International Journal of Anthropology and Archaeological Research

https://trpub.org/index.php/IJAAR

Vol 1, Issue 1, 2026

ANALYZING IRRIGATION QUALITY TO POTENTIALLY IMPROVE SUGARCANE YIELD USING AGRICULTURAL DATA ANALYTICS

[1]V.Sudha, [2]J.Hemalatha, [3]P. Devasudha

[1,2,3]Assistant Professor
[1,2]PG & Research Department of Computer Science
[1,2]St. Josephs College of Arts and Science, Cuddalore, Annamalai Unviersity, Tamilnadu.
[3]St. Martin's Engineering College, Secunderabad, Telangana, India

*Corresponding Author

E-mail : sudha@sjctnc.edu.in

ABSTRACT

Agriculture is the foundation of Indian economy. Sugarcane is one of the widely developed harvests in Tamilnadu. Big data are emerging précised and viable analytical tool in agricultural research field. This study paper facilitates the farmers to improve the crop yield. The object of the present study was to investigate the relationship between the chemical properties of water and sugarcane yield, by this means identifying the water parameters that determine the final productivity of the yield. The Sugarcane fields monitored during the cultivation for water samples and yield data collected annually. A random forest algorithm applied to investigate the influence of different water attributes on yield using by collected data, over the study period. The irrigation mainly depends on several parameters of water, which is termed as Electrical Conductivity (EC) node and Sodium Absorption Ratio (SAR) node under the Residual Sodium Carbonate (RSC) finalized. From the final report of the RSC that concluded with four types of Soil. The water irrigation prediction is important for farmers and it maintained with the help of previously available data. An irrigation influenced by salinity, alkalinity, sodicity and surface hardening of the soils. The results show that the amount of available water EC, PH, and Alkalinity. Sugarcane farmers are consistently in the expedition for higher harvest yields, higher expanded profit.

Keywords: Agriculture data, Crop yield, Data Analytics, Random forest, decision tree, prediction.

I. INTRODUCTION:

Android Apps are freely available on Google Play store, the official Android app store as well as third-party app stores for users to download. It is an open-source nature and popularity; malware writers are increasingly focusing on developing malicious applications for Android operating system. Despite various attempts by Google Play store to protect against malicious apps, they still find their way to mass market and cause harm to users by misusing personal information related to their phone book, mail accounts, GPS location information and others for misuse by third parties or else take control of the phones remotely.

Therefore, there is need to perform malware analysis or reverse-engineering of such malicious applications which pose serious threat to Android platforms. Broadly speaking, Android Malware analysis is of two types: Static Analysis and Dynamic Analysis. Static analysis basically involves analyzing the code structure without executing it while dynamic analysis is examination of the runtime behavior of Android Apps in constrained environment.

Given in to the ever-increasing variants of Android Malware posing zero-day threats, an efficient mechanism for detection of Android malwares is required. In contrast to signature-based approach which requires regular update of signature database, machine-learning based approach in combination with static and dynamic analysis can be used to detect new variants of Android Malware posing zero-day threats. In, broad yet lightweight static analysis has been performed achieving a decent detection accuracy of 94% using Support Vector Machine algorithm.

Nikola Milosevic et al. presented static analysis- based classification through two methodologies: one was permissions based while the other involved representation of the source code as a bag of words. Another approach based on identifying most significant permissions and applying machine learning on it for evaluation has been proposed. MADAM provides a multi-level analysis framework where behaviour of Android Apps is captured up to four levels: from package, user, application to kernel level, achieving detection accuracy as high as 96% with one disadvantage being that it could run only on rooted devices, making it incapable for commercial use. SAMADroid proposed a three-way novel host server-based methodology for improved performance as far as asset usage is concerned for malware detection on mobile devices.

II. LITERATURE REVIEW:

For the last few years, Android is known to be the most widely used operating system and this rapidly increasing popularity has attracted the malware developer's attention. Android allows downloading and installation of apps from other unofficial marketplaces. This gives malware developers an opportunity to put repackaged malicious applications in third-party app stores and attack Android devices. Many malware analysis and detection systems have been developed which uses static analysis, dynamic analysis, or hybrid analysis to keep Android devices secure from malware [1].Android users are constantly threatened by an increasing number of malicious applications (apps), generically called malware. Malware constitutes a serious threat to user privacy, money, device, and file integrity. In this paper, we note that, by studying their actions, we can classify malware into a small number of behavioral classes, each of which performs a limited set of misbehaviors that characterize them. These misbehaviors can be defined by monitoring features belonging to different Android levels [2]. With the widespread use of smartphones, the number of malware has been increasing exponentially. Among smart devices, android devices are the most targeted devices by malware because of their high popularity. This paper proposes a novel framework for android malware detection. Our framework uses various kinds of features to reflect the properties of android applications from various aspects, and the features are refined using our existence-based or similarity- based feature extraction method for effective feature representation on malware detection [3].

III. RESULTS AND DISCUSSIONS:

UML (Unified Modelling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. A use case diagram is used to represent the dynamic behaviour of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, stem/subsystem of an application. Fig. 1 represents the use case diagram which refers to all the functionalities which are provided to the user. There is only one end user, and he is functionalities.

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean, and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax. Implementation is one of the most important tasks in project is the phase in which one must be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample

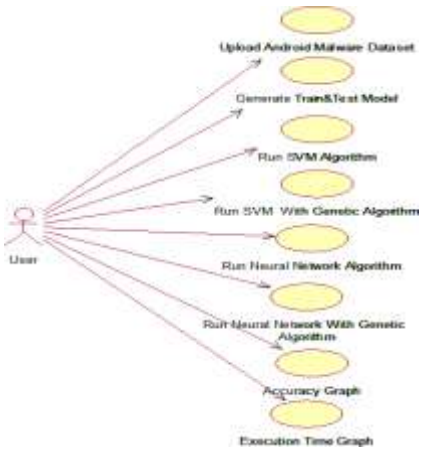

Fig.1: Use Case diagram for Android Malware Detection.

data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user. The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifying as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

Table 1 Acceptance Test

Test case Id	Test Case Name	Test Case Desc.	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Upload Android Malware Dataset	Test whether The Android Malware Dataset is Uploaded	If the Android Malware Dataset may not upload	we cannot do any further operations	we can do further operation	High	High
02	Generate Train & Test Model	Verify the Train & Test Model Generated or not	Without Generate the Train & Test model	we cannot do any further operations	we can do any further operations	High	High
03	Run SVM Algorithm	Test whether the SVM Algorithm run or not	If The SVM Algorithm may not be run	we cannot do any further operations	we can run SVM Algorithm	High	High
04	Run Neural Network Algorithm	Test whether the Neural Network Algorithm run or not	If the Neural Network Algorithm may not be run	we cannot do any further operations	we can run Neural Network Algorithm	High	High
05	Run SVM With Genetic Algorithm	Test whether the SVM with Genetic Algorithm run or not	If The SVM with Genetic Algorithm may not berun.	we cannot do any further operations	we can runSVM with Genetic Algorithm	High	High

IV. CONCLUSION AND FUTURE SCOPE

As the number of threats posed to Android platforms is increasing day to day, spreading mainly through malicious applications or malwares, therefore it is especially important to design a framework which can detect such malwares with accurate results. Where signature- based approach fails to detect new variants of malware posing zero- day threats, machine learning based approaches are being used. The proposed methodology attempts to make use of evolutionary Genetic Algorithm to get most optimized feature subset which can be used to train machine learning algorithms in most efficient way. From experimentations, a decent classification accuracy of more than 94% is maintained using Support Vector Machine and Neural Network classifiers while working on lower dimension feature-set, thereby reducing the training complexity of the classifiers. Further work can be enhanced using larger datasets for improved results and analyzing the effect on other machine learning algorithms when used in conjunction with Genetic Algorithm.
In future enhancements we would like to make the software run at runtime and built strategies that would have capabilities to detect new malware which attack the application and pry on user's sensitive information.

REFERENCES

[1] S. Arshad, M. A. Shah, A. Wahid, A. Mehmood, H. Song, and H. Yu,―SAMADroid: A Novel 3- Level Hybrid Malware Detection Model for Android Operating System,‖ IEEE Access, vol. 6, pp. 4321–4339, 2018.
[2] Saracino, D. Sgandurra, G. Dini, and F. Martinelli, ―MADAM: Effective and Efficient Behaviorbased Android Malware Detection and Prevention,‖ IEEE Trans. Dependable Secur. Comput., vol. 15, no. 1, pp. 83–97,2018.
[3] TaeGuen Kim, BooJoong Kang,Mina Rho,Sakir Sezer ,Eul Gyu Im―A Multimodal Deep Learning Method for Android Malware Detection using Various Features 21 August 2018.
[4] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa- An, and H. Ye, ―Significant Permission Identification for Machine- Learning-Based Android Malware Detection,‖ IEEE Trans. Ind. Informatics, vol. 14, no. 7, pp. 3216–3225, 2018.
[5] N. Milosevic, A. Dehghantanha, and K.
K. R. Choo, ―Machine learning aided Android malware classification,‖ Comput. Electr. Eng., vol. 61, pp. 266–