



A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree

Kyoungok Kim*

Information Technology Management Programme, International Fusion School, Seoul National University of Science & Technology (SeoulTech),
232 Gongreungno, Nowon-gu, Seoul 139-743, South Korea



ARTICLE INFO

Article history:

Received 7 December 2015

Received in revised form

21 April 2016

Accepted 25 April 2016

Available online 17 May 2016

Keywords:

Decision tree

Semi-supervised decision tree

Inhomogeneous measure

Subspace partitioning

ABSTRACT

Among data mining techniques, the decision tree is one of the more widely used methods for building classification models in the real world because of its simplicity and ease of interpretation. However, the method has some drawbacks, including instability, the nonsmooth nature of the decision boundary, and the possibility of overfitting. To overcome these problems, several works have utilized the relative advantages of other classifiers, such as logistic regression, support vector machine, and neural networks, in combination with a decision tree, in hybrid models which avoid the drawbacks of other models. Some hybrid models have used decision trees to quickly and efficiently partition the input space, and many studies have proved the effectiveness of the hybrid methods. However, there is room for further improvement by considering the topological properties of a dataset, because typical decision trees split nodes based only on the target variable. The proposed semi-supervised decision tree splits internal nodes by utilizing both labels and the structural characteristics of data for subspace partitioning, to improve the accuracy of classifiers applied to terminal nodes in the hybrid models. Experimental results confirm the superiority of the proposed algorithm and demonstrate the detailed characteristics of the algorithm.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Classification is a data mining task that predicts categorical class labels. A classifier is a function that maps input samples consisting of several predictors to one of the class labels. Many different classification techniques have been developed and applied to several classification problems in various fields. Logistic regression [1,2], decision tree [3,4], neural networks [5] and support vector machine (SVM) [6,7] are the most popular methods to build a classification model.

A decision tree is simple to use, easy to understand, and offers various advantages compared with other methods. Thus, decision trees have been intensively studied and established as a useful technique to solve real-world problems. However, the decision tree algorithms that were suggested in the early stages had some disadvantages, such as being unstable and prone to overfitting. One approach to overcoming the weak points of decision trees was to combine the decision tree algorithm with other classification models [8–11]. In most cases, the decision tree algorithm was applied to the dataset first, and then other classification models were built based on samples at terminal nodes. In such cases the

decision tree has the role of dividing a dataset into several subsets, and the combined classification models are used as classifiers for each subspace. These hybrid models have achieved better classification performance than single classification models.

Although combinations of decision trees and other models have been empirically proved to be effective for classification problems, the decision tree algorithm is not optimized for subspace partitioning. The decision tree creates child nodes by conducting a value test, which measures the ability of input features to estimate the dependent variable. As a result, subsets created by a decision tree often lack structural or topological homogeneity. Partitioning based on labels may improve the final classification accuracy of individual models at terminal nodes, because the ground assumption for classification is that observations with the same label are similar to each other in terms of explanatory variables.

However, the semi-supervised decision tree proposed in this paper is motivated by the likelihood that class distribution may be different depending on the regions where the samples are located. Unlike a traditional decision tree, the semi-supervised decision tree considers not only class labels but also distributions of input variables, which reflect and incorporate the structural or topological properties of the data. Split rules obtained by the proposed algorithm reflect the boundaries of the separate regions where input values are concentrated. This cutoff point is based on input

* Tel.: +82 29707286.

E-mail address: kyoungok.kim@seoultech.ac.kr

features only, so the term “semi-supervised” is used to describe the proposed algorithm.

The rest of the paper is organized as follows. In Section 2, decision tree and hybrid decision tree models are described. In Section 3, the proposed semi-supervised decision tree algorithm is explained in detail. In Section 4, the experimental results obtained from the proposed algorithm and other comparative algorithms are presented. Finally, conclusion of the paper is given in Section 5.

2. Literature reviews

2.1. Decision tree

The decision tree is one of the data mining algorithms widely used for both classification and regression. Each interior node corresponds to one of the input variables and is split into child nodes based on the values of the input variable. Each leaf or terminal node represents the particular value of a target variable, for example, the specific class of a categorical variable for classification problem, and the certain real value of a continuous variable for regression problems.

During the classification tree learning process, samples at each interior node are split into subsets based on an attribute, and this process is repeated on each derived subset in a recursive manner called “recursive partitioning.” The recursion is finished when a subset at a node has the same target value, when splitting does not improve prediction, or when splitting is impossible because of user-defined constraints.

At every step when growing a decision tree, one of the input variables is selected to split samples. Based on the chosen variable, the split point is determined by an attribute value test. Gini impurity and entropy are the most widely used tests for classification trees. Following two equations describe entropy [12] and Gini impurity [3], respectively:

$$H_e(S) = - \sum_{y \in C} p(y) \log_2 p(y) \quad (1)$$

$$H_g(S) = \sum_{y \in C} p(y)(1 - p(y)) = 1 - \sum_{y \in C} p(y)^2 \quad (2)$$

where S represents the dataset, C is a set of classes, and $p(y)$ is the proportion of the number of samples with the class label, y in C . Both Gini impurity and entropy have 0 when there is the only one class in C and reach the maximum value when all classes are equally probable.

The split rule is determined by a reduction in entropy and Gini impurity after the split, which is called information gain:

$$G(r, S) = H(S) - \sum_t p(t)H(t) \quad (3)$$

where r is a certain split rule and t represent the child nodes induced by r on the set S at the current node. $p(t)$ is the proportion of the number of samples corresponding to t . The attribute and the value for split rule are determined to obtain the maximum information gain at the current node.

After the growth of the full tree, the output classes of the samples are determined at terminal nodes. Each terminal node decides on a target value based on the majority class at the terminal node. When a new sample is observed, it is classified as one of the terminal nodes of the tree depending on input variables. Then, the target is predicted to be the majority class at the leaf node.

There are several variations in decision tree algorithms, with different details. Iterative Dichotomiser 3 (ID3) is a proposed algorithm which uses entropy to calculate information gain [4].

This algorithm tests all of the unused attributes of a dataset at each iteration and then splits each interior node into two children. C4.5 is an extension of the ID3 algorithm that builds a tree in the same way as ID3, i.e., by using entropy [13,14]. The difference in C4.5 properties is that C4.5 can handle both continuous and categorical attributes and learn from a dataset with missing values. Moreover, C4.5 introduces a pruning step after tree growth to avoid overfitting. One of the widely used decision tree algorithms is classification and regression tree (CART) [3]. Unlike ID3 and C4.5, CART can create both classification and regression trees. The basic procedures used for both classification and regression have several similarities but have some differences, such as the method to determine where to split.

2.2. Characteristics of decision trees

The decision tree has advantages when solving classification or regression problems. First, the interpretation of results in a tree is very simple. In the test phase, the tree is useful for rapidly predicting new observations, and the result is easily interpreted using a few simple if–then statements. This property is powerful for addressing real business problems because the decision tree can offer some intuition about the relationships between input and output.

Second, the decision tree is nonparametric. During training, it creates logical if–then rules which do not require an implicit assumption about the underlying distributions or relationships of the input and output variables. Thus, the decision tree is appropriate for general data mining problems where minimal prior knowledge about the data is common.

Third, the decision tree is inherently nonlinear. It can use the same variable several times to create split rules during tree growth, revealing a nonlinear relationship between the variables. Although introducing nonlinearity often increases model complexity and reduces the ability of interpretation in other data mining techniques, the decision tree can implement nonlinearity in a simple way.

Finally, the decision tree can easily handle categorical and numerical variables in the same algorithm, unlike many other data mining algorithms, such as SVM, generalized linear or logistic regression models, and neural networks. In most algorithms, handling categorical variables is complicated.

2.3. Hybrid decision tree combining with other models

The decision tree has many positive points, but it also has limitations. The decision boundary obtained from the trained tree is not smooth, because the decision tree considers one variable at each node and splitting is performed at a certain point. In addition to this point, the probabilities of classes estimated from the decision tree are finite, and the score of the specific terminal node is shared by all cases in that node. Another flaw of the decision tree is instability, in that small changes in input features can lead to large changes in the final tree.

To overcome these drawbacks, hybrid models, which combine a decision tree with other classifiers, have been proposed. One of the advantages of decision trees is that the decision tree algorithm can generate simple if–then statements to make subsets purer than the original dataset in terms of the target variable. Using generated rules, observations including train data and new data are assigned into one of the subsets, and this process can be viewed as subspace partitioning. Therefore, the decision tree acts as a pre-processing step to divide a dataset into small subsets.

In [15], CART was combined with logistic regression to analyze motor vehicle injury data. The shallow decision tree was produced by CART, and logistic regression models were trained on each

reduced set of terminal nodes. In [9], CART was hybridized with logistic regression for classification problems, but only one logistic regression model was trained over the entire training set. CART was used to create a new categorical variable representing terminal nodes.

Decision trees were also used together with naïve Bayes classifiers. In [8], different naïve Bayes classifiers were built on partitioned subsets in the same way described in [15]. In [11], SVM was applied to the most ambiguous leaf node in the tree built on satellite remote sensing data.

Entropy nets employed decision trees to fast and efficiently train neural networks [16]. Entropy nets were inspired by the research that showed that hidden layers provide the capability to create intricately curved partitions of space with complex nonlinear decision boundaries using the nonlinear sigmoid function [17]. Internal and terminal nodes were mapped to nodes in different layers of neural networks in [16]. Internal nodes were connected to the hidden layer called a partitioning layer. Another work related to neural networks directed the ability of decision trees to partition space. In this research, neural networks were used to modify the decision boundary of trees to obtain the hyperplane determined by every input variable [18].

Through several studies, hybrid models of decision trees and other classifiers have been proven to improve the performance of individual classifiers. However, no study has proposed that subspace partitioning considers topological or structural characteristics of data. In the next section, the improved subspace partitioning algorithm by the decision tree will be described in detail, which is differentiated in terms of capturing both information embedded in the target variable and input variables.

3. Semi-supervised decision tree

This section describes a semi-supervised decision tree which utilizes both label information and input predictors to split nodes. The key concepts of the proposed method are as follows:

1. Assumption 1: Different classes may distribute in different regions. Hence, it is better to split nodes to increase purity, and this criterion is the same as for splitting nodes in general decision trees.
2. Assumption 2: Objects of a dataset may be located in different subspaces, and in each subspace, classes are distributed in different ways. In this case, subspaces should be captured regardless of label information.

The main purpose of traditional decision trees is to accurately predict an output value for a new observation. Thus, Assumption 1 is the main criterion for splitting interior nodes. However, when the situation corresponding to Assumption 2 happens in real cases, traditional decision tree algorithms cannot partition the input space properly.

A new split criterion is proposed in this paper by considering label information and the characteristics or properties of input features, which is the main reason that the proposed algorithm is called a semi-supervised decision tree. The new algorithm tries to find a valley or boundary between peaks, where the peak is the point where observations densely locate, and the valley is the point where training samples sparsely distribute along a certain feature.

3.1. Measure of inhomogeneity

To recognize the regions where data points are relatively concentrated, various inhomogeneity measures were used.

Homogeneity describes the state or quality of a dataset being homogeneous, which is when the statistical properties of any one part of an overall dataset are the same as the other parts. Likewise, inhomogeneity describes a dataset in which any part of the dataset is distinct or distinguished from the other parts in terms of statistical properties. These statistical properties contain all aspects of the statistical distributions, including the location parameters.

The ground assumption of classification problems is that two points with similar input have a high probability of sharing the same label. Therefore, the distribution of observations within the same class may have high homogeneity. On the other hand, the distribution of a dataset consisting of two different classes may be inhomogeneous because two classes follow different families of distributions, or different distributions with different parameters. Decision tree algorithms partition input space based on this assumption that different classes are distributed in spatially different locations, and the split increases the homogeneity of partitioned subspaces in terms of distributions of the output classes.

Here, impurity measures, such as the Gini impurity, and entropy used in decision trees to calculate the inhomogeneity of target distribution, were introduced to the new algorithm. The difference between ordinary decision trees and the proposed method is that the additional impurity is computed by only using an input feature. The values of the impurity measures according to an input feature represent the inhomogeneity of the input feature. In an unsupervised setting for the proposed algorithm, the values of a predictor are regarded as classes. If an attribute is a categorical variable, then the calculation of the Gini impurity and entropy is simple. Otherwise, a continuous feature is discretized into several bins to transform the feature to be categorical, to apply the same measure to both types of variables. After obtaining the discretized feature, any impurity measure is computed in the same manner as that for the target variable in growing a tree:

$$H_e(x_i) = - \sum_a p(x_i^a) \log p(x_i^a) \quad (4)$$

$$H_g(x_i) = \sum_a p(x_i^a)(1 - p(x_i^a)) = 1 - \sum_a (p(x_i^a))^2 \quad (5)$$

In both equations, x_i represents the i th feature and a represents the specific categorical value or the a th bin. $p(x_i^a)$ is defined as the number of data points with the certain categorical value or in the certain bin divided by the total number of data points.

Selecting the number of bins may affect the final inhomogeneity; thus, the discretization step is important. Several methods are available to select the number of bins(k) or the bin width (h). In this study, some of the widely used methods were applied to determine the appropriate number of bins:

- *Sturges' formula*: It is derived from a binomial distribution and implicitly assumes a normal distribution [19]. It can perform poorly if the number of data points is less than 30 or the data does not follow a normal distribution:

$$k = \lceil \log_2 n + 1 \rceil \quad (6)$$

- *Doane's formula*: It is a modification of Sturges' formula which attempts to improve its performance on non-normally distributed data [20]:

$$k = 1 + \log_2 n + \log_2 \left(1 + \frac{|g_1|}{\sigma_{g_1}} \right) \quad (7)$$

where g_1 is the estimated 3rd-moment-skewness of a distribution [21]

$$\sigma_{g_1} = \sqrt{\frac{6(n-1)}{(n+1)(n+3)}} \quad (8)$$

In this paper, the 3rd-moment-skewness was estimated by Wang's method [22].

- **Scott's normal reference rule:** It calculates the optimal bin width for random samples of normally distributed data instead of the number of bins by the following equation [23]:

$$h = \frac{3.5\hat{\sigma}}{n^{1/3}} \quad (9)$$

where $\hat{\sigma}$ is the sample standard deviation. This approach minimizes the integrated mean squared error of the density estimate.

- **Freedman–Diaconis' rule:** It is an alternative of Scott's normal reference rule, which also calculates the bin width based on the interquartile range, denoted as IQR . It replaces $3.5\hat{\sigma}$ of Scott's normal reference rule with $2IQR$, which is less sensitive to outliers than the standard deviation [24]:

$$h = \frac{2IQR}{n^{1/3}} \quad (10)$$

When Sturge's formula and Doane's formula are used, the difference between minimum and maximum values of a feature is divided by k to calculate h . Then, the attribute is divided into k bins with the interval h such as $[x_{\min}, x_{\min} + h, x_{\min} + 2h, \dots, x_{\min} + k \cdot h = x_{\max}]$. For Scott's normal reference rule and Freedman–Diaconis' rule, k is approximated using the obtained h , $k = \lceil \frac{x_{\max} - x_{\min}}{h} \rceil$ and then h is recalculated again based on k to divide feature in the same way for cases with Sturge's formula and Doane's formula.

After binning, each bin is treated as a different class, and then entropy and Gini impurity are calculated. If observations are more concentrated in different bins along the specific input feature than other features, the split with this variable can reduce further inhomogeneity.

Instead of discretization, another inhomogeneity measure based on scatter matrices was also examined for continuous variables, which is similar to an analysis of variance (ANOVA). Two child nodes are considered as two different groups, and the sum of squares within each group is calculated to find an appropriate split point. The larger within-group variance represents the better split point, which implies that this value can be considered as a homogeneity measure. Otherwise, between-group variance can be used as an inhomogeneity measure. The relationship between the total sum of squares, the within-group sum of squares, and the between-group sum of squares are defined as follows:

$$\sum_{i=1}^n (x_i - \bar{x})^2 = \sum_{j=1}^2 \sum_{i \in t_j} (x_i - \bar{x}_j)^2 + \sum_{j=1}^2 n_j (\bar{x}_j - \bar{x})^2 \quad (11)$$

where $\bar{x}$ is the grand mean of the input variable x , t_j represents child nodes, $\bar{x}_j$ is the mean of values within the j -th child node, and n_j is the number of observations in the j -th child node. In this equation, the left-hand side is the total sum of squares, the first term of the right-hand side is the within group sum of squares, and the last term is the between group sum of squares. In addition to this, the between group sum of squares is divided by the total sum of squares because the scale of other measures ranges within the interval $[0,1]$.

3.2. Semi-supervised decision tree

The semi-supervised decision tree differs from typical decision trees in terms of the methods used to find the split point. The traditional decision tree considers the impurity of output classes to determine the best split. However, the semi-supervised decision tree utilizes distributions of input variables as well as labels to reflect the structure of a dataset when splitting internal nodes. To

balance the two different measures, the scale of both measures is restricted to the interval $[0,1]$, and parameter λ is introduced to control the trade-off between the two measures. The overall metric to find the best split is defined as follows:

$$H(x_i) = (1 - \lambda)H_s(x_i) + \lambda H_u(x_i) \quad (12)$$

where H_s and H_u represent impurity measurements, calculated in the supervised manner and inhomogeneity, obtained with the unsupervised method, respectively. The combined metric, $H(x_i)$, is calculated as the weighted sum of two values with the parameter λ . As λ increases, the impact of structural or topological properties of input variables on split also grows. When λ is 0, it is the same as traditional decision trees and when λ is 1, it can be viewed as the unsupervised decision tree to only consider the structure of data.

4. Experiments

4.1. Datasets

In this paper, the semi-supervised decision tree was tested on binary classification problems. Datasets for experiments were collected from the UCI repository [25]. Several datasets with various number of observations, from 1000 to more than 10,000, were considered to check the performance of the proposed algorithm on different data sizes. Datasets consisting of less than 1000 samples were not selected because subspace partitioning is more effective on relatively large datasets. Moreover, small data can result in lack of training samples at terminal nodes, which causes degradation of the final classification models. Finally, six datasets were chosen for experiments:

- **German:** This dataset consists of 1000 observations related to the credit information of people. 300 people are judged as bad credit. It has 20 features. Among them, 13 attributes are categorical.
- **Messidor:** It contains features extracted from the Messidor image set to predict whether an image contains signs of diabetic retinopathy or not. The total 19 attributes represent either a detected lesion, a descriptive feature of an anatomical part or an image-level descriptor. Among them, 3 attributes are categorical. The number of observations with signs of diabetic retinopathy is 611 out of 1151.
- **Abalone:** The purpose of this dataset is to predict the age of abalone from physical measurement, so it is originally not a classification problem. To convert it to a binary classification problem, age is divided into two groups, class 0 (age ≤ 10) and class 1 (age > 10). The number of samples is 4177 (1447 observations for class 0). This dataset consists of 9 input variables and sex is only nominal variable and remained attributes are continuous.
- **Bank:** This dataset is related to direct marketing campaigns of a Portuguese banking institution. The goal of this set is to determine whether the client has subscribed a term deposit. It consists of 16 attributes and 8 out of them are categorical variables. In the UCI repository, there are two sets such as the full dataset with all samples and the randomly sampled set. Smaller dataset containing 10% of examples (4521) is used in experiments. Among 4521 customers, the number of subscribing customers is 521.
- **Musk:** This dataset describes a set of 102 molecules of which 39 are judged to be musk and the remaining 63 molecules are judged to be non-musk and 166 features describe the shape and conformation of molecules. There is no categorical variable in this dataset. This data is generated by all the low-energy conformations of the molecules, so consists of 6598 conformations. Because of that, more than one rows belong to a molecule.

Table 1
Datasets from the UCI repository.

Dataset	# of objects	# of features	# of categorical features
German	1000	20	13
Messidor	1151	19	3
Abalone	4177	9	1
Bank	4521	16	8
Musk	6598	166	0
Coverttype	26,860	12	2

However, in this paper, classification performance is evaluated based on row-wise regardless of molecules. Among 6598 conformations, 1017 conformations are defined as musk.

- **Coverttype:** The purpose of this data is to predict the forest cover type from cartographic variables only. The actual forest cover type for a given observation was determined by US Forest Service. This dataset consists of 12 input features and 2 out of them are categorical variables such as soil type and wilderness area. Cover type is classified into 7 different categories. Among them, two classes, Aspen (9493 observations) and Douglas-fir (17,367 observations), were selected for experiments.

In Table 1, specifications of each dataset are summarized.

4.2. Experimental setting

Classification accuracy was the metric selected to evaluate the performance of the proposed method and for comparison with other models. Three different types of classifiers were built on each dataset. The first type was a classifier without subspace partitioning (denoted as CLF, abbreviation of *classifier*). Basic classifiers were logistic regression (logistic), lasso logistic regression (Lasso), naïve Bayes classifier (NB), and SVM. The second type was the hybrid model in which each basic classifier is combined with subspace partitioning using a traditional decision tree algorithm (denoted as SPCLF, abbreviation of *subspace partitioned classifier*) and the last type is the hybrid model with subspace partitioning using the proposed method, the semi-supervised decision tree (denoted as SS-SPCLF, abbreviation of *semi-supervised subspace partitioned classifier*).

The minimum number of samples at nodes to be split and the minimum number of samples at child nodes after splitting were set large enough to avoid a fully grown tree whose terminal nodes consist of a few observations. Except for the German dataset, nodes with more than or equal to 200 samples were considered for splitting, and a value larger than or equal to 100 was set for the minimum number of samples at child nodes. In experiments, pairs of the minimum number of samples were considered to be split, and the minimum samples for child nodes were set as (300, 200), (300, 100), and (200, 100).

As described in Section 3.1, the impurity-based and the variance-based metrics were introduced to find the best split. In the impurity-based method, each input value of categorical variables is considered as an independent class. Numerical variables, however, should be discretized first, and four different binning methods were compared through experiments. 10 equally spaced values within [0.1,1.0] were used for λ , the parameter to control the weight of inhomogeneity for deciding the split point.

Cross-validation was applied to obtain a more accurate estimate of model prediction performance. The number of validation sets was set to 5 or 10 depending on the size of a dataset. Datasets with more than 10,000 observations used 10-fold cross-validation. Otherwise, 5-fold cross-validation was applied to ensure enough samples in the validation sets. Parameters of classifiers such as Lasso and SVM were determined by taking parameters of the best classifiers in the training set.

Table 2
Comparison of classification models on German data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.7510	0.7530	0.7480	0.7550
SPCLF	Gini	–	0.7371	0.7530	0.7490	0.7490
	Entropy	–	0.7371	0.7530	0.7510	0.7490
SS-SPCLF	Gini	M1	0.7530	0.7560	0.7510	0.7570
		M2	0.7530	0.7550	0.7510	0.7570
		M3	0.7530	0.7560	0.7510	0.7570
		M4	0.7530	0.7550	0.7510	0.7570
		M5	0.7609	0.7600	0.7580	0.7609
	Entropy	M1	0.7530	0.7639	0.7540	0.7570
		M2	0.7530	0.7580	0.7490	0.7570
		M3	0.7530	0.7560	0.7510	0.7570
		M4	0.7530	0.7550	0.7510	0.7570
		M5	0.7609	0.7639	0.7580	0.7609

Table 3
Comparison of classification models on Messidor data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.7431	0.7439	0.5654	0.6948
SPCLF	Gini	–	0.7359	0.7411	0.5775	0.6771
	Entropy	–	0.7359	0.7359	0.5784	0.6693
SS-SPCLF	Gini	M1	0.7489	0.7671	0.6043	0.6857
		M2	0.7567	0.7688	0.5991	0.6874
		M3	0.7541	0.7541	0.6035	0.6883
		M4	0.7576	0.7654	0.5879	0.7048
		M5	0.7506	0.7550	0.5931	0.6874
	Entropy	M1	0.7463	0.7541	0.5939	0.7143
		M2	0.7481	0.7576	0.5957	0.7065
		M3	0.7515	0.7498	0.5957	0.7082
		M4	0.7472	0.7532	0.5983	0.7030
		M5	0.7506	0.7602	0.5931	0.6874

4.3. Experimental results

Tables 2–7 describe the best cross-validation accuracy of different classification models. Classifiers in SS-SPCLF with the same impurity measure were further divided by methods to calculate inhomogeneity, $H_u(x_i)$, for continuous variables. From M1 to M4, $H_u(x_i)$ was calculated using impurity measure with different binning methods (M1=Sturges', M2=Doane's, M3=Scott's, M4=Freedman–Dianonis') and M5 used the variance-based method.

For every dataset, the semi-supervised subspace partitioning outperformed the supervised subspace partitioning by traditional decision trees regardless of combined classification models and SS-SPCLF shows better performance than CLF. The results can be explained by two points of view. First, the improvement from SPCLF may be induced by relieving the unbalance on binary classes. Decision trees split internal nodes to increase the purity of class distribution. Therefore, one class accounts for the vast majority at terminal nodes. This phenomenon causes the class imbalance problem. Second, the superiority over CLF implies that there are subregions with different class distributions from other parts. Under this condition, the subspace partitioning considering input features can improve classification accuracy.

Another strength of the proposed method is the possibility to reduce computational complexity. Generally, the complexity of learning a classifier depends on the number of observations in a training set. For example, the computational complexity of SVM is $O(n^2)$ for solving the dual form of SVMs objective function when sequential minimal optimization, the widely used algorithm for quadratic programming, is applied and $O(n^2)$ for kernel matrix calculation [26,27]. However, the complexity to construct a decision tree is $O(d \cdot n \log(n))$ [28]. Here, n is the number of observations and d is the number of features. Hence, the hybrid model decreases the total computation time by subspace partitioning,

Table 4
Comparison of classification models on *Abalone* data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.7737	0.7766	0.6616	0.7717
SPCLF	Gini	–	0.7758	0.7337	0.6553	0.7748
	Entropy	–	0.7801	0.7538	0.6543	0.7780
SS-SPCLF	Gini	M1	0.7809	0.7907	0.6742	0.7801
		M2	0.7794	0.7880	0.6689	0.7823
		M3	0.7837	0.7928	0.6799	0.7840
		M4	0.7818	0.7876	0.6749	0.7758
		M5	0.7773	0.7739	0.6694	0.7830
	Entropy	M1	0.7806	0.7911	0.6731	0.7809
		M2	0.7859	0.7914	0.6667	0.7842
		M3	0.7828	0.7885	0.6796	0.7849
		M4	0.7888	0.7844	0.6652	0.7854
		M5	0.7797	0.7754	0.6694	0.7842

Table 5
Comparison of classification models on *Bank* data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.8826	0.8896	0.8661	0.8844
SPCLF	Gini	–	0.8907	0.8806	0.8661	0.8789
	Entropy	–	0.8909	0.8891	0.8661	0.8870
SS-SPCLF	Gini	M1	0.8950	0.9010	0.8661	0.8977
		M2	0.8957	0.9017	0.8667	0.8997
		M3	0.8950	0.9012	0.8661	0.8983
		M4	0.8972	0.9012	0.8661	0.8990
		M5	0.8950	0.9010	0.8661	0.8970
	Entropy	M1	0.8970	0.9039	0.8676	0.8970
		M2	0.8950	0.9032	0.8672	0.8977
		M3	0.8972	0.9032	0.8672	0.8979
		M4	0.8968	0.9025	0.8661	0.8986
		M5	0.8961	0.9012	0.8661	0.8970

Table 6
Comparison of classification models on *Musk* data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.9502	0.9493	0.8346	0.9268
SPCLF	Gini	–	0.9674	0.9652	0.8196	0.9337
	Entropy	–	0.9508	0.9670	0.8135	0.9370
SS-SPCLF	Gini	M1	0.9721	0.9777	0.8949	0.9391
		M2	0.9552	0.9621	0.8596	0.9412
		M3	0.9680	0.9785	0.9023	0.9409
		M4	0.9482	0.9567	0.8176	0.9355
		M5	0.9490	0.9508	0.8346	0.9268
	Entropy	M1	0.9612	0.9661	0.8464	0.9415
		M2	0.9600	0.9696	0.8440	0.9446
		M3	0.9702	0.9771	0.8723	0.9497
		M4	0.9620	0.9721	0.8558	0.9405
		M5	0.9490	0.9500	0.8346	0.9268

Table 7
Comparison of classification models on *Covertypes* data.

Model type	Impurity	Inhomogeneity	Logistic	Lasso	NB	SVM
CLF	–	–	0.9597	0.9596	0.9127	0.9894
SPCLF	Gini	–	0.9585	0.9469	0.9252	0.9912
	Entropy	–	0.9598	0.9610	0.9139	0.9917
SS-SPCLF	Gini	M1	0.9598	0.9805	0.9297	0.9943
		M2	0.9598	0.9808	0.9272	0.9943
		M3	0.9601	0.9802	0.9294	0.9944
		M4	0.9611	0.9814	0.9274	0.9941
		M5	0.9598	0.9780	0.9133	0.9943
	Entropy	M1	0.9649	0.9835	0.9645	0.9958
		M2	0.9642	0.9856	0.9565	0.9970
		M3	0.9627	0.9838	0.9428	0.9958
		M4	0.9633	0.9858	0.9497	0.9962
		M5	0.9627	0.9780	0.9435	0.9934

Table 8
Classification accuracies of single decision tree.

Data	German	Messidor	Abalone	Bank	Musk	Covertypes
Impurity						
Gini	0.6852	0.6364	0.7636	0.8853	0.9444	0.9407
Entropy	0.6952	0.6459	0.7636	0.8888	0.9396	0.9463

which reduces the number of training samples to build classifiers at the terminal nodes. Empirically, evaluation of 10-fold cross-validation using SVM takes about 1.5 min for *Covertypes* data, but the hybrid model took about 35 s when the inhomogeneity metric was calculated by Doane's formula for numerical variables, which requires the most computation.

Table 8 summarizes the classification accuracy of decision trees. SPCLF and SS-SPCLF generally outperformed decision trees. The exception occurred when NB was applied to terminal nodes, which shows that NB degrades classification performance in many datasets. During tree growth, the minimum number of samples to be split into child nodes was not restricted to certain values, unlike decision tree algorithm for subspace partitioning. Therefore, final decision trees in Table 9 usually had larger depth and smaller samples in terminal nodes than trees of SPCLF and SS-SPCLF.

Table 9 shows the parameter settings that achieved the best performance on each dataset. In this table, Min(split) represents the minimum number of samples to be split, and Min(child) is defined as the minimum number of samples in child nodes. Except *German* data, the impurity-based method outperformed the variance-based method for calculation of inhomogeneity metric of continuous variables and setting Min(split) as 300 obtained better performance than others. Among the four binning methods, Doane's formula showed the best performance, which is probably due to the fact that Doane's formula does not assume normality and considers skewness of a distribution. For λ , it would be better to select λ as close to 0 or 1 than as the middle of 0 and 1. Though, there was a little consistency of λ on each data.

5. Conclusion

A novel algorithm for subspace partitioning using a decision tree was proposed in this paper. The main contribution of the research is that the algorithm provides a way to incorporate the distributions of input variables to detect the valley between data-concentrated sub-regions in the input space. The suggested algorithm divides the input space with a tendency to separate different clusters, in addition to the separation of target classes. The proposed algorithm does not solely depend on the assumption that the distribution of a dependent variable is consistent with the distribution of input variables. This postulate is not always true in real world problems.

The semi-supervised decision tree algorithm finds better split points at internal nodes than traditional decision trees, and this was confirmed through experiments using six different datasets. Overall, the performance of the combined classifiers trained at terminal nodes was improved. The experimental results proved that subspace partitioning using the proposed method provides better training sets to build classifiers. The results also support the conclusion that some real world classification problems satisfy the main assumption of the proposed method: that data points may be located in different sub-spaces, and in each subspace, classes are distributed in different ways.

In this paper, the new algorithm was tested with several parameter settings, but consolidated guidelines for parameter selection have not been provided yet. Even though parameter setting does not seem to significantly affect classification performance, the optimal parameter selection can yield an additional

Table 9

The best setting of semi-supervised decision tree on each data.

Data	Classifier	Impurity	Inhomogeneity	Min(split)	Min(child)	λ
German	Logistic	Gini/Entropy	M5	200	100	0.8
	Lasso	Entropy	M5	200	100	0.8
	NB	Gini/Entropy	M5	200	100	0.7
	SVM	Gini/Entropy	M5	200	100	0.8
Messidor	Logistic	Gini	M4	200	100	0.2
	Lasso	Gini	M2	300	200	0.8
	NB	Gini	M2	300	100	0.3
	SVM	Entropy	M1	300	200	0.9
Abalone	Logistic	Entropy	M4	300	100	0.2
	Lasso	Gini	M3	300	200	0.1
	NB	Gini	M2	300	100	0.1
	SVM	Entropy	M4	300	200	0.2
Bank	Logistic	Entropy	M3	300	100	0.2
	Lasso	Entropy	M1	300	100	0.3
	NB	Entropy	M1	300	200	0.1
	SVM	Entropy	M3	300	100	0.3
Musk	Logistic	Entropy	M3	300	200	0.8
	Lasso	Gini	M3	300	200	0.7
	NB	Gini	M3	300	100	0.2
	SVM	Entropy	M3	300	100	0.1
Coverttype	Logistic	Entropy	M3	300	200	0.8
	Lasso	Gini	M3	300	200	0.7
	NB	Gini	M3	300	100	0.2
	SVM	Entropy	M3	300	100	0.1

benefit in the real world. Therefore, a more systematic and theoretical analysis on the semi-supervised decision tree will be performed in the future.

Conflict of interest

None declared.

Acknowledgment

This study was supported by the Research Program funded by the Seoul National University of Science and Technology.

References

- [1] D.R. Cox, The regression analysis of binary sequences (with discussion), *J. R. Stat. Soc. Ser. B* 20 (1958) 215–242.
- [2] S.H. Walker, D.B. Duncan, Estimation of the probability of an event as a function of several independent variables, *Biometrika* 54 (1967) 167–179.
- [3] L. Breiman, J. Friedman, R. Olshen, C. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, CA, 1984.
- [4] J.R. Quinlan, Induction of decision trees, *Mach. Learn.* 1 (1986) 81–106.
- [5] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed., Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.
- [6] C. Cortes, V. Vapnik, Support-vector networks, *Mach. Learn.* 20 (1995) 273–297.
- [7] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, Cambridge, United Kingdom, 2000, URL <https://books.google.co.kr/books?id=B-Y88GdO1yYC>.
- [8] R. Kohavi, Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid, in: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, AAAI Press, Palo Alto, California, 1996, pp. 202–207.
- [9] D. Steinberg, N.S. Cardell, The hybrid CART-Logit model in classification and data mining, in: *The Eighth Annual Advanced Research Techniques Forum*, American Marketing Association, Chicago, Illinois, United States, 1998.
- [10] J.M. Jerez-Aragónés, J.a. Gómez-Ruiz, G. Ramos-Jiménez, J. Muñoz Pérez, E. Alba-Conejo, A combined neural network and decision trees model for prognosis of breast cancer relapse, *Artif. Intell. Med.* 27 (2003) 45–63.
- [11] B.W. Heumann, An object-based classification of mangroves using a hybrid decision tree—support vector machine approach, *Remote Sens.* 3 (2011) 2440–2460.
- [12] C.E. Shannon, A mathematical theory of communication, *Bell Syst. Tech. J.* 27 (1948) 379–423.
- [13] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [14] J.R. Quinlan, Improved use of continuous attributes in C4.5, *J. Artif. Intell. Res.* 4 (1996) 77–90.
- [15] P.M. Kuhnert, K.-A. Do, R. McClure, Combining non-parametric models with logistic regression: an application to motor vehicle injury data, *Comput. Stat. Data Anal.* 34 (2000) 371–386.
- [16] I.K. Sethi, Entropy nets: from decision trees to neural networks, *Proc. IEEE* 78 (1990) 1605–1613.
- [17] A. Wieland, R. Leighton, Geometric analysis of neural network capabilities, in: *International Symposium on Neural Networks*, 1988.
- [18] R.P. Brent, Fast training algorithms for multilayer neural nets, *IEEE Trans. Neural Netw.* 2 (1991) 346–354.
- [19] H.A. Sturges, The choice of a class interval, *J. Am. Stat. Assoc.* 21 (1926) 65–66.
- [20] D.P. Doane, Aesthetic frequency classifications, *Am. Stat.* 30 (1976) 181–183.
- [21] J.R.M. Hosking, L-Moments: analysis and estimation of distributions using linear combinations of order statistics, *J. R. Stat. Soc. Ser. B (Methodol.)* 52 (1990) 105–124.
- [22] Q.J. Wang, Direct sample estimators of l moments, *Water Resour. Res.* 32 (1996) 3617–3619.
- [23] D.W. Scott, On optimal and data-based histograms, *Biometrika* 66 (1979) 605–610.
- [24] D. Freedman, P. Diaconis, On the histogram as a density estimator: L2 theory, *Z. Wahrscheinlichkeitstheorie Verwandte Geb.* 57 (1981) 453–476.
- [25] M. Lichman, *UCI Machine Learning Repository*. URL <http://archive.ics.uci.edu/ml/>, 2013.
- [26] I.W. Tsang, J.T. Kwok, P.-M. Cheung, Core vector machines: fast SVM training on very large data sets, *J. Mach. Learn. Res.* 6 (2005) 363–392.
- [27] L. Bottou, C.-J. Lin, Support vector machine solvers, *Large Scale Kernel Mach.* (2007) 301–320.
- [28] R.C. Barros, A. de Carvalho, A.A. Freitas, *Automatic Design of Decision-Tree Induction Algorithms*, SpringerBriefs in Computer Science, Springer International Publishing, New York City. URL <https://books.google.co.kr/books?id=PFqEBgAAQBAJ>, 2015.

Kyoungok Kim is an Assistant Professor in Seoul National University of Science and Technology. She received the B.S. and the Ph.D. degree from Pohang University of Science and Technology (POSTECH), in 2008 and 2013, respectively. Her research interests include developing several data mining and machine learning algorithms and their applications to business intelligence, text analytics, medical informatics, etc.