National College of Ireland

Configuration Manual

MSc Research Project
Cyber Security

Emin Varghese
Student ID: x23154250

School of Computing
National College of Ireland

Supervisor: Arghir Nicolae Moldovan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name**:**	Emin Varghese		
Student ID:	X23154250		
Programme:	MSc. Cybersecurity	Year**:**	2024 - 25
Module:	MSc. Research Project		
Supervisor:	Arghir Nicolae Moldovan		
Submission Due Date**:**	12/12/2024		
Project Title**:**	Detection of Phishing URLs using Machine Learning		
Word Count:	680 words excluding references	Page Count: 6	

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: Emin Varghese

Date: 12/12/2024

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

Emin Varghese
Student ID : x23154250

1 Introduction

The configuration manual covers all the required tools and frameworks used to implement the research model. This manual is subdivided into different categories. Section 2 refers to environment setup needed, Section 3 discusses about tools and software's being used and Section 4 discuss about implementation of the project.

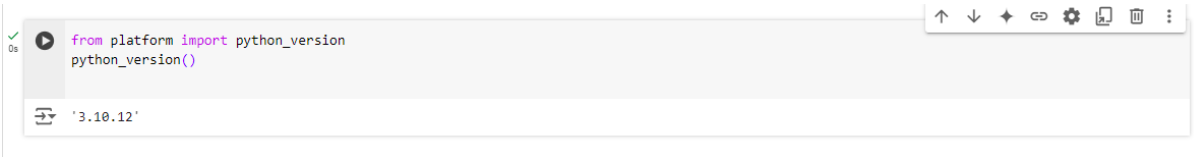
2 Environmental Setup

Processor : Intel i5
Memory : 16GB RAM
Programming Language : Python3
Python Environment : Jupyter Notebook, Anaconda Navigator, Google Colab

3 Tools and Software Used

Software Used : Google Colab (cloud-based), Python 3.10.12

Google Colab also known as Colaboratory is a powerful free Google cloud-based service for writing and running Python codes. It is integrated with Google Drive and pre-configured environment for the wide usage of data analysis, machine learning and prototyping.



4 Implementation

Step 1 : Go to Google Colab and login using your google account

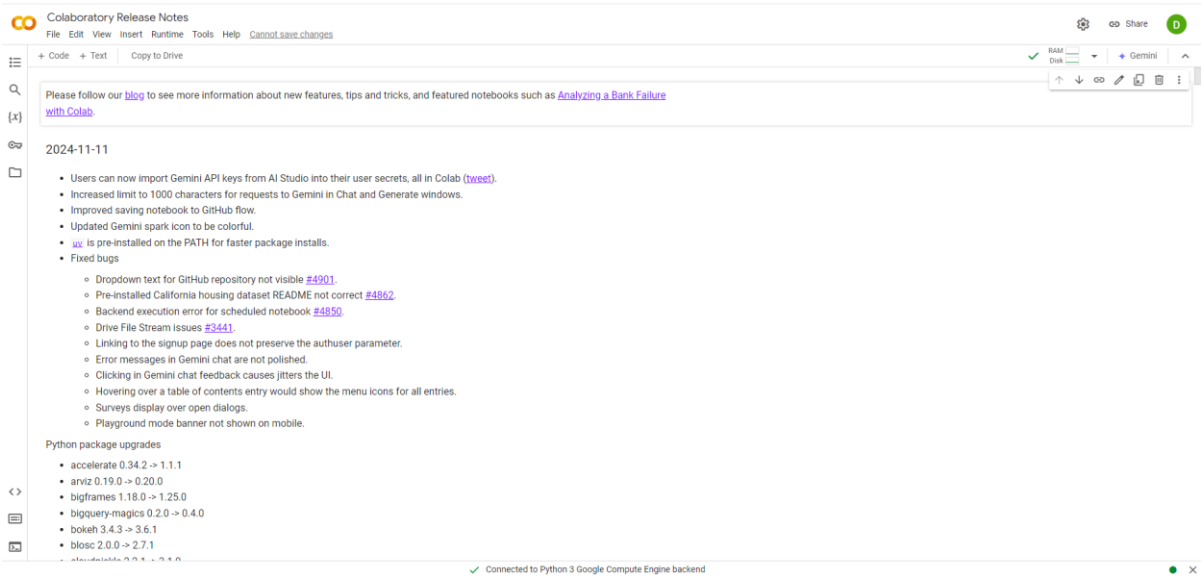


Figure 1 : Homepage of Google Colab

Step 2 : The dataset is downloaded

Step 3 : Importing essential libraries for the execution of file and machine learning algorithms.

```
#****************************************Import*************************************************
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import warnings
from sklearn.model_selection import cross_val_score
from sklearn import metrics
from sklearn import preprocessing
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
import time
```

Figure 2 : Importing python libraries

Step 4 : Performing feature extraction by applying preprocessing or transformation steps such as normalization, scaling and encoding

```python
import pandas as pd
import re
from urllib.parse import urlparse
import random
import string

# Load the dataset
file_path = '/content/Phishing URLs.csv'
phishing_urls = pd.read_csv(file_path)

# Specify the column name that contains URLs
url_column_name = 'url'  # Replace with the actual column name in your dataset

# Generate random file names
def generate_random_filename():
    return ''.join(random.choices(string.ascii_letters + string.digits, k=8)) + '.txt'

# Insert "File Name" column
phishing_urls.insert(0, 'File Name', [generate_random_filename() for _ in range(len(phishing_urls))])

# Feature extraction functions
def url_length(url):
    return len(url)
```

Figure 3 : Feature Extraction

```python
# Apply feature extraction
features_df = extract_features(phishing_urls, url_column_name)

# Save the resulting features to a new CSV file
output_path = 'Extracted_Phishing_URL_Features.csv'
features_df.to_csv(output_path, index=False)

# Display a preview of the extracted features
print(features_df.head())
```

```
      File Name                                              url      Type  \
0  bMqbUwbE.txt  https://docs.google.com/presentation/d/e/2PACX...  Phishing
1  DzsIlkln.txt    https://btttelecommunniccatiion.weeblysite.com/  Phishing
2  86ejFQdw.txt                          https://kq0hgp.webwave.dev/  Phishing
3  hmYaT0QF.txt  https://brittishtele1bt-69836.getresponsesite....  Phishing
4  WUOfKmTr.txt         https://bt-internet-105056.weeblysite.com/  Phishing

   URLLength                                  Domain  DomainLength  \
0        178                          docs.google.com            15
1         47     btttelecommunniccatiion.weeblysite.com            38
2         27                         kq0hgp.webwave.dev            18
3         50  brittishtele1bt-69836.getresponsesite.com            41
4         42          bt-internet-105056.weeblysite.com            33

   IsDomainIP  TLD  URLSimilarityIndex  CharContinuationRate  ...  Pay  \
0           0  com           0.184430              7.416667  ...    0
1           0  com           0.239637              6.714286  ...    0
2           0  dev           0.838010              3.857143  ...    0
3           0  com           0.171121              6.250000  ...    0
4           0  com           0.442283              4.666667  ...    0
```

Figure 4 : Preview of the extracted features

Step 5 : Loading the dataset with proper file path

```
df=pd.read_csv("/content/Extracted_Phishing_URL_Features.csv")
```

Figure 5 : File Path

Step 6 : Train the model by feeding the preprocessed data into machine learning or deep learning models

```
df = df.reset_index(drop=True)
LABEL = df.iloc[:,-1:].columns[0]
cols = ['LineOfCode', 'NoOfExternalRef', 'LargestLineLength', 'URLLength', 'NoOfImage', 'NoOfJS', 'NoOfSelfRef', 'NoOfCSS',
        'URLCharProb', 'CharContinuationRate', 'LetterRatioInURL', 'IsHTTPS', 'SpacialCharRatioInURL','NoOfEmptyRef',
        'NoOfOtherSpecialCharsInURL', 'HasDescription', 'HasSocialNet', 'DomainLength', 'DegitRatioInURL','NoOfDegitsInURL',
        'HasCopyrightInfo', 'NoOfLettersInURL','TLDLegitimateProb', 'DomainTitleMatchScore', 'IsResponsive',
        'HasHiddenFields', 'HasSubmitButton','NoOfSubDomain', 'HasFavicon', 'HasTitle',LABEL]
df = pd.DataFrame(df[cols]).copy()
df30, df70=train_test_split(df,test_size=.1)

df30 = df30.reset_index(drop=True)
df70 = df70.reset_index(drop=True)

yTrain = pd.DataFrame(df30[LABEL]).copy()
df30.drop(LABEL, axis=1, inplace=True)
xTrain = pd.DataFrame(df30).copy()

dfX = pd.DataFrame(df70[cols]).copy()
dfX.drop(LABEL, axis=1, inplace=True)
dfY = pd.DataFrame(df70[LABEL]).copy()
print("Finished Train-Test Split.")
len(xTrain.axes[0])
```

```
Finished Train-Test Split.
49325
```

```
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.linear_model import SGDClassifier
Models = []
Models.append(("BernoulliNB",BernoulliNB(alpha= 0.01, binarize= 0.0, class_prior= None, fit_prior= False, force_alpha= True)))
Models.append(("PassiveAggressive",PassiveAggressiveClassifier(C= 4, fit_intercept= False, n_iter_no_change= 50, n_jobs= 10, shuffle= True, verbose= 0)))
Models.append(("SGDClassifier",SGDClassifier(alpha= 0.01, eta0= 100, n_iter_no_change= 50, n_jobs= 1)))

start = time.time()
for name, model in Models:
    model.fit(xTrain, yTrain)
end = time.time()
OneTime = int(end - start)
print("Finished first training: ", OneTime)
```

Figure 6 : Model Training

Step 7 : Evaluation of the Model using test dataset

```
newRow = {}
newRow['Mode'] = "HighSecurity"
newRow['TP'] = TP
newRow['TN'] = TN
newRow['FP'] = FP
newRow['FN'] = FN
CM = pd.concat([CM, pd.DataFrame([newRow])], ignore_index=True)
```

```
4955   1   0   525
Accuracy    :  0.9042145593869731
Precision   :  1.0
Sensitivity :  0.9041970802919708
F1 Score    :  0.9496885481552467
MCC    :  0.04146089550490179
```

```
[ ] Metrics
```

	Mode	Accuracy	Precision	Sensitivity	F1Score	MCC	OneTime	IncTime
0	LowSecurity	0.999818	0.999818	1.000000	0.999909	0.000000	1	28
1	BestSecurity	0.999818	0.999818	1.000000	0.999909	0.000000	1	28
2	HighSecurity	0.904215	1.000000	0.904197	0.949689	0.041461	1	28

```
[ ] CM
```

	Mode	TP	TN	FP	FN
0	LowSecurity	5480	0	1	0
1	BestSecurity	5480	0	1	0
2	HighSecurity	4955	1	0	525

Figure 7 : Evaluation of the Model

- **CharConRate.ipynb** : Focused on analysing character-based metrics
- **Entropy_Domain_URL.ipynb** : Computes entropy metrics for domain names or URLs
- **IdentifyClassifiers.ipynb** : Evaluates different classifiers for a machine learning task.
- **MainModel_SplitBased_OneTime90.ipynb** : Primary model implementation with data splitting configurations.

References

GitHub - arvindbitm/PhiUSIIL: PhiUSIIL: A diverse security profile empowered phishing URL detection framework based on similarity index and incremental learning. (n.d.). Retrieved December 12, 2024, from https://github.com/arvindbitm/PhiUSIIL

Python Tutorial: Basic feature extraction - YouTube. (n.d.). Retrieved December 12, 2024, from https://www.youtube.com/watch?v=kp7VwkaoJjE