ANDROID BOTNET DETECTION USING SUPERVISED MACHINE LEARNING METHODS

SHALINI .S

(20PCA016)

Project Report Submitted

In partial fulfillment of the requirements for the Award of

Master of Computer Applications

DEPARTMENT OF COMPUTER SCIENCE

AVINASHILINGAM INSTITUTE FOR HOME SCIENCE AND HIGHER EDUCATION FOR WOMEN

COIMBATORE – 641043

MAY – 2022

ANDROID BOTNET DETECTION USING SUPERVISED

MACHINE LEARNING METHODS

SHALINI .S

(20PCA016)

Project Report Submitted

In partial fulfillment of the requirements for the Award of

Master of Computer Applications

DEPARTMENT OF COMPUTER SCIENCE

AVINASHILINGAM INSTITUTE FOR HOME SCIENCE AND

HIGHER EDUCATION FOR WOMEN

COIMBATORE – 641043

MAY – 2022

Signature of the Head of the Department **Signature of the Supervisor**

Viva-voce Examination held on _____

Signature of the Examiner

CERTIFICATE

CERTIFICATE

Avinashilingam Institute for Home Science and Higher Education for Women

(Deemed to be University under Estd. u/s 3 of UGC Act 1956, Category 'A' by MHRD)
Re-accredited with 'A++' Grade by NAAC. CGPA 3.65/4, Category 1 University by UGC
Coimbatore - 641 043, Tamil Nadu, India

DST - CURIE - AI Sponsored
Centre for Cyber Intelligence

CERTIFICATE OF APPRECIATION

This is to certify that Ms. Shalini S (20PCA016) ,Master of Computer Applications, Avinashilingam Institute for Home Science and Higher Education for Women, has successfully completed the project entitled "Andriod Botnet Detection using Supervised Machine Learning Methods" under Centre for Cyber Intelligence - Centre for Machine Learning and Intelligence - a DST - CURIE - AI facility during December 2021 - May 2022.

Dr. G. Padmavathi
Dean, School of PSCS
CCI - Principal Investigator

Dr. P. Subashini
Project Coordinator - DST - CURIE - AI

Dr. S. Kowsalya
Registrar

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **God** Almighty, for his constant love and grace that he has shown upon me, which kept me in good health, and sound mind without which my project would not have reached a successful end.

I would like to express my deep sense of reverential gratitude and sincere thanks to **Dr.S.P.Thyagarajan,** Chancellor, Avinashilingam Institute of Home Science and Higher Education for Women, Coimbatore, for the opportunity given to me for undertaking this study and for providing all the needed facilities during my study.

I owe my great deal of gratitude to **Dr.V.Bharathi Harishankar, Ph.D., FRSA Vice-Chancellor,** Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for extending all resources that facilitated the conduct of the present study.

I express my gratitude to **Dr.S.Kowsalya, Registrar, M.Sc., M.Phil., Ph.D.** Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for providing all facilities necessary for the study.

I would express my boundless thanks to **Dr. G. Padmavathi, M.Sc., M.Phil., Ph.D., and Dean,** School of Physical Sciences & Computational Sciences and Principal Investigator for Center for Cyber Intelligence, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, for granting the facility required.

I wish to place on record my deep sense of gratitude to **Dr. Vasantha Kalyani David, M.Sc., M.Phil(Mathematics)., M.Phil(CS)., Ph.D., Professor and Head,** Department of Computer Science for support and encouragement to complete the project.

I am grateful to the project coordinator **Dr. (Mrs.) P.Subashini, MCA., M.Phil., M.Sc.(Applied Psychology)., Ph.D., Professor of Computer Science**, who was instrumental in granting me the facilities required for doing a project.

I express my heart full gratitude to my esteemed mentor **Dr. (Mrs.) V.Srividhya** M**.Sc., M.Phil., Ph.D., Assistant Professor (SG),** Department of Computer Science, for imparting the tremendous assistance and well-timed support for the triumph of our project with guidance and constant supervision as well as for providing necessary resources for the project and also for her support in completing the project.

I also like to extend my gratitude **to Ms. A. Roshni, M.Sc., Technical Assistant of CCI,** Department of Computer Science, project Guidelines CCI always supported me and nurtured me with valuable advice and profound belief in my work and abilities.

It was a great feeling to finally complete my project with the full support of the **Center for Cyber Intelligence**. Which I had been working on for so long. Thank you so much for always paying close attention to my work and recognizing my accomplishments.

Finally, yet importantly, I would like to thank my **parents, family members, and friends** for their kind inspiration, support, encouragement, blessings, and prayers, which were instrumental in the successful completion of the project.

I have great pleasure in expressing my deep sense of gratitude to all other teaching and non-teaching staff members of the Department of Computer Science, who stood behind the screen for the completion of the project.

I would extend my hearty thanks to one and all that helped me directly or indirectly with the successful completion of my project.

ABSTRACT

ABSTRACT

In today's world, a cyberattack is any offensive maneuver that targets computer information systems, computer networks, network infrastructures, and mobile devices. Cyber threats are increasing day by day, botnet is one of the major cyber attack that affect mobile devices. A Botnet is a network of a computer infected by malware that are under the control of a single attacking party, known as the "bot-herder". Each individual machine under the control of the bot-herder is known as a bot. According to the recent statistics report, in Quartely3 2021, Spamhaus Malware Labs identified 2,656 botnet C&Cs (command and controllers) compared to 1,462 in Quartely2 2021. This was an 82% increase quarter on quarter. The monthly average increased from 487 per month in Quartely2 to 885 botnet C&Cs per month in Quartely3. Quartely3 has seen a massive 82% rise in the number of new botnet C&Cs identified by the research team.

This project, investigates the detection of Android Botnet using Machine learning techniques. Machine learning is a branch of Artificial Intelligence (AI) to provide automatic detection of botnet attacks without any human intervention. The process of Android Botnet Detection using Machine Learning methods consists of six phases. The Phase 1 is the data acquisition. In Phase 2, deals with data pre-processsing to remove unrelevant data. In Phase 3, the appropriate wrapper based feature selection methods are used to select the significant features. In Phase 4, deals with model building using supervised machine learning methods includes Random Forest, Naive Bayes, Decision Tree, Multi- layer Perceptron and Support Vector Machine. In Phase 5, the comparative analysis is made between the supervised machine learning models to suggest the suitable model for Android Botnet detection. The Evaluation of the models based on the performance metrics such as accuracy, precision, recall, f1 score in a significant way.

Keywords: **Android Botnet, Evaluation Metrics, Malware, Regression Error Metrics, Supervised Machine Learning.**

CONTENT

TABLE OF CONTENT

LIST OF FIGURES

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE PROJECT

A botnet is a collection of connected devices that individually operate one or more bots. Botnets can be used to launch DDoS assaults, steal data, send spam, and give the attacker access to the device and its connection. The botnet's owner can use command and control (C&C) software to manage it. The term "botnet" is a combination of "robot" and "network." Typically, the phrase has a nasty or hostile connotation. A botnet attack is a sort of cyberattack in which malware infects a huge number of internet-connected computers that are then controlled by a malicious hacker. Botnet assaults include spamming, data theft, stealing sensitive information, and launching unpleasant DDoS attacks.

A botnet is a logical collection of Internet-connected devices, such as computers, cellphones, or Internet of things (IoT) devices, whose security has been compromised and control has been relinquished to a third party. When software from a malware (malicious software) distribution infiltrates a device, it creates a compromised device known as a "bot." Through communication channels built by standards-based network protocols like IRC and Hypertext Transfer Protocol, the botnet's controller can command the operations of these infected computers (HTTP).

An Android botnet is one of the most dangerous types of malware attack even though it can be controlled remotelyby a botmaster and used to carry out damaging attacks. Android Botnet is similar to SPYWARE in that it allows attackers to gain complete control of a device after it has been installed. Matryosh is a botnet that targets Operating system that have exposed the Android Debug Bridge (ADB) debug interface to the internet. Matryosh, according to Connect different, is an ADB-targeting malware that hides it command and control nodes behind the Tor network. Several researchers have employed a variety of well-known Machine Learning (ML) techniques to distinguish Android botnet from benign application.

Android Botnet applications are malware that can be delivered through these marketplaces and downloaded by unsuspecting consumers onto their smartphones. Android botnets by using static code analysis to extract features from the source code after it has been reverse-engineered.

In Quartely3, Spamhaus Malware Labs research team discovered an increase of 82% of the total of new botnet commands and controllers (C&Cs). And it witnessed a rise in the deployment of backdoor viruses, with crooks using FastFlux as a cover. As a result, we've added several countries and internet providers to the Top 20 lists.

The Quartely3 2021 Spamhaus Botnet Attack Report is now available. In the third quarter of 2021, Section contains Software Labs found 2,656 botnet C&Cs (command and controllers), up from 1,462 in the previous quarter. It represents an 82% gain over the preceding quarter. The current average improved from 487 malware C&Cs in Quarter2 to 885 malware C&Cs in Quartely3.

1.2 MOTIVATION AND JUSTIFICATION

A botnet can be remotely controlled by attackers though malicious or illegal purposes, which can have a direct and indirect impact on users. Classifying the Android Botnet helps in detecting the attacks that are trying to access the user control.

1.3 PROBLEM STATEMENT

To identify fraudulent Android applications or malware attacks that are attempting to acquire sensitive data or get access to a user's device that is connected through a network.

1.4 OBJECTIVE

To develop a Supervised Machine learning models to detect and classify the Android Botnet attack that are trying to gain the access user control.

SYSTEM CONFIGURATION

CHAPTER – 2

SYSTEM CONFIGURATION

2.1. HARDWARE REQUIREMENT

PROCESSOR	: Intel i7 and above
RAM	: 8 GB
HARD DISK CAPACITY	: 1TB

2.2. SOFTWARE REQUIREMENT

OPERATING SYSTEM	: Windows10
PACKAGE	: Anaconda
FRONT END	: Python

2.3. ABOUT THE PLATFORM

2.3.1 ANACONDA

Anaconda is a Python and R programming language distribution aimed for simplifying package management and deployment in programming (data science, machine learning applications, large-scale data processing, predictive analytics, and so on). Data-science packages for Windows, Linux, and macOS are included in the release. Anaconda, Inc., created by Peter Wang and Travis Oliphant in 2012, is responsible for its development and maintenance. Anaconda Distribution or Anaconda Individual Edition are other Anaconda, Inc. goods, whereas Anaconda Team Edition and Anaconda Enterprise Edition, both are not free, are other Anaconda, Inc. products.

It is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage anconda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud

or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, mac OS and Linux.

The following applications are available in Anaconda Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code

2.3.2 JUPYTER NOTEBOOK

Jupyter Notebook is a popular programming environment an open source, interactive web tool that lets user create and share documents with interactive calculations, code, and graphics, among other things. Users can create interactive "story" that can edit and share by combining data, code, and visuals into a single notebook. Notebooks are documents that include both computer code(such as Python) and other text elements including paragraphs, markdown, figures, and links, among other things.

Jupyter is a programming notebook is well-known and well-documented, with a simple interface for generating, editing, and executing notebooks. The notebook is accessed through a web application known as the "Dashboard" or "Control panel", which displays local files and allows users to view notebook pages and run code snippets. The results are well formatted and presented in the browser.

The kernel is the notebook's other component. The kernel is a types of "computational engine" that runs the code in the notebook. It's similar to the application's backend. Jupyter notebook, the IPython kernel (jupyter was originally known as IPython notebook) is utilized to run Python code.

2.3.3 PYTHON PACKAGE

Common Importing Python package using Jupyter Notebook are:

The use of Jupyter Notebooks to import code is a common problem.

a) import pandas as pd

Pandas is typically installed using the pd alias. alias: In Python, an alias is a different name for the same item. Instead of pandas, the Pandas package is now known as pd.

b) import numpy as np

Python is told to introduce the NumPy library to the current environment via the import numpy section of the code. The as np subsection then instructs Python to assign the alias np to NumPy. Instead of typing numpy, it can use NumPy functions besides typing np. function name.

c) import matplotlib.pyplot as plt

Matplotlib is a Python package that allows it to create static, animated, and interactive visualisations. Matplotlib makes simple things simple and difficult things possible that can be Create plots that are suitable for publication, Customize the visual style and layout of dynamic figures which can zoom, pan, and update.

d) from sklearn.metrics

Metrics for classification to measure classification performance, the sklearn. metrics module implements several loss, score, and functionalities. Some metrics may demand positive class probability estimations, confidence variables, or binary decision values.

e) from sklearn.model_selection import train_test_split

Train test split is a Sklearn model selection method that divides information arrays into two subsets: test and train information. With this function, there should be no need to manually divide the dataset. By default, the Sklearn train test split generates random differences for the two subsets.

f) from sklearn.metrics import preprocessing

sklearn. preprocessing package contains several common transfer functions and transformer classes for converting the raw relevant features into an interpretation that is more suitable for downstream estimators. In general, the standardization of the set of data benefits supervised learning.

g) from sklearn.preprocessing import MinMaxScaler

sklearn.preprocessing.MinMaxScaler class sklearn.preprocessing. MinMaxScaler (feature range=0, 1) (copy=True) (feature [source] Scales each attribute to a set range to transform it. This estimate scales and translations each feature separately so that it falls inside the training set's provided range, i.e. among zero and one.

h) from sklearn.preprocessing import standardscaler

Instead of removing the mean and scaling to unit variance, StandardScaler standardises a feature. Divide all of the numbers by the standard deviation to get unit variance. The formal concept of scale that I introduced earlier does not apply to StandardScaler.

i) from sklearn import metrics

To measure classification performance, the sklearn. metrics module implements several loss, score, and utility functions. Some metrics may demand positive class probability estimations, confidence values, or binary decision values.

j) from sklearn.metrics import classification_report

In machine learning, a classification report is a performance evaluation indicator. It's used to demonstrate trained classification model's precision, recall, F1 Score, and support.

k) from sklearn.metrics import accuracy_score

This method computes subset accuracy in multilabel classification: any subset of label forecast for a sample must match exactly the corresponding set of keywords in y true. More information can be found in the User Manual. Labels that are true to the ground (accurate).

l) from sklearn.metrics import precision_score

sklearn.metrics.precision score. Precision is defined as tp / (tp + fp), in which tp is the number of actual true positives and fp seems to be the proportion of false positives. Precision is intuitively defined as the classifier's ability to avoid labeling something negative sample as positive.

m) from sklearn.metrics import recall_score

sklearn.metrics.recall score is used to calculate recall is defined as the proportion tp / (tp + fn), where tp represents the number of true positive rate and fn represents the number of false negatives rate. The recall is implicitly the classifier's ability to find all positive samples. The best possible value 1 but worst possible value is 0.

n) from sklearn.metrics import f1_score

Using the f1 score framework, we must first import the package listed below. F1 score is a component of the sklearn.metrics package. Here is the full syntax for the F1 score function. The required parameters are y true and y pred. Others are optional parameters that are not required.

REVIEW OF LITERATURE

CHAPTER 3

REVIEW OF LITERATURE

Table 3.1 Review of datasets and techniques used for analysis

S.NO	TITLE OF THE PAPER	AUTHOR & YEAR	ALGORITHM USED	DATASET USED	RESULT
1	Android Botnet Detection using machine learning models based on a comprehensive static analysis approach.	Wadi' Hijawi, Ja'far Alqarawna, Ala' M.Al Zoubi, Mohammad A.Hassonah (2021).	Random Forest, Multi-Layer Perceptron, Decision trees, Naive Bayes.	ISCX dataset	Random Forest 98% - Accuracy
2	Mobile Botnet Detection: A Deep Learning ApproachUsing Convolutional Neural Networks	Suleiman Y. Yerima and Mohammed K. Alzaylaee. (2021)	CNN Classifier	ISCX botnet dataset.	SVM 98.9% - accuracy
3	Android Botnet Detection Using Machine Learning .	Mohammad M. Rasheed, Alaa K. Faieq, Ahmed A. Hashim (2020)	Sequential minimal optimization, Random Tree, J48, Naïve Bayes and Logistic algorithms	ISCX dataset	Random Forest 85% - Accuracy

4	Botnet Detection using Machine Learning.	Shamsul Haq, Yashwant Singh (2020)	C4.5, REP Tree,Support vector machine , K-means	ISOT, ISCX, and CTU-13	C4.5 98.7% - Accuracy
5	A Study of Machine Learning Classifiers for Anomaly -Based Mobile Botnet Detection.	Ali Feizollah, Nor Badrul Anuar, Rosli Salleh, Fairuz Amalina, (2020)	Naïve Bayes, k-nn, decision tree, multi-layer perceptron, and support vector machine	MalGenome data	SVM 99.94% - Accuracy
6	A Static Approach towards Mobile Botnet Detection	Shahid Anwa, Jasni Mohamad Zain, Ahmad Karim, Zakira Inayat. (2020)	SVM, KNN, J48, Bagging, Naïve Bayes, and Random Forest.	UNB ISCX Android botnet dataset	Random Forest 93% - Accuracy
7	Detecting Botnet Traffic by using Machine learning	Pallavi, Vardhamane (2020)	SVM, Random Forest, Naive Bayes and Linear Regression algorithms	CTU-13 datasets	Random Forest 98.8% accuracy
8	Machine Learning Based Botnet Detection	Vaibhav Nivargi, Mayukh Bhaowal, Teddy Lee (2020)	Multinomial Naïve Bayes, linear SVM, kNN, Logistic Regression, Multiboost Adaboost	CTU-13 datasets	SVM 97.28% accuracy

9	Network Traffic Based Botnet Detection Using Machine Learning	Anand Ravindra Vishwakarma (2020)	Random forest, Decision tree, AdaBoost and XGBoost.	CTU-13 Dataset	Random Forest 86% - accuracy
10	Structural analysis and detection of android botnets using machine learning techniques.	G. Kirubavathi, R. Anitha.(2018)	Support Vector Machine, Naive Bayes, REP Tree	Android Malware Gnome project, Drebin, DroidAnalytics, ISCX.	SVM 92.10% - Accuracy
11	An Enhanced Android Botnet Detection Approach using Feature Refinement.	Shahid Anwar (2017)	Simple Logistic Regression,SVM, KNN, J48, Bagging, Naïve Bayes, and Random Forest	CTU-13 datasets	Random Forest 97.28% accuracy

Table description:

The above table 3.1 provides the details of dataset reviews and techniques used for Android botnet in Machine Learning classifier.

METHODOLOGY

CHAPTER 4

METHODOLOGY

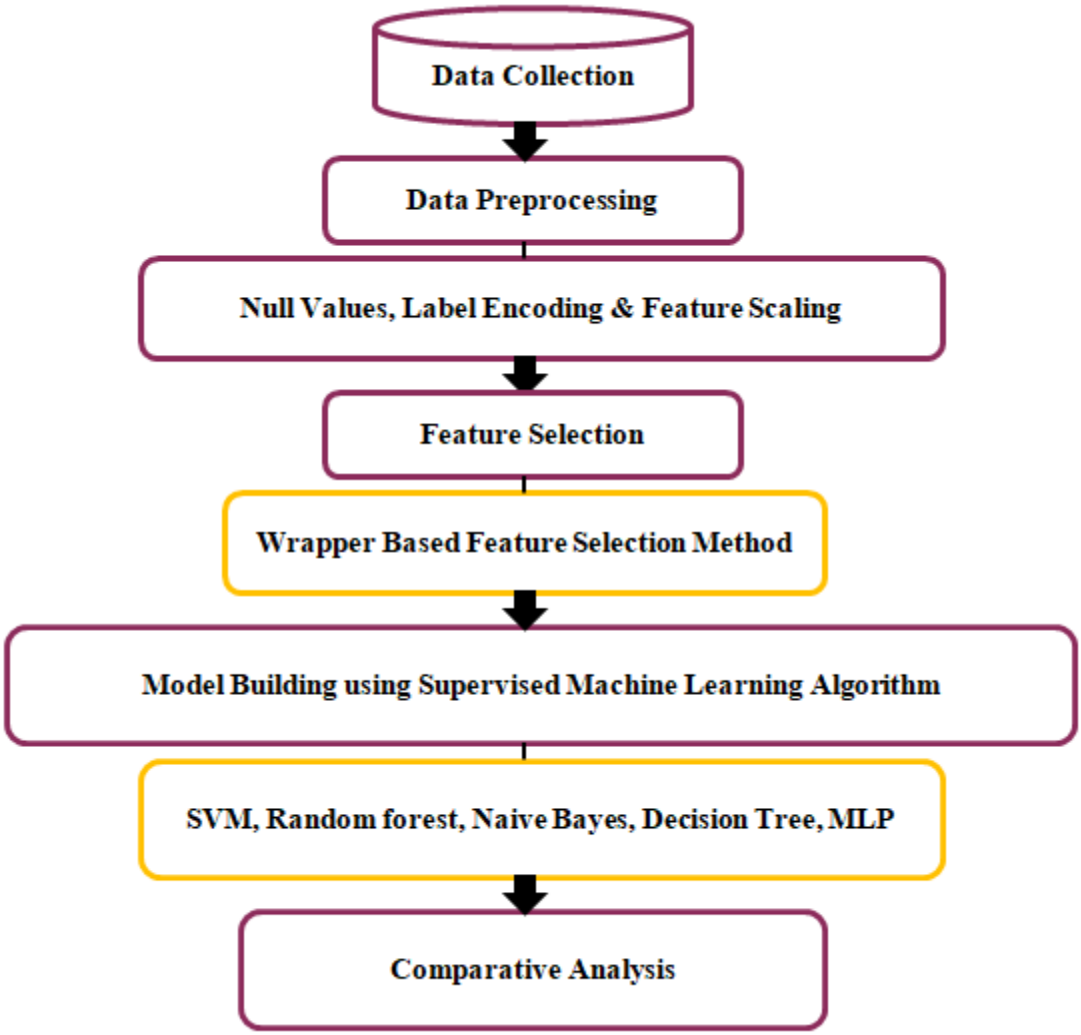


Figure 4.1 Methodology Overview

The following figure represemts, Android Botnet entire methodology is divded into five phases namely, Data collection, Data Pre-processing, Wrapper based Feature Selection method, Model Building using Supervised Machine learning algorithms and Comparative analysis based on Performance Evaluation Metrics.

4.1 PHASE 1: DATA COLLECTION

The term "Data collection" refers to the collection process data from relevant sources before it is stored, cleaned, preprocessed, and used in other methods. The process of acquiring and analysing data from a variety of sources is known as data collection, translating it into the relevant data form, and loading it into the efficient machine learning method.

Android Botnet ISCX Dataset: This dataset was created to train and evaluate machine learning models using two different applications: Android Botnet Application and Andoid Botnet Application. The ISCX Android Botnet dataset contains a huge number of Android botnets from 14 different botnet families. This dataset was created using 6802 Android applications, which included 4873 benign clean apps and 1929 botnet apps from the ISCX botnet dataset. This dataset contains 342 static features extracted from application files. These files extract features from API calls, Permissions, Extra files, Commands, and Intents, among other categories.

4.1.1 DATASET DESCRIPTION

The following table 4.1.1 consists of the Attribute, Attribute description of dataset used in ISCX dataset.

Table 4.1.1 Attribute and Attribute description of ISCX dataset

S.NO	ATTRIBUTE NAME	DESCRIPTION
1.	ACCESS_CHECKIN_PROPERTIES	Enables networks to the reservation database's "properties" table, allowing to edit the values that seem to be uploaded.
2.	ACCESS_COARSE_LOCATION	Enables an application to get an estimate of its location from web access sources like telephone lines and Wi-Fi.
3.	ACCESS_FINE_LOCATION	Enables an application to obtain exact position information from sources including Geolocation, communications systems, and Area network.

S.NO	ATTRIBUTE NAME	DESCRIPTION
4.	ACCESS_LOCATION_EXTRA_COMMANDS	Enables a programme to use additional location provider commands.
5.	ACCESS_MOCK_LOCATION	Enables applications to generate and test mock location providers.
6.	ACCESS_NETWORK_STATE	Enables applications to access network information.
7.	ACCESS_SURFACE_FLINGER	Enables applications to take advantage of SurfaceFlinger's low-level capabilities.
8.	ACCESS_WIFI_STATE	Enables Wi-Fi network information to be accessed by programmes.
9.	ACCOUNT_MANAGER	Asking permission to communicate with Account Authenticators.
10.	ADD_VOICEMAIL	Enables a programme to enter voicemails into the system.
11.	AUTHENTICATE_ACCOUNTS	Enables a programme to interact with the AccountManager as an AccountAuthenticator.
12.	BATTERY_STATS	Enables a programme to gather battery statistics.
13.	BIND_ACCESSIBILITY_SERVICE	To guarantee that only the computer can bind to it, an AccessibilityService must require it.
14.	BIND_APPWIDGET	Enables an app that shows the AppWidget provider which applications are allowed to use AppWidget's data.
15.	BIND_DEVICE_ADMIN	To guarantee that only the system may communicate with the device administration receiver, it must be necessary.
16.	BIND_INPUT_METHOD	To guarantee which only the computer can bind to an InputMethodService, it must be required.

S.NO	ATTRIBUTE NAME	DESCRIPTION
17.	BIND_REMOTEVIEWS	A RemoteViewsService must require this to guarantee which only the computer can bind to it.
18.	BIND_TEXT_SERVICE	A TextService must require it.
19.	BIND_VPN_SERVICE	An VpnService must require this to guarantee which only the computer can connect to it.
20.	BIND_WALLPAPER	A WallpaperService must require this to guarantee which only the computer can bind to it..
21.	BLUETOOTH	Asking permission to connect to Bluetooth devices that have been paired.
22.	BLUETOOTH_ADMIN	Enables Bluetooth devices to be discovered and paired by applications.
23.	BRICK	It is necessary to actually to turn off the device (this is really harmful!).
24.	BROADCAST_PACKAGE_REMOVED	Enables applications to transmit a notification about the removal of an application package.
25.	BROADCAST_SMS	Enables applications to send an SMS receipt notice to multiple users.
26.	BROADCAST_STICKY	Enables a programme to send out sticky intents.
27.	BROADCAST_WAP_PUSH	Enables a Gprs PUSH receipt notice to be broadcast by an application.
28.	CALL_PHONE	Enables applications to make a phone call without requiring to confirm the call by using the Dialer user interface.

S.NO	ATTRIBUTE NAME	DESCRIPTION
29.	CALL_PRIVILEGED	Enables applications to contact any contact details, even emergency numbers, without requiring the user to verify the call by using the Dialer user interface.
30.	CAMERA	To be able to use the camera device, must be able to view it.
31.	CHANGE_COMPONENT_ENABLED_STATE	Enables applications to control whether or not an external application component is enabled.
32.	CHANGE_CONFIGURATION	Enables a programme to make changes to the existing configuration, such as the locale.
33.	CHANGE_NETWORK_STATE	Enables developers to modify the state of their network connectivity.
34.	CHANGE_WIFI_MULTICAST_STATE	Enables Wi-Fi Multicast mode to be used by applications.
35.	CHANGE_WIFI_STATE	Enables developers to modify the state of Wi-Fi connectivity.
36.	CLEAR_APP_CACHE	Enables applications to delete the cache of all the device's installed apps.
37.	CLEAR_APP_USER_DATA	Enables a programme to delete user data.
38.	CONTROL_LOCATION_UPDATES	Enables to enable/disable radio location update notifications.
39.	DELETE_CACHE_FILES	Enables a programme to erase its cache files.
40.	DELETE_PACKAGES	Enables a programme to remove packages.
41.	DEVICE_POWER	Enables access to the power management at a low level.

S.NO	ATTRIBUTE NAME	DESCRIPTION
42.	DIAGNOSTIC	Enables diagnostic resources to be RW by programmes.
43.	DISABLE_KEYGUARD	The keyguard can be disabled by applications.
44.	DUMP	Enables applications to query system services for state dump information.
45.	EXPAND_STATUS_BAR	The status bar can be expanded or collapsed by an application.
46.	FACTORY_TEST	As the root user, execute as a manufacture test programme.
47.	FLASHLIGHT	Enables to use the flashlight.
48.	FORCE_BACK	Enables a programme to perform a BACK action on the currently active activity.
49.	GET_ACCOUNTS	Provides access to the Accounts Service's list of accounts.
50.	GET_PACKAGE_SIZE	Enables a programme to determine how much space each package takes up.
51.	GET_TASKS	Enables a programme to acquire information about tasks that are currently or recently running.
52.	GLOBAL_SEARCH	This authorization can be used by content providers to grant access to their data to the global search system.
53.	HARDWARE_TEST	Entrance to hardware peripherals is possible.
54.	INJECT_EVENTS	Enables a programme to insert input from the user) into the events pipeline and send them to just about any window.
55.	INSTALL_LOCATION_PROVIDER	Enables a location provider to be installed in the Location Manager by an application.

S.NO	ATTRIBUTE NAME	DESCRIPTION
56.	INSTALL_PACKAGES	Enables a programme to download and install packages.
57.	INTERNAL_SYSTEM_WINDOW	Enables applications to open windows for usage by system user interface components.
58.	INTERNET	Opens network sockets for applications.
59.	KILL_BACKGROUND_PROCESSES	KillBackgroundProcesses is a method that an application can use to kill background processes (String).
60.	MANAGE_ACCOUNTS	Enables applications to control the AccountManager's account list.
61.	MANAGE_APP_TOKENS	Enables applications to manage program tokens in the windows manager (create, destroy, and Z-order).
62.	MASTER_CLEAR	Enables Clean Master for Mobile to be a junk file remover as well as a computer optimizer.
63.	MODIFY_AUDIO_SETTINGS	Enables a programme to change global audio settings.
64.	MODIFY_PHONE_STATE	Enables a programme to change global audio settings.
65.	MOUNT_FORMAT_FILESYSTEMS	Enables file systems to be formatted for removable storage.
66.	MOUNT_UNMOUNT_FILESYSTEMS	Enables for the mounting and unmounting of removable storage file systems.
67.	NFC	Enables developers to use NFC to conduct I/O operations.
68.	PERSISTENT_ACTIVITY	Persistent activity is frequently observed in cortical regions that are engaged in more than merely information maintenance.

S.NO	ATTRIBUTE NAME	DESCRIPTION
69.	PROCESS_OUTGOING_CALLS	Enables applications to keep track of, change, or terminate incoming calls.
70.	READ_CALENDAR	Enables applications to read information from the user's calendar.
71.	READ_CALL_LOG	Enables a programme to access the user's call history.
72.	READ_CONTACTS	Enables a programme to read a user's contact information.
73.	READ_EXTERNAL_STORAGE	This property enables a program to read data from external storage.
74.	READ_FRAME_BUFFER	Enables applications to snap screen shots and access the contribute up data in general.
75.	READ_HISTORY_BOOKMARKS	Enables a programme to access (but not write) a user's bookmarks and browsing history.
76.	READ_INPUT_STATE	The current status of buttons and switches can be retrieved by an application.
77.	READ_LOGS	Enables a programme to read the system's low-level log files.
78.	READ_PHONE_STATE	Enables only read-only access to the phone's current state.
79.	READ_PROFILE	Enables a person's private profile data to be viewed by an application.
80.	READ_SMS	This property enables a program to receive SMS messages.
81.	READ_SOCIAL_STREAM	Enables a user's social stream to be read by an application.
82.	READ_SYNC_SETTINGS	Asking permission to access the sync preferences.

S.NO	ATTRIBUTE NAME	DESCRIPTION
83.	READ_SYNC_STATS	Asking permission to read sync statistics.
84.	READ_USER_DICTIONARY	Enables a programme to read the dictionary of the user.
85.	REBOOT	The ability to reboot the gadget is required.
86.	RECEIVE_BOOT_COMPLETED	Enables applications to hear the ActionBootCompleted signal, which is sent when the system has completed booting.
87.	RECEIVE_MMS	Enables applications to keep track of incoming MMS messages, record them, and process them.
88.	RECEIVE_SMS	Enables applications to keep track of incoming SMS messages, record them, and process them.
89.	RECEIVE_WAP_PUSH	Enables a programme to keep track of receiving WAP push messages.
90.	RECORD_AUDIO	Enables applications to make audio recordings.
91.	REORDER_TASKS	Enables applications to reorder tasks in the Z-order.
92.	RESTART_PACKAGES	Enables a programme to reload the packages.
93.	SEND_SMS	This property enables a program to send Calls and texts.
94.	SET_ACTIVITY_WATCHER	Enables a programme to monitor and regulate how activities are launched across the entire system.
95.	SET_ALARM	Enables applications to send out an Intent to the user to set an alarm.
96.	SET_ALWAYS_FINISH	Enables applications to control whether or not background actions are completed promptly.

S.NO	ATTRIBUTE NAME	DESCRIPTION
97.	SET_ANIMATION_SCALE	Change the animation scaling factor globally.
98.	SET_DEBUG_APP	Enable Set up a debugging environment for an application.
99.	SET_ORIENTATION	Enables low access to changing the screen's orientation.
100	SET_POINTER_SPEED	Enables for low-level control of the pointer speed.
101	SET_PREFERRED_APPLICATIONS	To establish itself as the Preferred HomeScreen, the permission SET PREFERRED APPLICATIONS is required.
102	SET_PROCESS_LIMIT	Enables applications to specify the highest limit of (non-essential) processing that can be active at the same time.
103	SET_TIME	Asking permission to change the time on the system.
104	SET_TIME_ZONE	The network time zone can be set by programmes.
105	SET_WALLPAPER	Asking permission to change the wallpaper.
106	SET_WALLPAPER_HINTS	Asking permission to customize wallpaper hints.
107	SIGNAL_PERSISTENT_PROCESSES	Enable an application to send a signal to all permanent processes at once.
108	STATUS_BAR	The over one and its icons can be opened, closed, or disabled by an application.
109	SUBSCRIBED_FEEDS_READ	Enables applications to access the ContentProvider's subscribed feeds.
110	SUBSCRIBED_FEEDS_WRITE	Enables applications to write Developer tools / private details to subscribed feeds. android.permission is a URI permissions.

S.NO	ATTRIBUTE NAME	DESCRIPTION
111	SYSTEM_ALERT_WINDOW	Enables applications to open windows of type SystemAlert that are shown above all other apps.
112	UPDATE_DEVICE_STATS	Enables a programme to update the device's statistics.
113	USE_CREDENTIALS	Enables applications to ask the AccountManager for authtokens.
114	USE_SIP	Enables a programme to use the SIP service.
115	VIBRATE	Gives an access to the vibrator.
116	WAKE_LOCK	Enables to use PowerManager WakeLocks to prevent the processor from going to sleep or the screen from dimming.
117	WRITE_APN_SETTINGS	Asking permission to write their own apn settings.
118	WRITE_CALENDAR	Enables applications to write to the user's calender data .
119	WRITE_CALL_LOG	Enables a programme to submit (but not access) a user's contact information.
120	WRITE_CONTACTS	Enables a programme to submit (but not read) a user's contact information.
121	WRITE_EXTERNAL_STORAGE	Enables a programme to save data to external storage.
122	WRITE_GSERVICES	Enables a programme to make changes to the Google services map.
123	WRITE_HISTORY_BOOKMARKS	Enables a programme to write the user's browser history and bookmarks (but not read them).
124	WRITE_PROFILE	Enables applications to write to the person's private profile data (but not read it).

S.NO	ATTRIBUTE NAME	DESCRIPTION
125	WRITE_SECURE_SETTINGS	Enables a programme to access the secured system settings and read or write them.
126	WRITE_SETTINGS	Enables a programme to read and write system settings.
127	WRITE_SMS	Enables a programme to send SMS messages.
128	WRITE_SOCIAL_STREAM	Enables an app to submit data from the user's social stream.
129	WRITE_SYNC_SETTINGS	Enables sync settings to be written by programmes.
130	WRITE_USER_DICTIONARY	Enables a programme to update the user dictionary.
131	intent.action.RUN	An intent allows to start an activity in another app by describing a simple action
132	NEW_OUTGOING_CALL	Provides access to the NEW OUTGOING CALL action for outgoing calls.intent android.content.Intent
133	USER_PRESENT	Android.content.intent.ACTION USER PRESENT and android.content.intent.ACTION BOOT COMPLETED are Java constants.
134	SMS_RECEIVED	"android.provider.Telephony.SMS RECEIVED" is the purpose for incoming SMS. The intent filter's priority can be changed, but it isn't required.
135	PACKAGE_REPLACED	MY PACKAGE REPLACED For testing purposes, it may also use the adb shell to manually activate the receiver:am broadcast adb shell.
136	PACKAGE_INSTALL	This is a platform-independent Android installer app that allows users to build Android applications directly to the computer.

S.NO	ATTRIBUTE NAME	DESCRIPTION
137	ACTION_MAIN	The major entry point of the programme is the Enable authorization to access MAIN action.
138	SEND_MULTIPLE	In Android, there are a variety of ways to transport numerous data from one application to another, but in this session, we'll use Bundle. In Android, a bundle is used to transfer information.
139	settings.APN_SETTINGS	To get online, set the APN to truphone.com and leave the rest of the settings alone (password, etc.).
140	wifi.WIFI_STATE_CHANGED	The android.net.wifi.STATE CHANGE activity receives a broadcasting on disconnection only if the smartphone is disconnected from a network
141	PICK_WIFI_WORK	Wi-Fi on Android allows programmes to view the status of wireless connections at a very low level of access.
142	PHONE_STATE	Enable an application to READ Device Status and display Dialog with the present state when the device state changes.
143	WAP_PUSH_RECEIVED	In Android, the Enables permission is used to receive WAP push messages.
144	CONNECTIVITY_CHANGE	Enable running apps to listen for CONNECTIVITY CHANGE on their main post if they use a Broadcast Receiver to request notice.
145	UMS_CONNECTED	Enable apps to use UMS CONNECTED, which is deprecated in current Android versions. The other issue is that it does not function with MTP supported devices.

S.NO	ATTRIBUTE NAME	DESCRIPTION
146	UMS_DISCONNECTED	Enable apps to use Java source for android.content.Intent.ACTION_UMS_DISCONNECTED
147	BATTERY_LOW	The system conversation "Low battery warning" corresponds to this communication.
148	BATTERY_OKAY	Following ACTION BATTERY LOW, this will be provided once the batteries have recovered from their low status.
149	BATTERY_CHANGED_ACTION	This is a sticky broadcast that contains information about the battery's charge state, level, and other features.
150	INPUT_METHOD_CHANGED	An input method has been changed.
151	SIG_STR	A signature setup consists of an encryption key site, Keystore passwords, key aliases, and a key password.
152	SIM_FULL	says "SIM card is filled," it most likely means that the storage capacity of the SIM card has been reached.
153	SEND_MESSAGE	It can use the SmsManager Sdk or the phone's built-in SMS application to send SMS on Android.
154	UID_REMOVED	Android Enthusiasts Based Processes is a question-and-answer site for Android fans and power users.
155	CAMERA_BUTTON	this button, a menu of features appears on the camera's LCD screen that can operate directly.
156	setRequestMethod(	setRequestMethod ():Protected void is the syntax. Protocol Exception Parameters are thrown by setRequestMethod (String method).

S.NO	ATTRIBUTE NAME	DESCRIPTION
157	getInputStream(	getInputStream () returns a stream that can be used to read bytes from this socket. If an Input/Output error happens while constructing the input stream, this socket is closed, the specified port is not linked, or the socket's input has been closed down using shutdownInput, an IOException will be thrown ().
158	getOutputStream(	getOutputStream () gives a stream that can be used to write bytes to this port. If an I/O fault appears during establishing the output stream, or if the connection is not connected, an IOException is thrown.
159	getCallingUid(	Access to the getCallingUid function is granted.
160	getCallingPid(	Enables users to access the android os Binder getCallingPid functionality.
161	transact(	Authorization to access Transaction is granted.
162	onBind	Marks a method onBind () that returns an IBinder that can be used to communicate with the service.
163	IRemoteService	IRemoteService is a foundation component that provides the appropriate connection parameters for accessing a distant RemObjects service.
164	ServiceConnection	Permits access to the connect ServiceConnection Context.
165	Context.bindService	To obtain a permanent connection to a service, use bindService ().
166	bindService	Other apps can bind to it and communicate with it because it implements the Service class.

S.NO	ATTRIBUTE NAME	DESCRIPTION
167	IBinder	IBinder is an interface that, among other things, provides IPC (Inter Process Communications).
168	Binder	Binder is an IPC (inter-process communication) medium that users can utilise.
169	getBinder	Binder is an interprocess communication and remote procedure invocation technology designed specifically for Android. That really is, one Android processes can call a procedure in an another Android process, identifying the function to invoke and passing the inputs between processes via binder.
170	MessengerService	MessengerService Class IncomingHandler Class handleMessage Method onCreate Method onDestroy Method onBind Method showNotification Method.
171	SecretKey	To keep secrets safe, the Keystore API employs both types of cryptography. A cryptographically RSA pair is created, which is saved in the keystore of the Android device and is normally protected by the device PIN.
172	KeySpec	keySpec Class specifies how the key material should be returned in the requested format.
173	Landroid/location/LocationManager.*getAllProviders(	Enables the getAllProviders function in android.location. LocationManager to be used.
174	android.hardware	Enables to acquire hardware packages is granted.
175	checkSignatures(	Analyse the signature of two programs to see if the signatures are the same in both.

S.NO	ATTRIBUTE NAME	DESCRIPTION
176	Chmod	Provides access to chmod, which modifies the permission of each file based on its mode.
177	Chown	Provides access to the chown command, which can be used to modify the ownership values.
178	\/system\/app	Enables an Android device to access an app stored in the /system/app folder.
179	\/system\/bin	Provides access to data and acts as a temporarily storage space on Android smartphones.
180	\/system\/bin\/su	The subinary is placed in /system/xbin/su and allows access to it.
181	\/system\/bin\/sh	Provides access to /system/xbin/sh, which is stored at /system/xbin/sh.
182	Mount	Enables to use the mounted command on Android device.
183	Remount	Enables users to access Mount system RO with the command mount -o ro,remount /system.
184	Grep	Enables to use the G rep command to search for text.
185	\/sh	Allow 'sh' commands to be executed by programmes.
186	\/bin	On Android phones, the.bin file is generally created when.apk files fail.
187	Insmod	In system/core/init/buildin.c, insmod is used to access Android init.
188	Stdout	Provides access to the Android system's stdout.

S.NO	ATTRIBUTE NAME	DESCRIPTION
189	Stderr	Provides access to the Android system and outputs stderr.
190	Killall	Enable background apps to be closed.
191	Reboot	Reboot the device; this is for system code to use.
192	\/dev\/net	Enables access to the device in order to connect to the network.
193	\/system	Provides access to the Android Operating System's System Services.
194	pm install	Provides access to Using the forward lock -r option, install the package, a previously installed app should be reinstalled.
195	Result	Show the android botnet result.

Table description:

The table 4.1.1 provides the detail about attribute and description of attribute in ISCX dataset.

The following table 4.1.2 represents kernel command lines attribute from ISCX dataset has been list.

S.NO	ATTRIBUTE NAME
1	android.intent.action.TIME_SET
2	android.intent.action.TIMEZONE_CHANGED
3	android.intent.action.BOOT_COMPLETED
4	android.intent.action.PACKAGE_ADDED
5	android.intent.action.PACKAGE_CHANGED
6	android.intent.action.PACKAGE_REMOVED
7	android.intent.action.PACKAGE_RESTARTED
8	android.intent.action.PACKAGE_DATA_CLEARED
9	android.intent.action.UID_REMOVED
10	android.intent.action.ACTION_POWER_CONNECTED
11	android.intent.action.ACTION_POWER_DISCONNECTED
12	android.intent.action.ACTION_SHUTDOWN
13	android.intent.action.PACKAGE_REPLACED
14	android.intent.action.BATTERY_LOW
15	android.intent.action.BATTERY_OKAY
16	android.intent.action.CALL
17	android.intent.action.CALL_BUTTON
18	android.intent.action.CAMERA_BUTTON

S.NO	ATTRIBUTE NAME
19	android.intent.action.NEW_OUTGOING_CALL
20	android.intent.action.REBOOT
21	android.intent.action.SCREEN_OFF
22	android.intent.action.SCREEN_ON
23	android.intent.action.SEND
24	android.intent.action.SENDTO
25	android.intent.action.SET_WALLPAPER
26	android.settings.NETWORK_OPERATOR_SETTINGS
27	android.intent.action.SEND_MULTIPLE
28	android.settings.APN_SETTINGS
29	.zip
30	.apk
31	.dex
32	.exe
33	.so
34	abortBroadcast
35	HttpPost.*init
36	HttpGet.*init
37	HttpUriRequest

S.NO	ATTRIBUTE NAME
38	Ljava.net.URLDecoder
39	System.*loadLibrary
40	Ljava\/lang\/Object.*getClass
41	Ljava\/lang\/Class.*getMethods(
42	Ljava\/lang\/Class.*forName
43	Ljava\/lang\/Class.*cast
44	Ljava\/lang\/Class.*getClasses
45	Ljava\/lang\/Class.*getCanonicalName
46	Ljava\/lang\/Class.*getDeclaredClasses
47	Ljava\/lang\/Class.*getDeclaredField
48	Ljava\/lang\/Class.*getField
49	Ljava\/lang\/Class.*getSigners
50	Ljava\/lang\/Class.*getResource
51	Ljava\/lang\/Class.*getPackage
52	DexClassLoader
53	DexFile.*loadClass
54	ClassLoader
55	findClass
56	defineClass

S.NO	ATTRIBUTE NAME
57	PathClassLoader
58	URLClassLoader
59	loadClass(
60	android.os.IBinder
61	onServiceConnected(
62	Ljavax\/crypto\/Cipher
63	Ljavax\/crypto\/spec\/SecretKeySpec
64	doFinal(
65	Runtime.*exec
66	createSubprocess
67	Runtime.*load
68	Runtime.*loadLibrary
69	ProcessBuilder
70	Process.*start
71	Process.*myPid
72	Runtime.*getRuntime
73	killProcess(
74	android.telephony.gsm.SmsManager
75	android.telephony.SmsManager

S.NO	ATTRIBUTE NAME
76	divideMessage
77	sendTextMessage(
78	android.content.pm.PackageInfo
79	android.content.pm.Signature
80	PackageInstaller
81	getInstalledPackages(
82	TelephonyManager.*getDeviceId
83	TelephonyManager.*getSubscriberId
84	TelephonyManager.*getSimSerialNumber
85	TelephonyManager.*getLine1Number
86	TelephonyManager.*getNetworkOperator
87	TelephonyManager.*getSimOperator
88	TelephonyManager.*getCallState
89	TelephonyManager.*isNetworkRoaming
90	getCellLocation(
91	TelephonyManager.*getSimCountryIso
92	Ljava.util.Timer
93	Ljava.util.Timer.*schedule
94	Ljava.util.TimerTask

S.NO	ATTRIBUTE NAME
95	Ljava.util.Date
96	AssetManager
97	getResources
98	Landroid.content.res.AssetManager
99	getAssets
100	getContentResolver.*query
101	content:\/\/sms
102	content:\/\/telephony
103	content:\/\/mail
104	content:\/\/downloads
105	content:\/\/browser
106	content:\/\/contacts
107	Ljava.net.InetSocketAddress
108	getDataDir(
109	getApplicationInfo(
110	getSystemService(
111	BatteryManager
112	AudioManager
113	CameraManager

S.NO	ATTRIBUTE NAME
114	NfcManager
115	SensorManager
116	UsbManager
117	WifiManager
118	BluetoothManager
119	addFlags(
120	setFlags(
121	getRunningServices(
122	getMemoryInfo(
123	restartPackage(
124	onActivityResult
125	getNetworkInfo(
126	getExtraInfo(
127	getTypeName(
128	isConnected(
129	getState(
130	setWifiEnabled(
131	getWifiState(
132	android.os.Handler

S.NO	ATTRIBUTE NAME
133	obtainMessage(
134	sendMessage(
135	DataInputStream.*available(
136	FileOutputStream.*write(
137	FileOutputStream.*write(
138	io.File.*delete(
139	io.File.*mkdir
140	io.File.*exists(
141	ZipInputStream.*read(
142	ZipInputStream.*close(
143	ZipInputStream.*getNextEntry(
144	getElementByTagName(
145	getAttribute(
146	getDocumentElement(
147	getSystemAvailableFeatures(

Table Description:

The above table 4.1.2 represents kernel command lines attribute from ISCX Dataset out off 342 features 147 kernel attributes presents.

4.2 PHASE 2: DATA PRE-PROCESSING

Data preprocessing is used in Machine Learning refers to the process of cleaning and organising raw data in order to make it appropriate for creating and training Machine Learning models. In simple terms, data preprocessing is a data mining technique used in Machine Learning that processes raw data into a readable and comprehensible format.

The process of removing incorrect, incomplete, and incorrect data from datasets, as well as filling missing values, is known as data cleaning. Data preprocessing is the transformation of raw data into data suitable for machine learning. It is perhaps the most efficient in developing a machine learning approach.

Real-world data may contain noise, missing values, or be in an inappropriate format that cannot be directly used in machine learning techniques. Data preprocessing is a necessary task for cleaning data and making data suitable for a machine learning algorithms, which improves the model's accuracy and efficiency.

There are a few methods for cleaning data

- Null Values
- Label Encoding
- Feature Scaling
    a) Normalization
    b) Standardization
- Train and Test Split

4.2.1 Null Values

Delete or Remove the rows or columns with null values to deal with missing values If many than quarter of the rows are filled in a column are null, the column might be eliminated entirely. The rows contain null values in one or more columns can also be removed.

4.2.2 Label Encoding

Label encoding refers to the transformation of labels into numeric form in order to convert them into a machine-readable format. In machine learning algorithms can correctly decide how these labels should be controlled. In supervised learning, it is a critical pre-processing measure during the integrated dataset.

4.2.3 Feature Scaling

Feature set is a method for normalizing a collection of independent factors or data elements. In data processing, it is also called as data normalization and is typically performed during in the data preprocessing stage. It is a method for trying to fit the data's independent characteristics into a predefined range.

a) Normalization

Normalization is a scaling process used in Machine Learning to convert the values of numeric columns in a dataset to a common scale during data preparation. It might not be required for each dataset in a model. It is only required when the ranges of features in machine learning models differ, all of the values are now in the range of 0 to 1. This is how the sklearn normalize() technique works.

b) Standardization

Another scaling strategy is standardisation, in which the values are centred around the mean with a unit standard deviation. The StandardScaler() method in the Python sklearn module allows to standardise data values into a standard format.

Syntax:

```
object = StandardScaler()

object.fit_transform(data)
```

It start with creating an object of the StandardScaler() function using the syntax above. We also use fit transform() with the provided object to transform and standardise the data.

4.2.4 Train and Test Split

A mechanism for measuring the performance of a machine learning algorithm is the train-test split. It can be used for any supervised learning technique and can be utilised for classification or regression tasks. Acquiring a dataset and splitting it into two subgroups is the technique.

It is desirable in the creation of machine learning techniques for the trained model to operate well on new, different datasets. To imitate the new, unseen data, the existing data is split into two parts and subjected to data splitting (sometimes referred to as the train-test split). The first portion is typically a large dataset that is used as the training set (e.g., comprising for 80% of the original data) while the second portion is typically a smaller subset that is used as testing set (the remaining 20 percent of the data). It's worth noting that this type of data split typically happens once.

The training dataset is then used to create a predictive model, which is subsequently used in the testing dataset (i.e. data that has never been seen before) to produce predictions.

Train test split () splits two sequences, x and y in this case, and produces four sequences (in this case NumPy arrays) in the following order:

a) x train: The first sequence's training phase (x)
b) x test: The initial sequence's test section (x)
c) y train: The second sequence's training phase (y)
d) y test: The second sequence's test section (y)

4.3 PHASE 3: FEATURE SELECTION

The feature selection procedure is based on a machine learning technique that we are attempting to apply to a given dataset. A feature is an aspect that has an impact on or is relevant for an issue, and feature selection is the process of selecting the most important features for the model. The process of discovering significant or influential variables from the target variable in the existing features set is known as feature selection.

A feature selection algorithm is composed of a search method for implementing new feature subsets and an evaluation measure for scoring the various feature subsets. The basic technique is to evaluate each feasible subset of features to find the one with the lowest error rate. It is an exhaustive study of the area and is computationally difficult for all but the smallest feature sets. The algorithm is heavily influenced by the evaluation metric chosen, and these are the performance measures that differentiate between the three types of feature selection methods: filters, wrapper and embedded methods.

Various methods or approaches, such as Decision Trees, Linear Regression, and Random Forest, can be used to choose features.

a) Wrapper Based Feature Selection Methods

The Wrapper methodology approaches feature set selection as a search problem, in which various combinations are prepared, evaluated, and compared to other combinations. A predictive model to analyze a feature set and assigns performance scores to the model.

Wrapper techniques use optimistic search algorithms to examine all possible feature combinations and select the one that produces better result for a specified machine learning algorithm. Wrapper approaches are frequently more accurate predictors than filter methods. It has some of techniques:

- Forward Feature Selection Method
- Backward Feature Selection Method
- Stepwise Feature Selection Method

4.3.1 Forward Feature selection

This is an iterative technique in which the best performing variable versus the target is being used as the starting point. This algorithm is an iterative technique in which we begin with a blank set of features and add a feature that improves our model the most with each iteration. This procedure is carried out again and again until the specified requirement is reached. Let me define some of the terms we use in SFS:

PSEDOCODE FOR FORWARD FEATURE SELECTION

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Installed mlxtend from which the feature selection methods to be imported.

STEP 4 **:** Using wrapper based feature selction method is implemented from sequential feature selector package using Linear Regression, and top 85 feature are selected.

STEP 5 **:** Top 85 feature selected based on greedy search approach.

STEP 6 **:** Fit the feature into X and y.

STEP 7 **:** Print performance and average score for selected features.

STEP 8 **:** Stop the process.

Top 85 features from ISCX Dataset selected using Forward Feature Selection Method:

S.NO	FEATURE NAME
1	ACCESS_CHECKIN_PROPERTIES
2	ACCESS_SURFACE_FLINGER
3	AUTHENTICATE_ACCOUNTS
4	BIND_DEVICE_ADMIN
5	BROADCAST_SMS
6	CAMERA
7	CHANGE_COMPONENT_ENABLED_STATE
8	CHANGE_WIFI_STATE
9	DELETE_PACKAGES
10	GET_PACKAGE_SIZE
11	INTERNET
12	KILL_BACKGROUND_PROCESSES
13	MODIFY_AUDIO_SETTINGS
14	MOUNT_FORMAT_FILESYSTEMS
15	MOUNT_UNMOUNT_FILESYSTEMS
16	NFC
17	PERSISTENT_ACTIVITY
18	PROCESS_OUTGOING_CALLS

S.NO	FEATURE NAME
19	READ_LOGS
20	READ_PROFILE
21	READ_SYNC_STATS
22	RECEIVE_SMS
23	RECEIVE_WAP_PUSH
24	REORDER_TASKS
25	SEND_SMS
26	VIBRATE
27	WAKE_LOCK
28	WRITE_CALENDAR
29	WRITE_CALL_LOG
30	WRITE_HISTORY_BOOKMARKS
31	WRITE_SETTINGS
32	android.intent.action.TIME_SET
33	android.intent.action.PACKAGE_DATA_CLEARED
34	android.intent.action.SCREEN_ON
35	android.settings.NETWORK_OPERATOR_SETTINGS
36	USER_PRESENT
37	SMS_RECEIVED

S.NO	FEATURE NAME
38	PHONE_STATE
39	UMS_DISCONNECTED
40	SIM_FULL
41	.zip
42	getOutputStream(
43	Ljava.net.URLDecoder
44	System.*loadLibrary
45	Ljava\\/lang\\/Class.*getMethods(
46	DexFile.*loadDex
47	PathClassLoader
48	IBinder
49	Binder
50	Ljavax\\/crypto\\/Cipher
51	Ljavax\\/crypto\\/spec\\/SecretKeySpec
52	doFinal(
53	Runtime.*exec
54	ProcessBuilder
55	killProcess(
56	sendTextMessage(

S.NO	FEATURE NAME
57	GetInstalledPackages(
58	TelephonyManager.*getDeviceId
59	TelephonyManager.*getSubscriberId
60	TelephonyManager.*getSimSerialNumber
61	TelephonyManager.*getLine1Number
62	TelephonyManager.*getCallState
63	TelephonyManager.*getSimCountryIso
64	Ljava.util.TimerTask
65	Ljava.util.Date
66	getResources
67	content:\\/\\/browser
68	Ljava.net.InetSocketAddress
69	getApplicationInfo(
70	BatteryManager
71	NfcManager
72	WifiManager
73	BluetoothManager
74	setFlags(
75	onActivityResult

S.NO	FEATURE NAME
76	IsConnected(
77	ObtainMessage(
78	DataInputStream.*available(
79	FileOutputStream.*write(
80	io.File.*delete(
81	io.File.*exists(
82	ZipInputStream.*getNextEntry(
83	checkSignatures(
84	\\/system\\/bin
85	killall

Table Description:

The above table 4.3.1 represent Forward feature selection is used to select top 85 features using k_features. According to ISCX dataset and scores, it is a random value.

4.3.2 Backward Feature selection

In machine learning model, backward elimination is a feature selection technique. It's used to get rid of features that don't have much of an impact on the dependent variable or output prediction. The Forward Feature Selection approach is the opposite of this technique. It start with all the features available and develop a model, then select the variable from the model that provides the best evaluation measure value. This method is repeated until the specified requirement is reached. Let me define some of the terms we use in SFS:

PSEDOCODE FOR BACKWARD FEATURE SELECTION

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Installed mlxtend from which the feature selection methods to be imported.

STEP 4 **:** Using wrapper based feature selction method is implemented from sequential feature selector package using Linear Regression, and top 85 feature are selected.

STEP 5 **:** Top 85 feature selected based on greedy search approach.

STEP 6 **:** Fit the feature into X and y.

STEP 7 **:** Print performance and average score for selected features.

STEP 8 **:** Stop the process.

Top 85 features from ISCX Dataset selected using Backward Feature Selection Method:

S.NO	FEATURE NAME
1	ACCESS_CHECKIN_PROPERTIES
2	ACCESS_SURFACE_FLINGER
3	AUTHENTICATE_ACCOUNTS
4	BIND_DEVICE_ADMIN
5	BROADCAST_SMS
6	CALL_PRIVILEGED
7	CAMERA
8	CHANGE_COMPONENT_ENABLED_STATE
9	CHANGE_NETWORK_STATE
10	CHANGE_WIFI_STATE
11	INTERNET
12	KILL_BACKGROUND_PROCESSES
13	MANAGE_ACCOUNTS
14	MODIFY_AUDIO_SETTINGS
15	MODIFY_PHONE_STATE
16	MOUNT_FORMAT_FILESYSTEMS
17	MOUNT_UNMOUNT_FILESYSTEMS
18	PERSISTENT_ACTIVITY

S.NO	FEATURE NAME
19	PROCESS_OUTGOING_CALLS
20	READ_HISTORY_BOOKMARKS
21	READ_PHONE_STATE
22	READ_PROFILE
23	READ_SMS
24	READ_SYNC_STATS
25	RECEIVE_SMS
26	REORDER_TASKS
27	SEND_SMS
28	VIBRATE
29	WAKE_LOCK
30	WRITE_CALL_LOG
31	WRITE_SECURE_SETTINGS
32	WRITE_SMS
33	android.intent.action.TIME_SET
34	android.intent.action.SCREEN_ON
35	android.settings.NETWORK_OPERATOR_SETTINGS
36	USER_PRESENT
37	SMS_RECEIVED

S.NO	FEATURE NAME
38	PACKAGE_REPLACED
39	UMS_DISCONNECTED
40	SIM_FULL
41	.zip
42	.dex
43	getOutputStream(
44	System.*loadLibrary
45	Ljava\\/lang\\/Class.*getMethods(
46	DexFile.*loadClass
47	DexFile.*loadDex
48	PathClassLoader
49	IBinder
50	Binder
51	Ljavax\\/crypto\\/Cipher
52	Ljavax\\/crypto\\/spec\\/SecretKeySpec
53	doFinal(
54	ProcessBuilder
55	sendTextMessage(
56	GetInstalledPackages(

S.NO	FEATURE NAME
57	TelephonyManager.*getDeviceId
58	TelephonyManager.*getSubscriberId
59	TelephonyManager.*getSimSerialNumber
60	TelephonyManager.*getLine1Number
61	TelephonyManager.*getCallState
62	TelephonyManager.*getSimCountryIso
63	Ljava.util.TimerTask
64	Ljava.util.Date
65	Landroid.content.res.AssetManager
66	getAssets
67	content:\\/\\/browser
68	Ljava.net.InetSocketAddress
69	getApplicationInfo(
70	BatteryManager
71	NfcManager
72	BluetoothManager
73	setFlags(
74	onActivityResult
75	IsConnected(

S.NO	FEATURE NAME
76	ObtainMessage(
77	DataInputStream.*available(
78	FileOutputStream.*write(
79	io.File.*delete(
80	io.File.*exists(
81	ZipInputStream.*getNextEntry(
82	checkSignatures(
83	chmod
84	\\/system\\/bin
85	killall

Table Description:

The above table 4.3.1 represent Backward feature selection is used to select top 85 features using k_features. According to ISCX dataset and scores, it is a random value.

4.3.3 STEP-WISE SELECTION

Stepwise selection was designed as a feature selection strategy for linear regression analysis initially. A forward stepwise regression method comprises a range of steps that allow features to enter the regression model one by one. This technique frequently converges on a subset of attributes. Let me define some of the terms we use in SFS:

PSEDOCODE FOR STEPWISE FEATURE SELECTION

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Installed mlxtend from which the feature selection methods to be imported.

STEP 4 **:** Using wrapper based feature selction method is implemented from sequential feature selector package using Linear Regression, and top 85 feature are selected.

STEP 5 **:** Top 85 feature selected based on greedy search approach.

STEP 6 **:** Fit the feature into X and y.

STEP 7 **:** Print performance and average score for selected features.

STEP 8 **:** Stop the process.

Top 85 features from ISCX Dataset selected using Stepwise Feature Selection Method:

S.NO	FEATURE NAME
1	ACCESS_CHECKIN_PROPERTIES
2	ACCESS_SURFACE_FLINGER
3	AUTHENTICATE_ACCOUNTS
4	BIND_DEVICE_ADMIN
5	BROADCAST_SMS
6	CAMERA
7	CHANGE_COMPONENT_ENABLED_STATE
8	CHANGE_WIFI_STATE
9	DELETE_PACKAGES
10	GET_PACKAGE_SIZE
11	INTERNET
12	KILL_BACKGROUND_PROCESSES
13	MODIFY_AUDIO_SETTINGS
14	MOUNT_FORMAT_FILESYSTEMS
15	MOUNT_UNMOUNT_FILESYSTEMS
16	NFC
17	PERSISTENT_ACTIVITY
18	PROCESS_OUTGOING_CALLS

S.NO	FEATURE NAME
19	READ_LOGS
20	READ_PROFILE
21	READ_SYNC_STATS
22	RECEIVE_SMS
23	RECEIVE_WAP_PUSH
24	REORDER_TASKS
25	SEND_SMS
26	VIBRATE
27	WAKE_LOCK
28	WRITE_CALENDAR
29	WRITE_CALL_LOG
30	WRITE_HISTORY_BOOKMARKS
31	WRITE_SETTINGS
32	android.intent.action.TIME_SET
33	android.intent.action.PACKAGE_DATA_CLEARED
34	android.intent.action.SCREEN_ON
35	android.settings.NETWORK_OPERATOR_SETTINGS
36	USER_PRESENT
37	SMS_RECEIVED

S.NO	FEATURE NAME
38	PHONE_STATE
39	UMS_DISCONNECTED
40	SIM_FULL
41	.zip
42	getOutputStream(
43	Ljava.net.URLDecoder
44	System.*loadLibrary
45	Ljava\\/lang\\/Class.*getMethods(
46	DexFile.*loadDex
47	PathClassLoader
48	IBinder
49	Binder
50	Ljavax\\/crypto\\/Cipher
51	Ljavax\\/crypto\\/spec\\/SecretKeySpec
52	doFinal(
53	Runtime.*exec
54	ProcessBuilder
55	killProcess(
56	sendTextMessage(

S.NO	FEATURE NAME
57	GetInstalledPackages(
58	TelephonyManager.*getDeviceId
59	TelephonyManager.*getSubscriberId
60	TelephonyManager.*getSimSerialNumber
61	TelephonyManager.*getLine1Number
62	TelephonyManager.*getCallState
63	TelephonyManager.*getSimCountryIso
64	Ljava.util.TimerTask
65	Ljava.util.Date
66	getResources
67	content:\\/\\/browser
68	Ljava.net.InetSocketAddress
69	getApplicationInfo(
70	BatteryManager
71	NfcManager
72	WifiManager
73	BluetoothManager
74	setFlags(
75	onActivityResult

S.NO	FEATURE NAME
76	IsConnected(
77	ObtainMessage(
78	DataInputStream.*available(
79	FileOutputStream.*write(
80	io.File.*delete(
81	io.File.*exists(
82	ZipInputStream.*getNextEntry(
83	checkSignatures(
84	\\/system\\/bin
85	killall

Table Description:

The above table 4.3.3 represent Stepwise feature selection is to select top 85 features using k_features. According to ISCX dataset and scores, it is a random value.

4.4 PHASE 4: MODEL BUILDING

Supervised classification learning, as well referred as supervised machine learning techniques , is an artificial intelligence and machine learning subcategory. It is defined by the use of labelled data to prepare techniques that accurately procedures for dealing or predict outcomes. As data is provided into the framework, the adjusting the weights until the model is correctly fitted, which occurs during cross-validation phase. Models built with supervised machine learning methods include,

- Random Forest
- Naive Bayes
- Decision Tree
- Multilayer Perceptron
- Support Vector Machine.

Depending on the data type (qualitative or quantitative) of the target attribute (often referred to as the Y variable), we will either develop a classification (if Y is qualitative) or regression (if Y is quantitative) model with the precisely gathered data.

4.4.1 RANDOM FOREST CLASSIFIER

Random Forest classifier is a well-known machine learning technique that employs supervised learning. It's used in machine learning for both regression and classification problems. It is based on ensemble learning, which appears to be a method of integrating multiple classifiers to solve a complex problem and improve model performance.

A forest is composed of branches, and more trees equals a healthier forest. The random forest technique, on the other hand, builds tree structure from sample data, extract prediction from all, and then decides on the best alternative. It's an ensemble learning method that's better than using a major decision node because that averages the outcomes to avoid overfitting.

According to the name, "Random Forest is a classification that includes a collection of decision trees based on different selections of a particular dataset and takes the average to boost the dataset's projected accuracy." The confusion matrix collects predictions out of each forest

and forecasts the final output depending on the majority of votes of projections, rather than depending on a decision tree classifier.

PSEDOCODE FOR RANDOM FOREST CLASSIFIER

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Create a function ensemble and give necessary parameters.

STEP 4 **:** Fit those function in trained dataset.

STEP 5 **:** Create another function to store the test dataset predicted accuracy.

STEP 6 **:** Print classification report to display precision, recall, f1-score and support.

STEP 7 **:** Display the train and test accuracy using accuracy_score.

STEP 8 **:** Display men squared error, mean absolute error, root mean squared error using Regression error metrics .

STEP 9 **:** The output is evaluated.

STEP 10 **:** Stop the process.

4.4.2 NAÏVE BAYES CLASSIFIER

The Naïve Bayesian technique is a supervised learning algorithm depending on the Bayes theorem for dealing with classification problems. It is most commonly used in text categorization that necessitate a large - scale training dataset. Naive Bayes is a significant and simple classification technique that assist in the development of efficient machine learning approaches capable of making accurate predictions.

It's a probabilistic classifier, which means it makes predictions based on an object's probability. Spam filtration, sentiment analysis, and article classification are all common uses of the Naive Bayes Algorithm.

PSEDOCODE FOR NAVIE BAYES CLASSIFIER

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Create a function GaussianNB and give necessary parameters.

STEP 4 **:** Fit those function in trained dataset.

STEP 5 **:** Create another function to store the test dataset predicted accuracy.

STEP 6 **:** Print classification report to display precision, recall, f1-score and support.

STEP 7 **:** Display the train and test accuracy using accuracy_score.

STEP 8 **:** Display men squared error, mean absolute error, root mean squared error using Regression error metrics .

STEP 9 **:** The output is evaluated.

STEP 10 **:** Stop the process.

4.4.3 DECISION TREE CLASSIFIER

Decision tree classifier is a tree method that looks like a flowchart, with each internal node representing an attribute test, each department representing a test outcome, and leaf nodes representing a class or subclass distribution. The root node is the maximum node in a tree. Decision Tree Classifier is a basic diagram for categorizing examples.

It's supervised machine learning in which data is constantly separated according to a parameter. Decision Tree Classifier is a Learning method that is supervised that can solve regression and classification problems but is most commonly used to solve classification problems. It is a tree-structured classifier, with internal nodes representing dataset features, and branches representing Each leaf node has its own set of decision rules representing the result.

PSEDOCODE FOR DECISION TREE CLASSIFIER

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Create a function tree and give necessary parameters.

STEP 4 **:** Fit those function in trained dataset.

STEP 5 **:** Create another function to store the test dataset predicted accuracy.

STEP 6 **:** Print classification report to display precision, recall, f1-score and support.

STEP 7 **:** Display the train and test accuracy using accuracy_score.

STEP 8 **:** Display men squared error, mean absolute error, root mean squared error using Regression error metrics .

STEP 9 **:** The output is evaluated.

STEP 10 **:** Stop the process.

4.4.4 MULTILAYER PERCEPTRON(MLP)

The MLP technique is a learning algorithm that is commonly used in MLP networks. Multilayer perceptron (MLP) is a feed-forward forward convolutional neural network that produces a set of results from a selection of inputs. An MLP is defined by multiple layers of input nodes connected as a graph structure between both the hidden and output layers.

PSEDOCODE FOR MLP CLASSIFIER

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Create a function neural network and give necessary parameters.

STEP 4 **:** Fit those function in trained dataset.

STEP 5 **:** Create another function to store the test dataset predicted accuracy.

STEP 6 **:** Print classification report to display precision, recall, f1-score and support.

STEP 7 **:** Display the train and test accuracy using accuracy_score.

STEP 8 **:** Display men squared error, mean absolute error, root mean squared error using Regression error metrics .

STEP 9 **:** The output is evaluated.

STEP 10 **:** Stop the process.

4.4.5 SUPPORT VECTOR MACHINE

Support vector Networks (SVMs, as well known as support vector machine) are supervised Machine learning algorithm with related learning algorithms used in machine learning for regression and classification analysis. Though we might also argue regression difficulties, categorization is the best fit. The purpose of the SVM methods is to determine a higher dimensional space in an N-dimensional region that group data points clearly. SVM is a classifier with a hyperplane which is separated. In other words, the algorithm produces an ideal hyperplane that categorises fresh samples given labelled train data (supervised learning).

PSEDOCODE FOR SVM CLASSIFIER

STEP 1 **:** Start the process.

STEP 2 **:** Import necessary libraries.

STEP 3 **:** Create a function SVC with kernel linear and give necessary parameters.

STEP 4 **:** Fit those function in trained dataset.

STEP 5 **:** Create another function to store the test dataset predicted accuracy.

STEP 6 **:** Print classification report to display precision, recall, f1-score and support.

STEP 7 **:** Display the train and test accuracy using accuracy_score.

STEP 8 **:** Display men squared error, mean absolute error, root mean squared error using Regression error metrics .

STEP 9 **:** The output is evaluated.

STEP 10 **:** Stop the process.

4.5 PHASE 5: COMPARATIVE ANALYSIS

4.5.1 PREFORMANCES EVALUTAION METRICS

Comparative analysis is actual data with predicted data and gives the number of correct and incorrect classifications. When the model predicts fraudulent claim as fraud, it is a true positive (TP) and if it predicts fraudulent as non-fraud, it is a false negative (FN). This study presents a comparison of five machine learning algorithms are used to develop the detection models.

- Confusion Matrix
- Accuracy
- Precision
- Recall
- F1-Score

Confusion Matrix:

The confusion metric is used for classification problems and is regarded as the most straightforward metric for performance evaluation. A confusion matrix is a simple table with two dimensions: 'actual' and 'predicted,' and it has the same sets of classes for both dimensions. There are two classes in a simple classification problem: positive (1) and negative (0).

Precision Score:

Model precision score is a useful measure of the success of prediction when the classes are very imbalanced.

Precision Score = TP / (FP + TP)

Recall Score:

Model Recall score is a useful measure of success of prediction when the classes are very imbalanced.

Recall Score = TP / (FN + TP)

Accuracy Score:

Model accuracy is a machine learning model performance metric that is defined as the ratio of true positives and true negatives to all positive and negative observations.

Accuracy Score = (TP + TN)/ (TP + FN + TN + FP)

F1 Score:

Model F1 score represents the model score as a function of precision and recall score.

F1 Score = 2* Precision Score * Recall Score/ (Precision Score + Recall Score)

4.5.2 REGRESSION ERROR METRICS

Machine Learning model cannot have 100% of accuracy efficiency otherwise the model is known as a biased model. Evaluation metrics which helps to better optimize the performance, fine-tune it, and obtain a better result. There are three error metrics that are commonly used for evaluating and reporting the performance of a regression model. They are Mean Squared Error(MSE), Root Mean Squared Error(RMSE) and Mean Absolute Error(MAE).

RESULTS AND DISCUSSION

CHAPTER – 5

RESULTS AND DISCUSSION

5.1 PHASE 1: DATA COLLECTION

The following figure represents relationship between the Android Botnet Application and Android Botnet Application from ISCX Dataset.

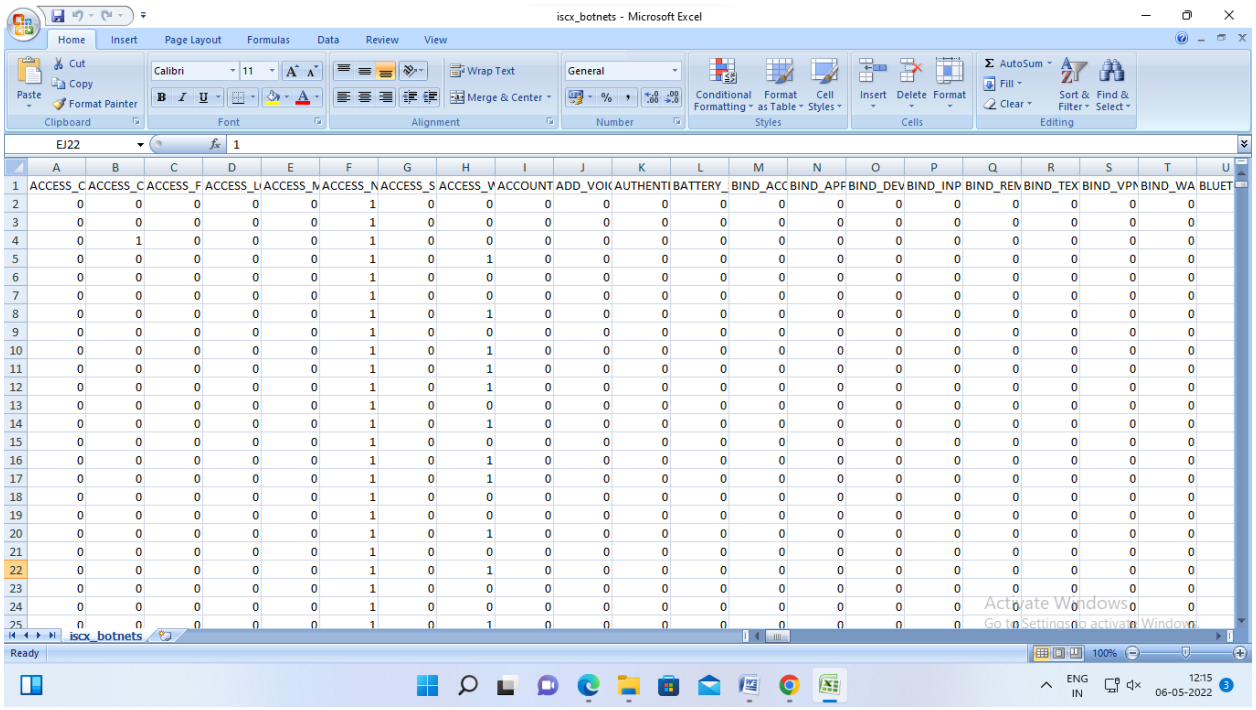


Figure 5.1 Data Collection

Description

The above figure 5.1 represents, ISCX Android Botnet dataset contains a huge number of Android botnets from 14 different botnet families. This dataset was created using 6802 Android applications, which included 4873 benign clean apps and 1929 botnet apps from the ISCX botnet dataset. This dataset contains 342 static features extracted from application files.

5.2 PHASE 2: DATA PRE-PROCESSING

5.2.1 EXPLORATORY DATA ANALYSIS

The following figure represents relationship between the Botnet as 1 and Benign as 0 from ISCX Dataset.

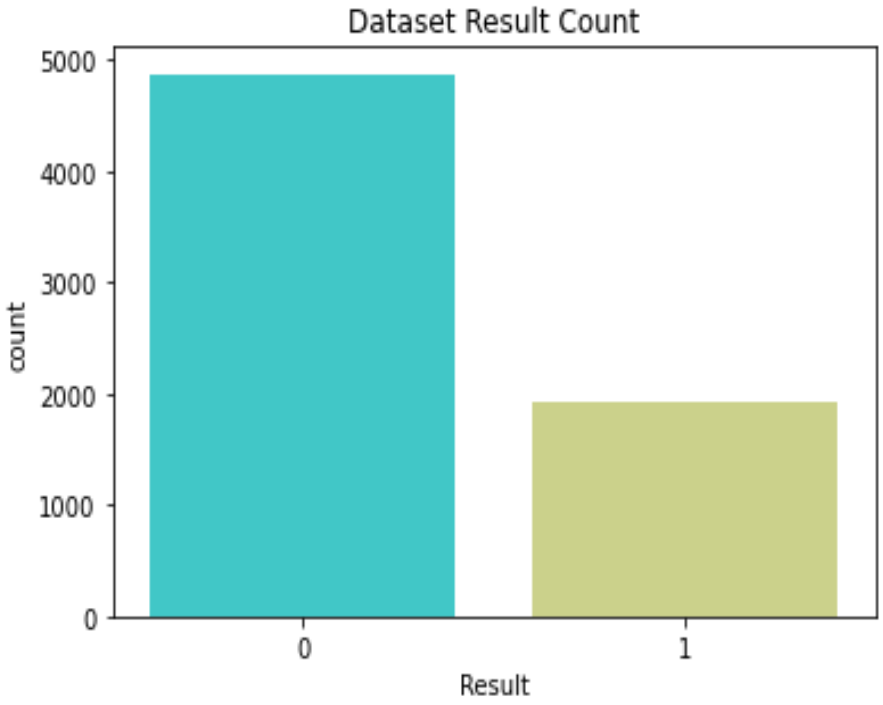


Figure 5.2.1: Exploratory Data Analysis after Label Encoding

Description

The above figure. 5.2.1 represents the Android Benign Application is used from the ISCX Dataset consisting of a large number of related variables such as 4873 Applications and the Android Botnet Application is consisting 1929 Applications. The observed relationships between Botnet and Benign.

5.2.2 Feature scaling using Standardization

```
array([[-0.08777075, -0.38799648, -0.42001137, ..., -0.42122171,
        -0.08691626,  1.58939591],
       [-0.08777075, -0.38799648, -0.42001137, ..., -0.42122171,
        -0.08691626,  1.58939591],
       [-0.08777075,  2.57734294, -0.42001137, ..., -0.42122171,
        -0.08691626,  1.58939591],
       ...,
       [-0.08777075, -0.38799648, -0.42001137, ..., -0.42122171,
        -0.08691626, -0.62916986],
       [-0.08777075, -0.38799648, -0.42001137, ..., -0.42122171,
        -0.08691626, -0.62916986],
       [-0.08777075,  2.57734294, -0.42001137, ..., -0.42122171,
        -0.08691626, -0.62916986]])
```

Figure 5.2.2: Feature scaling using Standardization

Description

The above figure 5.2.2 represents, Feature scaling using Standardization method.

5.2.3 Train and Test Split

```
from sklearn.model_selection import train_test_split
X = pd.DataFrame(dataset.iloc[:,:-1])
y = pd.DataFrame(dataset.iloc[:,-1])

# Splitting the dataset into train and test sets: 80-20 split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

```
((5441, 342), (5441, 1), (1361, 342), (1361, 1))
```

Figure 5.2.3: Training and Testing Splitting

Description

The above figure 5.2.3 represents Training and Testing Splitting, comprising for 80% train data and 20% for test data out off 342 features and 6802 records from ISCX Dataset.

5.3 PHASE 3: WRAPPER BASED FATURE SELECTION METHOD

5.3.1 Forward Feature Selection

	feature_idx	cv_scores	avg_score	feature_names
1	(255,)	[0.5223200314277512]	0.52232	(TelephonyManager.*getDeviceId,)
2	(92, 255)	[0.6756324606714466]	0.675632	(SEND_SMS, TelephonyManager.*getDeviceId)
3	(92, 255, 280)	[0.7405149056465065]	0.740515	(SEND_SMS, TelephonyManager.*getDeviceId, Ljav...
4	(92, 229, 255, 280)	[0.7525297180932282]	0.75253	(SEND_SMS, Binder, TelephonyManager.*getDevice...
5	(92, 228, 229, 255, 280)	[0.7639235231961226]	0.763924	(SEND_SMS, IBinder, Binder, TelephonyManager.*...
...	...	...	...	...
81	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8854849084681463]	0.885485	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
82	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8857053731278064]	0.885705	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
83	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8859366134825668]	0.885937	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
84	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8861259435872642]	0.886126	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
85	(0, 6, 10, 14, 24, 29, 30, 34, 39, 49, 57, 58,...	[0.886344257219875]	0.886344	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...

85 rows × 4 columns

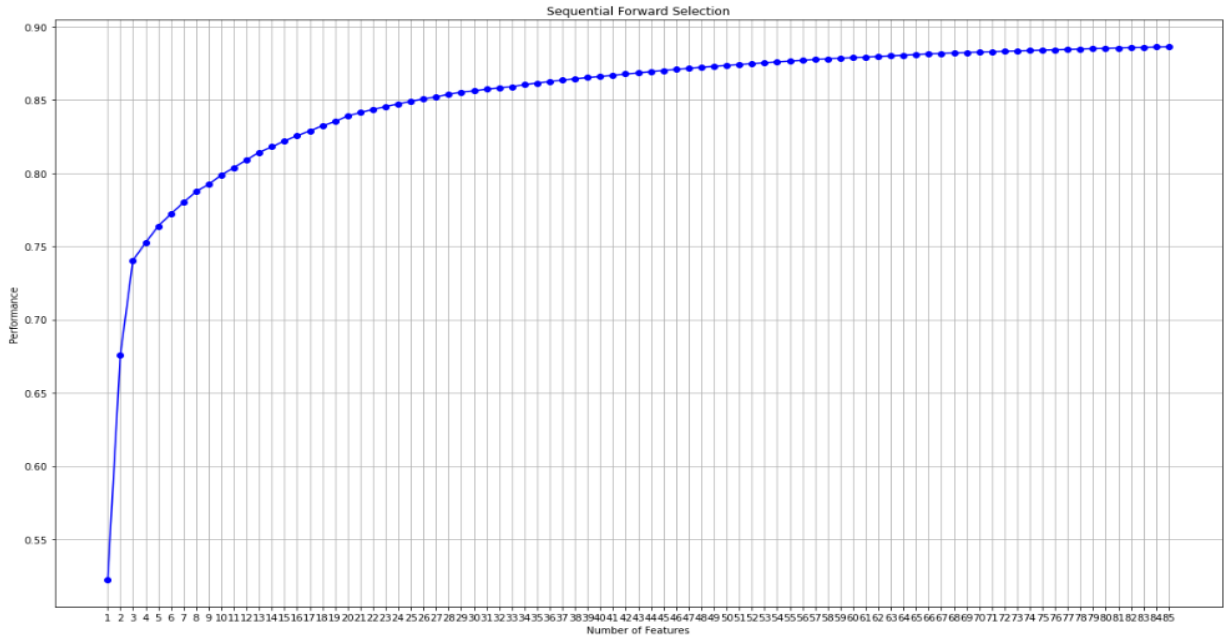


Figure 5.3.1: Top 85 features using Forward Feature Selection

Description

The above figure 5.3.1 represents, top 85 features selected using forward feature selection out off 342 features from ISCX Dataset.

5.3.2 Backward Feature Selection

	feature_idx	cv_scores	avg_score	feature_names
342	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9004281241641401]	0.900428	(ACCESS_CHECKIN_PROPERTIES, ACCESS_COARSE_LOCA...
341	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9004281327548341]	0.900428	(ACCESS_CHECKIN_PROPERTIES, ACCESS_COARSE_LOCA...
340	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9004281785472775]	0.900428	(ACCESS_CHECKIN_PROPERTIES, ACCESS_COARSE_LOCA...
339	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9004282098797457]	0.900428	(ACCESS_CHECKIN_PROPERTIES, ACCESS_COARSE_LOCA...
338	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...	[0.9004281371212516]	0.900428	(ACCESS_CHECKIN_PROPERTIES, ACCESS_COARSE_LOCA...
...	...	...	...	...
89	(0, 6, 10, 14, 24, 28, 29, 30, 32, 34, 57, 58,...	[0.8877294389355183]	0.887729	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
88	(0, 6, 10, 14, 24, 28, 29, 30, 32, 34, 57, 58,...	[0.8874978849360228]	0.887498	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
87	(0, 6, 10, 14, 24, 28, 29, 30, 32, 34, 57, 58,...	[0.8873328941909271]	0.887333	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
86	(0, 6, 10, 14, 24, 28, 29, 30, 32, 34, 57, 58,...	[0.8870558916113889]	0.887056	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
85	(0, 6, 10, 14, 24, 28, 29, 30, 32, 34, 57, 58,...	[0.8867991692041735]	0.886799	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...

258 rows × 4 columns

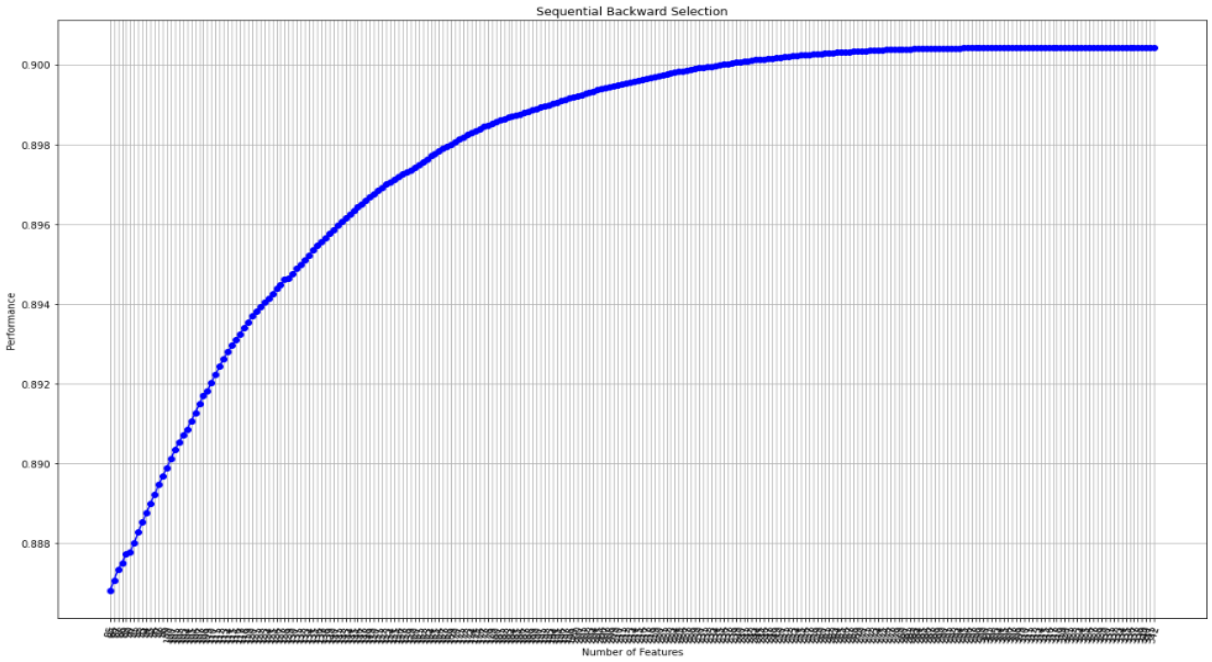


Figure 5.3.2: Top 85 features using Backward Feature Selection

Description

The above figure 5.3.2 represents, top 85 features selected using Backward feature selection out off 342 features from ISCX Dataset.

5.3.3 Stepwise Feature Selection

	feature_idx	cv_scores	avg_score	feature_names
1	(255,)	[0.5223200314277512]	0.52232	(TelephonyManager.*getDeviceId,)
2	(92, 255)	[0.6756324606714466]	0.675632	(SEND_SMS, TelephonyManager.*getDeviceId)
3	(92, 255, 280)	[0.7405149056465065]	0.740515	(SEND_SMS, TelephonyManager.*getDeviceId, Ljav...
4	(92, 229, 255, 280)	[0.7525297180932282]	0.75253	(SEND_SMS, Binder, TelephonyManager.*getDevice...
5	(92, 228, 229, 255, 280)	[0.7639235231961226]	0.763924	(SEND_SMS, IBinder, Binder, TelephonyManager.*...
...	...	...	...	...
81	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8854849084681463]	0.885485	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
82	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8857053731278064]	0.885705	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
83	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8859366134825668]	0.885937	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
84	(0, 6, 10, 14, 24, 29, 30, 39, 49, 57, 58, 62,...	[0.8861259435872642]	0.886126	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...
85	(0, 6, 10, 14, 24, 29, 30, 34, 39, 49, 57, 58,...	[0.886344257219875]	0.886344	(ACCESS_CHECKIN_PROPERTIES, ACCESS_SURFACE_FLI...

85 rows × 4 columns

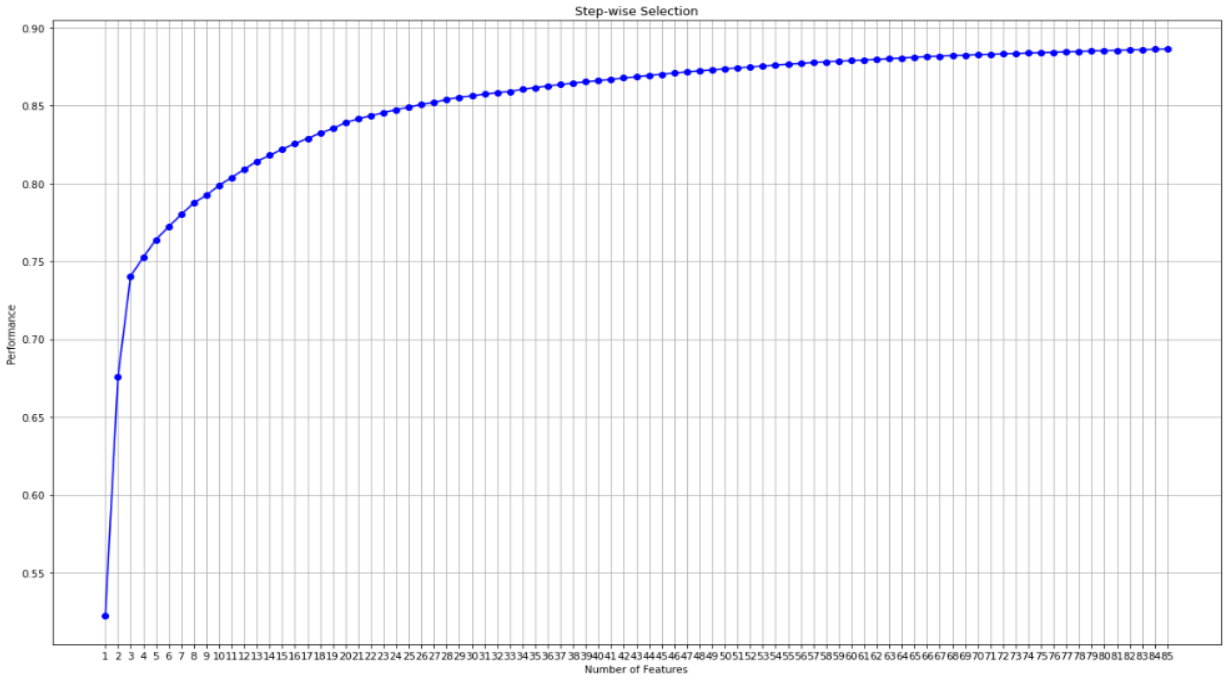


Figure 5.3.3: Top 85 features using Stepwise Feature Selection

Description

The above figure 5.3.3 represents, top 85 features selected using Stepwise feature selection out off 342 features from ISCX Dataset.

5.3.4 Train and Test Split after feature selection

The following figure represent training and testing splitting after complete Wrapper based Feature Selection Method.

```
from sklearn.model_selection import train_test_split
X = pd.DataFrame(df.iloc[:,:-1])
y = pd.DataFrame(df.iloc[:,-1])

# Splitting the dataset into train and test sets: 80-20 split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

```
((5441, 84), (5441, 1), (1361, 84), (1361, 1))
```

Figure 5.3.4: Training and Testing Splitting

Description

The above figure 5.3.4 represents Training and Testing Splitting. Top 85 features selected from Wrapper based Feature Selection Method and create another dataframe to split comprising for 80% train data and 20% for test data out off top 85 features and 6802 records from forward feature selection.

5.4 PHASE 4: MODEL BUILDING

5.4.1: SUPPORT VECTOR MACHINE

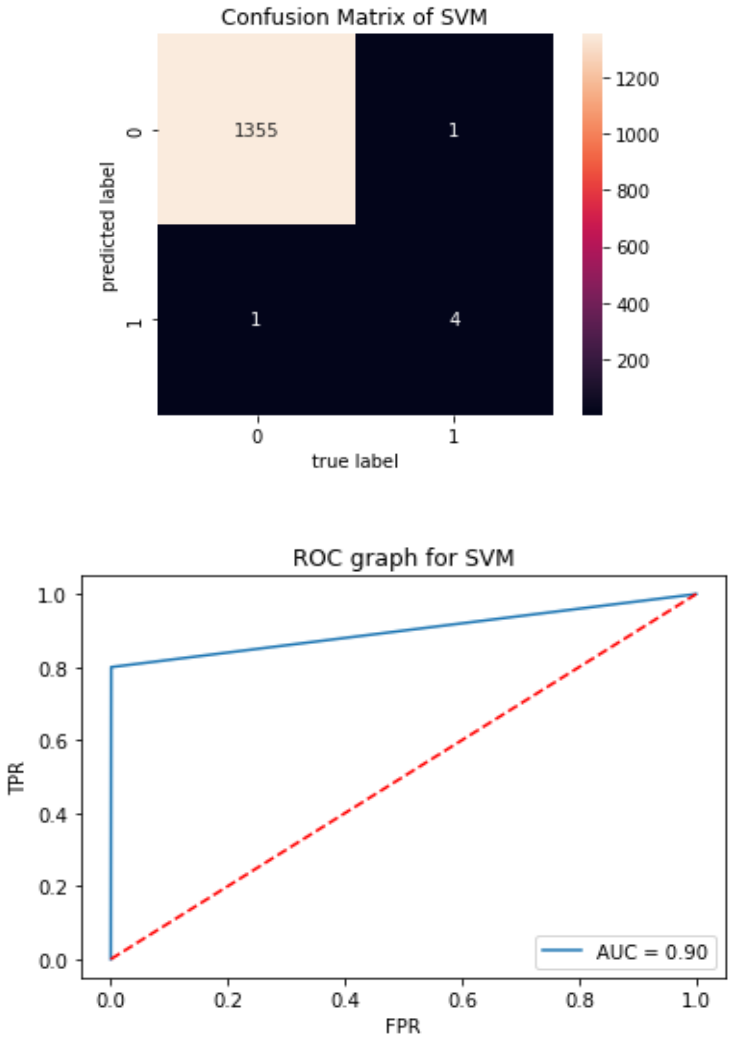


Figure 5.4.1: Confusion Matrix & ROC Curve using Support Vector Machine

Description

The above figure 5.4.1 represents Model Building SVM. This Matrix and ROC curve represents the performance model of the SVM classifier relationship between True positive and False positive to predicate model accuracy level.

5.4.2: NAVIE BAYES CLASSIFIER

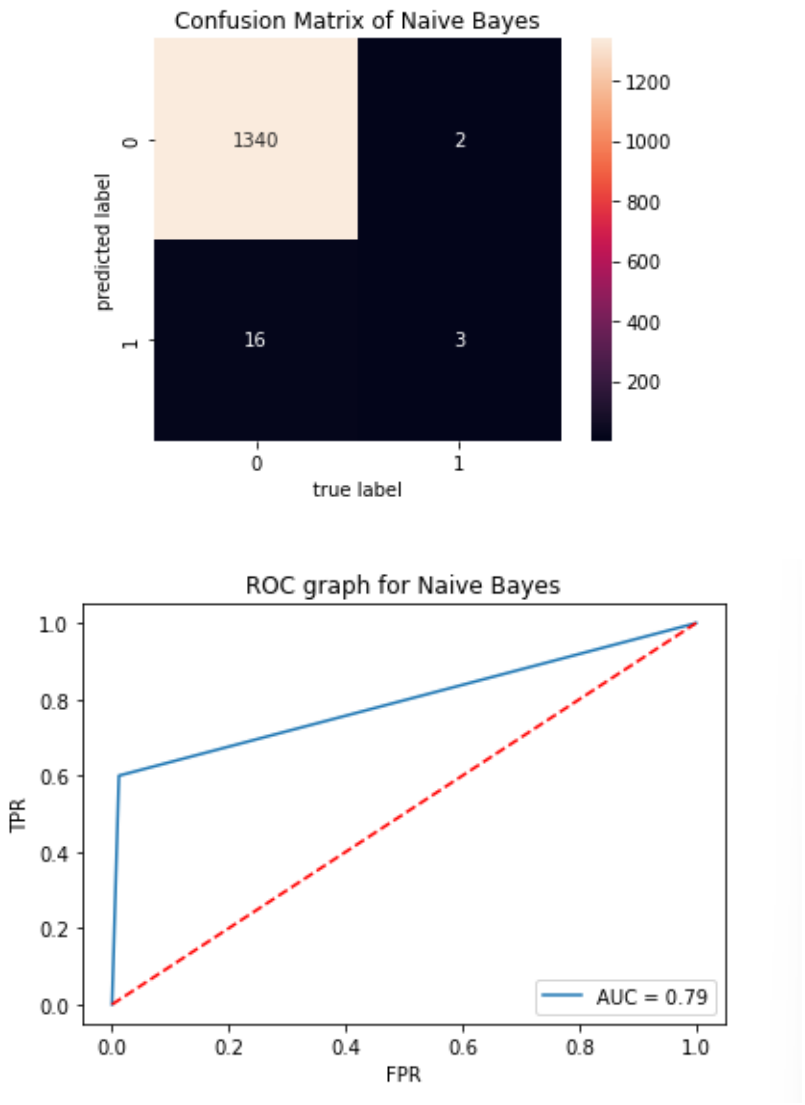


Figure 5.4.2: Confusion Matrix and ROC curve using Naive Bayes classifier.

Description

The above figure 5.4.2 represents Model Building Naive Bayes . This Matrix and ROC curve represents the performance model of the NB classifier relationship between True positive and False positive to predicate model accuracy level.

5.4.3: RANDOM FOREST CLASSIFIER

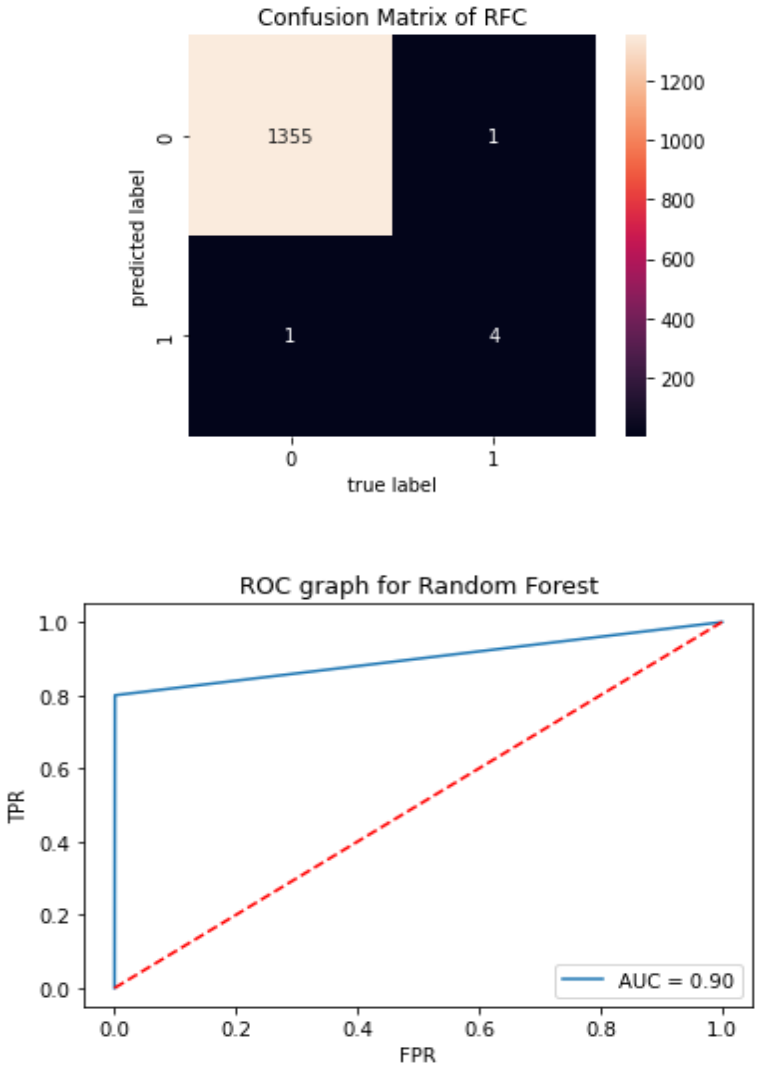


Figure 5.4.3: Confusion Matrix and ROC curve using Random Forest Classifier.

Description

The above figure 5.4.3 represents Model Building Random Forest. This Matrix and ROC curve represents the performance model of the Random forest classifier relationship between True positive and False positive to predicate model accuracy level.

5.4.4: DECISION TREE CLASSIFIER

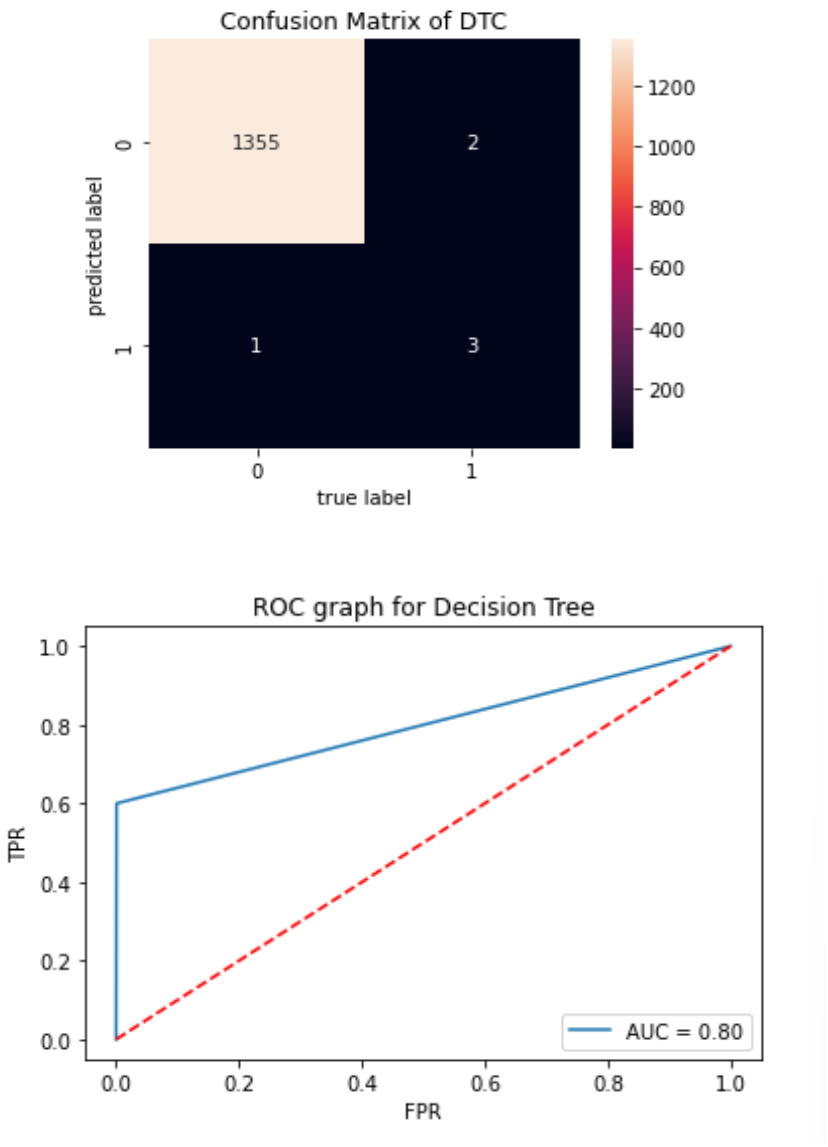


Figure 5.4.4: Confusion Matrix and ROC curve using Decision Tree Classifier.

Description

The above figure 5.4.4 represents Model Building Decision Tree. This Matrix and ROC curve represents the performance model of the Decision Tree classifier relationship between True positive and False positive to predicate model accuracy level.

5.4.5: MULTILAYER PERCEPTRON

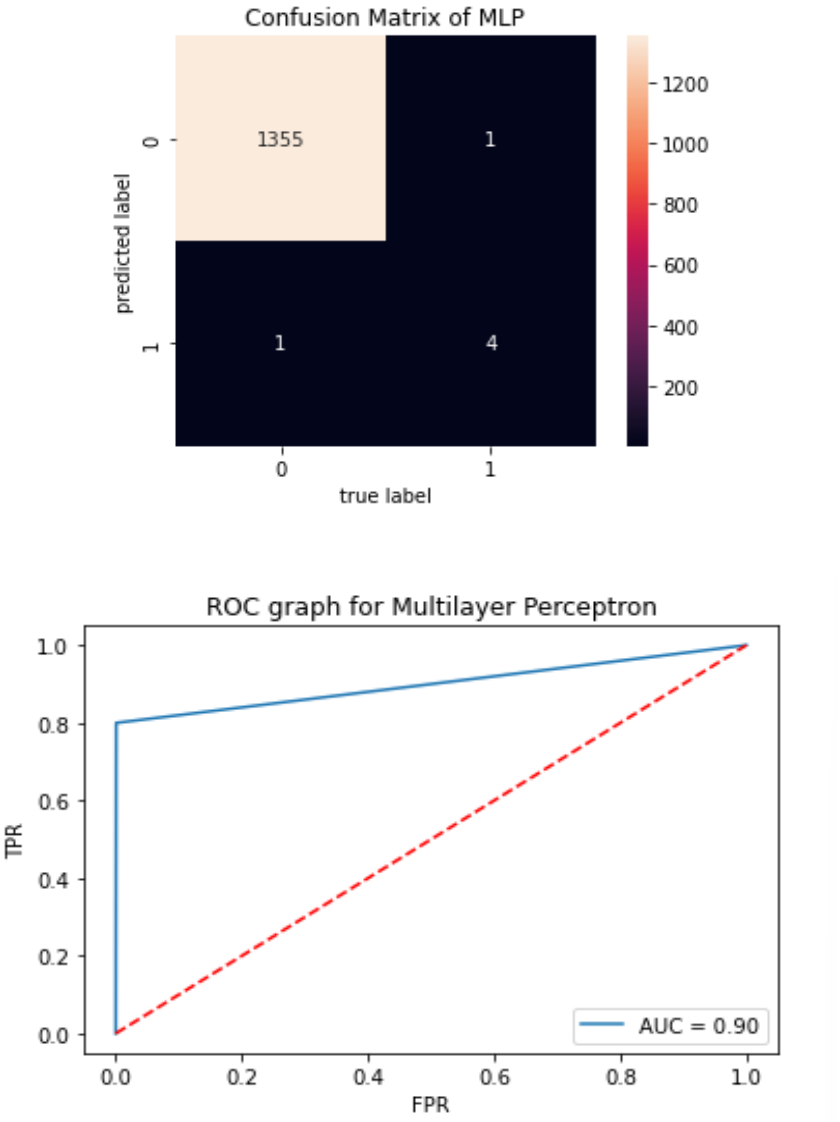


Figure 5.4.5: Confusion Matrix and ROC curve using Multilayer Perceptron (MLP).

Description

The above figure 5.4.5 represents Model Building MLP. This Matrix and ROC curve represents the performance model of the MLP classifier relationship between True positive and False positive to predicate model accuracy level.

5.5 PHASE 5: COMPARATIVE ANALYSIS

5.5.1 Model Train and Test Accuracy Score Comparison:

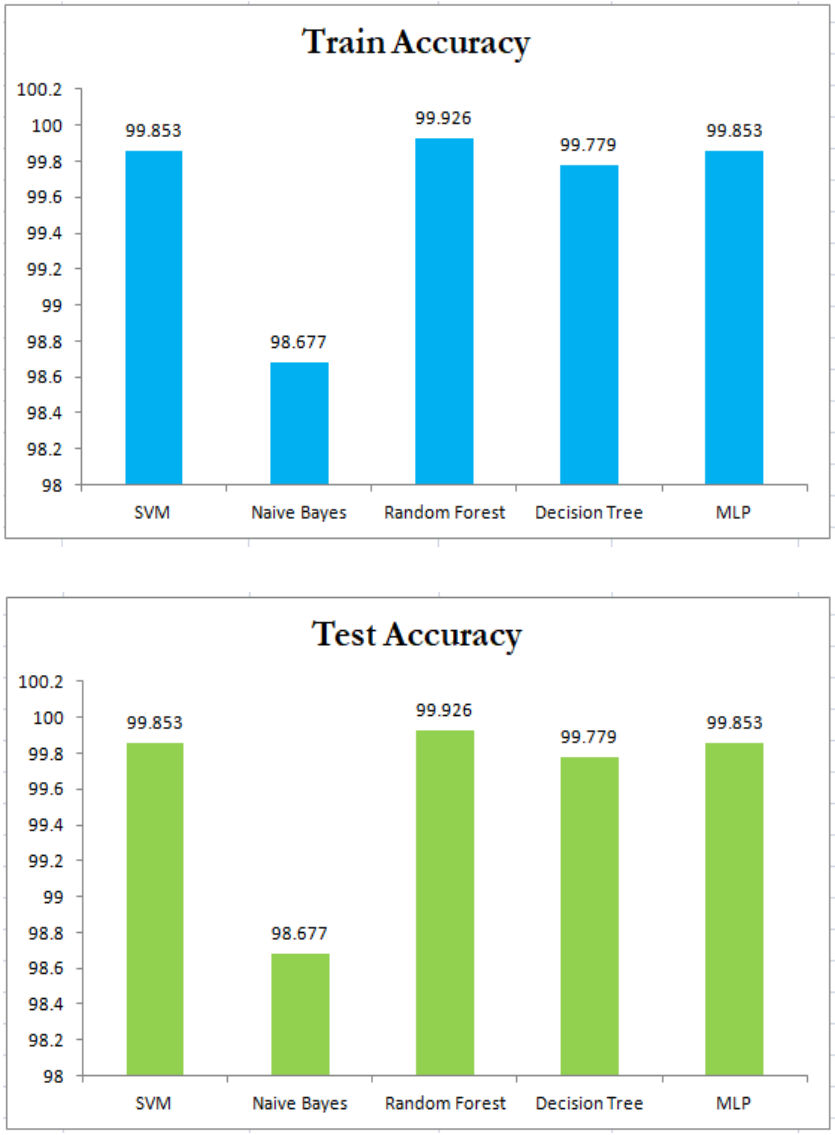


Figure 5.5.1: Model Train and Test Accuracy Comparison.

Description

The above figure 5.5.1 represents all the five Supervised Machine Learning classifier Model Train and Test accuracy Comparisons.

5.5.2 Model Precision and Recall Score Comparison

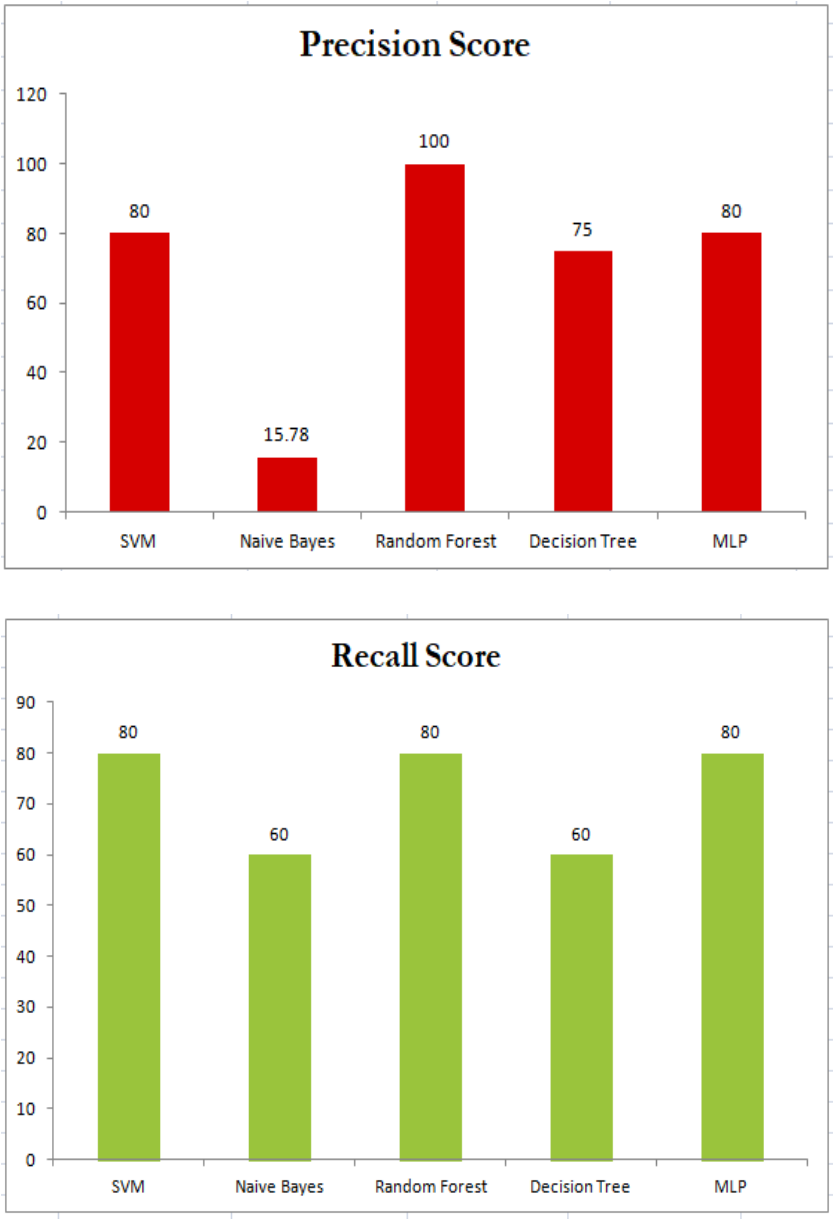
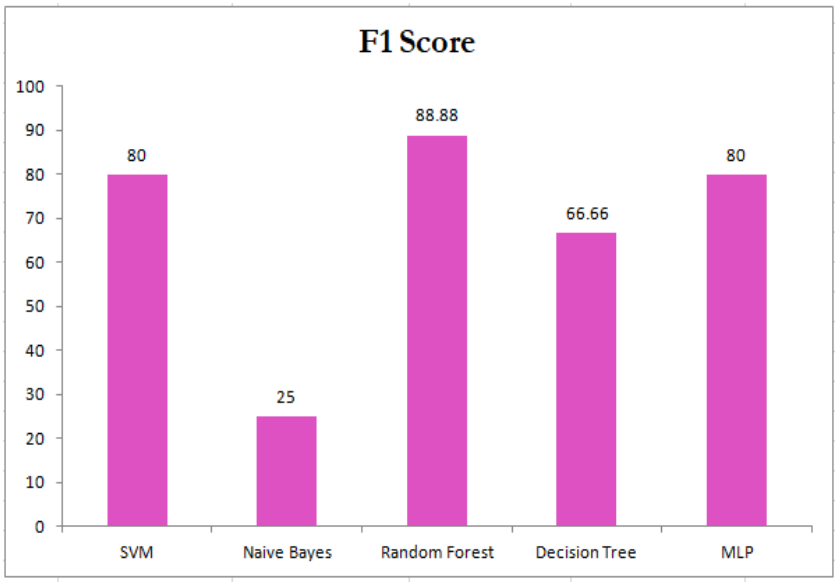


Figure 5.5.2: Model Precision and Recall Score Comparative Analysis.

Description

The above figure 5.5.2 represents all the five Supervised Machine Learning classifier Model Precision and Recall accuracy Comparisons.

5.5.3 Model F1 and AUC Score Comparison



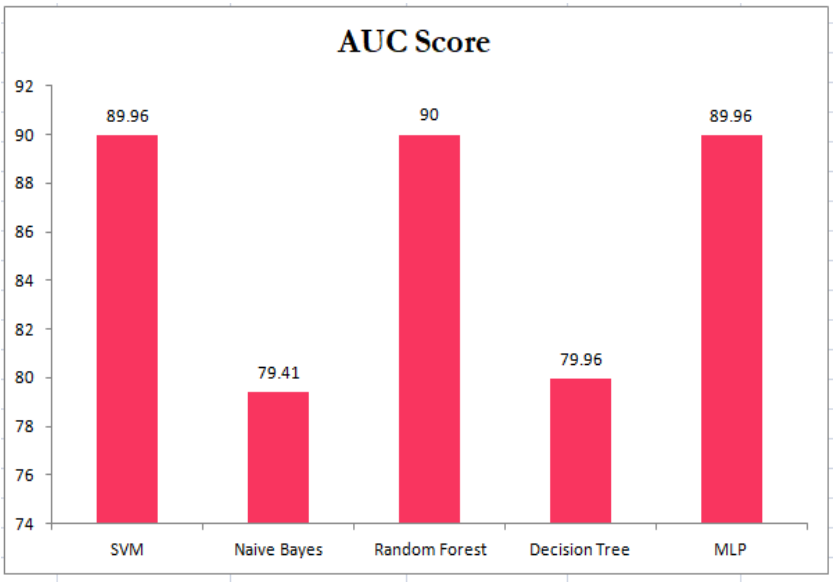


Figure 5.5.3: Model F1 and AUC Score Comparative Analysis.

Description

The above figure 5.5.3 represents all the five Supervised Machine Learning classifier Model F1 Score and AUC Score Comparisons.

5.5.4 Model ROC Curve and Mean Absolute Error Comparison

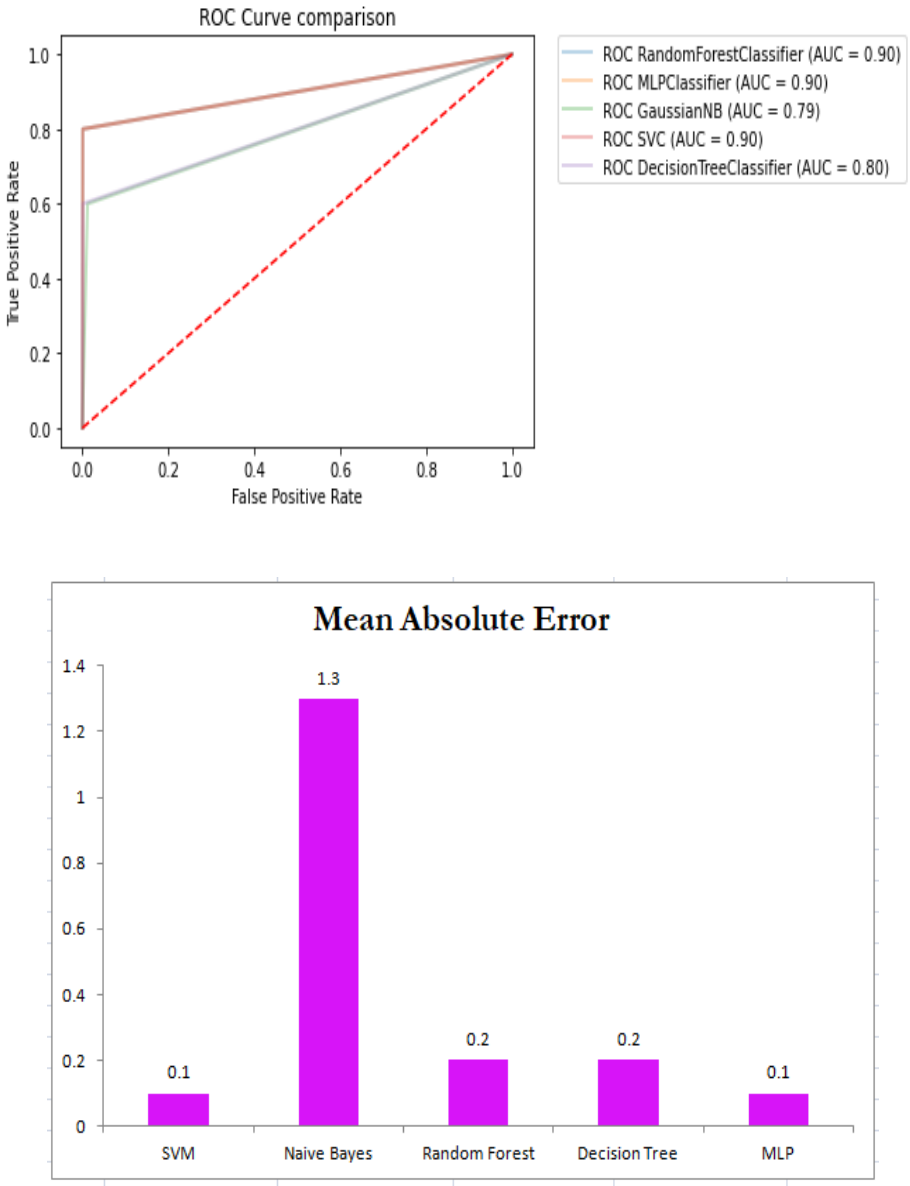


Figure 5.5.4: ROC Curve and Model Mean Absolute Error Comparison.

Description

The above figure 5.5.4 represents all the five Supervised Machine Learning classifier ROC Curve and Mean Absolute Error Comparisons.

5.5.5 Model Mean Squared Error and Root Mean Squared Error Comparison

Figure 5.5.5: Model Root Mean Square Error Comparison

Description

The above figure 5.5.5 represents all the five Supervised Machine Learning classifier Model Root Mean Squared Error Comparisons.

SCOPE FOR FUTURE DEVELOPMENT

CHAPTER – 6

SCOPE FOR FUTURE DEVELOPMENT

As devices becomes more popular, it become increasingly vulnerable to malicious. With the openness of the Android OS's design and its rising influence, Android malware is likely to become more prevalent. Checking the permissions of Android applications and being constantly aware of the devices behaviour and any unexpected behaviours are two further defensive tactics that a person might use. So, using different set of machine learning algorithms which is unsupervised and also deep learning algorithms, the model can be enhanced in a way such attacks can be reduced to minimum.

CONCLUSION

CHAPTER - 7

CONCLUSION

With the expanding number of Internet users and its commercial aspect, a corresponding number of crimes enter the market, posing a threat to authorized customers, Network infrastructure, and the timeliness of services offered by it. The point of this study worked and detected the android botnet attack, based on its classes. Here the Random Forest, SVM and MLP worked better than Decision tree and Naïve Bayes classifiers using top 85 features. While training the model, Random forest detect 99.92% of android botnet in top 85 feature.

The other algorithm fails to detect most of the android botnet. Thus it shows that random forest model is most sufficient than other algorithm in detecting the malware attack at top X feature. Thus, the Android Botnets are detected and classified, which helps in detecting the attacks that are trying to access the user control.

REFERENCES

CHAPTER - 8

REFERENCES

1. Ahmad Karim, Syed Adeel Ali Shah, Rosli Bin Salleh, Muhammad Arif, Rafidah Md Noor, Shahaboddin Shamshirband, (2015), *Mobile Botnet Attacks – an Emerging Threat: Classification,* Review and Open Issues. KSII Transactions on Internet and Information Systems, 9(4). DOI:10.3837/tiis.2015.04.012

2. Alharbi, A., & Alsubhi, K. (2021*). Botnet Detection Approach Using Graph-Based Machine Learning.* IEEE Access, 9, 99166–99180. DOI:10.1109/access.2021.3094183

3. FarhanTariq, Shamim Baig (2020), *Comparative Analysis of Network flow-based Botnet Detection Methods Using Supervised Machine Learning Algorithms*. International Journal of Advanced Trends in Computer Science and Engineering, 9(5), 8498–8503. DOI:10.30534/ijatcse/2020/229952020

4. Hashim, H. A. B., Saudi, M. M., & Basir, N. (2017a). *Android Botnet Features for Detection Mechanism*. Advanced Science Letters, *23*(6), 5314–5317. DOI.10.1166/asl.2017.7366

5. Hijawi, W., Alqatawna, J., Al-Zoubi, A. M., Hassonah,M. A., & Faris, H. (2021). *Android botnet detection using machine learning models based on a comprehensive static analysis approach.* Journal of Information Security and Applications, 58, 102735. DOI:10.1016/j.jisa.2020.102735

6. Kumar, K. (2021). *Comprehensive Method of Botnet Detection Using Machine Learning*. International Journal of Open Source Software and Processes, 12(4), 37–61. DOI:10.4018/ijossp.287613

7. Rasheed, M. M., Faieq, A. K., & Hashim, A. A. (2020). *Android Botnet Detection Using Machine Learning.* Ingénierie Des Systèmes d Information, 25(1), 127–130. DOI:10.18280/isi.250117

8. Shahid Anwar, Mohamad Fadli Zolkipli, Vitaliy Mezhuyev, Zakira Inayat, (2020), *A Smart Framework for Mobile Botnet Detection Using Static Analysis*. KSII Transactions on Internet and Information Systems, 14(6). DOI:10.3837/tiis.2020.06.015

9. Shatnawi, Ahmed S., et al. "*An Android Malware Detection Approach Based on Static Feature Analysis Using Machine Learning Algorithms*." Procedia Computer Science, vol. 201, 2022, pp. 653–58. *Crossref*, DOI.10.1016/j.procs.2022.03.086.

10. Shatnawi, A. S., Yassen, Q., & Yateem, A. (2022). *An Android Malware Detection Approach Based on Static Feature Analysis Using Machine Learning Algorithms.* Procedia Computer Science, 201, 653–658. DOI:10.1016/j.procs.2022.03.086

11. Yerima, S. Y., Alzaylaee, M. K., Shajan, A., & P, V. (2021). *Deep Learning Techniques for Android Botnet Detection.* Electronics, 10(4), 519. DOI:10.3390/electronics10040519

APPENDIX

CHAPTER - 9

APPENDIX

9.1 SAMPLE CODING:

```
#Import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn import metrics

#reading and display the dataset file
dataset = pd.read_csv('iscx_botnets.csv')
dataset
# finding null values in dataset
dataset.isnull()

#label encoding
from sklearn.preprocessing import LabelEncoder
Label_Encoder = LabelEncoder()
dataset["Result"] = Label_Encoder.fit_transform(dataset["Result"])
dataset

# label_encoder object knows how to understand word labels.
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
dataset['Result']= label_encoder.fit_transform(dataset['Result'])
```

```
dataset['Result'].unique()
dataset.columns


#Feature scaling using Normalization
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()

dataset=pd.DataFrame(scaler.fit_transform(dataset),
        columns=dataset.columns, index=dataset.index)
print(dataset)

#Feature selection using Forward based method
from mlxtend.feature_selection import SequentialFeatureSelector as SFS
from sklearn.linear_model import LinearRegression
sfs = SFS(LinearRegression(),
        k_features=85,
        forward=True,
        floating=False,
        scoring = 'r2',
        cv = 0)
#Use SFS to select the top 85 features
sfs.fit(X, y)
sfs.k_feature_names_

from sklearn.model_selection import train_test_split
X = pd.DataFrame(df.iloc[:,:-1])
y = pd.DataFrame(df.iloc[:,-1])

# Splitting the dataset into train and test sets: 80-20 split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape


from math import sqrt
from sklearn import ensemble
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import precision_score, recall_score, f1_score, auc,roc_curve
from sklearn import svm,model_selection, tree, neural_network, neighbors, naive_bayes,
ensemble, discriminant_analysis, gaussian_process
```

#declare Machine Learning Classifiers

```
ML_Model = [
    #Ensemble Methods
    ensemble.RandomForestClassifier(),

    #GLM
    neural_network.MLPClassifier(),

    #Navies Bayes
    naive_bayes.GaussianNB(),

    #SVM
    svm.SVC(),
    tree.DecisionTreeClassifier()
    ]
```

#Calculating Training accuarcy, testing accuracy, precision score, recall score and auc

```
ML_Model_columns = []
ML_Model_compare = pd.DataFrame(columns = ML_Model_columns)

row_index = 0
for alg in ML_Model:

    predicted = alg.fit(X_train, y_train).predict(X_test)
    fp, tp, th = roc_curve(y_test, predicted)
    ML_Model_name = alg.__class__.__name__
    ML_Model_compare.loc[row_index,'ML_Model Name'] = ML_Model_name
    ML_Model_compare.loc[row_index, 'ML_Model Train Accuracy'] = accuracy_score(y_test,
predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Test Accuracy'] = accuracy_score(y_test,
predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Precision'] = precision_score(y_test,
predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Recall'] = recall_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model F1 Score'] = f1_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model AUC'] = auc(fp, tp)

    row_index+=1

ML_Model_compare.sort_values(by = ['ML_Model Test Accuracy'], ascending = False, inplace
= True)
ML_Model_compare
```

9.2 SCREENSHOTS:

```
#Import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn import metrics
```

```
#reading and display the dataset file
dataset = pd.read_csv('iscx_botnets.csv')
dataset
```

Figure 9.2.1 Import Python Packages

```
#split data into training sets and test sets
from sklearn.model_selection import train_test_split
X = pd.DataFrame(dataset.iloc[:,:-1])
y = pd.DataFrame(dataset.iloc[:,-1])
```

```
from sklearn.model_selection import train_test_split
X = pd.DataFrame(df.iloc[:,:-1])
y = pd.DataFrame(df.iloc[:,-1])

# Splitting the dataset into train and test sets: 80-20 split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Figure 9.2.2 Training and Testing Splitting after feature selection

```
from math import sqrt
from sklearn import ensemble
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
from sklearn.metrics import precision_score, recall_score, f1_score, auc,roc_curve
from sklearn import svm,model_selection, tree, neural_network, neighbors, naive_bayes, ensemble, discriminant_analysis, gaussian_
```

```
#declare Machine Learning Classifiers

ML_Model = [
    #Ensemble Methods
    ensemble.RandomForestClassifier(),

    #GLM
    neural_network.MLPClassifier(),

    #Navies Bayes
    naive_bayes.GaussianNB(),
```

Figure 9.2.3 Model Building using Supervised Machine learning Algorithms

```
    #SVM
    svm.SVC(),
    tree.DecisionTreeClassifier()
    ]


#Calculating Training accuarcy, testing accuracy, precision score, recall score and auc

ML_Model_columns = []
ML_Model_compare = pd.DataFrame(columns = ML_Model_columns)

row_index = 0
for alg in ML_Model:

    predicted = alg.fit(X_train, y_train).predict(X_test)
    fp, tp, th = roc_curve(y_test, predicted)
    ML_Model_name = alg.__class__.__name__
    ML_Model_compare.loc[row_index,'ML_Model Name'] = ML_Model_name
    ML_Model_compare.loc[row_index, 'ML_Model Train Accuracy'] = accuracy_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Test Accuracy'] = accuracy_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Precision'] = precision_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model Recall'] = recall_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model F1 Score'] = f1_score(y_test, predicted)
    ML_Model_compare.loc[row_index, 'ML_Model AUC'] = auc(fp, tp)

    row_index+=1

ML_Model_compare.sort_values(by = ['ML_Model Test Accuracy'], ascending = False, inplace = True)
ML_Model_compare
```

```
#ML Model Training accuarcy, testing accuracy, precision score, recall score and auc Comparison
plt.subplots(figsize=(15,6))
sns.barplot(x="ML_Model Name", y="ML_Model Train Accuracy",data=ML_Model_compare,palette='Set3',edgecolor=sns.color_palette('dark
plt.xticks(rotation=90)
plt.title('ML Model Train Accuracy Comparison')
plt.show()

plt.subplots(figsize=(15,6))
sns.barplot(x="ML_Model Name", y="ML_Model Test Accuracy",data=ML_Model_compare,palette='Set2',edgecolor=sns.color_palette('dark'
plt.xticks(rotation=90)
plt.title('ML Model Test Accuracy Comparison')
plt.show()

plt.subplots(figsize=(15,6))
sns.barplot(x="ML_Model Name", y="ML_Model Precision",data=ML_Model_compare,palette='Pastel2',edgecolor=sns.color_palette('dark',
plt.xticks(rotation=90)
plt.title('ML Model Precission Comparison')
plt.show()

plt.subplots(figsize=(15,6))
sns.barplot(x="ML_Model Name", y="ML_Model Recall",data=ML_Model_compare,palette='Accent',edgecolor=sns.color_palette('dark',7))
plt.xticks(rotation=90)
plt.title('ML Model Recall Comparison')
plt.show()

plt.subplots(figsize=(15,6))
sns.barplot(x="ML_Model Name", y="ML_Model F1 Score",data=ML_Model_compare,palette='Set3',edgecolor=sns.color_palette('dark',7))
plt.xticks(rotation=90)
plt.title('ML Model F1_Score Comparison')
plt.show()
```

Figure 9.2.4 Performances Evaluation using Supervised Machine learning Algorithms