ABSTRACT

The challenging and constantly changing environment of cyber security landscape need to be addressed with latest tools approaches and architectures to combat cyber attacks and minimize or even eliminate vulnerabilities within information technology systems.

The cyber security industry is more and more embedding data-driven algorithms in security software to detect cyber threats cyber criminals widely develop and use malware to attack it infrastructures and systems.

Machine learning for malware detection is still in its early stages and adoption of deep learning to detect. Malware is gaining more and more importance for example antivirus software development this project is considering deep neural networks for classification of malware a widely used algorithm is convolutional neural network which is very effective for image classification tasks but how can an algorithm build for image classification be used for malware detection well an approach is to convert a malware into an image you can find many converters and pieces of codes to convert binary values to images in the internet for our project. We find which is the most efficient tool for the malware classification and detection.

We are testing both the CNN and machine Learning algorithms to see which is efficient in today advanced generation.

As the time grows the year passes the malware that is piece of software or code getting complicated. So we try to figure out the best solution possible to eradicate the malware from out files or from out devices.

INDEX TERMS

Malware, malware classification, malware detection, malware variants, deep neural networks, image classification, CNN(convonutional neural network), Cyber security, malicious.

I. INTRODUCTION

machine learning for malware detection is still in its early stages and adoption of deep learning to detect

malware is gaining more and more importance for example antivirus software development this project is considering deep neural networks for classification of malware a widely used algorithm is convolutional neural network which is very effective for image classification tasks but how can an algorithm build for image classification be used for malware detection well an approach is to convert a malware into an image you can find many converters and pieces of codes to convert binary values to

- images in the internet for our project

CNN

It has Input layer, hidden layer, and output layer

Input layer form is determined by the trained data

The number of units in the input layer should be correspond to input data

For neural network atleast one hidden layer is utilised

The hidden layer is depdends on the complexity of data received

the number of hidden layer units should be set

a deep learning training process consists of building the architecture model accuracy evaluation with loss function and finally optimizing weights of the neural network.

Linear and Non-Linear Layer in CNN

As neural network has multiple layer

So if we have the multiple linear layer doesn't have an learning impact on neural networks

So non-linear layer such as sigmoid , relu and tan h or used the torch.nn is a pytorch submodule for artificial neural networks that contains all modules for building various neural network architectures as characteristics a module can have parameter instances which are tensors whose values are optimized throughout the training process we have several subclasses such as sequential which is later used to design basic sequential layers of a network another subclass is linear which will be used later as well three arguments are allowed to be used with linear it is number of input and output features.

HOW CONVOLUTIONAL NUERAL NETWORKS WORK?

Generally we can understand that neural networks with more than one convolution operation is called convolution neural network

Generally we can understand as inputs for convolutional neural networks natural signals coming as multi-dimensional arrays the usage of convolutional neural networks is not limited to computer vision tasks for example one-dimensional convolutional neural networks can be applied to time series analysis test and sound processing and so on

For computer vision tasks convolutional neural network or
CNN are applicable due to the following reason

1. cnns consider correlation of nearby values in arrays thus probability that nearby color pixel values are same is given in images and videos
2. cnns are robust to variations of feature locations due to pooling layers which will be described later thus location for object detection or classification is irrelevant

for simplicity let us assume that we want to build a neural network model that can classify wolves in images
a wolf has certain features that provides the unique signals to the neural network like legs nose eyes ears and so on inputs for convolution neural networks for image processing are numerical values which are arranged as height width and channels or features each image provides a probability score belonging to a defined class in case a feed forward neural network is trained for image classification there exist some essential roadblocks to consider first flattening an image to an one-dimensional vector results in the loss of spatial information closely spaced nodes are critical because they contribute to the definition of an image's characteristics so we need a way to use spatial correlation of image features to see the wolf in our picture no matter where it is another disadvantage is when working with large images the number of weights quickly becomes unmanageable for example there are around 150 000 weights to be trained for a 224 by 224 pixel image with three color channels this creates problems while training and overfitting can take place.

THE FEED FORWARD NUERAL NETWORK

the feed forward neural network uses simple matrix multiplication as its core operation in its linear layers to describe interaction between input and output units thus every input unit interacts directly with the output unit
each element of the weight matrix is used only once

a convolutional neural network is using same weights in multiple convolution layers which can be understood as parameter sharing from input units to output units a filter is transforming meaningful features fewer parameters and memory is therefore required compared to neural networks.

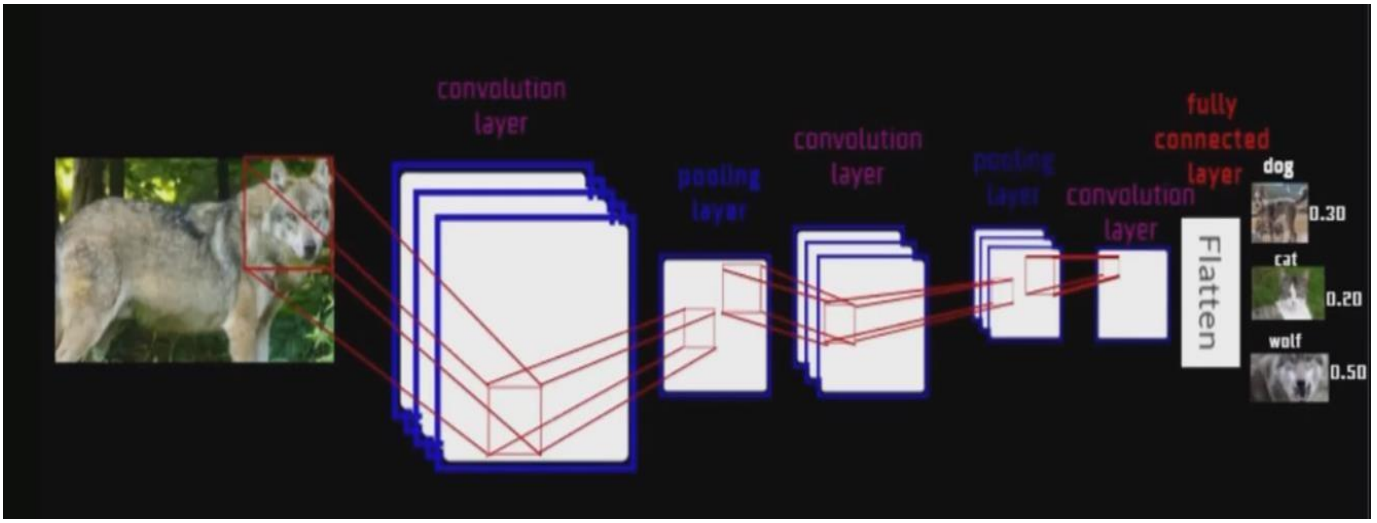
THREE MAIN COMPONENTS FOR A TYPICAL NUERAL NETWORK.

1. convolution layers
2. pooling layers and 3. fully connected layers within convolution layers features are learned by deep neural networks.

Then convolution neural network will look what features are present in an input image and finally

produce specific activation vectors these patterns are initiated by fully connected layers within the convolution neural network.

for instance a 3 by 3 filter would represent a 3 pixel by 3 3 selection of the image by using filters number of weights which are calibrated during training process of a neural network convolution filter values are continuously updated as the network is trained edge



HOW CNN LAYERS FOR IMAGE PROCESSING WORK
PROCESSING OF IMAGES

for three-dimensional convolutions input data has a shape of a tensor in form of channels depth batch size height and width

for example number for channels for a colored image is sum of red green and blue it is three the output shape of a convolution neural network consists of batch size and the other data will change by using filter
kernel size and padding convolution operation is done for all different
spatial localities in the input volume the convolutional layer extracts
different features from each part of an input image this layer performs a dot product between two matrices where one matrix is the set of learnable parameters also known as a filter and the other matrix
is the restricted portion of the receptive field the filter
assigns weights and biases to
any area of the input image

detection and sharpening are basic feature extraction operations by a filter the kernel or filter which is three by
three or five by five in size slides across the height and width of the image producing the image representation of
that receptive region this produces a two-dimensional representation of the image known as an activation or feature map that gives the response of the filter at each spatial position of the image the sliding size of the kernel is called a stride

finally the activation or feature map is taken through an activation function
the activation functions can be a relu sigmoid soft max and many others which
are used to decide whether a certain feature is present in a given location in an image more filtering layers can be added for deeper cnn.

POLLING PROCESS

now the pooling process comes into consideration spatial dimensions of each activation map is reduced by pooling while aggregating localized spatial information the pooling process squeezes the output from hidden layers along height and width.

the most popular pooling process is max pooling which reports the maximum output from the neighborhood additionally pooling supports averaging minimization and conversion to another function in this case the pooling layer collects only data that has been aggregated using a specific aggregation function its task is to reduce the height and width of the matrix by half while leaving the depth of the matrix intact because of this procedure we are able to reduce the risk of overfitting by identifying the most influential characteristics of our pooling layer and train the model faster.

PADDING and CONNECTING

the data from the last connecting layer is the input of a fully connected layer
the fc layer is responsible for flattening the vector value a fully connected layer takes the output of the convolutional layers and combine it through learning of nonlinear combinations of high level features it helps to map the representation between input and output the fully connected layer is learning a
possibly non-linear function in that space the final layer is in charge of
calculating the probability of belonging to a given class the softmax activation function is used for multi classification the sigmoid function is used for binary
classification

padding what happens at the edges of the image is a good question to ask yourself right now when we apply convolutions to a normal image the resulting image is down sampled by a factor proportional to the filter size
what are options if we do not want this to occur we can use padding padding equalizes the size of the feature maps produced by the filter kernels with the original image padding is very useful if the edges of a picture have significant functions that must be taken into consideration during the model training.

BACKPROPOGATION

back propagation after we defined a neural network architecture we need to understand the performance of the network and finally optimizing the network a deep learning model is trained by optimizing its parameters much like any other parameter-based machine learning model back propagation is used to tune the parameters and the final or output layer of the neural network
returns a loss

back propagation is the neural net training processes goal is to identify the weights w and biases b that minimize error the gradient descent algorithm does it to begin the weights are randomly initialized and then a process of iteratively changing the weights is undertaken until convergence each iteration begins with a forward pass in which the current prediction is produced and the model's error is evaluated using a loss function using gradient descent and the chain rule of differentiation this loss is then back propagated to the preceding layers
thus a backward pass is used to calculate the weights gradient to reduce the loss the parameters or weights at each layer are adjusted appropriately in order to find the minimum point of the loss function where the gradient is zero a coefficient ranging from zero to one often known as the learning rate determines the level of alteration and the algorithm iteratively moves in the direction of the steepest descent toward the minimum point the scalar learning rate is a hyper parameter which needs to be tuned the optimization schedule which is the process of changing the weights of a neural network has a substantial influence on how successfully a model is trained there are many variations of loss functions depending on the particular task of classification or regression and pytorch addresses all of them as we are creating models for classification tasks.

STOCHASTIC GRADIENT DESCENT - LOSS FUNCTIONS FOR CLASSIFICATION

stochastic gradient descent was developed in response to the constraints of gradient descent in order to address performance difficulties and accelerate convergence in the huge data sets probabilistic approximation of gradient descent is

stochastic gradient descent because at each step the algorithm calculates the gradient for one observation taken at random not the full data set stochastic gradient descent requires just that the predicted value of the random observation be a sub gradient of the function at point win large scale data sets stochastic gradient descent is more quicker and better suited however since the gradient is not computed for the full data set and instead is only applied to one random point on each iteration the updates have a greater variance when compared to gradient descent this causes the loss function to change more throughout each iteration making it more difficult for the algorithm to converge.

ENTROPY

Entropy is defined as the randomness or measuring the disorder of the information being processed in machine learning. Further, in other words we can say that entropy is the machine learning metric that measures the unpredictability or impurity in the system.

Multplication of different probabilities decreases the end result, so Entropy is much more suitable. The more question is required to find the correct options the more uncertainly exists, thus entropy is higher.

CROSS ENTROPY

The average number of bits required to send a message from distribution A to distribution B is referred as cross-entropy.Cross entropy is concept used in machine learning when algorithms are created to predict from the model. The construction of the model is based on a comparision of actual and exected results. a machine learning model predicts the likelihood of each data belonging to each class label which is between zero and one cross entropy measures the difference in entropy between two probability distributions over a random collection of events when projected probability differs more from anticipated probability cross entropy loss increases when projected probability differs less from anticipated probability cross entropy loss decreases so generally we can differentiate
between expected probability p and predicted probability q the cross entropy formula considers two distributions p(x) target probability and q(x) actual probability into consideration defined over the variable

cross entropy can help us here in which direction the parameter of a model needed to be tweaked to get finally a good model.

2. ACTIVATION TECHNIQUE

PROBLEM IN ACTIVATION FUNCTION

the main problem is in case a very deep neural network need to be trained the gradients need to propagate backwards throughout the network starting from the end

the issue here is that a chain of activation functions like sigmoid of a sigmoid above a sigmoid of a sigmoid etc
is used and finally a multiplication of derivatives of activation functions is done
when you multiply a chain of small numbers you will get even a smaller number

so the result is that weights closer to neurons are not trained at all within very deep neural networks.

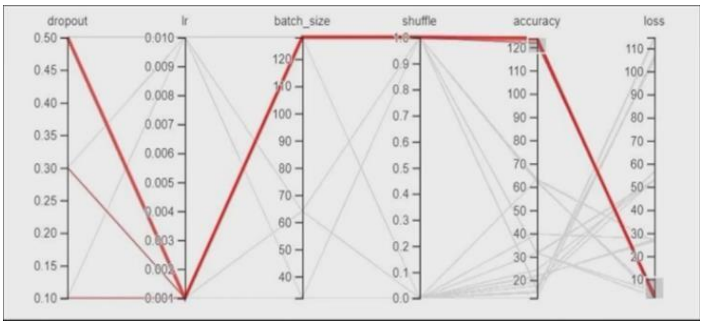
RELU

in recent years a relu or rectifier linear unit activation function has become more widely used as tanh and sigmoid are not very suitable to be used as you can see values and thus gradient is greater than zero the optimizer will find the right set of weights center and thus the training process is faster by using the relu activation function a learning rate need to be carefully modified to avoid that the activation function becomes non-responsive by using a leaky relu derivatives will be always positive and it is still a non-linear function

SIGMOID

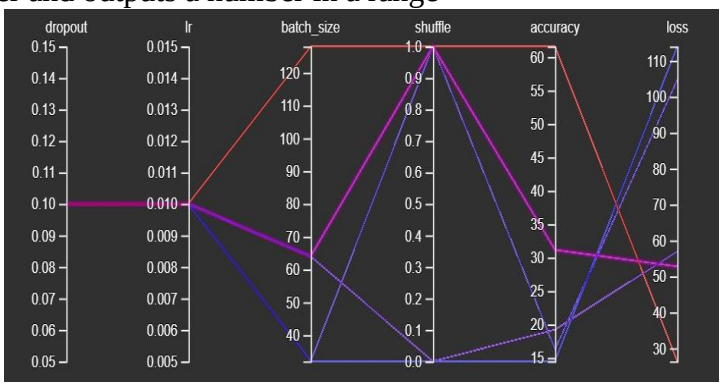
a sigmoid activation function maps inputs to outputs with a number in a range between zero and one this function is not widely used in neural network architectures this model is more suitable for binary classification problems.

the disadvantage is that for binary classification the output of sigmoid is close to zero or one the gradients for layers before sigmoid function are nearly zeros so the parameters of the previous layer get gradients close to zero and weights are not optimized.

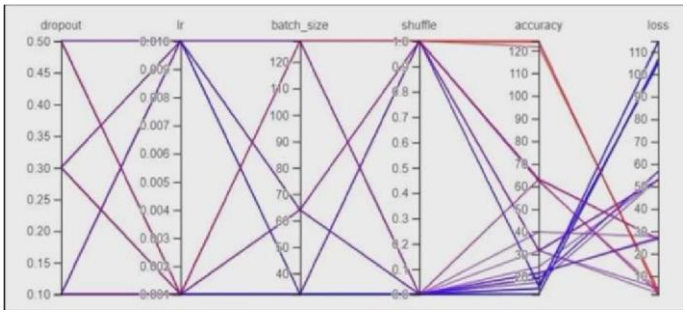


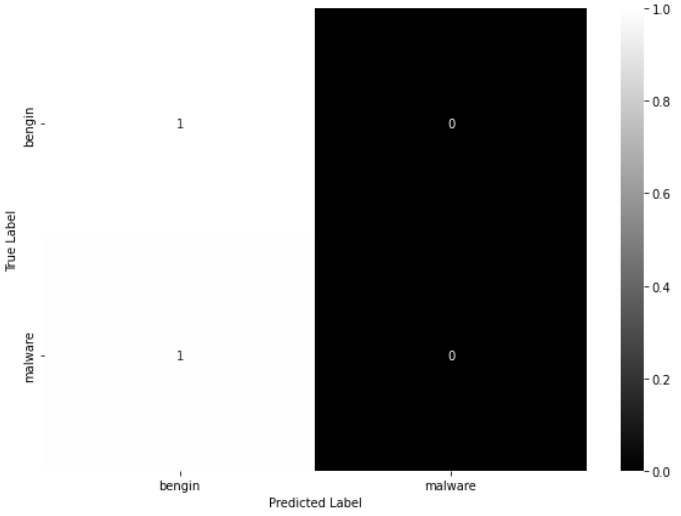
TAN H

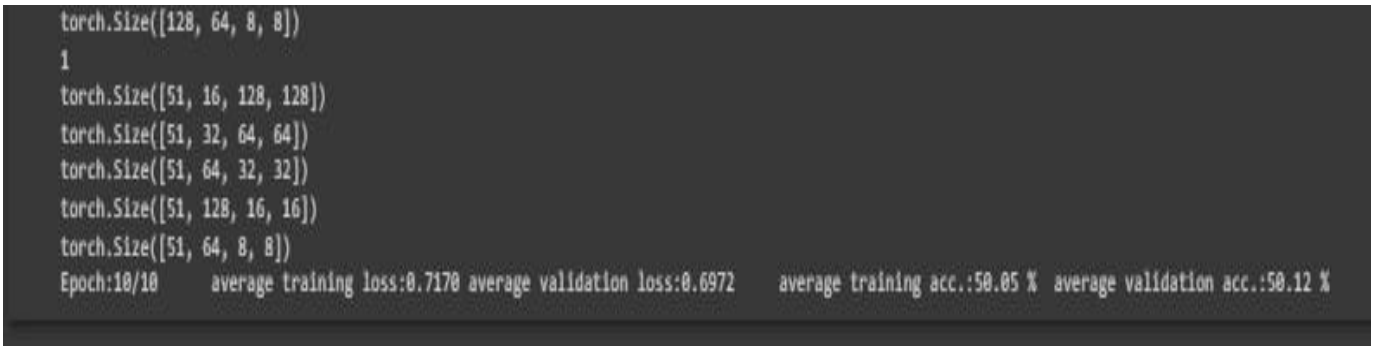
a tan h activation function takes a real value number and outputs a number in a range between -1 and 1.
Similar to a sigmoid function with the same shape and centered around zero
anyhow both s-shaped activation functions are not used widely today within neural networks.



CNN HYPERPARAMETER TENSORBOARD AND RESULTS







In CNN we get the accuracy as average training accuracy is 50.05% , average validation accuracy is 50.12% , average training loss is 0.7170, average validation loss is 0.6972 .

References for my Github repository of this above program is here

NOW USING MACHINE LEARNING ALGORITHMS

Now lets use various machine learning algorithms to get more accuracy than CNN. In this machine learning algorithm we used many metrics where this is discussed below.

USING MACHINE LEARNING ALGORITHMS

First we describe what is the different types of malware are present

There are many varieties of malware and every day there are new forms of malware that recreate previous methods of attack dynamically or implement radically new vulnerability tactics that leverage the specific characteristics of the target organization.

Generally it is possible to compile a classification of the most common types of malware in order to understand which are the most effective measures of prevention so here you can see an overview of most common malware.

1.TROJANS

first let's understand what a trojan is trojan is a special form of malware usually disguised as software that is legitimate i some form of social engineering victims usually get tricked into loading and executing trojans on their systems trojans will gain back door access to your device by cyber criminals and steal your sensitive data trojans can be for example key loggers to steal user names and password.

2. BOTNET

Botnets are now our infected computer networks and are typically remotely controlled by cyber criminals with the aim for financial gain or to initiate websites or network assaults a computer infected with this type of malware and plot of a botnet will communicate and receive instruction from command and control computers anywhere around the globe about what it should do what your computer is doing depends on what the cyber criminals are trying to do most botnets are intended to collect data like credit card numbers social security numbers telephone numbers addresses and other personal data.
The word botnet is a hybrid combination of the words robot and network.

3. RANSOMWARE

Ransomware is a form of malware that accesses blocks and encrypts files from a victim and then demands that the victim to pay around them to get back the hijacked files cyber criminals use these attacks to try to get victims to click on links or attachments that appear legitimate but actually contain malicious code payment is eventually you

away for target and the victim is forced to pga for the removal of the ransomware either by supplying a program that can decrypt the files.

4. ROOTKIT

In order to enable access to a computer or software region a rootkit which is a set of traditionally malicious software is used they ain't over root caters to hide presence or existence of other software detection of rootkit is difficult because it can manipulate the software which is installed to detect it as a detection approach an alternative operating system difference or signature scanning and behavioral based methods can be used if the root kit is embedded in the kernel a deletion is very difficult and nearly impossible it should be considered that the whole operation system needs to be reinstalled firmware root kits require in most cases in hardware replacement the word rootkit comes from root and then in the linux world.

5. DOWNLOADERS

first step for a cyber attack by cyber criminals to install malware on the target device is through a downloader after the downloader is installed on the target device it will try to connect to a remote instance like several website where additional files for download can be found or further instructions where to find additional download files a downloader is a piece of code in order to download a malware on your computer therefore the detection of downloaders is more difficult than bigger size of malware and cyber criminals have an easy step to download different types of malware on the target device.

6. PHISHING

Phishing is a cyber-attack method where cyber criminals send random emails to a wide audience in order to manipulate users to send sensitive information like bank account numbers account passwords or login passwords in most phishing cases and user is fooled into following a malicious link which may result in a malware download or disclosing sensitive information as start of a ransomware attack as countermeasures to defend a device from malware attacks carried out by cyber criminals malware detection strategies can be used like network management strategies constant system monitoring or hashing.

MALWARE ANALYSIS TOOLS

A network management system continuously tracks a computer network for elements which are delayed or missing and in case of anomalies or other issues the network administrator is notified protocols are used to monitor network infrastructure through system monitoring computer operations like ram or cpu usage is measured and in case of performance anomalies regarding software applications or hardware the cause of problems could be identified locally or remotely simple code analysis tools can also be used for analyzing if code of a software application contains malware purposes finally another common technique to identify malware is hashing in case a unique hash of software is determined it can be used for different use cases sharing labeled suspicious bit of code and identifying if a malware has been detected within the malware analysis community is a huge benefit through hashing perhaps a suspicious bit of code has been already identified as malware among the malware community hash said there's a fingerprint of the malware to perform further searches in order to find out what information is available.

MALWARE ANALYSIS TYPES

1. Static Malware Analysis
2. Dynamic Malware Analysis

Most common approaches for malware analysis is known as static and dynamic analysis of malware, static analysis detects malware without running it no deep diver because instructions is scope of a typical static analysis there are different tools and

methods used to analyze if a file is malicious or not by disclosing information about technical indicators to define signatures typical indicators are checksum file size and type or hashes before executing assembly instructions in the disassembled binary image of the malware external characteristics of the binary code reliability must be identified because of the immense presence of antianalysis methods this is not possible to be assumed a preliminary analysis to detect paths which encrypt portions of executable code need to be done up front cyber criminals developing countermeasures to prevent malware analysis rely on using patches and payloads encryption dynamic software analysis approach is able to detect these obstacles when using virtual machines the presence of the virtual environment can be detected by advanced malware.
to build a robust model we can think about binary classification problems if we want to apply machine learning algorithms for malware analysis so what we need to understand is the following question what does an imbalanced data set mean a highly unbalanced data set means that an initial distribution of classes is reflected in the data set.

ANAMOLIES IN HIGHLY IMBALANCED DATA

Classification if malware is available or not in highly imbalanced labelled dataset is main goal of applying machine learning within cyber security.

Identifying anomalies in data sets and other accurate classification is a difficult task the most critical consideration is that the data set is highly unbalanced for a machine learning use case within the cyber security area classification if malware is detected or not is critical

METRICS

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{(\text{True Positive} + \text{False Positive} + \text{True Negative} + \text{False Negative})}$$

$$\text{Balanced Accuracy Score} = \frac{\text{TPR} + \text{TNR}}{2}$$

$$\text{F1-Score} = 2 \times \frac{(\text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})}$$

$$\frac{\text{True Positive}}{\text{Actual Results}} \quad \text{Precision} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Positive})}$$

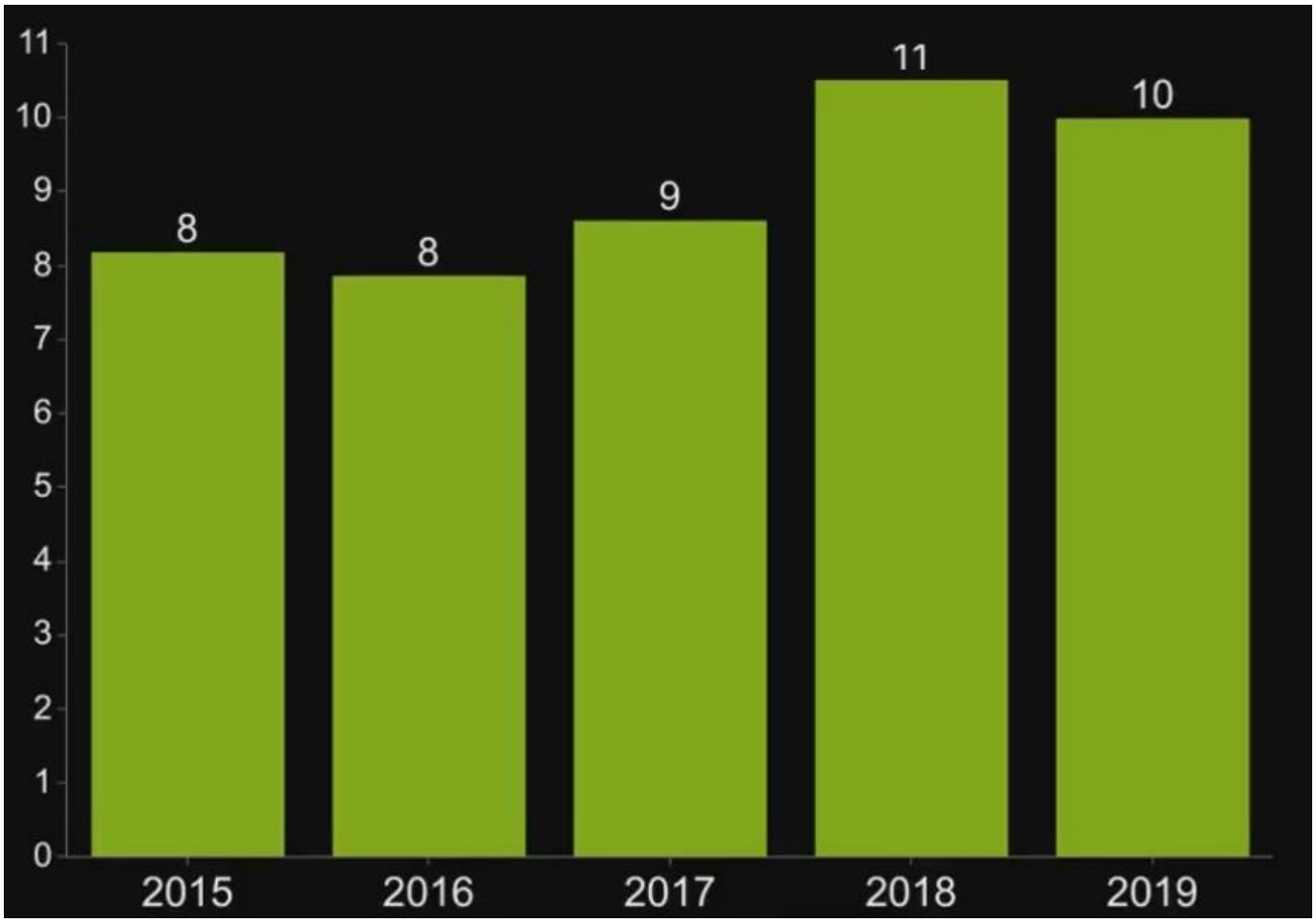
$$\frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{Recall} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})}$$

$$\text{True Positive Rate(TPR)} = \frac{\text{True Positive}}{(\text{True Positive} + \text{False Negative})}$$

$$\text{False Negative Rate(FNR)} = \frac{\text{False Negative}}{(\text{True Positive} + \text{False Negative})}$$

$$\text{True Negative Rate(TNR)} = \frac{\text{True Negative}}{(\text{False Positive} + \text{True Negative})}$$

$$\text{False Positive Rate(FPR)} = \frac{\text{False Positive}}{(\text{False Positive} + \text{True Negative})}$$



Above images shows the Annual Number of malware attacks worldwide from 2015 to 2019 [in billions]

EXPLANATION OF METRICS

Generally, precision is a suitable indicator in case the importance of false positives is high and important to pay attention when evaluating our machine learning model.

Recall is the important metric when we used to evaluate our machine learning model in case a high cost is associated with false negatives

Accuracy is in most machine learning use cases the key metric for evaluating the robustness of the machine learning model but accuracy does not consider class imbalance within a data set the percentage of correct predictive results by the classifier out of all available test data is the metric that can be understood as accuracy incorrect predicted results from a minority class within an imbalanced test data set are out of scope of the metric accuracy.

another important key indicator evaluating machine learning models for highly unbalanced data is therefore in f1score this metric takes both precision and recall into account so what is the difference between f1-score and accuracy. F1 score is balancing precision and recall on the positive class while accuracy looks at correctly classified observations both positive and negative.

one of the most important score for evaluating machine learning models for binary classification with highly unbalanced datasets is a balanced accuracy score this score considers two metrics first it is sensitivity and second is specificity specificity so what does these two metrics indicate

sensitivity score means sensitivity indicate how many of positive cases were correctly detected as positive within the total number of positive and negative cases

specificity indicates how many negative cases correctly classified as negative within the total number of negative and positive cases specificity indicates

Finally, the balanced accuracy score is sum of sensitivity and specificity divided by two

perhaps the most important metric for evaluating classification performance of machine learning models is with receiver operating characteristic curves what does a plotted receiver operating characteristic curve indicate it shows the trade-off between true positive rate and false positive rate across different decision thresholds these thresholds are implicit and not shown as an axis it can be assumed that the ideal decision threshold is set by 0.5 to separate true positive and true negative as well as false negative and false positive frequency in a classification model a random linear line from down left to upper right indicates the random classifier under are okay there's above the random classifier line the binary predicted probabilities has already mentioned a decision threshold by default 0.5 is taken to get a concrete class prediction for a given sample this means that in case a predicted probability is above 0.5 is considered to belong to class 1. The opposite belongs to a predicted probability which is below 0.5 and the sample belongs to class 0.

aroc start at false positive rate zero and true positive rate zero which is at the lower left hand corner where actually every example is classified as negative because all predicted probabilities are less than one the roc curve always ends at the upper hand corner with false positive rate 1 and true positive rate 1 and a decision threshold of 0 where every example is classified as positive because all predicted probabilities are greater than 0. the area beneath the receiver operating characteristic curve is called the "a u r o c" curve a and u stands for area under the x axis of "r o c" curve is the false positive rate and the y axis of "r o c" is the true positive root auro curve is plotted between the true positive rates on a y-axis and false positive rate on a x-axis at various threshold settings this graph is used how accurate the classification model is able to separate true positive and true negative so perhaps you can start to understand whether roc graphs however classifier performs as a so-called discriminator the area under a server operating characteristic curve indicates the performance of your classification model for instance a value of 0.5 is equivalent to a coin flipping scenario and generally this indicates not a good performance.

a value between 0.7 and 0.8 indicates a good performance a value greater than 0.8 is an excellent performance finally a value of 1.0 indicates a perfect classifier by calculating the true positive rate and false positive rate for different thresholds between one and zero the different points which create a roc curve can be determined in case more points are calculated the roc curve is a more smoother curve.

a precision and recall curve at this stage it needs to be underlined that precision and recall curve is not taking true negatives the majority labels class in an imbalanced data set into consideration precision and recall scores have been earlier in this course with necessary details introduced the precision recall curve is a plot of different probability thresholds and showing the relationship between recall or sensitivity on x-axis and the precision on y-axis

A high recall and low precision means, that your model is able to find all relevant instances in datasets, but can not precisely classify them.
A high precision and low recall means that your model is able precisely classify relevant datasets within all data, but not able to find all relevant instances in dataset.

In this case, your classifier, might be overfitted during the training with training data labelled as majority class.

PE HEADERS IN DATASETS (PORTABLE EXECUTABLES)

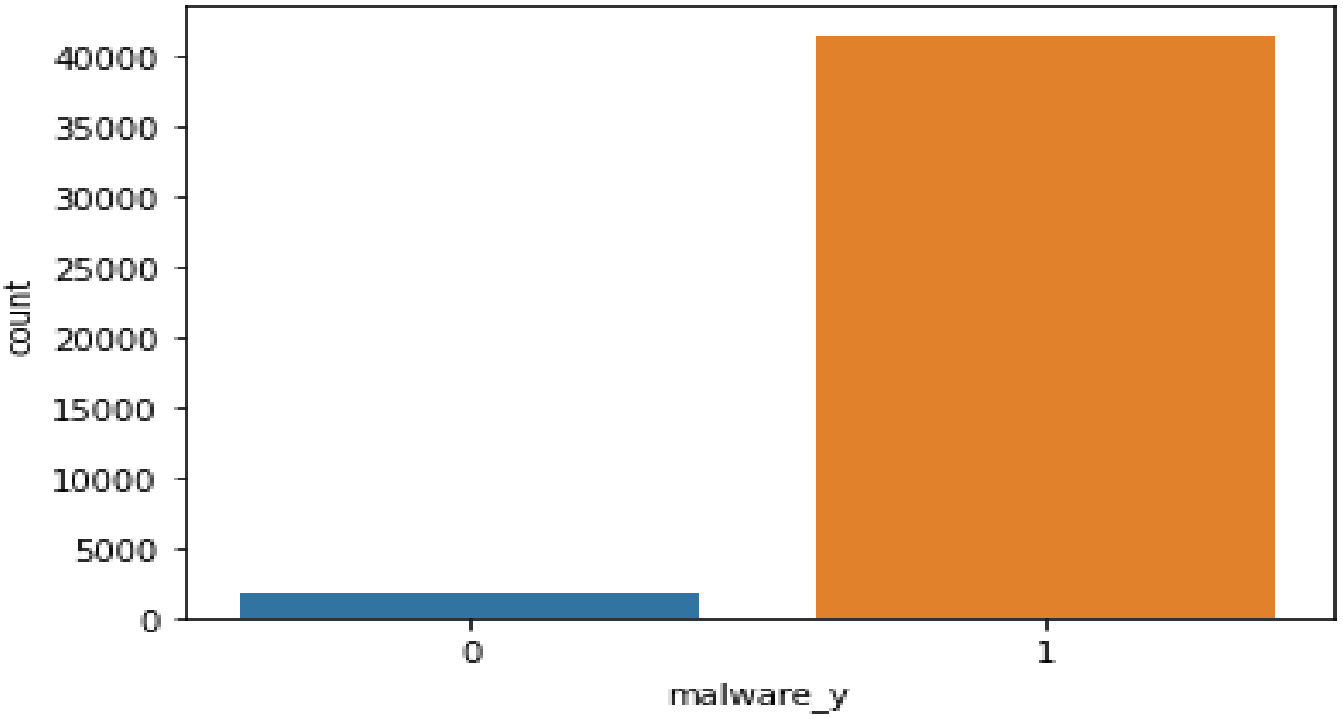
The dataset contains portable executable header information with many features and also labeled if malware available or not.

portable executable headers are portable executables are data structures which are required by the operating system to handle executable code.

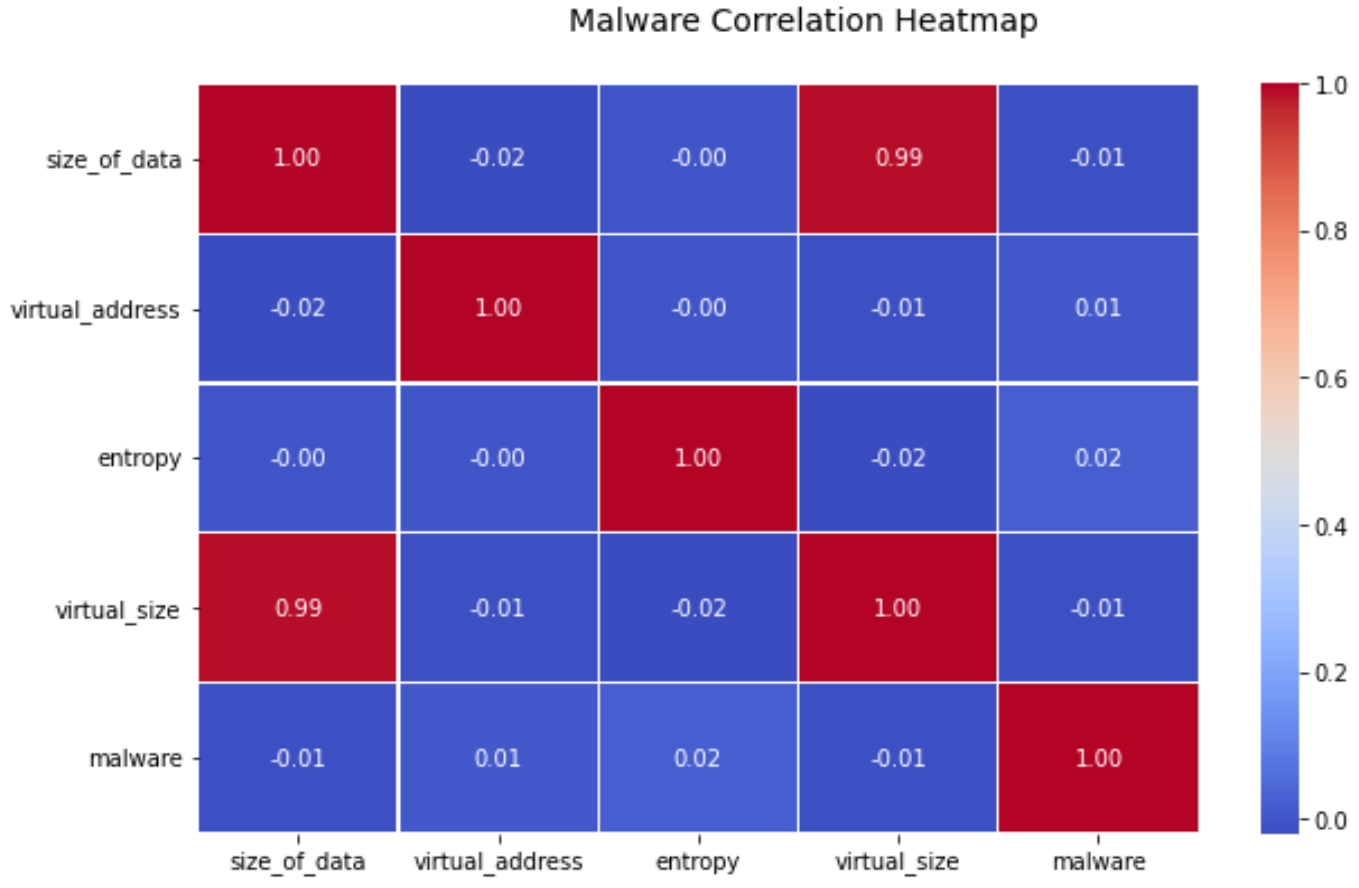
the pe format was designed to do the following steps first the pe format describes where and which parts of a file should be loaded into memory it describes where in the code the operating system should start executing the program then the pe format allocates resources of a running program which can be used for its execution these resources can be of any type like images videos or graphical user interfaces dialogs finally the pe format provides digital code signatures which the operating system uses to allocate a trusted source which the code comes from a pe format has a series of headers which communicates to the operating system how to load the program code into its memory one of these headers is the pe header which the operating system uses to load the program content from a file into the memory the pe header has also the address where the execution of the program starts and a list of functions on which the program code relies on the p header has the necessary information for helping the operating system to load and execute the program code properly the provided data sets have pe headers.

STARTING WITH CODING

In CSV files
we have normal file as 41449 and malware a
1724 See the graph ploted below,



MALWARE CORELLATION HEATMAP



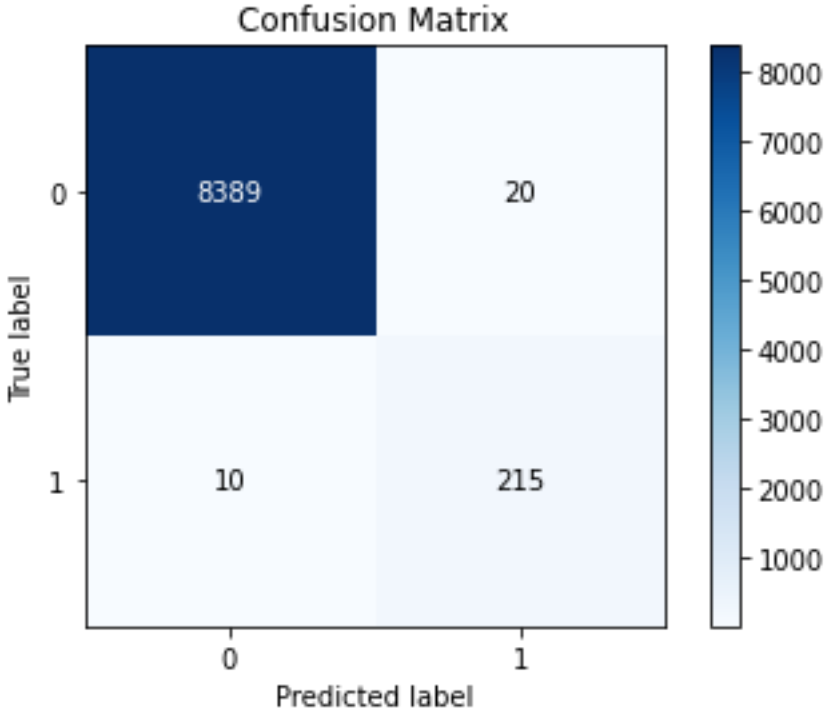
DIFFERENT MACHINE LEARNING ALGOITHMS

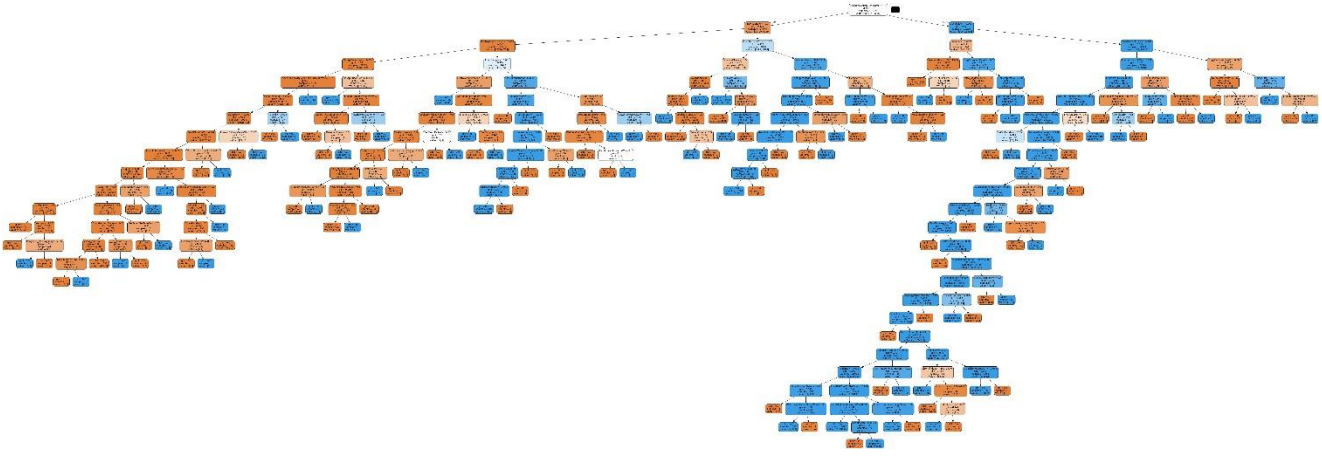
Metric, confusion matrix, ROC, precision-recall curve, and Learning Curve

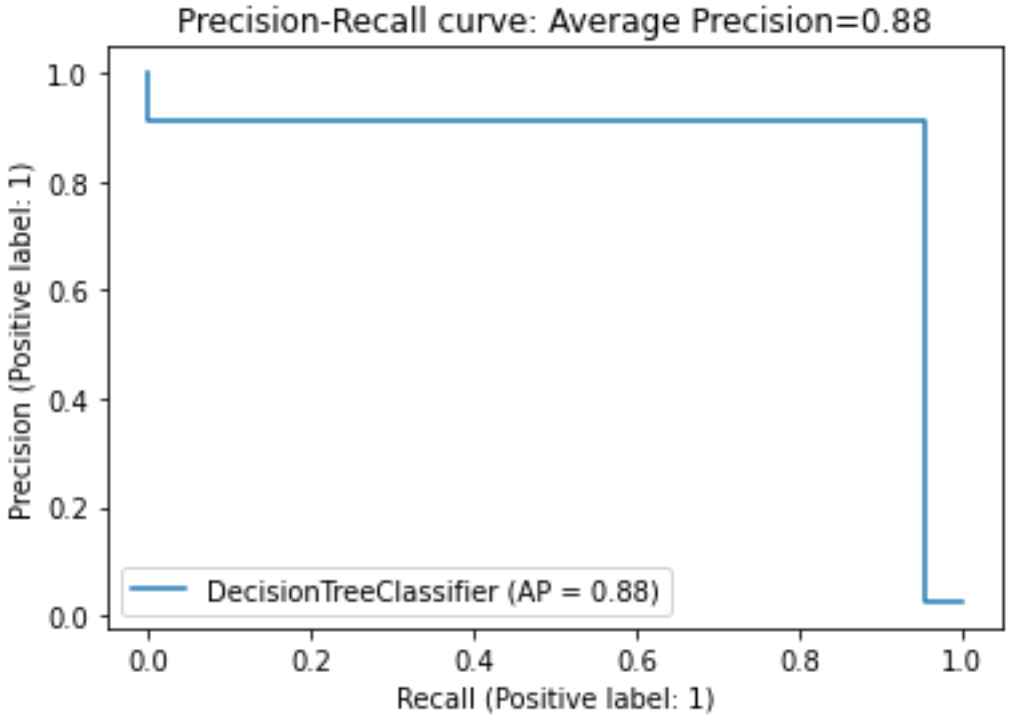
1. DECISION TREE CLASSIFIER

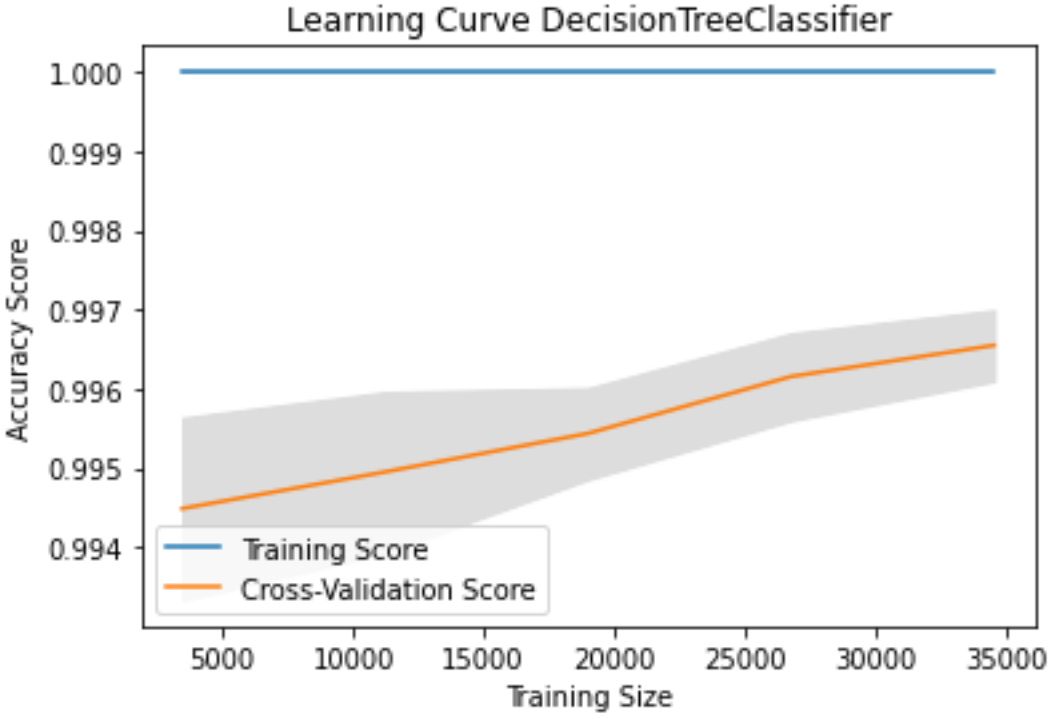
```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_dclf_f)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_dclf_f)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_dclf_f)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_dclf_f)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_dclf_f)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_dclf_f)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_dclf_f)))
```

```
F1-Score: 0.9348
Precision Score: 0.9149
Recall core: 0.9556
Accuracy Score: 0.9965
Balanced Accurarcy Score: 0.9766
Matthews Corrcoef: 0.9332
Average Precision Score: 0.8754
```



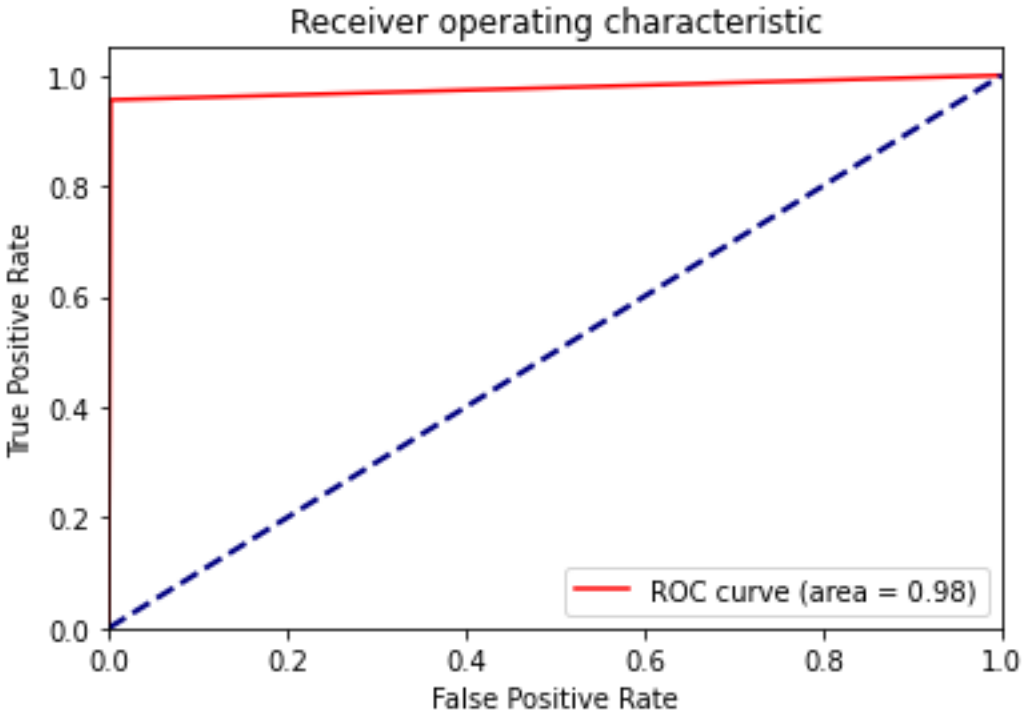







```
[38] #Reciever Operation curve
     from sklearn.metrics import roc_auc_score
     roc_auc_score(y1_test, y_pred_dclf_f)
```

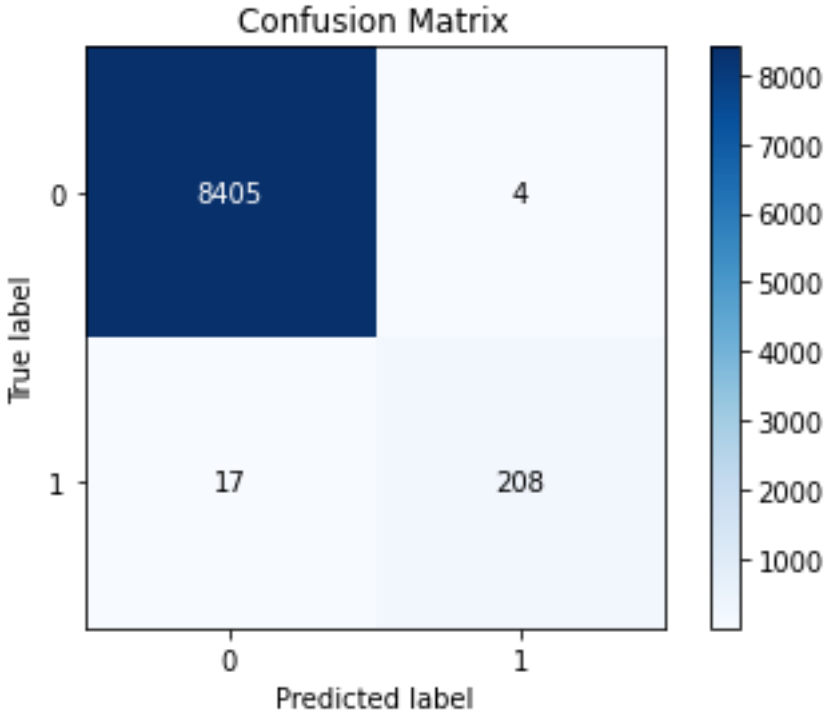
```
0.9765885757323504
```



---

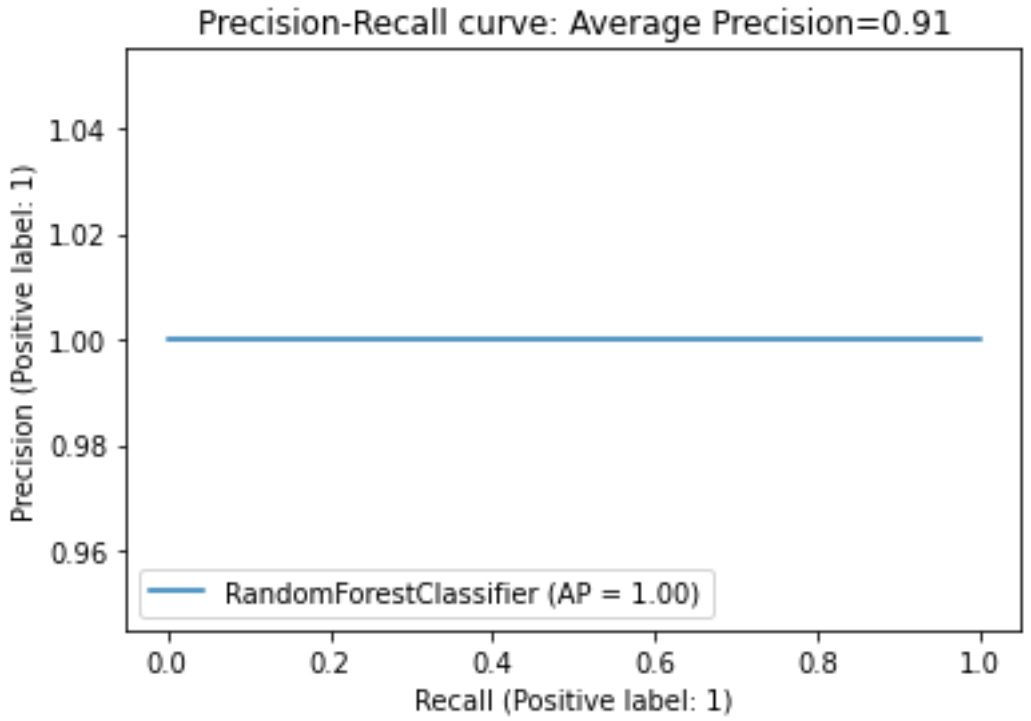
2. RANDOM FOREST CLASSIFIER

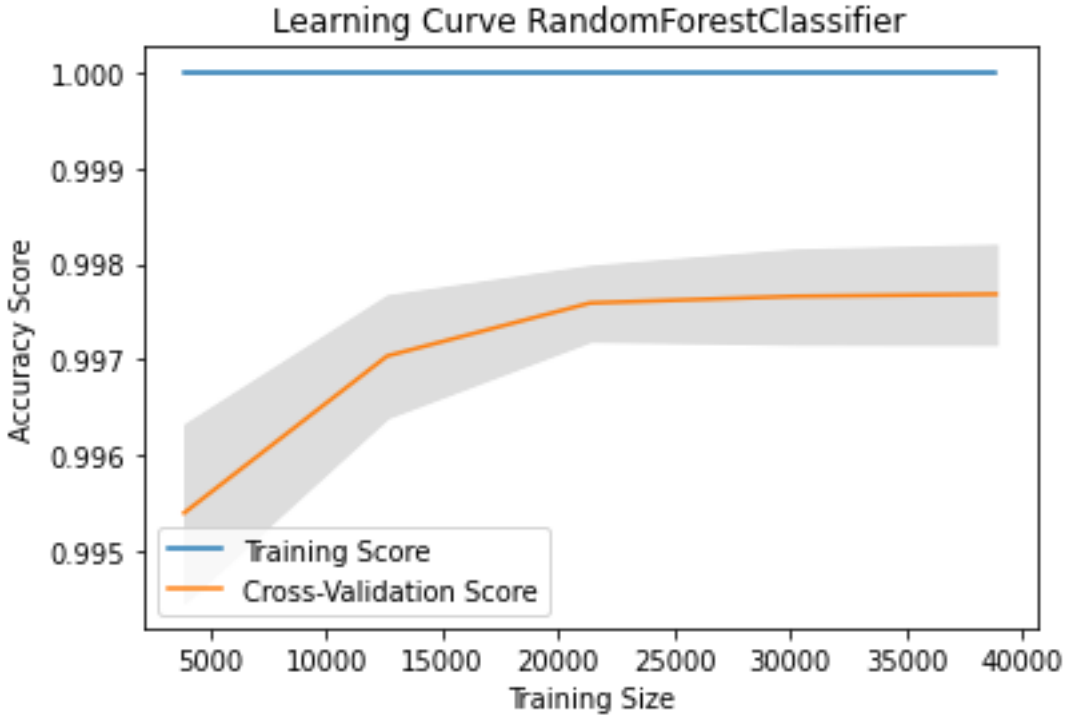
```
[45] f1_score(y1_test,y_pred_rfc)

     0.9519450800915331

[46] accuracy_score(y1_test, y_pred_rfc)

     0.9975677553856845

     balanced_accuracy_score(y1_test, y_pred_rfc)

     0.9619843818131367

[48] precision_score(y1_test, y_pred_rfc )

     0.9811320754716981

[49] recall_score(y1_test, y_pred_rfc)

     0.9244444444444444

[50] matthews_corrcoef(y1_test, y_pred_rfc)

     0.9511415975576181

[51] average_precision_score(y1_test, y_pred_rfc)

     0.9089710563619331
```

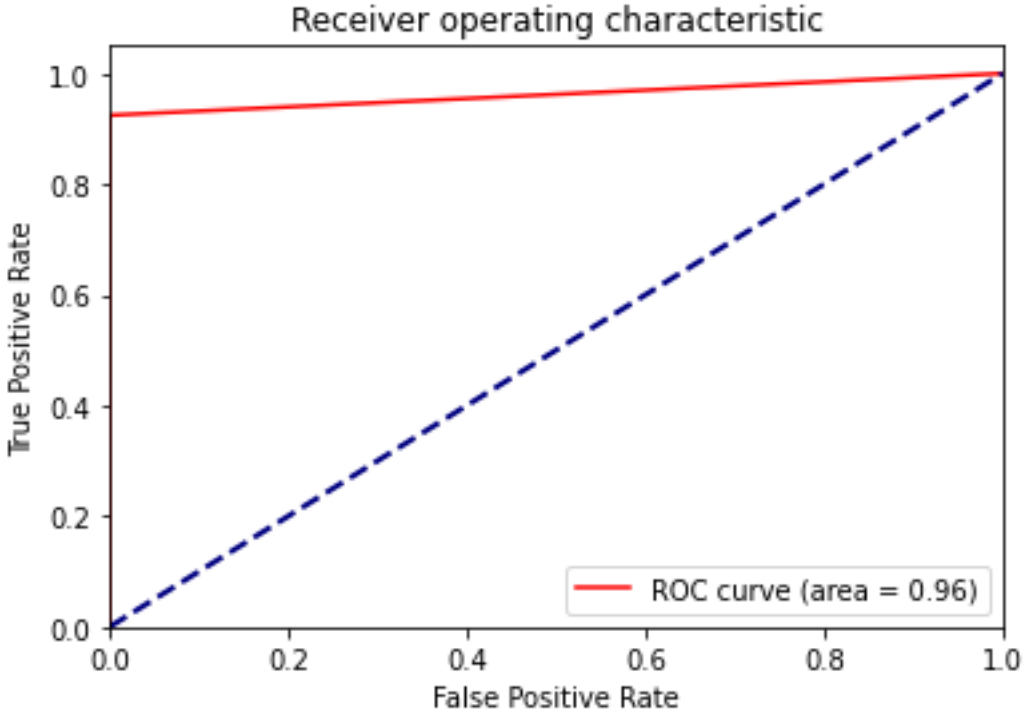



```
[55] from sklearn.metrics import roc_auc_score
     roc_auc_score(y1_test, y_pred_rfc)
```

```
0.9619843818131367
```




Learning Curve RandomForestClassifier
1.000
0.999
0.998
0.997
0.996
0.995
Accuracy Score
Training Score
Cross-Validation Score
5000
10000
15000
20000
25000
30000
35000
40000
Training Size

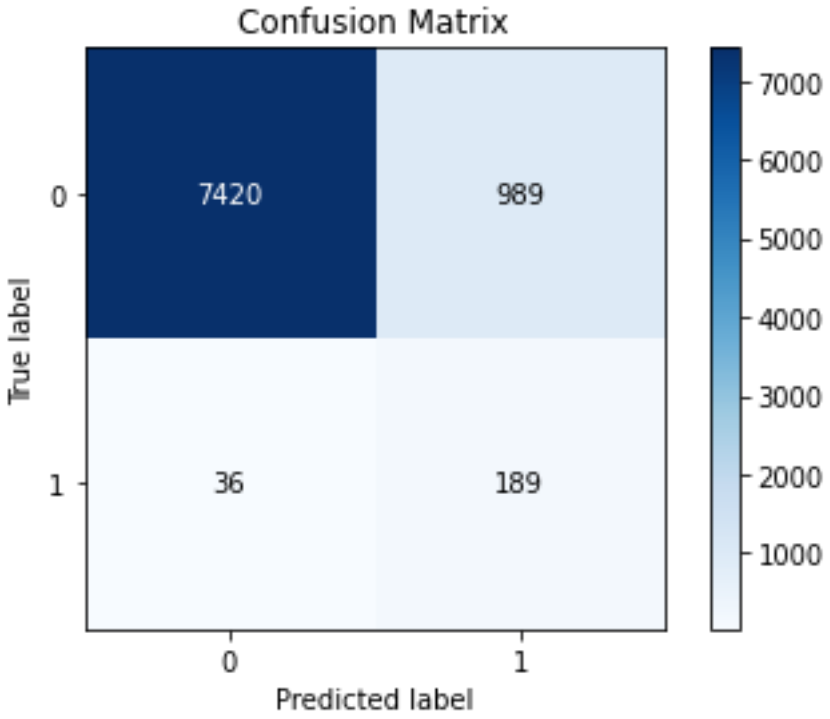

Receiver operating characteristic
1.0
0.8
0.6
0.4
0.2
0.0
True Positive Rate
ROC curve (area = 0.96)
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate

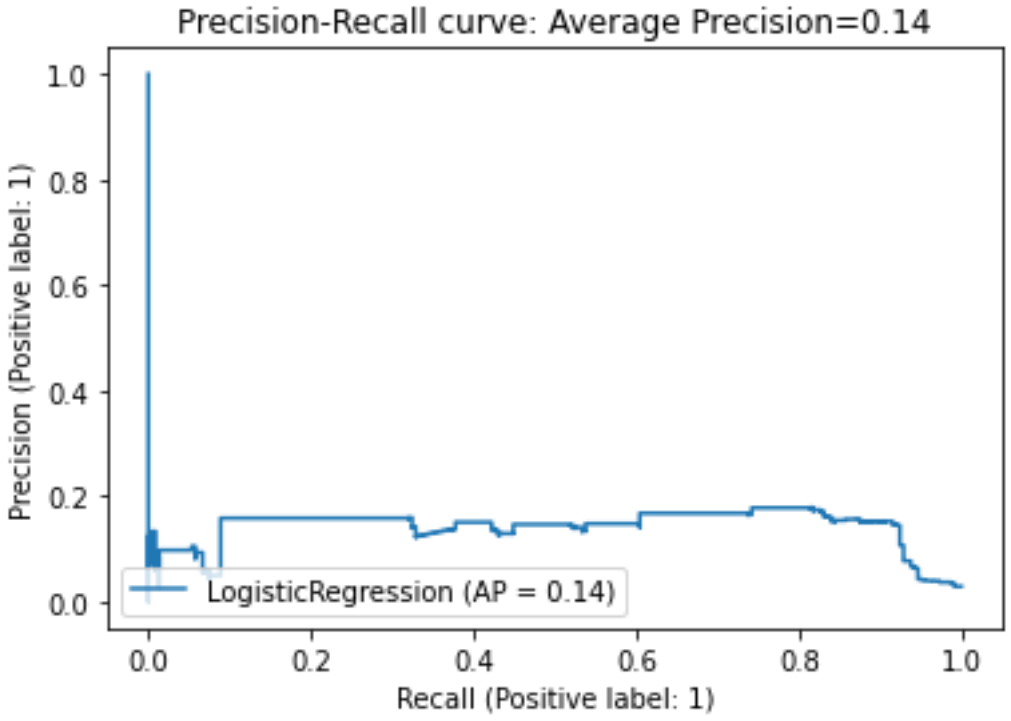
---

3. LINEAR REGRESSION

```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_lg)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_lg)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_lg)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_lg)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_lg)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_lg)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_lg)))
```

```
F1-Score: 0.2694
Precision Score: 0.1604
Recall core: 0.8400
Accuracy Score: 0.8813
Balanced Accurarcy Score: 0.8612
Matthews Corrcoef: 0.3353
Average Precision Score: 0.1389
```



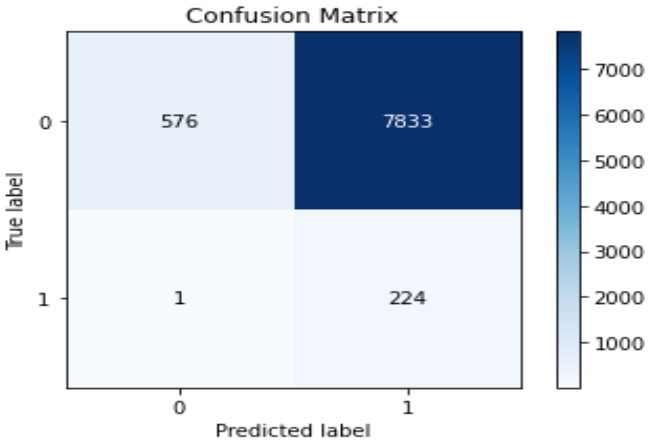


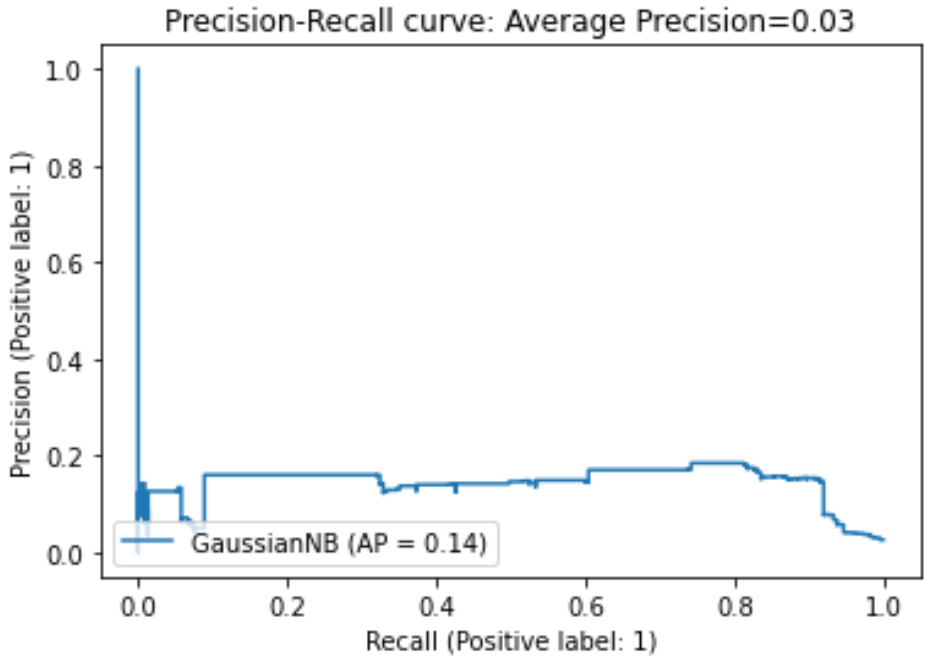
---

4. NAÏVE BAYES

```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_NB)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_NB)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_NB)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_NB)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_NB)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_NB)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_NB)))
```

```
F1-Score: 0.0541
Precision Score: 0.0278
Recall core: 0.9956
Accuracy Score: 0.0927
Balanced Accurarcy Score: 0.5320
Matthews Corrcoef: 0.0409
Average Precision Score: 0.0278
```



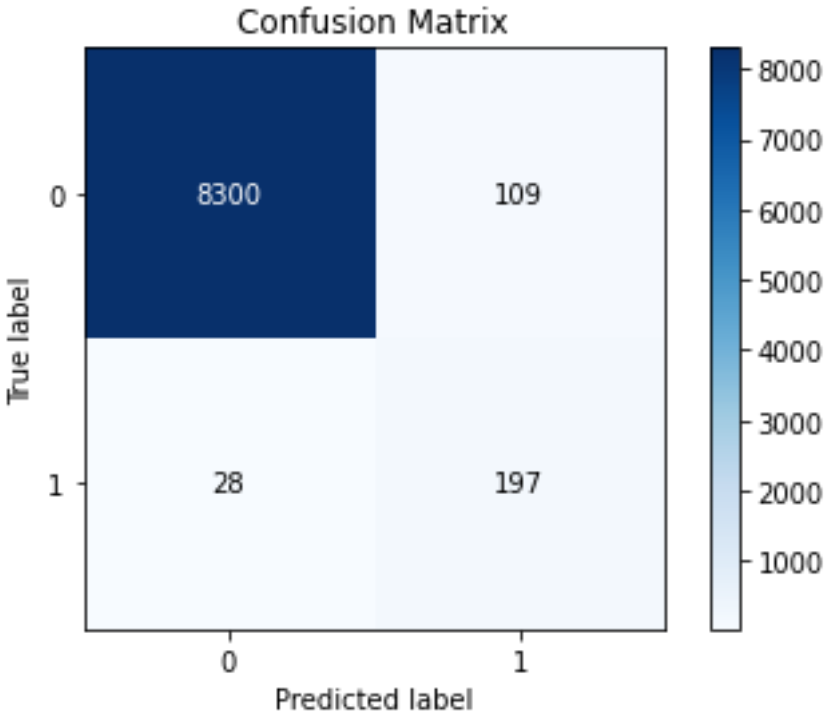


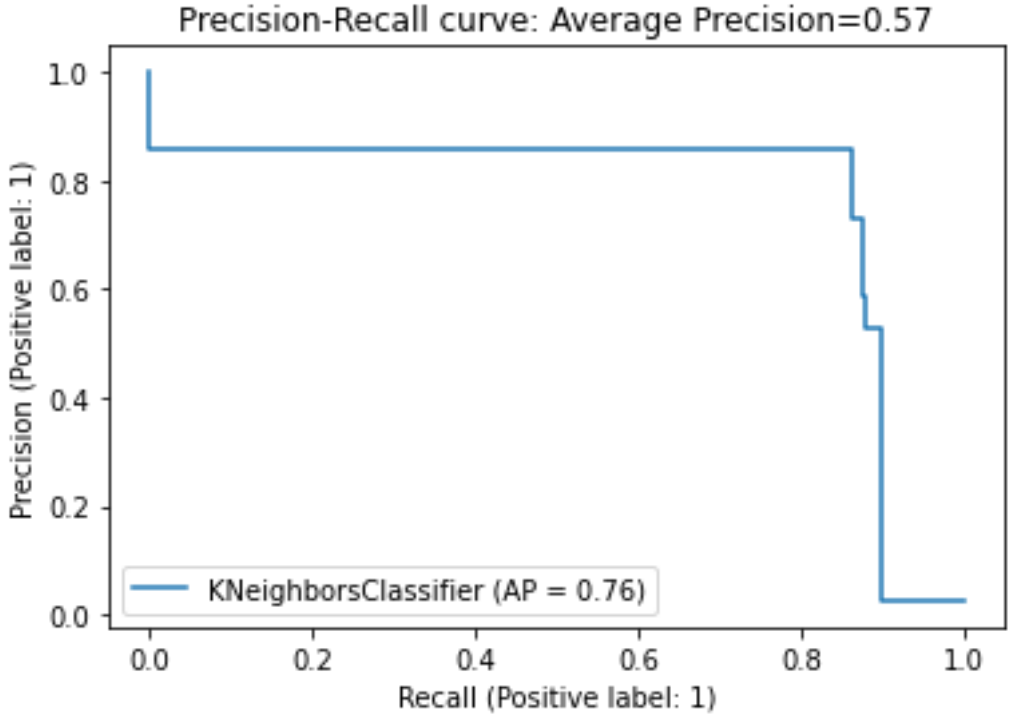
---

5. K NEAREST NEIGHBOUR CLASSIFIER

```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_knn)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_knn)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_knn)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_knn)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_knn)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_knn)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_knn)))
```

```
F1-Score: 0.7420
Precision Score: 0.6438
Recall core: 0.8756
Accuracy Score: 0.9841
Balanced Accurarcy Score: 0.9313
Matthews Corrcoef: 0.7433
Average Precision Score: 0.5669
```


Confusion Matrix
8300
109
28
197
0
1
True label
0
1
Predicted label
8000
7000
6000
5000
4000
3000
2000
1000


Precision-Recall curve: Average Precision=0.57
Precision (Positive label: 1)
Recall (Positive label: 1)
KNeighborsClassifier (AP = 0.76)
0.0
0.2
0.4
0.6
0.8
1.0

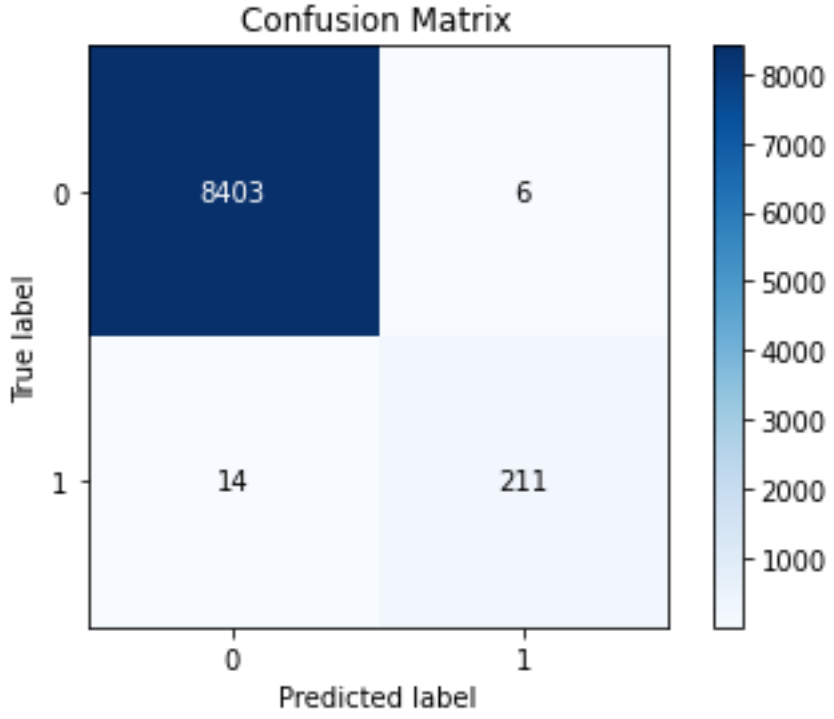
---

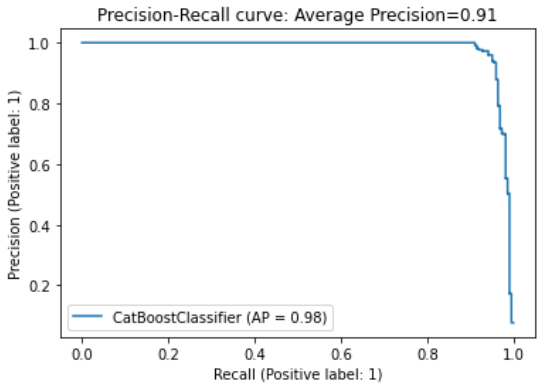
6. CATBOOST

CatBoost is an algorithm for gradient boosting on decision trees.

```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_cat)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_cat)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_cat)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_cat)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_cat)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_cat)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_cat)))
```

```
F1-Score: 0.9548
Precision Score: 0.9724
Recall core: 0.9378
Accuracy Score: 0.9977
Balanced Accurarcy Score: 0.9685
Matthews Corrcoef: 0.9537
Average Precision Score: 0.9135
```



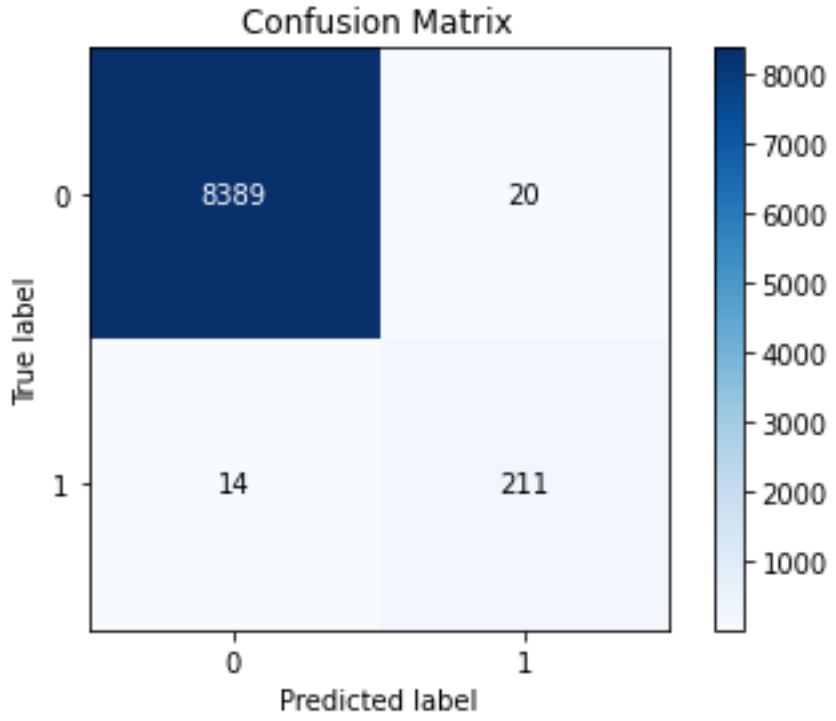


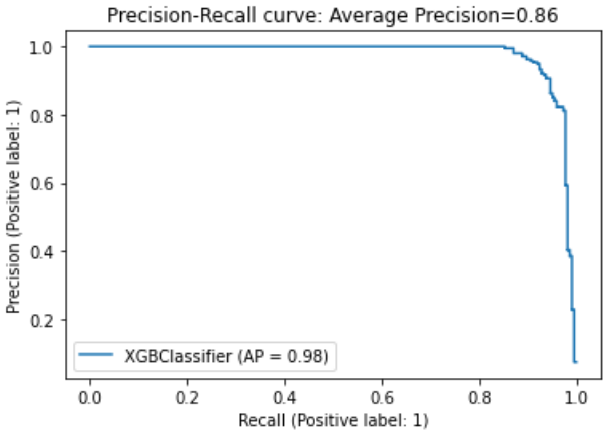
---

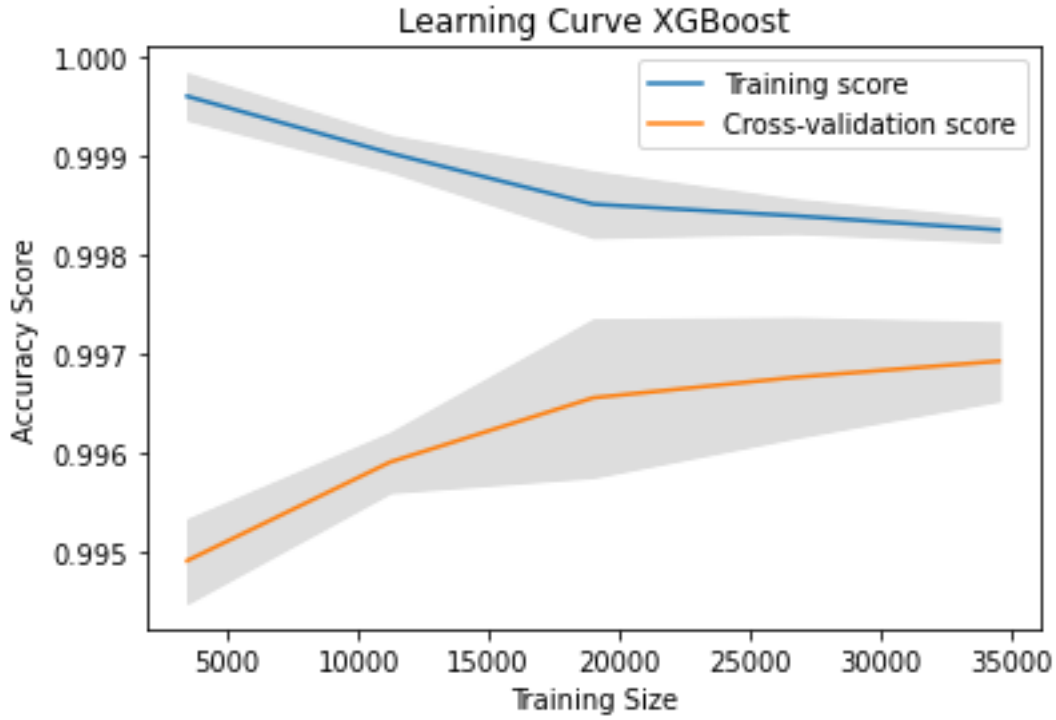
7. EXTREME GRADIENT BOOSTING

```
print("F1-Score: %.4f" % (f1_score(y1_test,y_pred_xgb)))
print("Precision Score: %.4f" %(precision_score(y1_test, y_pred_xgb)))
print("Recall core: %.4f" %(recall_score(y1_test, y_pred_xgb)))
print("Accuracy Score: %.4f" %(accuracy_score(y1_test, y_pred_xgb)))
print("Balanced Accurarcy Score: %.4f" %(balanced_accuracy_score(y1_test, y_pred_xgb)))
print("Matthews Corrcoef: %.4f" %(matthews_corrcoef(y1_test, y_pred_xgb)))
print("Average Precision Score: %.4f" %(average_precision_score(y1_test, y_pred_xgb)))
```

```
F1-Score: 0.9254
Precision Score: 0.9134
Recall core: 0.9378
Accuracy Score: 0.9961
Balanced Accurarcy Score: 0.9677
Matthews Corrcoef: 0.9235
Average Precision Score: 0.8582
```







References for my Github repository of this above program is here

LITERATURE SURVEY

What is Performance metrics?

Performance metrics are used to measure the behavior, activities, and performance of a business. This should be in the form of data that measures required data within a range, allowing a basis to be formed supporting the achievement of overall business goals

A New Malware Classification Framework Based on Deep Learning Algorithms.

Author with year:

ÖMER ASLAN 1 AND ABDULLAH ASIM YILMAZ 2

Received May 10, 2021, accepted June 9, 2021, date of publication June 15, 2021, date of current version June 24, 2021.

Methodolgy:

a novel deep-learning-based architecture is proposed which can classify malware variants based on a hybrid model.

The main contribution of the study is to propose a new hybrid architecture which integrates two wide-ranging pre-trained network models in an optimized manner.

Advantage:

The experimental results show that the suggested method can effectively classify malware with high accuracy which outperforms the state of the art methods in the literature. 97.8% of accuracy is obtained on malimg dataset.

Limitation:

However, Deep Learning method has not been used sufficiently in the cyber security field, especially in malware detection.

Performance metric:

method uses a new hybrid layer that involves two pre-trained models instead of one model.

method uses a new hybrid layer that involves two pre-trained models instead of one model.

2. Automatic malware classification and new malware detection using machine learning

Author with year:

Bao-sheng WANG, Bo YU, Qiu-xi ZHONG

Received June 12, 2016; Revision accepted Sept. 14, 2016; Crosschecked Sept. 15, 2017

Methodolgy:

a machine learning based malware analysis system, which is composed of three modules: data processing, decision making, and new malware detection(clustering). The method uses the texture of the malware to effectively describe the features of the different programs.

gray-scale image, op-code n-gram, and import function.

Advantage:

Based on the state of the malware that has been analyzed, malware analysis can be divided into static and dynamic analysis static analysis are its ability to find an author’s style and profile the code flow.

Dynamic analysis, on the other hand, enables the observation of the running state

of a program in a safe and controlled environment.

Limitation:

static ananlysis is thwarted easily by obfuscation techniques.

dynamic analysis is not affected by encryption, compression, metamorphosis, etc. However, this method spends time not only on debugging a program, but also in tracking and recording the running process of the program. ThereFore, dynamic analysis is usually far more inefficient than static analysis.

Performance metric:

experiment results showed that our system not only improves the accu racy of malware classification effectively, but also discovers new malware effectively.

3. Malware Classification Using Deep Learning Methods Author

with year:

Bugra Cakir

Erodgan Dogdu

ACM SE ’18, March 29–31, 2018, Richmond, KY, USA Methodolgy:

used a static methodology a shallow deep learning-based feature extraction method

(word2vec) is used for representing any given malware based on its opcodes.

Gradient Boosting algorithm is used for the classification task.

k-fold cross-validation is used to validate the model performance without sacrificing a validation split.

Classification sets of computer instructions consists ofa process so called "supervised learning" approach.feature vectors are used forsupervised learning which consists exactly two columns.First columns =>feature of vector describes malware classes. second columns => malware opcodes Advantage:

Deep learning models are shown to work much better in the analysis of long sequences of system calls.

deep learning is more accurate in malware detection.

Limitation: deep learning methods are time consuming in malware detection.whereas

ML machine learning algorithms are fast but not so much accurate.

Signature method is an old technique. However this method is not so effective

due to ever changing malware types.

Performance metric:

malware classification accuracy can easily reach up to between 94% and 96%.

If the cross-validation k-fold value is chosen greater instead of k=5, the accuracy might be higher. a wide range model lookups in the GBM tree search, leading to better accuracy

4. MALWARE CLASSIFICATION WITH RECURRENT NETWORKS

Author wth year:

Razvan Pascanu, Jack W Stokest Hennineh Sanossian, Mady Marinescu, Anil Thomas 2015

Methodology:

These models are trained in an un supervised fashion. A standard classifier uses these features to detect malicious files. We explore a few variants of ESNs and RNNs for the projection stage, including Max-Pooling and Half-Frame models which we propose.

Advantage:

learn to specialize in detecting different and potentially reordered temporal patterns.

Limitation:

the ESN models outperform the RNNs in the majority of the experiments.

Max-Pooling is more useful in this sit uation, as the ESN does not utilize the hidden state in the same way the RNN does.

Performance metrics:

Compared to the standard trigram of events model, it improves the true positive rate by 98.3% at a false positive rate of 0.1 % combining a recurrent model with a standard classifier can improve the true positive rate by a factor of three compared to a bag-of-events model.a factor of two given by a bag-of-trigrams model.

5. Ransomware Classification and Detection With Machine

Learning Algorithms.

Author wth year:

Mohammad Masum, Md Jobair Hossain Faruk, Hossain Shahriar

Kai Qian§, Dan Lo§, Muhaiminul Islam Adnan

January 2022

Methodology:

In this paper, we present a feature selection-based framework with adopting different machine learning algorithms including neural network-basedarchitectures to classify the security level for ransomware detection and prevention. We applied multiple machine learning algorithms: Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Logistic Regression (LR) as well as Neural Network (NN)-based classifiers on a selected number of features for ransomware classification. reported different evaluation metrics such as accuracy, F-beta score, precision, recall and area under ROC curve to assess the models' performance.

Advantage:

To improve the limitation, a static and dynamic-based or Behavior-based framework was introduced by [11] where analysis static-based technique analyze the application's code to determine malicious activities and dynamic-based analysis on the other hand monitoring the processes to determine the behavior of malicious intent and will be flagged as suspicious and terminated.

Limitation:

Both static and dynamic-based analysis has limitation in terms of the inability to detect unknown malware and ineffectiveness against code obfuscation, high variant output, and targeted attacks.

Performance metrics:

The model was evaluated, and the results indicate its performance in detecting ransomware between 76% to 97%.

According to the evaluation, the proposed LSTM based framework achieved 96.67% accuracy in classifying the ransomware behavior automatically from a large volume of malware datasets.

According to the evaluation report, a combination of DNN and LSTM provide effective in detecting new malware and achieved 91.63% accuracy.

The experimental results demonstrate that the Random Forest classifier outperformed other classifiers by achieving the highest accuracy, F-beta, and precision scores with reasonable consistency in the 10-fold cross-validation results.

CONCLUSION

Effectively detecting malware variants continues to pose a severe danger to cyber security despite extensive research on malware detection and classification.

Code obfuscation and packaging strategies make the domain difficult to Malware identification is an extremely difficult operation. This essay presented a cutting-edge deep learning architecture to find different malware types. The suggested architecture makes advantage of hybrid strategy This strategy comprises numerous thorough networks that have already been trained and rely on transfer learningmethod. At first, the malware data collecting was achieved by combining many thorough datasets. Then, Pre-trained networks are used to extract the features. Lastly, the deep learning architecture's training phase is carried out in relation to supervised learning.

Finally we can say that seeing above results that Machine Learning Algorithms is way better than convolutional neural network (CNN) because if you see the accuracy of Machine Learning Algorithm it is greater than 98% but the CNN model gives only 50.02% so using certain dataset we can say that Machine Learning Algorithm is way better than CNN. The test results confirmed that the Machine learning method can effectively classify malware with high precision, recall, accuracy ,f-score, matthews correlation coefficient, Specificity , Sensitivity, balanced accuracy score, and ROC. Summary of all the model created are shown in the image below.

	AVERAGE PRECISION	FALES NEGATIVES	BALANCED ACCURACY
Decision Tree	0.87	11	0.97
Random Forest	0.92	11	0.98
XG Boost	0.86	15	0.96
Catboost	0.89	8	0.96
Logistic Regression	0.14	988	0.86
Naive Bayes	0.03	7833	0.53
K Nearest Neighbors	0.67	55	0.92

By seeing this we can say that tree based model, catboost algorithm have a robust and good performance in detecting malware

REFERENCES :-

Base Research Paper was

A New Malware Classification Framework Based on Deep Learning Algorithms

By ÖMER ASLAN 1 AND ABDULLAH ASIM YILMAZ 2.

Reference from YouTube:

Malware Detection with Convolutional
Neural Networks by data bowl recipes Machine Learning for Malware Analysis (Part1)

Machine Learning for Malware Analysis (Part 2)