A Geometric Framework for Multiclass Ensemble Classifiers

Wu, S., Li, J., & Ding, W. (2023). A Geometric Framework for Multiclass Ensemble Classifiers. *Machine Learning*, 1-30. Advance online publication. https://doi.org/10.1007/s10994-023-06406-w

Link to publication record in Ulster University Research Portal

Published in:
Machine Learning

Publication Status:
Published online: 27/09/2023

DOI:
https://doi.org/10.1007/s10994-023-06406-w

Document Version
Publisher's PDF, also known as Version of record

General rights
Copyright for the publications made accessible via Ulster University's Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Ulster University's institutional repository that provides access to Ulster's research outputs. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact pure-support@ulster.ac.uk.

Download date: 20/10/2023

Machine Learning
https://doi.org/10.1007/s10994-023-06406-w

A geometric framework for multiclass ensemble classifiers

Shengli Wu[1,2] · **Jinlong Li**[1] · **Weimin Ding**[1]

Received: 9 November 2022 / Revised: 5 August 2023 / Accepted: 22 August 2023
© The Author(s) 2023

Abstract
Ensemble classifiers have been investigated by many in the artificial intelligence and machine learning community. Majority voting and weighted majority voting are two commonly used combination schemes in ensemble learning. However, understanding of them is incomplete at best, with some properties even misunderstood. In this paper, we present a group of properties of these two schemes formally under a geometric framework. Two key factors, every component base classifier's performance and dissimilarity between each pair of component classifiers are evaluated by the same metric—the Euclidean distance. Consequently, ensembling becomes a deterministic problem and the performance of an ensemble can be calculated directly by a formula. We prove several theorems of interest and explain their implications for ensembles. In particular, we compare and contrast the effect of the number of component classifiers on these two types of ensemble schemes. Some important properties of both combination schemes are discussed. And a method to calculate the optimal weights for the weighted majority voting is presented. Empirical investigation is conducted to verify the theoretical results. We believe that the results from this paper are very useful for us to understand the fundamental properties of these two combination schemes and the principles of ensemble classifiers in general. The results are also helpful for us to investigate some issues in ensemble classifiers, such as ensemble performance prediction, diversity, ensemble pruning, and others.

Keywords Ensemble learning · Geometric framework · Classification · Majority voting · Weighted majority voting

Editor: Zhi-Hua Zhou.

✉ Shengli Wu
swu@ujs.edu.cn

Jinlong Li
ljl3120@163.com

Weimin Ding
25581071@qq.com

[1] School of Computer Science, Jiangsu University, Zhenjiang 212013, China

[2] School of Computing, Ulster University, Belfast BT15 1AP, UK

Published online: 27 September 2023

1 Introduction

In the last three decades, ensemble learning has been investigated by many researchers. This technique has seen used in diverse tasks such as classification, regression, and clustering among others. Research has been conducted at multiple levels: from feature selection, component classifier generation and selection, to the ensemble model (Jurek et al., 2014; Dong et al., 2020). Some of the ensemble approaches have been very successful in international machine learning competitions such as Kaggle, KDD-Cups, etc. and the technologies have been extensively used in various application areas (Oza & Tumer, 2008).

Although some ensemble models (such as stacking, bagging, random forest, AdaBoost, gradient boosting machines, deep neural network-based models, and others) are more complicated, two relatively simple combination schemes, majority voting and weighted majority voting, have been used widely for the ensemble model (Sagi & Rokach, 2018). Even in those more complicated models, majority voting and weighted majority voting are still used frequently at intermediate or final combination stages.

How many component classifiers to use and how to select a subset from a large group for an ensemble are two related questions. Zhou et al. (2002) investigated the impact of the number of component classifiers on ensemble performance. It is found that the number of component classifiers is not always a positive factor for improving performance when majority voting is used for combination. Later their finding is often referred to as the "many-could-be-better-than-all" theorem.

However, this finding is not echoed by many others. More empirical evidence indicates that the size of an ensemble has a positive impact on performance (Hernández-Lobato et al., 2013). To balance ensemble performance and efficiency, many papers investigate how to achieve best possible performance by combining a fixed or a small number of classifiers selected from a large number of candidates (Latinne et al., 2001; Oshiro et al., 2012; Xiao et al., 2010; Dias & Windeatt, 2014; Bhardwaj et al., 2016; Ykhlef & Bouchaffra, 2017; Zhu et al., 2019).

Possibly inspired by Zhou et al' work (Zhou et al., 2002), Bonab and Can (2019) asserted that for weighted majority voting: the ideal condition for the ensemble to achieve maximum accuracy is for the number of component classifiers to equal the number of output classes. However, their theoretical analysis is limited by the strength of assumptions used, which experimental results do not always support.

Diversity among component classifiers is a factor that may influence ensemble performance (Bi, 2012; Jain et al., 2020; Zhang et al., 2020). However, there is no generally accepted definition of diversity (Kuncheva & Whitaker, 2003). Many measures, such as Yule's Q statistics, the correlation coefficient, the disagreement measure, F-score, the double fault measure, Kohavi-Wolpert's variance, Kuncheva's entropy, etc., have been proposed and investigated (Tang et al., 2006; Visentini et al., 2016). Moreover, many diversity measures such as the F-score, the disagreement measure, the double fault measure, etc. are or related to performance measures. This explains why the effect of diversity on ensemble performance is unclear in some cases (Tang et al., 2006; Bi, 2012).

In short, although ensemble learning has garnered considerable attention from researchers, much of the literature comprise methodological and empirical studies, while the theory is underrepresented. Many fundamental questions remain unclear. We list some of these below:

1. What is the difference between majority voting and weighted majority voting?
2. When should we use majority voting rather than weighted majority voting, or vice versa?
3. There are a lot of weighting assignment methods for weighted majority voting, which one is the best?
4. How does the number of component classifiers affect ensemble performance?
5. How does each component classifier affect ensemble performance?
6. How does performance of component classifiers affect ensemble performance?
7. How does diversity of component classifiers affect ensemble performance?

The goal of this paper is to set up a geometric framework for ensemble classifiers, thereby enabling us to clearly answer the above questions. In this framework, the result from each base classifier is represented as a point in a multi-dimensional space. We can use the same measure—the Euclidean distance—for both performance and diversity. Ensemble learning becomes a deterministic problem. Many interesting theorems about the relation of component classifiers and ensemble classifiers can be proven. Armed with this framework, we discover and present some useful features of majority voting and weighted majority voting. The major contributions of this paper is as follows:

1. A geometric framework is presented for ensemble classifiers.
2. An optimal weighting scheme is given for weighted majority voting.
3. A Euclidean distance-based measure of diversity is given. Unlike all other diversity measures proposed so far, it is orthogonal to performance.
4. Experiments on twenty data sets validate the theory for practical use.

The rest of this paper is organized as follows: Sect. 2 presents some related work. In Sect. 3, we present the geometric framework for ensemble classifiers. Some characteristics of majority voting and weighted majority voting are presented and compared. Discussion regarding the questions raised previously is given in Sect. 4, supported by theory. In Sect. 5, we present some empirical investigation results to confirm our findings in Sect. 3. Finally Sect. 6 is the conclusion.

2 Related work

Majority voting and weighted majority voting are commonly used in many ensemble models. Their features and some related tasks have been investigated by many researchers.

Zhou et al. (2002) investigated majority voting theoretically and empirically through an ensemble of a group of neural networks. It is found that selecting a subset may be able to achieve better performance than combining all available classifiers. Note that in both their theoretical and empirical study, weighted majority voting is mentioned, but not considered.

Fumera et al. (2008) set up a probabilistic framework to analyse the bagging misclassification rate when the ensemble size increases. Majority voting is used for combination. Both their theoretical analysis and empirical investigation show that bagging misclassification rate decreases when more and more component classifiers are combined. This is somewhat inconsistent with the finding of Zhou et al. (2002).

Latinne et al. (2001) used the McNemar test to decide if two sets of component classifiers are significantly different. More and more classifiers are added to the pool and the McNemar test is carried out repeatedly, the process stops if no significant difference can be observed.

Oshiro et al. (2012) raised the problem of how many component classifiers (trees) should be used for an ensemble (random forest). Experimenting with 29 real data sets in the biomedical domain, they observed that higher accuracy is achievable when more trees are combined by majority voting. However, when the number of trees is relatively large (say, over 32 or 64), the improvement is no longer significant. Therefore, a good choice is to consider both ensemble accuracy and computing cost. This issue is also addressed in Hernández-Lobato et al. (2013); Adnan and Islam (2016); Probst and Boulesteix (2017).

Many researchers find that if more base classifiers are combined, then the ensemble is able to achieve better performance. Although it is possible to achieve better performance by fusing more base classifiers, it costs more. Quite a few papers investigated the ensemble pruning problem that aims at increasing efficiency by reducing the number of base classifiers, without losing much performance at the same time. Various kinds of methods have been proposed. See Xiao et al. (2010); Dias and Windeatt (2014); Bhardwaj et al. (2016); Ykhlef and Bouchaffra (2017); Zhu et al. (2019); Mohammed et al. (2022) for some of them.

How to assign proper weights for weighted majority vote has been investigated by quite a few researchers. Some methods may consider different aspects of base classifiers for the weights: performance-based in Opitz and Shavlik (1996); Wozniak (2008), Bayesian model-based in Duan et al. (2007), prediction confidence-based in Schapire and Singer (1999), and Matthews correlation coefficient-based in Haque et al. (2016). Some methods may have different goals for weight optimisation: minimizing classification error in Kuncheva and Diez (2014); Mao et al. (2015); Bashir et al. (2015), reducing variance while preserving given accuracy in Derbeko et al. (2002), minimizing Euclidean distance in Bonab and Can (2018). Different search methods are also used: A linear programming-based method is presented in Zhang and Zhou (2011), Georgiou et al. (2006) applied a game theory to weight assignment, Liu et al. (2014) applied the genetic algorithm to weight assignment. Dynamic adjustment of weights is investigated in Valdovinos and Sánchez (2009).

Wu and Crestani (2015) proposed a geometric framework for data fusion in information retrieval. In this query-level framework, for a certain query each component retrieval system assigns scores to all the documents in the collection, which indicates their relevance to the query. Also predefined values are given to documents that are relevant or irrelevant to the query, which are ideal scores for the documents involved. Therefore, the scores from each component retrieval system can be regarded as a point in a multiple dimensional space, and the ideal scores form an ideal point. Under this framework, performance and dissimilarity can be measured by the same metric—the Euclidean distance. Both majority voting and weighted majority voting can be explained and calculated in the geometric space. However, the framework is set up at the query level, which is equivalent to the instance level in ensemble classifiers.

Bonab and Can (2018, 2019) adapted the model in Wu and Crestani (2015) to ensemble classifiers. Some properties of majority voting and weighted majority voting are presented. However, as in Wu and Crestani (2015), their framework is at the instance level. This

means that the theorems hold for each individual instance, but it is unclear if they remain true for multiple instances collectively. The latter is a more important and realistic situation we should consider. As we know, a training data set or a test data set usually comprises a group of instances. It is desirable to know the collective properties of an ensemble classifier over all the instances, rather than that of any individual instance. This generalization is the major goal of this paper.

To do this, we first define a dataset-level framework and then go about proving a number of useful theorems.

3 The geometric framework

In this section, we introduce the dataset-level geometric framework.

Suppose for a machine learning problem we have p classes $CL=\{cl_1,cl_2,\cdots,cl_p\}$, the ensemble has m component base classifiers $CF=\{cf_1,cf_2,\cdots,cf_m\}$, and the dataset DT has n instances $T=\{t_1,t_2,\cdots,t_n\}$. For every instance t_i and every class cl_j, each base classifier cf_k provides a score s_{ij}^k, which indicates the estimated probability score that t_i is an instance of class cl_k given by cf_j. $s_{ij}^k \in [0, 1]$. Each instance t_i has a real label for each class cl_k, which is 0 or 1. We may set up a $n * p$-dimensional space for the above problem. There are m points $\{S^1, S^2, \cdots, S^m\}$, each represents the scores given by a specific classifier to all the instances in the dataset for all the classes. Point S^k is:

$$\begin{aligned} S^k &= \{(s_{11}^k, s_{12}^k, \cdots, s_{1p}^k), (s_{21}^k, s_{22}^k, \cdots, s_{2p}^k), \cdots (s_{n1}^k, s_{n2}^k, \cdots, s_{np}^k)\} \\ &= \{s_1^k, s_2^k, \cdots, s_{n*p}^k\} \end{aligned}$$

As such, we can always organize all the scores first by instances and then by classes. Thus a two-dimensional array with n elements in one dimension and p in the other is transformed to a list of $n * p$ elements: s_{ij}^k becomes $s_{(i-1)*n+j}^k$ The ideal point is also represented in the same style.

$$\begin{aligned} O &= \{(o_{11}, o_{12}, \cdots, o_{1p}), (o_{21}, o_{22}, \cdots, o_{2p}), \cdots (o_{n1}, o_{n2}, \cdots, o_{np})\} \\ &= \{o_1, o_2, \cdots, o_{n*p}\} \end{aligned}$$

which indicates the real labels of every instance in the dataset to each of the classes involved: 1 for a true label and 0 for a false label. The notation used in this paper is summarized in Table 1.

This framework is a generalization of the one presented in Bonab and Can (2018, 2019). If the dataset only has one instance, then the above framework is the same as the one in Bonab and Can (2018, 2019). This framework is suitable for soft voting (Cao et al., 2015), in which each component classifier provides probability scores for every instance relating to each class. If no such probability scores are provided (hard voting), then it is still possible to apply it to the geometric framework if we transform estimated class labels into proper scores. The geometric framework is applicable to both single-label and multi-label classification problems.

Table 1 Notation used in this paper

Symbols	Meaning
C	Centroid of a group of points in X
CL	A set of classes
cl_j	One of the classes in CL
CF	A set of component classifiers
cf_k	One of the component classifiers in CF
DT	A dataset has components CF, CL, S, T, and O
$ed(S^i, S^j)$	the Euclidean distance between two points (S^i, S^j) in X
F	Fused point by linear combination of a group of points in X
m	Number of component classifiers in CF
n	Number of instances in T
O	All the real labels for T constitute the ideal point in X
o_{ij}	One of the elements of O
o_l	Another form of o_{ij}, in which $l = (i-1) * n + j$
p	Number of classes in CL
$\mathcal{S}$	Scores given by CF members relating to CL, a set of points in X
S^k	Scores given by cf_k for T relating to CL, a point in X
s_{ij}^k	A score given by cf_k for t_i relating to c_j
s_l^k	Another form of s_{ij}^k, in which $l = (i-1) * n + j$
T	A set of instances
t_i	One of the instances in T
w^k	Weight assigned to component classifier cf_k
X	a $n * p$-dimensional space relating to DT

Example 1 A dataset includes two instances t_1 and t_2, and three classes cl_1, cl_2, and cl_3. t_1 is an instance of cl_2 but not the other two, and t_2 is an instance of both class cl_1 and cl_3 but not cl_2. The scores got from the three base component classifiers cf_1, cf_2, and cf_3 are as follows:

Instance	Classifier	Class cl_1	Class cl_2	Class cl_3
t_1	cf_1	$s_{11}^1 = 0.5$	$s_{12}^1 = 0.6$	$s_{13}^1 = 0.3$
	cf_2	$s_{11}^2 = 0.4$	$s_{12}^2 = 0.7$	$s_{13}^2 = 0.2$
	cf_3	$s_{11}^3 = 0.6$	$s_{12}^3 = 0.8$	$s_{13}^3 = 0.4$
	Real label	$o_{11} = 0$	$o_{12} = 1$	$o_{13} = 0$
t_2	cf_1	$s_{21}^1 = 0.7$	$s_{22}^1 = 0.3$	$s_{23}^1 = 0.9$
	cf_2	$s_{21}^2 = 0.3$	$s_{22}^2 = 0.6$	$s_{23}^2 = 0.7$
	cf_3	$s_{21}^3 = 0.2$	$s_{22}^3 = 0.6$	$s_{23}^3 = 0.8$
	Real label	$o_{21} = 1$	$o_{22} = 0$	$o_{23} = 1$

In a 6-dimensional geometric space, we may set up four points to represent this scenario: $S^1 = \{0.5, 0.6, 0.3, 0.7, 0.3, 0.9\}$, $S^2 = \{0.4, 0.7, 0.2, 0.3, 0.6, 0.7\}$, $S^3 = \{0.6, 0.8, 0.4, 0.2, 0.6, 0.8\}$, and $O = \{0, 1, 0, 1, 0, 1\}$. □

In the following we use point and component result interchangeably if no confusion will be caused. We can calculate the Euclidean distance of two points S^u and S^v

$$ed(S^u, S^v) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{p}(s_{ij}^u - s_{ij}^v)^2} = \sqrt{\sum_{l=1}^{n*p}(s_l^u - s_l^v)^2} \tag{1}$$

We may use the Euclidean distance to evaluate the performance of classifier cf_u over n instances and p classes.

$$ed(S^u, O) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{p}(s_{ij}^u - o_{ij})^2} = \sqrt{\sum_{l=1}^{n*p}(s_l^u - o_l)^2} \tag{2}$$

It is an advantage of the geometric framework to evaluate both performance of a component result and dissimilarity of two component results by using the same metric. They will be referred to as performance distance and dissimilarity distance later in this paper.

Definition 1 (Performance distribution). In a geometric space X, there are m points $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ $(m \geq 1)$. O is the ideal point. Performance distribution of these m points in $\mathcal{S}$ is defined as $ed(S^i, O)$ for $(1 \leq i \leq m)$.

Definition 2 (Dissimilarity distribution). In a geometric space X, there are m points $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ $(m \geq 1)$. Dissimilarity distribution of these m points in $\mathcal{S}$ is $ed(S^i, S^j)$ for $(1 \leq i \leq m, 1 \leq j \leq m, i \neq j)$.

From their definitions, we can see that performance distribution and dissimilarity distribution are two completely different aspects of $\mathcal{S}$, and not related to each other. Their independence is a good thing for us to investigate the properties of ensembles, especially the effect of each on ensemble performance.

3.1 Majority voting

In a $n * p$-dimensional space, there are m points $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ $(m \geq 2)$. Combining them by majority voting can be understood to be finding the centroid of these m points. It is referred to as the centroid-based fusion method in Wu and Crestani (2015).

Theorem 1 *In a geometric space X, there are m points $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ $(m \geq 2)$. C is the centroid of these m points and O is the ideal point. The distance between C and O is no longer than the average distance between each of the m points and O:*

$$\sqrt{\sum_{l=1}^{n*p}(c_l - o_l)^2} \leq \frac{1}{m}\sum_{k=1}^{m}\sqrt{\sum_{l=1}^{n*p}(s_l^k - o_l)^2} \tag{3}$$

Proof Replace C with its definition $c_l = \frac{1}{m}\sum_{k=1}^{m} s_l^k$ for $(1 \le l \le n * p)$ in Eq. (3) and move $\frac{1}{m}$ on the right side to the left as m, we get

$$m\sqrt{\sum_{l=1}^{n*p}(\frac{1}{m}\sum_{k=1}^{m} s_l^k - o_l)^2} \le \sum_{k=1}^{m}\sqrt{\sum_{l=1}^{n*p}(s_l^k - o_l)^2} \tag{4}$$

The Minkowski Sum Inequality (Minkowski, 2020) is

$$(\sum_{l=1}^{n*p}(a_l + b_l)^q)^{q^{-1}} \le \sum_{l=1}^{n*p} a_l^q)^{q^{-1}} + \sum_{l=1}^{n*p} b_l^q)^{q^{-1}}$$

where $q > 1$ and $a_l, b_l > 0$. For our question, we let $q = 2$ and $a_l = s_l^k - o_l$. Then we have:

$$\begin{aligned}\sum_{k=1}^{m}\sqrt{\sum_{l=1}^{n*p}(s_l^k - o_l)^2} &= \sum_{k=1}^{m}\sqrt{\sum_{l=1}^{n*p}(a_l)^2} \\ &\ge \sqrt{\sum_{l=1}^{n*p}(a_1 + a_2)^2} + \sum_{k=3}^{m}\sqrt{\sum_{l=1}^{n*p}(a_l)^2} \\ &\ge \ldots \ge \sqrt{\sum_{l=1}^{n*p}(a_1 + a_2 + \ldots + a_m)^2} = \sqrt{\sum_{l=1}^{n*p}(\sum_{k=1}^{m} a_k)^2}\end{aligned}$$

Now notice the left side of Inequation 4

$$m\sqrt{\sum_{l=1}^{n*p}(\frac{1}{m}\sum_{k=1}^{m} s_l^k - o_l)^2} = \sqrt{\sum_{l=1}^{n*p}(\sum_{k=1}^{m} s_l^k - m * o_l)^2} = \sqrt{\sum_{l=1}^{n*p}(\sum_{k=1}^{m} a_k)^2}$$

□

This theorem tells us, if we take all the instances together and use the Euclidean distance as the performance metric, then the performance of the ensemble by majority voting is at least as good as the average performance of all component classifiers involved. This theorem confirms the observation by many researchers that the results from the ensemble are stable and good.

Example 2 Consider the points in Example 1. $C = \{0.5, 0.7, 0.3, 0.4, 0.5, 0.8\}$, $ed(S^1, O) = 0.83$, $ed(S^2, O) = 1.11$, $ed(S^3, O) = 1.26$, $ed(C, O) = 1.04$, $(ed(S^1, O) + ed(S^2, O) + ed(S^3, O))/3 = 1.06$. $ed(C, O)$ is slightly smaller than the average of $ed(S^1, O)$, $ed(S^2, O)$, and $ed(S^3, O)$. □

Theorem 2 *In a space X, suppose that* $\mathcal{S}=\{S^1, S^2, \cdots, S^m\}$ *and O are known points. C is the centroid of* $S^1, S^2, \cdots, S^m$. *The distance between C and O can be represented as*

$$ed(C, O) = \frac{1}{m}\sqrt{m\sum_{i=1}^{m} ed(S^i, O)^2 - \sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(S^i, S^j)^2} \tag{5}$$

Proof Assume that O=(0,...,0), which can always be done by coordinate transformation. According to its definition $C = (\frac{1}{m}\sum_{i=1}^{m} s_1^i, ..., \frac{1}{m}\sum_{i=1}^{m} s_{n*p}^i)$, we have

$$\begin{aligned} ed(C,O) &= \sqrt{(\frac{1}{m}\sum_{i=1}^{m} s_1^i)^2 + ... + (\frac{1}{m}\sum_{i=1}^{m} s_{n*p}^i)^2} \\ &= \frac{1}{m}\sqrt{\sum_{i=1}^{m}\sum_{k=1}^{n*p}(s_k^i)^2 + 2\sum_{k=1}^{n*p}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} s_k^i * s_k^j} \end{aligned} \tag{6}$$

Note that the distance between S^i and S^j is $ed(S^i, S^j) = \sqrt{\sum_{k=1}^{n*p}(s_k^i - s_k^j)^2}$ or $ed(S^i, S^j)^2 = \sum_{k=1}^{n*p}(s_k^i)^2 + \sum_{k=1}^{n*p}(s_k^j)^2 - 2\sum_{k=1}^{n*p} s_k^i * s_k^j$

Because $ed(S^i, O)^2 = \sum_{k=1}^{n*p}(s_k^i)^2$ and $ed(S^j, O)^2 = \sum_{k=1}^{n*p}(s_k^j)^2$, we get

$$2\sum_{k=1}^{n*p} s_k^i * s_k^j = ed(S^i, O)^2 + ed(S^j, O)^2 - ed(S^i, S^j)^2$$

Considering all possible pairs of points S^i and S^j and we get

$$2\sum_{k=1}^{n*p}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} s_k^i * s_k^j = (m-1)\sum_{i=1}^{m} ed(S^i, O)^2 - \sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(S^i, S^j)^2$$

In Eq. (6), we use the right side of the above equation to replace

$$2\sum_{k=1}^{n*p}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} s_k^i * s_k^j$$

Also note that $\sum_{i=1}^{m}\sum_{k=1}^{n*p}(s_k^i)^2 = \sum_{i=1}^{m} ed(S^i, O)^2$, we obtain

$$ed(C,O) = \frac{1}{m}\sqrt{m\sum_{i=1}^{m} ed(S^i, O)^2 - \sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(S^i, S^j)^2}$$

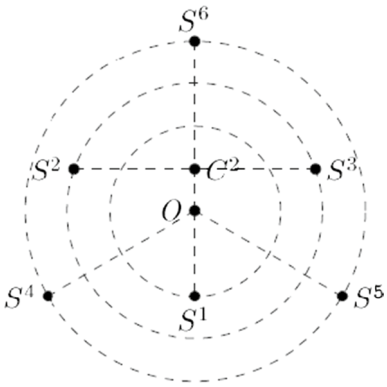
□

This theorem tells us that the ensemble performance is completely decided by $ed(S^i, O)$ (for $1 \le i \le m$) and $ed(S^i, S^j)$ (for $1 \le i \le m$, $1 \le j \le m$, $i \ne j$). The impact of both performance of component classifiers and dissimilarity of all pairs of component classifiers on ensemble performance can be seen clearly. According to Theorem 2, in order to minimize $ed(C, O)$, we need to minimize $\sum_{i=1}^{m} ed(S^i, O)^2$ and maximize $\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(S^i, S^j)^2$ at the same time. Therefore, for performance distribution, both average and variance affect ensemble performance. Lower average and lower variance lead to better performance. For dissimilarity distribution, both average and variance also affect ensemble performance. Higher average and higher variance lead to better performance.

Example 3 Assume that $\mathcal{S}^1$={S^1, S^2}, $\mathcal{S}^2$={S^3, S^4}, $ed(S^1, S^2) = ed(S^3, S^4)$, $ed(S^1, O)$=0.5, $ed(S^2, O)$=0.5, $ed(S^3, O)$=0.4, $ed(S^4, O)$=0.6. The centroid of S^1 and S^2 is C^1, and the centroid of S^3 and S^4 is C^2. We have $cd(C^1, O) < cd(C^2, O)$.

In this example, because dissimilarity distance is the same for both $\mathcal{S}^1$ and $\mathcal{S}^2$, we only need to consider performance distribution. The average of $ed(S^1, O)$ and $ed(S^2, O)$ is 0.5, and the average of $ed(S^3, O)$ and $ed(S^4, O)$ is also 0.5. The variance of $\mathcal{S}^1$ is smaller than that of $\mathcal{S}^2$. $ed(S^1, O)^2 + ed(S^2, O)^2$=0.50, $ed(S^3, O)^2 + ed(S^4, O)^2$=0.52. Because $ed(S^1, S^2) = ed(S^3, S^4)$, we get $ed(C^1, O) < ed(C^2, O)$. □

Example 4 In the figure below, there are six points S^i $(1 \le i \le 6)$. Among these points, S^1 is the closest to O. It is followed by S^2 and S^3, which are equally distant to O. Finally, S^4, S^5 and S^6 are equally distant to O and they are all further from O than the other three points. Now we try to work out a subset of these to maximize performance.



Now, if we select one point only, then S^1 is the best option; if we select two points, then combining S^2 and S^3 is the best option, in this instance the centroid would be C^2; if we select three points, then combining S^4, S^5 and S^6 is the best option, this gives the centroid O. Fusing all six points is also a good option, but it may not be as good as fusing S^4, S^5 and S^6. The "many-could-be-better-than-all" theorem seems reasonable (Zhou et al., 2002) in this case. Because the centroid of a group of points is decided by the positions of all the points collectively, each of which has an equal weight, removing or adding even a single point may change the position of the centroid of a group of points significantly. It also indicates that it is not an easy task to find the best subset from a large group of base classifiers. As a matter of fact, it is an NP-hard problem, as our next theorem proves. □

Theorem 3 *$\mathcal{S} = \{S_1, S_2, \cdots, S_m\}$ for $(m \ge 3)$ is a group of points and O is an ideal point. For a given number m' $(2 \le m' < m)$, the problem is to find a subset of m' points from $\mathcal{S}$ to minimize the distance of the centroid of these m' points to the ideal point O. The above question is NP-hard.*

Proof First let us have a look at the maximum diversity problem (MDP), which is a known NP-hard problem (Wang et al., 2014). The MDP is to identify a subset $\mathcal{E}\simeq$ of m' elements from a set $\mathcal{E}$ of m elements, such that the sum of the pairwise distance between the elements in the subset is maximized. More precisely, let $\mathcal{E} = \{e_1, e_2, \cdots, e_m\}$ be a group of elements, and d_{ij} be the distance between elements e_i and e_j. The objective of the MDP can be formulated as:

Maximize

$$f(x) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} d_{ij} * x_i * x_j \tag{7}$$

subject to $\sum_{i=1}^{m} x_i = m', x_i \in \{0,1\}, i = 1, \cdots, m$

where each x_i is a binary (zero–one) variable indicating whether an element $e_i \in \mathcal{E}$ is selected to be a member of $\mathcal{E}_{\simeq}$.

On the other hand, according to Theorem 2, our problem may be written as minimizing $ed(C,O)^2$. Here C is the centroid of m' selected points.

$$ed(C,O)^2 = \frac{1}{m'}\sum_{i=1}^{m} ed(S^i,O)^2 * x_i - \frac{1}{m' * m'}\sum_{i=1}^{m}\sum_{j=1}^{m} ed(S^i,S^j)^2 * x_i * x_j$$

subject to $\sum_{i=1}^{m} x_i = m', x_i \in \{0,1\}, i = 1, \cdots, m$

where each x_i is also a binary (zero–one) variable indicating whether a point $S^i \in \mathcal{S}$ is selected to be a member of $\mathcal{S}_{\simeq}$.

If we assume that $ed(S^i,O)$ equals to each other for all $i = 1, \cdots, m$, then $\frac{1}{m'}\sum_{i=1}^{m} ed(S^i,O)^2 * x_i$ becomes a constant and minimizing $ed(C,O)^2$ equals to maximizing $div(C,O)^2$

$$div(C,O)^2 = \frac{1}{m' * m'}\sum_{i=1}^{m}\sum_{j=1}^{m} ed(S^i,S^j)^2 * x_i * x_j$$

Let $g(x)$=$div(C,O)^2$ and $d_{ij} = \frac{2}{m'*m'} ed(S^i,S^j)^2$, then the above equation can be rewritten as

$$g(x) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} d_{ij} * x_i * x_j \tag{8}$$

Comparing Eqs. (7) and (8), we can see that a simplified version of our problem is an MDP problem. Therefore, our problem is an NP-hard problem. □

Theorem 3 tells us: for a given number of classifiers, choosing a subset for best ensemble performance by majority voting is an NP-hard problem.

Although the "many-could-be-better-than-all" theorem may be applicable to some cases, it does not tell the whole story. Now let us have a look at how the size of the ensemble impacts its performance. Because the performance of an ensemble is affected by a few different factors, we need to find a way of separating this from other factors.

Theorem 4 *In a space X, $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ and O are known points. C is the centroid of $S^1, S^2, \cdots, S^m$, C^1 is the centroid of $m-1$ points $S^2, S^3, \cdots, S^m$, C^2 is the centroid of $m-1$ points $S^1, S^3, \cdots, S^m$,$\cdots$, C^m is the centroid of $m-1$ points $S^1, S^2, \cdots, S^{m-1}$. We have*

$$ed(C,O) \le \frac{1}{m}\sum_{i=1}^{m} ed(C^i,O) \tag{9}$$

Proof According to the definition of $C^1, C^2, \cdots, C^m$, C is the centroid of these m points. The theorem can be proven by applying Theorem 1. □

Theorem 4 can be used repeatedly to prove more general situations in which a subset includes $m-2$,$m-3$,$\cdots$,or 2 points. This demonstrates that the number of component results has a positive effect on ensemble performance. □

Example 5 In a space X, $\{S^1, S^2, S^3, S^4\}$ and O are known points. There are four different combinations of three points and six combinations of two points. We use C^{1234} to represent the centroid of all 4 points. Similarly, C^{23} represents the centroid of S^2 and S^3, and so on. Applying Theorem 4 repeatedly, we have

$$\begin{aligned} ed(C^{1234}, O) \leq & \frac{1}{4}[ed(C^{123}, O) + ed(C^{124}, O) + ed(C^{134}, O) + ed(C^{234}, O)] \\ \leq & \frac{1}{6}[ed(C^{12}, O) + ed(C^{13}, O) + ed(C^{14}, O) \\ & + ed(C^{23}, O) + ed(C^{24}, O) + ed(C^{34}, O)] \\ \leq & \frac{1}{4}[ed(S^1, O) + ed(S^2, O) + ed(S^3, O) + ed(S^4, O)] \end{aligned}$$

□

Although Theorem 4 shows that the number of component classifiers has a positive effect on ensemble performance, it is not clear how significant the effect is. Next let us look at this matter quantitatively. In order to focus on the number of component classifiers, we make a few simplifying assumptions. Suppose that S^1, S^2,$\cdots$, S^m and O are known points in space X. $ed(S^i, O)$=c_p for any ($1 \leq i \leq m$), $ed(S^i, S^j)$=c_d for any ($1 \leq i \leq m$, $1 \leq j \leq m$, $i \neq j$), and $c_d = \theta * c_p$. According to Theorem 2 and the above assumptions, we have

$$\begin{aligned} ed(C, O)^2 = & \frac{1}{m^2}[m \sum_{i=1}^{m} ed(S^i, O)^2 - \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} ed(S^i, S^j)^2] \\ = & {c_p}^2 - \frac{m-1}{2m} {c_d}^2 \\ = & (1 - \frac{m-1}{2m}\theta^2) {c_p}^2 \end{aligned}$$

Therefore,

$$ed(C, O) = \sqrt{(1 - \frac{m-1}{2m}\theta^2)} * c_p \tag{10}$$

By definition, $ed(C, O)$ cannot be negative and $(1 - \frac{m-1}{2m}\theta^2) \geq 0$ should hold. Therefore, for a given m, θ must have a maximal limit. If $m = 2$ and $\theta = 2$, then $ed(C, O) = 0$. $\theta = 2$ must be the maximal value in this case. Likewise, when $m = 3$, the maximal value for θ is $\sqrt{3}$.

Figure 1 shows the values of $ed(C, O)$ in c_p unit for $\theta = 0.25, 0.5, 0.75, 1$ and $m = 2, 3, \cdots, 100$. From Fig. 1 we can see that in all four cases, $ed(C, O)$ decreases with m. However, as m becomes larger and larger, the rate of decrease becomes smaller and smaller. When m tends to infinity, $ed(C, O)$ approaches $\sqrt{1 - 0.5\theta^2}$ c_p units. They are 0.984, 0.935, 0.848, and 0.707 when θ equals to 0.25, 0.5, 0.75, and 1, respectively. However, adding more component results does not help much if there is already a large number. For example, when $\theta = 1.0$ and $m = 27$, $ed(C, O)$ is 0.720 c_p units. It is close to the limit of 0.707 c_p units. It suggests that fusing 30 or more component results may not be very useful

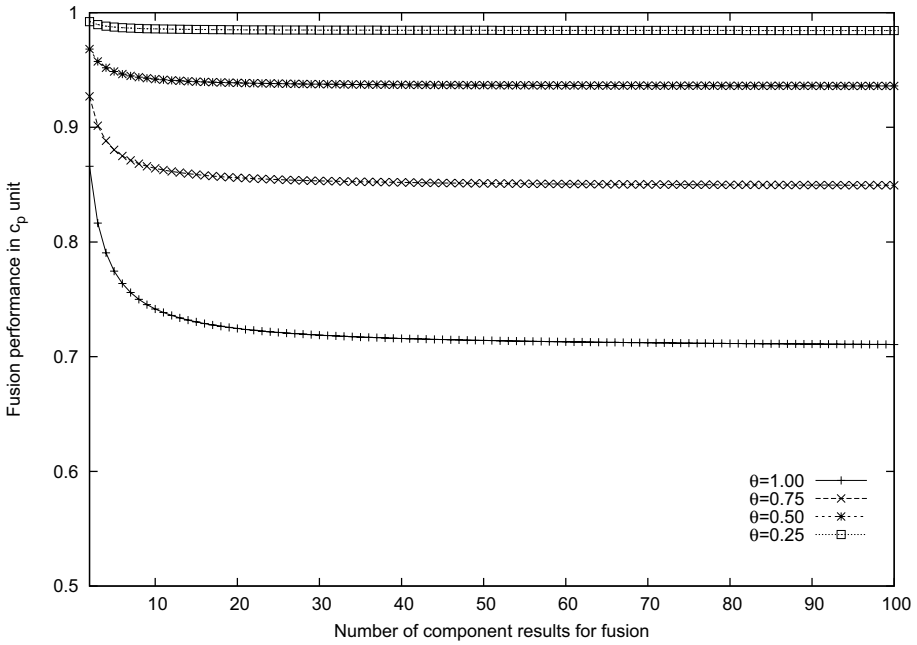


Fig. 1 The impact of component classifier number on ensemble performance

for further improving ensemble performance. This has been observed in some empirical studies before, such as in Oshiro et al. (2012), and others.

Theorem 1 tells us that the ensemble performance is at least as good as the average performance of all the component classifiers involved. This may not be positive enough for many applications of the technique. Theorem 3 indicates that it may take too much time to choose a subset from a large group of candidates for good ensemble performance. This problem may be solved in other ways. In particular, if we want the ensemble performance to be better than the best component classifier, more favorable conditions are required for those component classifiers. It means that we need to apply some restrictions to all the component classifiers involved. Theorem 5 can be useful for this.

Theorem 5 *In a space* X, $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ *and* O *are known points. At least one of the points in* $\mathcal{S}$ *is different from the others.* C *is the centroid of* $S^1, S^2, \cdots, S^m$. *If* $ed(S^1, O) = ed(S^2, O) = \cdots = ed(S^m, O)$, *then* $ed(C, O) < ed(S^1, O)$ *must hold.*

Proof According to Theorem 2, we have

$$
\begin{aligned}
ed(C,O)^2 &= \frac{1}{m}\sum_{i=1}^{m} ed(S^i,O)^2 - \frac{1}{m*m}\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(S^i,S^j)^2 \\
&< \frac{1}{m}\sum_{i=1}^{m} ed(S^i,O)^2 = ed(S^1,O)^2
\end{aligned}
$$

Therefore, we obtain $ed(C, O) < ed(S^1, O)$. □

Theorem 5 tells us if all the component classifiers are equally effective, then majority voting is able to do a better job than Theorem 1's guarantee. In practice, this has been implemented in various situations. For example, if using bagging with random forest or neural networks (Oshiro et al., 2012; Yang et al., 2013), then we are can generate a large number of almost equally-effective component classifiers. Good performance is achievable by fusing such classifiers. See Sect. 4 for more discussions.

3.2 Weighted majority voting

Weighted majority voting is a generalization of majority voting. It is more flexible than its counterpart because different weighting schemes can be de-fined. It might be believed that both are similar, and this is indeed true for the cases when weights across component classifiers are largely similar. However, for the weighted majority voting, we have little interest in the universe of all possible weighting schemes, but are rather focused on the optimum weighting scheme. We delve into the following two questions especially.

1. How to find the optimum weights for a group of component classifiers?
2. What are the properties of weighted majority voting with the optimum weights?

Let us begin with the first question. In a space X, $\mathcal{S} = \{S^1, S^2, \cdots, S^m\}$ and O are known points. Let F be the fused point for linear combination of m points in $\mathcal{S}$ with weighting $w^1, w^2, \cdots, w^m$.

$$ed(F,O)^2 = \sum_{i=1}^{n}\sum_{j=1}^{p}\left(\sum_{k=1}^{m}(w^k * s_{ij}^k) - o_{ij}\right)^2 \tag{11}$$

Our goal is to minimize $ed(F,O)^2$. Assuming $f(w^1, w^2, \cdots, w^m) = ed(F,O)^2$, we have

$$\frac{\partial f}{\partial w^q} = \sum_{i=1}^{n}\sum_{j=1}^{p} 2\left(\sum_{k=1}^{m}(w^k * s_{ij}^k) - o_{ij}\right)s_{ij}^q$$

Let $\frac{\partial f}{\partial w^q} = 0$ for (q=1,2,$\cdots$,m), then

$$\sum_{k=1}^{m} w^k \sum_{i=1}^{n}\sum_{j=1}^{p}(s_{ij}^k s_{ij}^q) = \sum_{i=1}^{n}\sum_{j=1}^{p} o_{ij}s_{ij}^q$$

Let $a_{qk} = \sum_{i=1}^{n}\sum_{j=1}^{p} s_{ij}^k s_{ij}^q$ for $(1 \le q \le m)$ and $(1 \le k \le m)$, and $b_q = \sum_{i=1}^{n}\sum_{j=1}^{p} o_{ij}s_{ij}^q$ for $(1 \le q \le m)$. Thus we obtain the following m linear equations with m variables $w^1, w^2, \cdots, w^m$:

$$\begin{pmatrix} a_{11}w^1 + a_{12}w^2 + \cdots + a_{1m}w^m = b_1 \\ a_{21}w^1 + a_{22}w^2 + \cdots + a_{2m}w^m = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{m1}w^1 + a_{m2}w^2 + \cdots + a_{mm}w^m = b_m \end{pmatrix} \tag{12}$$

The optimum weights can be calculated by finding the solution to these m linear equations. Note that minimizing $ed(F, O)^2$ and minimizing $ed(F, O)$ is equivalent for us to find the optimum weights because $ed(F, O)$ can not be negative.

Theorem 6 *In a $n * p$ dimensional space X, $\mathcal{S}$=$\{S^1, S^2, \cdots, S^m\}$ and O are known points. If every point in $\mathcal{S}$ is linearly independent from the others, then the above process and Eq. (12) can find the unique solution to the problem.*

Proof The independency of each point in $\mathcal{S}$ indicates that $m \leq n * p$ holds. For the same reason, any point that can be represented linearly by these m points has a unique representation. We may write $ed(F, O)$ as $f(w^1, w^2, \cdots, w^m)$, which is a continuous function. In the whole space, there is only one minima and no other saddle points or maxima. The point at which all partial derivatives of the function to all variables equal to zero must be the minimum point. The m equations set up in Eq. (12) are able to find the point with a unique representation of weights. □

Intuitively, in a $n * p$ dimensional space X, m points in $\mathcal{S}$=$\{S^1, S^2, \cdots, S^m\}$ comprise a subspace X' in X. For any point O, there exists one and only one point in X' that has the shortest distance to O. This point can be linearly represented by $S^1, S^2, \cdots, S^m$.

Theorem 6 can be explained as follows. For a (training) dataset with n instances, p classes, and m classifiers, each of the classifiers gives a score for each instance and each class. For each instance, we also have real labels relating to all the classes. Then we are able to find a group of weights $w^1, w^2, \cdots, w^m$ for $S^1, S^2, \cdots, S^m$ to achieve the best ensemble performance by weighted majority voting.

The following Theorem 7 answers the second question.

Theorem 7 *In a $n * p$ dimensional space X, $\mathcal{S}^1$=$\{S^1, S^2, \cdots, S^m\}$, $\mathcal{S}^2$=$\{S^1, S^2, \cdots, S^m, S^{m+1}\}$, and O is an ideal point. If the optimum weights are used for both $\mathcal{S}^1$ and $\mathcal{S}^2$, then the performance of Group $\mathcal{S}^2$ is at least as effective as that of Group $\mathcal{S}^1$.*

Proof Assume that $w^1, w^2, \cdots, w^m$ are optimum weights for $S^1, S^2, \cdots, S^m$ of $\mathcal{S}^1$ to obtain the best performance. For $\mathcal{S}^2$, if using the same weights $w^1, w^2, \cdots, w^m$ for $S^1, S^2, \cdots, S^m$, and 0 for S^{m+1}, then the ensemble performance of $\mathcal{S}^2$ will be the same as that of $\mathcal{S}^1$. Note that the above weighting scheme is by no means the best for $\mathcal{S}^2$ and it is also possible to find more profitable weights for $\mathcal{S}^2$. □

Theorem 7 can be explained as follows: for a given dataset, consider two groups of classifiers. Group 1 has n classifiers: $cf_1, cf_2, \cdots, cf_m$, and Group 2 has $m + 1$ classifiers, $cf_1, cf_2, \cdots, cf_m, cf_{m+1}$. m classifiers in both groups are the same. If using weighted majority voting with the optimum weights, then the ensemble performance of Group 2 is at least as effective as that of Group 1.

Corollary 7.1 *In a $n * p$ dimensional space X, assume that weighted majority voting is applied with the optimum weights. When more and more points are added, the ensemble performance is monotonically non-decreasing.*

Proof It can be proven by applying Theorem 7 repeatedly. □

Intuitively, when more and more points are added, the subspace becomes bigger and bigger. During this process, it is possible to find new points that has the shortest distance to the ideal point.

Corollary 7.1 can be explained in this way. Assume for a given dataset, an ideal ensemble is implemented by weighted majority voting with the optimum weights. When more and more classifiers are added into such an ensemble, its performance is non-decreasing monotonically.

Theorem 2 tells us about how individual classifier performance ($ed(S^i, O)$) and dissimilarity between classifiers ($ed(S^i, S^j)$) affect ensemble performance with the majority voting scheme. We may regard weighted majority voting as a variation of majority voting. Before applying Theorem 2, weighted majority voting changes the positions of all component results' position by a linear weighting scheme, thus Eq. (5) becomes

$$ed(C, O) = \frac{1}{m}\sqrt{m\sum_{i=1}^{m} ed(w^i \cdot S^i, O)^2 - \sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ed(w^i \cdot S^i, w^j \cdot S^j)^2} \tag{13}$$

where $w^i \cdot S^i = \sum_{j=1}^{n}\sum_{k=1}^{p} w^i * s^i_{jk}$. After that, both can be treated in the same way.

4 Discussion

In Sect. 3 we have set up a geometric framework and presented the properties of majority voting and weighted majority voting. Now we are in a good position to compare the two different levels of geometric frameworks and answer the questions raised in the first section of this paper.

4.1 Dataset-level vs. instance-level frameworks

A major objective of the ensemble problem is to try to provide a solution for all the instances in a dataset. A framework at different levels has certain impact on the way we can deal with the problem. For an instance-level framework, we need an approach to expand it to cover all the instances in the whole dataset, while the dataset-level framework does not need it.

In the dataset-level framework, all the instances in the whole dataset are concatenated to form a super-instance. Thus, all the properties stand in the instance-level framework also stand in the dataset-level framework.

However, a few differences need to be noted. One is the dimensionality of the geometric space involved. For a classification problem with p classes and a dataset with n instances, the dimensionality of the instance-level geometric space is p, while that of the dataset-level geometric space is $p * n$.

How to calculate optimal weights for weighted majority voting is another place where we may have different solutions. The solution given in Sect. 3 of this paper is to minimize the Euclidean distance between the linear combination of all the component points and the ideal point (refer to Equation 11). Recall that all the instances in the dataset is transformed to a single super-instance. However, for the instance-level framework, we still need to consider multiple instances together. One possible way is to minimize the sum of the distance over all instances, or

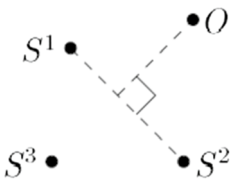
$$\sum_{i=1}^{n} ed(F,O) = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{p} (\sum_{k=1}^{m} (w^k * s_{ij}^k) - o_{ij})^2} \tag{14}$$

It is tricky to optimise Equation 14 directly. To simplify, we may optimise $\sum_{i=1}^{n} ed(F,O)^2$ instead (Wu & Crestani, 2015; Bonab & Can, 2018). In this way, it is the same as Equation 11. It demonstrates that there are connections between the two levels of frameworks. For the instance-level framework, optimising $\sum_{i=1}^{n} ed(F,O)^2$ approximates optimising $\sum_{i=1}^{n} ed(F,O)$. On the other hand, for the dataset-level framework, optimising $ed(F,O)^2$ is the same as optimising $ed(F, O)$. It means that the weights obtained are optimum.

4.2 The size of ensemble

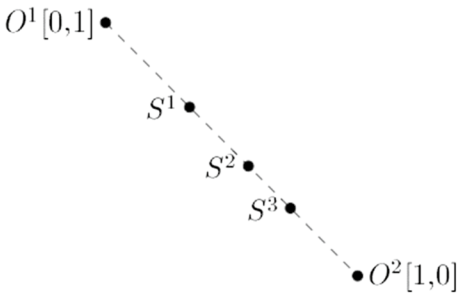
As discussed in Sect. 3, we proved that the number of component classifiers has positive impact on ensemble performance for both majority voting and weighted majority voting. However, this contradicts the assertion in Bonab and Can (2019): for a multi-classification problem with k classes, k is the ideal number of base classifiers to constitute an optimum ensemble by weighted majority voting. Let us analyse this further.

Example 6 Consider a classification problem with two classes and three base classifiers. One instance is shown in the figure below. Weighted majority voting is used for combination.



In the figure above, it shows a two-dimensional space with three points S^1, S^2, S^3 to be a combination. The ideal point is O. We can see that combining three points can lead to the optimal results of zero distance than fusing any two. Therefore, in this example the assertion in Bonab and Can (2019) even does not hold at the instance level. However, as shown in this example, for a n dimensional space, $n + 1$ independent points are enough. □

We may add some restrictions to the points involved. For example, for a binary classification problem, we let all the points to be on the line segment of [0,1] and [1,0]. The ideal point is either [1,0] or [0,1]. In this way, a maximum of two points are needed for the optimal fusion results. In the figure below, any two of the three points S^1, S^2, and S^3 are competent for this task.



Anyhow, the assertation at the instance level is not very useful. To consider the problem in a more realistic way, we need to look at it at the dataset level. A dataset usually comprises at least a good number of instances. If we consider three instances with a binary classification problem, then the dimensionality of the dataset-level geometric framework is up to 2*3=6, and not 2 any more. If the dataset has more instances, then we may include even more independent classifiers. With an increased number of base classifiers, we will likely get better ensemble performance (Corollary 7.1).

On the other hand, we can obtain the same conclusion as Corollary 7.1 even under the instance-level framework. Assume that the whole dataset has n instances. F_i and O_i are the fused point and the ideal point for instance t_i, respectively. The optimal weighting for m base classifiers are w^1,w^2,$\cdots$,w^m. Then we have

$$\sum_{i=1}^{n} ed(F_i, O_i) = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{p} (\sum_{k=1}^{m} (w^k * s_{ij}^k) - o_{ij})^2} \tag{15}$$

Now one more base classifier is added. We can set a new weighting scheme as $w_{new}^1 = w^1$,$w_{new}^2 = w^2$,$\cdots$,$w_{new}^m = w^m$, $w_{new}^{m+1} = 0$.

$$\sum_{i=1}^{n} ed_{new}(F_i, O_i) = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{p} (\sum_{k=1}^{m+1} (w_{new}^k * s_{ij}^k) - o_{ij})^2} \tag{16}$$

Then $\sum_{i=1}^{n} ed(F_i, O_i) = \sum_{i=1}^{n} ed_{new}(F_i, O_i)$. w^1,w^2,$\cdots$,w^m is the optimal weighting scheme for m base classifiers, while w_{new}^1,w_{new}^2,$\cdots$,w_{new}^m, w_{new}^{m+1} may not be optimal for $m + 1$ base classifiers. Therefore, if the optimal weighting scheme is used, then fusing $m + 1$ base classifiers can achieve at least the same performance as fusing m base classifiers. This is exactly what Corollary 7.1 tells us.

4.3 Answer to some questions

In Sect. 1, we listed some outstanding questions. Now let us discuss them one by one.

Question 1: What is the difference between majority voting and weighted majority voting?

In a sense, both weighting schemes can potentially enhance ensemble performance. However, there are certain aspects, including their abilities, to consider. Weighted majority

voting can be better than the best component classifier if optimum weights are used, while majority voting can be better than the average of all component classifiers. Majority voting is a "mild" method because all the component results are treated equally and the centroid is the solution, while weighted majority voting is an "extreme" method because it does not treat all component results equally and it tries to take the most effective solution from all possible ones.

Question 2: When should we use majority voting rather than weighted majority voting, or vice versa?

Theoretically, weighted majority voting is always better than or at least as good as majority voting if optimal weights are applied and Euclidean distance is used for performance evaluation. However, in reality the above two conditions may not be satisfied. The weights leant in the training dataset may not be the best in the test dataset. Therefore, a general answer is: in cases majority voting does not work well (e.g., ensemble performance is worse than that of the the best component classifier), then weighted majority voting should be used.

Now a further question is: when is majority voting a good method? Performance of all component results, dissimilarity of all pairs of component results, number of component results have positive impact on ensemble performance. A judicious decision should consider these factors thoroughly. More specifically, Theorem 2 answers this question quantitatively. One easy noticeable situation is that when all the component classifiers are of equal or very close performance, then majority voting may be able to achieve better ensemble performance than the best component classifier (Theorem 5).

Question 3: There are a lot of weighting assignment methods for weighted majority voting, which one is the best?

Optimality is a complicated issue and it is related to the goal of optimisation. There are many diffent measures and any individual method cannot do the best work for all those measures. The least squares is the best weighting assignment method for the measure of Euclidean distance. Compared with many others, it is efficient and effective at the same time. Almost all other weighting assignment methods are either heuristic or optimisation methods. For the former, its effectiveness is not guaranteed; for the latter, it is timing-consuming.

Question 4: How does the number of component classifiers affect ensemble performance?

The number of component classifiers has a positive effect on ensemble performance. The situation is straightforward for weighted majority voting. When more and more component results are added to an ensemble, its performance becomes better and better. Here we assume that the optimal weights are used in the ensemble.

On the other hand, the situation for majority voting is more complicated. Adding more component classifiers into an ensemble cannot always improve performance. From a group of candidates, to find a subset for best ensemble performance is a NP-hard problem (Theorem 3).

Question 5: How does each component classifier affect ensemble performance?

It is related to the first question. For majority voting, each contributes equally; for weighted majority voting, each contributes differently in order to get the optimal results for the whole data set. If a very good component classifier is added, then weighted majority voting can take advantage of it. On the other hand, for majority voting, if many component classifiers are poor, then a few good ones may not be able to improve performance very much.

Question 6: How does performance of component classifiers affect ensemble performance?

Performance of component classifiers is the most important aspect that affect ensemble performance. For majority voting, a high-performance point is able to move the centroid of the group closer to the ideal point. For weighted majority voting, a high-performance point very likely enables the subspace to expand with some points closer to the ideal point.

Question 7: How does diversity of component classifiers affect ensemble performance?

For diversity, there are many different types of definitions before. In this paper, we define it as the dissimilarity distribution of all pairs of component results. Apart from performance, diversity is another aspect that impacts ensemble performance significantly. For majority voting, a comparative investigation about it and performance has been done in Sect. 3.1. Based on a simplified situation, the importance ratio between diversity and performance is calculated to be in the range of (0.25,0.5], varying with the number of component classifiers. For weighted majority voting, high diversity among component results will make the subspace bigger, thus it is more likely to find closer points to the ideal point in such a space.

The advantage of the diversity measure defined in this paper is its orthogonality to performance, which makes its effect on ensemble performance very clear. It is also helpful when we try to select a subset of base classifiers from all available ones (classifier pruning (Mohammed et al., 2022)) for effective and efficient ensembling.

One final comment about the framework is: all the theorems in the geometric framework hold when the Euclidean distance is used for measuring performance. When other metrics are used, the conclusions we obtain may hold for many of the instances, but not every single instance. However, there is strong correlations between any other meaningful performance metrics and the Euclidean distance. If enough instances are observed, we may expect consistent conclusions.

5 Empirical investigation

In this section we are going to investigate how theoretical conclusions presented in Sect. 3 can be confirmed for practical use. Within the geometric framework, all the theorems hold perfectly when Euclidean distance is used as performance metric on the same dataset. We would like to see how they behave when the conditions are partially satisfied. Specifically, two points are considered:

- Quite a few weighting schemes have been proposed before for ensemble learning. We are going to see how the newly proposed optimal scheme is related to two typical weighting schemes proposed before. We also compare their performance and efficiency through experiments.
- Usually classification accuracy or some other metrics, rather than Euclidean distance, is used for performance evaluation. It is interesting to find the strength of the correlation between the Euclidean distance and other commonly used metrics.

Table 2 A data set used for training weights; it includes m base classiers, n instances, and three classes (a, b, and c); s^i_{jk} refers to the probability score given by base classifier cf_i for class k on instance number j

Instance	Class	Classifier cl_i		
		a	b	c
$Instance_1$	a	S^i_{1a}=0.90	S^i_{1b}=0.05	S^i_{1c}=0.05
$Instance_2$	b	S^i_{2a}=0.15	S^i_{2b}=0.70	S^i_{2c}=0.15
$Instance_3$	b	S^i_{3a}=0.10	S^i_{3b}=0.80	S^i_{3c}=0.10
$Instance_4$	c	S^i_{4a}=0.20	S^i_{4b}=0.20	S^i_{4c}=0.60
⋮	⋮	⋮	⋮	⋮
$Instance_n$	a	S^i_{na}=0.80	S^i_{1b}=0.10	S^i_{1c}=0.10

Table 3 Meta training set for class a, Stacking with MLR

Classifier cl_1			Classifier cl_2				Classifier cl_n			class
a	b	c	a	b	c		a	b	c	$=a$?
S^1_{1a}	S^1_{1b}	S^1_{1c}	S^2_{1a}	S^2_{1b}	S^2_{1c}	⋯	S^n_{1a}	S^n_{1b}	S^n_{1c}	1
S^1_{2a}	S^1_{2b}	S^1_{2c}	S^2_{2a}	S^2_{2b}	S^2_{2c}	⋯	S^n_{2a}	S^n_{2b}	S^n_{2c}	0
S^1_{3a}	S^1_{3b}	S^1_{3c}	S^2_{3a}	S^2_{3b}	S^2_{3c}	⋯	S^n_{3a}	S^n_{3b}	S^n_{3c}	0
S^1_{4a}	S^1_{4b}	S^1_{4c}	S^2_{4a}	S^2_{4b}	S^2_{4c}	⋯	S^n_{4a}	S^n_{4b}	S^n_{4c}	0
	⋮			⋮				⋮		⋮
S^1_{na}	S^1_{nb}	S^1_{nc}	S^2_{na}	S^2_{nb}	S^2_{nc}	⋯	S^n_{na}	S^n_{nb}	S^n_{nc}	1

To allow for reproducbility, the datasets, source code, raw results of this study are available in GitHub.[1]

5.1 Weighting schemes for ensemble learing

Before presenting the results of empirical investigation, let us further discuss the weighting schemes proposed in this paper and two others proposed before.

Recall the optimal weighting scheme, we may obtain it through minimizing the Euclidean distance between the ensemble point and the ideal point. See Eq. (11) for it. Actually, this can be implemented by applying multiple linear regression ,[2]. We illustrate it using an example taken from Seewald (2002) with some necessary modifications. Suppose that the training data set includes m base classifiers, a group of n instances, and three class labels a, b, and c. Table 2 shows an example of the original training set (left side) and class probability distribution of a base classifier (cl_i) (right side).

Stacking (Ting & Witten, 1999) and StackingC with multiple linear regression (MLR) (Seewald, 2002) are two typical weighting schemes for ensemble learning. Multiple linear regression is used for both of them. It is also noticeble that the optimal weights by minimizing the Euclidean distance can also be solved by multiple linear regression. This method is referred to as ED (Euclidean Distance) with MLR later in this paper. Although many other weighting schemes have also been proposed (e.g., in Caruana et al. (2004);

[1] https://github.com/MuxLZ/-Ensemble-Learning-experiment

[2] Linear regression is a stable method (Bousquet et al., 2003) its genelization error bound is given in Elisseeff (2000)

Table 4 Meta training set for class a, StackingC with MLR

$classifier_{cl_1}$ a	$classifier_{cl_2}$ a		$classifier_{cl_n}$ a	class $=a$?
S^1_{1a}	S^2_{1a}	$\cdots$	S^n_{1a}	1
S^1_{2a}	S^2_{2a}	$\cdots$	S^n_{2a}	0
S^1_{3a}	S^2_{3a}	$\cdots$	S^n_{3a}	0
S^1_{4a}	S^2_{4a}	$\cdots$	S^n_{4a}	0
$\vdots$	$\vdots$		$\vdots$	$\vdots$
S^1_{na}	S^2_{na}	$\cdots$	S^n_{na}	1

Table 5 Meta training set for all the classes, ED with MLR; the last collumn is not a part of the training data

$classifier_{cl_1}$	$classifier_{cl_2}$		$classifier_{cl_n}$	label	Note
S^1_{1a}	S^2_{1a}	$\cdots$	S^n_{1a}	1	Class a
S^1_{2a}	S^2_{2a}	$\cdots$	S^n_{2a}	0	Related
S^1_{3a}	S^2_{3a}	$\cdots$	S^n_{3a}	0	Scores
S^1_{4a}	S^2_{4a}	$\cdots$	S^n_{4a}	0	
$\vdots$	$\vdots$		$\vdots$	$\vdots$	
S^1_{nb}	S^2_{nb}	$\cdots$	S^n_{nb}	1	
S^1_{1b}	S^2_{1b}	$\cdots$	S^n_{1b}	0	Class b
S^1_{2b}	S^2_{2b}	$\cdots$	S^n_{2b}	1	Related
S^1_{3b}	S^2_{3b}	$\cdots$	S^n_{3b}	1	Scores
S^1_{4b}	S^2_{4b}	$\cdots$	S^n_{4b}	0	
$\vdots$	$\vdots$		$\vdots$	$\vdots$	
S^1_{nb}	S^2_{nb}	$\cdots$	S^n_{nb}	0	
S^1_{1c}	S^2_{1c}	$\cdots$	S^n_{1c}	0	Class c
S^1_{2c}	S^2_{2c}	$\cdots$	S^n_{2c}	0	Related
S^1_{3c}	S^2_{3c}	$\cdots$	S^n_{3c}	0	Scores
S^1_{4c}	S^2_{4c}	$\cdots$	S^n_{4c}	1	
$\vdots$	$\vdots$		$\vdots$	$\vdots$	
S^1_{nc}	S^2_{nc}	$\cdots$	S^n_{nc}	0	

Mao et al. (2015); Nguyen et al. (2019); Sen and Erdogan (2013); Ting and Witten (1999); Zhang and Zhou (2011)), these three are very closely related to each other. The training sets used by these three methods are shown in Tables 3, 4, 5, respectively.

In both Stacking and StackingC, a multiple linear regression model is set to learn the weights for each class label. Therefore p models are required for a data set with p classes. However, Stacking uses all available $m \times p$ variables, while StackingC uses m variables, which are related to the given class, in each regression model. ED only uses one regression model for all the classes and m variables are invovled in its regression model. In a sense, StackingC is a simplified version of Stacking and ED is a simplified version of StackingC.

Note that for binary classification problems, the three weighting schemes are almost the same. Although more variables are used in Stacking, half of the variables are redundant. This is because the probablity scores of any instance j obtained from a base classifier

Table 6 Statistics of the datasets used in the study

Data set	# Instances	# Features	# Classes
Abalone (*Ab*)	4177	8	3
Annealing (*An*)	798	38	5
Audiology (*Au*)	226	69	18
Bank Marketing (*Ba*)	4520	17	2
Breast-tissue (*Br*)	106	10	6
Bupa (*Bu*)	345	7	2
Car (*Ca*)	1728	6	4
Chart (*Ch*)	600	60	6
Conn (*Co*)	528	11	11
Connectionist (*Ct*)	208	60	2
Dermatology (*De*)	366	34	6
Energy (*En*)	768	8	3
Glass (*Gl*)	214	10	6
Iris (*Ir*)	214	4	3
Knowledge (*Kn*)	172	5	4
Leaf (*Le*)	340	15	30
Lenses (*Ls*)	24	4	3
Lymphography (*Ly*)	148	18	4
Pen-Based Recognition (*Pe*)	10992	16	10
Pittsburg-bridges-MATERIAL (*PM*)	108	13	3
Pittsburg-bridges-TYPE (*PT*)	108	13	6
Primary-tumor (*Pr*)	339	17	15
Seeds (*Se*)	210	7	3
Soybean (*So*)	683	35	18
Statlog (*St*)	846	18	4
Zoo (*Zo*)	101	17	7

cl_i sum to 1: $S^i_{ja} + S^i_{jb} = 1$). Using either S^i_{ja} or S^i_{jb} is enough. Due to the same reason, the weights learnt by either Stacking or StackingC are the same for both classes a and b. Therefore, it is enough to learn weights for one of the classes.

For multiclass classification problems, the model trained by Stacking is more accurate than the one by StackingC, and the model trained by StackingC is more accurate than the one by EU with regard to the training dataset. However, this may not always be the case with regard to the test dataset. More complex models have a bigger chance to be overfitting. Such a phenomenon is observed in Seewald (2002). It is also confirmed in our experiments. See later sections for detailed results.

5.2 Comparison of three weighting schemes

In this subsection we investigate empirically three weighting schemes including Stacking, StackingC, and ED. 26 datasets, downloaded from the UCI Machine Learning Repository,[3]

[3] https://archive.ics.uci.edu/ml/index.php

Table 7 Performance (in Accuracy) comparison of three weighting schemes (R denotes the performance difference between the best and worst base classifiers)

Data set	Stacking	StackingC	ED	Majority-Voting	R >10%
Ab	0.671	0.672	0.674	**0.691**	No
An	**0.941**	**0.941**	0.940	0.830	Yes
Au	0.912	**0.919**	0.914	0.911	Yes
Ba	**0.900**	**0.900**	**0.900**	**0.900**	No
Br	0.633	0.633	**0.701**	0.700	Yes
Bu	0.726	0.727	0.724	**0.747**	No
Ca	**0.970**	**0.970**	**0.970**	0.960	Yes
Ch	0.984	0.986	**0.987**	0.982	Yes
Co	**0.816**	0.775	0.759	0.762	No
Ct	0.832	0.832	0.827	**0.833**	No
De	0.970	0.974	**0.979**	0.970	Yes
En	0.940	0.940	**0.941**	0.900	Yes
Gl	0.684	0.706	**0.709**	0.702	Yes
Ir	0.956	0.956	0.957	**0.958**	No
Kn	0.884	**0.885**	0.877	0.866	No
Le	0.544	0.589	0.606	**0.607**	Yes
Ls	0.631	0.685	**0.725**	0.692	Yes
Ly	0.856	0.856	0.859	**0.860**	No
Pe	**0.993**	**0.993**	**0.993**	0.988	Yes
PM	0.845	0.858	0.860	**0.862**	Yes
PT	0.526	0.582	**0.601**	0.585	Yes
Pr	0.536	0.573	**0.589**	0.580	Yes
Se	0.939	**0.940**	0.937	0.936	No
So	0.941	**0.944**	0.941	0.931	Yes
St	0.764	0.768	0.766	**0.771**	Yes
Zo	0.971	**0.972**	0.951	0.926	No
Average	0.822	0.830	**0.834**	0.825	
Average(Yes)	0.801	0.816	**0.826**	0.811	

The best performance in each data set is shown in bold

are used for this. The main statistics of the these 26 datasets are listed in Table 6. These datasets vary in number of instances, features, and classes.

Three types of base classifiers, including decision tree, support vector machine, and logistic regression-based classifiers, were involved for the ensemble.

For each dataset, we divided it into five equal subsets. All the instances were randomly allocated to each subset. Two of the subsets were used to generate base classifiers, another two were used to train weights for the ensemble, and the remaining one were used for testing. All 30 different combinations were tried. In order to reduce the impact of randomness of selection, we repeat the above process 20 times for each dataset. The results are shown in Table 7. The figure is the average over 20 iterations and 30 combinations for each dataset.

From Table 7 we can see that StackingC and ED are close in performance, while Stacking is not as good as the others. However, the difference between them is small and not

Table 8 Time (in second) for weights training by different methods

Data set	Stacking	StackingC	ED
Ba	0.0013	0.0012	0.0010
Ca	0.0017	0.0013	0.0008
De	0.0017	0.0014	0.0006
En	0.0013	0.0011	0.0008
Gl	0.0016	0.0013	0.0006
Pe	0.0251	0.0047	0.0030
So	0.0162	0.0046	0.0009

significant if we consider all the cases together. In one of the datasets (*Ba*), all four weighting schems equally-performed. Out of the remaining 25 datasets, majority-voting wins eight of them. Stacking, StackingC, and ED are winners in 4, 8, and 10 datasets, respectively. Note that in two datasets *Ca* and *Pe*, these three are joint winners; and in one dataset *An*, Stacking and StackingC are joint winners.

In some cases, weighted ensemble does not perform as good as simple majority-voting. This happens when the condition is unfavorable to weighted ensemble. If all the base classifiers are close in performance, then it is harder for weighted ensemble to beat majority-voting. Therefore, we measured the performance of base classifiers and calculated the difference ratio of the best and worst *R*. Two groups are formed base on the difference (10% as the threshold). If only considering one group of 16 datasets in which R>10% (labelled "yes" in the column "R>10%" in Table 7), we find that the difference between Stacking and StackingC, and Stacking and ED is significant (paired-samples T test, two-sided, 0.010 and 0.008 respectively); while the difference between StackingC and ED is 0.055, just out of the signicance level of 0.05.

We also measured the time required for the training of these three weighting schemes. A personal desk computer with an i7-11700 CPU and 16 G RAM was used. Table 8 shows the results on seven selected datasets. All three methods run very fast. Understanably, ED is the fastest, Stacking is the slowest, while StackingC is in the middle. When a dataset has a small number of instances and a small number of classes, their difference is also small. This is the case for *Ba*, *Ca*, *De*, *En*, and *GL*. When a dataset has a larger number of classes, their difference becomes larger. *Pe* has over 10,000 instances and 10 classes, and *So* only has 683 instances but relatively a large number of 18 classes. So the time difference between Stacking and ED is over 8 times for *Pe* (Pen-Based Recognition) and 18 times for *So* (Soybean).

5.3 Correlation between Euclidean distance and other metrics

In this subsection we investigate the strength of correlation between Euclidean distance and other commonly used metrics.

In fact, Euclidean distance and all other reasonable metrics are directly related to probability scores. Considering one instance in a binary classification problem, a distance of below 0.5 always means a wrong classification, while a distance of above 0.5 alway means a correct classification. Therefore, an observable property of Euclidean distnace is: it is more sensitive to the change of probability scores than all other metrics. However, if a

Table 9 Correlation (by Pearson correlation coefficient) of different metric pairs)DS denotes Dataset, A denotes Accuracy, P denotes Precision, R denotes Recall, F denotes F1, and E denotes Euclidean distance)

DS	A &P	A &R	A &F	A &E	P &R	P &F	P &E	R &F	R &E	F &E
Ab	0.92	1.00	0.95	−0.91	0.94	0.95	−0.92	0.97	−0.91	−0.88
An	0.95	0.97	0.98	−0.94	0.96	0.97	−0.93	1.00	−0.92	−0.93
Au	0.73	0.71	0.73	−0.89	0.84	0.93	−0.67	0.90	−0.64	−0.66
Ba	0.76	0.91	0.91	−0.87	0.76	0.83	−0.76	0.99	−0.91	−0.91
Br	0.99	1.00	1.00	−0.99	1.00	1.00	−1.00	1.00	−0.99	−1.00
Bu	0.86	0.97	0.95	−0.92	0.89	0.93	−0.74	0.98	−0.93	−0.87
Ca	0.72	0.84	0.80	−0.92	0.94	0.98	−0.81	0.99	−0.86	−0.85
Ch	1.00	1.00	1.00	−0.99	1.00	1.00	−0.98	1.00	−0.99	−0.99
Co	0.94	1.00	0.99	−0.95	0.94	0.97	−0.90	0.99	−0.95	−0.94
Ct	0.98	1.00	1.00	−0.93	0.97	0.97	−0.93	1.00	−0.93	−0.93
De	0.99	1.00	1.00	−0.98	0.99	1.00	−0.99	1.00	−0.98	−0.98
En	0.81	0.96	0.91	−0.85	0.94	0.98	−0.95	0.99	−0.95	−0.96
Gl	0.98	0.99	0.99	−0.98	0.99	1.00	−0.96	1.00	−0.96	−0.96
Ir	1.00	1.00	1.00	−0.97	1.00	1.00	−0.97	1.00	−0.97	−0.97
Kn	0.98	0.99	0.99	−0.99	0.99	0.99	−0.98	1.00	−0.99	−0.99
Le	0.95	1.00	0.99	−0.92	0.95	0.98	−0.93	0.99	−0.93	−0.89
Ls	0.96	0.96	0.97	−0.68	0.99	1.00	−0.71	0.99	−0.75	−0.71
Ly	0.65	0.70	0.69	−0.93	0.98	0.98	−0.56	1.00	−0.59	−0.59
Pe	1.00	1.00	1.00	−1.00	1.00	1.00	−1.00	1.00	−1.00	−1.00
PE	0.98	0.99	0.98	−0.91	0.99	1.00	−0.90	1.00	−0.92	−0.92
Pi	0.91	0.94	0.94	−0.79	0.98	0.99	−0.77	1.00	−0.79	−0.79
Pr	0.98	0.99	0.99	−0.90	0.98	0.99	−0.87	1.00	−0.88	−0.88
Se	1.00	1.00	1.00	−0.95	1.00	1.00	−0.96	1.00	−0.95	−0.95
So	0.94	0.96	0.96	−0.95	0.98	0.99	−0.88	1.00	−0.89	−0.89
St	0.96	1.00	0.99	−0.98	0.96	0.97	−0.95	0.99	−0.98	−0.99
Zo	0.99	0.99	0.99	−0.82	1.00	1.00	−0.78	1.00	−0.79	−0.78

larger number of instances are considered together, we can expect a stronger (either negative or positive) correlation between Euclidean distance and other metrics such as precision and recall.

For all the instances in a dataset, we divide them into two equal partitions by selecting instances randomly. One partition is used to train a group of classifiers. Then we evaluate all the instances in the other partition by a few different metrics including Euclidean distance. Finally, the correlation between different metric pairs are calculated. The roles of training and testing are exchanged for the two partitions.

Two types of classifiers, decision tree-based (20) and support vector machine-based (20), were generated. To increase the diversity of those classifiers generated, different settings were tried. Rather than using all the features, we randomly selected a subset (2/3) of all the features. we also set different values for a few parameters including criterion and maximum deepth (for CART decision trees), and kernel and gamma (for SVMs).

Apart from Euclidean distance, Accuracy, Precision, Recall, and F1 are invovled. All the 26 datasets in Table 6 are also used in this experiment. For each dataset, we repeated

the above-mentioned process 20 times. Table 9 presents the Pearson correlation coefficients between each pair of those five metrics.

From Table 9, we can see that the correlation between all those pairs are very strong in most cases. In 12 datasets, the correlation coefficients are always above 0.9 for all different metric pairs. In a few datasets, the corrlelation is relatively weaker. The weakest is 0.56 (P &E in Ly). In general, it demonstrates that Euclidean distnace has strong correlation with all four metrics considered in the experiment. Therefore, we conclude that in general, the theorems we obtain from the geometric framework still make sense even when other metrics such as Accuracy is used for performance evaluation.

6 Conclusions

In this paper, we have presented a dataset-level geometric framework for ensemble classifiers. The most important advantage of the framework is it makes ensemble learning a deterministic problem. Euclidean distance has some good properties. One is its continuity. This makes it a good candidate as a target variable for optimization, such as using regression to deal with classification problems. Another property is it can measure both performance and dissimilarity, thus it is a good platform for us to understand the fundamental properties of ensembles clearly and investigate many issues in ensemble classifiers, such as the impact of multiple aspects on ensemble performance, predicting ensemble performance, selecting a small number of base classifiers to get efficient and effective ensembles, etc. Otherwise, it is very challenging to grasp even an incomplete picture. This is why up to now some of the properties of majority voting and weighted majority voting have not been fully understood.

Compared with the instance-level framework in Wu and Crestani (2015); Bonab and Can (2019), the dataset-level framework presented in this paper is a step forward. It maps the ensemble classifier problem for a whole dataset into one multi-dimensional space, thus it is more convenient for us to investigate the properties of ensembles. Otherwise, we have to deal with multiple spaces at the same time, each for one instance. To find out the collective properties in those spaces is more complicated. Based on the dataset-level framework, we have deduced some useful theorems which had not been found before.

An empirical investigation has also been conducted to see how those theorems in the geometric framework hold when Accuracy rather than the Euclidean distance is used for performance evaluation. The experimental results show that the theorems are still meaningful for other metrics.

In this geometric framework, we can present by a mathematical equation the exact relationship between ensemble performance and two major factors including all base classifiers' performance and diversity of the group, thus the profitability of ensembling a group of base classifiers can be calculated out very quickly. Ensemble pruning (Mohammed et al., 2022), which is to select a subset of component classifiers from all available ones for better performance and efficiency, has been investigated widely. This framework can be used for this task in different ways.

In this paper, the setting for the proposed framework is traditionally with a batch of training data. In recent years, data stream classification has attracted some attention (Gomes et al., 2017). How to adapt the geometric framework for this is worth further research. Especially, incorporating dynamic updates is a key point. Another research topic is multi-model data fusion (Gao et al., 2020). Again how to adapt the framework to support multi-modal data fusion is an interesting research issue. One possible solution is to use a

separate framework for each mode and then to combine them. These research issues remain to be our future work.

Author's contributions SW: theoretical part and paper writing. JL and WD: empirical part

Funding No funding was received for conducting this study.

Availability of data and material For the empirical investigation, the Python sklearn package and 26 datasets from the UCI Machine Learning Repository were used. They are publically available.

Code availability Our program is available on GitHub (https://github.com/MuxLZ/-Ensemble-Learning-experiment)

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Ethics approval The study is original and has not been submitted to any other journal/conference.

Consent to participate All three authors agree with the content and submission of the manuscript to this journal.

Consent for publication All three authors agree with the submission of the manuscript to this journal and possible publication afterwards.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Adnan, M. N., & Islam, M. Z. (2016). Optimizing the number of trees in a decision forest to discover a subforest with high ensemble accuracy using a genetic algorithm. *Knowledge-Based Systems, 110*, 86–97.

Bashir, S., Qamar, U., & Khan, F. H. (2015). Heterogeneous classifiers fusion for dynamic breast cancer diagnosis using weighted vote based ensemble. *Quality and Quantity, 49*, 2061–2076.

Bhardwaj, M., Bhatnagar, V., & Sharma, K. (2016). Cost-effectiveness of classification ensembles. *Pattern Recognition, 57*, 84–96.

Bi, Y. (2012). The impact of diversity on the accuracy of evidential classifier ensembles. *International Journal of Approximate Reasoning, 53*(4), 584–607.

Bonab, H.R., & Can, F. (2018). GOOWE: geometrically optimum and online-weighted ensemble classifier for evolving data streams. *ACM Transactions on Knowledge Discovery from Data*, *12*(2):25:1–25:33.

Bonab, H. R., & Can, F. (2019). Less is more: A comprehensive framework for the number of components of ensemble classifiers. *IEEE Transactions on Neural Networks and Learning Systems, 30*(9), 2735–2745.

Bousquet, O., Boucheron, S., & Lugosi, G. (2003). Introduction to statistical learning theory. Lecture Notes in Computer ScienceIn O. Bousquet, U. von Luxburg, & G. Rätsch (Eds.), *Advanced Lectures on Machine Learning, ML Summer Schools 2003, Canberra, Australia, February 2–14, 2003, Tübingen, Germany, August 4–16, 2003, Revised Lectures* (Vol. 3176, pp. 169–207). Springer.

Cao, J., Kwong, S., Wang, R., et al. (2015). Class-specific soft voting based multiple extreme learning machines ensemble. *Neurocomputing, 149*, 275–284.

Caruana, R., Niculescu-Mizil, A., Crew, G., et al. (2004). Ensemble selection from libraries of models. In: Brodley CE (ed) *Proceedings of the Twenty-first International Conference on Machine Learning* (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004, ACM International Conference Proceeding Series, vol 69. ACM.

Derbeko, P., El-Yaniv, R., & Meir, R. (2002). Variance optimized bagging. Lecture Notes in Computer ScienceIn T. Elomaa, H. Mannila, & H. Toivonen (Eds.), *Machine Learning: ECML 2002, 13th European Conference on Machine Learning, Helsinki, Finland, August 19–23, 2002, Proceedings* (Vol. 2430, pp. 60–71). Springer.

Dias, K., & Windeatt. T. (2014). Dynamic ensemble selection and instantaneous pruning for regression used in signal calibration. In: Wermter S, Weber C, Duch W, et al (eds) *Artificial Neural Networks and Machine Learning—ICANN 2014—24th International Conference on Artificial Neural Networks*, Hamburg, Germany, September 15–19, 2014. Proceedings, Lecture Notes in Computer Science, vol 8681. Springer, pp 475–482.

Dong, X., Yu, Z., Cao, W., et al. (2020). A survey on ensemble learning. *Frontiers of Computer Science, 14*(2), 241–258.

Duan, Q., Ajami, N. K., Gao, X., et al. (2007). Multi-model ensemble hydrologic prediction using bayesian model averaging. *Advances in Water Resources, 30*(5), 1371–1386.

Elisseeff, A. (2000). A study about algorithmic stability and their relation to generalization performances. Tech. rep., Laboratoire ERIC - Université Lyon 2, 5 Avenue Pierre Mendès France, 69676 BRON Cedex.

Fumera, G., Roli, F., & Serrau, A. (2008). A theoretical analysis of bagging as a linear combination of classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 30*(7), 1293–1299.

Gao, J., Li, P., Chen, Z., et al. (2020). A survey on deep learning for multimodal data fusion. *Neural Computation, 32*(5), 829–864.

Georgiou, H.V., Mavroforakis, M.E., & Theodoridis, S. (2006). A game-theoretic approach to weighted majority voting for combining SVM classifiers. In: Kollias, S. D., Stafylopatis, A., Duch, W., et al (eds) *Artificial Neural Networks—ICANN 2006, 16th International Conference, Athens, Greece*, September 10–14, 2006. Proceedings, Part I, Lecture Notes in Computer Science, vol 4131. Springer, pp 284–292.

Gomes, H.M., Barddal, J.P., Enembreck, F., et al. (2017). A survey on ensemble learning for data stream classification. *ACM Computing Surveys*, *50*(2):23:1–23:36.

Haque, M.N., Noman, N., Berretta, R., et al, (2016). Optimising weights for heterogeneous ensemble of classifiers with differential evolution. In: *IEEE Congress on Evolutionary Computation*, CEC 2016, Vancouver, BC, Canada, July 24–29, 2016. IEEE, pp 233–240.

Hernández-Lobato, D., Martínez-Muñoz, G., & Suárez, A. (2013). How large should ensembles of classifiers be? *Pattern Recogniton, 46*(5), 1323–1336.

Jain, S., Liu, G., Mueller, J., et al. (2020). Maximizing overall diversity for improved uncertainty estimates in deep ensembles. In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA*, February 7–12, 2020. AAAI Press, pp. 4264–4271.

Jurek, A., Bi, Y., Wu, S., et al. (2014). A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering Review, 29*(5), 551–581.

Kuncheva, L. I., & Diez, J. J. R. (2014). A weighted voting framework for classifiers ensembles. *Knowledge and Information Systems, 38*(2), 259–275.

Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning, 51*(2), 181–207.

Latinne, P., Debeir, O., & Decaestecker, C. (2001). Limiting the number of trees in random forests. Lecture Notes in Computer Science. In: Kittler, J. & Roli , F.(Eds.), *Multiple Classifier Systems, Second International Workshop, MCS 2001 Cambridge, UK, July 2–4, 2001, Proceedings* (Vol. 2096, pp. 178–187). Springer.

Liu, N., Cao, J., Lin, Z., et al. (2014). Evolutionary voting-based extreme learning machines. *Mathematical Problems in Engineering* 2014.

Mao, S., Jiao, L., Xiong, L., et al. (2015). Weighted classifier ensemble based on quadratic form. *Pattern Recognition, 48*(5), 1688–1706.

Minkowski, (2020). http://mathworld.wolfram.com/minkowskisinequalities.html.

Mohammed, A. M., Onieva, E., Wozniak, M., et al. (2022). An analysis of heuristic metrics for classifier ensemble pruning based on ordered aggregation. *Pattern Recognition, 124*(108), 493.

Nguyen, T. T., Dang, M. T., Liew, A. W., et al. (2019). A weighted multiple classifier framework based on random projection. *Information science, 490*, 36–58.

Opitz, D. W., & Shavlik, J. W. (1996). Actively searching for an effective neural network ensemble. *Connect Science, 8*(3), 337–354.

Oshiro, T.M., Perez, P.S., & Baranauskas, J.A. (2012). How many trees in a random forest? In: Perner, P. (ed) *Machine Learning and Data Mining in Pattern Recognition—8th International Conference, MLDM 2012, Berlin, Germany*, July 13–20, 2012. Proceedings, Lecture Notes in Computer Science, vol 7376. Springer, pp 154–168.

Oza, N. C., & Tumer, K. (2008). Classifier ensembles: Select real-world applications. *Information Fusion, 9*(1), 4–20.

Probst, P., & Boulesteix, A. (2017). To tune or not to tune the number of trees in random forest. *Journal of Machine Learning Research* 18:181:1–181:18

Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8(4).

Schapire, R. E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning, 37*(3), 297–336.

Seewald, A.K. (2002). How to make stacking better and faster while also taking care of an unknown weakness. In: Sammut, C., Hoffmann, A. G. (eds) *Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002), University of New South Wales, Sydney, Australia*, July 8–12, 2002. Morgan Kaufmann, pp 554–561.

Sen, M. U., & Erdogan, H. (2013). Linear classifier combination and selection using group sparse regularization and hinge loss. *Pattern Recognition Letters, 34*(3), 265–274.

Tang, E. K., Suganthan, P. N., & Yao, X. (2006). An analysis of diversity measures. *Machine Learning, 65*(1), 247–271.

Ting, K. M., & Witten, I. H. (1999). Issues in stacked generalization. *Journal of Artificial Intelligence Research, 10*, 271–289.

Valdovinos, R.M., & Sánchez, J.S. (2009). Combining multiple classifiers with dynamic weighted voting. In: Corchado, E., Wu, X., Oja, E., et al (eds) *Hybrid Artificial Intelligence Systems, 4th International Conference, HAIS 2009, Salamanca, Spain*, June 10–12, 2009. Proceedings, Lecture Notes in Computer Science, vol 5572. Springer, pp 510–516.

Visentini, I., Snidaro, L., & Foresti, G. L. (2016). Diversity-aware classifier ensemble selection via f-score. *Information Fusion, 28*, 24–43.

Wang, Y., Hao, J., Glover, F. W., et al. (2014). A tabu search based memetic algorithm for the maximum diversity problem. *Engineering Applications of Artificial Intelligence, 27*, 103–114.

Wozniak, M. (2008). Classifier fusion based on weighted voting—analytical and experimental results. In: Pan, J., Abraham, A., Chang, C. (eds) *Eighth International Conference on Intelligent Systems Design and Applications*, ISDA 2008, 26–28 November 2008, Kaohsiung, Taiwan, 3 Volumes. IEEE Computer Society, pp 687–692.

Wu, S., & Crestani, F. (2015). A geometric framework for data fusion in information retrieval. *Information Systems, 50*, 20–35.

Xiao, J., He, C., Jiang, X., et al. (2010). A dynamic classifier ensemble selection approach for noise data. *Information Science, 180*(18), 3402–3421.

Yang, J., Zeng, X., Zhong, S., et al. (2013). Effective neural network ensemble approach for improving generalization performance. *IEEE Transactions on Neural Networks and Learning Systems, 24*(6), 878–887.

Ykhlef, H., & Bouchaffra, D. (2017). An efficient ensemble pruning approach based on simple coalitional games. *Information Fusion, 34*, 28–42.

Zhang, L., & Zhou, W. (2011). Sparse ensembles using weighted combination methods based on linear programming. *Pattern Recognition, 44*(1), 97–106.

Zhang, W., Jiang, J., Shao, Y., et al. (2020). Efficient diversity-driven ensemble for deep neural networks. In: *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA*, April 20–24, 2020. IEEE, pp 73–84.

Zhou, Z., Wu, J., & Tang, W. (2002). Ensembling neural networks: Many could be better than all. *Artificial Intelligence, 137*(1–2), 239–263.

Zhu, X., Ni, Z., Ni, L., et al. (2019). Improved discrete artificial fish swarm algorithm combined with margin distance minimization for ensemble pruning. *Computers and Industrial Engineering, 128*, 32–46.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.