



Automatic Classifier Selection Based on Classification Complexity

Liping Deng¹, Wen-Sheng Chen^{1,2}, and Binbin Pan^{1,2}(✉)

¹ College of Mathematics and Statistics, Shenzhen University, Shenzhen 518060, People's Republic of China
{chenws, pbb}@szu.edu.cn

² Guangdong Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, People's Republic of China

Abstract. Choosing a proper classifier for one specific data set is important in practical application. Automatic classifier selection (CS) aims to recommend the most suitable classifiers to a new data set based on the similarity with the historical data sets. The key step of CS is the extraction of data set feature. This paper proposes a novel data set feature that characterizes the classification complexity of problems, which has a close connection with the performance of classifiers. We highlight two contributions of our work: firstly, our feature can be computed in a low time complexity; secondly, we theoretically show that our feature has connection with generalization errors of some classifiers. Empirical results indicate that our feature is more effective and efficient than the existing data set features.

Keywords: Automatic classifier selection · Data set feature
Data set similarity

1 Introduction

Classification is one of the most important tasks in machine learning. A great number of classifiers were putted forward in recent decades to tackle various kinds of classification problems arose in real world, such as support vector machine, decision tree, AdaBoost, artificial neural networks, and so on. Does there exist a classifier that significantly performs better than any other classifiers on most of data sets? Some literatures have done in-depth investigations on this problem. The No Free Lunch Theorem [1] tells us that there does not exist such classifier. If classifier \mathcal{A}_1 outperforms \mathcal{A}_2 on some data sets, then there must exist as many other data sets on which \mathcal{A}_2 outperforms \mathcal{A}_1 . In [2], authors analyzed the performances of three classifiers on some data sets and they did not observe which classifier is significantly better than the others. Furthermore, [3] conducted classification experiments using 179 classifiers and 121 data sets and showed that there is no optimal classifier. These results indicate that classifiers have preference on different types of data sets. Therefore, which classifier(s) would be selected for a given classification problem?

One idea is to use cross validation for all possible classifiers to find the best classifier. However, this procedure is time-consuming. An efficient alternative approach is automatic classifier selection based on data set similarity [4–7, 10], or classifier selection (CS) for short. We believe that the performances of classifiers on similar data sets should be close. Since different data sets may vary in sample size, dimensions, classes and attributes, how to measure the similarity between data sets is a critical step of CS. The common method is to extract data set feature by designing a feature extraction function (or called meta-learning) and then compute the similarity between these features. There is an intrinsic relationship between classifier performance and data set feature [9]. Therefore, the recommendation heavily depends on the effectiveness of data set feature. Furthermore, the feature should be calculated in a low time complexity, which is a bottleneck of CS.

A number of data set features have proposed. These features are extracted from different aspects of a data set: (i) statistics and information theory (SI) [7, 10]; (ii) model structure (MS) [5]; (iii) problem complexity (PC) [4]; (iv) landmarking (LM) [6]. Especially, PC and LM characterize the classification complexity of problems (we call it *complexity*) using a set of geometrical metrics or basic classifiers. The complexity is expected to highly correlate to the performances of classifiers [11]. In other words, the performances of classifiers on data sets that have similar complexity should be close. Therefore, complexity plays a vital role in CS. However, the data set features extracted by PC and LM have two shortages: (i) time-consuming; (ii) no theoretical connection with performances of classifiers. It is observed that PC and LM did not perform well in some literatures [5, 7], which means that they cannot characterize the complexity accurately.

To remedy the aforementioned shortcomings of PC and LM, this paper uses a set of geometrical and statistical metrics to describe the complexity of two-class data set, then these metrics are united as data set feature. We use KNN classifier as recommendation algorithm for CS. For multi-class classification problem, we split the problem into two-class problems using one-vs-one strategy. Compared with PC and LM, our work has improvements in two aspects: computation efficiency and theoretical guarantee. Empirical results demonstrate the effectiveness and efficiency of our method.

The rest of the paper is structured as follows. We briefly introduce the related works in Sect. 2. Section 3 presents our data set feature. The classifier selection algorithm is given in Sect. 4. Empirical investigations are discussed in Sect. 5 and conclusions are drew in Sect. 6.

2 Related Work

The key problem of CS is feature extraction. To the best of our knowledge, there are four kinds of features.

Statistical Feature: This feature can be categorized into two kinds. The first kind describes the data set using a group of statistical and information theory

characteristics [10]. The second kind is based on summary statistics. Song [7] characterizes the data set structure by computing the frequencies of itemsets generated from binary data sets. Non-binary data set needs to be transformed to binary data set, which would be time-consuming when the attributes of data set are continuous.

Problem Complexity Feature: Twelve measures are designed to describe the geometrical complexity of decision boundary of two-class problems [11]. Cano [12] claimed that some of the measures have little connection with the performances of classifiers. Bernado [4] selected six measures to characterize data set.

Landmarking Feature: This feature [6] utilizes the performances of a set of basic classifiers (called *landmarkers*) to describe the data set. Therefore, the similar features indicate that data sets may belong to the subspace of the same performance. The chosen landmarkers must be significantly different.

Model Structure Feature: The statistical information of a model generated from data set is collected as feature. In this category, decision tree is usually considered [5], from which we gather a set of statistics like maximum/minimum number of nodes, length of longest/shortest branches, and so on.

The aforementioned features belong to experimental origin. However, a theoretical investigation would be more persuasive. Furthermore, these features are computationally expensive.

3 Proposed Feature

In this section, we firstly propose several metrics of complexity for CS. Then the theoretical connections between two metrics and generalization errors of some classifiers are investigated. Finally, we present our data set feature and similarity measurement criterion.

3.1 Metrics of Complexity

Given a two-class data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$ in input space \mathcal{X} , where \mathbf{x}_i , $i = 1, 2, \dots, n$ are data points, and y_i is the binary class label, i.e., $y_i \in \{1, -1\}$. Let $\mathbf{y} = [y_1, y_2, \dots, y_n]^\top$ represents the vector formed with n labels. We use n_- and n_+ to represent the amount of samples labeled -1 or 1 , respectively. Note that $n_- + n_+ = n$.

For a given kernel function $k(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$, where ϕ is a nonlinear mapping that maps $\mathbf{x} \in \mathcal{X}$ to a reproduce kernel hilbert space (RKHS) \mathcal{H} , an $n \times n$ kernel matrix \mathbf{K} is generated from \mathcal{D} as

$$K_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle, i, j = 1, 2, \dots, n.$$

\mathbf{K} is a symmetric positive and semi-definite matrix that totally preserves the geometrical structure of \mathcal{D} . Our five metrics of complexity are based on \mathbf{K} .

Kernel Alignment. This metric, which is known as centered kernel target alignment (KA) [13], is defined as

$$KA(\mathbf{K}_c, \mathbf{y}\mathbf{y}^\top) = \frac{\langle \mathbf{K}_c, \mathbf{y}\mathbf{y}^\top \rangle_F}{\sqrt{\langle \mathbf{K}_c, \mathbf{K}_c \rangle_F \langle \mathbf{y}\mathbf{y}^\top, \mathbf{y}\mathbf{y}^\top \rangle_F}}, \tag{1}$$

where \mathbf{K}_c is a centralized kernel matrix of \mathbf{K} , $\langle \cdot, \cdot \rangle_F$ denotes Frobenius inner-product and $\mathbf{y}\mathbf{y}^\top$ is called the target matrix. $KA \in [0, 1]$ since $\langle \mathbf{K}_c, \mathbf{y}\mathbf{y}^\top \rangle_F \geq 0$.

The numerator of (1) can be expanded as

$$\begin{aligned} \langle \mathbf{K}_c, \mathbf{y}\mathbf{y}^\top \rangle_F &= \mathbf{y}^\top \mathbf{K}_c \mathbf{y} = \sum_{i=1}^n \sum_{j=1}^n y_i y_j (K_c)_{ij} \\ &= \sum_{y_i=y_j} (K_c)_{ij} - \sum_{y_i \neq y_j} (K_c)_{ij}. \end{aligned}$$

Therefore, KA measures the difference between the within-class and between-class distances of data set. A bigger KA indicates that the corresponding data set is more separable. The most time-consuming calculations of KA are the centralization of \mathbf{K} and $\langle \mathbf{K}_c, \mathbf{K}_c \rangle_F$, which take $O(n^2)$ time complexity.

Kernel Space-Based Separability. The centers of two classes in \mathcal{H} are calculated as

$$\begin{aligned} \phi_- &= \frac{1}{n_-} \sum_{y_i=-1} \phi(\mathbf{x}_i), \\ \phi_+ &= \frac{1}{n_+} \sum_{y_i=1} \phi(\mathbf{x}_i), \end{aligned} \tag{2}$$

respectively. KS [14] is defined as

$$KS(\mathbf{K}, \mathbf{y}) = \frac{std_- + std_+}{\|\phi_- - \phi_+\|_2}, \tag{3}$$

where

$$\begin{aligned} std_- &= \sqrt{\frac{\sum_{y_i=-1} \langle \phi(\mathbf{x}_i) - \phi_-, \mathbf{e} \rangle^2}{n_- - 1}}, \\ std_+ &= \sqrt{\frac{\sum_{y_i=1} \langle \phi(\mathbf{x}_i) - \phi_+, \mathbf{e} \rangle^2}{n_+ - 1}}, \end{aligned} \tag{4}$$

are the standard deviations of two classes projected along the direction $\mathbf{e} = \frac{\phi_- - \phi_+}{\|\phi_- - \phi_+\|_2}$ respectively, and $\|\cdot\|_2$ denotes 2-norm of vector.

$KS \in (0, +\infty]$ actually describes the samples' distribution along direction $\phi_- - \phi_+$. A smaller KS means that the data set is more separable. KS needs $O(n^2)$ time complexity.

Overlap Region. We propose a metric that compute the ratio of the overlapped region of two classes to the total region of two classes along direction \mathbf{e} , denoted as ROR. Suppose that the projected data of one class fall into $[a_1, b_1]$, where a_1, b_1 are the minimum and maximum values of the projected data, and the other class falls into $[a_2, b_2]$. Let $U = [a_1, b_1] \cap [a_2, b_2]$ and $V = [a_1, b_1] \cup [a_2, b_2]$ be intersection and union of these two intervals, respectively. ROR is defined as

$$\text{ROR} = \begin{cases} 0, & U = \emptyset, \\ \frac{\max(U) - \min(U)}{\max(V) - \min(V)}, & U \neq \emptyset, \end{cases} \tag{5}$$

where $\min(\cdot)$ and $\max(\cdot)$ are the maximum and minimum values of interval respectively and \emptyset represents empty set. $\text{ROR} \in [0, 1]$ since U is a subset of V . When data set is linear separable, ROR is expected to zero. However, ROR will increase if data set is nonlinear separable. ROR also needs $O(n^2)$ time complexity.

Test of Equality of Means. Now we treat kernel matrix \mathbf{K} as a similarity matrix. The following measure depends on the assumption that the similarity among within-class data is higher than between-class data. We first introduce two vectors extracted from \mathbf{K} :

$$\begin{aligned} \mathbf{k}_W &= \{K_{ij} | i < j \wedge y_i = y_j\}, \\ \mathbf{k}_B &= \{K_{ij} | i < j \wedge y_i \neq y_j\}. \end{aligned} \tag{6}$$

We denote $n_W = \frac{n-(n-1)}{2} + \frac{n+(n-1)}{2}$ and $n_B = n - n_+$ represent the size of vectors \mathbf{k}_W and \mathbf{k}_B respectively. We see that \mathbf{k}_W is the collection of within-class similarity and \mathbf{k}_B is the collection of between-class similarity.

TEM [15] is defined as a variant of t-test to evaluate the equality of means of \mathbf{k}_W and \mathbf{k}_B :

$$\text{TEM}(\mathbf{K}, \mathbf{y}) = \frac{1}{n} \left| \frac{\bar{k}_W - \bar{k}_B}{\sqrt{\frac{\sigma_W^2}{n_W} + \frac{\sigma_B^2}{n_B}}} \right|, \tag{7}$$

where \bar{k}_W and σ_W^2 denote the mean and variance of \mathbf{k}_W respectively, and \bar{k}_B and σ_B^2 denote the mean and variance of \mathbf{k}_B respectively. TEM is very sensitive to the nonlinearity of decision boundary. A larger TEM reflects that the data set is more likely to be linearly separable. Here we normalized TEM by multiplying the reciprocal of n to eliminate the influence of sample size. TEM only utilizes the upper triangle elements of \mathbf{K} , which needs $O(n^2)$ time complexity.

Test of Equality of Variances. Let $\mathbf{k}_{WB} = \mathbf{k}_W \cup \mathbf{k}_B$ be the union of \mathbf{k}_W and \mathbf{k}_B . We define three new vectors as follows:

$$\begin{aligned} \mathbf{z}_W &= |\mathbf{k}_W - \tilde{\mathbf{k}}_W|, \\ \mathbf{z}_B &= |\mathbf{k}_B - \tilde{\mathbf{k}}_B|, \\ \mathbf{z}_{WB} &= |\mathbf{k}_{WB} - \tilde{\mathbf{k}}_{WB}|, \end{aligned} \tag{8}$$

where $|\cdot|$ represents element-wise absolute value, $\tilde{\mathbf{k}}_W$, $\tilde{\mathbf{k}}_B$ and $\tilde{\mathbf{k}}_{WB}$ are the medians of \mathbf{k}_W , \mathbf{k}_B and \mathbf{k}_{WB} respectively. TEV [15] is defined using Brown-Forsythe test to measure the equality of variances of \mathbf{k}_W and \mathbf{k}_B ,

$$\text{TEV}(\mathbf{K}, \mathbf{y}) = \left(1 - \frac{2}{n}\right) \frac{n_W(\bar{z}_W - \bar{z}_{WB})^2 + n_B(\bar{z}_B - \bar{z}_{WB})^2}{\sum_{i=1}^{n_W} [(z_W)_i - \bar{z}_W]^2 + \sum_{i=1}^{n_B} [(z_B)_i - \bar{z}_B]^2}, \quad (9)$$

where \bar{z}_B , \bar{z}_W and \bar{z}_{WB} are the mean values of vectors \mathbf{z}_B , \mathbf{z}_W and \mathbf{z}_{WB} respectively, $(z_W)_i$ and $(z_B)_i$ represent the i^{th} element of \mathbf{z}_W and \mathbf{z}_B . The idea behind TEV is that if \mathbf{k}_W and \mathbf{k}_B have the same variance, then the data set should be difficult to separate. The high value of TEV rejects the hypothesis of equal variance and indicates compact within-class and mutually distant between-class distribution [15]. Here we also normalize TEV by multiplying $1/n$.

Like TEM, TEV also needs $O(n^2)$ time complexity, but TEV needs extra $O(n^2)$ to search the medians.

3.2 Theoretical Analysis

We theoretically investigate the relationship between metrics KA, KS and generalization errors.

Theorem 1. KA is defined as (1). Let $R(h) = \text{Pr}[yh < 0]$ be the error rate of Parzen window predictor

$$h(\mathbf{x}') = \frac{E_{\mathbf{x}}[y k_c(\mathbf{x}, \mathbf{x}')]}{\sqrt{E[k_c^2]}} \quad (10)$$

in binary classification. k_c is the centered kernel function and $E[\cdot]$ is an expectation operator. Suppose that $k(\mathbf{x}, \mathbf{x}) \leq S^2$ for all \mathbf{x} . Then for any $\delta > 0$, the following inequality holds with probability at least $1 - \delta$:

$$R(h) \leq 1 - \left(\text{KA}(\mathbf{K}_c, \mathbf{y}\mathbf{y}^\top) - 18\beta \left[\frac{3}{n} + 4\sqrt{\frac{\log \frac{6}{\delta}}{2n}} \right] \right) \cdot \frac{1}{\Gamma}, \quad (11)$$

where $\Gamma = \max_{\mathbf{x}'} \sqrt{\frac{E_{\mathbf{x}}[k_c^2(\mathbf{x}', \mathbf{x})]}{E_{\mathbf{x}, \mathbf{x}'}[k_c^2(\mathbf{x}', \mathbf{x})]}}$, $\beta = \max\left(\frac{S^2}{E[k_c^2]}, \frac{S^2}{E[k_c'^2]}\right)$ and $k'(\mathbf{x}_i, \mathbf{x}_j) = y_i y_j$.

Proof. According to Theorem 12 in [13], we have

$$\text{KA}(k_c, k_c') \geq \text{KA}(\mathbf{K}_c, \mathbf{y}\mathbf{y}^\top) - 18\beta \left[\frac{3}{n} + 4\sqrt{\frac{\log \frac{6}{\delta}}{2n}} \right],$$

where $\text{KA}(k_c, k_c') = \frac{E[k_c k_c']}{\sqrt{E[k_c^2]E[k_c'^2]}}$. Unifying Theorem 13 in [13]

$$R(h) \leq 1 - \text{KA}(k_c, k_c') \cdot \frac{1}{\Gamma},$$

We obtain the inequation (11) directly.

Theorem 2. [14] *KS is defined as (3). There is a separating hyperplane*

$$h(\mathbf{x}) = \mathbf{e} \cdot \phi(\mathbf{x}) - \mathbf{e} \cdot \frac{std_- \phi_+ + std_+ \phi_-}{std_- + std_+}, \quad (12)$$

such that the upper bound of training error of data set \mathcal{D} is

$$KSerr = \frac{KS(\mathbf{K}, \mathbf{y})^2}{1 + KS(\mathbf{K}, \mathbf{y})^2}. \quad (13)$$

Theorem 1 tells us that if there is a high KA and Γ is not too large, then the upper bound of generalization error of (10) on \mathcal{D} is small. Theorem 2 indicates if KS is small, then the upper bound of training error of (12) on \mathcal{D} is small, thus we can expect a low generalization error [14].

3.3 Data Set Feature

Based on the above analysis, we define data set feature as follows:

$$\mathbf{v} = [KA, 1 - KSerr, 1 - ROR, TEM, TEV]. \quad (14)$$

The computation of \mathbf{v} has a time complexity of $O(n^2)$. KA, KS and ROR mainly focus on the distributions and the degree of overlap of two classes from a geometrical point of view, while statistical tests (TEM, TEV) are used to characterize the nonlinearity of decision boundary. Employing different kernel functions would produce different features. We adopt Euclidean distance as similarity criterion:

$$\rho(\mathcal{D}, \mathcal{D}') := \|\mathbf{v} - \mathbf{v}'\|_2 = \sqrt{\sum_{i=1}^5 (v_i - v'_i)^2}. \quad (15)$$

The smaller $\rho(\mathcal{D}, \mathcal{D}')$ means that the similarity between data sets \mathcal{D} and \mathcal{D}' is higher.

4 Classifier Selection

Suppose that historical data sets $\mathcal{D}_1, \dots, \mathcal{D}_m$ and testing data set \mathcal{D} are two-class problems. Our CS algorithm is shown in Algorithm 1.

4.1 Recommendation Algorithm

In step 2 of Algorithm 1, we use KNN classifier as \mathcal{A}_R , where the data set similarity is the distance between data set features. Assuming $\mathcal{D}_j, j = 1, 2, \dots, K$ are the K most similar data sets for \mathcal{D} , the recommended classifier is selected as: (i) for each \mathcal{D}_j , we assign a rank to candidate classifiers according to its performances on this problem. The classifier with the best performance has rank 1, while the classifier with the worst performance has rank m . Classifiers with

Algorithm 1. CS for Two-class Problems

Input: historical data sets $\mathcal{D}_1, \dots, \mathcal{D}_m$, candidate classifiers $\mathcal{A}_1, \dots, \mathcal{A}_\ell$, testing data set \mathcal{D}

Output: classifier \mathcal{A}^*

- 1: Evaluate the performances of candidate classifiers on historical data sets using 10-fold cross validation.
 - 2: Design a recommendation algorithm \mathcal{A}_R based on similarity and the performances.
 - 3: Extract the data set features $\mathbf{v}_1, \dots, \mathbf{v}_m$ and \mathbf{v} as (14).
 - 4: Compute the data set similarities using (15).
 - 5: Output a best classifier \mathcal{A}^* for \mathcal{D} using \mathcal{A}_R .
-

the same performance have the same average rank; (ii) let $R_{i,j}, i = 1, 2, \dots, \ell$ denote the rank of classifier \mathcal{A}_i on \mathcal{D}_j , then the rank of classification algorithm \mathcal{A}_i on \mathcal{D} is computed as

$$R_{i,\mathcal{D}} = \frac{1}{K} \sum_{\mathcal{D}_j \in N_c(\mathcal{D})} R_{i,j}, j = 1, 2, \dots, K, \quad (16)$$

where $N_c(\mathcal{D})$ is a set contains the K most similar data sets of \mathcal{D} . In the end, the classifier with the lowest rank is the recommended classifier.

4.2 Multi-class Classification Problem

Our feature only suitable for two-class data sets. We handle multi-class problems as follow.

Step 1: Suppose that data set \mathcal{D} has c classes. We split \mathcal{D} into $m = \frac{c(c-1)}{2}$ two-class problems using one-vs-one strategy.

Step 2: For each sub-problem, we recommend one classifier based on Algorithm 1.

Step 3: The final decision is determined by using voting strategy.

The merit of this method is that we can select the most suitable classifier for each sub-problem, which would make the classification accuracy higher than that of the single classifier.

5 Experiments

We evaluate the proposed feature with three state-of-the-art features with respect to computational efficiency and recommendation performance.

5.1 Experimental Setup

Data Sets. We selected 67 classification problems from the UCI repository which include 49 historical data sets and 18 testing data sets (Table 1). Among

Table 1. Summary of testing data sets in terms of attributes, sample size and classes.

ID	Name	Att.	Ins.	Classes	ID	Name	Att.	Ins.	Classes
1	abalone	8	4117	3	10	page-blocks	10	5473	5
2	car	6	1728	4	11	seeds	7	210	3
3	contrac	9	1473	3	12	segment	18	2310	7
4	dermatology	34	366	6	13	st-landsat	36	6534	6
5	hayes-roth	5	132	3	14	st-vehicle	18	846	4
6	hill-valley	100	1212	2	15	synthetic-control	60	600	6
7	hill-valley-noise	100	1212	2	16	teaching	5	151	3
8	iris	4	154	3	17	waveform	21	2000	3
9	nursery	8	12598	4	18	wine	13	178	3

the historical data sets, the multi-class data sets are split into two-class data sets using one-vs-one technique, then those data sets that are easy to classify or have severely unbalanced/small samples in each class are deleted. We totally have 84 two-class historical data sets. The attributes of data sets are normalized into $[-1, 1]$.

Candidate Classifiers. We employ 20 candidate classifiers. Some candidate classifiers are KNN, LDA, logistics regression, SVM (linear, polynomial kernel, RBF kernel), naive bayes, decision tree C4.5, random forest, Bagging (tree) and AdaBoost (tree). These classifiers are run with the MATLAB statistic toolbox except SVM uses LIBSVM software.

The remaining classifiers are nearest mean classifier, Fisher’s least square linear discriminant, BP neural network, linear perceptron, Bayesian classifier, Gaussians mixture model, Parzen classifier, Parzen density classifier and radial basis neural network classifier, which are adopted from PrTools toolbox 5.0. We run all codes on MATLAB 2017a on Windows operating system with Inter(R) Core(TM) i5-6500 CPU @3.20GHz processor.

Comparative Classifiers. We evaluate 24 classifiers on testing data sets which include 20 candidate classifiers and 4 data set features.

- statistical feature (F_s) [7];
- problem complexity feature (F_p) [4];
- landmarking feature (F_l) [6] with landmarks KNN, C4.5, LR and NB;
- our data set feature using polynomial kernel (F_{poly}). We set $d = 3$.

The attributes of 4 data set features are normalized into $[0, 1]$. F_s , F_p and F_l adopt the CS framework in Algorithm 1. For each testing data set, 10% samples of each class are dropped as testing samples and the rests are used for training (the testing data set in Algorithm 1). The classification model of recommended classifier on training samples are trained using 10-fold cross validation. For the

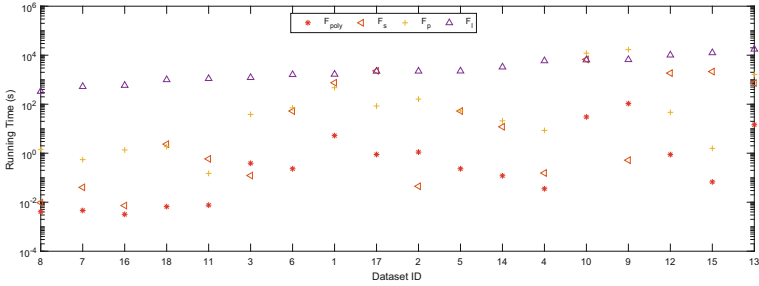


Fig. 1. Running times (s) of F_{poly} , F_s , F_p and F_l on testing data sets. The total times are 160.11s, 14662.85s, 31602.10s and 77109.74s, respectively.

sake of fairness, we also evaluate the performance of candidate classifiers on multi-class testing data sets using splitting and voting strategy.

Performance Metrics. We employ classification accuracy (CA), average recommendation performance ratio (ARPR) [8] and non-parameter statistical tests [16] to evaluate the performance of data set features.

5.2 Computational Efficiency

We collected the computation times of 4 data set features on 18 testing data sets (Fig. 1). The recorded time of each data set is the sum of times of its sub-problems. From Fig. 1, we see that our feature has the fastest computational speed, which spent 160 seconds on overall data sets. However, F_s , F_p and F_l have unacceptable low speeds. Although F_s outperformed our features on data sets 2, 3, and 9, we found that these data sets have discrete variables. For continuous variables, the efficiency of F_s would be degraded rapidly. Therefore, our feature outperforms F_s , F_p and F_l in terms of efficiency.

5.3 Performance Comparisons

In this section, we compare our F_{poly} with three state-of-the-art data set features: F_s , F_p and F_l , as well as 20 candidate classifiers. The comparisons of CA, ARPR and statistical test are listed in Table 2. We observe that F_{poly} has the highest CA and ARPR.

To check the statistical difference between different methods, we calculated the average rank of each feature and shown it in the last row of Table 2. F_{poly} has the lowest average rank 1.36, followed by F_s . F_p has the worst average rank. The Friedman statistic is distributed according to the F-distribution with $(4-1) = 3$ and $(4-1) \times (18-1) = 51$ degrees of freedom. The value of Friedman statistic is 11.64 and the critical value of $F(3, 51)$ is 2.79 at 0.05 significance level. Thus, the null hypothesis is rejected. Then we applied the Nemenyi test for pairwise

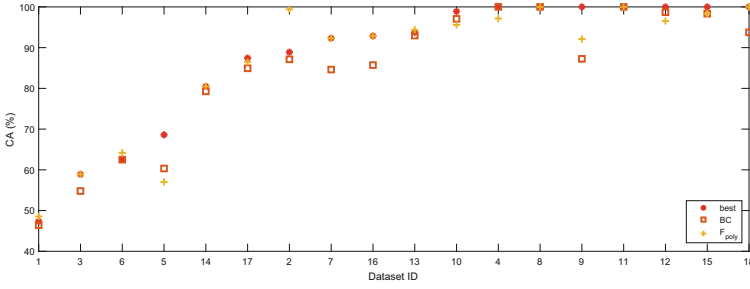


Fig. 2. CA (%) of *best*, BC and F_{poly} . *best* represents the CA of the best candidate classifier.

comparisons. The critical different is 1.11 which means that F_{poly} is significantly better than F_p and F_l .

Finally, we compare the CA of F_{poly} with that of the best candidate classifier and Bayesian classifier (BC) which has the highest ACA among 20 classifiers, shown in Fig. 2. We see that the CA of F_{poly} are very close to the CA of the best candidate classifier except on data sets 7 and 18. F_{poly} is equal to or higher than the best candidate classifier on 11 data sets. F_{poly} has the same CA as or outperforms BC in 14 out of 18 cases. On the 4 data sets that BC outperforms F_{poly} , we see that the CA of BC and F_{poly} are very close.

6 Conclusion

The difficulties of CS mainly stem from the similarity measurement among data sets. So far, people resolve this problem by characterizing data set feature and turn to comparing the similarity of features. In this paper, we proposed a new data set feature to describe the classification complexity of data set. Different

Table 2. CA (%) of F_{poly} , F_s , F_p and F_l on testing data sets. The first column shows the ID of data sets. The last row reports the average rank of each CS algorithm. \mathcal{A}_{best} and \mathcal{A}_{worst} indicate the best and worst CA of candidate classifiers.

ID	F_{poly}	F_s	F_p	F_l	\mathcal{A}_{best}	\mathcal{A}_{worst}	ID	F_{poly}	F_s	F_p	F_l	\mathcal{A}_{best}	\mathcal{A}_{worst}
1	48.56	46.63	41.59	43.99	47.36	44.47	12	96.54	95.67	96.10	95.67	100	98.27
2	99.42	74.85	76.02	91.81	88.89	77.19	13	94.38	93.59	91.72	90.63	93.75	90.47
3	58.90	58.22	50.00	54.79	58.90	53.42	14	80.49	74.39	75.61	74.39	80.49	74.39
4	97.14	94.29	88.57	94.29	100	100	15	98.33	98.33	98.33	95.00	100	98.33
5	57.02	57.02	52.07	52.07	68.60	52.89	16	92.86	78.57	85.71	71.43	92.86	78.57
6	64.17	64.17	50.83	55.83	66.94	48.33	17	86.43	81.41	83.92	84.42	87.44	78.89
7	92.31	92.31	76.92	84.62	92.31	69.23	18	100	100	93.75	100	100	93.75
8	100	100	100	100	100	100							
9	92.12	80.22	78.36	79.29	100	86.01	ACA	86.35	82.44	79.54	81.23		
10	95.59	94.30	92.28	93.93	98.90	95.96	ARPR	0.99	0.95	0.90	0.93		
11	100	100	100	100	100	85.71	Rank	1.36	2.44	3.19	3.00		

from previous works, our feature has merits like low computational complexity and theoretical support. We built a CS framework using the proposed feature. Experimental results show that our feature is effective and efficient. Our method outperforms three data set features, which means that the proposed feature can help to choose suitable classifiers for new classification problems.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Grant 61602308.

References

1. Wolpert, D.H.: The lack of a priori distinction between learning algorithms. *Neural Comput.* **8**(7), 1341–1390 (1996)
2. Maciá, N., Bernadó-Mansilla, E., Orriols-Puig, A., Kam, H.T.: Learner excellence biased by data set selection: a case for data characterisation and artificial data sets. *Pattern Recogn.* **46**(3), 1054–1066 (2013)
3. Cernadas, E., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? *J. Mach. Learn. Res.* **15**(1), 3133–3181 (2014)
4. Bernadó-Mansilla, E., Ho, T.K.: Domain of competence of XCS classifier system in complexity measurement space. *IEEE Trans. Evol. Comput.* **9**(1), 82–104 (2008)
5. Peng, Y., Flach, P.A., Soares, C., Brazdil, P.: Improved dataset characterisation for meta-learning. In: Lange, S., Satoh, K., Smith, C.H. (eds.) *DS 2002. LNCS*, vol. 2534, pp. 141–152. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36182-0_14
6. Pfahringer, B., Bensusan, H., Giraud-Carrier, C.G.: Meta-learning by landmarking various learning algorithms. In: *Seventeenth International Conference on Machine Learning*, vol. 11, no. 9, pp. 743–750. Morgan Kaufmann Publishers Inc. (2000)
7. Song, Q., Wang, G., Wang, C.: Automatic recommendation of classification algorithms based on data set characteristics. *Pattern Recogn.* **45**(7), 2672–2689 (2012)
8. Wang, G., Song, Q., Zhu, X.: An improved data characterization method and its application in classification algorithm recommendation. *Appl. Intell.* **43**(4), 892–912 (2015)
9. Kotthoff, L.: Algorithm selection for combinatorial search problems: a survey. In: Bessiere, C., De Raedt, L., Kotthoff, L., Nijssen, S., O’Sullivan, B., Pedreschi, D. (eds.) *Data Mining and Constraint Programming. LNCS (LNAI)*, vol. 10101, pp. 149–190. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50137-6_7
10. Kalousis, A., Theoharis, T.: NOEMON: design, implementation and performance results of an intelligent assistant for classifier selection. *Intell. Data Anal.* **3**(5), 319–337 (1999)
11. Ho, T.K., Basu, M.: Complexity measures of supervised classification problems. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(3), 289–300 (2002)
12. Cano, J.R.: Analysis of data complexity measures for classification. *Expert Syst. Appl.* **40**(12), 4820–4831 (2013)
13. Cortes, C., Mohri, M., Rostamizadeh, A.: Algorithms for learning kernels based on centered alignment. *J. Mach. Learn. Res.* **13**(2), 795–828 (2012)
14. Nguyen, C.H., Tu, B.H.: An efficient kernel matrix evaluation measure. *Pattern Recogn.* **41**(11), 3366–3372 (2008)
15. Chudzian, P.: Evaluation measures for kernel optimization. *Pattern Recogn. Lett.* **33**(9), 1108–1116 (2012)
16. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**(1), 1–30 (2006)