National College of Ireland

Configuration Manual

MSc Academic Internship
MSc Cyber Security

Anjali Pappachan Mulloly
Student ID: 21218897

School of Computing
National College of Ireland

Supervisor: Eugene Mclaughlin

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Anjali Pappachan Mulloly

Student ID: 21218897

Programme: Msc Cyber Security **Year:** 2023-2024

Module: MSc Academic Internship

Lecturer: **Eugene Mclaughlin**

Submission Due Date: **30**/01/2024

Project Title: SECURE DEPLOYMENT OF CLOUD INTEGRATED CYBERSECURITY APPLICATIONS: A COMPREHENSIVE CLOUD SECURITY MODEL

Word Count: 427 **Page Count: 5**

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: Anjali Pappachan Mulloly

Date: 30/01/2024

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

Anjali Pappachan Mulloly
Student ID: 21218897

1 Tools Used:

- Python > 3.1.0
- Juypter notebook > 7.0.6
- Aws- Cloud
- Data set: CMS buffer overflow

2 Libraries used.

- Pandas > 2.0.1
- Numpy > 3.1
- Matplotlib> 4.5.1
- Seaborn> 3.9.4
- Sklearn > 2.8.2
- Boto3 > 3.8

3 Algorithm Used:

Machine Learning

Logistic Regression
Decision Tree
Random Forest
Gradient Boosting
Random Forest and Extra Trees
Ada Boosting
Naïve Bayes
Extra tree
Ensemble Model
Neural Network
KNN
SVM

4 Implementation

Buffer Overflow threat detection

Importing libraries: This is the base lib which has been used.

In [1]:
```
import pandas as pd
import numpy as np
import warnings
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split


from sklearn.feature_selection import SelectKBest, f_classif


from sklearn.metrics import accuracy_score, precision_score, recall_score


from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, AdaBoostClassifier, StackingClassifier, ExtraTr
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier



warnings.filterwarnings("ignore")


warnings.filterwarnings("default", category=DeprecationWarning)
```

Data preprocessing: Data info

In [2]:
```
df = pd.read_csv('Data1.csv',sep=',')

df.head(15)
```

Label encoder

In [5]:
```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
df['vulnerability_name'] = label_encoder.fit_transform(df['vulnerability_name'])
df['required_action'] = label_encoder.fit_transform(df['required_action'])
df['cwe'] = label_encoder.fit_transform(df['cwe'])
df['vector'] = label_encoder.fit_transform(df['vector'])
df['complexity'] = label_encoder.fit_transform(df['complexity'])
df['severity'] = label_encoder.fit_transform(df['severity'])
df['product'] = label_encoder.fit_transform(df['product'])
df['cve_id'] = label_encoder.fit_transform(df['cve_id'])
df['vendor_project'] = label_encoder.fit_transform(df['vendor_project'])
df['date_added'] = label_encoder.fit_transform(df['date_added'])
df['pub_date'] = label_encoder.fit_transform(df['pub_date'])
df['grp'] = label_encoder.fit_transform(df['grp'])
df['short_description'] = label_encoder.fit_transform(df['short_description'])
df['notes'] = label_encoder.fit_transform(df['notes'])
df['cvss'] = label_encoder.fit_transform(df['cvss'])
df['due_date'] = label_encoder.fit_transform(df['due_date'])
```

Shape or Size

In [7]:
```
df.shape
```

Out[7]: (774, 16)

Distribution

In [15]:
```
if df.isnull().values.any():
    df = df.fillna(df.mean())

X = df.drop('severity', axis=1)

y = df['severity']


selector = SelectKBest(f_classif, k=10)

X_new = selector.fit_transform(X, y)


selected_features = X.columns[selector.get_support()]

print("Selected Features:", selected_features)
```

```
Selected Features: Index(['cve_id', 'vendor_project', 'date_added', 'required_action', 'due_date',
       'grp', 'pub_date', 'cvss', 'vector', 'complexity'],
      dtype='object')
```

Validation and training models: Implementing different machine learning models

In [18]:
```
models = {
    'Logistic Regression': LogisticRegression(max_iter=10000),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Gradient Boosting': GradientBoostingClassifier(),
    'AdaBoost': AdaBoostClassifier(),
    'SVM': SVC(probability=True),
    'K-Nearest Neighbors': KNeighborsClassifier(),
    'Gaussian Naive Bayes': GaussianNB(),
    'Neural Network': MLPClassifier(max_iter=10000),
    'Extra Trees': ExtraTreesClassifier()
}
```

Accuracy calculation

Out[20]:

	Model	Accuracy	Precision	Recall
0	Logistic Regression	0.993548	0.987455	0.993548
1	Decision Tree	0.993548	0.987455	0.993548
2	Random Forest	0.993548	0.987455	0.993548
3	Gradient Boosting	0.993548	0.987455	0.993548
4	AdaBoost	0.993548	0.987455	0.993548
7	Gaussian Naive Bayes	0.993548	0.987455	0.993548
9	Extra Trees	0.993548	0.987455	0.993548
10	Ensemble Model	0.993548	0.987455	0.993548
8	Neural Network	0.722581	0.730846	0.722581
6	K-Nearest Neighbors	0.612903	0.616132	0.612903

AWS integration
Install boto3

In [1]:
```
!pip install boto3
```

Import pandas

In [2]:
```
import boto3
import pandas as pd

s3Cloud = boto3.client('s3')
```

Establishing cloud connection

In [3]:
```
s3Cloud = boto3.resource(
    service_name='s3',
    region_name='us-east-1',
    aws_access_key_id='AKIA22TCIOZB5YNUG4T2',
    aws_secret_access_key='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'
)
```

Import s3fs

In [5]:
```
!pip install s3fs
```

Import os

In [6]:
```
import os
os.environ["AWS_DEFAULT_REGION"] = 'us-east-1'
os.environ["AWS_ACCESS_KEY_ID"] = 'AKIA22TCIOZB5YNUG4T2'
os.environ["AWS_SECRET_ACCESS_KEY"] = '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'
```

Importing JSON

In [7]:
```
# Upload files to S3 bucket
s3Cloud.Bucket('anjali-connection').upload_file(Filename='final_results.json', Key='final_results.json')
```

References

GeeksforGeeks (2023) *Machine learning with python tutorial*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/machine-learning-with-python/ (Accessed: 13 December 2023).

Hunt, M. (1994) *Free*, *Amazon*. Available at: https://aws.amazon.com/free/?gclid=Cj0KCQiAyeWrBhDDARIsAGP1mWSaAn1h5s2y5SHvBl2-IS-X292fqdZU4a2Fn1-nFW6MofU_CdGpgMUaAr1zEALw_wcB&trk=2738afd4-9401-4d18-8e3e-1b1c194dea07&sc_channel=ps&ef_id=Cj0KCQiAyeWrBhDDARIsAGP1mWSaAn1h5s2y5SHvBl2-IS-X292fqdZU4a2Fn1-nFW6MofU_CdGpgMUaAr1zEALw_wcB%3AG%3As&s_kwcid=AL%214422%213%21509606977827%21p%21%21g%21%21amazon+web+services%2112618685604%21120373367976&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc&awsf.Free+Tier+Types=%2Aall&awsf.Free+Tier+Categories=%2Aall (Accessed: 13 December 2023).