

Configuration Manual

MSc Data Analytics Research Project

Preetham Madeti 22142258

School of Computing National College of Ireland

Supervisor: Abdul Qayum

National College of Ireland



MSc Project Submission Sheet

School of Computing

Student Name:	Preetham Madeti		
Student ID:	22142258		
Programme:	MSc Data Analytics	Year:	2023/2024
Module:	Research Project		
Supervisor:	Mr. Abdul Qayum		
Date:	31/01/2024		
Project Title:	Implementing a Hybrid System for Accurate URLs with Machine Learning and Deep Learn	ely Dete ning Teo	cting Phishing chniques.

Word Count: 735

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.

<u>ALL</u> 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: Preetham Madeti

Date: 29/01/2024

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

Preetham Madeti 22142258

Introduction:

This manual provides detailed instructions for configuring and deploying the phishing URL detection system developed in this research project. The system employs a hybrid model integrating machine learning and deep learning techniques to accurately identify phishing URLs.

1 System Requirements:

To guarantee efficient model processing and to minimize the duration required, it's crucial to be equipped with the necessary hardware and software resources.

1.1. Hardware Requirements:

The implementation is performed on an HP Pavilion; the configuration of the device is as follows.

1.Processor:	Intel(R) Core (TM) i5-9300H CPU @ 2.40GHz
2.RAM:	8.00 GB (7.84 GB usable)
3. Hard Disk:	256GB SSD, 1 TB HDD
4.OS	Windows 10 Pro 64 – bit

1.2 Software Requirements:

Before beginning the model construction phase, the below mentioned software, libraries, and tools were set up and installed on the system.

Software/Tools	Version	Information
Python		To develop the model python is used
		in this project.
Anaconda		Anaconda stands as a highly
		favoured platform for the data
		science community, offering
		capabilities for computational work,
		managing libraries, and deploying
		models, all within a Windows-
		friendly environment
Pandas		It is particularly well-suited for
		dealing with tabular data, such as
		data stored in spreadsheets or
		databases.

NumPy	NumPy, an open-source tool from
	2023, is utilized for tackling intricate
	mathematical issues within data.
Sci-kit Learn	This library is employed for tasks
	like Classification, Regression, and
	data preprocessing. (Scikit-learn:
	Machine Learning in Python —
	scikit-learn 0.24.2 documentation,
	2023).

2. Implementation:

In this section there is a complete guide to run the project in any windows system.

1. Download and Install Anaconda Software in the windows system. (https://www.anaconda.com/products/individual)

	All applications Y ON	base (root)	D			
ironments	•	•	•	•	•	•
ning	DS	O	lab	Jupyter	\mathbf{O}	IPy
	DataSpell	CMD.exe Prompt 0.1.1	JupyterLab 3.4.4	Notebook 6.4.12	Powershell Prompt 0.0.1	Qt Console 522
munity	DetaSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jugyter notebooks with the intelligent	Run a cmd.exe terminal with your current environment from Navigator activated	An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.	Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.	Run a Powershell terminal with your current environment from Navigator activated	PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
	in one user-friendly environment.	Launch	Launch	Launch	Launch	Launch
	0	•	•	•	•	*
	1 No. 1	X		2	Ŭ.	Cloud Infrastructure
	Spyder	VS Code	Datalore	Deepnote	IBM Watson Studio Cloud	Oracle Data Science Service
_	5.2.2 Scientific Prthon Development EnviRonment, Powerful Python IDE with advanced edibing, interactive testing,	1.77.3 Streamlined code editor with support for development operations like debugging, task running and version control.	Kick-start your data science projects in seconds in a pre-configured environment. Enjoy coding assistance for Python, SQL,	Deepnote is a notebook built for collaboration. Create notebooks in your browser, spin up your conda environment	IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train	OCI Data Science offers a machine learning platform to build, train, manage, and deploy your machine learning models on
da Toolbox	debugging and introspection features		and R in Jupyter notebooks and benefit from no-code automations. Use Datalore	in seconds and share with a link.	machine learning models. Prepare data and build models, using open source data	the cloud with your favorite open-source tools
orged tebooks Taalbox	Launch	Launch	Launch	Launch	Launch	Launch
IT THE DOCT	•					
	DC					

2. Open the Jupyter Notebook from Anaconda.

💭 Jupyter		Quit	Logout
Files Running Clusters			
Select items to perform actions on them.		Upload	New 🕶 🖸
□ 0 🔹 🖿 / Desktop	Name 🕹	Last Modified	File size
۵		seconds ago	
C Batch - 17		3 years ago	
CV and Cover Letter		a month ago	
C Data Analytics		2 months ago	
Data Governance and Ethics (H9DGE)		a month ago	
C Documents		22 days ago	
C Final Year Project		6 months ago	
D DJOBS		7 months ago	
Ch My files		8 months ago	
Preetham M - Rajalakshmi Institute Of Technology - B.E		3 years ago	
Copy		5 months ago	

- 3. After opening jupyter notebook click on the new notebook (python 3).
- 4. In notebook, Import all the required libraries.

```
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from urllib.parse import urlparse, parse_qs
from collections import Counter
```

from sklearn.metrics import classification_report as urlphish_hyrder_E1
from sklearn.metrics import confusion_matrix as urlphish_hyrder_E2
from sklearn.metrics import ConfusionMatrixDisplay as urlphish_hyrder_E3
import time as urlphish_hyrder_E4

from sklearn.model_selection import GridSearchCV as urlphish_hyrder_E5

```
from sklearn.metrics import classification_report as urlphish_hyrder_E1
from sklearn.metrics import confusion_matrix as urlphish_hyrder_E2
from sklearn.metrics import ConfusionMatrixDisplay as urlphish_hyrder_E3
import time as urlphish_hyrder_E4
from sklearn.model_selection import GridSearchCV as urlphish_hyrder_E5
from sklearn.ensemble import AdaBoostClassifier as urlphish_hyrder_E6
from sklearn.ensemble import GaussianNB as urlphish_hyrder_E7
from sklearn.naive_bayes import GaussianNB as urlphish_hyrder_E8
from sklearn.neural_network import MLPClassifier as urlphish_hyrder_E10
from sklearn.ensemble import VotingClassifier as urlphish_hyrder_E11
```

5. Import the Provided Dataset.

urlphish_hyrd = urlphish_hyrde.read_csv('phishing_site_urls.csv')

6. Next Step will be Pre Processing Step will be performed using following Code.

```
']: urlphish_hyrd.info()
   <class 'pandas.core.frame.DataFrame'>
RangeIndex: 549346 entries, 0 to 549345
   dtypes: object(2)
   memory usage: 8.4+ MB
   pick null data out
i]: urlphish_hyrd.isna().any()
3]: URL
             False
   Label
             False
   dtype: bool
   pick duplicate data out
]: urlphish_hyrd.duplicated()
·]: 0
              False
              False
    1
    2
              False
   3
              False
   4
              False
              ...
True
   549341
    549342
               True
    549343
               True
    549344
               True
    549345
               True
   Length: 549346, dtype: bool
```

6. Exploratory Data Analysis has been Performed and Visualisation has been done using

following Code





```
urlphish_hyrd['url_length'] = urlphish_hyrd['URL'].apply(len)
plt.hist(urlphish_hyrd['url_length'], bins=50)
plt.title('Histogram of URL Lengths')
plt.xlabel('URL Length')
plt.ylabel('Frequency')
plt.show()
```



7. After Data Pre Processing the Data Splitting is Performed before Building a Model

In [11]: from sklearn.model_selection import train_test_split as urlphish_hyrderios

```
hish_hyrd_R = 99
hish_hyrd_Sa = 0.4
#.....train= 60% .......
inputN_hybrid, inputS_hybrid, outputN_hybrid, outputS_hybrid = urlphish_hyrderios(input_hybrid, output_hybrid, test_size=hish_hyr
hish_hyrd_Sb = 0.5
#.....test= 20% ,validation= 20% .......
inputV_hybrid, inputS_hybrid, outputV_hybrid, outputS_hybrid = urlphish_hyrderios(inputS_hybrid, outputS_hybrid, test_size=hish_hyr
print(inputN_hybrid.shape)
print(inputS_hybrid.shape)
4
```

8. Hybrid Models Implementation has been Performed with the following Code

```
In [16]:
           urlphish_vrt = {'voting': ['soft', 'hard']}
           urlphish_hyrder_m1 = urlphish_hyrder_E7(criterion= 'entropy', n_estimators= 20, n_jobs= 10)
           urlphish_hyrder_m2 = urlphish_hyrder_E10(activation= 'relu', learning_rate= 'constant', solver= 'adam')
           urlphish_vrtr = urlphish_hyrder_E11(estimators=[('randomforest', urlphish_hyrder_m1), ('mlp', urlphish_hyrder_m2)])
           urlphish_vrtr = urlphish_hyrder_E5(urlphish_vrtr, urlphish_vrt, cv=2)
urlphish_vrtr.fit(inputN_hybrid[:1000], outputN_hybrid[:1000])
           print(urlphish_vrtr.best_params_)
           print("score_value : ", urlphish_vrtr.best_score_)
            {'voting': 'soft'}
            score_value : 0.801
In [17]: urlphish_hyrder_a = urlphish_hyrder_E4.time()
hyrder_A = urlphish_hyrder_E11(estimators=[('randomforest', urlphish_hyrder_m1), ('mlp', urlphish_hyrder_m2)], **urlphish_vrtr.be
           hyrder_A.fit(inputN_hybrid, outputN_hybrid)
           urlphish_hyrder_b = urlphish_hyrder_E4.time()
print("\n Training-----time :", urlphish_hyrder_b-urlphish_hyrder_a,"\n")
           urlphish_hyrder_a = urlphish_hyrder_E4.time()
           P_hybrid = hyrder_A.predict(inputV_hybrid)
           print(urlphish_hyrder_E1(outputV_hybrid, P_hybrid))
E2 = urlphish_hyrder_E2(outputV_hybrid, P_hybrid)
E3 = urlphish_hyrder_E3(confusion_matrix = E2, display_labels = [0, 1])
           E3.plot()
           urlphish_hyrder_b = urlphish_hyrder_E4.time()
           print("\n validation-----time :", urlphish_hyrder_b-urlphish_hyrder_a,"\n")
```

9. The Accuracy is considered as evaluation factor after Model Implementation

```
urlphish_hyrder_a = urlphish_hyrder_E4.time()
hyrder_A = urlphish_hyrder_E11(estimators=[('randomforest', urlphish_hyrder_m1), ('mlp', urlphish_hyrder_m2)], **urlphish_vrtr.be
hyrder_A.fit(inputN_hybrid, outputN_hybrid)
urlphish_hyrder_b = urlphish_hyrder_E4.time()
print("\n Training-----time :", urlphish_hyrder_b-urlphish_hyrder_a,"\n")
urlphish_hyrder_a = urlphish_hyrder_E4.time()
P_hybrid = hyrder_A.predict(inputV_hybrid)
print(urlphish_hyrder_E1(outputV_hybrid, P_hybrid))
E2 = urlphish_hyrder_E2(outputV_hybrid, P_hybrid)
E3 = urlphish_hyrder_E3(confusion_matrix = E2, display_labels = [0, 1])
E3.plot()
urlphish_hyrder_b = urlphish_hyrder_E4.time()
print("\n validation------time :", urlphish_hyrder_b-urlphish_hyrder_a,"\n")
```

Training-----time : 768.395833492279

	precision	recall	f1-score	support
0	0.87	0.55	0.67	22821
1	0.88	0.98	0.93	78618
accuracy			0.88	101439
macro avg	0.88	0.76	0.80	101439
weighted avg	0.88	0.88	0.87	101439

prir E2 = E3 = E3.p	nt(urlphis = urlphish = urlphish plot() phish_hyrd	rder_A.predic h_hyrder_E1(o _hyrder_E2(ou _hyrder_E3(co er_b = urlphi	:t(inputS_ butputS_hy utputS_hyl onfusion_r ish_hyrder	_hybrid) ybrid, P_hy brid, P_hyt matrix = E2 r_E4.time()	/brid)) prid) 2, display_la	bels = [0, 1])	
prin	nt("\n tes	ting1	cime : .	uriphish r	ivruer p-urip	hish hyrder a, "\	n"
prir	nt("\n tes	precision	recall	f1-score	support	hish_hyrder_a,"\	n
prir	nt("\n tes	precision 0.87	recall 0.55	f1-score 0.68	support 22845	hish_hyrder_a,"\	n
prir	nt("\n tes 0 1	precision 0.87 0.88	recall 0.55 0.98	0.68 0.93	support 22845 78595	hish_hyrder_a,"\	n
prir	nt("\n tes 0 1 accuracy	precision 0.87 0.88	recall 0.55 0.98	f1-score 0.68 0.93 0.88	support 22845 78595 101440	hish_hyrder_a,`\	n
prir	nt("\n tes 0 1 accuracy macro avg	precision 0.87 0.88 0.88	recall 0.55 0.98 0.76	f1-score 0.68 0.93 0.88 0.80	support 22845 78595 101440 101440	hish_hyrder_a,`\	n

Total Execution Time:

• The overall duration taken to train the dynamic model was 768.395 seconds, while the time taken to test the data using the trained model amounted to 0.575 milliseconds.

The concluding code files are included within three distinct project files, titled 'Phishing URL Data Cleaning.ipynb', 'Implementing ML-DL Models.ipynb', and 'Hybrid Model with ML-DL Models.ipynb', respectively.

References

Anaconda. 2021. Anaconda | The World's Most Popular Data Science Platform. [online] Available at: .<https://www.anaconda.com/>.

Numpy.org. 2021. NumPy. [online] Available at: ">https://numpy.org/>.

Scikit-learn.org. 2021.

scikit-learn machine learning in Python — scikit-learn 0.24.2

documentation. [online] Available at: https://scikit-learn.org/stable/>.