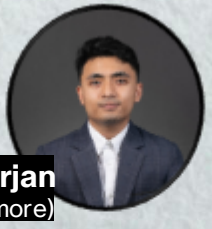
Poster Presentation

Student Research Symposium 2021

AI and Malware Signature

Use of Common Machine Learning Algorithms malware analysis and Artificial intelligence in Intrusion Detection System in Networks

Sijan Malla
Computer Science (Sophomore)



Pratik Maharjan
Computer Science (Sophomore)

University of Louisiana Monroe
School of Business and Social Sciences
BS in Computer Science

Research Introduction

Malware:
They are malicious software designed with intention to harm computer physically, its operation or performance or to steal or corrupt data or to keep tap on other people or organization.

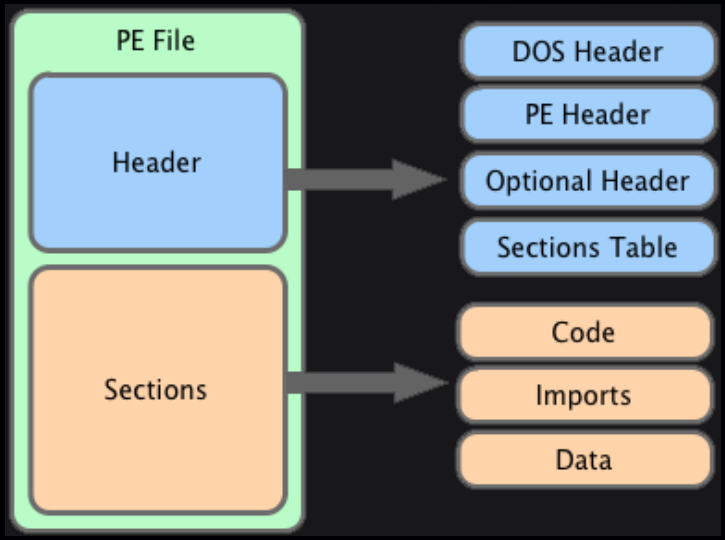
Static Malware Analysis:
Static Malware Analysis involves collecting information about the malicious binary and making decision based on these features.

Supervised Learning: Machine Learning Approach where we train our model, using labelled data. Machine knows the feature of data and the labels associated with the features. The machine learns to map the data to the label using the features.

Decision Trees: They are supervised learning algorithms that represents data as upside-down trees. They are used in classification and regression problems and are called CARTs. They were first introduced by Leo Breiman.

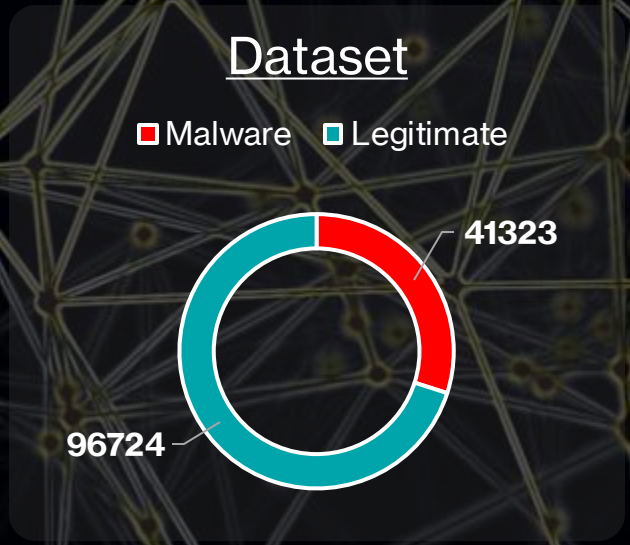
Malware Signature

Portable Executable(PE) Format files: These are files that contains information about all executable files, DLLs, object codes, etc. It can be read using software like PE Explorer, PEview, etc.



Structure of PE file

Malware Dataset



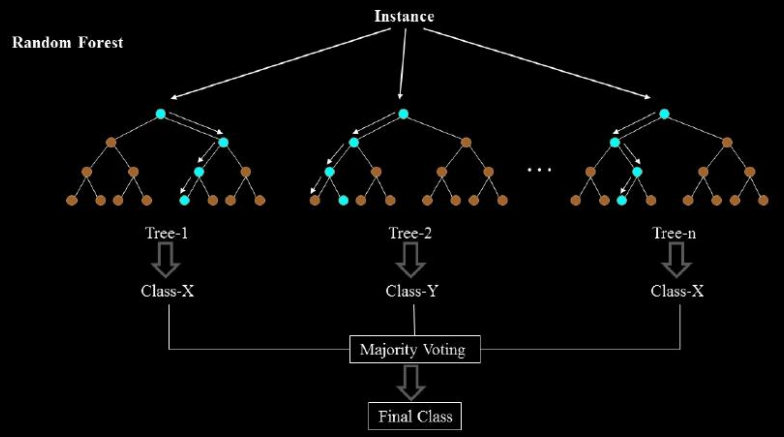
For our research project, we are using dataset provided by the security expert, Prateek Lalwani. This Dataset consists features of 41323 windows binaries files as legitimate files and 96724 malware files downloaded from VirusShare website.

Total Data: 138,048
Total Features: 56

ML Algorithms

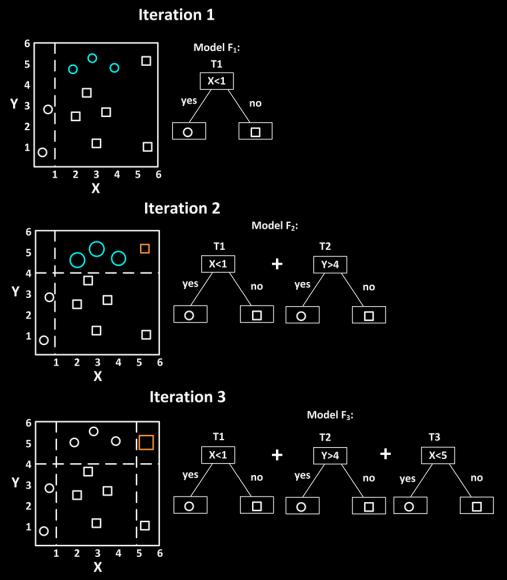
Random Forest: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

Source-Wikipedia



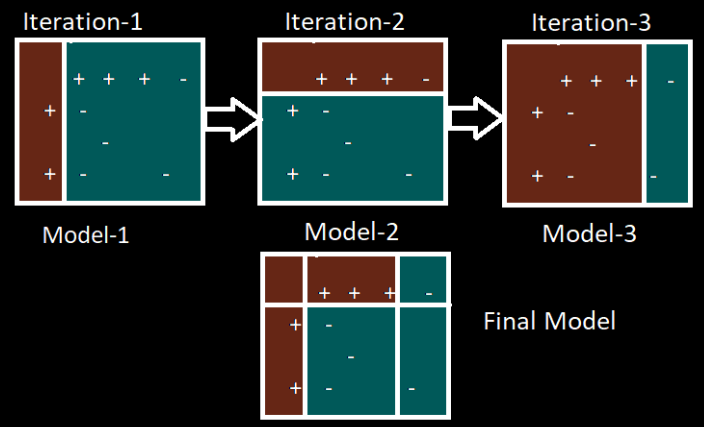
Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.[1][2] When a decision tree is the weak learner, the resulting algorithm is called gradient boosted trees, which usually outperforms random forest.

Source-Wikipedia



AdaBoost, short for Adaptive Boosting, is a machine learning meta-algorithm used in conjunction with many other types of learning algorithms to improve performance. The output of the other weak learning algorithms is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers.

Source-Wikipedia



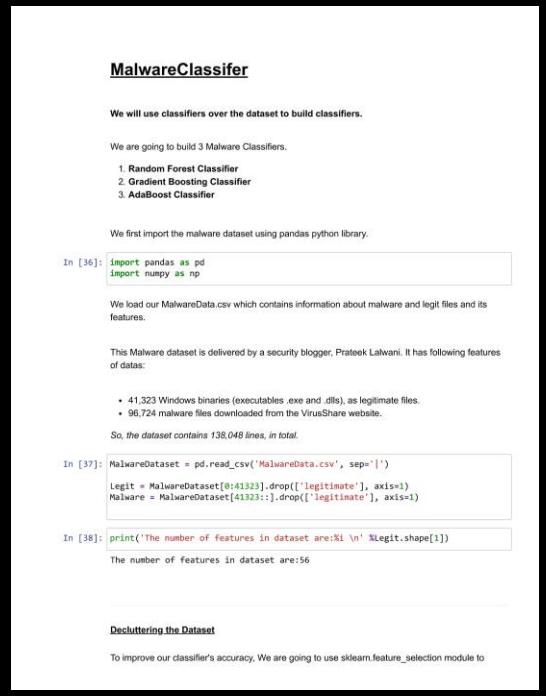
Procedure

We used the Linux environment (Ubuntu 20.04) and Python programming language as a choice. For carrying out coding, we choose Juypter Notebook.

First, we began with reducing the dimensionality of dataset. Since, our dataset had 56 features all of them are not important while training the classifiers. This benefits by reducing the overhead of the program and saves the resources consumed in computation.
After reduction, only 12 features were found to be important. They are:

1. DllCharacteristics
2. Characteristics
3. Machine
4. ResourcesMaxEntropy
5. SectionsMaxEntropy
6. Subsystem
7. VersionInformationSize
8. ImageBase
9. MajorSubsystemVersion
10. SizeOfOptionalHeader
11. ResourcesMinEntropy
12. MajorOperatingSystemVersion

Then, we fed dataset to the machine learning classifiers: Random Forests, Gradient-boosting classification and AdaBoost Classification



MalwareClassifer

We will use classifiers over the dataset to build classifiers.

We are going to build 3 Malware Classifiers.

1. **Random Forest Classifier**
2. **Gradient Boosting Classifier**
3. **AdaBoost Classifier**

We first import the malware dataset using pandas python library.

```
In [36]: import pandas as pd
         import numpy as np
```

We load our MalwareData.csv which contains information about malware and legit files and its features.

This Malware dataset is delivered by a security blogger, Prateek Lalwani. It has following features of datas:

- 41,323 Windows binaries (executables .exe and .dlls), as legitimate files.
- 96,724 malware files downloaded from the VirusShare website.

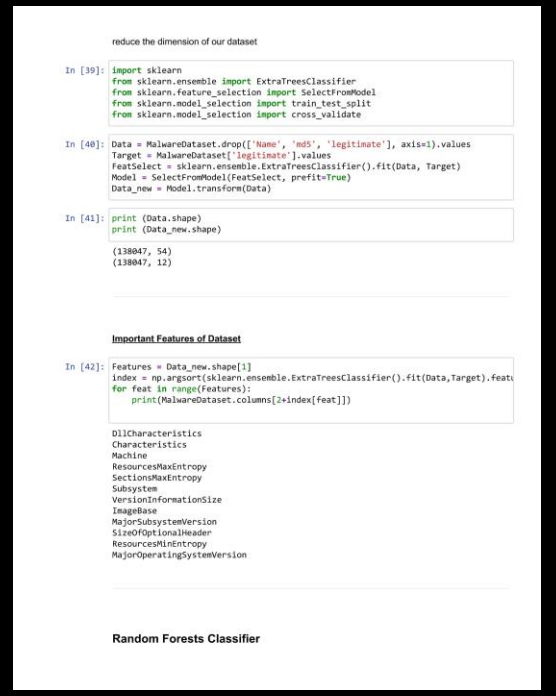
So, the dataset contains 138,048 lines, in total.

```
In [37]: MalwareDataset = pd.read_csv('MalwareData.csv', sep='|')

         Legit = MalwareDataset[0:41323].drop(['legitimate'], axis=1)
         Malware = MalwareDataset[41323::].drop(['legitimate'], axis=1)
```

```
In [38]: print('The number of features in dataset are:%i \n' %Legit.shape[1])

         The number of features in dataset are:56
```

Decluttering the Dataset

To improve our classifier's accuracy, We are going to use sklearn.feature_selection module to



reduce the dimension of our dataset

```
In [39]: import sklearn
         from sklearn.ensemble import ExtraTreesClassifier
         from sklearn.feature_selection import SelectFromModel
         from sklearn.model_selection import train_test_split
         from sklearn.model_selection import cross_validate
```

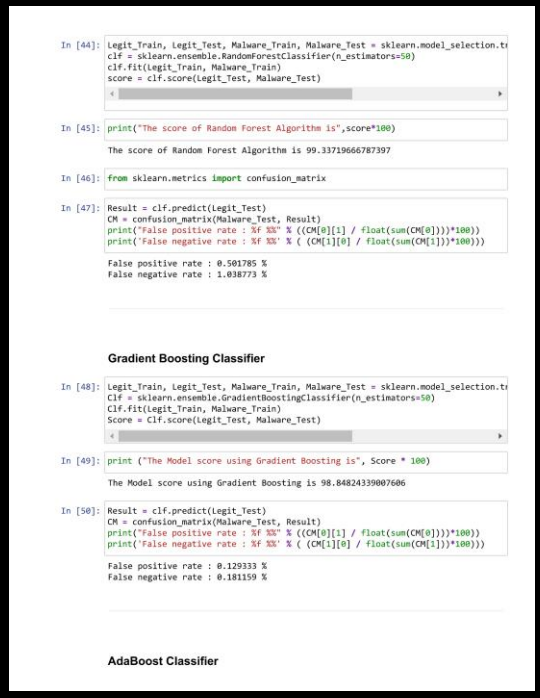
```
In [40]: Data = MalwareDataset.drop(['Name', 'md5', 'legitimate'], axis=1).values
         Target = MalwareDataset['legitimate'].values
         FeatSelect = sklearn.ensemble.ExtraTreesClassifier().fit(Data, Target)
         Model = SelectFromModel(FeatSelect, prefit=True)
         Data_new = Model.transform(Data)
```

```
In [41]: print (Data.shape)
         print (Data_new.shape)

         (138047, 54)
         (138047, 12)
```

Important Features of Dataset

```
In [42]: Features = Data_new.shape[1]
         index = np.argsort(sklearn.ensemble.ExtraTreesClassifier().fit(Data,Target).featu
         for feat in range(Features):
             print(MalwareDataset.columns[2+index[feat]])

         DllCharacteristics
         Characteristics
         Machine
         ResourcesMaxEntropy
         SectionsMaxEntropy
         Subsystem
         VersionInformationSize
         ImageBase
         MajorSubsystemVersion
         SizeOfOptionalHeader
         ResourcesMinEntropy
         MajorOperatingSystemVersion
```

Random Forests Classifier



```
In [44]: Legit_Train, Legit_Test, Malware_Train, Malware_Test = sklearn.model_selection.t
         clf = sklearn.ensemble.RandomForestClassifier(n_estimators=50)
         clf.fit(Legit_Train, Malware_Train)
         score = clf.score(Legit_Test, Malware_Test)
```

```
In [45]: print("The score of Random Forest Algorithm is",score*100)

         The score of Random Forest Algorithm is 99.33719666787397
```

```
In [46]: from sklearn.metrics import confusion_matrix
```

```
In [47]: Result = clf.predict(Legit_Test)
         CM = confusion_matrix(Malware_Test, Result)
         print("False positive rate : %f %%" % ((CM[0][1] / float(sum(CM[0])))*100))
         print('False negative rate : %f %%' % ( (CM[1][0] / float(sum(CM[1])))*100))

         False positive rate : 0.501785 %
         False negative rate : 1.038773 %
```

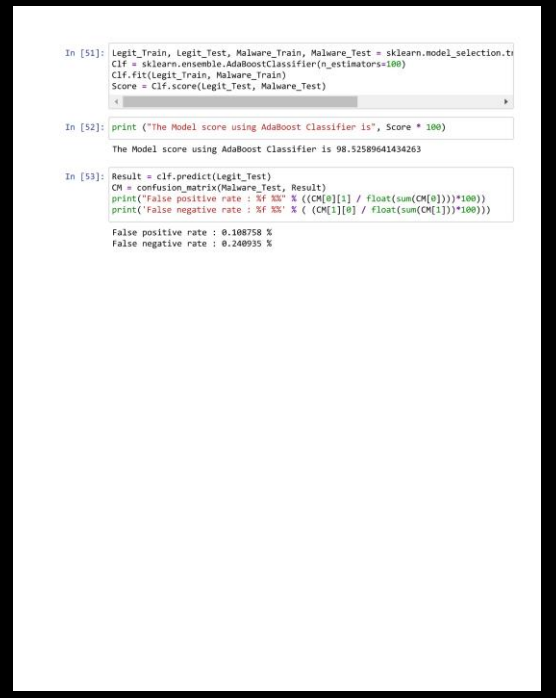
Gradient Boosting Classifier

```
In [48]: Legit_Train, Legit_Test, Malware_Train, Malware_Test = sklearn.model_selection.t
         Clf = sklearn.ensemble.GradientBoostingClassifier(n_estimators=50)
         Clf.fit(Legit_Train, Malware_Train)
         Score = Clf.score(Legit_Test, Malware_Test)
```

```
In [49]: print ("The Model score using Gradient Boosting is", Score * 100)

         The Model score using Gradient Boosting is 98.84824339007606
```

```
In [50]: Result = clf.predict(Legit_Test)
         CM = confusion_matrix(Malware_Test, Result)
         print("False positive rate : %f %%" % ((CM[0][1] / float(sum(CM[0])))*100))
         print('False negative rate : %f %%' % ( (CM[1][0] / float(sum(CM[1])))*100))

         False positive rate : 0.129333 %
         False negative rate : 0.181159 %
```

AdaBoost Classifier

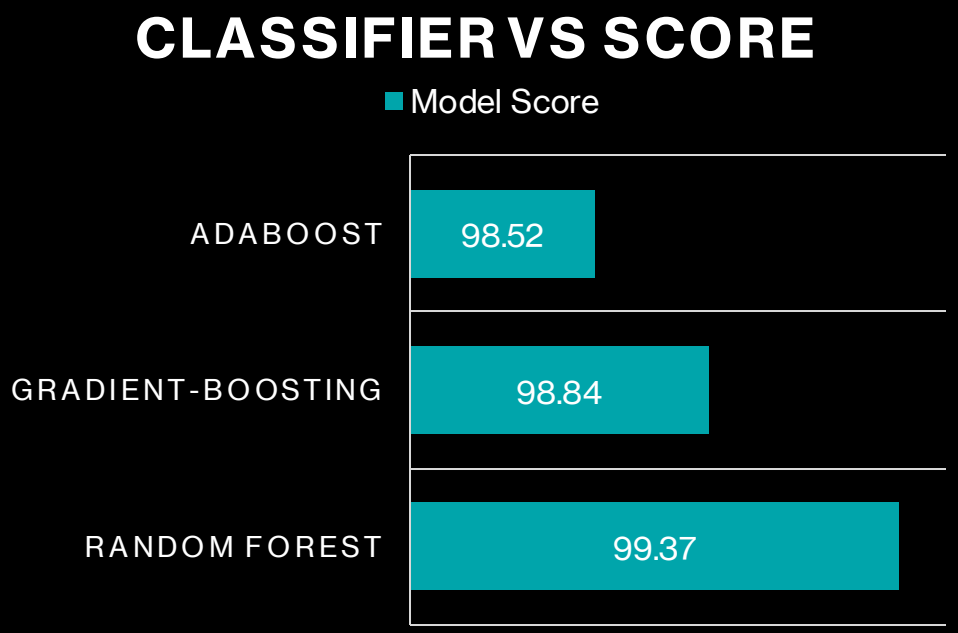


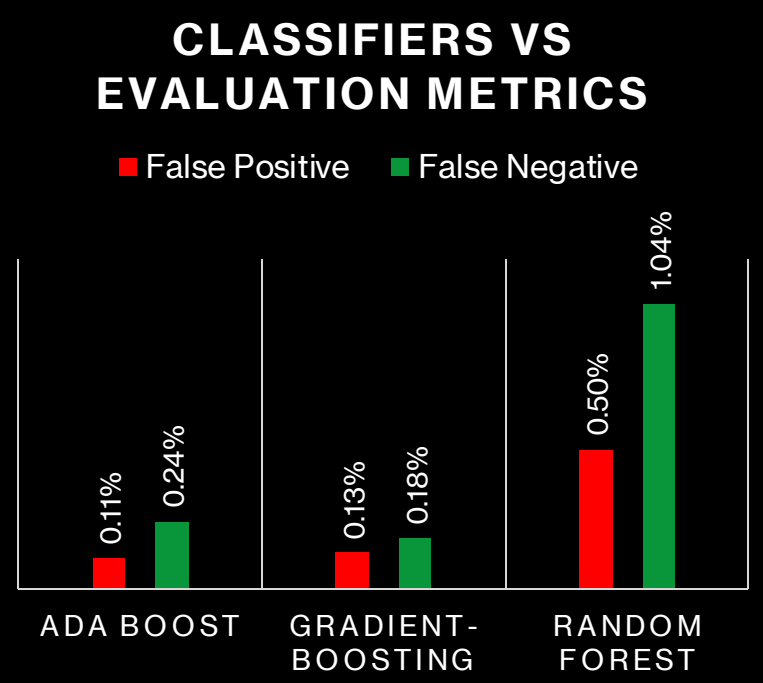
```
In [51]: Legit_Train, Legit_Test, Malware_Train, Malware_Test = sklearn.model_selection.t
         Clf = sklearn.ensemble.AdaBoostClassifier(n_estimators=100)
         Clf.fit(Legit_Train, Malware_Train)
         Score = Clf.score(Legit_Test, Malware_Test)
```

```
In [52]: print ("The Model score using AdaBoost Classifier is", Score * 100)

         The Model score using AdaBoost Classifier is 98.52589641434263
```

```
In [53]: Result = clf.predict(Legit_Test)
         CM = confusion_matrix(Malware_Test, Result)
         print("False positive rate : %f %%" % ((CM[0][1] / float(sum(CM[0])))*100))
         print('False negative rate : %f %%' % ( (CM[1][0] / float(sum(CM[1])))*100))

         False positive rate : 0.108758 %
         False negative rate : 0.240935 %
```

Findings



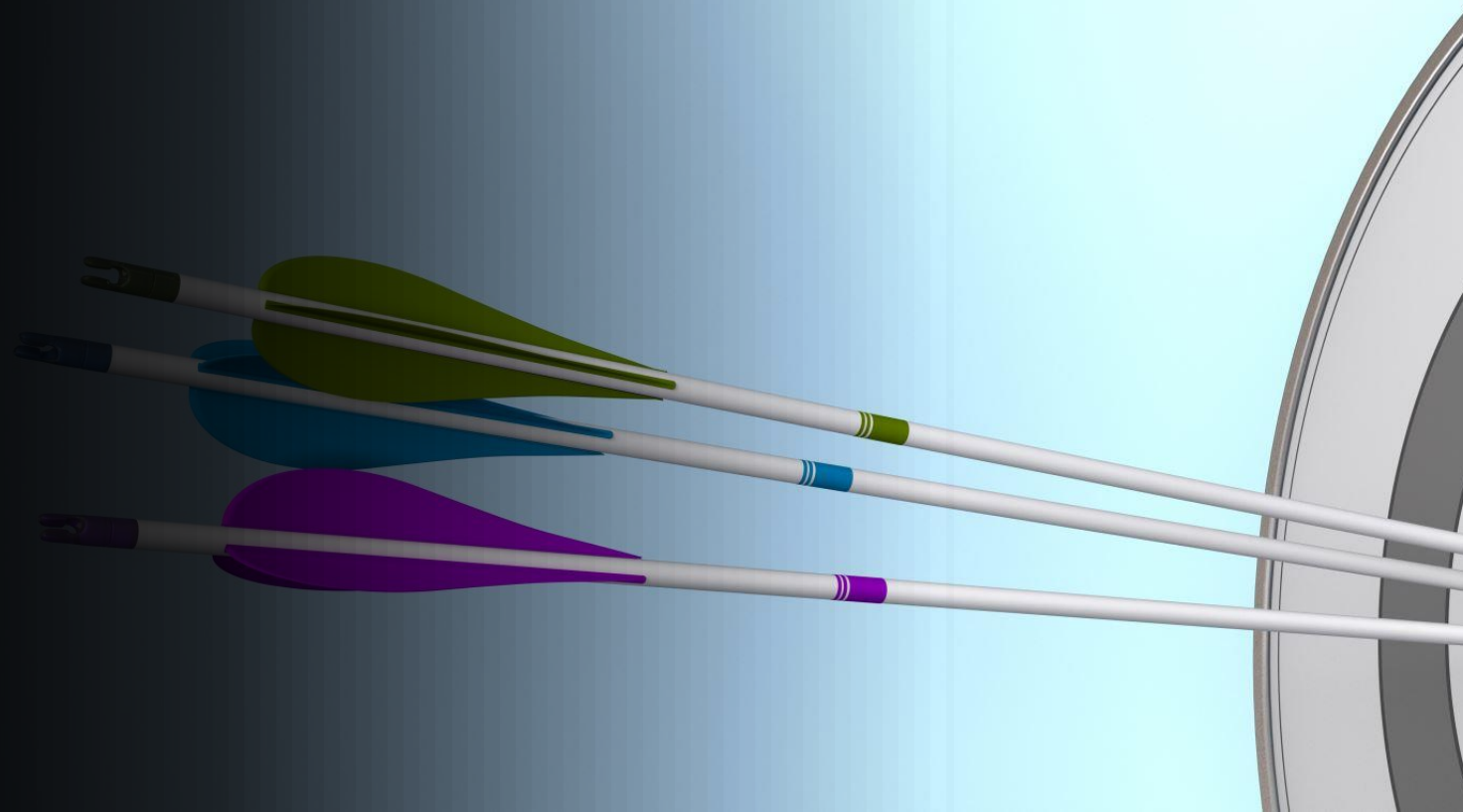


False Positive: Predicting negative class as positive
False Negative: Predicting positive class as negative

Results and Shortcomings

We used the three different Decisions trees to identify malwares in our dataset. Since the size of the dataset was relatively small Random Forest tend to perform best according to the score for detecting Malware. However, confusion matrix shows that there is marginal difference in AdaBoost and Gradient Boosting for higher accuracy of precision.

This kind of approach is called static malware analysis, however in recent era malware has evolved and they can mutate while passing in various nodes and bypass the security measures. Also, This models are needed to train on larger dataset and fewer dataset can lead to more inaccuracy.



Conclusion

Malware is always one of the top issue in IT fields. Every year, billions of computers are infected by malware which leads to loss in finance and work progress of individual and corporate. Machine Learning and Artificial Intelligence can be used to detect Malware. Various research are going to implement this approach in IDS by leading companies in security like Palo Alto Networks, Kaspersky, Norton Security, etc. As an IT enthusiasts, Malware analysis opens new fields and possibilities of study and integrating Machine Learning and Artificial Intelligence is crucial.

Codes for this project can be found at GitHub: https://github.com/season101/Student-Research-Symposium-2021-AI-and-Malware-Signature.git

References

- Wikipedia
- Chiheb Chebbi - Mastering Machine Learning for Penetration Testing_ Develop an extensive skill set to break self-learning systems using Python-Packt Publishing (2018)
- Python Docs
- Google
- And Several Others.

Thank You