National College of Ireland

Configuration Manual

MSc Research Project
Cyber Security

Bhavana Bhavya
Student ID: 20128126

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Bhavana Bhavya		
Student ID:	20128126		
Programme:	Cyber Security	**Year:**	2021-2022
Module:	MSc Research Project		
Lecturer:	Vikas Sahni		
Submission Due Date:	07/01/22		
Project Title:	Using IP Address as a Unique Attribute for Data Deduplication in Network Devices		
Word Count:	954	**Page Count:**	12

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: Bhavana Bhavya

Date: 07/01/22

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

Bhavana Bhavya
x20128126@student.ncirl.ie

1 Equipment Used

Model	Requirement
Processors Core	Intel Core i5
RAM Memory	8.00 GB
System Type	64-bit Operating System
Hard Disk Storage	1 TB
Network Used	Wifi
Language Used	Python3
IDE Used	Jupyter Notebook

2 Installation required

Tools	Versions
Python	3.9
Anaconda Navigator	3-2021.11-Windows-x86_64
Jupyter Notebook	6.4.5
Numpy Python Library	1.20.3
Pandas Python Library	1.3.4
RecordLinkage Python Library	0.14
Sklearn Python Library	0.24.2
Matplotlib Python Library	3.4.3
Glob Python Library	0.7

3 Logistic Regression Code

Importing required libraries:

```
# installing additional libraries
!pip install recordlinkage pandas numpy
```

```
# importing required libraries
import pandas as pd
import numpy as np
from recordlinkage import Compare
from recordlinkage.index import Block
from sklearn.model_selection import train_test_split
import os
import glob
import itertools
import seaborn as sns
import math
from pylab import *
from scipy.sparse import csr_matrix

from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
from matplotlib import style
import warnings
warnings.filterwarnings("ignore")

style.use('fivethirtyeight')
```

Loading dataset:

```
# loading dataset
dataset = pd.read_excel("./dataset/test_cmdb.xlsx")
```

Displaying top 5 rows:

```
# displaying top-5 rows
dataset.head()
```

	FQDN	IP_Address	Asset_State	Device_Subtype	Device_Discovery_Source
0	pc1apsconsole	10.58.90.28	Installed	NaN	ServiceNow
1	ps3apsdev	10.58.218.68	Installed	IP Firewall	Nlyte
2	pc1apsconsole	10.58.90.28	Installed	IP Switch	NaN
3	pc1aed	10.58.90.29	Installed	NaN	NaN
4	ps3aed	10.58.218.67	Installed	NaN	NaN

Listing rows and columns:

```
# shape of the datasets
dataset.shape
# discpritive analysis
dataset.describe()
```

	FQDN	IP_Address	Asset_State	Device_Subtype	Device_Discovery_Source
count	2693	2666	2693	701	2336
unique	2619	2502	3	10	15
top	pc1apsconsole	0.0.0.1	Installed	IP Firewall	Solarwinds
freq	2	24	2679	388	997

Checking Null values:

```
# preprocessing dataset
print("Checking for null values")
dataset.isnull().sum()
```

Filling 'NA' for Null values:

```
# filling null values
dataset["Device_Discovery_Source"].fillna(method='ffill', inplace = True)
dataset["Device_Subtype"].fillna(method='ffill', inplace = True)
```

Dropping rest of the rows:

```
dataset.dropna(subset=["IP_Address"],axis=0, inplace=True)
```

Checking Null values again:

```
dataset.isnull().sum()
```

Checking rows and columns again:

```
# checking dataset shape
dataset.shape
```

Converting dataset into upper case:

```
dataset = dataset.astype(str).apply(lambda x: x.str.upper())
```

Choosing IP Address as Index:

```
index = Block(on="IP_Address")
ipAddressIndex = index.index(dataset)
```

Printing number of duplicate pairs:

```
print("Table Records: {} records, No of Pairs: {} pairs".format(dataset.shape[0], len(ipAddressIndex)))
```

Keeping first row and dropping rest:

```
# dropping duplicates
ipAddressIndexPairs = ipAddressIndex.drop_duplicates(keep="first")
```

Displaying duplicate pairs:

```
ipAddressIndex
```

```
MultiIndex([(   2,    0),
            (  35,   33),
            ( 101,   48),
            (  84,   79),
            ( 120,  118),
            ( 378,  119),
            ( 305,  181),
            ( 368,  194),
            ( 322,  204),
            ( 371,  210),
            ...
```

```
dataset.columns
```

Comparing unique values using Jaro-winkler method:

```
compare = Compare()
compare.string('FQDN','FQDN', method='jarowinkler', label = 'FQDN_score')
compare.string('IP_Address','IP_Address', method='jarowinkler', label = 'IP_Address_score')
compare.string('Asset_State','Asset_State', method='jarowinkler', label = 'Asset_State_score')
compare.string('Device_Subtype','Device_Subtype', method='jarowinkler', label = 'Device_Subtype_score')
compare.string('Device_Discovery_Source','Device_Discovery_Source', method='jarowinkler', label = 'Device_Discovery_Source_score
comparison_vectors = compare.compute(ipAddressIndex,dataset)
```

Displaying top 5 rows:

```
comparison_vectors.head(5)
```

		FQDN_score	IP_Address_score	Asset_State_score	Device_Subtype_score	Device_Discovery_Source_score
2	**0**	1.000000	1.0	1.0	0.0	0.433333
35	**33**	0.975758	1.0	1.0	1.0	0.400000
101	**48**	0.973333	1.0	1.0	1.0	1.000000
84	**79**	0.973333	1.0	1.0	1.0	1.000000
120	**118**	1.000000	1.0	1.0	1.0	0.465079

Displaying description:

```
# describing similarity vector
comparison_vectors.describe()
```

	FQDN_score	IP_Address_score	Asset_State_score	Device_Subtype_score	Device_Discovery_Source_score
count	422.000000	422.0	422.000000	422.000000	422.000000
mean	0.676575	1.0	0.817228	0.867226	0.686064
std	0.255767	0.0	0.291993	0.222558	0.269264
min	0.000000	1.0	0.351852	0.000000	0.000000
25%	0.506586	1.0	0.351852	0.603367	0.447619
50%	0.594617	1.0	1.000000	1.000000	0.465079
75%	0.966667	1.0	1.000000	1.000000	1.000000
max	1.000000	1.0	1.000000	1.000000	1.000000

Creating Centroid function for seperating duplicate and unique values:

```
datasetPairs = comparison_vectors.reset_index()
centroids = {}
K = 2
for i in range(K):
    centroids[i] = datasetPairs.iloc[i,:].values
```

```
itr = 0
cnt = 0
centroids = {k:v[:2]for k, v in centroids.items()}
prevCentroids = dict(centroids).copy()
while itr < 300:
    predicted_labels = {i:[] for i in range(K)}

    for i, data in datasetPairs.iterrows():
        data = data.values
        distances = [round(np.sqrt(sum((a - b)**2 for a, b in zip(data, centroids[centroid]))), 2) for centroid in centroids]
        label = distances.index(min(distances))
        predicted_labels[label].append(data)

    for label in predicted_labels:
        centroids[label] = np.average(predicted_labels[label],axis=0)
    itr += 1
```

Considering input-output for training and testing data:

```
# input and output
x = []
y = []
for label, values in predicted_labels.items():
    for val in values:
        x.append(val)
        y.append(label)
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2,random_state=500)
len(x_train), len(x_test)
```

Applying Logistic Regression Algorithm:

```
# function to encode the output variable as indicator variable
def encoderFunction(Y):
    row = Y.shape[0]  # calculating number of rows
    temp = csr_matrix((np.ones(row), (Y, np.array(range(row)))))
    encodedY = np.array(temp.todense()).T  # encoding the values
    return encodedY


# function to calculate the softmax value
def softmaxFunction(z):
    z -= np.max(z)
    out = (np.exp(z).T / np.sum(np.exp(z), axis=1)).T
    return out


def lossFunction(w, x, y):
    row = x.shape[0]  # calculating the number of rows in the input
    yHat = encoderFunction(y)  # encoding the output value
    prob = softmaxFunction(np.dot(x, w))  # calculating the softmax
    cost = 1 / row * np.sum(-yHat * np.log(prob) - (1 - yHat) * np.log(1 - prob)) # loss calculation
    gradient = (1 / row) * np.dot(x.T, (yHat - prob))   # Performing gradient descent
    return cost, gradient
```

```
class LogisticRegression:

    def __init__(self, data, labels=None, numClasses=None):
        self.data = data
        self.labels = labels

    def train(self, eta, epoch):
        # initialize a random weight matrix whose size is the image dimension * num of classes
        weights = np.zeros([self.data.shape[1], len(np.unique(self.labels))])
        losses = []  # initializing the list for loss value

        for i in range(0, epoch):
            if i % 100 == 0:
                print("Completed : {} of {} epochs".format(i, epoch))
            loss, grad = lossFunction(weights, self.data, self.labels)
            losses.append(np.nan_to_num(loss))
            weights = weights + (eta * grad)
        print("Average loss value is {}".format(np.mean(losses)))
        return weights

    def test(self, Wt):
        probs = softmaxFunction(np.dot(self.data, Wt))
        preds = np.argmax(probs, axis=1)
        probality = [max(prob) for prob in probs]
        return preds, probality
```

```
ETA = 0.000001
EPOCH = 500
```

```
LR = LogisticRegression(data=np.array(x_train), labels=np.array(y_train))
Wt = LR.train(ETA, EPOCH)
```

```
LR = LogisticRegression(data=np.array(x_test), numClasses=shape(Wt)[1])
predicted_labels, probs = LR.test(Wt)

TP = 3
for i in range(len(y_test)):
    if y_test[i] == predicted_labels[i]:
        TP += 1

acc = round(100 * TP / float(len(predicted_labels)))
print('Recognition Rate = %0.1f' % acc)
```

Calculating Accuracy Percentage:

```
print(classification_report(predicted_labels, y_test))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         0
           1       1.00      0.95      0.98        85

    accuracy                           0.95        85
   macro avg       0.50      0.48      0.49        85
weighted avg       1.00      0.95      0.98        85
```

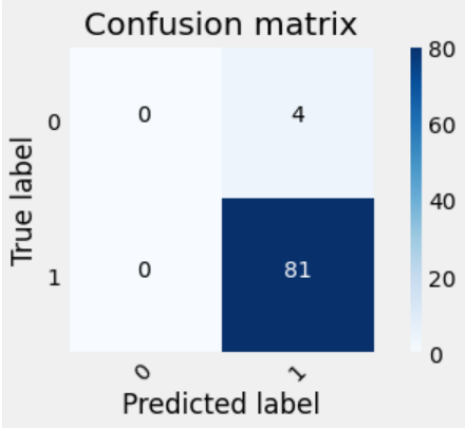
Generating Confusion-Matrix:

```
def plot_confusion_matrix(cm,
                          classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.grid(False)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(list(range(cm.shape[0])),
                                  list(range(cm.shape[1]))):
        plt.text(j,
                 i,
                 format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

```
cnf_matrix = confusion_matrix(y_test, predicted_labels)
np.set_printoptions(precision=2)
plt.figure()
plot_confusion_matrix(
    cnf_matrix,
    classes=[0,1])
```



4 Internship Activity

Student Name:	**Bhavana Bhavya**	Student number:	**20128126**
Company:	Dell Technologies	Month Commencing:	September 2021

Meeting with Senior Manager and brief introduction of project Brief introduction with the team and introductory meeting with assigned mentor	Attended IT and HR Induction and introduced to required licensed and unlicensed tools of asset inventory management Small chunks of tasks assigned to get acquainted with tools	Worked on Asset Inventory audits to ensure uniformity of asset related details across Solarwinds, Kenna, CMDB and Asset Registry tools	Updating tracker sheet and slides with updated asset device details for tracking improvements in uniformity of applications across four security tools (Solarwinds, CMDB, KENNA and Asset Registry)

Month Commencing:
October 2021

Spunk Fundamentals and Splunk Infrastructure Certificate Programme completion 'Code of Conduct' course completion	Training on Data Protection, Phishing, Ransomware, Incident Reporting during Security Awareness Month Program	Completed Policies, Standards and Best Practices for Secure Workplace course which comes under Dell's Global Ethics & Compliance Training Program	Updating tracker sheet and slides with updated asset device details for tracking improvements in uniformity of applications across all four security tools (Solarwinds, CMDB, KENNA and Asset

		‘Be The Change Essentials’ training completion	Registry)

Month Commencing:
November 2021

Palo Alto Networks Session: Keynote speaker Nikesh Arora (Palo Alto Networks CEO) and Jen Easterly (Director of Cybersecurity and Infrastructure Security Agency) on Rethinking Cybersecurity. Talk on Zero Trust Enterprise policy followed by present organizations by Nir Zuk (Palo Alto CTO) Latest in Cyber security Innovation talk by Lee Klarich (Palo Alto Networks Chief Product Officer)	Palo Alto Networks Session: Chat on diversity in storytelling and courage to step-up and respond to next opportunity by Lena Waithe (Producer/ Actor) and Liane Hornsey (Chief People Officer) Transformation journeys by Security leaders from KPMG, State of North Dakota, Sanofi and Investec	Palo Alto Networks Session: Discussion over threat landscape, its evolution and lessons learnt by Wendi Whitmore (Palo Alto Networks Senior Vice President) Discussion over future of mobility and race against climate change by Sylvain Filippi (Managing Director Envision Racing Formula E) and Zeynep Ozdemir (CMO Palo Alto Networks)	Updating tracker sheet and slides with updated asset device details for tracking improvements in uniformity of applications across four security tools (Solarwinds, CMDB, KENNA and Asset Registry) Follow-up and confirmation of legal approvals for using Asset Inventory data for research question implementation

Month Commencing:
December 2021

Background study, knowing basic architecture of data flow to select major tools for data extraction Data extraction from CMDB and Solarwinds UDDR data extraction with the help of team-mates	Data normalization and anonymisation for enhanced data security Character shuffling and character substitution method to anonymise dataset	Exploring Splunk and practicing exercises such as data search, dashboard creation, listing table, creating charts	Updating tracker sheet and slides with updated asset device details for tracking improvements in uniformity of applications across four security tools (Solarwinds, CMDB, KENNA and Asset Registry)

5 Internship Feedback

Employer comments

> Bhavana, integrated well with the team during her internship.

Student Signature: **Bhavana Bhavya** Date: 15/12/21
Industry Supervisor Signature: **Catherine Minogue** Date: 20/12/21