Malware Detection Using Machine Learning

A Report submitted

by

Sarang Pratap Chamola(170102083)

Shubham Negi(170102088)

Shubham Chauhan(170102093)

Prashant Garg(170102106)

Under the Guidance of

Mr. Sanjeev Sharma

Assistant Professor – CSE Department

In partial fulfilment of the requirements for the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING,

DIT UNIVERSITY, DEHRADUN

(State Private University through State Legislature Act No. 10 of 2013 of Uttarakhand and approved by UGC)

Mussoorie Diversion Road, Dehradun, Uttarakhand - 248009, India.

CANDIDATE/S DECLARATION

I hereby certify that the work, which is being presented in the report/ project report, entitled **Title of the Report**, in partial fulfillment of the requirement for the award of the Degree of **Bachelor of Technology** and submitted to the university is an authentic record of my/our own work carried out during the period *Month-Year* to *Month-Year* under the supervision of supervisor(s) name.

Date: Signature of the Candidate

This is to certify that the above statement made by the candidate is correct to the best of my /our knowledge.

Date: Signature(s) of the Supervisor (s)

TABLE OF CONTENTS

Chapter 1 Introduction

1.1 PURPOSE

The most threatening problem is, of the different malware that can attack and harm your system. Malware is one of the most serious security threats on the Internet today. In fact, most Internet problems such as spam e-mails and denial of service attacks have malware as their underlying cause. That is, computers that are compromised with malware are often networked together to form botnets, and many attacks are launched using these malicious, attacker-controlled networks.

In order to deal with the new malware generated, new techniques to detect them and prevent any damage caused by them.

1.2 Objective

To investigate on how to implement machine learning to malware detection in order to detection unknown malware. To develop a malware detection software that implement machine learning to detect unknown malware. To validate that malware detection that implement machine learning will be able to achieve a high accuracy rate with low false positive rate.

1.3 Overview

Traditional security product uses virus scanner to detect malicious code, these scanner uses signature which created by reverse engineering a malware. But with malware that became polymorphic or metamorphic the traditional signature-based detection method used by anti-virus is no long effective against the current issue of malware (Willems, G., Holz, T. & Freiling, F., 2007). In current anti-malware products, there are two main task to be carried out from the malware analysis process, which are malware detection and malware classification. In this paper, I am focusing on malware detection. The main objective of malware detection is to be able to detect malware in the system. There are two type of analysis for malware detection which are dynamic analysis and static analysis. For effective and efficient detection, the uses of feature extraction are recommended for malware detection (Ahmadi, M. et al., 2016). There are various type of detection method, the method that we are using will be detecting through hex and assembly file of the malware. Feature will be extracted from both hex view and assembly

view of malware files. After extracting feature to its category, all category is to be combine into one feature vector for the classifier to run on them (Ahmadi, M. et al., 2016). For feature selection, separating binary file into blocks to be compare the similarities of malware binaries. This will reduce the analysis overhead which cause the process to be faster (Kim, T.G., Kang, B. & Im, E.G., 2013). To build a learning algorithm, feature that are extracted with the label will be undergo classification with using any classification method for example Random Forest, Neural Network, N-gram, KNN and many others, but Support Vector Machine (VCM) is recommended for the presence of noise in the extracted feature and the label (Stewin, P. & Bystrov, I., 2016). As to generate result, the learning model is to test with dataset with label to generate a graph which indicate detection rate and false positive rate. To find the best result, repeat the process using many other classification and create learning model to test on the same dataset. The best result will the one graph that has the highest detection rate and lowest false positive rates (Lanzi, A. et al., 2010).

Chapter 3 System Requirements

2.1 Hardware interface

Device : Laptop, Smart Phones or Desktop Computer

Processor : core i3 3rd Gen (minimum) and above

RAM : 4GB(minimum) and above

Hard disk : 100 GB (minimum) and above

2.2 Software Requirements

Operating System: Windows, Linux – Ubuntu

Platforms: Jupyter, Spyder, Google Collab, Anaconda prompt, Virtual Box

Languages: Python

Web browsers: Chrome, Firefox

Chapter 3 Overall Description

3.1 Project Perspective

The idea of machine learning is to let the algorithm learn by itself the best parameters from data in order to make good predictions. There are many different applications, in our case we will consider using machine learning algorithm to classify binaries between legitimate and malicious binaries.

3.2 Project function

3.2.1 Initialization

The dataset is loaded from the file and is saved in memory.

```
import pandas as pd

import matplotlib as plt

import numpy as np

df=pd.read_csv('MalwareData.csv',sep='|')
```

3.2.2 Feature Extraction

Since every feature is in numeric form, therefore there is no need for this particular step.

3.2.3 Feature Selection

Since the Column “Name” only signifies the name of the file and Column “md5” is a hash function. So these values won’t affect the training model and therefore we can drop them.

```
df=df.drop(['Name','md5'],axis=1)
```

3.2.4 Splitting the data

```
X=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

Original data :

```
In [71]: df.shape
Out[71]: (138047, 55)
```

Training data :

```
In [73]: X_train.shape
Out[73]: (110437, 54)
```

Test data :

```
In [74]: X_test.shape
Out[74]: (27610, 54)
```

3.2.5 Data Pre-processing

1. Standard Scalar

Standardize features by removing the mean and scaling to unit variance

The standard score of a sample `x` is calculated as:

z = (x - u) / s

where `u` is the mean of the training samples or zero if `with_mean=False`, and `s` is the standard deviation of the training samples or one if `with_std=False`.

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using `transform`.

Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data.

from sklearn.preprocessing import StandardScaler

SS = StandardScaler()

Train = SS.fit_transform(Train)

3.2.5 Applying Classification Models

1. Logistic Regression

Logistic regression is a machine learning algorithm for classification. In this algorithm, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function.

Advantages: Logistic regression is designed for this purpose (classification), and is most useful for understanding the influence of several independent variables on a single outcome variable.

Disadvantages: Works only when the predicted variable is binary, assumes all predictors are independent of each other, and assumes data is free of missing values.

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()

lr.fit(X_train,y_train)

y_pred=lr.predict(X_test)
```

2. Naïve Bayes

Naive Bayes algorithm based on Bayes' theorem with the assumption of independence between every pair of features. Naive Bayes classifiers work well in many real-world situations such as document classification and spam filtering.

Advantages: This algorithm requires a small amount of training data to estimate the necessary parameters. Naive Bayes classifiers are extremely fast compared to more sophisticated methods.

Disadvantages: Naive Bayes is is known to be a bad estimator.

```
from sklearn.naive_bayes import GaussianNB

nb = GaussianNB()

nb.fit(X_train,y_train)
```

```
y_pred1 = nb.predict(X_test)
```

3. K-Nearest Neighbor

Neighbors based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbours of each point.

Advantages: This algorithm is simple to implement, robust to noisy training data, and effective if training data is large.

Disadvantages: Need to determine the value of K and the computation cost is high as it needs to computer the distance of each instance to all the training samples.

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
y_pred2=knn.predict(X_test)
```

4. Decision Tree

Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

Advantages: Decision Tree is simple to understand and visualise, requires little data preparation, and can handle both numerical and categorical data.

Disadvantages: Decision tree can create complex trees that do not generalise well, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated.

```
from sklearn.tree import DecisionTreeClassifier

tr = DecisionTreeClassifier()
```

```
tr.fit(X_train,y_train)

y_pred3=tr.predict(X_test)
```

5. Random Forest

Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.

Advantages: Reduction in over-fitting and random forest classifier is more accurate than decision trees in most cases.

Disadvantages: Slow real time prediction, difficult to implement, and complex algorithm.

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf.fit(X_train,y_train)
y_pred4=rf.predict(X_test)
```

Chapter 4 Conclusion

4.1 Conclusion

The desired feature extraction and representation methods were selected and the selected machine learning algorithms were applied and evaluated. The results of the Classification models used above is shown below.

1. Logistic Regression

```
Confusion Matrix :
[[19127   229]
 [  305  7949]]
Accuracy Score : 0.9806591814559942
Report :
              precision    recall  f1-score   support

           0       0.98      0.99      0.99     19356
           1       0.97      0.96      0.97      8254

    accuracy                           0.98     27610
   macro avg       0.98      0.98      0.98     27610
weighted avg       0.98      0.98      0.98     27610
```

2. Naïve Bayes

```
Confusion Matrix :
[[ 4822 14534]
 [   19  8235]]
Accuracy Score : 0.47290836653386453
Report :
              precision    recall  f1-score   support

           0       1.00      0.25      0.40     19356
           1       0.36      1.00      0.53      8254

    accuracy                           0.47     27610
   macro avg       0.68      0.62      0.46     27610
weighted avg       0.81      0.47      0.44     27610
```

3. K-Nearest Neighbor

```
Confusion Matrix :
[[19202   154]
 [  103  8151]]
Accuracy Score : 0.9906917783411807
Report :
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     19356
           1       0.98      0.99      0.98      8254

    accuracy                           0.99     27610
   macro avg       0.99      0.99      0.99     27610
weighted avg       0.99      0.99      0.99     27610
```

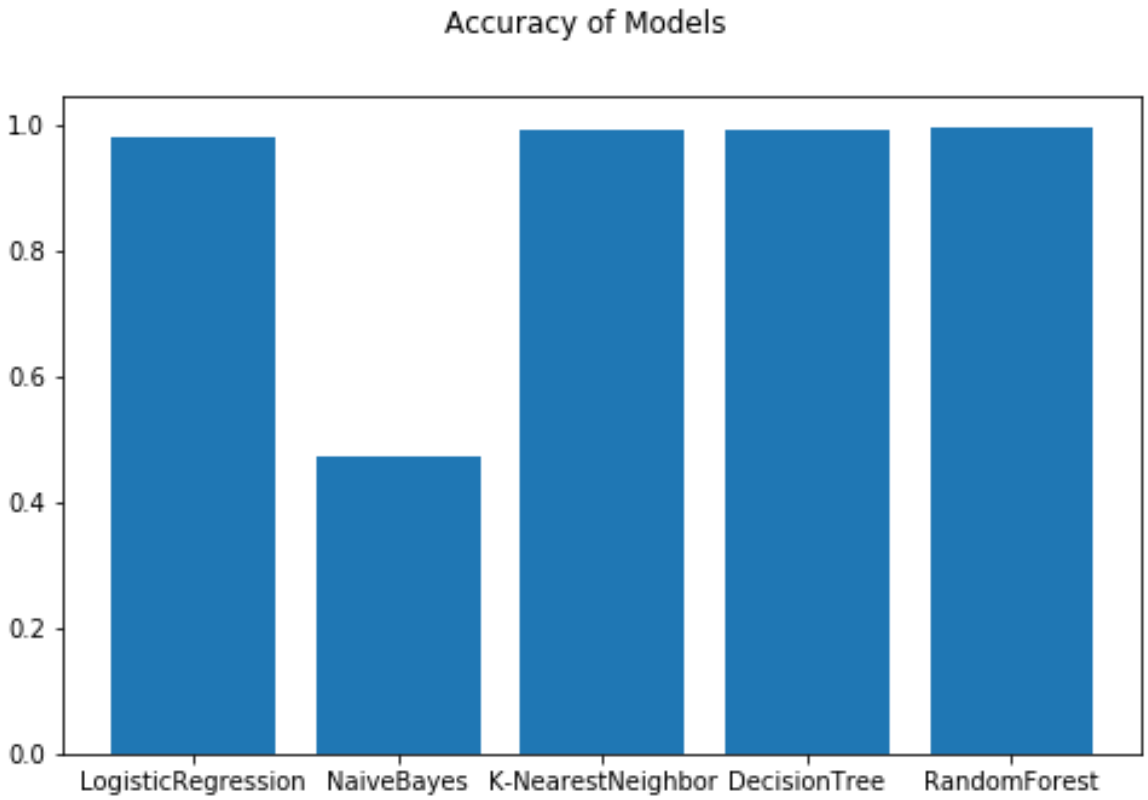
4. Decision Tree

```
Confusion Matrix :
[[19245   111]
 [  100  8154]]
Accuracy Score : 0.9923578413618255
Report :
              precision    recall  f1-score   support

           0       0.99      0.99      0.99     19356
           1       0.99      0.99      0.99      8254

    accuracy                           0.99     27610
   macro avg       0.99      0.99      0.99     27610
weighted avg       0.99      0.99      0.99     27610
```

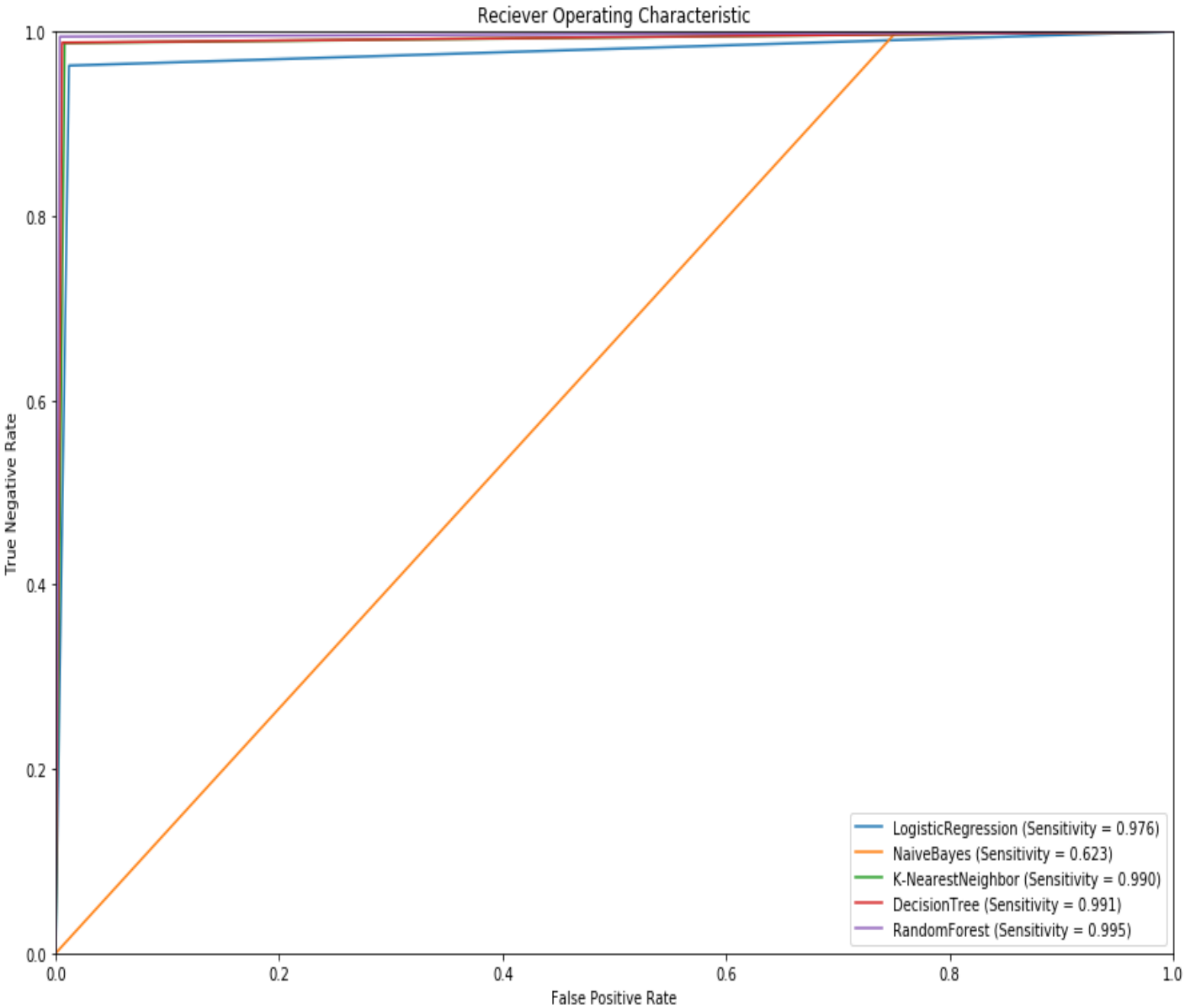
5. Random Forest

```
Confusion Matrix :
[[19286    70]
 [   46  8208]]
Accuracy Score : 0.99579862368707
Report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     19356
           1       0.99      0.99      0.99      8254

    accuracy                           1.00     27610
   macro avg       0.99      1.00      0.99     27610
weighted avg       1.00      1.00      1.00     27610
```

6. Comparison of Models



7. ROC Curve


Reciever Operating Characteristic
True Negative Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
LogisticRegression (Sensitivity = 0.976)
NaiveBayes (Sensitivity = 0.623)
K-NearestNeighbor (Sensitivity = 0.990)
DecisionTree (Sensitivity = 0.991)
RandomForest (Sensitivity = 0.995)

From the results above its clear that all models except Naïve Bayes are showing the best results on the given data. While Naïve Bayes shows poor result because of its probability function.

4.2 Future Scope

1. Advanced Preprocessing

   Till now the preprocessing used is of basic ground level. Further advanced preprocessing techniques like Normalization, Encoding and Dimensionality Reduction (e.g. PCA) can be applied on the dataset.

2. Use a wider dataset

   Currently we have used the data which is in csv format. Now will be using data which is in image format and in ASM & BYTES file format.

   Also the dataset that was used in this study is broad, covering most of the malware types that are relevant to the modern world, it does not cover all possible types. Collecting a malware dataset is a tedious task that requires a lot of time and effort. For more accurate evaluation of the predictors, it is advised to test the models on all the possible types of malware: spyware, adware, rootkits, backdoor, banking malware, etc. In addition to that, it is important to understand that the model will only be able to predict the samples of the families that it has seen earlier. In other words, in a real-world application, the maximum amount of possible families should be used before the launch of the project for real-world environments.

References

1.https://www.blackhat.com/docs/us-17/thursday/us-17-Anderson-Bot-Vs-Bot-Evading-Machine-Learning-Malware-Detection-wp.pdf

2. https://en.wikipedia.org/wiki/Malware https://www.kaggle.com/

3.https://www.infosecurity-magazine.com/opinions/malware-detection-signature4.

4.https://www.kaggle.com/

5. https://towardsdatascience.com/