Data mining tools -a case study for network intrusion detection

Soodeh Hosseini^{1,2} · Saman Rafiee Sardo¹

Received: 7 June 2020 / Revised: 21 August 2020 / Accepted: 16 September 2020 /

Published online: 2 October 2020

© The Author(s) 2020

Abstract

With the growth of data mining and machine learning approaches in recent years, many efforts have been made to generalize these sciences so that researchers from any field can easily utilize these sciences. One of the most important of these efforts is the development of data mining tools that try to hide the complexities from researchers so that they can achieve a professional output with any level of knowledge. This paper is focused on reviewing and comparing data mining and machine learning tools including WEKA, KNIME, Keel, Orange, Azure, IBM SPSS Modeler, R and Scikit-Learn to show what approach each of these methods has taken in the face of the complexities and problems of different scenarios of generalization of data mining and machine learning. In addition, for a more detailed review, this paper examines the challenge of network intrusion detection in two tools, Knime with graphical interface and Scikit-Learn with coding environment.

Keywords Data mining tools · Machine learning algorithms · Intrusion detection · Knime · Scikit-learn

1 Introduction

The growth and penetration of the Internet has led to the production of large amounts of data by companies. In addition, many software and databases have been developed to help companies maintain this data. There has also been a lot of research in recent years to extract

✉ Soodeh Hosseini
so_hosseini@uk.ac.ir

Saman Rafiee Sardo
saman.rafiie.rs@gmail.com

¹ Department of Computer Science, Faculty of Mathematics and Computer, Shahid Bahonar University of Kerman, Kerman, Iran

² Mahani Mathematical Research Center, Shahid Bahonar University of Kerman, Kerman, Iran

useful information from these data. These researches are very valuable for companies and as a result, have led to the growth of data mining and machine learning technologies. For example, the Chinese Electric Company data review, examined in [36], can be analyzed to discover the peak hours of power consumption.

Data mining is an appropriate extraction of hidden predictive information totally stored or captured in massive data centers. Recently, many free and commercial data mining and data analysis tools have been developed for solving problems across fields such as life sciences, financial services, telecom, and insurance [17]. Data mining or Knowledge Discovery from Data (KDD) tools allows us to analyze large datasets to solve decision problems. The data mining tools use historical information to build a model to predict customer's behavior e.g., which customers are likely to respond to a new product. Another example is intrusion detection in local systems or networks by analyzing the activity of system and network and processes them by the data mining algorithm in data mining tools. However, these tools are not all powerful enough and do not support all issues, like the research done by the researcher in [20, 35], which can be out of range of many of these tools. However, the goal of all data mining tools is to provide a simple environment for us. In general, when we choose a data mining tool to use, there are many affective factors. Does it run natively on our computer? Does the KDD tool provide all the methods we use? If not, how extensible is it? Does that extensibility use its own language or another language (e.g. R, Python, and SQL) that is generally accessible from many packages? [28]. In the following, we introduce the tools that have been studied in this paper.

KEEL (Knowledge Extraction based on Evolutionary Learning) is an open source (GPLv3) Java software tool that supports data management and a designer of experiments. This tool pays special attention to the implementation of evolutionary learning and soft computing based techniques for Data Mining problems containing regression, classification, clustering, pattern mining and so on [2].

The open source analytics platform KNIME is a modular environment that allows interactive execution and simple visual assembly of workflows [5]. KNIME like most of data mining tools is a graphical tool contain over than 1000 nodes that connected to each other and perform the data mining algorithm. In KNIME you can do classification, clustering, image processing, using WEKA and many more algorithms with programming languages such as python and R [17, 19].

Weka is the strong tool for data mining and machine learning that includes a perfect collection of preprocessing data tools and machine learning algorithms. Moreover, Weka can apply several learners to data and compare and evaluate their performance in order to choose the best learner for prediction [12].

Orange is a Python-based tool for data mining and machine learning suite, featuring a visual programming front-end for exploratory data analysis. It consists of a set of widgets for data preprocessing, features compute modeling, model comparison, and exploration methods. Components are called widgets and they range from simple data visualization, subset selection and preprocessing, to empirical evaluation of learning algorithms and predictive modeling [24].

Rapid Miner is a free open-source software for machine learning and data mining processes written in Java. RapidMiner has a flexible operator for a different format of input and output. It contains many learning schemes for regression, clustering and classification tasks. The graphical user interface of RapidMiner provides Plot View, Meta Data View, and a Data View in the result perspective when working with results.

The Azure Machine Learning service and its development environment are cloud-based and fully scalable which allow the user easily build an analytic model [7]. Another advantage of Azure is providing a circumstance for users that they can drag and drop analytic modules and data sets onto the experimental environment. In experimental environment users can connect two or more modules to each other, forming a model, editing and saving the model and use the model to learn and predict the new pattern.

IBM SPSS Modeler is one of the data mining software applications from IBM. It is a data mining and text analytics tool for building predictive models [1]. IBM SPSS Modeler has many types of modeling methods taken from artificial intelligence, machine learning, and statistics. The methods available on the Modeling menu allow you to deduce new information from your data and to create predictive models.

R is a statistical language that has a faster rate than code-centric software. One of the usages of programming language R is in data mining and text mining project, especially large-scale projects [33]. The R application includes statistical techniques (including linear and nonlinear modeling, statistical classical tests, time series analysis, classification, clustering, etc.) and graphical capabilities.

Scikit-learn is an open source machine learning software developed by David Cournapeau in 2007. Scikit-learn is released under the new BSD license. Scikit-learn offers a set of clustering and classification algorithms. It is offered as a package in Python language and is dependent on data science packages like Numpy and Pandas [26].

This paper is an extension of previous works on comparing introduced data mining tools. We want to compare a number of popular KDD tools, which are being used by organizations in taking proper business decisions and making optimal use of resources for business development. We present three of the most popular license tools and five open source tools. We provide comprehensive and simple tables for anyone who wants to compare tools with each other in short time in terms of important properties of data mining tools. Also, in the following, as a practical application, we present a case-study in which two of these tools challenged in terms of detecting attacks on computer networks to see how these can solve our problem.

The rest of paper is organized as follows. In Section 2, we explain some related work on comparing data mining tools. In Section 3, we provide examine and compare KDD tools ability to support different algorithms and scenarios, and present the results of this review in detail in several tables. In Section 4, we present a case study and use two of the tools to examine the challenge presented in Section 3. Finally, in Section 5, we conclude the paper.

2 Related works

Nowadays, most of the people and commercial companies which deal with data mining for solving their problem, use some known data mining and machine learning tools and do not program data mining algorithms from the beginning. These tools have several learning algorithms and several other packages for analyze data sets. In this section, we talk about similar related work which concentrates on comparing data mining and machine learning tools.

John et al. [9] presented a paper and talk about 17 data mining tools which very famous at that time. The authors examined the tools in terms of comprehensiveness, project construction steps, cost, and support for various classification, standalone algorithms, and objectives. One of their major problems is the lack of full coverage of data mining and machine learning

algorithms, especially clustering algorithms. Most of those tools do not exist in the market at this time or they simply lost popularity among users like WizWhy from WizSoft.

Goebel et al. [11] presented a survey of data mining tools in 1999. In this survey, they talked about common knowledge discovery tasks, suggestions to solve these tasks, and available tools which equipped with modules that can handle these tasks. The authors have studied 43 tools, which are well-researched in terms of operating system and support for various algorithms, but the tools under review are very weak compared to current tools.

In another survey, Mikut et al. [23] classified data mining tools into nine different types based on variant criteria such as user groups, data mining tasks and methods, data structures, visualization and interaction styles, import and export options for data and models, platforms, and license policies.

Abdulrahman et al. in [3] compared 19 open source data mining tools. The main aim of this paper was to present a comparative study, which prepares the features contained in each data mining tool for the user. The evaluation is carried out by scores provided by experts to produce a subjective judgment of each tool and objective analysis about which features are satisfied by each tool.

The report available on [29], like data miner survey in 2011, shows that decision tree and regression are two most popular data mining algorithm among data miners. This survey also shows that about 76% of analytic professionals use R for solving their data mining problem and most professional analytic select R as a primary tool since 2013. R has the first place in most used data mining tool since 2010.

However, KNIME and IBM SPSS Modeler are two tools which have won the highest satisfaction among users who work with these tools.

In [27], authors worked in three phase. In the first phase, they make a list of data mining tools from other papers which work on data mining tools. In the second phase, they delete out of dated tools from the list and in the last phase, the compared rest of tools with each other. They presented their results in the form of a table so that the reader can make the right decision with the support of each tool.

Alan et al. [2] described the properties of six most used tools for general data mining problem which available today: Rapid Miner, R, Weka, KNIME, Orange, and Scikit-learn. They compared these tools and concluded that there is no single best tool. Each tool has some advantage and disadvantage and Rapid Miner, R, Weka, and KNIME recommended for most of the data mining problems because of their user-friendly environment. In this paper, authors have not provided any exact reasons and documentation to support different algorithms and scenarios, and most of their efforts have been focused on studying different datasets.

Hong [15] introduced a prediction model which considers the recurrent support vector regression model with chaotic artificial bee colony algorithm to enhance the performance of forecasting. Fan et al. [10] presented a support vector regression model hybridized with the differential empirical mode decomposition (DEMD) method and auto regression (AR) to provide electric load forecasting with good accuracy. Hong et al. [16] introduced support vector regression based on a forecasting model with a new algorithm which is named chaotic genetic algorithm (CGA), to enhance the performance of forecasting. Hong et al. [36] hybridized several machine learning methods, such as the support vector regression model, the cuckoo search algorithm, the Tent chaotic mapping function, the out-bound-back mechanism, the VMD method, and the SR mechanism to improve the forecasting accuracy. Li et al. [20] implemented a periodogram estimation method (PEM) with considering the LSSVR model to improve the accuracy prediction. Hong et al. [35] proposed a novel electric load

model of forecasting, which considers quantum computing mechanism with intelligent models such as the support vector regression model.

3 Data mining algorithms and scenarios supported by the tools

In this chapter, the tools of WEKA, KNIME, Keel, Orange, Azure, IBM SPSS Modeler, R and Scikit-Learn have been examined and then, they have been introduced and compared based on a number of basic criteria.

Machine learning algorithms can be divided into four categories [8]:

Supervised learning In supervised learning, a set of samples with their labels is given to the machine and the machine should find a relationship between the samples and their labels. The goal in these algorithms is that the algorithm should be able to reduce the error in future examples. Classification and preference learning are some of its examples. Some examples of supervised learning algorithms include Decision tree, Random Forest, Artificial Neural Network, Support Vector Machine and Bayesian Network.

Unsupervised learning Samples used in unsupervised learning are unlabeled. In these algorithms a cost function and a distance measure are defined that, algorithms must reduce the value of the cost function according to the distance measure. Predicting future inputs, decision making, clustering or grouping, dimensionality reduction, and so on are part of the unsupervised learning subcategories. Some examples of unsupervised learning algorithms include K-means clustering, Markov chain model, Expectation maximization algorithm, Density-based spatial clustering of applications with noise (DBSCAN) and Apriori algorithm.

Semi-supervised learning The samples used in the semi-monitored approach are a combination of labeled and unlabeled samples. This approach requires less data than other methods, such as supervised learning and unsupervised learning, which reduces the cost of resources.

Reinforcement learning In this scenario, the machine is depicted as an agent and the surrounding as its environment. Also, information is not given to the machine in reinforcement learning. But the machine can interact with the environment with some actions and receive information and rewards from it. When, the machine receives a reward, it can learn how to improve itself so that it can receive more rewards in the future by doing some actions.

3.1 Support: Data mining algorithms and scenarios

After introducing some of the data mining tools, we are now reviewing these tools based on performance criteria. Tables 1, 2, 3 and 4 shows the main characteristics of these data mining tools. In Table 5, the supported machine learning algorithms for each data mining tool are summarized. Figure 1 also shows the different parts for creating a machine learning model. In this paper, we will look at how tools support these components.

We recognize four levels of support in these characteristics: none (N), basic support (B), intermediate support (I) and advanced support (A). The notation Yes (Y) is used for supporting, and No (N) is used for no-supporting, if characteristics do not have intermediate levels of support. The (+) specifies that the tools implement the algorithm, apply an external

Table 1 Summary of characteristics of software tools

Software	Open source	language	Run Type		Graphical Interface		Input / output			
			On-line	Off-line	Graph Representation	Data visualization	Data management	ARFF data format	Other data format	Data base connection
KEEL	Y	Java	Y	Y	Y	Y	Y	Y	Y	Y
KNIME	Y	Java	Y	N	Y	A	A	Y	Y	Y
WEKA	Y	Java	Y	N	Y	A	A	Y	Y	Y
ORANGE	Y	C/C++/ Python	N	Y	Y	A	A	N	Y	N
RAPIDMINER	Y	Java	Y	N	N	A	A	Y	Y	Y
AZURE MACHINE	Y	–	Y	Y	Y	Y	A	Y	Y	Y
IBM SPSS MODELER	N	Java	N	Y	Y	Y	Y	N	Y	Y
R	Y	C / Fortran/ R	Y	Y	Y	Y	Y	Y	Y	Y
SCIKIT-LEARN	Y	Python	Y	Y	Y	Y	Y	Y	Y	Y

Table 2 Summary of characteristics software tools: Pre-processing variety

Pre-processing variety					
Software	Discretization	Feature Selection	instance selection	Training Set Select	Missing Value Imputation
KEEL	A	A	A	A	A
KNIME	I	A	B	N	B
WEKA	A/I	I/A	B	N	B
ORANGE	A	I	B	N	B
RAPIDMINER	I	A	B	A	B
AZURE MACHINE	B	A	A	N	A
IBM SPSS MODELER	B	A	A	A	A
R	N	N	N	N	B
SCIKIT-LEARN	A	A	B	A	A

add-on (A) to support it; (S) indicates some degree of support for the method, or do not (–). Since most tools are upgraded in a constant state, the data in Tables should be considered temporarily. However, summarizing their capabilities is important and useful so that interested users can choose the suitable environment to handle their problem.

3.1.1 Pre-processing variety

This part contains discretization [21], feature selection [25], instance selection [34] and missing values imputation [4]. The most of the suites try to offer a good feature selection and discretization set of methods, but they ignore specialized methods of missing values imputation and instance selection. Usually, the contributions included are basic modules of replacing or generating null values and methods for sampling the data sets by random (stratified or not) or by value-dependence.

3.1.2 Learning variety

This is support over main areas of data mining, such as predictive tasks (classification, regression, anomaly/deviation detection), and descriptive tasks (clustering, association rule discovery, sequential pattern discovery) [31]. In addition, we take several novel data mining scenarios such as multiple instance learning (MIL), Semi-Supervised Learning (SSL), and Imbalanced Classification.

3.1.3 Advanced features

This part includes some of the less common criteria incorporated such as post-processing techniques, meta-learning, statistical tests, evolutionary algorithms (EAs), fuzzy learning schemes, multi-classifiers for extending the functionality of the software tool.

4 Case study: Intrusion detection challenge in selected tools

In this section, we challenge one of the introduced tools to review its function. The proposed challenge is network intrusion detection system (NIDS), which we use for NSL-KDD data.

Table 3 Summary of characteristics of software tools: learning variety

Learning Variety									
Software	Classification	Regression	Clustering	Association Rules	Subgroup Discovery	Imbalanced Classification	SSL	MIL	
KEEL	A	A	A	A	A	A	A	I	I
KNIME	A	A	A	A	N	N	N	N	N
WEKA	A	A	A	A	N	N	N	I	I
ORANGE	I	N	I	I	N	N	N	N	N
RAPIDMINER	A	A	A	B	B	A	Y	N	N
AZURE MACHINE	A	A	B	N	A	A	N	N	N
IBM SPSS MODELER	A	A	A	A	N	N	A	N	N
R	A	A	A	B	I	B	I	I	N
SCIKIT-LEARN	A	A	A	B	I	B	N	N	N

Table 4 Summary of characteristics of software tools: advanced features

Advanced Features						
Software	Post-processing	Meta - Learning	Statistical tests	EAs	Fuzzy Learning	Multi-Classifiers
KEEL	B	I	A	A	A	A
KNIME	N	N	I	B	B	N
WEKA	N	I	N/B	B	B	N
ORANGE	N	N	N	N	N	B
RAPIDMINER	N	A	B	I	Y	N
AZURE MACHINE	A	A	B	N	N	A
IBM SPSS MODELER	A	A	A	N	N	A
R	N	N	B	N	N	N
SCIKIT-LEARN	A	Y	N	N	B	B

This dataset requires proper preprocesses that the selected tool should be able to do. Also, the accuracy of the proposed model is of great importance to us. Finally, the tool should be able to give us a proper report. Figure 2 shows the proposed model for intrusion detection. We have implemented our model in two tools, Knime with graphical interface and Scikit-Learn with coding environment.

4.1 Intrusion detection challenge

A NIDS monitors activities in the network and can detect malicious activities. In general, the main goal is to examine network activities and provide a report on whether an attack has occurred [18]. In most networks, data is vital, so a NIDS must have the proper accuracy in detecting attacks. The method for doing this in NIDS is one of two methods:

A) **Anomaly Detection:** Any activity with abnormal behavior is known as an attack. These methods can better detect unknown attacks, but the rate of missing report is low [14].

B) **Misuse Detection:** In these methods, a standard pattern for known attacks is created, and an activity is detected as an attack, if it is similar to one of the stored patterns. These methods can well detect known attacks, but they are unable to detect new attacks. Also, the rate of misplaced report is high [6].

When designing a NIDS, we need to consider the following steps: data collection, preprocessing, intrusion detection and reporting. In this paper, what we expect from a tool is to be able to pre-process our data well, then train the proposed model for intrusion detection and finally provide a proper report for the test data set.

4.2 NSL-KDD dataset

NSL-KDD is a dataset to fix the KDD Cup 99 dataset problem. Some of the KDD Cup 99 problems have been resolved in this dataset but still have some problems [22, 32]. However, it is used as a standard data set. Each sample in the NSL-KDD has 41 features. This dataset contains 125,973 training data from 23 network attack types and a normal state and 18,794 test

Table 5 Data mining algorithms and procedures supported by tools

Name	Keel	KNIME	Weka	Orang	Rapid miner	Azure Machine	IBM SPSS Modeler	R	SCIKIT-LEARN
Data Import									
Textual files (txt, csv)	+	+	+	+	+	+	+	+	+
Specific input format files	+	+	+	+	+	+	+	+	+
Excel/spreadsheet Database table	+	A	–	–	+	+	+	A (xls)	+
Data from an URL	–	+	+	+	+	+	+	A (RODBC)	+
Feature selection	+	+	+	+	+	+	+	+	+
Filters	+	+	+	+	+	+	+	A (FSelecto)	+
Wrappers	+	+	+	+	+	–	+	A (FSelecto)	+
Feature transformation	+	+	+	+	+	+	+	A (RWeka)	+
Discretization	+	+	+	+	+	+	+	A (RWeka)	+
Normalization	+	+	+	+	+	+	+	+	+
PCA	–	+	+	+	+	+	–	+	+
ICA	–	–	–	–	–	–	–	+	+
MDS	–	+	+	+	–	–	–	+	+
SVD	–	–	+	A	+	–	–	+	+
Random projections	–	+	+	–	–	+	+	+	+
Classification									
ID3	+	A (Weka)	+	+	A (Weka)	–	–	–	+
C4.5 / C 5.0	+	–	+	+	A (Weka)	+	+	A (RWeka)	+
CART	+	A (Weka)	+	+	A (Weka)	–	+	A (RWeka)	+
Others	+	+	+	+	+	–	+	+	+
Bayesian networks									
Naive Bayes	–	+	+	+	+	+	+	+	+
Full Bayesian network	–	A (Weka)	+	–	A (Weka)	–	+	–	+
AODE	–	A (Weka)	+	–	A (Weka)	–	–	A	–

Table 5 (continued)

Name	Keel	KNIME	Weka	Orang	Rapid miner	Azure Machine	IBM SPSS Modeler	R	SCIKIT-LEARN
Function based learning	Others								
	ANN	+	A (Weka) +(DMNB WAODE) +(SMO), A (libsvm)	–	A (Weka)	–	+	–	+
Ensemble learning	SVM	+	+	–	+	+	+	A	A
	Other	–	–	+	+	+	+	–	+
	Bagging	A	+	+	+	+	+	A	+
	Adaboost	+	+	+	+	+	+	A	+
Regression	Regression	+	+	+	+	+	+	+	+
	Regression analysis	+	+	+	+	+	+	+	+
Association rules	Association								
	GSP	–	+	–	+	–	+	–	–
Cluster	Apriori	+	+	+	(sparse, attr.- value)	–	+	A	–
	FP-growth	+	+	–	+	–	–	–	–
	Eclat	+	–	–	A	–	+	A	–
	Tertius	–	A	–	+	–	–	A	–
	others	+	+	+	–	–	–	A	–
					(Apriori-SD)				
Data visualization	Clustering								
	k-means	+	+	+	+	+	+	+	+
	BIRCH	–	–	–	–	–	–	A	+
	EM	–	–	–	+	–	–	A	–
	DBSCAN	–	+	–	+	–	–	–	+
	OPTICS	–	+	–	A	–	–	–	+
	SOM	–	A	A	+	–	+	A	–
	fuzzy c-means	–	–	A	–	–	–	A	+
	Visualization								
	histograms	+	+	+	+	+	+	+	+
3D graphs	scatterplots	+	–	+	+	+	+	+	+
	other plots	+	–	+	+	+	+	+	+
	3D graphs	–	S	+	S	+	+	A	+

Table 5 (continued)

Name	Keel	KNIME	Weka	Orang	Rapid miner	Azure Machine	IBM SPSS Modeler	R	SCIKIT-LEARN
Preprocessing									
Discretization	+	+	+	+	+	+	+	A (RWeka)	A
Feature Selection	+	+	+	+	+	+	+	A (FSelecto)	A
instance selection	+	+	+	+	+	+	+	–	+
Training Set Select	+	–	+	–	–	–	+	–	+
Missing Value	+	S	+	+	+	+	+	+	+
Imputation									
Advance feature									
Post-processing	+	–	–	–	+	+	+	–	+
Meta - Learning	+	–	+	–	+	+	+	–	+
Statistical tests	+	+	+	–	+	+	+	+	–
EAs	+	+	–	–	+	–	–	–	–
Fuzzy Learning	+	+	–	–	–	–	–	–	+
Multi-Classifiers	+	–	+	+	–	+	+	–	+

Fig. 1 Different parts of a machine learning model

data that includes 38 attack types and normal mode. In our data, there are new types of attacks to check the compatibility of the model to new attacks. Table 6 provides the complete information of the NSL-KDD dataset.

As shown in Table 6, NSL-KDD has 5 classes including 1 normal and 4 types of attacks such as DoS, Probe, R2L, and U2R.

Denial of Service (DoS) attack: is an attack in which the attacker makes some computing (or memory) resource too busy (respectively, too full) to handle legitimate requests, or denies legitimate users access to a resource/service.

User to Root (U2R) attack: is an attack in which the attacker starts out with access to a normal user account on a system and is able to exploit some vulnerability to gain root access to the system.

Remote to Local (R2L) attack: occurs when an attacker who has the ability to send packets to a machine over a network but who does not have an account on that machine exploits some vulnerability to gain local access as a user on that machine.

Prob attack: is an attack that tries to find knowledge about how the computers are connected to each other by bypassing security controls.

4.3 Selected tools

Our goal is to choose tools that solve NIDS challenges well. Here we try to solve this challenge with two approaches: The first approach is choosing a graphical interface tool that suits those who are not strong in programming and conversely, in the second approach, we choose a tool without a graphical interface to suit those who are strong in programming. For the first approach, we have more choices than the second approach. We are looking for a tool that can pre-process the NSL-KDD data, as well as classification algorithms that can evaluate the

Fig. 2 Proposed model for intrusion detection

Table 6 NSL-KDD Details

NSL-KDD			
Category	Attack Type	Train	Test
Normal	Normal	67,343	9711
	DoS		
	Back	956	359
	Land	18	7
	Neptune	41,214	4657
Probe	Pod	201	41
	Smurf	2646	665
	Teardrop	892	12
	Ipsweep	3599	141
	Nmap	1493	73
	PortswEEP	2931	157
	Satan	3633	735
	R2L		
U2R	ftp_write	8	3
	guess_password	53	1231
	Imap	11	1
	Multihop	7	18
	Phf	4	2
	Spy	2	0
	warezclient	890	0
	warezmaster	20	944
	loadmodule	9	2
	buffer_overflow	30	20
Total	rootkit	10	13
	Perl	3	2
Total		125,973	18,794

model proper performance. The solution presented here for the NIDS challenge is not so complex that the tools analyzed above are not able to solve it. As the aim is more to work on a tool than to find an optimal model for NIDS, so almost all tools are able to implement the model with a slight difference in the solution. Accordingly, we select the KNIME tool with the view that the other tools are also suitable. For the second approach, among the tools reviewed are two R and SCIKIT-LEARN tools with no user interface and both are robust enough to solve the NIDS challenge. We choose SCIKIT-LEARN, which does not mean that R is weak or unable to solve the challenge.

So, we use KNIME and Scikit-Learn tools in this challenge. These tools are powerful enough and giving us two different views, one with the graphical environment and one with the coding environment. Figure 3 shows an overview of work flow in the KNIME tool. This implementation has three parts. The first part is to read data and perform preprocessing on the data to become the proper data. The second part is the training of a model on pre-processed data. The third step is to evaluate the model on the test data. Also, we follow all these steps in the Scikit-Learn tool and execute them all with coding.

4.3.1 Preprocessing

In the first implementation step, we perform the necessary preprocessing on the data to provide the proper data that we want to convert. In this step, we try to remove classes with fewer than 20 iterations because our model does not properly study them. Then, according to Table 6, we convert the existing class into the five main classes of DoS, Probe, R2L, U2R, and Normal. Now we have a 5-class problem and implement the model according to it. According to Fig. 1,

Fig. 3 The KNIME Work Flow

the preprocessing process in the KNIME tool is accomplished with the help of a node of the type “Table Creator” and two “Cell Replacer” and “Row Filter” nodes. In “Table Creator” node, a dictionary is defined to change the unnecessary classes to null, and convert the required classes into a 5-class format. And the “Cell Replacer” node executes the dictionary defined in the previous step on the label column. In the following, “Row Filter” node removes samples that are not labeled.

Also in the Scikit-Learn tool, we first use the Numpy library to find the number of iterations of each class and delete it in a loop of classes with less than 20 iterations. Then, we define a dictionary specifying that each key value is converted to a 5-class format. Figure 4 shows the preprocessing in Scikit-Learn.

4.3.2 Results

After converting the data to the desired shape, we run the Random Forest and Decision Tree algorithms 10-fold. We use accuracy, precision, recall and F1-score criteria to evaluate the algorithms, the formula of which is given below [13].

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)} \quad (1)$$

```
Labels_count = np.unique(Y_train, return_counts=True)
for i in range(len(Labels_count[0])):
    if Labels_count[1][i]>20:
        Y.append(Labels_count[0][i])
### For 5-Classes Format
for i in Y:
    Y_5.append(dic_5_classes[i])
```

Fig. 4 Pre-processing on Scikit-Learn

$$\text{Precision} = \frac{(TP)}{(TP + FP)} \quad (2)$$

$$\text{Recall} = \frac{(TP)}{(TP + FN)} \quad (3)$$

$$\text{F1-score} = 2 * \frac{(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})} \quad (4)$$

Figure 5 shows the model definition and implementation steps in the Scikit-Learn tool. Table 7 demonstrates the results of Random Forest and Decision Tree algorithms for each data class in KNIME and Table 8 shows these results in Scikit-Learn. Table 9 also shows the accuracy of each algorithm in KNIME and Table 10 shows these results in Scikit-Learn. Figure 6 indicates Train and Validation Accuracy for Random Forest and Decision Tree in Scikit-Learn.

The results show that, the accuracy of the Random Forest algorithm in the KNIME tool was better than the Scikit-Learn tool. However, the results for the Decision tree algorithm on the two tools are opposite, which could indicate differences in implementation. These two algorithms are in these two tools. Therefore, it can be concluded that in addition to support of tools for different algorithms, it is possible that the results of those algorithms are different in two tools on the same data set, and researchers should pay attention to this point.

We then perform a statistical test using the STAC tool [30] to evaluate the superiority of each of the algorithms. Figure 7 shows how to choose a suitable statistical test. The results of this test are shown in Table 11, STAC results show that although there are differences between the accuracy of the two algorithms in both tools, but both tools have the same rank in performance on this data set.

For comparing the examined algorithms, we apply a statistical test using STAC. Since the data distribution are not specified, we use a nonparametric test with considering two machine learning algorithms (Random Forest and Decision Tree) thus the number of algorithms are 2 ($k=2$), based on STAC diagram in Fig. 7, Mann Withney U test is chosen.

Table 12 shows the rank of algorithms in STAC. As shown in Table 12, the Random Forest receives the maximum score.

```
cv = StratifiedKFold(n_splits= 5, shuffle=True)
clfDT = tree.DecisionTreeClassifier(min_samples_leaf=3, splitter='random')
clfRF = RandomForestClassifier(n_estimators=35, max_depth=10)
for train_index, test_index in cv.split(X, Y):
    x_train, x_test = X[train_index], X[test_index]
    y_train, y_test = Y[train_index], Y[test_index]
    clfRF.fit(x_train, y_train)
    clfDT.fit(x_train, y_train)
```

Fig. 5 generating and training model in Scikit-Learn

Table 7 Results of Algorithms Implementation in KNIME

Criterion Algorithm	Type	Train			Test		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Random Forest	DDoS	100	100	100	98.8	99.0	98.9
	Normal	99.8	100	99.9	88.1	97.3	92.5
	Probe	99.9	99.5	99.7	83.1	99.9	90.8
	R2L	100	95.7	97.8	0.0	0	0.0
	U2R	100	66.7	80.8	0.0	0	0.0
Decision Tree	DDoS	100	99.9	100	96.3	96.3	97.4
	Normal	99.8	99.8	99.8	88.1	96.1	91.9
	Probe	99.5	99.7	99.6	68.8	99.5	81.3
	R2L	93.6	93.6	93.6	0	0.0	0.0
	U2R	75.0	100	85.7	66.7	20.0	30.8

Table 8 Results of Algorithms Implementation in Scikit-Learn

Criterion Algorithm	Type	Train			Test		
		Precision	Recall	F-measure	Precision	Recall	F-measure
Random Forest	DDoS	100	100	100	99.0	98.0	99.0
	Normal	99.0	100	99.0	87.0	98.0	92.0
	Probe	100	95.0	98.0	83.0	96.0	89.0
	R2L	99.0	85.0	92.0	0.0	0.0	0.0
	U2R	100	53.0	70.0	0.0	0.0	0.0
Decision Tree	DDoS	100	100	100	66.0	99.0	80.0
	Normal	100	100	100	84.0	70.0	77.0
	Probe	99.0	100	99.0	79.0	84.0	81.0
	R2L	95.0	96.0	95.0	0.0	0.0	0.0
	U2R	80.0	67.0	73.0	0.0	0.0	0.0

Table 9 Accuracy and Error Results in KNIME

Set	Train		Test	
	Accuracy	Error	Accuracy	Error
Decision Tree	99.79	0.21	89.64	10.36
Random Forest	99.90	0.10	91.18	8.82

Table 10 Accuracy and Error Results in Scikit-Learn

Set	Train		Test	
	Accuracy	Error	Accuracy	Error
Decision Tree	99.79	0.21	75.40	24.6
Random Forest	99.41	0.59	90.82	9.18

Fig. 6 Train and Validation Accuracy for Random Forest and Decision Tree in Scikit-Learn

5 Conclusions

In this paper, we compared a number of popular KDD tools such as WEKA, KNIME, KEEL, Orange, Azure, IBM SPSS Modeler, and R tools in terms of platforms, features, and algorithms. We presented comprehensive and simple tables to analyze and compare these tools with each other on important properties. Also, we reviewed the challenge of NIDS in the KNIME tool and, with the help of it, we created a model on the NSL-KDD dataset to examine how the KNIME and Scikit-Learn tools works.

Fig. 7 STAC assistant decision tree [30]

Table 11 Comparison of Scikit-Learn and KNIME with STAC tool

Rank	Tools
50,000	Scikit-learn
50,000	KNIME

Given the enormous growth of data in industry and science, the data analysis has become a significant problem. In recent years, tools for data mining and machine learning have grown enormously. In this paper, several data mining tools are considered together to help us to select the appropriate software for extracting the useful data. We have examined them based on support for various algorithms and scenarios, operating systems, open source, and more. The authors believe that all the tools under review have been developed to make it easier to use data mining and machine learning; however, there are differences in how they are implemented and supported, and newer versions can change their superiority; for example, support for video data can be very impressive.

Table 12 Ranking of algorithms

Algorithms	Ranking
Random Forest	8.50
Decision Tree	7.25

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Abdar M (2015) A survey and compare the performance of IBM SPSS modeler and rapid miner software for predicting liver disease by using various data mining algorithms. *J Sci (CSJ)* 36:1–12
2. Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *J Multiple-Valued Logic Soft Comput* 17:255–287
3. Altalhi AH, Luna JM, Vallejo M, Ventura S (2017) Evaluation and comparison of open source software suites for data mining and knowledge discovery. *Wiley Interdisc Rev: Data Mining Knowl Discov* 7(3):e1204
4. Batista GE, Monard MC (2003) An analysis of four missing data treatment methods for supervised learning. *Appl Artif Intell* 17(5–6):519–533
5. Berthold MR, Cebon N, Dill F, Gabriel TR, Köster T, Meinel T, Ohl P, Thiel K, Wiswedel B (2009) KNIME-the Konstanz information miner: version 2.0 and beyond. *AcM SIGKDD Explorations Newslett* 11(1):26–31
6. Casas P, Mazel J, Owezarski P (2012) Unsupervised network intrusion detection systems: detecting the unknown without knowledge. *Comput Commun* 35(7):772–783
7. Chappel, D. (2015) Introduction Azure machine learning: a guide for technical professionals. Chappel & Associates 2015: 1–17
8. Elavarasan D, Vincent DR, Sharma V, Zomaya AY, Srinivasan K (2018) Forecasting yield by integrating agrarian factors and machine learning models: a survey. *Comput Electron Agric* 155:257–282
9. Elder JF, Abbott DW (1998) A comparison of leading data mining tools. *Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 1–68
10. Fan G-F, Peng L-L, Hong W-C, Sun F (2016) Electric load forecasting by the SVR model with differential empirical mode decomposition and auto regression. *Neurocomputing* 173:958–970
11. Goebel M, Gruenwald L (1999) A survey of data mining and knowledge discovery software tools. *ACM SIGKDD Explorations Newslett* 1(1):20–33
12. Graczyk M, Lasota T, Trawiński B (2009) Comparative analysis of premises valuation models using KEEL, RapidMiner, and WEKA. *International conference on computational collective intelligence*: Springer. pp. 800–812
13. Hassan MM, Gumaei A, Alsanad A, Alrubaian M, Fortino G (2020) A hybrid deep learning model for efficient intrusion detection in big data environment. *Inf Sci* 513:386–396
14. Hodo E, Bellekens X, Hamilton A, Tachtatzis C, Atkinson R (2017) Shallow and deep networks intrusion detection system: A taxonomy and survey. *arXiv preprint arXiv:1701.02145*
15. Hong W-C (2011) Electric load forecasting by seasonal recurrent SVR (support vector regression) with chaotic artificial bee colony algorithm. *Energy* 36(9):5568–5578
16. Hong W-C, Dong Y, Zhang WY, Chen L-Y, Panigrahi B (2013) Cyclic electric load forecasting by seasonal SVR with chaotic genetic algorithm. *Int J Electr Power Energy Syst* 44(1):604–614
17. Jovic A, Brkic K, Bogunovic N (2014) An overview of free software tools for general data mining. 2014 37th international Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO): IEEE. pp. 1112–1117
18. Juniper Networks (2015) Juniper Networks - How many Packets per Second per port are needed to achieve Wire-Speed?" <https://www.kb.juniper.net/InfoCenter/index?page=content&id=KB14737>. Accessed 22 Feb 2020

19. Kadaru BB, UmaMaheswararao M (2017) An overview of general data mining tools. *Int Res J Eng Technol* 4(9):930–936
20. Li M-W, Geng J, Hong W-C, Zhang L-D (2019) Periodogram estimation based on LSSVR-CCPSO compensation for forecasting ship motion. *Nonlinear Dyn* 97(4):2579–2594
21. Liu H, Hussain F, Tan CL, Dash M (2002) Discretization: an enabling technique. *Data Min Knowl Disc* 6(4):393–423
22. McHugh J (2000) Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by Lincoln laboratory. *ACM Trans Info System Secur (TISSEC)* 3(4):262–294
23. Mikut R, Reischl M (2011) Data mining tools. *Wiley Interdisc Rev: Data Mining Knowl Discov* 1(5):431–443
24. Naik A, Samant L (2016) Correlation review of classification algorithm using data mining tool: WEKA, Rapidminer, Tanagra, Orange and Knime. *Procedia Comp Sci* 85:662–668
25. Oh I-S, Lee J-S, Moon B-R (2004) Hybrid genetic algorithms for feature selection. *IEEE Trans Pattern Anal Mach Intell* 26(11):1424–1437
26. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V (2011) Scikit-learn: machine learning in Python. *J Mach Learn Res* 12:2825–2830
27. Pop, D., & Iuhasz, G. Overview of machine learning tools and libraries. *Inst. e-Austria Timisoara*.
28. Raut R, Nathe A (2015) Comparative study of commercial data mining tools. *Int J Electron Commun Soft Comput Sci Eng (IJECSCE)* 2015:128–132
29. Rexer K, Gearan P, Allen H (2015) Data science survey. Rexer Analytics. Winchester, Massachusetts, 2015
30. Rodríguez-Fdez I, Canosa A, Mucientes M, Bugarín A (2015) STAC: a web platform for the comparison of algorithms using statistical tests. 2015 IEEE international conference on fuzzy systems (FUZZ-IEEE): IEEE. pp. 1–8
31. Tan, P.-N., Steinbach, M., & Kumar, V (2006) Introduction to data mining, Pearson education. Inc., New Delhi.
32. Tavallaei M, Bagheri E, Lu W, Ghorbani AA (2009) A detailed analysis of the KDD CUP 99 data set. *IEEE symposium on computational intelligence for security and defense applications: IEEE*. p. 1–6
33. Team RC (2000) R language definition. R foundation for statistical computing, Vienna
34. Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. *Mach Learn* 38(3):257–286
35. Zhang Z, Hong W-C (2019) Electric load forecasting by complete ensemble empirical mode decomposition adaptive noise and support vector regression with quantum-based dragonfly algorithm. *Nonlinear Dynamics* 98(2):1107–1136
36. Zhang Z, Hong W-C, Li J (2020) Electric load forecasting by hybrid self-recurrent support vector regression model with variational mode decomposition and improved cuckoo search algorithm. *IEEE Access* 8:14642–14658

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.