Wirtschaftsinformatik

und Maschinelles Lernen

Stiftung Universitat Hildesheim

Marienburger Platz 22

31141 Hildesheim

Prof. Dr. Dr. Lars Schmidt-Thieme

Eya Boumaiza

Master – Seminar Data Analytics I
A geometric framework for UNSUPERVISED ANOMALY DETECTION: Detecting Intrusions in Unlabelled Data

Winter Semester 2019/2020

Ajith Gumudavelly

305643, gumudavelly@uni-hildesheim.de

Abstract

Signature based estimation or mining based methods are quite normal to use by the intrusion detection system which depends upon labeled training data and this training data is quite expensive to produce. The main objective of the authors of this paper is to design a framework for unlabelled data by mapping the data elements in the feature space. They use two feature maps for the representation, the first one is data dependent normalization on network connection and the following one is spectrum kernel on system call traces. By implementing three different algorithms on two data-sets: KDD CUP 1999 dataset[1] for networks and 1999 Lincoln Labs DARPA for system call traces the authors tries to find which algorithm can detect the points present in the sparse region of the feature space. The success of the model is evaluated based on Detection rate and False positive rate.

Table of Contents

1 Introduction

The statistics of usage of the data[3] is exponentially dominating in various fields which have become an important aspect for the business to use. With the growth of data also arises an abundant problem around it. When some systems are developed around it, it would face problems or attacks either technically or manually and that can be solved by taking a few other systems developed. One that system the paper described is the Intrusion Detection System.

Signature based detection is one of the robust detections which is used for intrusion detection systems. These are useful to detect and avoid attack which causes harm to the system. Normally the attacks are manually checked for every attack on it, which is a tedious task to do. That is why there is a lot of work is done in the field of Machine Learning to detect these attacks which are mainly focused on labelled data.

Misuse detection and anomaly detection are the two important models for mining based detection systems. Anomaly detection[4] is of finding the data points which doesn't reside to the pattern expected. It builds a model from normal data and finds out the disturbance in the data from the model, whereas misuse detection[5] works opposite to the anomaly detection for finding computer attacks on labeled data. Here the problem is surrounded by some data and the authors main intention is to find whether the minute part of data belongs to normal or anomaly. This whole system is referred to as Supervised anomaly detection.

Supervised anomaly detection needs a set of data which is normal to train their model but if the data contains some intrusions it can't have the capability to detect further instances of the attacks. We usually don't have the data labeled or normal which makes usage of traditional data mining illogical. There are a lot of problems here in dealing with Supervised anomaly detections and that is why there is a lot of research[6] is going on in Unsupervised Anomaly detection as well.

Unsupervised anomaly detection has many uses comparatively with supervised anomaly detection because it doesn't require a labeled data. It performs even on unlabeled data which is quite useful to systems such as forensic analysis [7].

That is why they perform with Unsupervised anomaly detection and maps the data into feature space from the points R^d. From the points in the feature space, we find the points which are outliers based on their position such that we label the points the sparse region in the feature space irrespective of the algorithm. Here the algorithms can find anomalies because they are far from other points.

We use two types of data which are network connection records and system call traces for mapping the data points into feature space. Once the data is mapped, we can apply a different algorithm over the same data. Here the authors used three different algorithms to find the outliers in the feature space. The three are cluster based estimation, k-nearest neighbor and support vector machine.

The authors used three unsupervised algorithms over two types of data: first one is KDD CUP data[1] and the second one is 1999 Lincoln Labs DARPA for network connection and system call traces respectively.

2 Related work

Below are the related work done by different researchers on anomaly detection using different algorithms and process.

In [26] the researchers Wei Lu and Issa Traore conducted different experiments on different types of attacks using unsupervised anomaly detection framework to detect network intrusions online with metrices IP weight and an outlier detection algorithm which is based on Gaussian mixture model(GMM).

In [27] Markus Goldstein and Seiichi Uchida used a robust approach by applying nearly 19 different unsupervised anomaly detection algorithm on 10 different datasets from various fields. They worked on the real world tasks which gave fine results.

In [28] author Salima Benqdara approached a new system called AIDS (Anomaly-based Intrusion Detection Systems) based on both supervised an unsupervised anomaly detection which can detect attacks effectively with a low false positive rate.

3 Methodology

As discussed above even though many researchers conducted different experiments on unsupervised anomaly detection, the approach, the dataset are entirely different which gave our authors to do more using the different algorithms on the particular dataset which produced good results.

The authors here showed results in terms of a ROC curve which gave a clear view on the model.The fundamental problem is to find the points present in the sparse region of the feature space and find which algorithm us suitable. They also proved with a good accuracy.

Below are the experiments done by the authors to find the anomalies.

3.1 Unsupervised Anomaly Detection

The usage of Unsupervised over Supervised gives good results. Normally, it is quite hard to detect intrusions in the data, whereas Unsupervised can detect the intrusions over the labeled or unlabeled data moreover, the data shouldn't be labeled it works on both.

Two assumptions are made by Unsupervised anomaly detection. The first one is the number of normal instances are numerous than the number of anomalies, the latter one is anomalies are numerically different from normal instances. Though it is quite good to detect anomalies over unlabeled data, it has a certain type of limitless which is for example who has bad intention on a network that he/she is handling it. The model finds problems in the network but has vulnerability when coming to intentions.

Previously, the work of Unsupervised anomaly detection is, it builds a model on the training data and uses it to find the new data network is anomaly or not. The Unsupervised algorithm here is also used to find to detect anomalies and certain Machine Learning technique are also used.

Similar to Unsupervised anomaly detection, there is a lot of work done on distance-based outliers[8, 9, 10]. The difference between them is the nature of outliers.

Usually similar intrusions take place many times which means the number of same instances are many and those are less comparatively with normal instances.

3.2 Data Set Descriptions

The authors used two types of data. First one is for Network connection which is called KDD Cup 1999 Data[1] and for the second one System call we use BSM (Basic Security Module) of 1999 DARPA Intrusion Detection Evaluation which is created by MIT Lincoln Labs[11].

KDD Cup data contains a total of 4,900,000 data instances which obtained during simulated intrusions. This connection is by a sequence of TCP packets and also some from IP addresses. The TCP was built together by Bro program [12] modified with MADAM/ID[13] in which all records are collected. The attacks which are simulated comes under four types. They are probing, Denial of Service, R2L- Unauthorized access from a remote machine and U2R- Unauthorized access to superuser or root functions which are a total number of 24 attacks. The data consists of TCP connections under the type of transferred number of bytes, duration, protocol type, error status and some other were extracted through some domain knowledge. The login attempts which are failed are also included. In count there are 41 features in total.

This main aim here is to give the labeled methods which use labeled data for the model. That is why the KDD training data outnumbers the data we are taking for the model to compute. To make the data more realistic here, we filter many number of attacks such that obtained data consists of 1 to 1.5% of attack and 98.5% to 9% normal instances.

The second one, System call data has BSM (Basic Security Module) data of 5 weeks which runs on Solaris machine. The authors examined the three weeks of traces which were attacked during the three week time and they come under eject and ps. The important thing here to consider is we also take neighbouring process also because the malicious can happen to correspond on as they bound together.

Below is the list of a number of system calls and dataset of system call traces.

Program Name	Total # of Attacks	# Intrusion Traces	# Intrusion System Calls	# Normal Traces	# Normal System Calls	% Intrusion Traces
ps	3	21	996	208	35092	2.7%
eject	3	6	726	7	1278	36.3%

Figure 1: Lincoln Labs Data Summary [2]

3.3 Experimental Setup

For the data to be experimented the authors split the data into two portions. One is testing and the other one is training for deriving the results out of it. The parameters are defined according to the testing set and for every method on every datasets they found a threshold value from that they calculated false positive rate and detection rate with a ROC curve to compare.

For the first algorithm which is cluster based estimation, while performing network connection data they set 40 as fixed-width in feature space and for eject and ps it is 5 and 10 respectively.

For the second algorithm K nearest, while performing on KDD cup data, the value of k is 10,000 and for eject and ps the k value is 2 and 15 respectively.

For the third algorithm SVM, while performing on KDD cup data, the value of v is .01 and σ^2 is 12. while performing on system call data the value of v is .05 and σ^2 is 1.

3.4 A Geometric Framework for Unsupervised Anomaly Detection.

3.4.1 Feature Spaces

Once the data is collected from the audit stream, it is split into data elements $x_{1 \ldots . x_i}$ without loosing the generality. These data elements are defined as input (instance) of space *X*. The elements of input space is mapped in a feature space *Y*, but usually feature space is huge in dimension $d_{|} R^d{}_{|.}$

The feature map is defined as a function which takes input element in the space of input and maps in the feature space to a point. The equation of the feature map is defined as

$$\emptyset : X \rightarrow Y$$

The data element of image x is denoted to feature space point $Ø(x)$. The dot product of any two points in feature space is defined as $<y_1, y_2>$ because it is a Hilbert space. In the feature space, the norm of a point y is ||y|| which is defined as

$$||y|| = \sqrt{<y_1, y_2>}$$

The distance between two elements y_1 and y_2 in the feature space is

$$||y1 - y2|| = \sqrt{<y_1-y_2, y_1-y_2>}$$

$$= \sqrt{<y_1,y_2> -2 <y_1,y_2> + <y_2,y_2>}$$

The distance between two elements x_1 and x_2 in the input space is defined by $d_Ø(x_1, x_2)$ and the equation is

$$d_Ø(x_1, x_2) = || Ø(x_1) - Ø(x_2) ||$$

$$= \sqrt{<Ø(x_1), Ø(x_2)> -2 < Ø(x_1), Ø(x_2) > + <Ø(x_2) - Ø(x_2)>}$$

R^d is similar to Euclidean distance in feature space.

3.4.2 Kernel Functions

As the feature space is huge in dimension it is a tedious task to map data instance to a point in it. That is why we do dot products, if we can do dot products of two data elements of images it's not necessary to map the data elements to their images.

So we can use kernel functions here in the feature space to calculate the dot products. It is calculated as

$$K_Ø(x_1,x_2) = < Ø(x_1), Ø(x_2) >$$

And after redefining the distance measure [29] the equation is

$$d_Ø (x_1, x_2) = \sqrt{K_Ø (x_1, x_1) – 2K_Ø (x_1, x_2) + K_Ø (x_2, x_2)}$$

Kernel function is so efficient in calculating without mapping the elements from input space to their images. A kernel which maps implicitly is radial basis kernel and it is defined as

$$K_{rb}(x_1, x_2) = e^{-\frac{\|z_1 - z_2\|^2}{\sigma^2}}$$

Kernel functions are so useful that it can weigh many features from low or high which depends on domain knowledge.

3.4.3 Convolution Kernels

The kernels we use here are called convolution kernels[14, 15] and in this, we can define them directly with no need of converting the data into the vector in R^n in the first place. Kernels are capable of handling different types of data from numerical to sequences of system calls and event logs in a consistent framework but using different kernels and same defined algorithms under kernels.

3.5 Detecting outliers in Feature Space

Once the data points are in the feature space, we have to formalize the unsupervised anomaly detection problem. Its main idea is to find the points that are far away from other points which is similar to outlier detection.

Here the authors presented three different algorithms to detect anomalies and everything in the algorithm is computed through dot product and the work of algorithm is to detect points which lie in the sparse region and every algorithm performs uniquely.

The first algorithm is cluster based estimation [6], for every points, it approximates the density with respect to radius *w* across the point near the give point by calculating the number of points within the radius. There are two types of points here: normal and anomalies. Normal points are considered to be in the dense region and Anomalies are considered which lies in sparse region. Here they performed fixed-width clustering by a fixed width over the data points with radius *w* then they sort the clusters on size-based and such points which are small clusters are anomalies.

The second algorithm is k-nearest which detects anomalies based on calculating the k-nearest of each point. Anomalies are considered when the sum of distances to the k nearest neighbor is higher than threshold.

The third algorithm is SVM is used to find low support regions of probability distribution by calculating the convex optimization problem[17]. Here it works in a hierarchy way. The points are mapped into another feature space from the initial feature space using the Gaussian Kernel. Here the hyperplane is drawn to divide the data points from the region such that the remaining points are anomalies.

This section will describe how the authors conducted on two different types of data by applying three different algorithms.

4 Experiments

4.1 Algorithms

In the paper, the authors performed different algorithms and noted the results. To compare, they showed results in the form of ROC curve by considering both the detection rate and false positive rate.

4.1.1 First Algorithm - Cluster based estimation

The main goal of this algorithm is to calculate the number of points that are closer to each point in the feature space. Considered two points *x1* and *x2* the distance between them should be less than or equal to *w* (radius) and for every point *x*, the whole calculation is subject to *N(x)*, which is the number of points within the width *w* of that point *x*.

$$N(x) = |\{s|d(x,s) \leq w\}|$$

The complexity of the points is $O(n^2)$ for the points considered, the amount of width is calculated with the fixed width clustering, later the points of same clusters comes under anomalies.

The fixed width clustering is calculated as, every initial point is the centre of the cluster and succeeding point is, if the point is with in the cluster, then it is added to that cluster else is the midpoint of a new cluster. Some points might go through the addition of multiple clusters and the complexity is *O(cn)*, where c is the number of clusters and n is the number of data points.

For the dense region, the points might overlap so the accurate will be reduced. As we don't perform pairwise operations this algorithm is efficient in performing on larger data sets.

4.1.2 Second Algorithm - K means clustering

The main aim of this algorithm is to calculate whether the point is present in the sparse region or not by calculating the sum of the distances to k-nearest of that point. The KNN score[18] is calculated through KNN distance. As the points are bounded together in the dense region they will have a minute score and KNN score is useful for finding the attacks if and only if when the size of k is greater than the frequency of an attack in the data.

But the problem is, the KNN score is too much exorbitant to calculate as the complexity stands with $O(n^2)$. But to use the same algorithm to calculate we can use a technique called Canopy Clustering[19] without losing the result. It is used to speed up the algorithm over huge datasets while using the alternate algorithm doesn't make any sense of getting the output we use this method to find the accurate result.

Now we cluster the data using fixed-width clustering just like in the first algorithm but with a minute difference. Here each element is placed only in one cluster. Now the data is clustered with width w, hence we can use the following sources to calculate the KNN.

Here $c(x)$ is defined as a point, where c is the centre of the cluster and the x is the point. And the distance between the centre of the cluster and point is $d(x, c)$. For example, if two points x_1 and x_2 lies in the same cluster then the distance is defined as

$$d_{\emptyset}(x_1,x_2) \leq 2w$$

And for all the cases

$$d_{\emptyset}(x_1,x_2) \leq d_{\emptyset}(x_1,c(x_2)) + w$$

$$d_{\emptyset}(x_1,x_2) \leq d_{\emptyset}(x_1,c(x_2)) - w$$

The above three inequalities are used by the algorithm to find the K-nearest of a point x.

Here C is the set of clusters which contains the data of clusters and points which are possibly in k-nearest neighbour points are denoted with P and which are actually in k-nearest neighbour are denoted with K. Both K and P are empty initially. Now calculate the distance to each cluster from x. The points are added to P from C when the cluster is near to x and it is called opening. By using the below equation we can get a lower distance from all points in C clusters.

$$d_{min} = \min_{c \in C} d(x, c) - w$$

When $d(x, x_i) < d_{min}$ we can say x_i is closer the point x. Now we have to remove the point from P and add to K, only if P is empty. Now we can open the closer cluster. Whenever we take the cluster from C, the d_{min} increases and when K is with elements k we stop.

This is good when compare to pairwise calculation between the points but we have to choose width w efficiently so that it can divide the clusters accordingly and gives good result though it doesn't effect the knn score.

4.1.3 Third Algorithm - One Class SVM

SVM is a supervised algorithm, but the algorithm we use [20] is an unsupervised algorithm which has unlabelled data points. The work of Supervised SVM is it divides the data points by using a hyperplane[21], but here in Unsupervised, the main goal is to separate the whole data from the origin. We use the quadratic program to do this. Once the optimization is done, considering the hyperplane, the points which are separable from the origin are normal and which are on the different section of hyperplane are anomalous.

The main goal of the algorithm is to find the Hyperplane which separates that data. To have a clear view we define the points are which are in the region are +1 and which are not in the region are -1 based on the hyperplane which is separated from the origin with maximal margin. The best hyperplane is defined according to the optimization problem under a criteria which is described in this paper [21].

Let the origin be p and the normal vector of a hyperplane in the feature space is w and the equation for solving the optimization is

$$\min_{w\in Y, C_i\in R, p\in R} \quad \frac{1}{2}\| w\|^2 + \frac{1}{vl}\sum_i^l L_i - p$$

$$\textit{Subject to:} \quad (w.\Phi(x_i)) \geq p - L_i L_i \geq 0$$

And v stands with $0 < v < 1$ which controls the balance between data in the region which is separated by the hyperplane and the distance from the origin which corresponds to the anomalies ratio expected in the dataset.

Once the optimization problem is solved, the decision point of x is

$$f(x) = sgn((w \,.\, Ø(x)) - p)$$

Writing the whole optimization equation in Lagrangian multipliers a_i

$$\textit{Minimize:} \quad \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j K_\phi(x_i, x_j)$$

$$\textit{Subject to:} \quad 0 \leq \alpha_i \leq \frac{1}{vl}\sum_i \alpha_i = 1$$

For optimum, p can be calculated from Lagrange multipliers for x_i so that a_i satisfies $0 < \alpha_i < \frac{1}{vl}$

$$p = \sum_j \alpha_j K_\phi(x_j, x_i)$$

And the decision function in Lagrange is defined as

$$f(x) = \text{sgn}(\sum_i \alpha_i K_\phi(x_i, x) - p)$$

Computing the decision function is effective because the of the property of optimisation for many number of points a_i will be zero.

4.2 Feature Spaces for Intrusion Detection

To get the necessary information, we use feature space and it can get the information from the model. It's better to choose the needed application and a feature space for optimal performance.

Here in this experiment, we consider two data sets. One is network connection records with 41 features and the latter one is called system call traces and every entry is a sequence during the execution.

For two data-sets we use two different feature maps. First one is data-dependent normalization for records data and the second one is spectrum kernel for system call, as every trace is a sequence we use the kernel and it is called spectrum kernel which is before used for biological sequences [22]. To analyse system call we use short sub-sequences which are basic and used before [23 24 25].

4.3 Data-Dependent Normalization Kernels

As discussed before for connection records, the feature used is called data-dependent normalization feature map. It takes each feature so that it standardizes the distance between feature values in feature space.

In this kernel mapping, we handle both types of data - numerical or discrete attributes. Usually numerical is the number of bytes or the connections in the port. The structure of discrete are types of protocol. Some values of numerical might be discrete yet we handle both but the only problem with straightforward mapping it sometimes the number of one attribute outnumbers the other one which forms huge domination.

That is why we normalize and get the standard deviation from the mean such that the data becomes data dependent because the distance between any two points depends on standard deviation and mean which on the other hand depends on spreading of attribute over the data.

We use dependent data for distinct values. Possible values for distinct attributes i is defined as Σ_i and we use $|\Sigma_i|$ for each discrete value we have and for each attribute we have one coordinate and the attributes are mapped in the feature space under a specific value. The corresponding value has positive value *1/* $|\Sigma_i|$ and rest of corresponding values is zero. *2/* $|\Sigma_i|^2$ is for attribute i of a different value.

4.4 Kernel for Sequences: The Spectrum Kernel

Here we use spectrum kernel because of its results when applied to modelling biological sequences [22]. Over a sequences of input space, the spectrum kernel is defined and it is a long sequence from alphabet Σ. Defining feature space of k-spectrum kernel for $k > 0$ is $|\Sigma|^k$. For any sequence, the specific coordinate is defined as the number of times the neighbouring sub-sequence occurs and these are taken out from sequence by length k of sliding window.

As the dimension of feature space is exponential in k, it is difficult for the feature space to be stored. Taking advantage of feature vectors corresponds to the sequence in higher space, we can calculate kernels between sequences efficiently by a data structure described [26]. At the same time, it is also difficult because it is considered that feature space is nearly equal to 500,000 because of 26 system calls and 4 sub-sequences of length..

4.5 Performance Measures

The authors measure the performance based on two indicators, they are detection rate and the false positive rate. The detection rate is the ratio of number of the intrusion instances detected by the system to a total number of intrusion instances in the test set. The false-positive rate is the ratio of the number of normal instances which are wrongly described as intrusions to the total number of normal instances. These are calculated over labeled data to measure the performance.

By comparing these two methods the authors drew a set of results and the normal data is greater than intrusion data by the ratio of 100 : 1 and the system which thinks the data is normal will have the accuracy of 99%. That is why the authors plot a ROC(Receiver Operating Characteristics) [27] which draws the relationship between false positive rate and detection rate for one fixed training/test set.

5 Discussions and Conclusion

According to my analysis, this is one of the good approaches because of the usage of unsupervised data and also effective algorithms, Though it produced good results I feel that they should have compare the results by using those three algorithm on the supervised data as well which will have a clear view how much their approach is successful. As the individual human might be a concern because of his/her malicious intentions on the system, it is not considered but that might be a important issue as well. Apart from that everything is done by the authors.

The authors exponentially performed well on two datasets with the approach of unsupervised anomaly detection. Both system call and network connection did a great job while finding the anomalies but system call has done little good and the algorithms performed very well on the data whereas in network connection the authors found that there are few number of attacks to detect while few are not because some attacks use feature space just as normal data in the same region. Below is the data how three algorithms are performed and measured according to Detection rate and False positive rate.

Algorithm	Detection rate	False positive rate
Cluster	93%	10%
Cluster	66%	2%
Cluster	47%	1%
Cluster	28%	.5%
K-NN	91%	8%
K-NN	23%	6%
K-NN	11%	4%
K-NN	5%	2%
SVM	98%	10%
SVM	91%	6%
SVM	67%	4%
SVM	5%	3%

Figure 2: Selected points from the ROC curves of the performances of each algorithm over the KDD Cup 1999 Data [2]

This paper represents an unsupervised anomaly detection in a geometric framework. The authors nicely performed three algorithms on two different data-sets for finding the anomalies to discover the points in the sparse region of the feature space.

The authors used two feature maps. First one is data-dependent normalization to network connection records and the other one is spectrum kernel for system call traces. They performed three different algorithms, they are: cluster based approach, k-nearest neighbor and SVM approach. It is allowed that they can use these algorithms on any data.

The authors can detect intrusions on the unlabeled data and it is necessary to apply algorithms over raw collected system data because it is quite expensive and time taking process to do it manually.

Below is the ROC curve denoting the performance of three algorithms on the KDD data

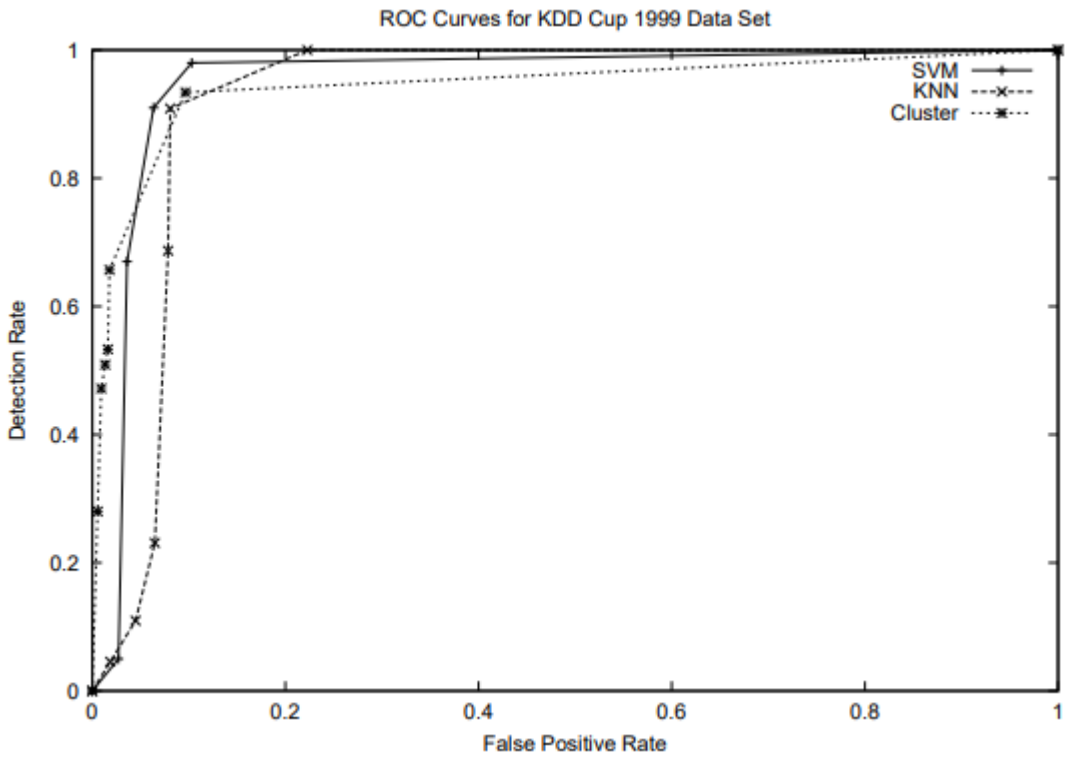

Figure 3: ROC curves showing the performance of the three algorithms over the KDD data set. The curves obtained by varying the threshold [2]

References

[1] The third international knowledge discovery and data mining tools competition dataset KDD99-Cup http://kdd.ics.uci.edu/databases/kddcuo99/kddcup99.html, 1999

[2] Eleazar Eskin, Andrew Arnold, Micheal Prerau, Leonid Portnoy, Sal Stolfo.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled data.

[3] How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone should read. [online] Available From: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#369ed41560ba [Accessed 29 March 2020]

[4] Anomaly Detection. [online] Available From: https://en.wikipedia.org/wiki/Anomaly_detection [Accessed 14 March 2020]

[5] Misuse Detection. [online] Available From: https://en.wikipedia.org/wiki/Misuse_detection [Accessed 14 March 2020]

[6] Leonid Portnoy, Eleazar Eskin and Salvatore J. Stolfo. Intrusion detection with unlabeled data using clustering. In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-), Philadelphia PA 200.

[7] Forensic science. [online] Available From: https://en.wikipedia.org/wiki/Forensic_science [Accessed 17 March 2020]

[8] Markus M. Breunig, Hans-Peter Kriegel, Raymond T, Ng, and Jorg Sander, LOF: identifying density-based local outliers. In ACM SIGMOID int. Conf. on Management of Data, pages 93-104, 2000.

[9] Edwin M.Knorr and Raymond T.Ng. Algorithms for mining distance-based outliers in large datasets. In Proc. Th Int. Conf. Verg Large Data Bases, VLDB, pages 392-403,24-27 1998.

[10] Ediwn M.knorr and Raymond T.Ng, Finding intentional knowledge of distance-based outliers. The VLDB Journal, pages 211-222, 1999.

[11] R.P.Lippmann, R.L.Cunningham, D.J.Fried, I. Graf, K.R.Kendall, S.W. Webster, and M. Zissman, Results of the 1999 darpa off-line intrusion detection evaluation. In Second International Workshop in Recent Advances in Intrusion Detection (RAID 1999), West Lafayette, IN. 1999.

[12] V. Paxson, Bro: A system for detecting network intruders in real-time. In Proceedings of the 7th USENIX Security Symposium, San Antonio, TX, 1998.

[13] W. Lee, S.J.Stolfo and P.K. Chan. Learning patterns from unix processes execution traces for intrusion detection. In AAAI Workshop on AI Approaches to Fraud Detection and Risk Management, pages 50-56. AAAI Press, 1997.

[14] D. Haussler. Convolution kernels on discrete structures. Technical Reports UCS-CRL-99-10, UC Santa Cruz. 1999.

[15] C.Watkins. Dynamic alignment kernels. In A.J.Smola, P.L. Bartlett, B.Scholkopf and D. Schuurmans editors, Advamces in Large Margin Classifiers pages 39-50, Cambridge, MA, 2000, MIT Press.

[16] B. Scholkopf J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, Estimating the support of a high dimensional distribution. Technical Report 99-87, Microsoft Research, 1999. To appear in Neural Computation, 2001.

[17] Building a k-Nearest-Neighbors (k-NN) Model with Scikit-learn. [Online] Available From: https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a [Accessed 17 March 2020]

[18] Canopy Clustering Algorithm. [Online] Available From: https://en.wikipedia.org/wiki/Canopy_clustering_algorithm [Accessed 17 March 2020]

[19] How SVM works. [Online] Available From: https://www.google.com/search?q=how+svm+works&oq=how+SVM+works&aqs=chrome.0.0l8.2535j0j7&sourceid=chrome&ie=UTF-8 [Accessed 17 March 2020]

[20] N. Cristianini and J. Shawe-Taylor. An Introduction to Support Vector Machines. Cambridge University Press, Cambridge, UK, 2000.

[21] Eleazar Eskin Christina Leslie and William Stafford Noble. The spectrum kernel: A string kernel for SVM protein classification. In Proceedings of the Pacific Symposium on Biocomputing (PSB-2002), Kauai, Hawaii, 2002.

[22] Eleazar Eskin, Wenke Lee and Salvatore J, Stolfo. Modeling system calls for intrusion detection with dynamic window sizes. In Proceedings of DARPA Information Survivability Conference and Exposition II (DISCEX II), Anaheim, CA, 2001.

[23] Stephanie Forrest, S. A. Hofmeyr, A. Somayaji, and T.A. Longstaff. A sense of self for unix processes. In 1996 IEEE Symposium on Security and Privacy, pages 120-128. IEEE Computer Society 1996.

[24] W. Lee, S.J. Stolfo and P.K. Chan. Learning patterns from unix processes execution traces for intrusion detection. In AAAI Workshop on AI Approaches to Fraud Detection and Risk Management, pages 50-56. AAAI Press, 1997.

[25] Foster Provost, Tom Fawcett and Ron Kohavi. The case against accuracy estimation for comparing induction algorithms. In Proceedings of the Fifteenth International Conference on Machine Learning, July 1998.

[26] Wei Lu, Issa Traore. A New Unsupervised Anomaly Detection Framework for Detecting Network Attacks in Real-Time, December 2005.

[27] Markus Goldstein, Seiichi Uchida. A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data, April 2016.

[28] Salima Benqdara. Anomaly Intrusion Detection System based on Unlabeled Data, November 2018.

[29] V. Barnett and T. Lewis. Outliers in Statistical Data John Wiley and Sons. 1994.