Journal of Big Data

RESEARCH

Open Access

Intrusion detection model using machine learning algorithm on Big Data environment

Suad Mohammed Othman[1*], Fadl Mutaher Ba-Alwi[1], Nabeel T. Alsohybe[1] and Amal Y. Al-Hashida[2]

*Correspondence:
Suad.m.othman@gmail.com
[1] Faculty of Computer Science and IT, Sana'a University, Sana'a, Yemen
Full list of author information is available at the end of the article

Abstract

Recently, the huge amounts of data and its incremental increase have changed the importance of information security and data analysis systems for Big Data. Intrusion detection system (IDS) is a system that monitors and analyzes data to detect any intrusion in the system or network. High volume, variety and high speed of data generated in the network have made the data analysis process to detect attacks by traditional techniques very difficult. Big Data techniques are used in IDS to deal with Big Data for accurate and efficient data analysis process. This paper introduced Spark-Chi-SVM model for intrusion detection. In this model, we have used ChiSqSelector for feature selection, and built an intrusion detection model by using support vector machine (SVM) classifier on Apache Spark Big Data platform. We used KDD99 to train and test the model. In the experiment, we introduced a comparison between Chi-SVM classifier and Chi-Logistic Regression classifier. The results of the experiment showed that Spark-Chi-SVM model has high performance, reduces the training time and is efficient for Big Data.

Keywords: Intrusion detection, Big Data, Apache Spark, Support vector machine (SVM), ChiSqSelector

Introduction

Big Data is the data that are difficult to store, manage, and analyze using traditional database and software techniques. Big Data includes high volume and velocity, and also variety of data that needs for new techniques to deal with it. Intrusion detection system (IDS) is hardware or software monitor that analyzes data to detect any attack toward a system or a network. Traditional intrusion detection system techniques make the system more complex and less efficient when dealing with Big Data, because its analysis properties process is complex and take a long time. The long time it takes to analyze the data makes the system prone to harms for some period of time before getting any alert [1, 2]. Therefore, using Big Data tools and techniques to analyze and store data in intrusion detection system can reduce computation and training time.

The IDS has three methods for detecting attacks; Signature-based detection, Anomaly-based detection, and Hybrid-based detection. The signature-based detection is designed to detect known attacks by using signatures of those attacks. It is an effective method of detecting known attacks that are preloaded in the IDS database. Therefore, it is often considered to be much more accurate at identifying an intrusion attempt of known attack [3]. However, new types of attack cannot be detected as its signature is not presented; the databases are frequently updated in order to increase their effectiveness of detections [4]. To overcome this problem Anomaly-based detection that compares current user activities against predefined profiles is used to detect abnormal behaviors that might be intrusions. Anomaly-based detection is effective against unknown attacks or zero-day attacks without any updates to the system. However, this method usually has high false positive rates [5, 6]. Hybrid-based detection is a combination of two or more methods of intrusion detection in order to overcome the disadvantages in the single method used and obtain the advantages of two or more methods that are used. Many researches proposed machine learning algorithm for intrusion detection to reduce false positive rates and produce accurate IDS. However, to deal with Big Data, the machine learning traditional techniques take a long time in learning and classifying data. Using Big Data techniques and machine learning for IDS can solve many challenges such as speed and computational time and develop accurate IDS. The objective of this paper is to introduce Spark Big Data techniques that deal with Big Data in IDS in order to reduce computation time and achieve effective classification. For this purpose, we propose an IDS classification method named Spark-Chi-SVM. Firstly, a preprocessing method is used to convert the categorical data to numerical data and then the dataset is standardization for the purpose of improving the classification efficiency. Secondly, ChiSqSelector method is used to reduce dimensionality on the dataset in order to further improve the classification efficiency and reduce of computation time for the following step. Thirdly, SVM is used for the data classification. More specifically, we use SVMWithSGD in order to solve the optimization, in addition, we introduce comparison between SVM classifier and Logistic Regression classifier on Apache Spark Big Data platform based on area under curve (AUROC), Area Under Precision-Recall curve (AUPR) and time metrics. The KDDCUP99 are tested in this study.

The rest of this work is organized as follows: A review of relevant works is conducted in “Related works” section. In “Methods” section, we introduced the proposed method. Also, each step in this method are described. Results and experiment settings are mentioned in “Result and discussion” section. Finally, we conclude our work and describe the future work in “Conclusion” section.

Related works

There are many types of researches introduced for intrusion detection system. With emerge of Big Data, the traditional techniques become more complex to deal with Big Data. Therefore, many researchers intend to use Big Data techniques to produce high speed and accurate intrusion detection system. In this section, we show some researchers that used machine learning Big Data techniques for intrusion detection to deal with Big Data. Ferhat et al. [7] used cluster machine learning technique. The authors used k-Means method in the machine learning libraries on Spark to determine whether the network traffic is an attack or a normal one. In the proposed method, the KDD Cup 1999 is used for training and testing. In this proposed method the authors didn’t use feature selection technique to select the related features. Peng et al. [8] proposed a clustering method for IDS based on Mini Batch K-means combined with principal component analysis (PCA). The principal component analysis method is used to reduce the dimension of the processed dataset and then mini batch K-means++ method is used for data clustering. Full KDDCup1999 dataset has been used to test the proposed model.

Peng et al. [9] used classification machine learning technique. The authors proposed an IDS system based on decision tree over Big Data in Fog Environment. In this proposed method, the researchers introduced preprocessing algorithm to figure the strings in the given dataset and then normalize the data to ensure the quality of the input data so as to improve the efficiency of detection. They used decision tree method for IDS and compared this method with Naïve Bayesian method as well as KNN method. The experimental results on KDDCUP99 dataset showed that this proposed method is effective and precise. Belouch et al. [10] evaluated the performance of SVM, Naïve Bayes, Decision Tree and Random Forest classification algorithms of IDS using Apache Spark. The overall performance comparison is evaluated on UNSW-NB15 dataset in terms of accuracy, training time and prediction time. Also, Manzoor and Morgan [11] proposed real-time intrusion detection system based on SVM and used Apache Storm framework. The authors used libSVM and C-SVM classification for intrusion detection. The proposed approach was trained and evaluated on KDD 99 dataset. In addition, Features selection techniques were used in a lot of researches. PCA Features selection technique implemented in some proposed IDSs like Vimalkumar and Randhika [12] proposed Big Data framework for intrusion detection in smart grid by using various algorithms like a Neural Network, SVM, DT, Naïve Bayes and Random Forest. In this approach, a correlation-based method is used for feature selection and PCA is used for dimensionality reduction. The proposed approach aimed to minimize the time of predicting attack and also to increase the accuracy of the classification task. This approach used Synchrophasor dataset for training and evaluation. The results of this proposed approach are compared by accuracy rate, FPR, Recall and specificity evaluation metrics. Dahiya and Srivastava [13] proposed a framework for fast and accurate detection of intrusion using Spark. In the proposed framework was used Canonical Correlation Analysis (CCA) and Linear Discriminant Analysis (LDA) algorithms for feature reduction, and seven classification algorithms(Naïve Bayes, REP TREE, Random Tree, Random Forest, Random Committee, Bagging and Randomizable Filtered). In the proposed work the two sets of UNSW-NB 15 dataset was used to evaluate the performance of all classifiers. The experiment result of the proposed method found the LDA and random tree algorithm approach is more effective and fast. The Results showed that AUROC = 99.1 for dataset1 and 97.4 for dataset2. In our model, we obtained the results of AUROC = 99.55. Therefore, our model is more effective and fast. Hongbing Wang et al. [14] proposed a parallel principal component analysis (PCA) combined with parallel support vector machine (SVM) algorithm based on the Spark platform (SP-PCA-SVM). PCA is used for analyzing data and feature extract for dimensionality reduction based on Bagging. The proposed approach used KDD99 for training and evaluation.

Natesan et al. [15] proposed optimization algorithm for feature selection. The authors proposed Hadoop based parallel Binary Bat algorithm method for intrusion detection. In this approach, the authors used parallel Binary Bat algorithm for efficient feature selection and optimized detection rate. The MapReduce of Hadoop is used to improve computational complexity and parallel Naïve Bayes provides a cost-effective classification. The proposed approach was trained and evaluated on KDD99 dataset. The proposed approach displayed that the detection rate is improved and the detection time is reduced. Table 1 shows differences between related works based on the Big Data tool that were used for developing the work and the machine learning algorithm that were used as a classifier in the work and the dataset that has been used to train and evaluate.

The researchers are still seeking to find an effective way to detect the intrusions with high performance, high speed and a low of false positive alarms rate. The main objective of this paper is to improve the performance and speed of intrusion detection within Big Data environment. In this method, the researchers used Apache Spark Big Data tools because it is 100 times faster than Hadoop [16], the feature selection that takes the amount of computation time, and this time can be reduced when using SVM on KDD datasets [17]. Therefore, we used SVM algorithm with Chi-squared for feature selection and compared it with Logistic Regression classifier based on area under curve (ROC), Area Under Precision Recall Curve and time metrics.

Methods

Spark Chi SVM proposed model

In this section, the researchers describe the proposed model and the tools and techniques used in the proposed method. Figure 1 shows Spark-Chi-SVM model. The steps of the proposed model can be summarized as follows:

1 Load dataset and export it into Resilient Distributed Datasets (RDD) and DataFrame in Apache Spark.
2 Data preprocessing.
3 Feature selection.
4 Train Spark-Chi-SVM with the training dataset.
5 Test and evaluate the model with the KDD dataset.

Dataset description

The KDD99 data set is used to evaluate the proposed model. The number of instances that are used are equal to 494,021. The KDD99 dataset has 41 attributes and the 'class' attributes which indicates whether a given instance is a normal instance or an attack. Table 2 provides a description of KDD99 dataset attributes with class labels.

Table 1 Related work comparative

Related work	Big Data tool	Algorithm	Dataset
[7]	Apache Spark	K-Means	KDD 99
[8]	Anaconda	K-Means++	KDD 99
[9]	Anaconda	Decision tree	KDD 99
[10]	Apache Spark	SVM, Naïve Bayes, Decision Tree and Random Forest	UNSW-NB 15
[11]	Apache Storm	C-SVM	KDD 99
[12]	Apache Spark	Neural network, SVM, DT, Naïve Bayes and Random forest	Synchrophasor
[13]	Apache Spark	Naïve Bayes, REP TREE, Random Tree, Random Forest, Random Committee, Bagging and Randomizable Filtered	UNSW-NB 15
[14]	Apache Spark	SVM	KDD 99
[15]	Hadoop	Parallel Naïve Bayes	KDD 99

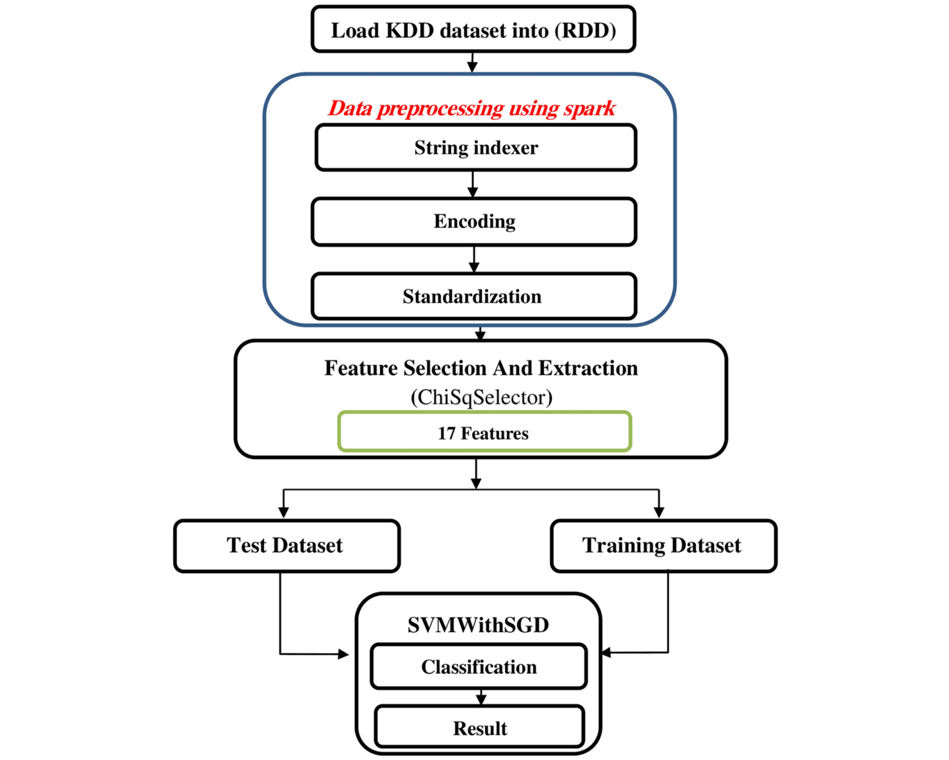

Fig. 1 Spark-Chi-SVM model. The sequence of steps that in Spark-Chi-SVM model

Table 2 KDD99 dataset attributes

No	Attribute name	No	Attribute name
1	Duration	22	Is_guest_login
2	Protocol_type	23	Count
3	Service	24	Serror_rate
4	Src_bytes	25	Rerror_rate
5	Dst_bytes	26	Same_srv_rate
6	Flag	27	Diff_srv_rate
7	Land	28	Srv_count
8	Wrong_fragment	29	Srv_serror_rate
9	Urgent	30	Srv_rerror_rate
10	Hot	31	Srv_diff_host_rate
11	Num_failed_logins	32	Dst_host_count
12	Logged_in	33	Dst_host_srv_count
13	Num_compromised	34	Dst_host_same_srv_rate
14	Root_shell	35	Dst_host_diff_srv_rate
15	Su_attempted	36	Dst_host_same_src_port_rate
16	Num_root	37	Dst_host_srv_diff_host_rate
17	Num_file_creations	38	Dst_host_serror_rate
18	Num_shells	39	Dst_host_srv_serror_rate
19	Num_access_files	40	Dst_host_rerror_rate
20	Num_outbound_cmds	41	Dst_host_srv_rerror_rate
21	Is_hot_login	42	class

Apache Spark

Spark [16] is a fast and general-purpose cluster computing system for large-scale in-memory data processing. Spark has a similar programming model to MapReduce but extends it with a data-sharing abstraction called Resilient Distributed Datasets or RDD [18]. A Spark was designed to be fast for iterative algorithms, support for in-memory storage and efficient fault recovery. Spark Core consists of two APIs which are the unstructured and structured APIs [19]. The unstructured API is RDDs, Accumulators, and Broadcast variables. The structured API consists of DataFrames, Datasets, Spark SQL, and it is the interface that most users should use. In this work the dataframe structure and RDD are used. Dataframe used to load and store the dataset, then it converted to RDD for processing by other process. Spark runs up to 100 times faster than Hadoop in certain environments [18]. Spark can be run with its standalone cluster mode, on Hadoop YARN, or on Apache Mesos or on EC2. In our model we use Spark standalone cluster mode. The main components of Apache Spark are Spark core, SQL, Streaming, MLlib, and GraphX. Figure 2 illustrates Spark on Hadoop ecosystem and it's main components.

Spark uses a master/slave architecture illustrated in Fig. 3. There is a driver that talks to a single coordinator called master that manages workers in which executors run. A Spark cluster has a single master and any number of slaves/workers.

Data preprocessing

Large-scale datasets usually contain noisy, redundant and different types of data which present critical challenges to knowledge discovery and data modeling. Generally, the intrusion detection algorithms deal with one or more of the raw input data types such as SVM algorithm that deals with numerical data only. Hence, we prepare data and convert categorical data in the dataset to numerical data.

Standardization

In machine learning, standardization is a key technique to get reliable results. Values for some features may diverge from small to very big numbers. Hence, analyzed processes may explode the scale [20]. In the Spark-Chi-SVM model we use the standardizes

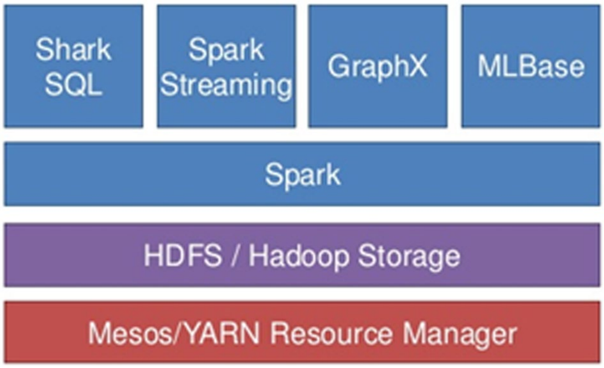

Fig. 2 Spark ecosystem and components. Spark ecosystem on Hadoop and Spark main components

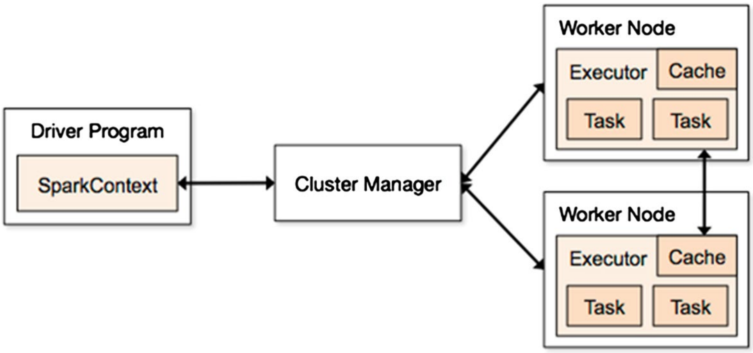

Fig. 3 Spark-architecture-official. Spark master/slave architecture

Table 3 The result of standardization

	The dataset record
The record before standardization	res1:org.apache.spark.mllib.regression.LabeledPoint = (1.0,[0.0,181.0, 5450.0,0.0,0.0,0.0,0.0,0.0,1.0, 0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,8.0,8.0, 0.0,0.0,0.0,0.0,1.0,0.0,0.0,9.0,9.0,1.0,0.0,0.11, 0.0,0.0,0.0,0.0,0.0])
The record after standardization	res2:org.apache.spark.mllib.regression.LabeledPoint = (1.0,[0.0,1.8315794844034117E-4,0.16495156759878019, 0.0,0.0,0.0, 0.0,0.0,2.814168444874875, 0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0, 0.03753270996838475,0.03247770581832668,0.0,0.0,0.0,0.0, 2.576061480099788,0.0,0.0,0.13900605646702138, 0.0848732827397667,2.434387313317322,0.0, 0.22854329046843286,0.0,0.0,0.0,0.0,0.0])

features process by scaling to unit variance in Spark Mllib. The unit variance method used corrected sample standard deviation which the obtained by the formula:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2} \tag{1}$$

Table 3 illustrates the first record in dataset after standardization operation.

Feature selection

Redundant and irrelevant features in the data have caused a problem in network traffic classification to slow down the process of classification and prevent making the accurate classification, especially when dealing with Big Data that have high dimensionality [21]. It is an important issue to determine the optimal feature subset which produce the high accuracy and eliminates diversions [22]. The Spark-Chi-SVM model combines ChiSqSelector and SVM, ChiSqSelector in the model for features selection. It used the Chi-Squared test of independence to decide which features to select. The feature selection that is applied to dataset features in our model is numTopFeatures method. In experiment, we implement different values of numTopFeatures parameter in ChiSqSelector method, the value of numTopFeatures = (40, 33, 30, 20, 19, 17, 15, 12, 11, 10).The numTopFeatures chooses a fixed number of top features according to a Chi-Squared test [16]. The result of this step dataset with 17 features. Table 4 shows some results of different values of numTopFeatures.

Model classifier

Support vector machine (SVM) is a supervised learning method that was introduced by Vapnik [23]. It analyzes data for use in classification and regression. SVM classifies data into different classes by an N-dimensional hyperplane. In the binary classification, SVM classifies the data into two classes by using linearly hyperplane, which is said to be linearly separable if a vector w exists and a scalar b such as:

$$w^T x + b \geq 1 \tag{2}$$

$$w^T x + b \leq 1 \tag{3}$$

where, w is the weight vector and b is a bias value.

SVM works by maximizing the margin to obtain the minimized error classification and best performance with the maximal margin between the vectors of the two classes that are named maximum margin classifier, showing in Fig. 4. The following equation is used to find the optimal separating hyperplane of a linear classification:

$$\min \frac{1}{2}\|w\|^2 \tag{4}$$

Subject to:

$$y_i(w.x_i + b) \geq 1; \quad \forall(x_i, y_i) \in D \tag{5}$$

The soft margin SVM is used to reduce the effects of outliers and misclassification error. The method introduces a non-negative slack variable to Eq. 4. Slack variable is user-defined constant to a tradeoff between the margin and misclassification error.

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \tag{6}$$

Table 4 AUROC result based on numTopFeatures

numTopFeatures	AUROC result (%)
25	99.49
22	99.51
17	99.55
15	99.49
11	92.81

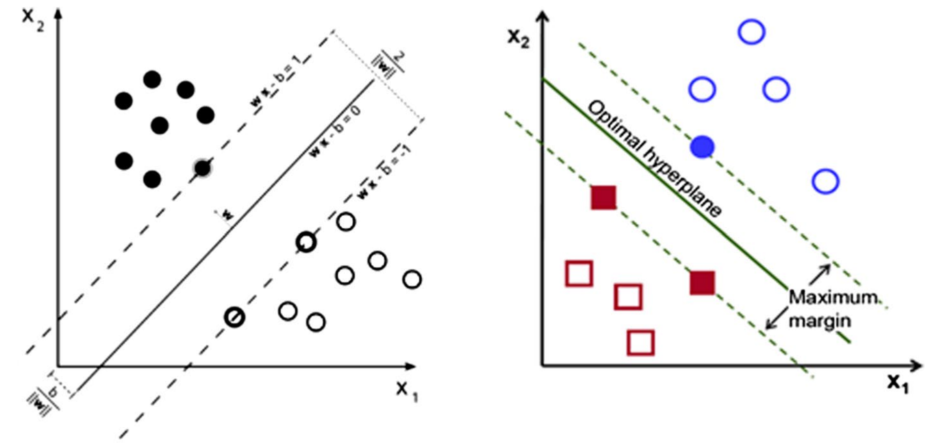

Fig. 4 SVM hyperplane. SVM Hyperplane is separate the data into two classes

Subject to:

$$y_i(w.x_i + b) \geq 1 - \xi_i; \quad \xi_i \geq 0, i = 1.......N \tag{7}$$

where ξ_i is the slack variable and C is a penalty parameter that controls the tradeoff between the cost of misclassification error and the classification margin, and the parameter C controls the tradeoff between the margin and the size of the slack variables [24]. In Spark-Chi-SVM model where the vectors $x_i \in R^d$at training dataset are represente by an RDD of LabeledPoint in MLlib and labels are class indices. The loss function in the SVM model given by the hinge loss:

$$L(w; x, y) := \max\{0, 1 - wy^T x\} \tag{8}$$

In our model, we use SVMWithSGD method. SVMWithSGD is trained with an L^2 regularization with the regularization parameter = 1.0

$$L^2 = \frac{1}{2}\|w\|\begin{matrix}2\\2\end{matrix} \tag{9}$$

High generalization and learning ability of SVM make it suitable for dealing with high dimensionality data, such as Big Data and intrusion detection [25, 26]. However, there are many challenges that need to be taken care about when implementing an IDS such as offering responses in real-time with a high intrusion detection rate and a low false alarm rate. Also, a large number of features and the difficulty to recognize the complex relationship between them make classification a difficult task [26]. SVM is computationally expensive [27]. Therefore, the execution time can be reduced by using Apache Spark, which is a distributed platform to execute many tasks in short time.

Results and discussion

This section shows the results of the Spark-Chi-SVM model that is used for intrusion detection. The proposed model was implemented in Scala programming using the Mllib machine learning library in Apache Spark. Tests were conducted on a personal computer with 2.53 GHZ $CORE^{TM}$ i5 CPU and 4GB of memory under windows7. For the

Table 5 Results of Spark-Chi-SVM model

Classifiers	AUROC (%)	AUPR (%)
SVM only	96.80	94.36
Chi-SVM	99.55	96.24
Logistic regression	92.70	92.77

Table 6 Training and predict time results

Classifiers	Training time	Predict time
SVM only	25.5 s	1.37 s
Chi-SVM	10.79 s	1.21 s
Logistic regression	25.44	1.58

evaluations, the researchers used the KDD dataset, Area under curve(AUROC), Area under Precision-Recall Curve and time measures. The Area under curve is a measure of a classifier's performance [28]. It is calculated by the formula:

$$AUROC = \int_0^1 \frac{TP}{P} d\left(\frac{FP}{N}\right) \tag{10}$$

The Area under Precision-Recall Curve (AUPR) "shows the tradeoff between precision and recall for the different threshold" [29]. It is calculated by the formula:

$$AUPR = \int_0^1 \frac{TP}{TP+FP} d\left(\frac{TP}{P}\right) \tag{11}$$

In "Methods" section we displayed the proposed model steps and Spark Big Data tool which are used in the implemented proposed model to make the model efficient for Big Data. In Table 3 we illustrated the result of data standardization process which standardizes features by scaling to unit variance. Table 4 showed the results of the model for some values that are selected to the numTopFeatures method that are used in the Chi-selector technique for features selection. The results of the experiment model illustrated in Table 5 with other methods are implemented to compare the proposed model with other methods. In Table 5 we displayed the result of implementing SVM classifier without Chi-selector technique for features selection and Logistic Regression classifier with Chi-selector technique based on AUROC and AUPR measures. The result of the experiment showed that the model has high performance and reduces the false positive rate. Table 6 showed the results based on training and predicting time. Figure 5 showed results of the proposed model. According to the comparison in Table 7 between Spark-Chi-SVM model and other researcher's methods based on training and predicting time the Chi-SVM is the best classifier.

Conclusion

In this paper, the researchers introduced Spark-Chi-SVM model for intrusion detection that can deal with Big Data. The proposed model used Spark Big Data platform which can process and analyze data with high speed. Big data have a high

Table 7 Training and predict time comparative

The work	Training time (s)	Predict time (s)
Spark-Chi-SVM	10.79	1.21
SVM Classifier in [10]	38.91	0.20
[15]	1467	792
SVM classifier in [30]	530.45	19.02
SVM classifier in [31]	561.044	26.369

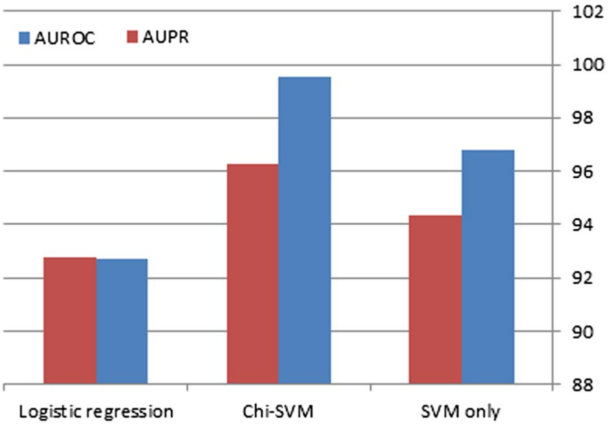

Fig. 5 The results of proposed model. The AUR AND AUPR results of proposed model

dimensionality that makes the classification process more complex and takes a long time. Therefore, in the proposed model, the researchers used ChiSqSelector to select related features and SVMWithSGD to classify data into normal or attack. The results of the experiment showed that the model has high performance and speed. In future work, the researchers can extend the model to a multi-classes model that could detect types of attack.

Authors' contributions

SMO took on the main role performed the literature review, implemented the proposed model, conducted the experiments and wrote manuscript. FB-A took on a supervisory role and oversaw the completion of the work. NTA reviewed the manuscript language and helped in edit the manuscript. AA-H helped in edit the manuscript, All authors read and approved the final manuscript.

Author details

[1] Faculty of Computer Science and IT, Sana'a University, Sana'a, Yemen. [2] University of Modern Science, Sana'a, Yemen.

Acknowledgements

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

All data used in this study are publicly available and accessible in the cited sources. KDD Dataset: including details of Dataset that used in experiment see the web site: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Consent for publication

The authors consent for publication.

Ethics approval and consent to participate

The authors Ethics approval and consent to participate.

Funding

The authors declare that they have no funding.

Publisher's Note

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

Received: 2 June 2018 Accepted: 14 September 2018
Published online: 24 September 2018

References

1. Tchakoucht TA, Ezziyyani M. Building a fast intrusion detection system for high-speed-networks: probe and DoS attacks detection. Procedia Comput Sci. 2018;127:521–30.
2. Zuech R, Khoshgoftaar TM, Wald R. Intrusion detection and big heterogeneous data: a survey. J Big Data. 2015;2:3.
3. Sahasrabuddhe A, et al. Survey on intrusion detection system using data mining techniques. Int Res J Eng Technol. 2017;4(5):1780–4.
4. Dali L, et al. A survey of intrusion detection system. In: 2nd world symposium on web applications and networking (WSWAN). Piscataway: IEEE; 2015. p. 1–6.
5. Scarfone K, Mell P. Guide to intrusion detection and prevention systems (idps). NIST Spec Publ. 2007;2007(800):94.
6. Debar H. An introduction to intrusion-detection systems. In: Proceedings of Connect, 2000. 2000.
7. Ferhat K, Sevcan A. Big Data: controlling fraud by using machine learning libraries on Spark. Int J Appl Math Electron Comput. 2018;6(1):1–5.
8. Peng K, Leung VC, Huang Q. Clustering approach based on mini batch Kmeans for intrusion detection system over Big Data. IEEE Access. 2018.
9. Peng K. et al. Intrusion detection system based on decision tree over Big Data in fog environment. Wireless Commun Mob Comput. 2018. https://doi.org/10.1155/2018/4680867.
10. Belouch M, El Hadaj S, Idhammad M. Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Comput Sci. 2018;127:1–6.
11. Manzoor MA, Morgan Y. Real-time support vector machine based network intrusion detection system using Apache Storm. In: IEEE 7th annual information technology, electronics and mobile communication conference (IEMCON), 2016. Piscataway: IEEE. 2016; p. 1–5.
12. Vimalkumar K, Radhika N. A big data framework for intrusion detection in smart grids using Apache Spark. In: International conference on advances in computing, communications and informatics (ICACCI), 2017. Piscataway: IEEE; 2017. p. 198–204.
13. Dahiya P, Srivastava DK. Network intrusion detection in big dataset using Spark. Procedia Comput Sci. 2018;132:253–62.
14. Wang H, Xiao Y, Long Y. Research of intrusion detection algorithm based on parallel SVM on Spark. In: 7th IEEE International conference on electronics information and emergency communication (ICEIEC), 2017 . Piscataway: IEEE; 2017. p. 153–156.
15. Natesan P, et al. Hadoop based parallel binary bat algorithm for network intrusion detection. Int J Parallel Program. 2017;45(5):1194–213.
16. https://spark.apache.org.
17. Akbar S, Rao TS, Hussain MA. A hybrid scheme based on Big Data analytics using intrusion detection system. Indian J Sci Technol. 2016. https://doi.org/10.17485/ijst/2016/v9i33/97037
18. Zaharia M, et al. Apache spark: a unified engine for big data processing. Commun ACM. 2016;59(11):56–65.
19. Chambers MZaB. Spark: The Definitive Guide: O?Reilly Media, Inc. , 1005 Gravenstein Highway North, Sebastopol, CA 95472. 2017.
20. Kato K, Klyuev V Development of a network intrusion detection system using Apache Hadoop and Spark. In: IEEE conference on dependable and secure computing, 2017 .Piscataway: IEEE. 2017; p. 416–423.
21. Deng Z, et al. Efficient kNN classification algorithm for big data. Neurocomputing. 2016;195:143–8.
22. Sung AH, Mukkamala S. The feature selection and intrusion detection problems. In: ASIAN. Berlin: Springer; 2004. p. 468–482.
23. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97.
24. Cherkassky V, Ma Y. Practical selection of SVM parameters and noise estimation for SVM regression. Neural Netw. 2004;17(1):113–26. https://doi.org/10.1016/S0893-6080(03)00169-2.
25. Karamizadeh S. et al. Advantage and drawback of support vector machine functionality. In: International conference on computer, communications, and control technology (I4CT), 2014. Piscataway: IEEE. 2014; p. 63–65.
26. Enache A-C, Sgârciu V. Enhanced intrusion detection system based on bat algorithm-support vector machine. In: 11th international conference on security and cryptography (SECRYPT), 2014 . Piscataway: IEEE; 2014. p. 1–6.
27. Bhavsar H, Ganatra A. A comparative study of training algorithms for supervised machine learning. Int J Soft Comput Eng (IJSCE). 2012;2(4):2231–307.
28. Bradley AP. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognit. 1997;30(7):1145–59.
29. http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html
30. Gupta GP, Kulariya M. A framework for fast and efficient cyber security network intrusion detection using Apache Spark. Procedia Comput Sci. 2016;93:824–31.
31. Kulariya M. et al. Performance analysis of network intrusion detection schemes using Apache Spark. In: International conference on communication and signal processing (ICCSP), 2016. Piscataway: IEEE; 2016. p. 1973–1977.