Securing Data with User and Entity Behavioural Analysis (UEBA) approach using Machine Learning Models

MSc Research Project
Cybersecurity

Sumeet
Student ID: 19236123

School of Computing
National College of Ireland

Supervisor: Dr. Imran Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sumeet		
Student ID:	19236123		
Programme:	MSc Cybersecurity	**Year:**	2021-22
Module:	MSc Research Project		
Lecturer:	Dr. Imran Khan		
Submission Due Date:	16 December 2021		
Project Title:	Securing Data with User and Entity Behavioural Analysis (UEBA) approach using Machine Learning Models		
Word Count:	1031	**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.
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: [illegible]

Date: 16 December 2021

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

Sumeet
19236123

1 Introduction

This configuration manual contains all the details and complete explanation of the environmental setup of the project of Securing Data using UEBA approach using XGBoost. All the kinds of software and hardware configurations are updated in the columns.

2 System Configuration

2.1 Hardware Requirements

To setup the project, a list of hardware requirements were kept in mind before performing the activity.

Features	Description
System Information	64-bit Operating System, x64-based processor
Operating System	Windows 10
Memory	8 GB
Space	512 GB
Processor	Intel(R) Core(TM) i-5 3360 CP@2.80 GHz

2.2 Software Requirements

The research project was performed with the help of Anaconda Navigator where we have Jupyter Notebook 6.3.0. Python 3.9 was used as writing scripts. Pythons codes were used with the help of Anaconda Navigator.

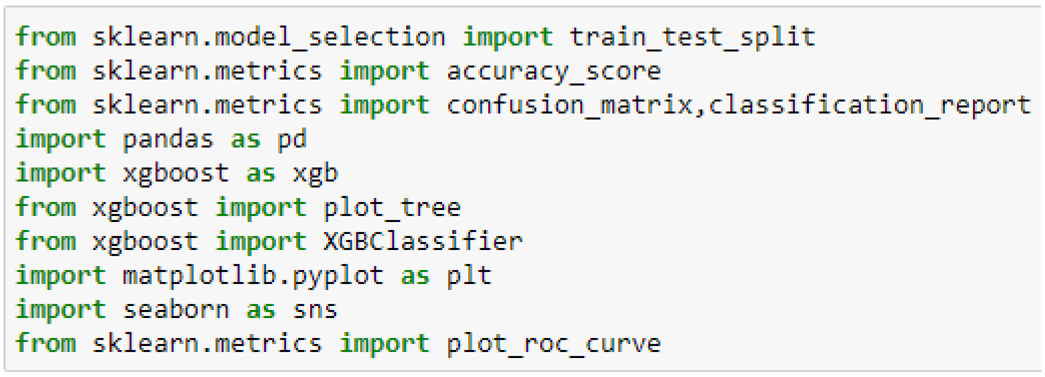
3 Data Selection

Behavioural dynamics was collected and gathered from the web chat from a random database. Two file information were gathered by (Martin, et al., 2020) as EVTRACKINFO where it has 347 number of records and also the static behaviour information are available.
The other one EVTRACKTRACK where 142691 records are available of which 29220 data for mouse dynamics and 112471 for keystrokes dynamics. There are a total of 11 users. To be more precise, a mean of 28524 ± 18541 are able to gather out of each user.

4 Implementation

4.1 XGBoost

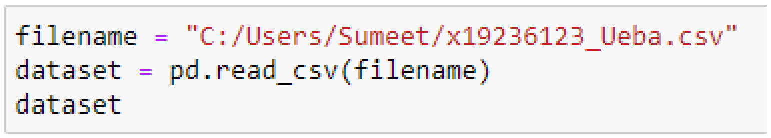
1. All the necessary libraries such as pandas, sklearn, etc. were imported for the use.



```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix,classification_report
import pandas as pd
import xgboost as xgb
from xgboost import plot_tree
from xgboost import XGBClassifier
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import plot_roc_curve
```

Figure 1: Libraries were imported

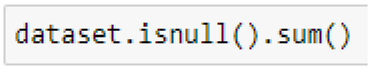
2. CSV file was read using Panda Libraries



```
filename = "C:/Users/Sumeet/x19236123_Ueba.csv"
dataset = pd.read_csv(filename)
dataset
```

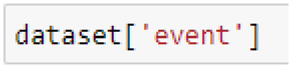
Figure 2: Read CSV file

3. Clean Data was checked such as null values if found.



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

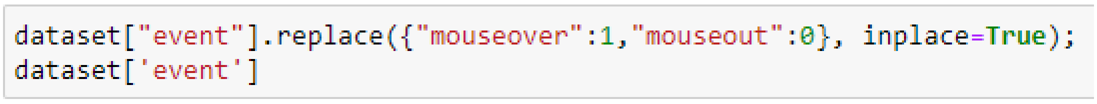
Figure 3: Clean Data check



```
dataset['event']
```

Figure 4: Check on Event in database

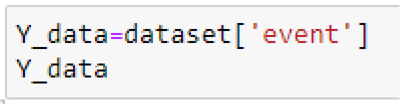
4. Converting Categorical Variable to Values of 0 and 1 for Machine Learning Model.



```
dataset["event"].replace({"mouseover":1,"mouseout":0}, inplace=True);
dataset['event']
```

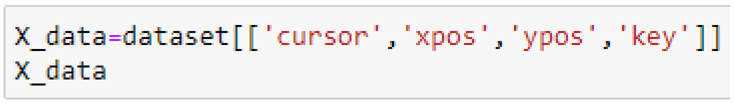
Figure 5: Declaration was predicted for 1 and 0

5. Separating Dependent and Independent Variables to be used in model.



```
Y_data=dataset['event']
Y_data
```

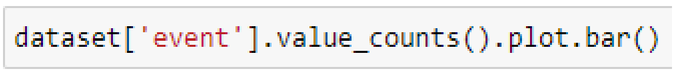
Figure 6: Variables were separated for Y



```
X_data=dataset[['cursor','xpos','ypos','key']]
X_data
```

Figure 7: Libraries were imported for X

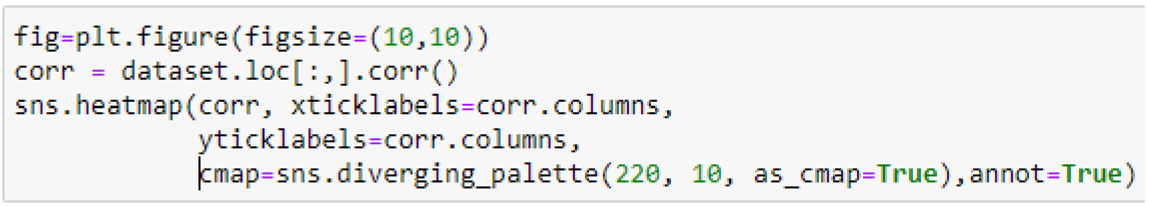
6. Checking event values present in the dataset separately.



```
dataset['event'].value_counts().plot.bar()
```

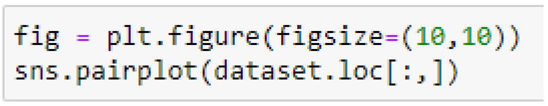
Figure 8: Event values were checked

7. Plotting Heatmap and Pairplot for checking correlation between independent variables.



```
fig=plt.figure(figsize=(10,10))
corr = dataset.loc[:,].corr()
sns.heatmap(corr, xticklabels=corr.columns,
            yticklabels=corr.columns,
            cmap=sns.diverging_palette(220, 10, as_cmap=True),annot=True)
```

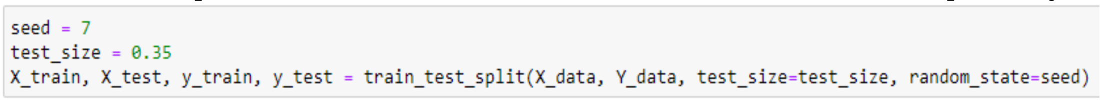
Figure 9: Heatmap generated



```
fig = plt.figure(figsize=(10,10))
sns.pairplot(dataset.loc[:,])
```

Figure 10: Pairplot generated

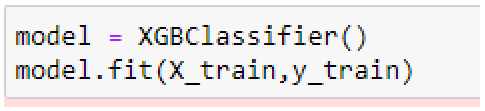
8. Datasets were splitted into train and test sets in ration of 65% and 35% respectively.



```
seed = 7
test_size = 0.35
X_train, X_test, y_train, y_test = train_test_split(X_data, Y_data, test_size=test_size, random_state=seed)
```

Figure 11: Libraries were imported

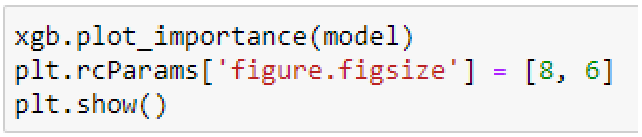
9. Implementing XGBoost Classifier.



```
model = XGBClassifier()
model.fit(X_train,y_train)
```

Figure 12: XGBoost Classifier Implemented

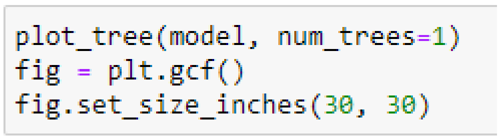
10. Feature Importance plot was plotted to fetch the importance of each feature in case of decision making.



```
xgb.plot_importance(model)
plt.rcParams['figure.figsize'] = [8, 6]
plt.show()
```

Figure 13: Feature importance plot

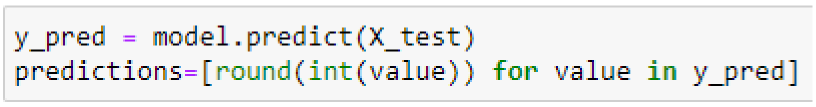
11. Decision Tree plot was used in the model.



```
plot_tree(model, num_trees=1)
fig = plt.gcf()
fig.set_size_inches(30, 30)
```

Figure 14: Decision Tree plot performed

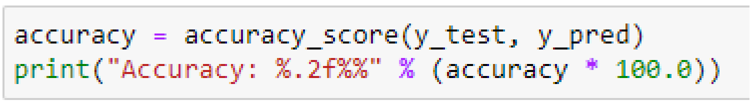
12. Predictions were made on the basis of model.



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

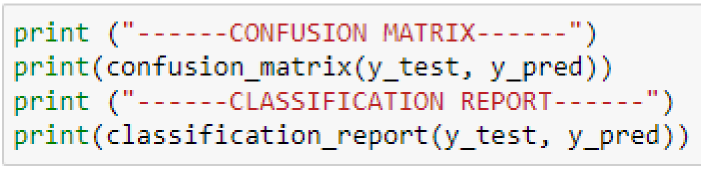
Figure 15: Predictions made

13. XGBoost model was evaluated on the test set using Classification Report and Accuracy.



```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
```

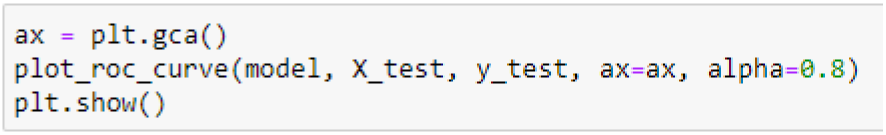
Figure 16: Accuracy obtained



```
print ("------CONFUSION MATRIX------")
print(confusion_matrix(y_test, y_pred))
print ("------CLASSIFICATION REPORT------")
print(classification_report(y_test, y_pred))
```

Figure 17: Confusion Matrix and Classification Report obtained

14. Receiver Operating Characteristics (ROC) Curve was plotted to observe Visualisation.

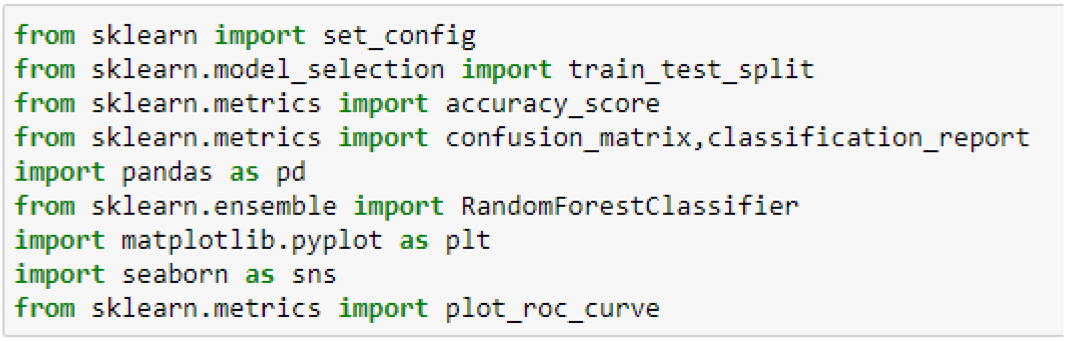


```
ax = plt.gca()
plot_roc_curve(model, X_test, y_test, ax=ax, alpha=0.8)
plt.show()
```

Figure 18: Plotted the ROC curve

4.2 Random Forest

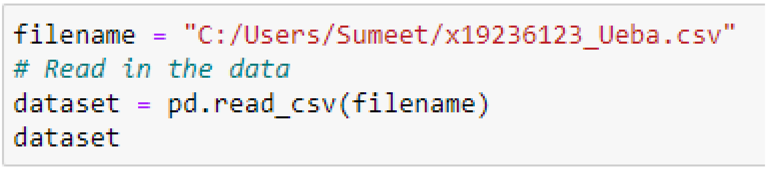
1. Importing all necessary libraries such as pandas, sklearn etc. to be used in this notebook.



```
from sklearn import set_config
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix,classification_report
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import plot_roc_curve
```

Figure 1: Libraries were used and imported

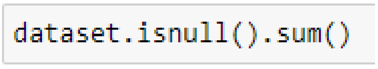
2. Reading CSV file data using Pandas Library.



```
filename = "C:/Users/Sumeet/x19236123_Ueba.csv"
# Read in the data
dataset = pd.read_csv(filename)
dataset
```

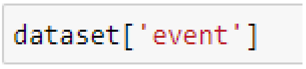
Figure 2: Panda Library used to read CSV

3. Checking for clean dataset (such as Null Values if present)



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

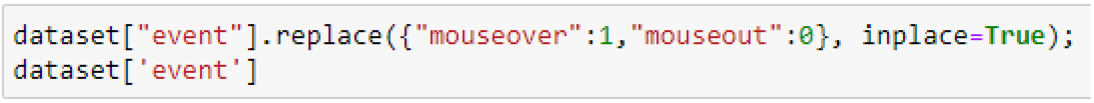
Figure 3: To check the null values



```
dataset['event']
```

Figure 4: Event dataset use

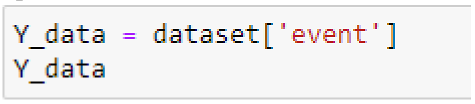
4. Converting categorical values to 0 and 1for machine learning



```
dataset["event"].replace({"mouseover":1,"mouseout":0}, inplace=True);
dataset['event']
```

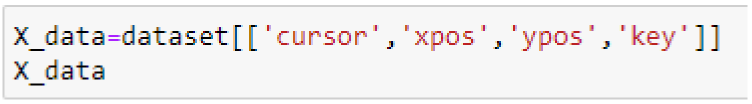
Figure 5: Categorical values

5. Separating independent and dependent variables to the model



```
Y_data = dataset['event']
Y_data
```

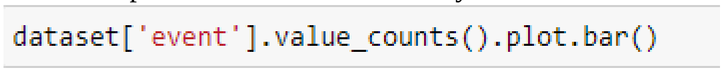
Figure 6: For Y



```
X_data=dataset[['cursor','xpos','ypos','key']]
X_data
```

Figure 7: For X

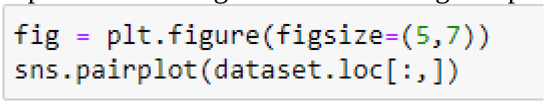
6. Checking event values present in the dataset already.



```
dataset['event'].value_counts().plot.bar()
```

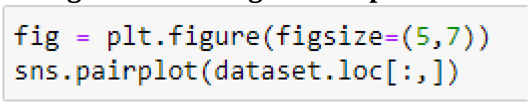
Figure 8: Bar plot print

7. Plotting Heatmap and Pairplot for checking correlation among independent variables.



```
fig = plt.figure(figsize=(5,7))
sns.pairplot(dataset.loc[:,])
```

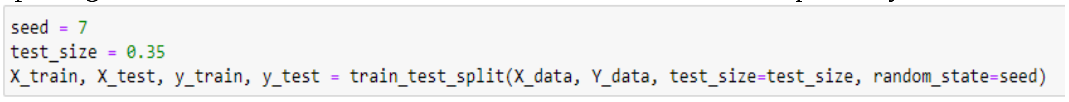
Figure 9: Plotting Heatmap



```
fig = plt.figure(figsize=(5,7))
sns.pairplot(dataset.loc[:,])
```

Figure 10: Pairplot

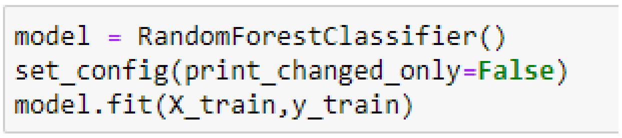
8. Splitting Datasets into Train and Test sets ratios of 65% and 35% repectively.



```
seed = 7
test_size = 0.35
X_train, X_test, y_train, y_test = train_test_split(X_data, Y_data, test_size=test_size, random_state=seed)
```

Figure 11: Split of datasets

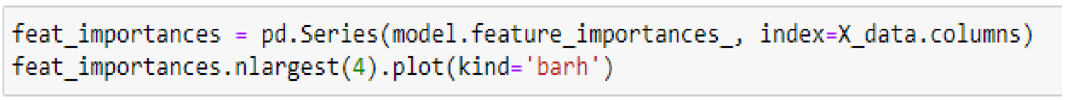
9. Importing Random Forest Classifier



```
model = RandomForestClassifier()
set_config(print_changed_only=False)
model.fit(X_train,y_train)
```

Figure 12: RF Implementation

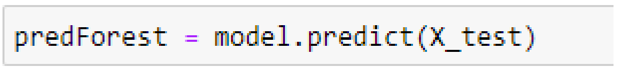
10. Plotting feature importance plot to get importance of each feature



```
feat_importances = pd.Series(model.feature_importances_, index=X_data.columns)
feat_importances.nlargest(4).plot(kind='barh')
```

Figure 13: Feature importance plot

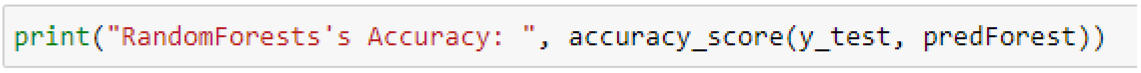
11. Making prediction on the basis of model



```
predForest = model.predict(X_test)
```

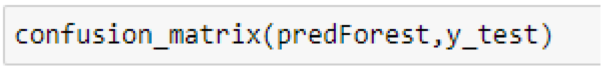
Figure 14: Predictions

12. Evaluating RF Model on the test set using Classification Report and Accuracy



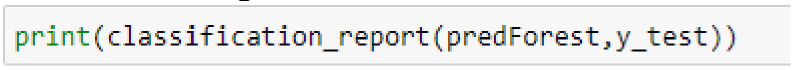
```
print("RandomForests's Accuracy: ", accuracy_score(y_test, predForest))
```

Figure 15: Check the accuracy



```
confusion_matrix(predForest,y_test)
```

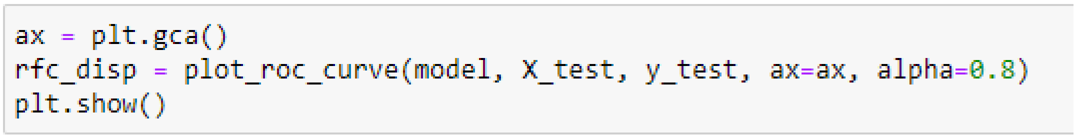
Figure 16: Confusion Matrix



```
print(classification_report(predForest,y_test))
```

Figure 17: Classification Matrix

13. Plotting ROC curve for evaluating model.



```
ax = plt.gca()
rfc_disp = plot_roc_curve(model, X_test, y_test, ax=ax, alpha=0.8)
plt.show()
```

Figure 18: ROC_auc slots full

5 Conclusion

The above steps helped to successfully implement UEBA data to XGBoost model and Random Forest.

6 References

Martin, A. G., Beltrán, M., Fernández-Isabel, A. & MARTIN, I., 2020. *Keystroke and Mouse Dynamics for UEBA Dataset.* [Online]
Available at: https://data.mendeley.com/datasets/f78jsh6zp9/2