Version Space Completeness for Novel Hypothesis Induction in Biomedical Applications

Jinyan Li
Advanced Analytics Institute, Faculty of Engineering and IT
University of Technology Sydney
PO Box 123, Broadway, NSW 2007, Australia

***Abstract*—Use of traditional discretization methods caused a heavy loss of hypotheses in the induction of version spaces. We present a new discretization method, named *two-point discretization*, to construct an interval covering all the positive data points of a variable as purely as possible. We prove that the two-point discretization is a necessary and sufficient condition to guarantee the completeness of version spaces (i.e., no loss of hypothesis). A linear complexity algorithm is proposed to implement these theories. The algorithm is also applied to real-world bioinformatics problems to induce significant biomedical hypotheses which have been never discovered by the traditional approaches.**

1. Introduction

Version space has been extensively studied for decades since its first publication in IJCAI77 [1]. The convexity of version spaces plays a long-standing role for inductive learning in the fields of data mining and artificial intelligence.

Let D be a data set of m training instances $r_1, r_2, \ldots, r_m$, where each instance r_i is an ordered list of n categorical or numeric values derived from n attributes (variables) $x_1, x_2, \ldots, x_n$. Every instance r_i is assigned with a class label c, denoted by $c(r_i)$, which is either 1 representing 'positive', or 0 representing 'negative'. When D is represented by a matrix $D_{m\times n} = [d_{i,j}]_{m\times n}$, each row corresponds to an instance while every column corresponds to an attribute. A *hypothesis* [2] is defined as a conjunction of interval or equality constraints respectively on some numeric or categorical attributes of D. For example, $(0.1 \leq x_3 \leq 8.0) \wedge (x_5 = \text{green}) \wedge (x_6 < -2.5)$ is a hypothesis which consists of two interval constraints (on the numeric attributes x_3 and x_6) and one equality constraint (on the categorical attribute x_5).

The *version space* of D is the set of all of the possible hypotheses that are *consistent* with D. A hypothesis h is consistent with D iff: (i) every positive instance of D satisfies the conjunction h of constraints, and (ii) none of the negative instances in D satisfies the conjunction h of constraints. Let $r \in D$ and $h(r)$ be a Boolean function on r. $h(r) = 1$ if r satisfies the conjunction h of constraints, otherwise $h(r) = 0$. Then, "a hypothesis h is consistent with D" can be rewritten as $consistent(h, D) \equiv (\forall r \in D)h(r) = c(r)$, and the version space of D, denoted by $VS(D)$, can be represented as

$$VS(D) = \{h \mid consistent(h, D)\}.$$

TABLE 1. WEATHER CONDITIONS 'YES' OR 'NO' FOR A SPORT

r	sky (x_1)	humidity (x_2)	wind (x_3)	water (x_4)	sport? (c)
1	sunny	normal	strong	warm	yes (1)
2	sunny	high	strong	warm	yes (1)
3	sunny	high	strong	cool	yes (1)
4	rainy	high	strong	warm	no (0)

We use two examples to illustrate this definition.

Example 1. Table 1 is a set D^{sport} of four instances (three positive and one negative). Hypothesis $h_1 = (x_1 = sunny)$ is consistent with D^{sport}. Hypothesis $h_2 = (x_3 = strong)$ is not consistent with D^{sport}, because the negative instance also satisfies this constraint. However, $h_3 = h_1 \wedge h_2$ is consistent with D^{sport}. $VS(D^{sport}) = \{h_1, h_3\}$.

Example 2. A version space can be infinite in size. Suppose a numeric data set $D_{30\times 10}$ has 30 variables and contains 10 instances (5 positive '+' and 5 negative '-'). The projection of the 10 instances on subspace x_1x_2 is shown in Figure 1. Then, $h = (a \leq x_1 \leq b) \wedge (c \leq x_2 \leq d)$ belongs to $VS(D_{30\times 10})$ for every $a \in (2, 4)$, $b \in (6, 8)$, $c \in (1, 3)$ and $d \in (6, 8)$.

It is an exponential complexity to obtain the *complete* version space of $D_{30\times 10}$, as some other *subspaces* may also form hypotheses consistent with $D_{30\times 10}$. It is a challenging task to identify any of such subspaces under a linear time complexity.

An influential research on the size and concise representation of version spaces is the convexity theory that: the most general and the most specific elements ($\mathcal{G}$ and $\mathcal{S}$) always exist in a version space such that all other elements can be enumerated directly from these boundary elements without access to data [3], [2]. That is, these boundary elements can be used as a concise representation of the version space. Algorithms to derive $\mathcal{G}$ include the classic Candidate-Elimination algorithm [3], the Incremental-Version-Space-Merging algorithm [4], and later algorithms by [5], [6], [7].

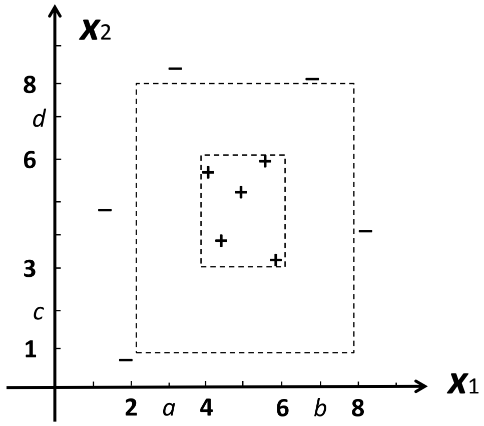


Figure 1. Consistent hypotheses at subspace x_1x_2.

Use of well-known discretization methods can cause information loss and hypotheses missing in the version space construction of a numeric data set. These discretization methods include the entropy-based discretization method [8], [9], [10], Gini-Index based method [11], Chi-square based method [12], [13], 1-rule method [14], [15], and unsupervised equal-length or equal-density methods. In fact, use of these discretization methods makes almost every version space empty as investigated by this work.

We propose a new discretization method, named two-point discretization, to split the value range of every numeric variable. Our two-point discretization results in an interval which covers all the data points from the positive class. We prove that two-point discretization is a necessary step to derive complete version spaces. Two-point discretization is also a sufficient condition when the interval is narrowed down to exclude other classes' data points as many as possible. Furthermore, it is a linear complexity to carry out the two-point discretization for a numeric data set D. It is also a linear complexity to test whether or not $VS(D) = \emptyset$. If $VS(D) \neq \emptyset$, only linear time is required to derive a subspace hypothesis consistent with D. Applied to high-dimensional real-life biomedical data sets, our algorithm can discover many hypotheses which are previously unknown and which can correct the misconception that version spaces are usually empty.

The remaining of the paper is organized as follows. In Section 2, we prove that the traditional discretization methods make almost every version space empty. Section 3 presents our proof on the necessity and sufficiency of our two-point discretization for the induction of complete version spaces. Section 4 describes a method for testing whether a version space is empty or not. Section 5 introduces a linear complexity algorithm to derive a hypothesis if the version space is not empty. In Section 6, we apply this algorithm to three contemporary genomics and molecular biology problems to draw novel biological hypotheses for domain investigation. Section 7 concludes the paper.

2. Traditional Discretization Makes Almost All Version Spaces Empty

The version spaces of many numeric data sets are *not* empty as shown in this work. But the traditional discretization methods cause information loss in the construction of these version spaces, making them almost all empty.

Let $disD_{m\times n} = [d_{i,j}]_{m\times n}$ be a matrix representing m training instances $r_1, r_2, \ldots, r_m$ on n variables $x_1, x_2, \ldots, x_n$ after *discretization*. Here, $d_{i,j}$ is a categorical value if x_j is a categorical variable, or it is a real number interval if x_j is a numeric variable.

Definition 1 (singleton hypothesis). A singleton hypothesis is a constraint on one attribute. For example, $x_1 = \text{red}$ is a singleton hypothesis. For a discretized numeric variable x_j (i.e., a special categorical variable), any of its discretized intervals, e.g. $d_{i,j} = (a, b]$, can form a singleton hypothesis $x_j = d_{i,j}$, meaning $a < x_j \leq b$. Sometimes we use $x_j(d_{i,j})$ to rewrite $a < x_j \leq b$.

Proposition 1. $VS(disD_{m\times n}) = \emptyset$, if there does not exist any singleton hypothesis of $disD_{m\times n}$ that can be satisfied by every positive instance.

Proof: By assumption, for a singleton hypothesis $h_{i,j} = (x_j = d_{i,j})$, $i = 1, 2, \ldots,$ or, m, $j = 1, 2, \ldots,$ or, n, not every positive instance satisfies $h_{i,j}$. Therefore, $h_{i,j}$ is not consistent with $disD_{m\times n}$, namely $h_{i,j} \notin VS(disD_{m\times n})$. Suppose h is the conjunction combination of a subset of these singleton hypotheses, then h cannot be satisfied by any positive instance either. Therefore, $VS(disD_{m\times n})$ is empty. □

Let $D^{numeric}_{m\times n} = [r_{i,j}]_{m\times n}$ be a matrix representing m training instances $r_1, r_2, \ldots,$ r_m on n *numeric* variables $x_1, x_2, \ldots, x_n$. Entropy-based discretization methods [8], [9], [10] normally discretize the value range of each variable x_i into two intervals. The value range of a variable is divided into such two intervals that the information gain of the splitting is maximal. A consequence of the entropy-based discretization is that the data of the same class are separated into two unbalanced proportions to maximize the object function.

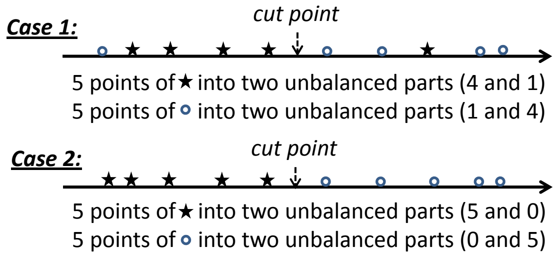

Case 1:
cut point
5 points of ★ into two unbalanced parts (4 and 1)
5 points of ○ into two unbalanced parts (1 and 4)
Case 2:
cut point
5 points of ★ into two unbalanced parts (5 and 0)
5 points of ○ into two unbalanced parts (0 and 5)

Figure 2. Entropy-based discretisation (one cut point).

Most often, the value range of a variable x_i is discretized as Case 1 of Figure 2, where neither of the two intervals can cover all the positive instances (the 'star' points). If the value range of a variable is two-side distributed for the two classes (Case 2 of Figure 2), then one of the two intervals can cover all the positive instances. Case 2 is extremely rare in real-life applications. We name the variable of Case 2 a *classification-perfect* variable.

We use $D_{m\times n}^{numeric-tpc} = [r_{i,j}]_{m\times n}$ to denote a numeric data matrix which does not contain any classification-perfect variables, namely the n numeric variables $x_1, x_2, \ldots, x_n$ are all *typically* distributed as Case 1 of Figure 2.

Proposition 2. Let $en\text{-}disD_{m\times n}^{numeric-tpc} = [r'_{i,j}]_{m\times n}$ be a discretized data matrix of $D_{m\times n}^{numeric-tpc}$ via an entropy-based discretization method. Then, there does not exist any singleton hypothesis of $en\text{-}disD_{m\times n}^{numeric-tpc}$ that can be satisfied by any positive instance.

Proof: Because none of the n numeric variables $x_1, x_2, \ldots, x_n$ of $D_{m\times n}^{numeric-tpc}$ is a classification-perfect variable, the value range of x_j, $1 \le j \le n$, is divided into such two intervals $(-\infty, a_j]$ and $(a_j, +\infty)$ that neither of them can cover all the positive instances. That is, not every positive instance can satisfy hypothesis $(-\infty < x_j \le a_j)$ or hypothesis $(a_j < x_j < +\infty)$. Therefore, there does not exist any singleton hypothesis $h_{i,j} = (x_j = r'_{i,j})$, $1 \le i \le m$, $1 \le j \le n$, that can be satisfied by any positive instance. □

Corollary 1. $VS(en\text{-}disD_{m\times n}^{numeric-tpc}) = \emptyset$.

Proof: According to Proposition 2, there does not exist any singleton hypothesis of $en\text{-}disD_{m\times n}^{numeric-tpc}$ that can be satisfied by every positive instance. According to Proposition 1, we have $VS(en\text{-}disD_{m\times n}^{numeric-tpc}) = \emptyset$. □

Proposition 2 and Corollary 1 both hold when the entropy-based discretization is recursively applied to have more than one cut points to split the value range of a variable.

However, $VS(D_{m\times n}^{numeric-tpc})$ is *not* empty for many typical numeric data sets. The emptiness of $VS(en\text{-}disD_{m\times n}^{numeric-tpc})$ is attributed to the discretization method which causes information loss specially for the construction of version spaces.

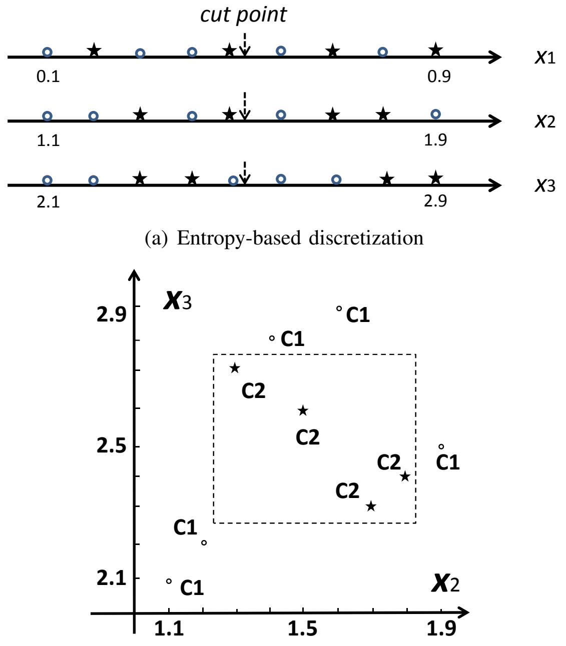
Table 2 shows a typical numeric data set which contains 9 instances on three variables: 4 instances are positive (class label C2) and the remaining 5 are negative (class label C1). It is denoted as $D_{9\times 3}^{numeric-tpc}$. The instance numbers 4 and 5 are purposely used for an example of unbalanced training data sets.

The entropy-based discretization method discretizes the value ranges of x_1, x_2 and x_3 by splitting these ranges in the middle (Figure 3(a)). We can see that none of the 6 intervals can be satisfied by all the positive instances (the C2 instances). According to Corollary 1, $VS(en\text{-}disD_{9\times 3}^{numeric-tpc}) = \emptyset$.

In fact, $VS(D_{9\times 3}^{numeric-tpc})$ is infinite in size. For example, hypothesis $h = (1.3 \le x_2 \le 1.8) \wedge (2.3 \le x_3 \le 2.7)$ belongs to $VS(D_{9\times 3}^{numeric-tpc})$. See Figure 3(b). It is easy to get other hypotheses by relaxing the two constraints of h similarly as in Example 2 and Figure 1. All these hypotheses are not detectable through the entropy-based discretization approach. We report real-life examples of this kind in Section 6.

TABLE 2. A SMALL DATA SET WHICH HAS A NON-EMPTY VERSION SPACE.

instance ID	x_1	x_2	x_3	class label
1	0.1	1.1	2.1	C1
2	0.2	1.7	2.3	C2
3	0.3	1.2	2.2	C1
4	0.4	1.4	2.8	C1
5	0.5	1.8	2.4	C2
6	0.6	1.6	2.9	C1
7	0.7	1.5	2.6	C2
8	0.8	1.9	2.5	C1
9	0.9	1.3	2.7	C2



(a) Entropy-based discretization

(b) Hypotheses in the subspace x_2x_3

Figure 3. Non-emptyness of $VS(D_{9\times 3}^{numeric-tpc})$.

The Gini-Index based discretization method, the chi-square based method, the 1-rule method, or the equal-length, equal-density bin methods also cause information loss in the construction of version spaces for numeric data sets. In fact, none of them can find any hypothesis for $VS(D_{9\times 3}^{numeric-tpc})$ although this version space contains infinite number of hypotheses.

3. Two-point Discretization: Necessity and Sufficiency

We introduce a new discretization method for the construction of complete version spaces. We name this method *two-point discretization*. A special case of two-point discretization is through the use of *compact interval*. A compact interval is a closed interval in the value range of a

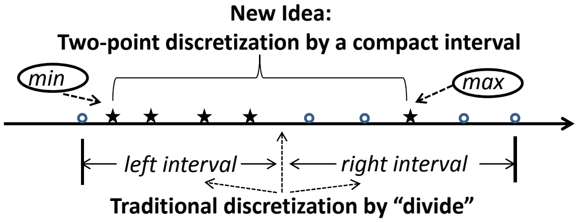

New Idea:
Two-point discretization by a compact interval
min
max
left interval
right interval
Traditional discretization by “divide”

Figure 4. Comparison of discretization methods.

numeric variable which covers all the data points from one class as densely as possible. Namely, a compact interval is bounded by the min and max points of the class. See Figure 4 for an illustration. We prove that two-point discretization is a necessary condition and can be refined to be a sufficient step in the construction of complete version spaces for all numeric data sets.

3.1. Necessity Proof

Definition 2 (two-point discretization). Let $D^{numeric}_{m\times n} = [r_{i,j}]_{m\times n}$ be a matrix representing m training instances $r_1, r_2, \ldots, r_m$ on n numeric variables $x_1, x_2, \ldots, x_n$. Two-point discretization of $D^{numeric}_{m\times n}$ is:

(i) to find an interval I_j which can cover all the positive points in the value range of variable x_j, $1 \leq j \leq n$, and

(ii) to replace $r_{i,j}$ with I_j if $r_{i,j} \in I_j$, otherwise replace $r_{i,j}$ with the negation of I_j, namely $\overline{I_j}$.

Theorem 1 (necessity). Two-point discretization is a necessary step to induce the complete version space of $D^{numeric}_{m\times n}$ unless it is empty.

Proof: Without loss of generality, suppose an interval V_j does not cover all the positive instances projected on variable x_j. Then hypothesis $x_j(V_j)$ is not consistent with $D^{numeric}_{m\times n}$. Its combination with any other hypotheses is not consistent with $D^{numeric}_{m\times n}$ either. Therefore, any hypothesis involving x_j gets lost. Thus, two-point discretization is a necessary discretization step to induce the complete version of $D^{numeric}_{m\times n}$. □

3.2. Sufficiency of Two-point Discretization When Using Compact Intervals

We present a sufficient condition of two-point discretization to construct complete version spaces, which involves the definition of a special interval called *compact interval*. The sufficiency by using a variant definition of compact intervals is proved in the later subsection.

Definition 3 (compact interval). Let $E = \{e_1, e_2, \ldots, e_m\}$ be a set of m real numbers (numeric points) in R. Each of them is associated with a class label c_1 or c_2. The *compact interval* of c_1 points in E is defined as the minimal closed interval that covers all the c_1 points in E. It is denoted by $CI(c_1, E)$.

Definition 4 (purity of compact intervals). Following Definition 3, the *purity* of c_1 points in $CI(c_1, E)$ is defined as the percentage of c_1 points over the total number of data points covered by $CI(c_1, E)$.

Example 3. Suppose $E = \{0.8(*), 1.0(+), 1.2(*), 1.4(+), 1.6(+), 1.8(*)\}$, where '+' or '*' is a class label, then $CI(+, E) = [1.0, 1.6]$. The purity of the '+' points in $CI(+, E)$ is 0.75 (3/4).

Two-point discretization using compact intervals can be implemented by the following steps. Suppose the m training instances of $D^{numeric}_{m\times n}$ are each assigned with a class label c_1 or c_2. When the class label c_1 is considered as positive, we compute the compact interval of the c_1 points in $E(x_j) = \{r_{i,j}, i = 1, 2, \ldots, m\}$ for all $j = 1, 2, \ldots,$ or n, namely $CI(c_1, E(x_j))$, $1 \leq j \leq n$. We then rewrite $CI(c_1, E(x_j))$ as o_j, and replace every instance of class c_1 as $o_1, o_2, \ldots, o_n$. For every instance r_i of class c_2, we replace $r_{i,j}$ as o_j if $r_{i,j} \in CI(c_1, E(x_j))$, or otherwise as $\overline{o}_j$ if $r_{i,j} \notin CI(c_1, E(x_j))$, $1 \leq j \leq n$.

The discretized data set is denoted by $ci\text{-}disD^{numeric}_{m\times n}$ (different from $en\text{-}disD^{numeric}_{m\times n}$ which is a notation for the discretized data sets using entropy).

Example 4. For the data set $D^{numeric-tpc}_{9\times 3}$ in Table 2, the compact interval of the *C2* points in the value range of x_1 is $[0.2, 0.9]$. Similarly, $CI(C2, E(x_2)) = [1.3, 1.8]$ and $CI(C2, E(x_3)) = [2.3, 2.7]$. Then $[0.2, 0.9]$ is rewritten as o_1, $[1.3, 1.8]$ is rewritten as o_2, and $[2.3, 2.7]$ is rewritten as o_3. Then o_1, o_2, o_3, $\overline{o_1}$, $\overline{o_2}$, or $\overline{o_3}$ is used to replace each value in $D^{numeric-tpc}_{9\times 3}$. The four *C2* instances (denoted as D_{C2}) after the two-point discretization are transformed as:

instance r	x_1	x_2	x_3	label c
2	o_1	o_2	o_3	C2
5	o_1	o_2	o_3	C2
7	o_1	o_2	o_3	C2
9	o_1	o_2	o_3	C2

and the five *C1* instances (denoted as D_{C1}) are transformed as:

instance r	x_1	x_2	x_3	label c
1	$\bar{o}_1$	$\bar{o}_2$	$\bar{o}_3$	C1
3	o_1	$\bar{o}_2$	$\bar{o}_3$	C1
4	o_1	o_2	$\bar{o}_3$	C1
6	o_1	o_2	$\bar{o}_3$	C1
8	o_1	$\bar{o}_2$	o_3	C1

The version space of a numeric data set is always infinite in size (if not empty), while the size of the version space after the two-point discretization (or any other discetization method) is always limited. To prove the completeness of version spaces after data discretization, we need an equivalence definition for all hypotheses having the same attributes (i.e., sharing the same subspace).

Definition 5. Suppose h_1 and h_2 are two hypotheses in the version space of a numeric data set. We say h_1 and h_2 are equal if and only if the variables involved in h_1 are exactly the same as the variables involved in h_2. It is denoted by $h_1 = h_2$.

When $h_1 = h_2$, it does not means the intervals in h_1 are the same as the intervals in h_2. Refer to Example 2 in the Introduction and Figure 1 for the rationality of this definition. For example, $h_1 = (2.1 \leq x_1 \leq 6.1) \wedge (1.1 \leq x_2 \leq 6.1)$ and $h_2 = (2.2 \leq x_1 \leq 6.2) \wedge (1.2 \leq x_2 \leq 6.2)$ in Figure 1 can be defined to be equal.

Because of Definition 5, the infinite number of hypotheses in the version space of a numeric data set becomes limited and counted. More examples can be seen from Figure 3(b).

Theorem 2 (sufficiency). For any numeric data set $D^{numeric}_{m\times n} = [r_{i,j}]_{m\times n}$, its version space is complete after two-point discretization, namely $VS(\textit{ci-dis}D^{numeric}_{m\times n}) = VS(D^{numeric}_{m\times n})$.

Proof: Suppose $h = \wedge_{j\in X} x_j(I_j) \in VS(D^{numeric}_{m\times n})$, where $X \subseteq \{1, 2, \ldots, n\}$, and I_j, $j \in X$, is an real number interval. Then singleton hypothesis $x_j(I_j)$ is satisfied by all the positive instances of $D^{numeric}_{m\times n}$, namely I_j covers all the positive instances projected on the variable x_j. Therefore, $CI(\textit{positive}, E(x_j)) \subseteq I_j$. As $\wedge_{j\in X} x_j(I_j)$ is not satisfied by any negative instance of $D^{numeric}_{m\times n}$, $\wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$ is also not satisfied by any negative instance of $D^{numeric}_{m\times n}$. As $\wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$ is satisfied by every positive instance of $D^{numeric}_{m\times n}$, $\wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$ is consistent with $D^{numeric}_{m\times n}$. Let $h' = \wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$. Then h and h' are two hypotheses of $VS(D^{numeric}_{m\times n})$ which involve exactly the same variables $\{x_j | j \in X\}$. Therefore, $h = h'$. After the two-point discretization of $D^{numeric}_{m\times n}$ using compact intervals, and rewriting h' as $\wedge_{j\in X}(x_j = CI(\textit{positive}, E(x_j)))$, it can be seen that $h' \in VS(\textit{ci-dis}D^{numeric}_{m\times n})$. Therefore, $h \in VS(\textit{ci-dis}D^{numeric}_{m\times n})$. (See Definition 1 for 'rewrite'.)

Suppose $h' = \wedge_{j\in X}(x_j = CI(\textit{positive}, E(x_j))) \in VS(\textit{ci-dis}D^{numeric}_{m\times n})$. Then h' can be rewritten as $\wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$. It is understood that hypothesis $\wedge_{j\in X} x_j(CI(\textit{positive}, E(x_j)))$ is consistent with $D^{numeric}_{m\times n}$. Therefore, $h' \in VS(D^{numeric}_{m\times n})$. □

Theorem 2 states that there are no hypothesis missing for $D^{numeric}_{m\times n}$ after our two-point discretization using compact intervals. Taking Theorem 1 and Theorem 2 together, we can see that two-point discretization is a necessary and sufficient discretization step towards the completeness of version spaces.

3.3. Sufficiency of Two-point Discretization When Using Variants of Compact Intervals

Some simple variants of compact intervals can also provide sufficiency for two-point discretization to induce complete version spaces.

Definition 6 (variants of compact interval). Let $E = \{e_1, e_2, \ldots, e_m\}$ be a set of m real numbers (numeric points) in R. Each of them is associated with a class label c_1 or c_2. A *compact interval* of the c_1 points in E, denoted by $CI(c_1, E)$, is defined as one of those intervals that must cover all the c_1 points in E and that must exclude the c_2 points as many as possible.

Let $min_{(c_1,E)}$ be the minimal point of class c_1 in E, and $max_{(c_1,E)}$ the maximal point. Let $E([min_{(c_1,E)}, max_{(c_1,E)}])$ be the set of points in E that are covered by $[min_{(c_1,E)}, max_{(c_1,E)}]$. By Definition 6, $CI(c_1, E)$ cannot contain any point in $E - E([min_{(c_1,E)}, max_{(c_1,E)}])$. It is true that Definition 3 is a special case of Definition 6.

Example 5. Following Example 3, a compact interval of the '+' points in E under Definition 6 can be $(0.9, 1.6]$, $[1.0, 1.7)$, or $(0.85, 1.75)$. On the other hand, $[0.8, 1.6]$ is not a compact interval of the '+' points in E. The reason is that this interval covers $0.8(*)$ which is a point belonging to $E - E([min_{(+,E)}, max_{(+,E)}])$, namely it does not exclude the '*' points as many as possible.

A compact interval under Definition 6 can be in the form of $(-\infty, b)$, $(-\infty, b]$, $(a, b]$, $[a, b)$, $[a, b]$, $(a, +\infty)$, or $[a, +\infty)$. We can prove that the version space of a discretized numeric data set by using a variant definition of compact intervals is exactly redundant to the version space of the discretized numeric data set by using the compact intervals of Definition 3. Proof is omitted in this manuscript. Based on this observation, the variant definitions of compact intervals also cause no information loss to construct the complete version spaces of numeric data sets. Therefore, we consider only the version space of the discretized numeric data set by using the compact intervals of Definition 3 in the subsequent work unless specified otherwise.

4. Non-emptiness Test for Version Spaces

The version spaces of some numeric data sets are empty. We present a linear complexity algorithm to easily test whether a data set has an empty version space or not.

Proposition 3. For data set $D^{numeric}_{m\times n} = [r_{i,j}]_{m\times n}$, $VS(D^{numeric}_{m\times n}) = \emptyset$ if there exists a row of $o_1, o_2, \ldots, o_n$ in the negative class after two-point discretization using compact intervals. Otherwise, $VS(D^{numeric}_{m\times n})$ is not empty.

Proof: Because the negative class contains an instance $\{o_1, o_2, \ldots, o_n\}$ after the two-point discretization, then any conjunction combination of singleton hypotheses $x_1(o_1), x_2(o_2), \ldots, x_n(o_n)$ can be satisfied by this negative instance. Therefore, none of these conjunction combinations is consistent with $\textit{ci-dis}D^{numeric}_{m\times n}$. Furthermore, there is no hypothesis containing singleton hypothesis $x_1(\overline{o_1}), x_2(\overline{o_2}), \ldots,$ or $x_n(\overline{o_n})$ that can be satisfied by any positive instance of $\textit{ci-dis}D^{numeric}_{m\times n}$. Therefore, no hypothesis is consistent with $\textit{ci-dis}D^{numeric}_{m\times n}$, namely, $VS(\textit{ci-dis}D^{numeric}_{m\times n}) = \emptyset$. Thus, $VS(D^{numeric}_{m\times n}) = \emptyset$ by Theorem 2.

Suppose the negative class does not contain any instance $\{o_1, o_2, \ldots, o_n\}$ after the two-point discretization, then at least the hypothesis $x_1(o_1) \wedge x_2(o_2) \wedge$

$\ldots \wedge\ x_n(o_n)$ is consistent with $ci\text{-}disD^{numeric}_{m\times n}$. Therefore, $VS(ci\text{-}disD^{numeric}_{m\times n})$ is not empty. Thus, $VS(D^{numeric}_{m\times n})$ is also not empty. □

When the negative class is considered as positive, it can be similarly tested whether or not the corresponding version space is empty.

Example 6. Following Example 4, there does not exist any row of $o_1\ o_2\ o_3$ in the discretized D_{C1}, thus, the version space is non-empty. In fact, at least $x_2(o_2) \wedge x_3(o_3)$, namely $(1.3 \leq x_2 \leq 1.8) \wedge (2.3 \leq x_3 \leq 2.7)$, belongs to $VS(D^{numeric-tpc}_{9\times 3})$. This hypothesis can be verified again from Figure 3(b).

The total time complexity of this test is linear to the size of D, namely $O(nm)$.

5. Linear Complexity for Inducing One Hypothesis

If a version space is tested as non-empty, we introduce a recursive linear complexity algorithm to induce one hypothesis at some subspace of the variables. Brute-force algorithms have an exponential complexity.

Definition 7. The compact interval A of the c points in E_1 is defined to be *purer* than the compact interval B of the c points in E_2 if the purity of the c points in A is higher than that in B.

Let $D = [r_{i,j}]_{m\times n}$ be a data set representing m training instances $r_1, r_2, \ldots, r_m$ on n numeric variables $x_1, x_2, \ldots, x_n$. Consider class label c of D as positive, the algorithm takes the following steps:

1) Determine the compact interval of the c points in $E(x_j) = \{r_{i,j}, i = 1, 2, \ldots, m\}$ for all $j = 1, 2, \ldots$, and n. Let $CI(c, E(x_k)) = [a_k, b_k]$ be the *purest interval* in these intervals.
2) Store singleton hypothesis $(a_k \leq x_k \leq b_k)$.
3) Group all those instances r_i of D which satisfy the singleton hypothesis $x_k = CI(c, E(x_k))$ to form D'.
4) Replace D by D'
5) If the purity of $CI(c, E(x_k)) \neq 100\%$, go to Step 1; otherwise go to Step 6.
6) Let h be the conjunction of all these $(a_k \leq x_k \leq b_k)$, then $h \in VS(D)$.

The recursion of this algorithm usually stops at depth of 3, 4 or 5 for high-dimensional numeric data sets (hundreds or thousands of variables, examples shown in next section). That means a subspace of 3, 4 or 5 variables from $x_1, x_2, \ldots, x_n$ can be fast identified, where hypotheses exist to be consistent with high-dimensional D. The worst case of the recursion is $O(n)$.

We use the data set D in Table 2 to illustrate this algorithm. Suppose $c = C_2$. Then $CI(C_2, E(x_1)) = [0.2, 0.9]$, its purity is $4/8 = 50\%$; $CI(C_2, E(x_2)) = [1.3, 1.8]$, its purity is $4/7 = 57.1\%$; and $CI(C_2, E(x_3)) = [2.3, 2.7]$, its purity is $4/5 = 80\%$. Therefore $k = 3$, namely the purest interval is $CI(C_2, E(x_3)) = [2.3, 2.7]$. Store singleton hypothesis $2.3 \leq x_3 \leq 2.7$. Then, group instances 2, 5, 7, 8 and 9, namely those satisfying this singleton hypothesis, to form D'. Replace D with D' and go to Step 1, then $CI(C_2, E(x_1)) = [0.2, 0.9]$, its purity is $4/5 = 80\%$; $CI(C_2, E(x_2)) = [1.3, 1.8]$, its purity is 4/4 = 100%; and of course, $CI(C_2, E(x_3)) = [2.3, 2.7]$, its purity is $4/5 = 80\%$. Therefore, $k = 2$, that is, the purest interval in the new D is $CI(C_2, E(x_2)) = [1.3, 1.8]$. Store singleton hypothesis $1.3 \leq x_2 \leq 1.8$. The algorithm goes to Step 5, as the purity reaches 100%. Then output hypothesis $(2.3 \leq x_3 \leq 2.7) \wedge (1.3 \leq x_2 \leq 1.8)$ which belongs to $VS(D)$. This is the same hypothesis shown in Figure 3(b).

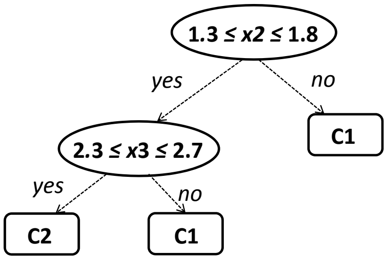


Figure 5. Decision tree derived by our two-point discretization.

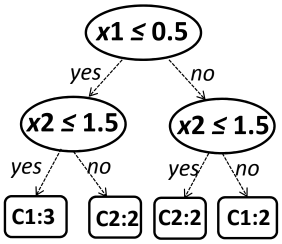


Figure 6. Decision tree derived via information gain discretization.

The time cost is $O(mn)$ with memory usage of $O(mn)$, where m is the number of instances in D and n is the number of variables. The complexity $O(m)$ is used to determine the purest interval for D or D' in the recursion.

A version space hypothesis is equivalent to a 100%-frequency decision rule. Traditionally, decision rules are derived by decision tree, e.g., by C4.5 [8]. C4.5 (without pruning) needs $O(m \log_2 m)$ under the memory usage of $O(mn)$ to determine the root node of the tree and needs $O(nm \log_2 m)$ to induce the whole tree. Thus, our algorithm is faster than C4.5 in the order of $\log_2 m$.

We note that decision tree C4.5 (information gain based tree induction method) or CART (gini impurity based tree induction method) is unable to derive this hypothesis because the class labels of the 9 instances are roughly distributed randomly over the variable x_1, x_2, or x_3. This hypothesis (100%-frequency) rule is equivalent to the decision tree in Fig. 5 which is different from C4.5 tree (Fig. 6).

6. Novel Hypotheses Discovered in Bioinformatics Problems

Recent research on genomics and molecular biology has aimed to draw *biomedical hypotheses* or insights from high-throughput numerical data sets for disease mechanism understanding and diagnosis [16], [17], [18], [19], [20], [21], [22], [23]. One area is to generate hypotheses based on the expression levels of miRNAs and mRNAs to infer the miRNA target mRNAs and these association interplays with the onset of diseases. As hypotheses in a version space are all strong statements to contrast the characteristics between positive and negative classes of data, it is interesting to see whether there exist hypotheses in the version spaces of these biomedical data sets under our new theories and algorithms. It is also interesting to see whether the computationally discovered hypotheses can be interpretable as biological hypotheses for domain investigation.

6.1. Novel Hypotheses Highlighting Leukemia Cell Expressions

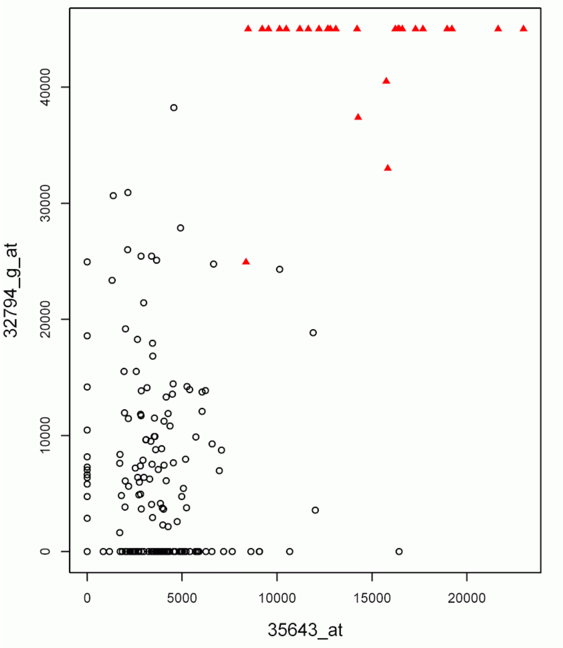


Figure 7. Novel hypotheses related to leukemia cell expressions.

The first data set is a classical mRNA expression data set [16], containing 215 childhood leukemia cell samples (28 positive cells of the T-ALL subtype vs 187 cells of other subtypes). The variables of this data set, i.e., the 12558 genes, are all numeric. The following hypothesis

$$(8361.1 \leq x_{5440} \leq 22975.2) \wedge (24924.7 \leq x_{7564} \leq 45000)$$

exists in the version space of this data set (Figure 7). It can be biologically interpreted as: if (i) the expression of $gene_{5440}$ (probe 35643_at) is between 8361.1 and 22975.2, and (ii) the expression of $gene_{7564}$ (probe 32794_g_at) is between 24924.7 and 45000, all such cell samples belong to the T-ALL subtype but not other subtypes of leukemia. This is a hypothesis underlying some principle of gene expression mechanism in the T-ALL subtype. The traditional decision tree approach is unable to detect this hypothesis because it uses entropy-based discretization techniques.

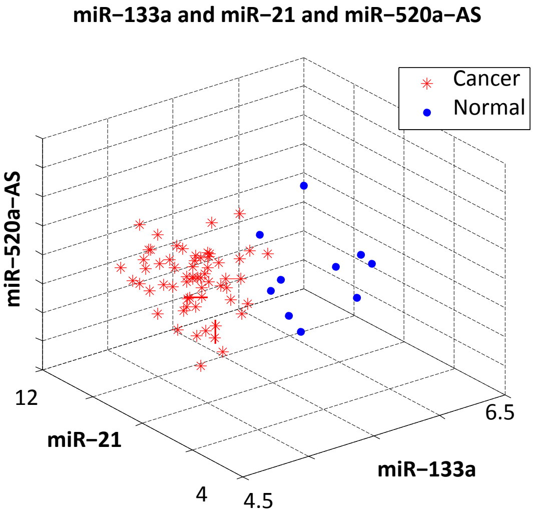


Figure 8. Novel hypotheses from a lung cell data set.

6.2. Novel Hypotheses Discovered from Lung and Liver Cancer Data Sets

The second data set D^{lung} [17] is a numeric data matrix of the expression values of 71 lung tissue cells on 328 miRNAs (variables). Of the 71 instances, 61 are cancer cells (positive class), and the remaining 10 are the adjacent normal lung cells. This data set is available at the National Center for Biotechnology Information Gene Expression Omnibus (GEO, http://www.ncbi.nlm.nih.gov/geo/ (accession number GSE16025).

None of the 328 variables is a classification-perfect variable. All the traditional discretization methods make $VS(D^{lung})$ empty. But in fact, $VS(D^{lung})$ is non-empty. Our algorithm can induce a hypothesis

$$(-\infty < x_1 \leq 5.85) \wedge (7.38 \leq x_2 < +\infty) \wedge (-\infty < x_3 \leq 5.23)$$

consistent with D^{lung}, where x_1, x_2 and x_3 represent variables miR-133a, miR-21, and miR-520a-AS, respectively (Figure 8). It is a novel hypothesis at a 3-variable subspace of the 328 variables discovered with linear complexity.

The third data set $D^{miLiver}$ [18] is about the expression levels of 470 miRNAs in human liver cells after the infection of Hepatitis C virus (HCV). This data set consists of 24 HCV positive samples and 12 negative samples. The fourth data set D^{mLiver} is about the same 36 tissue samples [18]. But D^{mLiver} represent all the mRNA expression levels instead of miRNA expression levels. There are 22575 mRNAs involved in D^{mLiver}. So, $D^{miLiver}$ and D^{mLiver} are paired data sets of the same instances under the description of different sets of variables, which are particularly useful for the discovery of miRNA-mRNA regulatory modules related to the infection of Hepatitis C virus. This pair of data sets is available from the NCBI Gene Expression Omnibus database under the (accession number GSE15387). Our algorithm discovered many hypotheses for $VS(D^{miLiver})$ and $VS(D^{miLiver})$.

For 10-fold cross validation on these data sets, our classification accuracy on each fold's test data is 5% - 15% higher than C4.5 decision tree, decision tree ensembles (Bagging or Boosting), and Support Vector Machine (SVM).

A significant contribution of our study is that we discovered version space hypotheses which could not be discovered by any traditional discretization method, as demonstrated in these real-life biomedical data sets.

7. Conclusion

This work has proved that two-point discretization is a necessary and sufficient step to guarantee the completeness of version spaces for all numeric data sets. The algorithm is in linear time to test whether a version space is empty. If not, it is also in linear time to induce one subspace hypothesis. This algorithm can be used iteratively to discover multiple hypotheses. Application to high-dimensional biomedical data sets has confirmed these theoretical results. Some of these hypotheses are the most general hypotheses of the version spaces (i.e., elements in $\mathcal{G}$, mentioned in Introduction). But, they are not always. It is an interesting future work to detect all the elements in $\mathcal{G}$ by an algorithm of high efficiency.

Acknowledgments

This research work was partly supported by Australia Research Council Discovery Project 130102124.

References

[1] T. Mitchell, "Version spaces: A candidate elimination approach to rule learning," in *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, R. Reddy, Ed., Cambridge, MA, 1977, pp. 305–310.

[2] T. M. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[3] T. Mitchell, "Generalization as search," *Artificial Intelligence*, vol. 18, pp. 203–226, 1982.

[4] H. Hirsh, "Generalizing version spaces," *Machine Learning*, vol. 17, pp. 5–46, 1994.

[5] M. Sebag, "Delaying the choice of bias: A disjunctive version space approach," in *Machine Learning: Proceedings of the Thirteenth International Conference*. Morgan Kaufmann, 1996, pp. 444–452.

[6] H. Hirsh, N. Mishra, and L. Pitt, "Version spaces and the consistency problem," *Artif. Intell.*, vol. 156, no. 2, pp. 115–138, Jul. 2004. [Online]. Available: http://dx.doi.org/10.1016/j.artint.2003.04.003

[7] E. Smirnov, G. Nalbantov, and N. Nikolaev, "k-version-space multi-class classification based on k-consistency tests," in *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, ser. ECML PKDD'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 277–292. [Online]. Available: http://dl.acm.org/citation.cfm?id=1889788.1889807

[8] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986. [Online]. Available: http://dx.doi.org/10.1023/A:1022643204877

[9] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning." in *IJCAI*, 1993, pp. 1022–1029. [Online]. Available: http://dblp.uni-trier.de/db/conf/ijcai/ijcai93.html#FayyadI93

[10] R. Kohavi and M. Sahami, "Error-based and entropy-based discretization of continuous features," in *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 114–119.

[11] L. Breiman, J. Friedman, C. J. Stone, and R. Olshen, *Classification and Regression Trees*. Chapman and Hall/CRC, 1998.

[12] R. Kerber, "Chimerge: discretization of numeric attributes," in *Proceedings of the tenth national conference on Artificial intelligence*, ser. AAAI'92. AAAI Press, 1992, pp. 123–128. [Online]. Available: http://dl.acm.org/citation.cfm?id=1867135.1867154

[13] H. Liu and R. Setiono, "Feature selection via discretization," *IEEE Trans. on Knowl. and Data Eng.*, vol. 9, no. 4, pp. 642–645, Jul. 1997. [Online]. Available: http://dx.doi.org/10.1109/69.617056

[14] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Mach. Learn.*, vol. 11, no. 1, pp. 63–90, Apr. 1993. [Online]. Available: http://dx.doi.org/10.1023/A:1022631118932

[15] R. C. Holte and C. Drummond, "Cost-sensitive classifier evaluation using cost curves," in *Proceedings of the 12th Pacific-Asia conference on Advances in knowledge discovery and data mining*, ser. PAKDD'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 26–29. [Online]. Available: http://dl.acm.org/citation.cfm?id=1786574.1786580

[16] E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing, "Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling," *Cancer Cell*, vol. 1, pp. 133–143, 2002.

[17] M. Raponi, L. Dossey, T. Jatkoe, X. Wu, G. Chen, H. Fan, and D. G. Beer, "MicroRNA classifiers for predicting prognosis of squamous cell lung cancer," *Cancer Res*, vol. 69, no. 14, pp. 5776–5783, 2009.

[18] X. Peng, Y. Li, K.-A. Walters, E. R. Rosenzweig, S. L. Lederer, L. D. Aicher, S. Proll, and M. G. Katze, "Computational identification of hepatitis C virus associated microRNA-mRNA regulatory modules in human livers," *BMC Genomics*, vol. 10, no. 1, p. 373, 2009.

[19] F. A. Karreth, M. Reschke, A. Ruocco, C. Ng, B. Chapuy, V. Léopold, M. Sjoberg, T. M. Keane, A. Verma, U. Ala *et al.*, "The BRAF pseudogene functions as a competitive endogenous RNA and induces lymphoma in vivo," *Cell*, vol. 161, no. 2, pp. 319–332, 2015.

[20] C. Vennin, N. Spruyt, F. Dahmani, S. Julien, F. Bertucci, P. Finetti, T. Chassat, R. P. Bourette, X. Le Bourhis, and E. Adriaenssens, "H19 non coding RNA-derived miR-675 enhances tumorigenesis and metastasis of breast cancer cells by downregulating c-Cbl and Cbl-b," *Oncotarget*, vol. 6, no. 30, pp. 29 209–23, 2015.

[21] C. Yang, D. Wu, L. Gao, X. Liu, Y. Jin, D. Wang, T. Wang, and X. Li, "Competing endogenous RNA networks in human cancer: hypothesis, validation, and perspectives." *Oncotarget*, vol. 7, no. 12, pp. 13 479–13 490, 2016.

[22] D. S. Sardina, S. Alaimo, A. Ferro, A. Pulvirenti, and R. Giugno, "A novel computational method for inferring competing endogenous interactions," *Briefings in Bioinformatics*, vol. 18, no. 6, p. bbw084, 2017.

[23] G. C. Jin, S. Park, H. L. Chae, S. K. Hong, Y. S. Seung, R. Tae-Young, S. Jong-Hyuk, W. Suh, H. Seok-Jin, L. Key-Hwan *et al.*, "ZNF224, Kruppel like zinc finger protein, induces cell growth and apoptosis-resistance by down-regulation of p21 and p53 via miR-663a." *Oncotarget*, vol. 7, no. 21, pp. 31 177–90, 2016.