Configuration Manual

MSc Research Project
Cyber Security

Prabhjeet Singh Multani
Student ID: x20153449

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	……Prabhjeet Singh Multani……………………………………………………………
Student ID:	……x20153449………………………………………………………………………………
Programme:	… MSc in Cyber Security ……….. **Year:** …2021-2022..
Module:	…MSc Internship ……………………………………………………………
Lecturer:	……Vikas Sahni ………………………………………………………………
Submission Due Date:	…07/01/2022………………………………………………………………………………
Project Title:	Intrusion Detection System for Industrial Control Systems using Classification Techniques
Word Count:	……733………….. **Page Count:** ………13………………………………

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: …Prabhjeet Singh Multani…………………………

Date: ……07/01/2022…………………………………………………………

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

Prabhjeet Singh Multani
Student ID: x20153449

1 Introduction

The configuration manual is detail about the hardware and software used in the research project. In this internship research project where Machine learning models were developed, implementation section will guide through the process carried out in the development phase along with the final results of the research.

2 System Configuration

In this research, the system used was personal during performing the internship as the internship was hybrid. The configuration of the system is as follows:

2.1 Hardware Configuration

- Operating system: Windows 10
- System Compatibility: 64-bit
- CPU: 11th Gen Intel(R) Core (TM) i5-1135G7 @ 2.40GHz 2.42 GHz
- RAM: 12.0 GB
- Storage: 320 Gb
- GPU: Intel® Iris® Xe Graphics

2.2 Software Configurations:

Prior to start the model building phase following software, tools and libraries were installed in the system.

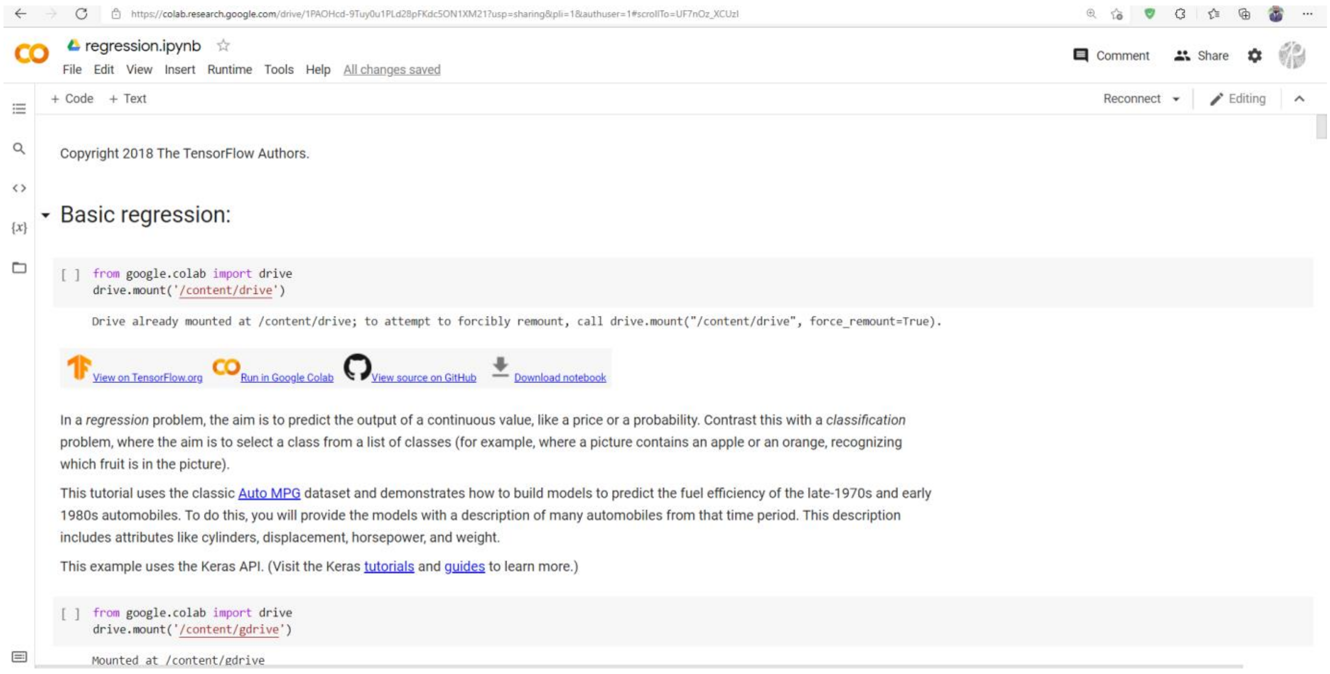
Programming Language Tools	Google Colaboratory (Cloud-based Jupyter notebook environment), Python
Email Account	Gmail account for Google Drive
Web Browser	Microsoft Edge or Google Chrome
Other software	Microsoft Word

3 Project Development

In this section, the environment setup and data collection details will be provided

3.1 Google Colaboratory Environment Setup

The google colab is used for this research implementation where python language has been used with powerful GPU and RAM. Google colaboratory is a free Jupyter notebook that runs in cloud (Google Colab - What is Google Colab?, 2022).



Copyright 2018 The TensorFlow Authors.

Basic regression:

```
[ ] from google.colab import drive
    drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

In a *regression* problem, the aim is to predict the output of a continuous value, like a price or a probability. Contrast this with a *classification* problem, where the aim is to select a class from a list of classes (for example, where a picture contains an apple or an orange, recognizing which fruit is in the picture).

This tutorial uses the classic Auto MPG dataset and demonstrates how to build models to predict the fuel efficiency of the late-1970s and early 1980s automobiles. To do this, you will provide the models with a description of many automobiles from that time period. This description includes attributes like cylinders, displacement, horsepower, and weight.

This example uses the Keras API. (Visit the Keras tutorials and guides to learn more.)

```
[ ] from google.colab import drive
    drive.mount('/content/gdrive')

    Mounted at /content/gdrive
```

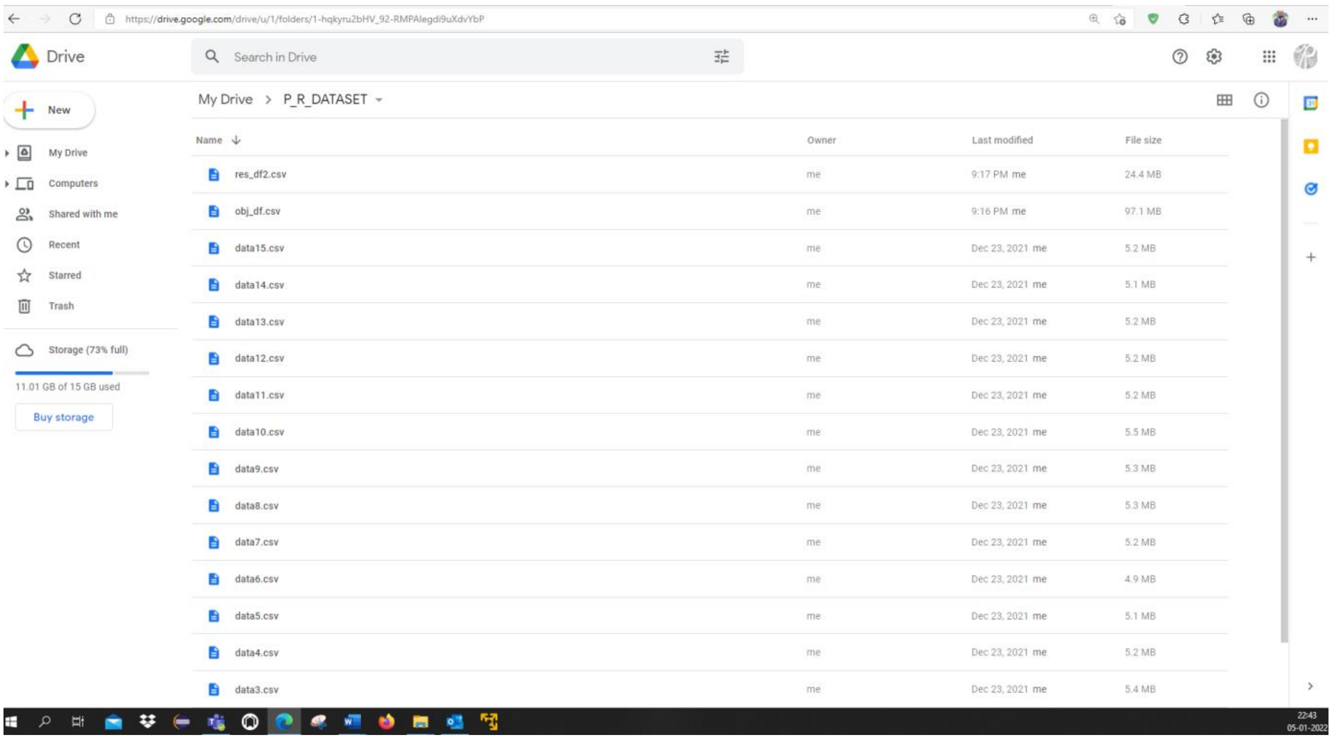
Figure 1 Google Colaboratory

3.2 Collection of Data

Dataset is obtained from the Mississippi university platform. It is publicly available. 29 parameters are used for trained and tested the data and after that implemented on Classification techniques.

3.3 Data Preparation

After getting the data from public platform, then uploaded the whole folder in Google Drive and named as P_R_DATASET.



My Drive > P_R_DATASET

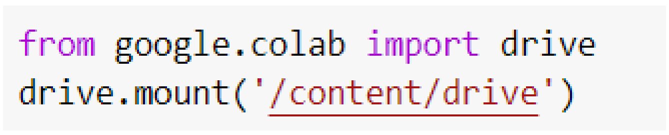
Name	Owner	Last modified	File size
res_df2.csv	me	9:17 PM me	24.4 MB
obj_df.csv	me	9:16 PM me	97.1 MB
data15.csv	me	Dec 23, 2021 me	5.2 MB
data14.csv	me	Dec 23, 2021 me	5.1 MB
data13.csv	me	Dec 23, 2021 me	5.2 MB
data12.csv	me	Dec 23, 2021 me	5.2 MB
data11.csv	me	Dec 23, 2021 me	5.2 MB
data10.csv	me	Dec 23, 2021 me	5.5 MB
data9.csv	me	Dec 23, 2021 me	5.3 MB
data8.csv	me	Dec 23, 2021 me	5.3 MB
data7.csv	me	Dec 23, 2021 me	5.2 MB
data6.csv	me	Dec 23, 2021 me	4.9 MB
data5.csv	me	Dec 23, 2021 me	5.1 MB
data4.csv	me	Dec 23, 2021 me	5.2 MB
data3.csv	me	Dec 23, 2021 me	5.4 MB

Figure 2 Google Drive Data

4 Implementation

Steps of implementation is mentioned below

4.1 Mounting Google Drive: To access the data, Google Drive must be mounted in Google Colab. It necessitates authentication using the Gmail account used for Colab.

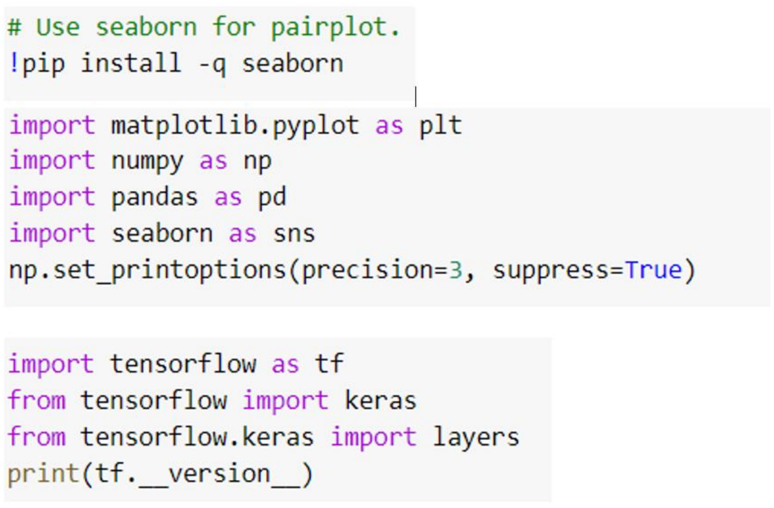


```
from google.colab import drive
drive.mount('/content/drive')
```

Figure 3 Mounting Google Drive on Colab Notebook

4.2 Libraries

In this part, the libraries required for implementation of this project are imported. Those libraries which are unavailable, use pip command can be installed.

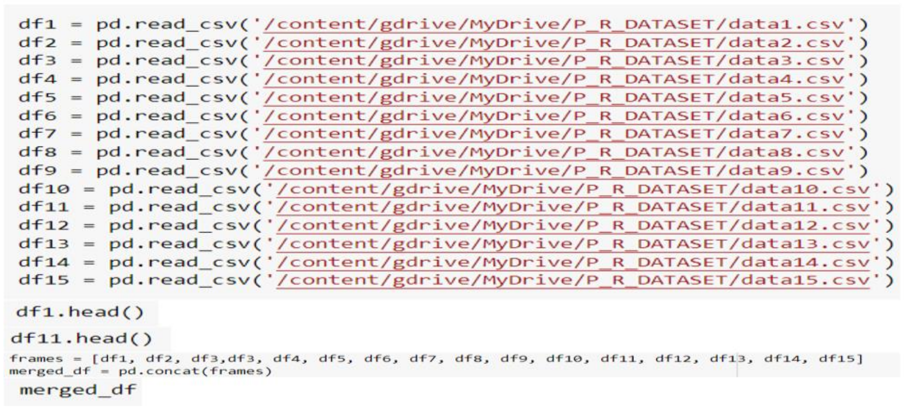


```
# Use seaborn for pairplot.
!pip install -q seaborn

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
np.set_printoptions(precision=3, suppress=True)

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
print(tf.__version__)
```

Figure 4 Imported Libraries

4.2 Dataset fetched in Google Colab:

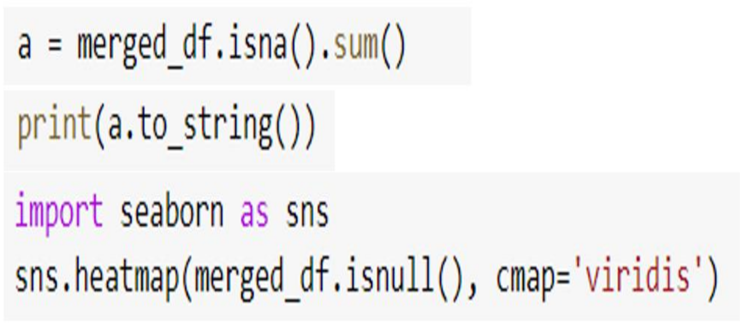
After mounted the dataset in Google colaboratory, then dataset got fetch using below commands and



```
df1 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data1.csv')
df2 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data2.csv')
df3 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data3.csv')
df4 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data4.csv')
df5 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data5.csv')
df6 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data6.csv')
df7 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data7.csv')
df8 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data8.csv')
df9 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data9.csv')
df10 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data10.csv')
df11 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data11.csv')
df12 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data12.csv')
df13 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data13.csv')
df14 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data14.csv')
df15 = pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/data15.csv')

df1.head()

df11.head()

frames = [df1, df2, df3,df3, df4, df5, df6, df7, df8, df9, df10, df11, df12, df13, df14, df15]
merged_df = pd.concat(frames)

merged_df
```

Figure 5 Dataset fetched from Google Drive

4.4 Data Cleaning: all the null and nan values will be removed

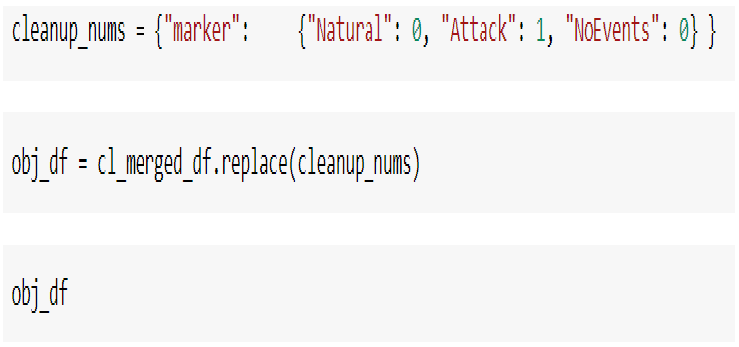


```
a = merged_df.isna().sum()

print(a.to_string())

import seaborn as sns
sns.heatmap(merged_df.isnull(), cmap='viridis')
```

Figure 6 Data Cleaning

4.5 Provide string values to dependent variables

Provide the string value to dependent variables where natural assigned to 0 and 1 allocated to attack.



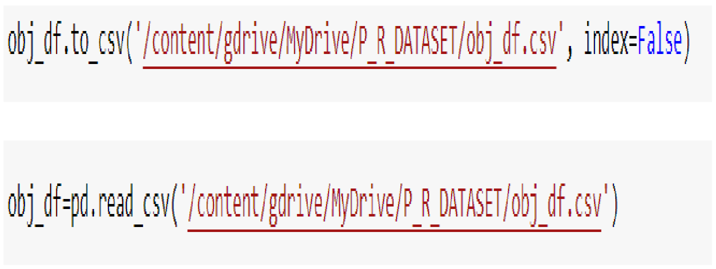
```
cleanup_nums = {"marker":     {"Natural": 0, "Attack": 1, "NoEvents": 0} }
```

```
obj_df = cl_merged_df.replace(cleanup_nums)
```

```
obj_df
```

Figure 7 Provided string values

4.6 Saving data frame to CSV file

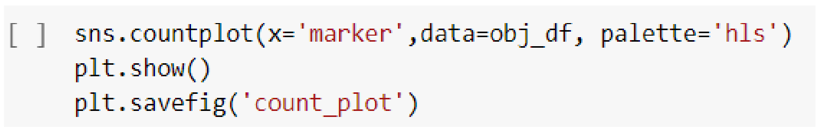


```
obj_df.to_csv('/content/gdrive/MyDrive/P_R_DATASET/obj_df.csv', index=False)
```

```
obj_df=pd.read_csv('/content/gdrive/MyDrive/P_R_DATASET/obj_df.csv')
```

Figure 8 Data frame to CSV

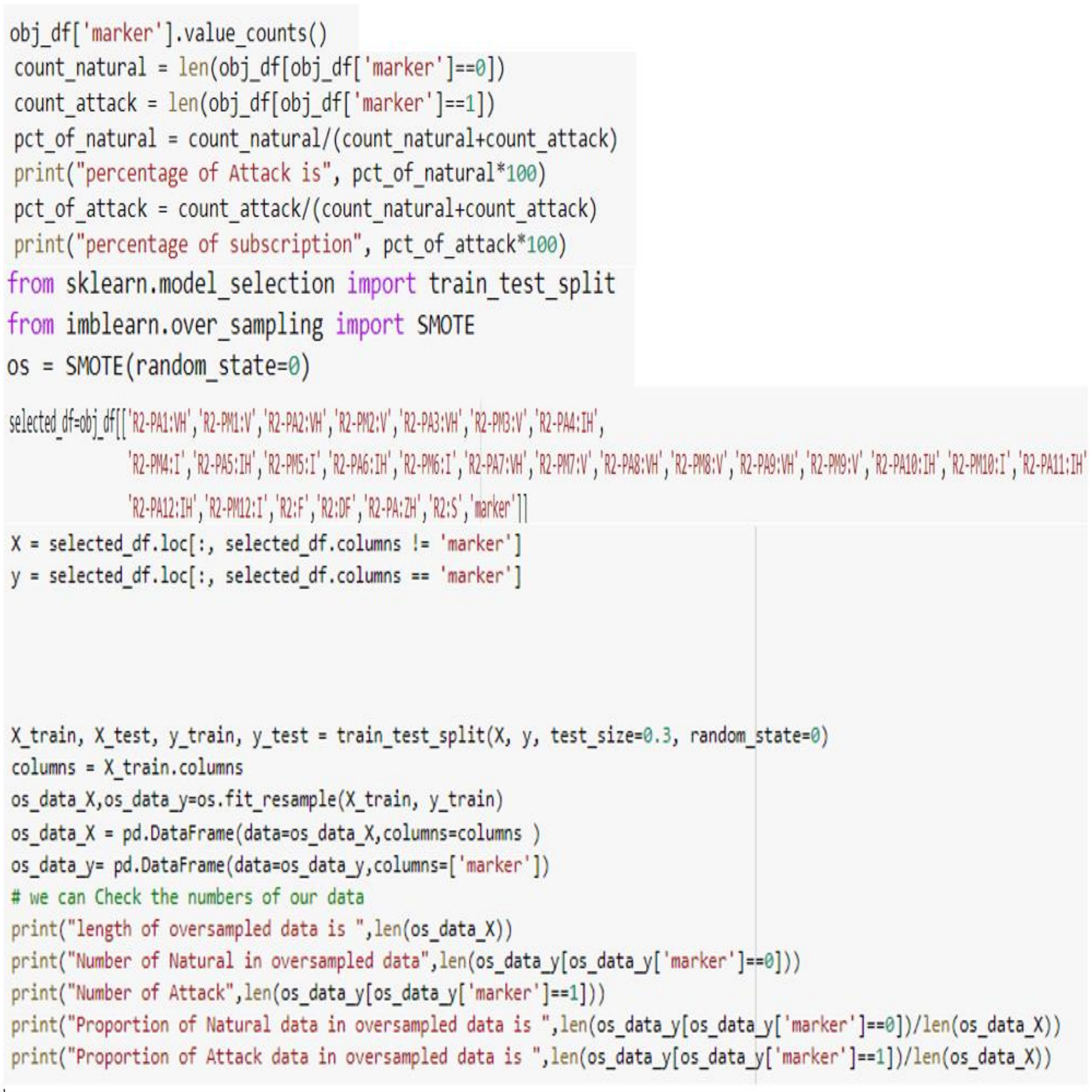
4.7 Check for Class imbalance



```
[ ]  sns.countplot(x='marker',data=obj_df, palette='hls')
     plt.show()
     plt.savefig('count_plot')
```

Figure 9 Class imbalance

4.8 Data over sampling using SMOTE



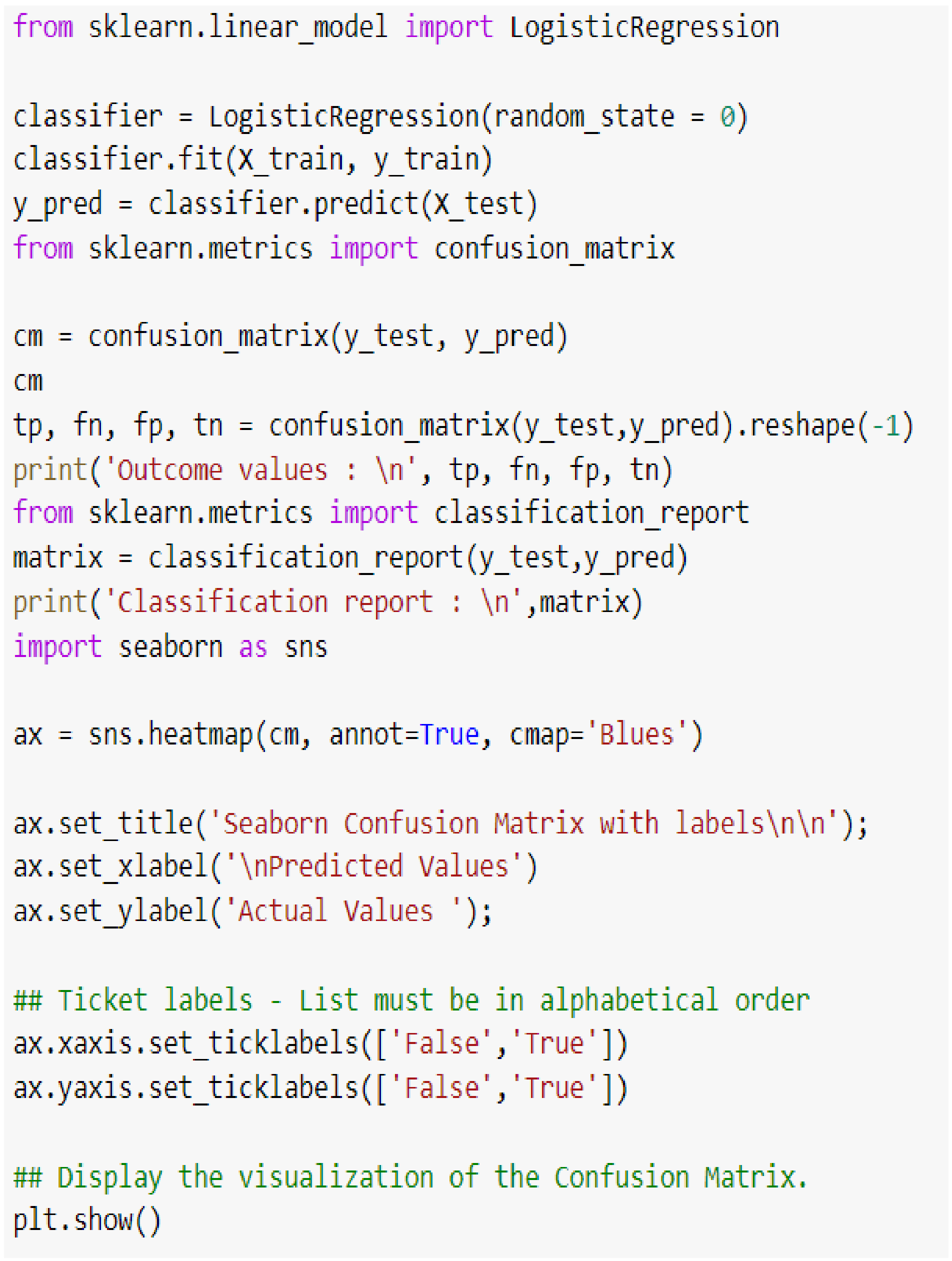
```
obj_df['marker'].value_counts()
 count_natural = len(obj_df[obj_df['marker']==0])
 count_attack = len(obj_df[obj_df['marker']==1])
 pct_of_natural = count_natural/(count_natural+count_attack)
 print("percentage of Attack is", pct_of_natural*100)
 pct_of_attack = count_attack/(count_natural+count_attack)
 print("percentage of subscription", pct_of_attack*100)
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE
os = SMOTE(random_state=0)
selected_df=obj_df[['R2-PA1:VH','R2-PM1:V','R2-PA2:VH','R2-PM2:V','R2-PA3:VH','R2-PM3:V','R2-PA4:IH',
              'R2-PM4:I','R2-PA5:IH','R2-PM5:I','R2-PA6:IH','R2-PM6:I','R2-PA7:VH','R2-PM7:V','R2-PA8:VH','R2-PM8:V','R2-PA9:VH','R2-PM9:V','R2-PA10:IH','R2-PM10:I','R2-PA11:IH'
              'R2-PA12:IH','R2-PM12:I','R2:F','R2:DF','R2-PA:ZH','R2:S','marker']]
X = selected_df.loc[:, selected_df.columns != 'marker']
y = selected_df.loc[:, selected_df.columns == 'marker']




X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
columns = X_train.columns
os_data_X,os_data_y=os.fit_resample(X_train, y_train)
os_data_X = pd.DataFrame(data=os_data_X,columns=columns )
os_data_y= pd.DataFrame(data=os_data_y,columns=['marker'])
# we can Check the numbers of our data
print("length of oversampled data is ",len(os_data_X))
print("Number of Natural in oversampled data",len(os_data_y[os_data_y['marker']==0]))
print("Number of Attack",len(os_data_y[os_data_y['marker']==1]))
print("Proportion of Natural data in oversampled data is ",len(os_data_y[os_data_y['marker']==0])/len(os_data_X))
print("Proportion of Attack data in oversampled data is ",len(os_data_y[os_data_y['marker']==1])/len(os_data_X))
```

Figure 10 Over sampling using SMOTE

5 Models and outputs

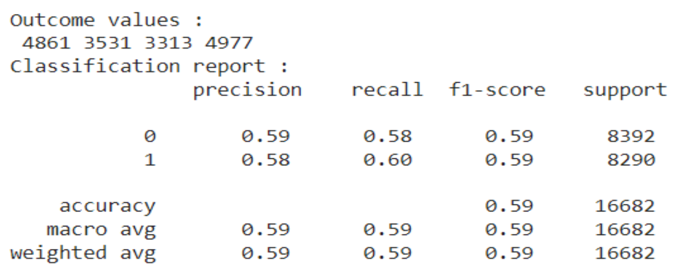
5.1 Logistic Regression

Logistic Regression is first model in the classification technique, where 29 parameters have selected for the implementation. Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable.



```python
from sklearn.linear_model import LogisticRegression

classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred)
cm
tp, fn, fp, tn = confusion_matrix(y_test,y_pred).reshape(-1)
print('Outcome values : \n', tp, fn, fp, tn)
from sklearn.metrics import classification_report
matrix = classification_report(y_test,y_pred)
print('Classification report : \n',matrix)
import seaborn as sns

ax = sns.heatmap(cm, annot=True, cmap='Blues')

ax.set_title('Seaborn Confusion Matrix with labels\n\n');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');

## Ticket labels - List must be in alphabetical order
ax.xaxis.set_ticklabels(['False','True'])
ax.yaxis.set_ticklabels(['False','True'])

## Display the visualization of the Confusion Matrix.
plt.show()
```

Figure 11 Logistic Regression



```
Outcome values :
 4861 3531 3313 4977
Classification report :
              precision    recall  f1-score   support

           0       0.59      0.58      0.59      8392
           1       0.58      0.60      0.59      8290

    accuracy                           0.59     16682
   macro avg       0.59      0.59      0.59     16682
weighted avg       0.59      0.59      0.59     16682
```

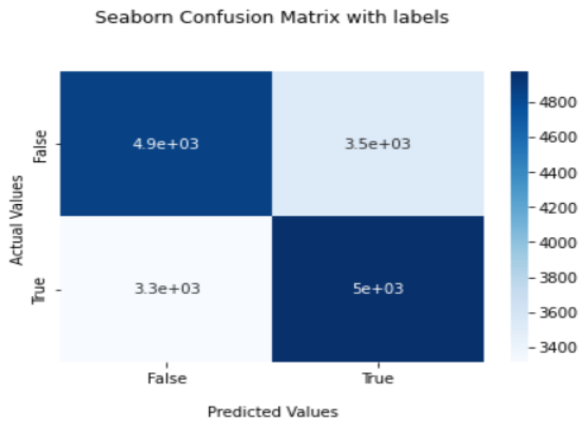
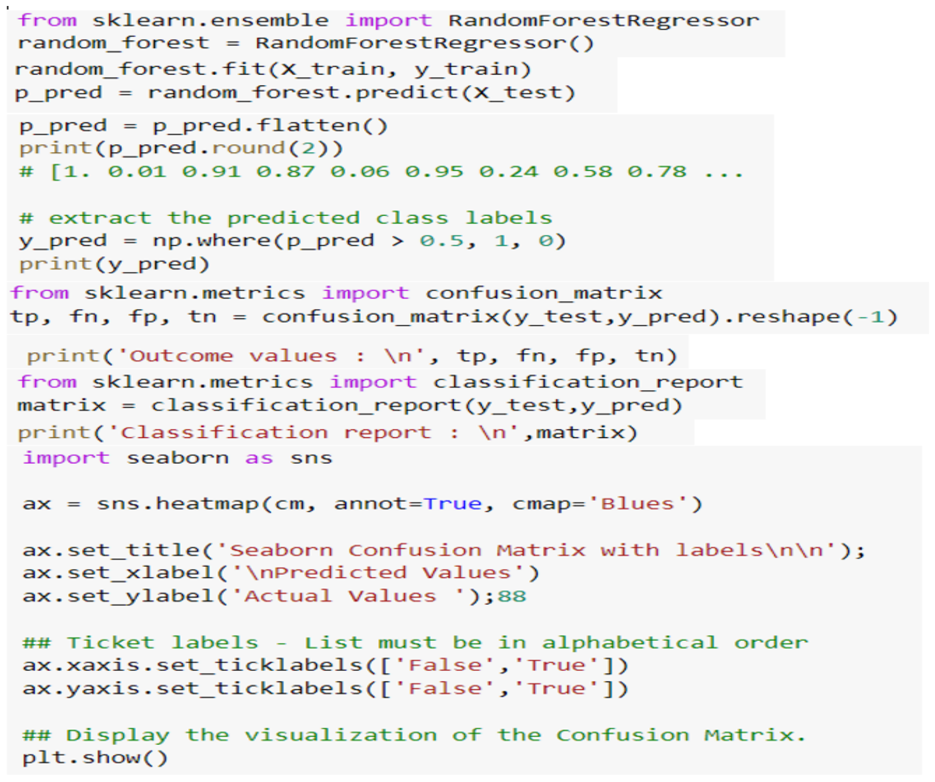


Figure 12 Result and Confusion Matrix

5.2 Random Forest.

Random forest is the second model for our implementation.



```
from sklearn.ensemble import RandomForestRegressor
random_forest = RandomForestRegressor()
random_forest.fit(X_train, y_train)
p_pred = random_forest.predict(X_test)
p_pred = p_pred.flatten()
print(p_pred.round(2))
# [1. 0.01 0.91 0.87 0.06 0.95 0.24 0.58 0.78 ...

# extract the predicted class labels
y_pred = np.where(p_pred > 0.5, 1, 0)
print(y_pred)
from sklearn.metrics import confusion_matrix
tp, fn, fp, tn = confusion_matrix(y_test,y_pred).reshape(-1)
print('Outcome values : \n', tp, fn, fp, tn)
from sklearn.metrics import classification_report
matrix = classification_report(y_test,y_pred)
print('Classification report : \n',matrix)
import seaborn as sns

ax = sns.heatmap(cm, annot=True, cmap='Blues')

ax.set_title('Seaborn Confusion Matrix with labels\n\n');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');88

## Ticket labels - List must be in alphabetical order
ax.xaxis.set_ticklabels(['False','True'])
ax.yaxis.set_ticklabels(['False','True'])

## Display the visualization of the Confusion Matrix.
plt.show()
```

Figure 13 Random Forest

```
Classification report :
               precision    recall  f1-score   support

           0       0.92      0.89      0.90      8392
           1       0.89      0.92      0.91      8290

    accuracy                           0.90     16682
   macro avg       0.91      0.90      0.90     16682
weighted avg       0.91      0.90      0.90     16682
```

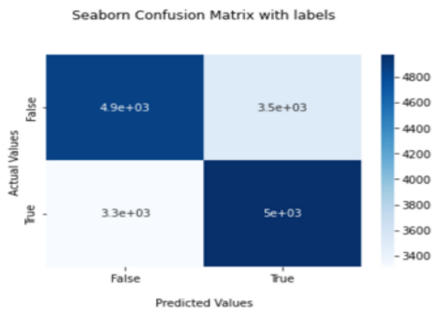


Figure 14 Result and Confusion Matrix

5.3 Decision Tree

Third model for our research is Decision Tree.

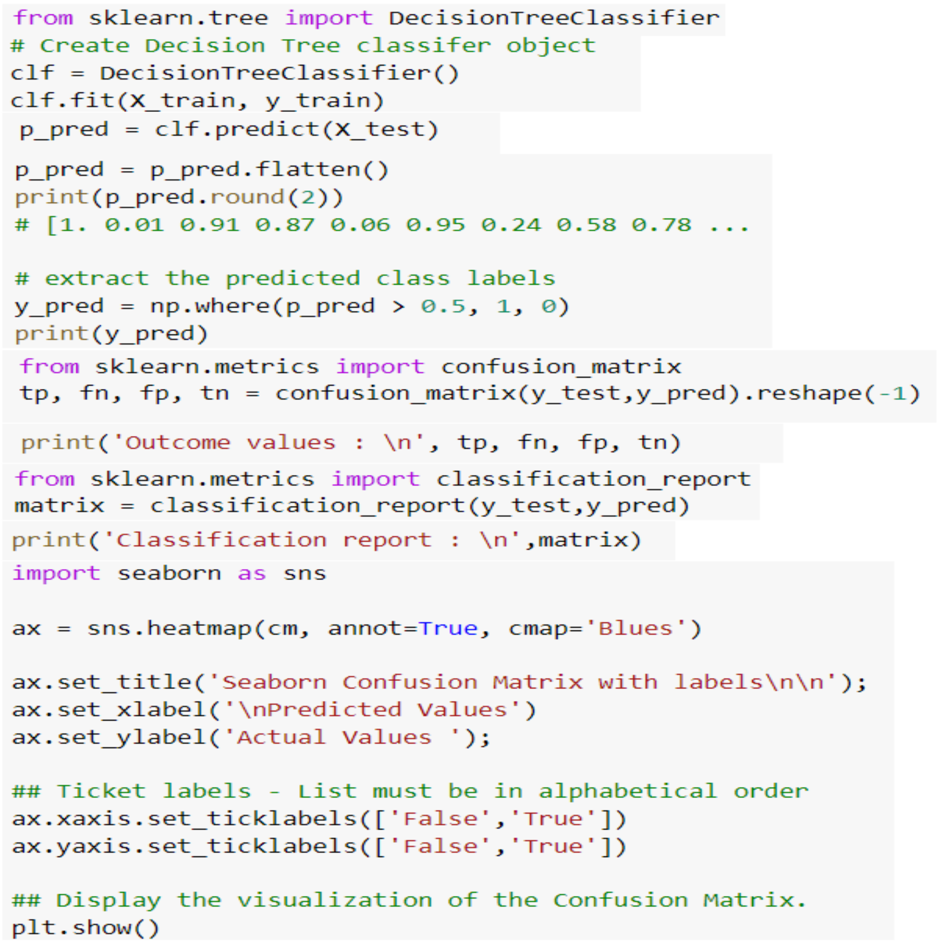
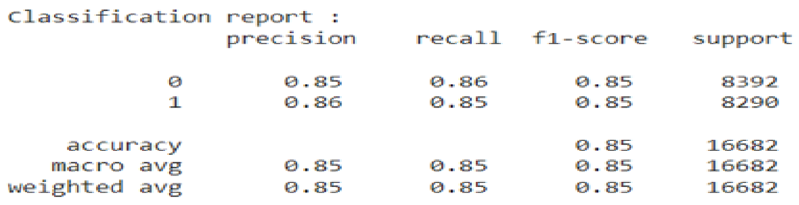

```
from sklearn.tree import DecisionTreeClassifier
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)
 p_pred = clf.predict(X_test)

p_pred = p_pred.flatten()
print(p_pred.round(2))
# [1. 0.01 0.91 0.87 0.06 0.95 0.24 0.58 0.78 ...

# extract the predicted class labels
y_pred = np.where(p_pred > 0.5, 1, 0)
print(y_pred)
from sklearn.metrics import confusion_matrix
tp, fn, fp, tn = confusion_matrix(y_test,y_pred).reshape(-1)

print('Outcome values : \n', tp, fn, fp, tn)
from sklearn.metrics import classification_report
matrix = classification_report(y_test,y_pred)
print('Classification report : \n',matrix)
import seaborn as sns

ax = sns.heatmap(cm, annot=True, cmap='Blues')

ax.set_title('Seaborn Confusion Matrix with labels\n\n');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');

## Ticket labels - List must be in alphabetical order
ax.xaxis.set_ticklabels(['False','True'])
ax.yaxis.set_ticklabels(['False','True'])

## Display the visualization of the Confusion Matrix.
plt.show()
```

Figure 15 Decision Tree



```
Classification report :
               precision    recall  f1-score   support

           0       0.85      0.86      0.85      8392
           1       0.86      0.85      0.85      8290

    accuracy                           0.85     16682
   macro avg       0.85      0.85      0.85     16682
weighted avg       0.85      0.85      0.85     16682
```

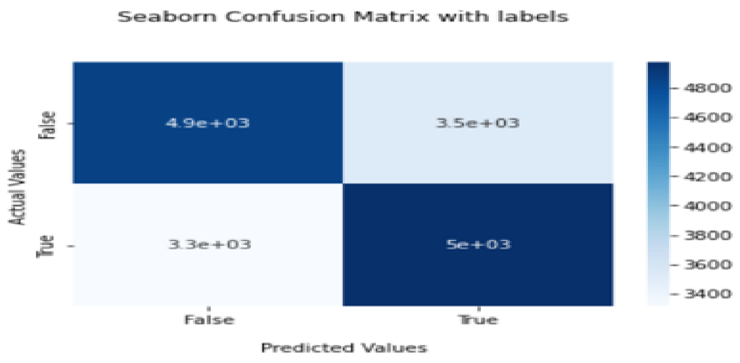
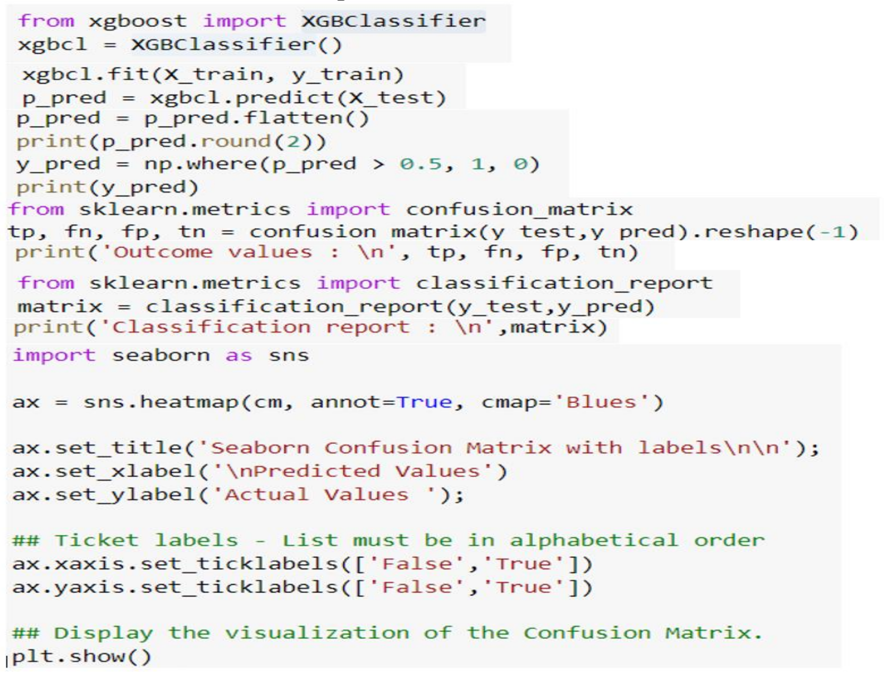


Figure 16 Result and Confusion Matrix

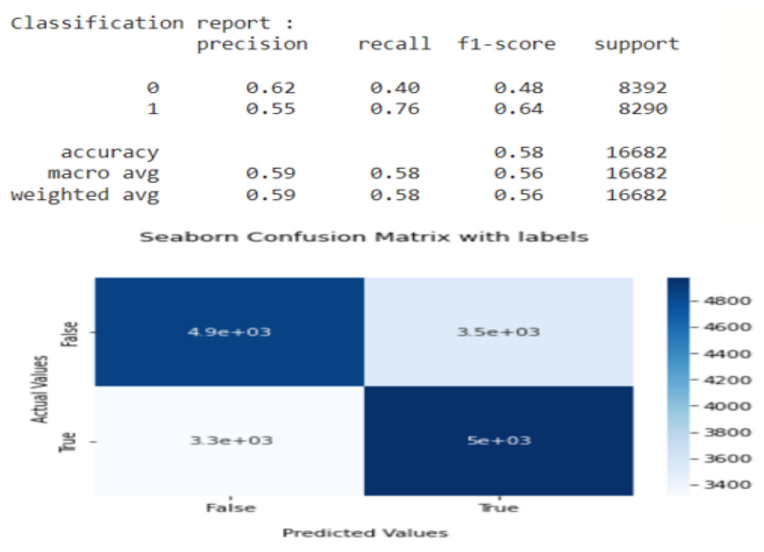
5.4 XGB Classifier

The last model we used for this implementation is XGB classifier.



```
from xgboost import XGBClassifier
xgbcl = XGBClassifier()
xgbcl.fit(X_train, y_train)
p_pred = xgbcl.predict(X_test)
p_pred = p_pred.flatten()
print(p_pred.round(2))
y_pred = np.where(p_pred > 0.5, 1, 0)
print(y_pred)
from sklearn.metrics import confusion_matrix
tp, fn, fp, tn = confusion_matrix(y_test,y_pred).reshape(-1)
print('Outcome values : \n', tp, fn, fp, tn)
from sklearn.metrics import classification_report
matrix = classification_report(y_test,y_pred)
print('Classification report : \n',matrix)
import seaborn as sns

ax = sns.heatmap(cm, annot=True, cmap='Blues')

ax.set_title('Seaborn Confusion Matrix with labels\n\n');
ax.set_xlabel('\nPredicted Values')
ax.set_ylabel('Actual Values ');

## Ticket labels - List must be in alphabetical order
ax.xaxis.set_ticklabels(['False','True'])
ax.yaxis.set_ticklabels(['False','True'])

## Display the visualization of the Confusion Matrix.
plt.show()
```

Figure 17 XGB Classifier



```
Classification report :
              precision    recall  f1-score   support

           0       0.62      0.40      0.48      8392
           1       0.55      0.76      0.64      8290

    accuracy                           0.58     16682
   macro avg       0.59      0.58      0.56     16682
weighted avg       0.59      0.58      0.56     16682
```

Figure 18 Result and Confusion Matrix

6 Result and Output

- Out of four models in this classification techniques are Random Forest, where it shows 90 percent of accuracy and second one is Decision Tree with 84 percent of accuracy rate.
- Around 60 percent of accuracy rate shown in Logistic Regression and XGB Classifier.

7 Internship Task Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you and what you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Prabhjeet Singh Multani Student number: x20153449

Company: Ornua Cooperative Limited Month Commencing: September-2021

Role Description:
Main aim and purpose of the internship was to research and study on Intrusion Detection System using Classification technique in Industrial Control System. Perform research to suggest a best possible solution or model to detect threat in ICS by using machine learning algorithms.
The task performed are:

1. Studied and analysed the Operational Technology Intrusion Detection System Documentation.
2. Carried out the study and research on the Machine Learning algorithms used in ICS.
3. Performed improvement activity for the proposed solution
4. Development of the models.
5. Evaluated the models and compare them to find the best model to detect threats.
6. Prepared the documentation of work performed.

Employer comments

- Prabhjeet performed exceptionally well throughout of this internship
- Prabhjeet's research on this topic provided positive results that can be implemented at our OT sites.
- Prabhjeet also used Machine Learning Techniques on public datasets and achieved good results.
- Prabhjeet is devoted to his given task and manages to complete it both in the office and remotely.

Student Signature: Prabhjeet Singh Multani Date: 07/01/2022

Industry Supervisor Signature: Declan Smith Date: 04/01/2022

8 References

Tutorialspoint.com. 2022. *Google Colab - What is Google Colab?*. [online] Available at: <https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm> [Accessed 6 January 2022].