Received 05 December 2025, accepted 24 April 2026, date of publication 30 June 2026
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.25.2.4657

Machine Learning Analysis of Potential Mobile Apps Threats on the Basis of Permissions

TETIANA HOVORUSHCHENKO, OLGA PAVLOVA, VITALII ALEKSEIKO
Computer Engineering and Information Systems Department, Khmelnytskyi National University, 29016, Khmelnytskyi, Ukraine

Corresponding author: Vitalii Alekseiko (e-mail: vitalii.alekseiko@khmnu.edu.ua).

ABSTRACT Nowadays, the use of mobile applications is extremely common. The number of new applications is constantly increasing, and accordingly, because of this, it is becoming increasingly difficult to determine the level of danger of a particular application. The issue of privacy and data security is especially acute, particularly because of the permissions that apps request. Such permissions can give applications access to sensitive user information and system resources, which, if misused, can lead to security risks. This research focuses on mobile application threat analysis using machine learning (ML) with a focus on the Android Permission dataset. By examining how different apps use and potentially abuse permissions, this research aims to identify high-risk apps that pose a serious threat to user privacy and cybersecurity. The importance of permissions as a vector for understanding mobile app security is undeniable, and a permission-based ML analysis approach can improve current app verification processes. This methodology can be further extended to include dynamic analysis where application behavior is monitored in real-time, providing an even more robust assessment of security risks in mobile environments.

KEYWORDS mobile application; permission; threat; machine learning; cybersecurity.

I. INTRODUCTION

Nowadays, information security is crucial for several reasons, especially in digital world where data is stored and transmitted online. In particular information such as personal identification numbers, health records, financial details and even browsing history can reveal a lot about personality.

Many users are seriously concerned that software products can abuse dangerous permissions to collect personal information [1]. For example, apps with access to location data can potentially track users' movements [2], and those with access to contacts can share users' social circles. Apps with access to the microphone or camera can potentially record conversations or take photos without the user's knowledge.

Machine learning techniques are increasingly being used to detect potentially malicious mobile apps by analyzing permission requests [3]. Statistical methods and supervised learning models are commonly used to identify significant relationships between permissions and malicious behavior. While such approaches are technically efficient, they also raise important legal and ethical debates related to data protection, privacy, accountability, fairness, and regulatory compliance.

Thus, the study of the influence of permissions on the classification of mobile applications is relevant.

II. DOMAIN ANALYSIS

It is worth noting that permissions to access sensitive information [4] pose a significant risk, as it can lead to information leakage, abuse, identity theft or surveillance. These permissions include full access to the Internet, the ability to read and send SMS, make calls, and view location. After gaining such access, the application can steal sensitive data or perform unwanted actions without the user's knowledge.

The study [5] presents a comprehensive systematic mapping review that focused on the application of the machine learning approach for the mitigation of security threats and attacks in the BYOD environment by highlighting the current trends in the existing studies.

In [6] the authors demonstrate that when concept drift is addressed, permissions can generate long-lasting and effective malware detection systems.

In paper [7] Machine Learning-Assisted Secure Mobile Electronic Payment Framework (ML-SMEPF) is proposed to detect the presence of malware, authentication issues, and fraud detection in mobile transactions.

In paper [8], in the first part, the authors have proposed a technique to check for correlation between features and classify using a supervised learning approach to avoid Multicollinearity problem is one of the drawbacks in the existing system.

In study [9], a supervised learning technique that demonstrates promising results in Android malware detection is presented.

The objective of [10] is to investigate the correlation between machine learning (ML) and cyber threat intelligence (CTI) to improve cybersecurity strategies.

In the research [11] the authors provide an extensive view of machine learning algorithms, emphasizing how they can be employed for intelligent data analysis and automation in cybersecurity through their potential to extract valuable insights from cyber data.

The paper [12] investigates the delicate balance between utilizing data for business intelligence and upholding ethical standards to protect consumer privacy.

The work [13] proposes a precise dynamic analysis approach to identify a slew of malicious attacks. The proposed method focus on behavioral analysis of malware that requires reconstructing the behavior of Android malware.

This study [14] starts by examining the unique architecture of 5G and its inherent vulnerabilities, contrasting them with emerging threat vectors.

In [15] we propose the methodology for determination legal responsibility or penalty for the cybersecurity breach. In [16] we develop the method for increasing the security of smart parking cyber-physical system, taking into account its software part. The paper [17] is focused on evaluating mutual correlations and weights of factors and consequences of mobile application insecurity.

Therefore, violation of permissions poses a significant threat to the security of both devices and users themselves. The conducted research aims to establish the most relevant permissions, on the basis of which it would be possible to classify software products as safe or malicious and to help minimize malicious actions performed through mobile applications.

In the initial stages, permission models were more generalized. Software products accessed a wide range of data with minimal user control, which in turn increased the risks of abuse and provided opportunities for attackers to breach security.

Although this study focuses on devices running the Android operating system, most modern OSes, particularly iOS, allow users to control permissions. Thus, the user can enable access to certain functionality only while using the program, which significantly reduces risks and prevents disclosure of information [18].

Unauthorized use of permissions is a very common malware attack vector [19]. Malware often requires excessive permissions, which can be used to steal personal data, track user activity, or even perform remote attacks on the device [20]. For example, malware that gains access to sensitive permissions can send data to external servers without the user's knowledge.

Therefore, permission analysis can be used to detect malware. Applying machine learning models will help identify unusual permission requests as an indicator of malicious behavior [21].

III. METHODOLOGY

The problem of detecting malware based on application permissions can be formulated as a binary classification task. Its goal is to predict whether a mobile application is safe or malicious.

The dataset for the task should include the following parameters:

– identifier of the mobile application;

– the name of the mobile application;

– columns with permissions, which are treated as binary functions: presence of permission is denoted as 1 and its absence as 0;

– a target label or class indicating that the application is safe (0) or malicious (1).

Since the number of permissions used by different mobile applications is very large, analyzing all permissions for classification is impractical. Therefore, the purpose of the research is to determine the most relevant features, on the basis of which the analysis will be carried out. Thus, the task is reduced to finding the parameters that exert the greatest influence on the binary classification.

During the research, methods of analysis, synthesis, comparison, formalization, and mathematical statistics were used.

An important step is to check the balance of the dataset and, if necessary, adjust the data to avoid class imbalance. Special techniques such as resampling or adjusting class weights may be used for this purpose.

The study was conducted using the Python programming language and its libraries. The matplotlib [22] library was used for visualization.

A. DATASET STRUCTURE

The article uses the Android Permission Dataset [23], which contains information about mobile applications and the permissions they request. All permissions are marked as safe (S) or dangerous (D). In addition, there is information about whether each application belongs to the class of benign or malware.

The dataset contains data on 9,999 benign and 20,000 malicious applications.

The analysis of the dataset (Fig. 1) showed that the largest number of applications belongs to the field of entertainment (more than 2500). There are also a large number of applications in the areas of travel, books and puzzles.

B. STATISTICAL PERMISSIONS ANALYSIS

It was determined that the most popular permissions (Fig. 2) required by applications are permissions related to Internet access. In particular, almost all applications require full access to the Internet. This permit is classified as dangerous. The vast majority of applications ask for permission to view the network status (belongs to the safe ones). Other popular permissions that can pose a potential threat to devices are access to storage, including USB ports and the ability to manage content on an SD card.

To evaluate the difference in permissions between two classes of mobile applications: benign (class 0) and malware (class 1), we will use the chi-square test for statistical evaluation of hypotheses. To do this, we will form the null (H0) and alternative (H1) hypotheses:

– H0: There is no difference in the distribution of permissions between harmless and malicious applications;

– H1: There is a difference in permission distribution between benign and malicious applications.

To calculate χ2 value, we will use the formula:

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}, \quad (1)$$

– H1: There is a difference in permission distribution between benign and malicious applications.

where:

O_i – observed frequency;

E_i – expected frequency.

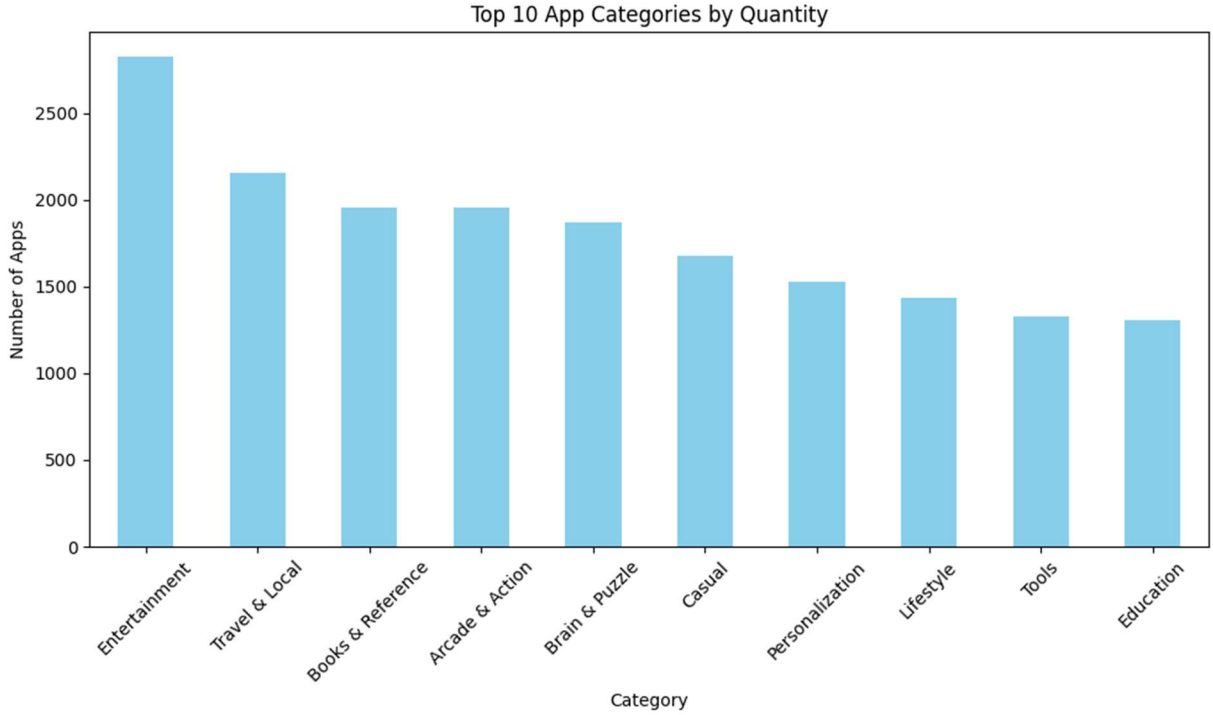


Figure 1. Top 10 applications' categories by quantity.

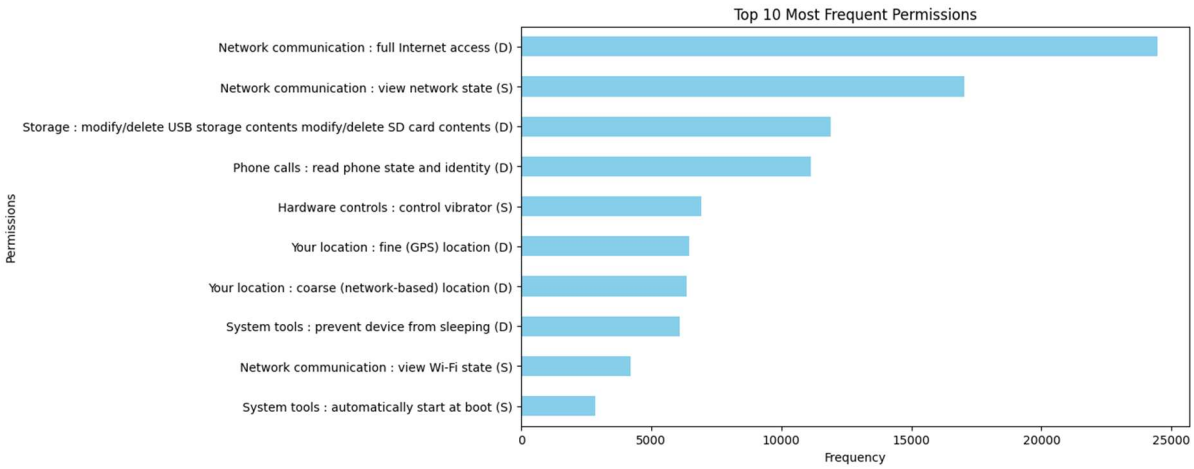


Figure 2. Top 10 most frequent permissions.

Figure 3 presents the results of the calculations. Permissions related to access to the Internet, browser history, phone calls and location are defined as key.

C. MACHINE LEARNING-BASED ANALYSIS

Since the dataset contains a large number of features (178), it is advisable to conduct an analysis of the importance of features using machine learning techniques. For feature selection, a comprehensive feature engineering was performed using the Gini index, permutation importance and Boruta algorithm.

The Gini index is most commonly used with decision tree models, such as the random forest, which was chosen as the main model in this study. In classification trees, the Gini index measures the impurity of a node, that is, how mixed the class labels are at that node [24]. When building a tree, the algorithm selects the features that reduce the impurity the most when splitting the data. For a dataset with class probabilities p_1, p_2, …, p_k the Gini impurity is defined as $1 - \sum_{i=1}^{k} p_i^2$. Because each permission in the dataset is represented as a binary attribute indicating whether a permission is requested, during training, the decision tree evaluates how well the partitioning of the dataset based on each permission separates safe applications from malicious ones. The overall reduction in impurity Gini caused by partitions that include a particular

permission is accumulated across all trees in the forest. This aggregated value represents the Gini importance, or average reduction in impurity, of that permission. Permissions with higher values are considered to be more influential in the model's classification decisions. However, the Gini-based measure can be biased toward attributes with higher variability or toward permissions that appear more frequently in the dataset.

Permutation importance is a model-independent method for estimating the contribution of each attribute to the predictive performance of a trained model. The method works after the model has already been trained. First, the baseline performance of the model is measured on a validation dataset. Then, the values of a single feature, such as a particular Android permission, are randomly permuted across the dataset, thereby breaking the relationship between that feature and the target variable. The model is re-evaluated on the modified dataset. If the model performance decreases significantly, this indicates that the permuted permission contained important predictive information. If the performance remains almost unchanged, the permission probably has little effect on the model. In Android malware analysis, the importance of permutations provides a straightforward interpretation of how much each permission affects the predictions, regardless of the type of model. One advantage of this method is that it reflects the importance of features in the context of the trained model and the distribution of the data, although it can be computationally intensive when many features are present [25].

Boruta algorithm is a method for selecting all relevant features, built on Random Forest models. Its goal is not only to find the minimum subset of features that optimizes predictive performance, but also to identify all features that carry statistically significant information about the target variable. In Android permission analysis, the algorithm starts by augmenting the dataset with "shadow features". These are artificial features created by randomly permuting the values of the original permissions, thereby destroying any real relationship with the target variable [26].

A random forest model is then trained on the combined dataset of real and shadow features, and feature importance scores are calculated. For each original permission, its importance is compared to the highest importance achieved among the shadow features. If the importance of the real permission is significantly higher than that of the shadow features, the permission is marked as important. If it is significantly lower, it is considered unimportant. Otherwise, the algorithm remains uncertain and repeats the process with additional iterations. In the context of Android malware detection, Boruta helps identify all permissions that contain meaningful information about malicious behavior, even if their individual contribution is relatively small. This makes the method particularly useful for exploratory analysis and for understanding the broader set of potentially risky permissions requested by applications.

Together, these three methods provide additional perspectives on the relevance of Android permissions in machine learning models. The Gini index provides a measure of the intrinsic importance obtained during tree construction, the importance of permutations assesses the impact of each permission on predictive performance, and Boruta algorithm offers a statistically sound method for identifying all relevant permissions in a dataset.

IV. CASE STUDY & RESULTS

Random Forest (RF) is an efficient method that offers several advantages for detecting relevant features in binary classification tasks.

RF can provide an importance score for each attribute, allowing to determine which permissions contribute most to predicting whether an application is safe or malicious. The model evaluates the impact of each feature by looking at how much splitting that feature improves the overall forecast.

Using this approach is particularly useful when working with large datasets with many features. This allows to reduce the dimensionality by focusing on the most important functions. RF handles high-dimensional data very well without requiring extensive data preprocessing. RF automatically reduces the weight of unimportant features during the learning process. An important advantage of this approach is that random forests have the ability to capture complex, non-linear relationships between features and the target variable. For example, certain combinations of permissions may indicate malware that would not be obvious using linear methods. RF determines not only the importance of an individual function, but also how functions interact with each other, revealing complex patterns useful for function identification. Due to its ensemble nature (building multiple decision trees and averaging their predictions), RF is less prone to overfitting compared to individual decision trees, even when dealing with many features. This robustness helps identify truly important features because the model does not overestimate noise or irrelevant features.

RF can handle class imbalance relatively well by using methods such as class weights or by adjusting the sampling strategy for the trees. Also, RF is relatively robust to noisy functions due to its ensemble approach. Even if some features are irrelevant or create noise, their impact will be minimized in the forest.

Thus, RF is an excellent tool for feature identification in binary classification tasks due to its ability to handle high-dimensional data, nonlinear relationships, and noise, while providing interpretable feature importance ratings. These advantages make it ideal for tasks such as malware detection, where defining the correct permissions (features) is critical to the model's success.

Figure 4 shows the most important features due to RF.

Top-20 permissions by Random Forest Gini importance and permutation importance are presented in Figure 5. ROC-AUC results for different numbers of selected permissions is presented in Figure 6.

The results of the feature engineering evaluation are summarized in Table 1 and Figure 7.

Table 1. Normalized feature importance for permissions.

Method	ROC-AUC	CV ROC-AUC	Number of Features
All Features	0.8016	0.8017	178
SelectKBest (80)	0.7986	0.7998	80
Boruta	0.7305	0.7375	1

V. DISCUSSION

Although the specifics of a particular application may require the use of certain permissions marked as unsafe without harming the system, the analysis of the set of permissions allows to make

some predictions about the classification of the application as malicious or safe. Figure 8 represents 10 of the most important permissions impacting on apps' classification due to χ^2 and Random Forest. Normalized feature importance for permissions presented in Table 2.

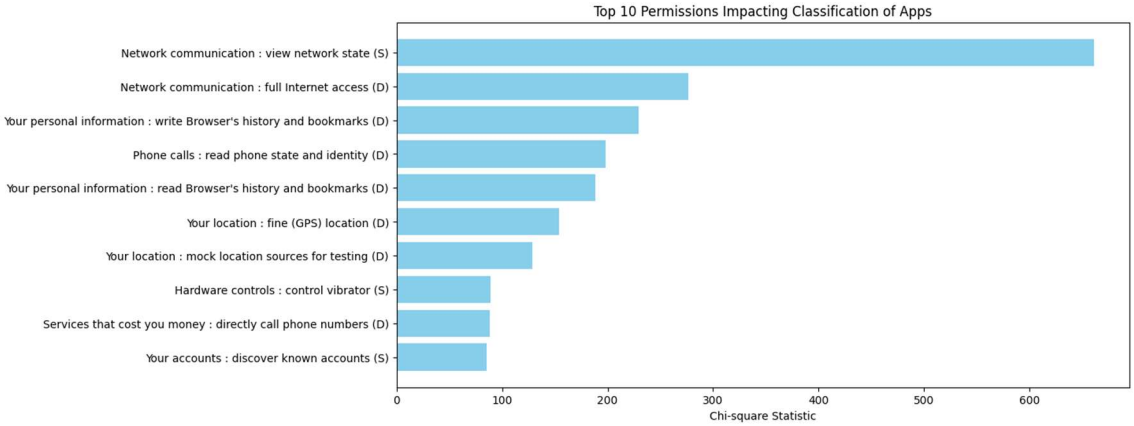


Figure 3. Top 10 permissions impacting on apps' classification due to $\chi 2$ test.

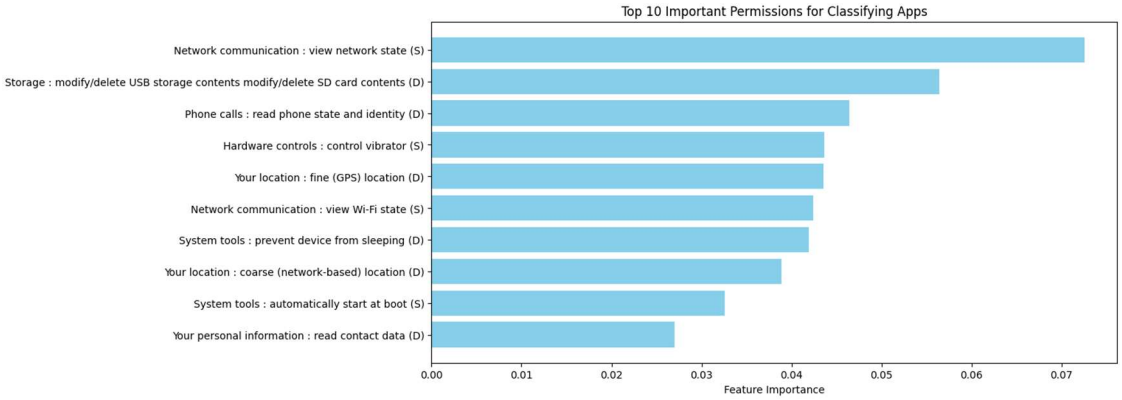
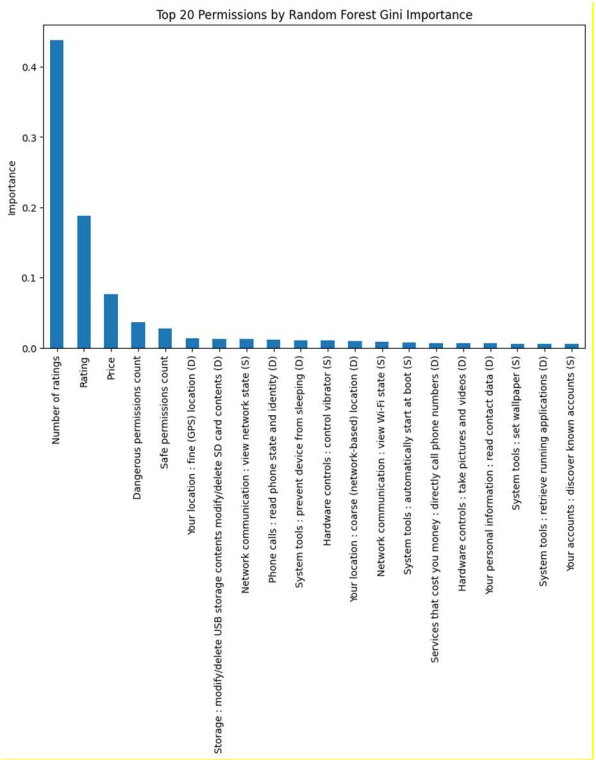
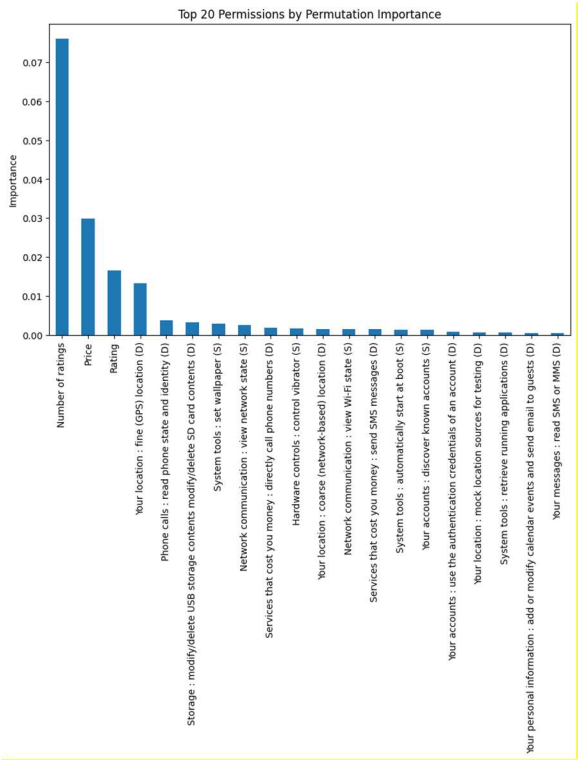


Figure 4. Top 10 permissions impacting on apps' classification due to Random Forest.



by Gini index



by permutation importance

Figure 5. Top-20 permissions.

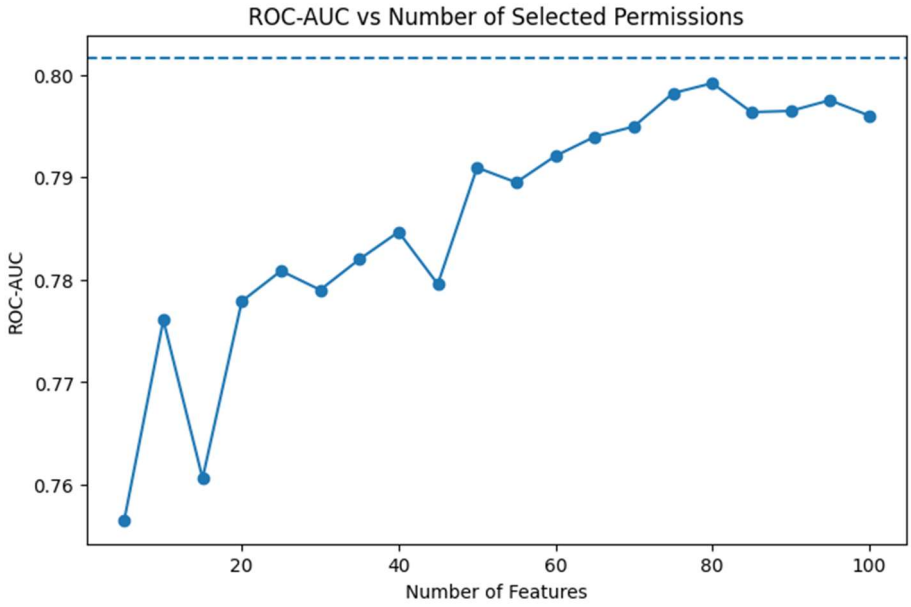


Figure 6. ROC-AUC results for different numbers of selected permissions.

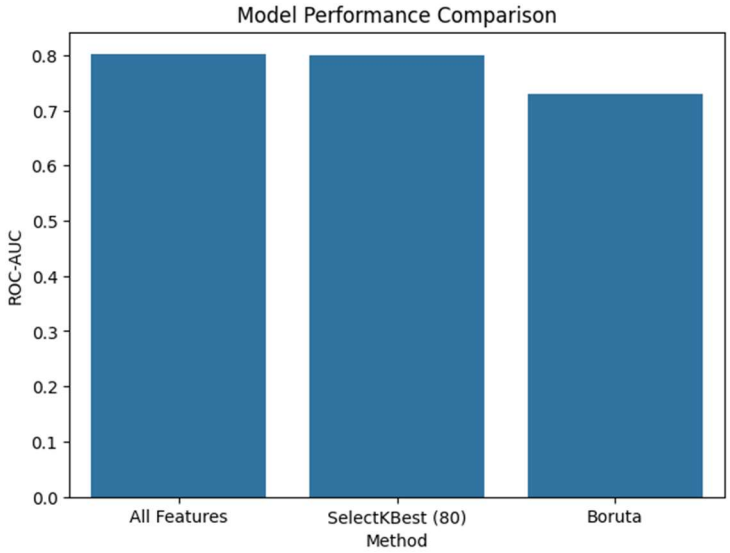


Figure 7. The results of the feature engineering evaluation.

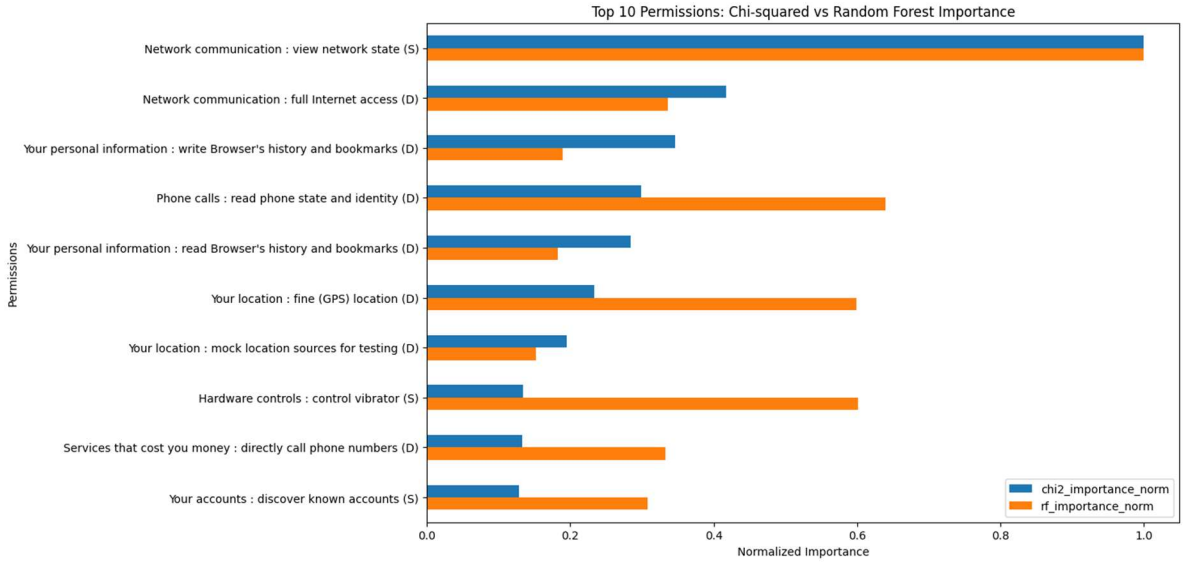


Figure 8. Top 10 permissions impacting on apps' classification due to $\chi 2$ and Random Forest.

Table 1. Normalized feature importance for permissions.

Permission category	Permission	Permission status	Normalized feature importance	
			χ^2	Random Forest
Network communication	View network state	Safe	1.0	1.0
	Full Internet access	Dangerous	0.417594	0.335793
Personal information	Write Browser's history and bookmarks	Dangerous	0.346781	0.189889
	Read Browser's history and bookmarks	Dangerous	0.284533	0.182284
Phone calls	Read phone state and identity	Dangerous	0.298957	0.639545
Location	Fine (GPS) location	Dangerous	0.232951	0.599754
	Mock location sources for testing	Safe	0.194531	0.152386
Hardware controls	Control vibrator	Safe	0.134174	0.601407
Services that cost money	Directly call phone numbers	Dangerous	0.133410	0.332492
Accounts	Discover known accounts	Safe	0.128554	0.307492

Permissions are typically displayed to users when they install software or perform certain actions, but users may inadvertently grant excessive permissions due to a lack of understanding or convenience, leading to potential security risks.

A constant challenge is the balance between security and usability. Some programs require special permissions to work effectively, and denying these permissions may impair the functionality of the program. At the same time, granting unnecessary permissions can create a security threat.

The initial dataset includes 178 features, and the ROC-AUC result at the level of 0.8 indicates the presence of a large number of weakly useful features. This increases the complexity of the model, makes it worse interpreted, and also increases the risks of overfitting.

Selecting the k best features allows us to evaluate the performance of the model when reducing the number of features. The model demonstrated the best result for 80 features. The ROC-AUC result is above 0.79, i.e. the difference in performance was less than 2% when the dimensionality was reduced by more than 2.2 times.

The Boruta method selected only one feature, which indicates that the data does not have strong individual predictors, and the signal is distributed among many variables

The latest versions of Android and iOS use runtime permission models that require apps to ask for permissions when using the app, not during installation. This approach gives users better control and reduces the chance of security vulnerabilities due to unused or unnecessary permissions.

It is common knowledge that apps from untrusted sources or third-party app stores are more likely to abuse permissions. Official stores such as Google Play and the Apple App Store usually check security, but even apps from these platforms can abuse permissions.

Another consideration is how app permissions change over time. That is, an application may initially request only necessary permissions, but subsequent updates may request additional, more dangerous permissions without clearly notifying the user, potentially leading to increased security risks.

VI. ETHICAL AND LEGAL ASPECTS

Mobile app threat analysis may involve the processing of user-related data, including device identifiers, usage logs, and behavioral metadata. In such cases, applicable data protection regulations, such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA), must be complied with.

According to the GDPR, personal data is broadly defined and can include device identifiers, IP addresses, or behavioral patterns associated with identified individuals. If machine learning models are trained on datasets containing such data, several legal obligations arise:

– Legal basis for processing, which can be either legitimate interest;

– The principle of data minimization;

– Purpose limitation;

– Storage limitation;

– Security of processing;

– Transparency and information rights.

When implementing statistical testing and machine learning models into recommender systems, the decisions they make will directly impact the developers or users. An example of such an impact is the removal of applications or restriction of access. Thus, Article 22 GDPR on automated decision-making may apply.

It should be noted that even when the data set consists solely of application permissions and metadata without direct personal identifiers, there may be risks of re-identification, especially in combination with behavioral logs.

Security research related to app permissions may intersect with cybersecurity legislation and digital platform regulations. In the European context, relevant instruments include:

– Directive on Security of Network and Information Systems (NIS Directive);

– Digital Services Act (DSA)

Entities that conduct ML-based threat analysis may be classified as digital service providers or security service operators, thereby assuming obligations regarding risk management, reporting, and transparency.

In the United States, federal and state cybersecurity laws and industry regulations may apply depending on the nature of the data processed.

Additionally, analyzing mobile apps for permissions may violate intellectual property laws. While permission metadata is usually publicly available through app manifests, deeper static or dynamic analysis may be considered reverse engineering.

Legal permissibility depends on the jurisdiction and purpose. In the EU, limited reverse engineering is permitted for

interoperability and security research under certain conditions. In the US, fair use doctrine and contractual terms may affect legality.

Therefore, when conducting such research, it is necessary to ensure compliance with the platform's terms of service and to avoid unauthorized access in accordance with computer misuse laws.

If a machine learning model incorrectly classifies an app as malicious, i.e. a false positive, this can cause reputational and financial damage to developers. Conversely, false negatives can expose users to harm.

Legal risks include:

– Defamation claims;

– Unfair competition claims;

– Contractual liability;

– Neglect (if security services fail to meet reasonable standards).

According to the GDPR, controllers must implement appropriate technical and organizational measures and be able to demonstrate compliance (accountability principle).

The methods considered in the study should also be evaluated from an ethical perspective. The chi-square test identifies statistically significant associations between permissions and malware classification. However, statistical significance does not imply causality. Permissions that are commonly used in certain categories of legitimate applications, such as status, navigation, may be overrepresented in malicious datasets, leading to biased feature selection.

Random forest models, while robust and interpretable compared to some other machine learning methods, can still propagate bias across datasets. Ethical issues arise if models rely on correlated but ethically irrelevant attributes.

Thus, ensuring representative and balanced training datasets is critical.

From an ethical perspective, data collection should be proportionate to the security objective. If permission-level metadata is sufficient to detect threats, collecting detailed behavioral logs may violate the principle of proportionality.

Applying the least intrusive method consistent with effective detection is both ethical and data protection.

A very important point is the risk of dual use, as methods designed to detect malware can also be adapted to evade detection. Responsible disclosure practices and controlled publication of sensitive methodological details are important ethical safeguards.

It is also important for developers to ensure informed consent and user autonomy if real user data is used in training or validation:

– Users should be informed about the purposes of data processing.

– Consent should be voluntary, specific, informed and unambiguous (in line with GDPR standards).

– Opt-out mechanisms should be available where appropriate.

Let's take a closer look at the methods used in the study from an ethical and legal perspective.

The chi-square test is generally considered to be low-risk from a legal perspective because:

– It is a statistical test of association;

– It does not inherently automate decisions.

However, ethical issues arise when statistically significant associations are interpreted as conclusive evidence of malicious intent. Overreliance on statistical thresholds without contextual analysis can lead to unjustified classifications.

Random forest models:

– Automate risk assessment;

– Aggregate multiple decision trees;

– Provide probabilistic results.

If used to automatically block applications, they may constitute automated decision-making systems under data protection law. Human oversight mechanisms should be integrated, especially for high-impact decisions.

Auditability is an important ethical requirement. Logging model decisions and supporting reproducible training pipelines supports accountability.

To ensure legal and ethical threat analysis based on machine learning, organizations should implement:

– Data Protection Impact Assessments (DPIAs) where appropriate;

– Privacy and security by design principles;

– Model documentation and reproducibility standards;

– Independent auditing and bias testing;

– Clear appeal mechanisms for affected developers;

– Adherence to platform terms and cybersecurity policies;

– Secure data storage and encryption.

VII. CONCLUSIONS

The growing ubiquity of mobile applications has introduced numerous security and privacy challenges, primarily stemming from the permissions these apps require. In this study, we have demonstrated how machine learning techniques can be effectively utilized to analyze the potential threats posed by mobile applications based on their permission requests. By employing classification models, we identified key permission patterns that correlate with malicious behavior, allowing for the prediction and mitigation of security risks. The permissions requested by mobile applications play a key role in the security of the software product and the potential impact on the device. Permissions determine what an app can access on a device, such as controlling Internet access, call control, camera usage, device location, access to contacts, memory, sensors, and more.

In summary, mobile device permissions have a significant impact on app security by controlling access to device functionality. Proper permission management, user awareness, and robust app validation are critical to minimizing security vulnerabilities related to mobile permissions. Understanding which permissions are highly predictive can help security experts and developers build more secure applications.

The conducted research confirmed hypothesis H1. Thus, determining whether a software product is safe or not can be done using permission analysis.

As mobile apps increasingly collect personal data, user concerns about privacy and security will continue to shape expectations and influence app downloads.

Developers can address some of the fears users have by providing clear contextual explanations for each permission request. Requests should be based only on the basic functions of the program. It is a good idea to provide users with control over permissions, allowing them to grant or revoke access at their discretion.

References

[1] I. Pekaric, C. Sauerwein, S. Laichner, and R. Breu, "How do mobile applications enhance security? An exploratory analysis of use cases and provided information," *Proceedings of the 2025 ACM Southeast*

Conference (ACMSE 2025). Association for Computing Machinery, New York, NY, USA, 2025, pp. 114–123. https://doi.org/10.1145/3696673.3723067.

[2] H. I. Maseeh, C. Jebarajakirthy, A. Sivapalan, M. Ross, and M. Rehman, "Understanding smartphone users' app usage with restricted permissions," *Information Technology & People*, vol. 38, no. 2, pp. 1045–1088, 2025. https://doi.org/10.1108/ITP-03-2022-0200.

[3] A. R. Nasab, M. Dashti, M. Shahin, M. Zahedi, H. Khalajzadeh, C. Arora, and P. Liang, "Fairness concerns in app reviews: A study on AI-based mobile apps," *ACM Trans. Softw. Eng. Methodol*, vol. 34, issue 2, article 51, pp. 1-30, 2025. https://doi.org/10.1145/3690633.

[4] M. Tahaei, R. Abu-Salma, and R. Awais, "Stuck in the permissions with you: Developer & end-user perspectives on app permissions & their privacy ramifications," *Proceedings of the ACM 2023 CHI Conference on Human Factors in Computing Systems (CHI'23)*, article 168, 2023, pp. 1–24. https://doi.org/10.1145/3544548.3581060.

[5] C. I. Eke, A. A. Norman, & M. Mulenga, "Machine learning approach for detecting and combating bring your own device (BYOD) security threats and attacks: a systematic mapping review," *Artificial Intelligence Review*, vol. 56, issue 8, pp. 8815-8858, 2023. https://doi.org/10.1007/s10462-022-10382-3.

[6] A. Guerra-Manzanares, H. Bahsi, & M. Luckner, "Leveraging the first line of defense: A study on the evolution and usage of android security permissions for enhanced android malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 19, issue 1, pp. 65-96, 2023. https://doi.org/10.1007/s11416-022-00432-3.

[7] F. Wang, N. Yang, P. M. Shakeel, & V. Saravanan, "Machine learning for mobile network payment security evaluation system," *Transactions on Emerging Telecommunications Technologies*, vol. 35, issue 4, e4226, 2024. https://doi.org/10.1002/ett.4226.

[8] V. J. Raymond, R. J. R. Raj, & J. Retna, "Investigation of Android malware with machine learning classifiers using enhanced PCA algorithm," *Comput. Syst. Sci. Eng.*, vol. 44, issue 3, pp. 2147-2163, 2025. https://doi.org/10.32604/csse.2023.028227.

[9] A. Gómez, & A. Muñoz, "Deep learning-based attack detection and classification in Android devices," *Electronics*, vol. 12, issue 15, 3253, 2023. https://doi.org/10.3390/electronics12153253.

[10] I. Naseer, "Machine learning applications in cyber threat intelligence: A comprehensive review," *The Asian Bulletin of Big Data Management*, vol. 3, issue 2, pp. 190-200, 2023. https://doi.org/10.62019/abbdm.v3i2.85.

[11] I. H. Sarker, "Machine learning for intelligent data analysis and automation in cybersecurity: current and future prospects," *Annals of Data Science*, vol. 10, issue 6, pp. 1473-1498, 2023. https://doi.org/10.1007/s40745-022-00444-2.

[12] N. B. Alom, "A comprehensive analysis of customer behavior analytics, privacy concerns, and data protection regulations in the era of big data and machine learning," *International Journal of Applied Machine Learning and Computational Intelligence*, vol. 14, issue 5, pp. 21-40, 2024.

[13] P. Bhat, S. Behal, & K. Dutta, "A system call-based android malware detection approach with homogeneous & heterogeneous ensemble machine learning," *Computers & Security*, vol. 130, 103277, 2023. https://doi.org/10.1016/j.cose.2023.103277.

[14] H. N. Fakhouri, S. Alawadi, F. M. Awaysheh, I. B. Hani, M. Alkhalaileh, & F. Hamad, "A comprehensive study on the role of machine learning in 5G security: challenges, technologies, and solutions," *Electronics*, vol. 12, issue 22, 4604, 2023. https://doi.org/10.3390/electronics12224604.

[15] T. Hovorushchenko, A. Herts, A. Boyarchuk, & O. Pavlova, "System for determination of legal responsibility/penalty for a cybersecurity breach," *Proceedings of the ITTAP'2022*, 2022, pp. 233-240.

[16] T. Hovorushchenko, O. Pavlova, & M. Kostiuk, "Method of increasing the security of smart parking system," *Journal of Cyber Security and Mobility*, vol. 12, issue 3, pp. 297–314, 2023. https://doi.org/10.13052/jcsm2245-1439.123.3.

[17] E. Zaitseva, T. Hovorushchenko, O. Pavlova, Y. Voichur, "Identifying the mutual correlations and evaluating the weights of factors and consequences of mobile application insecurity," *Systems*, vol. 11, 242, 2023. https://doi.org/10.3390/systems11050242.

[18] R. Mayya, and S. Viswanathan, "Delaying informed consent: An empirical investigation of mobile apps' upgrade decisions," *Management Science*, vol. 71, issue 8, pp. 7113-7135, 2024. https://doi.org/10.1287/mnsc.2022.00334.

[19] R. Mohammadi, M. M. Hosseini, R. Bahrami, "Uncovering security vulnerabilities through multiplatform malware analysis," *Security and Privacy*, vol. 8, issue 1, e455, 2025. https://doi.org/10.1002/spy2.455.

[20] L. Wei, H. Huang, S.C. Cheung, K. Li, "How far are app secrets from being stolen? A case study on Android," *Empir Software Eng*, vol. 30, article 90, 2025. https://doi.org/10.1007/s10664-024-10607-9.

[21] J. Apoorva, "AI-driven malware behaviour classification and detection systems," *International Journal of Advanced Research in Computer Science and Engineering (IJARCSE)*, vol. 1, issue 3, pp. 16–24, 2025. https://ijarcse.org/index.php/ijarcse/article/view/69.

[22] *Matplotlib – Visualization with Python*, 2024. [Online]. Available at: https://matplotlib.org/.

[23] *Android Permission Dataset*, 2021, May 29, Kaggle. [Online]. Available at: https://www.kaggle.com/datasets/saurabhshahane/android-permission-dataset.

[24] N. H. Saeed, A. A. Hamza, M. A. Sobh, and A. M. Bahaa-Eldin, "Efficient feature ranked hybrid framework for android IoT malware detection," *Sci Rep*, vol. 16, 3726, 2026. https://doi.org/10.1038/s41598-026-35238-6.

[25] A. Kadir, S.K. Peddoju, "PacDroid: lightweight android malware detection using permissions and intent features," *Multimed Tools Appl*, vol. 84, pp. 32351–32379, 2025. https://doi.org/10.1007/s11042-024-20455-w.

[26] A. M. H. Othman, M. F. A. Razak, S. M. Asmara, and S. N. Ramli, "Artificial intelligence-driven detection of Android malware using machine learning techniques," *International Journal on Advanced Science Engineering and Information Technology*, vol. 15, issue 5, pp. 1656–1662, 2025. https://doi.org/10.18517/ijaseit.15.5.13405.

***TETIANA HOVORUSHCHENKO,** DrSc (Engineering), Dean of the Faculty of Information Technologies, Professor of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Ukraine. Research interests: Software quality assessment; Intelligent agents based on the ontological approach; Intelligent information systems and technologies.*

***OLGA PAVLOVA,** PhD, Associate professor of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Ukraine. Research interests: Agent-oriented information technology for assessing the initial stages of the software life cycle based on the ontological approach.*

***VITALII ALEKSEIKO,** PhD Student majoring in Information Systems and Technologies, Assistant Lecturer of Computer Engineering & Information Systems Department, Khmelnytskyi National University, Ukraine. Research interests: Using AI technologies in applied tasks, Machine Learning, Responsible use and ethical aspects of artificial intelligence*