DUJASE Vol. 6 (2) 7-13, 2021 (July) https://doi.org/10.3329/dujase.v6i2.59214

A Relief Based Feature Subset Selection Method

Suravi Akhter,* Sumon Ahmed, Ahmedul Kabir and Mohammad Shoyaib

Institute of Information Technology Dhaka, Bangladesh

**E-mail: bsse0827@iit.du.ac.bd*

Received on 21 March 2021, Accepted for publication on 25 October 2021

ABSTRACT

Feature selection methods are used as a preliminary step in different areas of machine learning. Feature selection usually involves ranking the features or extracting a subset of features from the original dataset. Among various types of feature selection methods, distance-based methods are popular for their simplicity and better accuracy. Moreover, they can capture the interaction among the features for a particular application. However, it is difficult to decide the appropriate feature subset for better accuracy from the ranked feature set. To solve this problem, in this paper we propose Relief based Feature Subset Selection (RFSS), a method to capture more interactive and relevant feature subset for obtaining better accuracy. Experimental result on 16 benchmark datasets demonstrates that the proposed method performs better in comparison to the state-of-the-art methods.

Keywords: Relief, Feature Subset Selection, Relief based Feature Subset Selection (RFSS), Greedy Forward search

1. Introduction

Feature selection is necessary to extract important information from different areas including medical, cyber security, image processing, bioinformatics, agriculture and natural language processing. For example, to identify the nurse care activity from different sensor data which helps to identify the activity and performance evaluation of nurse [1]. The data produced in these fields often contain noisy and irrelevant features. Feature selection methods extract informative and relevant features [2, 3, 4] from a set of features to correctly predict the output which reduces the number of dimensions of the original data [5, 6].

Feature selection methods can be broadly categorized into two types: feature ranking methods and feature subset selection methods. In the ranking method, features are ranked based on some criteria that weights the importance of the features in an ordered way. The most important features are in the top ranked positions of the ranked feature list and the irrelevant features are in lower ranks as they are less important for the output prediction. From the ranked list p features are chosen as selected subset S. Popular feature ranking methods include minimal redundancy maximal relevance (MRMR) [7], joint mutual information (JMI) [8], relax MRMR [9], Relief F [10], MAPrelief [11], SURF [12] and MultiSURF [13], etc. In feature subset selection, the main goal is to find an optimal subset of features which gives better accuracy for a particular application. It can be classified into three categories: embedded, wrapper and filter [14]. Embedded methods are dependent on a classifier and implemented by optimizing an objective function during the training of the classifier [15]. The wrapper methods are also classifier-dependent, but their computational costs are relatively high. Filter methods, the most popular among these three, are classfier-independent and are built on the intrinsic properties of the features [16]. Our proposed method uses the filter approach. mDSM [17], DSbM [18], IGIS+ [19], BIRS [20] and DSM [21] are some of the feature subset selection methods. Various search mechanisms are used to find the optimal feature subset. Among them, forward search and backward elimination [22] are widely used. Forward search decides to add a feature to the list when the addition improves the value of a certain criterion. Backward elimination starts off with all the features, and then eliminates them one by one based on some criteria. At some point the elimination stops, and the remaining features are the selected subset of features. In this work, we use greedy forward search. For the feature selection criteria to select the features, there are various approaches for feature selection, such as Correlation [23, 24], Mutual Information (MI) [7, 17, 25, 26], distance-based [10, 11, 12, 27, 28, 29] methods. At the very beginning of feature selection method, correlation coefficient was used, but it can't remove the similar type of features and relation among features for the classification. Distance-based methods are easier to use and take less time to run than the other methods.

Relief [27] is the first algorithm of distance-based method where the feature is weighted according to the class separability of a feature in terms of a target instance. From the target instance, the nearest sameclass data instances are called 'hits' and different-class instances are called 'misses'. Relief takes 1 nearest neighbor (NN) from the target instance and measures how much one class data differs from another by taking the distance difference value from miss to hit. In Relief, the feature weight value is between -1 to 1. Negative weight means the feature creates noise and it is irrelevant to output prediction. More positive weight means more relevant information about the output is provided by the feature. Relief works in binary classification problems and it uses only one nearest neighbor in distance calculation. But taking a decision using only one NN creates noise in decision making in the existence of noisy features in a dataset [10]. ReliefA [10] introduced k nearest neighbour to handle the noisy data. However, in the case of missing data in a dataset, neither Relief nor ReliefA could weight the features. The next versions of Relief(BD) [10] are able to complete the missing data. Among them, ReliefD is best as it uses probabilistic estimation to fill the incomplete data. But the

above variation of the Relief algorithm can not handle the multiclass data. The next versions of Relief (E-F) method are designed to transform binary class solution to a multiclass problem solution. ReliefF [10] uses k (user-defined parameter) nearest neighbours for feature weighting. To fulfill the condition of k ReliefF needs to take instances that are far from the target instance. As it is a problem to fix the value of k, Spatially Uniform ReliefF (SURF) [30] uses a distance threshold value for selecting the next k nearest hit and miss instances for the target instance. To capture better interaction than the SURF method, SURF* [31] takes both the near and far instances from the target instance in feature weighting to detect two-way interactions. Here near instances are the ones whose distances are less than the distance threshold value, and the other ones are far instances. But as all instances are used in weighting, computation time is high in SURF*. MultiSURF* [32] discards the boundary region instances with a standard deviation ($\pm\sigma$) along with threshold distance and takes only the outside and inside instances. By discarding the instances MultiSURF* reduces the computation time of SURF*. However, when far weights are calculated then the original feature weight gets affected. In MultiSURF [13], the far weighting is avoided and near weighting is calculated for feature weighting. The methods discussed so far assume that the datasets are balanced (equal distribution of classes in the dataset). In the case of highly imbalanced data, these methods can output an erronous ranking of the features. To solve this problem, MAPrelief [11] was designed to handle imbalance and multiclass data. There still remains a problem of outliers in the dataset, which can mislead the weighting mechanism of the above methods. To handle outlier data, Iterative relief [29] is introduced. Apart from the above-discussed methods, there is another approach for feature weighting named EBFS [28] which calculates neighbors like Relief but rather than calculating distance among hit and miss it measures how much entropy is between the hit-miss feature value and the target feature value. However, Relief is feature subset selection method and the later version of Relief based methods give weights and serialize the features based on their performance [33].

Most of the existing feature selection methods are ranking methods, so it is difficult to decide up to which number of features should be taken for better classification. To get an optimal subset, a feature selection method is necessary. We modify the original ReliefF algorithm in combined weight calculation and propose a new feature selection method RFSS where the feature is selected when the combined feature set gives more margin difference among classes. Our proposed method is effective compared to the existing method because we take the features that increase the class separability of the data instances otherwise, we discard. As existing method are ranking method and combined feature performance is not measured, a feature will be in the ranked set where we remove the feature. In this work we experimented by applying this method only on the ReliefF algorithm. But it can also be applied in all the variations of the Relief algorithms discussed above.

Literature Review

The Relief algorithm considers feature dependency when it searches for the nearest hit and miss neighbour from the target instance. The same class neighbour from the target instance is called the hit instance and a different class is called the miss instance. For an attribute fc, Relief is an approximation of Eq.(1) two probability difference.

$$w[f_c] = P(\text{same value of } f_c | \text{hit class}) - P(\text{different value of } f_c | \text{miss class}) \quad (1)$$

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

$$w[f_c] = \sum_{n=1}^{N} \frac{d\left(x_n, miss(x_n)\right) - d\left(x_n, hit(x_n)\right)}{N} \quad (2)$$

Here, N is the number of instances, xn is the target instance, d calculates the distance of hit and miss from the target instance. Weight($w[fc]$) of a feature is always between 0 and 1. Distance calculation in Eq.(2) is different for numerical and categorical data. For categorical data there is Eq.(3),

$$d\left(x_n, NN(x_n)\right) = \begin{cases} 0 & same\ value\ x_n\ and\ NN(x_n) \\ 1 & otherwise \end{cases} \quad (3)$$

For numerical data difference is calculated as Eq.(4) gives the value between 0 and 1.

$$d\left(x_n, NN(x_n)\right) = |value(x_n) - value(NN(x_n))| \quad (4)$$

Relief(B-D) [10] develops several techniques to handle the missing data. In ReliefB [10], single instance value is missing then the distance is calculated using Eq.(5)

$$d\left(x_n, NN(x_n)\right) = 1 - \frac{1}{\#\ of\ values\ in\ f_c} \quad (5)$$

ReliefC ignores the missing data distance estimation. ReliefD performs better over ReliefB and ReliefC for the probabilistic estimation to calculate missing data. When one instance feature value is unknown ReliefD calculates the distance using the following form in Eq.(6)

$$d\left(x_n, NN(x_n)\right) = 1 - P(value(x_n, NN(x_n)) | y_n) \quad (6)$$

where y_n is the target instance class and C is the miss (different) classes of the nearest neighbour instance. In Eq.(7), both instance value is missing.

$$d\left(x_n, NN(x_n)\right) = \sum_{V}^{\#values(n)} \left(P(V|y_n)P(V|C)\right) \quad (7)$$

ReliefE and ReliefF is the solution to the binary class problem of Relief. ReliefE is the straight forward implementation of Relief. But in ReliefF, k-nearest miss is calculated for all different(miss) class. ReliefF optimizes the following Eq.(8)

$$w[f_c] = w[f_c] + \frac{1}{1 - P(y_n)} \sum_{C \neq y_n} \frac{[P(C)d(hit(x_n^{f_c}), miss((x_n^{f_c}))]}{N} \quad (8)$$

ReliefF [10] choose fixed k for the nearest hit and miss but sometimes the nearest neighbour may be so far from the target instance that it is the barrier for actual approximation. Spatially Uniform Relief (SURF) [12] takes the instances inside the threshold distance. The threshold distance is calculated by averaging all the instance pairwise distance. SURF* [31] takes the outside boundary instances to capture two-way interaction using far weighting. SURF* takes a long time for execution. MultiSURF* [32] eliminates the instances around the threshold boundary($\pm\sigma$) and it takes less time than SURF*. But the original goal of class separability cannot be achieved due to the more impact of far weighting. MultiSURF [13] calculates only the near weight of MultiSURF*.

The above-mentioned method is applicable in the balance dataset scenario. In MAPRelief [11], imbalanced and multiclass data is handled using the class prior probability. It takes *k*-nearest neighbour from the rest miss class without taking k-nearest neighbour from the different class respectively. MAPRelief optimizes the Eq.(9) to solve the class imbalance problem.

$$w[f_c] = -(1 - P(y_n))|x_n - hit(x_n)| + P(y_n)|x_n - miss(x_n)| \quad (9)$$

There is another kind of feature selection method namely Entropy Based Feature Selection (EBFS) which uses Eq.(10) entropy in feature weighting.

$$w[f_c] = H\left(val(x_n, miss(x_n))\right) - H(val(x_n, hit(x_n))) \quad (10)$$

When the nearest feature miss value is 1*;* 1*;* 1*;* 1 and hit value 0*;* 0*;* 0*;* 0 the feature separates nearest hits and misses perfectly, miss and hit in Eq.(10) provide same entropy value. To overcome the problem class label is also used when calculating feature entropy in Eq.(11).

$$w[f_c] = H(val(x_n, miss(x_n)), y_n) - H(val(x_n, hit(x_n)), y_n) \quad (11)$$

But all the above-discussed methods are ranking method and can not decide up to which number of feature we should take to achieve good classification. To solve this issue we propose a greedy forward selection algorithm (RFSS) which selects the feature-based provided new classification information of the candidate features with the selected subset.

2. Proposed Method

In this work, we modify the original ReliefF method to work for feature subset selection rather than only for feature ranking. In our proposed method, initial feature weights are calculated similar to ReliefF. But our contribution is that we propose a weighting mechanism for combined features. By maximizing the combined feature subset weight, we develop a forward feature subset selection method named RFSS. Here, the features are sequentially added to the subset, and when the new combined feature subset performs better than the previous feature subset, we conclude that the new candidate feature provides additional information with the selected subset about output prediction. In this section, we describe how we rank the features and what criteria we use to construct the optimal feature subset.

2.1 Ranking the features

Individual features are weighted by the output separation capability of a feature. We can represent the values of individual features by a straight line in Fig.1. Here, $x_n^{f_c}$ refers to the target instance x_n for a candidate feature f_c, while $hit(x_n^{f_c})$ and $miss(x_n^{f_c})$ refer to the same-class and different-class nearest neighbors respectively. For an individual feature, target $x_n^{f_c}$, $hit(x_n^{f_c})$ and $miss(x_n^{f_c})$ remain in linear space.

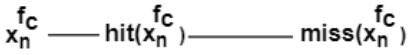

Fig. 1. Individual feature weighting

In Eq.12, hit miss instance distance difference from the target instance is equal to the distance between hits and misses. So it is clear that, for each instance it measures how much the hit and miss instances are far from each other.

$$w[f_c] = w[f_c] - \left|x_n^{f_c} - hit\left(x_n^{f_c}\right)\right| + \left|x_n^{f_c} - miss\left(x_n^{f_c}\right)\right| = w[f_c] + \left|hit\left(x_n^{f_c}\right) - miss\left(x_n^{f_c}\right)\right| \quad \textbf{(12)}$$

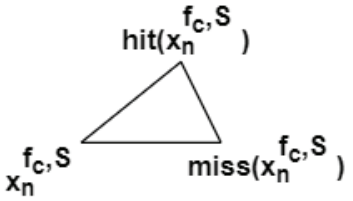

Fig. 2. Combined feature weight=

We calculate individual feature weights using Eq.13 and sort the features in descending order based on the feature weight.

$$w[f_c] = w[f_c] + \frac{1}{1 - P(y_n)} \sum_{C \neq y_n} \frac{[P(C)d(hit(x_n^{f_c}), miss((x_n^{f_c}))]}{N} \quad (13)$$

From w[fc], we get the feature weight value [−1; 1] where more positive weight means the feature is more relevant to the classes otherwise the feature is irrelevant.

2.2 Calculating the weight of combined features

Traditional ReliefF algorithm ranks the features individually, but does not have a process of weighting combined features. Our proposed method is a subset selection method where we incrementally add features to the subset in the descending order of ranking as calculated in the previous subsection. To decide whether to add a new candidate feature to the subset, a weighting criteria for the combined feature subset is required. A major contribution of our work is that we propose a weighting mechanism for combined features. When a candidate feature improves the performance, then the weight of the combined feature set is increased.

Fig. 2 is used to explain what happens to the hit-miss calculation when a set of features are used rather than one individual feature. In this figure, $x_n^{fc,S}$ refers to the target instance x_n when the candidate feature fc is combined with a feature subset S. hit($x_n^{fc,S}$) and miss ($x_n^{fc,S}$) correspond to the same-class and different class nearest neighbors as stated before. However, in contrast to the individual feature case, $x_n^{fc,S}$, hit ($x_n^{fc,S}$) and miss ($x_n^{fc,S}$) will most likely not be in linear space when features are combined. These points may now form a triangle like that shown in Fig. 2. In this scenario, traditional ReliefF weight calculation will measure the difference of hit and misses respectively from the target instance, and then subtract the results to get the weight. That is, traditional ReliefF takes the difference of the sides $x_n^{fc,S}$, hit ($x_n^{fc,S}$) and miss ($x_n^{fc,S}$) from the triangle of Fig. 2. However, this could give a very small value even if hit ($x_n^{fc,S}$) and miss ($x_n^{fc,S}$) are very far apart in the non-linear space. In our proposed method, we use the distance between hit and miss considering the dimension > 1, i.e., we take the length of the hit ($x_n^{fc,S}$) - miss ($x_n^{fc,S}$) side of the triangle as our weight for the combined feature set. This gives a much better estimate of the performance of the combined set of features.

The combined weight is calculated using Eq.14 to capture the actual difference of the predicted output.

$$w(fc,S) = w[fc,S] + 11 - P(yn)\,X\,C6 = yn\,[P(C)d(hit(x\,fc,S\,n),miss(x\,fc,S\,n))]\,N$$

$$w(fc,S) = w[fc,S] + \frac{1}{1-P(y_n)}$$

$$\sum_{C \neq y_n} \frac{[P(C)d(hit(x_n^{fc,S}),miss((x_n^{fc,S}))]}{N} \quad (14)$$

2.3 Constructing the feature subset

After getting the feature rank using Eq.8 we select the top feature from the ranked feature set and insert it into the selected subset S. Then the remaining features are taken sequentially one by one as candidate features. Combined weight is calculated by appending a candidate feature (f_c) to the selected subset S using Eq.14. When the candidate feature in combination with the selected subset improves the weight than before, it means the candidate feature should be added to the subset to achieve better accuracy. This greedy forward weight maximization technique is followed to capture the appropriate feature subset. The overall of view of how our proposed method works is shown in Algorithm 1.

2.4 Illustrative Example

To understand the overall method, let's take a look on the following example. In Table 1, we take 10 instances of two class data. We calculate the rank of feature using Eq.(8) for ReliefF which gives,

Algorithm 1: ReliefF Selection

Input: Dataset (D): instances, X ={x_1,x_2,x_3, ...x_n} and features, F={f_1,f_2,f_3, ...f_m}
Parameter: Number of neighbour (k) and threshold (T)
Output: Subset of features, S ⊆ F

```
Calculate the feature weight using Eq.8
Sort features(F) based on their weight in descending order.
Select f1 from the sorted feature rank.
S ← f1; Sc ← Sc \ f1;
for all i = 2: (m − |S|) do
Calculate weight w(fc, S) using the following Eq.14
if (w(fc, S) − w(S)) > T then
S = S ∪ fc
end if
m = m-1
end for
return S
```

Table 1. Example Dataset

F_1	1	2	1	1	-2	2	1	1	2	2
F_2	1	1	0	2	1	5	-2	3	0	5
F_3	1	4	2	1	0	0	1	7	6	6
F_4	2	1	2	2	8	4	1	-2	8	7
F_5	1	3	1	1	2	2	1	2	3	2
Class	+	-	+	+	+	-	+	+	-	-

<F_5, F_4, F_3, F_2, F_1 > rank. For our selection method, we rank the feature using Eq. 13. Then our selection process maximizes the Eq. 14 and the final selected feature subset is < F_5, F_4, F_2 >. We have measured the performance using SVM classifier where two test instances is used. Our method can classify both instances where Relief failed for one instance. Hence, our method is better than the Relief ranking method. Our method will also perform better over the existing ReliefF, SURF*, SURF, and MultiSURF as these methods are ranking method and selection mechanism will select most relevant and informative features.

3. Results and Discussion

In this section, we first describe the datasets and the implementation details, and then present a discussion on the experimental results.

Dataset description and implementation detail

For our experiments, we use 16 datasets from UCI machine learning repository [34]. The detail of the datasets is shown in Table II.

Table 2. Dataset Description

Dataset	Instance	Feature	Class
Parkinsons	195	22	2
Steel	1941	27	7
Breast	569	30	20
Dermatology	366	34	6
Wdbc	569	30	2
Sonar	208	60	2
Glass	214	9	6
German	1000	20	2
Musk	473	165	2
Page-blocks	5472	10	5
Glass123_vs_456	214	9	2
Glass6	214	9	2
Vehicle0	846	18	2
Vehicle1	846	18	2
Vehicle2	846	18	2
Southgerman	1000	20	2

In this experiment, we run 10 fold cross validation (CV) on each dataset and the final result is the average of accuracies and F-score of the 10 folds. Features values are normalized by using *maximum−minimum* normalization to keep the feature values between 0 and 1. For fair comparison the same training-testing is performed for all compared methods (such as ReliefF, SURF, MultiSURF*, MultiSURF). Python *skrebate* [35] library of Relief Based method is used for all the comparative method result generation. Results are generated in Intel core-i5 2.20GHz processor, 12GB RAM and 64-bit windows 10 operating system. To measure the relative efficiency of RFSS, we generate the accuracy of the existing methods (ReliefF, SURF, MultiSURF* and MultiSURF) using the RFSS-selected number of features.

In Table III and IV, our proposed method RFSS is compared with ReliefF, SURF, MultiSURF* and MultiSURF. The accuracy scores obtained by the respective methods are given in the method-wise columns. The number of selected features by RFSS is given in the parentheses of RFSS column. For example, in *musk dataset* RFSS accuracy is 83.1% and the number of selected features is 74. For the same dataset, accuracy of ReliefF (*77.9%*), SURF (*80.4%*), MultiSURF* (*81%*) and MultiSURF (*81.5%*) are obtained using the same number of features as selected by RFSS. To compare the overall performance of RFSS over the other methods, win/ tie/ loss are calculated. Win means RFSS classification accuracy / F-score is better than the compared method accuracy, tie occurs where the accuracy is the same as RFSS, loss when the compared method's accuracy / F-score is better than that of RFSS. For example, among the 16 datasets, RFSS wins in 14 datasets over ReliefF, is equal in 1 dataset and loses in 1 dataset. The best performing method's accuracy / F-score for each dataset is shown in bold format.

The results show that the proposed RFSS performs much better than all the existing methods we compared. In comparison to SURF method, RFSS loses in four datasets. One reason could be that SURF does not need to meet the criterion of k neighbors, so uninformative instances can be avoided in feature weighting. But our method takes k-nearest neighbour and to meet k we may need to take uninformative and far instances which do not provide information about the class separability. Presumably for this reason SURF performs better in these four (dermatology, pageblocks0, glass123 vs 456 and southgerman) datasets. Our method losses for glass123 vs 456 dataset against MultiSURF and MultiSURF*. As MultiSURF takes instances inside the different threshold boundary for each instance, it's not necessary to fulfill the k nearest neighbour instance.

Moreover, MultiSURF* capture two way interaction by weighting the feature both near and far scoring. That is the reason of doing better of these two methods against RFSS.

Table 3: Experimental Result Accuracy Using Svm Classifier

Dataset	ReliefF	SURF	MultiSURF*	MultiSURF	RFSS
Parkinsons	0.840	0.845	0.750	0.855	0.850(15)
Steel	0.674	0.639	0.672	0.654	**0.682(19)**
Breast	0.971	0.971	0.957	0.974	**0.976(20)**
Dermatology	0.968	**0.975**	0.970	0.975	0.970(33)
Wdbc	0.971	0.974	0.957	0.974	**0.976(19)**
Sonar	0.791	0.768	0.763	0.786	**0.795(47)**
Glass	0.535	0.534	0.595	0.534	**0.609(8)**
German	0.751	0.751	**0.756**	0.751	0.751(18)
Musk	0.779	0.804	0.810	0.815	**0.831(74)**
Page-blocks	0.914	**0.923**	0.899	0.920	0.914(4)
Glass123_vs_456	0.913	**0.926**	0.904	**0.926**	0.917(8)
Glass6	0.954	0.954	0.931	0.954	**0.959(8)**
Vehicle0	0.962	0.945	0.959	0.945	**0.957(15)**
Vehicle1	0.745	0.741	0.741	0.741	**0.759(10)**
Vehicle2	0.871	0.878	0.879	0.881	**0.906(12)**
Southgerman	0.768	**0.771**	0.770	0.770	0.756(19)
Win / tie / loss	**14 / 1 / 1**	**11 / 1 / 4**	**14 / 0 / 2**	**12 / 1 / 3**	

Table 4: Experimental Result F-Score

Dataset	ReliefF	SURF	MultiSURF*	MultiSURF	RFSS
Parkinsons	0.72	0.72	0.44	**0.73**	**0.73(15)**
Steel	0.59	0.55	0.63	0.654	**0.62(19)**
Breast	0.96	**0.97**	0.95	**0.97**	**0.97 (20)**
Dermatology	0.96	**0.97**	0.96	**0.97**	0.96(33)
Wdbc	0.96	**0.97**	0.95	**0.97**	**0.97(19)**
Sonar	0.78	0.75	0.75	**0.78**	**0.78(47)**
Glass	0.33	0.33	0.36	0.33	**0.38(8)**
German	0.66	0.66	**0.67**	0.66	0.66(18)
Musk	0.77	0.8	0.8	0.81	**0.82(74)**
Page-blocks	0.29	**0.31**	0.28	0.3	0.29(4)
Glass123_vs_456	**0.89**	**0.89**	0.85	**0.89**	0.88(8)
Glass6	0.89	0.89	0.85	0.89	**0.91(8)**
Vehicle0	0.92	0.92	0.94	0.92	**0.93(15)**
Vehicle1	0.43	0.42	0.42	0.42	**0.49(10)**
Vehicle2	0.8	0.82	0.82	0.83	**0.87(12)**
Southgerman	**0.69**	**0.69**	**0.69**	0.69	0.67(19)
Win / tie / loss	**10 / 4 / 2**	**9 / 3 / 4**	**11 / 1 / 4**	**7 / 5 / 4**	

4. Conclusion

In this paper, we propose a feature selection method RFSS, which constructs a feature subset by modifying ReliefF algorithm in such a way that maximizes the class separation capability of the combined set of features. RFSS follows a greedy forward selection method to add the features to the subset based on their ReliefF ranking. We introduce a mechanism to calculate the weight of combined features, and a candidate feature is added to the feature set if the addition increases the weight of combined feature set. Rigorous experiment shows that RFSS performs better over four state-of-the-art methods ReliefF, SURF, MultiSURF* and MultiSURF.

In the future, we will apply the same technique to other versions of the Relief family of algorithms. While experimental results show a favorable performance for RFSS, a thorough theoretical analysis is yet to be performed. For example, the adverse effect of outliers has not been measured for RFSS. Also, choosing the right value of k is a problem in RFSS. These limitations will be addressed in our future work.

Acknowledgement

This research is supported by the fellowship from ICT Division, Ministry of Posts, Telecommunications and Information Technology, Bangladesh. No - 56.00.0000.028.33.006.20-84; Dated 13.04.2021.

References

1. M. E. Kadir, P. S. Akash, S. Sharmin, A. A. Ali, and M. Shoyaib, "Can a simple approach identify complex nurse care activity?," in Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers, pp. 736–740, 2019.
2. J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan, "Feature selection based on structured sparsity: A comprehensive study," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 7, pp. 1490–1507, 2016.
3. M. Liu, C. Xu, Y. Luo, C. Xu, Y. Wen, and D. Tao, "Costsensitive feature selection by optimizing f-measures," *IEEE Transactions on Image Processing*, vol. 27, no. 3, pp. 1323– 1335, 2017.
4. Y. Wang, L. Feng, and J. Zhu, "Novel artificial bee colony based feature selection method for filtering redundant information," *Applied Intelligence*, vol. 48, no. 4, pp. 868–885, 2018.
5. Y.-T. P. Lee, Sungyoung and B. J. d'Auriol., "A novel feature selection method based on normalized mutual information," *Applied Intelligence*, vol. 37, no. 1, pp. 100 – 120, 2012.
6. M. Mafarja, I. Aljarah, A. A. Heidari, A. I. Hammouri, H. Faris, A.-Z. Ala'M, and S. Mirjalili, "Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems," *Knowledge-Based Systems*, vol. 145, pp. 25–45, 2018.
7. H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 8, pp. 1226–1238, 2005.
8. J. Moody and H. Yang, "Data visualization and feature selection: New algorithms for nongaussian data," *Advances in neural information processing systems*, vol. 12, pp. 687–693, 1999.
9. N. X. Vinh, S. Zhou, J. Chan, and J. Bailey, "Can highorder dependencies improve mutual information based feature selection?," *Pattern Recognition*, vol. 53, pp. 46–58, 2016.
10. K. Igor, "Estimating attributes: analysis and extensions of relief," *European conference on machine learning. Springer*, Berlin, Heidelberg, 1994.
11. S. H. Yang and B.-G. Hu, "Discriminative feature selection by nonparametric bayes error minimization," *IEEE Transactions on knowledge and data engineering*, vol. 24, no. 8, pp. 1422–1434, 2012.
12. C. S. Greene, N. M. Penrod, J. Kiralis, and J. H. Moore, "Spatially uniform relieff (surf) for computationally-efficient

filtering of gene-gene interactions," *BioData Mining*, vol. 2, no. 1, pp. 1–9, 2009.

13. P. M. M. M. Urbanowicz R. J. Olson, R. S. Schmitt and J. H., "Multiple threshold spatially uniform relieff for the genetic analysis of complex human diseases.," Benchmarking reliefbased feature selection methods for bioinformatics data mining. *Journal of Biomedical Informatics*, vol. 85, pp. 168–188, 2018.
14. M. I.Guyon, S.Gunn and L.A.Zadeh, "Feature extraction: Foundations and applications (studies in fuzziness and soft computing)," pringer-Verlag New York, Inc., 2006.
15. I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Machine Learning Research*, vol. 3, pp. 1157– 1182, 2003.
16. A. Senawi, H.-L. Wei, and S. A. Billings, "A new maximum relevance-minimum multicollinearity (mrmmc) method for feature selection and ranking," *Pattern Recognition*, vol. 67, pp. 47–61, 2017.
17. S. Sharmin, M. Shoyaib, A. A. Ali, M. A. H. Khan, and O. Chae, "Simultaneous feature selection and discretization based on mutual information," *Pattern Recognition*, vol. 91, pp. 162 – 174, 2019.
18. M. S. H. Khan, P. Roy, F. Khanam, F. H. Hera, and A. K. Das, "An efficient resource allocation mechanism for time-sensitive data in dew computing," in 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT), pp. 506–510, IEEE, 2019.
19. S. Nakariyakul, "A hybrid gene selection algorithm based on interaction information for microarray-based cancer classification," *Plos One*, vol. 14, no. 2, p. e0212333, 2019.
20. R. Ruiz, J. C. Riquelme, J. S. Aguilar-Ruiz, and M. Garc´ıaTorres, "Fast feature selection aimed at high-dimensional data via hybrid-sequential-ranked searches," *Expert Systems with Applications*, vol. 39, no. 12, pp. 11094–11102, 2012.
21. S. Sharmin, A. A. Ali, M. A. H. Khan, and M. Shoyaib, "Feature selection and discretization based on mutual information," in 2017 *IEEE International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 1–6, IEEE, 2017.
22. T. Naghibi, S. Hoffmann, and B. Pfister, "A semidefinite programming based search strategy for feature selection with mutual information measure," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 8, pp. 1529–1541, 2014.
23. L. Goh, Q. Song, and N. Kasabov, "A novel feature selection method to improve classification of gene expression data," in Proceedings of the second conference on Asia-Pacific bioinformatics-Volume 29, pp. 161–166, *Australian Computer Society*, Inc., 2004.
24. I. Jo, S. Lee, and S. Oh, "Improved measures of redundancy and relevance for mrmr feature selection," *Computers*, vol. 8, no. 2, p. 42, 2019.
25. D. D. Lewis, "Feature selection and feature extract ion for text categorization," Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26,1992, 1992.
26. J. C. Nguyen Xuan Vinh, Shuo Zhou and J. Bailey, "Can highorder dependencies improve mutual information based feature selection?," *Pattern Recognition*, vol. 53, pp. 46—-58, 2016.
27. Kira, Kenji, and L. A. Rendell, "The feature selection problem: traditional method and a new algorithm," *AAAI*, vol. 2, pp. 129–134, 1992. [28] F. Pisheh and R. Vilalta, "Filter-based information-theoretic feature selection," Proceedings of the 2019 3rd International Conference on Advances in Artificial Intelligence, 2019.
28. Y. Sun, "Iterative relief for feature weighting: algorithms, theories, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1035–1051, 2007.
29. C. S. Greene, N. M. Penrod, J. Kiralis, and J. H. Moore, "Spatially uniform relieff (surf) for computationally-efficient filtering of gene-gene interactions," *Biodata Mining*, vol. 2, no. 1, pp. 1–9, 2009.
30. D. S. K. J. Greene, C. S. Himmelstein and M. J. H., "The informative extremes: using both nearest and farthest individuals can improve relief algorithms in the domain of human genetics.," *In European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pp. 182–193, April, 2010.
31. D. Granizo-Mackenzie and J. H. Moore, "Multiple threshold spatially uniform relieff for the genetic analysis of complex human diseases.," *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics. Springer, Berlin, Heidelberg*, pp. 1–10, 2013.
32. R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, and J. H. Moore, "Relief-based feature selection: Introduction and review," *Journal of Biomedical Informatics*, vol. 85, pp. 189–203, 2018.
33. "Uci machine learning repository." https://archive.ics.uci.edu/ml. [Online; accessed 16-February-2021].
34. "Scikit rebate library." https://github.com/EpistasisLab/scikit-rebate. [Online; accessed 3-March-2021].