National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Abhay Singh Bangari
Student ID: X21153507

School of Computing
National College of Ireland

Supervisor: Cristina Hava Muntean

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Abhay Singh Bangari
Student ID:	X21153507
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Cristina Hava Muntean
Submission Due Date:	15/12/2022
Project Title:	Configuration Manual
Word Count:	489
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:	
Date:	1st February 2023

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

Abhay Singh Bangari
X21153507

1 Introduction

Detailed configuration manual gives out essential information regarding the system, hardware, and software specifications. It also shows the entire flow used to implement the research on the "A Comparative Evaluation of Machine Learning Models and EDA through Tableau Using CICIDS2017 Dataset". In section 2, the manual describes a system specification like hardware and software configurations. Next section 3 tells the setting up of the environment, importing of important libraries and pre-processing, In the 4th section model building and evaluation of those models are discussed.

2 System Configuration

System configuration discussed the hardware and software requirements in order to carry out the research.

2.1 Hardware Specifications

Operating System	Windows 10
RAM	16GB
Processor	i7intel
Speed	3.2GHz
Disk Space	3GB approx.
GPU	NVIDIA RTX 2060

2.2 Software Specifications

Programming Language	Python 3.9 version
Other Softwares	Excel, Tableau, & Anaconda
Web Browsers	Google Chrome & Microsoft Edge

3 Environment Setup

3.1 Launching Jupyter on Anaconda

The first step is to launch anaconda application. Anaconda provides you with much useful software to fulfill your needs. Here, Jupyter Notebook was used for implementing the

code, it comes with the latest version of python. Also contains useful python resources for analysis.

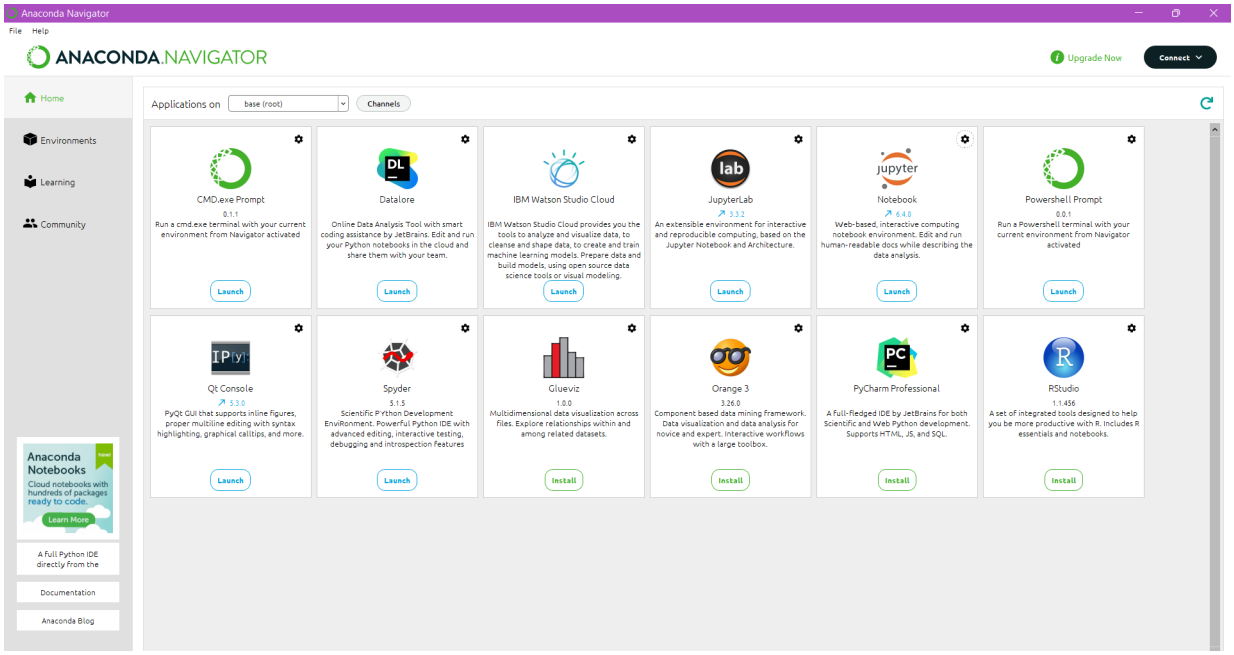


Figure 1: anaconda interface

This is the home page for jupyter notebook, create a new ipynb file to start coding.

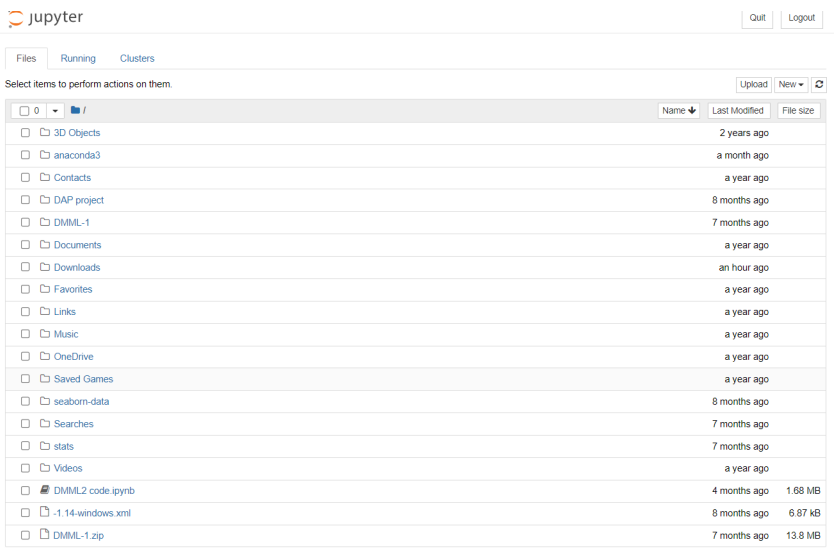


Figure 2: Jupyter

3.2 Data Preparation

Keep all the dataset files in one folder before diving into the code. All the required files are in one place in the folder MachineLearningCVE.

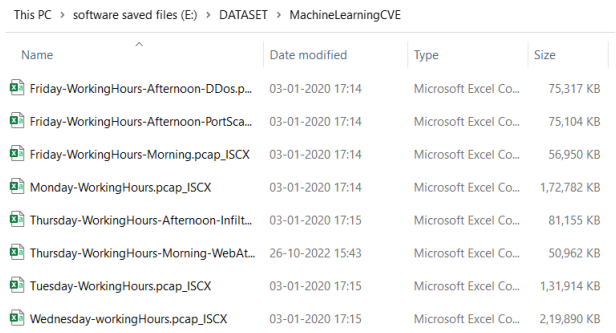


Figure 3: CSV Files

3.3 Importing Important Libraries

It shows all libraries necessary for the implementation and for libraries that require installation can be installed using "pip" . Majorly sklearn library is used throughout the research.

Importing Neccesary Libraries

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.ensemble import AdaBoostClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classification_report
import warnings
from imblearn.over_sampling import SMOTE
from collections import Counter
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
```

Figure 4: Importing Libraries

3.4 Importing Dataset

All the CSVs are imported into the python data frame using the pandas library.

Importing Datasets

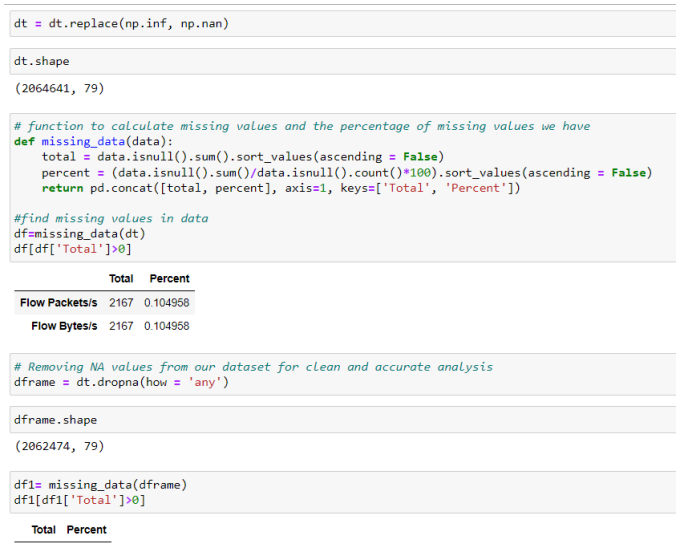
```
data1 = pd.read_csv("E:/DATASET/MachineLearningCVE/Monday-WorkingHours.pcap_ISCX.csv")
data2 = pd.read_csv("E:/DATASET/MachineLearningCVE/Tuesday-WorkingHours.pcap_ISCX.csv")
data3 = pd.read_csv("E:/DATASET/MachineLearningCVE/Wednesday-workingHours.pcap_ISCX.csv")
data4 = pd.read_csv("E:/DATASET/MachineLearningCVE/Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv")
data5 = pd.read_csv("E:/DATASET/MachineLearningCVE/Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv")
data = [data1,data2,data3,data4,data5]
dt = pd.concat (data, axis=0, sort=False, ignore_index=True)
dt

del [data1,data2,data3,data4,data5]
```

Figure 5: Important Dataset

3.5 Data Pre-processing

In this section steps like removing missing values, outliers, etc are performed. Data is converted into structured data which can later be used for modeling.



```
dt = dt.replace(np.inf, np.nan)
```

```
dt.shape
```

```
(2064641, 79)
```

```
# function to calculate missing values and the percentage of missing values we have
def missing_data(data):
    total = data.isnull().sum().sort_values(ascending = False)
    percent = (data.isnull().sum()/data.isnull().count()*100).sort_values(ascending = False)
    return pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])

#find missing values in data
df=missing_data(dt)
df[df['Total']>0]
```

	Total	Percent
Flow Packets/s	2167	0.104958
Flow Bytes/s	2167	0.104958

```
# Removing NA values from our dataset for clean and accurate analysis
dframe = dt.dropna(how = 'any')
```

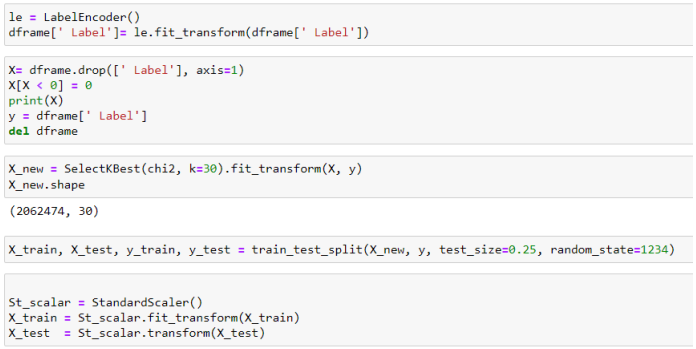
```
dframe.shape
```

```
(2062474, 79)
```

```
df1= missing_data(dframe)
df1[df1['Total']>0]
```

	Total	Percent

Figure 6: Data Pre-processing



```
le = LabelEncoder()
dframe[' Label']= le.fit_transform(dframe[' Label'])
```

```
X= dframe.drop([' Label'], axis=1)
X[X < 0] = 0
print(X)
y = dframe[' Label']
del dframe
```

```
X_new = SelectKBest(chi2, k=30).fit_transform(X, y)
X_new.shape
```

```
(2062474, 30)
```

```
X_train, X_test, y_train, y_test = train_test_split(X_new, y, test_size=0.25, random_state=1234)
```

```
St_scalar = StandardScaler()
X_train = St_scalar.fit_transform(X_train)
X_test  = St_scalar.transform(X_test)
```

Figure 7: More Data Pre-processing Steps

3.6 Data Visualization

This section tells you about all the insights that are drawn from the data. It is performed in tableau. The dashboard generated was uploaded on tableau's public online platform. The link is Given below. `https://public.tableau.com/app/profile/abhay.singh.bangari/viz/BusinessDashboard_16710555971690/Dashboard1?publish=yes`

3.7 Modeling

This section displays how data is fit into different models and trained for prediction analysis. Once the data is ready and properly spilt into training and test data models training is conducted. In this research 6 models have been used and compared against

each other. Later best performing model is further evaluated based on a different number of features. Below are the images of all six machine-learning models.

```
abc = AdaBoostClassifier(n_estimators=50,
                         learning_rate=1)
# Train Adaboost Classifer
model = abc.fit(X_train, y_train)
```

```
#Predict the response for test dataset
y_pred = model.predict(X_test)
```

Figure 8: AdaBoostClassifier

```
model =  XGBClassifier(max_depth=15, learning_rate=1.0, n_estimators=15, seed=34567)
model.fit(X_train, y_train)
```

```
XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
              colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
              early_stopping_rounds=None, enable_categorical=False,
              eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
              importance_type=None, interaction_constraints='',
              learning_rate=1.0, max_bin=256, max_cat_to_onehot=4,
              max_delta_step=0, max_depth=15, max_leaves=0, min_child_weight=1,
              missing=nan, monotone_constraints='()', n_estimators=15, n_jobs=0,
              num_parallel_tree=1, objective='multi:softprob', predictor='auto',
              random_state=34567, reg_alpha=0, ...)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
y_pred = model.predict(X_test)
predictions = [round(value) for value in y_pred]
```

Figure 9: XGBoost Classifier

```
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(X_train, y_train)
```

```
KNeighborsClassifier(n_neighbors=3)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
predsknn=neigh.predict(X_test)
```

Figure 10: KNeighborClassifier

```
RFC= RandomForestClassifier()
RFC.fit(X_train, y_train)
```

```
RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
preRFC = RFC.predict(X_test)
```

Figure 11: Random Forest Classifier

```
DTC= DecisionTreeClassifier()
DTC.fit(X_train, y_train)
```

```
DecisionTreeClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
preDTC = DTC.predict(X_test)
```

Figure 12: Decision Tree

```
LDA = LinearDiscriminantAnalysis()
```

```
LDA.fit(X_train, y_train)
```

```
LinearDiscriminantAnalysis()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

55preLDA = LDA.predict(X_test)

Figure 13: Linear Discriminant Analysis