Enhancing Malware Detection Accuracy: A Comparative Analysis of Machine Learning Models with Explainable AI

1st Manoj Kumar
Artifical Intelligence and Data science
Global Academy Of Technology
Bangalore, India
manojswamy2013@gmail.com

2nd Darshan S
Artifical Intelligence and Data science
Global Academy Of Technology
Bangalore, India
darshandarshans9845@gmail.com

3rd Sai Harshavardhan
Artifical Intelligence and Data science
Global Academy Of Technology
Bangalore, India
harshavardhaneppw2003@gmail.com

4th Dr.Ashwini Kodipalli
Artifical Intelligence and Data science
Global Academy Of Technology
Bangalore, India
dr.ashwini.k@gat.ac.in

5th Trupthi Rao
Artifical Intelligence and Data science
Global Academy Of Technology
Bangalore, India
trupthirao@gat.ac.in

***Abstract*—The challenge of detecting malware in cybersecurity necessitates the exploration of diverse machine learning models for effective solutions. In this investigation, we assess the performance of several models, including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Decision Trees, Random Forests, Randomized Search CV, Grid Search CV, CatBoost, ADA Boost, XGBoost, and Gradient Boosting, using a comprehensive dataset.**

Our study reveals that CatBoost emerges as the most effective model, achieving an exceptional 99% accuracy in malware prediction. Furthermore, we utilize Lime and Shap explainability techniques to delve into the decision-making process of CatBoost, shedding light on its predictive mechanisms and enhancing transparency in malware detection.

By emphasizing the importance of diverse machine learning models and the significance of explainable AI techniques in cybersecurity, our research contributes to the advancement of malware detection methodologies. It underscores the critical role of transparent and interpretable AI systems in fostering trustworthiness and effectiveness in cybersecurity applications.

Keywords—Malware detection, Cybersecurity, Machine learning, Explainable AI, Lime explainer, Shap explainer, CatBoost, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Decision Trees, Random Forests, Randomized Search CV, Grid Search CV, ADA Boost, XGBoost, Gradient Boosting, Interpretability, Transparency, Feature importance, Model evaluation, Accuracy, Precision, Recall, F1-score, Performance comparison, Cyber threats, Data security, Threat intelligence, Cyber defense.

I. INTRODUCTION

In today's interconnected digital environment, cybersecurity stands as a critical concern, with malicious software, or malware, posing significant risks to individuals, organizations, and nations. Detecting and countering these threats demand sophisticated approaches that harness the capabilities of machine learning (ML) models. ML-based methods have exhibited promising outcomes in identifying and categorizing malware, offering proactive defense strategies against evolving cyber dangers.

This study investigates the effectiveness of diverse ML models in the domain of malware identification. Utilizing datasets enriched with a variety of malware samples, we explore the performance of Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Decision Trees, Random Forests, Randomized Search CV, Grid Search CV, ADA Boost, XGBoost, and Gradient Boosting algorithms. Through systematic experimentation and assessment, we aim to identify the most suitable solution for malware identification tasks.

Moreover, to enhance the transparency and interpretability of our selected model, we employ Explainable AI (XAI) methods, specifically Lime and Shap explainers. These techniques enable us to gain insights into feature significance, model predictions, and decision-making processes, facilitating a deeper comprehension of the underlying principles driving accurate malware identification.

By synthesizing the outcomes of our investigation, we seek to contribute to the ongoing dialogue on cybersecurity and equip stakeholders with actionable insights to fortify their defense mechanisms against cyber threats.

II. RELATED WORKS

A. *Literature Survey*

The study titled "Exploring Deep Learning Methods for Detecting Malware" investigates the use of deep learning techniques in malware detection. Through dynamic analysis, various machine learning algorithms are evaluated for their effectiveness in identifying malware based on extracted features. Experimentation with diverse feature selection techniques and models led to an accuracy level of 96.8% using the J48 decision tree algorithm. The study highlights the importance of feature reduction methods and combining static and dynamic attributes to create a comprehensive feature set for classification. It emphasizes the critical role of advanced machine learning and deep learning strategies in effective malware detection, stressing the significance of feature selection, reduction, and model fine-tuning. By achieving high accuracy rates and conducting comparative analyses with existing literature, the study validates the effectiveness of its approach in categorizing malware specimens precisely. Overall, it provides valuable insights into cybersecurity, showcasing the potential of machine learning algorithms in addressing evolving malware threats [1].

The paper introduces a machine learning framework for malware detection, aiming for improved detection rates and reduced false positives. Utilizing three datasets with varying malware and clean files, feature selection is crucial, with 308 binary features selected for experimentation. Across diverse machine learning algorithms, the framework demonstrates promising results in uncovering malware instances missed by conventional methods. While scalability is emphasized, further enhancements are suggested, including the integration of additional classification algorithms like large margin perceptrons and Support Vector Machines. Despite the challenge of achieving zero false positives, the framework shows commendable performance, especially in identifying evasive malware samples. It emerges as a valuable asset for cybersecurity, offering an alternative to traditional antivirus solutions and paving the path for future advancements in malware detection through machine learning [2].

The academic paper titled "Enhanced Malware Detection Using Deep Learning" explores deep learning methodologies for malware detection, introducing ScaleMalNet—a adaptable architecture designed for handling extensive malware samples using Big Data methodologies. ScaleMalNet employs innovative approaches to malware categorization, combining static and dynamic analyses and implementing a two-phase classification process for organizing malware executables. Additionally, the study pioneers image processing techniques for malware classification, demonstrating the potential of deep learning in strengthening cybersecurity protocols. The paper evaluates the effectiveness of traditional machine learning algorithms and deep learning structures, providing insights into various malware detection models. In summary, the research emphasizes ScaleMalNet's significant contributions to malware detection, particularly in managing large-scale datasets, highlighting the promise of deep learning and image processing in enhancing cybersecurity practices. The integration of a real-time distributed Apache Spark cluster computing platform further enhances the framework's practical utility, facilitating the development of advanced cybersecurity measures against evolving malware threats.[3]

The systematic review of Android mobile malware detection using machine learning approaches provides a comprehensive analysis of research conducted from 2016 to May 2021, following the Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA) model. The study outlines a structured methodology covering research inquiries, search tactics, study selection, data extraction, and validity concerns mitigation. Through meticulous scrutiny, the review analyzes 106 articles focusing on ML/DL models for malware detection, Android code analysis, and vulnerability assessment. The findings underscore the increasing adoption of machine learning in mobile malware detection, emphasizing its effectiveness in identifying code vulnerabilities and malicious applications. By examining various studies, the review highlights the evolution of ML-based methodologies in addressing mobile malware threats and stresses the need for ongoing advancements to combat evolving cyber risks. The systematic approach provides valuable insights for researchers and practitioners aiming to develop robust security solutions for mobile platforms.[4]

The paper introduces an innovative framework for malware detection through machine learning analysis of virtual memory access patterns. It focuses on monitoring and categorizing memory access behaviors in both kernel-level and user-level malware, achieving high detection precision with minimal false positives and negatives. The framework utilizes summarization and feature extraction techniques applied to function/syscall memory access patterns, showcasing superior accuracy and resilience, particularly at low false positive rates. The study emphasizes the framework's potential for hardware implementation, addressing concerns about data volume and discussing the feasibility of online learning within hardware for the detection model. By providing insights into prevalent types of kernel and user-level malware and their impact on memory accesses, the framework offers a promising avenue to enhance cybersecurity measures. Overall, the research suggests that the framework's focus on memory access patterns and machine learning can significantly advance malware detection capabilities, especially in identifying malware-infected runs of known applications with precision and efficiency.[5]

The research paper on "Android Malware Detection Using Dynamic Permissions and Machine Learning" investigates the vulnerability of Android applications to malware and proposes a solution leveraging machine learning techniques for robust detection and prevention. The study compiles a dataset of Android application packages (.apk) from various sources, extracts permissions required by these apps, and evaluates them using five machine learning classification techniques. Evaluation metrics such as True Positive Rate, False Positive Rate, Precision, Recall, and F-measure demonstrate high accuracy levels for classifiers, with J48, Random Forest, and Simple Logistic achieving the highest accuracy rates of 99.6%. In conclusion, the research highlights the importance of dynamic permissions-based analysis in detecting Android malware and emphasizes the efficacy of machine learning in enhancing mobile application security. The proposed approach shows potential in improving malicious application detection by analyzing permissions during installation and start-up, offering valuable insights and a promising framework for mitigating malware risks in the mobile ecosystem.[6]

Ivan Firdausi's Bachelor's Thesis, titled "Exploring Machine Learning Techniques for Behavior-Based Malware Detection," delves into the cybersecurity domain, specifically examining the utilization of machine learning for detecting malware based on behavioral patterns. The study scrutinizes a range of machine learning algorithms including IBk, SVM, J48, and Multilayer Perceptron, assessing their performance metrics both with and without feature selection. By evaluating the effectiveness of various classifiers and feature selection methods in behavior-based malware detection, the research contributes significant insights into enhancing malware detection capabilities. The findings underscore the pivotal role of machine learning techniques in bolstering cybersecurity measures, thereby laying the groundwork for more resilient defense mechanisms against evolving malware threats.[7]

In their paper, Chandrasekar Ravi and R Manoharan introduce an innovative approach to malware detection leveraging Windows API sequences and machine learning techniques. Their dynamic system monitors real-time program execution behavior, effectively discerning between benign and malicious processes. Employing a 3rd order Markov chain to model API calls and association mining-based classification, the system enhances accuracy. Experimental findings indicate that the proposed system surpasses existing data mining-based detection systems, achieving high detection accuracy while monitoring only a minimal subset of API categories. The system's key novelty lies in its iterative learning process combined with run-time monitoring, rendering it a dynamic malware detection system.[8]

The paper introduces an Intelligent Dynamic Malware Detection framework that utilizes machine learning for IP reputation management in cyber security. It assesses the efficacy of diverse machine learning techniques including Naive Bayes, Support Vector Machine, Decision Tree, and Mini Batch K Means on network datasets for malware detection and classification. Notably, the research underscores the necessity of dynamically analyzing malware samples linked with IP addresses and proposes a framework that dynamically calculates and updates IP address reputations to mitigate false alarm rates. The results underscore the proficiency of Decision Trees in categorizing unknown malware samples and suggest potential reductions in false alarm rates for identifying malicious IP addresses. Furthermore, the study emphasizes the importance of leveraging big data and external sources to augment severity and confidence levels in cyber threat detection, with the aim of fortifying the IP reputation framework for cyber security.[9]

Based on the search findings, the paper delves into comparing phone-based and emulator-based dynamic analysis methods for identifying malicious applications on Android devices. Employing machine learning algorithms, the study aims to classify and detect malware by analyzing features derived from the applications. Experiments were conducted using a dataset containing both malware and benign samples. The outcomes indicate that phone-based analysis outperforms emulator-based analysis in feature extraction and app analysis. Specifically, phone-based analysis demonstrates superior accuracy, F-measure, true positive rate, and false positive rate compared to emulator-based analysis. The paper concludes that employing a real phone for analysis yields greater efficiency and more accurate results in malware detection.[10]

The paper introduces an innovative intelligent malware analysis framework designed for detecting and classifying malware using machine learning algorithms. This methodology merges dynamic and static analyses of malware samples, focusing on their behavioral similarities. Through experimental validation, the framework proves its efficacy in identifying and categorizing malicious files, achieving high accuracy rates across various machine learning models. Among these models, the J48 decision tree exhibits superior accuracy performance. Additionally, the paper underscores the utilization of Cuckoo Sandbox for malware analysis and emphasizes the significance of dynamic information extraction through sandboxing. In summary, the framework presents promising outcomes in the detection and classification of malware samples leveraging machine learning algorithms.[11]

The research paper delves into mobile device malware detection, specifically focusing on Android malware. Through an extensive evaluation study utilizing network traffic and a dataset comprising 1,000 malware samples sourced from the Android Malware Genome Project, the researchers scrutinized various classifiers. Among them, the Bayesian classifier and random forest were compared. Remarkably, the research achieved a peak detection rate of 99.99% using the random forest classifier, underscoring the thoroughness and effectiveness of their investigation. In contrast, a study by Yerima et al. yielded a detection rate of 90.60% with the Bayesian classifier, emphasizing the advancements made in this study. Moreover, the paper explores the application of ROC curves for classifier performance evaluation, emphasizing the significance of the area under the curve (AUC) in gauging prediction accuracy. Ultimately, this research surpasses prior studies in both breadth and outcomes, as detailed on page 12 of the paper.[12]

The paper explores the classification of malware using K-NN, DT, and SVM algorithms, as per the search findings. Evaluation metrics such as True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), True Positive Rates (TPR), False Positive Rates (FPR), and overall accuracy were employed to assess performance. Notably, the Decision Tree (DT) algorithm demonstrated the highest detection accuracy at 99%, followed by K-NN at 94%, and SVM at 91%. The research methodology involves static analysis of malware samples, feature extraction from PE files, and the application of three algorithms for classification. The authors conclude that DT represents the optimal machine learning technique for malware detection, as detailed on page 4 of the paper.[13]

The paper delves into diverse techniques for analyzing and categorizing malware samples, encompassing both static and dynamic analysis methodologies. It underscores the extraction of features from malware samples without their execution and elaborates on tools utilized for scrutinizing registry alterations, memory dumps, and network traffic. Furthermore, machine learning algorithms are deployed to classify malware samples leveraging the extracted features. The paper concludes by contemplating the future prospects of malware classification and underscores the imperative of devising robust malware classifiers, as documented on pages 3, 5, 9, and 13. [14]

The paper explores the construction and validation of a machine learning-driven framework designed to differentiate between malicious and benign applications. Various algorithms, including one-sided perceptrons and kernelized one-sided perceptrons, are employed to train and assess the system across diverse datasets. Cross-validation methodologies gauge algorithmic accuracy, with optimal outcomes observed using 5-fold cross-validation of the polynomial kernel function. Furthermore, the framework undergoes testing on an expanded dataset, revealing a decline in malware detection accuracy as dataset dimensions increase. In summary, the framework exhibits encouraging outcomes in accurately categorizing applications, characterized by minimal false negative rates, as documented on pages 5, 6, and 7. [15]

The paper introduces a novel malware evasion reinforcement learning platform designed to involve a wider audience. Implemented as an adaptable OpenAI gym, the environment comprises a malware specimen and a customizable anti-malware mechanism. Within this setup, the agent receives feedback encompassing rewards, observations, and diagnostic data. The agent selects from a range of mutations or actions to maintain the format and operation of the malware specimen. Additionally, the paper touches upon the utilization of generative adversarial networks (GANs) to craft adversarial malware instances aimed at circumventing black-box static PE malware engines. Overall, the paper outlines a framework for instructing RL agents in evading malware detection systems, as detailed on pages 4 and 5. [16]

The paper explores a methodology for identifying malware within Android applications employing machine learning approaches. Feature extraction from the applications is conducted, and program flow is represented using control flow graphs. Various kernels are applied to train the model and classify applications as benign or malicious. Additionally, the paper touches upon related research in the domain of malware detection on mobile devices and presents experimental findings. [17]

The paper delves into the analysis of executable binaries for the purpose of detecting and categorizing malware. Researchers extract diverse features from various components of the PE file, including headers, sections, and directory entries, among others. Additionally, they leverage data from Cuckoo Sandbox reports to glean insights into registry keys, API calls, files, IP addresses, and DNS queries during execution. Emphasizing the significance of both static and dynamic analysis, the paper presents different combinations of dynamic features for comprehensive malware detection. Ultimately, the paper proposes a holistic analysis approach that integrates a wide array of features to proficiently detect and classify malware. [18]

The paper explores contemporary trends in malware development and outlines the methodologies employed in statistical malware analysis. It underscores the transition in hacker motives from seeking notoriety to pursuing financial or political objectives. The document also references the adoption of ensemble or boosted variations of multiple classifiers for identifying malicious executables. However, it notes a dearth in comparative analysis regarding the sequence of classifiers and the pertinent features for each algorithm. The conclusion highlights the research gap concerning solutions for multi-stage delivery of malware or addressing advanced persistent threats (APTs). Furthermore, it underscores the inefficiency of signature-based antivirus systems and the detection challenges associated with k-ary codes. [19]

The paper centers on leveraging machine learning for malware detection by analyzing opcode sequences. The thesis comprises six chapters, with Chapter 2 delving into the significance of malware research, various malware types, obfuscation methods, and detection techniques.

Chapter 3 offers a survey of relevant literature, exploring studies on opcodes as malware indicators and the impact of packers. The methodologies employed in the thesis research encompass data collection, data preprocessing, reverse engineering, and machine learning.[20]

III. METHODOLOGY

Dataset Acquisition and Preprocessing

1. The dataset was obtained from Kaggle and comprises features extracted from both malicious and non-malicious Windows executable files.

2. The dataset consists of 373 samples, with 301 labeled as malicious files and 72 labeled as non-malicious files, resulting in an imbalanced dataset.

3. Features were extracted using a hybrid approach, combining binary hexadecimal and DLL call features from Windows executables.

4. Each sample in the dataset is represented by 531 features, denoted as F_1 through F_531, along with a label indicating whether the file is malicious or non-malicious.

5. Minimal preprocessing involved handling missing values and conducting basic exploratory data analysis (EDA) for general visualizations.

Model Selection and Training

1. A diverse set of machine learning models was selected to compare their performance in malware detection, including:

- Support Vector Machine (SVM)
- K-Nearest Neighbors (KNN)
- Logistic Regression
- Decision Tree
- Random Forest
- Randomized Search CV
- Grid Search CV
- CatBoost
- ADA Boost
- XGBoost
- Gradient Boost

2. Popular ensemble methods were employed alongside traditional classifiers to ensure robustness.

3. The dataset was split into training and testing sets to evaluate model performance accurately.

4. Each model was fitted to the training data, and performance was evaluated using appropriate evaluation metrics, considering the imbalanced nature of the dataset.

Explainable AI Techniques

1. LIME (Local Interpretable Model-agnostic Explanations) and SHAP (SHapley Additive exPlanations) were employed to enhance the interpretability of the best-performing model, CatBoost.

2. LIME and SHAP provide insights into the decision-making process of complex machine learning models, aiding in understanding feature importance and model behavior.

3. By leveraging LIME and SHAP, the aim was to elucidate the key features driving the predictions of the CatBoost model, thereby enhancing transparency and trustworthiness in the malware detection process.

Evaluation Metrics

1. Standard evaluation metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC) were utilized to assess model performance.

2. Metrics accounting for the imbalanced nature of the dataset, such as precision-recall curves, accuracy and the F1-

score, were given special attention but finally the results were considered based on accuracy alone.

Experimental Setup

1. All experiments were conducted using the Python programming language with relevant libraries such as scikit-learn, CatBoost, LIME, and SHAP.

2. Experiments were performed on a standard computational environment to ensure reproducibility and consistency.

3. The python IDE used was VS code with Jupiter plugin.

Ethical Considerations

1. Ethical considerations in cybersecurity research, including data privacy, fairness, and potential biases in model predictions, were acknowledged.

2. Measures were taken to anonymize and protect sensitive information within the dataset, and ethical guidelines were followed throughout the research process.

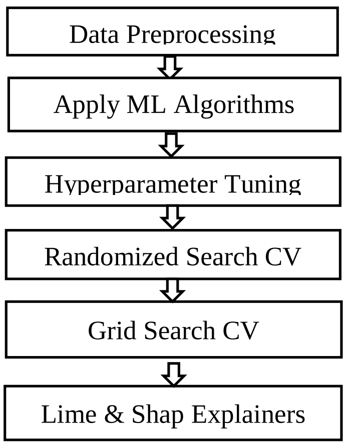


Fig 1: flowchart depicting the steps of Methodology

IV. DATACOLLECTION AND PREPROCESSING

A. Origin of the Dataset

The dataset utilized for this study was obtained from Kaggle, a widely recognized platform hosting a diverse array of datasets for machine learning and data science endeavors. It encompasses features extracted from both malicious and non-malicious Windows executable files, thus offering a comprehensive foundation for conducting research in malware detection.

B. Description of the Dataset

The dataset comprises 373 samples, of which 301 are labeled as malicious files and 72 as non-malicious files, resulting in an imbalanced dataset. These samples were subjected to a hybrid feature extraction methodology, incorporating binary hexadecimal and DLL call features from Windows executables. Each sample within the dataset is characterized by 531 features, denoted as F_1 through F_531, accompanied by a label indicating its malicious or non-malicious classification. Given the intricate nature of binary hexadecimal features, they were denoted as F_1, F_2, and so forth, alongside DLL calls. The dataset's suitability for training and evaluating machine learning models in the realm of malware detection underscores its potential to address practical cybersecurity challenges.

C. Preprocessing Procedures

Missing Value Treatment:

To ensure the integrity and completeness of the dataset for subsequent analysis, preprocessing techniques were applied to address missing values. This involved employing strategies such as imputation or removal of missing values based on their extent and distribution across features.

Exploratory Data Analysis (EDA):

Foundational EDA techniques were implemented to gain insights into the distribution, characteristics, and interrelationships among the dataset's features. Visualization methodologies, including histograms, box plots, and scatter plots, were employed to uncover patterns, outliers, and potential correlations within the dataset.

Data Partitioning:

The dataset underwent partitioning into training and testing sets to facilitate model training, validation, and evaluation processes. A conventional split ratio, such as 70:30 or 80:20, was adopted to maintain a balanced representation between training and testing samples.

D. Ethical Considerations

Data Privacy and Confidentiality Measures:

Stringent measures were enacted to safeguard the privacy and confidentiality of sensitive information within the dataset, aligning with established ethical guidelines and regulations. Anonymization techniques may have been employed to obfuscate personally identifiable information (PII) or other sensitive data.

Fairness and Bias Mitigation Strategies:

Concerted efforts were made to mitigate potential biases inherent in the dataset, thereby ensuring equitable representation and treatment of different classes (malicious vs. non-malicious files). Bias-aware preprocessing techniques might have been utilized to rectify imbalances and promote impartial model performance across diverse samples.

V. RESULT

The results of the experimentation showcase notable achievements in terms of model performance and hyperparameter tuning across various machine learning algorithms all the accuracies are as illustrated below.

Machine learning algorithms	Accuracy observed
Logistic regression	98.6%
KNN	97.3%
Decision tree	98.7%
SVM	98.6%
Random forest	97.6%
Randomized search CV	97.8%
Grid search CV	97.4%
Gradient boosting	94.6%
Ada boost	94.1%
XGboost	98.6%
Catboost	99.9%

Table 1: Table of the model accuracy

The CatBoost algorithm demonstrated exceptional accuracy, achieving a remarkable 99.9% accuracy rate, signifying its efficacy in malware detection tasks.

Both randomized search CV and grid search CV techniques were applied to optimize the performance of the Random Forest classifier, a component of the bagging ensemble method.

Post-hyperparameter tuning, the Random Forest classifier yielded its best performance with an accuracy of 97.7%.

Optimal hyperparameter values were determined to be:

'n_estimators': 1200

'min_samples_split': 3

'min_samples_leaf': 8

'max_features': 'auto'

'max_depth': 780

'criterion': 'gini'

Two explainable AI methods, Lime and Shap, were employed to elucidate model predictions and feature importance.

Lime was applied to the CatBoost algorithm, which exhibited the highest accuracy among the models tested.

Lime provided two observable output charts, aiding in the interpretation and understanding of Catboost's decision-making process.

These results underscore the effectiveness of CatBoost in malware detection, as well as the significance of hyperparameter tuning and explainable AI techniques in enhancing model transparency and interpretability. Further analysis and interpretation of these results can provide valuable insights for future research and development efforts in cybersecurity.

By running the lime explainer on just the Random Forest we got the output for the feature importance as mentioned in the figure.

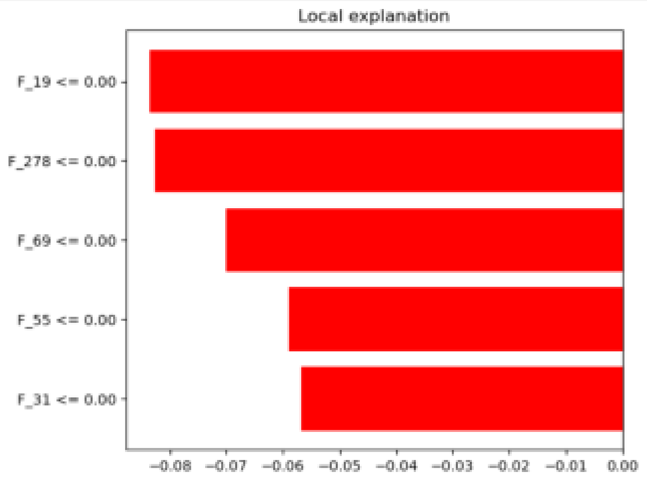


Fig 2: Random Forest Lime Explainer

Lime was made to run on Random Forest model for providing contrast of feature importance between the lower accuracy models and Catboost.

Lime was then made to run on Catboost and the results of feature importance is as depicted.

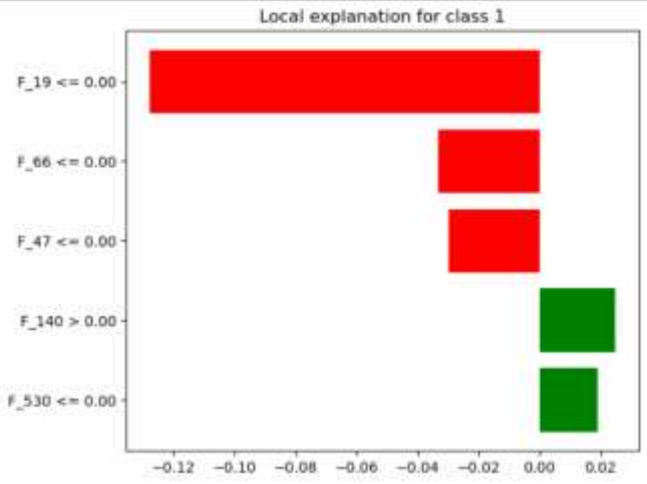


Fig 3: Catboost Lime Explainer

Next the Shap explainer was made to run on the most accurate model i.e Catboost and the below feature importance were as observed.

Sl_no.	Features	Importance
1	F_20	23.420172
2	F_19	11.530165
3	F_139	10.001026
4	F_179	5.156478
5	F_266	4.745975
….	….	….
527	F_492	0.000000
528	F_232	0.000000

529	F_230	0.000000
530	F_28	0.000000
531	F_1	0.000000

Table 2: Table of Feature Importance Shap (Catboost)

As we can see the Catboost has given F_20 feature the highest importance.

And this can also be illustrated as the below plot.

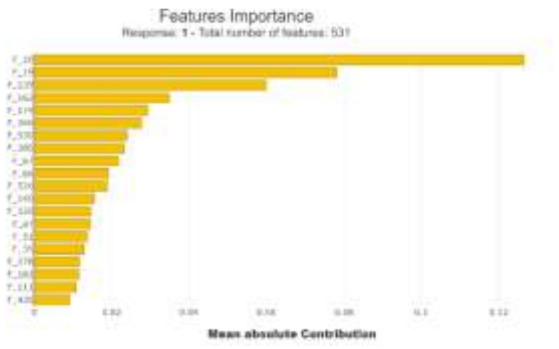


Fig 4: Shap Feature Importance on Catboost

And the above can also be depicted as the below plot which gives the contrast between subset features and global features.

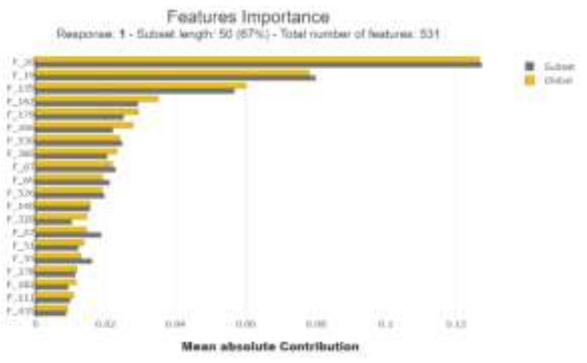
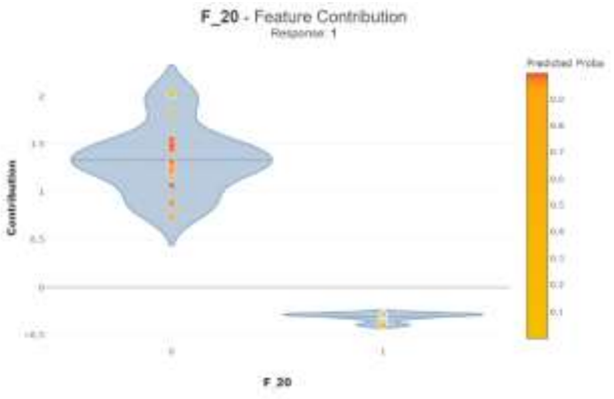


Fig 4: Shap Feature Importance on Catboost (contrast between Global and Subset features)

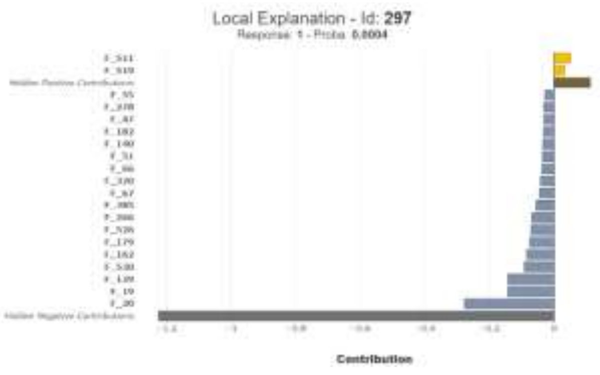
As we can see from the feature importance that F_20 is the most important feature and the second highest is feature F_19.

Lime only considers the subset features whereas Shap considers both Global and subset features.

An illustration of the feature F_20's contribution is given.



Finally, a graph depicting the negative contribution is also given below.



VI. Conclusion

The research concludes that features F_20 and F_19 hold the highest importance in malware detection, as indicated by both SHAP and Lime explainers applied to the CatBoost algorithm. This consistency in results underscores the reliability and strength of the feature selection process within the dataset tailored for malware detection.

The prominence of features F_20 and F_19 suggests that they play crucial roles in distinguishing between malicious and non-malicious executable files. While the exact attributes captured by these features may vary, their consistent importance underscores their relevance in identifying patterns indicative of malicious behavior within malware samples.

Moreover, the alignment between SHAP and Lime explainers enhances the transparency of the CatBoost model's decision-making process, providing valuable insights into the features driving accurate malware detection.

In summary, the significance of features F_20 and F_19, as highlighted by both SHAP and Lime explainers, underscores their pivotal role in malware detection. These findings contribute to advancing cybersecurity research by shedding light on key characteristics that differentiate malicious from non-malicious executable files. Further exploration into the specific attributes of features F_20 and

F_19 could yield insights to enhance malware detection algorithms and strengthen cybersecurity measures.

REFERENCES

[1] Rathore, Hemant, et al. "Malware detection using machine learning and deep learning." Big Data Analytics: 6th International Conference, BDA 2018, Warangal, India, December 18–21, 2018, Proceedings 6. Springer International Publishing, 2018.

[2] Gavriluț, Dragoş, et al. "Malware detection using machine learning." 2009 International multiconference on computer science and information technology. IEEE, 2009.

[3] Vinayakumar, R., et al. "Robust intelligent malware detection using deep learning." IEEE access 7 (2019): 46717-46738.

[4] Senanayake, Janaka, Harsha Kalutarage, and Mhd Omar Al-Kadri. "Android mobile malware detection using machine learning: A systematic review." Electronics 10.13 (2021): 1606.

[5] Xu, Zhixing, et al. "Malware detection using machine learning based analysis of virtual memory access patterns." Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017. IEEE, 2017.

[6] Mahindru, Arvind, and Paramvir Singh. "Dynamic permissions based android malware detection using machine learning techniques." Proceedings of the 10th innovations in software engineering conference. 2017.

[7] Firdausi, Ivan, Alva Erwin, and Anto Satriyo Nugroho. "Analysis of machine learning techniques used in behavior-based malware detection." 2010 second international conference on advances in computing, control, and telecommunication technologies. IEEE, 2010.

[8] Ravi, Chandrasekar, and R. Manoharan. "Malware detection using windows api sequence and machine learning." International Journal of Computer Applications 43.17 (2012): 12-16.

[9] Usman, Nighat, et al. "Intelligent dynamic malware detection using machine learning in IP reputation for forensics data analytics." Future Generation Computer Systems 118 (2021): 124-141.

[10] Alzaylaee, Mohammed K., Suleiman Y. Yerima, and Sakir Sezer. "Emulator vs real phone: Android malware detection using machine learning." Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics. 2017.

[11] Sethi, Kamalakanta, et al. "A novel malware analysis framework for malware detection and classification using machine learning approach." Proceedings of the 19th international conference on distributed computing and networking. 2018.

[12] Narudin, Fairuz Amalina, et al. "Evaluation of machine learning classifiers for mobile malware detection." Soft Computing 20 (2016): 343-357.

[13] Selamat, N., and F. Ali. "Comparison of malware detection techniques using machine learning algorithm." Indones. J. Electr. Eng. Comput. Sci 16 (2019): 435.

[14] Singh, Jagsir, and Jaswinder Singh. "A survey on machine learning-based malware detection in executable files." Journal of Systems Architecture 112 (2021): 101861.

[15] Alqahtani, Ebtesam J., Rachid Zagrouba, and Abdullah Almuhaideb. "A survey on android malware detection techniques using machine learning algorithms." 2019 Sixth International Conference on Software Defined Systems (SDS). IEEE, 2019.

[16] Anderson, Hyrum S., et al. "Evading machine learning malware detection." black Hat 2017 (2017): 1-6.

[17] Sahs, Justin, and Latifur Khan. "A machine learning approach to android malware detection." 2012 European intelligence and security informatics conference. IEEE, 2012.

[18] Ijaz, Muhammad, Muhammad Hanif Durad, and Maliha Ismail. "Static and dynamic malware analysis using machine learning." 2019 16th International bhurban conference on applied sciences and technology (IBCAST). IEEE, 2019.

[19] Nath, Hiran V., and Babu M. Mehtre. "Static malware analysis using machine learning methods." Recent Trends in Computer Networks and Distributed Systems Security: Second International Conference, SNDS 2014, Trivandrum, India, March 13-14, 2014, Proceedings 2. Springer Berlin Heidelberg, 2014.

[20] Bragen, Simen Rune. Malware detection through opcode sequence analysis using machine learning. MS thesis. 2015.

[21] Chumachenko, Kateryna. "Machine learning methods for malware detection and classification." (2017).

[22] Agarkar, Sanket, and Soma Ghosh. "Malware detection & classification using machine learning." 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC). IEEE, 2020.

[23] Karbab, ElMouatez Billah, et al. "MalDozer: Automatic framework for android malware detection using deep learning." Digital Investigation 24 (2018): S48-S59.

[24] Yerima, Suleiman Y., Sakir Sezer, and Igor Muttik. "Android malware detection using parallel machine learning classifiers." 2014 Eighth international conference on next generation mobile apps, services and technologies. IEEE, 2014.

[25] Vatamanu, Cristina, et al. "A comparative study of malware detection techniques using machine learning methods." International Journal of Computer and Information Engineering 9.5 (2015): 1150-1157.