ISSN: 2508-7894 © 2022 KODISA & KAIA.
KJAI website: http://acoms.kisti.re.kr/kjai
Doi: http://dx.doi.org/10.24225/kjai. 2023.11.2.19

Network Traffic Measurement Analysis using Machine Learning

Hae-Duck Joshua Jeong[1]

Received: May 06, 2023. Revised: May 28, 2023. Accepted: June 02, 2023.

Abstract

In recent times, an exponential increase in Internet traffic has been observed as a result of advancing development of the Internet of Things, mobile networks with sensors, and communication functions within various devices. Further, the COVID-19 pandemic has inevitably led to an explosion of social network traffic. Within this context, considerable attention has been drawn to research on network traffic analysis based on machine learning. In this paper, we design and develop a new machine learning framework for network traffic analysis whereby normal and abnormal traffic is distinguished from one another. To achieve this, we combine together well-known machine learning algorithms and network traffic analysis techniques. Using one of the most widely used datasets KDD CUP'99 in the Weka and Apache Spark environments, we compare and investigate results obtained from time series type analysis of various aspects including malicious codes, feature extraction, data formalization, network traffic measurement tool implementation. Experimental analysis showed that while both the logistic regression and the support vector machine algorithm were excellent for performance evaluation, among these, the logistic regression algorithm performs better. The quantitative analysis results of our proposed machine learning framework show that this approach is reliable and practical, and the performance of the proposed system and another paper is compared and analyzed. In addition, we determined that the framework developed in the Apache Spark environment exhibits a much faster processing speed in the Spark environment than in Weka as there are more datasets used to create and classify machine learning models.

Keywords: Network traffic measurement, Machine learning, Network traffic analysis, Logistic regression, Support vector machine.

Major Classification Codes: Artificial Intelligence, Machine Learning

1. Introduction

With the advent of the Fourth Industrial Revolution, Internet traffic is increasing exponentially alongside the rapid increase in the usage of Internet of Things (IoT), mobile networks that connect to the Internet by embedding sensors, and communication functions in various objects and devices. Further, the rapid growth in usage of social networking services (SNS) such as Facebook and YouTube as well as the impact of COVID-19, whereby communication methods globally have shifted to online platforms, have inevitably led to an increase in Internet traffic. Within this context, research on machine learning-based network traffic analysis is drawing considerable attention (Alqudah & Yaseen, 2020; Barford & Plonka, 2001; Kelleher, Namee & D'Arcy, 2014).

In addition, information security issues regarding various Internet infringement incidents and various types of network attacks are emerging as serious issues. In previous

[1] First & Corresponding Author, Professor, Dept. of Computer Software, Korean Bible University, South Korea,
Email: joshua@bible.ac.kr

studies, network attack traffic was also detected by utilizing various classifiers such as SVM (Support Vector Machine) and logistic regression, which mainly use network traffic information such as end-to-end connection information, domain information, and data transmission information as feature extraction. analysis methods have been proposed. However, there are still limitations in perfectly detecting increasingly intelligent and advanced types of network attacks (Abbasi, Shahraki & Taherkordi, 2021; Jeong, Ryu, Ji, Cho, Ye & Lee, 2016; Lee, Ye & Jeong, 2014; Jeong, Ahn, Kim & Lee, 2017). Therefore, a machine learning-based study is urgently needed as an alternative to detecting a novel type of attack that is not detected by existing intrusion detection systems (Almomani, Almaiah, Alsaaidah, Smadi, Mohammad & Althunibat, 2021; Gitau, Rodrigues & Abuonji, 2020; Khan & Goodridge, 2019; Kulariya, Saraf, Ranjan & Gupta, 2016).

The main objectives in this paper are to design and develop a new machine learning framework in data mining toolkit, Waikato Environment for Knowledge Acquisition (Weka) and Spark based on Hadoop and Yarn. In addition, KDD CUP'99, one of the most widely used datasets for evaluating network traffic analysis systems, is used and machine learning is applied based on the well-known SVM method and logistic regression analysis model among (Tavallaee, Bagheri, Lu & Ghorbani, 2009), supervised learning methods (Kang & Choi, 2021; Saranya, Sridevi, Deisy, Chung & Khan, 2020). In two different environments, Weka and Spark compare and analyze time series type analysis, feature extraction, data formalization, network traffic measurement tools, and the results derived.

The structure of this paper is as follows. In Section 2, we look at machine learning algorithms that are widely used, and in Section 3, we look at performance evaluation indicators to evaluate machine learning algorithms. Section 4 we discuss the designing and development of the machine learning framework, and Section 5 compares and analyzes the performance evaluation results for machine learning algorithms by indicator, and finally concludes.

2. Machine Learning Algorithms

Machine learning algorithms are largely divided into supervised machine learning applied to prediction, estimation, and classification, and unsupervised machine learning types are applicable to pattern/rule, grouping, dimension reduction, video, image, text, and signal processing (Kim & Song, 2018). In this paper, we investigate how to classify traffic which includes both normal and abnormal traffic via the well-known support vector machine and logistic regression algorithm among machine learning techniques (Casas, Vanerio & Fukuda, 2017; Cortes & Vapnik, 1995; Murphy, 2012; Parihar & Yadav, 2022; Pentreath, 2015).

2.1. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning model for pattern recognition and data analysis as one of the machine learning fields, and is mainly used for classification and regression analysis. Given a set of data belonging to one of two categories, the SVM algorithm creates a non-probabilistic binary linear classification model that determines which category new data belongs to based on the given data set. The created classification model is expressed as a boundary in the data-mapped space, and the SVM algorithm is an algorithm that finds the boundary with the largest width among them. In addition to linear classification, SVM can also be used for non-linear classification. In order to perform nonlinear classification, it is necessary to map the given data into a high-dimensional feature space, and kernel tricks are used to do this efficiently (Perveen, Shahbaz, Guergachi & Keshavjee, 2016; Yuan, Li, Guan & Li, 2010).

2.1.1. Definition of SVM

In general, SVMs consist of a hyperplane or set of hyperplanes that can be used for classification or regression analysis. Intuitively, if the hyperplane has a large difference from the nearest training data point, the classification error is small. Thus, for a good classification, we need to find the hyperplane that has the furthest distance from the closest training data point to any classified point. In general, the initial problem is dealt with in a finite-dimensional space, but problems often occur in which the data are not linearly separated. In order to solve this problem, a method to facilitate separation by mapping from the finite dimension of the initial problem to a higher dimension has been proposed. In order to prevent an increase in the amount of calculation in the process, an SVM structure defining a kernel function $k(x,y)$ appropriate for each problem is designed so that the dot product operation can be effectively calculated using the variables of the initial problem (Cortes & Vapnik, 1995). A hyperplane in a high-dimensional space is defined as the dot product of a set of points and a constant vector. The vectors defined in the hyperplane are chosen to be a linear combination with the image vector parameters appearing in the database. In this selected hyperplane, the points $x$corresponding to the hyperplane have the following relationship.

$$\sum_i \alpha_i \ k(x_i, x) \ = \ constant. \qquad (1)$$

If $k(x,y)$ gets smaller as x and y move further apart, each sum represents the degree of proximity between the test point x and the corresponding data point x_i. In this way, the sum of the above kernel equations can be used to measure the relative proximity between the test points and the data points in the set you want to distinguish. When the point x in the non-convex set in the initial space is mapped to a higher dimension, it can become rather more complicated and difficult, but attention needs to be given to this aspect.

Classifying data is a common task in machine learning. Assuming that given data points belong to each of the two classes, the goal is to determine which of the two classes a new data point belongs to. In SVM, given a p-dimensional vector (a list of p numbers), we want to see if we can classify these data points into a (p-1)-dimensional hyperplane. This task is called linear classification. Hyperplanes that classify data can come in many cases. One reasonable way to select a hyperplane is to choose the hyperplane with the largest classification or margin between the two classes. So, we choose the hyperplane that maximizes the distance between the data points of each class closest to the hyperplane. If such a hyperplane exists, the hyperplane is called the maximum margin hyperplane, and the linear classifier is called the maximum margin classifier.

2.1.2. Linear SVM

We may define a given training data set D (a set of N points) as follows.

$$D = \{(x_i, y_i) | x_i \in R^p, y_i \in \{-1, 1\} \}_{i=1}^{n} \qquad (2)$$

Each x_i is a p-dimensional vector of real numbers, and y_i has a value of 1 or -1 indicating which class x_i belongs to. When the above training data set can be linearly separated according to the y_i value, the separation of the data set is called a hyperplane and can be represented as a set of points X that satisfy the following conditions. $wx - b = 0$ is the inner product operator, and w is the normal vector of the hyperplane.

The support vector (X^+, X^-) of a given hyperplane is defined as:

- X^+ : Among the data in y_i = +1, the data closest to the hyperplane
- X^- : Among the data in y_i = -1, the data closest to the hyperplane.

A hyperplane that passes through X^+ and has the same normal vector as a given hyperplane can be denoted by $wx - b = +1$, and similarly, a hyperplane that passes through X^- is denoted as $wx - b = -1$. The hyperplane margin means the distance between the hyperplanes passing through each support vector. If you find the distance between two hyperplanes with geometry, that is, the margin, it is known that $\frac{b}{\|w\|}$ is the SVM, and SVM is an algorithm that maximizes the margin.

Since there should be no data points between the hyperplanes passing through the support vectors, the following formula holds.

$wx_i - b \geq +1$ for x_i with y_i = +1, and
$wx_i - b \leq -1$ for x_i with y_i = +1.

The above two expressions can be expressed as:

$$y_i(wx_i - b) \geq 1, for\ all\ 1 \leq i \leq n. \qquad (3)$$

The SVM problem that follows the hyperplane condition and seeks the maximum value of the margin can be expressed as the following optimization problem.

$$arg \min_{(w,b)} \| w \|, \quad (4)$$

but $y_i(wx_i - b) \geq 1, for\ all\ 1 \leq i \leq n.$

For more details on circular form, dual form, biased and unbiased hyperplanes, soft margin, dual form and nonlinear classification, refer to Murphy (2012) and Bell (2015).

2.2. Logistic Regression

Regression analysis is the most widely used analysis method when analyzing the correlation between one dependent variable and several independent variables. General regression analysis assumes that the change in the dependent variable changes linearly with the independent variables. On the other hand, logistic regression analysis assumes that the relationship between the dependent variable and the independent variable is non-linear and can analyze the relationship between the independent variable and the dependent variable that has only two values to estimate the logistic regression coefficient (Murphy, 2012; Perveen, Shahbaz, Guergachi & Keshavjee, 2016).

Logistic regression is a statistical model in mathematics that uses a logistic function or logit function as an equation between x and $f(x)$. As shown in Equation (5) below, $f(x)$ is a dependent variable or a response variable, and $x_1, x_2, \ldots, x_k$ is an independent variable. $\beta_0, \beta_1, \ldots, \beta_k$ in this model is a parameter or regression coefficient, and is usually estimated through MLE (Maximum Likelihood Estimate). The logistic function can also be used to obtain the failure-to-success ratio or the log probability, which is mathematically calculated as $\frac{p}{1-p}$, and the log probability is $ln(\frac{p}{1-p})$.

$$f(x) = \frac{1}{1 - e^{-x}} \qquad (5)$$

$$ln(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k. \qquad (6)$$

This method tests various beta values over multiple iterations to optimize for the best fit line of log odds. Each iteration in this way creates a log likelihood function, and logistic regression analysis maximizes this function to find the optimal parameter estimate. Once we have found the optimal coefficient (or multiple coefficients if we have more than one independent variable), we can calculate the conditional probability of each observation, take its log, and sum them to get the predicted probability. For binary classification, we predict 0 if the probability is less than 0.5 and 1 if it is greater than 0.5. After the model is calculated, it is best to evaluate how well the model predicts the dependent variable, which is called goodness-of-fit. The Hosmer–Lemeshow test is the most used method to evaluate model goodness-of-fit.

When comparing logistic regression with other machine learning algorithms, it has the advantage of being able to process large amounts of data at high speed because it is mathematically simple and requires less computational capacity such as memory and processing performance.

2.2.1. Three types of logistic regression models

There are three types of logistic regression analysis: binary logistic regression model, multinomial logistic regression model, and ordinal logistic regression model. In the case of the binary logistic regression model dealt with in this study, the dependent variable (or response variable) can only belong to one of two categories, that is, the dependent variable can have only two values such as yes or no, or 0 and 1. Even though the logistic function is calculated with a range of values between 0 and 1, the binomial regression model rounds the answer to the nearest value. In general, answers less than 0.5 are rounded down to 0, and answers greater than 0.5 are rounded up to 1, so the logistic function returns a binomial result.

Multinomial logistic regression model can analyze problems that can have more than 3 outcomes under the premise that the number of outcomes is finite. For example, one can predict whether house prices will increase by 25%, 50%, 75% or 100% based on population data, but cannot predict the exact price of a house. Multinomial logistic regression works by mapping the resulting value to some other value between 0 and 1. Since the logistic function can return a range of continuous data, such as 0.1, 0.11, and 0.12, multinomial regression analysis can more accurately predict housing prices by grouping the output values into the closest possible values.

Table 1: Three types of logistic regression models

	Binomial logistic regression	Multinomial logistic regression	Ordinal logistic regression
Number of categories for dependent variable	2	3 or more	3 or more
Does order of categories matter?	No	No	Yes

An ordinal logistic regression analysis or ordinal logit model is a special type of multinomial regression analysis for solving problems in which numbers represent ranks rather than actual values. For example, one can use ordinal regression to predict answers to survey questions asking customers to rate a service as poor, good, very good, or excellent based on a numeric value, such as the number of items purchased in a year. Table 1 summarizes the three models.

3. Machine Learning Performance Evaluation Index

In supervised learning, a predictive model is created through machine learning using pre-existing training data. Using that model, it is to determine which class the newly introduced data belongs to. As such, when creating a model, it is created based on training data. To evaluate such supervised learning, cross validation is performed through data set classification. Cross-verification can evaluate training performance by dividing data into a training set, a validation set, and a test set and measuring precision and recall for trained functions through cross-verification. In addition, the F1-score value and accuracy can be found (Murphy, 2012; Perveen, Shahbaz, Guergachi & Keshavjee, 2016).

The factor that evaluates the classification model can eventually be defined as the relationship between the predicted answer from the classification model and the actual correct answer. The answer is divided into True and False, and the classification model also gives an answer of True or False. Therefore, as shown in Table 2, it can be divided into 4 cases with a 2x2 confusion matrix.

Table 2: 2 x 2 Confusion matrix

		Predicted values	
		True(T)	False(F)
Actual values	True(T)	TP(True Positive)	FP(False Positive)
	False(F)	FN(False Negative)	TN(True Negative)

True Positive (TP) predicts the answer that is True correctly as True and is the correct answer. False Positive (FP) predicts the answer that is actually False incorrectly as True and is the incorrect answer. False Negative (FN) predicts the answer that is actually False as True and is an incorrect answer. Finally, True Negative (TN) predicts the correct answer that is actually False correctly as False and is the correct answer. In this way, the performance of the classification model for each case can be evaluated through evaluation data such as precision, recall, accuracy, and F-score.

3.1 Precision

Precision refers to the proportion of answers that are actually true among those that the classification model classifies as true. This can be expressed in the same way as Equation (7), which is also called Positive Predictive Value (PPV).

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

3.2 Recall

Recall refers to the ratio of answers predicted by the classification model to be true among those that are actually true as in Equation (8). Recall is also used as sensitivity in statistics and hit rate in other fields.

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

While both precision and recall are interested in the occasions where the model predicts that the correct answer is True, the only difference is that these each take a different point of view. Precision looks at cases whereby the correct answer is correct from the perspective of the classification model, while recall looks at cases whereby the correct answer is correct from the perspective of the actual correct answer (data). Precision and recall can be used interchangeably, and the higher both indicators are, the better the model.

3.3 Accuracy

Both the above precision and recall indicators are only applicable when they correctly predict True as True. However, when False is predicted to be False, it is also true. In this case, the applicable indicator is accuracy, shown as Equation (9).

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad (9)$$

Accuracy is an evaluation index that can most intuitively represent the performance of a model. However, it is crucial to recognize the bias of the domain. Therefore, indicators to compensate for this are required.

3.4 F1-score

F1-score consists of the harmonic average of precision and recall as shown in Equation (10). The F1-score can accurately evaluate the performance of the model when the data label has an imbalanced structure and can express the performance as a single number. Using the harmonic average reduces the bias that would impact the result value greatly compared to using the arithmetic average.

$$F1\text{-}score = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (10)$$

Therefore, using the forementioned four indicators, we analyze and evaluate the experimental results obtained on the newly designed and developed machine learning framework.

4. Design and Development of a Machine Learning Framework for Network Traffic Measurements

4.1 Weka Tool and Spark

We design and develop two versions of a machine learning framework based on Weka, a machine learning and data mining toolkit, and Spark, an open source cluster computing framework. The machine learning framework in Weka is designed and developed in Java in the Windows OS environment, and the machine learning framework for network traffic measurements (NTM) is designed and developed in the Apache Spark environment running on top of Linux OS and Hadoop 2.x version of YARN (Pakdel, 2019; Pentreath, 2015).

4.1.1 Weka Tools

Weka (Waikato Environment for Knowledge Acquisition) is a machine learning and data mining toolkit written in Java at the University of Waikato in New Zealand. It provides learning and visualization tools using workbench programs or command-line tools. Weka can import data from existing data sources with a JDBC driver, support access to big data by connecting to Hadoop, and can perform various machine learning tasks such as data mining, classification, regression, clustering, and association rules (Witten & Frank, 2002; Witten, Frank & Hall, 2011).

4.1.2 Spark

Spark is an open-source cluster computing framework for large-scale data processing. Spark can quickly process large amounts of data by using a distributed memory-based data processing method. It is well utilized in various fields such as machine learning, graph processing, and streaming processing. In addition, since Spark can be developed in various programming languages such as Scala, Java, Python, and R, this compatibility enables a broad range of users to use it (Bell, 2015; Pentreath, 2015).

Spark MLlib is a machine learning library provided by Spark that provides various machine learning algorithms such as classification, regression, clustering, and collaborative filtering. As per the advantages that Spark itself provides, MLlib is capable of processing large amounts of data using a distributed memory-based processing method. It enables efficient processing of machine learning tasks that involve large amounts of data (Barford, & Plonka, 2001; Kelleher, Namee & D'Arcy, 2014). Spark MLlib's classification and regression include linear model SVMs, logistic regression, and linear regression, as well as Nave Bayes and decision trees. Clustering includes k-means, Gaussian mixture, and power iteration clustering.

In this paper, as shown in Figure 1, the proposed machine learning framework for network traffic measurements (NTM) was linked with Spark in the Yarn environment based on Hadoop on Linux. The system consisting of one master and three multi node clusters (3 slaves) was also built in a Spark-based distributed environment. And, as shown in Figure 2, a predictive model was created through machine learning with the training dataset, and then the model was designed and developed to determine which class the newly introduced data belongs to.

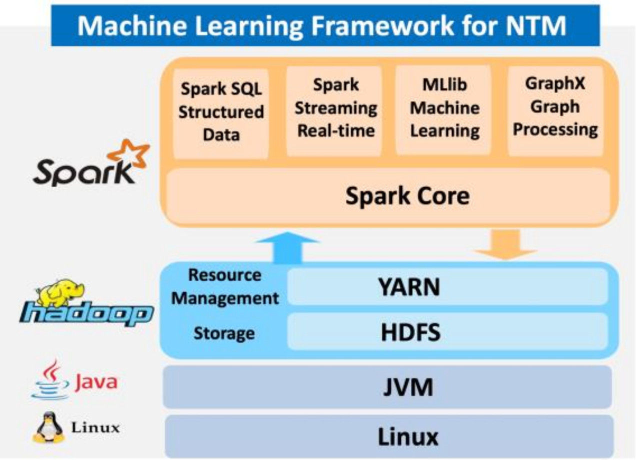

Figure 1: The proposed machine learning framework for network traffic measurements (NTM)

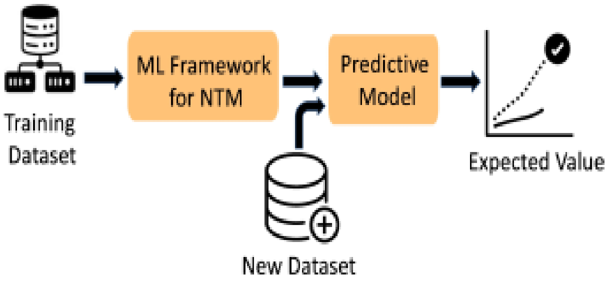

Figure 2: The proposed supervised learning-based machine learning prediction model

4.2 KDD Dataset

KDD CUP'99, as part of research support for DARPA, includes traffic with normal and abnormal connections on the network, and is one of the most widely used datasets for objective evaluation of the performance of IDS (Intrusion Detection System) (Choudhary & Kesswani, 2020; Gurung, Ghose & Subedi, 2019; Tavallaee, Bagheri, Lu & Ghorbani, 2009). This dataset has 18 million packet headers and is largely divided into 4 attack methods: Probe, DoS, U2R, R2L, and normal packets. The meaning of each packet is as follows:

- Normal: Normal packets.
- Probe: A packet that collects preliminary data (port, etc.) of the system before attempting an actual attack.
- DoS: Denial of Service. Packet attempting a denial-of-service attack.
- U2R: User to Root. Packets attempting to gain administrator (root) privileges.
- R2L: Remote to Local. Packets that an unauthorized user is attempting to gain access from outside.

4.3 Data Processing and Collection

4.3.1 KDD Dataset Parsing

Parsing, the process of decomposing and analyzing data from the KDD Dataset, reconstructing it in a desired form, then extracting it again, is as follows. ‘protocol_type’ consists of three types: TCP, UDP, and ICMP. ‘service’ consists of 70 items including ‘ftp data’, ‘flag’ consists of 11 items, and class consists of 23 items including ‘normal’. All strings in the dataset format are converted to integer data.

4.3.2 Building Support Vector Machine Model and Logistic Regression Model

4.3.2.1 Support Vector Machine Model

This Java code is a process for creating input/output objects of KDD Dataset file for SVM model.

```
public class KddSet_Parsing_SupportVectorMachineModel {
     public static void main(String args[]) throws IOException {
              BufferedReader fr = new BufferedReader(new
     FileReader(new File("data\\Weka_kddSet_Original.csv")));
              BufferedWriter fw = new BufferedWriter(new
     FileWriter(new
     File("data\\Spark_mllib_kddSet_SVM.txt")));
                         String sLine = null;
```

4.3.2.2 Logistic Regression Model

This Java code is a process for creating input/output objects of KDD Dataset file for the logistic regression model.

```
public class KddSet_Parsing_LogisticRegressionModel {
     public static void main(String args[]) throws IOException
     {
              BufferedReader fr = new BufferedReader(new
     FileReader(new
     File("data\\Weka_kddSet_Original.csv")));
              BufferedWriter fw = new BufferedWriter(new
     FileWriter(new
     File("data\\Spark_mllib_kddSet_LogisticRegression.txt"))
     );
                         String sLine = null;
```

4.3.3 Machine Learning Test

4.3.3.1 Support Vector Machine Test

This Java code is used to run the training algorithm to build the support vector machine model.

```
public class SVMWithSGDTest {
     public static void main(String[] args) throws IOException
     {
              SparkConf conf = new
     SparkConf().setAppName("JavaSVMWithSGDExample")
     ;
              SparkContext sc = new SparkContext(conf);
              String Modelpath =
     "data/SupportVectorMachine/kddSet_SVM_AllDataSet.tx
     t";
              String Predictpath =
     "data/SupportVectorMachine/kddSet_SVM_AllDataSet.tx
     t";
              JavaRDD<LabeledPoint> Modeldata =
     MLUtils.loadLibSVMFile(sc, Modelpath).toJavaRDD();
              JavaRDD<LabeledPoint> Predictdata =
     MLUtils.loadLibSVMFile(sc, Predictpath).toJavaRDD();
              JavaRDD<LabeledPoint> training = Modeldata;
              JavaRDD<LabeledPoint> test = Predictdata;
                         final SVMModel model =
SVMWithSGD.train(training.rdd(), numIterations);
```

4.3.3.2 Logistic Regression Test

This Java code is used to run the training algorithm to build the logistic regression model.

```
public class LogisticRegressionTest {
     public static void main(String[] args) {
              SparkConf conf = new
     SparkConf().setAppName("Multi class Classification
     Metrics Example");
              SparkContext sc = new SparkContext(conf);
              String Modelpath =
     "data/LogisticRegression/kddSet_LogisticRegression_All
     DataSet.txt";
              String Predictpath =
     "data/LogisticRegression/kddSet_LogisticRegression_All
     DataSet.txt";
              JavaRDD<LabeledPoint> Modeldata =
     MLUtils.loadLibSVMFile(sc, Modelpath).toJavaRDD();
              JavaRDD<LabeledPoint> Predictdata =
     MLUtils.loadLibSVMFile(sc, Predictpath).toJavaRDD();
              JavaRDD<LabeledPoint> training = Modeldata;
              JavaRDD<LabeledPoint> test = Predictdata;
                         final LogisticRegressionModel
model = new
LogisticRegressionWithLBFGS().setNumClasses(23).ru
n(training.rdd());
```

5. Numerical Results

In this paper, based upon the support vector machine and logistic regression machine learning algorithms that are widely used with high accuracy, a new machine learning framework for network traffic measurements was designed and developed in two versions based on Java language. The first one was developed using the Weka.jar and LibSVM.jar libraries in the general Windows OS environment, and the second one was developed using the MLlib.jar library in the Apache Spark environment running on Linux OS and Hadoop 2.x version of YARN.

Table 3: Numerical results obtained from SVM and logistic regression models on Weka and Spark

Experimental Environment	ML Model	No. Datasets	Accuracy (%)	Precision	Recall	F1-score
Weka	SVM	1,000	98.80	0.983	0.988	0.984
		10,000	98.40	0.984	0.984	0.983
		125,973	97.60	0.976	0.976	0.976
	ogistic	1,000	100.0	1.000	1.000	1.000
		10,000	99.94	0.999	0.999	0.999

		125,973	97.10	0.971	0.971	0.971
Spark	SVM	1,000	96.61	0.976	0.998	0.987
		10,000	99.11	0.991	0.991	0.991
		125,973	98.24	0.979	0.979	0.979
	ogistic	1,000	100.0	1.000	1.000	1.000
		10,000	98.98	0.990	0.990	0.990
		125,973	99.03	0.990	0.990	0.990

As shown in Table 3, the support vector machine and logistic regression machine learning algorithms using the KDD dataset were analyzed based on Weka and Spark with four representative evaluation indicators: accuracy, precision, recall, and F1-score. As a result, the logistic regression algorithm performed slightly better than the support vector machine algorithm. In addition, when the number of datasets used for machine learning model creation and classification is 10,000 (ML model used is SVM), Apache Spark's processing time is 10 times faster than Weka's. In the case of Logistic, Apache Spark shows a processing speed that is 99 times faster than Weka.

As a result of analyzing the performance of the NTM system proposed in this paper and the (Kulariya, Saraf, Ranjan & Gupta, 2016) paper, the proposed system was superior to their paper with an average of 6.50% higher in terms of accuracy in the case of SVM and a higher average of 5.68% in the case of logistic regression.

6. Conlcusion

In this paper, we designed and developed two versions of a new machine learning framework for network traffic analysis based on support vector machines and logistic regression algorithms for network traffic measurements. These two versions are both written based on the Java language. The first was designed and developed using the Weka.jar and LibSVM.jar libraries in the general Windows OS environment, and the second was designed and developed using the MLlib.jar library in the Apache Spark environment running on YARN of the Linux OS and Hadoop 2.x version.

When analyzing the support vector machine and logistic regression machine learning algorithms with the representative evaluation indicators, accuracy, precision, recall, and F1-score, using the KDD dataset based on Weka and Spark, both showed excellent results, while the logistic regression algorithm was found to perform slightly better. Therefore, we demonstrate that the design and development method of our newly proposed machine learning framework is reliable and practical. In addition, it was confirmed that the newly developed framework in the Apache Spark environment shows a significantly faster processing speed than Weka as the number of data sets used for machine learning model creation and classification increases.

If the machine learning framework-based network traffic measurement tool proposed in this paper is utilized, the computer can recognize abnormal traffic by making predictions based on the similarity of previous traffic and what they have learned for traffic and patterns that are not yet known. In addition, as the computer learns by itself, rather than merely making judgements based on static data, it can flexibly determine traffic referring to previously learned traffic and current traffic. As a result of analyzing the performance of the NTM system proposed in this paper with the (Kulariya, Saraf, Ranjan & Gupta, 2016) paper, it was found to be 6% superior.

As a future research project, it seems necessary to develop and study a method to analyze abnormal traffic by allowing computers to recognize, judge, and apply by themselves through various machine learning methods in the machine learning framework environment designed and developed in this study. It will also be beneficial to conduct further testing using other datasets such as the Canadian Institute for Cybersecurity datasets.

References

Abbasi, A, Shahraki, A & Taherkordi, A. (2021). Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey, *Computer Communications*, *170*, 19-41.

Almomani, O., Almaiah, M. A., Alsaaidah, A., Smadi, S., Mohammad, A. H., & Althunibat, A. (2021, July). Machine learning classifiers for network intrusion detection system: comparative study. In *2021 International Conference on Information Technology (ICIT)* (pp. 440-445).

Alqudah, N., & Yaseen, Q. (2020). Machine Learning for Traffic Analysis: A Review, *Procedia Computer Science*, 170, 911-916.

Barford, P., & Plonka, D. (2001). Characteristics of Network Traffic Flow Anomalies. Proc. *1st ACM SIGCOMM Workshop on Internet Measurement*, San Francisco, California, USA, 69-73.

Bell, J. (2015). *Machine Learning* (Indianapolis, IN: John Wiley & Sons, Inc.).

Casas, P., Vanerio, J. & Fukuda, K. (2017). "GML learning, a generic machine learning model for network measurements analysis," *2017 13th International Conference on Network and Service Management (CNSM)*, Tokyo, Japan, 1-9.

Choudhary, S., & Kesswani, N. (2020). Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT. *Procedia Computer Science*, *167*, 1561-1573.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks, *Machine Learning*, *20*(3), 273-297.

Gitau, J.M., Rodrigues, A.J., & Abuonji, P. (2020). Prototype Intelligent Log-Based Intrusion Detection System, *International Journal of Advanced Networking and Applications*, 12, 4519-4527.

Gurung, S., Ghose, M. K., & Subedi, A. (2019). Deep learning approach on network intrusion detection system using NSL-KDD dataset. *International Journal of Computer Network and Information Security*, *11*(3), 8-14.

Jeong, H.-D., Ahn, W., Kim, H., & Lee, J.-S.R. (2017). Anomalous Traffic Detection Self-Similarity Analysis in the Environment of ATMSim, *Cryptography, 1*(3), 1-19.

Jeong, H.-D.J., Ryu, M.-U., Ji, M. -J., Cho, Y. -B., Ye, S. -K., & Lee, J.-S.R. (2016). DDoS Attack Analysis Using the Improved ATMSim, *Journal of Internet Computing and Services*, *17*(2), 19-28.

Kang, M., & Choi, E. (2021). *Machine Learning: Concepts, Tools and Data Visualization*, World Scientific.

Kelleher, J.D., Namee, B.M., & D'Arcy, A. (2014). *Fundamentals of Machine Learning for Predictive Data Analysis: Algorithms, Worked Examples, and Case Studies* (Cambridge, MA: The MIT Press).

Khan, K., & Goodridge, W. (2019). A Survey of Network-based Security Attacks, *International Journal of Advanced Networking and Applications, 10*(5), 3981-3989.

Kim, K.-P., & Song, S.-W. (2018). A Study on Prediction of Business Status Based on Machine Learning. *Korea Journal of Artificial Intelligence*, 6(2), 23–27. https://doi.org/10.24225/KJAI.2018.6.2.23.

Kulariya, M., Saraf, P., Ranjan, R., & Gupta, G. P. (2016). Performance analysis of network intrusion detection schemes using Apache Spark. In *2016 International Conference on Communication and Signal Processing (ICCSP)* (pp. 1973-1977).

Lee, J.-S., Ye, S.-K., & Jeong, H.-D. (2014). ATMSim: An Anomaly Teletraffic Detection Measurement Analysis Simulator, *Simulation Modeling Practice and Theory, 49*, 98-109.

Murphy, K.P. (2012). *Machine Learning: A Probabilistic Perspective* (Cambridge, Massachusetts: The MIT Press).

Pakdel, R. (2019). *Cloud-based Machine Learning Architecture for Big Data Analysis*, PhD thesis, National University of Ireland, Cork.

Parihar, V., & Yadav, S. (2022). Comparative Analysis of Different Machine Learning Algorithms to Predict Online Shoppers' Behaviour, *International Journal of Advanced Networking and Applications*, 13(6), 5169-5182.

Pentreath, N. (2015). *Machine Learning with Spark*, (Packt Publishing, London).

Perveen, S., Shahbaz, M., Guergachi, A., & Keshavjee, K. (2016). Performance Analysis of Data Mining Classification Techniques to Predict Diabetes, *Procedia Computer Science*, *82*, 115-121.

Saranya, T., Sridevi, S., Deisy, C., Chung, T. D., & Khan, M. A. (2020). Performance analysis of machine learning algorithms in intrusion detection system: A review. *Procedia Computer Science*, *171*, 1251-1260.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A.A. (2009). A detailed analysis of the KDD CUP 99 data set, *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Ottawa, ON, Canada, 1-6.

Yuan, R., Li, Z., Guan, X. & Li, X. (2010). An SVM-based machine learning method for accurate internet traffic classification. *Information Systems Frontiers,* 12, 149–156.

Witten, I.H., & Frank, E. (2002). Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, *ACM SIGMOD Record, 31*(1), 76–77.

Witten, I.H., Frank, E., & Hall, M.A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd Edition (New York, NY, Morgan Kaufmann).