RMAM: A Rule-based Filter Method with Modified Huffman Ranking for Variable Selection in Supervised Learning

Talib Alshehhi[1*], Yingjie Yang[1], Feng Chen[1], Aladdin Ayesh[2]
[1]De Montfort University, Faculty of Computing Engineering and Media, Institute of Artificial Intelligence, School of Computer Science and Informatics, De Montfort University, Leicester, UK
[2]School of Natural and Computing Sciences, University of Aberdeen, King College Aberdeen, UK
E-mail: P2602836@my365.dmu.ac.uk, yyang@dmu.ac.uk, fengchen@dmu.ac.uk, aladdin.ayesh@abdn.ac.uk

Keywords: Dimensionality reduction, variable selection, machine learning, rules, supervised learning

Received: March 10, 2026

Most existing filter-based variable selection methods compute variable scores using static mathematical formulations over fixed contingency tables and do not provide principled cut-off mechanisms to distinguish influential from redundant variables thus leaving final subset selection to end users. To address this limitation, this paper proposes a new filter-based variable selection method called Rule Mining Analysis Method (RMAM). For classification tasks involving supervised learning, this method dynamically generates rules from the training data by iteratively selecting variable values that maximize information gain with respect to the target class. Each generated rule is assigned a weight based on rule coverage and uncovered training data instances, and variable scores are subsequently computed by aggregating the weights of all rules in which they appear. To automate variable subset determination, RMAM integrates a new Modified Huffman Ranker (MHR) that computes an adaptive cut-off threshold to distinguish essential from redundant variables. Experimental evaluation across 27 UCI benchmark datasets demonstrates that RMAM reduced variable search space by an average of 75.2% relative to the original datasets thereby outperforming Information Gain and Chi-Square methods which achieved average reductions of 44.7% and 50.0%, respectively. When evaluated using Naïve Bayes classification algorithm, RMAM achieved comparable or superior predictive performance on the majority of datasets while offering substantially smaller and more interpretable variable subsets. These results demonstrate RMAM's effectiveness as a practical variable selection approach for supervised learning applications.

Povzetek: RMAM učinkovito samodejno izbere najpomembnejše spremenljivke ter zmanjša njihovo število brez bistvenega poslabšanja napovedne uspešnosti.

1 Introduction

Variable selection using filter methods involves evaluating the relevance of variables based on statistical measures, such as correlation or mutual information, independently of the machine learning model. It helps remove irrelevant or redundant variables, improving model performance and efficiency [24]. Many filter methods are dependent on domain experts. In the process of choosing the relevant variables, domain experts typically check the set of variables generated by the filter method in an attempt to retain the most influential. This process is not easy, time-consuming, and it is not cost effective because it requires constant attention from the user especially in medical applications like dementia diagnosis [20] [26] [13]. Similar challenges regarding uncertainty reduction and adaptive decision support have also been highlighted in broader intelligent systems and control domains where robust adaptive and fuzzy-based frameworks are employed to improve decision making under uncertainty [17] [2].

To minimise the impact of these issues, the majority of current variable selection methods generate scores associated with each variable in the dataset and then using the scores they adopt a Ranker search method to sort the variables ranging from the top down. These scores are computed based on mathematical models like Chi Square Testing [CHI-SQR], which calculates the variables' scores using observed and expected probabilities in the dataset [14]. Overall, ranking of variables in supervised learning problems is an essential mechanism in variable selection methods since it sorts variables based on their associations with the target variable; this eases the final selection of which variables to keep and which ones to discard by end users [1] [22].

Users typically rely on the following assumptions when they use variable selection methods that are of type filter:

- Default threshold to distinguish between favourable and non-favourable variables
- Higher scores associated with variables lead to better output quality
- More variables tend to enhance predictive performance

However, the above assumptions rely on the function used within the filter method, the input data characteristics, and the default threshold value used in the

Ranker search procedure. Considering that results of the filter methods may vary significantly depending on the above facts, it would be gainful to develop a method that considers correlations in the format of human interpretable rules from the data, and then creates a scoring rank for each generated rule. Based on such ranks each variable can be assigned a new score based on computational metric such as the number of times the variable-value appears in the rules, and the information gained when adding the variable-value into the rule's body.

This approach can be seen as a way forward of variable selection that takes into account the correlations among the variables themselves and correlations among the independent variables and the dependent variable in the training dataset. Such adaptive ranking philosophies conceptually align with intelligent optimization principles employed in modern adaptive control and synchronization systems, where dynamic weighting and iterative refinement improve system robustness and interpretability [4] [18].

Additionally, developing a new method that can guide the users like physicians in medical diagnosis applications on the number of variables to consider, especially during medical evaluation, can be crucial to improve diagnostic decisions. The Ranked search method should not necessarily require human involvement during pre-processing, and it will suggest to the user a threshold pinpointing to a set of dissimilar variables regardless of the input dataset size. This approach will improve the efficiency and limits user's subjectivity during variables selection methods.

To address the limitations mentioned earlier, this study is guided by the following research questions:

- RQ1: Can a rule-based variable selection approach, driven by entropy reduction, reduce variable set size while maintaining classification performance compared to traditional filter methods such as Information Gain (IG) and CHI-SQR on classification datasets?
- RQ2: Can the integration of rule mining and variable weighting mechanisms improve the identification and elimination of redundant and overlapping variables?
- RQ3: Can an automated cut-off mechanism based on rule scoring and Huffman-based encoding determine the optimal number of variables without user intervention?
- RQ4: How does the proposed RMAM method perform across diverse datasets in terms of dimensionality reduction, interpretability, and predictive accuracy?

The primary objective of this research is to develop a fully automated variable selection pipeline that not only ranks variables but also determines an optimal subset size, reducing reliance on domain expertise and improving applicability in real-world domains such as medical diagnosis. This method is called Rule Mining Analysis Search [RMAM] that minimises data dimensionality. The basis for which variables are offered follows the development of rules using a reduction of uncertainty among independent variables and the dependent variable in the dataset, and using Huffman method [7] to compute how many variables should retain prior of supervised learning. These rules are developed from the training dataset using computed independent variables-dependent variable associations, thus weak variables are identified and thereby early discarded. When a rule is constructed based on the reduction of entropy a score is assigned for that rule by amalgamating the computed frequencies of the variables' values in the rule. The rules' scores are then utilised to assign a true weight per variable by totaling the rules' scores in which the variable has appeared. More details are given in Section 3.

The proposed variable selection method considers only variables with adequate correlation with the dependent variable that is measured using information gained can be added into the rule's body. In addition, a total-score will be assigned to each rule by iterating over the rule's body items (variable computed score). The process of rule construction, and rules' weights assignment continues until no further rules can be grown from the training dataset. This is the point when the RMAM method terminates.

Empirical initial results, using several applications' data from the University of California Irvine Repository (UCI) [5] show that RMAM is a competitive variable selection method in terms of data dimensionality reduction. When it is used for data pre-processing, the chosen variables are processed by classification algorithms, then competitive classifiers are generated and contrasted with those chosen by other variable selection methods such as Information Gain (IG) [15], and CHI-SQR. The adaptive and iterative nature of the proposed framework further reflects broader trends in intelligent optimization and adaptive computational systems where iterative refinement mechanisms are employed to improve performance and robustness across diverse application domains [19].

This paper consists of different sections; section 2 discusses relevant works, and section 3 details RMAM phases such as rule discovery, score computations, and variables' assignments. Section 4 discusses results and analysis, and lastly section 5 lists the key characteristics of the RMAM method besides conclusion.

2 Literature review

Ranking variables is an essential process in filter-based variable selection methods to identify their contribution to classification's model performance [26]. However, variable redundancy and overlapping are persistent issues that affect the accuracy of derived classification models. While techniques such as Mutual Information and Correlation-based rankings [12] address redundancy, they often do not handle variable sharing in data observations, leading to some irrelevant variables being retained by the filter methods [26]. Variable redundancy arises when multiple variables convey similar information, which unnecessarily increases model complexity without adding values to performance. For instance, Pearson correlation coefficients are widely used to detect variable overlapping

using heat maps; however, these techniques often lack the ability to capture nonlinear dependencies.

To deal with variable ranking in data related to oil industry [25] introduced a ranking method to forecast crude oil returns. Their work emphasizes ranking variables based on their individual contributions but lacks explicit mechanisms for identifying overlapping variables. While their approach demonstrated improvements in accuracy, the appearance of overlapping variables in the outcome may reduce the interpretability and computational efficiency of the classification model. Overlapping variables often inflate variable space, leading to additional computational time during model evaluation, a significant issue for large-scale datasets.

An improvement to integrate multiple rankings was introduced by [3]. The authors proposed the Weighted Rank Difference Ensemble (WRDE) method, which combines multiple rankings by assigning weights to variables based on the difference in their rankings across different methods. WRDE enhances the robustness of rankings by mitigating the biases of individual methods; however, it does not explicitly address variable interdependencies. As a result, redundant variables that rank high across multiple methods can persist, despite their limited individual value. This highlights a critical need for methods that not only rank variables effectively but also incorporate mechanisms to eliminate overlapping variables systematically.

Another ranking approach that takes into account multiple variables as a group instead of individual variables was proposed by [27] by introducing a permutation-based method that analyzes variables groups contribution to classification accuracy, and reduces the risk of redundancy caused by variable overlapping. However, evaluating variable groups increases computational complexity, particularly for high-dimensional datasets, making it less practical for real-time or large-scale applications. Moreover, the method does not explicitly handle redundancy at an individual variable level, leaving room for further enhancement. These limitations underscore the need for novel filter-based methods that incorporate both variable ranking and redundancy elimination.

The lack of effective mechanisms to define a cutoff for distinguishing relevant from irrelevant variables in filter methods also represents a significant research gap. Existing methods, such as the one developed by [23], attempt to combine the strengths of filter- and wrapper-based methods to improve performance. [23] employs a filter-based approach for initial variable reduction, followed by wrapper-based optimization to refine the selection further. While effective, this method lacks a systematic way to determine a cutoff threshold within the initial filtering stage. The reliance on arbitrary or dataset-specific thresholds hinders the generalizability of the method and limits its efficiency. Other researchers such as [9] suggested that heuristic search techniques combined with learning-based approaches may offer a promising direction for identifying relevant variables. However, these methods still lack a ranker mechanism that clearly determines cutoffs, particularly in filter-based approaches where computational simplicity is dominant.

Identifying a universal or semi-automated cutoff mechanism would significantly enhance the utility of filter-based methods, particularly for large-scale datasets where exhaustive evaluations are computationally expensive [1] [26] [12]. A promising approach that used association rule mining with ranking method was proposed by [1]. [1] introduced a filter method that leverages class association rules to identify variables with strong associations to class labels.

By mining frequent patterns between variable values and class labels, this method effectively reduced the number of variables while improving classification accuracy. Despite its strengths, the CAR-based method focused primarily on discrete datasets and frequent patterns, limiting its applicability to continuous datasets where patterns are less explicit. Moreover, the method did not embed a ranker mechanism to optimize variable cutoff thresholds. A hybrid approach that combines CARs with statistical or learning-based methods could enhance both the ranking process and cut-off determination.

Rule-mining methods play a significant role in filter-based variable selection by assigning weights to variables based on their relationships to target outcomes. Rule mining allows for the discovery of meaningful patterns that are often overlooked by traditional statistical techniques. The work of [1] demonstrated promising direction of how class association rules could be used in variable selection to reduce variable overlapping and offer a smaller number of retained variables. However, the method focuses on frequent itemsets without assigning explicit weights to variables rather a variable can be assigned a weight from multiple frequent itemsets. This limitation reduces the ability to differentiate between strongly associated and moderately relevant variables, particularly in high-dimensional datasets.

A recent hybrid filter method has been developed by [16] to further emphasize the importance of rule-based methods in their study on neuropathological variables in dementia classification. By ranking and filtering variables using rule-mining techniques, the study demonstrated improvements in the interpretability and accuracy of machine learning models. Rule-based approaches offer a unique advantage in variable ranking by leveraging both statistical associations and domain-specific patterns.

However, their integration into filter-based methods remains limited in research work in the context of variable selection, particularly in terms of assigning variable weights. A combined approach that integrates rule-based weighting with redundancy elimination mechanisms could significantly enhance filter-based variable selection. This suggestive approach, when linked with a ranking method that suggest a cutoff of the retained number of variables, can be highly beneficial especially in medical applications such as dementia classification when time and resources are limited.

The combination of ranker mechanisms, systematic cut-off thresholds, and rule-mining methods has the potential to address the major limitations of existing filter-based variable selection methods. For example, a ranker

mechanism embedded into a filter method could evaluate variable importance while simultaneously detecting and eliminating overlaps. This approach would allow for a more precise ranking of variables, enhancing both interpretability and computational efficiency. By incorporating rule-mining techniques, such as rule induction, covering algorithms, or association rules, variables could be assigned weights based on their predictive associations, further improving the robustness of the selection process.

Lastly, meta-classifiers such as Ensemble learner can be used in improving filter methods especially reducing the decision bias of choosing variables. For example, [3] demonstrated the value of ensemble methods in mitigating biases associated with individual techniques. Their Weighted Rank Difference Ensemble method highlights the importance of combining multiple variable selection methods to enhance robustness. However, the method does not consider the issue of cut-off determination to easy the process of user selection in filter-based methods.

Table 1 depicts the summary of the relevant papers based on the literature review conducted.

Table 1: Summary of the relevant literature

Study	Dataset(s) Used	Method	Variables Selected	Quantified Results	Key Limitations
[1] Al-Dhaheri (2021)	UCI datasets (e.g., Mushroom, Breast Cancer)	Class Association Rules (CAR)	Reduced from full set (exact numbers vary per dataset)	Accuracy improved by up to ~3–5% over baseline classifiers	No automated cut-off; limited to discrete data; no explicit variable weighting
[3] Begum et al. (2024)	8 medical datasets (e.g., Heart, Diabetes, Breast Cancer)	WRDE (Weighted Rank Difference Ensemble)	Reduced variables (not fixed; varies per dataset)	Achieved average accuracy improvements of ~1.5–3% over single ranking methods	Does not remove redundant variables; no cut-off mechanism
[9] Karimi et al. (2023)	Breast Cancer (Wisconsin)	Learn-Heuristic Variable Selection	Reduced variables from 30 to ~10–15	Accuracy improved to ~97–99% depending on classifier	No rank-based cut-off; requires tuning
[23] Yu et al. (2020)	KDD Cup 99 (Intrusion Detection)	TSDR (Filter + Wrapper)	Reduced variables from 41 to ~10–15	Detection accuracy increased by ~2–4%	No automatic cut-off; multi-stage complexity
[25] Zhao et al. (2024)	Crude Oil Price Dataset	Importance-based ranking	Not specified	RMSE reduced by ~5–10% compared to baseline models	No redundancy handling; no variable subset selection
[27] Zubair et al. (2024)	UCI datasets (e.g., Iris, Wine, Glass)	Permutation-based Group FS	Group-based selection (size varies)	Accuracy improved by ~2–6% over baseline ranking	High computational cost; no individual redundancy control
[16] Rajab et al. (2024)	ADNI (Dementia dataset)	Rule-based variable filtering	Reduced neuropathology variables (exact count varies)	Classification accuracy improved by ~2–5%	No explicit weighting; no automated cut-off
[12] Li et al. (2020)	Multiple benchmark datasets	Relevance–Redundancy–Complementarity	Reduced variable subsets	Accuracy gains reported (~2–4%)	Limited nonlinear dependency handling
[21] Thabtah et al. (2020)	UCI datasets (e.g., Ionosphere, Sonar)	Least Loss (Filter)	Reduced variable sets (dataset dependent)	Competitive accuracy with fewer variables (~similar to IG/CHI-SQR)	No automatic subset size recommendation

3 RMAM method

The proposed filter method (RMAM) (Figure 1) consists of a rule generation function. The resulting rule sets are subsequently used to assign weights within the preceding stage in which a threshold, named the reduction in the Entropy, is employed to grow the rules from the training dataset. It is used to check whether an independent variable value has any significance before adding that value into the rule by checking its correlation with the class label. Therefore, any variable value plus the class label (V_v, Class_c) with a reduction in the entropy will be added into the current rule that is being grown.

To retain rules that are strong enough before the variables' scores are calculated by RMAM method, a rule is continuously grown by adding variable values into its body until the rule's information gain cannot be improved any further. In other words, when adding the variable

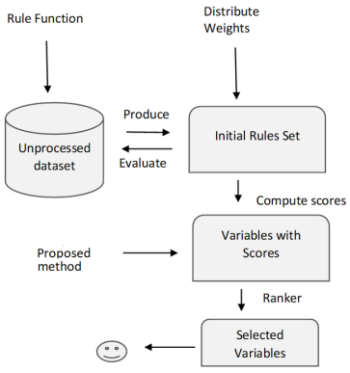

Figure. 1. RMAM filter method

value (V_v) to a rule:Ri, and Ri's information gain is not improved then a rule grown process is stopped. When this scenario occurs, Ri will be generated since its entropy cannot be minimised any further. The stoppage in grown a rule can be seen as a rule trimming process since it determines when a rule can be generated, and it avoids appending biased and/or redundant variable values into the current rule's body.

The RMAM filter method produces all rules using the "generate rule" function; whenever a rule is discovered through this function, any connected training data instances are erased, and a new weight is assigned to the rule. The training dataset shrinks, and the next rule is discovered from the updated training dataset in the same way. The process is repeated until all rules are discovered. Hence, the "generate rule" function returns a set of rules each of which is associated with a computed weight. One advantage of the RMAM "generate rule" function is that it uses straightforward computations for assigning weights using the variables values' frequencies with the class labels.

Once the rules set is returned iteration over each rule is performed to assign a score to each variable. The variables' scores are assigned using the rules' weights per Algorithm 1 and a cut-off is then calculated based on a proposed search Ranker called Modified Huffman Ranker [MHR] (Section 3.2 gives further details). Lastly, RMAM deals with both nominal (pre-set possible values) and continuous variables (numbers and decimals) and treats any missing value as any other value in the training dataset by adopting the replace_missing_value filter; any continuous variable is implicitly discretized.

3.1 Definitions

The variable selection problem is defined as follows: Given an input dataset TR with cardinality |TR| which consists of n variables Vr1, Vr2, … ,Vrn, and the class variable (class). The key purpose is to select a set of variables that can have comparable performance with TR when processed by learning algorithms. Below are the main definitions related to the proposed filter method.

Term 1: A variable value in TR with a class is represented as: [(Vri, vri), class].

Term 2: A record/tuple in TR is comprised of [(Vrj1, vrj1), …, (Vrjv, vrjv), class] and is normally called a training data instance.

Term 3: A rule r: Vr_1∧ Vr_2 ∧….∧ Vr_j→Class, consists of a body and a target class.

Term 4: The variables' values frequency (Vr_freq) for a rule in TR is the number of training instances in TR that are identical to the rule's body.

Term 5: The rule's frequency in TR (r_freq) is the number of training instances in TR that are identical to r.

Term 6: The rule's covered data in TR (r_data) is the training data instances in TR that are identical to r.

Term 7: The weight of a rule r (r_weight) is computed as $r\_freq \times (|TR| - |r\_data|)$

Term 8: A score of a variable (Vr_score) is computed as $\sum_1^n (\mathrm{r_{weight}}) + \mathrm{Attributes\_ranks}$

In the RMAM "generate rule" function, we store the training instances in a data structure (array) in which each instance is associated with a unique identifier (line number, column number).

```
Input:
- Training dataset TD
Output:
- Ranked set of variables with scores and selected subset based on cut-off
Pseudocode:
// Step 1: Initialize
Attributes_rank ← array of size |A| initialized to 0
num_uncovered_instances ← number of instances in TD
// Step 2: Generate rules
RSet ← Generate_Rule(TD)
// Step 3: Compute attribute scores
for each rule Ri in RSet do
    ri_weight ← frequency(Ri) × num_uncovered_instances
    num_uncovered_instances ← num_uncovered_instances − frequency(Ri)
   for each attribute Aj in Ri do
       Attributes_rank[j] ← Attributes_rank[j] + ri_weight
   end for
end for
// Step 4: Apply MHR for cut-off selection
C_score ← MHR(Attributes_rank)
// Step 5: Select top-ranked attributes
sort attributes in descending order based on Attributes_rank
Selected_Set ← top C_score attributes
return Selected_Set
```

Algorithm 1: RMAM Pseudocode

This data transformation eases processing data during iterating over the training data and counting the variable values' frequencies. In addition, this data transformation enabled us to limit data computation especially the process of constructing the rules. For example, we can obtain the frequency of the variable values and the rules by locating their line and column numbers stored in the array rather than performing extensive counting per instance.

3.2 Rules and weights

Since the rules are utilised to assign the scores of the variables in the input dataset by the proposed filter method, this section expands on the way rules are built within the "generate rule" function of RMAM. This function takes the training dataset and returns a complete set of rules each of which is assigned a computed weight. The function iterates over the complete set of training instances to consider the variable values with the highest information gain with respect to the class label. The reduction of entropy, or "information gain," is a concept in determining how well a variable (or attribute) splits the data into groups that are more homogeneous (or pure) regarding the target variable. Entropy (Eq. 1), in this context, measures the impurity or disorder in a dataset.

By focusing on entropy reduction, the rule learning algorithm of the RMAM method aims to construct a decision rule that is both simple and highly predictive of the target variable. The goal is to use that rule in calculating the total weight that will be later assigned to the available variable in the training dataset. These variables are then passed to the supervised learning algorithm to develop predictive models that can generalize

well from the training data to unseen test data instances, effectively capturing the underlying patterns in the data.

$$IG(T, V) = \text{Entropy}(T) - \sum ((|T_a| / |T|) * \text{Entropy}(T_a)) \quad (1)$$

Where,

$$\text{Entropy}(T) = \sum -[P_l \, log_2 \, P_l] \quad (2)$$

Where, P_l = the probability that a dataset T is associated with class l.

T_a = A dataset of T

Where variable $\text{Entropy}(T) = \sum -[P_l \, log_2 \, P_l]$le V is associated with a value v

$|T_a|$ = the number of instances that match T_a,

$|T|$ = the number of instances in the training dataset D.

Algorithm 2 shows the detailed steps used to generate the rules by the RMAM filter method in which the process starts by converting the data into Line_Space data format [6] (Hammoud, 2009) to simplify the process of constructing rules. In converting the data to Line_Space the filter method iterates over the input dataset to convert the data instances into a (key:values) data form as discussed earlier. Prior building a rule, information gains of each possible variable value in the training dataset are calculated.

The process begins by transforming the input dataset into the Line_Space data format to facilitate efficient rule construction and frequency computation. In this representation, each data instance is converted into a structured (key:value) format, where each variable-value pair is associated with its corresponding record identifier (e.g., row index). For example, a record such as (Age = Young, Income = High, Class = Yes) is transformed into indexed pairs like (Age:Young → {1}), (Income:High → {1}), (Class:Yes → {1}). Across the dataset, identical variable-value pairs are aggregated, resulting in compact mappings such as (Age:Young → {1, 3, 5}), which indicate all records where the condition holds.

This structure enables rapid computation of variable frequencies and rule data coverage using simple set operations (e.g., intersection of record indices), rather than repeatedly scanning the dataset. As a result, rule items can be efficiently evaluated, and their IG with respect to the class label can be computed. Prior to rule construction, IG is calculated for all possible variable-value pairs, allowing the algorithm to iteratively select the most informative conditions while minimizing computational overhead and redundancy during rule generation. The reason for calculating the information gains is to know the amount of information that can be gained when considering each variable value with the class label. A high information gain in a rule reflects low uncertainty when estimating the class label.

Using the computed information gains, an empty rule: R is then initiated, and the possibility of adding a variable into R is considered. The variable value with the largest information gain is added (Line 15), and all its corresponding training data is isolated (Line 16) to possibly account for adding other variables if possible later on (expanding the current rule). The data instances associated with R is then isolated and sent along with R into "Check_Improvement_Rank" function to check whether adding another variable value to R may improve R's gain (Line 17). If the "Check_Improvement_Rank" function yields "true" then R gets amended (expanded) by appending the best variable value computed from R's data instances, and the process is repeated until R's information gain cannot be improved anymore. When R's information gain is not improving any more then R is added to the rules set, and the original data is updated by discarding R's data instances. The process of discovering more rules continues from the updated training dataset until all rules are produced from the training dataset.

Input:
- Training dataset TD

Output:
- Set of generated rules R_set

Pseudocode:

```
1: // Step 1: Preprocessing
2: Convert TD into Line_Space format
3: Compute frequency counts for all variable values (Vr, Vv)
4: // Step 2: Initialization
5: R_set ← Ø
6: T_storage ← TD
7: // Step 3: Rule generation loop
8: while T_storage is not empty do
9:    r ← Ø  // initialize empty rule
10:   // Step 3.1: Grow rule
11:   repeat
12:      for each rule item RI = ((Vr, Vv), class) in T_storage do
13:         compute Information Gain IG(RI)
14:      end for
15:      r_i ← rule item with maximum IG
16:      T_temp ← T_storage − instances covered by r_i
17:      C ← Check_Improvement_Rank(r, T_temp)
18:      if C = true then
19:         add r_i to rule r
20:         T_storage ← T_temp
21:      end if
22:   until C = false
23:   // Step 3.2: Finalize rule
24:   R_set ← R_set U {r}
25:   remove instances covered by r from T_storage
26: end while
27: return R_set
```

Algorithm. 2: RMAM Generate_Rule Function

The "generate rule" function of the RMAM method constructs rules as described above and the training dataset at each iteration shrinks after removing each rule's data instances. The function terminates when there are no more can be discovered. When this condition holds, the function returns the rules sets along with the associated rules' weights. The pseudocode of the "generate rule" and "Check_Improvement_Rank Function" functions of RMAM are depicted in Algorithms 2 and 3 respectively.

Input: *r*'s (Data): T_ storage', Rule: *r*

Output: Boolean: Whether the current rule is improved after adding a RI

1. If $IG(r_i) > IG(r_{i-1})$
2. $return\ (true)$
3. $else\ return\ (false)$ // empty rule

Algorithm. 3: Check_Improvement_Rank Function

The "generate rule" function of RMAM relies on the reduction of entropy as a quality assurance metric for preventing weak variables' from taking any role in computing the weights of the rules. When variables become part of a rule's body, the "generate rule" function ensures that such variables are correlated with the target class and therefore RMAM checks any possible information gain improvement of the rule to ensure that the rule is significant. Thus, the tests performed by proposed filter method are vital to prune the weak variable value(s) from being added into a rule.

3.3 Variables' scores

Once the rules are sent back by the "rule generate" function of the RMAM filter method then they are utilised to assign a score for variables. RMAM iterates over the set of rules generated and for each rule an internal iteration over the rule's variables is performed. A score is assigned to each variable appearing in such a rule equal to the rule's weight. When the variable appears in multiple rules then the summation of these rules' weights denotes the score of that variable.

The RMAM filter method considers that earlier derived rules should be associated with larger weights; when these rules are produced larger numbers of uncovered training instances remain. Considering that one of the main components of the rule's weight is the number of uncovered training instances then rules derived first have larger weights than rules derived later; therefore, variables appearing in the first few rules often have larger scores. These variables have a higher chance of being chosen by RMAM as they are associated with larger scores. It should be noted that RMAM favours higher ranked rules in ascending order as these rules are more predictive and often cover a sizable portion of the dataset. Thus, assigning these rules weights to the variables that appeared in them, gives such variables the best opportunity of being presented to the end user as influential.

As discussed in the literature review, one of the major issues associated with filter methods is the inability of these methods in recommending a final subset of variables to the end users, especially novice users. Having adopted a basic Ranker search method all sets of variables are returned sorted in descending order based on their assigned scores. To deal with this issue, RMAM has developed a new Ranker method called Modified Huffman Ranker [MHR] that automatically identifies an intelligent cut-off to separate influential variables from those that are redundant. MHR was inspired by the Huffman greedy search algorithm [7], which was originally developed to efficiently decode data and to utilize the least storage possible, especially in data compression applications. The classic Huffman algorithm represents strings as binary codes without compromising data loss in which coding is achieved using the probabilities of the input strings or symbols. The Huffman algorithm offers the shortest weighted average length per symbol to encode that symbol. To be exact, when a symbol appears more frequently its corresponding binary representation will be short and therefore the data accommodation requirement will be minimised using this kind of encoding.

The proposed MHR method is shown in Algorithm 4 in which the variables' values, instead of symbols and their frequencies, are utilised as input for the Ranker function, which in turn returns the cut-off value. The input parameters for the Ranker function are given after the training dataset has been transformed and scanned by the filter method (in our case RMAM). The cut-off value returned by the MHR function denotes the recommended number of variables to the end user. The function initially normalizes the weights of the variables' values as shown in Algorithm 4 (lines 1-4). The weights of the variables' values are then sorted in a descending manner to compute the sum of all possible weights. This sum is then used to compute the cut-off value by inserting it as a power in a base 2 formula as indicated in line 21 of Algorithm 4. Detailed calculations of the Ranker are shown later in this section.

Mathematically, we define a set of variables V = {v1, v2, …, vn} with associated weights wi ≥ 0 obtained from rule based scoring. First, normalize the weights:

$$p_i = \frac{wi}{\sum_{j=1}^{n} wj}, where \sum_{i=1}^{n} pi = 1$$

Let P = {p1, p2, …, pn} be the normalized weights sorted in ascending order. The MHR applies an iterative merging process:

At each step, select the two smallest values pa and pb and merge them:

$$p_{\{ab\}} = p_a + p_b$$

This merged value replaces pa and pb. Let S be the cumulative sum of all merged values:

$$S = \sum_{k=1}^{n=1} p_{k^{(merge)}}$$

where p_(k^((merge))) represents the merged weights at each iteration.

Finally, the cut-off threshold is computed as:

$$C_{\{score\}} = 2^{\{S\}}$$

By using the proposed MHR function, variables that are common in the training dataset have higher weights and therefore they are significant and often ranked higher by RMAM than in any other filter methods. Therefore, such variables will typically be chosen by RMAM as they appear in the top ranked results. Variables that are not common in the training dataset will have very little weight and therefore they will be associated with a low rank once the results are generated by the filter method. Thus, such variables will have little chance of being detected by the filter method since their ranks will be below the cut-off value computed by the MHR method. This simple function of calculating the cut-off is useful for domain experts as well as novice users as it offers a line to discriminate variables. Users are expected to spend more time in data analysis rather in manually assessing the outcomes of filter methods. The MHR is not dependent on a specific filter method and can be utilized by any existing filter methods as long as they produce scores per variables in the dataset.

The cut-off procedure is designed to decide when adding more variables is no longer useful. In simple terms, the method first ranks variables based on how often they appear in strong predictive rules and how much they help reduce uncertainty about the target outcome. Variables that appear frequently in important rules receive higher weights, while less useful or rarely contributing variables receive lower weights. The cut-off is then determined by frequent and infrequent variables using a process inspired by data compression. This process gradually merges lower-weight variables and keeps track of their overall contribution. When the combined contribution stabilizes, the method identifies a stopping point. This approach works well since it balances two goals: keeping informative variables while avoiding unnecessary ones thus it provides a automatic way to select a compact set of variables without relying on arbitrary thresholds or user experience.

```
Input:
- Set of variables V = {v1, v2, ..., vn}
- Corresponding frequencies F(vi) for each variable
Output:
- Cut-off threshold C_score
Pseudocode:
// Step 1: Normalize frequencies
total ← Σ F(vi) for all vi ∈ V
for each vi ∈ V do
    F(vi) ← F(vi) / total
end for
// Step 2: Sort variables by ascending frequency
sort V in ascending order based on F(vi)
// Step 3: Initialize
Sum ← 0
// Step 4: Iteratively combine lowest-weight variables
while |V| > 1 do
    va ← first element in V
    vb ← second element in V
    merged_weight ← F(va) + F(vb)
    remove va and vb from V
    insert new variable vab into V with weight merged_weight
    Sum ← Sum + merged_weight
    re-sort V in ascending order based on weights
end while
// Step 5: Compute cut-off score
C_score ← 2^(Sum)
return C_score
```

Algorithm. 4: The Cut-off score function (MHR function) of RMAM filter method

To demonstrate how MHR search method works consider Figure 2 with eight variables (A-H) and their corresponding weights. For any two variables such as { A, B}, their weight W(S) can be represented as W(A) + W(B) = 0.30 + 0.20 = 0.50 hence we can represent the new derived variable as {A, B} (0.50). Based on the variables of Figure 2, the following steps are performed by MHR:

- Create a variable Sum = 0;
- Loop over the input variable List: [{A}(.30), {B}(.20), {C}(.15), {D}(.10), {E}(.08), {F}(.07), {G}(.05), {H}((.05)], length = 8, Sum = 0.0
- Union the lowest weight sets as List [{A}(.30), {B}(.20), {C}(.15), {D}(.10), {E}(.08), {F}(.07), {G, H}(.10)], length = 7 so now Sum = 0.10
- Second union: List [{A}(.30), {B}(.20), {C}(.15), {D}(.10), {E, F}(.15), {G, H}(.10)], length = 6, Sum = 0.10 + 0.15 = 0.25
- Third union: List [{A}(.30), {B}(.20), {C}(.15), {D, G, H}(.20), {E, F}(.15)], length = 5, Sum= 0.25 + 0.20 = 0.45
- Fourth union: List [{A}(.30), {B}(.20), {C, E, F}(.30), {D, G, H}(.20)], length = 4,
- Sum= 0.45 + 0.30 = 0.75
- Fifth union: List [{A}(.30), {B, D, G, H}(.40), {C, E, F}(.30)], length = 3, Sum= 0.75 + 0.40 = 1.15
- Sixth union: List [{A, C, E, F}(.60), {B, D, G, H}(.40)], length = 2, Sum= 1.15 + 0.60 = 1.75
- Seventh union: List [{A, C, E, F, B, D, G, H}(1.00)], length = 1, Sum= 1.75 + 1.00 = 2.75
- End while loop
- Lastly, compute cut off point = 2 ^ 2.75 = 6.727

It should be noted that we initially normalized the variables weights, this process can also be performed later on after all calculations without normalized variables and then for the final Sum value we can divide it by the sum of all weights

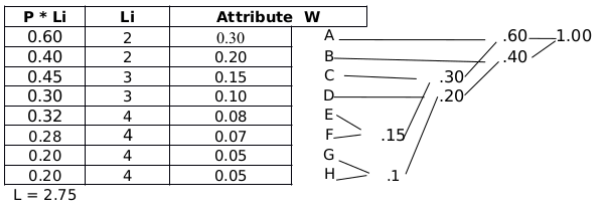

Figure 2: Sample variables and their scores

The key variables of this search method are given below:

- Easy to understand
- Efficient to calculate
- Straightforward results to interpret
- A clear cut-off value that recommends to the end-user the number of suggested variables
- Minimizes the search space of variables when used with a filter method
- Reduces the manually tedious process of checking all variables in the results set.

3.4 RMAM characteristics

In this section, the primary differences between RMAM and other existing filter methods are provided:

- RMAM adopts a special learning based on inducing rules from different parts of the training dataset. These rules are primarily utilized to assign scores to the influential variables; existing filter methods do not involve learning rules when they calculate variables' scores.
- The majority of the existing filter methods consider generating scores using mathematical formulas without considering that variables may

have common training instances. RMAM ensures that the scores assigned to the variables are calculated based on predictive rules learnt from the input dataset. Hence, the chance of redundant variables in the selected variables are reduced as rules used for the score's assignment are not redundant

- The mathematical formula for assigning the scores to the variables is novel since the weights of the rules that have been used to compute the scores are derived based on the reduction of entropy of both of the rules as well as the remaining uncovered instances. This is unlike existing filter methods which are primarily based on a contingency table of variables and class values frequencies.
- RMAM utilizes a novel Ranker search method to produce a cut-off value computed regardless of the input dataset. This search method balances between the performance of the selected variables when processed (error rate) and the information gained when selecting these variables. This is unlike existing filter methods which utilize a conventional Ranker that sorts the variables based on the scores without pinpointing to the set of variables that can be selected by the end user; this is time-consuming and a burden on the users especially novices.
- RMAM often produces fewer variables when compared with other filter methods and maintains the performance of the predictive models derived when processing its distinctive variables sets.

4 Data and results

4.1 Experimental protocol

The experimental protocol followed a structured process as outlined below:

1) Datasets Selection: A total of 27 benchmark datasets from the UCI repository were selected, covering small, medium, and large datasets across multiple domains, including medical diagnosis. The selection considered dimensionality, dataset size, missing values, continuous attributes, and application diversity.
2) Data Preprocessing: Missing values were handled using the MissingMerg setting. Continuous variables were discretized using entropy-based discretization, and data instances were transformed into Line_Space format to support efficient rule construction and frequency counting.
3) RMAM Implementation: The proposed RMAM method was implemented in Java and integrated into WEKA within the attribute selection module. RMAM generated entropy-guided rules by selecting variable values that improved information gain with respect to the class label, while pruning weak or redundant variables when no further gain was achieved.
4) Scoring and Cut-off: Rule weights were used to compute variable scores, and the Modified Huffman Ranker (MHR) automatically determined the recommended variable subset size.
5) Comparison Methods: RMAM was compared with Information Gain (InfoGainAttributeEval) and Chi-Squared (ChiSquaredAttributeEval) filter methods in WEKA.
6) Parameter Settings: Naïve Bayes: Probabilistic classifier assuming conditional independence, using normal (Gaussian) distribution for numeric attributes and frequency-based estimation for nominal attributes; kernel estimation and supervised discretization were not enabled.
7) Information Gain: Entropy-based evaluation using InfoGainAttributeEval, computing variable relevance based on reduction in entropy with respect to the class variable; no frequency discretization or binning beyond preprocessing.
8) Chi-Squared: ChiSquaredAttributeEval using contingency tables between variable values and class labels, computing χ^2 statistics based on observed and expected frequencies; operates on discretized attributes.
9) Search Methods: Ranker search for Information Gain and Chi-Squared; Modified Huffman Ranker (MHR) for RMAM.
10) Evaluation Procedure: All experiments were conducted using stratified 10-fold cross-validation, where datasets were partitioned into ten equal folds with stratification, iteratively training on nine folds and testing on the remaining fold to ensure robustness and generalization.
11) Performance Metrics: Performance was evaluated using search-space reduction, error rate, precision, recall, F1-score, and confusion matrix analysis.

Several different datasets from the UCI repository have been adopted to evaluate the performance of RMAM and MHR methods; Table 2 shows the 27 datasets that have been utilised in the experimental analysis. The selection of these datasets is based on factors such as their size, dimensionality, the availability of missing values, the availability of continuous variables and data business domains. For fair comparison we have used big, medium and small datasets that belong to various domains including medical diagnosis and implemented a discretization method of the continuous variables based on the Entropy within RMAM. Further, any missing values have been treated using the MissingMerg method as a parameter in the RMAM interface. This can be set to accomplish the distributes of the missing values across other values using their actual frequencies in the training dataset.

4.2 Settings and datasets

To reveal the performance of RMAM in terms of dimensionality reduction and classifiers' predictive power

Table 2: The datasets considered

Dataset	# variables	# of cases	Missing Values	Numeric Variables
anneal	38	898	Yes	Yes
arrhythmia	279	452	Yes	Yes
audiology	69	226	Yes	No
autos	25	205	Yes	Yes
breast-cancer	9	286	Yes	No
car	6	1728	No	Yes
cleve	11	690	Yes	Yes
cmc	9	1773	No	Yes
credit-rating	15	690	Yes	Yes
dermatology	34	366	Yes	Yes
flags	29	194	No	No
Glass	9	214	No	Yes
heart-statlog	13	270	No	Yes
hepatitis	19	155	Yes	Yes
hypothyroid	29	3772	Yes	Yes
ionosphere	34	351	No	Yes
labor	16	57	Yes	Yes
lung-cancer	56	32	Yes	No
lymphography	18	148	No	Yes
mushroom	22	8124	Yes	No
nursery	8	12960	No	No
pima_diabetes	8	768	No	Yes
primary-tumor	17	339	Yes	No
segment	19	2310	No	Yes
tic-tac-toe	9	958	No	No
vehicle	18	846	No	Yes
Wis-breast-cancer	9	699	Yes	Yes

on the selected subsets we selected a number of competitive filter methods for comparison purposes. Specifically, we used IG, and CHI-SQR as these methods have been tested on several benchmarks in the research literature, e.g. [11] [10] The choice of these filter methods was based on the following:

a) They produce a score per variable using different mathematical formulas
b) They use search methods to rank the scores
c) They have been previously utilized by scholars on different datasets

To evaluate the effectiveness of the variables sets derived by the filter methods we used the Naïve Bayes algorithm [8]. This algorithm adopts a probabilistic theorem to build fast and effective classifying models. Naïve Bayes creates a contingency table based on the observed and expected probabilities of variables' values and the class variable. These are then used to produce a forecasted class of a test case without the need to build a classifier from the training dataset. The reason for adopting the Naïve Bayes algorithm as it has shown efficient and effective performance in predicting class labels as accurately as possible when compared with other conventional classification algorithms including statistical functions, decision trees and even Artificial Neural Network. In all experiments, WEKA (Waikato Environment for Knowledge Analysis) machine learning platform was used to implement the filter methods including ours and the Naïve Bayes algorithm. The experiments were conducted on a computing machine with a Core i7 computer with a 3.1 GHz processor and 16.0 GB RAM. Further, different evaluation metrics were adopted to reveal the advantages and disadvantages of RMAM when contrasted with other filter methods (IG, CHI-SQR).

4.3 Search space results

Table 3 depicts the number of recommended variables chosen by the RMAM, IG, and CHI-SQR filter methods respectively when integrating the new MHR Ranker

Table 3: Number of selected variables of RMAM and the other filter methods

MHR Ranker suggested cut off point				
Dataset	**# variables**	**CHI**	**IG**	**RMAM**
anneal	38	18	19	7
arrhythmia	279	88	94	4
audiology	69	38	32	8
autos	25	16	18	9
breast-cancer	9	7	7	9
car	6	5	5	3
cleve	11	10	10	10
cmc	9	8	7	7
credit-rating	15	10	9	13
dermatology	34	32	31	6
flags	29	9	14	4
Glass	9	8	8	9
heart-statlog	13	10	10	12
hepatitis	19	13	13	12
hypothyroid	29	5	5	4
ionosphere	34	33	33	12
labor	16	12	12	6
lung-cancer	56	42	42	4
lymphography	18	14	16	9
mushroom	22	15	14	4
nursery	8	3	3	4
pima_diabetes	8	7	7	7
primary-tumor	17	15	15	7
segment	19	15	15	8
tic-tac-toe	9	6	6	7
vehicle	18	16	16	16
breast-cancer	9	9	9	4

method. The figures shown in Table 3 have been rounded up to convert the values into an integer rather than a decimal. For example, for the Anneal dataset, the cut-off computed value by RMAM when using the Ranker method is 6.373 thus this value was rounded up to 7 variables. The figures within the table clearly show superiority of RMAM in reducing the search space of variables when contrasted with the other considered filter methods.

RMAM was able to derive fewer variables on 18 out of 28 datasets when compared with each considered filter method. For instance, on the Anneal, Arrhythmia and Audiology datasets, RMAM recommended 7, 4, and 8 sets of variables respectively whereas for the same datasets IG, and CHI-SQR have recommended (19, 93, 32), and (18, 88, 37) respectively. These figures, if limited, demonstrate that RMAM minimises the search space substantially when compared with known filter methods. To be exact, the reduction of the search space on these three datasets by RMAM when contrasted with IG, and CHI-SQR is (62.20%, 95.70%, 78.90%), and (63.32%, 95.50%, 75.00%), respectively. Notice that on the Arrythmia dataset which consists of 279 variables, RMAM was able to efficiently offer the end user just 4 variables, reducing over 95% of the search space of the variables when contrasted with the IG, and CHI-SQR filter methods. Later on, we show the influence of reducing the search space on evaluation metrics such as error rate, precision, recall and harmonic mean among others.

The results pinpointed that on a few occasions RMAM produced slightly more variables than CHI-SQR, and IG, such as on the Breast-Cancer, Credit-Rating and Heart-Statlog datasets. Yet the difference in the number of variables selected on these datasets between RMAM and each other filter method was small and these datasets contain a limited number of variables, i.e. Breast-Cancer, Credit-Rating and Heart-Statlog datasets contain 9, 15, and 13 variables respectively. The numbers of variables selected results also show consistency between CHI-SQ and IG filter methods on most of the datasets considered as there were slight differences on the figures generated by these two methods. To be precise, on average, RMAM, IG and CHI-SQR have selected 7.59, 17.18 and 17.40 variables respectively from the considered datasets. If we compare these average figures with the average of the available variables in the original datasets (30.66) we show that RMAM reduced the search space on average by 75.20% whereas IG and CHI-SQR have reduced the search space by 44.66% and 50.00% respectively. All these results have been generated using the MHR method.

One reason for consistently recommending fewer variables to the end user when processing the dataset is the mechanism employed by RMAM during the process of assigning scores to variables. In this only fit variable that have been chosen to be part of the rules are considered during the process. Any variable value with a low reduction in the entropy is discarded at the preliminary phase and only variables' values that a) have good correlation with the class and b) when integrated with a rule have a possible rule's information gain improvement, can be selected by RMAM. Moreover, RMAM guarantees that these recommended variables have little correlation when testing variable-to-variable correlations yet sustain

Table 4: Relative difference in numbers of selected variables of RMAM and the other filter methods

Relative diff of RMAM vs Filter methods			**Relative diff of All variable's vs Filter methods**		
Dataset	**CHI**	**IG**	**RMAM**	**CHI**	**IG**
anneal	0.611	0.632	0.816	0.526	0.500
arrhythmia	0.955	0.957	0.986	0.685	0.663
audiology	0.789	0.750	0.884	0.449	0.536
autos	0.438	0.500	0.640	0.360	0.280
breast-cancer	**-0.286**	**0.286**	0.000	0.222	0.222
car	0.400	0.400	0.500	0.167	0.167
cleve	0.000	0.000	0.091	0.091	0.091
cmc	0.125	0.000	0.222	0.111	0.222
credit-rating	**-0.300**	**-0.444**	0.133	0.333	0.400
dermatology	0.813	0.806	0.824	0.059	0.088
flags	0.556	0.714	0.862	0.690	0.517
Glass	**-0.125**	**-0.125**	0.000	0.111	0.111
heart-statlog	**-0.200**	**-0.200**	0.077	0.231	0.231
hepatitis	0.077	0.077	0.368	0.316	0.316
hypothyroid	0.200	0.200	0.862	0.828	0.828
ionosphere	0.636	0.636	0.647	0.029	0.029
labor	0.500	0.500	0.625	0.250	0.250
lung-cancer	0.905	0.905	0.929	0.250	0.250
lymphography	0.357	0.438	0.500	0.222	0.111
mushroom	0.733	0.714	0.818	0.318	0.364
nursery	**-0.333**	**-0.333**	0.500	0.625	0.625
pima_diabetes	0.000	0.000	0.125	0.125	0.125
primary-tumor	0.533	0.533	0.588	0.118	0.118
segment	0.467	0.467	0.579	0.211	0.211
tic-tac-toe	**-0.167**	**-0.167**	0.222	0.333	0.333
vehicle	0.000	0.000	0.111	0.111	0.111
breast-cancer	0.556	0.556	0.556	0.000	0.000

high correlations between variable to class. This has been accomplished by a process of whenever a variable value is detected to be part of a rule RMAM immediately erases training instances connected with it hence minimizing any possible overlap among variables in the data.

Table 4 shows the relative difference in terms of the number of variables offered by RMAM when compared with IG, and CHI-SQR computed according to Equation (1) and using the MHR search method. The relative difference represents the percentage of variables RMAM has selected for a dataset in contrast to the other considered filter methods when using the MHR search method. The relative difference figures in terms of the number of chosen variables (Columns 1-2) show a reduction by RMAM and for most of the datasets when compared with IG, and CHI-SQR. The bold figures in Table 4 pinpoint the datasets in which RMAM selected more variables than any other considered filter method.

$$Relative\ Difference = \frac{(FilterMethod_{Numof-var} - RMAM_{Nun-of-var})}{FilterMethod_{Nun-of-var}} \quad (1)$$

Table 4 contains the relative difference between each filter method including RMAM and the number of variables in the original datasets respectively (Columns 3-5). The results indicate that RMAM has reduced the search space of the original datasets and is superior to the other considered filter methods for most of the datasets. This is apparent in datasets such as Anneal, Arrhythmia, Audiology, Autos, Car, Dermatology and Lung-cancer. RMAM has minimised the search space for these datasets 81.60%, 98.60%, 88.40%, 64%, 50%, 77.80% and 90.50% respectively. The results also show a good reduction on the search space of the original dataset for CHI-SQR and IG methods yet RMAM has outperformed these methods on most of the datasets. There were a few datasets in which the considered filter methods have consistency in selecting the number of variables such as Cleve, Pima-diabetes and Vehicle. Consequently, for these three datasets, the filter methods almost derive the complete variables set in the original dataset. Hence such datasets appear to contain highly correlated variables with the target variable.

4.4 Error rate results

Initially, we generated the error rates results per dataset and for each filter method using the Naïve Bayes classifier. We derived the error rate and other performance metrics per variables' set by examining 1-Variable, 2-Variable, 3-Variable, …, N-Variable where N denotes the number of variables in a dataset. Figures 3-10 illustrate error rate behavior for 8 sample datasets and for the considered variable selection methods. All the results have been obtained after deriving the weights and ranking them regardless of the filter method used. In other words, 1-Variable error denotes the error rate obtained on the "top ranked" variable for a filter method and using the designated scores of that filter method for fair comparison.

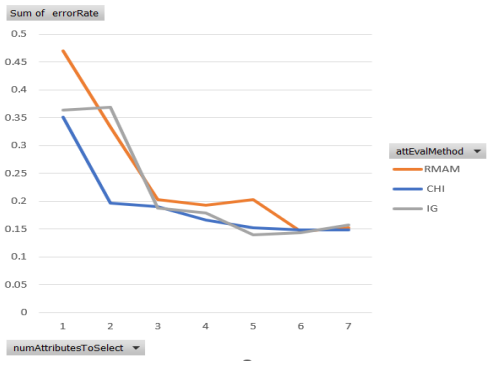

Figure 3: The errors behaviour on the Ecoli dataset by the filter methods

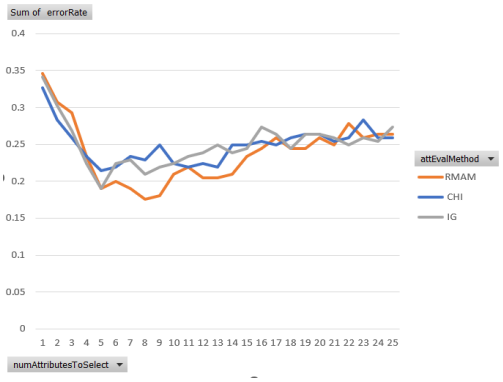

Figure 4: The errors behaviour on the Autos dataset by the filter methods

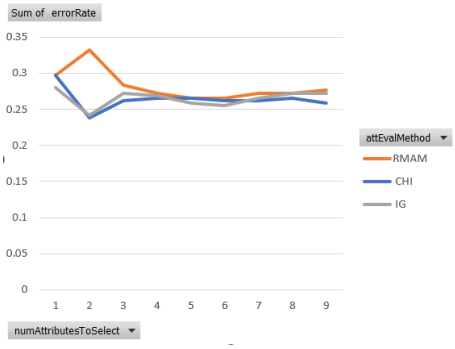

Figure 5: The errors behaviour on the Breast-cancer dataset by the filter

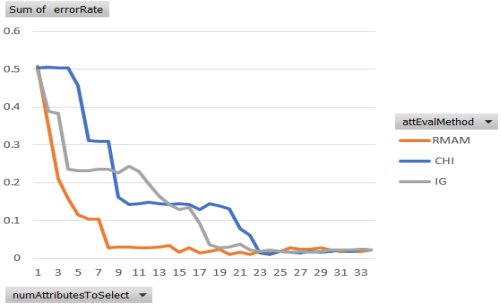

Figure 6: The errors behaviour on the Dermatology dataset by the filter methods

The displayed results show that RMAM scales well in terms of error rate with the other filter methods. For example, on the Mushroom and Dermatology datasets (Figures 10 and 6), the sum of errors using RMAM selected variables are below those of CHI-SQR and IG. Whereas, in Figures 9 and 8 (Cleve and Flag datasets) the error figures of RMAM were a bit higher than those of CHI-SQR and IG especially before 25 and 7 –Variables respectively. Lastly, Figure 4 depicts that the considered filter method has a similar performance pattern in terms of sum of errors. There are no clear patterns on the behavior of error rate with respect to the number of selected variables in most of the considered datasets. In addition, there is no clear pattern on the performance of the classifiers produced against the sets of variables per dataset using the considered filter methods.

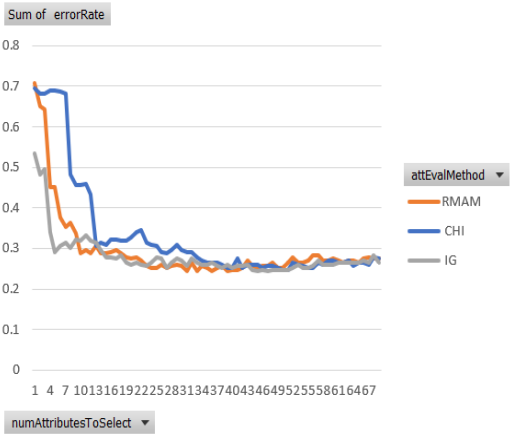

Figure 7: The errors behaviour on the Audiology dataset by the filter methods

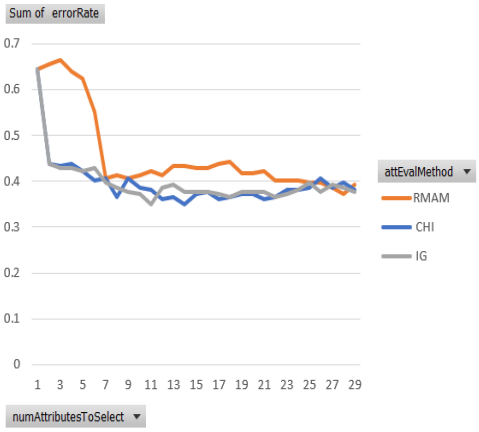

Figure 8: The errors behaviour on the Flag dataset by the filter methods

For example, in the Autos dataset (Figure 4) the error rates generated against RMAM's variables sets were decreasing from 1-Variable to 7-Variable sets and then not improving after the 7-Variable set. For the same dataset, the error rate generated against IG's Variables sets decreases from 1-Variable to 4-Variable subsets, increased from Variable-4 onward. The same pattern was shown for CHI-SQR. However, for the error rates derived against the Dermatology dataset (Figure 6), the pattern is similar for the filter methods' variables sets derived classifiers with notable error's stabilization for RMAM method after only selecting 7-variable set. Whereas IG and CHI-SQR error rate was stabilized after 19-variable set.

It is also noted in some Figures (e.g. Figure 4) that in many cases when the number of variables increases this has nothing to do with the error rate; this potentially falsifies any claim that more variables lead to improved performance of the classifiers. For instance, Figure 4 shows an increase in the error rate from after processing 5-variable subset in most cases regardless of the filter method used.

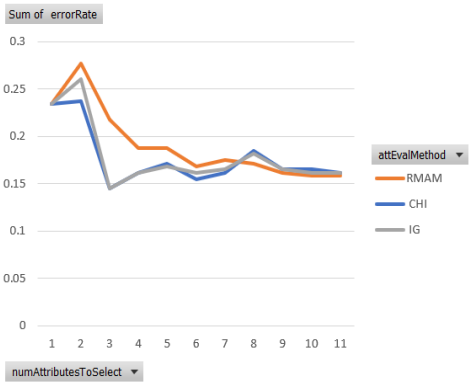

Figure 9: The errors behaviour on the Cleve dataset by the filter methods

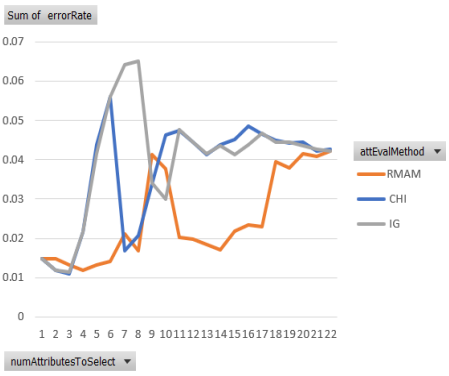

Figure 10: The errors behaviour on the Mushroom dataset by the filter methods

Table 5 displays the error rate, precision, recall and F1 results derived on the datasets based on the cut-off points of the MHR method and for each filter method on the considered datasets using the Naive Bayes classifier. In other words, we show in that table the classifiers' results derived against the subsets of variables based on the cut-off points per filter method and on each dataset. The bold figures in column 2 reveal the datasets that processed by Naive Bayes achieved the lowest error percentage when employing the RMAM method with MHR. There was better performance in terms of error rate

on 13 datasets when subsets of variables were chosen by RMAM and processed by Naïve Bayes classifiers These results on the datasets considered indicate that the RMAM filter method was able to select good sets of variables according to the suggested cut-off values computed by the MHR Ranker method. This is clear in the results generated against Anneal, Autos, Ionosphere and Lung-cancer among others. For these datasets and others not only did RMAM with MHR Ranker select fewer variables and reduce the search space of variables but also classifiers derived against their subsets were superior to those selected by other filter methods including CHI-SQR and IG. This is beneficial for the following reasons:

1) A minimal set of variables is offered with a high chance of good performance when compared to the complete dataset variables or variables chosen by other filter methods
2) The end user can exploit and manage the entire dataset using the offered variables
3) A reduction in the search space and computing resources
4) The process of checking variable selection outcomes manually by the recommended cut-off points is replaced regardless of the filter method used
5) A mathematical configuration of the cut-off is used rather than a complete variable set ranked based on scores

Table 5: Error, precision, recall and F1 rates derived by Naïve Bayes algorithm on the selected variables of RMAM and the other filter methods

	Error %			Precision %			Recall %			F1-Measure %		
Dataset	**RMAM**	**IG**	**CHI-SQR**	**RMAM**	**IG**	**CHI-SQR**	**RMAM**	**IG**	**CHI-SQR**	**RMAM**	**IG**	**CHI-SQR**
anneal	**9.02**	14.92	13.36	94.01	93.11	93.50	90.98	85.07	86.63	91.57	87.40	88.64
arrhythmia	37.61	31.85	31.41	54.15	66.70	67.12	62.38	68.14	68.58	57.10	66.66	67.10
audiology	35.39	26.10	25.22	55.47	67.92	68.51	64.60	73.89	74.77	59.22	39.92	70.47
autos	**38.04**	40.97	41.95	60.46	62.74	62.64	61.95	58.02	58.04	60.31	58.93	58.33
breast-cancer	27.62	27.62	27.62	70.79	71.12	71.12	72.37	72.37	72.37	71.40	71.53	71.53
car	20.89	14.64	14.64	74.39	84.91	84.91	79.10	85.35	85.35	76.54	84.57	84.57
cleve	**15.85**	17.16	17.16	84.18	82.83	82.83	84.15	82.83	82.83	84.16	82.83	82.83
cmc	**48.26**	49.89	49.89	53.14	52.14	52.14	51.47	50.10	50.10	52.11	50.58	50.58
credit-rating	23.18	23.04	23.04	78.61	78.63	78.63	76.81	76.95	76.95	75.89	76.08	76.08
dermatology	8.74	1.63	1.63	91.08	98.72	98.72	91.25	98.36	98.36	90.07	98.34	98.34
flags	43.81	37.13	39.17	50.57	60.58	59.09	56.18	62.88	60.82	52.13	60.77	59.29
Glass	**51.40**	52.23	52.23	49.55	49.52	49.52	48.59	47.66	47.66	45.25	44.63	44.63
heart-statlog	**15.18**	15.92	15.92	84.80	84.07	84.07	84.81	84.07	84.07	84.77	84.01	84.01
hepatitis	14.83	13.54	12.90	86.13	86.97	87.78	85.16	86.45	87.09	85.54	86.47	87.36
hypothyroid	5.38	5.27	5.27	93.32	93.62	93.62	94.61	94.72	94.72	93.52	93.77	93.77
ionosphere	**8.83**	17.83	17.83	91.15	84.17	84.17	91.16	82.62	82.62	91.16	82.90	82.90
labor	14.03	10.52	12.28	85.83	89.47	87.90	85.96	89.47	87.71	85.78	89.47	87.78
lung-cancer	**9.07**	21.87	21.87	90.49	77.47	77.47	90.62	78.12	78.12	90.44	77.71	77.71
lymphography	18.24	16.21	19.59	81.52	83.98	81.03	81.75	83.78	80.40	81.45	83.61	80.54
mushroom	**1.18**	4.28	4.38	98.84	95.95	95.80	98.81	95.71	95.80	98.81	95.70	95.60
nursery	**12.26**	12.34	12.34	85.83	85.67	85.67	87.73	87.65	87.65	86.69	86.61	86.61
pima_diabetes	**23.17**	23.56	23.56	76.37	75.94	75.94	76.82	76.43	76.43	76.47	76.03	76.03
primary-tumor	58.99	50.14	50.14	35.25	43.84	43.84	41.00	49.85	49.85	35.09	45.48	45.48
segment	**19.56**	20.99	20.99	81.06	80.67	80.67	80.43	79.00	79.00	80.56	76.58	76.58
tic-tac-toe	26.93	26.82	26.82	72.34	72.49	72.49	73.06	73.17	73.17	70.87	70.96	70.96
vehicle	**54.27**	56.38	56.38	52.74	50.49	50.49	45.27	43.61	43.61	41.22	39.40	39.40
wisconsin-breast-cancer	4.14	4.00	4.00	95.97	96.15	96.15	95.85	95.99	95.99	95.87	96.02	96.02

The figures of Tables 2&4 suggest that in some cases when RMAM cut-offs are larger than those of the considered filter methods, the error rates of Naïve Bayes on RMAM's variables sets are better than those derived on the those of CHI-SQR and IG respectively. Noticeable results were obtained on the Arrhythmia, Audiology, and Dermatology datasets among others in which the error rates derived by the Naïve Bayes classifier against RMAM's variables subsets were larger than those derived from the other filter method subsets. After a careful investigation of these three datasets RMAM with the MHR search method determines the cut-off point at an earlier iteration than other filter methods.

For example, for the Audiology dataset RMAM stopped adding variables when the variable set size was 8. As a result, RMAM loses some predictive accuracy performance on this dataset instead of appending more variables for up52 to reach the optimal performance when compared to the other filter methods. In other words, for a saving of around 85% of the search space RMAM had to sacrifice around 8% predictive accuracy.

A similar case is the Arrhythmia dataset in which RMAM stops adding variables early at the 4-Variable set instead of continuously appending more variables to reach an optimal performance with respect to error rate at the 52-Variable set. On these two datasets, CHI-SQR and IG methods with MHR search methods retained (38, 32) and (88, 94) variables respectively, aiming to enhance the performance in the final sets offered to the end user. This has resulted in an increase of predictive accuracy performance by (6.20%, 6.42%) and (10.17%, 9.29%) when processing the variables subsets of CHI-SQR and IG on Arrhythmia and Audiology dataset respectively than those of RMAM variables sets.

The precision and recall results shown in Table 5 are consistent with the results reported earlier on error rate. Naïve Bayes derived better classifiers when processed RMAM variables sets on 12, 13 and 13 datasets respectively in terms of precision, recall and F1 metrics
To give insight on the precision and recall results we take two datasets as samples:

- One where Naïve Bayes on RMAM variables set produced higher results and
- One that produced low results when compared with results obtained on IG, and CHI-SQR

For the first case, we used the Ionosphere dataset that contains 351 instances, 34 variables and a target class with two possible values (Good, Bad). This dataset is concerned with radar data to categorize the type of signals in the ionosphere. The true positives, false positives, true negatives and false negatives derived by Naïve Bayes against each filter method's variables sets on the Ionosphere dataset are shown in Tables 6A-6C. In these tables it is obvious that the numbers of misclassification for class Good are higher than class Bad and for all subsets of variables regardless of the filter method used.

However, these numbers of misclassification are smaller when RMAM variables sets are processed by Naïve Bayes algorithm; the false positives on RMAM are 14 for class Bad whereas 17 on the remaining filter methods. In addition, Naïve Bayes achieved a less false negatives when it processed a RMAM variables set than for those using other remaining filter methods. There were 15 false negatives on RMAM whereas IG and CHI-SQR variables set when processed by Naïve Bayes resulted in 43 and 45 false negatives respectively These results contributed to improving the recall and precision rates of the derived classifiers on the RMAM variables set when compared with those derived from IG and CHI-SQR variables sets on this dataset.

For the second case we used the Car dataset and the confusion matrix derived by the Naïve classifier on the distinctive variables sets of IG CHI-SQR and RMAM are shown in Tables 7A-7C. This dataset consists of 6 variables excluding the target class and contains 1,728 instances and 4 possible class values (Unacc, Acc, Good, VGood). The dataset is imbalanced in terms of the class labels as Unacc class has 1,210 frequencies followed by 384, 69, and 65 frequencies respectively for Acc, Good and VGood class labels. As a result, the confusion matrix results show clearly that in most cases instances that belong to class labels Acc, VGood and Good have been misclassified by Naïve Bayes to Unacc regardless of the variables set processed. The case is clear when Naïve Bayes processed a RMAM variables set in which 484 misclassifications are encountered on Acc, VGood and Good class labels.

Further, excluding variables Lug_boot and Maint from the original Car dataset by RMAM the imbalanced class labels issued have resulted in larger false negatives at least for class labels Acc, VGood and Good. When these two variables are included then Naïve Bayes derived better classifiers with respect to precision and recall (in the case of the IG and CHI-SQR variables sets). These results,

Table 6A: confusion matrix derived by Naive Bayes on the RMAM variables set of Ionosphere dataset

	Bad	Good
Bad	112	14
Good	15	210

Table 6B: confusion matrix derived by Naive Bayes on the IG variables sets of Ionosphere dataset

	Bad	Good
Bad	109	17
Good	43	182

Table 6C: confusion matrix derived by Naive Bayes on the CHI-SQR variables sets of Ionosphere dataset

	Bad	Good
Bad	109	17
Good	45	180

if limited, show that RMAM with the MHR Ranker method is sensitive to the class imbalance issue in classification datasets. Another probable reason is the limited number of variables in the original Car dataset, i.e. 6, in which when 2 variables have not been included by RMAM the performance of the classifiers deteriorated. The precision and recall results on the Car dataset are approximately 6% lower for classifiers generated from RMAM variables set when compared with those generated from IG and CHI-SQR.

The harmonic mean (F1) considers both recall and precision especially in cases when datasets are imbalanced with respect to class labels such as the Car dataset. Thus, a weighted average of both the recall and precision can be useful as error rate or predictive accuracy metrics do not work well in cases when false negatives and false positives are substantially different. The F1 scores obtained on the different variable sets of the filter methods using the Naïve Bayes classifier are shown in Table 5. The F1 scores results are consistent with the error rate, recall and precision results reported earlier. When Naive Bayes has been used to process the RMAM variables set the

Table 7A: confusion matrix derived by Naive Bayes on the RMAM variables set of Car dataset

	Unacc	Acc	Vgood	Good
Unacc	1152	56	2	0
Acc	97	277	10	0
Vgood	0	45	18	6
Good	0	37	0	28

Table 7B: confusion matrix derived by Naive Bayes on the IG variables sets of Car dataset

	Unacc	Acc	Vgood	Good
Unacc	1142	52	11	5
Acc	336	46	2	0
Vgood	37	17	12	3
Good	27	28	9	1

Table 7C: confusion matrix derived by Naive Bayes on the CHI-SQR variables sets of Car dataset

	Unacc	Acc	Vgood	Good
Unacc	1194	16	0	0
Acc	364	20	0	0
Vgood	63	6	0	0
Good	57	8	0	0

classifiers produced have superiority over those derived from the variables sets of IG and CHI-SQR at least on 13 datasets.

4.5 Discussion

Across the 28 datasets, the proposed method demonstrated strong performance while significantly reducing the number of variables. In terms of dimensionality reduction, it selected fewer variables than competing methods on 18 out of 28 datasets, achieving an average reduction of 75.20% compared to 44.66% and 50.00% for the alternative methods. Importantly, this reduction did not come at the cost of performance in most cases. The method achieved lower error rates on 13 datasets, while also producing superior results in precision, recall, and F1-score on 12–13 datasets. In a smaller number of datasets, the method accepted minor reductions in predictive performance (approximately 6–10%) in exchange for dimensionality reduction particularly in high dimensional settings. These trade-offs highlight the method's ability to balance performance and simplicity.

A key strength of the proposed approach is that it goes beyond merely assigning scores to variables as it provides a recommendation for the number of variables to retain, offering a complete variable selection outcome. This is particularly valuable in practical domains such as medicine, finance, and security, where practitioners require clear guidance rather than long, ambiguous ranked lists of variables. By integrating both a scoring mechanism and an automated cut-off procedure within a unified data driven approach, the proposed method delivers a full variable selection pipeline. This reduces reliance on user judgment, minimizes the risk of subjective decisions, and improves usability for both expert and non-expert users.

In interpreting the confusion matrices results, it is important to consider the context in which the models are applied for users working with imbalanced datasets. For example, in the Ionosphere dataset, RMAM achieved fewer false positives and false negatives compared to other methods, leading to improved precision and recall. However, in datasets such as Car, where class distribution is highly skewed, most data instances belong to a dominant class (Unacc), which can bias the classifier toward that class. As a result, minority classes (e.g., Acc, Good, VGood) are more likely to be misclassified, regardless of the variable selection method used. Therefore, practitioners like those in domains such as healthcare or fraud detection where minority classes are critical should pay close attention to recall and precision for minority classes rather than relying on overall accuracy. In such cases, metrics like F1-score provide a more balanced evaluation of model performance.

In practice, the proposed method can be integrated in multiple domain applications to support managers, and other stakeholders in key relate decision making. For instance, the development of computer-aided diagnostic tools, particularly for early dementia detection, the proposed method can be integrated to help clinicians understand variables, their interactions and more importantly their importance to the disease progression. In this context, cognitive and clinical data can be input into the system, where the method identifies a compact set of the most informative variables. These variables can then be used to train classification models that assist clinicians in identifying early signs of cognitive decline besides mapping such variables to important cognitive domains. The automated cut-off ensures that only the relevant variables are retained thus simplifying interpretation and decision-making. Once deployed within a clinical decision-support system, the model can help in pinpointing to key clinical and cognitive areas associated with each patient over time thus providing timely alerts for potential progression toward dementia.

We have now clarified the computational aspects of RMAM and MHR in the revised manuscript. From a theoretical perspective, RMAM involves two main stages: rule generation and variable scoring. The rule generation process requires iterating over the training dataset and computing IG for candidate variable-value pairs, leading to an approximate complexity of O(n·m·logm), where n is the number of variables and m is the number of instances. The MHR ranking stage involves sorting and iterative merging operations, with a complexity of O(nlogn), which is comparable to standard ranking procedures.

From an empirical perspective, RMAM introduces additional overhead compared to traditional filter methods such as IG and CHI-SQR, primarily due to the rule generation phase. However, this cost is incurred during preprocessing and is offset by the significant reduction in feature space (on average ~75%), which can reduce downstream model training time.

In addition to computational considerations, the method exhibits some sensitivity to design choices, particularly in the discretisation of continuous variables and the handling of missing values, which may influence rule generation and subsequent variable ranking. The quality and granularity of discretisation can affect calculations thus potentially impacting which variables are selected or discarded. Furthermore, RMAM is primarily designed for structured, tabular data and may require adaptation for non-tabular data types such as text, images, or graph-based representations, where feature interactions are less explicit and may not be easily captured through rule-based formulations.

The approach may also be sensitive to class imbalance, as rule generation is driven by frequency and entropy reduction, which can bias the selection towards dominant class patterns in skewed datasets. In addition, while the MHR cut-off mechanism provides a practical and automated threshold, its behaviour under different data distributions has not been fully theoretically characterised and may vary depending on dataset properties. These limitations highlight potential areas for future enhancement, including improving scalability through optimisation or parallelisation, incorporating imbalance-aware mechanisms, refining discretisation strategies, and extending the framework to support more complex and unstructured data modalities.

5 Conclusion

This paper presented RMAM and its associated ranking mechanism (MHR) as a filter variable selection approach aimed at reducing data dimensionality while providing a practical recommendation for variable subset size. A key contribution of this work lies in the integration of rule-based variable weighting with an adaptive cut-off mechanism thus enabling a complete variable selection pipeline that not only ranks variables but also determines how many should be retained without user intervention. The empirical evaluation conducted on 27 benchmark datasets demonstrated that RMAM consistently produces compact variable subsets by selecting fewer variables than competing methods on 18 datasets and achieving an average reduction of approximately 75.20%, compared to 44.66% and 50.00% for IG and Chi-Square, respectively. Importantly, this reduction was achieved while maintaining comparable predictive performance in most cases, with lower error rates observed on 13 datasets and improved precision, recall, and F1-scores on 12–13 datasets. The results further show that the MHR cut-off mechanism effectively identifies a practical stopping point, supported by observed error rate stabilisation when incrementally adding variables. In some high-dimensional cases, RMAM accepts a small reduction in predictive accuracy (approximately 6–10%) in exchange for substantial dimensionality reduction thereby highlighting a practical balance between performance and simplicity. From an applied perspective, the method offers clear benefits by reducing the need for manual variable inspection and supporting users in identifying concise and interpretable variable sets.

Despite these encouraging results, several limitations should be acknowledged. The evaluation was conducted using a single classifier (Naïve Bayes), and performance may vary when integrated with other learning algorithms. Additionally, while consistent empirical trends were observed across heterogeneous datasets, further statistical validation could strengthen the findings. The method also showed sensitivity in datasets with strong class imbalance or very small variable spaces. Moreover, the discretisation of continuous variables and treatment of missing values may influence the generated rules and subsequent rankings. The reliance on entropy-based measures may also limit sensitivity to complex non-linear relationships. Future work will extend the evaluation to multiple classifiers such as ensemble methods, investigate robustness under varying data characteristics, and explore integration with domain specific applications particularly in medical contexts such as dementia diagnosis. Additional directions include deeper analysis of cut-off behaviour to enhance generalisability and practical applicability.

Declaration

‘Funding’ and/or ‘Competing interests’.
We declare that all authors contributed to this research article.
We declare that there is no funding obtained to conduct this research.
We declare that there is no conflict of interests.

Data and reproducibility statement

The datasets used in this study are publicly available from the UCI Machine Learning Repository (https://archive.ics.uci.edu/), and are cited within the manuscript. All experiments were conducted using these benchmark datasets in WEKA tool with standard preprocessing steps, including entropy-based discretisation and handling of missing values as described in Section 4.1.

The proposed RMAM method was implemented in Java and integrated within the WEKA environment. Due to current institutional and project constraints, the Java

implementation code is not publicly available at the time of submission. However, the methodology, algorithms, and parameter settings have been described in sufficient detail within the manuscript to enable replication of the study.

References

[1] Al-Dhaheri, S. (2021). A New Variable Selection Method Based on Class Association Rule (Doctoral dissertation, City University of New York).

[2] Azar, A. T., Boubellouta, A., Bouzeriba, A., & Zouari, F. (2025). Novel fixed-time fuzzy sliding-mode synchronization schemes of two uncertain dissimilar chaotic systems. Soft Computing. Advance online publication

[3] Begum, A. M., Mondal, M. R. H., Podder, P., & Kamruzzaman, J. (2024). Weighted Rank Difference Ensemble: A New Form of Ensemble Variable Selection Method for Medical Datasets. BioMedInformatics, 4(1), 477-488.

[4] Boulkroune, A., Boubellouta, A., Bouzeriba, A., & Zouari, F. (2025). Practical finite-time fuzzy synchronization of chaotic systems with non-integer orders: Two chattering-free approaches. Journal of Systems Science and Systems Engineering, 34, 334–359.

[5] Dua, D., & Graff, C. (2019). UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences. https://doi.org/10.24432/C5J30R

[6] Hammoud, M. (2009). Introduction to key-value stores. Proceedings of the IEEE Computer Society Annual Symposium on VLSI, 198-203. https://doi.org/10.1109/ISVLSI.2009.44

[7] Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. Proceedings of the IRE, 40(9), 1098-1101. https://doi.org/10.1109/JRPROC.1952.273898

[8] John, G. H., & Langley, P. (1995). Estimating continuous distributions in Bayesian classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI '95), 338-345. Morgan Kaufmann Publishers Inc. https://doi.org/10.5555/2074158.2074196

[9] Karimi, K., Ghodratnama, A., & Tavakkoli-Moghaddam, R. (2023). Two new variable selection methods based on learn-heuristic techniques for breast cancer prediction: a comprehensive analysis. Annals of Operations Research, 328(1), 665-700.

[10] Khan, M. A., & Afzal, H. (2019). Efficient variable selection for sentiment analysis in social media data using hybrid method. Journal of Ambient Intelligence and Humanized Computing, 10(9), 3527-3545. https://doi.org/10.1007/s12652-019-01367-6

[11] Kumar, M., & Kaur, H. (2020). Variable selection for high-dimensional data using hybrid model based on information gain and chi-square. International Journal of Advanced Computer Science and Applications (IJACSA), 11(4), 435-440. https://doi.org/10.14569/IJACSA.2020.0110455

[12] Li, Chao, Xiao Luo, Yanpeng Qi, Zhenbo Gao, and Xiaohui Lin (2020) A new variable selection algorithm based on relevance, redundancy and complementarity. Computers in biology and medicine 119 (2020): 103667.

[13] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2018). Variable selection: A data perspective. ACM Computing Surveys (CSUR), 50(6), 1-45.

[14] McHugh, M. L. (2013). The Chi-square test of independence. Biochemia Medica, 23(2), 143-149. https://doi.org/10.11613/BM.2013.018

[15] Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81-106. https://doi.org/10.1007/BF00116251

[16] Rajab, M. D., Taketa, T., Wharton, S. B., Wang, D., & Cognitive Function and Ageing Neuropathology Study, and for the Alzheimer's Disease Neuroimaging Initiative. (2024). Ranking and filtering of neuropathology variables in the machine learning evaluation of dementia studies. Brain Pathology, e13247.

[17] Rigatos, G., Busawon, K., Abbaszadeh, M., & Zouari, F. (2020). Flatness-based adaptive fuzzy control for the Uzawa–Lucas endogenous growth model. AIP Conference Proceedings, 2293(1), 310003.

[18] Rigatos, G., Abbaszadeh, M., & Zouari, F. (2026). Flatness-based control in successive loops for dual-arm robotic manipulators. Journal of Vibration and Control, 32(3–4), 488–507.

[19] Rigatos, G., Busawon, K., Abbaszadeh, M., Pomares, J., Gao, Z., & Zouari, F. (2024). Flatness-based control in successive loops for autonomous quadrotors. Journal of Dynamic Systems, Measurement, and Control, 146(2), Article 011106

[20] Salehi, H., Rashidi, H., & Abadeh, M. S. (2021). Hybrid variable selection methods: A survey and critique of filter, wrapper, and embedded strategies. Artificial Intelligence Review, 54, 4775-4827. https://doi.org/10.1007/s10462-021-09984-x

[21] Thabtah F., Kamalov F., Hammoud S., & Shahamiri S. R. (2020). Least Loss: A simplified filter method for variable selection. Information Science Journal, Volume 534, September 2020, Pages 1-15

[22] Tran, H. N., & Huynh, H. T. (2022). Variable selection methods for high-dimensional data: A comparative study and a new method based on metaheuristics. Knowledge-Based Systems, 243, 108424. https://doi.org/10.1016/j.knosys.2022.108424

[23] Yu, T., Liu, Z., Liu, Y., Wang, H., & Adilov, N. (2020, November). A New Variable Selection Method for Intrusion Detection System Dataset–TSDR method. In 2020 16th international conference on computational intelligence and security (CIS) (pp. 362-365). IEEE.

[24] Zhang, Y., Liu, X., & Wang, J. (2023). A comprehensive review on variable selection methods

in machine learning. IEEE Access, 11, 10574-10588. https://doi.org/10.1109/ACCESS.2023.3245678

[25] Zhao, Y., Huang, Y., Wang, Z., & Liu, X. (2024). A new variable selection method based on importance measures for crude oil return forecasting. Neurocomputing, 581, 127470.

[26] Zhao, Z., & Liu, H. (2020). On the challenges in variable selection with big data. IEEE Intelligent Systems, 35(2), 40-45. https://doi.org/10.1109/MIS.2020.2972094

[27] Zubair, I. M., Lee, Y. S., & Kim, B. (2024). A New Permutation-Based Method for Ranking and Selecting Group Variables in Multiclass Classification. Applied Sciences, 14(8), 3156.