COMP3217 Assignment 2: ML For Security

By Gherman Nicolisin (gn2g21)

Contents

1. Part A

1.1. Problem

Binary classification of 6000 system traces traces

1.2. Solution

The training data was split into 80/20 ratio. With 20% of data used for testing.

Logistic Regression Classifier was initally used and produced results of decent accuracy (**~90%**). Further action was taken to optimise a hyperparameter C, which resulted in the boost of **~3%**. **Decision Tree Classifier** was then used as it does not require scaling and normalisation of data and produced a model with **~96%** accuracy. **Support Vector Machine** was also tried but did not yield accurate model even with tuning. **Random Forest Classifier** was also tried since it operates well on continious data which is present in the dataset. The model yielded **~98%** accuracy.

Finally, during the part B classification, the **XGBoost** showed remarkable performance over the Random Forest Classifier when performing multiclass classification. Therefore, it was also tested in the part A with binary classification and showed a significant improvement in accuracy up to **~99%**.

1.3. Results

- Model: **XGBoost**
- Accuracy: **0.99 (99.00%)**
- F1 score: **0.9899959983993598 (99.00%)**
- Labesls

```
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
```

2. Part B

2.1. Problem

Multiclass classification of 6000 system traces traces

2.2. Solution

Similarly, as in part A, the data was split into 80/20 ration for training and testing purposes.

Random Forest Classifier was again tested and produced a consistent good accuracy of **~95%**. **Logic Regression Classifier** performed even worse than in the part A even with hyperparameter tuning with the accuracy of **~67%**, and significant downgrade in runtime speed. The **XGBoost** was discovered due to its incorporated regularization while working with this dataset and better runtime performance. The raw model perfomed faster and slightly more accurately with the score of **~96%** than the Random Tree Forest. Further tuning took place using `RandomizedSearchCV` to find optimum values for `colsample_bytree`, `gamma`, `learning_rate`, `max_depth`, `n_estimators`, `subsample` as well as running on scaled features. The calculated hyperparameters was then used in the refined model and gave further improvement in accuracy up to **~97%**.

2.3. Results

- Model: **XGBoost (tuned)**
- Accuracy: **0.97 (97.00%)**
- F1 score: **0.9699129151738807 (96.99%)**
- Labels:

```
[0 0 0 0 0 0 2 2 2 2 2 2 0 0 0 0 1 2 2 2 0 0 0 0 0 0 0 0 1 1 1 1 1 1 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 2
0 0 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

3. Code

The code can be acccess in Google Collab at
https://colab.research.google.com/drive/19A6xIVhpaDiVfeExI3M0p__8oQl3rU7D?usp=sharing

3.1. Usage

- Navigate to the link
- Executed code blocks as instructed at the beginning of the section A and the Section B
- (Optional) Executed other blocks to see the accuracy results for other models.

3.2. Local settings

- The notebook has been configured to use the local running image for better performance, therefore, it is important to change remote instance if you do not have the local execution environment running.
- If you run hyperparameter fitting for verifying the results, adjust `n_jobs` according to your environment computational capabilities.
- Finally, adjust the paths to the datasets' CSVs in the preludes depending on your configuration of the environment.