Incomplete Data Classification Based on Multiple Views

Ming Sun, Hongzhi Wang(✉), Fanshan Meng, Jianzhong Li, and Hong Gao

Departmemt of Computer Science and Technology,
Harbin Institute of Technology, Harbin, China
{mingsun,wangzh,lijzh,honggao}@hit.edu.cn, fanshan.mfs@gmail.com

Abstract. Missing values have negative impacts on big data analysis. However, in absence of extra knowledge, exact imputation can hardly be conducted for many data sets. Therefore, we have to tolerate missing values and perform data mining on incomplete data sets directly. To achieve high quality data mining on incomplete data, we propose a classification approach based on multiple views. We use various complete views of the data set to generate the base classifiers and combine the results of base classifiers. Since the amount of base classifiers will affect the effectiveness and efficiency of the classification, we aim to find proper view sets. We prove that the view set selection problem is an NP-hard problem and develop an approximation algorithm with approximate ratio $ln|S| + 1$ where S is the feature set of original data set. Extensive experimental results demonstrate the efficiency and effectiveness of the proposed approaches.

1 Introduction

Classification is an important kind of techniques of data mining. Clearly, missing values will affect the quality of classification result. Taking medical data sets as an example, patient A's blood pressure value is lost, and we cannot predict what it is exactly in absence of extra knowledge. If we fill in a value which is in the blood pressure normal range but the true data is beyond the scope, the model being trained will be affected by the wrong data and emerge fatal mistakes, and vice versa. Someone's life will be lost regrettably due to the mistakes. Thus, incompleteness has to be handled for classification.

Current solutions can be divided into two types. One solution [1–6] trains the model from complete data subset and imputes missing values based on that model. The other deletes samples with missing values, and then performs classification directly. In absence of extra knowledge, the cost of the former solution is high, and it can hardly get exact values, which may lead to a low accuracy of classification and inefficiency. The efficiency of the latter one indeed wins, but the accuracy is disturbed by the samples with missing values.

Motivated by this, in this paper, we attempt to conduct classification on incomplete data directly to acquire relatively high-quality results. To achieve this goal, we need to utilize the complete share in data sufficiently. We use

F. Li et al. (Eds.): APWeb 2016, Part II, LNCS 9932, pp. 239–250, 2016.
DOI: 10.1007/978-3-319-45817-5_19

Table 1. A dirty data set D

	s_1	s_2	s_3	s_4	s_5	s_6	T
r_1	√	×	√	√	×	√	1
r_2	√	√	√	√	×	√	1
r_3	√	×	√	√	√	√	0
r_4	×	×	√	√	√	√	1
r_5	×	√	√	×	√	×	0
r_6	√	√	√	×	√	√	0
r_7	×	√	×	√	√	√	1

Table 2. Part of extracted views

V_i	S_i	R_i
V_1	$S_1 = \{s_1, s_3, s_4, T\}$	$R_1 = \{r_1, r_2, r_3\}$
V_2	$S_2 = \{s_2, s_4, s_5, T\}$	$R_2 = \{r_7\}$
V_3	$S_3 = \{s_1, s_2, s_5, T\}$	$R_3 = \{r_6\}$
V_4	$S_4 = \{s_2, s_5, s_6, T\}$	$R_4 = \{r_6, r_7\}$
V_5	$S_5 = \{s_3, s_4, T\}$	$R_5 = \{r_1, r_2, r_3, r_4\}$
V_6	$S_6 = \{s_2, s_6, T\}$	$R_6 = \{r_2, r_6, r_7\}$
...	...	

an example to illustrate this point. Consider an incomplete data set D with the feature set $S(s_1, s_2, s_3, s_4, s_5, s_6, T)$, where T is the column of type and the sample set $R(r_1, r_2, r_3, r_4, r_5, r_6, r_7)$, as shown in Table 1, where "√" denotes the data is available while "×" denotes it is missing. Table 2 shows the part of views from D. V_i denotes the ith view queried, S_i is the feature set of the ith view, and R_i is the sample set. Each view is a complete share of original data. The challenge is that the complete share of data may not be sufficient enough for classification due to the difference in incomplete values. Fortunately, the data have relevance. That is, some part of data could be implied from other part.

With the consideration of keeping relevance among data and not trying useless imputation, we propose a novel method which considers different complete share of the incomplete data. A base classifier is trained from each view. With these base classifiers, we adopt the voting mechanism to generate the final classification results. Clearly, the amount of views affects the efficiency as well as effectiveness of the algorithms. To achieve high performance, we do not have to use all the views, but only to find a combination of views which can cover all the features of dirty data set and a great majority of samples.

The contributions of this paper could be summarized as three points.

1. We take different views of incomplete data into account and regard them as base classifiers, which mitigates decline of accuracy caused by missing data.
2. To keep all features in the data set during training and ensure the efficiency, we propose view selection problem which we prove to be NP-hard and develop an efficient weighted greedy algorithm called *VS* with approximate ratio, $ln|S| + 1$, where S is the feature set of original data.
3. Experimental results show that our approach performs better and spend less time than the imputation-based solution.

The remainder of this paper is organized as follows. We introduce our framework in the next section which can acquire a relatively high-quality classification result from an incomplete data set directly. Section 3 introduces our view selection phase. Experimental results are reported in Sect. 4 and conclusions are presented in Sect. 5.

2 The Framework

In this section, we introduce the framework of our approach. Our method has three phases, preprocessing, view selection and combination. We acquire views satisfying the rules in preprocessing phase, which prepares for view selection phase. After selecting the views used to train base classifiers, we combine their classification results to obtain the final result. To describe the approach clearly, we use $\mathcal{F}$ to represent the set of views satisfying the rules which will be described concretely in Sect. 2.1, and use ℓ to represent the combination of views selected to be the base classifiers.

2.1 Preprocessing

In this phase, we give some rules to limit the amount of views for selection which mainly reduce the calculation burden on view selection phase.

Rule 1: $\forall\ V_i \in \mathcal{F}, \mid R_i \mid \geq \theta \mid R \mid, \theta \in [0.1, \delta]$.

With this rule, we restrict that the sample coverage of each view should exceed a threshold θ which avoids the extreme case that one view indeed covers many features but only cover few samples. We also restrict the value of θ should be less than δ which agrees with common sense.

Rule 2: $\forall\ V_i \in \mathcal{F}, \mid S_i \mid > 2$.

This rule ensures that the views only containing two features (including the column T) are abandoned because it is ridiculous to classify with a single feature.

Rule 3: If $\exists\ V_i$ and V_j, where $R_i = R_j$ and $S_i \subseteq S_j$, then delete V_j from $\mathcal{F}$.

Once there exists two views satisfying the rule above, the view with more features will be deleted. From the point of samples, they both have the same performance. The only distinctive between them is that one or more extra features are considered redundant.

Rule 1 makes this phase similar to the mining of frequent item set because the set of threshold θ is same as the support count in Apriori [7], while Rule 3 change the process into the mining of frequent closed item sets. As the combination of these three rules, we choose the algorithm in [8] to filter the views for $\mathcal{F}$.

2.2 View Selection

Since the first phase have filtered the views for $\mathcal{F}$, in this phase, our goal is to find the combination ℓ which contain the fewest views but can generally express the incomplete data set with current available data from $\mathcal{F}$.

We will formally define view selection problem which is proven to be NP-hard in Sect. 3. Furthermore, we develop an approximate algorithm whose approximate ratio is $ln|S| + 1$.

2.3 Combination

In this phase, each view selected into ℓ will be trained to be a base classifier and we adopt the voting mechanism to generate the final classification results. We can use any mining technique to train the base classifiers.

Given an instance X_q to be classified, because of the voting mechanism, the final result $F(X_q)$ depends on the result $f_i(X_q)$ of all the base classifiers. We regard the result of each classifier equally, that is,

$$F(X_q) \leftarrow \arg\max_{V_i \in \ell} f_i(X_q).$$

3 View Selection

In this section, we focus on view selection problem in the second phase. We formally define the view selection problem as follows:

$$\min \ \mid \ell \mid, \ \ell \subseteq \mathcal{F}$$
$$s.t. \quad \begin{cases} \bigcup\limits_{V_i \in \ell} S_i = S, \\ \mid \bigcup\limits_{V_i \in \ell} R_i \mid \geq \delta \mid R \mid \quad \delta \in [0.5, 1]. \end{cases}$$

In this definition, on condition that we attempt to select the fewest views to cover all the features and a great majority of samples, each view abundantly keeps the relationship among different features and guarantees a good generalization ability.

Theorem 1. View selection problem is NP-hard.

Proof (Sketch). To prove that view selection problem is NP-hard, we show that set-covering problem [9] could be reduced to view selection problem. In other words, we need to show how to reduce any instance of set-covering problem to an instance of view selection problem in polynomial time. When $\delta = 1$ and $\forall \ V_i \in \mathcal{F}, \mid R_i \mid = \delta \mid R \mid$, we can use induction to express any instance of set covering as that of view selection. Thus, if the view selection problem has a solution, the set covering problem has a solution. Since set-covering is NP-hard, view selection problem is NP-hard. □

Our problem is similar to the set-covering problem in two dimensions. Unlike the set-covering problem, we cannot directly use the greedy strategy. Concretely speaking, when we want to use the greedy strategy to pick a view, we may indeed choose the one that covers the greatest number of remaining uncovered elements of one dimension, but we cannot guarantee that how many uncovered elements of the other dimension we have covered. We must consider the coverage of features and the coverage of samples simultaneously. Moreover, we do not need to cover all the elements of samples while we must cover all the elements of features. With regard to our problem, we decide to change one dimension into the weight of the other dimension which will affect the selection of view.

Algorithm 1. View Selection $(S, R, \mathcal{F})$

1: $U = S, T = R$
2: $\ell = \varnothing, M = \varnothing$
3: **while** $U \neq \varnothing$ **do**
4: **if** $\mid M \mid \geq \delta \mid R \mid$ **then**
5: **for all** i such that $V_i \in \mathcal{F}$ **do**
6: $\omega_i = 1$
7: **end for**
8: **else**
9: **for all** i such that $V_i \in \mathcal{F}$ **do**
10: $\omega_i = \mid R_i \backslash M \mid$
11: **end for**
12: **end if**
13: select an $V_i \in \mathcal{F}$ that maximizes $\omega_i \times \mid S_i \cap U \mid$
14: $U = U \setminus S_i, T = T \backslash R_i$
15: $\ell = \ell \cup \{V_i\}, M = M \cup R_i$
16: **end while**
17: return ℓ

The algorithm works as follows. The set U and T contains, at each stage, the set of remaining uncovered features and samples. The set ℓ is used to contain which views have been chosen while the set M is to record which samples have been covered so that we can compute the weight of every view. Line 4–12 update the weight of each subset. When the second constraint that the samples should be covered at least $\delta \mid R \mid$ is satisfied, ω_i is 1 which means the value of weight will not affect the selection of view any more. Otherwise, ω_i is $\mid R_i \backslash M \mid$. Line 13 is the greedy decision-making step, choosing a view V_i whose product between ω_i and the amount of features will covered the first time is the largest. After V_i is selected, line 14 removes its features from U and its samples from T, and line 15 places V_i into ℓ and add R_i, i.e. the samples of V_i into M. When the algorithm terminates, the set ℓ contains a subset of $\mathcal{F}$ that covers S.

Time Complexity Analysis. We can easily verify *VS* to run in time polynomial in $\mid S \mid$, $\mid R \mid$ and $\mid \mathcal{F} \mid$. Since the number of iterations of the loop on lines 3–16 is bounded from above by $\min(\mid S \mid, \mid \mathcal{F} \mid)$, and we can implement the loop body to run in time $O(\mid R \mid\mid \mathcal{F} \mid + \mid S \mid\mid \mathcal{F} \mid)$. For most data sets, the amount of samples is over that of features a lot ($\mid R \mid \gg \mid S \mid$), so $O(\mid R \mid\mid \mathcal{F} \mid + \mid S \mid\mid \mathcal{F} \mid) = O(\mid R \mid\mid \mathcal{F} \mid)$. Even a straightforward implementation of our algorithm runs in time $O(\mid R \mid\mid \mathcal{F} \mid \min(\mid S \mid, \mid \mathcal{F} \mid)$.

Approximate Ratio Bound Analysis. We now show that the weighted greedy algorithm returns a combination of views with the cost not too much larger than an optimal combination. For convenience, in this paper we denote the dth harmonic number $H_d = \sum_{i=1}^{d} 1/i$ by $H(d)$. As a boundary condition, we define $H(0) = 0$.

Theorem 2. *VS* algorithm is polynomial-time $\rho(n)$-approximation algorithm, where $\rho(n) = \ln |S| + 1$.

Proof. We have already shown that *VS* runs in a polynomial time in time complexity analysis. We define the cost of the algorithm is the sum of the reciprocal of the weight when the view is selected in that our problem attempt to find the minimal views satisfying two constraints. We try to select a view with a larger weight even though it may not cover the most uncovered features till the second constraint is satisfied. To show that *VS* is a $\rho(n)$-approximation algorithm, we denote $V(i)$ as the ith view selected by *VS*. The algorithm incurs a cost of $\frac{1}{\omega(i)}$ when it adds $V(i)$ to ℓ where $\omega(i)$ is the weight of $V(i)$ when $V(i)$ is selected. We spread this cost of selecting $V(i)$ evenly among the features covered for the first time by $S(i)$. Let c_x denote the cost allocated by to feature x, for each $x \in S$. Each feature is assigned a cost only once, when it is covered for the first time. If x is covered for the first time by V(i), then $c_x = \frac{1}{\omega(i)|S(i)-(S(1)\cup S(2)\cup\cdots\cup S(i-1))|}$.

The cost of ℓ is the sum of the cost of every feature in S after the implementation of *VS*, so we have

$$cost(\ell) = \sum_{x \in S} c_x \tag{1}$$

Each element $x \in S$ is in at least one set in the optimal cover ℓ^*, and so we have

$$\sum_{V_j \in \ell^*} \sum_{x \in S_j} c_x \geq \sum_{x \in S} c_x \tag{2}$$

Combining Eq. (1) and inequality (2), we have that

$$cost(\ell) \leq \sum_{V_j \in \ell^*} \sum_{x \in S_j} c_x \tag{3}$$

Now, we estimate the equation $\sum_{x \in S_j} c_x$. Consider any j such that $V_j \in \mathcal{F}$ and $i = 1, 2, \ldots, |\ell|$, and let $u_i = |S_j - (S(1) \cup S(2) \cup \cdots \cup S(i-1))|$ be the number of features in S_j that remain uncovered after the algorithm has selected sets $V(1), V(2), \ldots, V(i-1)$. We define $u_0 =| S_j |$ to be the number of features of S_j, which are all uncovered. Let k be the least index such that $u_k = 0$, so that every feature in S_j is covered by at least one of sets $S(1), S(2), \ldots, S(k)$ and some features in S_j is uncovered by $S(1) \cup S(2) \cup \cdots \cup S(k-1)$. Then, $u_{i-1} \geq u_i$, and $u_{i-1} - u_i$ features of S_j are covered for the first time by S_j, for $i = 1, 2, \ldots, k$. Thus, $\sum_{x \in S_j} c_x = \sum_{i=1}^{k} (u_{i-1} - u_i) \cdot \frac{1}{\omega(i)|S(i)-(S(1)\cup S(2)\cup\cdots\cup S(i-1))|}$. Observe that $|S(i)-(S(1)\cup S(2)\cup\cdots\cup S(i-1))| \geq |S_j-(S(1)\cup S(2)\cup\cdots\cup S(i-1))| = u_{i-1}$, because the greedy choice of $S(i)$ guarantees that S_j cannot have a bigger product between the weight and the amount of new features covered than $S(i)$ does (otherwise, the algorithm would have chosen S_j). Consequently, we obtain

$$\begin{aligned}
\sum_{x\in S_j} c_x &\le \sum_{i=1}^{k}(u_{i-1}-u_i)\cdot\frac{1}{\omega(i)\cdot u_{i-1}} \le \sum_{i=1}^{k}(u_{i-1}-u_i)\cdot\frac{1}{\omega_j\cdot u_{i-1}} \\
&= \frac{1}{\omega_j}\sum_{i=1}^{k}(u_{i-1}-u_i)\cdot\frac{1}{u_{i-1}} \\
&= \frac{1}{\omega_j}\sum_{i=1}^{k}\sum_{q=u_i+1}^{u_{i-1}}\frac{1}{u_{i-1}} \\
&\le \frac{1}{\omega_j}\sum_{i=1}^{k}\sum_{q=u_i+1}^{u_{i-1}}\frac{1}{q} = \frac{1}{\omega_j}\sum_{i=1}^{k}(\sum_{q=1}^{u_{i-1}}\frac{1}{q}-\sum_{q=1}^{u_i}\frac{1}{q}) \\
&= \frac{1}{\omega_j}\sum_{i=1}^{k}(H(u_{i-1})-H(u_i)) = \frac{1}{\omega_j}[H(u_0)-H(u_k)] \\
&= \frac{1}{\omega_j}[H(u_0)-H(0)] = \frac{1}{\omega_j}H(u_0) \\
&= \frac{1}{\omega_j}H(\mid S_j \mid) \le \frac{1}{\omega_j}H(\max\mid S_q \mid: V_q\in\mathcal{F})
\end{aligned} \tag{4}$$

From the inequality (3) and (4), we have

$$\begin{aligned}
cost(\ell) &\le \sum_{V_j\in\ell^*}\sum_{x\in S_j} c_x \le \sum_{V_j\in\ell^*}\frac{1}{\omega_j}H(\max\mid S_q \mid: V_q\in\mathcal{F}) \\
&= H(\max\mid S_q \mid: V_q\in\mathcal{F})\sum_{V_j\in\ell^*}\frac{1}{\omega_j} \\
&= H(\max\mid S_q \mid: V_q\in\mathcal{F})\sum_{V_j\in\ell^*} cost(\ell^*)
\end{aligned}$$

For harmonic progression $H(n)$, there is $\ln(n+1) \le H(n) = \sum_{k=1}^{n} 1/k \le \ln n + 1$. Since $\mid S_q \mid \le \mid S \mid$, $cost(\ell) \le H(\mid S\mid)cost(\ell^*) \le (\ln \mid S \mid +1)cost(\ell^*)$, thus proving the theorem. □

4 Experimental Results

In this section, we conduct extensive experiments to verify the effectiveness and efficiency of the proposed framework on real data sets. The basic information of data sets from UCI machine learning repository[1] we used is shown in Table 3. The amount of features and samples of both data set is suitable for our approach. Besides, Heart Disease Data Set hardly has missing values so that we need to randomly inject missing values, while Chronic Kidney Disease Data Set itself has some real missing values. We generate the incomplete data sets by randomly

[1] http://archive.ics.uci.edu/ml/.

Table 3. Data sets used in experiments (from UCI)

Data set	Features	Samples
Heart disease	14	300
Chronic kidney disease	25	400

removing some values evenly to control the ratio of missing values. We generate the test data set from the original data set with three parameters, the number of extracted tuples (#tuple), the number of extracted features (#feature) and the missing ratios (#missing). The default values of these parameters are 300, 13 and 0.2. The default values of θ and δ are 0.5 and 0.8, respectively. Moreover, we choose KNN [10] to be the classification technique of basic classifiers and comparative methods where k is set to be 3. The reason why we choose KNN is its insensitivity to the amount of features in each base classifiers, unlike neural networks. Experimentally, we have discovered the accuracy we acquired is the best when $k = 3$. We use run time and accuracy to measure the efficiency and effectiveness, respectively. Accuracy is defined as $Accuracy = \frac{|D_{right}|}{|D|}$ where D represents the original data set and D_{right} represents the set containing the tuples which is classified correctly.

4.1 Comparison to Other Methods

First of all, we compared our approach with another two approaches. The former one classification on the data set which is imputed based on Bayesian network [11]. The latter one performs classification on the incomplete data set directly.

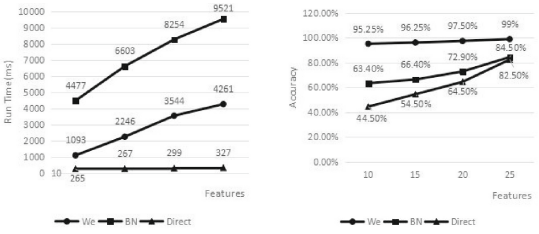
(a) Run Time (b) Accuracy

Fig. 1. Scalability in tuples

Varying #tuple. To test the scalability, we use Heart Disease Data Set and vary #tuple from 150 to 300. The experimental results are shown in Fig. 1.

The experiment shows that with the increase of sample amount, the accuracy increases while it costs more time. More samples means a better generalization of original data set. Besides, we can observe that our framework acquires higher

accuracy than another two methods but run time only outperforms BN [11]. DirectKNN is disturbed by the samples with missing values. The fewer amount of samples indeed reduce the run time but sacrifices the accuracy. The results of BN [11] are also interpretable. It first spends much to use the correlation between features to construct Bayesian networks and then uses it to predict the possible value, whose cost is large. Our framework just finds a suitable combination of views which regards each sample as a entirety, but not considering the inner relationship of features in each sample. In terms of accuracy, BN [11] utilizes the correlation between features to impute the missing values which usually appear in some certain features. However, when the correlation is weak and the missing value appears evenly, the classification accuracy decreases reasonably.



(a) Run Time (b) Accuracy

Fig. 2. Scalability in features

Varying #feature. To test the scalability, we use Chronic Kidney Disease Data Set and vary #feature from 10 to 25. The experimental results are shown in Fig. 2.

The experiment shows that with the increase of sample feature, the accuracy is improved significantly while it costs more time, since more features and relationships are kept to express the original data set. The comparison of run time and accuracy between these three methods is similar as the experimental results of varying #tuple for the same reasons. Besides, we observe that the impact of feature amount on run time is more significant on that of the number of samples. Because more features lead to larger computation cost especially in the phase of filtering views for $\mathcal{F}$.

Varying #missing. To test the impacts of missing ratio, we use Heart Disease Data Set and vary #missing from 0.1 to 0.4. The results are shown in Fig. 3.

From Fig. 3, it is observed that the increase of ratio will reduce run time and accuracy. In terms of run time, more missing values decrease the scale of $\mathcal{F}$ and Bayesian networks to great extent. With regard to accuracy, its decrease with the increase of ratio agrees with common sense since the larger missing ratio means the fewer amount of available data.

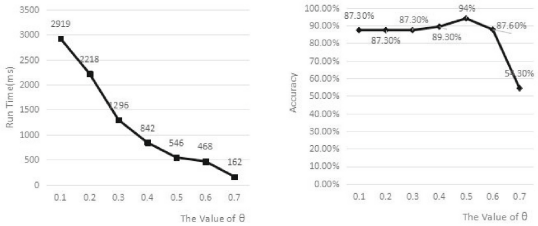
(a) Run Time (b) Accuracy

Fig. 3. Impacts on missing ratio

4.2 Impacts of Parameters

Our framework has two important parameters θ and δ, which affect the run time and accuracy. Thus, we conduct experiments to analyze the impact of them.

Impacts of θ**.** The set of θ mainly affects the construction of $\mathcal{F}$ in the pre-processing phase. We use Heart Disease Data Set and vary the value of θ from 0.1 to 0.7. The results are shown in Fig. 4.

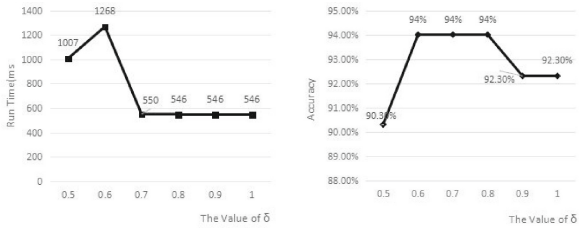


(a) Run Time (b) Accuracy

Fig. 4. Impacts of θ

In terms of run time, the increase of θ heightens the standard to filtering views for $\mathcal{F}$, then fewer views meet the standard, which leads to the decrease of run time. Considering accuracy, it is observed that when $\theta = 0.5$, the accuracy is the highest. The reason is that when θ is less than 0.5, some useless views whose amount of sample is few added to $\mathcal{F}$ disturb the view selection because of the low ability to generalize. When θ is more than 0.5, some views with more samples but fewer features are selected preferentially which cannot represent the data set more complicatedly and some views with suitable samples and features are ignored. That is why in most experiments, we keep the value of θ 0.5.

Impacts of δ**.** The set of δ mainly affects the implementation of *VS* in the view selection phase. We use Heart Disease Data Set and vary the value of δ from 0.5 to 1. The results are shown in Fig. 5.



(a) Run Time (b) Accuracy

Fig. 5. Impacts of δ

With regard to run time, the increase of δ extends the time for calculation of the weight of each view in $\mathcal{F}$ which causes the increase of run time. For accuracy, when δ is 0.5, too few samples are not covered so the low ability to generalize leads to the low accuracy. When δ is 0.9 or 1, some views with fewer features but more samples are selected preferentially which can not represent the data set more complicatedly and some views with suitable samples and features are ignored. However, for this data set, when δ is from 0.6 to 0.8, we get the same accuracy which means *VS* may select the same combination on the condition that the view selected finally covers at least 20 % uncovered samples. Setting $\delta = 0.8$ is a balance action for both feature coverage and sample coverage.

We should take attention to the fact that the most suitable value of θ and δ in this part varies due to different data sets.

4.3 Effectiveness of View Selection Algorithm

To test the effectiveness of our *VS* algorithm. we use Heart Disease Data Set and the experimental results are shown in Table 4. For comparison, we randomly generate 10 groups of views in $\mathcal{F}$ with the same number of the views selected with our approach, train the learners with each group and report the run time and accuracy. We also compare with the optimal brute-force searching algorithm.

The approximate answer obtained from *VS* whose accuracy is still high spends far less time than that of the optimal solution. In addition, either the maximal accuracy, the average one or the minimal one, the random selection of views cannot beat *VS*. Both facts show the effectiveness of our *VS* algorithm.

Table 4. Comparison

	OPT	*max*	*min*	*avg*	*OPT**
Time (ms)	1521	–	–	–	4223
Accuracy	94 %	87.6 %	84.3 %	86 %	–

5 Conclusions

In this paper, we study the classification method of incomplete data which considers different views of it. To ensure the effectiveness and efficiency, we propose view selection problem which is proven to be NP-hard and develop an efficient weighted greedy algorithm called *VS* with approximate ratio, $ln|S| + 1$, where S is the feature set of original data set. Then we use a voting mechanism to generate the final classification results. Experiments results show that our *VS* performs well in classification and spend less time than an imputation-based solution. Our future work includes taking inconsistent data into consideration and performing more experiments and comparisons.

Acknowledgement. This paper was partially supported by National Sci-Tech Support Plan 2015BAH10F01 and NSFC grant U1509216,61472099,61133002 and the Scientific Research Foundation for the Returned Overseas Chinese Scholars of Heilongjiang Provience LC2016026.

References

1. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, T., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics **17**(6), 520–525 (2001)
2. Oba, S., Sato, M.A., Takemasa, I., Monden, M., Matsubara, K.I., Ishii, S.: A Bayesian missing value estimation method for gene expression profile data. Bioinformatics **19**(16), 2088–2096 (2003)
3. Zhu, X., Zhang, S., Jin, Z., Zhang, Z., Xu, Z.: Missing value estimation for mixed-attribute data sets. IEEE Trans. Knowl. Data Eng. **23**(1), 110–121 (2011)
4. Setiawan, N.A., Venkatachalam, P.A., Hani, A.F.M.: Missing attribute value prediction based on artificial neural network and rough set theory. In: International Conference on BioMedical Engineering and Informatics, BMEI 2008. IEEE (2008)
5. Abdella, M., Marwala, T.: The use of genetic algorithms and neural networks to approximate missing data in database. In: IEEE 3rd International Conference on Computational Cybernetics, ICCC 2005. IEEE (2005)
6. Hagan, M.T., Demuth, H.B., Beale, M.H., De Jesús, O.: Neural Network Design. PWS publishing company, Boston (1996)
7. Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: Proceedings of 20th International Conference very large Data Bases, VLDB (1994)
8. Pei, J., Han, J., Mao, R., et al.: Closet: an efficient algorithm for mining frequent closed itemsets. In: ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (2000)
9. Christofides, N.: Graph Theory–An Algorithmic Approach. Academic Press Inc., New York (1975)
10. Cover, T.M., Hart, P.E.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theor. **13**(1), 21–27 (1967)
11. Jin, L.: Research on missing value imputation of incomplete data. Harbin Institute of Technology (2013)