© April 2025 | IJIRT | Volume 11 Issue 11 | ISSN: 2349-6002

AI-Powered Android Malware Prediction Using Stacked Ensemble Learning

K. Sai Nithin Reddy[1], M. Dekshith Rao[2], N. Rohith Reddy[3], Ms. Kasturi Anoopama[4]
[1,2,3] UGC Students, Sreenidhi Institute of Science and Technology, Telangana, India
[4]Assistant Professor, Sreenidhi Institute of Science and Technology, Telangana, India

***Abstract*—The rapid growth of Android-based applications has led to a rise in malicious apps that compromise user privacy and security. Traditional signature-based malware detection methods struggle against evolving threats, making machine learning (ML)- based approaches essential. In this paper, we evaluate multiple ML classifiers and propose a stacking-based ensemble model to enhance malware detection. Our approach integrates pre- dictions from base classifiers using a meta-classifier, improving accuracy, recall, and ROC-AUC. Experimental results show that the stacking model outperforms individual classifiers, offering better generalization and resilience against modern malware obfuscation techniques.**

***Index Terms*—Android Malware Detection, Stacking Ensemble Learning, Cybersecurity, Feature Engineering, Explainable AI**

I. INTRODUCTION

The unprecedented growth of Android applications has resulted in an upsurge in malware threats that put users' privacy and security at risk. Conventional methods of malware detection such as signature-based and heuristic approaches have proven weak against more advanced forms of malware which are well-versed in obfuscation and polymorphism. Therefore, we require advanced detection mechanisms, among which machine learning (ML) has emerged as one of the more promising mechanisms.In this study, we propose a Stacking-based ensemble learning approach for Android malware detection, leveraging the strengths of multiple ML classifiers. We initially train and evaluate various classifiers, including Random Forest, Decision Tree, XGBoost, CATBoost, and other ensemble models with hyperparameter optimization, and select the top-performing ones for the final Stacking model. The meta-classifier in our Stacking framework combines the predictions of these classifiers to enhance malware detection accuracy while minimizing false positives.For experimentation, we have utilized an Android malware dataset collected from Kaggle and online sources, comprising both benign and malicious applications with permissions data. We extracted only the most relevant static and dynamic features using feature engineering, eliminating unnecessary attributes to im- prove model efficiency. Key features include permissions, API calls, Internal permissions, which were used to train and evaluate our models.Experimental results demonstrate that the Stacking model outperforms individual classifiers in terms of accuracy, recall, and ROC-AUC, making it a robust approach for real-world malware detection. This study highlights the effectiveness of ensemble learning in strengthening Android security and suggests future improvements by incorporating deep learning techniques and real-time behavioral analysis.

II. LITERATURE SURVEY

The increasing sophistication of Android malware necessitates advanced detection mechanisms that go beyond traditional methods. Recent research has demonstrated that ensemble learning techniques, particularly stacking models, significantly improve malware detection accuracy by combining multiple classifiers to minimize false positives and enhance generalization [3]. Studies have shown that hybrid models integrating convolutional neural networks (CNNs) and long short-term memory (LSTM) networks further strengthen detection capabilities by capturing complex patterns in malicious applications [4]. Additionally, incorporating feature selection techniques such as analyzing permissions, API calls, and opcodes has been found crucial in optimizing performance and reducing computational overhead [5]. The effectiveness of stacking-based ensemble models has been validated in multiple studies, showing that combining different classifiers leads to superior performance compared to individual models [1]. One study proposed a stacking-based model that integrates decision trees, XGBoost, and random forests, achieving a higher detection rate than individual classifiers [1]. Another research work highlighted the significance of analyzing sensitive API calls, demonstrating that focusing on critical API interactions can enhance malware detection accuracy while reducing unnecessary computations [6]. Similarly, ensemble clustering methods have been applied to predict Android malware families, im- proving classification robustness against evolving threats [5]. A comparative study revealed that XGBoost-based ensemble models outperform traditional classifiers by efficiently handling high-dimensional feature spaces and improving detection rates [2]. The use of deep learning-based models, such as CNNs and LSTMs, in combination with ensemble learning techniques, has further refined detection accuracy and resilience against obfuscation techniques [4]. Moreover, research has demonstrated that stacking models can dynamically adapt to new malware strains, making them more effective in real-world deployment scenarios [3]. Another study emphasized the role of feature engineering in malware detection, showing that selecting relevant features and eliminating unnecessary ones significantly enhances model efficiency [5]. Furthermore, an evaluation of multiple feature extraction methods has revealed that stacking models incorporating permission-based and opcode-based analysis demonstrate improved efficiency in distinguishing between benign and malicious applications [3]. Recent advancements in the field also indicate that integrating real-time behavioral analysis with ensemble models could further improve malware detection frameworks [6]. By leveraging these advancements, researchers continue to refine malware detection methodologies, ensuring more robust and adaptive security solutions for Android users.

III. PROBLEM STATEMENT

The rapid increase of Android apps has resulted in the expansion of malware threats, and these significantly violate user privacy and security. The traditional detection techniques, such as signature-based and heuristic techniques, struggle with new malware that develops techniques of concealing its code, modifying its appearance, and encryption to evade detection. Due to this, new detection techniques are needed that will be effective in identifying malicious apps without triggering false alarms. Several techniques of enhancing malware detection have been studied in the past, but most existing models still have challenges, particularly in handling dynamic threats.To address these issues, this study introduces a stacking-based model from ensemble learning approach that strengthens malware detection by combining the predictive capabilities of multiple classifiers, including Random Forest, Decision Tree, XGBoost, CATBoost and Gradient Boosting . Unlike traditional single-model approaches, this method incorporates feature selection and hyperparameter tuning to improve detection efficiency. By analyzing application permissions, the proposed model effectively differentiates between safe and harmful applications, providing a more accurate, reliable and adaptable detection strategy. This approach enhances classification performance and ensures better resilience against emerging modern threats. Experimental findings confirm that stacking model surpasses individual classifiers in terms of accuracy, recall, and precision, making it a dependable and scalable solution for Android malware detection. The ability of identifying new and modern evolving malware underscores its practical value in security applications, app verification systems, and enterprise level protection solutions. By overcoming the constraints of earlier methods, this research contributes to the advancement of more effective, reliable, and scalable security solutions designed to counter the modern malware security and privacy related risks.

IV. PROPOSED SYSTEM

The proposed system introduces a novel approach to An- droid malware detection by utilizing machine learning-based stacking technique to enhance accuracy, precision, and re- call. With an increasing number of Android applications, risk of malware infections has become a major concern, as at- tackers continuously develop sophisticated methods to evade traditional detection systems. Existing techniques such as signature-based detection rely on predefined patterns and known malware signatures, making them ineffective against newly emerging or obfuscated malwares/viruses. Behavior- based methods, on the other hand, monitor app activities and interactions, but they require extensive computational resources, making them less practical for real-time detection. To overcome these limitations, our system takes a lightweight yet highly effective approach by predicting an application’s malicious nature solely based on the permissions it

requests. The model takes binary input, where each permission is represented as 1 if requested and 0 otherwise, allowing for a straightforward yet powerful feature representation.
In order to obtain the highest predictive accuracy, multiple machine learning classifiers are trained with hyperparameter tuning to maximize their working efficiency. The best four classifiers with highest performance are selected and combined into a stacking ensemble, where their predictions are once again processed to produce a final, more precise output. Stacking is especially useful since it allows different classifiers to synergistically work together, thus reducing bias and maximizing generalization on different types of malware. Stacking is contrasted with traditional single-model methods, which are plagued by high variance or overfitting behavior. Stacking takes advantage of the strengths of different machine learning models to construct a stronger detection system. Through the use of this method, our system is capable of performing better accuracy in learning, thus reducing the rate of false positives and false negatives, which are common in malware classification tasks.The given system is an efficient and scalable solution that can be easily adopted into already present cybersecurity infrastructure without requiring the employment of resource-intensive dynamic analysis. It has the ability to serve as a stand-alone tool for app stores, security researchers, or even ordinary users to check the safety of apps before installation. Moreover, its minimal size guarantees efficient working on handheld devices with less disruption to total performance. With advancements in the future, the approach may be bolstered through the integration of deep learning frameworks like convolutional neural networks (CNNs) or re- current neural networks (RNNs) so that automated generation of higher-order features as well as discovery of more complex behavior patterns within malware may become feasible. As cyber security as well as artificial intelligence keep growing stronger, the given system is bound to evolve into an even stronger malware detector, thus increasing the security level of Android ecosystems for users across the world.

V. METHODOLOGY

A. Data Collection

For training and testing our malware detection model, we employ a dataset on Kaggle as well as other available public repositories containing permission data for Android applications. The dataset contains both benign and malicious Android applications, with each application characterized by the set of permissions it demands. The permissions are the most important signs of possible malicious behavior because malware will demand an unusually long list of irrelevant permissions to be able to access user information and system resources. The data set is a heterogeneous collection of applications categorized based on their safety, thereby offering an equal sample of both safe and malicious applications. The permission requests of each application are offered in a formatted binary form where 1 means a requested permission and 0 means the lack of a requested permission. The formatted data simplifies our ma- chine learning algorithms' ability to look for patterns regarding malware. The data includes numerous Android application categories, thereby offering the model generalization across numerous categories and avoiding overfitting to any given subset of applications.In addition to improving data quality, we pre-process our dataset by eliminating duplicate records, managing missing values, and feature normalization where appropriate. We also conduct exploratory data analysis (EDA) to determine the most requested permissions by malware, which allows us to understand their intents better and improve our model. With real-world permission data from diverse sources, our dataset forms a solid base for training a solid and accurate malware detection system.

B. Data Preprocessing

In this project, we aim to predict whether an application contains malware based solely on the permissions it requests. The dataset consists of permissions extracted from both mal- ware and benign apps, serving as input features for our ma- chine learning model. To ensure data quality and consistency, missing values were handled by replacing them with the mode of the respective feature, as this method retains the most frequently occurring value without introducing bias. Since the dataset contained a mix of numerical and string-based values, all non-numeric data was removed to maintain compatibility with machine learning models. Additionally, feature selection played a crucial role in improving model performance and interpretability. We utilized SHAP (SHapley Additive exPla- nations) to analyze feature importance and retain only the most relevant permissions while discarding less significant ones.

This approach helped reduce noise, prevent overfitting, and ensure that the model was trained on meaningful and highly influential features. By applying these preprocessing steps, we refined the dataset, making it more suitable for classification tasks and improving the overall accuracy of malware detection.

C. Machine Learning Models

TABLE I: Performance of Stacking Model

Metric	Value
Precision (%)	97.55
Recall (%)	98.2
Accuracy (%)	97.15
ROC Score (%)	99.66
TP	3935
FP	99
FN	72
TN	1894

TABLE II: Performance of XGBoost Model

Metric	Value
Precision (%)	98.00
Recall (%)	98.0
Accuracy (%)	97.00
ROC Score (%)	97.00
TP	3917
FP	78
FN	90
TN	1915

TABLE III: Performance of CatBoost Model

Metric	Value
Precision (%)	98.00
Recall (%)	99.0
Accuracy (%)	97.00
ROC Score (%)	97.00
TP	3952
FP	96
FN	55
TN	1897

TABLE IV: Performance of LightGBM Model

Metric	Value
Precision (%)	97.00
Recall (%)	98.0
Accuracy (%)	97.00
ROC Score (%)	96.00
TP	3940
FP	123
FN	67
TN	1870

TABLE V: Performance of Gradient Boosting Model

Metric	Value
Precision (%)	97.31
Recall (%)	98.25
Accuracy (%)	97.02
ROC Score (%)	96.39
TP	3937
FP	109
FN	70
TN	1884

TABLE VI: Performance of Extra Trees Model

Metric	Value
Precision (%)	97.33
Recall (%)	98.38
Accuracy (%)	97.12
ROC Score (%)	96.48
TP	3942
FP	108
FN	65
TN	1885

Harnessing the power of machine learning, we developed an advanced malware detection system that integrates multiple models to enhance accuracy and robustness. By leveraging Grid Search CV, we fine-tuned various algorithms and selected the most effective ones for a stacking ensemble, ensuring a more generalized and resilient detection framework. XGBoost (XG) : A gradient-boosting algorithm known for its scalability and efficiency, handling missing values natively and reducing overfitting with L1/L2 regularization. Its ability to process sparse data makes it highly effective for cybersecurity appli- cations. CatBoost (Cat) : A boosting method optimized for categorical features, reducing data preprocessing needs while improving performance on imbalanced datasets. With ordered boosting, it minimizes target leakage and enhances real-world applicability. LightGBM (Light) : A histogram-based gradi- ent boosting framework that efficiently handles large-scale datasets. It achieves superior speed and memory efficiency by using leaf-wise tree growth, making it ideal for real-time threat detection.

Gradient Boosting (Grad) : A sequential, tree- based algorithm that focuses on minimizing residual errors, enhancing model accuracy through adaptive learning. It is particularly useful for capturing complex patterns in malware behavior. Extra Trees : An ensemble learning method that increases randomness in decision trees by selecting split points arbitrarily, reducing variance and improving robustness against adversarial attacks.

D. Stacking Ensemble

In addition to enhancing model performance, we also applied a stacking ensemble approach. This approach uses XGBoost, CatBoost, LightGBM, Gradient Boosting, and Extra Trees' predictions as input to a meta-learner. By learning an optimal-weighted ensemble of the models, the stacking ensemble overcomes each model's failure and enhances overall detection accuracy, making for a more adaptive and robust malware detector.

E. Training and Testing

The dataset was split in the proportion of 80:20, reserving 80% for training and 20% for testing, it ensures facilitating overall assessment of the performance of the model. The training subset was utilized to train the machine learning algorithms in a way that they could identify intricate patterns and relationships in the data. In order to avoid overfitting and enhance the ability of the models to generalize, Grid Search Cross-Validation was utilized during training to optimize hy- perparameters. The models were tested on the unseen 20% of the data set after training to evaluate their predictive accuracy, stability, and ability to detect malware. This split provided a balanced evaluation, reasserting the ability of the models to perform in real-world cybersecurity settings.

F. Model Deployment

For deployment, the trained stacking ensemble model was integrated into a Streamlit application, offering an interactive and user-friendly interface for real-time malware detection. The model was serialized using Joblib for efficient storage and fast inference. Users could input application permissions, and the model would analyze them to determine potential malware risks. The app provided instant feedback, highlighting suspicious permissions that could indicate malicious intent. With its lightweight and efficient design, the application en- sured seamless performance across various platforms, making malware detection more accessible and effective.

G. Key Features

The proposed Artificial Intelligence-based Android Malware Detection System is designed to efficiently identify malicious applications depending on the permissions they make requests for. The system is highly precise and convenient to oper- ate, thus it can be utilized by non-experts and experts in the cybersecurity department. The following are the primary characteristics of the project:

1) High Accuracy and Robust Performance – The system achieves an impressive 97.15
2) Lightweight and Efficient – The model uses only 46 selected permission features, reducing computational overhead and making it a lightweight yet effective solution. Feature engineering ensures that only the most relevant permissions are considered, eliminating unnecessary or redundant features.
3) User-Friendly Interface – The application is deployed on Streamlit Cloud, providing an intuitive and easy-to-use interface. Users simply input the permissions requested by an Android app from a predefined list, and the system instantly predicts whether the app is malicious or safe.
4) Advanced Feature Selection Using SHAP – SHAP (SHapley Additive exPlanations) is used to analyze feature importance, ensuring that only the most significant permissions contribute to the model's decision-making. This enhances accuracy and model interpretability.
5) Stacking Ensemble Model for Improved Detection – The system employs a stacking-based ensemble learning approach, combining multiple machine learning models to enhance gen- eralization and minimize bias. This leads to superior detection capabilities compared to single-model approaches.
6) Scalable and Deployable – The application is designed for real-world use, with deployment on Streamlit Cloud mak- ing it accessible to users worldwide. Its lightweight nature ensures it can function on various devices without requiring high-end hardware.
7) Preprocessed and Optimized Dataset – The dataset, sourced from Kaggle and other online repositories, is thor- oughly cleaned by replacing missing values with the mode and removing string values, resulting in a completely binary dataset. This ensures consistency and enhances the model's efficiency.
8) Future Scope for Improvements – The model can

be extended further to incorporate deep learning models, dynamic analysis methods, and real-time monitoring capabilities for even improved malware detection in the future.

VI. RESULTS

The result of our malware detection system proved that the stacking ensemble model performed far better than single classifiers with the highest accuracy, precision, and recall. By combining the models XGBoost, CatBoost, LightGBM, Gradient Boosting, and Extra Trees, the stacking model could identify intricate patterns in the data, minimizing false pos- itives and false negatives. The ensemble method capitalized on the strength of the constituent models, thereby being more robust and providing better generalization in actual use.The stacking model achieved a very high test accuracy of 97.15% against standalone models such as XGBoost (97%), CatBoost (97%), LightGBM (97%), Gradient Boosting (97.02%), and Extra Trees (97.12%). It also achieved the highest recall rate of 98.2% and precision rate of 97.55%, thus the most balanced and most accurate approach to malware detection.

One of the most critical measures in classification problems, the ROC AUC measure, also reaffirmed the performance of the stacking model with an impressive 99.66%, much better than that of individual models. This is evidence of its great proficiency in differentiating between benign and malicious software with very few errors. The stacking model identified 3,935 true positives (TP) and kept the false positives (FP) to just 99 and only 72 false negatives (FN), testifying to its high efficiency in malware detection with high accuracy.In contrast, while individual models performed well, they had slightly higher misclassification rates. XGBoost achieved 98% precision but had 90 false negatives, potentially allowing more malware to slip through. CatBoost showed the best recall at 99%, detecting more malware cases but with a slightly higher false positive count of 96. Gradient Boosting and Extra Trees performed competitively, but their lower ROC AUC scores (96.39% and 96.48%, respectively) indicated slightly weaker distinction capabilities compared to the stacking model.These results emphasize the effectiveness of the stacking ensemble approach, which not only improves individual model weaknesses but also enhances the overall malware detection performance. The combination of multiple models ensured a more adaptive, precise, and reliable cybersecurity solution, making the stacking model a highly effective tool for detecting malicious applications based on their permissions

Fig. 1: Web Interface for Android Malware Detection

Fig. 2: Detecting Malware through Permissions

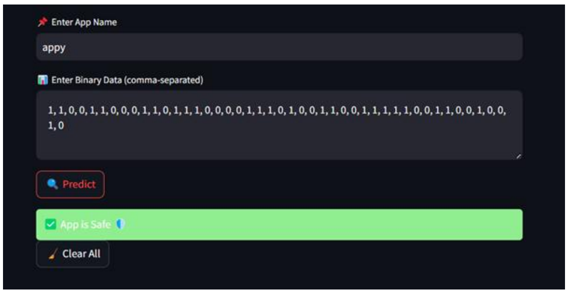

Fig. 3: Classifying App as Benign

VII. CONCLUSION

The AI-based Android Malware Detection System intro- duces an innovative and highly effective approach to identifying malicious applications using permission-based analysis. By leveraging machine learning, feature engineering, and a stacking ensemble model, the system achieves remarkable ac- curacy, with 97.15% test accuracy and 99% training accuracy, ensuring a robust and reliable detection mechanism. Unlike traditional detection techniques that rely on signature-based methods, which struggle against zero-day threats, or behavior- based models, which require high computational resources, our system provides a lightweight, scalable, and efficient alternative.One of the key advantages of this system is its ability to function using only 46 optimized permission-based features, which were carefully selected using SHAP-based feature selection to eliminate unnecessary attributes and enhance model performance. The entire dataset was preprocessed by replacing missing values with the most frequently occurring values (mode) and ensuring that all features remained in a binary format, making the model simple yet highly effective. The system's ability to generalize well across diverse malware samples makes it suitable for real-world cybersecurity applications.

Furthermore, the user-friendly interface, deployed on Streamlit Cloud, allows users to easily input permissions from a predefined list and instantly receive a malware prediction. This ease of access ensures that the system can be used by security researchers, app developers, and even general users to assess application safety before installation. The solution is lightweight and does not require deep system-level access, making it a practical choice for integration into app stores, security tools, and enterprise cybersecurity frameworks.In the next phase, the project can be upgraded by enabling real-time scan capabilities, incorporating deep learning technologies such as CNNs or RNNs to improve pattern analysis, and incorporating a new database of malware for handling new risks. Furthermore, the integration of explainability features can help in informing users that why an app is detected as malware, thus establishing trust among users in AI-driven security offerings.

Lastly, the project provides a solid platform for permission- based malware detection with a high-precision, scalable, and computationally efficient solution compared to existing approaches. As the Android platform grows, this system can be a significant contribution towards improving mobile security and protecting users from cyber attacks.

VIII. FUTURE SCOPE

The suggested AI-driven Android malware detection system has shown high accuracy and efficiency in detecting malicious apps based on permissions. Nevertheless, there are some areas of improvement. Future research can investigate deep learning methods, including CNNs, RNNs, and transformer models, to learn complex permission usage patterns autonomously. Reinforcement learning can also be employed to facilitate self-adaptive malware detection, enabling the system to learn and counter novel threats. Another direction is the hybrid method, integrating permission-based analysis with behavioral analysis, system call monitoring, and network activity tracking to enhance detection accuracy. Moreover, real-time detection capabilities can be implemented, providing on-device scanning through Edge AI and federated learning, preserving user privacy while boosting security.The system is also supported by other operating systems, for instance, Windows and iOS, thus forming a cross-platform security platform. Further, Explainable AI (XAI) techniques, for instance, SHAP and LIME, can be employed to generate interpretable explanations for malware classification, thus enhancing the credibility of the system among users and security professionals. As cybersecurity and AI keep advancing, the system has immense potential to serve as an effective real-time malware detection system, thus enhancing mobile security across the world.

REFERENCES

[1] Yerima, S. Y., Sezer, S., & Muttik, I. (2015). High Accuracy Android Malware Detection Using Ensemble Learning. IET Information Security, 9(6), 313-320.

[2] Lakshmanarao, A., & Shashi, M. (2022). An Efficient Android Malware Detection Framework with Stacking Ensemble Model. IJETT, 70(4), 294- 302.

[3] Wang, J., Jing, Q., & Gao, J. (2022). MFDroid: A Stacking Ensemble Framework for Android Malware Detection. Sensors, 22(7), 2597.

[4] Lu, T., Du, Y., et al. (2020). Android Malware Detection Based on a Hybrid Deep Learning Model. Security and Communication Networks.

[5] Chakraborty, T., Pierazzi, F., & Subrahmanian, V. S. (2020). EC2: En- semble Clustering and Classification for Android Malware. IEEE TDSC, 17(2), 262–277.

[6] Junhui, Y., et al. (2021). Android Malware Detection Using Ensemble Learning on Sensitive APIs. In AISC (Vol. 1242).