Configuration Manual

MSc Research Project
Cyber Security

Yashvardhan Pant
Student ID: X21132399

School of Computing
National College of Ireland

Supervisor: Prof. Imran Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Yashvardhan Pant		
Student ID:	X21132399		
Programme:	MSc in Cyber Security	**Year:**	2022-2023
Module:	MSc Internship		
Lecturer:	Prof. Imran Khan		
Submission Due Date:	15/12/2022		
Project Title:	Malware Detection in executable files using XGBoost Algorithm		
Word Count:	………………980…………	**Page Count:**	……………10……..……..………

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Yashvardhan Pant

Date: …………15/12/2022………………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Malware Detection in executable files using XGBoost Algorithm

Yashvardhan Pant
Student ID: x21132399

1 Introduction

2

The study conducted as part of the Academic Research Project is summarized and analyzed in detail inside the configuration manual booklet. This document will describe the methods and tools that were used throughout the project's development and research phases. The approach followed throughout the development phase and the findings from the study will be detailed in the implementation section. This guidebook also includes information on the internship assignment report.

3 System Configuration

The system used while performing the activity was personal as the research project was Remote. The configuration of the system is as follows:

3.1 Hardware Configuration

- Operating system: Windows 11
- Processor: Intel i7-11th gen
- System Compatibility: 64-bit
- Hard Disk: 1 TB SSD
- RAM: 16 GB

3.2 Software Configurations:

Prior to start the model building phase following software, following tools and libraries were installed in the system:

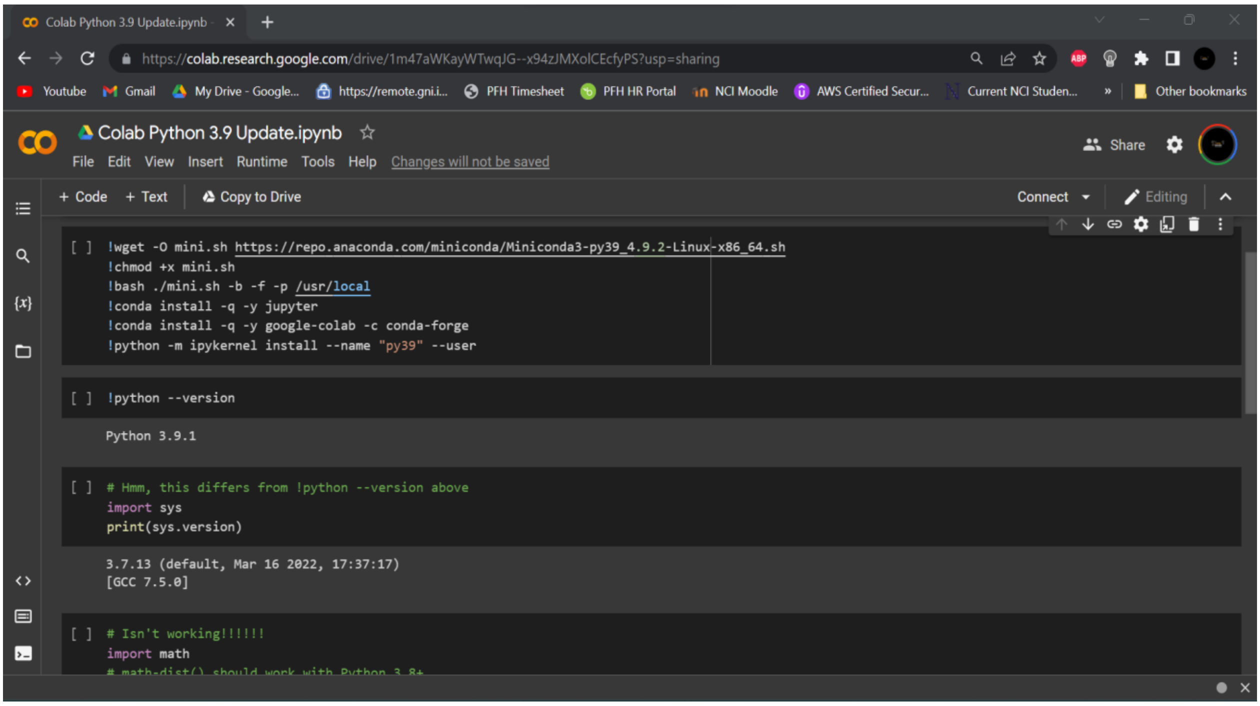
Software/Tools	Version	Information
Python	3.9	Python is utilized to import and implement Machine Learning model in this project.
Google Collab	https://colab.research.google.com/	With Colab, anyone can write and run any Python script in the browser. It is especially helpful for teaching, machine learning, and data analysis.
Cuckoo Sandbox	2.0.7	Cuckoo is an automated method for the investigation of malware that is free

		source. While the malicious software is working inside of an isolated operating system, it is utilized to automatically execute and analyze files and gather complete analysis findings that define what the malware performs.
NumPy	1.23.5	NumPy is an abbreviation for Numerical Python, and it is a fundamental scientific computing package in Python. It offers efficient multi-dimensional array objects as well as a variety of functions for working with these array objects.
Sci-Kit Learn	1.2.0	Scikit-learn is an open-source Python package for data analysis that offers a variety of unsupervised and supervised learning techniques.
XGBoost	1.7.2	XGBoost is a distributed gradient boosting toolkit that has been developed to be very efficient, adaptable, and portable. It uses the Gradient Boosting framework to construct machine learning algorithms. XGBoost offers parallel tree boosting to address numerous data science tasks quickly and accurately.

4 Implementation

This section contains a step-by-step instruction for running the project on any Windows machine.

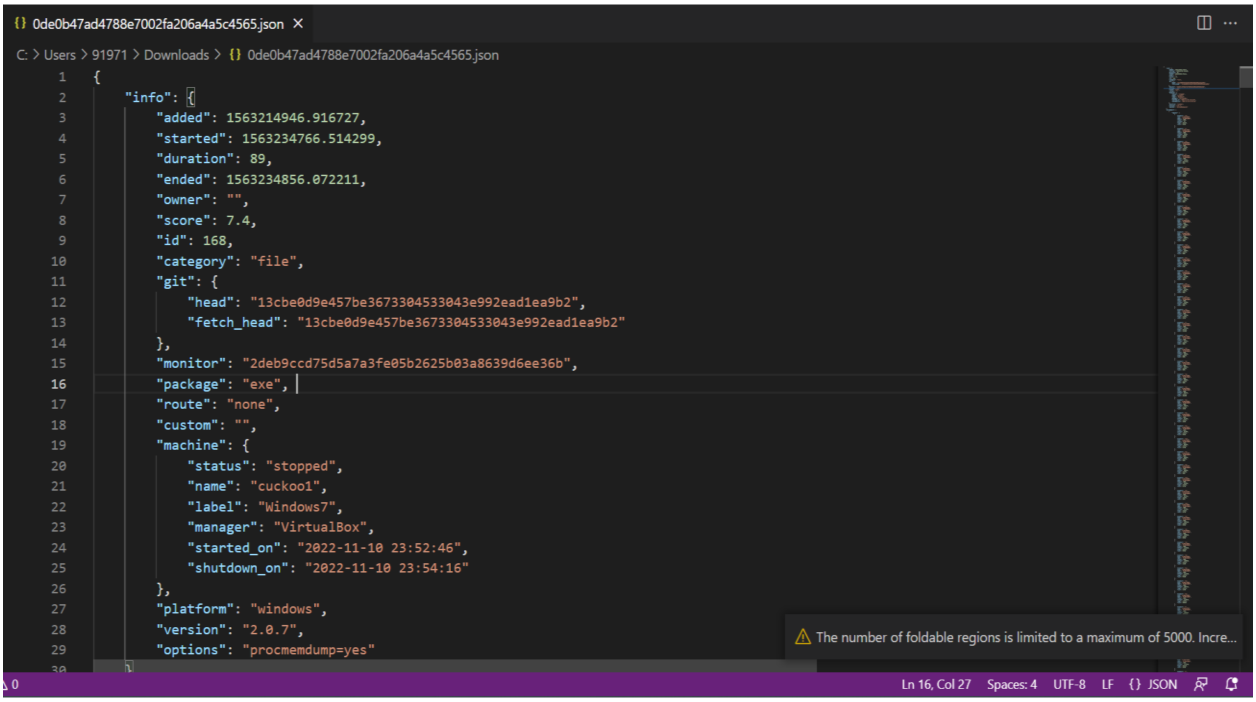
1. Browse the Google Collab for python URL: https://colab.research.google.com. The following User Interface would open:



```
[ ] !wget -O mini.sh https://repo.anaconda.com/miniconda/Miniconda3-py39_4.9.2-Linux-x86_64.sh
    !chmod +x mini.sh
    !bash ./mini.sh -b -f -p /usr/local
    !conda install -q -y jupyter
    !conda install -q -y google-colab -c conda-forge
    !python -m ipykernel install --name "py39" --user

[ ] !python --version

    Python 3.9.1

[ ] # Hmm, this differs from !python --version above
    import sys
    print(sys.version)

    3.7.13 (default, Mar 16 2022, 17:37:17)
    [GCC 7.5.0]

[ ] # Isn't working!!!!!!
    import math
    # math.dist() should work with Python 3.8+
```

Fig.1 Google Collab UI

2. Next step is to parse the JSON based output report from Cuckoo Sandbox, into Python environment in Google collab UI :

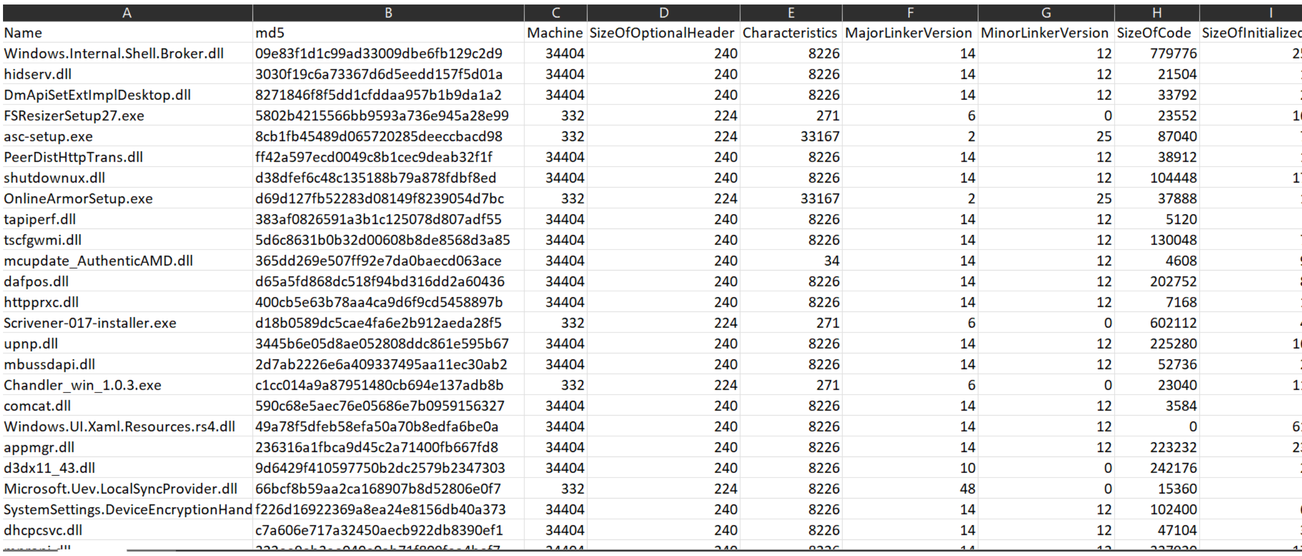


0de0b47ad4788e7002fa206a4a5c4565.json
C: > Users > 91971 > Downloads > 0de0b47ad4788e7002fa206a4a5c4565.json
The number of foldable regions is limited to a maximum of 5000. Incre...
0
Ln 16, Col 27
Spaces: 4
UTF-8
LF
{} JSON

```
{
    "info": {
        "added": 1563214946.916727,
        "started": 1563234766.514299,
        "duration": 89,
        "ended": 1563234856.072211,
        "owner": "",
        "score": 7.4,
        "id": 168,
        "category": "file",
        "git": {
            "head": "13cbe0d9e457be3673304533043e992ead1ea9b2",
            "fetch_head": "13cbe0d9e457be3673304533043e992ead1ea9b2"
        },
        "monitor": "2deb9ccd75d5a7a3fe05b2625b03a8639d6ee36b",
        "package": "exe",
        "route": "none",
        "custom": "",
        "machine": {
            "status": "stopped",
            "name": "cuckoo1",
            "label": "Windows7",
            "manager": "VirtualBox",
            "started_on": "2022-11-10 23:52:46",
            "shutdown_on": "2022-11-10 23:54:16"
        },
        "platform": "windows",
        "version": "2.0.7",
        "options": "procmemdump=yes"
    }
```

Fig.2 JSON File extracted from Cuckoo Sandbox

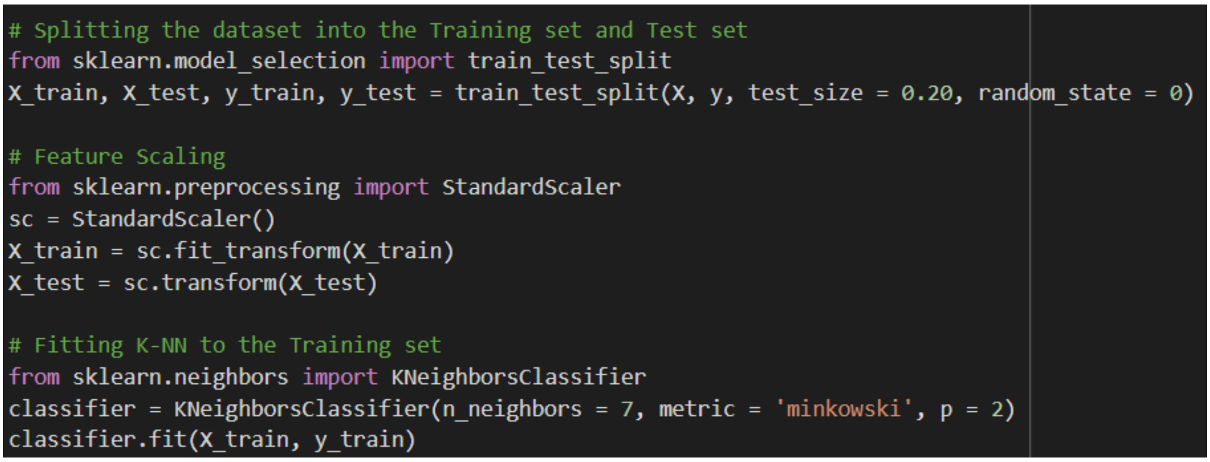
3. PE header information were extracted from JSON by parsing them through extract.py. Extrac.py is python script which converts PE header information to a CSV format. Following is the screenshot of extracted dataset:



Name	md5	Machine	SizeOfOptionalHeader	Characteristics	MajorLinkerVersion	MinorLinkerVersion	SizeOfCode
Windows.Internal.Shell.Broker.dll	09e83f1d1c99ad33009dbe6fb129c2d9	34404	240	8226	14	12	779776
hidserv.dll	3030f19c6a73367d6d5eedd157f5d01a	34404	240	8226	14	12	21504
DmApiSetExtImplDesktop.dll	8271846f8f5dd1cfddaa957b1b9da1a2	34404	240	8226	14	12	33792
FSResizerSetup27.exe	5802b4215566bb9593a736e945a28e99	332	224	271	6	0	23552
asc-setup.exe	8cb1fb45489d065720285deeccbacd98	332	224	33167	2	25	87040
PeerDistHttpTrans.dll	ff42a597ecd0049c8b1cec9deab32f1f	34404	240	8226	14	12	38912
shutdownux.dll	d38dfef6c48c135188b79a878fdbf8ed	34404	240	8226	14	12	104448
OnlineArmorSetup.exe	d69d127fb52283d08149f8239054d7bc	332	224	33167	2	25	37888
tapiperf.dll	383af0826591a3b1c125078d807adf55	34404	240	8226	14	12	5120
tscfgwmi.dll	5d6c8631b0b32d00608b8de8568d3a85	34404	240	8226	14	12	130048
mcupdate_AuthenticAMD.dll	365dd269e507ff92e7da0baecd063ace	34404	240	34	14	12	4608
dafpos.dll	d65a5fd868dc518f94bd316dd2a60436	34404	240	8226	14	12	202752
httpprxc.dll	400cb5e63b78aa4ca9d6f9cd5458897b	34404	240	8226	14	12	7168
Scrivener-017-installer.exe	d18b0589dc5cae4fa6e2b912aeda28f5	332	224	271	6	0	602112
upnp.dll	3445b6e05d8ae052808ddc861e595b67	34404	240	8226	14	12	225280
mbussdapi.dll	2d7ab2226e6a409337495aa11ec30ab2	34404	240	8226	14	12	52736
Chandler_win_1.0.3.exe	c1cc014a9a87951480cb694e137adb8b	332	224	271	6	0	23040
comcat.dll	590c68e5aec76e05686e7b0959156327	34404	240	8226	14	12	3584
Windows.UI.Xaml.Resources.rs4.dll	49a78f5dfeb58efa50a70b8edfa6be0a	34404	240	8226	14	12	0
appmgr.dll	236316a1fbca9d45c2a71400fb667fd8	34404	240	8226	14	12	223232
d3dx11_43.dll	9d6429f410597750b2dc2579b2347303	34404	240	8226	10	0	242176
Microsoft.Uev.LocalSyncProvider.dll	66bcf8b59aa2ca168907b8d52806e0f7	332	224	8226	48	0	15360
SystemSettings.DeviceEncryptionHand	f226d16922369a8ea24e8156db40a373	34404	240	8226	14	12	102400
dhcpcsvc.dll	c7a606e717a32450aecb922db8390ef1	34404	240	8226	14	12	47104

Fig.3 Extracted CSV containing variables to be analyzed by Machine Learning algorithms

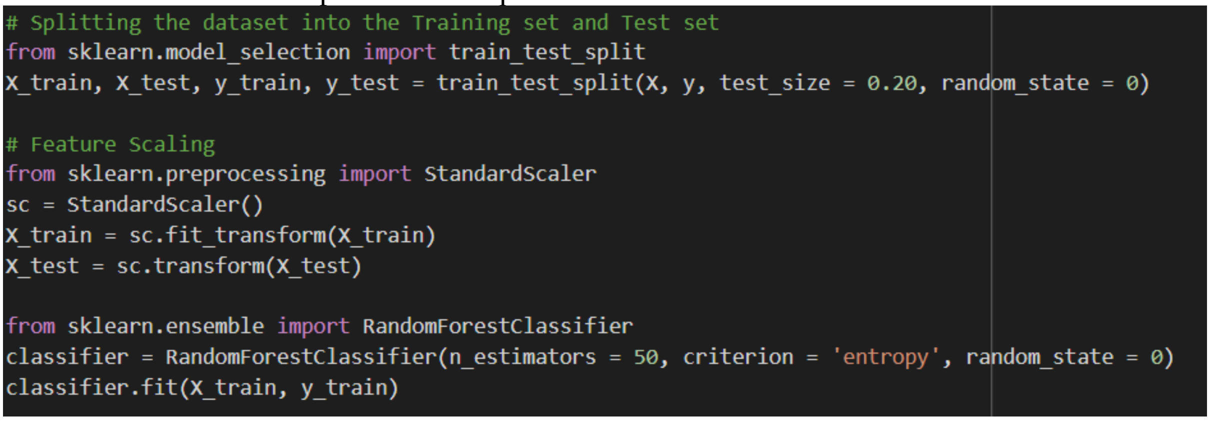
4. Now, apply the Machine Learning algorithms (KNN , Random Forest, XGBoost) on extracted CSV file in step 3, as shown below :



```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting K-NN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 7, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)
```

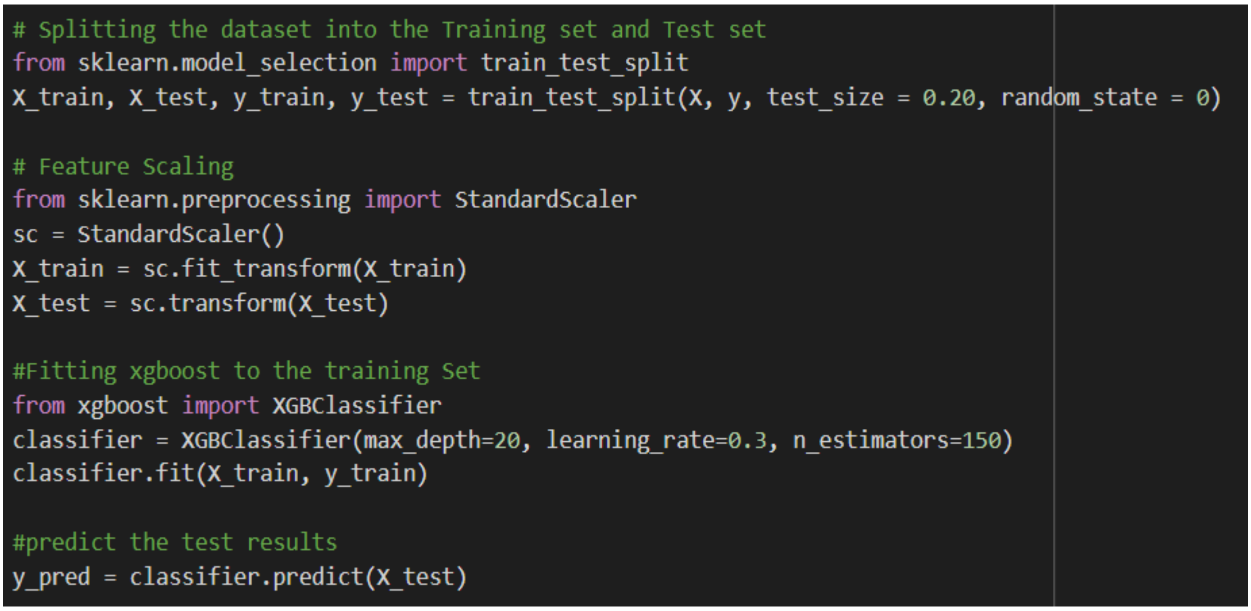
Fig.4 Applying K -NN ML Algorithm on extracted CSV file

5. In new notebook first import all the required libraries.



```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 50, criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
```

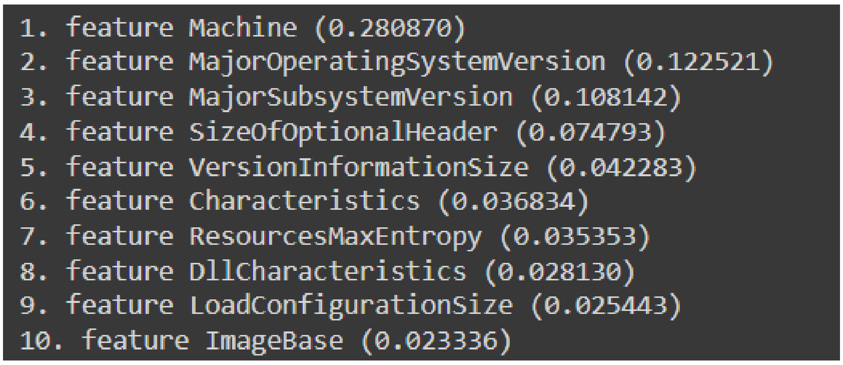
Fig.5 Applying Random Forest ML Algorithm on extracted CSV file



```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

#Fitting xgboost to the training Set
from xgboost import XGBClassifier
classifier = XGBClassifier(max_depth=20, learning_rate=0.3, n_estimators=150)
classifier.fit(X_train, y_train)

#predict the test results
y_pred = classifier.predict(X_test)
```

Fig.6 Applying XGNoost ML Algorithm on extracted CSV file

6. Next, use Feature selection method to identify variables that affect the .As shown below, out 57 variables, 10 variables have been selected on which Machine Learning algorithms (KNN , Random Forest, XGBoost) will be applied again.



```
1. feature Machine (0.280870)
2. feature MajorOperatingSystemVersion (0.122521)
3. feature MajorSubsystemVersion (0.108142)
4. feature SizeOfOptionalHeader (0.074793)
5. feature VersionInformationSize (0.042283)
6. feature Characteristics (0.036834)
7. feature ResourcesMaxEntropy (0.035353)
8. feature DllCharacteristics (0.028130)
9. feature LoadConfigurationSize (0.025443)
10. feature ImageBase (0.023336)
```

Fig.7 Feature selection -10 Variables selected

7. After applying the 3 ML algorithms on the 10 variables identified in step 6. The data from our dataset was then utilized to train a model or algorithm, and the results were gathered. The confusion matrix for each model was used to compute accuracy, precision, recall, and F1-score.
8. Confusion Matrix are created i.e. the final output of the models are plotted for both static and dynamic model as show below:
   a) Without Feature Selection method:

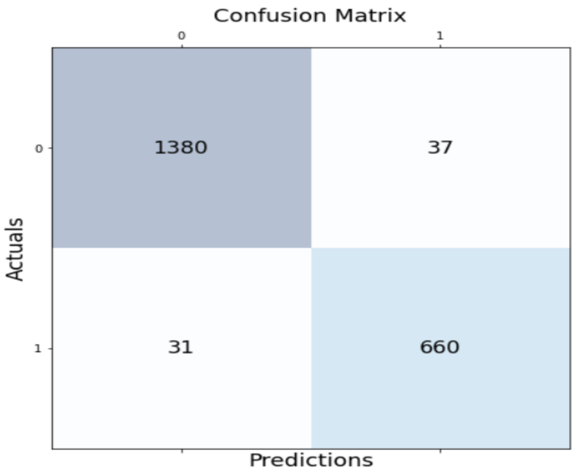


Fig.8 Confusion Matrix- KNN Algorithm without Feature Selection variables

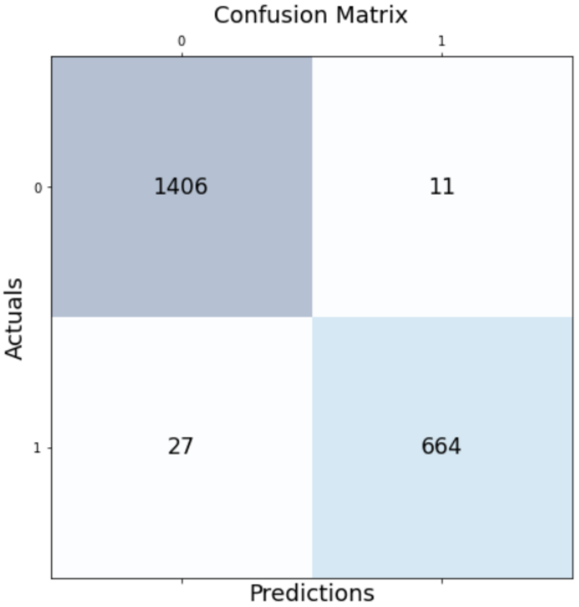


Fig.9 Confusion Matrix- Random Forrest Algorithm without Feature Selection variables

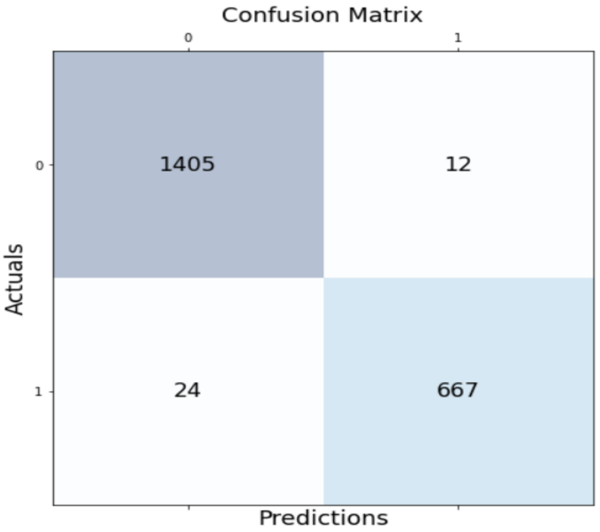


Fig.10 Confusion Matrix- XGBoost Algorithm without Feature Selection variables

b) With Feature Selection:

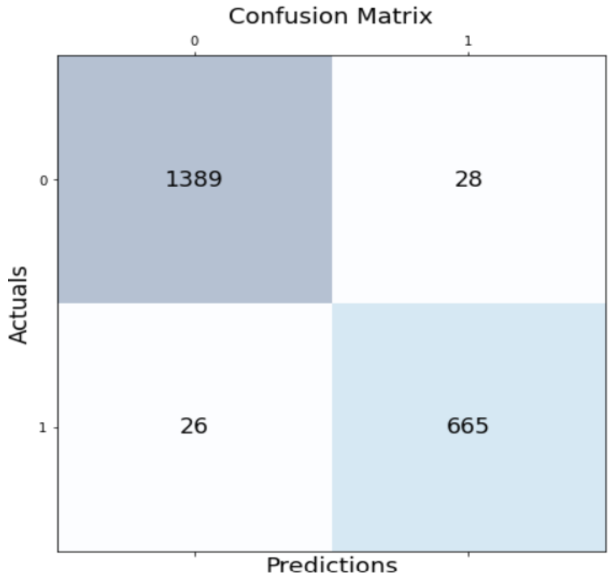


Fig.11 Confusion Matrix- XGBoost Algorithm with Feature Selection variables

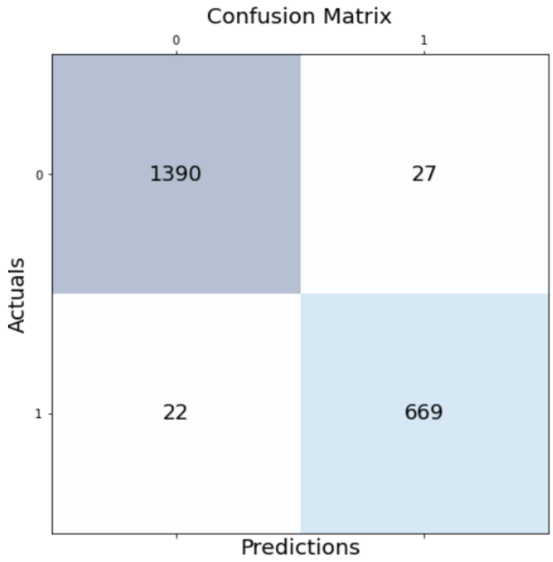


Fig.12 Confusion Matrix- XGBoost Algorithm with Feature Selection variables

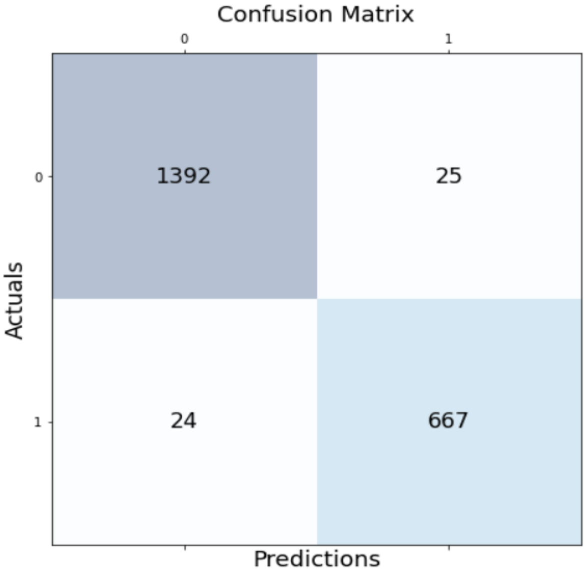


Fig.13 Confusion Matrix- XGBoost Algorithm with Feature Selection variables