- **Graph Construction**: These graphs are built from the binaries' execution flow, showing the actions and interactions within the malware.
- **Feature Extraction**: Features from these graphs capture complex relationships and execution patterns, aiding in the detection of sophisticated malware.

Chapter 9 provides a detailed discussion on constructing function call graphs, extracting features, and using Graph Neural Networks (GNNs) for malware detection. The datasets are publicly available and are reported in Table 3.5.

Table 3.5: Datasets and Their Availability Sources

Dataset	Availability Source
Windows Malware Dataset	https://forms.gle/iDqw9jGv84tqgxwZ8
Android Malware Dataset	https://t.ly/vDRy9
Image-Based Malware Feature Dataset	https://forms.gle/iDqw9jGv84tqgxwZ8
Function Call Graph (FCG) Dataset	https://forms.gle/iDqw9jGv84tqgxwZ8

3.7 Discussion

In this chapter, the importance of malware datasets and their pivotal role in evaluating malware detection systems are explored. Assessing malware detection systems requires a robust foundation of diverse and representative datasets for thorough testing and validation. Recognizing this need, two novel malware feature datasets tailored for distinct platforms—Windows and Android—are introduced. The TUMALWD dataset is meticulously curated for the Windows platform, incorporating host-level and network-level features. By encompassing a wide variety of malware characteristics, this dataset enables a more thorough and effective evaluation of malware detection systems. For the Android platform, the TUANDROMD dataset is presented. This dataset consists of permission and API-based features, acknowledging the distinct attributes of the Android platform. Together, these datasets serve as invaluable resources for benchmarking and refining malware detection systems and contribute to a more nuanced understanding of the challenges posed by malicious activities across different operating systems.

Chapter 4

An Enhanced Feature Selection Method for Imprecise Data

4.1 Introduction

In machine learning and data science, a dataset comprises of a set of data points, each of which is represented by a set of numbers or values. These numbers are called variables, attributes, or features of the data point. The cardinality of the feature set is also called the dimensionality of the dataset. The dimensionality of data plays an important role while building machine learning models. When the dimensionality of data is very high, a set of issues known as the curse of dimensionality arises which affects the performance of the learning model [35]. In addition, the high dimensional feature space make the learning model overfitted and also increases the memory requirements and computational cost. To address these issues, feature selection is used as a data preprocessing approach to reduce the dimensionality [36] [37]. All the features present in the real world are not useful; the feature selection strategy helps to select a relevant feature subset using some evaluation criteria [38] [39] [40] [41]. The feature selection has several advantages like it improves the performance and generalization ability of the learning model, the computational efficiency, and decreases the memory requirements.
Feature selection algorithms are classified based on class label information and selection strategies [42]. Based on the availability of ground truth knowledge, there are three types of such methods, namely, supervised, unsupervised, and semi-supervised. Supervised approach [43] aims to select a relevant feature subset by taking into consideration of the class label information. The feature

relevance is usually calculated by using various measures like mutual information and correlation. The selected feature subset helps to build a better learning model. Unsupervised approach [44] [45] [46] produces a relevant feature subset from unlabeled data. Since the ground truth knowledge is not present, the feature relevance is usually calculated using different measures such as data similarity, and local discriminative information. The selected subset of features is able to extract clusters of all instances. Semi-supervised approach [47] [48] [49] generates a relevant feature subset by taking into consideration of both labeled and unlabeled samples. It is similar to the supervised approach except it uses the partial label information.
Based on the selection strategy there are three types of feature selection techniques, namely, filter, wrapper, and embedded [37]. Filter methods [50] produce the optimal feature subset by studying the characteristics of data using some statistical criteria. It does not use the learning algorithms during the feature selection phase which makes them computationally efficient. On the other hand, wrapper methods [51] use the learning algorithms as selection criteria. The wrapper method selects an optimal feature subset in such a way that it achieves improved predictive accuracy for a given classifier. It is computationally very intensive when the dimension of the dataset is very high and also biased to the given classifier. The wrapper methods are considered to be performed better than filter methods but it is computationally expensive than filter methods. The embedded methods [52] are intermediate warppers and filters. It exploit the benefits of both the methods. First, it selects subsets of candidate features using some statistical measure like filter methods and then it selects the optimal subset with the highest classification accuracy. It is computationally less expensive than a wrapper.

4.1.1 Motivation

The motivation for implementing a feature selection method is to create more efficient and interpretable machine learning models. Many datasets contain numerous features, some of which are redundant or minimally contribute to model performance. By selecting the most informative features, model accuracy can be improved, computational complexity reduced, and data patterns better understood. This enhances the overall efficiency of machine learning models for practical applications. In malware detection, feature selection is crucial as it helps identify key characteristics that distinguish malware from goodware. Focusing on these key features improves detection accuracy and speed, essential for combating evolving malware threats.

4.1.2 Contribution

Rough set theory is a mathematical approach used to handle the uncertainty present in data. It has been widely applied in various feature selection methods. This chapter introduces a supervised filter-based feature selector that leverages rough set theory. The method utilizes class-label information to calculate the significance of each feature, introducing a new criterion for identifying the most relevant features. The proposed feature selection method, referred to as FSR, has demonstrated superior performance compared to other competing methods. Its effectiveness has been evaluated on datasets related to malware and malware-based attacks, where it has shown promising results.

4.2 Background

Feature selection is a crucial step in the process of preparing data for machine learning models. It involves choosing a subset of relevant features from the original set of features to improve model performance, reduce overfitting, and enhance interpretability. Feature selection approaches are classified into four categories as shown in figure 4-1, such as filter approach, wrapper approach, embedded approach, and hybrid approach.

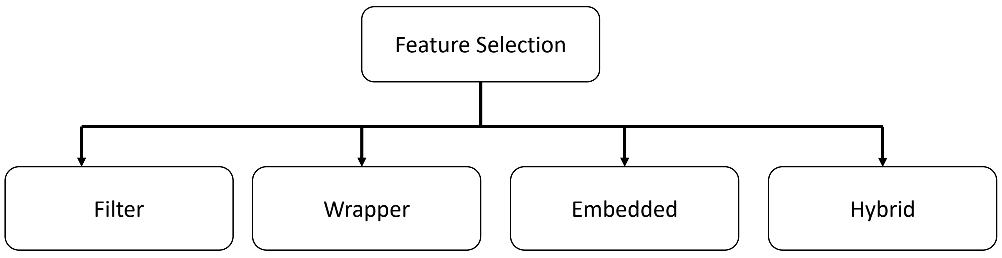


Figure 4-1: Feature selection methods

4.2.1 Filter Methods

Filter methods are a class of feature selection techniques that operate independently of a particular machine learning algorithm, focusing on the inherent characteristics of individual features. These methods involve evaluating each feature's relevance to the target variable through statistical measures or heuristics. For instance, correlation-based filter methods assess the statistical correlation between each feature and the target variable, ranking or selecting features based on these relationships. Variance thresholding, another filter method, eliminates features with low variance, assuming they contribute less information. Filter methods are computationally efficient and serve as a preprocessing step before model training, aiding in dimensionality reduction by selecting a subset of the most informative features. While filter methods may overlook feature interactions, their speed and simplicity make them particularly valuable for high-dimensional datasets where a quick assessment of feature importance is needed.

4.2.2 Wrapper Methods

Wrapper methods for feature selection involve evaluating the performance of a machine learning model with different subsets of features. Unlike filter methods, wrapper methods incorporate the actual learning algorithm in the evaluation process, using a specific model's performance as the criterion for feature selection. Recursive Feature Elimination (RFE) is a common wrapper method that iteratively removes the least important features based on the model's performance until a desired number of features is reached. Forward selection adds features one at a time, selecting the one that maximizes model performance at each step. Backward elimination starts with all features and eliminates one at a time based on their impact on performance. Wrapper methods are computationally more intensive than filter methods since they involve training the model multiple times, but they can capture feature interactions and dependencies more effectively, making them suitable for scenarios where the relationship between features is crucial for accurate modeling.

4.2.3 Embedded Methods

Embedded feature selection methods seamlessly integrate the feature selection process into the model training itself. These methods optimize both the model's predictive performance and the relevance of features simultaneously. Lasso regression, a popular embedded method, introduces a regularization term based on the absolute values of the coefficients during the model training process. This encourages sparsity in the coefficient values, effectively performing feature selection by driving some coefficients to zero. Similarly, tree-based methods, such as Random Forest and Gradient Boosting, inherently perform feature selection as they build decision trees by selecting features that contribute the most to reducing impurity or error. Embedded methods are advantageous because they consider feature importance within the context of the specific algorithm, making them more sensitive to the model's intricacies and potentially resulting in more accurate and efficient feature selection.

4.2.4 Hybrid Methods

Hybrid feature selection methods combine elements from different types of feature selection approaches to capitalize on their respective strengths. These methods often leverage the advantages of filter, wrapper, or embedded techniques to create a more robust and effective feature selection strategy. For example, genetic algorithms, a popular hybrid approach, simulate the process of natural selection and evolution to optimize feature subsets based on their performance with a particular model. Simulated annealing, another hybrid method, iteratively explores the solution space, allowing moves that decrease performance to escape local optima. By integrating diverse techniques, hybrid methods aim to exploit the complementary nature of different feature selection strategies, providing a more comprehensive and flexible approach to identifying the most relevant features for a given machine learning task.

Feature selection approaches are also classified into supervised, unsupervised, and semi-supervised categories as shown in figure 4-2 based on the availability of labeled data and the nature of the learning task. The classification serves to highlight the different strategies and considerations involved in feature selection, depending on the context of the problem being addressed.

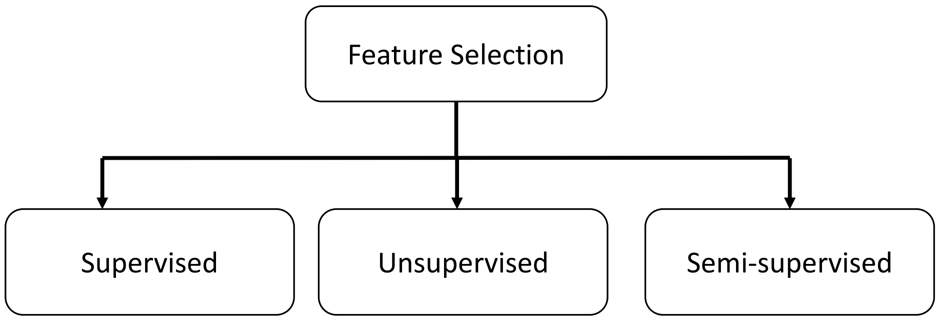


Figure 4-2: Feature selection methods

4.2.5 Supervised Method

Supervised feature selection methods take advantage of labeled training data, where both the input features and corresponding target variables are known. These methods assess the relevance of features with respect to the target variable and aim to retain those that contribute most to predictive accuracy. Common techniques include filter methods, wrapper methods, and embedded methods. Filter methods evaluate feature importance independently of the learning algorithm, wrapper methods use the predictive performance of a specific model for selection, and embedded methods incorporate feature selection within the model training process. Supervised feature selection is particularly useful in classification and regression tasks where the goal is to build predictive models.

4.2.6 Unsupervised Method

Unsupervised feature selection methods operate in scenarios where the data lacks labeled data. Instead of relying on explicit class information, these methods explore the inherent structure and relationships within the feature space. Clustering-based approaches, such as hierarchical clustering or k-means clustering, can be employed to group similar features and identify representative ones from each cluster. Additionally, techniques like Principal Component Analysis (PCA) and autoencoders aim to capture the most important information in the data without requiring labeled information. Unsupervised feature selection is valuable in exploratory data analysis and dimensionality reduction tasks.

4.2.7 Semi-supervised Method

Semi-supervised feature selection methods strike a balance between the availability of labeled and unlabeled data. In scenarios where obtaining labeled data is expensive or time-consuming, these methods leverage a combination of labeled and unlabeled samples. Techniques like self-training or co-training use the available labeled data to iteratively label unlabeled instances and refine feature selection. This approach is particularly beneficial when only a fraction of the data is labeled, making it more adaptable to real-world situations where obtaining fully labeled datasets can be challenging. Semi-supervised feature selection bridges the gap between the advantages of supervised and unsupervised methods, offering a flexible solution for various applications.

Each type of feature selection approach has its strengths and weaknesses, and the choice depends on the specific characteristics of the dataset and the goals of the analysis. Smith et. al. introduces a hybrid feature selection method combining genetic algorithms and classification performance evaluation. The authors propose an evolutionary approach to optimize feature subsets, demonstrating improved classification accuracy on various datasets. Xi et. al. explores sparse feature learning within Convolutional Neural Networks (CNNs). By leveraging the sparsity-inducing properties, the authors demonstrate enhanced interpretability and efficiency in extracting discriminative features from image data. Zhang et. al introduces an embedded feature selection method within deep neural networks for natural language processing tasks. By integrating feature selection within the network architecture, the authors enhance the model's ability to capture salient linguistic features. Mitchell et. al addresses feature selection in financial time series forecasting using an ensemble learning approach. The authors propose an algorithm that combines multiple feature selection methods to improve the stability and accuracy of predictions in dynamic financial markets. Ryan C. Turner et. al presents a semi-supervised feature selection method tailored for cybersecurity applications, with a case study in intrusion detection. The authors leverage both labeled and unlabeled data to identify relevant features, enhancing the robustness of intrusion detection systems. Ren et. al proposes a feature selection method based on regularized regression. The authors address the challenges of high-dimensional remote sensing datasets and demonstrate the efficiency of their approach in extracting relevant features for accurate environmental monitoring.

4.3 Problem Statement

For a given dataset D of size $M \times N$ with M instances and N features, with a feature set F $=\{f_1, f_2, f_3, ..., f_N\}$, the goal is to find an optimal feature subset F' such that $F' \subset F$ and for F', the learning model will give the best possible generalization performance. In other words, we aim to reduce the dimension of the dataset D by removing the unimportant features without compromising the performance of the learning model. The reduced feature set is obtained by minimizing the redundancies among features and maximizing the feature-class relevance.

4.4 Proposed Method: FSR

The proposed method selects the optimal feature subset based on rough set. The method uses the concepts of indiscernibility relation, approximation of sets, and attribute dependency for the generation of the optimal feature subset. The theoretical basics that are used in the design of FSR are described next.

4.4.1 Rough Set

A mathematical procedure that is used in data mining for the analysis of imprecise, vague and uncertain data [53]. The indiscernibility relation is the main basis of the rough set theory. Data objects that share the same knowledge or information are called as indiscernible. There exists some boundary line elements in each rough set and these elements can neither be the members of the set or its complement with certainty. A non-empty boundary region defines the target set as rough set otherwise, it is a crisp set [54].

For a given dataset in a tabular form where each row represents a data object or instance and each column represents the attribute value of each data instance. Such table, in the concepts of rough set theory is termed as data or information table. More formally, the pair $I = (U, A)$ is the data table, here U represents the universal set which is finite and A represents the set of attibutes such that $x : U \times A \rightarrow V_x$, where $x \in A$ and V_x is the set of values of a.

4.4.1.1 Indiscernibility Relation

For a given set B⊆A of an $I = (U, A)$, the indiscernibility realtion I_B is stated as follows.

$$I_B{=}\{(x,y)\epsilon U^2|\ \forall a\,\epsilon\, B, a(x) = a(y)\}$$

where, $a(x)$ is the attribute value for the element x. If any $(x, y)\ \epsilon\ I_B$, then x, y are cannot be distinguishable by B. The relation I_B is also called an equivalence relation. The U/B or U/I_B i.e., the partition of U based on B can be stated as follows.

$$\mathrm{U}/I_B{=}\{[x_i]_B \mid x_i\ \epsilon\ U\}$$

where, $[x_i]_B$ is equivalence class of I_B. The equivalence classes of I_B and the empty set ϕ are referred to as $B-$ *elementary concepts*.

4.4.1.2 Approximation of sets

For a given data or information table $I = (U, A)$, let $B \subseteq A$ and $X \subseteq U$, the set X cannot be expressed precisely based on the knowledge available in B but it can be approximated by the formation of two crisp sets namely, $B - Upper(\overline{B}X)$ and $B - Lower(\underline{B}X)$ approximations of X respectively.

The $\underline{B}X$ consists of all the elementary sets which are subsets of X. This is stated as follows.

$$\underline{B}X{=}\cup\{[x_i]_B \mid [x_i]_B \subseteq X\}$$

In other words, the $\underline{B}X$ consists of all the elements that belong to the target set X with probability 1. The $\underline{B}X$ is also known as the positive region of X. Based on this, the set approximation accuracy is calculated as follows.

$$\alpha_B(X) = \frac{|\underline{B}X|}{|\overline{B}X|}$$

The $\overline{B}X$ comprises a set of elementary sets and the intersection of it with the target set X is non-empty. This is stated as follows.

$$\overline{B}X = \cup \{[x_i]_B \mid [x_i]_B \cap X \neq \phi\}$$

In other words, $\overline{B}X$ comprises a set of elements that belong to the target set X with some probability. The set $\overline{B}X$ is also called the negative region.

The difference $\overline{B}X$ - $\underline{B}X$ which is denoted as $BR(X)$ that contains the boundary elements of X. The $BR(X)$ comprises all the elements that are not possible to assign to the target set X or its complement based on the information contained in B.

4.4.1.3 Dependency of attributes

For a given dataset or data table, if it contains the class label for each data object present in the system, then the attribute set can be distinguished into two sets namely, conditional and decision attributes. The data objects are uniquely classified into one of the decision attribute values based on information contained in conditional attributes. However, sometimes it is not possible to predict the ground truth of the data objects with the available conditional attribute values but can be determined with some approximations. Approximations help determine the relationship between conditional and decision attributes or any two attribute variables. For a given I(U, A), here $A = C \cup D$, the dependency of D (dependency attribute) on C (conditional attributes) can be stated as follows.

$$\gamma(C, D) = \frac{|POS_C(D)|}{|U|} \tag{4.1}$$

where $POS_C(D) = \cup \underline{C}X$ i.e., the members of U/C which are positively belong to the partition of U/D and — . — is the length of the set. Depending on the value of $\gamma(C, D)$, there are three types of dependency as follows.

$\gamma(C, D)$=1, D completely dependent on C
0¡$\gamma(C, D)$¡1, D partially dependent on C
$\gamma(C, D)$=0, D does not depends on C

Similarly, the dependency measure also helps to determine the feature or attribute's significance pertaining to the decision attribute. The change in dependency is the measure of an attribute's significance after it has been deleted from the conditional attribute set. For example, consider an attribute $A \epsilon C$ and its significance is calculated as follows:

$$\delta_C(D, A) = \gamma(C, D) - \gamma(C - A, D) \tag{4.2}$$

The zero value of $\delta_C(D, A)$ indicates that the attribute A is not useful. The higher value of $\delta_C(D, A)$ signifies that the attribute a is of higher significance and the lower value indicates the attribute is less significant.

Example: Consider a data table (U,A) which is shown in Table 4.1. Here,

Table 4.1: A data table

	A1	A2	A3	Label
D_1	1	2	0	C
D_2	1	2	0	C
D_3	2	0	0	D
D_4	0	0	1	D
D_5	2	1	0	C
D_6	0	0	1	D
D_7	2	0	0	C
D_8	2	1	2	D
D_9	2	1	0	D
D_{10}	2	0	0	C

U=$D_1, D_2, ..., D_{10}$ and $A = C \cup D$ where C=$\{A_1, A_2, A_3\}$ and D=$\{Label\}$. According to indiscernibility relation, the set U can be partitioned based on different subsets of attributes which are shown below.

- U/$I_{\{A_1\}}$=$\{\{D_1, D_2\}, \{D_3, D_5, D_7, D_8, D_9, D_{10}\}, \{D_4, D_6\}\}$
- U/$I_{\{A_2\}}$=$\{\{D_1, D_2\}, \{D_3, D_4, D_6, D_7, D_{10}\}, \{D_5, D_8, D_9\}\}$
- U/$I_{\{A_3\}}$=$\{\{D_1, D_2, D_3, D_5, D_7, D_9, D_{10}\}, \{D_4, D_6\}\{D_8\}\}$
- U/$I_{\{A_1,A_2\}}$=$\{\{D_1, D_2\}, \{D_3, D_7, D_{10}\}, \{D_4, D_6\}, \{D_5, D_9\}, \{D_8\}\}$
- U/$I_{\{A_1,A_3\}}$=$\{\{D_1, D_2\}, \{D_3, D_5, D_7, D_9, D_{10}\}, \{D_4, D_6\}, \{D_8\}\}$
- U/$I_{\{A_2,A_3\}}$=$\{\{D_1, D_2\}, \{D_3, D_7, D_10\}, \{D_4, D_6\}, \{D_5, D_9\}, \{D_8\}\}$
- U/$I_{\{A_1,A_2,A_3\}}$=U/C=$\{\{D_1, D_2\}, \{D_3, D_7, D_{10}\}, \{D_4, D_6\}, \{D_5, D_9\}, \{D_8\}\}$
- U/$I_{\{Label\}}$=U/D=$\{\{D_1, D_2, D_5, D_7, D_{10}\}, \{D_3, D_4, D_6, D_8, D_9\}\}$

Consider a target set X={D $|Label(D) = C\}and$C$\subset$ A, the approximation sets–upper and lower approximation ($\overline{C}X$ *and* $\underline{C}X$) are created using the available knowledge contained in C.

$$\underline{C}X = \{D_1, D_2, D_4, D_6\}$$
$$\overline{C}X = \{D_1, D_2, D_3, D_4, D_5, D_6, D_7, D_9, D_{10}\}$$

Based on lower and upper approximation sets, we can identify boundary region elements of X as follows.

$$BR(X) = \overline{C}X - \underline{C}X = \{D_3, D_5, D_7, D_9, D_{10}\}$$

The positive region $POS_C(D)$ consists of all the blocks of the partition U/C which uniquely belong to the partition U/D.

$$POS_C(D) = \{\{D_1, D_2\}, \{D_4, D_6\}, \{D_8\}\} = \{D_1, D_2, D_4, D_6, D_8\}$$

The dependency between D and C can be computed using the Equation 4.1.

$$\gamma(C, D) = \frac{|POS_C(D)|}{|U|} = \frac{5}{10} = 0.5$$

Similarly, the relevance of each feature, $A_i \,\epsilon\, C$ can be computed using the Equation 4.1 and report in Table 4.2 below.

Table 4.2: Relevance score of each feature

$\gamma(A_1, D)$	$\gamma(A_2, D)$	$\gamma(A_3, D)$
0.4	0.2	0.3

4.4.1.4 Proposed feature selection method

The proposed method-FSR uses the concepts of rough set to select the most effective subset of features from a given feature set. Initially, we compute the relevance score $\gamma(a_i, D)$ for each feature, a_i, in respect of the decision attribute using the

Equation 4.1 and the feature with the highest feature-class relevance score is selected. The feature is then removed from the initial feature set and added into the newly selected feature subset.

If we select features based on only relevance score, then there is a possibility of inclusion of redundant features. Suppose, if two features are selected based on their high relevance score only but they are highly correlated and in that case, any one of them should be removed. Therefore, we calculate the significance score to identify the unnecessary features. To do so, for each a_j, which is an unselected feature, the significance score is calculated with respect to each selected feature, a_i.

The significance score $Z(a_i, a_j)$ of a_j in respect a_i is calculated using equation 4.2 i.e.,

$$Z(a_i, a_j) = \delta_{a_i,a_j}(D, a_j) = \gamma(a_i, a_j, D) - \gamma(a_i, D) \tag{4.3}$$

Now, for each unselected feature, we have a feature-class relevance score and a set of $k(k \geq 1)$ significance values where k is the number of selected features. The calculated values are used as input to a function given in Equation 4.4 based on which the $(k+1)^{th}$ feature is selected.

$$FS(a_j) = \gamma(a_j, D) + min((max - min)Z(a_i, a_j) + Z(a_i, a_j)) \tag{4.4}$$

The feature that gives the maximum value of $FS(a_j)$ is selected as the $(k+1)^{th}$ feature. The flow-diagram of our method FSR is depicted in Figure 4-3. For better understanding, following example is helpful.

Example: Let's consider an information system given in Table 4.1 where each object is of three features F=$\{A_1, A_2, A_3\}$. We compute the relevance scores of each feature which are given in Table 4.2. Initially, we select the feature which has highest relevance score. In our case, it is 0.4 for the feature A_1. Next, the significance score is calculated for unselected features using equation 4.3.

Selected features=$\{A_1\}$, Unselected features=$\{A_2, A_3\}$

$$Z(A_1, A_2)=\gamma(A_1, A_2, D) - \gamma(A_1, D) = 0.5 - 0.4 = 0.1$$
$$Z(A_1, A_3)=\gamma(A_1, A_3, D) - \gamma(A_1, D) = 0.5 - 0.4 = 0.1$$

Now, based on the caluclated values, a feature score is calculated for each unselected feature using Equation 4.4.

$$FS(A_2) = 0.2 + 0.1 = 0.3$$
$$FS(A_3) = 0.3 + 0.1 = 0.4$$

Next, we select the feature which gives the highest feature score $FS(a_j)$. In our case, it is A_3. So, the final order of the selected features are $\{A_1, A_3, A_2\}$. In case of tie, we select the feature with highest value of $\gamma(a_i, D)$. If both parts of the $FS(a_j)$ same for two features, then we select any one of them.

Algorithm 1: FSR

```
Input: The dataset(U,C,D), where U={$o_1, o_2, ....o_m$} is the number of
       instances, C={$f_1, f_2, ....f_n$} is the attribute set, and D=class label
Output: $C'$=the optimal feature subset
initialization;
$C'$=$\emptyset$;
for i=1 to n do
    for i=1 to m do
        compute $\gamma(f_i, D)$
    end
end
Select the feature $f_i$ which gives the maximum value i.e.,
  $f_i$=arg $max\{\gamma(f_i, D)\}$;
$C'$=$C'$+ {$f_i$};
$C$=$C$− {$f_i$};
x=1;
while $x \leq k$ do
    for each $f_j$ in C do
        for for each $f_i$ in $C'$ do
            Compute $Z(f_i, f_j)$
        end
        Compute $FS(f_j)$
    end
    Select the feature $f_i$ which gives the maximum value i.e.,
      $f_j$=arg $max\{FS(f_j)\}$;
    $C'$=$C'$+ {$f_j$};
    $C$=$C$− {$f_j$};
end
Return $C'$;
```

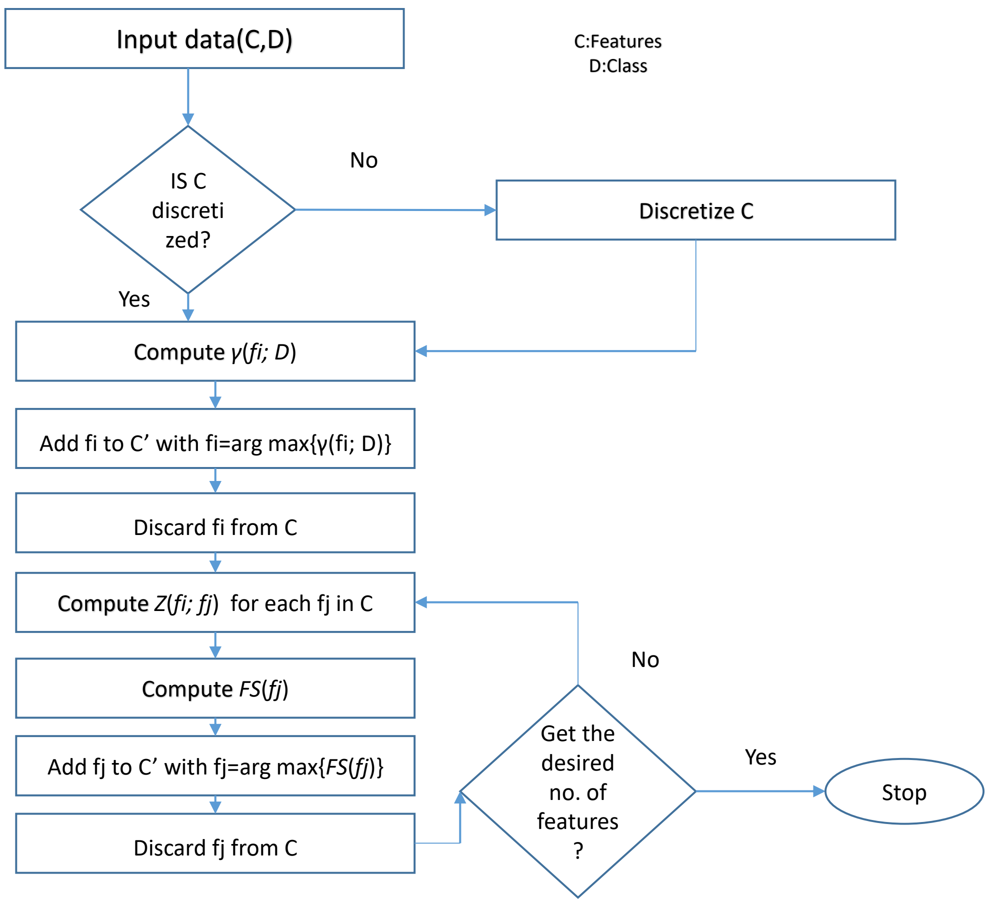

Input data(C,D)
C:Features
D:Class
IS C discretized?
No
Discretize C
Yes
Compute γ(fi; D)
Add fi to C' with fi=arg max{γ(fi; D)}
Discard fi from C
Compute Z(fi; fj) for each fj in C
Compute FS(fj)
Add fj to C' with fj=arg max{FS(fj)}
Discard fj from C
Get the desired no. of features ?
No
Yes
Stop

Figure 4-3: Workflow of the proposed method

Complexity analysis: The proposed method, FSR, is composed of two major steps a) finding the relevance score and b) the significance score. The total complexity also depends on the size of the dataset. The time complexity for computing the relevance score of n features with m instances is $O(mn)$. The time complexity for finding the top relevant feature out of n features is $O(n)$. The time complexity of calculating the significance score regarding the previously chosen k features is $O((k)(n-k))$. So, the overall complexity of FSR is $O(mn)$.

Proposition 1. *The subset identified by FSR is optimal*

Explanation: Suppose for the sake of contradiction, we assume that for a dataset, the feature subset given by FSR is not optimal and $|Featuresubset| = k$. Let's say for the $|Featuresubset| + 1$, the highest classification accuracy is obtained. However, Figure (4-4) to (4-8) shows the highest possible accuracy obtained is for the top k features. So, there is no improvement in accuracy after adding or deleting any further features. Hence, the initial non-optimality assumption is false. Hence, the feature subset is optimal.

4.4.2 Significance of FSR

There are some other methods of feature selection that use the rough set theory approach. Maji et al. [55] use a rough set approach to select the relevant features. But unlike our method, the author calculate significance score $Z(a_i, a_j)$ of each unselected feature a_j with respect to the k selected features as average value i.e., $\frac{1}{k}\sum_{i=1}^{k} Z(a_i, a_j)$. In this case, if an unselected feature has zero value of Z with one of the selected features i.e, the unselected feature is redundant and if it has a very large value of Z with one of the selected features, then the significance score $Z(a_i, a_j)$ of unselected feature will be high which result in the selection of the feature though it is redundant. Similarly, in another literature, the significance score is calculated as $\frac{min(Z(a_i,a_j))}{max(Z(a_i,a_j))}\, min(Z(a_i, a_j))$ for each unselected feature a_j with respect to the k selected features where $i = 1$ to k. In this case, if an unselected feature has equal values of significance score with all of the selected features, then that feature will always get preference over the other important unselected features (if unselected features have equal values of $\gamma(a_j, D)$).

Similarly, in [56], an incremental feature selector is proposed that uses the rough set theory like our method. But they also use a genetic algorithm to select the optimal and relevant feature subset. Unlike our method, the proposed algorithm has polynomial time complexity which limits its usability when the size of the dataset is large. Zhang et al. [57] propose a feature selector that uses a fuzzy rough set-theoretic approach for feature selection. They use a filter-wrapper approach along with a novel entropy measure to select the optimal subset of features. Since they use a hybrid method for feature selection, the time complexity is high compared to our proposed method.

4.5 Performance Analysis

FSR has been implemented in Python using a Dell Precision 7810 workstation with 2x Intel Xeon (R) W-2145 comprising 8 cores, 64GB RAM, NVIDIA Tesla K80 GPU with 12GB VRAM, and Ubuntu OS. Materials used and performance achieved are discussed next.

Table 4.3: Dataset details

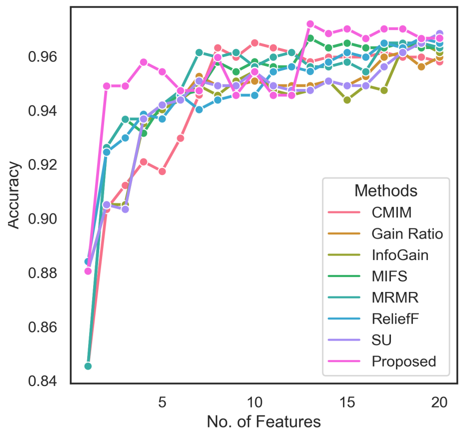
Dataset	# Instances	# Features	# Classes
Sonar	208	60	2
Parkinson	756	754	2
Breast Cancer Wisconsin	569	32	2
Ionosphere	351	34	2
Wine	178	13	3
Pima	768	8	2
Seed	210	7	3
IRIS	150	4	3
TUANDROMD	4465	241	2
XSSD	1301	11	2

4.5.1 Datasets and Preprocessing

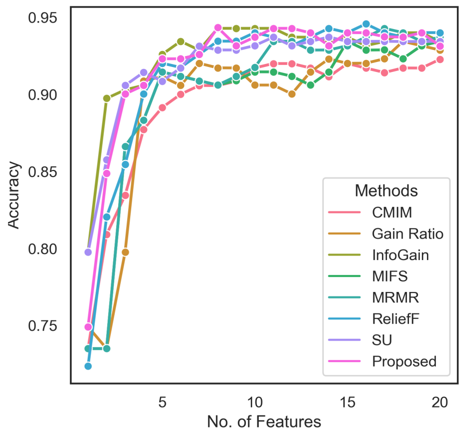
To understand the effectiveness of our method, it has been tested on ten real-world datasets, including two datasets focused on malware attacks. This allows us to observe the method's effectiveness in both malware and malware-based attack contexts. The attributes of these datasets include both numeric and categorical values. The dimensionality of these datasets are ≥ 4. The datasets except the XSSD are collected from UCI repository [58]. The XSSD dataset is collected from [59]. The detailed description of these datasets are given in Table 6.3. We use the data discretization technique [60] to convert the continuous attribute values to discrete values since the rough set theory deals with discrete-valued attributes.

4.5.2 Result Analysis and Performance Comparison

The performance of our method is assessed based on the classification accuracy. For this, we have used the Random forest classifier. The 10-fold cross validation is used to estimate the performace of the classifier on the selected features. We also compare our method with seven filter-based feature selectors, namely, Gain Ratio, Maximum Relevance Minimum Redundancy (MRMR), Information Gain, Conditional Mutual Information Maximization Criterion (CMIM), ReliefF, Mutual Information Feature Selection (MIFS), and Symmetric Uncertainty (SU) . The comparison results of FSR with the other feature selectors for the ten datasets are shown in Figure 4-4-4-8. From the experimental results, it can be observed that the proposed method is a top performer on most of the datasets or atleast at par with the seven other competing methods. The size of the optimal feature subset selected by the FSR varies for different datasets because it depends on a number of features (or dimensions), their relevance to the given class labels, and

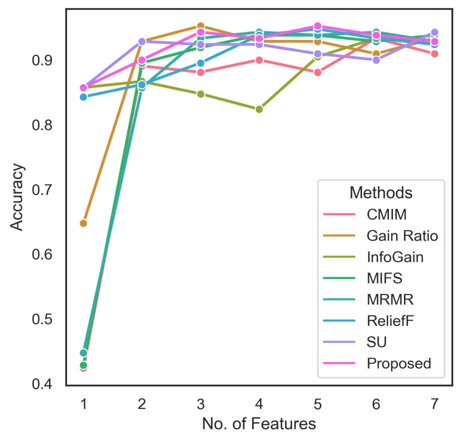


(a) Accuracy on breast cancer dataset

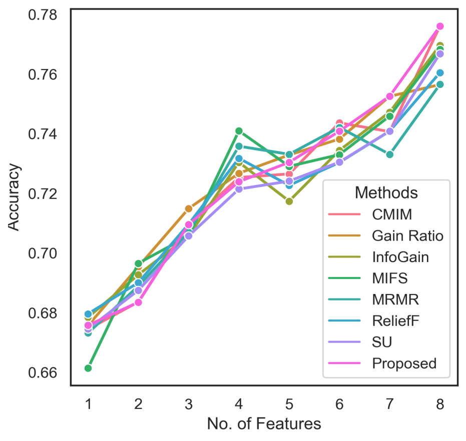


(b) Accuracy on ionosphere dataset

Figure 4-4: Performance of FSR in terms of accuracy



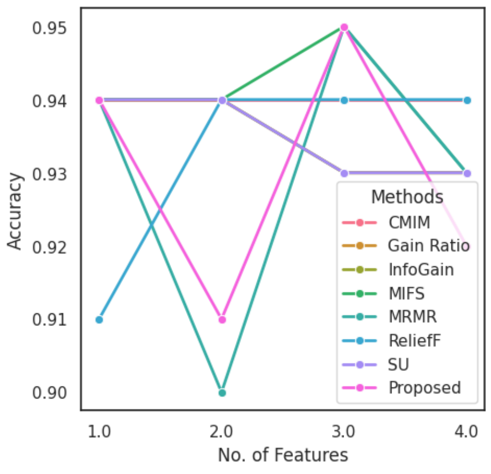
(a) Accuracy on seed dataset



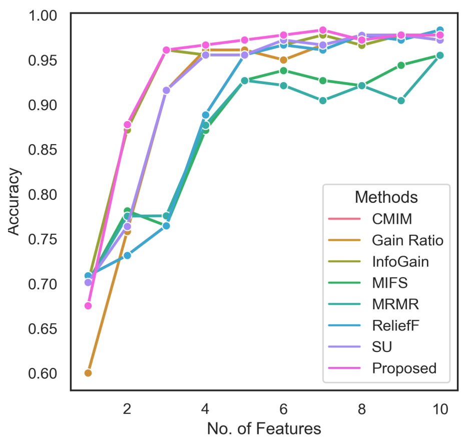
(b) Accuracy on pima dataset

Figure 4-5: Performance of FSR in terms of accuracy

feature-feature correlations or redundancies. The top k features selected by each of the eight methods including the proposed one are shown in Table 4.4a - 4.8. As we can see from the table some common features are selected by each method.

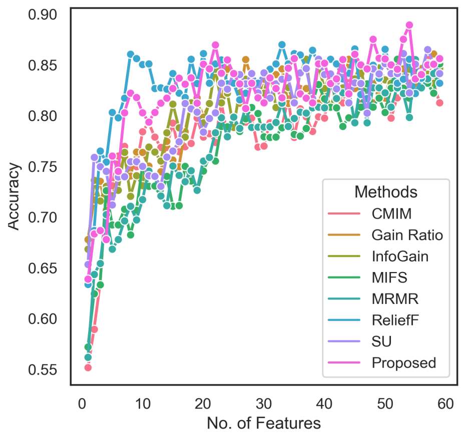


(a) Accuracy on IRIS dataset

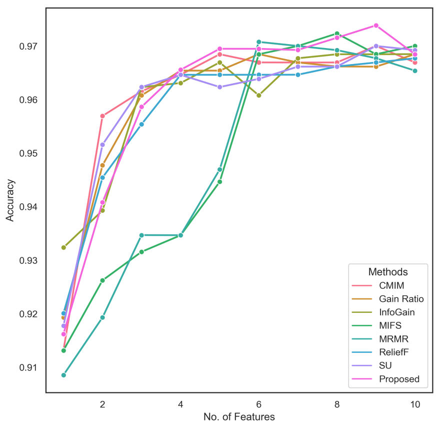


(b) Accuracy on wine dataset

Figure 4-6: Performance of FSR in terms of accuracy

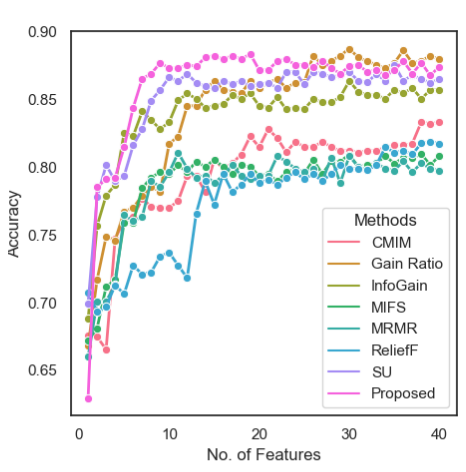


(a) Accuracy on sonar dataset

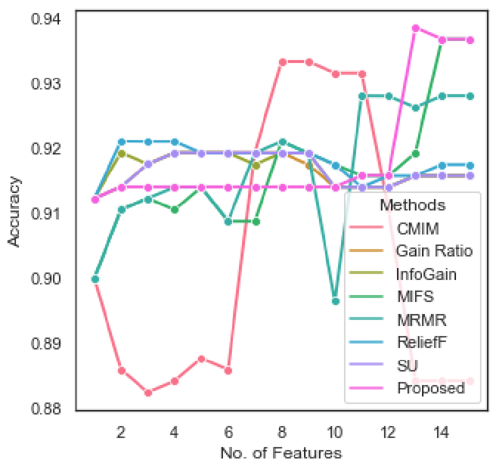


(b) Accuracy on XSSD dataset

Figure 4-7: Performance of FSR in terms of accuracy

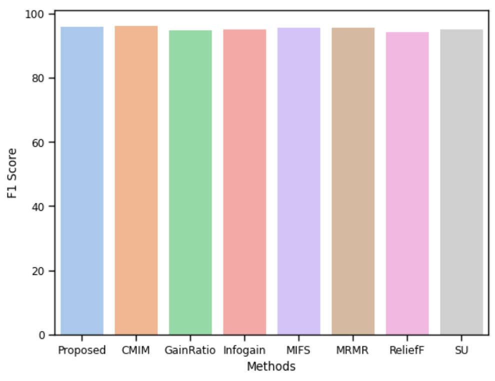


(a) Accuracy on parkinson dataset

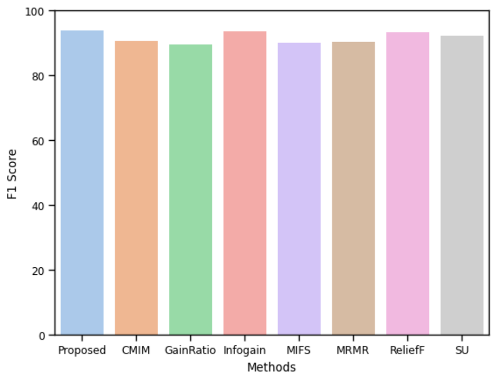


(b) Accuracy on TUNADROMD

Figure 4-8: Performance of FSR in terms of accuracy

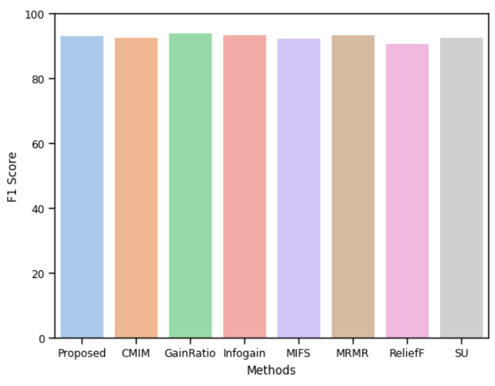


(a) F1 score on breast cancer dataset

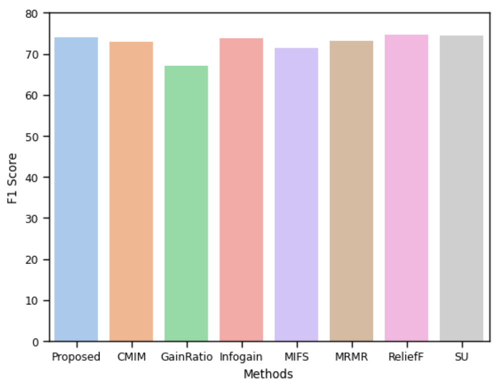


(b) F1 score on ionosphere dataset

Figure 4-9: Performance of FSR in terms of F1 score

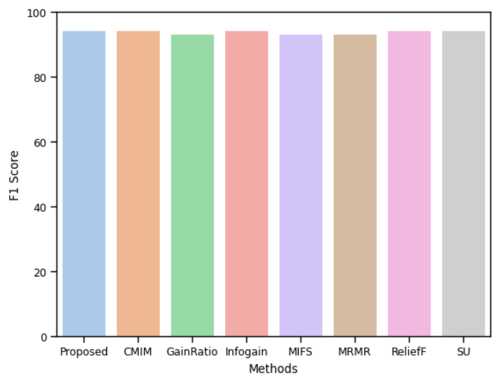


(a) F1 score on seed dataset

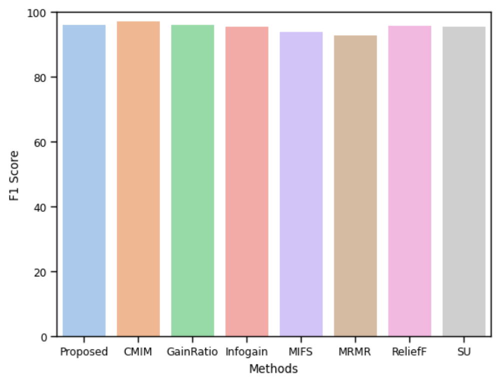


(b) F1 score on pima dataset

Figure 4-10: Performance of FSR in terms of F1 score

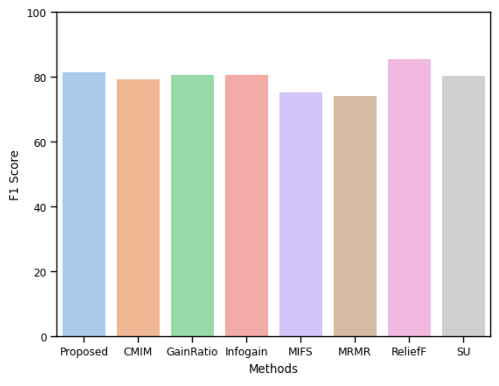


(a) F1 score on IRIS dataset

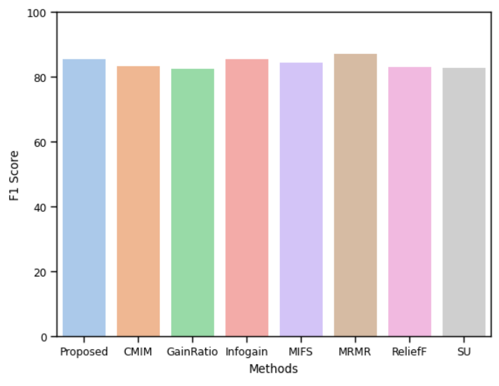


(b) F1 score on wine dataset

Figure 4-11: Performance of FSR in terms of F1 score



(a) F1 score on sonar dataset



(b) F1 score on XSS dataset

Figure 4-12: Performance of FSR in terms of F1 score

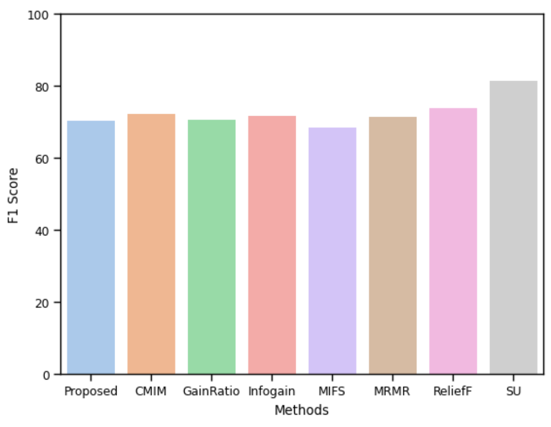


Figure 4-13: Performance of FSR in terms of F1 score for Parkinson dataset

(a) Breast cancer dataset

Methods	Selected features
Proposed	22,24,4,29,8,28,19,15,9,14
CMIM	7,0,6,8,12,15,20,21,25,26
Gain Ratio	22,20,23,27,7,6,26,2,3,0
Infogain	22,23,20,27,7,2,3,0,6,13
MIFS	7,20,24,8,0,4,27,1,18,9
MRMR	7,20,24,0,8,27,1,4,17,2
ReliefF	23,3,22,13,2,21,20,1,0,12
SU	22,20,23,27,7,2,6,3,0,26

(b) Ionosphere dataset

Methods	Selected features
Proposed	0,3,1,2,13,15,9,7,4, 5
CMIM	26,14,2,23,6,4,7,18,32,21
Gain Ratio	0,26,16,3,5,18,22,31,4,25
Infogain	3,4,31,27,1,19,32,6,11,5
MIFS	26,0,3,1,5,13,27,29,7,11
MRMR	26,0,3,1,5,27,13,29,7,31
ReliefF	22,6,25,3,1,32,12,27,4,5
SU	3,5,4,26,31,27,1,25,19,32

Table 4.4: Top ten features of the datasets

(a) IRIS dataset

Methods	Selected features
Proposed	2, 3, 1, 0
CMIM	3, 0, 2, 1
Gain Ratio	0,2, 3, 1
Infogain	3,0,2,1
MIFS	1, 2, 3, 0
MRMR	1, 2, 3, 0
ReliefF	3, 0, 2, 1
SU	2,0,3, 1

(b) Pima dataset

Methods	Selected features
Proposed	0, 5, 7, 1, 2
CMIM	6, 1, 2, 5, 7
Gain Ratio	1, 5, 7, 0, 4
Infogain	1, 5, 7, 4, 3
MIFS	6, 0, 3, 2, 7
MRMR	6, 0, 3, 2, 7
ReliefF	1, 4, 7, 3, 5
SU	1, 5, 7, 4, 0

Table 4.5: Top five features of the datasets

(a) Seed dataset

Methods	Selected features
Proposed	0, 5, 6, 1, 3
CMIM	5, 0, 2, 3, 4
Gain Ratio	6, 0, 3, 1, 4
Infogain	0, 1, 4, 3, 6
MIFS	5, 1, 6, 4, 3
MRMR	5, 6, 1, 4, 3
ReliefF	0, 1, 5, 6, 3
SU	0, 6, 1, 4, 3

(b) Wine dataset

Methods	Selected features
Proposed	9, 12, 6, 0, 10
CMIM	12, 4, 3, 9, 6
Gain Ratio	11, 6, 9, 12, 0
Infogain	6, 12, 9, 11, 10
MIFS	6, 3, 7, 4, 10
MRMR	6, 7, 3, 4, 10
ReliefF	12, 4, 3, 9, 6
SU	6, 11, 9, 12, 0

Table 4.6: Top five features of the datasets

(a) Sonar dataset

Methods	Selected features
Proposed	11,35,29,36,30,31,23,15,16,27
CMIM	9,5,6,11,16,32,3,7,17,21
Gain Ratio	10,11,8,43, 12,53,9,44, 46,47,
Infogain	10,11,8,9,12,47,48,50,46,44
MIFS	9, 26, 59, 58, 56, 55, 57, 54, 53, 52
MRMR	9, 59, 58, 56, 57, 55, 54, 53, 52, 51
ReliefF	11, 35, 10, 9, 20, 8, 44, 36, 30, 19
SU	10,11, 8, 9,12,47,48,44,43,46

(b) XSSD dataset

Methods	Selected features
Proposed	3, 6, 4, 0, 8, 2, 9, 1, 7, 5
CMIM	2, 8, 3, 9, 0, 6, 1, 7, 5, 4
Gain Ratio	0, 8, 6, 1, 3, 9, 7, 2, 4, 5
Infogain	6, 3, 8, 0, 1, 7, 9, 2, 4, 5
MIFS	2, 8, 3, 9, 0, 6, 1, 7, 5, 4
MRMR	2, 4, 7, 5, 1, 9, 6, 8, 0, 3
ReliefF	0, 2, 8, 9, 6, 1, 3, 7, 5, 4
SU	0, 8, 6, 3, 1, 7, 9, 2, 4, 5

Table 4.7: Top ten features of the datasets

Table 4.8: Top ten features for the Parkinson dataset

Methods	Selected features
Proposed	116,58,118,132,404,419,427,121,23,134
CMIM	6, 4, 5, 10, 11, 139, 541, 565, 567, 568
Gain Ratio	532,389,272,339,354,355,746,356,345,419
Infogain	126,135,133,112,477,59,476,134,136,137
MIFS	6, 0, 230, 33, 505, 35, 10, 139, 517, 11
MRMR	6, 0, 230, 33, 505, 35, 517, 10, 139, 516
ReliefF	199, 198, 197, 196, 195, 194, 193, 192, 191, 190
SU	58,440,404,368,584,132,347,125,476,9