PERFORMANCE EVALUATION OF MACHINE LEARNING ALGORITHMS IN DETECTING KEYLOGGERS

ABSTRACT

The development of more innovative and effective malware defense mechanisms has been regarded as an urgent requirement in the cybersecurity community. This study is focused on predicting the existence of keyloggers in a system using typical Machine Learning (ML) classification techniques. The science of building algorithms that can learn without being led by humans is known as ML. In this context, "learning" refers to the development of algorithms capable of ingesting data, making sense of it within a subject of expertise, and using that data to make autonomous judgments. This paper examines the presence of keyloggers that might pose as cyber hazards in a system utilizing traditional ML Techniques. The accuracy of the results from the seven algorithms was compared to show which algorithm performed better on the prediction, and this work was also further compared to other existing works in the cyber security space to answer the question " Can classical machine learning classification techniques predict the existence of a keylogger better than Deep Learning (DL) counterparts?". The ultimate accuracy of the ML algorithm in this work is 97% (0.97) and is as a consequence of prediction using the Decision Tree algorithm. Thus, the Decision Tree algorithm demonstrated that simple ML classification algorithms will accurately predict the presence of keyloggers in a system better than the DL classification algorithms.

Keywords: Machine Learning, Deep Learning, Cyber Security, Decision Tree, K-Nearest Neighbor (KNN), Logistics Regression, Gaussian Naïve Bayes, Decision Tree, Random Forest, Extreme Gradient Boosting, Light Gradient Boosting Machine, Algorithms, Keyloggers, Prediction.

1 INTRODUCTION

Cyber security is the collection of technologies, methods, and practices that are meant to secure networks, devices, programs, and data against attack, damage, or illegal access. Information technology security is another term for cyber security. Cyber security is critical because governments, military groups, corporations, financial institutions, and medical institutions acquire, analyze, and store massive quantities of data on computers and other devices. A considerable amount of the data may contain sensitive information, such as intellectual property, financial data, personal information, or other sorts of data whose unlawful access or disclosure may have negative effects. In the course of doing business, organizations send sensitive data over networks and to other devices, and cyber security refers to the discipline committed to securing that information and the systems that handle or store it. As early as March 2013, top intelligence officials in the United States warned that cyber-attacks and digital surveillance constitute the most serious danger to national security, surpassing even terrorism. For successful cyber security, a company must coordinate its activities throughout its whole information system, particularly in terms of network security. There are different kinds of threat to the security of the cyber space and this paper focuses solely on ***"keylogging threat."***

A keylogger is a sort of malware that is difficult to detect. Keyloggers are software apps that monitor your activities and provide hackers access to your personal information. Passwords and credit card information, as well as websites visited, are all recorded by watching your keyboard strokes. The

application is installed on your computer and logs each keystroke. The log file is subsequently transferred to a server, where fraudsters are waiting to take advantage of all of this sensitive data. They are not always illegal and can serve useful and beneficial functions. IT workers, for example, routinely use keyloggers to troubleshoot problems and systems. If there is malevolent intent, keylogging turns bad and becomes a menace.

Types of Keyloggers:

Software Keyloggers: These keyloggers monitor the system by collecting keystroke data from the operating system, storing it in any memory location, and sending it to the attacker who inserted the keyloggers. The keylogger software program is made up of two files: a dynamic link library file that records and an executable file that installs and activates the DLL file. Keyloggers are used by IT firms to track out and resolve technical issues with computer networks and business networks. Keyloggers can be used lawfully by families and businesses to track network activity without their users' awareness. Even Microsoft has openly revealed that the current version of its operating system includes a keylogger to help with writing and typing abilities.

Hardware Keyloggers: These are linked to the keyboard and the computer. These keyloggers do not require any software to be installed since they exist on a hardware level in a machine. The hardware keyloggers come in three types.

- Inline devices – these are linked to the cable of the keyboard.
- Devices that can be installed within the regular keyboards.
- Replacement keyboards containing the already installed keylogger.

2 PROBLEM STATEMENT

The internet has grown at an exponential rate during the last decade. There are currently more than five times as many Internet users as in 2000. And, as previously said, the global Internet user base is approaching two billion, and may do so before the end of this year. The Internet has not only grown in size, but it has also expanded out and become more worldwide. The top ten nations accounted for 73% of all Internet users in 2000. In 2010, that figure had dropped to 60%. While the internet has clearly offered new benefits, it has also introduced new concerns as cyber thieves seek to take advantage of our apparently ever-increasing dependency on connection. Phishing emails, virus and ransomware assaults, or having your bank account information, passwords, and other personal information stolen - the internet has offered malevolent hackers with a plethora of new methods to gain money and disturb your life.

Malware is the primary choice of weapon to carry out evil intents in cyberspace, according to (Jang-Jaccard and Nepal, 2014), either by exploiting existing vulnerabilities or utilizing unique properties of developing technology. This increases awareness of the need to develop systems and procedures that can identify malicious intent (keylogging) in near real-time, before it becomes a threat.

3 LITERATURE REVIEW

We analyze prior works in this cyber security arena labeled "keylogger detection" to keep up with the level of work done here, identify new ideas and research problems, and determine how this work may be distinctive in its contribution to the cyber security community. (Pillai and Siddavatam, 2018) had previously worked on detecting keyloggers using a machine learning (Support Vector Regression) approach. Using the I/O hook mechanism, this work developed its own keyboard features by calculating the time differential between the send event process and the get output process. It then fits the obtained dataset to an SVR method and achieves an 83% accuracy performance. This study resulted in a keylogger detection system that could only work on Windows operating system.

(Solairaj et al., 2021) emphasized the existing keylogger detection tools at the time, which omitted ML and AI (AI). It discussed the anti-hook method, which is based on the fact that any process, whether hidden or visible, uses the hooks API for the purpose of hooking. This approach employs hooks to scan all processes, static executables, DLLs (Dynamic Link Libraries), and detect suspicious programs or files. Overall, this strategy captures all of the information about the process or file that use hooks. This paper also investigated the HoneyID mechanism, which is a Spyware detection mechanism capable of detecting unknown spywares. The attacker is enticed by this method. To activate the spyware's operations, bogus events are produced. The hidden spyware among running processes is detected.

(Wazid et al., 2013) created a unique framework for keylogger detection and prevention in a system. To test the suggested system, the group built its own keylogging assault scenario. The proposed system employed two techniques: the honeypot base monitoring, which acts as a keylogger to the keylogger spyware posing a threat and monitors the keylogger's activities, and the prevention side, which analyzes the honeylog file from the honeypot monitor to determine whether or not there is a threat.

TABLE 1: Comparison of different Methodologies.

S/N	PAPER TITLE	AUTHOR(S)	YEAR	CONFERENCE	METHODOLOGY	CONCLUSION
1	A modified framework to detect keyloggers using machine learning algorithm.	Divyadev Pillai. Irfan Siddavatam.	2018	International Journal of Information Technology. Vol 11, No 4.	Support Vector Machine Regression (SVR) Algorithm.	With an accuracy of 83%, the SVM algorithm successfully predicted keylogging actions in a Windows operating system. It acquired data from previous keyloggers' behavioral features.

2	Mobile Keylogger Detection by Using Machine Learning Technique.	S.vinothkumar. S.Arunasankaralingam.	2020	IJEDRCP1403011 International Journal of Engineering Development and Research.	Support Vector Machine Classifier (SVC) Algorithm.	Using behavioral and intent data, the system was able to distinguish between legitimate and unlawful keylogging. It had an 87% accuracy rate.
3	Detection of Mobile Keyloggers Using Deep Learning.	Ch. Madhuri. K. Rama Siritha. S. Sai Ram. G. Rama Koteswara. Rao	2019	International Journal of Innovative Technology and Exploring Engineering (IJITEE). Vol 8, No 8.	Deep Learning's Stochastic Gradient Descent (SGD) Algorithm.	The existence of keyloggers was successfully recognized in the dataset including the characteristics of the APK files. This was done with around 90% accuracy.
4	Enhancing Keylogger Detection Performance of the Dendritic Cell Algorithm by an EnticementStrategy.	Jun Fu. Huan Yang. Yiwen Liang. Chengyu Tan.	2014	Journal of Computers.	Dendritic Cell Algorithm (DCA) to perform multi-sensor data fusion on a set of input signals containing keylogger information.	This study used keystroke simulation to successfully enhance the behaviors (features) of keyloggers. The DCA compares these patterns in order to find the keylogger process as quickly as feasible in order to

						prevent privacy loss.
5	Keylogger Detection using Memory Forensic and Network Monitoring.	Md Bayzid Ahmed. Mohiuddin Shoikot. Jafrul Hossain. Anisur Rahman.	2019	International Journal of Computer Applications.	Memory analysis-based detection method via Memory Forensic and Network Monitoring.	Successfully developed a keylogger mitigation system capable of detecting keylogging attacks in their early stages without requiring advanced technical expertise.
6	Keylogger Detection Using a Decoy Keyboard.	Julian Rrushi. Seth Simms. Margot Maxwell. Sara Johnson.	2017	Lecture Notes in Computer Science.	A decoy keyboard that emulates and exposes keystrokes modeled after actual users through a low-level driver.	Successfully created a counterfeit keyboard with a filter drive capable of shadowing an actual keyboard in order to identify malware corpus. This technology was capable of detecting threats in seconds.

4. METHODOLOGY

We present a technique for detecting keyloggers in a system in real time. This system accomplishes this by training itself across different ML classification algorithms, picking the best fitting algorithm, then evaluating fresh input with that best fit algorithm in real time to provide detection predictions. It is divided into five stages:

- **Phase 1: Data Collection –** The dataset is an open-source dataset that may be downloaded from the University of California, Irvine (UCI) Machine Learning Repository. It is a comprehensive library of datasets, domain theories, and data generators utilized by the machine learning community for empirical evaluation of machine learning algorithms. It is hosted and maintained by the University of California, Irvine's Center for Machine Learning and Intelligent Systems. David Aha developed it while a PhD student at UC Irvine.
- **Phase 2: Data Information** – The dataset is (65139, 86) in size and required to be trained on a GPU using the Google Collaboratory (Google CoLab) infrastructure. There was a total of 182 missing values (NaNs) in the data, which is quite little when compared to the entire size of the dataset, therefore they are not as significant. The dataset contains 85 input features for the learning algorithms and 1 output feature "Class" for the prediction algorithms. The output feature is a binary class feature with two distinct entries: Keylogger (indicating the existence of a keylogger) and Benign (signifying the absence of a keylogger). The distribution of the output feature is 38,287 items for Benign or 0 value and 26,852 entries for Keylogger or 1 value.
- **Phase 3: Data Cleaning and Preparation** – To maintain credibility, NaNs were filled rather than dropped, the target feature "Class" was encoded so that 0 represents Benign and 1 represents Keylogger - these encodings were created as a new feature called "target" while the previous target feature "Class" was dropped, the useless features with low correlation value to the target feature were dropped - this left the data with 72 highly correlating features and the numerical feasibility.
- **Phase 4: Modeling and Training –** The dataset was divided into three components during the data preparation phase (train data, test data, and validation data). The train dataset was fitted with the top seven machine learning classification algorithms: logistic regression, gaussian nave bayes, random forest, k-nearest neighbor, decision tree, extreme gradient boosting, and mild gradient boosting. Because of the complexity of the computation and the size of the dataset, this model training step was performed on the GPU.

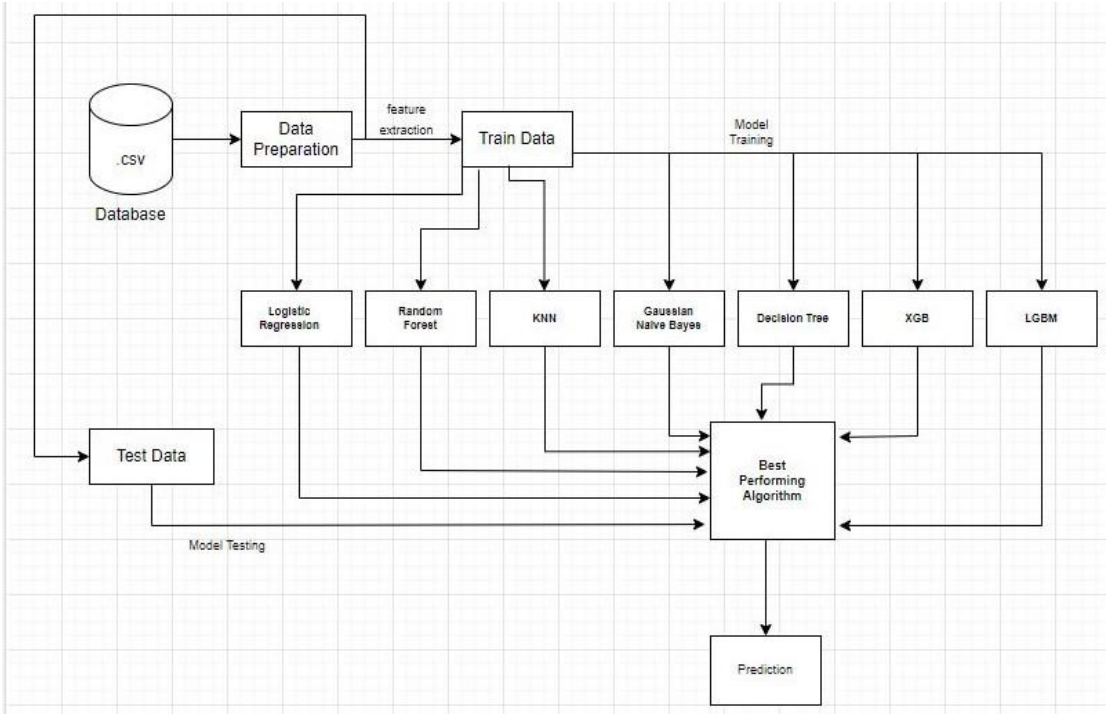


Fig 1: Proposed workflow architecture.

Phase 5: Evaluation and Validation – Following training on the train dataset, the algorithms were assessed (i.e., tested) on the test data by generating predictions. During this phase, it was discovered that performance was not sufficient, with some algorithms performing as badly as 59% - 62%.

	Model	Train Error	Test Accuracy
0	Decision Tree	0.2968	0.7264
1	LGBM Classifier	0.3126	0.6946
2	XGB Classifier	0.3552	0.6490
3	KNN	0.3854	0.6193
4	Logistics Regression	0.4038	0.5988
5	Random Forest	0.2454	0.5988
6	Gaussian Naive Bayes	0.4464	0.5902

Fig 2: Performance after initial training.

This poor performance necessitated a thorough examination of the dataset, which revealed that it was unbalanced.

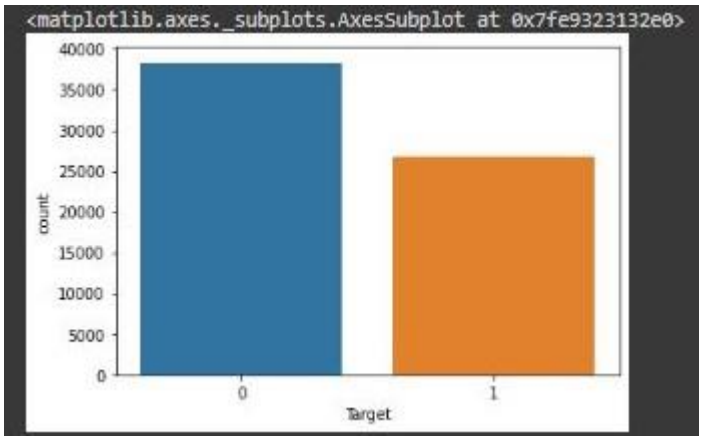


Fig 3: Target feature distribution chart shows imbalanced dataset.

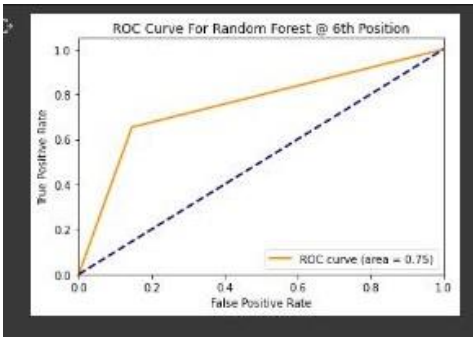
Training was repeated once adequate mitigating actions (data up sampling approaches) were followed, and the algorithms showed substantial improvement, however the least performing algorithms in the prior training still performed badly.

	Model	Train Error	Test Accuracy
0	Decision Tree	0.0325	0.9735
1	KNN	0.0716	0.9358
2	LGBM Classifier	0.1024	0.8980
3	XGB Classifier	0.1095	0.8910
4	Logistics Regression	0.1101	0.8903
5	Random Forest	0.0395	0.8903
6	Gaussian Naive Bayes	0.8793	0.1212

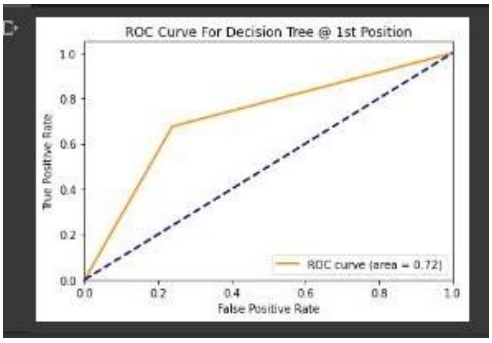
Fig 4: Performance after final training.

ROC Metrics Evaluation After Initial Training (four most performing algorithms).

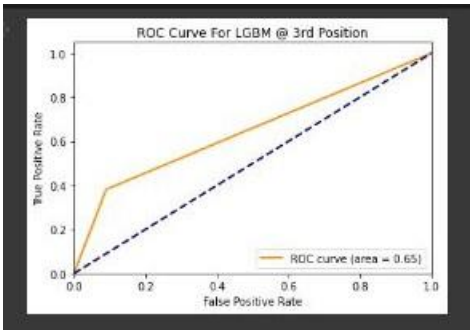
a. ROC for Random Forest



b. ROC for Decision Tree



c. ROC for LGBM



d. ROC for KNN

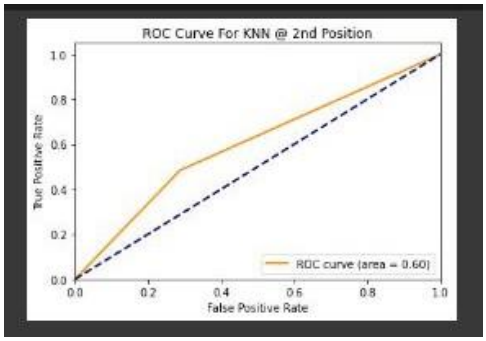
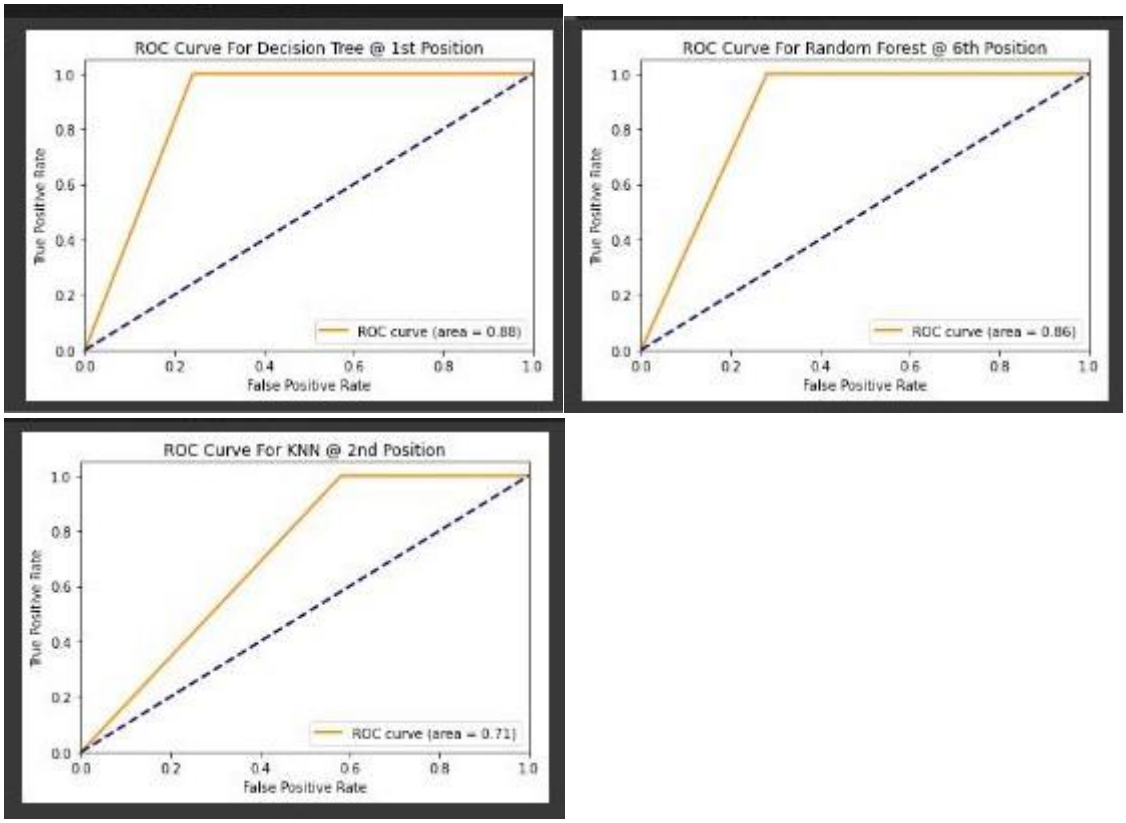


*Fig 5: **Evaluating top four performing algorithms after initial data training using the Area Under Curve – Receiver Operator Characteristics (AUC-ROC) metric.** a) – the random forest algorithm is the top performing algorithm with an AUC of 75%. b) – the decision tree algorithm is the second top performing algorithm with 72% AUC. c) – the LGBM algorithm has an AUC of 65%. d) – the knn algorithm has and AUC of 60%.*

ROC Metrics Evaluation After Up Sampling and Final Training (three most performing algorithms).

a). ROC for Decision Tree

b). ROC for Random Forest



c). ROC for KNN

Fig 6: ***Evaluating top three performing algorithms after up sampling and retraining dataset using the Area Under Curve – Receiver Operator Characteristics (AUC-ROC) metric.*** *a) – the decision algorithm is the top performing algorithm with an AUC of 97%. b) – the random forest algorithm is the second top performing algorithm with 89% AUC. c) – the KNN algorithm has an AUC of 93%.*

5. RESULTS AND DISCUSSIONS

Research Question - "Can classical machine learning classification techniques predict the existence of a keylogger better than deep learning counterparts?" This work was divided into three major areas, each with its own set of performance requirements that each algorithm must meet before moving on to the next. For our study question to be addressed, the algorithms had to meet all three criteria.

- **Category 1 – performance comparison after initial training.**

Table 2: Comparing performance to threshold > 50%.

S/N	Algorithm	Performance (%)	Status (% > 50)
1	Decision Tree	0.72	✓
2	LGBM	0.69	✓
3	XGB	0.64	✓

4	KNN	0.61	✓
5	Logistic Regression	0.59	✓
6	Random Forest	0.59	✓
7	Gaussian Naïve Bayes	0.59	✓

- **Category 2 – performance comparison after up sampling and retraining data.**

Table 3: Further comparing performance to threshold > 80%.

S/N	Algorithm	Performance (%)	Status (% > 80)
1	Decision Tree	0.97	✓
2	LGBM	0.89	✓
3	XGB	0.89	✓
4	KNN	0.93	✓
5	Logistic Regression	0.89	✓
6	Random Forest	0.89	✓
7	Gaussian Naïve Bayes	0.12	×

- **Category 3 – comparing top performing algorithm in this work to top performing algorithms in previous works.**

Table 4: This work VS previous works, which is better?

S/N	Algorithm	Performance (%)	Status (% > 90)
1	SVR	0.83	×
2	SVC	0.87	×
3	Stochastic Gradient Descent	0.90	×
4	Decision Tree	0.97	✓
5	LGBM	0.89	×
6	XGB	0.89	×
7	KNN	0.93	✓
8	Logistic Regression	0.89	×
9	Random Forest	0.89	×

According to the following tables, each algorithm was trained under comparable settings yet performed significantly differently. With this type of dataset, the tree algorithms (Decision Tree and Random Forest) appear to perform better.

Table 2 shows that all of the algorithms passed the performance requirement and progressed to the next category. The dataset was thoroughly cleaned and feature engineered in the new category 2. It also used data up sampling to compensate for the dataset's imbalance. Table 3 compares their performance, demonstrating how poorly the gaussian naive bayes algorithm performed. In this step, the decision tree and knn algorithms performed admirably. To be effective candidates that answer our study question, the algorithms that made it out of the second group must reach the criterion of 90%, which is the greatest performance of the DL algorithm (SGD) in earlier studies by (Madhuri, C., Rama Siritha, et al, 2019). And the final table 4 reveals that only our decision tree and knn algorithms passed this test.

6. CONCLUSION AND FUTURE SCOPE

As a result, we may infer that classical ML “Decision Tree” and “K-Nearest Neighbors” algorithms can more accurately and confidently predict the presence of a keylogger in a system than the DL techniques tested, albeit more complex DL algorithms have yet to be utilized in this study domain. This creates a space for future work. Which would be to employ new and sophisticated DL techniques for this same area, such as Long-Short Term Memory (LSTM), Neural Networks, Transformers, and so on. This is a viable work focus since these algorithms have repeatedly shown to be new and cutting-edge. In addition, as a future work area, the best performing algorithm in this study will be incorporated in real life settings (ie; cyberspace) to combat the widespread keylogging danger, as well as monitored and observed to see whether it works as well in real-world scenarios as it did in this study.

REFERENCES

1. Aporras, 2018. *10 Reasons for loving Nearest Neighbors algorithm ⋆ Quantdare*. [online] Quantdare. Available at: <https://quantdare.com/10-reasons-for-loving-nearest-neighbors-algorithm/>.

2. De Groot, J., 2020. *What is Cyber Security? Definition, Best Practices & More*. [online] Digital Guardian. Available at: <https://digitalguardian.com/blog/what-cyber-security>.

3. Dhiraj, K., 2020. *Top 5 advantages and disadvantages of Decision Tree Algorithm*. [online] Medium. Available at: <https://dhirajkumarblog.medium.com/top-5-advantages-and-disadvantages-of-decision-tree-algorithm-428ebd199d9a>.

4. Fu, J., Yang, H., Liang, Y. and Tan, C., 2014. Enhancing Keylogger Detection Performance of the Dendritic Cell Algorithm by an Enticement Strategy. *Journal of Computers*, 9(6). https://doi.org/10.4304/jcp.9.6.1347-1354.

5. Jang-Jaccard, J. and Nepal, S., 2014. A survey of emerging threats in cybersecurity. *Journal of Computer and System Sciences*, [online] 80(5), pp.973–993. https://doi.org/10.1016/j.jcss.2014.02.005.

6. Madhuri, C., Rama Siritha, K., Sai Ram, S. and Rama Koteswara Rao, G., 2019. Detection of Mobile Keyloggers Using Deep Learning. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 8(8). https://doi.org/<H6845068819 /19©BEIESP>.

7. Pillai, D. and Siddavatam, I., 2018. A modified framework to detect keyloggers using machine learning algorithm. *International Journal of Information Technology*, 11(4), pp.707–712. https://doi.org/10.1007/s41870-018-0237-6.

8. Rahman, A., Ahmed, B., Shoikot, M. and Hossain, J., 2019. Keylogger Detection using Memory Forensic and Network Monitoring Keylogger Detection using Memory Forensic and Network Monitoring. *International Journal of Computer Applications*, 177(11), pp.975–8887.

9. Raj, A., 2021a. *Naive Bayes | Gaussian Naive Bayes with Hyperpameter Tuning in Python*. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/01/gaussian-naive-bayes-with-hyperpameter-tuning/>.

10. Raj, A., 2021b. *The Perfect Recipe for Classification Using Logistic Regression*. [online] Medium. Available at: <https://towardsdatascience.com/the-perfect-recipe-for-classification-using-logistic-regression-f8648e267592#:~:text=Logistic%20regression%20is%20easier%20to>.

11. Rrushi, J., Simms, S., Maxwell, M. and Johnson, S., 2017. Keylogger Detection Using a Decoy Keyboard. *Lecture Notes in Computer Science*, 10(6). https://doi.org/10.1007/978-3-319-61176-1_24.

12. Sharma, H., 2021. *XGBoost- The Miracle Worker*. [online] AlmaBetter. Available at: <https://medium.com/almabetter/xgboost-the-miracle-worker-7518dd55abf3>.

13. Solairaj, A., Prabanand, S., Mathalairaj, J., Prathap, C. and Vignesh, L., 2021. *KEYLOGGERS SOFTWARE DETECTION TECHNIQUES*.

14. Vinothkumar, S. and Aruna, S., 2020. *Mobile Keylogger Detection By Using Machine Learning Technique*. *IJEDRCP1403011 International Journal of Engineering Development and Research*, p.53.

15. Wazid\, M., Katal, A., Goudar, R., Singh, D., Tyagi, A., Sharma, R. and Bhakuni, P., 2013. *A Framework for Detection and Prevention of Novel Keylogger Spyware Attacks*.