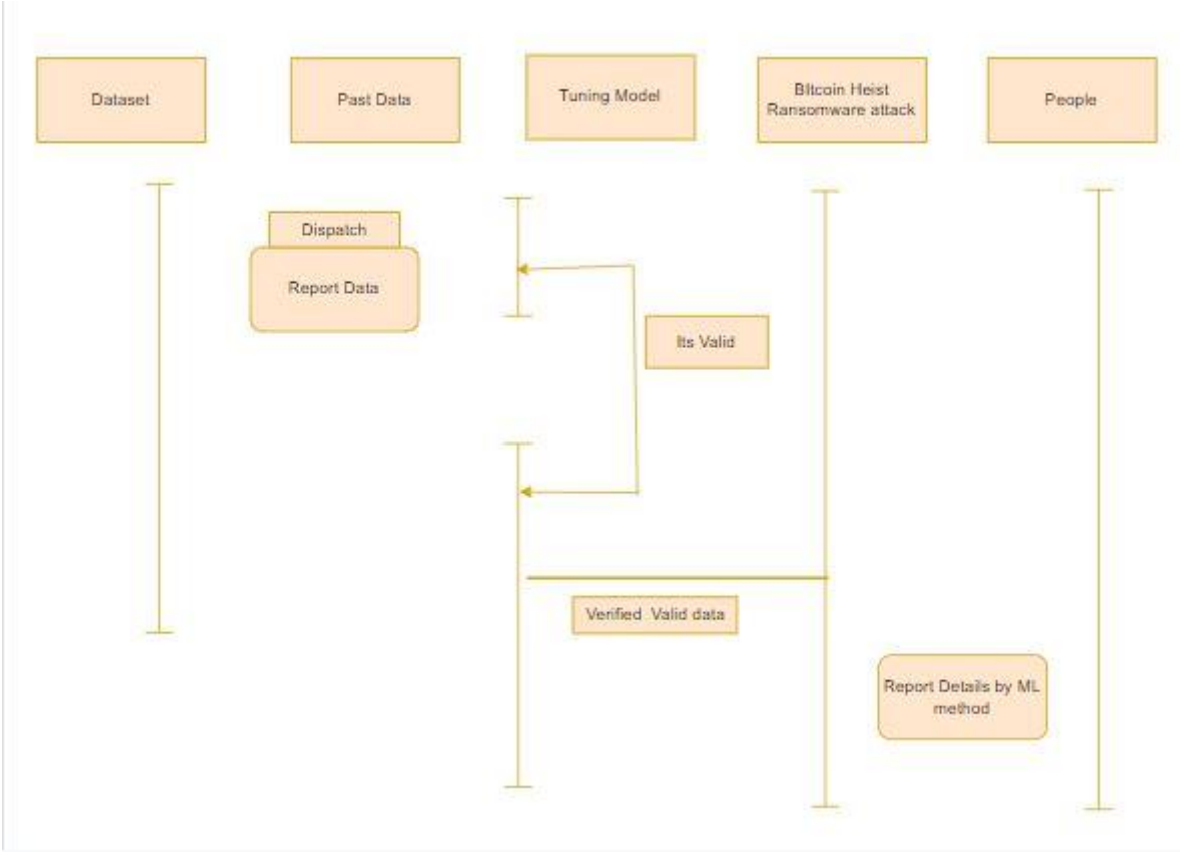


Fig5.2.1 Sequence Diagram

5.3 USECASE DIAGRAM

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development.

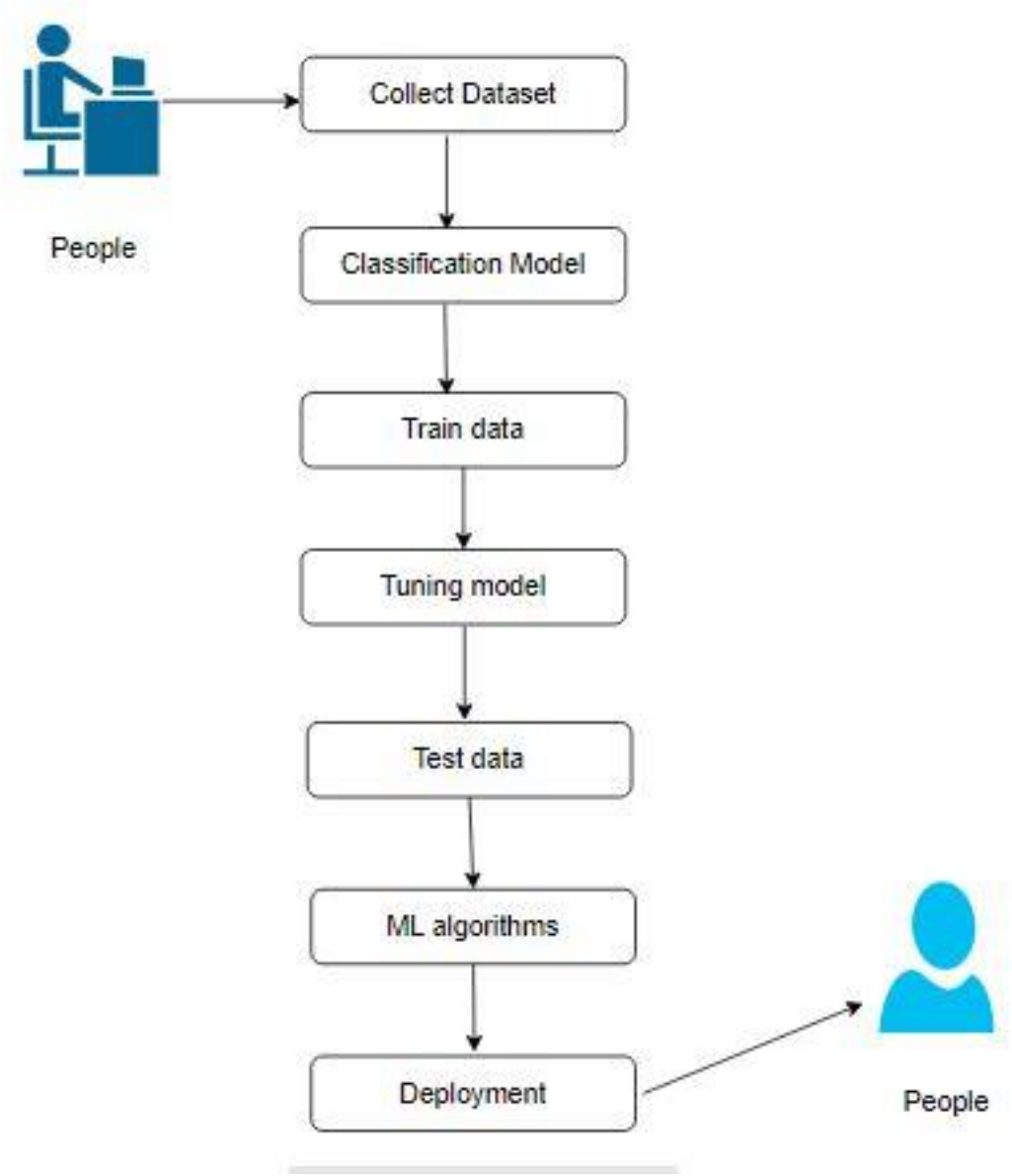


Fig 5.3.2 Use case Diagram

5.4 ACTIVITY DIAGRAM

- Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency.
- An activity diagram shows the overall flow of control.
- The most important shape types:
    i. Rounded rectangles represent activities.
    ii. Diamonds represent decisions.
    iii. Bars represent the start or end of concurrent activities.
    iv. A black circle represents the start of the workflow.
    v. An encircled circle represents the end of the workflow.

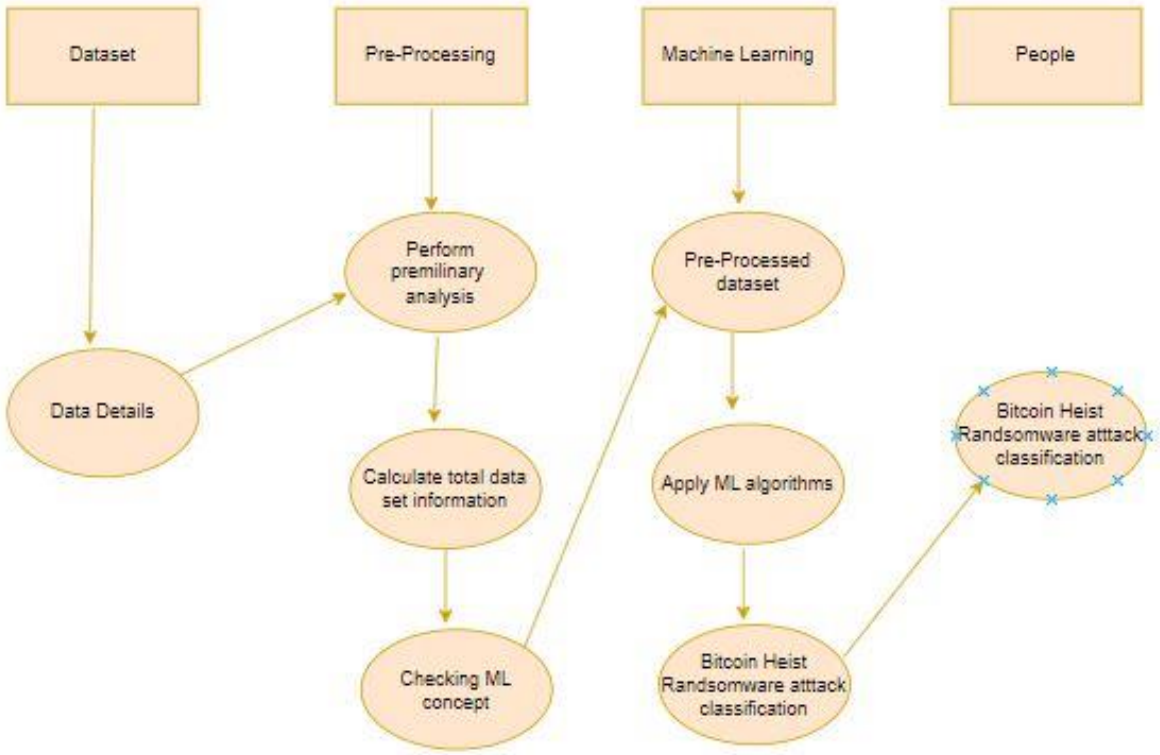


Fig.5.4.1ActivityDiagram

5.5 ENTITY RELATIONSHIP DIAGRAM (ERD)

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

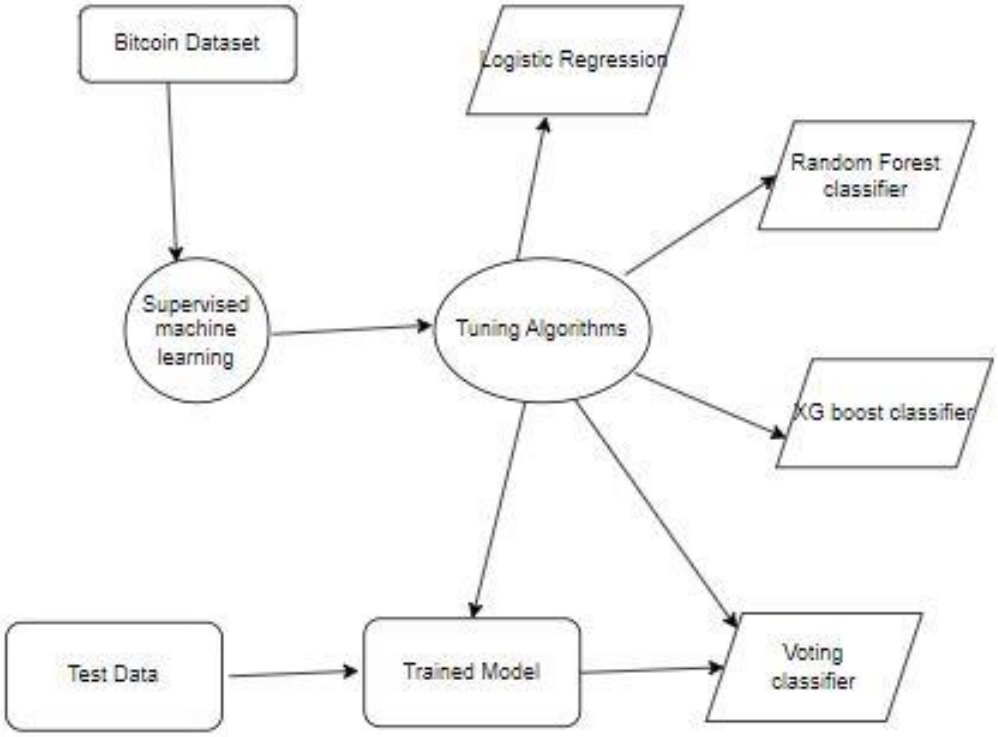


Fig.5.5.1 Entity Relationship Diagram

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 MODULES IMPLEMENTATION

A modular design reduces complexity, facilities change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different part of system. Software with effective modularity is easier to develop because function may be compartmentalized and interfaces are simplified. Software architecture embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

A module implementation consists in a sequence of implementation phrases. An implementation phrase either opens a module, is a type definition or is a sequence of definitions.

- The instruction open modifies the list of opened modules by adding the module name to the list of opened modules, in first position.
- The type definition defines the type for the implementation phrases following the definition.
- The value definition defines some global values.

DATA SCIENCE

Data science is an interdisciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from structured and unstructured data, and apply knowledge and actionable insights from data across a broad range of application domains.

Data science is the field of study that combines domain expertise, programming skills, and knowledge of mathematics and statistics to extract meaningful insights from data.

Data science can be defined as a blend of mathematics, business acumen, tools, algorithms and machine learning techniques, all of which help us in finding out the hidden insights or patterns from raw data which can be of major use in the formation of big business decisions.

DATA SCIENTIST

Data scientists examine which questions need answering and where to find the related data. They have business acumen and analytical skills as well as the ability to mine, clean, and present data. Businesses use data scientists to source, manage, and analyze large amounts of unstructured data.

REQUIRED SKILLS FOR A DATA SCIENTIST

- Programming: Python, SQL, Scala, Java, R, MATLAB.
- Machine Learning: Natural Language Processing, Classification, Clustering.
- Data Visualization: Tableau, SAS, D3.js, Python, Java, R libraries.
- Big data platforms: MongoDB, Oracle, Microsoft Azure, Cloudera.

ARTIFICIAL INTELLIGENCE (AI)

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. The term may also be applied to any machine that exhibits traits associated with a human mind such as learning and problem-solving.

Artificial intelligence (AI) is intelligence demonstrated by machines, as opposed to the natural intelligence displayed by humans or animals. Leading AI textbooks define the field as the study of "intelligent agents" any system that perceives its environment and takes actions that maximize its chance of achieving its goals. Some popular accounts use the term "artificial intelligence" to describe machines that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving", however this definition is rejected by major AI researchers.

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, and speech recognition and machine vision.

In general, AI systems work by ingesting large amounts of labelled training data, analysing the data for correlations and patterns, and using these patterns to make predictions about future states. In this way, a chatbot that is fed examples of text chats can learn to produce life like exchanges with people, or an image recognition tool can learn to identify and describe objects in images by reviewing millions of examples. AI programming focuses on three cognitive skills: learning, reasoning and self-correction.

LEARNING PROCESSES

This aspect of AI programming focuses on acquiring data and creating rules for how to turn the data into actionable information. The rules, which are called algorithms, provide computing devices with step-by-step instructions for how to complete a specific task.

REASONING PROCESSES

This aspect of AI programming focuses on choosing the right algorithm to reach a desired outcome.

SELF-CORRECTION PROCESSES

This aspect of AI programming is designed to continually fine-tune algorithms and ensure they provide the most accurate results possible.

AI is important because it can give enterprises insights into their operations that they may not have been aware of previously and because, in some cases, AI can perform tasks better than humans. Particularly when it comes to repetitive, detail-oriented tasks like analysing large numbers of legal documents to ensure relevant fields are filled in properly, AI tools often complete jobs quickly and with relatively few errors.

Artificial neural networks and deep learning artificial intelligence technologies are quickly evolving, primarily because AI processes large amounts of data much faster and makes predictions more accurately than humanly possible.

NATURAL LANGUAGE PROCESSING (NLP)

Natural language processing (NLP) allows machines to read and understand human language. A sufficiently powerful natural language processing system would enable natural-language user interfaces and the acquisition of knowledge directly from human-written sources, such as newswire texts. Some straightforward applications of natural language processing include information retrieval, text mining, question answering and machine translation. Many current approaches use word co-occurrence frequencies to construct syntactic representations of text. "Keyword spotting" strategies for search are popular and scalable but dumb; a search query for "dog" might only match documents with the literal word "dog" and miss a document with the word "poodle". "Lexical affinity" strategies use the occurrence of words such as "accident" to assess the sentiment of a document. Modern

statistical NLP approaches can combine all these strategies as well as others, and often achieve acceptable accuracy at the page or paragraph level. Beyond semantic NLP, the ultimate goal of "narrative" NLP is to embody a full understanding of common-sense reasoning. By 2019, transformer-based deep learning architectures could generate coherent text.

MACHINE LEARNING (ML)

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python. Process of training and prediction involves use of specialized algorithms. It feed the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated in to three categories. There are supervised learning, unsupervised learning and reinforcement learning. Supervised learning program is both given the input data and the corresponding labelling to learn data has to be labelled by a human being beforehand. Unsupervised learning is no labels. It provided to the learning algorithm. This algorithm has to figure out the clustering of the input data. Finally, Reinforcement learning dynamically interacts with its environment and it receives positive or negative feedback to improve its performance.

Data scientists use many different kinds of machine learning algorithms to discover patterns in python that lead to actionable insights. At a high level, these different algorithms can be classified into two groups based on the way they “learn” about data to make predictions: supervised and unsupervised learning. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function from input variables(X) to discrete output variables(y). In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc.

Process Of Machine Learning

Supervised Machine Learning is the majority of practical machine learning uses supervised learning. Supervised learning is where have input variables (X) and an output variable (y) and use an algorithm to learn the mapping function from the input to the output is y = f(X).

PREPARING THE DATASET

This dataset contains 159664 records of features extracted from bitcoin, which were then used to find the bitcoin heist.

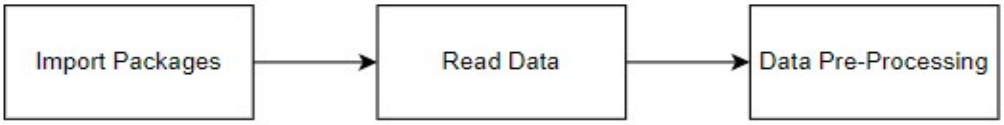
LIST OF MODULES

1. Data pre-processing
2. Data visualization
3. Logistic regression
4. Random Forest
5. XG Boost classifier
6. Voting classifier
7. Deployment

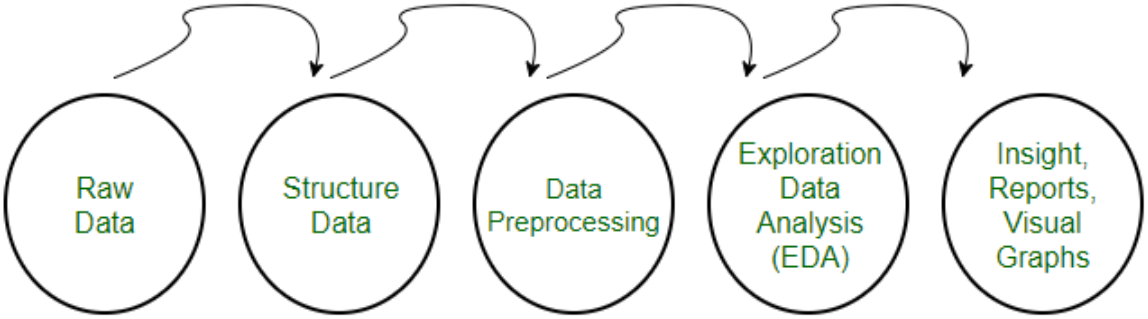
6.2 MODULE DESCRIPTION

DATA PRE-PROCESSING

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters.



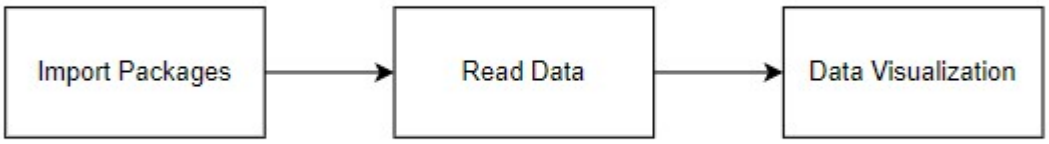
The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. It as machine learning engineers uses this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model.



DATA VISUALIZATION

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.



In business, for numerous periods, it happens that we need to look at the exhibitions of two components or two situations. A conventional methodology is to experience the massive information of both the circumstances and afterward examine it. This will clearly take a great deal of time.

It can tackle the difficulty of placing the information of both perspectives into the pictorial structure. This will unquestionably give a superior comprehension of the circumstances. For instance, Google patterns assist us with understanding information identified with top ventures or inquiries in pictorial or graphical structures.

With the representation of the information, organizations present another arrangement of correspondence. Rather than sharing the cumbersome information, sharing the visual data will draw in and pass on across the data which is more absorbable.

VOTING CLASSIFIER

A Voting Classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. The idea is instead of creating separate dedicated models and finding the accuracy for each them, we create a single model which trains by these models and predicts output based on their combined majority of voting for each output class.

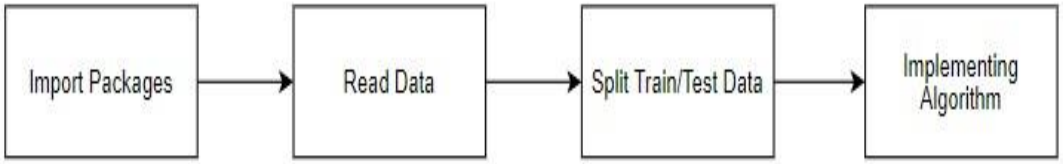
Voting Classifier supports two types of voting namely,

1. **HARD VOTING**

   In hard voting, the predicted output class is a class with the highest majority of votes i.e., the class which had the highest probability of being predicted by each of the classifiers. Suppose three classifiers predicted the *output class (A, A, B)*, so here the majority predicted *A* as output. Hence *A* will be the final prediction.

2. **SOFT VOTING**

   In soft voting, the output class is the prediction based on the average of probability given to that class. Suppose given some input to three models, the prediction probability for class *A = (0.30, 0.47, 0.53)* and *B = (0.20, 0.32, 0.40)*. So, the average for class *A is 0.4333* and *B is 0.3067*, the winner is clearly class *A* because it had the highest probability averaged by each classifier.

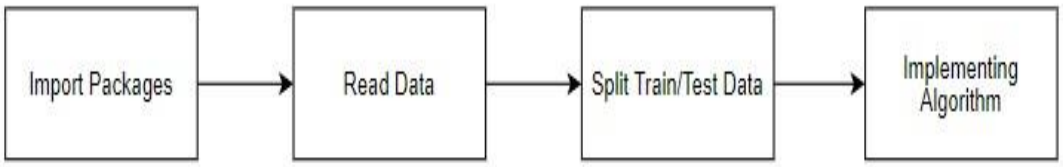


The key advantage of using a voting classifier is its ability to improve the overall accuracy and robustness of a model. By aggregating the predictions of multiple classifiers, the ensemble can overcome the limitations and biases of individual models. It effectively harnesses the collective wisdom of diverse classifiers, capturing different aspects of the data and decision boundaries. This can lead to better generalization, reduced overfitting, and improved performance on unseen data.

RANDOM FOREST CLASSIFIER

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.



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.

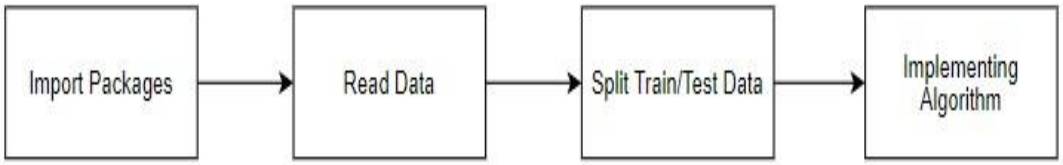
As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

LOGISTIC REGRESSION

Logistic regression is another powerful supervised ML algorithm used for binary classification problems (when target is categorical). The best way to think about logistic regression is that it is a linear regression but for classification problems. Logistic regression essentially uses a logistic function defined below to model a binary output variable. The primary difference between linear regression and logistic regression is that logistic regression's range is bounded between 0 and 1. In addition, as opposed to linear regression, logistic regression does not require a linear relationship between inputs and output variables.



The logistic regression algorithm estimates the parameters of the sigmoid function by minimizing a cost function, typically using optimization techniques like gradient descent. The most common cost function used in logistic regression is the log-loss function, which measures the difference between the predicted probabilities and the true class labels. The goal is to find the optimal values for the parameters that minimize the log-loss and provide the best fit to the data.

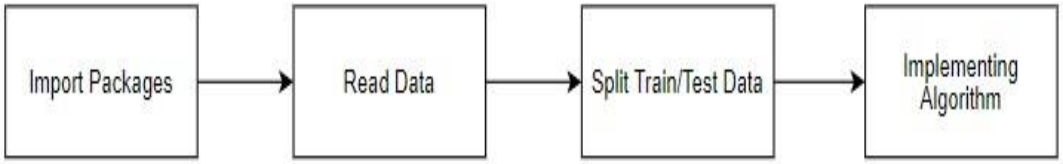
During training, logistic regression learns the weights or coefficients associated with each feature, indicating their importance in predicting the target class. These coefficients can be interpreted as the log-odds or log-likelihood of the target class given the input features. By applying the sigmoid function to these log-odds, we obtain the predicted probabilities of the target class.

logistic regression remains a widely used and versatile algorithm in machine learning. It serves as a fundamental building block for more advanced techniques and can provide a solid baseline for binary classification tasks. With careful feature engineering and model tuning, logistic regression can deliver reliable and interpretable predictions in various domains.

XG BOOST CLASSIFIER

XG Boost is an optimized distributed gradient boosting library designed for efficient and scalable training of machine learning models. It is an ensemble learning method that combines the predictions of multiple weak models to produce a stronger prediction.

XG Boost stands for “Extreme Gradient Boosting” and it has become one of the most popular and widely used machine learning algorithms due to its ability to handle large datasets and its ability to achieve state-of-the-art performance in many machine learning tasks such as classification and regression.



One key innovation of XG Boost is its focus on optimizing the objective function through the introduction of gradient information. This allows XG Boost to make more informed decisions during training, leading to faster convergence and better overall performance. The algorithm calculates the gradients of the loss function with respect to the predictions, and these gradients are used to guide the construction of subsequent trees.

The algorithm also supports parallelization and distributed computing, making it highly scalable and efficient for handling large datasets. It optimizes memory usage and computational resources, enabling faster training and prediction times. Moreover, XG Boost provides built-in support for handling missing values and handling categorical features without the need for pre-processing.

XG Boost has gained popularity and achieved state-of-the-art results in numerous machine learning competitions and real-world applications. Its exceptional performance and versatility make it suitable for a wide range of tasks, including fraud detection, text classification, customer churn prediction, and recommendation systems.

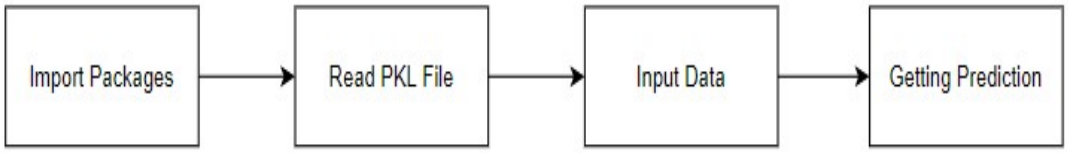
To use XG Boost, one must specify the hyperparameters that control the model's behaviour, such as the learning rate, number of trees, and regularization parameters. These hyperparameters can be tuned through techniques like cross-validation to optimize the model's performance on the specific task and dataset at hand.

DEPLOYMENT

Flask is an API of Python that allows us to build up web-applications. It was developed by Armin Ronacher.

Flask is a web framework for Python, meaning that it provides functionality for building web applications, including managing HTTP requests and rendering templates and also, we can add to this application to create our API.

Flask's framework is more explicit than Django framework and is also easier to learn because it has less base code to implement a simple web-Application.



The deployment consists of Flask framework implemented with basic HTML and CSS which are interconnected by the machine learning algorithms (modules).

Deploying a module in a project is a critical step in bringing the functionality to users and ensuring a smooth integration within the overall system. Following proper development practices, thorough testing, and effective deployment strategies contribute to successful module deployment.

ADVANTAGES OF FLASK

- Higher compatibility with latest technologies.
- Technical experimentation.
- Easier to use for simple cases.
- Codebase size is relatively smaller.
- High scalability for simple applications.
- Easy to build a quick prototype.
- Routing URL is easy.
- Easy to develop and maintain applications.

CHAPTER 7

TESTING

7.1 TEST PROCEDURE

SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example, the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walk through. Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases. Static analysis is used to investigate the structural properties of the Source code. Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

7.2 TEST DATA AND OUTPUT

UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

7.3 FUNCTIONALTESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

PERFORMANCE TESTING

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

STRESS TESTING

Stress Test is those tests designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

STRUCTURED TESTING

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all oops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.
- Checking attribute for their correctness.
- Handling end of file condition,
- I/O errors, buffer problem sand textual errors in output information

INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 CONCLUSION

The analytical process started from data cleaning and processing, missing value, exploratory analysis and finally model building and evaluation. The best accuracy on public test set of higher accuracy score algorithm will be find out. The founded one is used in the application which can help to find the Bitcoin Heist ransomware attack.

8.2 FUTURE ENHANCEMENT

- Deploying the project in the cloud.
- To optimize the work to implement in the IOT system.

APPENDIX-1

SAMPLE CODING

FLASK DEPLOYMENT

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

app = Flask(__name__)
model = joblib.load('voting.pkl')

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

@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    print(final_features)
    prediction = model.predict(final_features)

    output = prediction[0]
    print(output)
    if output == 0:
        output ='Cerber'
    elif output == 1:
        output ='Crypto'
    elif output == 2:
```

```
        output ='CryptoLocker'
    elif output == 3:
        output ='CryptoWall'
    elif output == 4:
        output ='Locky'
    elif output == 5:
        output ='White'


    return render_template('index.html', prediction_text='THE BITCOIN ATTACK IS
PREDICTED AS : {}'.format(output))

if __name__ == "__main__":
    app.run(host="localhost", port=5000)
```

MODULE - 1

DATA PRE-PROCESSING

```
import pandas as p
import numpy as n
import warnings
warnings.filterwarnings('ignore')
data = p.read_csv('Data.csv')
data.head()
data.shape
df = data.dropna()
df.shape
df.isnull().sum()
df.columns
df.describe()
df.length.unique()
p.crosstab(df.label,df.year)
```

```
p.Categorical(df['label']).describe()
p.Categorical(df['year']).describe()
print("Days: ", sorted(df['day'].unique()))
df['length'].value_counts()
df.duplicated()
df.duplicated().sum()
df.columns
print("Minimum Income : ", df.income.min())
print("Maximum Income : ", df.income.max())
df.info()
df['label']
from sklearn.preprocessing import LabelEncoder
var_mod = ['label']
le = LabelEncoder()
for i in var_mod:
  df[i] = le.fit_transform(df[i]).astype(int)
df['label']
```

MODULE - 2

#DATA VISUALIZATION

```
import library packages
import pandas as p
import matplotlib.pyplot as plt
import seaborn as s
import numpy as n
import warnings
warnings.filterwarnings('ignore')
df = p.read_csv("Data.csv")
df
df.columns
df['year'].hist(figsize=(10,4), color='c')
plt.xlabel('YEAR')
```

```
plt.ylabel('COUNT')
plt.title('Yearwise Count')
#Propagation by variable
def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable, fontsize = 15)
    return n.round(dataframe_pie/df.shape[0]*100,2)

PropByVar(df, 'label')
#Propagation by variable
def PropByVar(df, variable):
    dataframe_pie = df[variable].value_counts()
    ax = dataframe_pie.plot.pie(figsize=(8,8), autopct='%1.2f%%', fontsize = 12)
    ax.set_title(variable, fontsize = 15)
    return n.round(dataframe_pie/df.shape[0]*100,2)
PropByVar(df, 'year')
# Heatmap plot diagram
fig, ax = plt.subplots(figsize=(15,5))
s.heatmap(df.corr(), ax=ax, annot=True, cmap='CMRmap')
plt.figure(figsize=(12,5))
plt.scatter(df["year"],df["label"],color="m")
plt.title('Scatter Plot')
plt.show()
plt.plot(df["year"], df["label"], color='red')
plt.xlabel('year')
plt.ylabel('Attack Types')
plt.title('Distribution of Year and Attack_Types')
plt.show()
import seaborn as s
s.boxplot(df['day'], color='y')
plt.figure(figsize= (10,4))
df["year"].plot(kind='density')
```

MODULE - 3

#LOGISTIC REGRESSION

```
import library packages
import pandas as p
import warnings
warnings.filterwarnings('ignore')
#Load given dataset
data = p.read_csv("Data.csv")
df = data.dropna()
df
del df['Unnamed: 0']
del df['address']
df['label']=df['label'].map({'montrealCryptXXX':'Crypto','montrealCryptoLocker':'CryptoLoc
ker','paduaCryptoWall':'CryptoWall','princetonCerber':'Cerber','princetonLocky':'Locky','whit
e':'White'})
df.info()
df['label'].unique()
from sklearn.preprocessing import LabelEncoder
col = ['label']
label = LabelEncoder()
for i in col:
    df[i] = label.fit_transform(df[i]).astype(int)
df['label'].unique()
df
X = df.drop(labels='label', axis=1)
y = df.loc[:,'label']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)
print("Number of Training Dataset: ", len(X_train))
print("Number of Testing Dataset: ", len(X_test))
```

```
print("Total Number of Dataset: ", len(X_train)+len(X_test))
Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,
hamming_loss, plot_confusion_matrix
Training Process
lr = LogisticRegression()
lr.fit(X_train,y_train)
predicted_lr = lr.predict(X_test)
Getting Accuracy
accuracy = accuracy_score(y_test,predicted_lr)
print('Accuracy of Logistic Regression is: ',accuracy*100)
Finding Loss
loss = hamming_loss(y_test,predicted_lr)
print('Loss of Logistic Regression is: ',loss*100)
Finding Clasiification Report
cr = classification_report(y_test,predicted_lr)
print('Classification report\n----------------------\n',cr)
Finding Confusion Matrix
cm = confusion_matrix(y_test,predicted_lr)
print('Confusion matrix\n----------------\n',cm)
import matplotlib.pyplot as plt
from sklearn.metrics import ConfusionMatrixDisplay
cm = confusion_matrix(y_test, predicted_lr, labels=lr.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=lr.classes_)
disp.plot()
plt.show()
import matplotlib.pyplot as plt
DF = p.DataFrame()
DF["y_test"] = y_test
DF["predicted"] = predicted_lr
DF.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(DF["predicted"][:100], marker='x', linestyle='dashed', color='red')
```

```
plt.plot(DF["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE - 4

#RANDOM FOREST CLASSIFIER

```
import library packages
import pandas as p
import warnings
warnings.filterwarnings('ignore')
#Load given dataset
df = p.read_csv("Data.csv")
df.tail()
df = df.dropna()
df.head()
del df['Unnamed: 0']
del df['address']
df['label']=df['label'].map({'montrealCryptXXX':'Crypto','montrealCryptoLocker':'CryptoLoc
ker','paduaCryptoWall':'CryptoWall','princetonCerber':'Cerber','princetonLocky':'Locky','whit
e':'White'})
df.info()
df['label'].unique()
df.head()
from sklearn.preprocessing import LabelEncoder
col = ['label']
label = LabelEncoder()
for i in col:
    df[i] = label.fit_transform(df[i]).astype(int)
df['label'].unique()
df.head()
X = df.drop(labels='label', axis=1)
y = df.loc[:,'label']
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)
print("Number of Training Dataset: ", len(X_train))
print("Number of Testing Dataset: ", len(X_test))
print("Total Number of Dataset: ", len(X_train)+len(X_test))
print("Number of Training Dataset: ", len(y_train))
print("Number of Testing Dataset: ", len(y_test))
print("Total Number of Dataset: ", len(y_train)+len(y_test))
```

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,
hamming_loss, plot_confusion_matrix
```

Training Process

```
rfc = RandomForestClassifier()
rfc.fit(X_train,y_train)
predicted_rfc = rfc.predict(X_test)
```

Getting Accuracy

```
accuracy = accuracy_score(y_test,predicted_rfc)
print('Accuracy of Random Forest Classifier is: ',accuracy*100)
```

Finding Loss

```
loss = hamming_loss(y_test,predicted_rfc)
print('Loss of Random Forest Classifier is: ',loss*100)
```

Finding Clasiification Report

```
cr = classification_report(y_test,predicted_rfc)
print('Classification report\n----------------------\n',cr)
```

Finding Confusion Matrix

```
cm = confusion_matrix(y_test,predicted_rfc)
print('Confusion matrix\n----------------\n',cm)
import matplotlib.pyplot as plt
from sklearn.metrics import ConfusionMatrixDisplay
cm = confusion_matrix(y_test, predicted_rfc, labels=rfc.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=rfc.classes_)
disp.plot()
plt.show()
```

```
import matplotlib.pyplot as plt
DF = p.DataFrame()
DF["y_test"] = y_test
DF["predicted"] = predicted_rfc
DF.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(DF["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(DF["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE - 5

#XGBOOST CLASSIFIER

```
#import library packages
import pandas as p
import warnings
warnings.filterwarnings('ignore')
#Load given dataset
data = p.read_csv("Data.csv")
df = data.dropna()
df
del df['Unnamed: 0']
del df['address']
df['label']=df['label'].map({'montrealCryptXXX':'Crypto','montrealCryptoLocker':'CryptoLoc
ker','paduaCryptoWall':'CryptoWall','princetonCerber':'Cerber','princetonLocky':'Locky','whit
e':'White'})
df.info()
df['label'].unique()
from sklearn.preprocessing import LabelEncoder
col = ['label']
label = LabelEncoder()
for i in col:
```

```
    df[i] = label.fit_transform(df[i]).astype(int)
df['label'].unique()
df
X = df.drop(labels='label', axis=1)
y = df.loc[:,'label']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)
print("Number of Training Dataset: ", len(X_train))
print("Number of Testing Dataset: ", len(X_test))
print("Total Number of Dataset: ", len(X_train)+len(X_test))
```

XGBoost Classifier

```
from xgboost import XGBClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,
hamming_loss, plot_confusion_matrix
```

Training Process

```
xg = XGBClassifier()
xg.fit(X_train,y_train)
predicted_xg = xg.predict(X_test)
```

Getting Accuracy

```
accuracy = accuracy_score(y_test,predicted_xg)
print('Accuracy of XGBoost Classifier is: ',accuracy*100)
```

Finding Loss

```
loss = hamming_loss(y_test,predicted_xg)
print('Loss of XGBoost Classifier is: ',loss*100)
```

Finding Clasiification Report

```
cr = classification_report(y_test,predicted_xg)
print('Classification report\n----------------------\n',cr)
```

Finding Confusion Matrix

```
cm = confusion_matrix(y_test,predicted_xg)
print('Confusion matrix\n----------------\n',cm)
import matplotlib.pyplot as plt
from sklearn.metrics import ConfusionMatrixDisplay
cm = confusion_matrix(y_test, predicted_xg, labels=xg.classes_)
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=xg.classes_)
disp.plot()
plt.show()
import matplotlib.pyplot as plt
DF = p.DataFrame()
DF["y_test"] = y_test
DF["predicted"] = predicted_xg
DF.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(DF["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(DF["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()
```

MODULE – 6

#VOTING CLASSIFIER

```
import library packages
import pandas as p
import warnings
warnings.filterwarnings('ignore')
#Load given dataset
data = p.read_csv("Data.csv")
df = data.dropna()
df
del df['Unnamed: 0']
del df['address']
df['label']=df['label'].map({'montrealCryptXXX':'Crypto','montrealCryptoLocker':'CryptoLoc
ker','paduaCryptoWall':'CryptoWall','princetonCerber':'Cerber','princetonLocky':'Locky','whit
e':'White'})
df.info()
```

```
df['label'].unique()
from sklearn.preprocessing import LabelEncoder
col = ['label']
label = LabelEncoder()
for i in col:
    df[i] = label.fit_transform(df[i]).astype(int)
df['label'].unique()
df
X = df.drop(labels='label', axis=1)
y = df.loc[:,'label']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1,
stratify=y)
print("Number of Training Dataset: ", len(X_train))
print("Number of Testing Dataset: ", len(X_test))
print("Total Number of Dataset: ", len(X_train)+len(X_test))
```

Voting Classifier

```
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import VotingClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score,
hamming_loss, plot_confusion_matrix
```

Training Process

```
xg = XGBClassifier()
rf = RandomForestClassifier()
lr = LogisticRegression()
vc = VotingClassifier(estimators=[('XGBoost', xg), ('RandomForestClassifier', rf),
('LogisticRegression', lr)], voting='hard')
vc.fit(X_train,y_train)
pred_vc = vc.predict(X_test)
```

Getting Accuracy

```
accuracy = accuracy_score(y_test,pred_vc)
print('Accuracy of Voting Classifier is: ',accuracy*100)
```

```
Finding Loss
loss = hamming_loss(y_test,pred_vc)
print('Loss of Voting Classifier is: ',loss*100)
Finding Clasiification Report
cr = classification_report(y_test,pred_vc)
print('Classification report\n----------------------\n',cr)
Finding Confusion Matrix
cm = confusion_matrix(y_test,pred_vc)
print('Confusion matrix\n----------------\n',cm)
import matplotlib.pyplot as plt
from sklearn.metrics import ConfusionMatrixDisplay
cm = confusion_matrix(y_test, pred_vc, labels=vc.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=vc.classes_)
disp.plot()
plt.show()
import matplotlib.pyplot as plt
DF = p.DataFrame()
DF["y_test"] = y_test
DF["predicted"] = pred_vc
DF.reset_index(inplace=True)
plt.figure(figsize=(20, 5))
plt.plot(DF["predicted"][:100], marker='x', linestyle='dashed', color='red')
plt.plot(DF["y_test"][:100],  marker='o', linestyle='dashed', color='green')
plt.show()

Saving Model
import joblib
joblib.dump(vc,'voting.pkl')
```

APPENDIX - 2

SNAPSHOTS

DATA PRE-PROCESSING

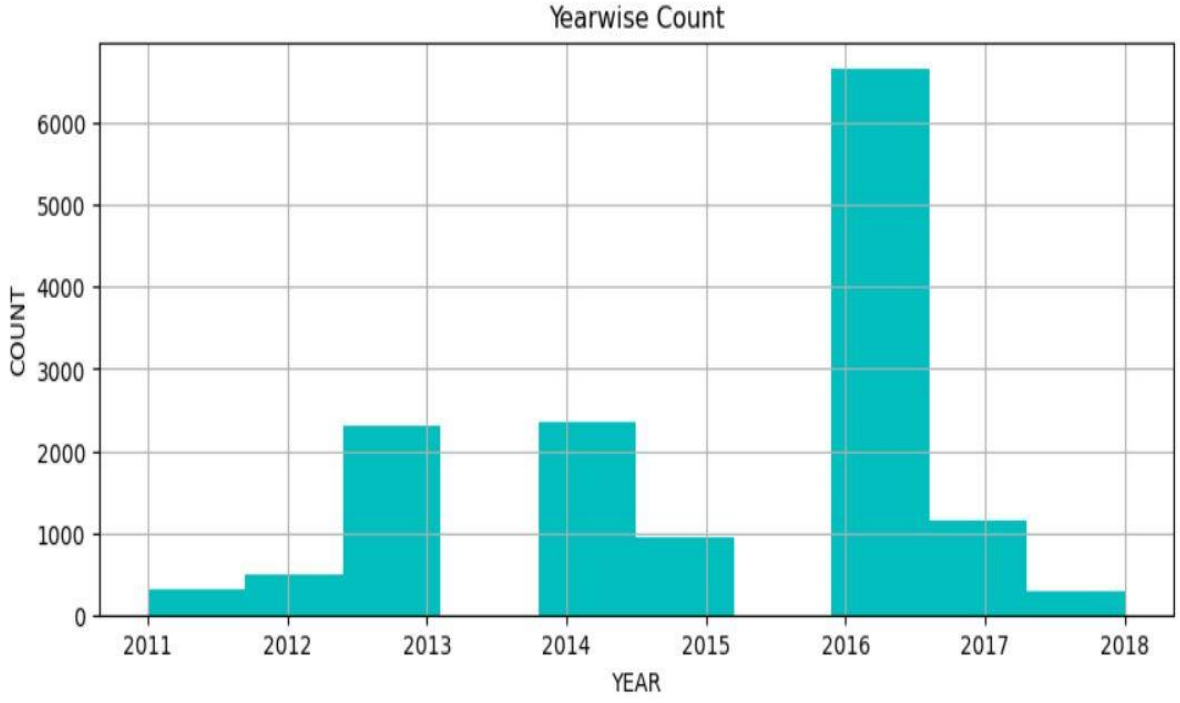
year	2011	2012	2013	2014	2015	2016	2017	2018
label								
montrealCryptXXX	0	0	0	0	0	2419	0	0
montrealCryptoLocker	21	181	1958	259	0	0	0	0
paduaCryptoWall	0	0	0	1778	641	0	0	0
princetonCerber	0	0	0	0	0	1544	875	0
princetonLocky	0	0	0	0	0	2406	13	0
white	297	318	341	306	306	284	272	295

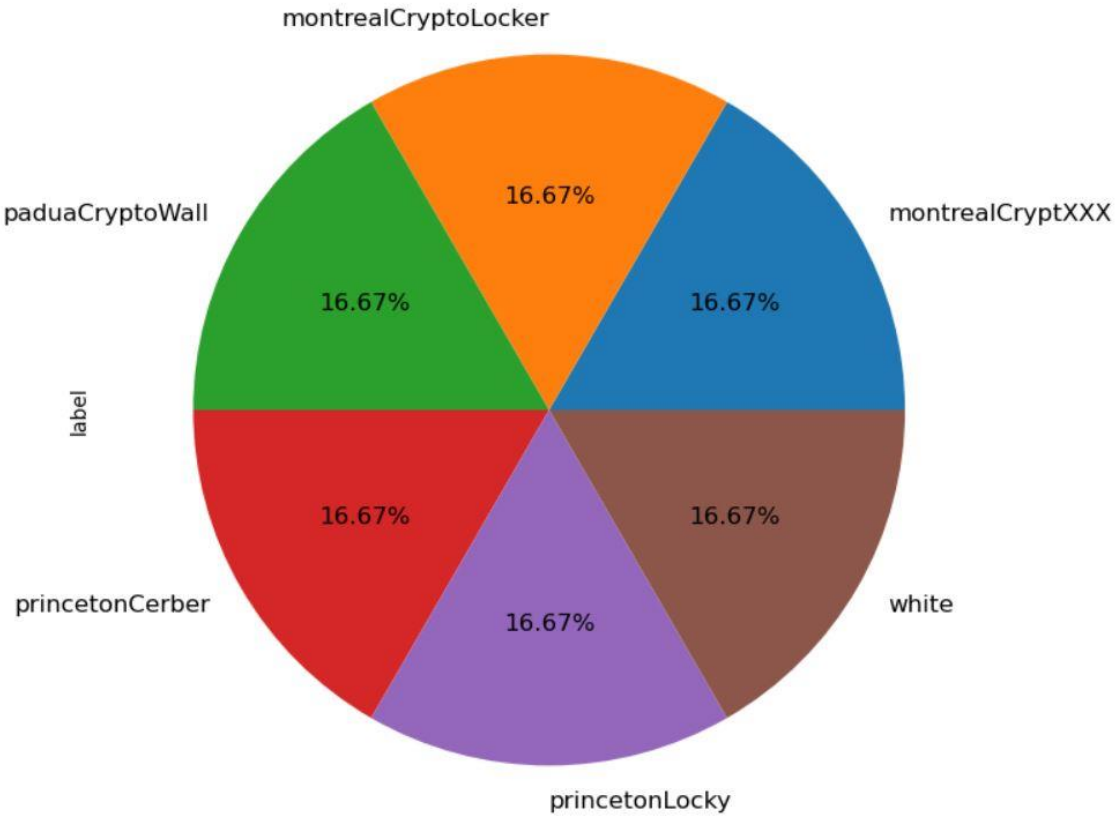
	counts	freqs
categories		
montrealCryptXXX	2419	0.166667
montrealCryptoLocker	2419	0.166667
paduaCryptoWall	2419	0.166667
princetonCerber	2419	0.166667
princetonLocky	2419	0.166667
white	2419	0.166667

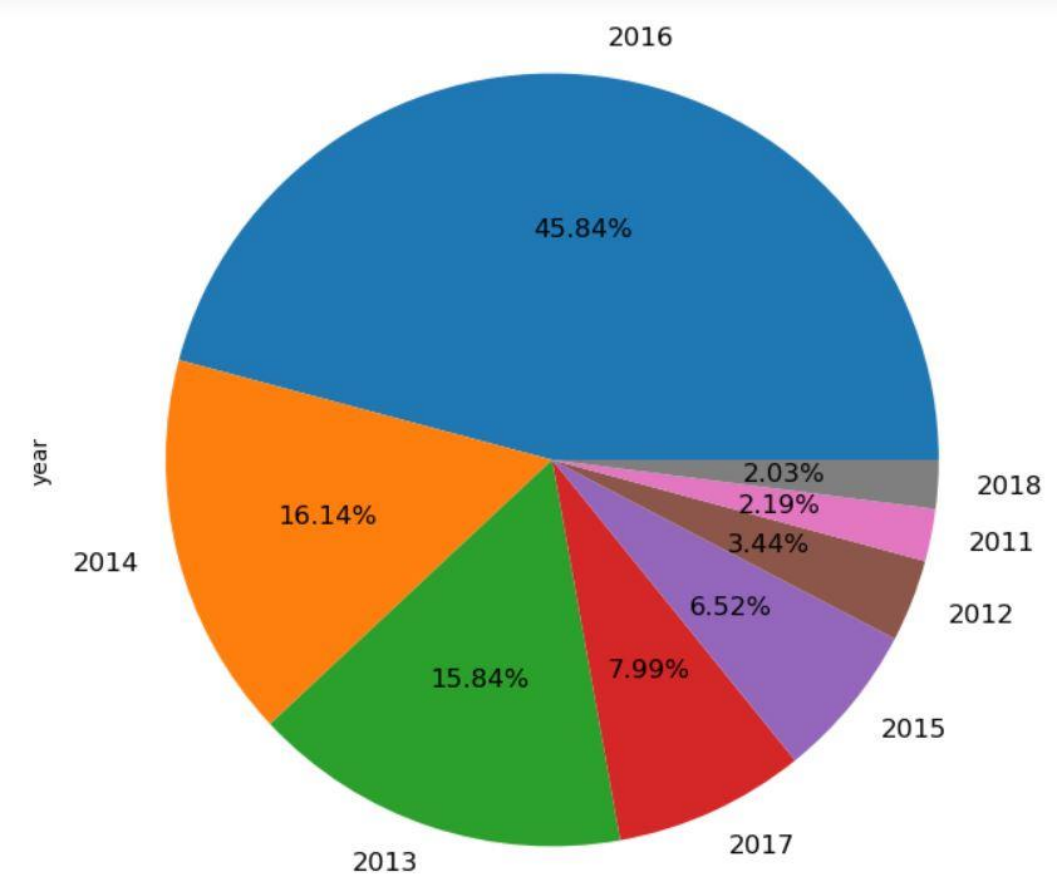
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14514 entries, 0 to 14513
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   14514 non-null  int64
 1   address      14514 non-null  object
 2   year         14514 non-null  int64
 3   day          14514 non-null  int64
 4   length       14514 non-null  int64
 5   weight       14514 non-null  float64
 6   count        14514 non-null  int64
 7   looped       14514 non-null  int64
 8   neighbors    14514 non-null  int64
 9   income       14514 non-null  float64
 10  label        14514 non-null  object
dtypes: float64(2), int64(7), object(2)
memory usage: 1.2+ MB
```

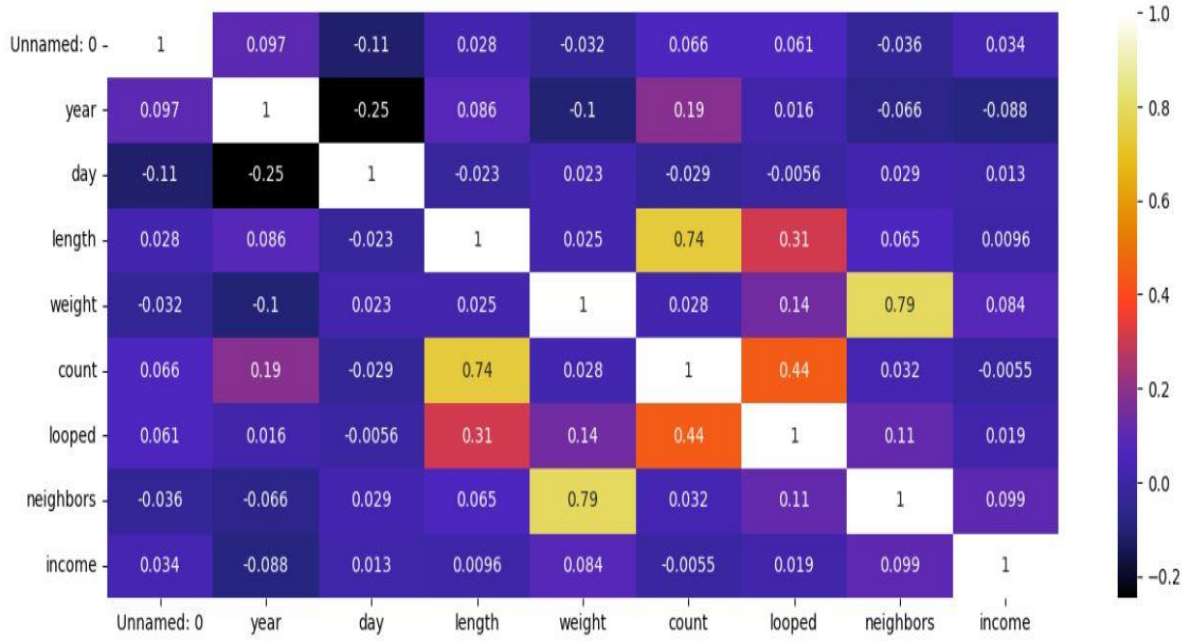
categories	counts	freqs
2011	318	0.021910
2012	499	0.034381
2013	2299	0.158399
2014	2343	0.161430
2015	947	0.065247
2016	6653	0.458385
2017	1160	0.079923
2018	295	0.020325

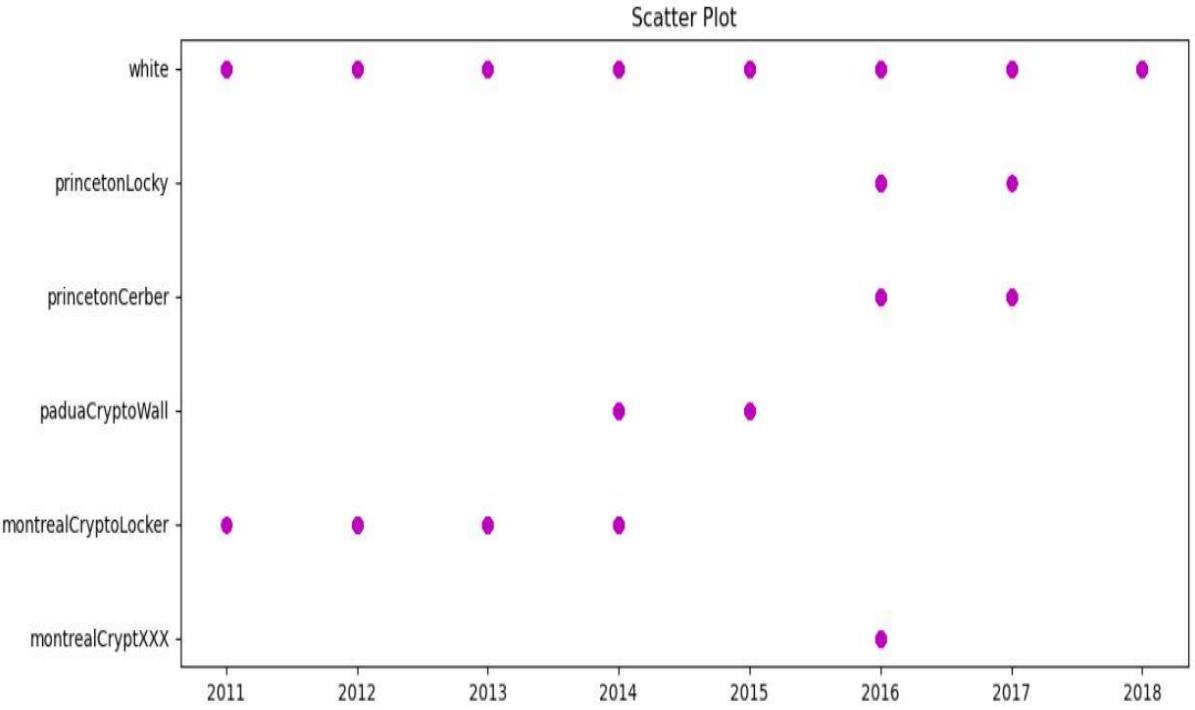
DATA VISUALIZATION

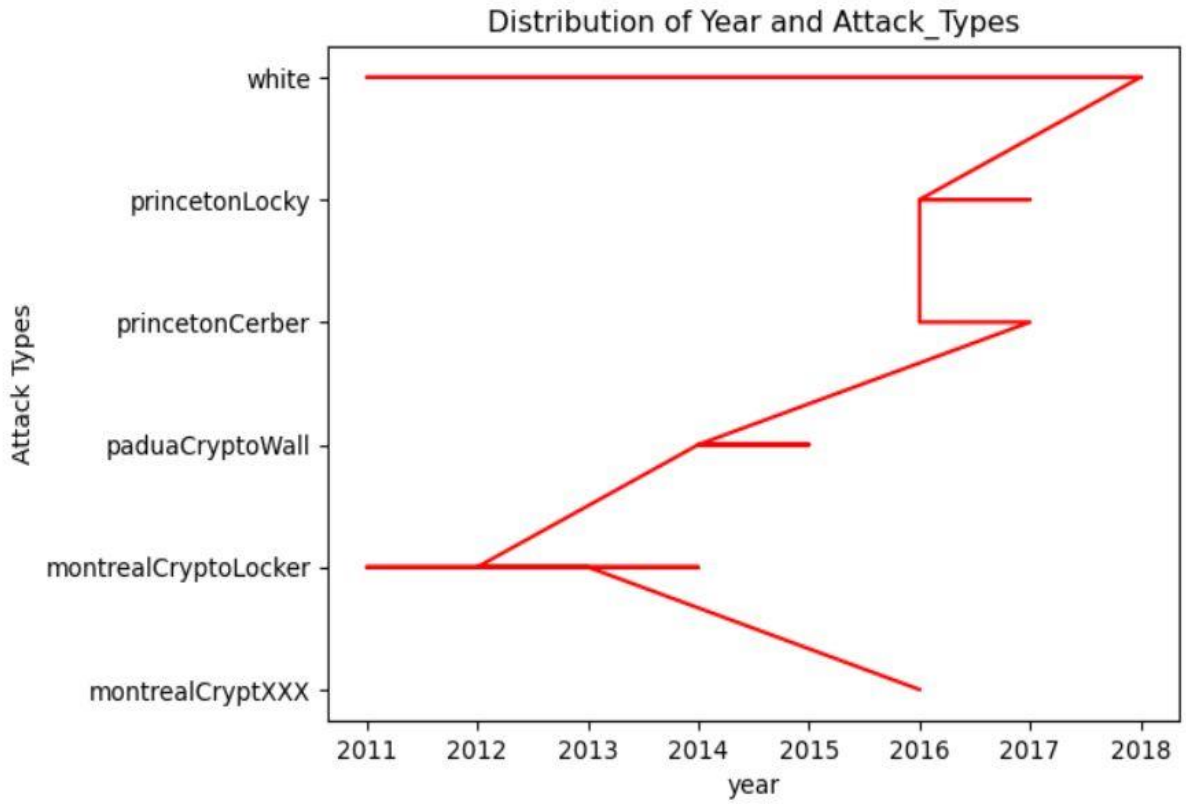


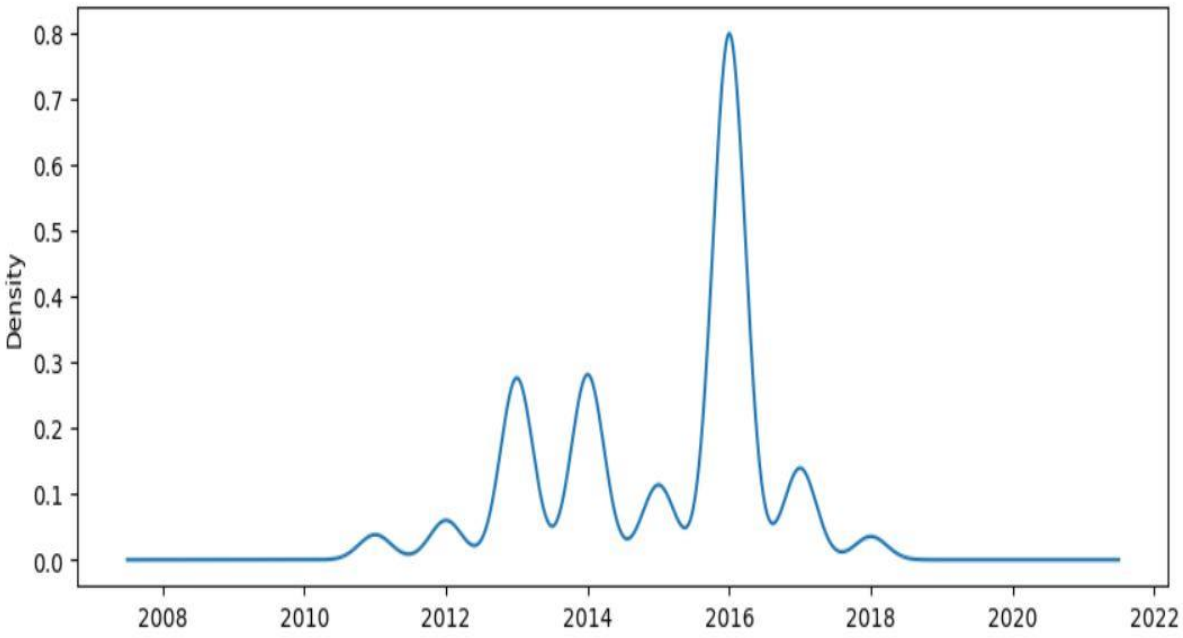

montrealCryptoLocker
paduaCryptoWall
montrealCryptXXX
16.67%
16.67%
16.67%
label
16.67%
16.67%
princetonCerber
16.67%
white
princetonLocky


2016
45.84%
year
2.03%
2018
16.14%
2.19%
2011
2014
3.44%
2012
6.52%
15.84%
7.99%
2015
2013
2017

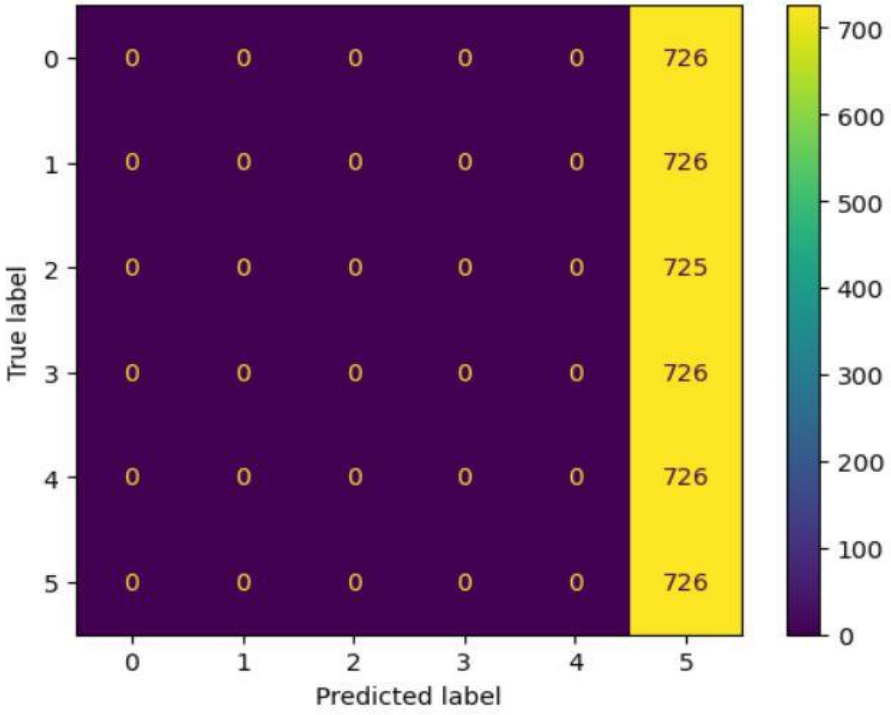

Unnamed: 0
year
day
length
weight
count
looped
neighbors
income
1 0.097 -0.11 0.028 -0.032 0.066 0.061 -0.036 0.034
0.097 1 -0.25 0.086 -0.1 0.19 0.016 -0.066 -0.088
-0.11 -0.25 1 -0.023 0.023 -0.029 -0.0056 0.029 0.013
0.028 0.086 -0.023 1 0.025 0.74 0.31 0.065 0.0096
-0.032 -0.1 0.023 0.025 1 0.028 0.14 0.79 0.084
0.066 0.19 -0.029 0.74 0.028 1 0.44 0.032 -0.0055
0.061 0.016 -0.0056 0.31 0.14 0.44 1 0.11 0.019
-0.036 -0.066 0.029 0.065 0.79 0.032 0.11 1 0.099
0.034 -0.088 0.013 0.0096 0.084 -0.0055 0.019 0.099 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

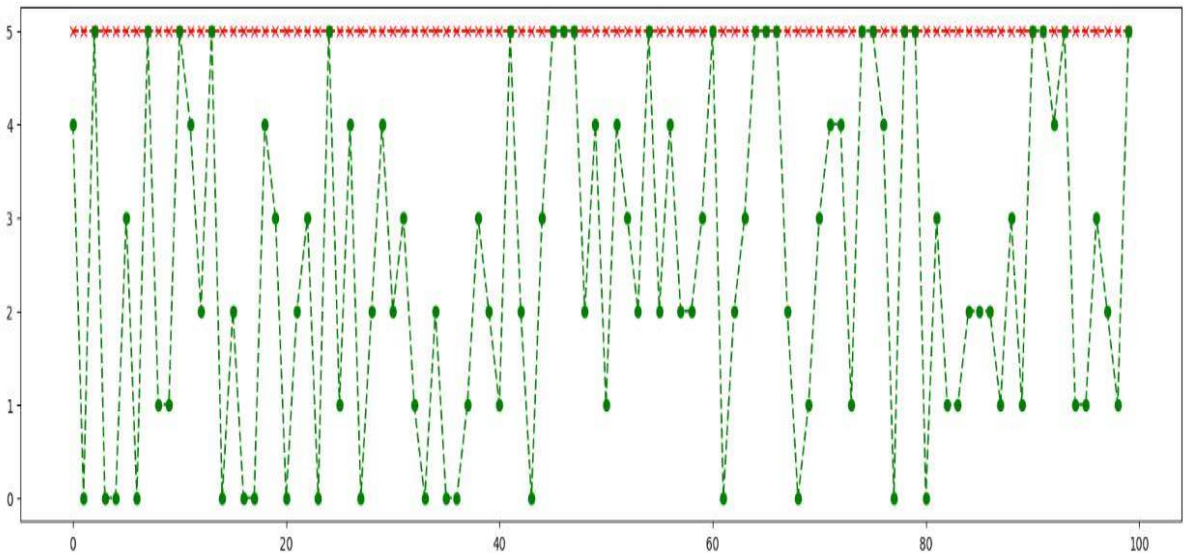

Scatter Plot
white
princetonLocky
princetonCerber
paduaCryptoWall
montrealCryptoLocker
montrealCryptXXX
2011
2012
2013
2014
2015
2016
2017
2018


Distribution of Year and Attack_Types
white
princetonLocky
princetonCerber
paduaCryptoWall
montrealCryptoLocker
montrealCryptXXX
Attack Types
2011
2012
2013
2014
2015
2016
2017
2018
year

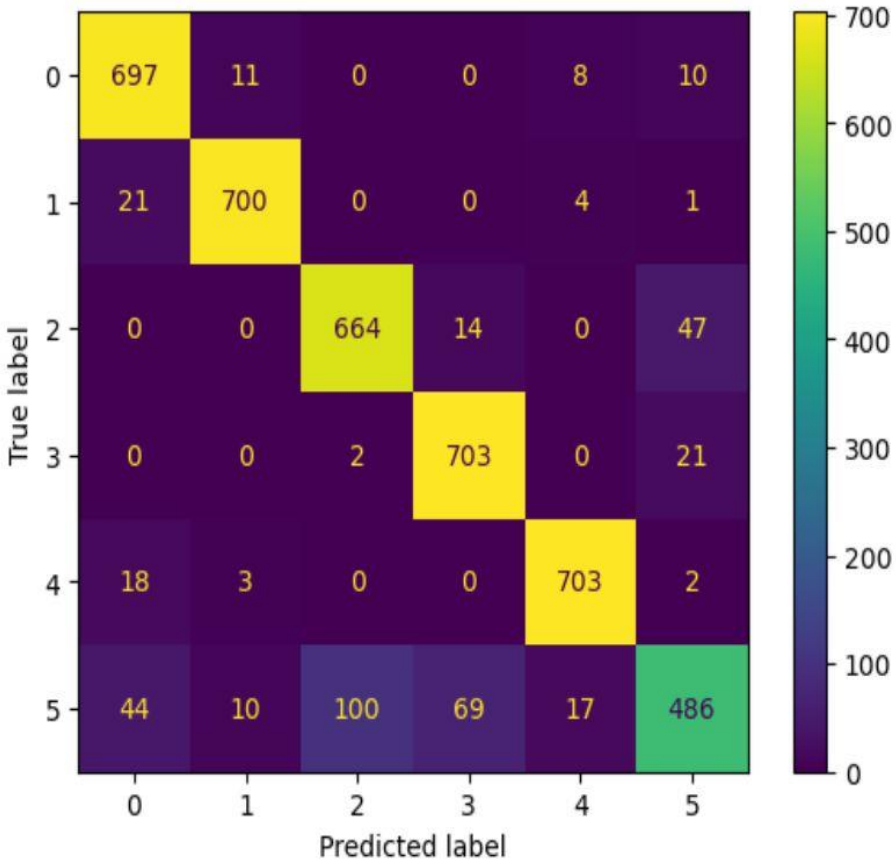

Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
2008
2010
2012
2014
2016
2018
2020
2022

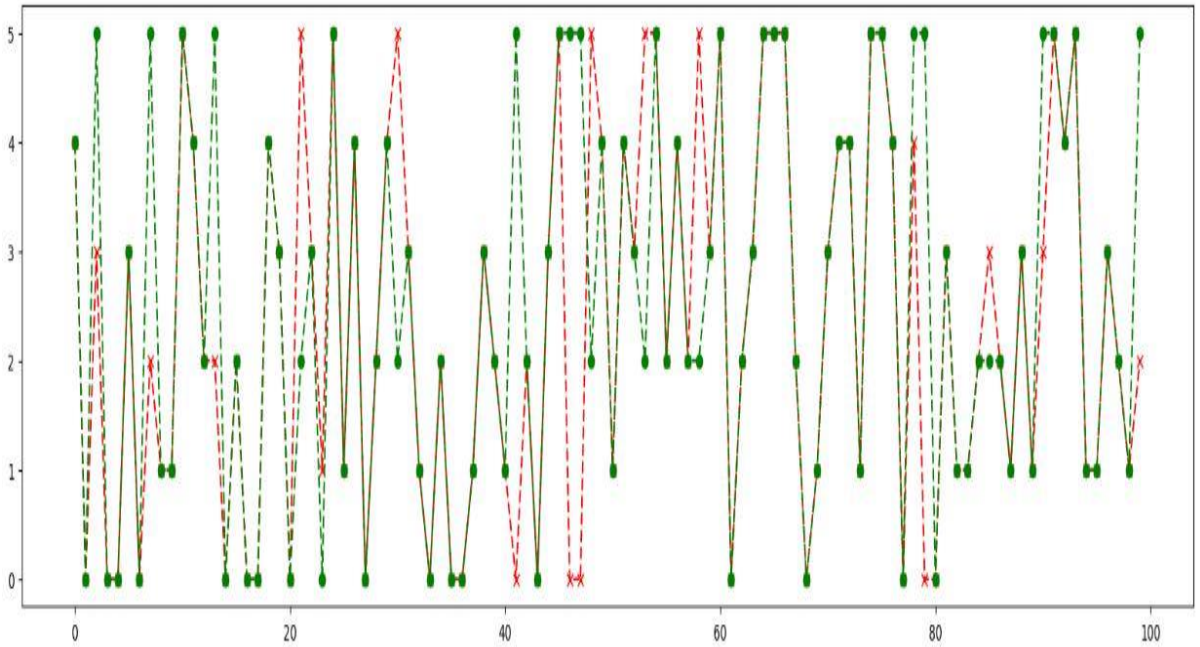
LOGISTIC REGRESSION



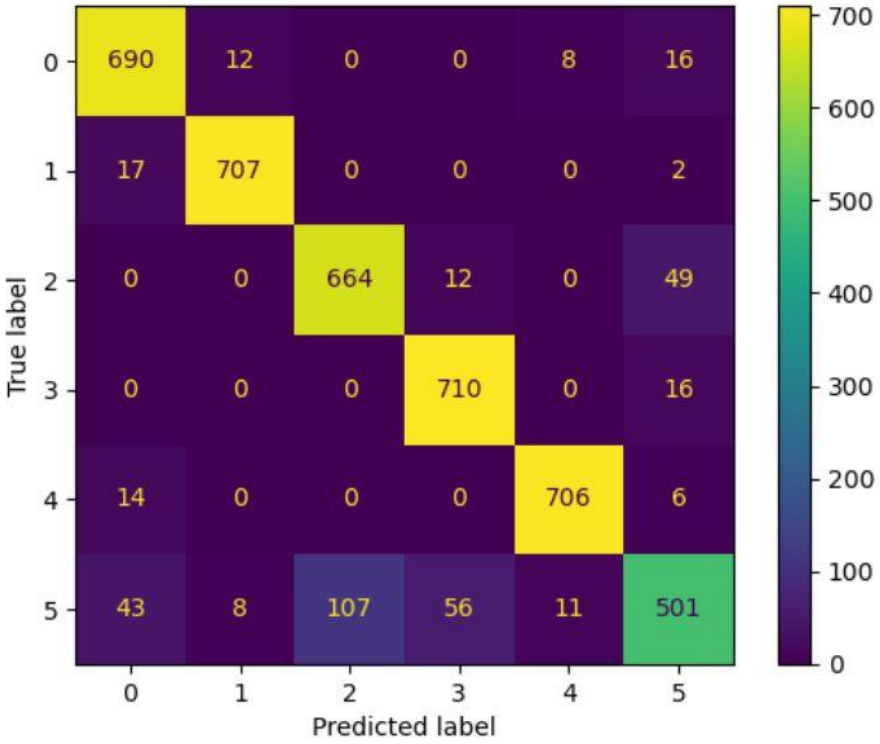


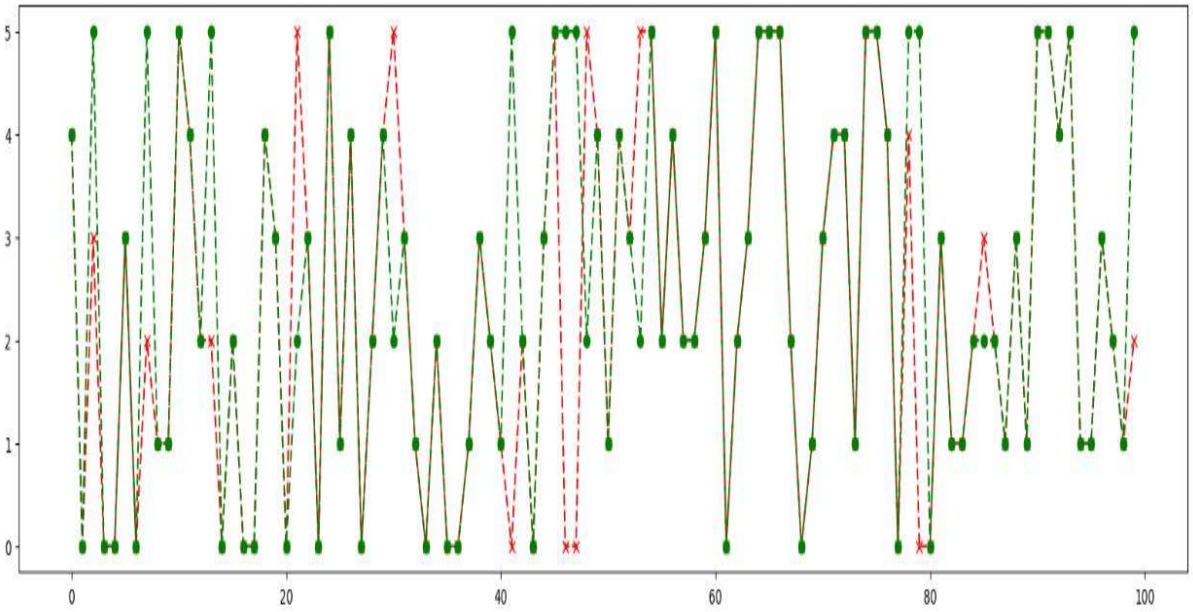
RANDOM FOREST CLASSIFIER



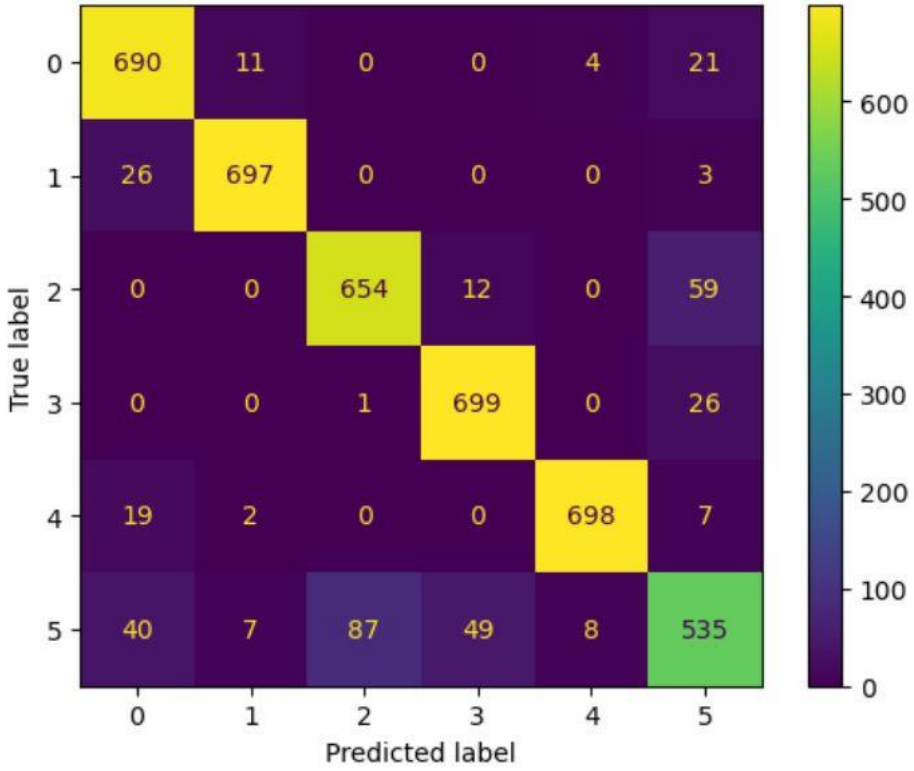


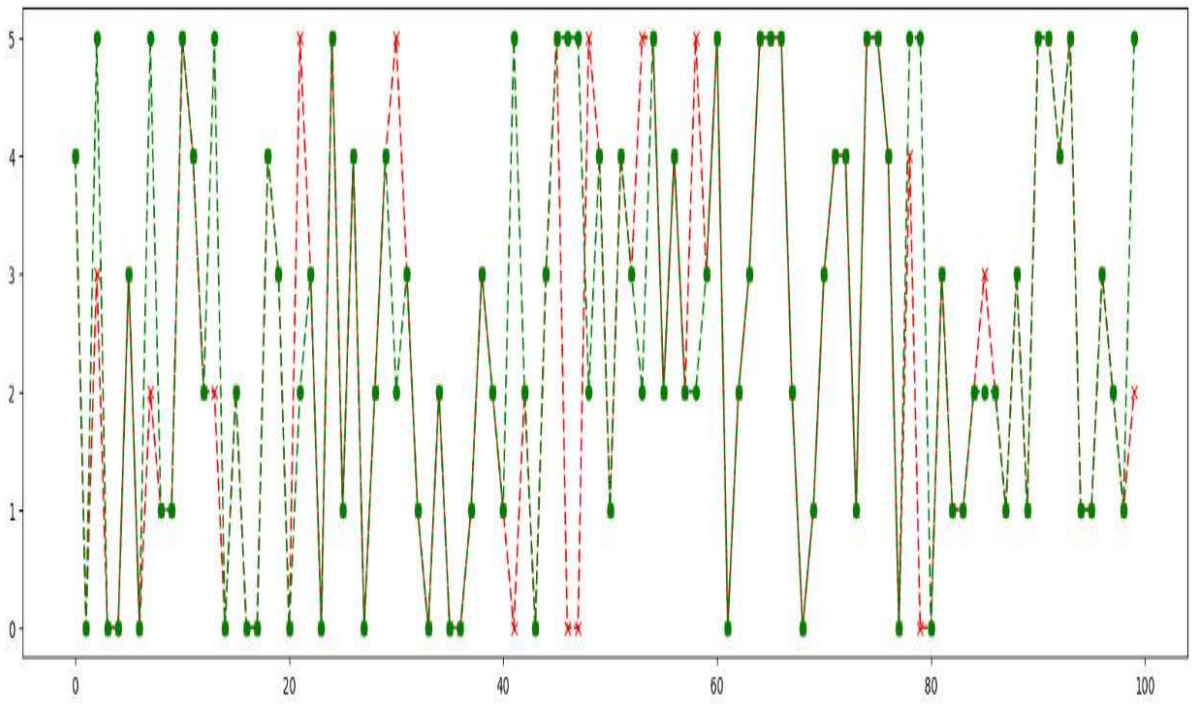
XG BOOST CLASSIFIER





VOTING CLASSIFIER





OUTPUT


BITCOIN ATTACK PREDICTION USING MACHINE LEARNING
YEAR
2016
COUNT
20
DAY
156
LOOPED
4
LENGTH
0.75
NEIGHBORS
0
WEIGHT
1
INCOME
1.4
Predict
THE BITCOIN ATTACK IS PREDICTED AS : Crypto

REFERENCES

- Muniye, Temesgen & Rout, Minakhi & Mohanty, Lipika & Satapathy, Suresh. (2020). Bitcoin Price Prediction and Analysis Using Deep Learning Models. 10.1007/978-981-15-5397-4_63.
- Al Harrack, Micheline, The BitcoinHeist: Classifications of Ransomware Crime Families (October 2021). International Journal of Computer Science & Information Technology (IJCSIT) Vol 13, No 5, October 2021.
- Çağlar, Ersin & Kirikkaleli, Dervis. (2020). The Crypto-currency and Cyber-attack: Evidence from Causality Techniques. International Journal of Engineering Trends and Technology. 68. 1-4. 10.14445/22315381/IJETT-V68I9P201.
- Mahajan, Rahul & Roychaudhary, Reema. (2021). Protective Mechanism form DDoS Attack for Cryptocoin. 12. 1421.
- Shah, Naman & Dave, Sonal. (2022). Review on Types of Attacks on Bitcoin and Ethereum crypto currencies. International Journal of Engineering Research in Computer Science and Engineering. 9. 1-6. 10.36647/IJERCSE/09.10.Art001.
- Talabani, Hardi & Abdulhadi, Hezha. (2022). Bitcoin Ransomware Detection Employing Rule-Based Algorithms. Science Journal of University of Zakho.
- Shengyun Xu. 2021. The Application of Machine Learning in Bitcoin Ransomware Family Prediction. In 2021 the 5th International Conference on Information System and Data Mining (ICISDM 2021). Association for Computing Machinery, New York, NY, USA, 21–27.
- Hairil, N. D. W. Cahyani and H. H. Nuha, "Ransomware Detection on Bitcoin Transactions Using Artificial Neural Network Methods," 2021 9th International Conference on Information and Communication Technology (ICoICT), Yogyakarta, Indonesia, 2021, pp.
- Huang, Danny & Aliapoulios, Maxwell & Li, Vector & Invernizzi, Luca & Bursztein, Elie & McRoberts, Kylie & Levin, Jonathan & Levchenko, Kirill & Snoeren, Alex & Mccoy, Damon. (2018). Tracking Ransomware End-to-end.
- Bistarelli, Stefano, Matteo Parroccini and Francesco Santini. “Visualizing Bitcoin Flows of Ransomware: WannaCry One Week Later.” Italian Conference on Cybersecurity (2018).

CERTIFICATE
of Publication

IJARASEM

International Journal of Advanced Research in Arts, Science, Engineering & Management

(A High Impact Factor, Bimonthly, Peer Reviewed & Referred Journal)

Web : www.ijarasem.com Email: editor@ijarasem.com, ijarasem@gmail.com

This is hereby Awarding this Certificate to

A.ANBUMANI

Assistant Professor, Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Chennai, India

Published a paper entitled

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

in IJARASEM, Volume 10, Issue 3, May 2023

Impact Factor 6.551

ISSN: 2395-7852

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Editor-in-Chief
IJARASEM

CERTIFICATE
of Publication

IJARASEM

International Journal of Advanced Research in Arts, Science, Engineering & Management

(A High Impact Factor, Bimonthly, Peer Reviewed & Referred Journal)

Web : www.ijarasem.com Email: editor@ijarasem.com, ijarasem@gmail.com

This is hereby Awarding this Certificate to

GEETH AKSHAY KUMAR M

UG Scholar, Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Chennai, India

Published a paper entitled

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

in IJARASEM, Volume 10, Issue 3, May 2023

Impact Factor 6.551

ISSN: 2395-7852

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Editor-in-Chief
IJARASEM

CERTIFICATE
of Publication

IJARASEM

International Journal of Advanced Research in Arts, Science, Engineering & Management

(A High Impact Factor, Bimonthly, Peer Reviewed & Referred Journal)

Web : www.ijarasem.com Email: editor@ijarasem.com, ijarasem@gmail.com

This is hereby Awarding this Certificate to

SHAIK ABDUL ALEEM

UG Scholar, Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Chennai, India

Published a paper entitled

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

in IJARASEM, Volume 10, Issue 3, May 2023

Impact Factor 6.551

ISSN: 2395-7852

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Editor-in-Chief
IJARASEM

CERTIFICATE
of Publication

IJARASEM

International Journal of Advanced Research in Arts, Science, Engineering & Management

(A High Impact Factor, Bimonthly, Peer Reviewed & Referred Journal)

Web : www.ijarasem.com Email: editor@ijarasem.com, ijarasem@gmail.com

This is hereby Awarding this Certificate to

SANJAY M

UG Scholar, Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Chennai, India

Published a paper entitled

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

in IJARASEM, Volume 10, Issue 3, May 2023

Impact Factor 6.551

ISSN: 2395-7852

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Editor-in-Chief
IJARASEM

CERTIFICATE
of Publication

IJARASEM

International Journal of Advanced Research in Arts, Science, Engineering & Management

(A High Impact Factor, Bimonthly, Peer Reviewed & Referred Journal)

Web : www.ijarasem.com Email: editor@ijarasem.com, ijarasem@gmail.com

This is hereby Awarding this Certificate to

THIYAGARAJAN P G

UG Scholar, Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Chennai, India

Published a paper entitled

BITCOIN ATTACK PREDICTION USING MACHINE LEARNING

in IJARASEM, Volume 10, Issue 3, May 2023

Impact Factor 6.551

ISSN: 2395-7852

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Editor-in-Chief
IJARASEM