Classification of KDDCup dataset using SVM Algorithm with Elastic net regularization.

1 Introduction

* 1. Introduction

A network intrusion detection system (NIDS) is crucial for network security because it enables you to detect and respond to malicious traffic. The primary purpose of an intrusion detection system is to ensure IT personnel is notified when an attack network intrusion might be taking place.Intrusion prevention systems control the access to an IT network and protect it from abuse and attack. These systems are designed to monitor intrusion data and take the necessary action to prevent an attack from developing.

Many data mining techniques exist like classification, clustering, association rule. The of data is time consuming but it identifies complex type of IDS data. Association rule technique is based on attribute/ value pair. It identifies multi-feature correlation between attributes of IDS and based on this correlation classification of attack is done. Support vector machine (SVMs) is used for binary classification. SVM prove that SVM is fit for the highly variable and high dimensional data. Data acquired in IDS domain is more complex. So we introduce SVM to the research for intrusion detection system (IDS). SVM is a classifier which can be store the result in two ways: partial or true. Generally, SVM is a simplest linear form.

Support Vector Machine (SVM) Algorithm is specially used for solving the classification problem. The fundamental concept of SVM is to find decision surface that separates the best data vectors into two classes for a given problem which is defined over the vector space SVM is a simplest linear form generally it is a hyperplane that separates the positive examples from the negative example. All vectors lie on one side of the hyperplane labeled as +1 and other vectors lies on another side of the hyperplane labeled as -1. The training objects are lying closest to the hyperplane are called the support vectors.

1.2 Problem statement:

Classification of KDDCup dataset using SVM Algorithm with Elastic net regularization.

1.3 Objective

* The objective of this project is used to classify the KDDCup dataset using SVM with elastic net regularization to optimise the performance of the algorithm
* Enhance the time complexity of the model for KDDCup dataset.

1.4 Scope

• Increase the scalability performance of the algorithm for the large dataset

• Enhances complexity of large dataset

2 Literature survey

* 1. Existing system

1. Evaluation of Different Data Mining Algorithms with KDD CUP 99 Data Set

1. Our work in this paper survey is for the most algorithms Data Mining using KDD CUP 99 data set in the classification of attacks and compared their results which have been reached, and being used of the performance measurement such as, True Positive Rate (TP), False Alarm Rate(FP), Percentage of Successful Prediction (PSP) and training time (TT) to show the results, the reason for this survey is to compare the results and select the best system for detecting intrusion(classification).
2. In this paper, a comprehensive set of algorithms will be evaluated on the KDD dataset being tried to detect attacks on the four attack categories: Probe, DoS, U2R, R2L. These four attacks have distinct unique execution dynamics and signatures, which motivates us to explore if in fact certain, but not all, detection algorithms are likely to demonstrate superior performance for a given attack category.
3. Comparison was made to obtain the accuracy from above algorithms and determine Training Time for each them.
4. Out of all 20 Algorithms implemented, SVM gives Accuracy of 81.38% of predicting algorithm.

2. classification of DDoS Attacks using Enhanced Support Vector machines with Real time Generated Dataset

1. Using the generated DDoS dataset the Enhanced Multi Class Support Vector Machines (EMCSVM). The performance of the EMCSVM is evaluated over SVM with various parameter values and kernel functions. It is inferred that EMCSVM produces better classification rate for the DDoS dataset with ten types of latest DDoS attacks when compared with the kddcup 99 dataset which has six types of DoS attacks.
2. When Compared to KDDCup Dataset and Developed Dataset using ESVM, developed dataset gave better performance in detecting the attack than KDDCup dataset.

3. SVM Intrusion Detection Model Based on Compressed Sampling

1. Compressed sensing is a new data processing theory. Currently, massive data processing is the performance bottleneck of network software and hardware equipment. In the phase of data acquisition, if the dimension of data can be reduced and characteristic information of network data can be directly obtained the efficiency of the detection will be greatly improved [14, 15].
2. SVM intrusion detection technology based on compressed sensing uses the compressed sampling technology of compressed sensing theory to get a small amount of data concerning network behaviour characteristics and then uses the support vector machine (SVM) to establish an intrusion detection model, so as to realize rapid judgment of intrusion behaviour.
3. They arrived at the conclusion that, under the low sampling frequency the detection rate was lower with the increase of sampling frequency the detection rate was increased accordingly.
4. The large Number of network datasets need rapid and real time detections it can be seen that intrusion detection based on compressed sensing has provided a real time network security protection mechanism.
   1. Proposed system

The experiments on KDDCUP99 dataset using SVM and elastic net regularizers. classification is done using SVM classifier. A commonly used model of regression is the Elastic Net regularizes In addition to setting and choosing a lambda value elastic net also allows us to tune the alpha parameter where 𝞪 = 0 corresponds to ridge and 𝞪 = 1 to lasso. Simply put, if you plug in 0 for alpha, the penalty function reduces to the ridge term and if we set alpha to 1 we get the lasso term. Therefore, we can choose an alpha value between 0 and 1 to optimize the elastic net. Effectively this will shrink some coefficients and set some to 0 for sparse selection.

KDDcup dataset contained 42 features, out of which 8 features has been extracted using co-relation matrix.

Eight extracted features from dataset

|  |  |  |
| --- | --- | --- |
| Sl. no | Feature | Details |
| 1. | Count | The total no. of connections in the previous two seconds of the same host |
| 2. | destination bytes | The total no. of bytes of data sent from destination to the source |
| 3. | protocol type | Connection protocol |
| 4. | same srv rate | The total no. of connections of the same service in terms of percentage |
| 5. | Service | Destination service |
| 6. | src bytes | The total no. of bytes of data communicated from source to the destination |
| 7. | srv count | The total no. of connections in the previous two seconds for the same service |
| 8. | Label | Anomaly or normal behaviour |

Using these eight extracted features we train and test the model using SVM classifier with elastic net regularizers and get the accuracy of predicting the attack.

3 Specific Requirements

* 1. Functional Requirements

Introduction: - In this project is used to classify the KDDCup dataset using SVM with elastic net regularization to optimise the performance of the algorithm

Inputs:-In our project we are using Bench Mark Data Set KDDCup data set as input

Processing: - One-Hot-Encoding is used to transform all categorical features into

binary features. The One-Hot-encoding takes a matrix of integers, denoting the values on by categorical features. The output will be a sparse matrix where each column corresponds to one possible value of one feature. Therefore, the features first need to be transformed with Label Encoder, to transform every category to a number. In the second step, we will be applying SVM with elastic regularization. This algorithms analyse the large datasets and mechanism which show the intrusion in the given KDDCup dataset with accuracy.

|  |  |  |
| --- | --- | --- |
| * + - 1. Sl.No | * + - 1. Algorithm Name | * + - 1. Accuracy |
| * + - 1. 1. | * + - 1. SVM with Elastic\_net\_regulaiers | 98.451 % |

3.2 Non-functional requirements

|  |  |
| --- | --- |
| Type | Description |
| Performance | 1.The system should be able to classify anomalies and normal packets with the accuracy of more than 95%.  2. It get accuracy |
| Usability | 1.The system should be available all the time. |

4 System Design

4.1 Architectural Design

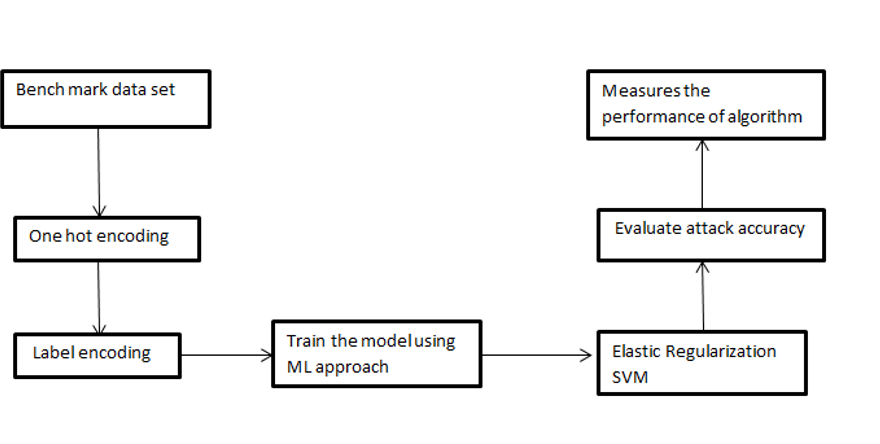


Figure No:1.0

* 1. Data flow diagram

DFD Level-0

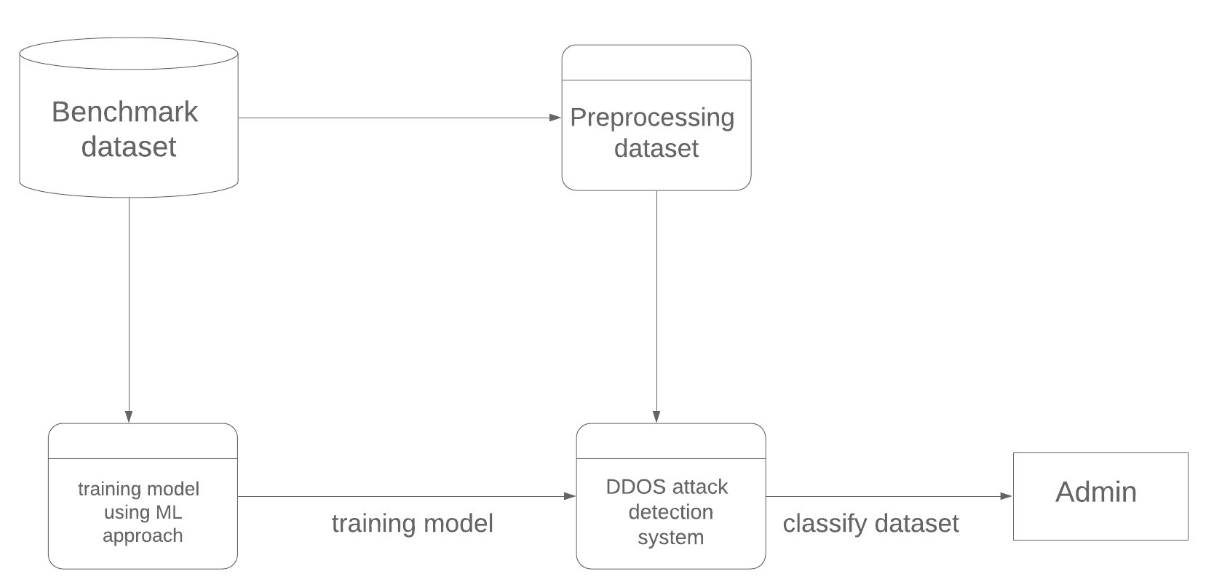


Figure No:1.1

DFD Level-1

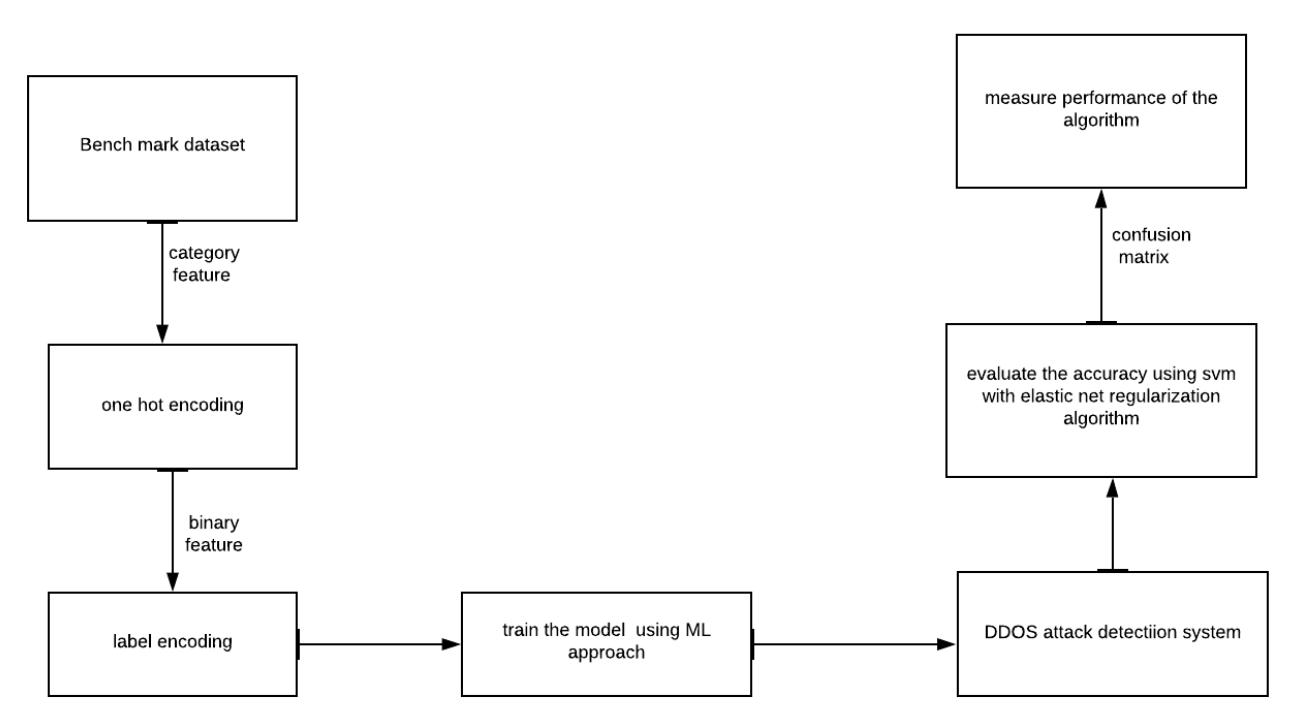


Figure no:1.2

4.3 Use case Diagram

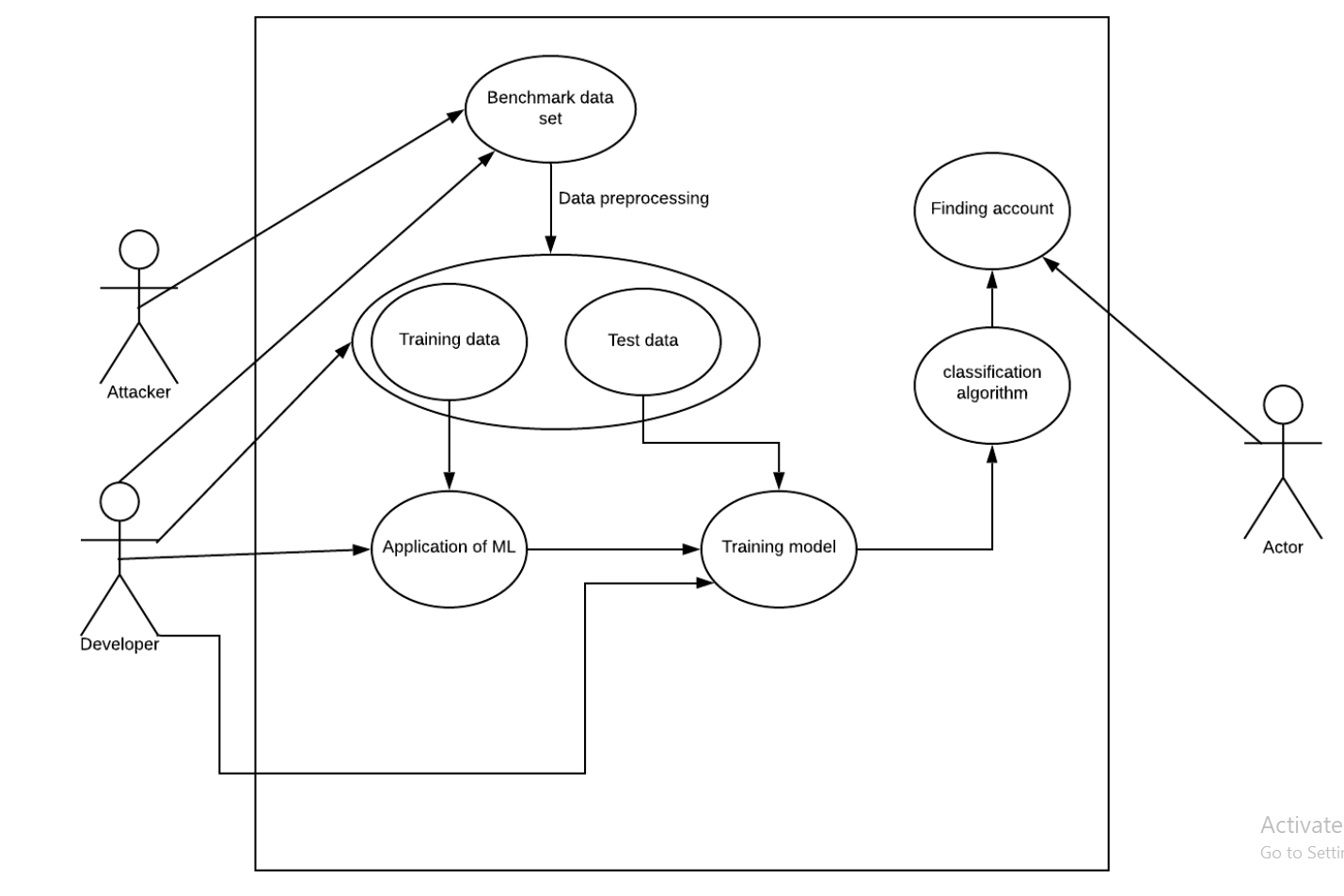


Figure no:1.3

5. Implementation

5.1Introduction

1. Anaconda

Anaconda is a freemium open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. Its package management system is conda. Anaconda distribution comes with more than 1,000 data packages as well as the Conda package and virtual environment manager, called Anaconda Navigators it eliminates the need to learn to install each library independently.

2. Jupyter

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebooks documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context.

3. Python

Python is an interpreted, high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms. It includes object oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation.

5.2 Algorithms used for Implementation

1. Correlation matrix

A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses. Each [random variable](https://www.statisticshowto.com/random-variable/) (Xi) in the table is correlated with each of the other values in the table (Xj).

2. Support vector machine: -

Support Vector Machine (SVM) proposed by is considered as a statistical learning method for classification and regression. Afterward, SVM has been adapted to non-linear problems with using kernel methods.

The kernel function is defined as the following:

k(X, X)'=Φ(X)∙Φ(X') (2)

Φ(x) is defined for solving non-linear classification problem and project the original input data χ to new feature space Η where the classification problem has a linear solution. In our case, we use the Liner kernel function.

f(X)=k(||X−Xc||)

is the center of the vector space. The objective of this learning method is to find a hyper plane separator to classify the input data which can be described mathematically as the following:

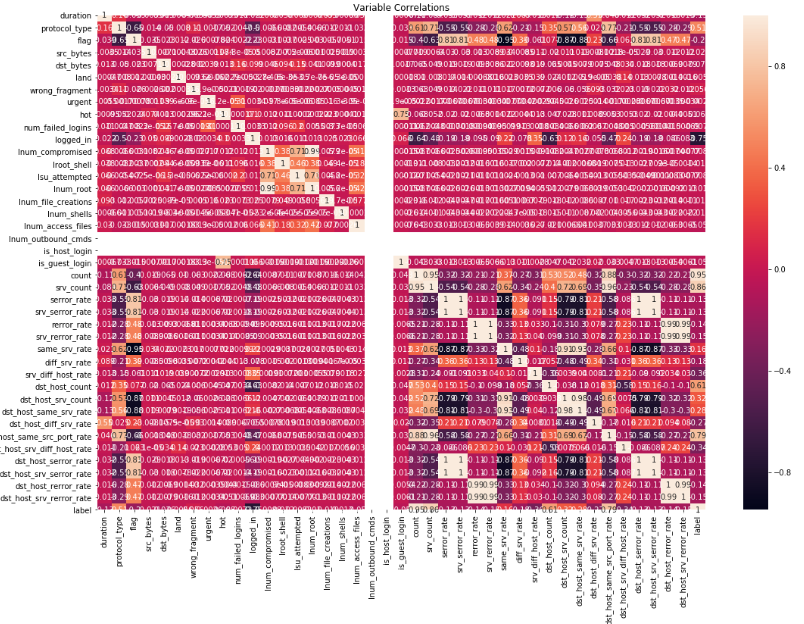
y(X)=sgn(f(X))

Data Set Contents

|  |  |  |
| --- | --- | --- |
|  | Feature Name | Description |
| 1. | Count | No. of connections to the same host as the current connection in the last two seconds |
| 2. | destination bytes | Bytes sent from destination to source |
| 3. | diff srv rate | percentage of connections to different  services |
| 4. | dst host count | count of connections having the same  destination hosts |
| 5. | dst host diff srv rate | percentage of different services on the current host |
| 6. | dst host rerror rate | percentage of connections to the current host that has an RST error |
| 7. | dst host same src port rate | percentage of connections to the current host having the same src port |
| 8. | dst host same srv rate | percentage of connections having the  same destination host and using the same service |
| 9. | dst host serror rate | percentage of connections to the current host that have an S0 error |
| 10. | dst host srv count | count of connections having the same  destination host and using the same service |
| 11. | dst host srv diff host rate | percentage of connections to the same  service coming from different hosts |
| 12. | dst host srv rerror rate | percentage of connections to the current host and specified service that have an RST error |
| 13. | dst host srv serror rate | percentage of connections to the current host and specified service that  have an S0 error |
| 14. | Duration | Duration of the active connection. |
| 15. | Flag | Status flag of the connection |
| 16 | Hot | No. of "hot" indicators |
| 17. | is guest login | One if the login is a "guest.''  login; Otherwise 0 |
| 18. | is host login | One if the login belongs to  the "host''; otherwise 0 |
| 19. | Land | One if the connection is  from/to the  same host/port;  Otherwise 0 |
| 20. | logged in | One if successfully logged  in; otherwise 0 |
| 21. | num access files | No. of operations on  access control files |
| 22. | num compromised | No. of compromised  conditions |
| 23 | num failed logins | No. of failed logins |
| 24. | num file creations | No. of file creation  operations |
| 25. | num outbound cmds | No. of outbound  commands in an ftp  session |
| 26. | num root | No. of "root'' accesses |
| 27. | num shells | No. of shell prompts |
| 28. | protocol type | Connection protocol  (e.g. tcp, udp). |
| 29. | rerror rate | percentage of  connections that have  “REJ'' Errors |
| 30. | root shell | One if the root shell is  obtained; otherwise 0 |
| 31. | same srv rate | percentage of  connections to the same  service |
| 32. | serror rate | percentage of  connections that have  “SYN'' Errors |
| 33. | Service | Destination service (e.g.  telnet, ftp) |
| 34. | src bytes | Bytes sent from source to destination |
| 35. | srv count | No. of connections to the same service as the current connection in the last two seconds |
| 36. | srv diff host rate | percentage of connections to different  hosts |
| 37. | srv rerror rate | percentage of connections that have  “REJ'' errors |
| 38. | srv serror rate | percentage of connections that have  “SYN'' Errors |

6. Result and analysis

6.1 Correlation Matrix



6.2 Snap Shots of Accuracy of the algorithms

SVM algorithm with elastic net\_ regularization



**7.** Conclusion and future work

In this project, the problem was addressed by Network intrusion Detection for finding the best algorithm that classifying the attack more efficiently, using the machine learning model on KDD Cup benchmark dataset. We proposed a Network Intrusion detection system framework comprising three tasks, namely, data pre-processing, machine learning model, and performance evaluation. We used one different architectures to classify the behaviours and found that the SVM algorithm. As future work, the proposed framework can be extended by using variants of deep learning architectures with more robust features to detect intrusion with real time network data.

8. References

* V. Cherkassky and F. Mulier, Learning from Data: Concepts, Theory and Methods, John Wiley and Sons, 1998.
* Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. \Core Vector Machines :Fast SVM Training on Very Large Data Sets". In: Journal of Machine Learning Research 6 (2005).
* T. Joachims. \Making large-scale support vector machine learning practical”. In: Advances in kernel methodsh (1999).
* Alsirhani, S. Sampalli and P. Bodorik, "DDoS Attack Detection System: Utilizing Classification Algorithms with Apache Spark," 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, 2018
* C. Hsieh and T. Chan, "Detection DDoS attacks based on neural-network using Apache Spark," 2016 International Conference on Applied System Innovation (ICASI), Okinawa, 2016
* X. Yuan, C. Li and X. Li, "DeepDefense: Identifying DDoS Attack via Deep Learning," 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, 2017
* B. Zhou, J. Li, J. Wu, S. Guo, Y. Gu and Z. Li, "Machine-Learning-Based Online Distributed Denial-of-Service Attack Detection Using Spark Streaming," 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, 2018
* V. Cherkassky and F. Mulier, Learning from Data: Concepts, Theory and Methods, John Wiley and Sons, 1998.
* Ivor W. Tsang, James T. Kwok, and Pak-Ming Cheung. \Core Vector Machines :Fast SVM Training on Very Large Data Sets". In: Journal of Machine Learning Research 6 (2005).
* T. Joachims. \Making large-scale support vector machine learning practical”. In: Advances in kernel methodsh (1999).