School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

uOttawa

Report

CSI5388 - 2021 FALL Assignment 2

Kafka Server

Student Kun Yan kyan008@uOttawa.ca (300259303)

Task 1 – Binary Classification Problem

1 Algorithms description

In order to detect Network Intrusion, two solutions were proposed to perform a binary class classification and predict whether there is an ATTACK or not (BENIGN). One solution was based on ***a static RandomForestClassifier model*** using an initial csv data and test the model with data from Kafka server. Another solution was to construct and train ***an incremental learning model*** on ***KNNClassifier*** and to test it with stream data from Kafka server.

- A Static Solution

The first step was to deal with feature preprocessing for the initial csv data, including checking missing items, dropping NaN and infinity values, dropping 433 duplicate rows, transforming categorical 'Source' feature with OneHotEncoder[1] method, and using LabelEncoder[2] to transforming the Label feature to 0 and 1.

Secondly, before making a decision which algorithm can be used for my assignment, two types of anomaly detection (DBSCAN and IsolationForest Outlier Detector) on three algorithms were delivered, that is, RandomForestClassifier (RF), GradientBoostingClassifier (GRB), and MLPClassifier (MLP), respectively. Although GRB and MLP predictive performance improved after DBSCAN, ***the F1-score of RF before DBSCAN is still the highest*** (F1: 0.9874723655121592). Meanwhile, three algorithms got poorer Mean Absolute Error (MAEs) after doing iForest anomaly detection. Also, Liu[1] demonstrate in his paper that IsolationForest is not suitable for Random Forest. As a result, ***RandomForestClassifier was chosen as the classification estimator in my assignment***.

Finally, the static solution was implemented with below algorithms on initial csv dataset.

1) Executed the MinMaxScaler to scale and translate all features values between zero and one, so that they are dimensionless.
2) Chose the SMOTE algorithm to oversampling the minority class (Attack class) since the dataset is imbalanced.
3) Used the VarianceThreshold method to remove all low-variance features and select features.
4) Implemented classification with the RandomForestClassifier algorithm, which fits a number of decision tree classifiers on various sub-dataset.
5) Delivered the Pipeline algorithm to assemble above steps that can be cross-validated together by (Stratified)KFold algorithms.

- An Incremental Learning Solution

The online learning solution was designed and implemented using a Python package (River) on initial csv dataset and kafka stream data.

1) As with the static solution, all features values were scaled and translated between zero and one with the MinMaxScaler algorithm.
2) By comparing HoeffdingAdaptiveTreeClassifier (Accuracy: 90.11%, F1:0.604753) and KNNClassifier (Accuracy: 96.80%, F1:0.837577) algorithms, the latter was chosen for the adaptive model algorithm to predict 100,000 entries.
3) Pipeline algorithm was used to compose the above algorithms and steps so that I can predict and train the model adaptively.

2 Algorithm Evaluation: correct metrics, hyperparameters, performance estimation

- A Static Solution

In order to evaluate the static model algorithm, scoring parameter of cross_validate evaluation was adopted to obtain average values and standard deviation of five performance metrics, such as ***f1, precision, accuracy, recall and roc_auc***.

Considering the performance of the machine, the grid search was not adopted for parameter tuning, but the loop search was used to find the most parameters. The parameters and values of RandomForestClassifier algorithm were n_estimators=115, max_depth=23, max_features=17, min_samples_leaf=1(default), min_samples_split=2(default), and criterion='entropy'.

The Kafka data was used to test the classification model and solution. For the static solution, 100,000 Kafka data were read and transformed to a DataFrame, and then were predicted whether there was a network intrusion. The performance estimation metrics also were recorded as well.

- An Incremental Learning Solution

First of all, the adaptive model was constructed with the initial csv data and evaluated with accuracy and f1 metrics by reading the data as stream.

Then, the 100,000 Kafka stream data was predicted by the adaptive model (KNNClassifier). The evaluation metrics are accuracy and f1.

3 Presentation of results (tables)

Then, the 100,000 Kafka stream data was predicted by the adaptive model (KNNClassifier). The evaluation metrics are accuracy and f1.

Table 1-1: evaluation metrics of the static solution

The Static Solution	initial dataset	Kafka data
fit_time	35.943109	92.763388
score_time	0.182465	0.245371
test_precision	0.990226	0.993116
test_accuracy	0.998505	0.99881
test_f1	0.991753	0.993967
test_reacall	0.993313	0.994826
test_roc_auc	0.99965	0.999892

Table 1-2: Evaluation metric of both solutions

	initial dataset		Kafka data	
	test_accuracy	test_f1	test_accuracy	test_f1
Static Solution	0.998505	0.991753	0.99881	0.993967
Incremental Learning Solution	0.9697	0.833037	0.968	0.837577

4 Results discussion (comparison of results, advantages/ limitations, knowledge learned)

- Comparison of results

The test_accuracy shown the best performance on Kafka data with static solution (refer to table 1-2). Also, static solution demonstrated the better predictive score on Kafka data, while incremental learning solution has the higher test_accuracy score on initial dataset and the better test_f1 on Kafka data.

- Advantages / Limitations

One of the advantage of a static model is that the full data can be observed and do feature preprocessing, which are benefit to improve the predictive performance. By comparison, this is almost impossible for stream data.

The online learning model is more suitable for network intrusion detection because it is based on network flow. That is to say, when the network intrusion happened, the incremental learning model is swifter to find the attack.

- Knowledge learned

I have learned how to use the River package and skmultiflow package to construct an incremental learning model, and how to update it at any time.
Also, the assignment let me know how to consume the Kafka data and use it to test static model and dynamic model.

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Task 2 – Multiclass Classification Problem

1 Algorithms description

There were two solutions to implement the multiclass classification and prediction about IOT Botnet attack. In the static solution, the ***RandomForestClassifier*** algorithm also can be trained as a multiclass model on the initial dataset. In the incremental learning solution, One-vs-the-rest (OvR) multiclass strategy and the KNNClassifier algorithm were adopted to train the dynamic model and predict the Kafka stream data.

- A Static Solution

Based on the preprocessing of initial csv dataset, there were two ways to train the model. One was to split the dataset into train dataset and test dataset, then used the Pipeline to assemble the MinMaxScaler, SMOTE, feature selection (VarianceThreshold) and RandomForestClassifier steps. Another was to adpote cross_validate to run the Pipeline by 10 folds. Kafka data was download and used to test the model.

- An Incremental Learning Solution

The River package was used to implement the solution. The initial csv dataset was fitted to an adaptive OneVsRestClassifier (KNNClassifier ()) model as stream data. Each time, the data was predicted, then the metrics and model were update. The algorithm supports incremental learning using the learn_one method, and the window_size allowed to store the last 1000 observed samples.
Kafka steam data was predicted online and got the predictive performance metrics.

2 Algorithm Evaluation: correct metrics, hyperparameters, performance estimation

- A Static Solution

Since the static solution was developed by Scikit-learn, ***classification_report*** can be used to show the main classification metrics by multiclass names. In this assignment, the multiclass names were the types of attack about IOT Botnet. Also, ***cross_validate*** evaluation was adopted to get the specified metrics, such as f1_micro, precision_micro, balanced_accuracy, recall_micro, and so on.
Parameters tuning and performance estimation of RandomForestClassifier were done by loop searching since the computer was too slow to support the GridSearchCV. These parameters are n_estimators, max_depth, max_features, min_samples_leaf, min_samples_split, criterion, and so forth.

- An Incremental Learning Solution

In the River package, Accuracy and ClassificationReport classes provide a set of metrics for multiclass classification in a dynamic environment and update each of them every time. Performance estimation could be printed at any time during the model's lifetime.

3 Presentation of results (tables)

table2-1: Evaluation metrics of the static solution on initial dataset and Kafka data (method 1: cross validation)

Cross_validate	Static Solution - initial dataset	Static Solution - Kafka prediction
fit_time	233.721119	154.770235
score_time	0.08576558	0.15186474
test_recall_micro	0.999564921	0.99167027
test_f1_micro	0.999564921	0.99167027
test_precision_micro	0.99956492	0.99167027
test_balanced_accuracy	0.97017106	0.79606306

table 2-2: Evaluation metrics of the static solution on initial dataset and Kafka dataset (method 2: split dataset into train and test datasets)

	Static Solution - initial dataset				Static Solution - Kafka data			
Split dataset	precision	recall	f1-score	support	precision	recall	f1-score	support
BENIGN	1	1	1	6639	0.999861915	0.999988491	0.999925199	86892
mirai_udp_attack	1	1	1	4	0.95	0.513513514	0.666666667	37
mirai_ack_attack	1	1	1	104	0.988556898	0.998715478	0.993610224	1557
gafgyt_scan_attack	1	1	1	53	1	0.997442455	0.99871959	782
mirai_scan_attack	1	1	1	51	0.998851894	0.996563574	0.997706422	873
gafgyt_tcp_attack	1	1	1	126	0.999408983	0.998818665	0.999113737	1693
gafgyt_udp_attack	1	1	1	30	1	0.996996997	0.998496241	333
gafgyt_junk_attack	1	1	1	26	1	1	1	411
gafgyt_combo_attack	1	1	1	28	1	0.997624703	0.998810939	421
mirai_syn_attack	1	1	1	522	0.999855721	0.999423132	0.99963938	6934
mirai_udpplain_attack	1	1	1	3	1	1	1	70
macro avg	1	1	1	7586	0.994230492	0.954462455	0.968426218	100003
weighted avg	1	1	1	7586	0.999653323	0.99966001	0.999632134	100003
accuracy	1				0.99966001			

Table2-3: Evaluation metrics of the adaptive solution on initial dataset and Kafka data

initial dataset					kafka data				
	Precision	Recall	F1	Support		Precision	Recall	F1	Support
0	0.998	0.996	0.997	21962	0	0.998	0.996	0.997	86826
1	0.000	0.000	0.000	18	1	0.945	0.990	0.967	6956
2	0.875	0.987	0.928	377	2	0.067	0.006	0.011	32
3	0.746	0.645	0.692	2	3	0.744	0.639	0.688	793
4	0.291	0.300	0.296	213	4	0.597	0.534	0.563	399
5	0.572	0.700	0.630	446	5	0.302	0.308	0.305	883
6	0.188	0.031	0.054	96	6	0.564	0.692	0.621	168
7	0.657	0.512	0.575	86	7	0.874	0.981	0.925	1589
8	0.864	0.376	0.524	101	8	0.200	0.020	0.037	49
9	0.939	0.993	0.965	1774	9	0.732	0.363	0.485	452
10	0.000	0.000	0.000	11	10	1.000	0.036	0.069	56
Macro	0.557	0.504	0.515		Macro	0.638	0.506	0.515	
Micro	0.973	0.973	0.973		Micro	0.973	0.973	0.973	
Weighted	0.971	0.973	0.971		Weighted	0.971	0.973	0.971	
97.3% accuracy					97.3% accuracy				

4 Results discussion (comparison of results, advantages/ limitations, knowledge learned)

- Comparison of results

On the whole, f1-micro on initial dataset with static models is the highest performance (0.999564921). Specifically, as for static solution, the performance on initial dataset is better than that on Kafka data (refer to table 2-1). For the incremental learning solution, the accuracy, precision_micro, recall_micro and f1-micro performances are almost the same on initial dataset with the Kafka data (refer to table 2-3). Also in the static solution, by type of attack, there is no significant different on initial dataset, while on the Kafka stream data, the BENIGN (Majority class) shows the best performance (refer to table 2-2). It's obvious that for the dynamic Kafka data, the adaptive solution obtained the best accuracy performance (97.3%).

- Advantages / limitations

The advantage of a static model is that the model could get better understand about the dataset. However, it could not be able to detect the attack in time for the stream data that is time-sensitive.

The advantage of an Incremental learning model is suitable for online data stream, and it allows to predict data and update model at any time during the model's lifetime. One of the limitation is it's challenged to do data preprocessing, and it is not realistic to use all samples as they arrive.

- Knowledge learned

I have learned how to do multiclass classification both on csv dataset and Kafka data, especially how to calculate the correct metrics by attack types.
Also, some of metrics do not able to calculate for multiclass, like roc_curve.
Otherwise, it is necessary to do concept drift detection, and then partial fit the model or update the model, which can help keep the predictive performance.

[1] Liu, Fei Tony; Ting, Kai Ming; Zhou, Zhi-Hua (2008). "Isolation Forest". 2008 Eighth IEEE ICDM: 413-422