MALWARE DETECTION IN SELF-DRIVING VEHICLES USING ARTIFICIAL INTELLIGENCE

A FINAL YEAR PROJECT REPORT

Submitted by

T. NISHANTH
(17ECR134)

S. PASUPATHY
(17ECR141)

P. RAGUL
(17ECR154)

in particular fulfillment of the requirements
for the award of the degree
of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

ENGINEERING

SCHOOL OF COMMUNICATION AND COMPUTER SCIENCES

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI ERODE – 638060

APRIL 2021

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638060

APRIL 2021

BONAFIDE CERTIFICATE

This is to certify that the Project report entitled is **MALWARE DETECTION IN SELF-DRIVING VEHICLES USING ARTIFICIAL INTELLIGENCE** is the bonafide record of project work done by NISHANTH.T(17ECR134), PASUPATHY.S(17ECR141), RAGUL.P (17ECR154) in partial fulfilment of the requirements for the award of the Degree of Bachelor of Engineering in Electronics and Communication Engineering of Anna university Chennai during the year 2020 - 2021.

SUPERVISOR	**HEAD OF THE DEPARTMENT**
Ms. A. Vennila BE., ME	**Dr. T. Meeradevi BE., ME., PhD**
Assistant Professor	Professor & Head
Department of ECE	Department of ECE
Kongu Engineering College	Kongu Engineering College
Perundurai - 638060	Perundurai - 638060

Date:

Submitted for the end semester viva voce examination held on __________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638060

APRIL 2021

DECLARATION

We affirm that the Project report titled **MALWARE DETECTION IN SELF-DRIVING VEHICLES USING ARTIFICIAL INTELLIGENCE** being submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Date:

(Signature of the candidate)

NISHANTH T
(17ECR134)

PASUPATHY S
(17ECR141)

RAGUL P
(17ECR154)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date: **Name and Signature of the Supervisor with seal**

ABSTRACT

The recent trend to connect vehicles to vehicles, unspecified devices and infrastructure creates the potential for various external threats to vehicle cyber security especially in the self-driving cars. Therefore, intrusion detection combined with machine learning is an effective network safety key in vehicles with open connections in self-driving vehicles. Specifically, when a vehicle is connected to an external device via an internal smartphone or when the vehicle communicates with external infrastructure, security technologies are required to protect the software network within the vehicle. The technology available with this function includes car gates and access systems. However, it is difficult to block malicious code based on app behavior. In this project, we propose a artificial intelligence based data analysis method to accurately detect unusual behavior due to high network traffic malware in real time. We describe the design of the acquisition, which is required by the access module to detect and block malware attempting to touch the car with a smartphone and also developed an effective algorithm to detect malicious behaviors in the network environment and perform tests to verify the accuracy of the algorithms.

ACKNOWLEDGEMENT

First and foremost, we sincerely thank our respected Correspondent **THIRU.P. SACHITHANANDAN** for providing all the necessary facilities to complete the course successfully.

We wish to express our profuse thanks to our respected principal **Dr. V. BALUSAMY B.E (HONS)., MTech., PhD.,** for his encouragement during the course of study.

We solemnly express our gratitude to our Head of the Department **Dr. T. MEERADEVI B.E., M.E., Ph.D.,** for her encouragement and constant support that initiated us to complete this project.

We express our sincere thanks to our Project Coordinator. **Dr. A. ARULMURUGAN B.E., M.E., PhD.,** Assistant Professor, Department of ECE who had been a source of external encouragement to us.

We extent our sincere gratitude to our guide **Ms. A. VENNILA BE., ME.,** Professor, Department of ECE who showered us with multiple ideas.

Finally, we wish to express our sincere thanks to all faculty and staff of Department of Electronics and Communication Engineering and our friends for their support towards the successful completion of this project.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	**iv**
	LIST OF TABLES	**v**
	LIST OF FIGURES	**viii**
	LIST OF TABLES	**ix**
1	**INTRODUCTION**	**1**
2	**LITERATURE REVIEW**	**3**
3	**GAP ANALYSIS AND PROBLEM STATEMENT IDENTIFICATION**	**4**
	3.1 MALWARE FAMILIES	4
	3.2 PROBLEM STATEMENT	6
	3.3 EXISTING SYSTEM	6
4	**PROPOSED METHOD**	**8**
	4.1 MACHINE LEARNING BASED INTRUSION DETECTION MODULE	8
	4.2 DATA PROCESSING FLOWCHART	9
	4.3 DATASET	10
	4.4 PREPROCESSING TECHNIQUES	12
	4.4.1 SELECT K-BEST	12
	4.4.2 SK LEARN	12
	4.4.3 VARIATION INFLATION FACTOR	13
	4.5 ALGORITHM USED	14
	4.5.1 LINEAR REGRESSION	14
	4.5.2 LOGISTIC REGRESSION	15
	4.5.3 DUMMY CLASSIFIER (ZERO R)	16
	4.5.4 DECISION TREE ALGORITHM	16

4.5.5 RANDOM FOREST ALGORTIHM 17

4.5.6 KNN ALGORITHM 18

5 IMPLEMENTATION AND RESULT ANALYSIS 18

5.1 ACCURACY OF TRAINING AND TESTING DATA 18

5.1.1 LINEAR REGRESSION 18

5.1.2 LOGISTIC REGRESSION 19

5.1.3 DUMMY CLASSIFIER (ZERO R) 20

5.1.4 DECISION TREE ALGORITHM 20

5.1.5 RANDOM FOREST ALGORITHM 21

5.1.6 K NEAREST NEIGHBOURS 21

5.2 OUTPUT ACCRUACY TABLE 22

6 CONCLUSION AND FUTURE SCOPE 22

6.1 CONCLUSION 24

6.2 FUTURE SCOPE 24

7 REFERENCES 25

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.3	Block diagram of existing system	7
4.1.a	Head unit connected to an android mobile device	8
4.1.b	CAN topology with gateway and intrusion detection module	9
4.2	Proposed model flow chart	10
4.5.1	Relationship between dependant and independent variables	14
4.5.2	‘S’ curve of logistic function	15
4.5.4	Structure of Decision tree algorithm	16
4.5.5	Architecture of Random forest (RF)	17
4.5.6	KNN model graph	18
5.1.1	Linear Regression algorithm Result	19
5.1.2	Results of Logistic Regression algorithm	19
5.1.3	Results of Dummy classifier algorithm	20
5.1.4	Confusion matrix result for Decision Tree Algorithm	20
5.1.5	Result of Random forest algorithm.	21
5.1.6	Predicted output of KNN	21

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
4.3	Classification of malware families with number of samples	25
4.4.3	VIF features based on ANOVA and mutual information gain	13
5.2.a	Accuracy and f1 scores of algorithms	22
5.2.b	Predicted output of each algorithm	23

CHAPTER 1

INTRODUCTION

As cars become more intelligent, so are travel systems. New business needs in the automotive market and advances in automotive communication technologies increase automotive communication. This massive interaction reveals the increased likelihood of future car crashes. Therefore, it is necessary to prepare various anti-aircraft missiles to combat threats to vehicle safety.

For example, in 2015, Miller and Valasek hijacked a moving Jeep Cherokee to control wipers, air protectors, steering wheel and brakes, suggesting that an unresolved cyber security system could threaten the safety of drivers. In addition, in 2016 and 2017, Keen Security Lab hacked into a Tesla vehicle to signal security threats and potential attacks related to connected vehicles. Typically, connected vehicles are a closed environment that only accepts remote control commands via an authorized communication system, such as a manufacturer-built server or dedicated services published by the manufacturer. In a closed period, environment, unauthorized commands are blocked.

However, the latest self-driving cars share their control signals and internal data not only with the controllers inside the car, but also with various unspecified vehicles, infrastructure, and smart devices outside the car in real time. Therefore, vehicle network protection should be a priority in open spaces.

Self-driving car safety is directly related to passenger safety, it is necessary to carefully consider the various attacking vectors against vehicles based on the integrity, discovery, and confidentiality of their cybersecurity. When the connected car software is updated, it is important to ensure the integrity of the software. Attackers can use malicious programs like malwares to steal rights illegally or gain access, re-install software installed on a car by injecting incorrect code, various kinds of viruses and make malicious modified applications.
Machine learning is the study of computer algorithms that develop automatically through experience and data usage. It seems to be part of the artificial intelligence.

Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

By using Machine learning for detecting the malwares and classifying it using five different types of algorithms which reduces analysis time and improve access accuracy. By including a greater number of malware families in order to get more accuracy during prediction gives a more accurate result at real time environment.

(i) Introduce a way to detect adware and malware in the area of self-driving vehicles.

(ii) Defines the structure of the access point module required to detect malware and prevent it from crashing into a car via smartphone.

(iii) Comparing the accuracy of acquisition with the cost of different algorithms and present the most advanced algorithm.

CHAPTER 2

LITERATURE REVIEW

Seunghyun Park and Jin-Young Choi has developed a machine learning-based data analysis method to accurately detect abnormal behaviours due to malware in large-scale network traffic in real time. They define a detection architecture, which is required by the intrusion detection module to detect and block malware attempting to affect the vehicle via a smartphone. Proposed an efficient algorithm (Random Forest, KNN, Decision tree) for detecting malicious behaviors in a network environment.

Daniel Gibert, Carles Mateu and Jordi Planes proposed a systematic and detailed overview of machine learning techniques for malware detection and in particular, deep learning techniques. This model provides a complete description of the methods and features in a traditional machine learning workflow for malware detection and classification it analyses recent trends and developments in the field with special emphasis on deep learning approaches. They uses Select-K best Techniques while preprocessing the Datasets. The 30% Datasets were used for testing with the accuracy of 90%.

Sherif Saad, William Briguglio and Haytham Elmiligi give a detailed report of malware detection in the wild present unique challenges for the current state-of-the-art machine learning techniques. They defined three critical problems that limit the success of malware detectors powered by machine learning in Smartphones. They use the Machine learning algorithms (Random forest, KNN, K-Neighbors) in different types of malware families by using Confusion Matrix for precision and accuracy. It gives a better accuracy result of 92.08% in prediction at real time networks.

Saurabh Jha**,** Timothy Tsai had introduced an attack detection model, a method to deploy the attack in the form of smart malware, and an experimental evaluation of its impact on production-grade autonomous driving software. Using Deep Learning (K means algorithm) they achieved the higher accuracy in determining Trojan and Adware malware types. They use the samples which takes longer time for testing more than 30 mins for two malware families.

Balaji Baskaran and Anca Ralescu research focused on machine learning algorithms that analyze features from malicious application and use those features to classify and detect unknown malicious applications. This study summarizes the evolution of malware detection techniques based on machine learning algorithms focused on the Android OS. It Connected to a networking layer enables to scan the incoming data from the Android OS for any malicious code or application which root cause to the stealing of personal data or hacking the access.

CHAPTER 3

GAP ANALYSIS AND PROBLEM STATEMENT IDENTIFICATION

Malware family paves the major role in hijacking the self-driving vehicles. So, it is important to identify all types of malware families during the malware detection system.

3.1 MALWARE FAMILIES

Malware is a program designed to gain access to computer systems, usually for the benefit of another person, without the user's consent. Malware includes computer viruses, worms, Trojan horses, malware, spyware and other malicious programs.

Virus

Virus is a malicious portable code attached to another usable file. The virus is spread when an infected file is uploaded from system to system. Viruses are harmless or may change or delete data. Opening a file can create a virus. Once the system virus is running, it will infect other programs on the computer.

Worms

Worms replicate themselves in the system, attaching themselves to separate files and searching for ways between computers, such as a computer network that shares common file storage locations. Worms often slow down networks. The virus needs a host system to work but worms can work on their own. After the worm hits the administrator, it is able to spread very quickly on the network.

Spyware

Its purpose is to steal confidential information from a third-party computer program. Spyware collects information and sends it to hackers.

Trojan horse

Trojan Horse is a malware that performs dangerous functions under the required functionality such as playing online games. Trojan horse differs from virus because Trojan commits to unsuccessful files, such as image files, audio files.

Ransomware

Ransomware hosts a computer program or data content until the victim pays. Ransomware encrypts data on a computer with an unknown user key. The user has to pay a fine (price) to the criminals to get the data. Once the amount has been paid the victim can continue to use their plan.

Adware

Adware (or advertising software) is the term used for various pop-up advertisements that show up on your computer or mobile device. Adware has the potential to become malicious and harm your device by slowing it down, hijacking your browser and installing viruses and/or spyware.

Benign

A prank virus that does not cause damage. It does such things as randomly displaying a message on screen or causing the computer to make a clicking sound every time a key is pressed. Fortunately, most viruses are benign.

Keyloggers

Keylogger records everything a user types on their computer system to retrieve passwords and other sensitive information and send it to the source of the keylogging program.

Background doors

The backdoor exceeds the standard authentication used to access the system. The purpose of the external environment is to give cyber criminals access to the future of the system even if the organization fixes the initial vulnerability used to attack the system.

Rootkits

The rootkit modifies the OS to make it a back door. The attackers use the back door to access the computer remotely. Most rootkits take advantage of software vulnerabilities to modify system files.

Scareware

A malware tactic that manipulates users into believing they need to download or buy malicious, sometimes useless, software. Most often initiated using a pop-up ad, it uses social engineering to take advantage of a user's fear, coaxing them into installing fake anti-virus software.

SMS malware

SMS attacks involve the creation and distribution of **malware** by cybercriminals designed to target a victim's mobile device. These Trojan, in turn, are designed to make unauthorized calls or send unauthorized texts without the user's knowledge or consent.

3.2 PROBLEM STATEMENT

As malwares are in vast categories so it is difficult to make a system with limited number of malware types. Because of less safety in existing system, we use detection system.

Detection system detects the android auto malware of the automated vehicle. Detection system consists of two phases preprocessing and modelling. At the end of modeling detects the which type of malware that attacks the vehicle system by increasing the accuracy of the prediction.

3. 3 EXISTING METHOD

In malware detection system we have a tendency to area unit victimization navie Bayes formula for classification with accuracy. A positive use of naive Bayes is that it solely needs a little quantity of coaching information to estimate the parameters (means and variances of the variables) necessary for classification. As a result of freelance variables area unit assumed, solely the variances of the variables for every category have to be compelled to be determined and not the complete variance matrix.

The existing work describes the architecture in various modules which are

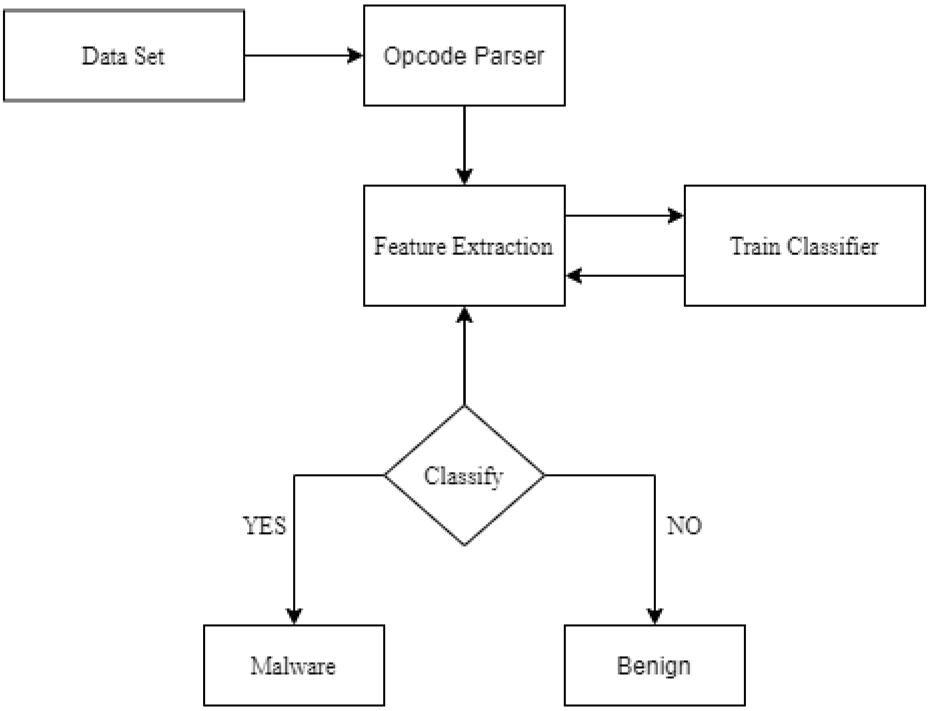


Figure 3.3: block diagram of existing system

There also not much more malware family is detected only malware, benign and benign is detected. They pre-process the algorithm in order to reduce the dataset, but it reduces the accuracy of the algorithm.

The number of malware families used for detection is less so it may not detect when unauthorized malware is attacked. This reduces the safety of the system when in implementation.

CHAPTER 4

PROPOSED METHOD

4.1 MACHINE LEARNING BASED INTRUSION DETECTION MODULE

Connected or self-driving vehicles are connected to external or public networks outside the vehicle via various connectors. TCUs or CCUs have modems and external communication channels to enable Global Positioning System signal reception and access to mobile networks. In-car infotainment systems, which provide entertainment and information, enable various applications by installing an embedded OS, such as QNX OS or Android OS. If security configurations are ignored in wireless networks, these frameworks may be misused as a form of computer-sensitive commands or malicious commands to enter the automotive network. Basically, the embedded OS environment can be controlled from malicious commands or malicious commands where these malicious processes violate the OS-level security standard or derive root authority from copyright infringement. Therefore, to prevent malicious commands from accessing embedded OS controls, the construction of a CAN gate that includes an internal access module which detects the malicious behavior when devices based on Android OS are connected to a car.

Machine learning module was included in the vehicle's IDS, which could detect CAN input from any abnormality, so that the head unit or ECU was protected from malicious code. Such detection methods are performed in the form of computer-based software modules to monitor malware injection or malicious behavior on the vehicle. The software can be installed as part of a car login detection module or as an anti-virus agent in the main head.

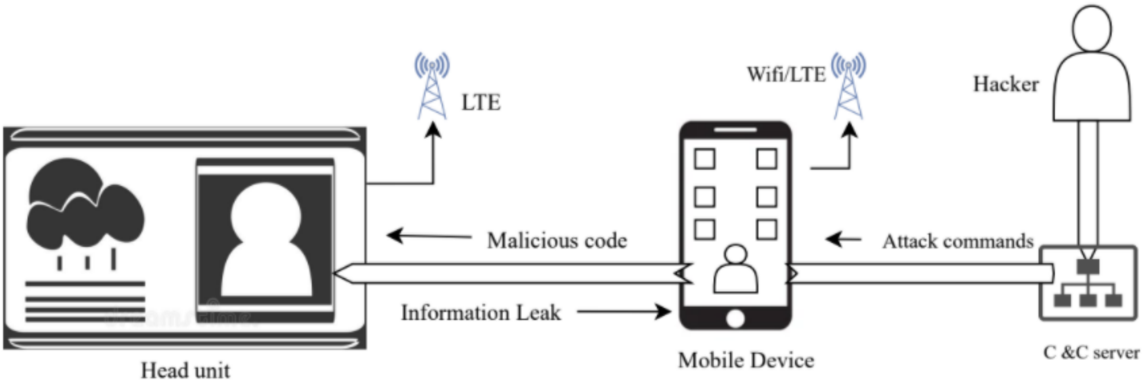


Figure 4.1.a: Head unit connected to an Android mobile device.

The proposed acquisition software has modules for installation, analysis, testing and reporting. CAN-entered traffic is analyzed by the input module and integrated into the analysis module, which is fitted with a machine learning algorithm. The analysis mode detects an intrusion or abnormal behavior based on the learned model and provides details of the login behavior to the user or control center in real time. This machine-based learning module can improve model accuracy by frequently reading, verifying, and checking message patterns. In addition, the rules for the detection of malicious behavior can be reviewed at the car gate and controls each individual to receive malicious code.

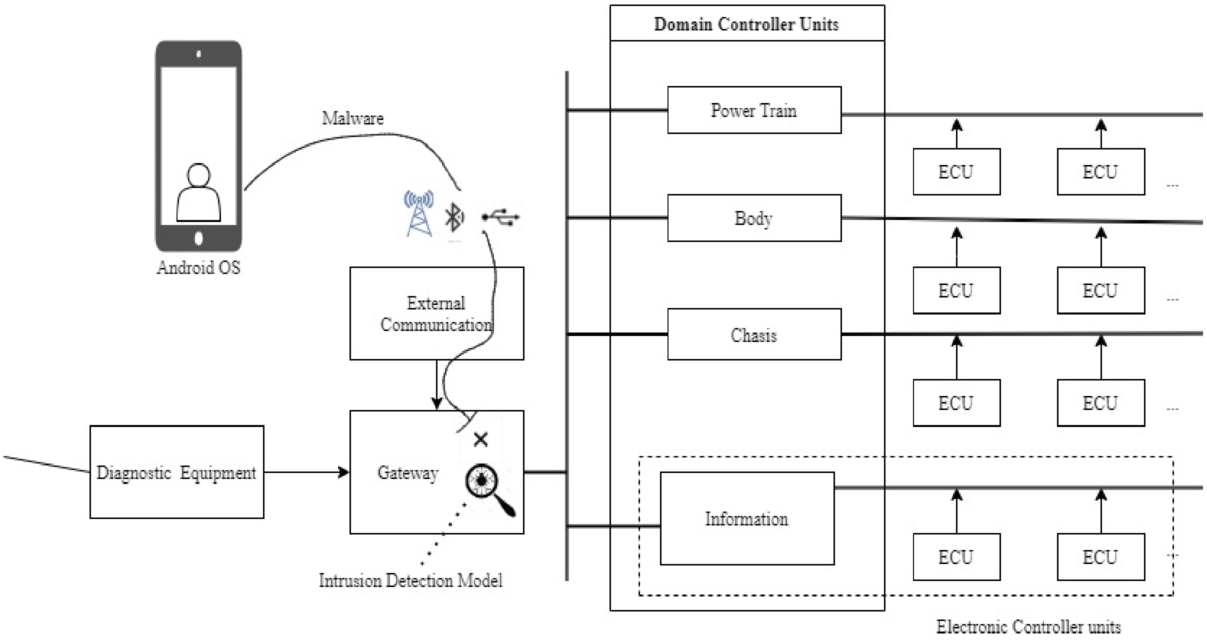


Figure 4.1.b: CAN topology with gateway and intrusion detection module

4.2 DATA PROCESSING FLOWCHART

Detection module based on the proposed machine learning detects the Android auto malware of the automated vehicle and labels its type (e.g., adware or common malware). The process, based on detecting network deviations in the Android OS, is divided into three phases. The first phase focuses on data expansion. Feature selection is made to select the most important features of all the measurement features in the database. The second category consists of modeling. Using a ten-fold race, this section trains a machine learning model using 75% of the database and suggests the most appropriate hyperparameters for the retraining model. In addition, this section uses 25% of the database to test and evaluate the proposed access module. Therefore, a machine learning model prepared for hyperparameters is created using a training database and using a model test database. In the third stage, the Intrusion detection module can detect bad behavior in real time when real data enters the self -driving vehicle.

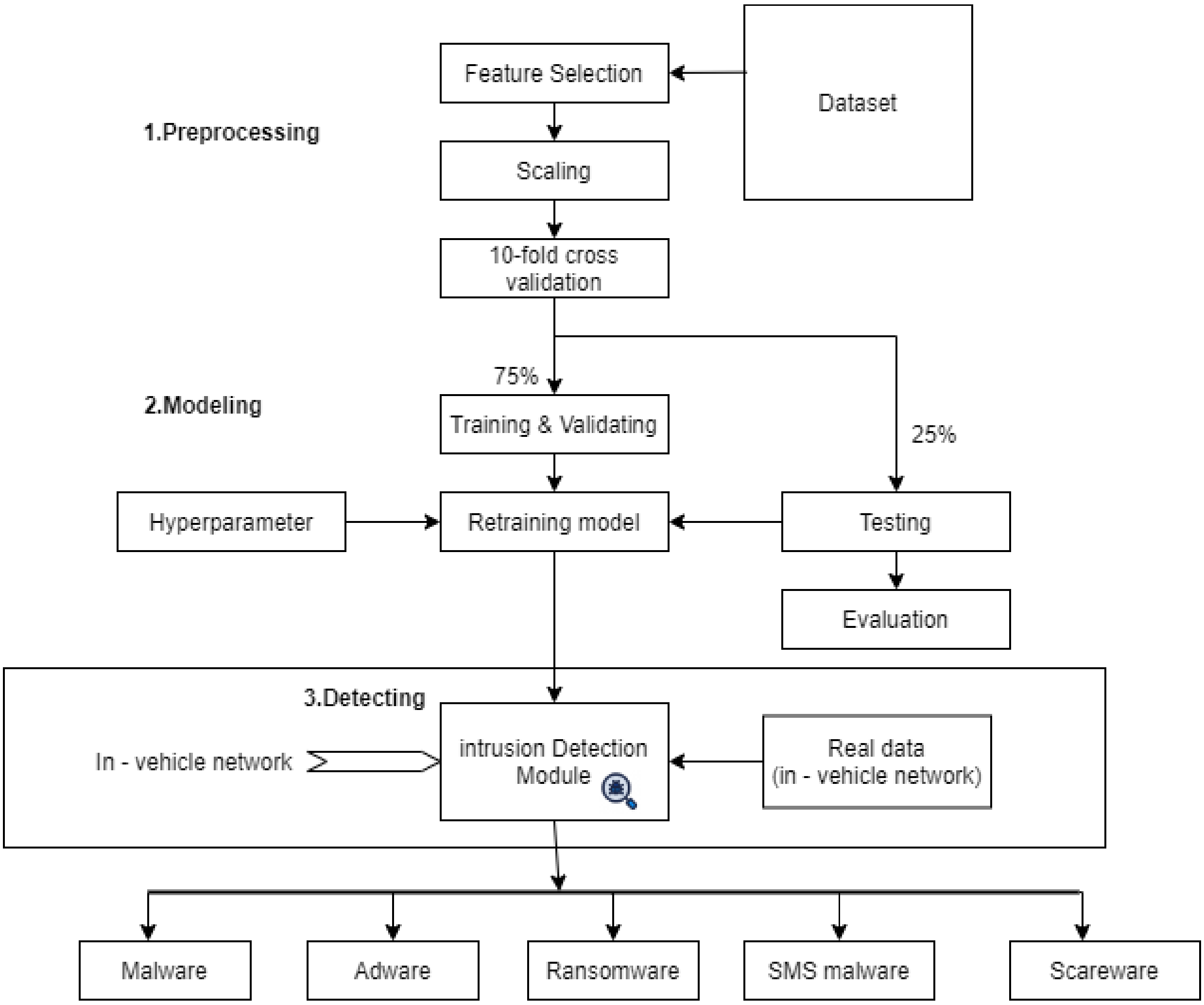


Figure 4.2: Proposed model flow chart

The dataset is taken from the Canadian Institute of Cybersecurity (CIC-AndMal2017) which contains the data of Adware, Benign, Ransomware, Scareware and SMS malware. It also has 42 unique malware families.

4.3 DATASET

The dataset is gathered from Canadian Institute of Cyber security (UNB) portal. The name of the dataset is CIC-AndMal2017 which consists of 5 major types of malware families with a collection of 42 unique malware families. This dataset is taken from the real time android network devices. The dataset contains more than 352,992 samples (4,354 - malware, 72802 -

Benign, 84556 - Adware, 70230 - Ransomware, 80161 - Scareware and 45544 - SMS malware) from several sources.

MALWARE TYPES	**SAMPLES**	**MALWARE TYPES**	**SAMPLES**
BENIGN	72802	SMSMALWARE_BIIGE	6686
ADWARE_GOOLIGAN	18695	SCAREWARE_FAKETAOBAO	6669
SCAREWARE_ANDROIDDEFENDER	11460	SMSMALWARE_SMSSNIFFER	6617
ADWARE_FEIWO	11311	RANSOMWARE_WANNALOCKER	6581
RANSOMWARE_SVPENG	10933	ADWARE_KOODOUS	6433
RANSOMWARE_PORNDROID	9167	SCAREWARE_FAKEJOBOFFER	6189
RANSOMWARE_KOLER	8952	ADWARE_MOBIDASH	6116
SMSMALWARE_NANDROBOX	8876	RANSOMWARE_LOCKERPIN	5264
SCAREWARE_FAKEAPPAL	8732	RANSOMWARE_JISUT	5096
ADWARE_EWIND	8729	SCAREWARE_ANDROIDSPY	5071
SCAREWARE_AVFORANDROID	8503	SCAREWARE_VIRUSSHIELD	4755
RANSOMWARE_RANSOMBO	8170	SMSMALWARE_FAKENOTIFY	4469
SCAREWARE_AVPASS	8144	MALWARE	4354
SCAREWARE_FAKEAV	7972	SCAREWARE_PENETHO	4348
SMSMALWARE_PLANKTON	7952	SMSMALWARE_FAKEINST	2963
ADWARE_DOWGIN	7871	ADWARE_SELFMITE	2585
RANSOMWARE_CHARGER	7851	SMSMALWARE_BEANBOT	2419
ADWARE_SHUANET	7825	SMSMALWARE_ZSONE	1927
ADWARE_KEMOGE	7658	SCAREWARE	1927
ADWARE_YOUMI	7336	SMSMALWARE_FAKEMART	1303
RANSOMWARE_SIMPLOCKER	7262	SMSMALWARE_MAZARBOT	1235
SCAREWARE_FAKEAPP	6853	SMSMALWARE_JIFAKE	1127

Figure 4.3: Classification of malware families with number of samples

Each malware is grouped with 85(columns) values of data which gives in-depth of the data collections. The types of malware included in the dataset is Adware, Ransomware, SMS malware, Scareware and Benign. Apart from benign all has its different types of malware. the dataset contains a total of 3,52,882 samples. This classification types of malwares contain its own individual characteristics so that it can be more accurate for the identification of malware family.

4.4 PREPROCESSING TECHNIQUES:

The base of any AI projects is laid by the datasets fed into the model. For a precise result, the obtained dataset must be tuned or reshaped before used in model, which is then termed as the pre-processing of the dataset. Pre-processed the data in our proposed model in terms of different factors,

i. Select K Best
ii. SK Learn
iii. Variation Inflation Factor

4.4.1 SELECT K-BEST

The classes in the sklearn.feature_selection module can be used for feature selection /dimensionality reduction on sample sets, either to improve estimators' accuracy scores or to boost their performance on very high-dimensional datasets.

As dataset contains 353882 rows and 85 columns. Our primary work is to remove the unwanted columns in the dataset in order to get higher accuracy. By using Select K-best module for knowing the weight of every column and relieving the other lower accuracy columns in order to save the run time of the program.

4.4.2 SK Learn

Scikit-learn is probably the most useful library for machine learning in Python. The SKlearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.Some popular groups of models provided by scikit-learn include:

- **Clustering**: for grouping unlabelled data such as KMeans.
- **Cross Validation**: for estimating the performance of supervised models on unseen data.
- **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behaviour.
- **Dimensionality Reduction**: for reducing the number of attributes in data for summarization, visualization, pre-processing the dataset and feature selection such as Principal component analysis.
- **Feature extraction**: for defining attributes in image and text data.

- **Feature selection**: for identifying meaningful attributes from which to create supervised models.
- **Manifold Learning**: For summarizing and depicting complex multi-dimensional data.
- **Supervised Models**: a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

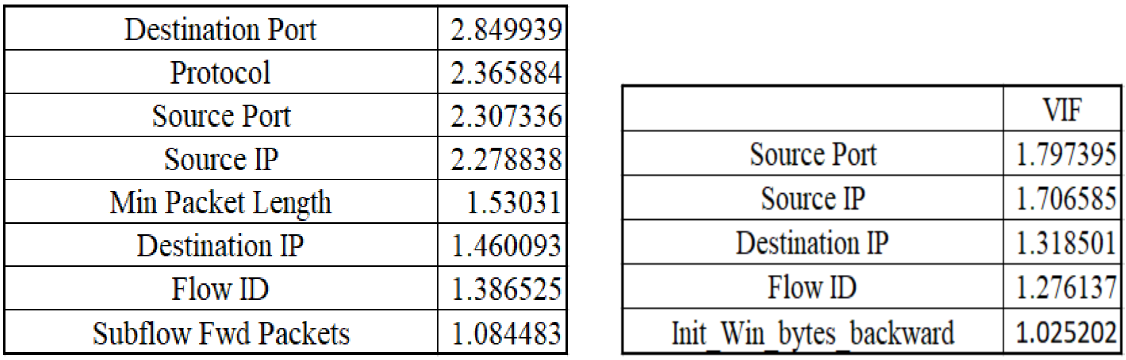
4.4.3 VARIATION INFLATION FACTOR

Variance inflation factor (VIF) is a measure of the amount of multicollinearity in a set of multiple regression variables. Mathematically, the VIF for a regression model variable is equal to the ratio of the overall model variance to the variance of a model that includes only that single independent variable. This ratio is calculated for each independent variable. A high VIF indicates that the associated independent variable is highly collinear with the other variables in the model.

It also helps to identify the degree of multicollinearity. A multiple regression is used when a person wants to test the effect of multiple variables on a particular outcome. The dependent variable is the outcome that is being acted upon by the independent variables—the inputs into the model. Multicollinearity exists when there is a linear relationship, or correlation, between one or more of the independent variables or inputs.

A rule of thumb for interpreting the variance inflation factor:

- 1 = not correlated;
- between 1 and 5 = moderately correlated;
- greater than 5 = highly correlated.



Destination Port	2.849939
Protocol	2.365884
Source Port	2.307336
Source IP	2.278838
Min Packet Length	1.53031
Destination IP	1.460093
Flow ID	1.386525
Subflow Fwd Packets	1.084483

	VIF
Source Port	1.797395
Source IP	1.706585
Destination IP	1.318501
Flow ID	1.276137
Init_Win_bytes_backward	1.025202

Figure 4.4.3: VIF features based on ANOVA and mutual information gain:

4.5 ALGORITHM USED

In its basic operation, machine learning uses structured algorithms that detect and analyze input data to predict output values at an acceptable range. Algorithms like Random Forest, Linear Regression, Logistic Regression,

4.5.1 LINEAR REGRESSION

Line Regression is one of the easiest and most popular ways to learn mechanically. It is a mathematical method used for forecasting analysis. Line reversals make predictions for continuous / real or statistical variations such as sales, salary, age, product price, etc. The Linear regression algorithm shows the parallel relationship between dependence (y) and one or more independent variants (y), hence its term as linear regression. As the reversal of the line indicates a direct relationship, it means that it finds that the value of the variance on which it depends varies according to the value of the independent variance. The linear regression model provides a straight dashed line that represents the relationship between the variables. Consider the picture below:

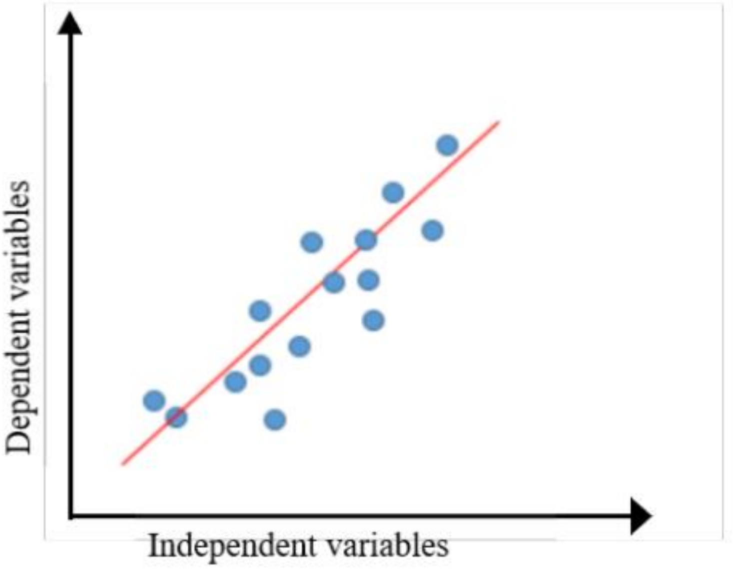


Figure 4.5.1: Relationship between dependant and independent variables

Mathematically, we can represent a linear regression as:

$y= a_0+a_1x+ \varepsilon$

Here,

Y= Dependent Variable (Target Variable)
X= Independent Variable (predictor Variable)
a0= intercept of the line (Gives an additional degree of freedom)
a1 = Linear regression coefficient (scale factor to each input value).
ε = random error.

4.5.2 LOGISTIC REGRESSION

Retirement is one of the most popular forms of machine learning, which falls under a supervised learning approach. It is used to predict phase-based fluctuations using a given set of independent variables. A depreciation of assets predicts the release of class-based variability. Therefore, the result should be a separate or different value. It can be Yes or No, 0 or 1, True or False, etc. But instead of giving a direct value like 0 and 1, it gives values that are probably between 0 and 1. Logistic Regression is almost identical to Linear Regression regardless of how it is used. Linear Regression is used to solve Regression problems, while Logistic regression is used to solve division problems. In Logistic regression, instead of inserting a regression line, we measure the logistic function "S" which predicts two higher values (0 or 1). A curve from the function of the material indicates the potential for something like cancer cells or not, the mouse is overweight or not based on its weight, etc. Logistic Regression is an important machine learning algorithm because it has the ability to provide opportunities and separate new data using continuous and different data sets. Logistic Regression can be used to classify recognition using a variety of data types and can easily determine the most effective variables used for classification. The image below shows the function of objects:

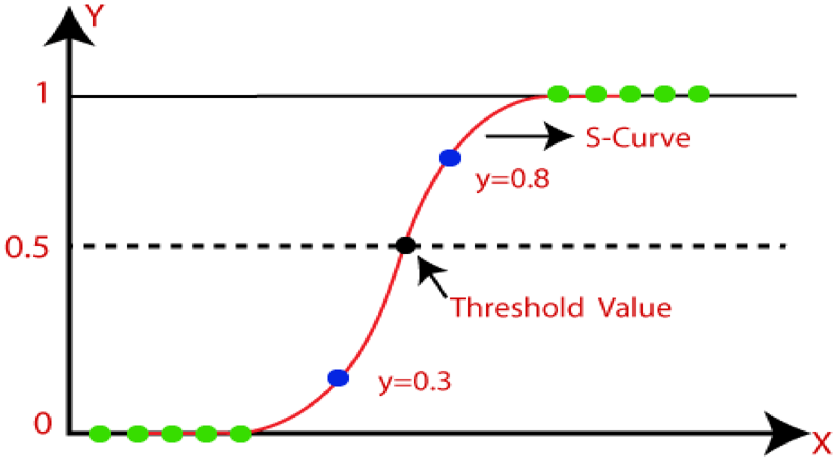


Figure 4.5.2: 'S' curve of logistic function

4.5.3 DUMMY CLASSIFIER (ZERO R)

A dummy splitting is a type of separation that does not generate understanding about the data and separates the given data using only simple rules. Organizer behavior is entirely based on training data as the trend of training data is completely ignored and instead uses one of the class label prediction strategies. It is only used as a simple basis for other partitions that any other partitions are expected to perform better on a given database. It is especially useful for data sets where class inequality is ensured. It is based on the philosophy that any method of analyzing the problem of segregation should be better than a random guessing method.

4.5.4 DECISION TREE ALGORITHM

Decision Tree is a supervised learning technique that is useful for classification and regression problems but is more likely to solve taxonomic problems. It is a tree-structured taxonomy, where the internal nodes represent the characteristics of the dataset, the branches represent the decision rules, and each leaf node represents the result. In the decision tree, there are two nodes, namely the decision node and the leaf node. Decision nodes make any decision and have multiple branches, while leaf nodes have no output and other branches of those decisions the decision or test is made based on the characteristics of a given dataset.

The diagram below illustrates the general structure of the decision tree.

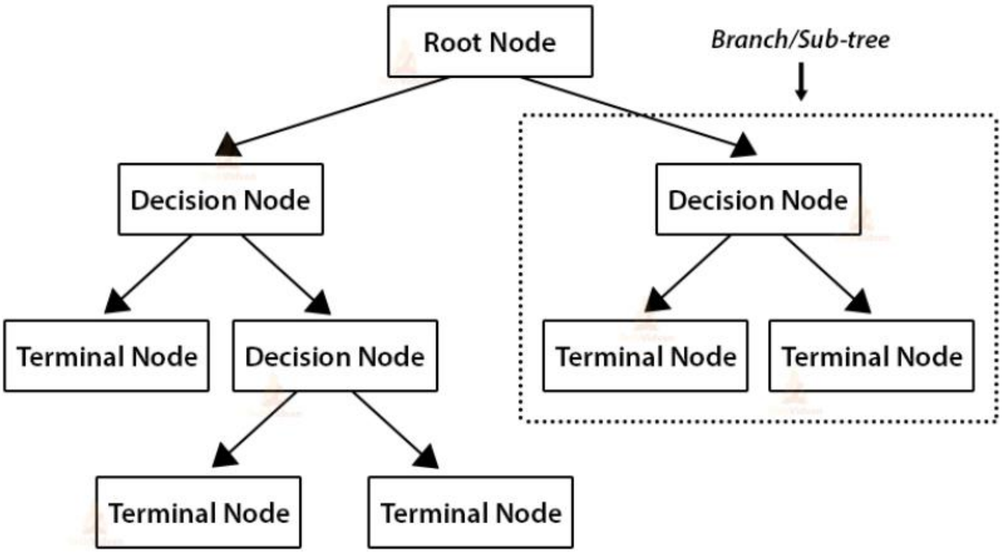


Figure 4.5.4: Structure of Decision tree algorithm

Graphical representation is the process of obtaining all possible solutions to a problem or decision based on the given circumstances. Like a tree it starts at the root node and then spreads to the branches to form a tree-like structure. To construct the tree, we use the cart algorithm, which is a classification and regression tree algorithm. The decision tree asks just one question and based on the answer (yes / no), it divides the tree into subtypes.

4.5.5 RANDOM FOREST (RF)

Random Forest is a popular algorithm for machine learning that is a supervised learning method. It can be used for both separation and Regression problems in ML. It is based on the concept of learning together, which is the process of combining multiple variables to solve a complex problem and improve model performance.

As the name suggests, Random Forest is a divider that contains a number of decision trees in the various subsets of a given database and takes a measure to improve the accuracy of the speculation of that database." Instead of relying on a single decision-making tree, the random forest takes predictions from each tree and relies on multiple predictable votes and predicts the final outcome.

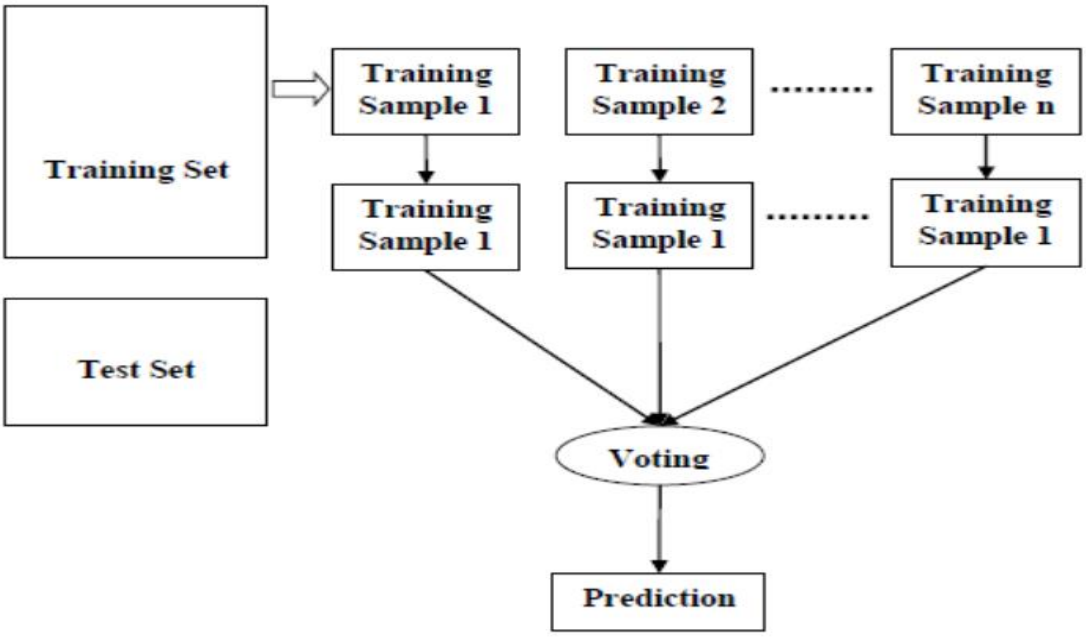


Figure 4.5.5 Architecture of Random forest (RF)

4.5.6 K-NEAREST NEIGHBORS

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

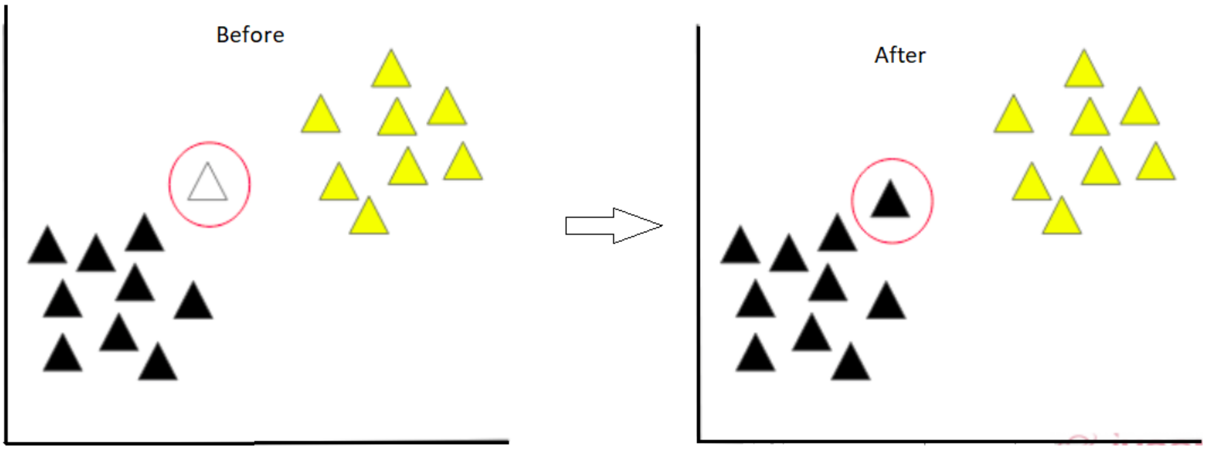


figure 4.5.6 KNN model graph

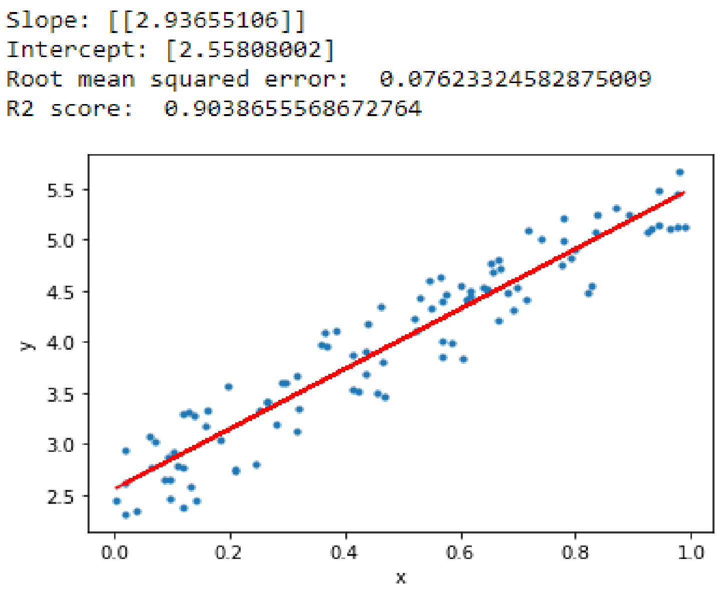
CHAPTER 5

IMPLEMENTATION AND RESULT ANALYSIS

5.1 ACCURACY OF TESTING RESULTS

5.1.1 LINEAR REGRESSION ALGORITHM

The algorithm gives an accuracy of 90.38% with intercept and slope for 2.9 and 2.5. This model works more accurate and gives small loss of data.



```
Slope: [[2.93655106]]
Intercept: [2.55808002]
Root mean squared error:  0.07623324582875009
R2 score:  0.9038655568672764
```

Figure 5.1.1: Linear regression algorithm Result

5.1.2 LOGISTIC REGRESSION ALGORITHM

This algorithm makes an accuracy of 86.54%.

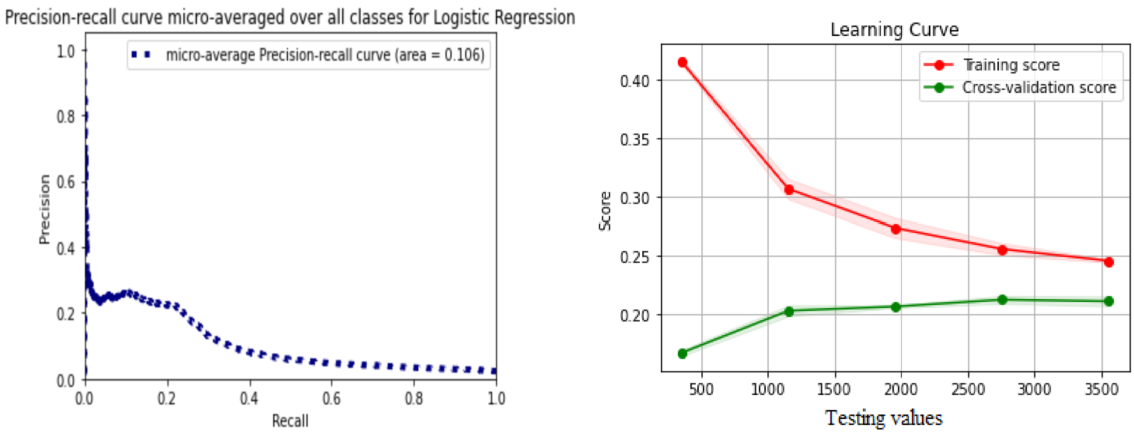


Figure 5.1.2: Results of Logistic Regression algorithm

5.1.3 DUMMY CLASSIFIER ALGORITHM

This algorithm gives an accuracy of 88.34%.

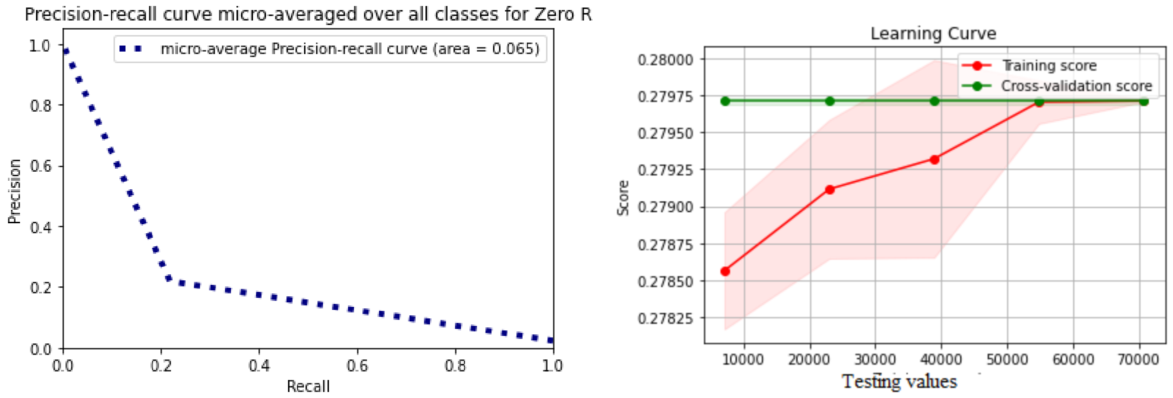
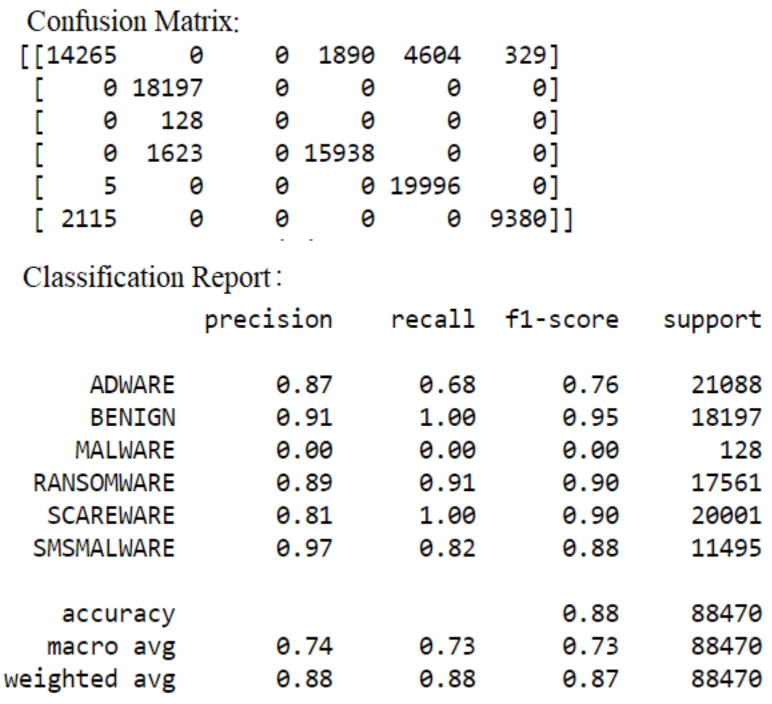


Figure 5.1.3: Results of Dummy classifier algorithm

5.1.4 DECISION TREE ALGORITHM

This algorithm works with an accuracy of 87.93% with best cross validation score of 90.42%. The following image shows the test result and its predicted output results.



Confusion Matrix:

```
[[14265     0     0  1890  4604   329]
 [    0 18197     0     0     0     0]
 [    0   128     0     0     0     0]
 [    0  1623     0 15938     0     0]
 [    5     0     0     0 19996     0]
 [ 2115     0     0     0     0  9380]]
```

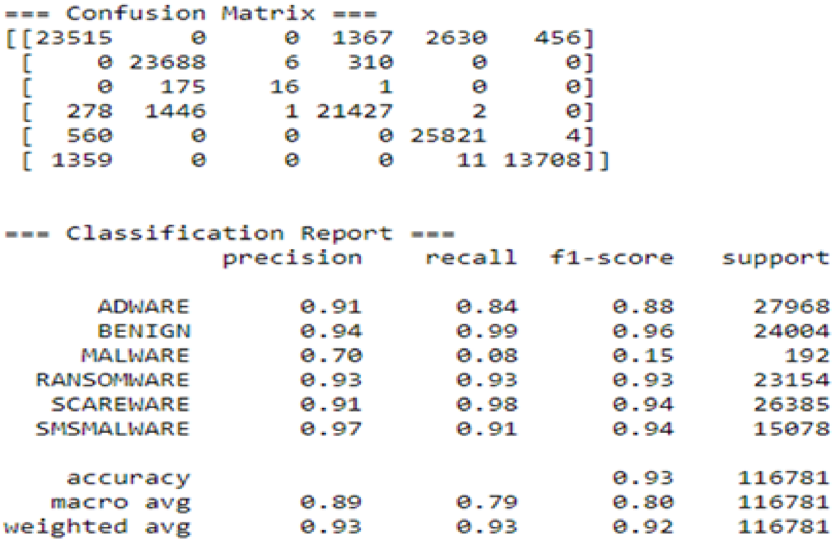
Classification Report :

```
              precision    recall  f1-score   support

     ADWARE        0.87      0.68      0.76     21088
     BENIGN        0.91      1.00      0.95     18197
    MALWARE        0.00      0.00      0.00       128
 RANSOMWARE        0.89      0.91      0.90     17561
  SCAREWARE        0.81      1.00      0.90     20001
 SMSMALWARE        0.97      0.82      0.88     11495

   accuracy                            0.88     88470
  macro avg        0.74      0.73      0.73     88470
weighted avg       0.88      0.88      0.87     88470
```

Figure 5.1.4: Confusion matrix result for Decision Tree Algorithm

5.1.5 RANDOM FOREST ALGORITHM

This algorithm gives an accruacy of 92.86% which is the higher accruacy of all the algorithm used. We also plot the confusion matrix which gives the separate result for each malware by f1-scores.

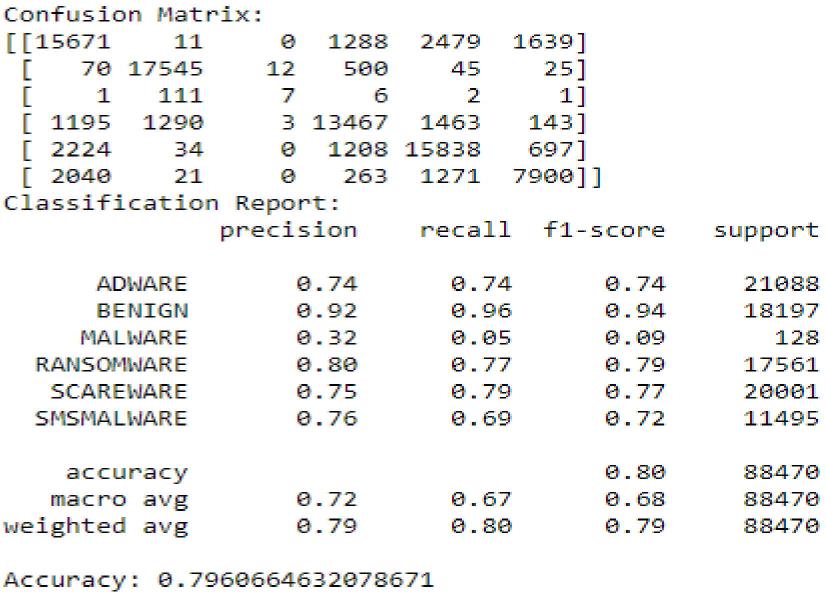


```
=== Confusion Matrix ===
[[23515     0     0  1367  2630   456]
 [    0 23688     6   310     0     0]
 [    0   175    16     1     0     0]
 [  278  1446     1 21427     2     0]
 [  560     0     0     0 25821     4]
 [ 1359     0     0     0    11 13708]]

=== Classification Report ===
              precision    recall  f1-score   support

      ADWARE       0.91      0.84      0.88     27968
      BENIGN       0.94      0.99      0.96     24004
     MALWARE       0.70      0.08      0.15       192
  RANSOMWARE       0.93      0.93      0.93     23154
   SCAREWARE       0.91      0.98      0.94     26385
  SMSMALWARE       0.97      0.91      0.94     15078

    accuracy                           0.93    116781
   macro avg       0.89      0.79      0.80    116781
weighted avg       0.93      0.93      0.92    116781
```

Figure 5.1.5 Result of Random forest algorithm

5.1.6 K – NEAREST NEIGHBOURS

This algorithm works on the principle of sk learn module by confusion matrix relations. It gives an accruacy of 79.45%.



```
Confusion Matrix:
[[15671    11     0  1288  2479  1639]
 [   70 17545    12   500    45    25]
 [    1   111     7     6     2     1]
 [ 1195  1290     3 13467  1463   143]
 [ 2224    34     0  1208 15838   697]
 [ 2040    21     0   263  1271  7900]]
Classification Report:
              precision    recall  f1-score   support

      ADWARE       0.74      0.74      0.74     21088
      BENIGN       0.92      0.96      0.94     18197
     MALWARE       0.32      0.05      0.09       128
  RANSOMWARE       0.80      0.77      0.79     17561
   SCAREWARE       0.75      0.79      0.77     20001
  SMSMALWARE       0.76      0.69      0.72     11495

    accuracy                           0.80     88470
   macro avg       0.72      0.67      0.68     88470
weighted avg       0.79      0.80      0.79     88470

Accuracy: 0.7960664632078671
```

Figure 5.1.6: Predicted output of KNN

5.2. OUTPUT ACCRUACY TABLE

This table shows that the relationship among each algorithm by the help of confusion matrix precision, recall and F1 scores. By comparing the F1 scores of Random forest, Decision Tree, Logistic Regression and KNN algorithms it gives an exact output of malware family which having higher count.

CLASSIFICATIONS REPORT												
MALWARES	**PRECISION**				**RECALL**				**F1-SCORES**			
Algorithms	DTC	RF	LR	KNN	DTC	RF	LR	KNN	DTC	RF	LR	KNN
Adware	0.87	0.91	0.57	.74	0.68	0.84	0.68	0.74	0.76	0.87	0.62	0.74
Benign	0.91	0.94	0.91	0.92	1.00	0.99	1.0	0.96	0.95	0.96	0.95	0.94
Ransomware	0.89	0.93	0.89	0.80	0.91	0.93	0.91	0.77	0.90	0.93	0.90	0.79
Scareware	0.81	0.91	0.81	0.75	1.0	0.97	1.0	0.79	0.90	0.94	0.90	0.77
SMS Malware	0.97	0.97	0.76	0.76	0.82	0.91	0.07	0.69	0.88	0.93	0.13	0.72
Accuracy									**0.88**	**0.92**	**0.78**	**0.80**
Micro Average	0.74	0.92	0.66	0.72	0.73	0.79	0.61	0.67	0.73	0.80	0.58	0.68
Weighted Average	0.88	0.92	0.78	0.79	0.88	0.92	0.78	0.80	0.87	0.92	0.74	0.79

figure 5.2.a: accuracy and f1 scores of algorithms

As the input testing and training sets were common to Random forest, Decision tree and K-Neighbors algorithm the predicted values during the valuating the dataset were given below the tables. The colored and highlighted values are the wrong predicted malwares from the dataset.

TESTING RESULTS	**PREDICTED RESULTS**		
	RANDOM FOREST	**DECISION TREE**	**KNN**
RANSOMWARE	RANSOMWARE	RANSOMWARE	**SCAREWARE**
ADWARE	ADWARE	ADWARE	ADWARE
ADWARE	ADWARE	ADWARE	ADWARE
ADWARE	ADWARE	ADWARE	ADWARE
ADWARE	ADWARE	ADWARE	ADWARE
SCAREWARE	SCAREWARE	SCAREWARE	SCAREWARE
BENIGN	BENIGN	BENIGN	BENIGN
RANSOMWARE	RANSOMWARE	RANSOMWARE	RANSOMWARE
ADWARE	ADWARE	ADWARE	ADWARE
ADWARE	ADWARE	ADWARE	ADWARE
RANSOMWARE	RANSOMWARE	**BENIGN**	**BENIGN**
ADWARE	ADWARE	ADWARE	ADWARE
BENIGN	BENIGN	BENIGN	BENIGN
RANSOMWARE	RANSOMWARE	RANSOMWARE	RANSOMWARE
SCAREWARE	SCAREWARE	SCAREWARE	SCAREWARE
ADWARE	ADWARE	ADWARE	ADWARE
BENIGN	BENIGN	BENIGN	BENIGN
ADWARE	**SCAREWARE**	SCAREWARE	**SCAREWARE**
BENIGN	BENIGN	BENIGN	BENIGN
ADWARE	ADWARE	**RANSOMWARE**	**RANSOMWARE**

Figure 5.2.b: Predicted output of each algorithm

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

Increasing traffic connections have also increased safety threats. Malicious code may flow into the car's internal network when a malicious code-infected device is connected to a car via an external communication channel. High accuracy and speed are key to discovery malicious behaviors in embedded motor situations, in which responses have to be considered in real time. This study, therefore, analyzed the security threats from adware and malware on the Android OS inside the self-driving car. Network mobility was analyzed to detect malicious patterns on the network in this module. In addition, a learning module for learning malware detection was suggested. Finally, we have developed a machine learning algorithm that can detect Android auto malware with high accuracy and short time. We compared algorithm availability and speed with the highest recommended parameters to the five machine learning algorithms.

6.2 FUTURE SCOPE

As the technology improving automatic driving cars paves a major role in transportation process. It is necessary to build a highly secured environment for the driving model in order to save from the crashes and steeling of vehicles. The machine learning provides a higher accuracy among others and the time for predicting the malware is reduced. Everyone having smartphones so we build a android malware detection system which checks the vehicle is attacked by malware or not. It also a simple technique which can be takeover by all the drivers for their safety purposes. Increasing the number of malware detection types by optimizing the algorithm for higher accuracy may give a substantial growth in future.

CHAPTER 7

REFERENCES

1. Mohit Sewak, Sanjay K Sahay, and Hemant Rathore, “Comparison of deep learning and the classical machine learning algorithm for the malware detection”, 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pages 293–296. IEEE, 2018
2. Seunghyun Park and Jin-Young Choi, “Malware Detection in Self-Driving Vehicles Using Machine Learning Algorithms”, Journal of Advanced Transportation Volume 2020, Article ID 3035741.
3. Daniel Gibert, Carles Mateu, Jordi Planes,” The rise of machine learning for detection and classification of malware: Research developments, trends and challenges”, Journal of Network and Computer Applications (153), 2020.
4. Sherif Saad, William Briguglio and Haytham Elmiligi “The Curious Case of Machine Learning In Malware Detection”, 5th International Conference on Information Systems Security and Privacy, 2019.
5. Azmoodeh, A., Dehghantanha, A., Conti, M., and Choo, K.-K. R. (2018). Detecting crypto-ransomware in iot networks based on energy consumption footprint. Journal of Ambient Intelligence and Humanized Computing, 9(4):1141–1152.
6. Choi, W., Joo, K., Jo, H. J., Park, M. C., and Lee, D. H. (2018). Voltageids: Low-level communication characteristics for automotive intrusion detection system. IEEE Transactions on Information Forensics and Security, 13(8):2114–2129.
7. Hassen, M., Carvalho, M. M., and Chan, P. K. (2017). Malware classification using static analysis-based features. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pages 1–7.
8. Naeem, H., Guo, B., and Naeem, M. R. (2018). A lightweight malware static visual analysis for iot infrastructure. In 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), pages 240– 244.
9. OSahn, D., Kural, O. E., Akleylek, S., and Kilic¸, E. (2018). New results on permission based static analysis for android malware. In 2018 6th International Symposium on Digital Forensic and Security (ISDFS), pages 1–4.
10. Selcuk, A. A., Orhan, F., and Batur, B. (2017). Undecidable problems in malware analysis. In 2017 12th International Conference for Internet Technology andSecured Transactions (ICITST), Pages 494 – 497.

11. Saurabh Jha, Timothy Tsai, Shengkun Cui, Subho S. Banerjee, "ML-driven Malware that Targets AV Safety", 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2020.
12. Balaji Baskaran and Anca Ralescu, "A Study of Android Malware Detection Techniques and Machine Learning", Modern AI and Cognitive Science Conference - MAICS 2016.
13. W. Wang, M. Zhu, X. Zeng, X. Ye, Y. Sheng, Malware traffic classification using convolutional neural network for representation learning, in: 2017 International Conference on Information Networking (ICOIN), IEEE, 2017, pp. 712–717.
14. Han, M.L.; Kwak, B.I.; Kim, H.K. Anomaly intrusion detection method for vehicular networks based on survival analysis. Veh. Commun. 2018, 14, 52–63.
15. Choi, W.; Joo, K.; Jo, H.J.; Park, M.C.; Lee, D.H. VoltageIDS: Low-level communication characteristics for automotive intrusion detection system. IEEE Trans. Inf. Forensics Secur. 2018, 13, 2114–2129.
16. Zhang, Y.; Chen, X.; Jin, L.; Wang, X.; Guo, D. Network Intrusion Detection: Based on Deep Hierarchical Network and Original Flow Data. IEEE Access 2019, 7, 37004–37016.
17. Song, H.M.; Woo, J.; Kim, H.K. In-vehicle network intrusion detection using deep convolutional neural network. Veh. Commun. 2020, 21, 100198.
18. . M. A. Alheeti, A. Gruebler and K. D. McDonald-Maier, "An Intrusion Detection System against Black Hole Attacks on the Communication Network of Self-Driving Cars," Proceedings of sixth International Conference on Emerging Security Technologies (EST), Braunschweig, pp. 86-91, 2015.
19. M. . R. Moore, R. A. Bridges, F. L. Combs, M. S. Starr, and S. J. Prowell, ''Modeling inter-signal arrival times for accurate detection of can bus signal injection attacks: A data-driven approach to in-vehicle intrusion detection,'' in Proc. 12th Annu. Conf. Cyber Inf. Secur. Res., 2017, p. 11.
20. A. Bezemskij, G. Loukas, D. Gan, and R. Anthony, ''Detecting cyberphysical threats in an autonomous robotic vehicle using Bayesian networks,'' in Proc. IEEE Cyber, Phys. Social Comput. (CPSCom), Aug. 2017, pp. 1–6.
21. K. M. A. Alheeti, A. Gruebler, and K. McDonald-Maier, ''Intelligent intrusion detection of grey hole and rushing attacks in self-driving vehicular networks," Computers, vol. 5, no. 3, p. 16, 2016.

22. T. P. Vuong, ‘‘Cyber-physical intrusion detection for robotic vehicles,’’ Ph.D. dissertation, Dept. Comput. Inf. Syst., Univ. Greenwich, London, U.K., 2017