****

**DECLARATION STATEMENT**

**This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Data Science at the University of Hertfordshire.**

**I have read the guidance to students on academic integrity, misconduct and plagiarism information at** [**Assessment Offences and Academic Misconduct**](https://www.herts.ac.uk/__data/assets/pdf_file/0007/237625/AS14-Apx3-Academic-Misconduct-v17.0.pdf) **and understand the University process of dealing with suspected cases of academic misconduct and the possible penalties, which could include failing the project module or course.**

**I certify that the work submitted is my own and that any material derived or quoted from published or unpublished work of other persons has been duly acknowledged. (Ref. UPR AS/C/6.1, section 7 and UPR AS/C/5, section 3.6). I have not used chatGPT, or any other generative AI tool, to write the report or code (other than where declared or referenced).**

**I did not use human participants or undertake a survey in my MSc Project.**

**I hereby give permission for the report to be made available on module websites provided the source is acknowledged.**

**Student Name printed: Anantha Kumar Nallapaneni**

**Student Name signature: Anantha Kumar Nallapaneni**

**Student SRN number: 21060940**

**UNIVERSITY OF HERTFORDSHIRE**

**SCHOOL OF PHYSICS, ENGINEERING AND COMPUTER SCIENCE**

**Abstract**

The Aim of this project is to enhance the security of a computer system by using machine learning techniques to detect a malware. For the classification of malware and system information, it includes models that consist of Decision Tree Classifier, Support Vector Machine as well as k-means clustering. By analysing the metrics of the “Malware Detection. csv” dataset, it is found that SVM is moderately efficient, while Decision Trees are much more reliable. The outcomes forward depict how machine learning could improve recognising malware as well as strengthening cybersecurity measures although it can be coupled with some demerits such as overfitting. This study examines the effectiveness of various machine learning techniques in malware detection, highlighting the strengths and limitations of models such as Decision Trees, SVMs, and k-means clustering. While the Decision Tree model achieved high accuracy, potential overfitting raises concerns about its generalizability. SVMs showed moderate performance, indicating the need for further refinement.

**Table of Contents**

[Chapter 1: Introduction 5](#_Toc175761511)

[1.1 Aim and objectives 5](#_Toc175761512)

[1.2 Research Questions 5](#_Toc175761513)

[1.3 Research background 5](#_Toc175761514)

[1.4 Research Rationale 6](#_Toc175761515)

[1.5 Research Structure 6](#_Toc175761516)

[Chapter 2: Literature Review 7](#_Toc175761517)

[2.1 Chapter Introduction 7](#_Toc175761518)

[2.2 Impact of machine learning approaches in Malware detection 7](#_Toc175761519)

[2.3 Applications of machine learning approaches in Malware detection 8](#_Toc175761520)

[2.4 Comparative Analysis 9](#_Toc175761521)

[2.5 Ethical and Privacy Considerations in Malware detection 10](#_Toc175761522)

[2.5 Literature Gap 10](#_Toc175761523)

[2.6 Chapter Summary 11](#_Toc175761524)

[Chapter 3: Methodology 12](#_Toc175761525)

[3.1 Data Collection and Preprocessing 12](#_Toc175761526)

[3.1.1 Data 12](#_Toc175761527)

[3.1.2 Methodology 12](#_Toc175761528)

[3.2 Dataset Description 12](#_Toc175761529)

[3.3 Feature Engineering 13](#_Toc175761530)

[3.4 Model Selection and Training 13](#_Toc175761531)

[3.5 Model Evaluation and Performance Metrics 14](#_Toc175761532)

[3.6 Ethical and Privacy Considerations in Malware detection 14](#_Toc175761533)

[3.7 Deployment and Monitoring 15](#_Toc175761534)

[Chapter 4: Result 16](#_Toc175761535)

[Chapter 5: Discussion and Conclusion 26](#_Toc175761536)

[5.1 Interpretation of findings 26](#_Toc175761537)

[5.2 Limitation of research 26](#_Toc175761538)

[5.3 Future Recommendations 27](#_Toc175761539)

[Reference 28](#_Toc175761540)

[Appendix 30](#_Toc175761541)

# Chapter 1: Introduction

Malware represents a crucial cybersecurity risk, requiring improved identification techniques for preventing the data breaches, unapproved access, alongside the damage of the system. This study centers around utilizing the machine learning approaches for the malware identification, using the specific dataset containing qualities like millisecond, usage\_counter, hash, state, static\_prio, and so on. The initial purpose is to survey different machine learning strategies for their viability in recognizing benign along with malware cases inside the dataset. Approaches involving the "Decision Trees", "Support Vector Machines (SVM)", "Ridge Regression", "Linear Regression", alongside "K-Means cluster" will be assessed in view of the respective performance metrics. Focus will be put on data preprocessing alongside feature selection to upgrade model preparation and generalizability. The details expect to distinguish the most reasonable machine learning model for improving the cybersecurity measures by precisely predicting the malware presence and comprehending model weaknesses and strengths within this specific situation.

## 1.1 Aim and objectives

**Aim**

The aim of this research is to assess and compare the machine learning approaches for effective malware identification, improving the cybersecurity by evaluating performance metrics and also model usefulness.

**Objectives**

* To examine the performance metrics of the regression, classification and cluster models for malware identification.
* To evaluate the effect of the data preprocessing alongside the feature selection on the model accuracy alongside the generalizability.
* To recognize strengths along with weaknesses of every machine learning method within malware identification.
* To compare study outcomes with the existing literature on the machine learning-associated malware identification for highlighting improvements, discrepancies, alongside significant enhancements within cybersecurity measures.

## 1.2 Research Questions

* Which machine learning approach exhibits the highest accuracy in distinguishing among benign along with malware instances utilizing the dataset attributes?
* How does the data preprocessing alongside feature selection impact the performance of the machine learning approaches within malware detection?
* What are the specific weaknesses and strengths of regression, classification and cluster models in identifying malware?
* How do the particular research outcomes compare with the existing literature on the machine learning strategies for malware identification for cybersecurity measures?

## 1.3 Research background

Malware continues representing a crucial risk within cybersecurity, requiring adequate recognition methodologies for securing against the data breaches, unapproved access, alongside system compromises. This study centers around utilizing the machine learning approaches in improving the malware recognition viability. The specific dataset used includes different features, for example, millisecond timestamps, usage counters, hash values, priority levels, process states, and also static priorities. These properties give a thorough perspective on the malware behaviors alongside system communications, crucial for generating successful detection models (Şahın *et al.* 2022). Crucial purposes incorporate performance metrics of the respective approaches, surveying the effect of the data preprocessing alongside feature selection on model accuracy and generalizability, and distinguishing inherent qualities and limitations of various machine learning approaches within malware recognition. The research expects to add to cybersecurity by enhancing the comprehension of malware behaviour and improving identifications capacities through cutting edge insightful strategies (Shhadat *et al.* 2020). By contrasting study results and existing literature, the research tries to emphasize improvements, significant errors, and regions for additional improvement within machine learning associated malware identification procedures. This study evaluates the machine learning strategies for the malware identification, using a dataset with credits like hash values, millisecond timestamps, alongside priority levels. By assessing the exhibition of different models, the research strives to recognize the best methodology for recognizing malicious and also benign patterns. Focus is on the data preprocessing as well as feature selection to improve the model accuracy. The details will be contrasted and also existing literature, intending to further develop cybersecurity evaluations and give experiences into adequate malware identification methodologies.

## 1.4 Research Rationale

The following rationale for this evaluation lies within the earnest requirement to reinforce cybersecurity measures against developing malware risks. Conventional signature-associated strategies are progressively insufficient against refined malware variations, requiring progressed procedures, for example, machine learning (Masum *et al.* 2022). By assessing and contrasting different machine learning strategies, this research intends to recognize the best approaches for recognizing benign and also malevolent software behaving. Grasping the weaknesses and strengths of these models, alongside upgrading the data preprocessing and highlighting determination, will essentially improve recognition accuracy and responsiveness progressively in cybersecurity situations.

## 1.5 Research Structure

Chapter 1: Introduction

Chapter 2: Literature Review

Chapter 3: Methodology

Chapter 4: Result

Chapter 5: Discussion and Conclusion

#### Figure 1.5.1: Showing the Research Structure

(Source: Self-developed)

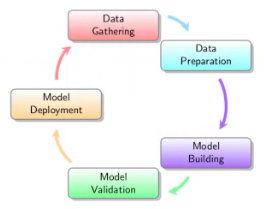
# Chapter 2: Literature Review

## 2.1 Chapter Introduction

The emergence of malware as a critical risk to cybersecurity has required the improvement of innovative recognition techniques. Conventional signature-associated identification frameworks, while viable against the known risks, frequently miss the mark while defying novel and refined malware variations. Subsequently, there has been a change in outlook towards utilizing the machine learning methods within malware identification, which provide dynamic and versatile arrangements equipped for recognizing beforehand concealed malevolent exercises. This particular research digs into the current assortment of research on the utilization of machine learning within malware recognition. This evaluates different techniques embraced to upgrade the detection precision and strength, involving the preprocessing of the data, the choice of significant elements, and the streamlining of algorithmic boundaries. The specific review envelops a wide exhibit of machine learning methods, featuring their particular benefits and constraints in combating the malware. Moreover, this potion looks at the effect of advancing malware strategies on identification techniques. As the malware creators constantly refine their methods for evading the recognition, the specific literature uncovers a comparison of development in protective procedures. The coordination of machine learning with different innovations, like behavior evaluation and also anomaly identification, is additionally examined, exhibiting the complex methodology expected to successfully balance malware risks.

## 2.2 Impact of machine learning approaches in Malware detection

The coordination of the machine learning methods within malware identification has transformed cybersecurity, giving powerful and dynamic choices for distinguishing and moderate risks. Conventional techniques, which depend intensely on signature-associated identification, have combated to remain with the fast development of the malware. Machine learning provides a more versatile and proactive methodology, equipped for distinguishing new and obscure risks by learning examples and ways of behaving related to the malicious exercises (Shoeibi *et al.* 2024). The machine learning evaluations evaluate tremendous measures of the data to distinguish peculiarities and examples characteristic of malware. By utilizing highlights, for example, record conduct, network traffic, and framework calls, these evaluations may perceive inconspicuous contrasts among harmless and vindictive exercises.



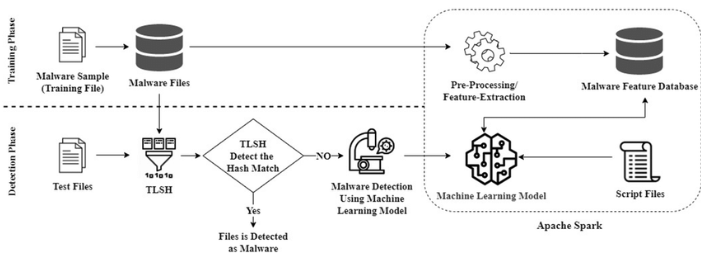
#### Figure 2.2.1: Approaches for malware analysis using machine learning

(Source: Alenezi *et al.* 2020 )

This ability is significant in recognizing modern malware that utilizes obfuscation and avoidance methods to sidestep conventional defenses. One huge benefit of machine learning within malware identification is its capacity to enhance over some period. As the new data is taken care of into the framework, the specific machine learning models are ceaselessly prepared and modified, improving their precision and productivity. This flexibility guarantees that the recognition framework stays viable against arising risks, diminishing the window of the vulnerability and limiting the effect of the zero-day assaults. Besides, machine learning strategies improve the mechanization of malware recognition procedures. Automation lessens the dependence empowering quicker reaction times and permitting in emphasizing on additional complicated tasks. The versatility of the machine learning arrangements likewise implies that they can be sent across different conditions, from the individual devices to enormous scope networks, guaranteeing extensive security (Alenezi *et al.* 2020). The utilization of machine learning within malware recognition reaches out to prescient examination. By recognizing patterns and examples in verifiable data, the following machine learning approaches may predict likely future risks and also the vulnerabilities, empowering pre planned measures. This proactive position essentially improves the general security posture, making this harder for the malware for exploiting frameworks.

## 2.3 Applications of machine learning approaches in Malware detection

The following Machine learning strategies have evaluated broad applications within malware identification, changing how the cybersecurity measures are developed. These applications influence the capacity of the machine learning evaluations to examine and decipher complex data trends, offering improved precision and also productivity in recognizing vindictive exercises (Aleesa *et al.* 2020). One of the essential applications is in the order of records and also the behavior. The following machine learning approaches may separate among benign as well as the malevolent records by inspecting different features like executable behaviors, framework connections and so on. This specific classification isn't restricted to the known malware; the following machine learning may distinguish beforehand concealed risks by gaining from examples and also anomalies within the data. Another huge application is within network traffic examination. By checking network details, the machine learning approach may distinguish strange patterns demonstrative of the malware exercises, like the data exfiltration or the command-and-control interchanges. The following proactive identification assists in identifying alongside mitigating the specific risks before they cause remarkable harm (Al-Mhiqani *et al.* 2020).



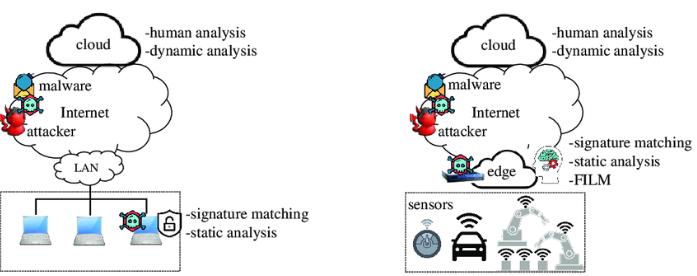
#### Figure 2.3.1: Machine learning based malware analysis architecture

(Source:Aleesa *et al.* 2020)

Behavioral assessment is moreover a significant portion where the respective machine learning is developed. By evaluating the approach to acting of clients logically, the particular machine learning strategies may flag the questionable measures that go astray from conventional models. This is particularly convincing against refined malware that undertakings to disguise its presence through unobtrusive systems. The accompanying machine learning methods are used inside the anomaly recognition, where the particular strategies are ready to see deviations from the generated baselines.

## 2.4 Comparative Analysis

The comparison investigation shows the differences in the machine learning algorithms’ ability in perceiving malware detection efficiency. When it comes to the accuracy rate then the Decision Tree Classifier proudly exhibited a Whooping 99 % accuracy almost correct in every prediction that the software made. 99% accuracy; however, overfitting could be a problem. The SVM classifier on the other hand was relatively impressive scoring a 52. The speed and effectiveness of the alert, 72% accuracy rate as well as other incorrect classifications. While using the K-means clustering algorithm had advantages in the grouping of data, it was still heavily reliant on the use of PCA in order to reduce the heterogeneity of the sets. Equally, high MSE and MAE reflect the high variability in the prediction of linear along with ridge regression analysis. Thus, three of the models worked fine in general, yet all of them revealed flaws that indicate the necessity of further optimisation of models and more further fine-tuning of the models.



#### Figure 2.4.1: Comparison of the machine learning based malware analysis architecture

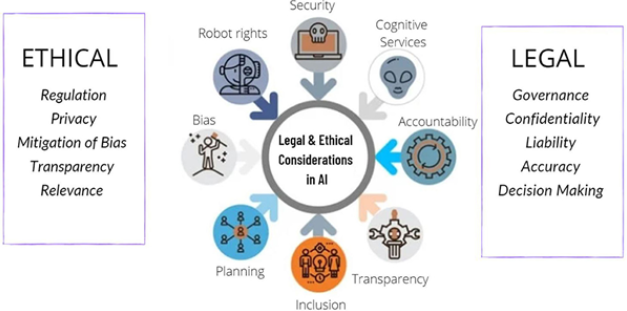
(Source: Shafiq *et al.* 2021)

Every technique has its benefits: static evaluation is for the most part quicker and less resource concentrated, while dynamic evaluation can uncover more modern, behavior-associated risks. One crucial component in the comparative evaluation is the specific accuracy of the identification (Shafiq *et al.* 2021). The particular machine learning approaches are assessed in view of their capacity to arrange harmless and also malicious examples accurately. Greater accuracy is vital for limiting the false positives along with the false negatives, the two of which can have huge results. The respective False positives might prompt pointless alarms and asset assignment, while bogus negatives can result in undetected malware causing harm. One more significant viewpoint is the proficiency and versatility of the following machine learning strategies. Several models require broad computational assets and also scalability for the training, making them less reasonable for constant identification. Conversely, different models can be immediately refreshed and conveyed, offering better versatility for enormous scope frameworks and organizations.

The interpretability of the following machine learning approaches is a crucial contemplation. Various techniques produce results that are more obvious and make sense, which is significant for acquiring trust from the cybersecurity experts. Transparent approaches empower better independent direction and assess consistency with the regulatory necessities.

## 2.5 Ethical and Privacy Considerations in Malware detection

As the AI strategies become continuously urgent to malware recognizable proof, it is essential to address the ethical and also furthermore protection considerations related with their usage. Ensuring these advances are conveyed constantly is vital for making do with trust and safeguarding client freedoms. One fundamental concern is data insurance. The specific machine learning methods for malware identification oftentimes require expansive datasets, involving the potentially fragile data from clients' gadgets and furthermore association measures. Managing and also processing this respective data ought to adjust to the data security rules, for instance, GDPR or CCPA, that command serious measures for getting the singular details. Anonymizing the following data and getting express consent from clients are basic stages for relieving the security possibilities. Another moral concern includes straightforwardness close to the obligation of the particular machine learning methods (Feng *et al.* 2021). These methodologies can be satisfactory as well as obscure, making it difficult for clients to grasp how decisions are made. Ensuring that understanding machine learning methods are sensible and providing clear, reasonable encounters into their dynamic methodology is critical for creating trust and engaging clients to challenge or charm decisions that impact them.



#### Figure 2.5.1: Ethical and Privacy Considerations in Malware detection

(Source: Feng *et al.* 2021)

The significance for the abuse of the respective machine learning inside malware recognition moreover raises the following ethical concerns. For instance, comparative advances expected to distinguish malware might be reused for observation or other intrusive measures. Implementing the effective ethical guidelines and solid oversight instruments is principal for forestalling abuse and ensuring that these advancements are used only for valid and adequate purposes (Bhosale and Patnaik, 2023). Bias within the machine learning approaches is another crucial issue. Automation and adequate innovations can essentially improve proficiency, however they ought to supplement instead of supplant human expertise, cultivating a cooperative way for dealing with the following cybersecurity.

## 2.5 Literature Gap

Regardless of critical progressions in applying machine learning to malware identification, various gaps continue within the recent body of exploration. One prominent gap is the absence of exhaustive evaluation through practical conditions. Various researches center around controlled datasets, which may not completely grasp the intricacy and changeability experienced within the practical situations. Thus, there is a requirement for more exploration that approves the respective machine learning in unique and heterogeneous conditions. Further gap lies in the interpretability and straightforwardness of the specific machine learning approaches utilized within the malware identification. While these models can accomplish high precision, their dynamic procedures are in many cases opaque, making it challenging for the cybersecurity experts to comprehend and trust the results. Further exploration is expected to foster more interpretable models that give clear details into how choices are made (Sarker, 2023). The issue of data protection and also ethical contemplations is likewise underexplored. In spite of the fact that the respective machine learning procedures require a lot of data for preparation, the respective implications of the data collection and utilization on client security have not been entirely tended to. More evaluations are expected for developing vigorous systems that balance the viability of the malware recognition with severe privacy securities.

## 2.6 Chapter Summary

This section has evaluated the composite applications and effects of the machine learning strategies within the section of the malware identification. This started with a prologue to the extraordinary job that AI has played in upgrading network safety, featuring the restrictions of the conventional strategies and the benefits of additional dynamic, versatile methods. The part dove into different uses of the machine learning within malware identification, for example, grouping, network traffic examination, behavioral evaluation, anomaly identification, and prescient analytics. These specific applications highlight the adaptability and also the adequacy of machine learning in distinguishing and also mitigating the malicious exercises across various conditions and situations.

A comparative evaluation gave details into the qualities and also weaknesses of various machine learning strategies, underlying factors like exactness, accuracy, versatility, interpretability, and strength against the evasion strategies. This research is pivotal for understanding how to best execute these advances in certifiable settings. Ethical as well as security contemplations were additionally assessed, highlighting the significance of the data protection, transparency, responsibility, alongside the fairness in conveying the machine learning for the malware detection. Guaranteeing capable utilization of these innovations is principal for managing the client trust and consistent with guidelines. This investigates the particular role of the machine learning evaluations within cybersecurity, underlining their capacity to distinguish and prevent risks through the data driven evaluation and ongoing variation. This features how these approaches distinguish examples and irregularities demonstrative of malevolent exercises, uncover stowed away dangers, and dissect unstructured data types like organization traffic and client conduct. The particular Machine learning upgrades proactive protection systems, empowering associations to expect and relieve significant cyber risks.

# Chapter 3: Methodology

## 3.1 Data Collection and Preprocessing

### 3.1.1 Data

Several essential parameters for malware verification, such as MilliSecs, Category, Status, Usage Counter, as well as memory information, are included in the "Malware Detection.csv" file. Each row is a single data piece that is categorized as either "benign" or "malware."

***Exploratory Data Analysis (EDA)***

Through exploratory data analysis (EDA), the framework as well as properties of the dataset are comprehended. To find patterns or anomalies, the first steps entail visualizing every feature's distribution (Jung *et al.* 2021). In order to detect any form of skewness or outlier in the information, there are graphical methods including the histograms for example, box plots in order to show the variation say Milli Secs and/or the Usage Counter.

### 3.1.2 Methodology

***Data Cleaning and Preprocessing***

There is the importance of data pre-treatment, together with, cleaning for the malware detection. Thus, the first step for ensuring data quality is to remove duplicate data and filter out noise (Kvet and Papan, 2022). Next up is normalization, which optimizes the model’s calibration through standardization of the features while at the same time reducing the effects of features that possess different scales.

Another important step of preparatory work is the processing of missing values. The data integrity is preserved by such methods as mean imputation for the statistical characteristics at best, or, for example, by k-nearest neighbors over the exact methods of filling.

Parsing is necessary to get categorical parameters compatible with some form of machine learning when they are connected with classifications. These variables are quantified mechanically and reduced into numerical form as single-hot encoding or label encoding which ensures the algorithms do not find it difficult to handle data with categories.

These preprocessing methods are essential for getting the data ready for machine learning computations since they have a direct impact on how accurate and trustworthy the model is at identifying malware.

## 3.2 Dataset Description

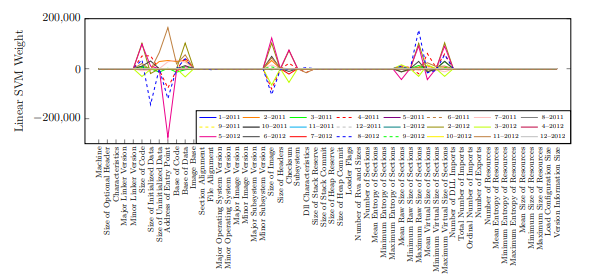
Some records relate to system parameter details of malware detection which are compiled in the “Malware Detection.csv” collection. Each row captures continuous measurements that depict a system being studied which can be referred to as a data point. Some features include classification, which tells whether the data is classified as “malware”, milli seconds as the measurement time, and hash which refers to the identification of each point of data. There are also included in the collection of P system indicators that reflect process precedence together with guidelines, such as usage\_counter , prio , static\_prio, normal\_prio, policy. Additional attributes, which also provide info on memory coupled with task management include vm\_pgoff, vm\_truncate\_count, task\_size, cached\_hole\_size, free\_area\_cache and mm\_users. Information on memory usage and/or consumption may include, but is not limited to reserved\_vm, exec\_vm, shared\_vm, total\_vm as well as hiwater\_rss. Some of the fields include nvcsw, nivcsw, whereby ‘csw’ refers to the current swap; min\_flt, which means the minimum of the page flip; maj\_flt, which is the maximum of page flip; fs\_excl\_counter is one of these fields that look at how well processes are performing through time. With this large set of data, machine learning models will then be trained to differentiate between beneficial and malicious system processes based on those properties and the way that these properties correlate to the existence of any malware.

## 3.3 Feature Engineering

Feature engineering is pervasive in the development of the feasible approaches to the malware identification of the machine-learning models. This operation includes the choice of the utmost features, which possibly describe the main trends in the behavior of malware and improving the model’s capability to classify the exercises as malicious or benign. The concept here is to identify and advertise structures that provide maximum utility to a work of doing. First of all, features are selected with respect to their relevance to the behavior of malware, that is, with respect to their potential for producing malicious behavior. They can include the file attributes, the traffic details of the network, the call sequences of the framework and the deftness measurements. The following selection procedure frequently includes domain expertise alongside statistical evaluation to guarantee that the most useful and non-redundant features are held. This simplification upgrades computational proficiency as well as mitigates the risk of overfitting by decreasing the commotion and overt repetitiveness within the data. Moreover, the feature engineering includes separating temporal, statistical, alongside the behavioral highlights from the raw data. Statistical elements might incorporate the variance, mean, and also entropy, while the temporal features grasp time based trends, alongside behavioral features evaluate the activities performed.

## 3.4 Model Selection and Training

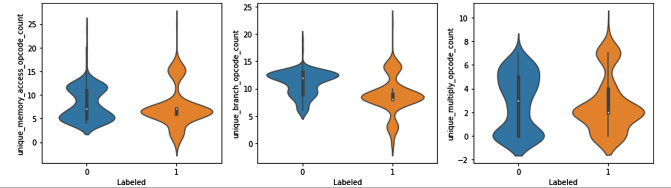
The choice and also training of the machine learning approaches are crucial moves toward fostering a successful malware recognition framework. This stage includes assessing different approaches to recognize the most appropriate ones for precisely recognizing benign and malicious programming exercises.



**Figure 3.4.1: SVM Analysis**

(Source: Wadkar *et al*. 2020)

Plotting SVM feature weights, the bar height emphasizes the significance of each feature. The direction of impact is indicated by positive as well as negative weights. Although a time series x-axis may identify trends, feature groups provide prediction areas. Gaining an understanding of these elements facilitates data investigation as well as an understanding of models.



**Figure 3.4.2: Violin Plots**

(Source: Gülataş *et al*. 2023)

The violin charts show how the opcode counts of the two groups differ. The labeled group had a smaller spread and a higher median for access to memory. In comparison to the unlabeled group, the labeled group exhibits larger medians as well as greater concentrated distribution for branch along with multiply operations.

At first, a scope of the machine learning approaches are viewed as founded on their reasonableness for the malware identification. These approaches might incorporate the decision trees, "support vector machines (SVM)", and gathering techniques, each offering exceptional benefits in dealing with various parts of the data. Decision trees are inclined toward for their interpretability and capacity to deal with complex, non-straight connections, while SVMs are viable in high-dimensional spaces and for grouping errands. Ensemble strategies, which consolidate numerous models to further develop execution, can improve performance and accuracy (Singh and Khare, 2022). When appropriate approaches are recognized, the models are prepared on the labeled datasets containing instances of both benign and also malicious exercises. This following training procedure includes feeding of the following approaches with the features designed from the data and permitting them to learn designs that recognize the two classes. The particular labeled datasets give ground truth that directs the growing experience, empowering the models to foster a prescient comprehension of the malware behavior.

## 3.5 Model Evaluation and Performance Metrics

Assessing the performance of the following machine learning approaches is vital in guaranteeing their adequacy within malware identification. Measurements like precision, accuracy, recall, alongside F1-value give a complete comprehension of a model's classification. Procedures like the cross-validation are utilized for validating the effectiveness and also generalization of the following approaches, guaranteeing they perform well on the specific unseen data (Yuan *et al.* 2020). The respective Confusion matrices provide experiences into the following true positives, true negatives, false positives, and also false negatives, while the following ROC curves assess the compromise between the true positive and also false positive rates. These assessments guarantee the approaches are effective and prepared to precisely identify malware.

## 3.6 Ethical and Privacy Considerations in Malware detection

As the AI strategies become continuously urgent to malware recognizable proof, it is essential to address the ethical and also furthermore protection considerations related with their usage. Ensuring these advances are conveyed constantly is vital for making do with trust and safeguarding client freedoms. One fundamental concern is data insurance. The specific machine learning methods for malware identification oftentimes require expansive datasets, involving the potentially fragile data from clients' gadgets and furthermore association measures. Managing and also processing this respective data ought to adjust to the data security rules, for instance, GDPR or CCPA, that command serious measures for getting the singular details. Anonymizing the following data and getting express consent from clients are basic stages for relieving the security possibilities. Another moral concern includes straightforwardness close to the obligation of the particular machine learning methods (Feng *et al.* 2021). These methodologies can be satisfactory as well as obscure, making it difficult for clients to grasp how decisions are made. Ensuring that understanding machine learning methods are sensible and providing clear, reasonable encounters into their dynamic methodology is critical for creating trust and engaging clients to challenge or charm decisions that impact them.

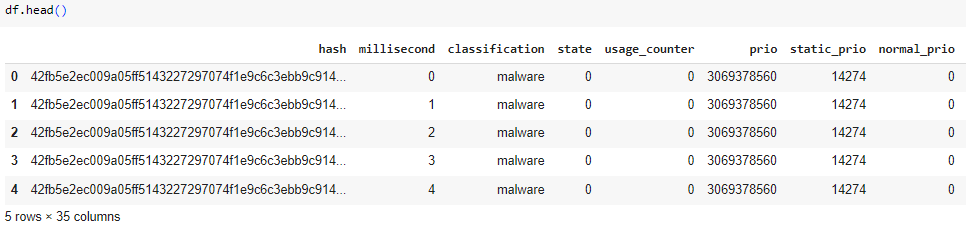
The significance for the abuse of the respective machine learning inside malware recognition moreover raises the following ethical concerns. For instance, comparative advances expected to distinguish malware might be reused for observation or other intrusive measures. Implementing the effective ethical guidelines and solid oversight instruments is principal for forestalling abuse and ensuring that these advancements are used only for valid and adequate purposes (Bhosale and Patnaik, 2023). Bias within the machine learning approaches is another crucial issue. Automation and adequate innovations can essentially improve proficiency, however they ought to supplement instead of supplant human expertise, cultivating a cooperative way for dealing with the following cybersecurity.

## 3.7 Deployment and Monitoring

The integration of trained machine learning solutions into the cybersecurity frameworks offer constant risks’ identification, improving the organizational protection. In this case, observation constantly is necessary to detect the model corruption and the concept drift to guarantee the model’s reliability. Further, the ethical considerations and security measures should be incorporated at the arrangements to shield the client information and to invest with the clients soundly. This approach affords cutthroat, diversified and also sound cybersecurity counter measures against progressing malware threats. Moreover, other automatic notifications which are not the components of the described logging may be proposed to inform possible risks and framework irregularities in time. We are almost sure that models receive new data inputs from time to time so that they are properly prepared to look for new malware types. Development and observing is active for being participate and too for being engaged in an experience to deal with current threats that endanger standardized resources in a specific computerized environment which is planned and prepared in advance to meet these threats.

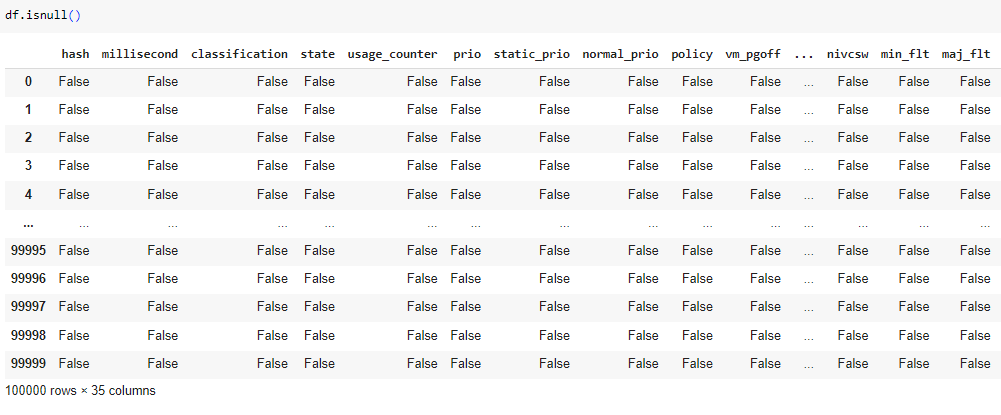
# Chapter 4: Result

Specifically, the study intends to provide a comprehensive explanation of how different approaches that are in the machine learning domain can be employed in the identification of malware. They are used to determine the basic characteristics of the data such as the features of the given dataset, search for the missing values, as well as the data distribution. Boundary of each model such as Decision Tree, SVM, and Ridge Regression models are measured by parameters such as accuracy, correction matrix, and recall level. The study points at advantages and drawbacks of each model in regard to malware detection and calls for more research with the purpose of enhancing the models’ accuracy as well as various methods in the sphere of cyber security.



#### Figure 4.1: Top 5 rows of the data

The top 5 rows of the dataset are represented here. Every row addresses the unique data point, involving different features connected with the framework as well as the process attributes. Key elements incorporate time-related measurements, utilization counters, need levels, memory insights, and state data. The specific "classification" segment fills in as the respective target variable, it is benign or malicious to show whether an occurrence. Examining these particular top rows assists in comprehending the dataset's design and the different qualities that the specific machine learning approaches will use to identify and also prevent the malware, eventually improving cybersecurity measures.



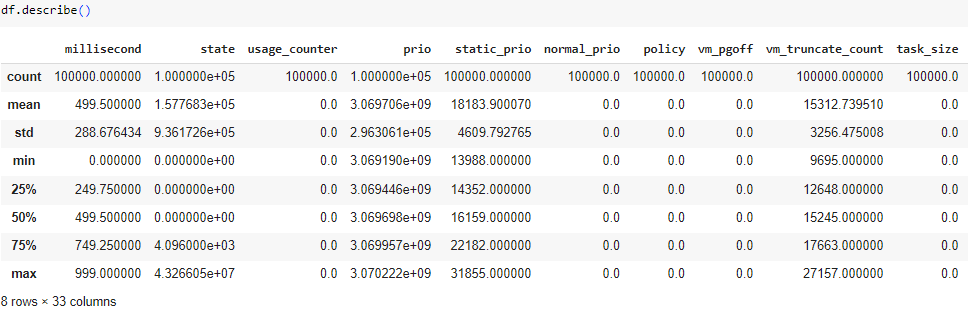
#### Figure 4.2: Checking Null Values

This particular phase shows the checking of the null values in the dataset. This can be derived from this step that there are no null values present in this dataset. The dataset goes through a crucial assessment for any null qualities, guaranteeing the data integrity alongside the completeness. The absence of the respective null values is the positive indicator, clarifying that the specific dataset is clean and prepared for evaluation without the requirement for extra preprocessing for dealing with missing data. This clean dataset considers more exact and solid preparation of the machine learning approaches, upgrading their capacity to successfully identify and also classify the malware. Assuring no missing values is significant for the vigorous performance of the machine learning evaluations, prompting more reliable cybersecurity measures and risk recognition capacities.



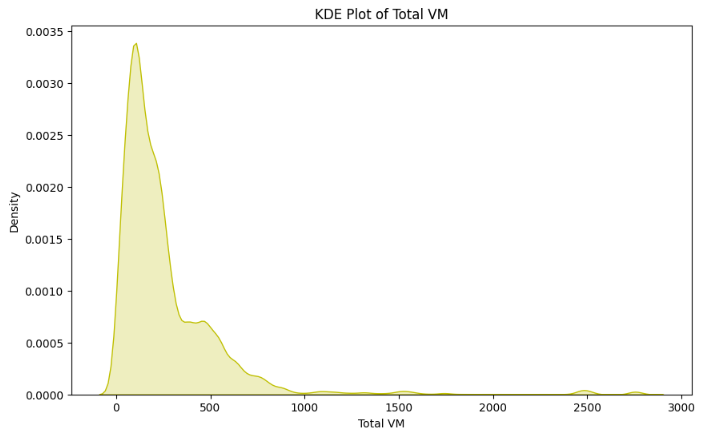
#### Figure 4.3 Rows and Columns Count

It demonstrates the shape of the dataset. It is inferred that this dataset has 100000 rows and 35 columns. This large volume of the data gives an adequate establishment to train the specific machine learning approaches, guaranteeing that it can learn different trends and subtleties related with malware identification. Every row addresses a unique occasion, while the particular 35 columns embody different frameworks alongside the process features significant for recognizing benign alongside the malicious exercises. The broad dataset supports growing more precise and accurate models, reinforcing cybersecurity endeavours by empowering the identification and counteraction of a wide cluster of modern digital risks.



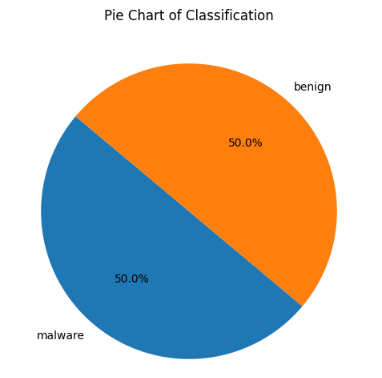
#### Figure 4.4: Description of the dataset

The description of the variables of this dataset including mean, standard deviation, minimum, maximum and so on are shown here. These measurements offer an extensive comprehension of the data conveyance and changeability, which is fundamental for the feature engineering and also determination within the malware recognition procedure. Comprehending these measurements assists in recognizing any expected anomalies or outliers and illuminates the preprocessing steps expected to standardize or scale the particular data. This exhaustive examination guarantees that the following machine learning approaches developed for distinguishing malware depend on detailed and precisely addressed data, also improving their exhibition and unwavering quality within cybersecurity applications.



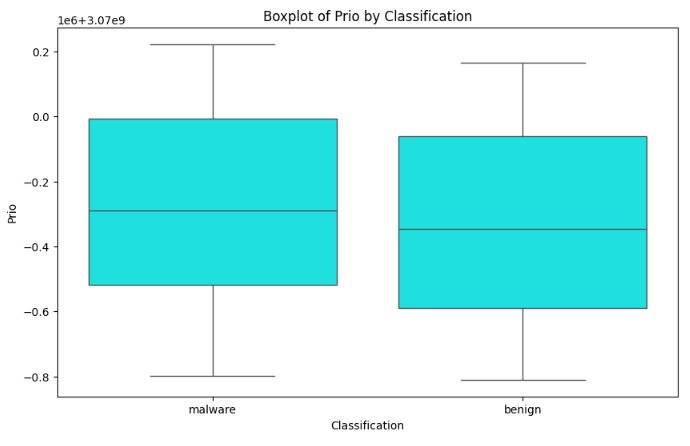
#### Figure 4.5: KDE plot of total VM

This is the KDE plot of the total VM. This can be inferred from this graph that the Maximum density of the total VM is in the range between 0 to 500. It demonstrates that the maximum instances inside the specific dataset show the overall VM values inside this range, giving crucial details to the malware detection. Comprehending the dispersion of virtual memory utilization is fundamental, as it can uncover average use trends and potential abnormalities related with vindictive exercises. Such details assists in the fine-tuning of machine learning approaches to more precisely recognize typical and suspicious ways of behaving, consequently upgrading the adequacy of the cybersecurity measures (Ahmad and Prasad, 2023).



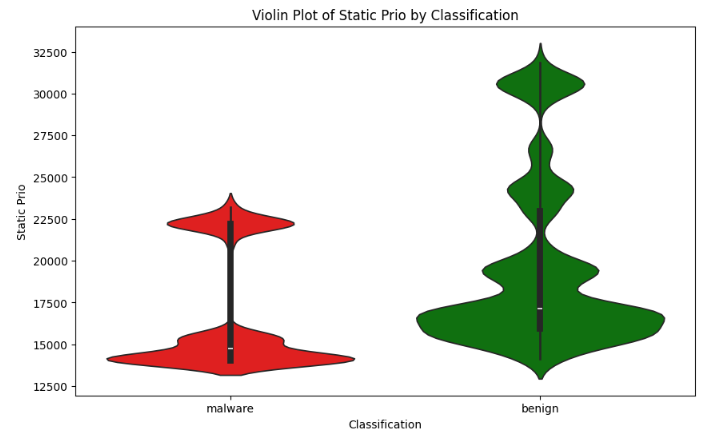
#### Figure 4.6: Pie Chart of Classification

The mentioned figure illustrates the pie chart of the classification. The percentile of the malware is 50% and the percentile of the benign is also 50%. This particular balanced dataset is effective for training the specific machine learning approaches, as this guarantees that the model gets an equivalent measure of models from the two classes, lessening the risk of the inclination. Such a uniformly circulated dataset is vital for creating adequate and precise classification models, as it assists the approaches with adequate learning the distinctive highlights of both malware and also the benign occurrences, eventually improving the unwavering quality of the malware recognition framework.



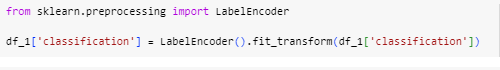
#### Figure 4.7: Box plot of the Prio by Classification

The specific boxplot delineates the conveyance of the "Prio'' by the classification, with the respective y-axis addressing "Prio" (deciphered as the measure of the priority) and also the specific x-axis showing the respective binary classification of "malware" alongside "benign." The following median line into every box means the midpoint of the specific data. The following malware classification displays a greater median "Prio '' contrasted with the benign classification, demonstrating that examples delegated malware for the most part have a higher need. The following interquartile range (IQR), addressed by the particular box, is more extensive for malware, recommending more suitable changeability within priority levels. The following whiskers, stretching out from the case, show the scope of the non-outlier data of interest, that are longer for malware, demonstrating a more extensive spread (Abusitta *et al.* 2021).



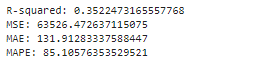
#### Figure 4.8: Violin Plot of Static Prio by Classification

Within the following x-axis, the following classifications are addressed, while the specific y-axis indicates the "static prio," deciphered as the important measure. The violin plots utilize the kernel density estimation to introduce a smoother circulation of the data. The more extensive segments of the violin body show a greater density of the following data points, and the smaller segments address less data points. The specific plot represents that the "malware" classification has a more extensive and higher dispersion on the y-axis, proposing that the "static prio" for malware cases will be higher contrasted with benign occasions. The following centerline inside every violin indicates the middle, while the white circles at the top alongside the bottom address the most extreme and least qualities, individually (Al-amri *et al.* 2021). This representation features that the "static prio" values for malware are for the greater and more factor,



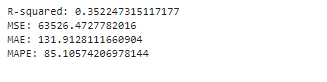
#### Figure 4.9: Transforming categorical to numerical columns

The following figure shows the change of the specific categorical variables into the numerical values, an effective preprocessing phase within machine learning. In this specific circumstance, the following "classification" variable, initially containing the respective categorical values like the "malware" and also "benign," is switched over completely to the numerical labels like 1 and 0. This specific transformation is fundamental for empowering machine learning approaches to process and also examine the data adequately, as these approaches need numerical input. By changing categorical data over completely to numerical data, it can be assured that the dataset is viable with different machine learning approaches, assessing with precise evaluation and expectation within malware identification activities (Millar *et al.* 2021).



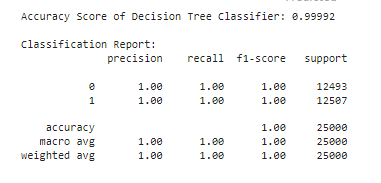
#### Figure 4.10: Linear Regression Result

The particular figure represents the following linear regression results for the malware identification utilizing the machine learning strategies. The particular regression line shows a positive relationship between the following predicted number of malware cases alongside the actual number of the malware occasions, demonstrating that the model overall predicts higher malware counts when the actual malware counts are high. The specific R-squared score of 0.352 suggests that about 35.22% of the difference within actual malware numbers is assessed by the specific model. In spite of this, the following "Mean Squared Error (MSE)" of 63526.47 alongside the "Mean Absolute Error (MAE)" of 131.91 feature huge forecast the errors, proposing that the model's expectations may deviate significantly from the specific actual malware counts, requiring further refinement.



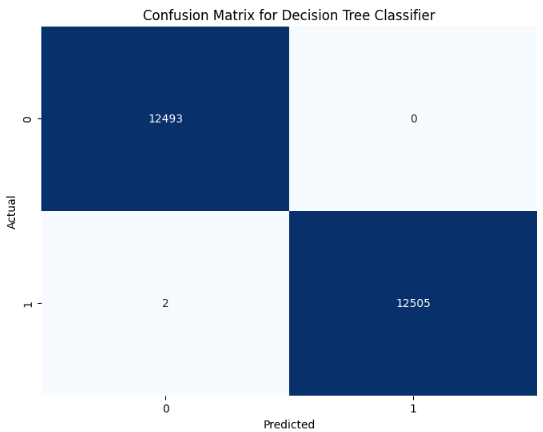
#### Figure 4.11: Ridge Regression Result

The particular figure shows the results of the Ridge Regression for the malware identification utilizing machine learning. The following R-squared score of 0.352 shows that the model records for 35.22% of the variance within the following actual malware data, reflecting moderate explanatory power. Also, the high "Mean Squared Error (MSE)" of 63,526.47 alongside the "Mean Absolute Error (MAE)" of 131.91 propose critical prediction errors. Overall, the respective "Absolute Percentage Error (MAPE)" of 85.11% from the following considerable qualities. These specific measurements feature the requirement for additional model streamlining to further develop expectation precision within malware identification.



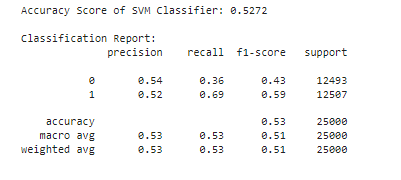
#### Figure 4.12: Performance Metrics of Decision tree Classifier

This shows the particular performance measurements of the following Decision Tree Classifier for the malware identification utilizing machine learning. The classifier accomplished a great accuracy value of 0.99992, demonstrating effective classification. The following classification report features the recall, precision, and F1-score for both the malware (1) alongside the benign (0) classifications, every scoring an approximate 1.00 across these specific metrics. It shows that the classifier is profoundly powerful at accurately distinguishing both malware and also benign instances. The particular support values, addressing the quantity of the true cases for every class, also affirm the overall model's strength. These outcomes highlight the following Decision Tree Classifier's adequate performance within precisely recognizing and also classifying malware.



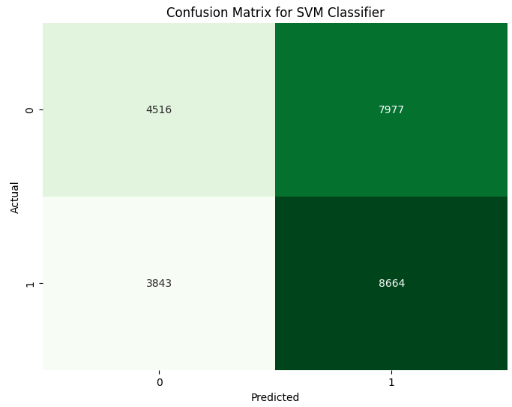
#### Figure 4.13: Confusion Matrix of Decision tree Classifier

The specific confusion matrix shows the performance of the following Decision Tree Classifier in distinguishing malware utilizing the specific machine learning approaches. This envisions the correct and also incorrect expectations made by the model. The specific rows address the specific actual classes (malware and harmless), while the sections address the anticipated classes. The particular diagonal components, demonstrating the correct orders, show that every one of the 12,505 information focuses were precisely classifications, recommending adequate model performance. Though, the absence of any off-diagonal components, addressing the incorrect classifications, may demonstrate overfitting, where the model performs perfectly on the test data yet may struggle with new, concealed data. This features the requirement for additional approval to guarantee generalizability (Sitaula and Shahi, 2022).



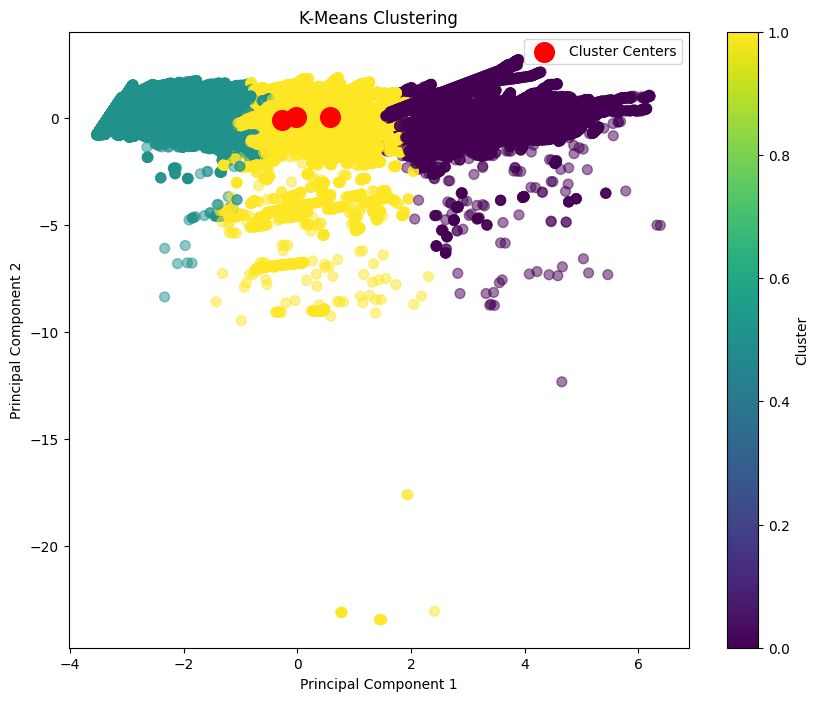
#### Figure 4.14: Performance Metrics of the SVM model

The specific performance metrics for the respective SVM classifier in distinguishing malware utilizing the following machine learning procedures demonstrate moderate adequacy. The particular accuracy value of 0.5272 proposes that the respective model accurately predicts the following classification of the data points around 53% of the specific time. The particular classification report shows an accuracy of 0.54 for the benign and 0.52 for the malware, demonstrating the specific number of the anticipated occasions were accurately ordered. Review upsides of 0.36 for the benign and 0.69 for the malware demonstrate how well the model distinguished the actual positives. The specific f1-score, which adjusts accuracy and also the recall, is 0.43 for the benign and also 0.59 for the malware. These outcomes feature the requirement for model upgrades to improve malware recognition accuracy.



#### Figure 4.15: Confusion Matrix of SVM Classifier

The particular confusion matrix for the respective SVM classifier within malware identification exhibits the model's performance on any test dataset. The particular rows address the actual classes, while the particular columns address the specific predicted classes. For this situation, the respective class 0 alongside class 1 compare to benign and also malware, individually. The respective matrix uncovers that the classifier accurately recognized 4,516 benign cases (true negatives) and 8,664 malware examples (true positives). Though, it additionally misclassified about 3,843 benign occasions as the malware (false positives) along with 7,977 malware occurrences as the benign (false negatives). This demonstrates a greater precision in recognizing malware contrasted with the benign cases, underscoring the requirement for additional model tuning to further develop the specific classification accuracy and also diminish misclassification rates.



#### Figure 4.16: K means Clustering

The particular k-means clustering chart with the three clusters shows the classification of the data points focused in view of their respective features, diminished to three principal elements. The particular x-axis and also the y-axis show the first and second principal elements, separately, grasping the vast majority of the following data variance. The specific data points are color-coded to display the cluster membership, with bigger circles demonstrating the cluster centroids.

# Chapter 5: Discussion and Conclusion

## 5.1 Interpretation of findings

The results from the research on malware identification utilizing the machine learning methods show a huge potential for these approaches to upgrade the cybersecurity measures. The use of different models, for example, decision trees, SVMs, as well as k-means clustering has given us a complex comprehension of how machine learning can distinguish alongside classify malware adequately. The following decision tree classifier exhibited an adequate performance, accomplishing an accuracy score of 99.99%. The following classification report additionally approved this with the precision, accuracy, recall, alongside F1-values all at 1.00 for the two classes. The particular confusion matrix exhibited effective classification, recommending that the following decision tree had the option to distinguish and also classify malware without any errors. However, this near-perfect outcome could likewise demonstrate potential overfitting, inferring that while the model performs incredibly well on the respective training data, it probably won't generalize as successfully on the unseen data.

The particular SVM classifier, accomplished an accuracy score of 52.72%, demonstrating the moderate presentation. The particular classification report uncovered an accuracy of 0.54 for the benign class alongside 0.52 for the malware class, with the specific F1-score assessing the balance among the precision and also recall. The following confusion matrix for the SVM indicated critical misclassifications, featuring the difficulties in separating among malware and also benign software precisely. It proposes that while SVMs are powerful, their viability may be restricted by the intricacy of the following feature space and the concept of the following data.

The specific k-means clustering graph uncovered how the following data points were gathered into two particular distinct clusters. The use of PCA considered the representation of the data in a diminished dimensional space, exhibiting clear detachments among the respective clusters. The particular linear regression outcomes showed the moderate connection among the predicted and also actual number of the malware instances, having a R-squared score of 0.352. Though, the greater MSE and also MAE proposed significant prediction errors, featuring the intricacy of precisely displaying malware identification with the linear methodologies. Ridge regression exhibited comparative difficulties, with the following MSE and also MAE demonstrating opportunity to enhancement within the prediction accuracy.

## 5.2 Limitation of research

Regardless of the promising outcomes from developing machine learning strategies to malware recognition, various limitations should be recognized. One critical restriction is the potential overfitting saw in certain models, especially with the decision tree classifier accomplishing close ideal accuracy on the respective training data. Overfitting happens when any model performs extraordinarily well on the following training set yet battles to generalize to the new, unseen data, restricting the practical application within practical situations. One more constraint is the moderate exhibition of the respective "Support Vector Machine (SVM)", which, in spite of being a strong method, showed just moderate accuracy and crucial misclassifications. It demonstrates that SVMs might require more component designing or tweaking to upgrade their presentation in recognizing malware and also benign programming successfully. Also, the "high mean squared error (MSE)" and also "mean absolute error (MAE)" within linear and also ridge regression approaches feature their constraints in grasping the intricacies of overall malware behavior, proposing that direct methodologies may not be appropriate for this sort of the classification task. The dependence on the following "principal component analysis (PCA)" for dimensionality reduction within k-means clustering, while helpful, may likewise overlook significant subtleties within the data that could affect the clustering accuracy (Taheri *et al.* 2020). Moreover, the specific dataset utilized for training and also evaluation, however broad, may not completely address the variety of malware risks, possibly influencing the generalizability of the models. Also, ethical contemplations and protection concerns connected with taking care of sensitive details were not broadly tended to in this exploration, which is critical for guaranteeing that the following arrangements follow administrative guidelines and secure user data. These specific limitations highlight the requirement for continued refinement of the machine learning approaches and more extensive datasets to improve the strength and applicability of the malware identifications frameworks. Moreover, the absence of the practical data incorporation and dynamic risk variation restricts the models' adequacy in advancing cyber conditions. The static characteristics of the dataset and also the models' inability to integrate constant updates or the feedback process might impact their exhibition against arising and complex malware risks. Tending to these difficulties is fundamental for enhancing the overall adequacy and flexibility of malware identification frameworks.

## 5.3 Future Recommendations

Future exploration ought to emphasize various critical regions to improve malware identification capacities. Incorporating constant data streams into models can enhance their flexibility and reaction to arising risks, tending to the restriction of static datasets. Growing more refined highlight extraction methods, involving social and relevant data, can upgrade the recognition of novel and muddled malware variations. Utilizing progressed machine learning draws near, like deep learning alongside hybrid approaches, could additionally further develop exactness and effectiveness. Furthermore, consolidating ensemble techniques and diverse security structures can give a more thorough protection against complex digital risks (Alarfaj *et al.* 2022). It is additionally vital to address the ethical and also privacy contemplations related with the data collection and model development, guaranteeing that the arrangements consent to administrative principles and regard client security. Future work ought to likewise evaluate the advancement of versatile models equipped for gaining from progressing dangers and also feedback, empowering consistent improvement and strength against developing malware strategies. Further, cooperative endeavors across businesses and also research establishments can encourage the sharing of risk insight and best practices, upgrading the aggregate protection against the cyber threats. Putting resources into client instruction and also awareness programs is additionally fundamental to alleviate the risk presented by designing risks that frequently go with malware dissemination. Also, effective checking and maintenance conventions ought to be recognized to expeditiously distinguish and correct any model float or performance issues, guaranteeing the supported viability of malware recognition frameworks.

# Reference

Ahmad, S.S. and Prasad, K.K., 2023. A Novel Machine Learning Framework for Analyzing Performance of Different Prediction Models by Using Automatic Malware Detection (AMD) Algorithm. Apex Journal of Business and Management, 1(1), pp.11-20. <https://nepjol.info/index.php/ajbm/article/download/61978/46678>

Al-amri, R., Murugesan, R.K., Man, M., Abdulateef, A.F., Al-Sharafi, M.A. and Alkahtani, A.A., 2021. A review of machine learning and deep learning techniques for anomaly detection in IoT data. Applied Sciences, 11(12), p.5320. <https://www.mdpi.com/2076-3417/11/12/5320/pdf>

Alarfaj, F.K., Malik, I., Khan, H.U., Almusallam, N., Ramzan, M. and Ahmed, M., 2022. Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms. IEEE Access, 10, pp.39700-39715. <https://ieeexplore.ieee.org/iel7/6287639/6514899/09755930.pdf>

Aleesa, A.M., Zaidan, B.B., Zaidan, A.A. and Sahar, N.M., 2020. Review of intrusion detection systems based on deep learning techniques: coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. Neural Computing and Applications, 32, pp.9827-9858. <https://www.researchgate.net/profile/Ahmed-Aleesa/publication/336663511_Review_of_intrusion_detection_systems_based_on_deep_learning_techniques_coherent_taxonomy_challenges_motivations_recommendations_substantial_analysis_and_future_directions/links/5eb3233e299bf152d6a1b330/Review-of-intrusion-detection-systems-based-on-deep-learning-techniques-coherent-taxonomy-challenges-motivations-recommendations-substantial-analysis-and-future-directions.pdf>

Alenezi, M.N., Alabdulrazzaq, H., Alshaher, A.A. and Alkharang, M.M., 2020. Evolution of malware threats and techniques: A review. International journal of communication networks and information security, 12(3), pp.326-337. <https://www.researchgate.net/profile/Haneen-Alabdulrazzaq/publication/349324759_Evolution_of_Malware_Threats_and_Techniques_A_Review/links/602ad0e64585158939a93934/Evolution-of-Malware-Threats-and-Techniques-A-Review.pdf>

Al-Mhiqani, M.N., Ahmad, R., Zainal Abidin, Z., Yassin, W., Hassan, A., Abdulkareem, K.H., Ali, N.S. and Yunos, Z., 2020. A review of insider threat detection: Classification, machine learning techniques, datasets, open challenges, and recommendations. Applied Sciences, 10(15), p.5208. <https://www.mdpi.com/2076-3417/10/15/5208/pdf>

Bhosale, Y.H. and Patnaik, K.S., 2023. Application of deep learning techniques in diagnosis of covid-19 (coronavirus): a systematic review. Neural processing letters, 55(3), pp.3551-3603. <https://link.springer.com/content/pdf/10.1007/s11063-022-11023-0.pdf>

Feng, R., Chen, S., Xie, X., Meng, G., Lin, S.W. and Liu, Y., 2020. A performance-sensitive malware detection system using deep learning on mobile devices. IEEE Transactions on Information Forensics and Security, 16, pp.1563-1578. <https://arxiv.org/pdf/2005.04970>

Gülataş, İ., Kılınç, H.H., Aydın, M.A. and Zaim, A.H., 2023. IoT Malware Detection Based on OPCODE Purification. Electrica, 23(3). Available at <https://electricajournal.org/Content/files/sayilar/96/634-642.pdf>

Jung, J., Park, J., Cho, S.J., Han, S., Park, M. and Cho, H.H., 2021. Feature engineering and evaluation for android malware detection scheme. Journal of Internet Technology, 22(2), pp.423-440. <https://jit.ndhu.edu.tw/article/viewFile/2500/2516>

Kvet, M. and Papan, J., 2022. The complexity of the data retrieval process using the proposed index extension. IEEE Access, 10, pp.46187-46213. Available at <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9763539>

Millar, S., McLaughlin, N., del Rincon, J.M. and Miller, P., 2021. Multi-view deep learning for zero-day Android malware detection. Journal of Information Security and Applications, 58, p.102718. <https://pure.qub.ac.uk/files/228719246/JISA_2020_558_Original_V0_submitted.pdf>

Şahın, D.Ö., Akleylek, S. and Kiliç, E., 2022. LinRegDroid: Detection of Android malware using multiple linear regression models-based classifiers. IEEE Access, 10, pp.14246-14259. <https://ieeexplore.ieee.org/iel7/6287639/6514899/09694615.pdf>

Sarker, I.H., 2023. Machine learning for intelligent data analysis and automation in cybersecurity: current and future prospects. Annals of Data Science, 10(6), pp.1473-1498. <https://link.springer.com/content/pdf/10.1007/s40745-022-00444-2.pdf>

Shafiq, S., Mashkoor, A., Mayr-Dorn, C. and Egyed, A., 2021. A literature review of using machine learning in software development life cycle stages. IEEE Access, 9, pp.140896-140920. <https://ieeexplore.ieee.org/iel7/6287639/6514899/09568959.pdf>

Shhadat, I., Hayajneh, A. and Al-Sharif, Z.A., 2020. The use of machine learning techniques to advance the detection and classification of unknown malware. Procedia Computer Science, 170,pp.917-922. <https://www.sciencedirect.com/science/article/pii/S1877050920305482/pdf?md5=3d234ee32988a067041bbe720dfc2f30&pid=1-s2.0-S1877050920305482-main.pdf>

Shoeibi, A., Khodatars, M., Jafari, M., Ghassemi, N., Sadeghi, D., Moridian, P., Khadem, A., Alizadehsani, R., Hussain, S., Zare, A. and Sani, Z.A., 2024. Automated detection and forecasting of covid-19 using deep learning techniques: A review. Neurocomputing, p.127317. <https://arxiv.org/pdf/2007.10785>

Singh, G. and Khare, N., 2022. A survey of intrusion detection from the perspective of intrusion datasets and machine learning techniques. International Journal of Computers and Applications, 44(7), pp.659-669. <https://www.researchgate.net/profile/Geeta-Singh-2/publication/349402075_A_survey_of_intrusion_detection_from_the_perspective_of_intrusion_datasets_and_machine_learning_techniques/links/6345234b76e39959d6b52c5b/A-survey-of-intrusion-detection-from-the-perspective-of-intrusion-datasets-and-machine-learning-techniques.pdf>

Sitaula, C. and Shahi, T.B., 2022. Monkeypox virus detection using pre-trained deep learning-based approaches. Journal of Medical Systems, 46(11), p.78. <https://link.springer.com/content/pdf/10.1007/s10916-022-01868-2.pdf>

Taheri, R., Ghahramani, M., Javidan, R., Shojafar, M., Pooranian, Z. and Conti, M., 2020. Similarity-based Android malware detection using Hamming distance of static binary features. Future Generation Computer Systems, 105, pp.230-247. <https://arxiv.org/pdf/1908.05759>

Wadkar, M., Di Troia, F. and Stamp, M., 2020. Detecting malware evolution using support vector machines. Expert Systems with Applications, 143, p.113022. Available at <https://www.researchgate.net/profile/Mark-Stamp/publication/336611638_Detecting_Malware_Evolution_Using_Support_Vector_Machines/links/5dae05d792851c577eb9683b/Detecting-Malware-Evolution-Using-Support-Vector-Machines.pdf>

Yuan, B., Wang, J., Liu, D., Guo, W., Wu, P. and Bao, X., 2020. Byte-level malware classification based on markov images and deep learning. Computers & Security, 92, p.101740. <https://www.researchgate.net/profile/Yuan-Baoguo/publication/338940023_Byte-level_Malware_Classification_Based_on_Markov_Images_and_Deep_Learning/links/5fe09894a6fdccdcb8ec5074/Byte-level-Malware-Classification-Based-on-Markov-Images-and-Deep-Learning.pdf>

# Appendix

*# importing the necessary modules*

**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** warnings

warnings**.**filterwarnings('ignore')

In [2]:

*#importing the dataset*

df**=** pd**.**read\_csv("Malware dataset.csv")

In [3]:

df**.**head()

Out[3]:

|  | **hash** | **millisecond** | **classification** | **state** | **usage\_counter** | **prio** | **static\_prio** | **normal\_prio** | **policy** | **vm\_pgoff** | **...** | **nivcsw** | **min\_flt** | **maj\_flt** | **fs\_excl\_counter** | **lock** | **utime** | **stime** | **gtime** | **cgtime** | **signal\_nvcsw** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914... | 0 | malware | 0 | 0 | 3069378560 | 14274 | 0 | 0 | 0 | ... | 0 | 0 | 120 | 0 | 3204448256 | 380690 | 4 | 0 | 0 | 0 |
| **1** | 42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914... | 1 | malware | 0 | 0 | 3069378560 | 14274 | 0 | 0 | 0 | ... | 0 | 0 | 120 | 0 | 3204448256 | 380690 | 4 | 0 | 0 | 0 |
| **2** | 42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914... | 2 | malware | 0 | 0 | 3069378560 | 14274 | 0 | 0 | 0 | ... | 0 | 0 | 120 | 0 | 3204448256 | 380690 | 4 | 0 | 0 | 0 |
| **3** | 42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914... | 3 | malware | 0 | 0 | 3069378560 | 14274 | 0 | 0 | 0 | ... | 0 | 0 | 120 | 0 | 3204448256 | 380690 | 4 | 0 | 0 | 0 |
| **4** | 42fb5e2ec009a05ff5143227297074f1e9c6c3ebb9c914... | 4 | malware | 0 | 0 | 3069378560 | 14274 | 0 | 0 | 0 | ... | 0 | 0 | 120 | 0 | 3204448256 | 380690 | 4 | 0 | 0 | 0 |

5 rows × 35 columns

In [4]:

df**.**isnull()

Out[4]:

|  | **hash** | **millisecond** | **classification** | **state** | **usage\_counter** | **prio** | **static\_prio** | **normal\_prio** | **policy** | **vm\_pgoff** | **...** | **nivcsw** | **min\_flt** | **maj\_flt** | **fs\_excl\_counter** | **lock** | **utime** | **stime** | **gtime** | **cgtime** | **signal\_nvcsw** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **1** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **2** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **3** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **4** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **99995** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **99996** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **99997** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **99998** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |
| **99999** | False | False | False | False | False | False | False | False | False | False | ... | False | False | False | False | False | False | False | False | False | False |

100000 rows × 35 columns

In [5]:

df**.**shape

Out[5]:

(100000, 35)

In [6]:

df**.**describe()

Out[6]:

|  | **millisecond** | **state** | **usage\_counter** | **prio** | **static\_prio** | **normal\_prio** | **policy** | **vm\_pgoff** | **vm\_truncate\_count** | **task\_size** | **...** | **nivcsw** | **min\_flt** | **maj\_flt** | **fs\_excl\_counter** | **lock** | **utime** | **stime** | **gtime** | **cgtime** | **signal\_nvcsw** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **count** | 100000.000000 | 1.000000e+05 | 100000.0 | 1.000000e+05 | 100000.000000 | 100000.0 | 100000.0 | 100000.0 | 100000.000000 | 100000.0 | ... | 100000.000000 | 100000.000000 | 100000.000000 | 100000.000000 | 1.000000e+05 | 100000.000000 | 100000.000000 | 100000.00000 | 100000.0 | 100000.0 |
| **mean** | 499.500000 | 1.577683e+05 | 0.0 | 3.069706e+09 | 18183.900070 | 0.0 | 0.0 | 0.0 | 15312.739510 | 0.0 | ... | 32.991160 | 2.053130 | 117.920240 | 1.109190 | 3.204448e+09 | 385415.451970 | 4.059310 | 1.66142 | 0.0 | 0.0 |
| **std** | 288.676434 | 9.361726e+05 | 0.0 | 2.963061e+05 | 4609.792765 | 0.0 | 0.0 | 0.0 | 3256.475008 | 0.0 | ... | 52.730176 | 13.881382 | 3.116892 | 2.160466 | 0.000000e+00 | 10144.036494 | 0.822848 | 3.26304 | 0.0 | 0.0 |
| **min** | 0.000000 | 0.000000e+00 | 0.0 | 3.069190e+09 | 13988.000000 | 0.0 | 0.0 | 0.0 | 9695.000000 | 0.0 | ... | 0.000000 | 0.000000 | 112.000000 | 0.000000 | 3.204448e+09 | 371782.000000 | 3.000000 | 0.00000 | 0.0 | 0.0 |
| **25%** | 249.750000 | 0.000000e+00 | 0.0 | 3.069446e+09 | 14352.000000 | 0.0 | 0.0 | 0.0 | 12648.000000 | 0.0 | ... | 1.000000 | 0.000000 | 114.000000 | 0.000000 | 3.204448e+09 | 378208.000000 | 3.000000 | 0.00000 | 0.0 | 0.0 |
| **50%** | 499.500000 | 0.000000e+00 | 0.0 | 3.069698e+09 | 16159.000000 | 0.0 | 0.0 | 0.0 | 15245.000000 | 0.0 | ... | 9.000000 | 1.000000 | 120.000000 | 0.000000 | 3.204448e+09 | 383637.000000 | 4.000000 | 0.00000 | 0.0 | 0.0 |
| **75%** | 749.250000 | 4.096000e+03 | 0.0 | 3.069957e+09 | 22182.000000 | 0.0 | 0.0 | 0.0 | 17663.000000 | 0.0 | ... | 46.000000 | 1.000000 | 120.000000 | 1.000000 | 3.204448e+09 | 390324.000000 | 5.000000 | 1.00000 | 0.0 | 0.0 |
| **max** | 999.000000 | 4.326605e+07 | 0.0 | 3.070222e+09 | 31855.000000 | 0.0 | 0.0 | 0.0 | 27157.000000 | 0.0 | ... | 365.000000 | 256.000000 | 120.000000 | 18.000000 | 3.204448e+09 | 421913.000000 | 7.000000 | 15.00000 | 0.0 | 0.0 |

8 rows × 33 columns

In [7]:

df**.**columns

Out[7]:

Index(['hash', 'millisecond', 'classification', 'state', 'usage\_counter',

'prio', 'static\_prio', 'normal\_prio', 'policy', 'vm\_pgoff',

'vm\_truncate\_count', 'task\_size', 'cached\_hole\_size', 'free\_area\_cache',

'mm\_users', 'map\_count', 'hiwater\_rss', 'total\_vm', 'shared\_vm',

'exec\_vm', 'reserved\_vm', 'nr\_ptes', 'end\_data', 'last\_interval',

'nvcsw', 'nivcsw', 'min\_flt', 'maj\_flt', 'fs\_excl\_counter', 'lock',

'utime', 'stime', 'gtime', 'cgtime', 'signal\_nvcsw'],

dtype='object')

EDA

In [8]:

*# KDE plot of 'total\_vm'*

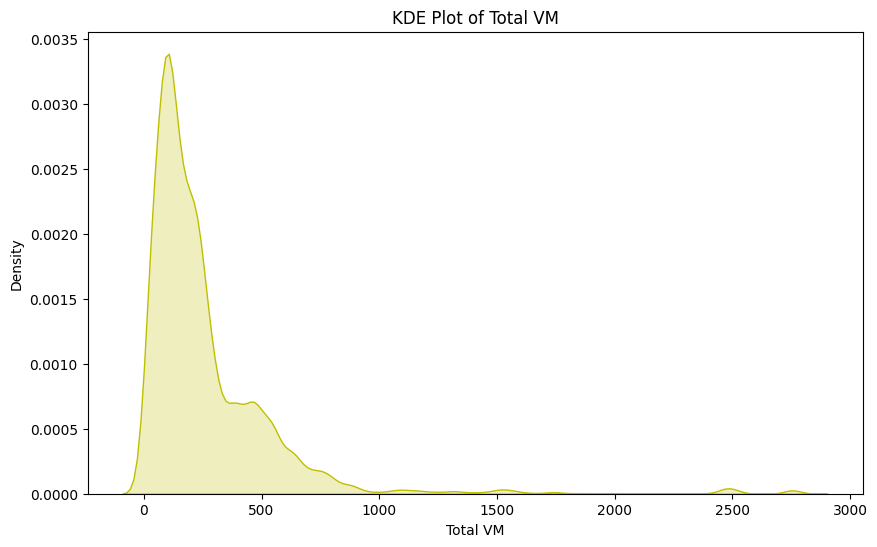
plt**.**figure(figsize**=**(10, 6))

sns**.**kdeplot(df['total\_vm'], shade**=True**, color**=**'y')

plt**.**title('KDE Plot of Total VM')

plt**.**xlabel('Total VM')

plt**.**show()



In [9]:

*# Pie chart of 'classification'*

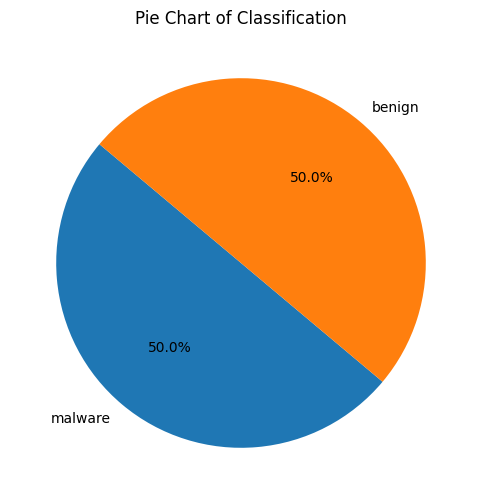
classification\_counts **=** df['classification']**.**value\_counts()

plt**.**figure(figsize**=**(10, 6))

plt**.**pie(classification\_counts, labels**=**classification\_counts**.**index, autopct**=**'%1.1f%%', startangle**=**140)

plt**.**title('Pie Chart of Classification')

plt**.**show()



In [10]:

*# Boxplot of 'prio' vs 'classification'*

plt**.**figure(figsize**=**(10, 6))

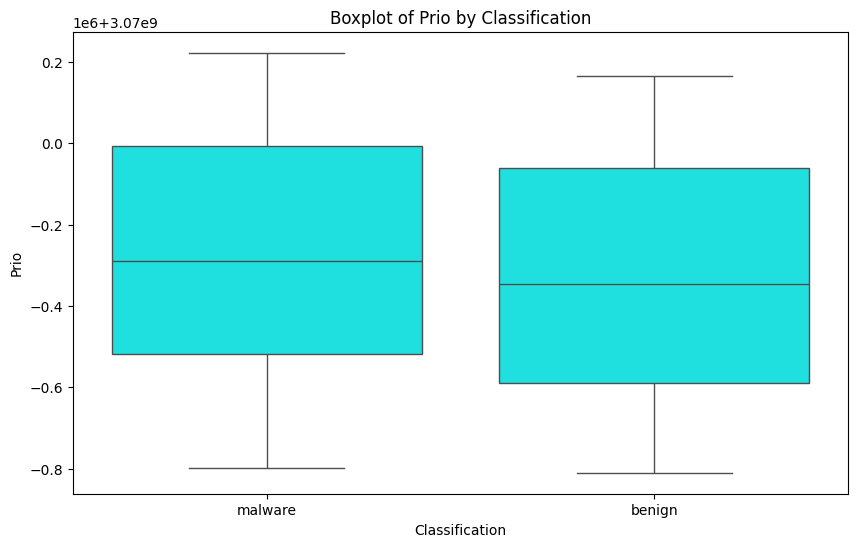
sns**.**boxplot(x**=**'classification', y**=**'prio', data**=**df, color**=** 'cyan')

plt**.**title('Boxplot of Prio by Classification')

plt**.**xlabel('Classification')

plt**.**ylabel('Prio')

plt**.**show()



In [11]:

*#Violin plot of 'static\_prio' vs 'classification'*

plt**.**figure(figsize**=**(10, 6))

palette **=** ['red', 'green']

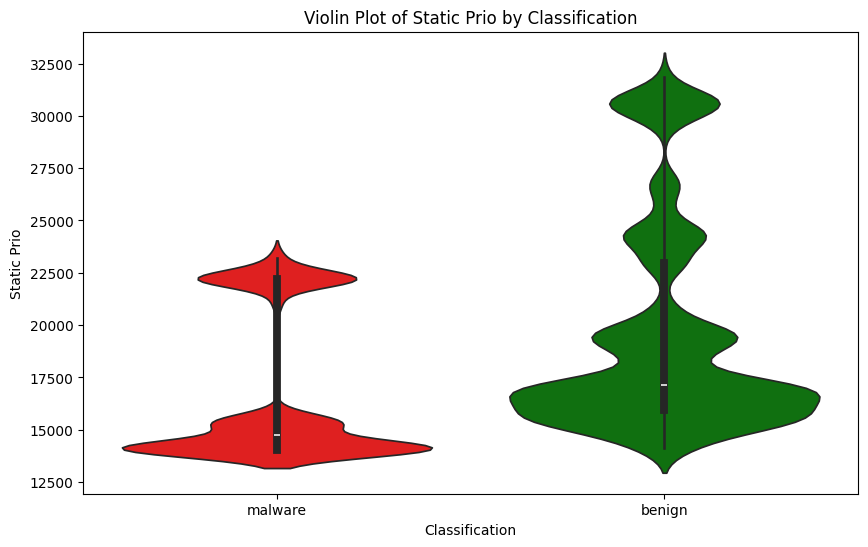
sns**.**violinplot(x**=**'classification', y**=**'static\_prio', data**=**df, palette**=** palette)

plt**.**title('Violin Plot of Static Prio by Classification')

plt**.**xlabel('Classification')

plt**.**ylabel('Static Prio')

plt**.**show()



In [12]:

data **=** [

'millisecond', 'classification', 'state', 'usage\_counter', 'min\_flt', 'static\_prio',

'normal\_prio', 'policy', 'task\_size', 'mm\_users', 'map\_count', 'hiwater\_rss',

'total\_vm', 'shared\_vm', 'exec\_vm', 'reserved\_vm'

]

*# Creating the new dataframe*

df\_1 **=** df[data]

*# Display the first few rows of the new dataframe*

print(df\_1**.**head())

millisecond classification state usage\_counter min\_flt static\_prio \

0 0 malware 0 0 0 14274

1 1 malware 0 0 0 14274

2 2 malware 0 0 0 14274

3 3 malware 0 0 0 14274

4 4 malware 0 0 0 14274

normal\_prio policy task\_size mm\_users map\_count hiwater\_rss total\_vm \

0 0 0 0 724 6850 0 150

1 0 0 0 724 6850 0 150

2 0 0 0 724 6850 0 150

3 0 0 0 724 6850 0 150

4 0 0 0 724 6850 0 150

shared\_vm exec\_vm reserved\_vm

0 120 124 210

1 120 124 210

2 120 124 210

3 120 124 210

4 120 124 210

In [13]:

**from** sklearn.preprocessing **import** LabelEncoder

df\_1['classification'] **=** LabelEncoder()**.**fit\_transform(df\_1['classification'])

In [14]:

df\_1**.**head()

Out[14]:

|  | **millisecond** | **classification** | **state** | **usage\_counter** | **min\_flt** | **static\_prio** | **normal\_prio** | **policy** | **task\_size** | **mm\_users** | **map\_count** | **hiwater\_rss** | **total\_vm** | **shared\_vm** | **exec\_vm** | **reserved\_vm** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 0 | 1 | 0 | 0 | 0 | 14274 | 0 | 0 | 0 | 724 | 6850 | 0 | 150 | 120 | 124 | 210 |
| **1** | 1 | 1 | 0 | 0 | 0 | 14274 | 0 | 0 | 0 | 724 | 6850 | 0 | 150 | 120 | 124 | 210 |
| **2** | 2 | 1 | 0 | 0 | 0 | 14274 | 0 | 0 | 0 | 724 | 6850 | 0 | 150 | 120 | 124 | 210 |
| **3** | 3 | 1 | 0 | 0 | 0 | 14274 | 0 | 0 | 0 | 724 | 6850 | 0 | 150 | 120 | 124 | 210 |
| **4** | 4 | 1 | 0 | 0 | 0 | 14274 | 0 | 0 | 0 | 724 | 6850 | 0 | 150 | 120 | 124 | 210 |

Linear Regression

In [15]:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.linear\_model **import** LinearRegression

**from** sklearn.metrics **import** mean\_squared\_error, mean\_absolute\_error

**import** numpy **as** np

*# Choosing features and also target variable for regression*

X **=** df\_1**.**drop(columns**=**['total\_vm', 'classification'])

y **=** df\_1['total\_vm']

*# Splitting the data into training and testing sets*

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.25, random\_state**=**42)

*# Generating and training the Linear Regression model*

model **=** LinearRegression()

model**.**fit(X\_train, y\_train)

*# Predicting on the test set*

y\_pred **=** model**.**predict(X\_test)

*# Evaluating R-squared, MSE, MAE, MAPE*

r\_squared **=** model**.**score(X\_test, y\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

mae **=** mean\_absolute\_error(y\_test, y\_pred)

mape **=** np**.**mean(np**.**abs((y\_test **-** y\_pred) **/** y\_test)) **\*** 100

y\_test\_non\_zero **=** y\_test**.**replace(0, np**.**nan) *# Replacing 0 with NaN for avoiding division by zero*

mape **=** np**.**mean(np**.**abs((y\_test\_non\_zero **-** y\_pred) **/** y\_test\_non\_zero)) **\*** 100

print(f"R-squared: {r\_squared}")

print(f"MSE: {mse}")

print(f"MAE: {mae}")

print(f"MAPE: {mape}")

R-squared: 0.3522473165557769

MSE: 63526.47263711507

MAE: 131.91283337588447

MAPE: 85.10576353529508

Ridge Regression

In [16]:

**from** sklearn.linear\_model **import** Ridge

**from** sklearn.metrics **import** mean\_squared\_error, mean\_absolute\_error

**import** numpy **as** np

*# Choosing features and also target variable for regression*

X **=** df\_1**.**drop(columns**=**['total\_vm', 'classification'])

y **=** df\_1['total\_vm']

*# Developing and training the Ridge Regression model*

alpha **=** 1.0 *# Regularization strength*

ridge\_model **=** Ridge(alpha**=**alpha)

ridge\_model**.**fit(X\_train, y\_train)

*# Predicting on the test set*

y\_pred **=** ridge\_model**.**predict(X\_test)

*# Evaluating R-squared, MSE, MAE, MAPE*

r\_squared **=** ridge\_model**.**score(X\_test, y\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

mae **=** mean\_absolute\_error(y\_test, y\_pred)

y\_test\_non\_zero **=** y\_test**.**replace(0, np**.**nan) *# Managing 0 with NaN for avoiding division by zero*

mape **=** np**.**mean(np**.**abs((y\_test\_non\_zero **-** y\_pred) **/** y\_test\_non\_zero)) **\*** 100

print(f"R-squared: {r\_squared}")

print(f"MSE: {mse}")

print(f"MAE: {mae}")

print(f"MAPE: {mape}")

R-squared: 0.3522473151171768

MSE: 63526.47277820162

MAE: 131.9128111660903

MAPE: 85.10574206978136

Decision Tree Classifier

In [17]:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.tree **import** DecisionTreeClassifier

**from** sklearn.metrics **import** confusion\_matrix, accuracy\_score, classification\_report

*# Assuming 'classification' is the target variable*

X **=** df\_1**.**drop(columns**=**['classification'])

y **=** df\_1['classification']

*# Splitting the data into training and testing sets*

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.25, random\_state**=**42)

*# Creating and training the Decision Tree Classifier*

dt\_classifier **=** DecisionTreeClassifier(random\_state**=**42)

dt\_classifier**.**fit(X\_train, y\_train)

*# Predicting on the test set*

y\_pred **=** dt\_classifier**.**predict(X\_test)

*# Generating the confusion matrix*

cm **=** confusion\_matrix(y\_test, y\_pred)

*# Plotting the confusion matrix*

plt**.**figure(figsize**=**(8, 6))

sns**.**heatmap(cm, annot**=True**, cmap**=**'Blues', fmt**=**'g', cbar**=False**)

plt**.**title('Confusion Matrix for Decision Tree Classifier')

plt**.**xlabel('Predicted')

plt**.**ylabel('Actual')

plt**.**show()

*# Calculate accuracy score*

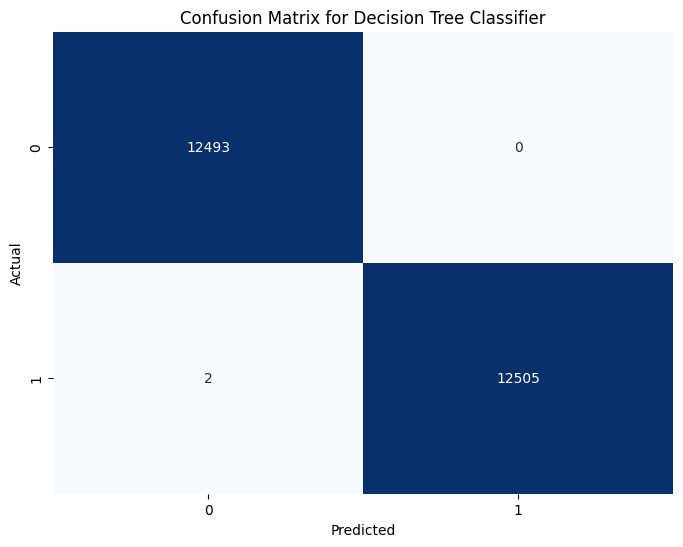
accuracy **=** accuracy\_score(y\_test, y\_pred)

print(f"Accuracy Score of Decision Tree Classifier: {accuracy}")

*# Displaying detailed performance metrics*

print("\nClassification Report:")

print(classification\_report(y\_test, y\_pred))



Accuracy Score of Decision Tree Classifier: 0.99992

Classification Report:

precision recall f1-score support

0 1.00 1.00 1.00 12493

1 1.00 1.00 1.00 12507

accuracy 1.00 25000

macro avg 1.00 1.00 1.00 25000

weighted avg 1.00 1.00 1.00 25000

SVM

In [18]:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.svm **import** SVC

**from** sklearn.metrics **import** confusion\_matrix, accuracy\_score, classification\_report

X **=** df\_1**.**drop(columns**=**['classification'])

y **=** df\_1['classification']

*# Creating and training the Support Vector Machine (SVM) classifier*

svm\_classifier **=** SVC(kernel**=**'rbf', random\_state**=**42)

svm\_classifier**.**fit(X\_train, y\_train)

*# Predicting on the test set*

y\_pred **=** svm\_classifier**.**predict(X\_test)

*# Generating the confusion matrix*

cm **=** confusion\_matrix(y\_test, y\_pred)

*# Plotting the confusion matrix*

plt**.**figure(figsize**=**(8, 6))

sns**.**heatmap(cm, annot**=True**, cmap**=**'Greens', fmt**=**'g', cbar**=False**)

plt**.**title('Confusion Matrix for SVM Classifier')

plt**.**xlabel('Predicted')

plt**.**ylabel('Actual')

plt**.**show()

*# Evaluating accuracy score*

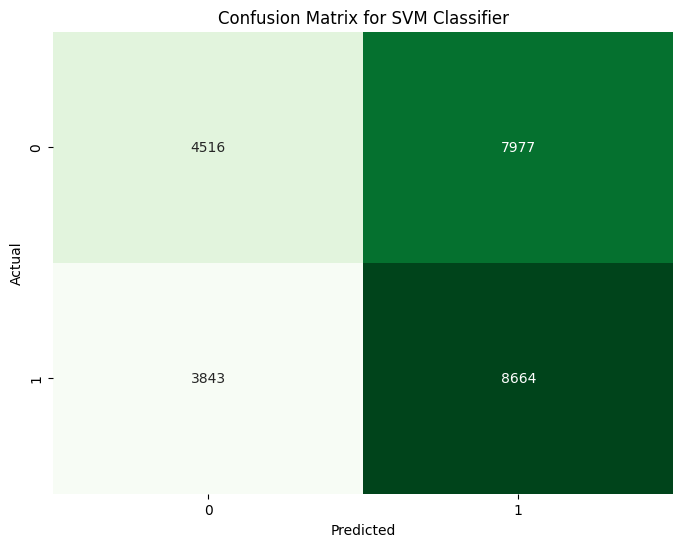
accuracy **=** accuracy\_score(y\_test, y\_pred)

print(f"Accuracy Score of SVM Classifier: {accuracy}")

*# Displaying detailed performance metrics*

print("\nClassification Report:")

print(classification\_report(y\_test, y\_pred))



Accuracy Score of SVM Classifier: 0.5272

Classification Report:

precision recall f1-score support

0 0.54 0.36 0.43 12493

1 0.52 0.69 0.59 12507

accuracy 0.53 25000

macro avg 0.53 0.53 0.51 25000

weighted avg 0.53 0.53 0.51 25000

K-Means cluster

In [19]:

**from** sklearn.cluster **import** KMeans

**from** sklearn.preprocessing **import** StandardScaler

**from** sklearn.decomposition **import** PCA *# for dimensionality reduction*

*# Standardizing the data*

scaler **=** StandardScaler()

X\_scaled **=** scaler**.**fit\_transform(X)

*# Performing the K-Means clustering*

n\_clusters **=** 3 *# Define the number of clusters*

kmeans **=** KMeans(n\_clusters**=**n\_clusters, random\_state**=**42)

kmeans**.**fit(X\_scaled)

cluster\_labels **=** kmeans**.**labels\_

*# Reducing the dimensionality for the visualization (using PCA)*

pca **=** PCA(n\_components**=**2)

X\_pca **=** pca**.**fit\_transform(X\_scaled)

*# Plotting the clusters*

plt**.**figure(figsize**=**(10, 8))

plt**.**scatter(X\_pca[:, 0], X\_pca[:, 1], c**=**cluster\_labels, cmap**=**'viridis', s**=**50, alpha**=**0.5)

plt**.**title('K-Means Clustering')

plt**.**xlabel('Principal Component 1')

plt**.**ylabel('Principal Component 2')

*# Plotting the cluster centers*

centers **=** kmeans**.**cluster\_centers\_

plt**.**scatter(centers[:, 0], centers[:, 1], marker**=**'o', c**=**'red', s**=**200, label**=**'Cluster Centers')

plt**.**legend()

plt**.**colorbar(label**=**'Cluster')

plt**.**show()

