Subclass-based semi-random data partitioning for improving sample representativeness

Han Liu[a], Shyi-Ming Chen[b,*], Mihaela Cocea[c]

[a]*School of Computer Science and Informatics, Cardiff University, Queen's Buildings, 5 The Parade, Cardiff, CF24 3AA, United Kingdom*
[b]*Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan*
[c]*School of Computing, University of Portsmouth, Buckingham Building, Lion Terrace, Portsmouth, PO1 3HE, United Kingdom*

Abstract

In machine learning tasks, it is essential for a data set to be partitioned into a training set and a test set in a specific ratio. In this context, the training set is used for learning a model for making predictions on new instances, whereas the test set is used for evaluating the prediction accuracy of a model on new instances. In the context of human learning, a training set can be viewed as learning material that covers knowledge, whereas a test set can be viewed as an exam paper that provides questions for students to answer. In practice, data partitioning has typically been done by randomly selecting 70% instances for training and the rest for testing. In this paper, we argue that random data partitioning is likely to result in the sample representativeness issue, i.e., training and test instances show very dissimilar characteristics leading to the case similar to testing students on material that was not taught. To address the above issue, we propose a subclass-based semi-random data partitioning approach. The experimental results show that the proposed data partitioning approach leads to significant advances in learning performance due to the improvement of sample representativeness.

Keywords: Machine learning, Classification, Data mining, Decision tree

*Corresponding author
Email addresses: liuh48@cardiff.ac.uk (Han Liu), smchen@mail.ntust.edu.tw (Shyi-Ming Chen), mihaela.cocea@port.ac.uk (Mihaela Cocea)

learning, Rule learning, If-then rules

1. Introduction

It has been a very popular strategy to adopt machine learning approaches for the purposes of knowledge discovery (e.g., [11, 39, 43]) and predictive modelling (e.g., [1, 32, 33]), which involves the training and testing stages. The former stage aims at learning a model from data, whereas the latter stage aims at evaluating the confidence/prediction performance by using new data. In order to fulfill the above aims, data partitioning is usually needed to obtain a training set and a test set. For knowledge discovery tasks, the use of a training set aims for discovering new patterns, whereas the use of a test set aims for evaluating the degree to which the discovered patterns can be trusted. For predictive modelling tasks, the use of a training set aims for building a model that can make predictions on new data, whereas the use of a test set aims for evaluating the prediction accuracy of the model. In the rest of this paper, we focus on classification, which is a special type of machine learning tasks.

In practical machine learning [42], it has been the usual practice to partition a data set by randomly selecting 70% of the instances to form the training set and 30% of the instances to form the test set. As argued in [21], random data partitioning is likely to result in the sample representativeness issue, in which the test data show the characteristics very dissimilar to the ones of the training data. In a machine learning task, it is essential to make sure that testing is on something that can be learned from the training data. Otherwise, it would lead to the case similar to testing students on material that was not taught. In this case, it could not only fail to get a good prediction performance but also to make an effective judgment of the learning ability of the chosen algorithm, while in reality the poor performance is due to the sample representativeness rather than to the learning ability of the algorithm.

In this paper, in order to address the sample representativeness issue, we propose a subclass-based semi-random partitioning approach. The contributions

of this paper include the following:

- A new approach of data partitioning is proposed, which leads to an effective increase of the similarity between training data and test data, such that it can be more effective to judge the learning ability of an algorithm.
- We demonstrate a novel application of decision tree learning algorithms in the setting of data partitioning in a semi-random way, i.e., each rule, which is extracted from a decision tree trained on a data set, is used to group instances of a specific class to form a subclass of this class, such that semi-random partitioning of data can be achieved by randomly selecting training and test instances from each subclass separately.
- We compare the proposed approach of semi-random data partitioning with the random data partitioning one as well as another more recent one in terms of the classification performance on various data sets. Our experimental results show that our proposed approach of semi-random data partitioning leads to significant advances in the classification performance due to the improvement of the sample representativeness.

The rest of this paper is organized as follows. A review of existing approaches of data partitioning is provided in Section 2 and some limitations of the approaches are identified as well. In Section 3, we illustrate the proposed approach of semi-random data partitioning and justify its significance in improving the sample representativeness. In Section 4, we make experiments by using 20 UCI data sets [18] and the results are discussed in depth to show the effectiveness of the proposed approach of semi-random data partitioning. The conclusions of this paper are drawn in Section 5 by summarizing the contributions and suggesting some further directions for this research area.

2. Related work

In machine learning, data partitioning can be done in several ways for experimentation, and the most popular ways include training/test partitioning and

cross validation [9, 12, 15]. As discussed in [28], cross validation is considered as a way of measuring the learning ability of an algorithm on a data set, i.e., it can be adopted to judge if an algorithm qualifies to get employed towards learning effectively from this data set leading to a good model. The way of training/test partitioning is usually taken towards learning a highly generalizable model from the training data set and evaluating the model confidence in a proper way by using the test data set. In reality, it is more important to measure effectively the generalizability and the confidence of each model trained on a given data set, towards proper employment of the models for classifying new instances. Therefore, this paper focuses on investigating the way of training/test partitioning.

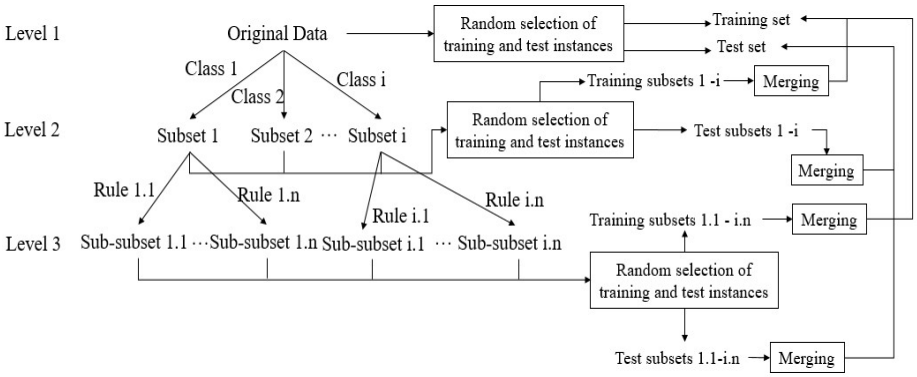

Fig. 1. Three level framework of data partitioning [21]

In terms of training/test partitioning, a three level framework is proposed in [21], as shown in Fig. 1. The three levels are outlined as follows:

1. Level 1: Data is partitioned through the random sampling of a training set D_1 and a test set D_2 separately from the original data set D.
2. Level 2: A number of sub-sets $(S_1, S_2, ..., S_n)$ are obtained through dividing the original data set D, and each subset S_i $(1 \leq i \leq n)$ contains instances that belong to class c_i. Each subset S_i is randomly partitioned into training and test subsets ($S_{i.1}$ and $S_{i.2}$). All the training and test subsets $[(S_{1.1}, S_{1.2}, ..., S_{1.n})$ and $(S_{2.1}, S_{2.2}, ..., S_{2.n})]$ are merged, respectively, for obtaining the final training and test data sets (D_1 and D_2).

3. Level 3: Each (S_i) of the subsets $(S_1, S_2, ..., S_n)$, which is obtained in Level 2, is subdivided into a number of sub-subsets $(T_1, T_2, ..., T_l)$, where each of the sub-subsets T_i contains instances that belong to a subclass of class C_i. Each sub-subset T_i is randomly partitioned into a training sub-subset and a test sub-subset. All the training and test sub-subsets $[(T_{1.1}, T_{1.2}, ..., T_{1.m})$ and $(T_{2.1}, T_{2.2}, ..., T_{2.m})]$ are merged, respectively, for obtaining the final training and test data sets (D_1 and D_2).

The three level framework essentially indicates that the design of the strategies of semi-random data partitioning involved in level 2 and level 3 is aimed at the control of the consistency between the training and test sets in terms of their characteristics, since the strategy involved in level 1 (random data partitioning) has no such control resulting in the class imbalance and sample representativeness issues as mentioned in Section 1. In particular, the strategy, which was proposed in [21] and is involved in level 2, aims at preserving the degree of class balance (frequency distribution among classes) in both training and test data sets. Since this strategy involves dividing the original data set D into several subsets $(S_1, S_2, ..., S_n)$ by putting instances of the same class into the same subset, the strategy is referred to as class-based semi-random data partitioning.

Another strategy of data partitioning, which is referred to as stratified sampling [10, 17, 40], can also achieve preserving the frequency distribution among classes in both training and test data sets. However, for stratified sampling, it is needed to calculate the sampling probability ($P_{training}$ or P_{test}) for each class c_i according to the selected partitioning ratio $(r : 1 - r)$ and the class weight W_{c_i} as defined in Eqs. (1) and (2).

$$P_{training}(class = c_i) = W_{c_i} \cdot r \tag{1}$$

$$P_{test}(class = c_i) = W_{c_i} \cdot (1 - r) \tag{2}$$

A comparison between stratified sampling and the class-based semi-random partitioning approach was made in [21], and the experimental results indicate

that the latter approach outperformed the former one on the majority of the data sets. It is also argued in [21] that the latter approach pays more attention to balancing the training and test data sets in comparison with the former one.

Furthermore, it is very likely that instances of the same class still show high diversity, especially in the era of big data . In other words, separate data partitioning for each class still cannot make sure that the obtained training and test data have highly similar characteristics and thus can result in the sample representativeness issue mentioned in Section 1. Therefore, it is necessary to propose a data partitioning approach in a new semi-random strategy, which needs to be involved in level 3 and deals effectively with the sample representativeness.

3. Subclass-based semi-random data partitioning

In this section, we propose a subclass-based semi-random data partitioning approach, which is involved in level 3 of the partitioning framework shown in Fig. 1. In particular, we illustrate how the concept of decision tree learning can be used for subclass identification. We also justify the significance of our proposed approach in terms of improving the sample representativeness.

3.1. Procedure

The key feature of our proposed data partitioning approach is the heuristic identification of sub-classes through a decision tree learning algorithm, such as ID3 [36] and C4.5 [37]. In particular, a decision tree DT could have more than one leaf node labelled with the same class c_i. In this context, each of these branches ending with the same leaf node (labelled c_i) could be converted into an if-then rule, which is considered to cover a collection of instances that form a subclass $c_{i.j}$ of c_i. The whole procedure of this partitioning approach is described as follows:

- **Step 1**: Train a decision tree DT on the original data set D;
- **Step 2**: For each class c_i, extracts a number (n) of rules $rule_{c_i}[n]$;

- **Step 3**: For each rule $rule_{c_i}[j]$, finds all its covered instances that form a subclass $c_{i.j}$ and groups these instances in a subset $T_{i.j}$ of D;
- **Step 4**: Training and test instances are randomly sampled from $T_{i.j}$ in a specific ratio $r : 1 - r$.

Table 1
Weather data set [23].

Outlook	Temperature	Humidity	Windy	Play?
1	1	1	0	N
1	1	1	1	N
2	1	1	0	Y
3	2	1	0	Y
3	3	2	0	Y
3	3	2	1	N
2	3	2	1	Y
1	2	1	0	N
1	3	2	0	Y
3	2	2	0	Y
1	2	2	1	Y
2	2	1	1	Y
2	1	2	0	Y
3	2	1	1	N

We use the weather data set [1] as an example for illustrating the above four steps by using the ID3 algorithm for subclass identification. The data set is shown in Table 1 and the original attribute values and class labels are replaced with specific symbols.

[1] `http://storm.cis.fordham.edu/~gweiss/data-mining/weka-data/weather.nominal.arff`

As described in [27], the aim of decision tree learning is at recursive attribute selection to obtain the root node and each internal node until the leaf node of each tree branch is obtained, as shown in Figure 2.

Input : A set of training instances;
Output: A decision tree *DT*

```
if the stopping criterion is satisfied then
    create a leaf that corresponds to all remaining training instances
end
else
    choose the best (according to some heuristics) attribute A_x
    label the current node with A_x
    for each value v_{x.j} of attribute A_x do
        label an outgoing edge with value v_{x.j}
        recursively build a subtree based on a subset of training instances
         that meet the condition 'A_x = v_{x.j}'
    end
end
```

Fig. 2. Decision tree learning algorithm [16]

The ID3 algorithm is designed to use the information entropy defined in Eq. (3) and Eq. (4) for attribute selection, i.e., the attribute that obtains the minimum entropy is selected, shown as follows.

$$CS(A_x = v_{x.j}) = -\sum_{i=0}^{c} p(class_i|A_x = v_{x.j}) \log_2 p(class_i|A_x = v_{x.j}) \quad (3)$$

where A_x is an attribute with the index x, $v_{x.j}$ is a value with the index j of the attribute A_x, and $p(class_i|A_x = v_{x.j})$ is the conditional probability of

classifying an instance to $class_i$ given that $A_x = v_{x.j}$.

$$Entropy(A_x) = \sum_{j=1}^{k} w_{x.j} CS(A_x = v_{x.j}) \tag{4}$$

where k is the number of values for attribute A_x and $w_{x.j}$ represents the weight of attribute value $v_{x.j}$. A more detailed illustration of the ID3 algorithm can be found in [26].

In the following, we illustrate the procedure of attribute selection for the root node of a decision tree using the Weather data set shown in Table 1. In particular, a frequency table needs to be created for each of the four attributes, namely, 'Outlook', 'Temperature', 'Humidity' and 'Windy'. The four frequency tables are shown in Tables 2-5.

Table 2

Frequency table for the attribute 'Outlook'

Class label	Outlook= 1	Outlook= 2	Outlook= 3
Y	2	4	3
N	3	0	2
Total	5	4	5

According to Table 2, we can calculate the average entropy $Entropy(outlook)$ of the attribute 'Outlook' as follows:

$Entropy(Outlook) = \frac{5}{14} CS(Outlook = 1) + \frac{4}{14} CS(Outlook = 2) + \frac{5}{14} CS(Outlook = 3) = \frac{5}{14} \times (-\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5}) + \frac{2}{7} \times (-\frac{4}{4} \log_2 \frac{4}{4} - 0) + \frac{5}{14} \times (-\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5}) = 0.69$

Table 3

Frequency table for the attribute 'Temperature'

Class label	Temperature= 1	Temperature= 2	Temperature= 3
Y	2	4	3
N	2	2	1
Total	4	6	4

According to Table 3, we can calculate the average entropy $Entropy(Temperature)$ of the attribute 'Temperature' as follows:

$Entropy(Temperature) = \frac{4}{14}CS(Temperature = 1) + \frac{6}{14}CS(Temperature = 2) + \frac{4}{14}CS(Temperature = 3)$=$\frac{2}{7} \times (-\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4})+\frac{3}{7} \times (-\frac{4}{6}\log_2\frac{4}{6} - \frac{2}{6}\log_2\frac{2}{6})+ \frac{2}{7} \times (-\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4})$=1.39

Table 4

Frequency table for the attribute 'Humidity'

Class label	Humidity= 1	Humidity= 2
Y	3	6
N	4	1
Total	7	7

According to Table 4, we can calculate the average entropy $Entropy(Humidity)$ of the attribute 'Humidity' as follows:

$Entropy(Humidity) = \frac{7}{14}CS(Humidity = 1) + \frac{7}{14}CS(Humidity = 2)$=$\frac{1}{2} \times (-\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7})+\frac{1}{2} \times (-\frac{6}{7}log_2\frac{6}{7} - \frac{1}{7}log_2\frac{1}{7})$=0.79

Table 5

Frequency table for the attribute 'Windy'

Class label	Windy= 1	Windy= 0
Y	3	6
N	3	2
Total	6	8

According to Table 5, we can calculate the average entropy $Entropy(Windy)$ of the attribute 'Windy' as follows:

$Entropy(Windy) = \frac{6}{14}CS(Windy = 1) + \frac{8}{14}CS(Windy = 0)$=$\frac{3}{7} \times (-\frac{3}{6}log_2\frac{3}{6} - \frac{3}{6}log_2\frac{3}{6})+\frac{4}{7} \times (-\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4})$=0.89

Because the attribute 'Outlook' obtains the minimum average entropy (i.e., 0.69), the attribute 'Outlook' is selected for the root node leading to an incom-

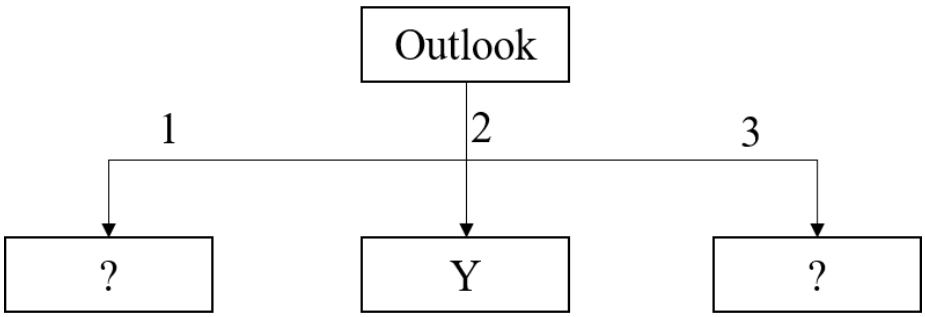

Fig. 3. Incomplete decision tree trained on the weather data set

plete decision tree as illustrated in Fig. 3. The attribute 'Outlook' has three values, namely, '1', '2' and '3', so the root node leads to three branches as judgment criteria. The conditional entropy of the attribute-value pair 'Outlook=2' is 0, i.e., all instances that meet this condition belong to the 'Y' class, as shown in Table 1, so this branch ends up with a leaf node labelled 'Y'. However, the conditional entropy values for the other two attribute value pairs are not 0, so each of the corresponding branches needs to be grown further by repeating the attribute selection procedure on the basis of the other three attributes, i.e., attribute selection needs to be done further for each of the two nodes labelled with the symbol '?', until each branch covers instances of a single class.

On the basis of the above description, following **Step 1**, a complete decision tree can be trained on the weather data set, as shown in Fig. 4.

Following **Step 2**, we can obtain three rules for the class 'Y' and two rules for the class 'N', shown as follows:

- Rule 1: $Outlook = 2 \rightarrow class = Y$,
- Rule 2: $Outlook = 1 \wedge Humidity = 2 \rightarrow class = Y$,
- Rule 3: $Outlook = 3 \wedge Windy = 0 \rightarrow class = Y$,
- Rule 4: $Outlook = 1 \wedge Humidity = 1 \rightarrow class = N$,
- Rule 5: $Outlook = 3 \wedge Windy = 1 \rightarrow class = N$.

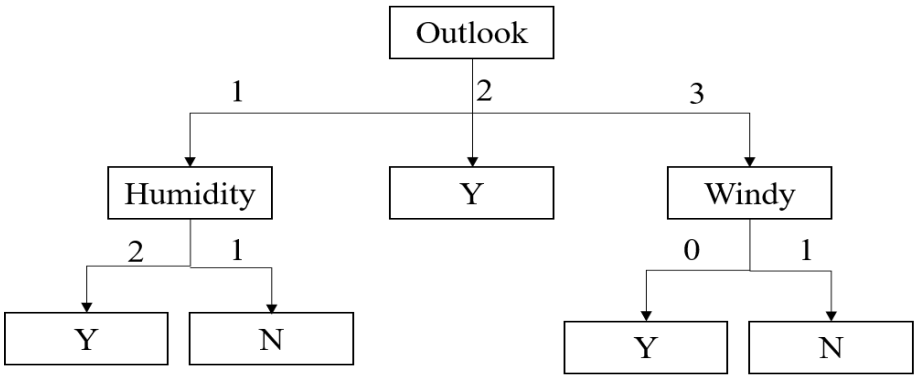

Fig. 4. Complete decision tree trained on the weather data set.

Table 6

Subset $T_{1.1}$ of the weather data set.

Outlook	Temperature	Humidity	Windy	Play?
2	1	1	0	Y
2	3	2	1	Y
2	2	1	1	Y
2	1	2	0	Y

Table 7

Subset $T_{1.2}$ of the weather data set.

Outlook	Temperature	Humidity	Windy	Play?
1	3	2	0	Y
1	2	2	1	Y

Following **Step 3**, Rule 1, Rule 2, Rule 3 cover three subsets of instances shown in Tables 6, 7 and 8, respectively, which form three sub-classes of the class 'Y'. Moreover, Rule 4 and Rule 5 cover two subsets of instances shown in Tables 9 and 10, which form two sub-classes of the class 'N'.

Following **Step 4**, five training subsets and five test subsets are obtained through random partitioning of the five subsets shown in Tables 6-10. The final

Table 8

Subset $T_{1.3}$ of Weather Dataset

Outlook	Temperature	Humidity	Windy	Play?
3	2	1	0	Y
3	3	2	0	Y
3	2	2	0	Y

Table 9

Subset $T_{2.1}$ of the weather data set

Outlook	Temperature	Humidity	Windy	Play?
1	1	1	0	N
1	1	1	1	N
1	2	1	0	N

Table 10

Subset $T_{2.2}$ of the weather data set

Outlook	Temperature	Humidity	Windy	Play?
3	3	2	1	N
3	2	1	1	N

training set is obtained by merging all the five training subsets and the final test set is obtained in the same way.

The above example shows that each of the five subsets is made up of instances of the same class. Therefore, Step 4 is simply taken for drawing the training and test data sets. However, in reality, it is not always the case (Case 1) that each subset consists of instances of a single class, i.e., a subset may contain instances that belong to different classes, which could happen from the following cases:

- Case 2: In decision tree learning, a branch of the tree has reached its maximum length, which means that all the attributes have been selected for growing this branch. Unfortunately, this branch still covers a subset

of instances of more than one class.

- Case 3: Since attribute selection involved in the procedure of decision tree learning is essentially the partitioning of a larger training subset into several smaller subsets, the partitioning may result in a smaller subset that does not contain any attributes that can lead to reduction of the average uncertainty in distinguishing instances of different classes, while the smaller subset still contains instances of different classes.

In both Case 2 and Case 3, the proposed semi-random data partitioning approach is designed to partition each subset (containing instances of different classes) into sub-subsets further to Step 4, such that each sub-subset contains instances that belong to the same class, prior to instances selection for drawing the training and test data sets.

3.2. Mathematical justification

As proved mathematically and experimentally in [21], when a data set contains n classes with the frequency distribution of p_1, p_2, ... , p_n, the strategy involved in level 2 of the partitioning framework is able to ensure that the above distribution is preserved in the training and test sets resulting from partitioning of the data set. In this sub-section, we prove that the proposed partitioning approach involved in level 3 of the framework is also able to ensure the preservation of the above distribution after data partitioning. In particular, we suppose that the original data set has m instances, such that each class c_i has mp_i instances.

Given that the percentage of the training data set is q, i.e., the percentage of the test data set is $1-q$, following the proposed partitioning approach, the proof of the preservation of the original distribution of the class frequency involves the four steps, shown as follows:

- **Step 1**: After a decision tree is trained, each class c_i is decomposed into r sub-classes; each subclass $c_{i.j}$ of c_i obtains $mp_ip_{i.j}$ instances, i.e., the resulting frequency distribution among the sub-classes of c_i is $p_{i.1}$, $p_{i.2}$, ... , $p_{i.r}$ (their summation is 1).

- **Step 2**: For each subclass $c_{i.j}$, the instances that belong to it are randomly selected into either a training subset or a test subset. The numbers of instances in the training subset and in the test subset are $mp_ip_{i.j}q$ and $mp_ip_{i.j}(1-q)$, respectively.

- **Step 3**: The final training subset is obtained by merging the training subsets resulting from Step 2. The frequency distribution among the sub-classes is $mp_1p_{1.1}q : mp_1p_{1.2}q : ... : mp_1p_{1.r}q : mp_2p_{2.1}q : mp_2p_{2.2}q : ... : mp_2p_{2.r}q : ... : mp_np_{n.1}q : mp_np_{n.2}q : ... : mp_np_{n.r}q$, which is equivalent to $p_1p_{1.1} : p_1p_{1.2} : ... : p_1p_{1.r}: p_2p_{2.1} : p_2p_{2.2} : ... : p_2p_{2.r} : ... : p_np_{n.1} : p_np_{n.2} : ... : p_np_{n.r}$. Furthermore, the above distribution can be rearranged to $p_1(p_{1.1}+p_{1.2}+...+p_{1.r}) : p_2(p_{2.1}+p_{2.2}+...+p_{2.r}) : ... : p_n(p_{n.1}+p_{n.2}+...+p_{n.r})$. Because the summation of $p_{i.1}$, $p_{i.2}$, ... , $p_{i.r}$ is 1, the above distribution can be simplified to $p_1 : p_2 : ... : p_n$, i.e., the original distribution.

- **Step 4**: The final test subset is obtained by merging the test subsets resulting from **Step 2**. The frequency distribution among the sub-classes is $mp_1p_{1.1}(1-q) : mp_1p_{1.2}(1-q) : ... : mp_1p_{1.r}(1-q) : mp_2p_{2.1}(1-q) : mp_2p_{2.2}(1-q) : ... : mp_2p_{2.r}(1-q) : ... : mp_np_{n.1}(1-q) : mp_np_{n.2}(1-q) : ... : mp_np_{n.r}(1-q)$, which is equivalent to $p_1p_{1.1} : p_1p_{1.2} : ... : p_1p_{1.r} : p_2p_{2.1} : p_2p_{2.2} : ... : p_2p_{2.r} : ... : p_np_{n.1} : p_np_{n.2} : ... : p_np_{n.r}$. Furthermore, the above distribution can be rearranged to $p_1(p_{1.1}+p_{1.2}+...+p_{1.r}) : p_2(p_{2.1}+p_{2.2}+...+p_{2.r}) : ... : p_n(p_{n.1}+p_{n.2}+...+p_{n.r})$. Because the summation of $p_{i.1}$, $p_{i.2}$, ... , $p_{i.r}$ is 1, the above distribution can be simplified to $p_1 : p_2 : ... : p_n$, i.e., the original distribution.

According to the above proof, we can also see that the proposed partitioning approach not only preserves in both the training and test data sets the frequency distribution $(p_1 : p_2 : ... : p_n)$ among the classes $(c_1, c_2, ..., c_n)$ but also the frequency distribution $(p_1p_{1.1} : p_1p_{1.2} : ... : p_1p_{1.r} : p_2p_{2.1} : p_2p_{2.2} : ... : p_2p_{2.r} : ... : p_np_{n.1} : p_np_{n.2} : ... : p_np_{n.r})$ among the sub-classes of each class c_i.

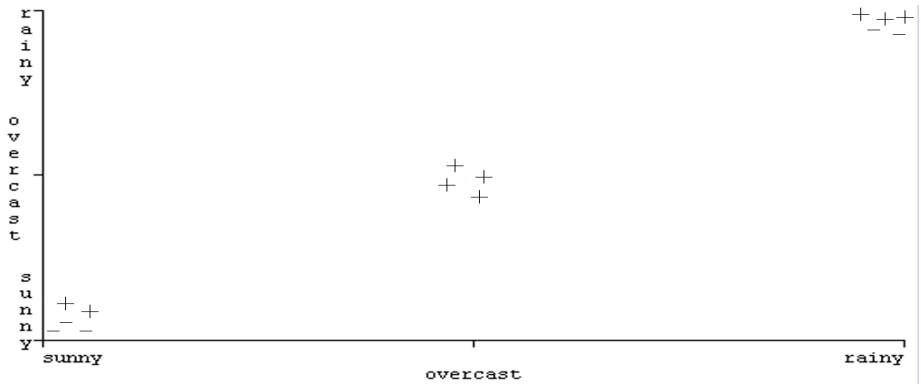

Fig. 5. Visualization of the weather data set.

In practice, preserving in both the training and test data sets the frequency distribution among sub-classes of each class is an effective way of managing the sample representativeness. As mentioned in Section 1, instances of the same class could still be highly diverse, which indicates that only the preservation of the frequency distribution among classes is not sufficient to address the sample representativeness issue, since it is still possible that the instances of class c_i in the training data set are very dissimilar to the ones of the same class in the test data set. For example, as shown in Fig. 5, the visualization result of the Weather data set indicates that the instances of the 'Y' class (in the "+" sign) are distributed in three different areas. Also, the instances of the 'N' class (in the "−" sign) are distributed in two different areas.

In the above context, if the instances belong to the same class but their distribution is sparse, it would be likely to result in the sample representativeness issue if the strategies involved in level 1 and level 2 of the partitioning framework are adopted. For example, according to Fig. 5, the patterns learned from the four instances of the class 'Y' (four "+" points located in the central area) can not be used effectively to classify other instances of the same class (the "+" points located in the top right or bottom left corner). Therefore, it is necessary to identify sub-classes for each class towards addressing the above issue of sample representativeness.

From the perspective of decision tree learning, the instances (of class c_i) covered by the same rule have common characteristics. Thus, learning from these instances is more likely to lead to patterns that are representative of all these instances, in comparison with learning from instances (of the same class) covered by other rules. Furthermore, the nature of decision tree learning guarantees that different branches of a tree do not cover common instances, i.e., these rules cover disjoint subsets of instances, which indicates that the sub-classes identified through the rules resulting from decision tree learning are not overlapping. The above argumentation indicates that decision tree learning would be considered as an effective way of subclass identification leading to the improvement of the sample representativeness.

On the other hand, it is possible that decision tree learning leads to the generation of only one rule for each class. For example, as reported in [23], the 'contact lenses' data set [2] contains three classes, namely, 'hard lenses', 'soft lenses' and 'no lenses', and three rules are learned from this data set (one rule for each class). In this case, the proposed approach and the class-based semi-random partitioning approach would make no difference in terms of the partitioning strategy, i.e., both approaches lead to random data sampling from each class of instances for forming the training and test sets. If the above case occurs in practice, the proposed approach of data partitioning would not really lead to considerable advances in the classification performance, in comparison with the class-based semi-random partitioning approach.

4. Experimental results

In this section, we make experiments for evaluation of the proposed approach of subclass-based semi-random partitioning by comparing it with the random partitioning one and the class-based semi-random partitioning one in terms of their impacts on classification accuracy and standard deviation.

In terms of classification accuracy, each data set is partitioned into a training set and a test set in the ratio of 70:30 for conduct of the experiments. The data

partitioning is repeated 10 times for each data set and the average accuracy is taken for comparative evaluation. We also report the standard deviation of the classification accuracy obtained on each data set over the 10 runs.

The above procedures are followed by using C4.5 [37], Naive Bayes (NB) [38] and K Nearest Neighbours (KNN) [45] (k=5), respectively, for training classifiers on 20 UCI data sets [18], towards testing the performance of the proposed approach of data partitioning. We choose the above three learning algorithms for classifiers training, since these algorithms are all very popular in real applications and are sensitive to the changes in the training sample leading to negative impacts on their performance. In other words, the use of C4.5, NB and KNN for classifiers training would lead to the effective evaluation of the impact of each data partitioning approach on classification accuracy and standard deviation. The proposed approach of subclass-based semi-random partitioning is implemented by using C4.5 for subclass identification as part of the procedure, due to the presence of continuous attributes in some of the 20 data sets. The 20 data sets are described in Table 11 in terms of the data characteristics.

For the 20 data sets used in our experiments, some of them contain missing values. All the missing values are replaced with the majority value (the most frequently occurring one) for each discrete attribute, or with the average of the values in the entire domain for each continuous attribute. Also, for the 'Hypothyroid' and 'Sick' data sets, the last second (28th) attribute 'TBG' is deleted since the entire domain of this attribute is full of missing values.

In Tables 12 and 14, C4.5 I, NB I and KNN I represent that the three algorithms are used to learn classifiers from training data obtained through random data partitioning; C4.5 II, NB II and KNN II represent that the class-based approach of semi-random data partitioning is adopted to obtain the training data for the three algorithms to learn classifiers; C4.5 III, NB III and KNN III represent that the three algorithms are used to learn classifiers when subclass-based approach of semi-random data partitioning is adopted to obtain the training and test data. The results on the average accuracy of classification are shown in Table 12, whereas the results on the standard deviation of the classification

Table 11
Data sets

Data sets	Attribute types	Number of attributes	Number of instances	Number of classes
Audiology	discrete	69	226	24
Autos	discrete, continuous	25	205	7
Breast-w	continuous	9	699	2
Colic	discrete, continuous	22	368	2
Diabetes	discrete, continuous	20	768	2
Dermatology	discrete, continuous	34	366	6
Ecoli	continuous	7	336	8
Haberman	continuous	3	306	2
Heart-c	discrete, continuous	13	303	5
Heart-h	discrete, continuous	13	294	5
Heart-stalog	continuous	13	270	2
Hypothyroid	discrete, continuous	29	3772	4
Iris	continuous	4	150	3
Kr-vs-kp	discrete	36	3196	2
Lymph	discrete, continuous	18	148	4
Sick	discrete, continuous	29	3772	2
Soybean	discrete	35	683	19
Spambase	continuous	57	4601	2
Vowel	discrete, continuous	13	990	11
Vote	discrete	16	435	2

accuracy are shown in Table 14.

From Table 12, we can see that the results indicate that in most cases, the proposed subclass-based semi-random (SSR) data partitioning approach leads to considerable advances in the average accuracy of classification, in comparison with the random (R) data partitioning approach and class-based semi-random

Table 12

Average accuracy of classification

Data sets	C4.5 I	C4.5 II	C4.5 III	NB I	NB II	NB III	KNN I	KNN II	KNN III
Audiology	0.62	0.76	**0.83**	0.44	0.67	**0.73**	0.39	0.56	**0.64**
Autos	0.54	0.75	**0.88**	0.46	0.56	**0.67**	0.45	0.58	**0.68**
Breast-w	0.93	**0.95**	**0.95**	0.94	**0.96**	**0.96**	0.96	**0.97**	**0.97**
Colic	0.73	0.83	**0.87**	0.72	0.77	**0.79**	0.67	0.79	**0.82**
Dermatology	0.76	0.93	**0.94**	0.79	0.98	**0.99**	0.82	0.96	**0.97**
Diabetes	0.70	0.73	**0.75**	0.71	**0.76**	**0.76**	0.71	0.73	**0.75**
Ecoli	0.66	0.80	**0.84**	0.72	0.83	**0.87**	0.72	0.84	**0.89**
Haberman	**0.75**	0.72	0.71	**0.77**	0.74	0.75	0.71	0.71	**0.73**
Heart-c	**0.78**	0.76	0.77	0.81	0.81	**0.84**	0.79	**0.82**	**0.82**
Heart-h	0.74	0.78	**0.81**	0.82	**0.84**	**0.84**	**0.84**	0.82	**0.84**
Heart-stalog	0.71	0.78	**0.81**	0.81	0.84	**0.87**	0.76	0.79	**0.82**
Hypothyroid	**0.99**	**0.99**	**0.99**	**0.95**	**0.95**	**0.95**	**0.93**	**0.93**	**0.93**
Iris	0.83	**0.95**	0.94	0.85	0.94	**0.95**	0.84	0.95	**0.96**
Kr-vs-kp	0.98	**0.99**	**0.99**	0.87	0.87	**0.88**	0.90	**0.96**	**0.96**
Lymph	0.71	0.77	**0.83**	0.76	0.80	**0.81**	0.74	0.80	**0.86**
Sick	0.97	**0.99**	**0.99**	0.92	0.92	**0.93**	0.95	**0.96**	**0.96**
Soybean	0.69	0.92	**0.93**	0.74	0.92	**0.93**	0.61	0.91	**0.92**
Spambase	0.89	0.92	**0.93**	0.79	**0.80**	**0.80**	0.86	0.89	**0.90**
Vowel	0.50	0.75	**0.80**	0.48	0.61	**0.67**	0.31	0.82	**0.89**
Vote	0.95	0.95	**0.96**	0.89	0.90	**0.91**	0.90	0.92	**0.94**

(CSR) data partitioning approach.

In particular, when C4.5 is used for training classifiers (see columns 2-4 in Table 12), the SSR approach outperforms the other two approaches in 13 out of 20 cases. For the other 7 cases, SSR still performs the best but the same as one or both of the others in 4 cases. In the remaining 3 cases, SSR performs marginally worse than R or CSR. When NB is used for training classifiers (see columns 5-7

in Table 12), the SSR approach outperforms the other two approaches in 14 out of 20 cases. For the other 6 cases, SSR still performs the best but the same as one or both of the others in 5 cases. For the remaining case, SSR performs marginally worse than R. When KNN is used for training classifiers (see columns 8-10 in Table 12), the SSR approach outperforms the other two approaches in 14 out of 20 cases. For the other 6 cases, SSR still performs the best but the same as one or both of the others.

To further investigate the performance of SSR, statistical analysis through the Wilcoxon sign rank test is conducted to investigate if SSR leads to higher average classification accuracy in comparison with R and CSR, respectively. It has been shown in [8] that the use of the Wilcoxon sign rank test is more appropriate than the use of the paired t-test.

In general, the comparison between 2 methods across several data sets is allowed through using the Wilcoxon sign rank test [8]. If the performance of the two classifiers is not significantly different, it will generally indicate the case of null hypothesis. The differences in performance between different methods need to be calculated and ranked, such that two sums of ranks can be calculated, respectively, for the positive differences and the negative differences. The distribution of the signs (positive and negative) over the ranks is compared and the significance level is calculated.

The details of the Wilcoxon's test are displayed in Table 13 to show the comparison between each pair of classifiers trained on the 20 data sets in our experiments, e.g. 'C4.5: R vs SSR' indicates that C4.5 is used to learn two classifiers from two training samples obtained, respectively, by adopting the R and SSR data partitioning approaches, and the performance of the two learned classifiers is compared.

The number (N) of positive ranks indicates the number of cases that one approach outperforms the other one in terms of the average accuracy of classification, whereas the number of negative ranks indicates the number of the opposite cases. The sums of ranks (i.e., mean ranks $\times$ N) indicate the relative ranking of the compared classifiers. When the sums of ranks for the positive

Table 13

Wilcoxon sign rank tests for average accuracy.

Compared classifiers	Ranks	N	Mean ranks	Sum of ranks	z-score	p-value
C4.5: R vs SSR	Negative ranks	17	10.68	181.50	-3.48	p=0%
	Positive ranks	2	4.25	8.50		
	Ties	1				
	Total	20				
C4.5: CSR vs SSR	Negative ranks	14	9.07	127.00	-3.05	p=0.10%
	Positive ranks	2	4.50	9.00		
	Ties	4				
	Total	20				
NB: R vs SSR	Negative ranks	18	10.25	184.50	-3.60	p=0%
	Positive ranks	1	5.50	5.50		
	Ties	1				
	Total	20				
NB: CSR vs SSR	Negative ranks	15	8.00	120.00	-3.41	p=0%
	Positive ranks	0	0	0		
	Ties	5				
	Total	20				
KNN: R vs SSR	Negative ranks	18	9.50	171.00	-3.70	p=0%
	Positive ranks	0	0	0		
	Ties	2				
	Total	20				
KNN: CSR vs SSR	Negative ranks	15	8.00	120.00	-3.41	p=0%
	Positive ranks	0	0	0		
	Ties	5				
	Total	20				

and negative ranks are very close in value, the difference between the compared classifiers is negligible, i.e., they have the similar performance. When one of

the sums of ranks (either positive or negative) is considerably higher than the other, it is an indicator of a considerable difference; the level of significance of the difference is calculated using the z-score, based on the results on positive and negative ranks; The difference is significant if the p-value is less than 0.05.

For example, when comparing C4.5 using random partitioning (C4.5 R) with C4.5 using subclass-based semi-random partitioning (C4.5 SSR), the sum of negative ranks, i.e., 181.5, (indicating that the performance of C4.5 R is worse than the performance of C4.5 SSR) is much higher than the sum of positive ranks, i.e., 8.5, (indicating that the performance of C4.5 R is better than the performance of C4.5 SSR). The z-score (-3.48) and it's corresponding p-value (0%) indicate that the C4.5 SSR performs significantly better than C4.5 R.

The results obtained through Wilcoxon rank test indicate the adequate condition of rejecting the null hypothesis is met, which means that the average accuracy obtained by adopting the SSR data partitioning approach is higher than the accuracy obtained by adopting each of the other approaches and that it is unlikely to get this difference by chance. In other words, the SSR data partitioning approach performs significantly better than the R and CSR approaches in terms of average accuracy.

From Table 14, we can see that in most cases, the proposed SSR data partitioning approach leads to smaller values of the standard deviation, in comparison with the R and CSR data partitioning approaches. As the partitioning process is repeated for each run, the standard deviation is an indicator of the influence of the partitioning process on the performance; a low standard deviation is an indicator of the consistency for the performance of the used learning algorithms (C4.5, NB and KNN) across the 10 runs.

In particular, when C4.5 is used for training classifiers (see columns 2-4 in Table 14), the SSR approach outperforms the other two approaches in 17 out of 20 cases. For the remaining 3 cases, the SSR approach performs marginally worse than the CSR approach. When NB is used for training classifiers (see columns 5-7 in Table 14), the SSR approach outperforms the other two approaches in 18 out of 20 cases. For the remaining 2 cases, the SSR approach

Table 14
Standard deviation

Data sets	C4.5 I	C4.5 II	C4.5 III	NB I	NB II	NB III	KNN I	KNN II	KNN III
Audiology	13.90%	2.32%	**1.09%**	13.63%	4.25%	**1.61%**	18.03%	3.91%	**2.25%**
Autos	24.55%	8.80%	**3.68%**	21.80%	6.07%	**2.97%**	19.92%	5.40%	**2.01%**
Breast-w	4.49%	**1.21%**	1.29%	4.40%	**0.85%**	1.96%	5.17%	1.37%	**0.75%**
Colic	18.49%	3.27%	**2.29%**	19.20%	4.38%	**3.10%**	8.31%	2.40%	**2.22%**
Dermatology	18.23%	2.52%	**1.63%**	22.63%	1.27%	**0.80%**	13.19%	1.62%	**1.12%**
Diabetes	11.71%	**2.44%**	2.59%	8.82%	2.81%	**1.43%**	8.21%	2.49%	**1.84%**
Ecoli	23.73%	**3.87%**	3.91%	21.16%	2.65%	**2.07%**	18.62%	2.68%	**1.96%**
Haberman	11.31%	4.34%	**1.85%**	7.30%	2.80%	**0.98%**	10.15%	4.04%	**2.80%**
Heart-c	12.00%	5.09%	**2.76%**	7.72%	4.06%	**2.53%**	9.76%	3.34%	**2.47%**
Heart-h	12.75%	4.11%	**2.34%**	8.78%	4.45%	**1.69%**	9.95%	3.24%	**2.02%**
Heart-stalog	6.33%	4.35%	**2.98%**	12.54%	5.63%	**2.82%**	10.74%	4.64%	**2.66%**
Hypothyroid	0.91%	0.20%	**0.15%**	2.19%	**0.22%**	0.31%	1.39%	**0.29%**	0.34%
Iris	18.33%	3.45%	**2.78%**	16.62%	2.85%	**1.63%**	18.83%	2.95%	**1.52%**
Kr-vs-kp	1.97%	0.27%	**0.19%**	4.04%	1.25%	**1.04%**	5.10%	0.75%	**0.58%**
Lymph	21.07%	5.28%	**2.61%**	12.83%	4.48%	**2.38%**	14.53%	6.33%	**2.33%**
Sick	1.38%	0.56%	**0.24%**	2.85%	1.42%	**0.65%**	1.58%	0.54%	**0.23%**
Soybean	22.64%	1.92%	**0.62%**	15.35%	1.55%	**0.63%**	21.19%	1.18%	**0.99%**
Spambase	4.98%	0.82%	**0.37%**	3.04%	1.09%	**0.58%**	2.30%	0.59%	**0.49%**
Vowel	14.31%	3.40%	**2.62%**	13.32%	3.34%	**1.86%**	22.05%	3.62%	**2.07%**
Vote	5.27%	2.01%	**1.21%**	5.40%	2.09%	**1.54%**	5.83%	1.59%	**1.35%**

performs marginally worse than the CSR approach. When KNN is used for training classifiers (see columns 8-10 in Table 14), the SSR approach outperforms the other two approaches in 19 out of the 20 cases. For the remaining case, the SSR approach performs marginally worse than the CSR approach.

In order to investigate in more depth the performance of the SSR approach, statistical analysis is conducted again through the Wilcoxon sign rank test to

investigate if SSR leads to a lower standard deviation in comparison with R and CSR, respectively. The results are shown in Table 15. In this table, the number (N) of positive ranks indicates the number of cases that one approach shows a higher standard deviation than the other one, whereas the number of negative ranks indicates the number of the opposite cases. For example, when comparing the R and SSR data partitioning approaches in terms of their impacts on the standard deviation of the classification performance of C4.5, i.e., C4.5: R vs SSR, the results show that the number of negative ranks is 0 and the number of positive ranks is 20, which indicate that the standard deviation obtained by using the R approach is higher than the one obtained by using the proposed SSR approach in all the 20 cases.

The Wilcoxon rank test results shown in Table 15 indicate the adequate condition of rejecting the null hypothesis is fulfilled, which means that the standard deviation obtained by adopting the proposed SSR data partitioning approach is lower than the standard deviation obtained by adopting each of the other two approaches and that it is unlikely to get this difference by chance. In other words, the SSR data partitioning approach performs significantly better than the R and CSR data partitioning approaches in terms of standard deviation.

Through looking at the results shown in Tables 12-15, we can see that the proposed SSR approach leads to significant improvements of the average accuracy and considerable reduction of the standard deviation in comparison with the R and CSR approaches. The results also show that in some cases SSR and CSR perform very similarly or even the same, e.g., on the 'Dermatology' and 'Diabetes' data sets. The above phenomenon could be partially explained by the argumentation made in Section 3.2 that if decision tree learning leads to only one rule for each class then SSR and CSR would make no difference in terms of the partitioning strategy and thus lead to very similar or even the same accuracy of classification. Furthermore, there are also some cases that the three data partitioning approaches (R, CSR and SSR) all perform very similarly or even the same, e.g., on the 'Breast-w', 'Hypothyroid' and 'Sick' data sets. The above phenomenon is very likely due to the case that the original data set is

Table 15

Wilcoxon sign rank tests for standard deviation

Compared classifiers	Ranks	N	Mean ranks	Sum of ranks	z-score	p-value
C4.5: R vs SSR	Negative ranks	0	0	0	3.92	p=0%
	Positive ranks	20	10.50	210.00		
	Ties	0				
	Total	20				
C4.5: CSR vs SSR	Negative ranks	3	3.33	10.00	3.55	p=0%
	Positive ranks	17	11.77	200.00		
	Ties	0				
	Total	20				
NB: R vs SSR	Negative ranks	0	0	0	3.92	p=0%
	Positive ranks	20	10.50	210.00		
	Ties	0				
	Total	20				
NB: CSR vs SSR	Negative ranks	2	5.00	10.00	3.55	p=0%
	Positive ranks	18	11.11	200.00		
	Ties	0				
	Total	20				
KNN: R vs SSR	Negative ranks	0	0	0	3.92	p=0%
	Positive ranks	20	10.50	210.00		
	Ties	0				
	Total	20				
KNN: CSR vs SSR	Negative ranks	1	1.00	1.00	3.88	p=0%
	Positive ranks	19	11.00	209.00		
	Ties	0				
	Total	20				

already nicely cleaned and sufficiently representative, i.e., for each class in the data set, the instances are located closely and random partitioning is not likely

to lead to negative impacts from breaking the balance level of the training data.

In addition, the results show that the impact of adopting the SSR data partitioning approach in several cases is different when different learning algorithms are used to train classifiers. For example, for the 'Lympth' data set, the SSR approach leads to considerable advances in the classification accuracy when C4.5 and KNN are used for training classifiers, whereas the SSR approach leads to a much smaller improvement of the classification accuracy when NB is used for training classifiers. The above phenomenon is likely due to the case that different learning algorithms have different suitability for training classifiers on the same data set.

In order to analyze in more depth the impact of the proposed semi-random data partitioning approach, we present additional results shown in Table 16. As emphasized in Section 3.1, decision tree learning may lead to three different cases, where Case 1 represents the ideal outcome that a training subset covered by a branch of the trained decision tree contains instances that all belong to the same class; both Case 2 and Case 3 represent unexpected outcomes, which indicate that a training subset covered by a tree branch contains instances that belong to different classes.

From Table 16, we can see that the percentage of Case 1 is higher than 80% on eight data sets, i.e., 'Audiology', 'Colic', 'Dermatology', 'Hypothyroid', 'Iris', 'Lymph', 'Sick' and 'Vote', which indicates that the eight trained decision trees well fit the eight data sets, respectively. According to the results shown in Table 12 on some of the above data sets, e.g., the 'Audiology' and 'Colic' data sets, we can see that the use of the proposed partitioning approach generally results in a considerable improvement of the average classification accuracy, in comparison with the other two partitioning approaches.

Furthermore, the results show again different impacts of the proposed semi-random data partitioning approach on the classification performance of C4.5, NB and KNN on different data sets. For example, on the 'Audiology', 'Colic', 'Lymph' and 'Vowel' data sets, the proposed data partitioning approach leads to a considerable improvement of the average accuracy for at least two of the

Table 16

Distribution among different cases of decision tree learning

Data sets	Percentage of Case 1	Percentage of Case 2	Percentage of Case 3
Audiology	86.15%	0%	13.85%
Autos	65.22%	15.22%	19.56%
Breast-w	70.00%	30.00%	0%
Colic	87.50%	10.00%	2.50%
Dermatology	96.15%	0%	3.85%
Diabetes	44.44%	55.56%	0%
Ecoli	25.00%	12.50%	62.50%
Haberman	21.43%	78.57%	0%
Heart-c	64.71%	35.29%	0%
Heart-h	64.71%	0%	35.29%
Heart-statlog	71.43%	21.43%	7.14%
Hypothyroid	82.14%	0%	17.86%
Iris	80.00%	20.00%	0%
Kr-vs-kp	76.32%	2.63%	21.05%
Lymph	89.29%	3.57%	7.14%
Sick	80.00%	0%	20.00%
Soybean	78.43%	1.96%	19.61%
Spambase	87.93%	3.45%	8.62%
Vowel	29.63%	14.81%	55.56%
Vote	88.24%	11.76%	0%

above three learning algorithms. On the 'Dermatology' data set, a larger improvement of the average accuracy is achieved for C4.5, NB and KNN through using the proposed data partitioning approach, in comparison with the random data partitioning approach, but the improvement of the average accuracy is very

small (only 1%), in comparison with the class-based random data partitioning approach. On the 'Hypothyroid', 'Iris', 'Sick' and 'Vote' data sets, the proposed data partitioning approach results in a very small or even no improvement of the average accuracy for C4.5, NB and KNN, in comparison with the better performing one of the other two approaches of data partitioning. The results on the above four data sets indicate that there is no much space for achieving significant advances in the average accuracy through adopting the proposed data partitioning approach, when the average accuracy is already high enough (above 90%) through adopting the other two approaches of data partitioning.

There are several cases that the percentage of Case 1 is not so high (about 65%) but the adoption of the proposed data partitioning approach still leads to considerable advances in the classification performance. For example, on the 'Autos' data set, the proposed data partitioning approach leads to a significant improvement of the average accuracy for C4.5, NB and KNN, comparing with the other two approaches of data partitioning. As mentioned in [25, 29], Case 2 and Case 3 may result from the two situations of decision tree learning as follows: (1) The training set contains noise; (2) The set of attributes is not enough to fully distinguish instances of different classes and additional attributes are needed to grow the decision tree. The second situation may not necessarily result in a negative impact on decision tree learning but may also be helpful for avoiding the case of overfitting. From the above viewpoint, the increase of the percentage of Case 2 and Case 3 may not be a negative point against the effectiveness of the proposed data partitioning approach. In other words, the phenomenon on the 'Autos' data set is likely due to the possibility that Case 2 and Case 3 result from the above mentioned second situation of decision tree learning and lead to positive impacts on the effectiveness of the proposed approach.

Moreover, on the 'Diabetes', 'Ecoli', 'Haberman' and 'Vowel' data sets, the percentage of Case 1 is very low (no greater than 50%). On the 'Ecoli' and 'Vowel' data sets, the proposed data partitioning approach leads to a considerable improvement of the average accuracy for C4.5, NB and KNN in comparison with the other two approaches of data partitioning, while the percentage of Case

3 is higher (above 55%). In contrast, the proposed data partitioning approach leads to a small improvement of the average accuracy for the above three learning algorithms on the 'Diabetes' set and even leads to a small drop in the average accuracy for two of the three learning algorithms on the 'Haberman' data set, while the percentage of Case 2 is much higher (above 55%) on the two data sets. The above description indicates the point that Case 3 seems to be more likely to result in a positive impact than Case 2 on the effectiveness of the proposed data partitioning approach, which is worthy of future research in more depth.

5. Conclusions

In this paper, we have proposed a subclass-based approach of semi-random data partitioning, which outperforms the random data partitioning approach and the class-based approach of semi-random partitioning in most cases according to the experimental results shown in Section 4. We have also identified that the random data partitioning approach can result in two issues: class imbalance and sample representativeness. Although the class imbalance issue can be addressed by using the class-based approach of semi-random partitioning, the sample representativeness issue still needed to be addressed leading to the subclass-based approach of semi-random data partitioning proposed in this paper. We have also proved through the experiments that the proposed approach of semi-random data partitioning not only keeps the class balance of both the training data and the test data but also improves the sample representativeness, leading to a more effective judgment of the learning ability of an algorithm and a significant improvement of the classification performance.

On the other hand, we have identified that the random data partitioning approach is likely to result in a high variance of classification performance when the same learning algorithm is used on the same data set with different training/test partitions. We have also proved through the experiments that the use of the two semi-random data partitioning approaches result in significant reduction of the variance in comparison with the use of the random data partitioning

approach. The experimental results show that the proposed subclass-based approach of semi-random data partitioning leads to further reduction of the variance in comparison with the class-based approach in most cases.

In the future, we will investigate clustering techniques [3, 4, 5, 6, 13, 19, 34] towards decomposing big data into multiple contexts, such that the training data obtained within each cluster can be more representative in the corresponding context. It is also worth to investigate granular computing techniques [7, 14, 31, 35, 41, 44] for the decomposition of each class in more depth [20, 22, 24, 30], towards further improvements of sample representativeness.

Acknowledgements

This work is supported by the University of Portsmouth, UK, under the Research Development Fund, and is supported by the Ministry of Science and Technology, Republic of China, under Grant MOST 107-2221-E-011-122 -MY2.

References

[1] P. Branco, L. Torgo, R. P. Ribeiro, A survey of predictive modeling on imbalanced domains, ACM Computing Surveys 49 (2) (2016) 31.

[2] J. Cendrowska, Prism: An algorithm for inducing modular rules, International Journal of Man-Machine Studies 27 (1987) 349–370.

[3] S. M. Chen, Y. C. Chang, Multi-variable fuzzy forecasting based on fuzzy clustering and fuzzy rule interpolation techniques, Information Sciences 180 (24) (2010) 4772–4783.

[4] S. M. Chen, H. R. Hsiao, A new method to estimate null values in relational database systems based on automatic clustering techniques, Information Sciences 169 (1-2) (2005) 47–69.

[5] S. M. Chen, K. Tanuwijaya, Fuzzy forecasting based on high-order fuzzy logical relationships and automatic clustering techniques, Expert Systems with Applications 38 (12) (2011) 15425–15437.

[6] S. M. Chen, N. Y. Wang, J. S. Pan, Forecasting enrollments using automatic clustering techniques and fuzzy logical relationships, Expert Systems with Applications 36 (8) (2009) 11070–11076.

[7] G. D'Aniello, A. Gaeta, V. Loia, F. Orciuoli, A granular computing framework for approximate reasoning in situation awareness, Granular Computing 2 (3) (2018) 141–158.

[8] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research 7 (2006) 1–30.

[9] P. A. Devijver, Pattern Recognition: A Statistical Approach, Prentice-Hall, London, 1982.

[10] M. S. Esfahani, E. R. Dougherty, Effect of separate sampling on classification accuracy, Bioinformatics 30 (2) (2014) 242–250.

[11] E. Frias-Martinez, S. Y. Chen, X. Liu, Survey of data mining approaches to user modeling for adaptive hypermedia, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 36 (6) (2006) 734–749.

[12] S. Geisser, Predictive Inference, Chapman and Hall, New York, 1993.

[13] Y. J. Horng, S. M. Chen, Y. C. Chang, C. H. Lee, A new method for fuzzy information retrieval based on fuzzy hierarchical clustering and fuzzy inference techniques, IEEE Transactions on Fuzzy Systems 13 (2) (2005) 216–228.

[14] B. Huang, H. Li, Distance-based information granularity in neighborhood-based granular space, Granular Computing 3 (1) (2018) 75–92.

[15] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann, San Mateo, CA, 1137–1143, 1995.

[16] I. Kononenko, M. Kukar, Machine Learning and Data Mining: Introduction to Principles and Algorithms, Horwood Publishing Limited, Chichester, West Sussex, 2007.

[17] K. Lang, E. Liberty, K. Shmakov, Stratified sampling meets machine learning, in: Proceedings of the 33rd International Conference on Machine Learning, JMLR.org, New York, 2320–2329, 2016.

[18] M. Lichman, UCI Machine Learning Repository, http://archive.ics.uci.edu/ml, 2013.

[19] P. Lingras, F. Haider, M. Triff, Granular meta-clustering based on hierarchical, network, and temporal connections, Granular Computing 1 (1) (2016) 71–92.

[20] H. Liu, M. Cocea, Fuzzy information granulation towards interpretable sentiment analysis, Granular Computing 2 (4) (2017) 289–302.

[21] H. Liu, M. Cocea, Semi-Random partitioning of data into training and test Sets in granular computing context, Granular Computing 2 (4) (2017) 357–386.

[22] H. Liu, M. Cocea, Granular Computing Based Machine Learning: A Big Data Processing Approach, Springer, Berlin, 2018.

[23] H. Liu, M. Cocea, Induction of classification rules by Gini-Index based rule generation, Information Sciences 436-437 (2018) 227–246.

[24] H. Liu, M. Cocea, W. Ding, Multi-Task learning for intelligent data processing in granular computing context, Granular Computing 3 (3) (2018) 257–273.

[25] H. Liu, A. Gegov, Induction of modular classification rules by information entropy based rule generation, in: V. Sgurev, R. R. Yager, J. Kacprzyk, V. Jotsov (Eds.), Innovative Issues in Intelligent Systems, vol. 623, 217–230, 2016.

[26] H. Liu, A. Gegov, M. Cocea, Generation of classification rules, in: Rule Based Systems for Big Data: A Machine Learning Approach, vol. 13, 29–42, 2016.

[27] H. Liu, A. Gegov, M. Cocea, Rule Based Systems for Big Data: A Machine Learning Approach, Springer, Switzerland, 2016.

[28] H. Liu, A. Gegov, M. Cocea, Unified framework for control of machine learning tasks towards effective and efficient processing of big Data, in: Data Science and Big Data: An Environment of Computational Intelligence, Springer, Switzerland, 123–140, 2017.

[29] H. Liu, A. Gegov, F. Stahl, Categorization and construction of rule based systems, in: 15th International Conference on Engineering Applications of Neural Networks, Sofia, Bulgaria, 183–194, 2014.

[30] H. Liu, L. Zhang, Fuzzy rule-based systems for recognition intensive classification in granular computing context, Granular Computing 3 (4) (2018) 355–365.

[31] Q. Liu, Q. Liu, L. Yang, G. Wang, A multi-granularity collective behavior analysis approach for online social networks, Granular Computing 3 (4) (2018) 333–343.

[32] E. J. Parish, K. Duraisamy, A paradigm for data-driven predictive modeling using field inversion and machine learning, Journal of Computational Physics 305 (2016) 758–774.

[33] J. Patel, S. Shah, P. Thakkar, K. Kotecha, Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques, Expert Systems with Applications 42 (1) (2015) 259–268.

[34] G. Peters, R. Weber, DCC: a framework for dynamic granular clustering, Granular Computing 1 (1) (2016) 1–11.

[35] A. Piegat, M. Landowski, Solving different practical granular problems under the same system of equations, Granular Computing 3 (1) (2018) 39–48.

[36] J. R. Quinlan, Induction of decision trees, Machine Learning 1 (1) (1986) 81–106.

[37] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Francisco, 1993.

[38] I. Rish, An empirical study of the naive bayes classifier, IJCAI 2001 workshop on empirical methods in artificial intelligence 3 (22) (2001) 41–46.

[39] P. Ristoski, H. Paulheim, Semantic Web in data mining and knowledge discovery: A comprehensive survey, Web Semantics: Science, Services and Agents on the World Wide Web 36 (2016) 1 – 22.

[40] C. E. Srndal, B. Swensson, J. Wretman, Model Assisted Survey Sampling, Springer, New York, 1992.

[41] T. O. William-West, D. Singh, Information granulation for rough fuzzy hypergraphs, Granular Computing 3 (1) (2018) 75–92.

[42] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2016.

[43] X. Wu, X. Zhu, G. Wu, W. Ding, Data mining with big data, IEEE Transactions on Knowledge and Data Engineering 26 (1) (2014) 97–107.

[44] W. Xu, W. Li, X. Zhang, Generalized multigranulation rough sets and optimal granularity selection, Granular Computing 2 (4) (2017) 271–288.

[45] J. Zhang, Selecting typical instances in instance-based learning, in: Proceedings of the Ninth International Workshop on Machine Learning, Aberdeen, United Kingdom, 470–479, 1992.