A

Major Project

On

ARTIFICIAL INTELLIGENCE FOR CYBERSECURITY: A SYSTEMATIC MAPPING OF LITERATURE

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

S.Neha(187R1A0549)

A.Manasa(187R1A0505)

N.Anusha(187R1A0540)

Under the Guidance of

A.UDAY KIRAN

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC,NBA,Permanently Afiliated to JNTUH,Approved by AICTE,New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2018-22

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"ARTIFICIAL INTELLIGENCE FOR CYBERSECURITY:A SYSTEMATIC MAPPING OF LITERATURE"** being submitted by **S.NEHA(187R1A0549), A.MANASA(187R1A0505) & N.ANUSHA(187R1A0540)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Mr. A.Uday Kiran
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
Head Of Department

EXTERNAL EXAMINER

Submitted for viva voice Examination held on_____.

ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.We take this opportunity to express my profound gratitude and deep regard to my guide**.**

We take this opportunity to express my profound gratitude and deep regard to my guide **A.Uday Kiran**, Assistant Professor for his exemplary guidance, monitoring and constant encouragement through out the project work.The blessing help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to **Project Review Committee (PRC) Mr. A.Uday Kiran, Mr. J.Narasimharao, Dr. T.S.Mastan Rao, Mrs. G.Latha, Mr. A.Kiran Kumar** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A.Raji Reddy**, Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. Ch. Gopal Reddy, Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

S.NEHA(187R1A0549)

A.MANASA(187R1A0505)

N.ANUSHA(187R1A0540)

ABSTRACT

Due to the ever-increasing complexities in cybercrimes, there is the need for cybersecurity methods to be more robust and intelligent. This will make defense mechanisms to be capable of making real-time decisions that can effectively respond to sophisticated attacks. To support this, both researchers and practitioners need to be familiar with current methods of ensuring cybersecurity (CyberSec). In particular,the use of artificial intelligence for combating cybercrimes. However, there is lack of summaries on artificial intelligent methods for combating cybercrimes. To address this knowledge gap, this study sampled 131 articles from two main scholarly databases (ACM digital library and IEEE Xplore). Using a Systematic mapping, the articles were analyzed using quantitative and qualitative methods. It was observed that artificial intelligent methods have made remarkable contributions to combating cybercrimes with significant improvement in intrusion detection systems. It was also observed that there is a reduction in computational complexity, model training times and false alarms. However, there is a significant skewness within the domain. Most studies have focused on intrusion detection and prevention systems, and the most dominant technique used was support vector machines. The findings also revealed that majority of the studies were published in two journal outlets. It is therefore suggested that to enhance research in artificial intelligence for CyberSec, researchers need to adopt newer techniques and also publish in other related outlets.

LIST OF FIGURES/TABLES

FIGURE NO.	FIGURE NAME	PAGE NO.
FIGURE 2.1	JOB ROLES OF PYTHON DEVELOPERS	8
FIGURE 2.2	PROGRAMMING LANGUAGES COMPANY SIZE BREAKDOWN	11
FIGURE 5.1	USE CASE DIAGRAM OF PROPOSED STRUCTURE	16
FIGURE 5.2	COMPONENT DIAGRAM OF PROPOSED STRUCTURE	17
FIGURE 5.3	DEPLOYMENT MODEL OF PROPOSED STRUCTURE	18
FIGURE 5.4	LEVEL 0 DFD	20
FIGURE 5.5	LEVEL 1 DFD	20
FIGURE 5.6	LEVEL 2 DFD	21
FIGURE 5.7	BLOCK DIAGRAM OF PROPOSED STRUCTURE	21

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
SCREENSHOT 7.1	Loading the libraries required for the model	32
SCREENSHOT 7.2	Loading the data and seeing the few lines of the dataset	32
SCREENSHOT 7.3	Featuring the headers to the dataset	33
SCREENSHOT 7.4	Label encoder for converting data to numeric format	33
SCREENSHOT 7.5	Chi 2 model for identifying the parameters	34
SCREENSHOT 7.6	Dividing data to training and testing	34
SCREENSHOT 7.7	SVC model for analyzing the data	35
SCREENSHOT 7.8	Random forest model for analyzing the data	35
SCREENSHOT 7.9	Adaboost classifier model for analyzing the data	36
SCREENSHOT 7.10	XGB classifier for model	36

TABLE OF CONTENTS

ABSTRACT i

LIST OF FIGURES/TABLES ii

LIST OF SCREENSHOTS iii

1.INTRODUCTION 1

2.LITERATURE SURVEY 4

2.1 BASE PAPER 4

2.1.1 Intrusion Detection Systems 4

2.1.2 Signature Based Intrusion Detection System(SIDS) 4

2.1.3 Anomaly-Based Intrusion Detection System(AIDS) 5

2.1.4 Knowledge Based Techniques 5

2.1.5 Finite State Machine(FSM) 5

2.2 PYTHON 6

2.2.1 Uses of Python 6

2.2.2 Uses of Python in Data Science 7

2.2.3 Python Scientific libraries and tools 8

2.2.4 Python Keywords 11

3. SYSTEM ANALYSIS 12

3.1 Problem Statement 12

3.2 Existing System 12

3.3 Hardware Requirements 12

3.4 Software Requirements 12

4.PROPOSED SYSTEM 13

4.1 Methodology 13

4.2 Feasibility Study 13

4.2.1 Economical Feasibility 14

4.2.2 Technical Feasibility 14

4.2.3 Operational Feasibility 15

5.EXPERIMENTAL ANALYSIS 16

5.1 UML Diagrams 16

5.1.1 Use Case Diagram 16

5.1.2 Component Diagram 17

5.1.3 Deployment Model 18

5.2 Data Flow Diagrams 19

5.2.1 Level 0: System input/output level 20

5.2.2 Level 1: Sub System level data flow 20

5.2.3 Level 2: File level detail data flow 21

5.3 Architecture 21

5.4 Modules 22

5.4.1 Feature Extraction 22

6.TESTING 23

6.1 Introduction To Testing

23 6.2 Testing Strategies

23

6.2.1 Unit Testing 24

6.2.2 Black Box Testing 24

6.2.3 White Box Testing 24

6.3 Test Approach **25**

6.3.1 Bottom Up Approach 25

6.3.2 Top Down Approach 25

6.4 Validation **26**

7.DISCUSSION OF RESULTS **27**

7.1 Sample Code **27**

7.2 Results **32**

8.CONCLUSION AND FUTURE ENHANCEMENT **37**

8.1 Conclusion **37**

9.REFERENCES **38**

1.Introduction

Intrusion Detection System is a security software that analyzes network traffic for suspicious action and issues alert signals when such action is found. It examines the network by collecting an adequate amount of data and detecting sensor nodes' abnormal behavior. Intrusion Detection System(IDS) also checks illegal access to the system and inappropriate use of the system. Such detection methods are instrumental in identifying unauthorized access, hackers and traders, masquerading software's, etc. [1]. IDS congregate data from the traffic within a computer system or from a network and is known as audit data. This audit data is analyzed to detect any violation in the system security policy, and in case any security breach is identified, a security break is concluded. This violation in security is possible from two ends, one from inside the network or from the outside the network. There are two methods for intrusion detection misuse detection and the anomaly detection [2]. In the misuse detection method, IDS examines the data it collects and relates it with an extensive database of known attack patterns. Attack patterns are kept in the database, and each packet is matched with patterns in the database; if it is a malicious packet, an alert is generated. Anomaly detection method aims to reveal abnormal behavior of the system.

With the recent interest and progress in the development of internet and communication technologies over the last decade, network security has emerged as a vital research domain. It employs tools like firewall, antivirus software, and intrusion detection system (IDS) to ensure the security of the network and all its associated assets within a cyberspace.1 Among these, network‐based intrusion detection system (NIDS) is the attack detection mechanism that provides the desired security by constantly monitoring the network traffic for malicious and suspicious behavior .

The evolution of malicious software (malware) poses a critical challenge to the design of intrusion detection systems (IDS). Malicious attacks have become more sophisticated and the foremost challenge is to identify unknown and obfuscated malware, as the malware authors use different evasion techniques for information concealing to prevent detection by an IDS. In addition, there has been an increase in security threats such as zero-day attacks designed to target internet users. Therefore, computer security has become essential as the use of information technology has become part of our daily lives.

The idea of IDS was first proposed by Jim Anderson in 1980.4 Since then, many IDS products were developed and matured to satisfy the needs of network security.5 However, the immense evolution in the technologies over the last decade has resulted in a large expansion in the network size, and the number of applications handled by the network nodes. As a result, a huge amount of important data is being generated and shared across different network nodes. The security of these data and network nodes has become a challenging task due to the generation of a large number of new attacks either through the mutation of an old attack or a novel attack. Almost every node within a network is vulnerable to security threats. For instance, the data node may be very important for an organization. Any compromise to the node's information may cause a huge impact on that organization in terms of its market reputation and financial losses. Existing IDSs have shown inefficiency in detecting various attacks including zero‐day attacks and reducing the false alarm rates (FAR).6 This eventually results in a demand for an efficient, accurate, and cost‐effective NIDS to provide strong security to the network.

To fulfill the requirements of an effective IDS, the researchers have explored the possibility of using machine learning (ML) and deep learning (DL) techniques. Both ML and DL come under the big umbrella of artificial intelligence (AI) and aim at learning useful information from the big data.7 These techniques have gained enormous popularity in the field of network security, over the last decade due to the invention of very powerful graphics processor units (GPUs).8 Both ML and DL are powerful tools in learning useful features from the network traffic and predicting the normal and abnormal activities based on the learned patterns.

The ML‐based IDS depends heavily on feature engineering to learn useful information from the network traffic.9 While DL‐based IDS do not rely on feature engineering and are good at automatically learning complex features from the raw data due to its deep structure.

2. Literature Survey

2.1 Base Paper

2.1.1 Intrusion detection systems

Intrusion can be defined as any kind of unauthorised activities that cause damage to an information system. This means any attack that could pose a possible threat to the information confidentiality, integrity or availability will be considered an intrusion. For example, activities that would make the computer services unresponsive to legitimate users are considered an intrusion. An IDS is a software or hardware system that identifies malicious actions on computer systems in order to allow for system security to be maintained (Liao et al., 2013a). The goal of an IDS is to identify different kinds of malicious network traffic and computer usage, which cannot be identified by a traditional firewall.

2.1.2 Signature-based intrusion detection systems (SIDS)

Signature intrusion detection systems (SIDS) are based on pattern matching techniques to find a known attack; these are also known as Knowledge-based Detection or Misuse Detection (Khraisat et al., 2018). In SIDS, matching methods are used to find a previous intrusion. In other words, when an intrusion signature matches with the signature of a previous intrusion that already exists in the signature database, an alarm signal is triggered. For SIDS, host's logs are inspected to find sequences of commands or actions which have previously been identified as malware. SIDS have also been labelled in the literature as Knowledge-Based Detection or Misuse Detection (Modi et al., 2013).

2.1.3 Anomaly-based intrusion detection system (AIDS)

AIDS has drawn interest from a lot of scholars due to its capacity to overcome the limitation of SIDS. In AIDS, a normal model of the behavior of a computer system is created using machine learning, statistical-based or knowledge-based methods. Any significant deviation between the observed behavior and the model is regarded as an anomaly, which can be interpreted as an intrusion. The assumption for this group of techniques is that malicious behavior differs from typical user behavior.

2.1.4 Knowledge-based techniques

This group of techniques is also referred toas an expert system method. This approach requires creating a knowledge base which reflects the legitimate traffic profile. Actions which differ from this standard profile are treated as an intrusion. Unlike the other classes of AIDS, the standard profile model is normally created based on human knowledge, in terms of a set of rules that try to define normal system activity.

2.1.5 Finite state machine (FSM)

FSM is a computation model used to represent and control execution flow. This model could be applied in intrusion detection to produce an intrusion detection system model. Typically, the model is represented in the form of states, transitions, and activities. A state checks the history data. For instance, any variations in the input are noted and based on the detected variation transition happens (Walkinshaw et al., 2016). An FSM can represent legitimate system behaviour, and any observed deviation from this FSM is regarded as an attack.

2.2 Python

Python is a programming language, which means it'a a language both people and computers can understand. Python was developed by a Dutch software engineer named Guido van Rossum, who created the language to solve some problems he saw in computer languages of the time.

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released program.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. C Python is managed by the non-profit Python Software Foundation.

2.2.1 Uses Of Python

One significant advantage of learning Python is that it's a general-purpose language that can be applied in a large variety of projects. Below are just some of the most common fields where Python has found its use:

- Data science
- Scientific and mathematical computing
- Web development
- Computer graphics
- Basic game development

2.2.2 Uses of Python in Data Science

There are plenty of Python scientific packages for data visualization, machine learning, natural language processing, complex data analysis and more. All of these factors make Python a great tool for scientific computing and a solid alternative for commercial packages such as MatLab. The most popular libraries and tools for data science are:

- **Pandas**: a library for data manipulation and analysis. The library provides data structures and operations for manipulating numerical tables and time series.
- **NumPy**: the fundamental package for scientific computing with Python, adding support for large, multi-dimensional arrays and matrices, along with a large library of high-level mathematical functions to operate on these arrays.
- **SciPy**: a library used by scientists, analysts, and engineers doing scientific computing and technical computing.

As a result of this popularity there are plenty of Python scientific packages for data visualization, machine learning, natural language processing, complex data analysis and more. All of these factors make Python a great tool for scientific computing and a solid alternative for commercial packages such as MatLab.

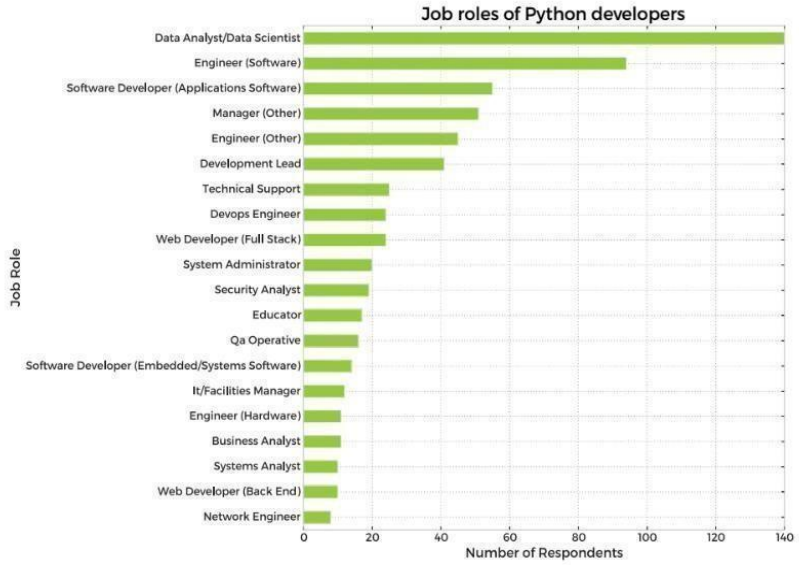


FIG 2.1 JOB ROLES OF PYTHON DEVELOPERS

2.2.3 Python scientific libraries and tools

Astropy

The Astropy Project is a collection of packages designed for use in astronomy. The core astropy package contains functionality aimed at professional astronomers and astrophysicists, but may be useful to anyone developing astronomy software.

Biopython

Biopython is a collection of non-commercial Python tools for computational biology and bioinformatics. It contains classes to represent biological sequences and sequence annotations, and it is able to read and write to a variety of file formats.

Pandas

Pandas is a library for data manipulation and analysis. The library provides data structures and operations for manipulating numerical tables and time series.

Matplotlib

Matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib allows you to generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, and more.

NumPy

NumPy is the fundamental package for scientific computing with Python, adding support for large, multi-dimensional arrays and matrices, along with a large library of highlevel mathematical functions to operate on these arrays.

NetworkX

NetworkX is a library for studying graphs which helps you create, manipulate, and study the structure, dynamics, and functions of complex networks.

SciPy

SciPy is a library used by scientists, analysts, and engineers doing scientific computing and technical computing. SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

Scikit-learn

Scikit-learn is a machine learning library. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Scikit-image

Scikit-image is a image processing library. It includes algorithms for segmentation, geometric transformations, color space manipulation, analysis, filtering, morphology, feature detection, and more.

Graph-tool

Graph-tool is a module for the manipulation and statistical analysis of graphs.

Python Saves Time

Even the classic “Hello, world” program illustrates this point:

```
print("Hello, world")
```

For comparison, this is what the same program looks like in Java: public class

```
HelloWorld
{
public static void main(String[] args)
        {
 System.out.println("Hello, world");
      }
}
```

CMRTC

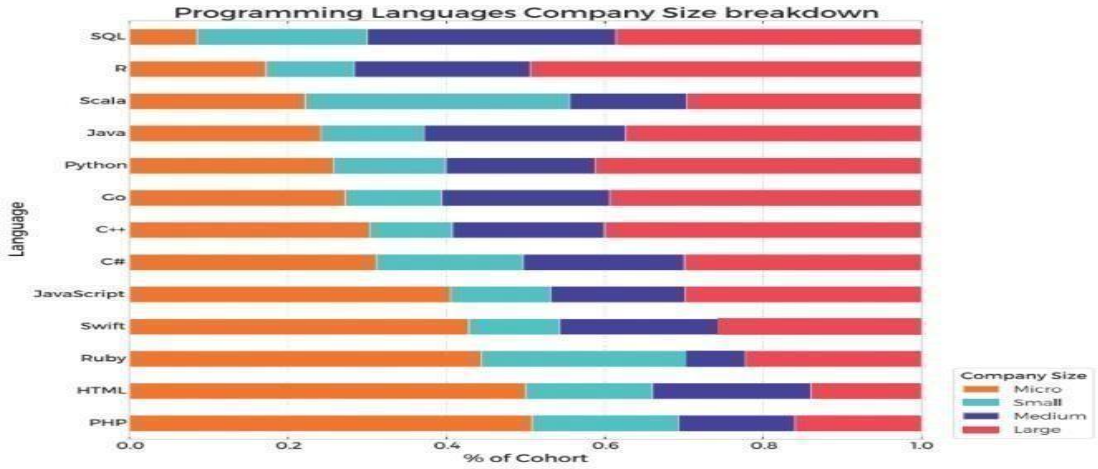


FIG 2.2:PROGRAMMING LANGUAGES COMPANY SIZE BREAKDOWN

2.2.4 Python Keywords

Keywords are the reserved words in Python.

We cannot use a keyword as variable name, function name or any other identifier. They are used to define the syntax and structure of the Python language.

In Python, keywords are case sensitive.

3.System Analysis

3.1 Problem Statement:

In the field of machine learning, the problem of category imbalance has always been a challenge. Therefore, intrusion detection also faces enormous challenges in network traffic with extremely imbalanced categories. Therefore, many scholars have begun to study how to improve the intrusion recognition accuracy of imbalanced network traffic data.

3.2 Existing System:

Some scholars applied machine learning methods in intrusion detection. However, due to the limitation of computer storage and computing power at that time, machine learning failed to attract attention. With the rapid development of computers and the emergence and promotion of Artificial Intelligence(AI) and other technologies, many scholars have applied machine learning methods to network security. They have achieved certain results

3.3 Hardware Requirements

- RAM : 4GB min
- Hard Disc : 500 GB min

Graphical Processing unit if required

3.4 Software Requirements

- OS : Windows 10
- Technology : Python
- Domain : Machine Learning / Deep Learning

4.Proposed System

Faced with imbalanced network traffic data, we propose a novel Difficult Set Sampling Technique(DSSTE) algorithm to tackle the class imbalance problem in network traffic. This method effectively reduces the imbalance and makes the classification model learning difficult samples more effective. We use classic machine learning and deep learning algorithms to verify on two benchmark datasets. The specific contributions are as follows. (1) We use the classic NSL-KDD and the up-to-date CSECIC-IDS2018 as benchmark datasets and conduct detailed analysis and data cleaning. (2)This work proposes a novel DSSTE algorithm,reducing the majority samples and augmenting the minority samples in the difficult set, tackling the class imbalance problem in intrusion detection so that the classifier learns the differences better in training. (3) The classification model uses Random Forest(RF), Support Vector Machine(SVM), .

4.1 Methodology:

Step1: Collect and store the Data.

Step2: Data pre processing

Step3: Extract the features from dataset

Step4: Make model to learn from the different input features.

Step5: Predict the results

4.2 Feasibility Study:

A credibility contemplate expects to fair-mindedly and soundly uncover the qualities and inadequacies of a present business or proposed meander, openings and threats present in nature, the benefits required to bring through, and in the long run

the prospects for advance. In its most clear terms, the two criteria to judge believability are incurred significant injury required and motivator to the fulfilled.

An inside and out arranged feasibility ponder should give a recorded establishment of the business or wander, a delineation of the thing or organization, accounting explanations, purposes of enthusiasm of the operations and organization, publicizing examination and game plans, budgetary data, authentic necessities and cost duties. All things considered, plausibility looks at go before specific change and wander utilization.

There are three sorts of attainability

- Economical Feasibility
- Technical Feasibility
- Operational Feasibility

4.2.1 Economical feasibility

The electronic structure manages the present existing system's data stream and technique absolutely and should make each one of the reports of the manual structure other than a substantial gathering of the other organization reports. It should be filled in as an electronic application with specific web server and database server. Advance a segment of the associated trades happen in different ranges. Open source programming like TOMCAT, JAVA, MySQL and Linux is used to restrict the cost for the Customer.No extraordinary wander need to manage the instrument.

4.2.2 Technical feasibility

Surveying the particular probability is the trickiest bit of a believability consider. This is in light of the fact that, starting at the present moment, not a lot of point by point layout of the system, making it difficult to get to issues like execution, costs on (by excellence of the kind of development to be passed on) et cetera.

Different issues must be considered while doing a particular examination. Grasp the differing progressions required in the proposed system. Before starting the wander, we should be clear about what are the advances that are to be required for the change of the new system. Check whether the affiliation by and by has the required advancements.

Is the required development open with the affiliation?

In case so is the utmost sufficient?

For instance – "Will the present printer have the ability to manage the new reports and structures required for the new system?"

4.2 3 Operational feasibility

Proposed wanders are profitable just if they can be changed into information systems that will meet the affiliations working necessities. Simply communicated, this trial of probability asks with reference to whether the structure will work when it is made and presented. Are there genuine obstacles to Implementation? Here are questions that will help test the operational achievability of a wander.

- Is there sufficient help for the wander from organization from customers? In case the present structure is particularly cherished and used to the extent that individuals won't have the ability to see purposes behind change, there may be resistance.
- Are the present business methodologies qualified to the customer? If they are not, Users may welcome a change that will accomplish a more operational and supportive systems.
- Have the customer been locked in with the orchestrating and change of the wander?

Early commitment decreases the chances of impenetrability to the structure.

5. Experimental Analysis

5.1 UML Diagrams

UML (Unified Modeling Language) is a standard vernacular for choosing, envisioning, making, and specifying the collectibles of programming structures. UML is a pictorial vernacular used to make programming blue prints. It is in like way used to exhibit non programming structures similarly like process stream in a gathering unit and so forth.

5.1.1 Use case Diagram:

The use case graph is for demonstrating the direct of the structure. This chart contains the course of action of use cases, performing pros and their relationship. This chart might be utilized to address the static perspective of the structure.

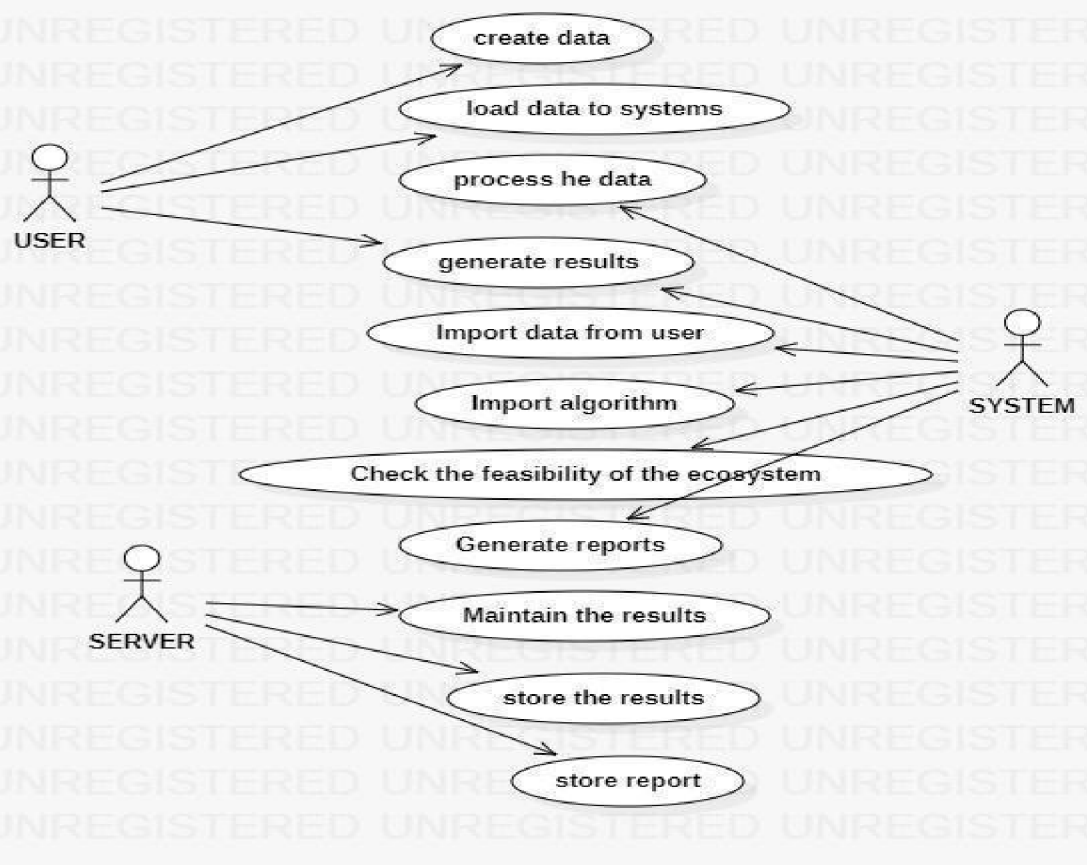


FIG 5.1: Use Case Diagram Of Proposed Structure

5.1.2 Component Diagram

The imperative portion of part format is segment. This diagram demonstrates within parts, connectors and ports that understand the piece. Precisely when section is instantiated, duplicates of inside parts are besides instantiated.

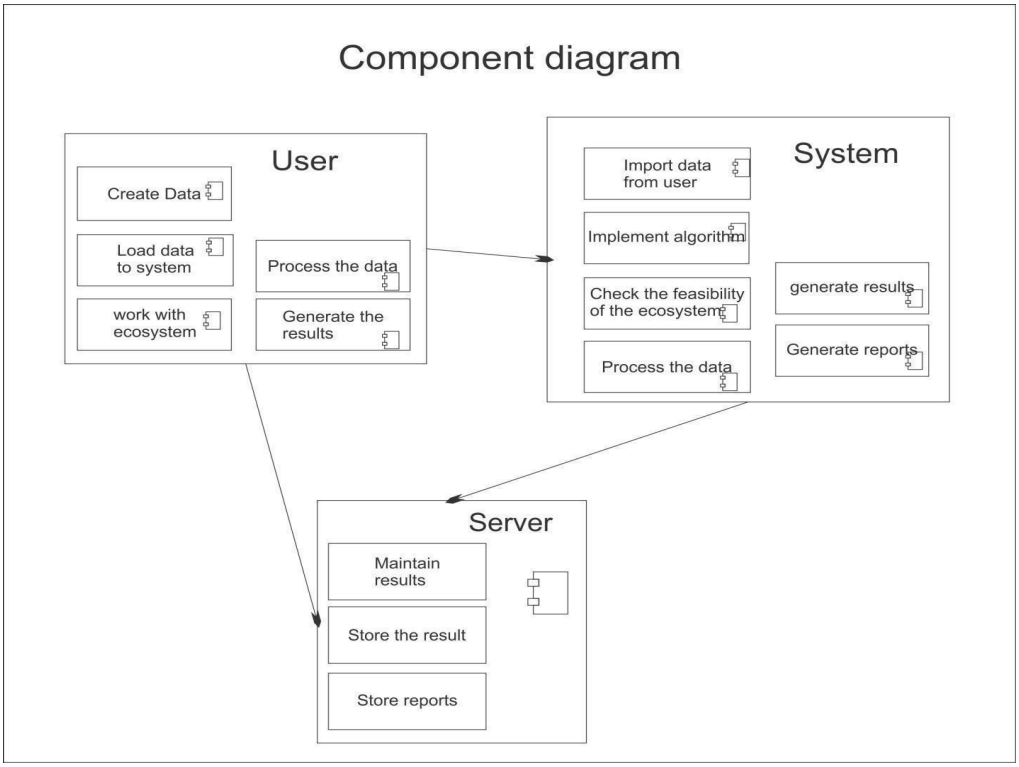


Fig 5.2: Component Diagram Of Proposed structure

A part outline is spoken to utilizing segment. A part is a physical building piece of the framework. It is spoken to as a rectangle with tab. Part outline portrays the inward handling of the venture. The information is sent to the Hadoop where sqoop is utilized for information cleaning and the reports are produced utilizing hive.

5.1.3 Deployment Model

The fundamental fragment in game-plan layout is a middle point. The strategy of focus focuses and their relationship with other is tended to utilizing sending plot. The sending outline is identified with the area diagram, that is one focus purpose obviously of activity format frequently includes no short of what one sections. This outline is in like way critical for tending to the static perspective of the framework.

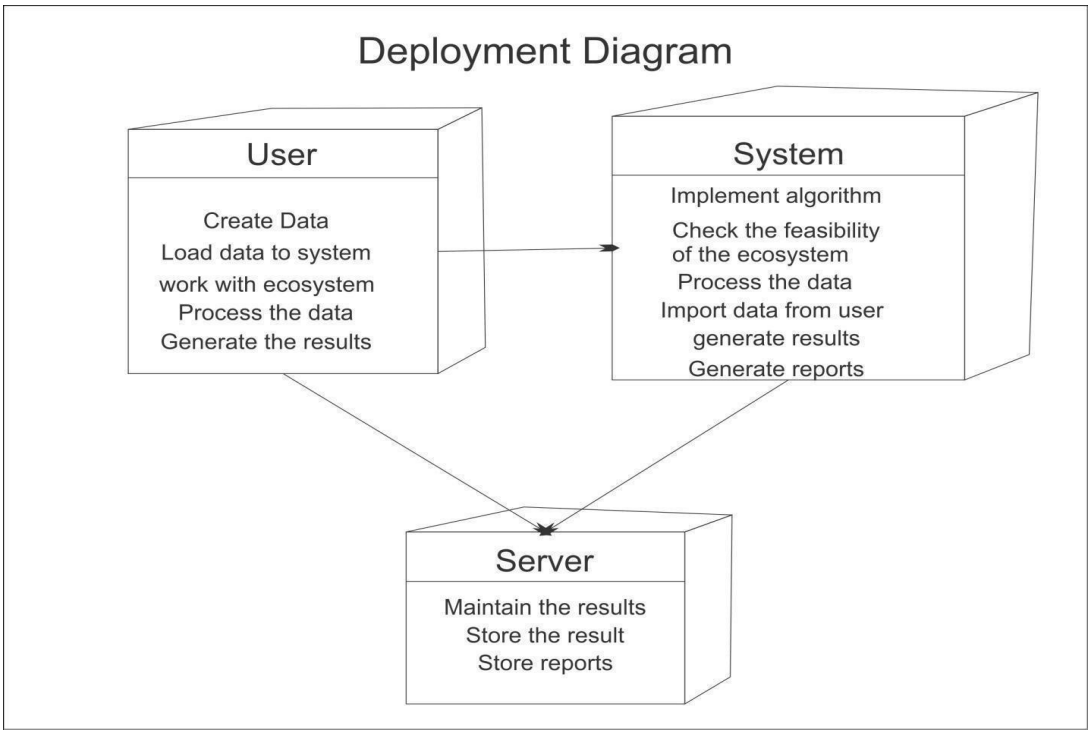


Fig 5.3: Deployment Model Of Proposed Structure

An arrangement graph is spoken to utilizing hub. A hub is a physical asset that executes code parts. They are likewise used to portray run time handling of hubs. The information is sent to the Hadoop where sqoop is utilized for information cleaning and the reports are produced utilizing hive.

5.2 Data Flow Diagrams:

An information stream design (DFD) is a graphical portrayal of the "stream" of information through a data framework, demonstrating its strategy edges. A DFD is a significant part of the time utilized as a preparatory stroll to make an overview of the framework, which can later be cleared up. DFDs can in like way be utilized for the depiction of information prepare. A DFD indicates what sort of data will be sense of duty regarding and yield from the structure, where the information will begin from and go to, and where the information will be secured. It doesn't demonstrate data about the organizing of process or data about whether strategy will work in game-plan or in parallel.

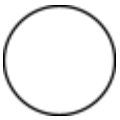
DFD Symbols:

In the DFD, there are four symbols

- A square defines a source or destination of system data.

- An arrow identifies data flow. It is the pipeline through which the information flows.

- A circle represents a process that transforms incoming data flow into outgoing data flow.



5.2.1 Level 0: System input/ output level

A level 0 DFD describes the system wide boundaries, dealing input to and output flow from the system and major processes.

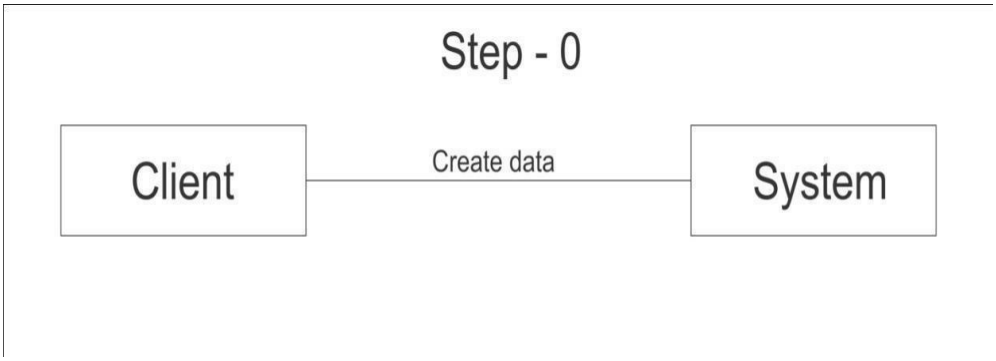


Fig 5.4: Level 0 DFD

DFD Level 0 is in like way called a Context Diagram. It's a urgent review of the entire structure or process being bankrupt down or appeared. It's required to be an at first watch, demonstrating the framework as a particular surprising state handle, with its relationship to outside substances.

5.2.2 Level 1: Sub system level data flow

Level 1 DFD delineates the accompanying level of purposes of enthusiasm with the data stream between subsystems. The Level 1 DFD exhibits how the system is secluded into sub-structures (shapes), each of which oversees no less than one of the data streams to or from an outside pro, and which together give most of the helpfulness of the system as a rule.

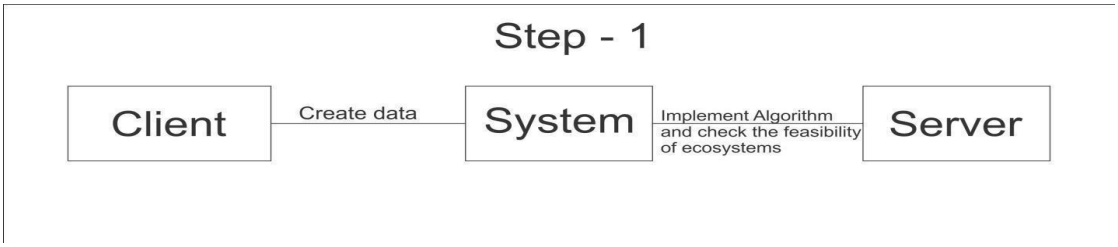


Fig 5.5: Level 1 DFD

5.2.3 Level 2: File level detail data flow

Plausibility and danger examination are connected here from various perspectives. The level 2 DFD elucidates the fundamental level of understanding about the system's working.

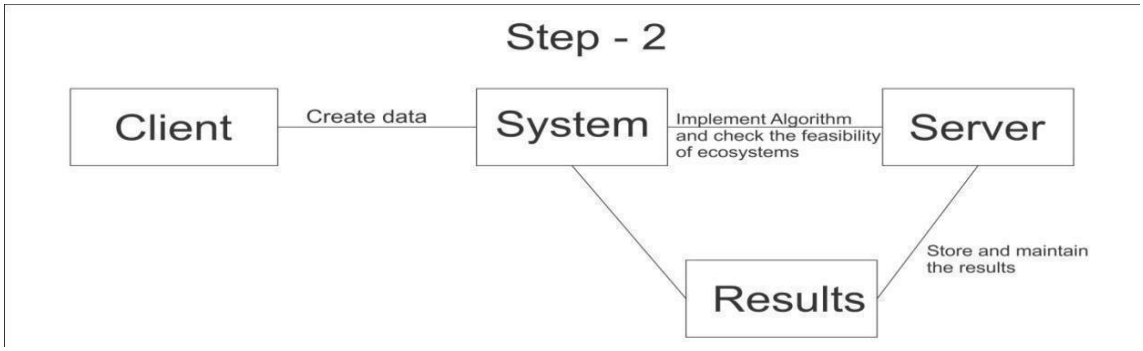
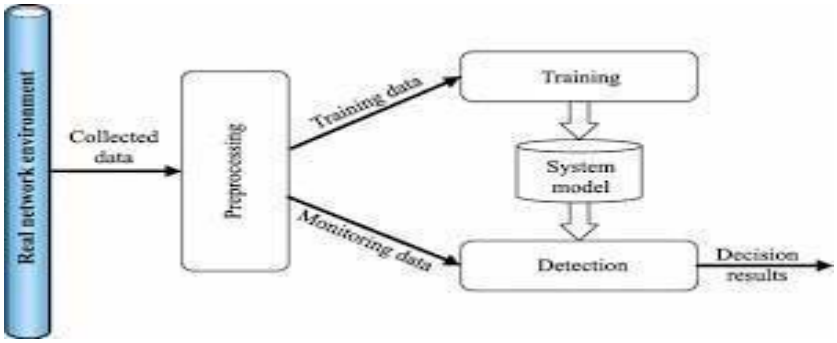


Fig 5.6: Level 2 DFD

5.3 Architecture:



5.7: BLOCK DIAGRAM OF PROPOSED SYSTEM

5.4 Modules

Data Source KDD CUP dataset is used for the experiment. It has 42 attributes classified under numeric, nominal, and binary. Two types of classes are there Normal and Anomaly. Anomaly class has some attacks categorized as dos, probe, u2r, r2l. This dataset has enough volume of records for training and testing purposes.

5.4.1 Feature Extraction:

Feature selection is made in a dataset to lessen the computational effort and find the optimal subset of the dataset that produces higher classification accuracy. A few parts of a dataset include irrelevant and redundant features, and such features distract the classifiers. PSO is used in this experiment for the extraction of features. 9 relevant features are selected among 42 features of the KDD-CUP dataset. .

6. Testing

6.1 Introduction To Testing

Testing is a procedure, which uncovers blunders in the program. Programming testing is a basic component of programming quality affirmation and speaks to a definitive audit of determination, outline and coding. The expanding perceivability of programming as a framework component and chaperon costs related with a product disappointment are propelling variables for we arranged, through testing. Testing is the way toward executing a program with the plan of finding a mistake. The plan of tests for programming and other built items can be as trying as the underlying outline of the item itself It is the significant quality measure utilized amid programming improvement. Amid testing, the program is executed with an arrangement of experiments and the yield of the program for the experiments is assessed to decide whether the program is executing as it is relied upon to perform.

6.2 Testing Strategies

A technique for programming testing coordinates the outline of programming experiments into an all around arranged arrangement of steps that outcome in fruitful improvement of the product. The procedure gives a guide that portrays the means to be taken, when, and how much exertion, time, and assets will be required. The procedure joins test arranging, experiment configuration, test execution, and test outcome gathering and assessment. The procedure gives direction to the specialist and an arrangement of points of reference for the chief. Due to time weights, advance must be quantifiable and issues must surface as ahead of schedule as would be prudent.

Keeping in mind the end goal to ensure that the framework does not have blunders, the distinctive levels of testing techniques that are connected at varying periods of programming improvement are:

6.2.1 Unit Testing

Unit Testing is done on singular modules as they are finished and turned out to be executable. It is restricted just to the planner's prerequisites. It centers testing around the capacity or programming module. It Concentrates on the interior preparing rationale and information structures. It is rearranged when a module is composed with high union

- Reduces the quantity of experiments
- Allows mistakes to be all the more effectively anticipated and revealed**.**

6.2.2 Black Box Testing

It is otherwise called Functional testing. A product testing strategy whereby the inward workings of the thing being tried are not known by the analyzer. For instance, in a discovery test on a product outline the analyzer just knows the information sources and what the normal results ought to be and not how the program touches base at those yields. The analyzer does not ever inspect the programming code and does not require any further learning of the program other than its determinations.

6.2.3 White Box testing

It is otherwise called Glass box, Structural, Clear box and Open box testing . A product testing procedure whereby express learning of the inner workings of the thing being tried are utilized to choose the test information. Not at all like discovery testing, white box testing utilizes particular learning of programming code to inspect yields. The test is precise just if the analyzer comprehends what the program should do.

6.3 Test Approach

A Test approach is the test system usage of a venture, characterizes how testing would be done. The decision of test methodologies or test technique is a standout amongst the most intense factor in the achievement of the test exertion and the precision of the test designs and gauges.

Testing should be possible in two ways

- Bottom up approach
- Top down approach

6.3.1 Bottom up Approach

Testing can be performed beginning from littlest and most reduced level modules and continuing each one in turn. In this approach testing is directed from sub module to primary module, if the fundamental module is not built up a transitory program called DRIVERS is utilized to recreate the principle module. At the point when base level modules are tried consideration swings to those on the following level that utilization the lower level ones they are tried exclusively and afterward connected with the already inspected bring down level modules

6.3.2 Top down Approach

In this approach testing is directed from fundamental module to sub module. in the event that the sub module is not built up an impermanent program called STUB is utilized for mimic the sub module. This sort of testing begins from upper level modules. Since the nitty gritty exercises more often than not performed in the lower level schedules are not given stubs are composed. A stub is a module shell called by upper level module and that when achieved legitimately will restore a message to the calling module demonstrating that appropriate association happened.

6.4 Validation

The way toward assessing programming amid the improvement procedure or toward the finish of the advancement procedure to decide if it fulfills determined business prerequisites. Approval Testing guarantees that the item really addresses the customer's issues. It can likewise be characterized as to exhibit that the item satisfies its proposed utilize when sent on proper condition.

The framework has been tried and actualized effectively and along these lines guaranteed that every one of the prerequisites as recorded in the product necessities determination are totally satisfied.

7. Discussion Of Results

7.1 Sample Code

" numpy as np\n",

"import pandas as pd\n",

"import matplotlib.pyplot as plt\n",

"from sklearn.feature_selection import SelectKBest, chi2\n",

"from sklearn.ensemble import ExtraTreesClassifier\n",

"from sklearn.model_selection import train_test_split\n",

"from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix\n",

"from sklearn.svm import SVC\n",

"from sklearn.neighbors import KNeighborsClassifier\n",

"from sklearn.metrics import classification_report, confusion_matrix, accuracy_score\n",

"from sklearn.ensemble import RandomForestClassifier\n",

"from sklearn.ensemble import AdaBoostClassifier\n",

"from xgboost import XGBClassifier"

"from google.colab import drive\n",

"drive.mount('/content/drive')"

"df = pd.read_csv(\"/content/drive/MyDrive/dataset.csv\")\n",

"df.head()"

"columns=(['real','protocol_type','service','flag','src_bytes','dst_bytes','land','wrong_fragment','urgent','real1','hot','num_failed_logins','logged_in','num_compromised','root_she

```
ll','su_attempted','num_root','num_file_creations','num_shells','num_access_files','num
_outbound_cmds','is_host_login','is_guest_login','count','srv_count','serror_rate','srv_se
rror_rate','rerror_rate','srv_rerror_rate','same_srv_rate','diff_srv_rate','srv_diff_host_rat
e','dst_host_count','dst_host_srv_count','dst_host_same_srv_rate','dst_host_diff_srv_rat
e','dst_host_same_src_port_rate','dst_host_srv_diff_host_rate','dst_host_serror_rate','ds
t_host_srv_serror_ra te','dst_host_rerror_rate', 'class'])\n",

"\n",

"df.columns = columns\n",

"pd.set_option('display.max_columns', 500)\n",

"df.head()"

"df_new = df\n",


"from sklearn.preprocessing import LabelEncoder\n",

"labelencoder = LabelEncoder()\n",

"df_new['protocol_type'] = labelencoder.fit_transform(df_new['protocol_type'])\n",

"df_new['service'] = labelencoder.fit_transform(df_new['service'])\n",

"df_new['flag'] = labelencoder.fit_transform(df_new['flag'])\n",

"df_new['class'] = labelencoder.fit_transform(df_new['class'])\n"


"df_new.head()\n"


"X = df_new.iloc[:,0:41] #independent columns\n",

"y = df_new.iloc[:,41] #target column i.e price range"


"from sklearn.feature_selection import SelectKBest, chi2\n",

"bestfeatures = SelectKBest(score_func=chi2, k=10)\n",
```

"fit = bestfeatures.fit(X,y)\n",

"dfscores = pd.DataFrame(fit.scores_)\n",

"dfcolumns = pd.DataFrame(X.columns)"

"dfscores = pd.DataFrame(fit.scores_)\n",

"dfcolumns = pd.DataFrame(X.columns)"

"featureScores = pd.concat([dfcolumns,dfscores],axis=1)\n",

"featureScores.columns = ['Specs','Score']#namingthe dataframe columns\n",

"print(featureScores.nlargest(30,'Score')) #print 10 best features"

"from sklearn.ensemble import ExtraTreesClassifier\n",

"import matplotlib.pyplot as plt\n",

"model = ExtraTreesClassifier()\n",

"model.fit(X,y)\n",

"print(model.feature_importances_) #use inbuilt class feature_importances of tree based classifiers\n",

"#plot graph of feature importances for better visualization\n",

"feat_importances = pd.Series(model.feature_importances_, index=X.columns)\n",

"feat_importances.nlargest(10).plot(kind='barh')\n",

"plt.show()"

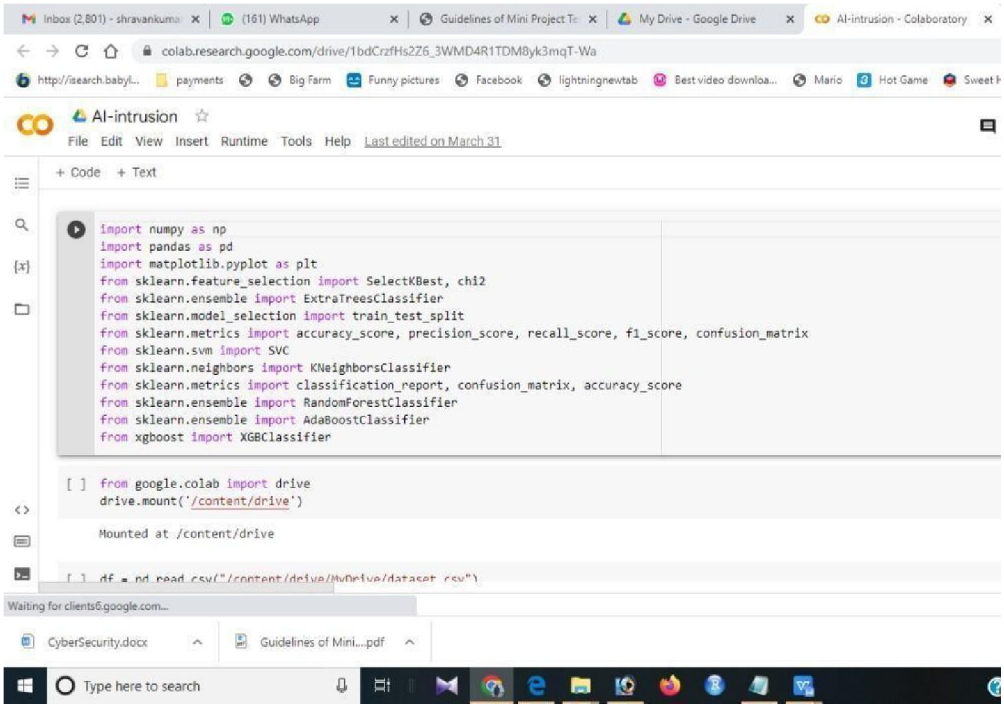
"from sklearn.model_selection import train_test_split\n",

```
"from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix\n",

"X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=0)"
"print(X_train.shape)\n",

"print(y_train.shape)"


"from sklearn.svm import SVC\n",
"model=SVC()\n",
"model.fit(X_train, y_train)\n",
"pred=model.predict(X_test)\n",
"from sklearn.metrics import classification_report, confusion_matrix\n",
"print(classification_report(y_test, pred))\n",
"print('Accuracy: {:.3f}'.format(accuracy_score(y_test, pred)))\n"


"from sklearn.neighbors import KNeighborsClassifier\n",
"classifier = KNeighborsClassifier(n_neighbors=12)\n",
"classifier.fit(X_train, y_train)\n",
"y_pred = classifier.predict(X_test)\n",
"from sklearn.metrics import classification_report, confusion_matrix\n",
"print(confusion_matrix(y_test, y_pred))\n",
"print(classification_report(y_test, y_pred))\n"


"from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score\n",
```

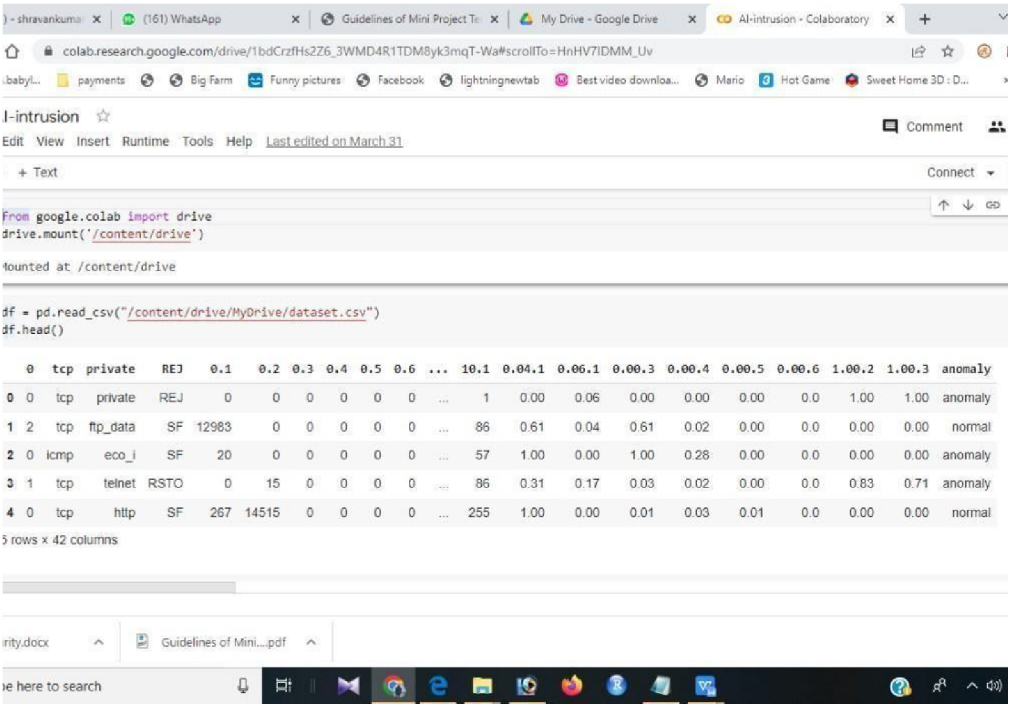
```
"from sklearn.ensemble import RandomForestClassifier\n",
"regressor = RandomForestClassifier(n_estimators=20, random_state=0)\n",
"regressor.fit(X_train, y_train)\n",
"y_pred_class=regressor.predict(X_test)\n",
"print(confusion_matrix(y_test,y_pred_class))\n",
"print(classification_report(y_test,y_pred_class))\n",
"rf = accuracy_score(y_test, y_pred_class)"

"from sklearn.ensemble import AdaBoostClassifier\n",
"model = AdaBoostClassifier()\n",
"model.fit(X_train, y_train)\n",
"y_pred_class=model.predict(X_test)\n",
"y_pred_class = model.predict(X_test)\n",
"print(confusion_matrix(y_test,y_pred_class))\n",
"print(classification_report(y_test,y_pred_class))\n",
"abc = accuracy_score(y_test, y_pred_class)\n",
"abc"

"from xgboost import XGBClassifier\n",
"model = XGBClassifier()\n",
"model.fit(X_train, y_train)\n",
"y_pred_class=model.predict(X_test)\n",
"print(confusion_matrix(y_test,y_pred_class))\n",
"print(classification_report(y_test,y_pred_class))\n",
"xgb = accuracy_score(y_test, y_pred_class)\n",
```

"xgb"

7.2 Results

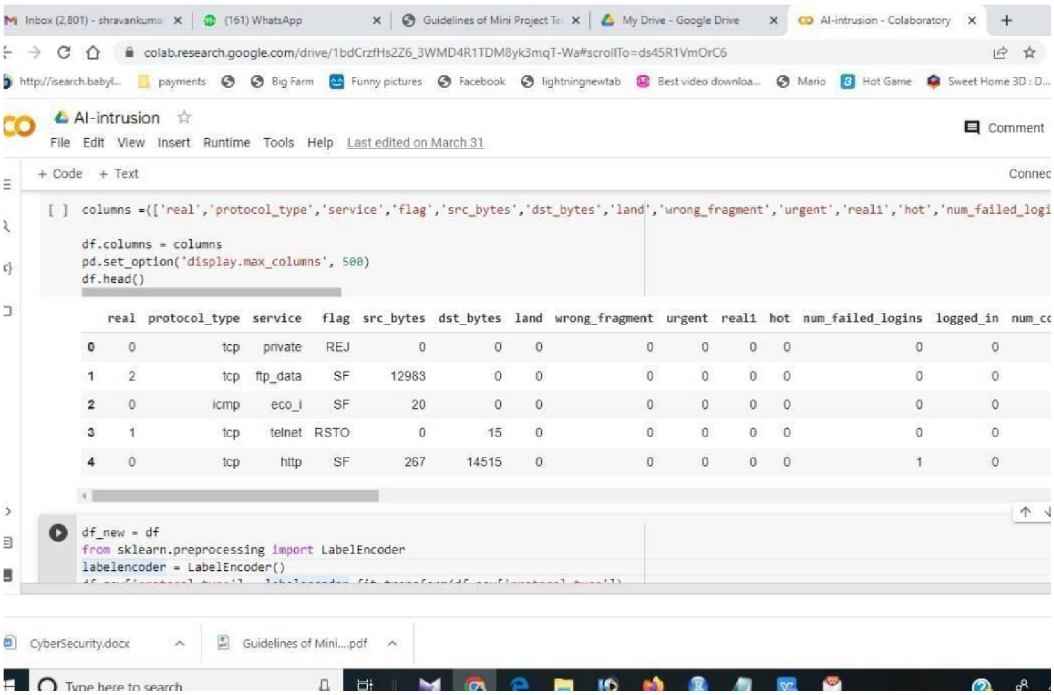


Screenshot 7.1 : Loading the libraries required for the model

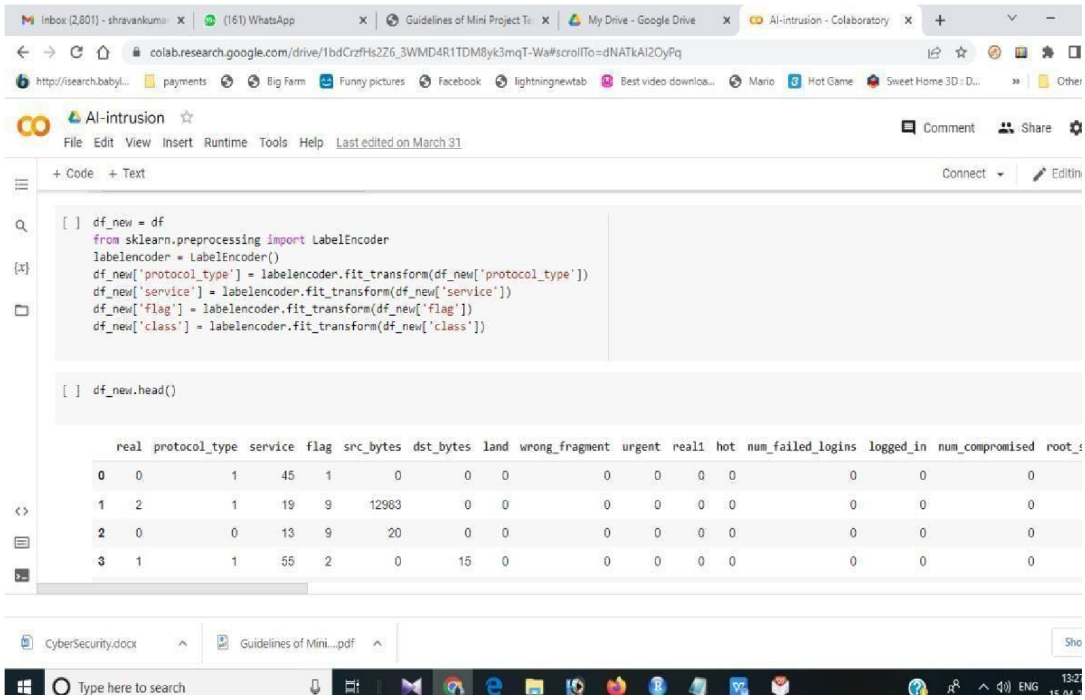


	0	tcp	private	REJ	0.1	0.2	0.3	0.4	0.5	0.6	...	10.1	0.04.1	0.06.1	0.00.3	0.00.4	0.00.5	0.00.6	1.00.2	1.00.3	anomaly
0	0	tcp	private	REJ	0	0	0	0	0	0	...	1	0.00	0.06	0.00	0.00	0.00	0.0	1.00	1.00	anomaly
1	2	tcp	ftp_data	SF	12983	0	0	0	0	0	...	86	0.61	0.04	0.61	0.02	0.00	0.0	0.00	0.00	normal
2	0	icmp	eco_i	SF	20	0	0	0	0	0	...	57	1.00	0.00	1.00	0.28	0.00	0.0	0.00	0.00	anomaly
3	1	tcp	telnet	RSTO	0	15	0	0	0	0	...	86	0.31	0.17	0.03	0.02	0.00	0.0	0.83	0.71	anomaly
4	0	tcp	http	SF	267	14515	0	0	0	0	...	255	1.00	0.00	0.01	0.03	0.01	0.0	0.00	0.00	normal

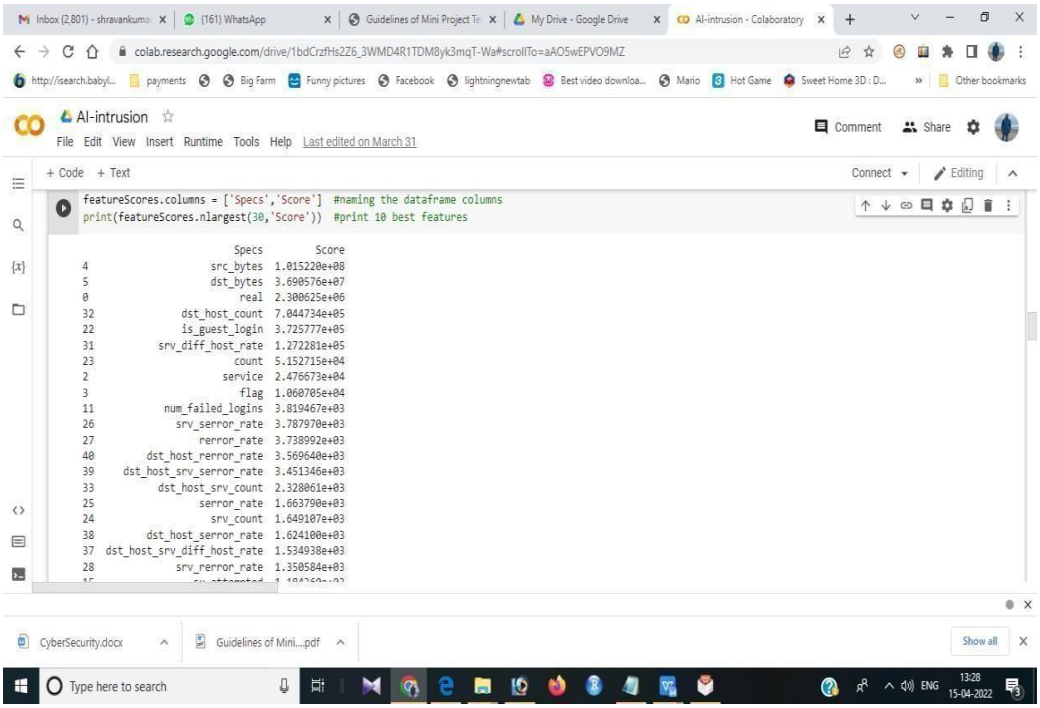
Screenshot 7.2: Loading the data and seeing the few lines of the dataset



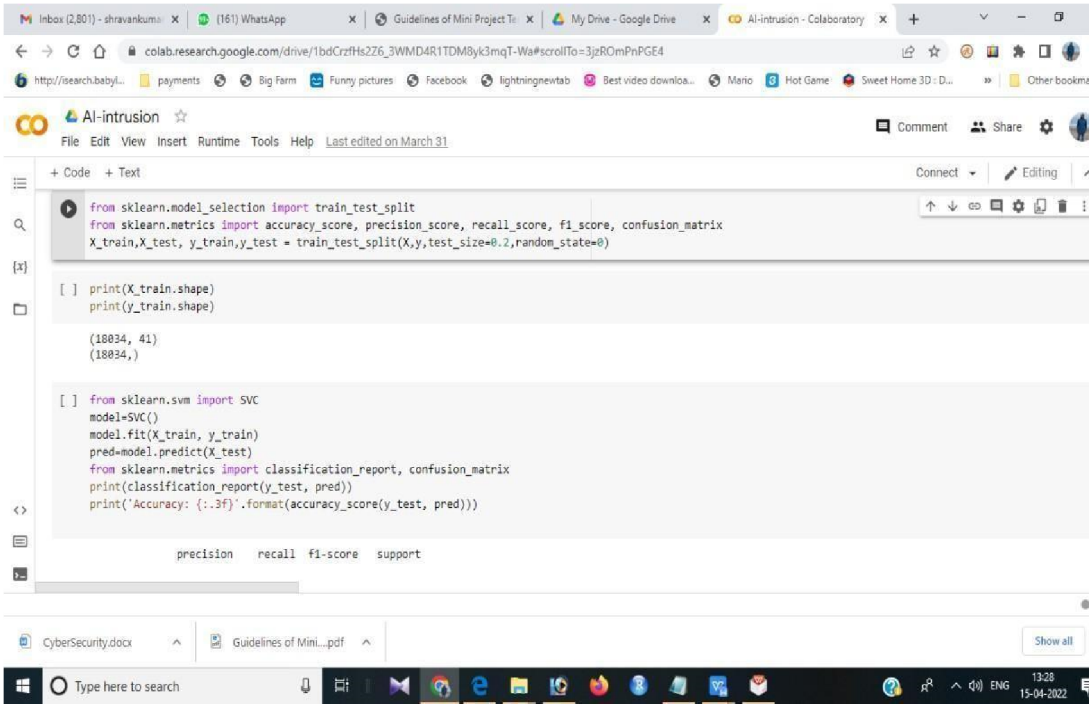
Screenshot 7.3 : Featuring the headers to the dataset



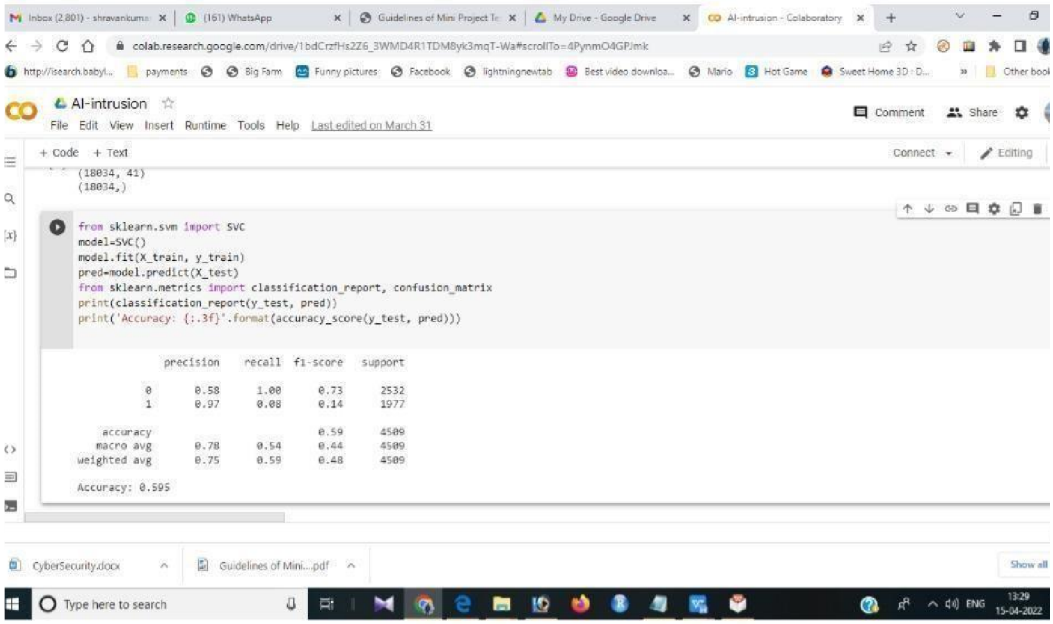
Screenshot 7.4: Label encoder for converting data to numeric format



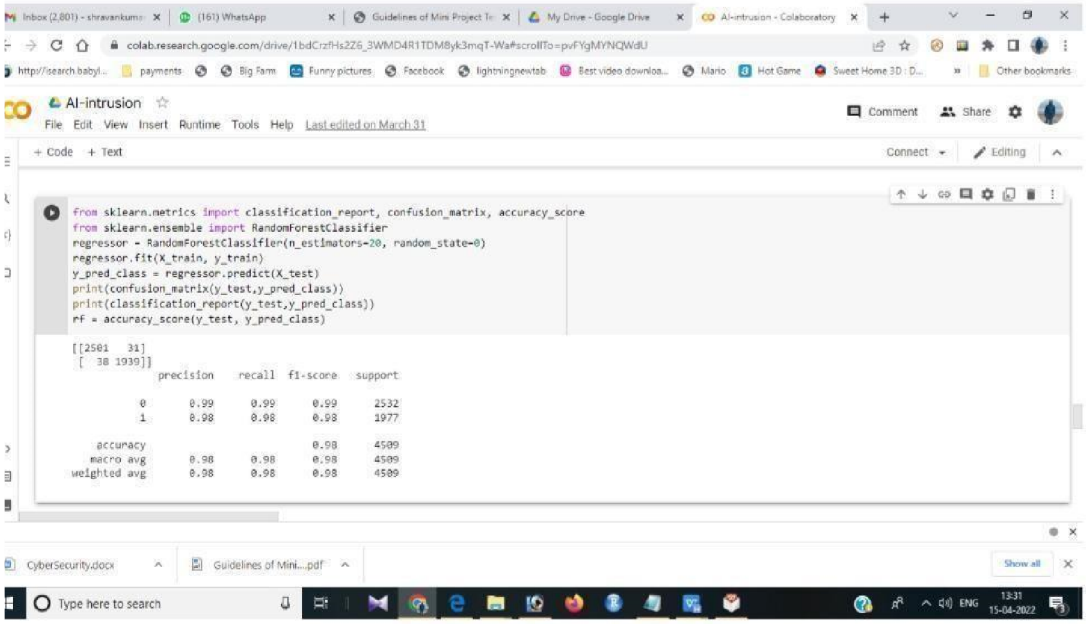
Screenshot 7.5: Chi2 model for identifying the parameters



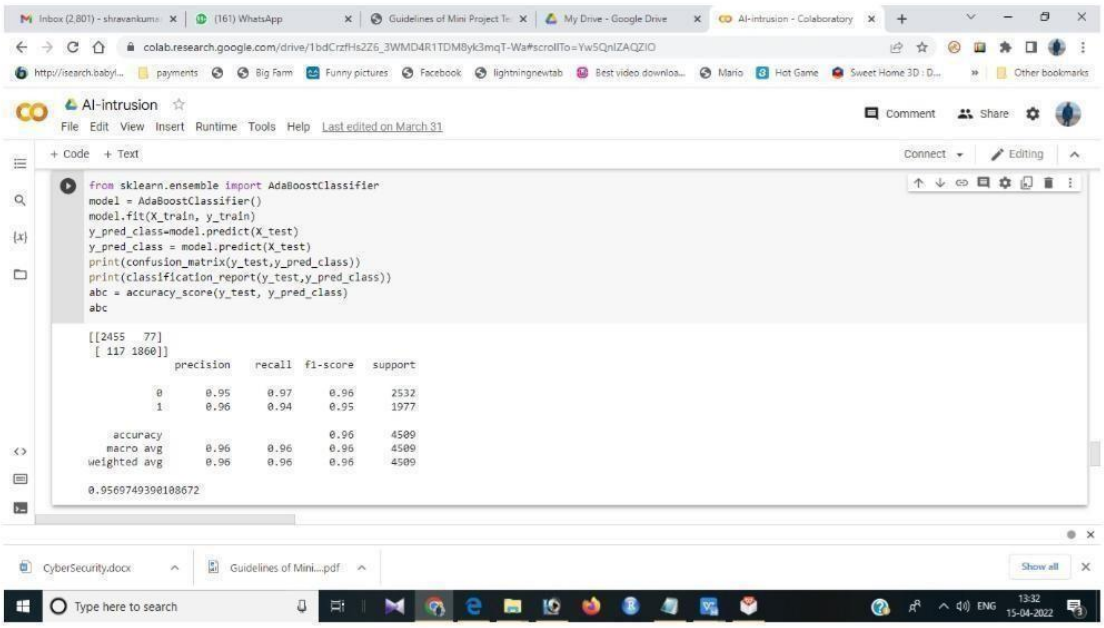
Screenshot 7.6 : Dividing data to training and testing



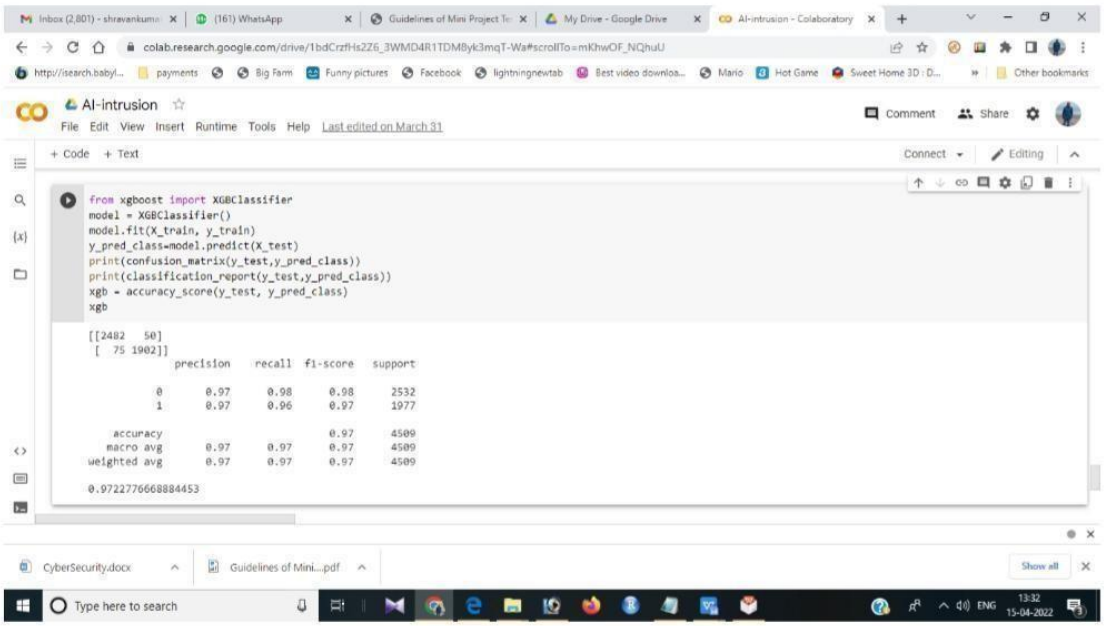
Screenshot 7.7:SVC model for analyzing the data



Screenshot 7.8: Random Forest model for analyzing the data



Screenshot 7.9: Adaboost Classifier model for analyzing the data



Screenshot 7.10: XGB Classifier for model

8.Conclusion & Future Enhancement

8.1 Conclusion

We presented ML technologies in a steel plate production line. We focused on finding high-scored ML algorithms which can be used for the roll force and plate thickness prediction at each rolling pass, so that one can find thebestcontrolconditionstoproducehighqualitysteelplate products.Peer to peer lending is understood as an online activity where P2P platforms act as the marketplace to connect investors and borrowers of funds electronically. There is no need of any prior relationship between both the parties. A simple registration on the platform by lenders and orrowers is followed by a rigorous credit assessment process. The platform analyzes the creditworthiness and then classify loan requests into different risk categories leaving the decision to extend or not expend funds on the lenders only. The present exploratory study has analyzed the reasons for the exponential growth of P2P lending, the challenges and risk factors for different parties.

9. Reference

[1] D. E. Denning, ‘‘An intrusion-detection model,’’ IEEE Trans. Softw. Eng., vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[2] N. B. Amor, S. Benferhat, and Z. Elouedi, ‘‘Naive Bayes vs decision trees inintrusiondetectionsystems,’’inProc.ACMSymp.Appl.Comput.(SAC), 2004, pp. 420–424.

[3] M. Panda and M. R. Patra, ‘‘Network intrusion detection using Naive Bayes,’’Int.J.Comput.Sci.Netw.Secur.,vol.7,no.12,pp. 258–263,2007.

[4] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, ‘‘Support vector machine and random forest modeling for intrusion detection system (IDS),’’ J. Intell. Learn. Syst. Appl., vol. 6, no. 1, pp. 45–52, 2014.

[5] N. Japkowicz, ‘‘The class imbalance problem: Significance and strategies,’’ in Proc. Int. Conf. Artif. Intell., vol. 56, 2000, pp. 111–117.

[6] Y. LeCun, Y. Bengio, and G. Hinton, ‘‘Deep learning,’’ Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[7] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, ‘‘Deep learning for visual understanding: A review,’’ Neurocomputing, vol. 187, pp. 27–48, Apr. 2016.

[8] T. Young, D. Hazarika, S. Poria, and E. Cambria, ‘‘Recent trends in deep learning based natural language processing [review article],’’ IEEE Comput. Intell. Mag., vol. 13, no. 3, pp. 55–75, Aug. 2018

[9] N.Shone,T.N.Ngoc,V.D.Phai,andQ.Shi,‘‘Adeeplearningapproachtonetwork intrusion detection,’’ IEEE Trans. Emerg. Topics Comput. Intell., vol. 2, no. 1, pp. 41–50, Feb. 2018.

[10] D. A. Cieslak, N. V. Chawla, and A. Striegel, ‘‘Combating imbalance in network intrusion datasets,’’ in Proc. IEEE Int. Conf. Granular Comput., May 2006, pp. 732–737.

[11] M. Zamani and M. Movahedi, ‘‘Machine learning techniques for intrusion detection,’’
2013, arXiv:1312.2177. [Online]. Available: http://arxiv. org/abs/1312.2177

[12] M. S. Pervez and D. M. Farid, ‘‘Feature selection and intrusion classification in NSL-
KDD cup 99 dataset employing SVMs,’’ in Proc. 8th Int. Conf. Softw., Knowl., Inf. Manage. Appl. (SKIMA), Dec. 2014, pp. 1–6.

[13] H. Shapoorifard and P. Shamsinejad, ‘‘Intrusion detection using a novel hybrid method incorporating an improved KNN,’’ Int. J. Comput. Appl., vol. 173,
no. 1, pp. 5–9,
Sep. 2017.

[14] S.Bhattacharya,P.K.R.Maddikunta,R.Kaluri,S.Singh,T.R.Gadekallu, M. Alazab, and U.
Tariq,‘‘A novel PCA-firefly based XGBoost
classificationmodelforintrusiondetectioninnetworksusingGPU,’’Electronics, vol. 9, no. 2, p.
219, Jan. 2020.

[15] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, ‘‘A deep learning approach for network intrusion detection system,’’ in Proc. 9th EAI Int. Conf.
BioinspiredInf.Commun.Technol.(FormerlyBIONETICS),2016,pp. 21–26.

[16] P.Torres,C.Catania,S.Garcia,andC.G.Garino,‘‘Ananalysisofrecurrent neural networks for botnet detection behavior,’’ in Proc. IEEE Biennial Congr. Argentina

(ARGENCON), Jun. 2016, pp. 1–6.

[17] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, ‘‘Malware traffic classificationusingconvolutionalneuralnetworkforrepresentationlearning,’’ in Proc. Int.

Conf. Inf. Netw. (ICOIN), 2017, pp. 712–717.

[18] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, ‘‘A survey of deep learningbased network anomaly detection,’’ Cluster Comput., vol. 22, pp.

949–961, 2019.

[19] B. A. Tama, M. Comuzzi, and K.-H. Rhee, ‘‘TSE-IDS: A two-stage

classifierensembleforintelligentanomaly-basedintrusiondetectionsystem,’’ IEEE Access, vol. 7, pp. 94497–94507, 2019.

[20] P. Jeatrakul, K. W. Wong, and C. C. Fung, ‘‘Classification of imbalanced data by combining the complementary neural network and smote algorithm,’’ in Proc.

Int.

Conf.

Neural Inf. Process. Springer, 2010, pp. 152–159.