Improving Discretization by Post-Processing Procedure

Taijun Han[#1], Sangbum Lee[*2], Sejong Oh[#3]
[#]Dept of NanoBioMedical Science, Dankook University, Cheonan, 330-714, Korea
[1] hantaegune@gmail.com
[3] sejongoh@dankook.ac.kr
[*]Dept of Computer Science, Dankook University, Cheonan, 330-714, Korea
[2]sblee@dankook.ac.kr

Abstact. Bioinformatics and data mining require data analysis schemes. Many methods of analysis, such as those focusing on entropy, have been developed and assume that the input data has discrete values. Therefore, when using continuous data, discretization needs to be performed before analysis can begin. Many discretization algorithms have been proposed, and these discretize a given dataset attribute-by-attribute. Although such methods assume that the attributes are independent from each other, in reality these attributes interact with and influence the results of the analysis as a group, not individually. In this paper we propose a post-processing method that can improve the quality of discretization. After the normal discretization process, we adjust the boundary point of the discretization for each attribute, and then after evaluating the group effect of the adjusted point, we update the original boundary point by adjusting it if it has a positive influence on the attribute. The results of the empirical experiments show that the adjusted dataset improves the classification accuracy. Proposed method can be used with any discretization algorithms, and improves their discretization power.

Keywords: Discretization, Classification Accuracy, Bioinformatics, Attribute Interaction

I. INTRODUCTION

In the genomics age, high-throughput data techniques can produce an abundance of high-dimensional biomedical data. The analysis of such data presents significant analytical and computational challenges, and for this reason various data mining techniques have been developed [8]. Many types of data mining and bioinformatics tools require for data to be discretized and, for example, microarray data discretization is a basic pre-process for many algorithms that are used for gene regulatory network inference [16]. Major data analysis schemes – such as a t-test, a Bayesian network, mutual information, and symmetric uncertainty – are based on probability theory and on entropy theory, and they require discrete data as an input in order to calculate some probability or entropy value, which illustrates why discretization is required for continuous data.

Dozens of discretization methods have been proposed since the early computer age, and Liu et al. [7] suggested that discretization methods could be grouped as follows: (1) **supervised** vs. **unsupervised** discretization, were classification depends on whether class information is used during discretizing; (2) **local** vs. **global** discretization, where a local method discretizes in a localized region of the instance space whereas a global method uses the entirety of the instance space; (3) **top-down** vs. **bottom-up** discretization, where top-down methods start with an empty list of cut-points (split-points) and new ones are added to the list by 'splitting' the intervals whereas bottom-up methods start with the complete list of all the continuous values as cut-points and gradually remove some of them by 'merging' the intervals.

In this study, we consider supervised and top-down/bottom-up approaches because the supervised method is more efficient than an unsupervised one. Top-down or bottom-up methods generally contain the following steps [7]:

(1) sort continuous data for discretization

(2) select a candidate cut-point or adjacent intervals

(3) invoke appropriate evaluation measures

(4) if the evaluation value is satisfied, split or merge the intervals, else go to (2)

(5) if a stopping criterion is satisfied, finish the discretization, else go to (2)

The limitation of the supervised methods that were previously proposed is that they do not consider attribute interaction. Each attribute is discretized under a relationship with a class attribute, and other attributes have no

influence in any way. This means that previous supervised methods assume an independence between the attributes. In reality, however, attributes interact with each other in supervised learning [4, 5]. This interaction influences the performance of the classification accuracy, among other measures. Therefore, when we develop a discretization method, we need to consider the interactions between the attributes.

In the context of a supervised prediction task, typically for classification, attribute interaction can be defined in the following manner [2]. Suppose there are three attributes C, X_1, and X_2 where C is the class attribute that is to be predicted, and X_1 and X_2 are predictor attributes. X_1 and X_2 interact when the prediction or magnitude of the relationship between C and X_1 depend on the value of X_2. We then call this a two-way interaction. Higher-order attribute interactions can therefore be defined in a similar way. Let us suppose *acc*(X, C) is the classification accuracy by attribute X onto class C. If $acc(\{X_1, X_2\}, C) > max[acc(X_1, C), acc(X_2, C])$, then a positive interaction exists between X_1 and X_2 . If $acc(\{X_1, X_2\}, C) < min[acc(X_1, C), acc(X_2, C)]$, then there is a negative interaction between X_1 and X_2.

In this study we suggest a post-processing technique that can be used after discretization to reflect attribute interaction. Proposed method is not new discretization algorithm; it makes enhancement of given discretization algorithm. After discretization is finished using any algorithms, we adjust the cut-points considering the attribute interaction. In the result of the empirical experiments, we confirm that the adjusted dataset brings an improvement for classification accuracy.

II. Post-Processing Method for Discretization

In an ordinary discretization process, we use continuous values as an input of a given attribute, and then the discretization algorithm finds the proper cut-points that correspond to the evaluation criterion. Finally, the continuous values are converted into discrete values by using these cut-points (*see* Fig. 1). Every attribute is converted to discrete values in the same way. As we had previously mentioned, this approach does not consider the interaction between the attributes. Therefore, we could inject the effect of the interaction into the discretization algorithm, but this would be difficult and complex. Our strategy is to separate the discretization and to reflect the attribute interaction. The advantage of this approach is that we can separately develop a discretization method and then apply a method to take into account attribute interaction.

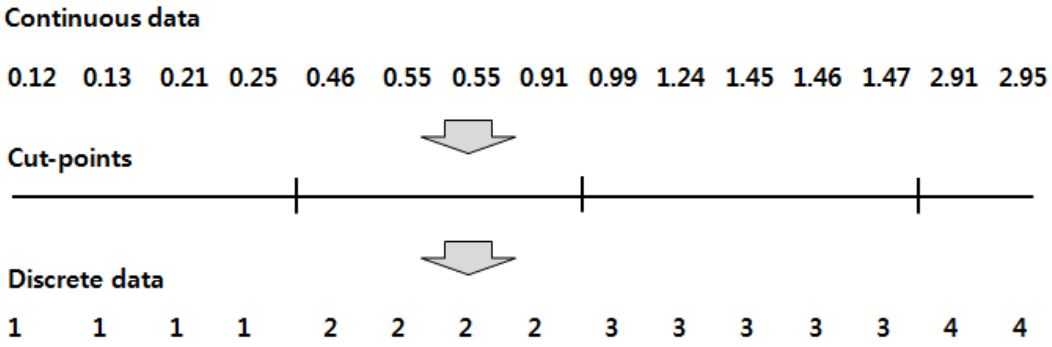

Fig. 1. Ordinary discretization process

Now, we describe the post-processing method that is used for the discretization method. Our basic idea is to adjust the cut-points, and to make a new discretization according to the adjusted points. We then re-evaluate the discretized dataset in a manner that takes into account the interaction of the attributes. If adjusting a cut-point brings an improvement in the evaluation of the value, we change the original cut-point to the adjusted cut-point and re-generate the discrete dataset. Let us then suppose that our method is given an original continuous dataset (CD), a discretized dataset (DD), and cut-points of all attribute for discretization (CUT-POINTS). The proposed post-processing method is as follows:

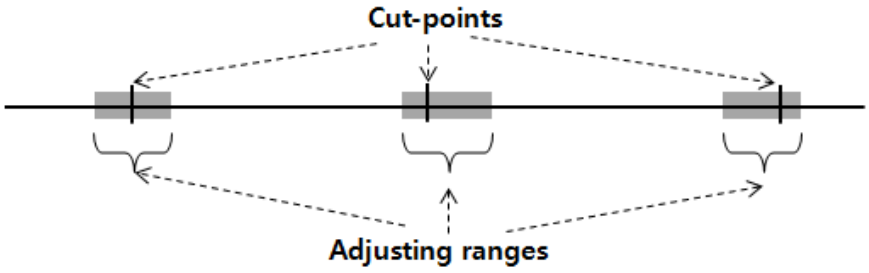

Fig. 2. Adjusting ranges (gray region) for cut-points

(1) evaluate the relevance of each attribute in DD and of the class attribute

(2) sort attributes in DD, CD, CUT_POINTS in descending order of relevance.

(3) calculate the base performance criterion value, *bp*, using CD

(4) take *m*% of best relevant attributes as a candidate for the adjusting process. (In our experiment, *m* = 10.)

(5) take an attribute *x* from the candidate attributes

(6) take a cut-point *p* of *x* and find the adjusting range

(7) move *p* to *p'* and re-discritize *x* according to *p'*

(8) calculate the new performance criterion value *cp* using the adjusted DD

(9) if *cp* > *bp* change *p* = *p'*and *bp* = *cp*. Go to (6)

(10) repeat steps 5–9 until there are no candidate attributes to adjust.

There are three important issues related to the implementation of the proposed post-processing method. The main issue are the performance criteria of *bp* and *cp* in steps (3) and (8), which are related to the attribute interaction. Measure attribute interaction is an important issue for data mining [2], and various measures have been proposed to achieve as much. Mutual information [2], interaction gain [3], synergy [18], symmetric uncertainty, ITERACT [18], and merit function [2] are well-known methods, and all of them deal with two-way interactions. *N*-way interactions, on the other hand, are those for which $N > 2$ is measured by an average of two-way interactions for pairs of attributes. It is not proper in such cases to capture *N*-way attribute interaction, and the only way to measure *N*-way interaction is to use the classification accuracy. Classification is a supervised learning method which is dimension-free. Therefore, tens of thousands of attributes are no problem for classification. We can think that classification accuracy reflects the quality of the attributes and the attribute interaction. Therefore, we can indirectly measure the attribute interaction of the known classification algorithms. Jakulin et al. [4] used naïve Bayesian classification to measure the attribute interaction. In our experiment we adopt a C5.0 classifier [11,19] because it is widely used to test the quality of discretized datasets.

The second issue is that of reducing the problem space. Some datasets have many attributes, which means that there are many cut-points that we must adjust. In order to reduce the computation time, we select *m*% of best relevant attributes in step 4 as candidates to adjust the test. This process assumes that the change of strongly relevant attributes influences the classification accuracy more than weakly relevant attributes do. In order to find relevant attributes, we evaluate each feature using the attribute selection algorithm from step 1. FSDD [9], Relief [15], MRMR [5], and RFS [10] are known algorithms. In our experiment, we simply use mutual information between each attribute and the class to measure relevancy because mutual information is simple and allows for a high speed of computation.

The third issue is that of adjusting the range in step 6. Each attribute contains many data values, and we cannot test the adjustment of the cut-points for every data value. Instead, we just test the data values that are near the cut-points, which is the adjusting range. The adjusting range for a specific cut-point contains 10% forward data values and 10% backward data values.

III. Experiments and Results

To test the efficiency of proposed post-processing method, we choose 30 continuous datasets from the UCI machine learning repository (http://archive.ics.uci.edu/ml/) and KEEL-dataset site (http://sci2s.ugr.es/keel/category.php?cat=clas). Table I summarizes the basic list of the benchmarking datasets, and we also choose three supervised (CAIM, CACC, AMEVA) and three unsupervised (INTERVAL, FREQUENCY,

CLUSTER) discretization methods, as listed in Table II. The proposed method was considered for supervised discretization, but we also experiment with unsupervised discretization in order to compare both methods.

TABLE I

Summary of the datasets

ID	Dataset name	ID	Dataset name
D1	Abalone	D16	page-blocks
D2	Ecoli	D17	parkinsons
D3	faults.NNA	D18	pima-indians-diabetes
D4	Glass	D19	segmentation
D5	hayes-roth	D20	smoke
D6	Ionosphere	D21	sonar
D7	Iris	D22	spectrometer
D8	letter-recognition	D23	statlog_segment
D9	Lung	D24	wdbc
D10	SRBCT	D25	wine
D11	multi_tissues	D26	winequality-white
D12	Satimage	D27	yeast
D13	SPECTF	D28	newthyroid
D14	Waveform	D29	Wholesale_data
D15	Liver	D30	bupa

TABLE II

List of discretization methods

Method	Description	Ref.
CAIM	Class-Attribute Interdependence Maximization	[13]
CACC	Class-Attribute Contingency Coefficient	[6]
AMEVA	Ameva algorithm	[12]
INTERVAL	equal interval width	[5]
FREQUENCY	equal frequency	[5]
CLUSTER	k-means clustering	[5]

We use the R language (http://www.r-project.org) to implement the proposed method and the benchmark test program. To test three supervised discretization algorithms, the *discretization* package in R is used, and the *arules* package is used for the three unsupervised discretization algorithms. A classification accuracy from the C5.0 classifier [11] is adopted as a performance criterion for the proposed method, as implemented in the *C50* package. To compare the improvement of the classification accuracy between the original dataset and the post-processed dataset, we use the C5.0 classifier. In order to avoid the overfitting problem, we apply 10-fold cross validation and repeat it 5 times. For our proposed method, therefore, there is no improvement in the classification that is a result of chance.

Tables III and IV present the comparison for the classification accuracy between normal discretization and post-processed datasets. From Tables III and IV, we can observe as follows:

- Overall, the discretized datasets due to the supervised discretization induce better classification accuracy than by some unsupervised discretization. (*see* values of 'N' columns in Tables III and IV.)
- The adjusted datasets from the supervised discretization show a much more improved number of cases than from some unsupervised discretization. In the improved case, the classification accuracy is improved after applying the proposed post-processing method.
- Though the proposed method is not well-matched with unsupervised discretization, some datasets show a high improvement in classification accuracy, such as D6 with INTERVAL and FREQUENCY or D11 with FREQUENCY. Post-processing brings 7%, 5.3%, and 12.6% improvements to classification accuracy in these

cases.

- If the number of adjusted attributes/instances is small in a dataset, it tends to show a small improvement in classification accuracy, and vice versa. This may imply that each dataset has a different degree of interactions between the attributes.
- Though the proposed method fits well with supervised discretization, some discretization methods present low efficiency in terms of their improvement. In Table III, CACC produces a lower improvement than CAIM and AMEVA do.

TABLE III

Comparison of classification accuracy between normal supervised discretization and post-processed datasets

	CAIM			**CACC**			**AMEVA**		
	N	**P**	**C**	**N**	**P**	**C**	**N**	**P**	**C**
D1	0.546	0.550	6/6	0.543	0.548	4/4	0.543	0.548	4/4
D2	0.837	0.864	2/2	0.817	0.826	4/4	0.817	0.826	4/4
D3	0.731	0.728	6/6	0.748	0.753	11/11	0.744	0.743	12/12
D4	0.736	0.739	3/3	0.682	0.708	1/1	0.716	0.724	3/3
D5	0.837	0.855	1/1	0.863	0.863	0/0	0.851	0.851	0/0
D6	0.902	0.914	4/4	0.905	0.919	4/4	0.913	0.913	3/3
D7	0.940	0.953	1/1	0.940	0.953	1/1	0.940	0.953	1/1
D8	0.880	0.880	1/1	0.880	0.880	1/1	0.880	0.880	1/1
D9	0.731	0.734	1/1	0.789	0.789	0/0	0.758	0.765	2/2
D10	0.871	0.879	2/2	0.919	0.919	0/0	0.890	0.911	2/2
D11	0.829	0.833	3/3	0.767	0.767	0/0	0.791	0.819	3/3
D12	0.864	0.864	9/7	0.859	0.857	7/5	0.859	0.857	7/5
D13	0.730	0.853	4/4	0.728	0.755	2/2	0.740	0.813	3/3
D14	0.773	0.783	16/15	0.789	0.787	12/12	0.787	0.790	14/14
D15	0.696	0.699	1/1	0.688	0.709	4/4	0.673	0.686	5/5
D16	0.966	0.967	2/2	0.970	0.970	3/3	0.970	0.970	3/3
D17	0.862	0.911	8/8	0.904	0.911	7/7	0.909	0.920	5/5
D18	0.743	0.760	3/3	0.772	0.774	3/3	0.767	0.784	4/4
D19	0.870	0.865	1/1	0.898	0.906	1/1	0.899	0.911	1/1
D20	0.425	0.446	3/3	0.418	0.418	0/0	0.538	0.555	3/3
D21	0.800	0.780	8/8	0.782	0.782	0/0	0.810	0.859	3/3
D22	0.841	0.847	1/1	0.863	0.859	4/4	0.860	0.867	4/4
D23	0.961	0.962	4/4	0.969	0.970	4/4	0.968	0.970	5/5
D24	0.955	0.965	5/5	0.946	0.965	6/6	0.949	0.951	9/9
D25	0.937	0.957	3/3	0.949	0.970	2/2	0.944	0.973	2/2
D26	0.564	0.564	8/8	0.593	0.595	4/4	0.573	0.576	7/7
D27	0.598	0.599	2/2	0.572	0.585	2/2	0.576	0.585	2/2
D28	0.940	0.957	2/2	0.943	0.953	1/1	0.943	0.953	1/1
D29	0.912	0.909	1/1	0.918	0.922	3/3	0.919	0.924	4/4
D30	0.696	0.699	1/1	0.686	0.706	3/3	0.652	0.656	3/3

N: accuracy by normal discretization

P: accuracy by proposed method

C: number of adjusted features/ number of adjusted instanced

TABLE IV

Comparison of classification accuracy between normal unsupervised discretization and post-processed datasets

	INTERVAL			FREQUENCY			CLUSTER		
	N	**P**	**C**	**N**	**P**	**C**	**N**	**P**	**C**
D1	0.54	0.54	0/0	0.539	0.544	5/5	0.541	0.541	0/0
D2	0.64	0.64	0/0	0.655	0.652	1/1	0.647	0.647	0/0
D3	0.72	0.72	7/7	0.685	0.684	12/12	0.722	0.728	7/7
D4	0.71	0.71	1/1	0.697	0.701	1/1	0.704	0.704	0/0
D5	0.63	0.64	1/1	0.644	0.644	0/0	0.746	0.746	0/0
D6	0.83	0.90	3/3	0.813	0.866	11/11	0.876	0.907	5/5
D7	0.97	0.97	0/0	0.977	0.977	0/0	0.957	0.957	0/0
D8	0.88	0.88	1/1	0.880	0.881	9/4	0.878	0.878	1/1
D9	0.75	0.75	1/1	0.723	0.723	0/0	0.750	0.750	0/0
D10	0.88	0.88	1/1	0.797	0.841	2/2	0.823	0.804	2/2
D11	0.79	0.79	0/0	0.677	0.803	1/1	0.822	0.822	0/0
D12	0.86	0.86	0/0	0.862	0.862	1/1	0.858	0.858	0/0
D13	0.73	0.73	2/2	0.745	0.745	1/1	0.640	0.640	4/4
D14	0.77	0.77	0/0	0.768	0.768	6/5	0.771	0.771	0/0
D15	0.68	0.68	1/1	0.629	0.650	3/3	0.659	0.659	0/0
D16	0.96	0.96	1/1	0.968	0.968	1/1	0.970	0.970	0/0
D17	0.88	0.88	0/0	0.884	0.884	0/0	0.881	0.881	0/0
D18	0.73	0.73	0/0	0.732	0.728	4/4	0.740	0.740	0/0
D19	0.45	0.45	0/0	0.490	0.500	1/1	0.475	0.475	0/0
D20	0.46	0.44	2/2	0.374	0.392	4/4	0.373	0.365	2/2
D21	0.75	0.75	0/0	0.753	0.751	1/1	0.708	0.708	0/0
D22	0.81	0.81	8/8	0.827	0.827	2/2	0.827	0.830	1/1
D23	0.59	0.59	0/0	0.606	0.606	0/0	0.592	0.592	0/0
D24	0.94	0.94	0/0	0.931	0.931	0/0	0.948	0.942	1/1
D25	0.93	0.93	0/0	0.916	0.916	0/0	0.955	0.955	0/0
D26	0.58	0.58	0/0	0.585	0.585	7/7	0.586	0.586	0/0
D27	0.52	0.53	1/1	0.575	0.571	6/6	0.550	0.560	1/1
D28	0.939	0.939	0/0	0.920	0.920	1/1	0.940	0.940	0/0
D29	0.895	0.895	0/0	0.904	0.904	0/0	0.903	0.903	0/0
D30	0.670	0.670	0/0	0.647	0.648	1/1	0.681	0.681	0/0

Table V summarizes the experiment results. The proposed method has a high ratio of cases for improvement (63%–80%) with CAIM, CACC, and AMEVA. Their average improved classification accuracies are of 1.63%, 1.19%, and 1.49%, which are meaningful improvements for the classification test. Unsupervised discretization does not consider the correlation between the attributes and class information. Therefore, the adjustment of the cut points is not influenced by the class information, and the proposed method does not work well.

TABLE V

Summary of experiments

Method	# of improved cases	# of same cases	# of decreased cases	Average improved accuracy (%)
CAIM	23 (76.7)	3 (10.0)	4 (13.3)	1.63
CACC	19 (63.3)	8 (26.7)	3 (10.0)	1.19
AMEVA	24 (80.0)	4 (13.3)	2 (6.7)	1.49
INTERVAL	3 (10.0)	26 (86.7)	1 (3.3)	3.17

( ) : proportion of the cases against 30 cases

IV. CONCLUSIONS

Discretization is one of the forms of basic pre-processing that must be done before many machine learning tasks are employed. Previous discretization algorithms discretize each attribute in a dataset independently, and in order to consider the interaction between the attributes, we suggested a post-processing step that can be performed upon discretized datasets. The experimental results show that the proposed method can improve the quality of the datasets and can bring classification accuracy, and this method works well with various supervised discretization algorithms.

In this study, we use a C5.0 classifier to measure the performance criterion, since this performance criterion works well for classification tasks. If discretization is performed for other tasks, then some other proper performance criterion can be designed. Choosing the performance criterion should depend on the targeted machine learning task, which shares the same principle with a wrapper or embedded feature selection model [17]. The implementation of the proposed method is posted on 'http://biosw.dankook.ac.kr/biosw/postDiscretization'.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2012S1A2A1A01028576). Corresponding author is Sejong Oh.

REFERENCES

[1] A. A. Freitas. Understanding the Crucial Role of Attribute Interaction in Data Mining. ARTIF INTELL REV, 2001, 16:177--199.

[2] A. Jakulin. Machine Learning Based on Attributes Interactions, PhD Dissertation, University of Ljubljana, 2005.

[3] A. Jakulin, I. Bratko. Analyzing Attribute Dependencies. LECT NOTES ARTIF INT, PKDD 2003, 2003, 229—240

[4] A. Jakulin, I. Bratko, D. Smrke. Attribute Interactions in Medical Data Analysis. LECT NOTES ARTIF INT 2780, AIME 2003, 2003, 229--238.

[5] C. Ding, H. Peng. Minimum Redundancy Feature Selection from Microarray Gene Expression Data. in: Proceedings of the IEEE Computer Society Conference on Bioinformatics, IEEE Computer Society, 2003, 523.

[6] C. J. Tsai, C. I. Lee, W. P. Yang. A discretization algorithm based on Class-Attribute Contingency Coefficient. Information Sciences, 2008, 178:714--731.

[7] H. Liu, F. Hussain, C. L. Tan, M. Dash. Discretization: An Enabling Technique. DATA MIN KNOWL DISC, 2002, 6:393--423.

[8] J. L. Lustgarten, S. Visweswaran, V. Gopalakrishnan, G. F. Cooper. "pplication of an efficient Bayesian discretization method to biomedical data. BMC Bioinformatics, 2011, 12(309):1--15.

[9] J. Liang, S. Yang, A. Winstanley. Invariant optimal feature selection: A distance discriminant and feature ranking based solution. PATTERN RECOGN, 2008, 41:1429--1439.

[10] J. Lee, N. Batnyam, Oh S. RFS: Efficient feature selection method based on R-value. COMPUT BIOL MED, 2013, 43:91--99.

[11] L. Breiman, J. Friedman, R. Olshen, C. Stone. Classification and Regression Trees, New York: Chapman and Hall, 1984.

[12] L. Gonzalez-Abril, F. J. Cuberos, F. Velasco, J. A. Ortega. Ameva: An autonomous discretization algorithm. EXPERT SYST APPL, 2009, 36:5327—5332

[13] L. A. Kurgan, K. J. Cios. CAIM Discretization Algorithm, IEEE T KNOWL DATA EN, 2004, 16:145--153.

[14] M. Hahsler, B. Gr¨un, K. Hornik, C. Buchta. Introduction to arules – A computational environment for mining association rules and frequent item sets. J STAT SOFTW, 2005, 14 (15):1--25.

[15] M. Robnik-Sikonja, I. Kononenko. Theoretical and Empirical Analysis of ReliefF and RReliefF. Mach Learn, 2003, 53:23--69.

[16] Y. Li, L. Liu, X. Bai, H. Cai, W. Ji, D. Guo, Y. Zhu. Comparative study of discretization methods of microarray data for inferring transcriptional regulatory networks. BMC Bioinformatics, 2010, 11(520): 1--6.

[17] Y. Saeys, I. Inza, P. Larranaga. A review of feature selection techniques in bioinformatics. Bioinformatics, 2007, 23(19):2507--2517.

[18] Z. Zhao, H. Liu. Searching for interacting features. INTELL DATA ANAL, 2009, 13:207--228.

[19] S. Garcia, J. Luengo, J. A. Sáez, V. López, A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning, IEEE Transactions on Knowledge and Data Engineering, 2013, 25(4): 734-750.

AUTHOR PROFILE

Taijun Han is a master student at the Nanobiomedical Science of Dankook University. His main research interests are data classification and bioinformatics.

Sangbum Lee is a professor of Department of Computer Science at Dankook University, Korea. His main research interests are software engineering, ontology, and bioinformatics.

Sejong Oh received a Doctor, Master, and Bachelor degree in Computer Science from Sogang University, Korea, in 2001, 1991, and 1989. From 2001 to 2003, he was a postdoctoral fellow in the laboratory for Information Security Technology at George Mason University, USA. Since 2003 he joined the Department of Computer Science at Dankook University, Korea, and is currently associate professor in WCU Research Center of NanoBioMedical Science. His main research interests are bioinformatics, information system, and information system security.