Cyber Security of Critical Infrastructure
Assignment 4 Report

Gouni Anuraag Kiran(18111017)
Hariom(18111019)
M V Ravi Kumar(18111037)
Shikhar Barve(18111065)
Siddharth Kumar(18111068)
Kamlesh Kumar Biloniya(160317)

In our Assignment, we are provided with SWAT_Dataset_v0.xls and attack_merged.xls dataset which has :-

- Number of features = 51 which are the readings and status of different sensors and actuators present in different stages of Water Treatment Plant. These readings could be water flow rate,water level in a tank,chemical properties of water and possible states of actuators which could be on/off, flow/no_flow etc. There are 24,237 such records which we would use for Training, Validation and Testing. Training and Test set contains 18177 and 6060 records respectively.
- There is a labels associated with each record 0 to 7 depending on whether System is in 'Normal' condition or in one of the 6 'Attack' Conditions.
- We have 20000 records of Normal Condition and 4237 records of 6 different types of attack.

SWaT Architecture:-

Secure Water Treatment (SWaT), is a testbed for research in area of cyber security. SWaT consists of 6 Processes.

- Raw water supply and storage.
- Pre-treatment
- Ultrafilteraion and backwash
- Dechlorination System
- Reverse Osmosis
- RO Permeate Transfer, UF backwash, and Cleaning

Various water tanks, chemical tanks, water pumps chemical pumps are used in these stages. The cyber security portion of SWaT consists of a communications network, Programmable Logic Controllers (PLCs),Supervisory Control and Data Acquisition (SCADA) workstation and Human Machine Interfaces (HMIs). Sensors collect the data and transfer it to PLCs which is later transferred to SCADA station.

Data Analysis:-

Given data is real-life data prepared by *iTrust* to analyze the normal and attack conditions on a testbed. The SWaT Dataset systematically generated from the testbed over 7 days under normal operation and 4 days with attack scenarios.We are using only a part of whole dataset. The Main Aim was being able to predict normal and different type of attack scenarios in real-time by seeing readings/status from some or all of the sensors and actuators.

- Some data-features are real-valued.
- Some Data-features have few fixed states.
- Classes are imbalanced i.e. not of equal size.If we simply train our model for such data, model would be biased for a particular class. To resolve this problem we could use various methods.

    1. Upsampling :- A strategy to handle imbalanced classes by repeatedly sample with replacement from minority class to make it of equal size as the majority class.
    2. Downsampling :- we creates a balanced dataset by matching the number of samples in the minority class with a random sample from the majority class.
    3. Penalised Models :- Penalized classification imposes an additional cost on the model for making classification mistakes on the minority class during training. These penalties can bias the model to pay more attention to the minority class.
    4. class_weights :- using parameter *class_weight="balanced"* as a parameter we can give same weight to each class i.e. entries of minority class would have higher weight and those of Majority class would have lesser weights.

    Upsampling each class upto points in normal class increased data without giving any additional benefit over downsampling, so we used downsampling for normal class data and class_weights in our models.

- At each stage of process we could use data from any other stage.
- We have numerous classifiers namely $K-$NN, Linear SVM, Decision Tree, Logistic Regression, Random forest and many more. We can choose any one which suits our data best.

Model Selection :-

There are several models and each of them have there own pros and cons. We incorporated many of them in our code so we can find best model.

- Before training the model, we are removing duplicate examples from the dataset then *randomly* dividing whole data into 75% training and 25%testing data.
- We used *stratify=Y*, which takes a fixed percentage of data points from each class while extracting test points from data. This would maintain similar ratio of data points in training and testing data.
- We are using Leave-One-Out Cross-validation(LOOC) on 25% of entire dataset five times with different splits. Therefore, each execution of program will have different training and testing dataset which may lead to different accuracies.
- But in this way, we could not compare two models that's why we are adding a parameter *random state* fixing it to a particular value say 40 for reproducing ability of same test and training set. In that way we can compare two models.

1.Random Forest Classifier Model:-

It is an ensemble algorithm. It combines one or more than one algorithm of same or different kinds.It creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object.The subsets in different decision trees created may overlap.

Validation Accuracy :- 99.9967%
Testing Accuracy :- 100.00%

Forest	P_Attack1	P_Attack2	P_Attack3	P_Attack4	P_Attack5	P_Attack6	P_Normal
Attack1	232	0	0	0	0	0	0
Attack2	0	109	0	0	0	0	0
Attack3	0	0	106	0	0	0	0
Attack4	0	0	0	115	0	0	0
Attack5	0	0	0	0	69	0	0
Attack6	0	0	0	0	0	412	0
Normal	0	0	0	0	0	0	4996

2.SVM with Linear Kernel:-

It is highly efficient in dealing with extra large data sets.It is an ideal model for solving multiclass classification problems.Since Linear kernel used it would learn only linear boundaries.

Validation Accuracy :- 99.9801%
Testing Accuracy :- 100.00%

Since Test Accuracy is 100.00% here as well,then it would have same confusion matrix as above.

3.Bernoulli Naive Bayes Classifier:-

It is efficient classifier and suitable for discrete data.It is designed for Binary/Boolean features.

Validation Accuracy :- 99.9967%

Testing Accuracy :- 100.00%

Since Test Accuracy is 100.00% here as well,then it would have same confusion matrix as above.

4.ExtraTree or ExtraTrees classifier:-

Validation Accuracy Extrees :- 100.00%

Validation Accuracy Extree :- 99.9967%

Testing Accuracy :- 100.00%

Since Test Accuracy is 100.00% here as well,then it would have same confusion matrix as above.

5.Gaussian Naive Bayes classifier:-

Validation Accuracy :- 99.9967%

Testing Accuracy :- 100.00%

Since Test Accuracy is 100.00% here as well,then it would have same confusion matrix as above.

6. K Nearest neighbours:-

Validation Accuracy :- 100.00%

Testing Accuracy :- 100.00%

Since Test Accuracy is 100.00% here as well,then it would have same confusion matrix as above.

7.Logistic Regression:-

It is uses one vs all strategy to classify dataset. In this way our model learns 7 models and point is assigned to the class which has maximum probability.

Validation Accuracy :- 99.8940%

Testing Accuracy :- 99.9006%

Logistic	P_Attack1	P_Attack2	P_Attack3	P_Attack4	P_Attack5	P_Attack6	P_Normal
Attack1	232	0	0	0	0	0	0
Attack2	0	109	0	0	0	0	0
Attack3	0	0	106	0	0	0	0
Attack4	0	0	0	115	0	0	0
Attack5	0	0	0	0	69	0	0
Attack6	0	0	0	0	0	412	0
Normal	0	4	1	1	0	0	4990

8.Nearest Centroid Method(NCM):-
Validation Accuracy :- 100.00%
Testing Accuracy :- 99.9834%

NCM	P_Attack1	P_Attack2	P_Attack3	P_Attack4	P_Attack5	P_Attack6	P_Normal
Attack1	232	0	0	0	0	0	0
Attack2	0	109	0	0	0	0	0
Attack3	0	0	106	0	0	0	0
Attack4	0	0	0	115	0	0	0
Attack5	0	0	0	0	69	0	0
Attack6	0	0	0	0	0	412	0
Normal	1	0	0	0	0	0	4995

There are some other classifiers which we have used in our code and they have similar performance over this data.

Conclusion:-

From all models that we used in our code, **Tree-based classifiers** are the best options for the reasons which are as follows.

- It is very fast at testing time. Once trained, it takes only few comparison before predicting label.
- It does not get affected by unbalanced dataset.
- Multiple Decision trees ensure better accuracy.

We could have also trained two models 1^{st} model would check whether a point belongs to Normal Or Attack class. If the point belongs to Attack class then we could further use a multiclass classifier to decide which attack class it actually belongs to among 6 attack classes.But, we choose to do it with a single multiclass classifier model only.