A Performance Evaluation Method of a Machine Learning Programming of Simple Logistic Regression

Yong Suk Suh*, Seung Ki Shin, Dane Baang, Sang Mun Seo, Jong Bok Lee
Research Reactor System Design Div., Korea Atomic Energy Research Institute (KAERI), Daedeok-Daero 989-111, Yuseong-Gu, Daejeon, 34057, Korea
Corresponding author: yssuh@kaeri.re.kr

1. Introduction

A machine learning programming (MLP) of simple logistic regression (SLR) was briefly reviewed in previous paper, which described the derivation of a logistic function and a cost function [1]. The SLR is used to classify the independent data into binary state: true (1) or false (0). The MLP of SLR uses a sigmoid function to predict response (dependent) data of trained (independent) data. The MLP of SLR can be applied in nuclear power plants to classify measured data (events or symptoms) into the binary state on the basis of experience, analysis or engineering judgment. The result of the MLP of SLR depends on the initial value, learning rate and epoch [1].

This paper briefly reviews a method for evaluating the performance of the MLP of SLR. The results in this paper are generated using Python [2] programming language and Tensorflow [3] and Scikit-learn [4] library under Anaconda [5] development environment for programming the MLP of SLR.

2. Performance evaluation of MLP of SLR

A threshold (or a cut-off value) in the SLR is used as a decision value to classify the independent data into binary state: true (1) or false (0). Thus, the binary state of independent data depends on the threshold. In order to explain this, we assume four test cases Case-1, 2, 3 and 4 with arbitrary x values as independent data and y values as dependent data (Table 1). When we run the MLP of SLR with initial value set to 1, learning rate set to 0.01 and epoch set to 1000 for the four assumed test cases, we obtain four S-shaped curves (Fig. 1).

Table 1: Test cases for logistic programming

Case-1	x	-4	-3	-2	-1	0	1	2	3	4	5
	y	0	0	0	0	0	1	1	1	1	1
Case-2	x	-4	-3	-2	-1	0	1	2	3	4	5
	y	0	0	1	1	1	1	1	1	1	1
Case-3	x	-4	-3	-2	-1	0	1	2	3	4	5
	y	0	0	0	0	0	1	0	1	1	1
Case-4	x	-4	-3	-2	-1	0	1	2	3	4	5
	y	0	0	0	0	0	1	0	0	1	1

When we assume three thresholds set to 0.1, 0.5 and 0.9 represented by the black dotted lines shown in Fig. 1, we obtain the results of SLR according to the thresholds (Table 2). Thus, the thresholds are used as decision values to convert the probabilities into the binary states. In Table 2, values in "y" are trained data and values in "$\widehat{y_{0.1}}$" are predicted data when threshold is set to 0.1, values in "$\widehat{y_{0.5}}$" are predicted data when threshold is set to 0.5, and so on.

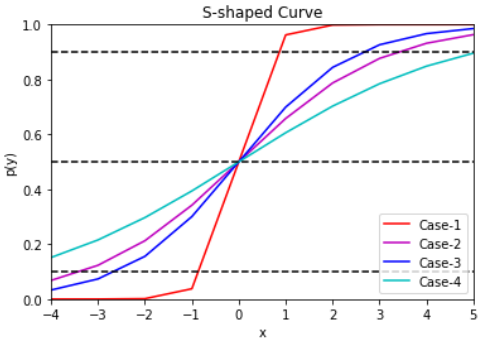

Fig. 1. S-shaped curves of MLP of SLR

Table 2: Predicted data at thresholds of 0.1, 0.5, 0.9

Case-1	y	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.1}}$	0	0	0	0	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	1	1	1	1	1
Case-2	y	0	0	1	1	1	1	1	1	1	1
	$\widehat{y_{0.1}}$	0	1	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	0	1	1
Case-3	y	0	0	0	0	0	1	0	1	1	1
	$\widehat{y_{0.1}}$	0	0	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	1	1	1
Case-4	y	0	0	0	0	0	1	0	0	1	1
	$\widehat{y_{0.1}}$	1	1	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	0	0	0

In Table 2, the yellow highlighted data do not match their y data counterparts. The test case Case-1 shows accurate classifications because the trained y data are well classified. The Case-2 shows worse classification than the Case-1 because the y data are biased to 1 but better than the Case-3 and 4 because no false y data exist. The Case-3 shows better classification than the Case-4 because less false y data exist.

A confusion matrix is used as a metric to evaluate the predicted data generated by a logistic regression [6]. The confusion matrix is defined as a specific table layout that shows the performance of a supervised MLP of logistic regression (Table 3). The performance of an unsupervised MLP of logistic regression can be shown using the matching matrix [6]. This paper only considers the supervised MLP of logistic regression.

As shown in Table 3, there are four cases in evaluating the outcomes of SLR: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). TP means that the SLR correctly predicts the true outcome as actual data is true. FP means that the SLR incorrectly predicts the true outcome as actual data is false, which is called a false alarm and Type I error. TN means that the SLR correctly predicts the false outcome as actual data is false, which is Type II error. FN means that the SLR incorrectly predicts the false outcome as actual data is true.

There are various measurement terms: accuracy, precision, sensitivity, specificity, and so on [6]. In order to explain the terms, we assume that the SLR predicts six true and four false data from ten actual data containing five true and five false data (Table 3).

Table 3: Confusion matrix (typical example)

		Actual data=10	
		True=5	False=5
Predicted data=10	True=6	TP=4	FP=2
	False=4	FN=1	TN=3

The accuracy is overall correctness of prediction by calculating: accuracy = (TP + TN) / (TP + FP + TN + FN) = 0.7.

The precision is overall correctness of true prediction by calculating: precision = (TP) / (TP + FP) = 0.67.

The sensitivity (called a recall) is how many actual true data are predicted as true data by calculating: sensitivity = TP / (TP + FN) = 0.8.

The specificity is how many actual false data are predicted as false data by calculating: specificity = TN / (TN + FP) = 0.6.

Table 4 shows a confusion matrix for Case-1, 2, 3 and 4 of Table 1. In Table 4, the FP and FN values are highlighted in yellow.

Table 4: Confusion matrix of Case-1, 2, 3 and 4

Case	Threshold=0.1		Threshold=0.5		Threshold=0.9	
1	TP=5	FP=1	TP=5	FP=0	TP=5	FP=0
	FN=0	TN=4	FN=0	TN=5	FN=0	TN=5
2	TP=8	FP=1	TP=5	FP=0	TP=2	FP=0
	FN=0	TN=1	FN=3	TN=2	FN=6	TN=2
3	TP=4	FP=4	TP=4	FP=1	TP=3	FP=0
	FN=0	TN=2	FN=0	TN=5	FN=1	TN=6
4	TP=3	FP=7	TP=3	FP=2	TP=5	FP=0
	FN=0	TN=0	FN=0	TN=5	FN=3	TN=2

Table 5 shows the accuracy, precision, sensitivity and specificity for Case-1, 2, 3 and 4 from the confusion matrix of Table 4.

Table 5: Performance matrix of Case-1, 2, 3 and 4

Case	Threshold	Accuracy	Precision	Sensitivity	Specificity
1	0.1	9/10	5/6	5/5	4/5
	0.5	10/10	5/5	5/5	5/5
	0.9	10/10	5/5	5/5	5/5
2	0.1	9/10	8/9	8/8	1/2
	0.5	7/10	5/5	5/8	2/2
	0.9	4/10	2/2	2/8	2/2
3	0.1	6/10	4/8	4/4	2/6
	0.5	9/10	4/5	4/4	5/6
	0.9	9/10	3/3	3/4	6/6
4	0.1	3/10	3/10	3/3	0/7
	0.5	8/10	3/5	3/3	5/7
	0.9	7/10	5/5	5/8	2/2

A receiver operating characteristic (ROC) curve is used to find a proper threshold. ROC is a method that came from a field called a signal detection theory developed during World War II for the analysis of radar images [7]. An ROC curve is a plot of true positive rate (TPR) against false positive rate (FPR). TPR is defined as a rate of sensitivity. FPR is defined as a rate of one minus specificity (1-specificity). We can easily plot the ROC curves of Case-3 and 4 with the performance matrix of Table 5 (Fig. 2). The pairs of TPR and FPR of Case-3 are (1.0, 0.66), (1.0, 0.16), and (0.75, 0.0) at threshold 0.1, 0.5 and 0.9, respectively. The pairs of TPR and FPR of Case-4 are (1.0, 1.0), (1.0, 0.28), and (0.61, 0.0) at threshold 0.1, 0.5 and 0.9, respectively.

Among the three thresholds, the threshold of 0.9 shows the best performance in terms of achieving a high TPR and low FPR.

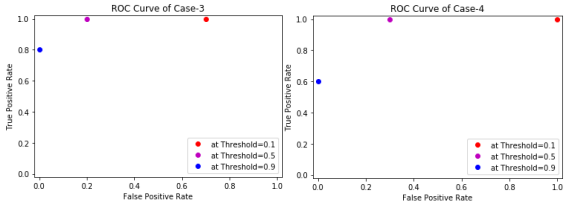

Fig. 2. ROC curves of Case-3 and 4

A Scikit-learn library can be used to easily plot the ROC curves (Fig. 3), which is free software for the Python programming language. Fig. 3 shows ROC curves of the Case-1, 2, 3 and 4. The Scikit-learn library selected thresholds as shown in Table 6 to plot the ROC curves.

The library also calculates the area under curve (AUC). The bigger the AUC is, the better the outcomes generated by the model.

Table 6: Thresholds selected by Scikit-learn library

Case	Thresholds					
1	1.99	0.99	0.96	2.3e-6		
2	1.96	0.96	0.21	0.07		
3	1.98	0.98	0.93	0.84	0.71	0.03
4	1.89	0.89	0.85	0.71	0.61	0.15

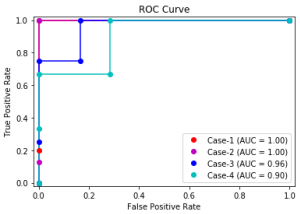

Fig. 3. ROC curves and AUC of Case-1, 2, 3 and 4

A good threshold can be found in an area representing a high TPR and low FPR. From Fig. 3, we can select a proper threshold: 0.9, 0.8 and 0.7 for Case-1, 2 and 3, respectively. After the predicted data are generated with some assumed good thresholds, the actual good threshold can be found (Table 7). For Case-1, an actual good threshold lies between 0.5 and 0.9. For Case-2, the threshold is 0.2. For Case-3, the threshold 0.9 is better than 0.5 because taking false negative outcome is better than taking false positive outcome. For Case-4, the actual good threshold is 0.8. These thresholds make us obtain a high TPR and low FPR rate.

Table 7: Predicted data at each threshold

Case-1	y	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.1}}$	0	0	0	0	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	1	1	1	1	1
Case-2	y	0	0	1	1	1	1	1	1	1	1
	$\widehat{y_{0.1}}$	0	1	1	1	1	1	1	1	1	1
	$\widehat{y_{0.2}}$	0	0	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	0	1	1
Case-3	y	0	0	0	0	0	1	0	1	1	1
	$\widehat{y_{0.1}}$	0	0	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.7}}$	0	0	0	0	0	0	1	1	1	1
	$\widehat{y_{0.8}}$	0	0	0	0	0	0	1	1	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	1	1	1
Case-4	y	0	0	0	0	0	1	0	0	1	1
	$\widehat{y_{0.1}}$	1	1	1	1	1	1	1	1	1	1
	$\widehat{y_{0.5}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.6}}$	0	0	0	0	0	1	1	1	1	1
	$\widehat{y_{0.7}}$	0	0	0	0	0	0	1	1	1	1
	$\widehat{y_{0.8}}$	0	0	0	0	0	0	0	0	1	1
	$\widehat{y_{0.9}}$	0	0	0	0	0	0	0	0	0	0

3. Conclusions

The results of a machine learning programming (MLP) of simple logistic regression (SLR) have been evaluated using a confusion matrix and receiver operating characteristic (ROC) curves. A Scikit-learn library can be used to plot the ROC curves. A proper threshold can be found in an area representing a high true positive rate (TPR) and low false positive rate (FPR) in the ROC curves. The performance evaluation methods for multiple (multi-variable) logistic regression and multinomial (multi-class) logistic regression need to be studied in the future.

REFERENCES

[1] Yong Suk Suh, et al., A Brief Review of a Machine Learning Programming of Simple Logistic Regression, Transactions of the Korean Nuclear Society Autumn Meeting, Yeosu, Korea, October 25-26, 2018.
[2] https://www.python.org/
[3] https://www.tensorflow.org/
[4] https://www. scikit-learn.org/
[5] https://www.anaconda.com/
[6] https://en.wikipedia.org/wiki/Confusion_matrix
[7] https://en.wikipedia.org/wiki/Receiver_operating_Characteristic