International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 10 No. 4, 2024
https://www.theaspd.com/ijes.php

Rank Based Iterative Relief Technique For Spam Mail Recognition

B. Aruna Kumari[1], **C. Nagaraju**[2]
[1]Research Scholar, YSR Engineering College of Yogi Vemana University, Computer Science & Engineering, Proddatur, arunakumarib1421@gmail.com
[2]Professor, YSR Engineering College of Yogi Vemana University, Computer Science & Engineering, Proddatur, nagaraju.c@yvu.edu.in

Abstract
Spam emails are unsolicited bulk messages often used for malicious purposes, such as phishing and identity theft. They clutter inboxes, consume bandwidth, and strain resources. Detecting spam is challenging since it frequently shares characteristics with legitimate emails, and irrelevant features can hinder classification accuracy. Feature selection methods eliminate irrelevant features by identifying the most relevant ones. This leads to improved performance, minimized overfitting, reduced dimensionality, and enhanced interpretability. The original Relief algorithm is a feature selection technique that allocates weights to features according to their effectiveness in differentiating instances of distinct classes. Nonetheless, it has drawbacks, including sensitivity to noisy data and challenges in managing redundant features. Relief variants (A–D) improve weight updates and feature dependency management but still struggle with outliers. To enhance spam detection, an iterative version of the Relief algorithm is integrated with the Random Forest classifier. This approach removes irrelevant features and improves efficiency. Evaluation results indicate that this method outperforms the original Relief algorithm and its variants (Relief A–D).
Keywords: *Feature Selection, Relief, ReliefA-D, Iterative Relief*

INTRODUCTION

The Internet is a vital aspect of modern life, influencing nearly every facet of our daily activities. More than two-thirds of the global population utilizes the Internet to connect with others, communicate, and share information. Email is one of the most popular ways to communicate online [1]. It effectively allows people to send messages and share information over long distances. Email is widely used, but it also has challenges, especially with spam attacks. Spam includes unwanted messages, such as fake promotions, misleading offers, or harmful content. These can damage our device or steal our personal information. Spam emails are dispatched to a large number of recipients simultaneously, with the primary objective of deceiving individuals into disclosing sensitive information. This can result in significant financial loss or identity theft [2]. Distinguishing between spam and legitimate emails can be challenging. The methods used by spammers are constantly evolving, making it harder for users to differentiate authentic emails from fraudulent ones. To combat the increasing threat of spam, various anti-spam tools and strategies have been introduced. Companies use different methods to stop spam emails. They have email systems that block spam before it reaches users. There are also mail filtering gateways that check incoming messages for anything suspicious. Additionally, businesses can hire antispam services that offer expert help to tackle this problem. Despite these efforts, many solutions remain ineffective, as spammers continually adapt to evade filters. Many models have been created and tested over the years to address these ongoing challenges. However, none have reached the level of accuracy that users need.

In machine learning and pattern recognition, data often has many features. This variety makes the analysis deeper and more effective. Many spam emails look similar to real emails because they use the same format, words, and structure. This makes it hard to tell them apart. The presence of overlapping features in the classification process can result in the inclusion of irrelevant features. This not only prolongs the time needed to accurately categorize emails but also reduces the overall precision of the classification outcomes. In the realm of machine learning and data analysis, selecting the most relevant features is crucial for building effective models.

We are particularly enthusiastic about relief-based methods, as they effectively rank features through strategically assigned weights, thereby enhancing the overall quality of feature selection. This method has the advantage of using the Relief algorithm, which is different from wrapper methods. The Relief algorithm does not require exhaustive searches for combinations and works well with various learning algorithms. It often works better than other methods for choosing important features. This is because it receives helpful feedback from a non-linear classifier while searching for relevant features [3]. Relief-based methods effectively demonstrate the connections among features by utilizing local information. They also provide a broader perspective that aids in identifying the most relevant features.

Relief

Kira and Rendell [4,5], created the original Relief algorithm, which is based on instance-based learning. Relief is a method used for selecting important features in data. It works as a tool for evaluating each feature individually. The text discusses a method for assessing the relevance of features in predicting a target concept, specifically the endpoint value. It involves calculating a proxy statistic for each feature to provide a quantitative measure of its quality. This statistic indicates the effectiveness of each feature in distinguishing between various classes or outcomes. Relief looks at how features interact and impact the target variable. It identifies which features help make accurate predictions. This process is useful for choosing the most important variables in machine learning or statistical modelling tasks. When analyzing a target instance, the nearest data points in the same class are called 'hits', while those in a different class are referred to as 'misses'. Hits are relevant examples similar to the target, while misses are dissimilar instances or belong to different categories. The Relief feature selection algorithm looks at the nearest neighbour (NN) to understand how different classes of data compare to each other. It finds the nearest neighbour for a given target instance and then checks the differences between the two classes. The classification decision regarding the target instance is determined through a thorough analysis of its nearest neighbours across both classes. Feature weights are statistics associated with individual characteristics and are represented as W[A] [6], where 'A' signifies a specific feature. Informally, these weights can also be referred to as feature 'scores.' Each feature score reflects the importance or relevance of that feature, with a possible range from -1 to +1. A score of -1 indicates that the feature has a negative impact or is considered the least favorable, while a score of +1 signifies a highly positive influence, representing the most favorable outcome. Feature weights are statistics associated with individual characteristics and are represented as W[A], where 'A' signifies a specific feature [7]. Informally, these weights can also be referred to as feature 'scores.' Each feature score reflects the importance or relevance of that feature, with a possible range from -1 to +1. A score of -1 indicates that the feature has a negative impact or is considered the least favorable, while a score of +1 signifies a highly positive influence, representing the most favorable outcome. For an attribute A, Relief is an approximation of the difference between two probabilities, as described in Eq (1).

$$W[A] = P\left(\frac{\text{same value of A}}{\text{hit class}}\right) - P\left(\frac{\text{different value of A}}{\text{miss class}}\right) \qquad \text{Eq. (1)}$$

Each feature weight (w[A]) is calculated for every target instance and is updated accordingly in Eq. (2).

$$W[A] = \sum_{n=1}^{N} \frac{d(x_n, miss(x_n)) - d(x_n, hit(x_n))}{N} \qquad \text{Eq. (2)}$$

In this context, let N represent the total number of instances and x_n the target instance. The variable d calculates the difference between hits and misses relative to the target instance. The feature weight, W[A], is limited to a range of 0 to 1. The method for calculating differences varies for numerical and categorical data, with Equation 2 for numerical data and Equation 3 for categorical data.

$$d(x_n, NN\,(x_n)) = \begin{cases} 0, & \text{same value } x_n \text{ and } NN(x_n) \\ 1, & \text{otherwise} \end{cases} \qquad \text{Eq. (3)}$$

The difference in numerical data is computed according to Eq. (4), yielding values that range from 0 to 1.

$$d(x_n, NN\,(x_n)) = |value(x_n) - value(NN(x_n))| \qquad \text{Eq. (4)}$$

Relief methods help evaluate how important different features are in binary classification, which involves distinguishing between two classes. However, relying only on the nearest neighbour can add a lot of noise to the decision-making process. This issue becomes serious when the dataset has noisy or irrelevant features, missing values, or unknown values. Outliers can also distort the results. As a result, this may cause incorrect classifications or misunderstandings about feature importance.

ReliefA

ReliefA is a significant improvement over the original Relief algorithm, specifically designed for noisy datasets. ReliefA improves the evaluation of neighbouring instances by incorporating contributions from multiple nearest hits and misses, rather than relying on just one. Averaging these contributions gives the algorithm a better understanding of feature relationships, enhancing the robustness and performance of ReliefA across diverse datasets. This multi-neighbour approach improves the algorithm's noise filtering and feature identification, resulting in better feature weighting and enhanced model performance. The algorithm focuses on local relationships among features, enhancing its understanding of their interactions in specific contexts. The difference in numerical data is assessed using Eq. (5), producing values that fall within the range of 0 to 1.

$$d(x_n, NN\ (x_n)) = \frac{|value(x_n) - value(NN(x_n))|}{max(A) - min(A)} \qquad \text{Eq. (5)}$$

Here We use the Max-min method to evaluate the difference between maximum and minimum values of A across all instances. This process normalizes the weights to stay between 0 and 1, which helps prevent any single feature from having too much influence on the model. To update the weight vector W[A], we divide the difference by m, the number of samples. This keeps the updates consistent and normalizes the final weights. Ultimately, this ensures that all weight values are between -1 and 1, improving the model's stability and performance during training.

However, both Relief and ReliefA algorithms struggle with missing values, limiting their ability to accurately weigh features. This challenge will be addressed by the ReliefB algorithm.

ReliefB

The ReliefB algorithm is a powerful advancement over the original Relief algorithm, designed specifically for feature selection in machine learning. It significantly improves performance by effectively identifying the most relevant features. Relief-B improves its algorithm to handle missing values more effectively [8]. When feature A is missing for a specific instance or for one of its nearby instances (called H or M), the algorithm uses a method to calculate distances based on probabilities.

The difference function d(x_n,y) for a feature A in Relief-B is defined as follows:

$$d(x_n, NN\ (x_n)) = 1 - \frac{1}{\#\ of\ values\ in\ A} \qquad \text{Eq. (6)}$$

Where x_n denotes the target instance and $NN\ (x_n)$ specifies one nearest hit and miss without unknown values and which are normalized to scaling factor in between 0 and 1 are shown in Eq. (6).

When the algorithm finds a missing value for A in the dataset or its nearest neighbours, it fills the gap using an estimated value based on the available observed values of A. The algorithm assumes that all known values of A have an equal chance of being selected. This method helps it measure how similar or different the data points are, which improves the accuracy of its predictions. When handling known values, the distance metric measures the actual difference—either numerical or categorical—between the feature values of the sample point and its nearest neighbour. To ensure a balanced contribution from different features, we use a scaling factor of $\frac{1}{\#\ of\ values\ in\ A}$. This scaling factor helps balance the impact of A, which may have different numbers of potential values for the features.

Despite the advancements made with the Relief-B algorithm in comparison to its original counterpart, it is important to acknowledge certain limitations associated with the handling of unknown values. These limitations stem from the intrinsic challenges in estimating distances and relevance within incomplete datasets. The ReliefC algorithm, however, is poised to effectively address these concerns.

Relief C

ReliefC is an updated version of the original Relief algorithm. It is designed to work with datasets that have missing values. This algorithm improves the process of selecting important features by assessing how well each feature helps to tell apart different classes. It does this by examining how features relate to each other and their role in identifying class differences, even when the data is incomplete. In Relief-C, the rule from Relief-B that requires at least one missing value in the dataset for selecting the k value is no longer followed. Instead of picking the k value randomly, it is calculated as the square root of the total number of records in the dataset. This method has been found to provide better accuracy than all other ways of choosing k, including random selection.

$$d(x_n, NN\ (x_n)) = 1 - P(value\frac{(x_n, NN(x_n))}{y_n}) \qquad \text{Eq. (7)}$$

In above Eq. (7), NN (x_n) represents one nearest hit and one nearest miss which are having unknown values and y_n denotes the class of the target instance.

ReliefC has problems when it comes to dealing with outliers in a dataset. Outliers can greatly affect the way we identify near-hits and near-misses, which are important for updating feature weights accurately. When these extreme values are present, they can lead to incorrect results and ultimately weaken the algorithm's effectiveness.

ReliefD

ReliefD works better than both ReliefB and ReliefC when predicting and filling in missing data. This means ReliefD is more effective at accurately completing datasets [8]. As a result, it helps create more reliable data analyses and supports better decision-making. In instances where a feature value is unknown, ReliefD utilizes a specific formula to accurately compute the distance. in Eq. (8).

$$d(x_n, NN\,(x_n)) = \sum_V^{\#values(n)} (P(\frac{V}{y_n})P(\frac{V}{C})) \qquad \text{Eq. (8)}$$

In this context, y_n signifies the target instance's class, while C refers to the classes of the nearest neighbor. In Eq. (8), the values for both instances are missing.

ReliefD is an effective algorithm for managing missing values in datasets, particularly in incomplete data scenarios. It performs well with noisy data and handles irrelevant features without affecting model accuracy. However, it struggles with significant outliers, which can distort feature relevance assessments and lead to misleading results. Thus, caution is advised when using ReliefD with datasets containing outliers.

To improve the original Relief algorithm and its extensions, we introduced a new version that is an iterative version of Relief. This updated method aims to solve the problems and limitations of previous versions, making the algorithm more effective and adaptable in different situations.

Proposed Method - Iterative Relief

Iterative Relief is an improved version of the original Relief algorithm, designed for picking the best features in machine learning. It examines different features and gives them weights based on how well they separate different classes of data. The algorithm looks at how relevant and important each feature is in telling one class apart from another. It assesses each feature's ability to distinguish between classes and to group similar instances within the same class. In each step, the algorithm randomly picks an instance and looks at its nearest neighbours from both the same and different classes. It updates the feature weights based on how well they can tell similar instances apart and separate those in different classes. By focusing on the most helpful features and reducing the effect of irrelevant ones, the algorithm improves its performance.

The Iterative Relief method is effective at handling irrelevant features and can be used for both binary and multiclass classification with some adjustments. However, this method requires searching for the nearest neighbors for all instances during each step, which can be slow for larger datasets, especially if they have many dimensions. Although Iterative Relief works well with noisy data, having outliers during the neighbor selection can distort the feature weights. The Iterative Relief process is a systematic approach aimed at enhancing the quality of data representation in a dataset. It consists of the following key steps:

1. Random Instance Selection: Begin by randomly selecting an instance from the dataset, which will serve as the foundation for analysis.
2. Finding Neighbours: Identify the nearest neighbour that belongs to the same class as the selected instance, referred to as the nearest hit. Similarly, locate the closest neighbour from a different class, known as the nearest miss.
3. Feature Weight Update: Analyze the differences in feature values between the selected instance and its nearest neighbours to update the feature weights. This step is critical for improving the accuracy of the model.
4. Iteration for Refinement: Repeat this process over multiple iterations, refining the feature weights with each cycle. This ongoing adjustment enhances the overall representation of the data and contributes to a more effective model.

The Iterative Relief feature weight update equation is represented as follows:

$$W[A] = W[A] - \frac{1}{m}\sum_{i=1}^{m} diff(A, R\_i, H_i) + \frac{1}{m}\sum_{i=1}^{m} diff(A, R\_i, M_i) \qquad \text{Eq. (9)}$$

Where W[A] denotes the weight of feature A, R_i indicates randomly selected target instance and H_i represents i^{th} hit and M_i specifies i^{th} miss. The function diff (A, R_i, Hi) measures the dissimilarity between R_i (the target instance) and H_i (the hit) for feature (A). The function diff (A, R_i, M_i) used to quantify the difference between R_i and M_i for feature (A). m denotes the number of samples taken into account during the iteration.

Classification

The Random Forest algorithm can be further improved by combining it with Iterative Relief, a feature selection method that significantly enhances classification accuracy compared to using Random Forest [9] alone. The Integrated Random Forest (IRF) works by utilizing the predictions made by each individual tree in the forest. These predictions are aggregated through a majority voting mechanism, which determines the final classification outcome based on the consensus of the trees.

Experimental Results

This study uses the 'Spam' dataset to evaluate various detection methods, containing 1,813 spam emails and 2,788 non-spam emails, for a total of 4,601 emails with 57 attributes. A confusion matrix assesses performance through four outcomes: True Negatives (TN), True Positives (TP), False Positives (FP), and False Negatives (FN) [11]. This allows for the calculation of precision, recall, accuracy, and F1-score, presented through tables and graphs.

Accuracy: The accuracy measures how many of the total predictions were correct.

$$\text{Accuracy} = \frac{\text{truepositive+truenegative}}{\text{truepositive+falsepositive+truenegative+falsenegative}} \quad \text{Eq. (1)}$$

Precision: The percentage of times the model correctly identifies positive cases among its positive predictions. This is found by dividing true positives by the total of true positives and false positives.

$$\text{Precision} = \frac{\text{truepositive}}{\text{truepositive+falsepositive}} \quad \text{Eq. (2)}$$

Recall: Recall is a metric that quantifies the proportion of true positive predictions in relation to all actual positive instances. It is determined by dividing the number of true positives by the sum of true positives and false negatives. This measurement is essential for evaluating the performance of predictive models in various applications [10].

$$\text{Recall} = \frac{\text{truepositive}}{\text{truepositive+falsenegative}} \quad \text{Eq. (3)}$$

F1-score: A metric utilized to evaluate a model's accuracy by integrating both precision and recall. This metric is defined as the harmonic mean of precision and recall, providing a comprehensive assessment of the model's performance.

$$\text{F1} = \frac{2*\text{precision}*\text{recall}}{\text{precision+recall}} \quad \text{Eq. (4)}$$

Mean Absolute Error (MAE): The average of the absolute values of the errors, which represent the differences between predicted and actual values, provides a measure of accuracy in the predictions.

$$\text{MeanAbsoluteError(MAE)} = \frac{1}{N}\sum_{i=1}^{N}|\mathbf{y_i} - \hat{y}_i| \quad \text{Eq. (5)}$$

Mean Squared Error (MSE): The average of the squared differences between the actual values and the predicted values in a given data set serves as a critical measure of the variance of the residuals.

$$\text{MeanSquaredError(MSE)} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{y_i} - \hat{y}_i)^2 \quad \text{Eq. (6)}$$

Root Mean Squared Error (RMSE): The square root of the Mean Squared Error (MSE) is a valuable metric that quantifies the standard deviation of the residuals. This metric provides important insights into the accuracy of the model's predictions.

$$\text{RootMeanSquaredError(RMSE)} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{y_i} - \hat{y}_i)^2} \quad \text{Eq. (7)}$$

t-test: The means of two groups were compared by t-test. The below formula is used for the t-test.

$$\text{t} - \text{test} = \frac{\text{mean difference}}{\text{Standard error}} \quad \text{Eq. (8)}$$

f-test: The F-test, commonly referred to as the F-ratio, is a statistical method used to compare the variances of two or more groups.

$$\text{F-test} = \frac{\text{variance between sample mean}}{\text{variance expected by error}} \quad \text{Eq. (9)}$$

p-test: The p-value is a statistical measure that quantifies the strength of evidence against the null hypothesis. A smaller p-value suggests a more substantial degree of evidence against the null hypothesis.

Table 1. Accuracy

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	91.77489	90.47619	92.64069	80.08658	76.62337	93.20779
0.1	91.32321	91.32321	91.32321	83.08026	79.82646	93.75704
0.15	90.88277	91.31693	91.75108	82.34442	79.88422	93.8958
0.2	90.33659	91.85667	91.74809	82.95331	81.10749	93.39956
0.25	91.48566	91.7463	92.61511	82.36316	80.97306	93.35447
0.3	91.45546	91.60028	92.03475	82.40405	80.44895	93.46922
0.35	90.99937	91.99255	92.42706	83.24022	80.63314	93.99255
0.4	90.87452	92.2868	92.2868	82.12927	81.04291	93.39543

Table 2. Precision

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	93.45454	94.29411	94.5909	78.49462	77.10843	95.50561
0.1	92.79768	92.78531	92.78531	83.05084	78.33333	94.85714
0.15	93.81818	93.88489	94.26523	80.53691	78.54671	94.60431
0.2	92.87749	92.45283	93.61111	81.23324	78.42105	94.19889
0.25	92.93849	92.22222	93.23333	79.05982	76.77824	93.48859
0.3	93.38374	92.13893	92.53187	80.47445	77.36185	92.76672
0.35	92.88079	90.5775	92.46467	80.75117	77.0416	92.24683
0.4	92.07492	91.44021	92.36111	79.17241	77.11978	92.97752

Table 3. Recall

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	84.84848	81.81818	85.85858	73.73737	64.64646	85.85858
0.1	84.10256	85.1282	85.1282	75.38461	72.30769	85.1282
0.15	84.86842	85.85526	86.51315	78.94736	74.67105	86.51315
0.2	83.58974	87.94871	86.41025	77.6923	76.41025	87.43589
0.25	85.89473	87.36842	88.42105	77.89473	77.26315	87.78947
0.3	85.61525	87.34835	88.04159	76.4298	75.21663	88.90814
0.35	84.61538	89.89441	88.83861	77.82805	75.41478	87.93363
0.4	84.9734	89.49468	88.43085	76.32978	76.1968	88.03191

Table 4. F1-score

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	89.83957	88.04347	90.90909	76.04166	70.32967	90.42553
0.1	89.13043	89.24731	89.24731	79.03225	75.2	89.72972
0.15	89.11917	89.69072	90.22298	79.73421	76.55986	90.378
0.2	87.9892	90.14454	89.86666	79.42332	77.40259	90.69148
0.25	89.27789	89.72972	90.81081	78.47295	77.01993	90.45553
0.3	89.33092	89.67971	90.2309	78.4	76.27416	90.79646
0.35	88.55564	90.23467	90.61538	79.26267	76.21951	90.03861
0.4	88.38174	90.45698	90.35326	77.72511	76.65551	90.43715

Table 5. Mean Absolute Error

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	0.082	0.095	0.073	0.199	0.233	0.077
0.1	0.086	0.086	0.086	0.169	0.201	0.082
0.15	0.091	0.086	0.082	0.176	0.201	0.081
0.2	0.096	0.081	0.082	0.17	0.188	0.076
0.25	0.085	0.082	0.073	0.176	0.19	0.076
0.3	0.085	0.083	0.079	0.175	0.195	0.075
0.35	0.09	0.08	0.075	0.167	0.193	0.08
0.4	0.091	0.077	0.077	0.178	0.189	0.076

Table 6. Mean Squared Error

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	0.082	0.095	0.073	0.199	0.233	0.077
0.1	0.086	0.086	0.086	0.169	0.201	0.082
0.15	0.091	0.086	0.082	0.176	0.201	0.081
0.2	0.096	0.081	0.082	0.17	0.188	0.076
0.25	0.085	0.082	0.073	0.176	0.19	0.076
0.3	0.085	0.083	0.079	0.175	0.195	0.075
0.35	0.09	0.08	0.075	0.167	0.193	0.08
0.4	0.091	0.077	0.077	0.178	0.189	0.076

Table 7: Root Mean Squared Error

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	0.286	0.308	0.271	0.446	0.483	0.279
0.1	0.294	0.294	0.294	0.411	0.449	0.287
0.15	0.301	0.294	0.287	0.42	0.448	0.284
0.2	0.31	0.285	0.287	0.412	0.434	0.275
0.25	0.291	0.287	0.271	0.419	0.436	0.276
0.3	0.292	0.289	0.282	0.419	0.442	0.274
0.35	0.3	0.282	0.275	0.409	0.44	0.282
0.4	0.302	0.277	0.277	0.422	0.435	0.275

Table 8. t-test values

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	1.041	1.33	1.04	0.565	1.52	0.945
0.1	1.479	1.207	1.2	1.207	1.005	1.343
0.15	1.581	1.416	1.36	0.325	0.814	1.416
0.2	1.853	0.898	1.42	0.803	0.472	1.327
0.25	1.533	1.062	1.06	0.296	0.126	1.19
0.3	1.864	1.161	1.08	1.122	0.618	0.928
0.35	2.128	0.179	0.93	0.861	0.501	1.113
0.4	1.957	0.53	1.07	0.907	0.301	1.346

Table 9. p-test values

Test size	Relief	ReliefA	ReliefB	ReliefC	ReliefD	Iterative Relief
0.05	0.298	0.184	0.29	0.572	0.12	0.344
0.1	0.139	0.227	0.22	0.227	0.315	0.179
0.15	0.114	0.156	0.17	0.745	0.415	0.156
0.2	0.063	0.368	0.15	0.421	0.636	0.184
0.25	0.125	0.288	0.28	0.766	0.899	0.233
0.3	0.062	0.245	0.27	0.261	0.536	0.353
0.35	0.033	0.857	0.35	0.389	0.615	0.265
0.4	0.05	0.59	0.21	0.364	0.762	0.178

Table 10. f-test values

Test size	**Relief**	**ReliefA**	**ReliefB**	**ReliefC**	**ReliefD**	**Iterative Relief**
0.05	0.962	0.949	0.96	0.982	0.94	0.967
0.1	0.96	0.969	0.96	0.969	0.975	0.964
0.15	0.972	0.975	0.97	0.995	0.987	0.975
0.2	0.966	0.985	0.97	0.987	0.992	0.977
0.25	0.973	0.982	0.98	0.995	1.001	0.98
0.3	0.971	0.983	0.98	0.983	0.991	0.987
0.35	0.967	0.997	0.98	0.988	0.993	0.984
0.4	0.972	0.99	0.985	0.987	0.996	0.981

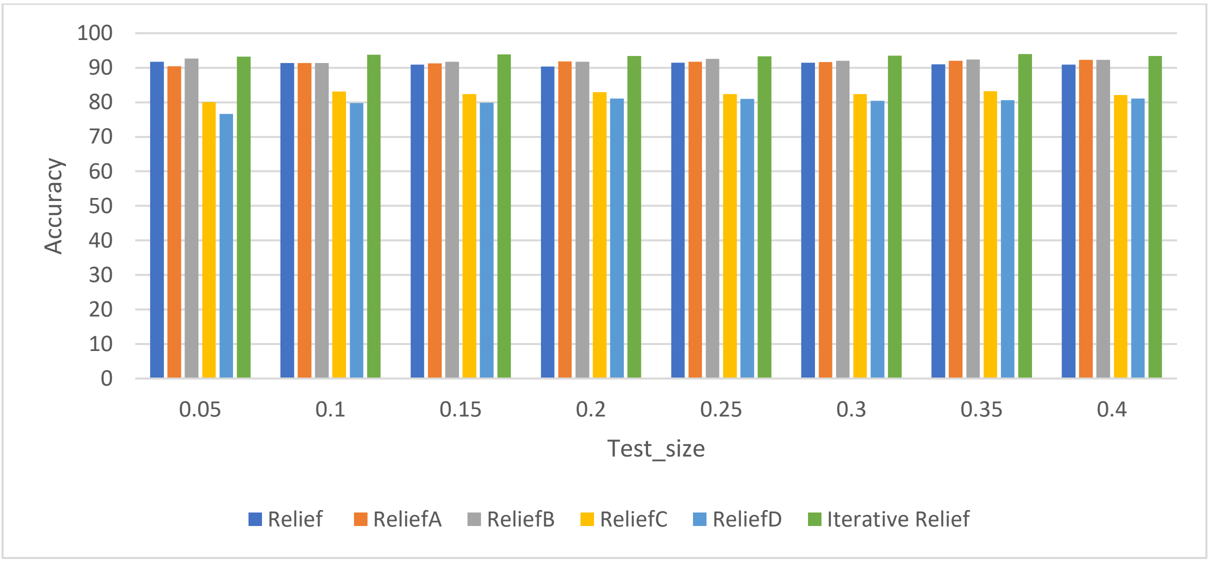

Fig 1. Test_size Vs. Accuracy

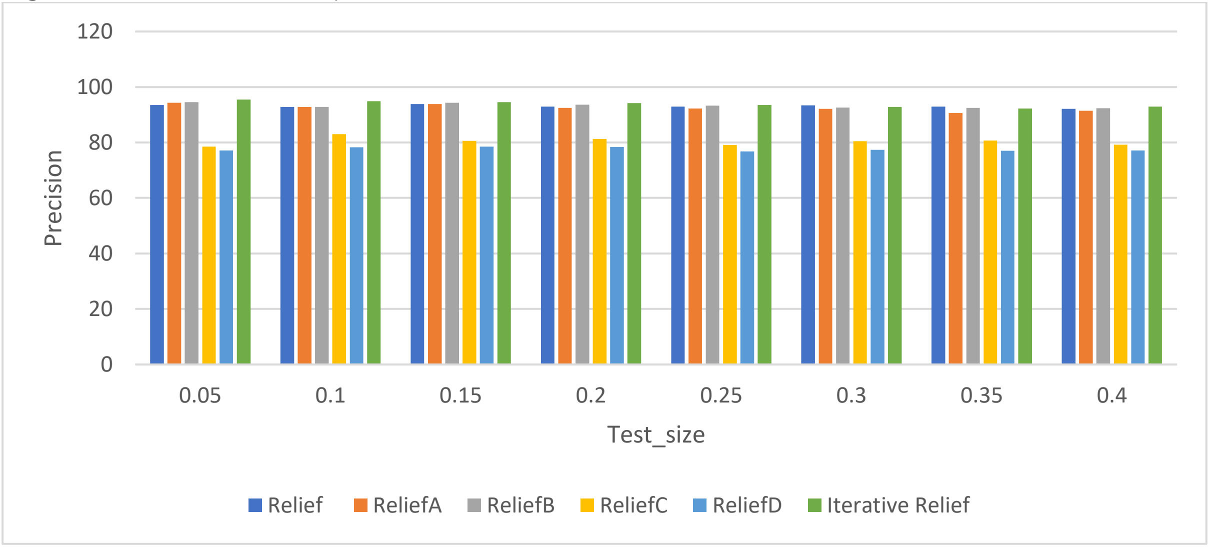

Fig 2. Test_size Vs. Precision

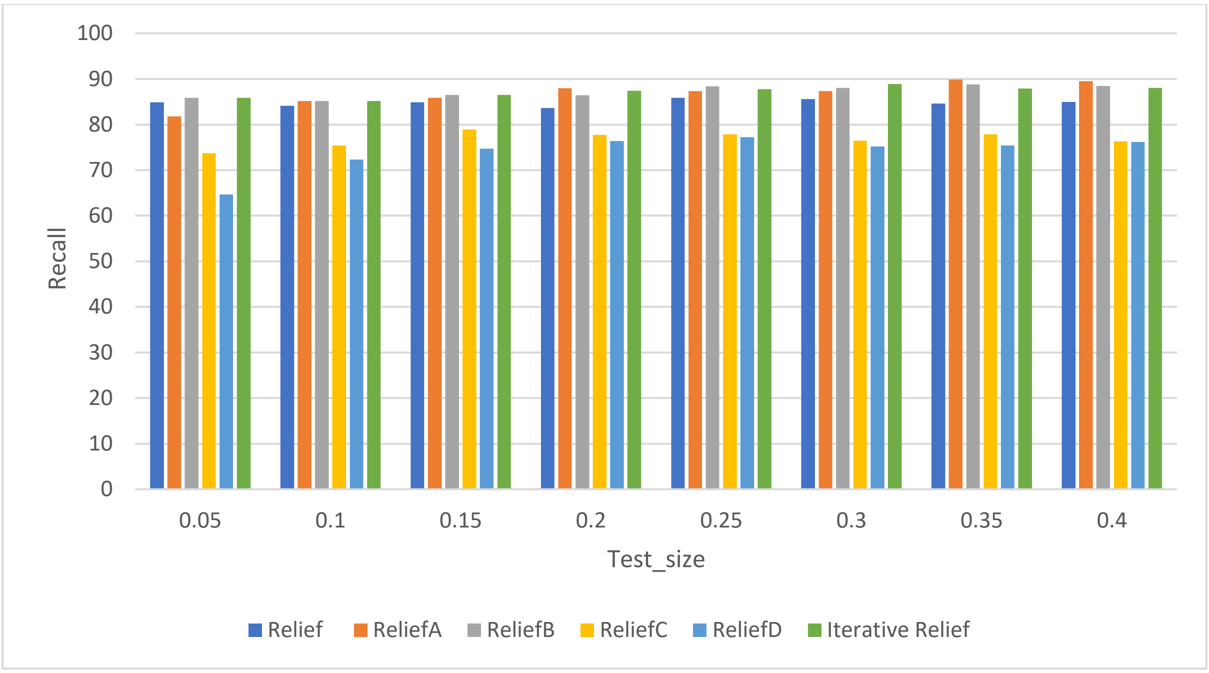

Fig 3. Test_size Vs. Recall

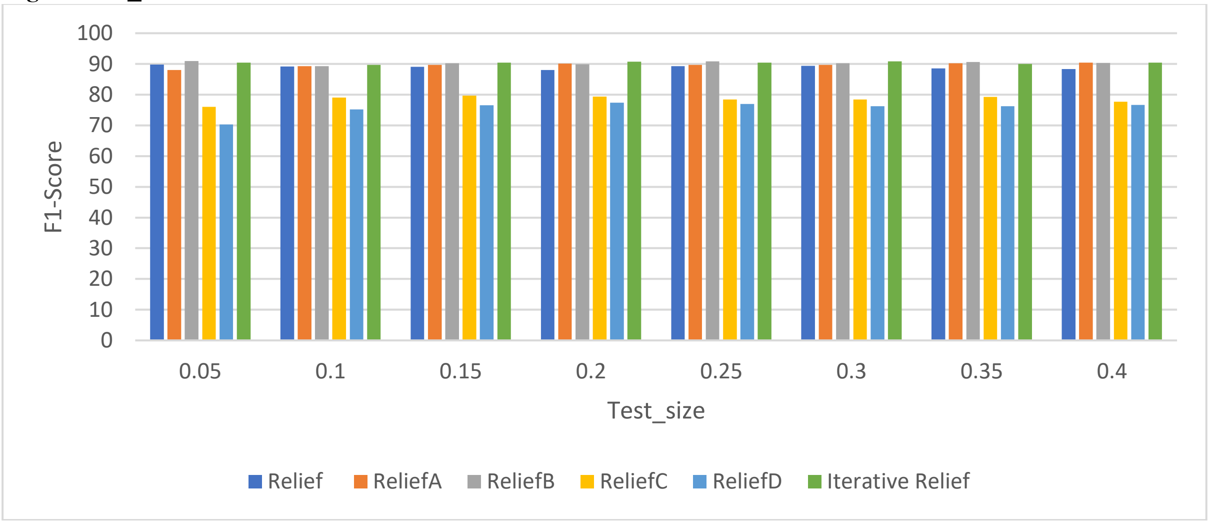

Fig 4. Test_size Vs. F1-score

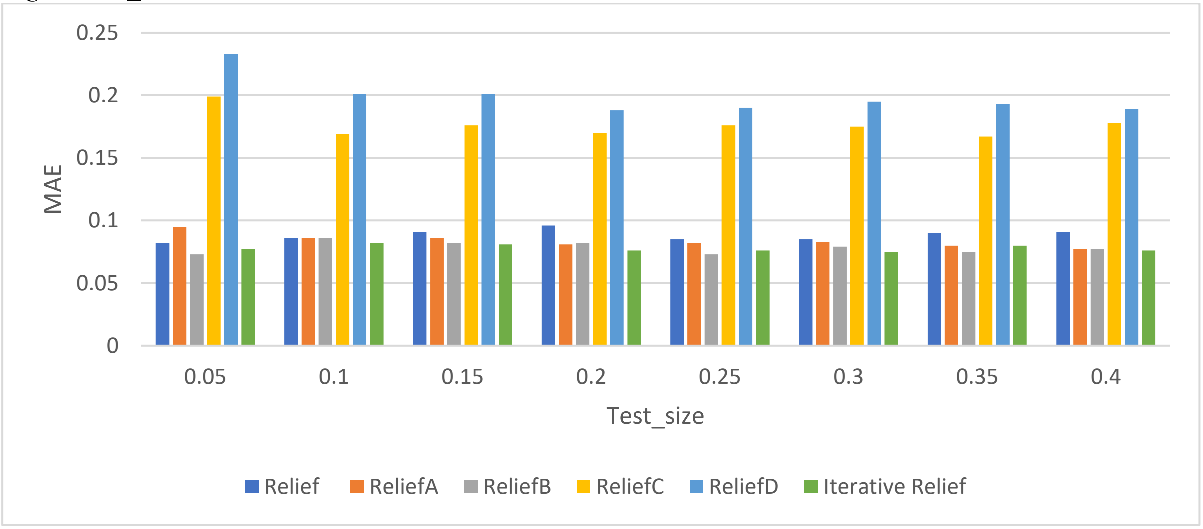

Fig 5. Test_size Vs. MAE values

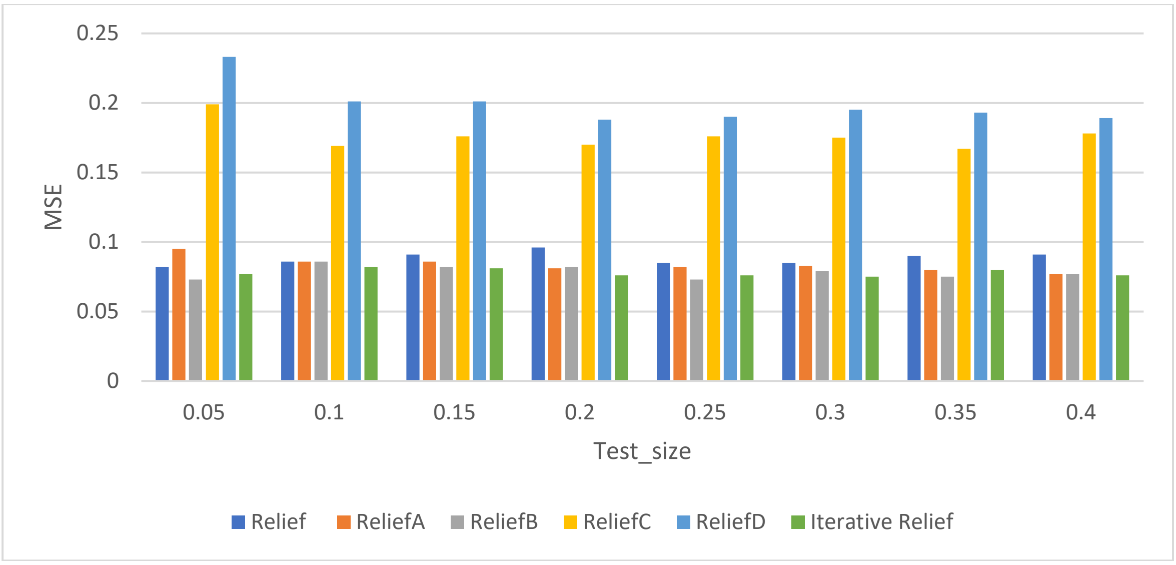

Fig 6. Test_size Vs. MSE values

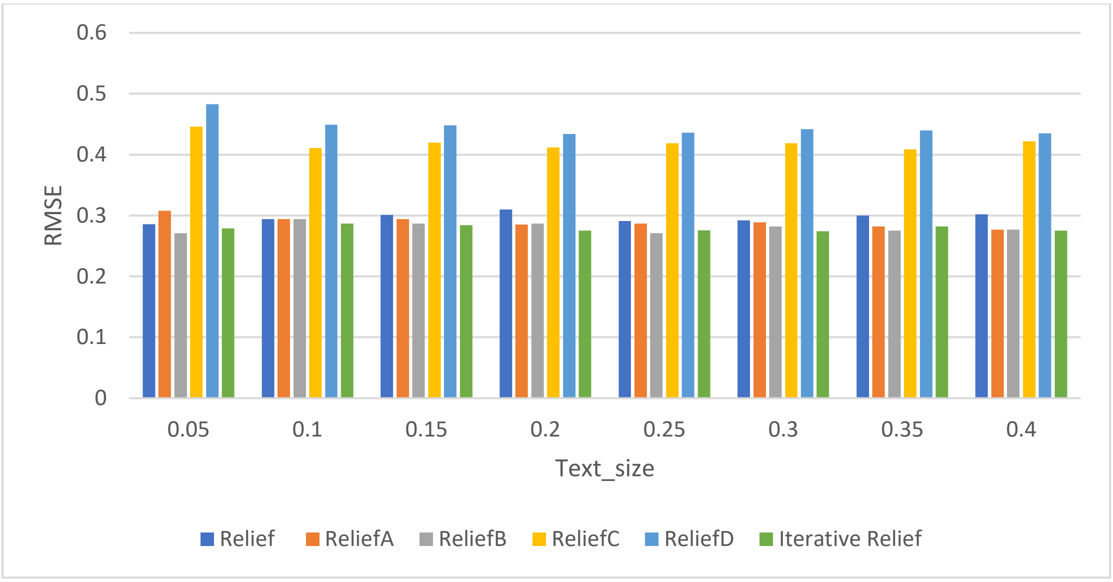

Fig 7. Test_size Vs. RMSE values

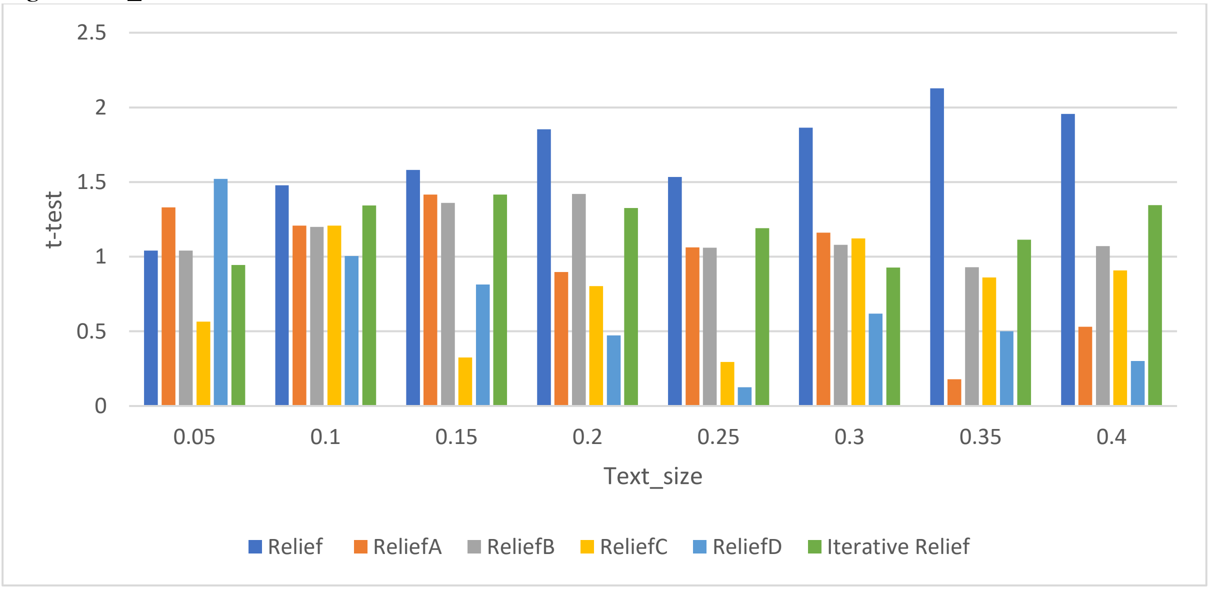

Fig 8. Test_size Vs. t-test values

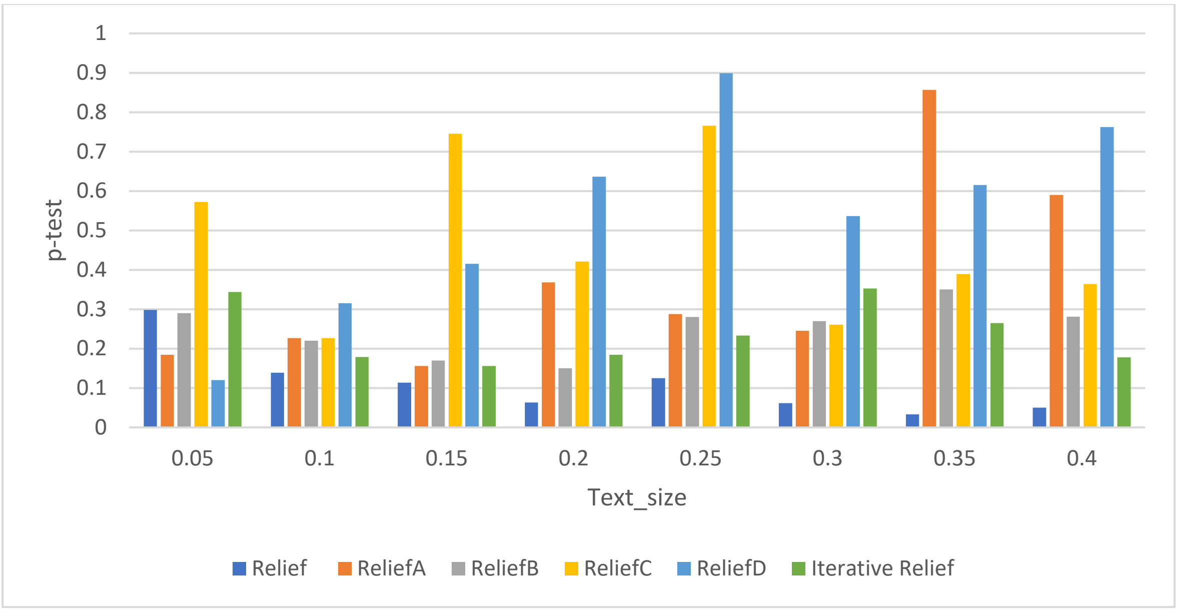

Fig 9. Test_size Vs. p-test values

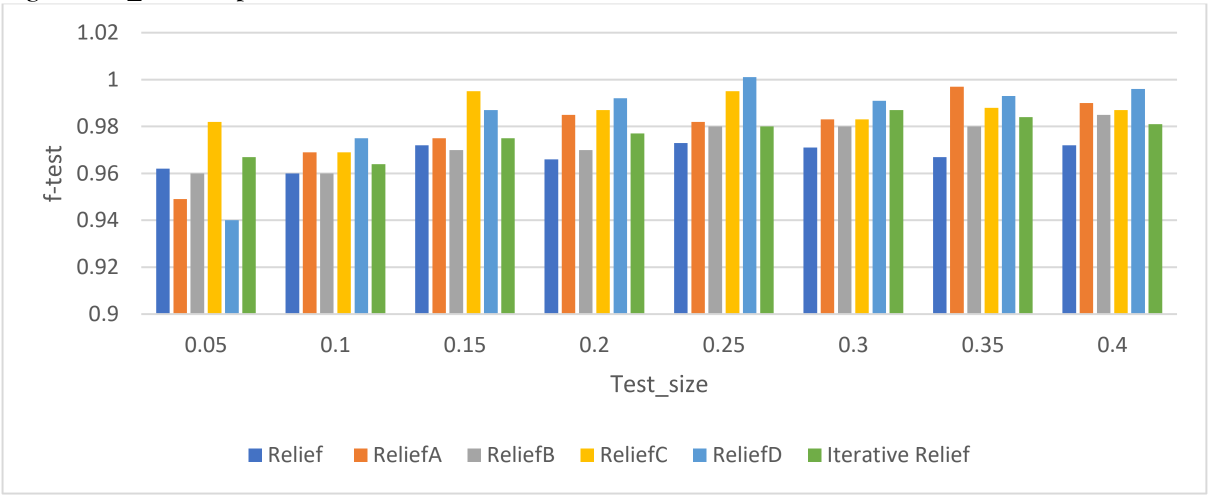

Fig 10. Test_size Vs. f-test values

The relationship between test_size and accuracy is detailed in Table 1 and Figure 1. A high level of accuracy was attained with a test_size of 0.35. Furthermore, Tables 1 through 10 and Figures 1 through 10 illustrate the correlation between accuracy and other statistical factors.

CONCLUSIONS

This study explored several relief-based feature selection methods, including Relief, ReliefA, ReliefB, ReliefC, and ReliefD. The algorithms have made improvements to address issues like noise sensitivity and missing data. We introduced the Iterative Relief method, an enhancement of the traditional Relief framework that refines feature weights through multiple iterations. This approach aims to provide more accurate assessments of feature importance, leading to improved performance in predictive modelling. The Iterative Relief method enhances the robustness and accuracy of feature rankings, especially in complex datasets. Our analysis indicates that it is computationally efficient and effective at identifying relevant features, even in noisy data situations, a common challenge in data analysis.

REFERENCES

1. Lamiaa M. EI Bakrawy, Hybrid Particle Swarm Optimization and Pegasos Algorithm for Spam Email Detection, Advances in Systems Science and Applications, Vol. 3, 2019, PP: 11-22.
2. Jai Batra, et.al., A comprehensive study of spam detection in e-mails using bio-inspired optimization techniques, International Journal of Information Management Data Insights, 2021, PP: 1-13.
3. Yijun Sun, Sinisa Todorovic, and Steve Goodison. Local-learning-based feature selection for high-dimensional data analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 32, Issue-9, 2010, PP:1610–1626.
4. K. Kira, L.A. Rendell, The feature selection problem: traditional methods and a new algorithm, in: AAAI, Vol. 2, 1992a, PP: 129–134.

5. K. Kira, L.A. Rendell, A practical approach to feature selection, in: Proceedings of the Ninth International Workshop on Machine Learning, 1992b, PP: 249–256.
6. Ryan J. Urbanowicz, et.al., Relief-based feature selection: Introduction and review, Journal of Biomedical Informatics, Vol. 85, 2018, PP: 189-203.
7. Suravi Akhter, et.al., A Relief Based Feature Subset Selection Method, Dhaka University Journal of Applied Science and Engineering, Vol. 6, Issue-2, 2021, PP: 7-13.
8. Suravi Akhter, et.al., A Relief Based Feature Subset Selection Method, Dhaka University Journal of Applied Science and Engineering, Vol. 6, Issue-2, 2021, PP: 7-13.
9. B. Aruna Kumari, et.al., Robust Machine Learning Technique for Detection and Classification of Spam Mails, Novyi MIR Research Journal, Vol. 8, Issue-7, 2023, PP:196-204.
10. C. Nagaraju, et.al., MultiSURF: Optimal Feature Selection Technique for Spam Mail Detection and Classification, Nanotechnology Perceptions, Vol. 20, No. S8, 2024, PP: 452-461.
11. B. Aruna Kumari, C. Nagaraju, A Generalized Two-Level Ensemble Method for Spam Mail Detection, Journal of Electrical Systems, Vol. 20, No. 2, 2024, PP: 1570-1579.