Assessment Task 3: Problem solving task

SIT-720 Machine Learning

October 9, 2022

T2-2022

Thanh Nguyen

STUDENT ID 218583133

COURSE - Bachelor of IT Honours (S470)

Contents

1 The Problem, Motivation and Related Works

The increase development of technologies comes with security problem. Many attack can negatively affect the operation of the system due to such security gaps. In this assignment, we conduct a comparative study to investigate the performance of different machine learning models in detecting potential security breaches using the datase *NSL-KDD*. The results of our study will then be compared to the paper by Ilhan et al [3].

We implement a range of machine learning models according to Ilhan et al [3] and propose our own novelty solution. Overall, our voting model has reached the score close to the highest

1.1 The Dataset and Pre-processing

We will be using the dataset *NSL-KDD* as the data input for our models. NSL-KDD dataset contains approximately 15000 records of different attacks types. In general, we can classify the attack types into four classes [1, 4]:

- **DoS**: [apache2, back, pod, processtable, worm, neptune, smurf, land, udpstorm, teardrop]
- **Probe**: [satan, ipsweep, nmap, portsweep, mscan, saint]
- **R2L**: [guess_passwd, ftp_write, imap, phf, multihop, warezmaster, warezclient, spy, xlock, xsnoop, snmpguess, snmpgetattack, httptunnel, sendmail, named]
- **U2R**: [buffer_overflow, xterm, sqlattack, perl, loadmodule, ps, rootkit]

For each attack type in the dataset, we map to the equivalence attack class. According to the paper by Ilhan et al. [3] the *Normal* class has 6817 records, 11617 records for *DoS*, 988 for *Probe*, 53 for *R2L* and 3086 for *U2R*. We thus reduce the number of record by randomly select the entries belongs to each classes for our data to match with the dataset used in the literature [3].

1.2 Machine Learning Models and Train/Test Spliting Strategy

In the origrinal paper by Ilhan et al. [3], there are eight machine learning models employed to classify our data, the specification of each machine learning model are implemented in SKlearn library [2]:

- **SVM Linear**: the `svm.LinearSVC` class.
- **SVM Quadratic**: `svm.SVC` class with polynomial kernel of degree 2.
- **SVM Cubic**: `svm.SVC` class with polynomial kernel of degree 3.
- **KNN Fine**: the `KNeighborsClassifier` class with 1 neighbor.
- **KNN Medium**: the `KNeighborsClassifier` class with 10 neighbors.
- **KNN Cubic**: `KNeighborsClassifier` class of Power parameter 3 with 10 neighbors.
- **Decision Tree Fine**: the `DecisionTreeClassifier` class with at least 100 splits. We have chosen the depth 7, which would result in 127 splits in the tree.
- **Decision Tree Medium**: the `DecisionTreeClassifier` class with at least 20 splits. We have chosen the depth 5, which would result in 31 splits in the tree.

The performance of the dataset will be compared by the *Accuracy*, *Precision*, *Recall*, *Geometric Meam* and *F1* metrics. The equations for each metric are given as Eqs.1, 2, 4, 3, and 5. We will be running 10 K-fold cross validation of 100 iterations for each classifiers.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2}$$

$$\text{Geometric\_mean} = \sqrt{\frac{TP * TN}{(TP + FN) * (TN + FP)}} \tag{3}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

$$\text{F1} = \frac{2TP}{2TP + FP + FN} \tag{5}$$

2 Our Solution

In our solution, we employ two ensemble machine learning models. Ensemble learning is a combination of many machine learning models to solve a problem. Machine learning models have their own limitations and weaknesses. If we combine those models, then we can boost the overall accuracy by aggregating the output of those models. Ensemble works best when the base models are not correlated with each other. The idea is that those models can fulfil each other's weaknesses.

The first ensemble method we implement is a Voting classifiers. This will utilise the results from four different weak learners to perform prediction. The four learners in our case are: a linear SVC, a KNN fine, a decision tree fine, and a linear classifier with stochastic gradient descent training.

The secone ensemble method is AdaBoosting. AdaBoost (adaptive boosting) is a machine learning model that used as an ensemble method. The idea is to use multiple classifiers to increase the accuracy of overall classifier. The classifiers are built on top of each other iteratively. Any machine learning model can be used the base classifier. We have chosen Linear Support Vector Machine as our base estimator, the booster will use 50 of this estimator to obtain the prediction.

The same dataset and K-fold cross validation process are used for both classifiers. The result of our solutions is discussed in Section 3

3 Results and Analysis

We present the results of our obsevation for the eight classifiers as the Table 1. Overall, we can achieve closely to the results as proposed by Ilhan et al [3]. We present the results of comparison between our implementation and the proposed results in Figure 1.

Overall, the accuracy scores of our implemented models are close to those of proposed results. However, our precision, recall, geometricall mean and f1 are mostly higher. The best score for intrusion detection as proposed by Ilhan et al. [3] was 99.46% for SVM Cubic, 99.64% for KNN fine and 99.92% for fine tree. While our SVM Cubic score lower at 98.27%, KNN fine and fine tree was comparatively at 99.73% and 99.64%.

Interestingly, in our own solutions, the score for accuracy, precision, recall and f1 are equal, that means the number of false positive prediction is equal to false negative. While adaptive boost model has overall lower score, our voting model can reach the range of 99%.

model	result	accuracy	precision	recall	geo mean	f1
SVM Linear	best	0.981826	0.981826	0.981826	0.988618	0.981826
	mean	0.971675	0.971675	0.971675	0.982234	0.971675
	std	0.003544	0.003544	0.003544	0.002228	0.003544
SVM Quadratic	best	0.978280	0.978280	0.978280	0.986391	0.978280
	mean	0.969557	0.969557	0.969557	0.980906	0.969557
	std	0.003411	0.003411	0.003411	0.002147	0.003411
SVM Cubic	best	0.982270	0.982270	0.982270	0.988896	0.982270
	mean	0.971948	0.971948	0.971948	0.982408	0.971948
	std	0.003395	0.003395	0.003395	0.002138	0.003395
KNN Fine	best	0.996897	0.996897	0.996897	0.998060	0.996897
	mean	0.992173	0.992173	0.992173	0.995104	0.992173
	std	0.001782	0.001782	0.001782	0.001116	0.001782
KNN Medium	best	0.992908	0.992908	0.992908	0.995564	0.992908
	mean	0.983041	0.983041	0.983041	0.989374	0.983041
	std	0.002766	0.002766	0.002766	0.001734	0.002766
KNN Cubic	best	0.994238	0.994238	0.994238	0.996396	0.994238
	mean	0.987720	0.987720	0.987720	0.992313	0.987720
	std	0.002246	0.002246	0.002246	0.001408	0.002246
Tree Fine	best	0.983156	0.983156	0.983156	0.989452	0.983156
	mean	0.966288	0.966288	0.966288	0.978844	0.966288
	std	0.003827	0.003827	0.003827	0.002406	0.003827
Tree Medium	best	0.998670	0.998670	0.998670	0.999169	0.998670
	mean	0.993561	0.993561	0.993561	0.995972	0.993561
	std	0.001847	0.001847	0.001847	0.001156	0.001847

Table 1: The results observed from different classifiers.

model	result	accuracy	precision	recall	geo mean	f1
Voting	best	0.995567	0.995567	0.995567	0.997228	0.995567
	mean	0.987880	0.987880	0.987880	0.992413	0.987880
	std	0.002269	0.002269	0.002269	0.001421	0.002269
AdaBoost	best	0.965426	0.965426	0.965426	0.978305	0.965426
	mean	0.876065	0.876065	0.876065	0.921126	0.876065
	std	0.050445	0.050445	0.050445	0.032936	0.050445

Table 2: The results from our solution.

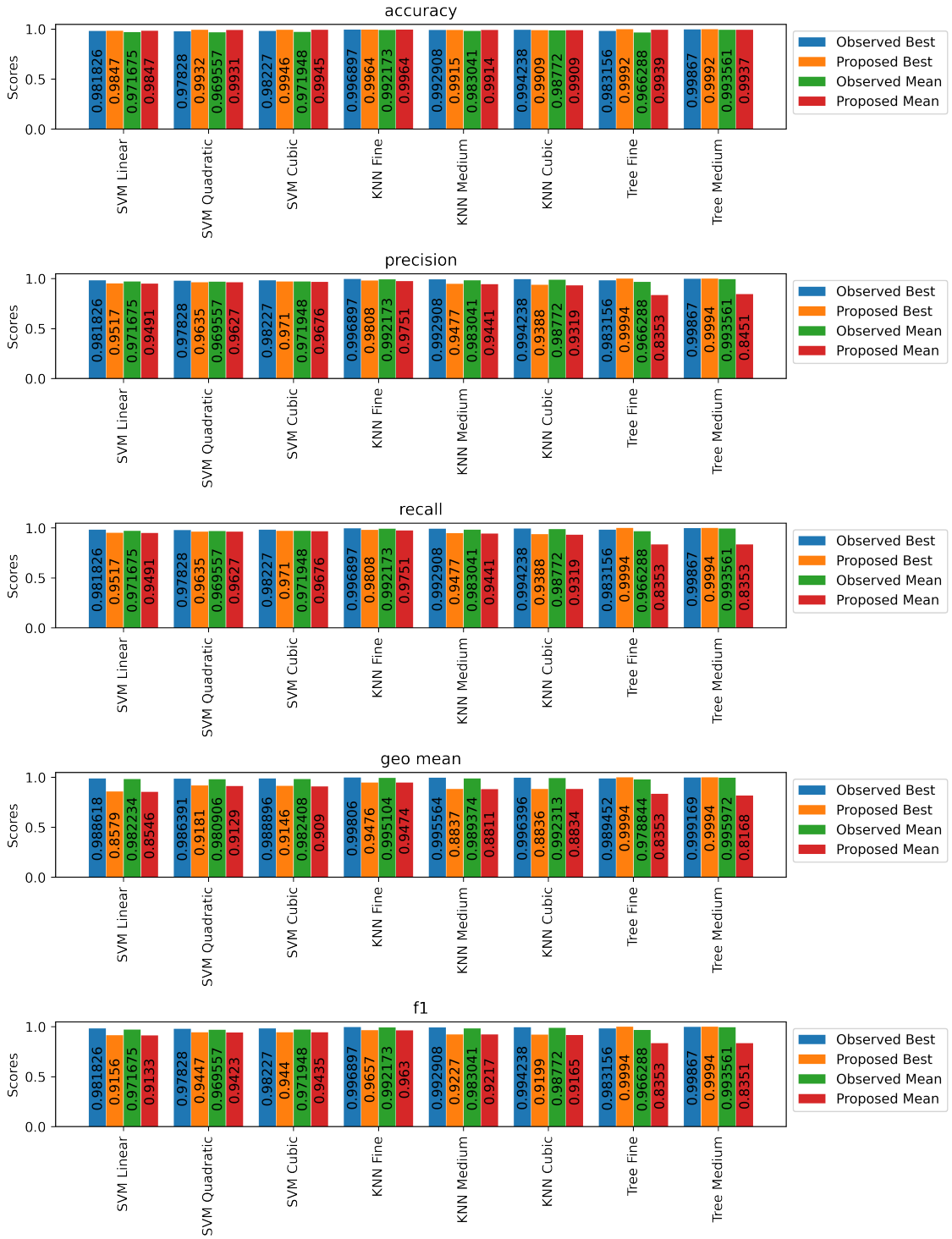

Figure 1: Comparison between our implementation and proposed models

References

[1] L. Dhanabal and S. Shantharajah. A study on nsl-kdd dataset for intrusion detection system based on classification algorithms. *International journal of advanced research in computer and communication engineering*, 4(6):446–452, 2015. 1.1

[2] J. M. Johnson and A. Yadav. Fault detection and classification technique for hvdc transmission lines using knn. In *Information and communication technology for sustainable development*, pages 245–253. Springer, 2018. 1.2

[3] I. F. Kilincer, F. Ertam, and A. Sengur. Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Computer Networks*, 188:107840, 2021. 1, 1.1, 1.2, 3

[4] A. M. Mahfouz, D. Venugopal, and S. G. Shiva. Comparative analysis of ml classifiers for network intrusion detection. In *Fourth international congress on information and communication technology*, pages 193–207. Springer, 2020. 1.1