Shapley-Based Data Valuation for Weighted k-Nearest Neighbors

Guangyi Zhang
Shenzhen Technology University
zhangguangyi@sztu.edu.cn

Qiyu Liu*
Southwest University
qyliu.cs@gmail.com

Aristides Gionis*
KTH Royal Institute of Technology and Digital Futures
argioni@kth.se

Abstract

Data valuation quantifies the impact of individual data points on model performance, and Shapley values provide a principled approach to this important task due to their desirable axiomatic properties, albeit with high computational complexity. Recent breakthroughs have enabled fast computation of exact Shapley values for unweighted k-nearest neighbor (kNN) classifiers. However, extending this to weighted kNN models has remained a significant open challenge. The state-of-the-art methods either require quadratic time complexity or resort to approximation via sampling. In this paper, we show that a conceptually simple but overlooked approach — data duplication — can be applied to this problem, yielding a natural variant of weighted kNN-Shapley. However, a straightforward application of the data-duplication idea leads to increased data size and prohibitive computational and memory costs. We develop an efficient algorithm that avoids materializing the duplicated dataset by exploiting the structural properties of weighted kNN models, reducing the complexity to near-linear time in the original data size. Besides, we establish theoretical foundations for this approach through axiomatic characterization of the resulting values, and empirically validate the effectiveness and efficiency of our method.

1 Introduction

In the era of data-driven machine learning, understanding the value and contribution of individual data points has emerged as a critical challenge. Data valuation—the process of quantifying the impact of each training instance on model performance—plays a pivotal role in numerous applications, including identifying influential samples, detecting mislabeled data, and designing data markets [1, 2]. As machine-learning systems continue to proliferate across domains, the development of robust and scalable data-valuation methods has become increasingly essential.

Among various approaches for data valuation [3, 4, 5], the Shapley value from cooperative game theory has attracted significant attention due to its unique desirable axiomatic properties [6]. When applied to machine learning, data Shapley values provide a principled framework for distributing the model's performance among training instances, where each data point's Shapley value represents its average marginal contribution across all possible subsets of the dataset [1].

Despite its theoretical appeal and desirable properties, the exact computation of data Shapley values presents substantial challenges, as it requires enumerating and re-training over all possible subsets

*Corresponding authors

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

of data points—a task whose complexity grows exponentially with the dataset size. Indeed, the computation has been proven to be $\#\mathbf{P}$-hard in certain games [7]. Fortunately, a notable recent advancement in this domain by Jia et al. [8] leverages the structural properties of unweighted k-nearest neighbor (kNN) classifiers to efficiently calculate exact data Shapley values in closed form, referred to as unweighted kNN-SV hereafter. The kNN approach is a classic algorithm in machine learning, and it has been shown that running kNN over pre-trained embeddings can yield comparable performance with more advanced models [9].

However, a significant limitation of the approach by Jia et al. [8] lies in its restriction to unweighted kNN models. In practice, weighted kNN models, which assign different importance to neighbors based on their distances, offer greater flexibility and typically yield superior performance. Computing Shapley values for weighted kNN models poses unique challenges due to the lack of a closed-form solution analogous to the unweighted case. To date, the only successful attempt has been a hard-label variant of weighted kNN-SV [10], which invokes a sophisticated and time-consuming dynamic-programming approach that is similar in spirit to those for power indexes [11].

In this paper, we offer a novel variant of weighted kNN-SV, dubbed as DkNN-SV for duplication-based weighted kNN-SV, whose underlying classifier is equivalent to the standard weighted kNN classifier. We demonstrate that this weighted variant can be effectively reduced to the unweighted case by a duplication technique. Since naïve duplication could significantly increase the dataset size and computational burden, we further develop an efficient algorithm that exploits the structural properties of the weighted kNN models, thereby eliminating the additional computational costs associated with data duplication. In other words, the data duplication has been made *conceptual* and does not need to be materialized. Our algorithm runs in near-linear time $\mathcal{O}(n \log n)$, while the state-of-the-art methods either require a pseudo-polynomial time complexity of $\mathcal{O}(Wk^2n^2)$ [10] or settle for approximation by sampling from $n!$ permutations [8], where n is the dataset size and W is the maximum weight. We analyze the theoretical properties of the proposed DkNN-SV scheme, and empirically validate its effectiveness and efficiency against other kNN-SV variants. For example, our algorithm performs better at the task of noisy label detection.

The rest of the paper is organized as follows. First, we review the preliminaries in Section 2 and the unweighted kNN-SV in Section 3. Next, we introduce the new weighted DkNN-SV in Section 4, and develop an efficient algorithm for it in Section 5. We discuss the related work in Section 6. Finally, we empirically evaluate the proposed method in Section 7. We conclude in Section 8. All proofs are deferred to Appendix A.

2 Preliminaries

In this section, we review the framework for data valuation based on Shapley values (SV) and establish our notation for applying this concept to the k-nearest neighbor (kNN) model.

2.1 Cooperative game theory and Shapley values

The concept of data valuation can be elegantly formalized through the lens of cooperative game theory [12]. In this framework, we consider a collection of players who form coalitions to generate collective utility. Formally, a cooperative game consists of a pair (I, util), where $I = \{1, \ldots, n\}$ represents a set of n players and $\text{util} : 2^I \to \mathbb{R}$ is a utility function that assigns a real value to each possible coalition $S \subseteq I$.

A central question in cooperative game theory concerns fair allocation: how should the total utility be distributed among individual players based on their contributions? The Shapley value, introduced by Lloyd Shapley [6], provides a rigorous solution to this problem. The Shapley value $s(i)$ for player i represents their expected marginal contribution when joining the coalition in a random order. Let Π denote the set of all permutations of players in I. For a permutation $\pi \in \Pi$, let π_i represent the set of players that precede player i in π. Then, the Shapley value equals

$$s(i) = \frac{1}{n!} \sum_{\pi \in \Pi} \left[\text{util}(\pi_i \cup \{i\}) - \text{util}(\pi_i) \right]. \tag{1}$$

An equivalent expression represents the Shapley value as the average marginal contribution across all possible coalition formations, that is,

$$s(i) = \frac{1}{n} \sum_{S \subseteq I \setminus \{i\}} \binom{n-1}{|S|}^{-1} \left[\text{util}(S \cup \{i\}) - \text{util}(S)\right]. \tag{2}$$

It is well known that the Shapley value is the *unique* allocation mechanism that satisfies the following desirable properties: *efficiency*, *symmetry*, *null player*, and *additivity* [6]. These properties make the Shapley values a principled approach for data valuation, relying on an axiomatic framework.

To apply the Shapley-value framework to the data-valuation setting, we interpret individual data points as players in a coalition game, and a performance measure of a model as a utility function. The value of the utility function for a coalition of players corresponds to the performance measure of the model trained on the respective subset of data. This framework allows for quantifying the contribution of each training data point to the overall model performance.

2.2 kNN-based Shapley values

In this section we discuss how the Shapley-value framework described above can be used for data valuation with a kNN model [8]. We refer to this method as kNN-SV. The idea is first presented for a single test data point, and then is extended to multiple test data points.

Consider a dataset D with n training data points, where each point $z = (x, y) \in D$ comprises $x \in \mathbb{R}^d$ and $y \in \mathcal{Y}$, where $\mathcal{Y}$ is the label space. Consider also a single test data point $z_{\text{test}} = (x_{\text{test}}, y_{\text{test}})$. We want to compute the Shapley value $s(z \mid z_{\text{test}})$ of each training point $z \in D$ with respect to the test point z_{test}. For each point z we consider a rational weight $w(z \mid z_{\text{test}}) \in \mathbb{Q}_+$, often determined by the distance between x and x_{test} (see discussion below). For a subset $S \subseteq D$ of the training data, the kNN utility of S is defined as

$$\text{util}(S) = \begin{cases} \frac{\sum_{i=1}^{\min\{|S|,k\}} w(z_{\alpha_i(S)} | z_{\text{test}})\, \mathbb{1}(y_{\alpha_i(S)} = y_{\text{test}})}{\sum_{i=1}^{\min\{|S|,k\}} w(z_{\alpha_i(S)} | z_{\text{test}})}, & \text{if } |S| > 0 \\ \frac{1}{C}, & \text{if } |S| = 0, \end{cases} \tag{3}$$

where $C = |\mathcal{Y}|$ is the number of classes, $\alpha_i(S)$ is the index of the i-th closest point of S to x_{test}, and i is the *rank* of $z_{\alpha_i(S)}$ in S.

For defining the weight $w(z \mid z_{\text{test}})$, in the case of unweighted kNN, we simply set $w(z \mid z_{\text{test}}) = 1$. More generally, by using the distance $\text{dist}(z, z') = \|x - x'\|$, we can define the weight $w(z \mid z_{\text{test}})$ as the Gaussian kernel $w(z \mid z_{\text{test}}) = \mathcal{K}(\text{dist}(z, z_{\text{test}})) = \exp(-\text{dist}(z, z_{\text{test}})^2 / 2\sigma^2)$, where σ measures the *width* of the kernel. Clearly, any other distance metric can be used.

Having defined the utility function for subsets $S \subseteq D$, the Shapley value of a data point $z \in D$ can be computed using Eq. (2). Formally, the Shapley value of a training data point $z \in D$ concerning a single test data point z_{test} is defined as

$$s(z \mid z_{\text{test}}) = \frac{1}{n} \sum_{S \subseteq D \setminus \{z\}} \binom{n-1}{|S|}^{-1} \left[\text{util}(S \cup \{z\}) - \text{util}(S)\right]. \tag{4}$$

In practice, when given multiple test data points, the data Shapley value of a data point z can be straightforwardly extended as the average over all test points. That is, when given a test dataset D_{test} of n_{test} data points, we have

$$s(z) = \frac{1}{n_{\text{test}}} \sum_{z_{\text{test}} \in D_{\text{test}}} s(z \mid z_{\text{test}}). \tag{5}$$

3 Exact computation for unweighted kNN-SV

In this section, we introduce the analytical solution to unweighted kNN-SV, introduced by Jia et al. [8] and a subsequent note [13]. This solution results in a dramatic improvement in the time complexity for exact computation, from $\mathcal{O}(2^n)$ to $\mathcal{O}(dn + n \log n)$ for a single test point, over the naïve approach that enumerates all possible subsets S of the dataset D.

We fix a test point z_{test} throughout this section. Given a subset $S \subseteq D$, recall that $\alpha_i(S)$ is the index of the i-th closest element of S to z_{test}. When the context is clear, we write $z_{\alpha_i(D)}$ as z_i and $w(z_{\alpha_i(D)} \mid z_{\text{test}})$ as w_i for simplicity. Similarly, we denote by s_i the Shapley value $s(z_i \mid z_{\text{test}})$ for data point z_i.

We restate the main result from Wang and Jia [13] for unweighted kNN-SV. For ease of exposition, we assume $n \geq 2$ and $n \geq k$ throughout the paper.

Theorem 1 (Wang and Jia [13])**.** *Assume $n \geq 2$ and $n \geq k$. For the unweighted kNN classifier, the Shapley value of a data point z_i can be computed as follows.*

$$s_n = \frac{1}{n}\left(\mathbb{1}_n - \sum_{i=1}^{n-1} \frac{\mathbb{1}_i}{n-1}\right)\left(\sum_{j=1}^{k-1} \frac{1}{j+1}\right) + \frac{\mathbb{1}_n - C^{-1}}{n}, \tag{6}$$

where $\mathbb{1}_i = \mathbb{1}[y_i = y_{test}]$, and for $i < n$,

$$s_i = s_{i+1} + \frac{\mathbb{1}_i - \mathbb{1}_{i+1}}{n-1}\left(\sum_{j=1}^{k} \frac{1}{j} + \frac{1}{k}\left(\frac{\min\{k,i\}(n-1)}{i} - k\right)\right). \tag{7}$$

Following Theorem 1, we can compute the Shapley values by first sorting the data points by increasing distance to z_{test}, and then iteratively computing the Shapley values in the reverse order, as specified by Eq. (6) and Eq. (7).

However, extending the previous result to the weighted kNN is challenging. Given a subset S, we refer to the quantity $\text{util}(S \cup \{z_i\}) - \text{util}(S)$ as the *marginal contribution* (MC) of z_i to S. The key principle behind the result in Theorem 1 is that for unweighted kNN, the MC can only take a few distinct values, which turns kNN-SV problem into a combinatorial counting problem. For example, for any $|S| \geq k$, we have

$$\text{util}(S \cup \{z_i\}) = \text{util}(S) \quad \text{or} \quad \text{util}(S \cup \{z_i\}) - \text{util}(S) = \frac{1}{k}(\mathbb{1}[y_i = y_{\text{test}}] - \mathbb{1}[y_{\alpha_k(S)} = y_{\text{test}}]),$$

the latter of which only takes three possible distinct values. Thus, one can simply count the number of subsets S for each distinct MC value, and aggregate the results, without evaluating the MC for all possible subsets. However, for weighted kNN, as in Eq. (3), the MC may take arbitrary values due to the weights and the normalization term.

4 Weighted kNN-SV via data duplication

In this section, we show how to overcome the challenge in the weighted kNN-SV problem. We do so by introducing a new variant and reducing it to the unweighted case using a novel data-duplication technique. We relate the new variant to the concept of *Owen values* [14], and prove adherence to axiomatic properties. We further compare it with other kNN-SV variants analytically in Section 4.2.

Our main idea is to create $w(z \mid z_{\text{test}}) - 1$ copies of every point $z \in D$. For ease of exposition, we assume that all weights are integers. This assumption can be easily removed by scaling the weights appropriately. Let the new dataset containing D and its copies be D'. We will then apply the unweighted kNN classifier to D' with parameter k'.

To obtain an equivalent weighted kNN classifier, we adaptively set k' for each test point z_{test} [15]. Specifically, we adjust k' so that it includes the top-k nearest neighbors of z_{test} in D and their copies. Formally, we set

$$k' = \sum_{i=1}^{k} w(z_{\alpha_i(D)} \mid z_{\text{test}}). \tag{8}$$

We show that this leads to an equivalent weighted kNN classifier.

Proposition 2. *Running unweighted kNN classifier on a duplicated dataset D' with parameter k' defined in Eq.* (8) *is equivalent to running weighted kNN classifier on the original dataset D for any test point z_{test}.*

Fixing a test point z_{test}, we define the utility of unweighted kNN on D' for any subset S' of D' as

$$\text{util}'(S') = \begin{cases} \frac{\sum_{i=1}^{\min\{|S'|,k'\}} \mathbb{1}(y_{\alpha_i(S')}=y_{\text{test}})}{\min\{|S'|,k'\}}, & \text{if } |S'|>0 \\ \frac{1}{C}, & \text{if } |S'|=0. \end{cases} \tag{9}$$

The Shapley value of a data point $z \in D'$ concerning a given test point z_{test} is then simply

$$s'(z \mid z_{\text{test}}) = \frac{1}{n'} \sum_{S' \subseteq D' \setminus \{z\}} \binom{n'-1}{|S'|}^{-1} \left[\text{util}'(S' \cup \{z\}) - \text{util}'(S')\right]. \tag{10}$$

The DkNN-SV of a data point $z \in D$ is the sum of the Shapley values of all its copies in D', i.e.,

$$\phi(z \mid z_{\text{test}}) = \sum_{z' \in D' : z'=z} s'(z' \mid z_{\text{test}}) = w(z \mid z_{\text{test}}) s'(z \mid z_{\text{test}}). \tag{11}$$

Note that duplicating data points according to their weights may result in a dataset D' much larger than D, which may cause significant scalability issues. We will address this challenge in Section 5 by proposing an efficient algorithm that avoids materializing D'.

4.1 Duplication-based weighted kNN-SV as group values

An alternative way to interpret the DkNN-SV $\phi(z \mid z_{\text{test}})$ is to view a data point z and its copies as a "group" with z being the representative of the group, and $\phi(z \mid z_{\text{test}})$ measuring the value of the whole group. This is similar to the classic concept of *Owen values* [14], which characterizes the value of players in a game with *coalition structure*.

Formally, to compute the Owen values, we are given non-overlapping groups $\mathcal{G} = \{G_1, \ldots, G_m\}$ such that $\cup_{i=1}^m G_i = D'$ and $G_i \cap G_j = \emptyset$ for all $i \neq j$. Owen values are calculated by considering permutations of players that are compatible with the group structure. A permutation π of the player set D' is called *group-compatible* if players within the same group G_i appear consecutively in the permutation. The Owen value of a player z' is its average marginal contribution over all group-compatible permutations, computed in a way similar to Eq. (1). The only difference is that the Owen value sums over all group-compatible permutations instead of all permutations. The Owen value of a group G_i is then the sum of the Owen values of all players $z' \in G_i$.

In our setting, each group is a set of copies of a single point. The main difference is that we do not require all players in a group to always act as one unit, and thus do not require the permutations to be group-compatible. Instead, we allow each player (data point) to act independently in the game, and eventually aggregate their contributions to obtain the group value in the same manner. This relaxation allows us to design an efficient algorithm. Our approach can also be justified axiomatically. These axiomatic properties ensure that data valuation is conducted in a fair, theoretically sound, and interpretable manner for downstream tasks. Moreover, additional properties about symmetry between z and its copies enable us to devise a fast algorithm without materializing D'; see Section 5.

Theorem 3. *The group value $\phi(z \mid z_{test})$ defined in Eq. (11) satisfies the following axioms:*

1. ***Efficiency:*** *$\sum_{z \in D} \phi(z \mid z_{test}) = util'(D') - util'(\emptyset) = util(D) - util(\emptyset)$, that is, the sum of all group values equals the total utility.*
2. ***Symmetry:*** *If z_1 and z_2 are such that $util'(S' \cup \{z_1'\}) = util'(S' \cup \{z_2'\})$ for all $S' \subseteq D' \setminus \{z_1', z_2'\}$ where z_1' and z_2' are copies of z_1 and z_2 respectively, then $\phi(z_1 \mid z_{test})/w(z_1 \mid z_{test}) = \phi(z_2 \mid z_{test})/w(z_2 \mid z_{test})$.*
3. ***Dummy Player:*** *If z is such that $util'(S' \cup \{z'\}) = util'(S')$ for all $S' \subseteq D' \setminus \{z'\}$ where z' is a copy of z, then $\phi(z \mid z_{test}) = 0$.*
4. ***Additivity:*** *If $util_1'$ and $util_2'$ are two utility functions, and ϕ_1 and ϕ_2 are their corresponding values, then the value ϕ corresponding to $util_1' + util_2'$ satisfies $\phi = \phi_1 + \phi_2$.*

Theorem 3 can be easily extended to more general settings, where each group is formed by an arbitrary subset of players instead of a set of copies.

4.2 Comparison with other kNN-SV

Comparison with standard weighted kNN-SV. Although Proposition 2 shows that the duplication strategy provides an equivalent weighted kNN classifier, it may not produce the same Shapley values. This is because we need equivalent classification over all possible subsets instead of merely the entire dataset D. More generally, we show that their values remain different regardless of the value of parameter k'. We give a counterexample below. The intuition is that the effect of increasing the weight of a point and increasing its number of copies on its value is similar over most subsets, but may differ over certain subsets.

Proposition 4. *In the genral case, the group Shapley values in Eq.* (11) *may be different from the Shapley values of weighted* kNN-SV *in Eq.* (4)*.*

Hence, our duplication strategy offers a new variant for computing data Shapley values for weighted kNN-SV. Due to the lack of a closed-form derivation for the vanilla weighted kNN-SV, we compare the different methods empirically in the experiments, and show that they are correlated, however, the values in the proposed approach can be computed more efficiently.

Comparison with the scaled unweighted kNN-SV. One might also wonder how the group values $\phi(z \mid z_{\text{test}})$ compare to simply scaling the unweighted kNN-SV values $s(z \mid z_{\text{test}})$ in the original dataset D. A natural scaling approach might be $s(z \mid z_{\text{test}}) \cdot w(z \mid z_{\text{test}})$. However, such scaling is fundamentally different from our duplication strategy, and fails to capture the inherent weighted nature of the weighted kNN-SV, as illustrated in the analysis below.

Given an arbitrary test point z_{test}, we first examine the scaled value $\tilde{s}_n$ of the farthest point z_n.

$$\tilde{s}_n = s_n w_n \frac{n}{n'} = \frac{w_n}{n'} \left(\mathbb{1}_n - \sum_{i=1}^{n-1} \frac{\mathbb{1}_i}{n-1} \right) \left(\sum_{j=1}^{k-1} \frac{1}{j+1} \right) + w_n \frac{\mathbb{1}_n - \frac{1}{C}}{n'},$$

where $w_n = w(z_n \mid z_{\text{test}})$ and $\mathbb{1}_i = \mathbb{1}(y_i = y_{\text{test}})$. To simplify the analysis, we set k' to be the same as k. If we compare the scaled value $\tilde{s}_n$ with the group value $\phi(z_n \mid z_{\text{test}})$, we have

$$\tilde{s}_n - \phi(z_n \mid z_{\text{test}}) = \frac{w_n}{n'} \left(\left[\mathbb{1}_n - \underbrace{\sum_{i=1}^{n-1} \frac{\mathbb{1}_i}{n-1}}_{\text{unweighted avg}} \right] - \left[\mathbb{1}_n - \underbrace{\frac{(w_n - 1)\mathbb{1}_n + \sum_{i=1}^{n-1} w_i \mathbb{1}_i}{n' - 1}}_{\text{weighted avg}} \right] \right) \left(\sum_{j=1}^{k-1} \frac{1}{j+1} \right).$$

Thus, the difference is driven by the choice of the reference mean used in the comparison. Define unweighted and weighted reference means as

$$\mu_{\text{unw}} := \sum_{i=1}^{n-1} \frac{\mathbb{1}_i}{n-1}, \qquad \text{and} \qquad \mu_{\text{w}} := \frac{(w_n - 1)\mathbb{1}_n + \sum_{i=1}^{n-1} w_i \mathbb{1}_i}{n' - 1}.$$

Then the scaled value $\tilde{s}_n$ uses the deviation $\mathbb{1}_n - \mu_{\text{unw}}$, which measures the contribution of z_n by the deviation of its label from this unweighted reference mean, whereas the group value considers the deviation from the weighted reference mean by using $\mathbb{1}_n - \mu_{\text{w}}$.

Moving on to the scaled value of the i-th closest point z_i, we have

$$\begin{aligned}
\tilde{s}_i :&= s_i w_i \frac{n}{n'} \\
&= s_{i+1} w_i \frac{n}{n'} + w_i \frac{n}{n-1} \frac{\mathbb{1}_i - \mathbb{1}_{i+1}}{n'} \left(\sum_{j=1}^{k} \frac{1}{j} + \frac{1}{k} \left(\frac{\min\{k,i\}(n-1)}{i} - k \right) \right) \\
&\approx s_{i+1} w_i \frac{n}{n'} + w_i \frac{\mathbb{1}_i - \mathbb{1}_{i+1}}{n'-1} \left(\sum_{j=1}^{k} \frac{1}{j} + \frac{1}{k} \left(\frac{\min\{k,i\}(n-1)}{i} - k \right) \right),
\end{aligned}$$

where the last approximation follows by taking $\frac{n-1}{n} \frac{n'}{n'-1} \approx 1$. Then, if we compare the scaled value $\tilde{s}_i$ with the group value $\phi(z_i \mid z_{\text{test}})$, we have

$$\tilde{s}_i - \phi(z_i \mid z_{\text{test}})$$

Algorithm 1: Fast algorithm for duplicate-based weighted kNN-SV

Input: Integer k, weight function w, datasets D and D_{test}

Initialize ϕ_z with default value 0 for every $z \in D$

for $z_{test} \in D_{test}$ **do**

  Let $z_1, \ldots, z_n$ be the points in D sorted by increasing distance to z_{test}

  $n' \leftarrow \sum_{i=1}^{n} w(z_i \mid z_{test})$

  $k' = \sum_{i=1}^{k} w(z_i \mid z_{test})$

  $s'_{z_n} \leftarrow \frac{1}{n'}\left(\mathbb{1}_n - \frac{(w_n-1)\mathbb{1}_n + \sum_{i=1}^{n-1} w_i \mathbb{1}_i}{n'-1}\right)\left(\sum_{j=1}^{k'-1} \frac{1}{j+1}\right) + \frac{\mathbb{1}_n - 1/C}{n'}$, where $w_i = w(z_i \mid z_{test})$ and $\mathbb{1}_i = \mathbb{1}[y_i = y_{test}]$

  for $i = n-1, \ldots, 1$ **do**

    $i' \leftarrow \sum_{j=1}^{i} w(z_j \mid z_{test})$

    $s'_{z_i} \leftarrow s'_{z_{i+1}} + \frac{\mathbb{1}_i - \mathbb{1}_{i+1}}{n'-1}\left(\sum_{j=1}^{k'} \frac{1}{j} + \frac{1}{k'}\left(\frac{\min(k', i')(n'-1)}{i'} - k'\right)\right)$

  for $i = n, \ldots, 1$ **do**

    $\phi_{z_i} \leftarrow \phi_{z_i} + w(z_i \mid z_{test}) s'_{z_i}$

for $z \in D$ **do**

  $\phi_z \leftarrow \phi_z / n_{test}$

Return values $\{\phi_z\}_{z \in D}$

$$\approx w_i \left(\frac{s_{i+1} n}{n'} - \frac{\phi(z_{i+1})}{w_{i+1}}\right) + w_i \frac{\mathbb{1}_i - \mathbb{1}_{i+1}}{n'-1} \frac{1}{k} \left(\frac{\min\{k,i\}(n-1)}{i} - \frac{\min\{k,i'\}(n'-1)}{i'}\right).$$

First, the difference between these two values accumulates from the previous $(i+1)$-th point. In addition, the contribution from the i-th point to the scaled value $\tilde{s}_i$ is roughly weighted by $\frac{w_i(n-1)}{i(n'-1)}$, while the contribution to the group value $\phi(z_i \mid z_{\text{test}})$ is roughly weighted by $\frac{w_i}{i'}$. That is, the former scales up every rank i uniformly by a factor of $\frac{n'-1}{n-1}$, while the latter uses the rank i' of the i-th point in the duplicated dataset D'. Therefore, the scaled value tends to amplify the contribution of nearest neighbors with a large weight more than the group value.

5 Fast algorithm for duplication-based weighted kNN-SV

Recall that we duplicate data points in D according to their weights to form a new dataset D'. However, the size of D' may be much larger than that of D. This poses a significant challenge to the computational cost of the recursive computation of the Shapley values. Specifically, it requires $\mathcal{O}(n' \log n')$ time for each given test point, where n' is the size of D'. In this section, we propose a fast algorithm for the duplication-based weighted kNN-SV, with a time complexity as small as $\mathcal{O}(n \log n)$. The algorithm successfully obtains the Shapley values while avoiding materializing the duplicated dataset D'.

The proposed algorithm is displayed in Algorithm 1. The key idea is to leverage the *symmetry* property of the Shapley values to avoid materializing the duplicated data points.

Lemma 5. *Fix a test point z_{test}. Let z be a point in D. Then, every copy z' of z has the same Shapley value $s'(z' \mid z_{test})$ as that of z. This continues to hold when the values are obtained by applying the recursive formula in Theorem 1 with an arbitrary order among z and its copies.*

Therefore, when applying the recursive formula in Theorem 1, we only process every point z in D once, and skip its copies by directly multiplying the Shapley value by the number of its copies. This is valid as a result of Lemma 5; imagine we are taking an order where z is behind all its copies and processed first.

Note that the partial sum of the harmonic series can be accurately approximated by the Euler-Maclaurin formula [16], or retrieved from a pre-computed table. Asymptotically, the Euler-Maclaurin

formula [16] $F(j)$ below converges to the harmonic series $H_j = \sum_{i=1}^{j} \frac{1}{i}$ as $j \to \infty$.

$$F(j) = \ln(j) + \gamma + \frac{1}{2j} - \frac{1}{12j^2},$$

where $\gamma \approx 0.5772156649$ is the Euler-Mascheroni constant. In practice, we adopt the following more accurate approximation scheme. When j is a small constant (e.g., $j \leq 1000$), we can afford to compute H_j exactly. Otherwise, we use $F(j) - F(1000) + H_{1000}$ as an approximation (in $O(1)$ time where H_{1000} is pre-computed) to eliminate the noticeable error when j is relatively small. We verify that the absolute error is at most 4.167×10^{-8} for j up to 1 million and continues to decrease as j increases. We believe this approximation is sufficient for our purpose.

Hence, the total time complexity is dominated by the sorting step, which is $\mathcal{O}(n \log n)$. In addition, distance computations cost $\mathcal{O}(dn)$ time. We summarize the results in the following theorem.

Theorem 6. *Algorithm 1 computes duplication-based weighted kNN-SV in $\mathcal{O}(dn + n \log n)$ time.*

6 Related work

Data Valuation. Data valuation aims to assign importance scores to training examples [3, 4, 5]. The dominant approaches are based on the concept of leave-one-out (LOO), which measures the marginal contribution of a data point to the utility function (e.g., model accuracy) when it is removed from the training procedure. DataShapley [1] and its variants such as BetaShapley [17], DataBanzhaf [18], least core [19], are all based on the LOO principle, but differ in the way the marginal contributions are aggregated. We discuss several notable options beyond Shapley values below. Feldman and Zhang [20] simulate the data values by LOO retraining albeit constrained on a small sample of the training data, while DataModels [21] sacrifice the exactness of LOO to achieve better scalability by model predictions. Another line of popular methods is gradient-based. TracIn [22] estimates the importance of a training example by tracing the change in the test loss caused by the example during the training process. Variations of influence functions [23, 24] have their roots in robust statistics [25], and offer a gradient-based approximation of the LOO values.

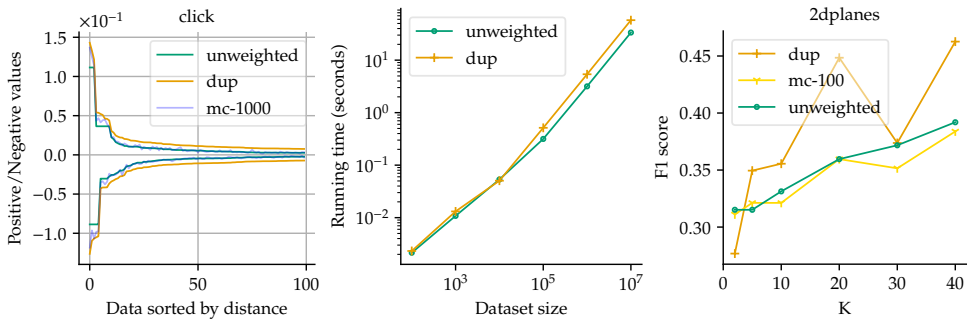
Shapley Values. Shapley values [6] originated in cooperative game theory as a method for fairly distributing gains among players, and have been widely adopted in multiple fields such as economics [26]. Computing exact Shapley values is well-known to be expensive, i.e., $\#\mathbf{P}$-hard in certain games [7]. This computational challenge has motivated various approximation techniques, including mostly Monte Carlo sampling [27, 28, 29, 30], and specialized algorithms for specific games [31]. Our work falls into the latter category. Early work on Shapley values with exogenous coalitions distributes among players from a group G the utility util(G), respecting the so-called *relative efficiency* axiom. Owen [14] and subsequent work [32] further consider a game between coalitions. Our duplication technique creates natural coalitions in the game.

kNN-based Shapley Values (kNN-SV). The k-Nearest Neighbor (kNN) model provides a unique opportunity for efficient computation of data Shapley values. Jia et al. [8] are the first to discover an efficient algorithm in time $\mathcal{O}(n_{test} n \log n)$ for unweighted kNN-SV. Wang and Jia [13] provide refinements to the unweighted kNN utility function. The weighted kNN case turns out to be more challenging due to the normalization factor in the utility function. Wang et al. [10] propose a time-consuming dynamic-programming algorithm for weighted kNN-SV with a hard-label utility function. Our work offers a more efficient approach for weighted kNN-SV.

7 Experiments

We investigate the following research questions in the experiments: (1) How does Algorithm 1 perform compared with the existing methods? We study this question with a task of noisy label detection. See Section 7.1. (2) How does the DkNN-SV deviate from those of the unweighted and weighted kNN-SV formulations? We visualize and compare them in Section 7.2. (3) How is the scalability of Algorithm 1? We study this in Fig. 1b and further in Appendix B.3. (4) What is the effect of the parameters on Algorithm 1? We study this in Fig. 1c and more in Appendix B.4.

Datasets. We evaluate the proposed methods on 11 datasets, whose statistics are listed in Table A1. The size of the datasets ranges from 5K to 1M. Many of them are chosen to be of a moderate size, so



(a) (b) (c)

Figure 1: Visualization of the Shapley values of different methods concerning a random test point (Fig. 1a). The running time on a random 1-dimensional dataset of different sizes (Fig. 1b). The effect of the parameter k on the performance of different methods (Fig. 1c).

as to allow us to compare with more costly baselines. By default, we randomly select 1% the data up to 100 points as the testing set.

Baselines. We include the following baselines in the experiments: (1) unweighted kNN-SV (*unweighted*), (2) scaled unweighted kNN-SV (*scaled*) in Section 4.2, (3) weighted kNN-SV (*mc*) by fast Monte Carlo sampling [8] with different number of samples, (4) random selection (*random*), and finally (5) our duplication-based DkNN-SV in Algorithm 1 (*dup*). The hard-label weighted kNN-SV [10] is not included as it fails to finish within 5 hours on small datasets (with default discretization bits $n_{bits} = 3$), so we report comparison results with it on tiny datasets in Appendix B.1. Our code can be found at a Github repository.[2] By default, we run each algorithm three times and report the average results.

7.1 Noisy label detection

To evaluate the performance of different methods, we follow the setup in previous works [8, 13, 10] and use the task of noisy label detection. For each dataset, we randomly flip the labels of 5% of the training data points, which forms a noisy subset of size $n/20$. We predict the noisy subset by the top-t data points with the lowest Shapley values. We set $t = 500$ for all methods. Intuitively, stronger data valuation methods should be able to detect noisy data points more accurately.

We tune the key parameter, kernel width σ, of all algorithms that use a kernel function as follows. We randomly select 5% of the training data as a validation set. We train a weighted kNN classifier on the rest of the data, and evaluate its accuracy on the validation set. We select the value σ that gives the highest accuracy from a list of candidates. Note that no information about the noisy labels is leaked to the validation process.

The results are shown in Tables 1 and 2 with standard deviation. Our proposed algorithm *dup* consistently outperforms all the other methods on most datasets. Its running time is slightly higher than that of the unweighted kNN-SV due to the additional cost in weighting the data points, but still very efficient. The weighted kNN-SV (*mc*) achieves better performance as the number of samples increases; however, it only reaches the performance of unweighted kNN-SV when the sample number is as large as 1000 and is nearly 1000 times slower. The scaled kNN-SV (*scaled*) is comparable to the unweighted kNN-SV. All methods perform significantly better than random selection.

7.2 Visualization of Shapley values

We visualize the Shapley values of different methods with respect to a random test point z_{test} as follows. We sort all data points by an increasing distance to z_{test} and plot their Shapley values. Besides, we plot the positive and negative values separately. We limit the number of points to the top 100 as the rest are all close to zero.

[2] `https://github.com/Guangyi-Zhang/weighted-knnsv-via-duplication`

Table 1: F1 scores of different methods on the noisy label detection task. The best one is highlighted in bold, and the second best is underlined. '-' indicates a timeout.

	unweighted	dup	scaled	random	mc-10	mc-100	mc-1000
phoneme	**0.318** $\pm$ 0.074	0.283 $\pm$ 0.167	0.232 $\pm$ 0.090	0.032	0.225	0.298	<u>0.318</u> $\pm$ 0.074
wind	0.286 $\pm$ 0.026	**0.357** $\pm$ 0.017	0.277 $\pm$ 0.009	0.043	0.240	0.283	<u>0.289</u> $\pm$ 0.017
cpu	<u>0.507</u> $\pm$ 0.089	**0.565** $\pm$ 0.014	0.444 $\pm$ 0.000	0.049	0.365	0.486	0.507 $\pm$ 0.096
2dplanes	0.342 $\pm$ 0.021	**0.368** $\pm$ 0.017	0.330 $\pm$ 0.027	0.055	0.226	0.339	<u>0.351</u> $\pm$ 0.016
apsfail	0.746 $\pm$ 0.004	**0.795** $\pm$ 0.007	0.737 $\pm$ 0.020	0.055	0.520	0.732	<u>0.749</u> $\pm$ 0.003
click	<u>0.070</u> $\pm$ 0.021	**0.072** $\pm$ 0.033	0.059 $\pm$ 0.011	0.055	0.054	0.067	0.069 $\pm$ 0.017
creditcard	0.119 $\pm$ 0.040	**0.132** $\pm$ 0.050	<u>0.120</u> $\pm$ 0.039	0.055	0.102	0.118	0.118 $\pm$ 0.036
fraud	0.879 $\pm$ 0.003	**0.888** $\pm$ 0.010	0.874 $\pm$ 0.019	0.055	0.737	0.853	<u>0.881</u> $\pm$ 0.011
pol	0.420 $\pm$ 0.003	**0.447** $\pm$ 0.021	0.425 $\pm$ 0.024	0.055	0.266	0.390	<u>0.430</u> $\pm$ 0.000
vehicle	**0.140** $\pm$ 0.007	0.126 $\pm$ 0.024	0.119 $\pm$ 0.031	0.055	0.111	0.128	<u>0.138</u> $\pm$ 0.007
poker	<u>0.150</u> $\pm$ 0.002	**0.305** $\pm$ 0.001	0.137 $\pm$ 0.001	0.050	0.069	-	-

Table 2: Running time (seconds) of different methods on the noisy label detection task. The best one is highlighted in bold, and the second best is underlined. '-' indicates a timeout.

	unweighted	dup	scaled	random	mc-10	mc-100	mc-1000
phoneme	<u>1.2</u> $\pm$ 0.2	1.6 $\pm$ 0.2	1.3 $\pm$ 0.1	**0.0**	9.8	92.1	874.1 $\pm$ 25.5
wind	<u>1.5</u> $\pm$ 0.0	2.4 $\pm$ 0.2	1.9 $\pm$ 0.1	**0.0**	14.9	129.2	1278.2 $\pm$ 26.6
cpu	<u>2.3</u> $\pm$ 0.2	3.7 $\pm$ 0.2	3.2 $\pm$ 0.3	**0.0**	22.3	206.7	1975.4 $\pm$ 12.3
2dplanes	<u>3.5</u> $\pm$ 0.3	5.8 $\pm$ 0.0	4.7 $\pm$ 0.2	**0.0**	33.5	307.5	2961.1 $\pm$ 69.8
apsfail	<u>3.6</u> $\pm$ 0.4	5.7 $\pm$ 0.6	4.7 $\pm$ 0.5	**0.0**	33.7	299.0	2940.1 $\pm$ 47.7
click	<u>3.4</u> $\pm$ 0.3	5.5 $\pm$ 0.6	4.5 $\pm$ 0.3	**0.0**	33.9	302.4	2931.1 $\pm$ 39.7
creditcard	<u>3.4</u> $\pm$ 0.4	5.4 $\pm$ 0.4	4.7 $\pm$ 0.4	**0.0**	33.9	305.7	2952.4 $\pm$ 49.5
fraud	<u>3.4</u> $\pm$ 0.2	5.3 $\pm$ 0.3	4.5 $\pm$ 0.3	**0.0**	33.9	300.0	2923.2 $\pm$ 27.9
pol	<u>3.6</u> $\pm$ 0.3	5.6 $\pm$ 0.9	4.7 $\pm$ 0.5	**0.0**	33.1	297.2	2937.4 $\pm$ 29.1
vehicle	<u>3.5</u> $\pm$ 0.0	5.6 $\pm$ 0.8	4.7 $\pm$ 0.3	**0.0**	34.6	307.2	2948.2 $\pm$ 4.2
poker	<u>368.7</u> $\pm$ 41.9	637.4 $\pm$ 74.3	496.5 $\pm$ 36.7	**0.0**	3606.3	-	-

The representative visualization is shown in Fig. 1a and more in Appendix B.2. All methods follow a similar pattern and assign a larger absolute values to nearer points. One common property that is shared by *dup* and *mc* is that they are able to differentiate the nearest points, while the unweighted kNN-SV often assigns an equal value to, for example, the top 10 points.

8 Conclusion

In this paper, we introduced a novel variant of weighted kNN-SV that leverages a duplication technique to reduce the weighted case to the unweighted one effectively. This variant successfully captures the weighted nature of the kNN models and maintains the desirable axiomatic properties of Shapley values, while being amenable to efficient computation.

We discuss limitations and directions for future work. Our method does not solve the original weighted kNN-SV problem but rather a new variant of it. Though the resulting data values can be justified axiomatically, they are not Shapley values. Several promising directions for future work include extending our approach to kNN regression tasks, and investigating the applicability of our duplication technique to other kNN-related scenarios.

Acknowledgments and Disclosure of Funding

This research is supported by the ERC Advanced Grant REBOUND (834862), the Swedish Research Council project ExCLUS (2024-05603), the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, Guangdong Provincial College Youth Innovative Talent Project (Grant No. 2025KQNCX075), Natural Science Foundation of Top Talent of SZTU (Grant No. GDRC202520), SZTU University Research Project (No. 20251061020002), and fundamental research funds for the central universities of Ministry of Education of China (SWU-KR24043).

References

[1] Amirata Ghorbani and James Y. Zou. Data shapley: Equitable valuation of data for machine learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.

[2] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1167–1176. PMLR, 2019.

[3] Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: a survey. *Mach. Learn.*, 113(5):2351–2403, 2024.

[4] Kevin Fu Jiang, Weixin Liang, James Y. Zou, and Yongchan Kwon. Opendataval: a unified benchmark for data valuation. *Advances in Neural Information Processing Systems*, 36:28624–28647, 2023.

[5] Rachael Hwee Ling Sim, Xinyi Xu, and Bryan Kian Hsiang Low. Data valuation in machine learning: "ingredients", strategies, and open challenges. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5607–5614. International Joint Conferences on Artificial Intelligence Organization, 2022.

[6] Lloyd S. Shapley. A value for n-person games. In *Contributions to the Theory of Games, Volume II*, pages 307–318. Princeton University Press, Princeton, NJ, USA, 1953.

[7] Xiaotie Deng and Christos H. Papadimitriou. On the complexity of cooperative solution concepts. *Math. Oper. Res.*, 19(2):257–266, 1994.

[8] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gürel, Bo Li, Ce Zhang, Costas J. Spanos, and Dawn Song. Efficient task-specific data valuation for nearest neighbor algorithms. *Proc. VLDB Endow.*, 12(11):1610–1623, 2019.

[9] Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8239–8247, 2021.

[10] Jiachen T. Wang, Prateek Mittal, and Ruoxi Jia. Efficient data Shapley for weighted nearest neighbor algorithms. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, pages 2557–2565. PMLR, 2024.

[11] Tomomi Matsui and Yasuko Matsui. A survey of algorithms for calculating power indices of weighted majority games. *Journal of the Operations Research Society of Japan*, 43(1):71–86, 2000.

[12] Edith Elkind and Jörg Rothe. Cooperative game theory. *Economics and computation: an introduction to algorithmic game theory, computational social choice, and fair division*, pages 135–193, 2016.

[13] Jiachen T. Wang and Ruoxi Jia. A note on "efficient task-specific data valuation for nearest neighbor algorithms", 2023.

[14] Guilliermo Owen. Values of games with a priori unions. In *Mathematical economics and game theory: Essays in honor of Oskar Morgenstern*, pages 76–88. Springer, 1977.

[15] Akshay Balsubramani, Sanjoy Dasgupta, Shay Moran, et al. An adaptive nearest neighbor rule for classification. *Advances in Neural Information Processing Systems*, 32, 2019.

[16] Ralph P Boas Jr and John W Wrench Jr. Partial sums of the harmonic series. *The American Mathematical Monthly*, 78(8):864–870, 1971.

[17] Yongchan Kwon and James Zou. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 8780–8802. PMLR, 2022.

[18] Jiachen T. Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 6388–6421. PMLR, 2023.

[19] Tom Yan and Ariel D Procaccia. If you like shapley then you'll love the core. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):5751–5759, 2021.

[20] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33: 2881–2891, 2020.

[21] Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Understanding predictions with data and data with predictions. In *Proceedings of the 39th International Conference on Machine Learning*, pages 9525–9587. PMLR, 2022.

[22] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33: 19920–19930, 2020.

[23] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017.

[24] Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8):8179–8186, 2022.

[25] R. Dennis Cook and Sanford Weisberg. *Residuals and Influence in Regression*. Chapman and Hall, New York, NY, USA, 1982.

[26] Alvin E. Roth. *The Shapley Value: Essays in Honor of Lloyd S. Shapley*. Cambridge University Press, Cambridge, UK, 1988.

[27] Javier Castro, Daniel Gómez, and Juan Tejada. Polynomial calculation of the shapley value based on sampling. *Comput. Oper. Res.*, 36(5):1726–1730, 2009.

[28] S. Shaheen Fatima, Michael J. Wooldridge, and Nicholas R. Jennings. A linear approximation method for the shapley value. *Artif. Intell.*, 172(14):1673–1699, 2008.

[29] Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. Bounding the estimation error of sampling-based shapley value approximation, 2013.

[30] Jiayao Zhang, Qiheng Sun, Jinfei Liu, Li Xiong, Jian Pei, and Kui Ren. Efficient sampling approaches to shapley value approximation. *Proceedings of the ACM on Management of Data*, 1(1):1–24, 2023.

[31] David Liben-Nowell, Alexa Sharp, Tom Wexler, and Kevin M. Woods. Computing shapley value in supermodular coalitional games. In *Computing and Combinatorics - 18th Annual International Conference, COCOON 2012, Sydney, Australia, August 20-22, 2012. Proceedings*, pages 568–579, Berlin, Heidelberg, 2012. Springer.

[32] Sergiu Hart and Mordecai Kurz. Endogenous formation of coalitions. *Econometrica: Journal of the econometric society*, pages 1047–1064, 1983.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We believe that the claims made in the abstract and introduction accurately reflect contributions and scope of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See the discussion in the conclusion (Section 8).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We have clearly stated every major assumption and provided complete and correct proofs for all the theorems.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided sufficient information in the paper and the associated source code to reproduce the main experimental results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have released the code with usage instructions, and all data used in the experiments are publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have provided all the details necessary to understand the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have reported standard deviation of key metrics in the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have provided sufficient information on the computer resources in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and we believe that the research conducted in the paper conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We propose a new methodology for the task of data valuation to facilitate fair and interpretable distribution of the total utility generated by data, which can have a positive societal impact. We are not aware of any potential negative societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any models or data that have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly credited and mentioned the license and terms of use of the data and code used in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We have provided sufficient documentation for the released code.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not involve research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We do not use LLMs as an important, original, or non-standard component of the core methods in this research.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Missing proofs

Theorem 3. *The group value $\phi(z \mid z_{test})$ defined in Eq.* (11) *satisfies the following axioms:*

1. ***Efficiency:*** *$\sum_{z \in D} \phi(z \mid z_{test}) = util'(D') - util'(\emptyset) = util(D) - util(\emptyset)$, that is, the sum of all group values equals the total utility.*
2. ***Symmetry:*** *If z_1 and z_2 are such that $util'(S' \cup \{z_1'\}) = util'(S' \cup \{z_2'\})$ for all $S' \subseteq D' \setminus \{z_1', z_2'\}$ where z_1' and z_2' are copies of z_1 and z_2 respectively, then $\phi(z_1 \mid z_{test})/w(z_1 \mid z_{test}) = \phi(z_2 \mid z_{test})/w(z_2 \mid z_{test})$.*
3. ***Dummy Player:*** *If z is such that $util'(S' \cup \{z'\}) = util'(S')$ for all $S' \subseteq D' \setminus \{z'\}$ where z' is a copy of z, then $\phi(z \mid z_{test}) = 0$.*
4. ***Additivity:*** *If $util_1'$ and $util_2'$ are two utility functions, and ϕ_1 and ϕ_2 are their corresponding values, then the value ϕ corresponding to $util_1' + util_2'$ satisfies $\phi = \phi_1 + \phi_2$.*

Proof of Theorem 3. We prove each property separately.

Efficiency: It directly follows from the efficiency of Shapley values $s'(z' \mid z_{\text{test}})$ in the duplicated game. The equality $\text{util}'(D') = \text{util}(D)$ holds because of Proposition 2.

$$\sum_{z \in D} \phi(z \mid z_{\text{test}}) = \sum_{z \in D} \sum_{z' \in D' : z' = z} s'(z' \mid z_{\text{test}}) = \sum_{z' \in D'} s'(z' \mid z_{\text{test}}) = \text{util}'(D') - \text{util}'(\emptyset).$$

Symmetry: If two points z_1 and z_2 make identical marginal contributions through their copies, then by the symmetry of Shapley values, each copy receives the same value, i.e., $s'(z_1' \mid z_{\text{test}}) = s'(z_2' \mid z_{\text{test}})$. Thus,

$$\frac{\phi(z_1 \mid z_{\text{test}})}{w(z_1 \mid z_{\text{test}})} = \frac{w(z_1 \mid z_{\text{test}}) s'(z_1' \mid z_{\text{test}})}{w(z_1 \mid z_{\text{test}})} = s'(z_1' \mid z_{\text{test}}) = s'(z_2' \mid z_{\text{test}}) = \frac{\phi(z_2 \mid z_{\text{test}})}{w(z_2 \mid z_{\text{test}})}.$$

Dummy Player: If a point z contributes nothing through any of its copies, then by the dummy player property of Shapley values, $s'(z' \mid z_{\text{test}}) = 0$ for all copies z' of z. Therefore,

$$\phi(z \mid z_{\text{test}}) = w(z \mid z_{\text{test}}) s'(z \mid z_{\text{test}}) = w(z \mid z_{\text{test}}) \cdot 0 = 0.$$

Additivity: Let util_1' and util_2' be two utility functions with corresponding values ϕ_1 and ϕ_2. Let s_1' and s_2' be the Shapley values in the duplicated games for util_1' and util_2'. By the additivity of Shapley values, $s'(z') = s_1'(z') + s_2'(z')$ for the combined utility $\text{util}' = \text{util}_1' + \text{util}_2'$. Therefore:

$$\begin{aligned}
\phi(z \mid z_{\text{test}}) &= w(z \mid z_{\text{test}}) s'(z \mid z_{\text{test}}) \\
&= w(z \mid z_{\text{test}})(s_1'(z \mid z_{\text{test}}) + s_2'(z \mid z_{\text{test}})) \\
&= w(z \mid z_{\text{test}}) s_1'(z \mid z_{\text{test}}) + w(z \mid z_{\text{test}}) s_2'(z \mid z_{\text{test}}) \\
&= \phi_1(z \mid z_{\text{test}}) + \phi_2(z \mid z_{\text{test}}).
\end{aligned}$$

□

Proposition 4. *In the genral case, the group Shapley values in Eq.* (11) *may be different from the Shapley values of weighted k*NN-SV *in Eq.* (4)*.*

Proof of Proposition 4. Consider a dataset $D = \{z_1, \ldots, z_n\}$ where all points share z_{test}'s label. Note that since all points have the correct label, the marginal contribution of adding any point to a subset S is non-zero only when S is empty, regardless of the value of k. Let $w(z_1) = 2$ and $w(z_i) = 1$ for $i \neq 1$.

We first discuss the weighted kNN-SV on D. It is easy to see that

$$s_1 = \frac{1}{n}\left(1 - \frac{1}{C}\right).$$

Next, we consider the unweighted kNN-SV on D'. By symmetry, we have $s_{z_1}' = s_{z_1'}'$, so

$$\phi(z_1) = s_{z_1}' + s_{z_1'}' = \frac{2}{n+1}\left(1 - \frac{1}{C}\right).$$

Equality $s_1 = \phi(z_1)$ holds *only* when $n = 1$. For $n > 1$ it holds $\frac{2}{n+1} > \frac{1}{n}$. □

Table A1: Statistics of the datasets used in the experiments.

Dataset	n	d	$\lvert\mathcal{Y}\rvert$
phoneme	5404	5	2
wind	6574	14	2
cpu	8192	21	2
2dplanes	10000	10	2
apsfail	10000	170	2
click	10000	11	2
creditcard	10000	23	2
fraud	10000	30	2
pol	10000	48	2
vehicle	10000	100	2
poker	1000000	10	10

Lemma 5. *Fix a test point z_{test}. Let z be a point in D. Then, every copy z' of z has the same Shapley value $s'(z' \mid z_{test})$ as that of z. This continues to hold when the values are obtained by applying the recursive formula in Theorem 1 with an arbitrary order among z and its copies.*

Proof of Lemma 5. The first statement is a direct consequence of the symmetry property of the Shapley values. The second statement follows by observing that any two copies of z share the same label, which turns the second term in Eq. (7) into zero. Thus, their values do not depend on the order of z and its copies. □

Theorem 6. *Algorithm 1 computes duplication-based weighted kNN-SV in $\mathcal{O}(dn + n \log n)$ time.*

Proof of Theorem 6. Following Lemma 5, we take an order where each point z is behind all its copies. Then we apply the recursive formula in Theorem 1 to compute the Shapley values. After processing each point $z \in D$, we can skip its copies by multiplying $s'(z \mid z_{\text{test}})$ by the number of its copies, as instructed by Lemma 5. Thus, no copies need to be materialized, and the time complexity is dominated by the sorting step, which is $\mathcal{O}(dn + n \log n)$. □

B Additional experimental details

Experimental Environment. All algorithms were implemented in Python 3.11. All experiments were carried out on a Linux server equipped with 64 CPUs of Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60 GHz and 1511 GB RAM.

See Table A1 for the statistics of the datasets used in the experiments.

B.1 Comparison with hard-label weighted kNN-SV

Instead of resorting to the approximate version, we choose to compare with the exact version of this baseline [10] over small datasets. The hard-label weighted kNN-SV is denoted as *dp* below. We set the data size to be 300 by randomly sampling from the original datasets. We tune the parameter of kernel width in the same fashion as in the main experiments. Very surprisingly, as reported in Table A2, the *dp* is only slightly better than random guessing for the task of noisy label detection, and is far behind the performance of the unweighted kNN-Shapley and our method. To make sure this is not caused by bugs in the code, we further verify that its values are indeed consistent with those by Monte-Carlo sampling with a hard-label utility function.

After careful inspection, we identify the key reason: a hard-label utility function may be less suitable for the task of noisy label detection, or require stronger signal from distance-based weights. Unlike the soft-label utility, it is unable to capture the fine-grained contribution of a data point. It requires the noisy point to be a game changer for the prediction of its neighbors in order to be considered harmful, which is not the case for most mislabeled points. A mislabeled point is often surrounded by well-labeled points, and its ability to change the prediction of its neighbors is often negligible. Note

Table A2: F1 scores of different methods on the noisy label detection task. The best one is highlighted in bold, and the second best is underlined.

	dup	unweighted	random	dp
2dplanes	<u>0.232</u> $\pm$ 0.090	**0.250** $\pm$ 0.092	0.036	0.071 $\pm$ 0.058
apsfail	**0.786** $\pm$ 0.143	<u>0.696</u> $\pm$ 0.090	0.036	0.071 $\pm$ 0.058
click	**0.036** $\pm$ 0.041	<u>0.036</u> $\pm$ 0.041	0.036	0.036 $\pm$ 0.041
cpu	**0.607** $\pm$ 0.071	<u>0.536</u> $\pm$ 0.149	0.036	0.071 $\pm$ 0.101
creditcard	<u>0.107</u> $\pm$ 0.124	**0.125** $\pm$ 0.036	0.036	0.054 $\pm$ 0.068
fraud	<u>0.821</u> $\pm$ 0.137	**0.893** $\pm$ 0.071	0.036	0.036 $\pm$ 0.041
phoneme	**0.321** $\pm$ 0.092	<u>0.321</u> $\pm$ 0.041	0.000	0.018 $\pm$ 0.036
pol	**0.339** $\pm$ 0.236	<u>0.339</u> $\pm$ 0.147	0.036	0.071 $\pm$ 0.058
vehicle	<u>0.143</u> $\pm$ 0.154	**0.179** $\pm$ 0.189	0.036	0.107 $\pm$ 0.092
wind	**0.250** $\pm$ 0.092	<u>0.214</u> $\pm$ 0.154	0.089	0.036 $\pm$ 0.041
poker	**0.071** $\pm$ 0.000	<u>0.071</u> $\pm$ 0.000	0.054	0.000 $\pm$ 0.000

that resorting to a hard-label utility function is the key modification that enables the DP algorithm in Wang et al. [10] to work.

B.2 Visualization of the Shapley values

See Fig. A1 for more visualization of the Shapley values.

B.3 Scalability

We compare the running time of *unweighted* and *dup* with different dataset sizes. We vary the size of a random 1-dimensional dataset D from 100 to 10 M. The results are shown in Fig. 1b. As we can see, the running time of *dup* follows that of the unweighted kNN-SV closely, confirming its near-linear time complexity.

B.4 Effect of the parameter k

We examine the effect of the parameter k on the performance of representative methods. See Fig. A2 for more results on the effect of the parameter k. Overall, the performance of all methods is robust to the choice of k.

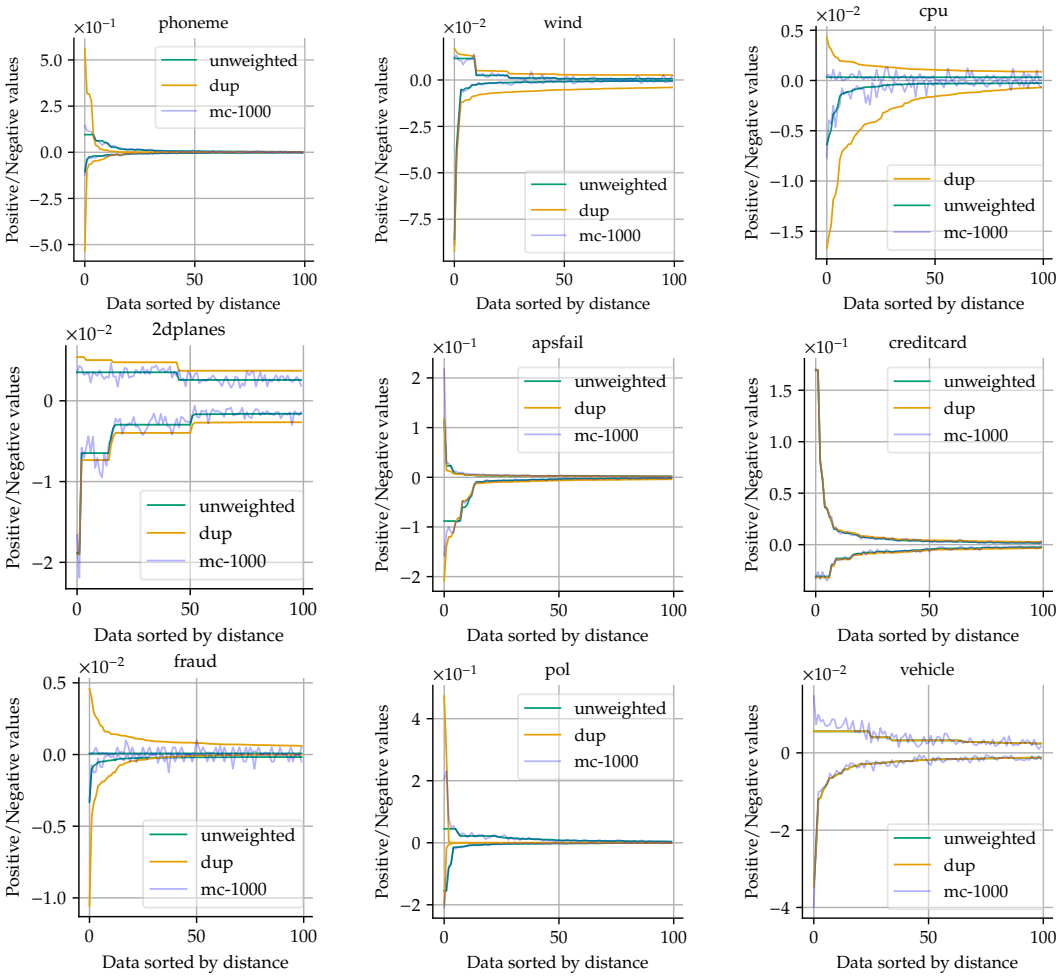


Figure A1: Visualization of the Shapley values of different methods concerning a random test point.

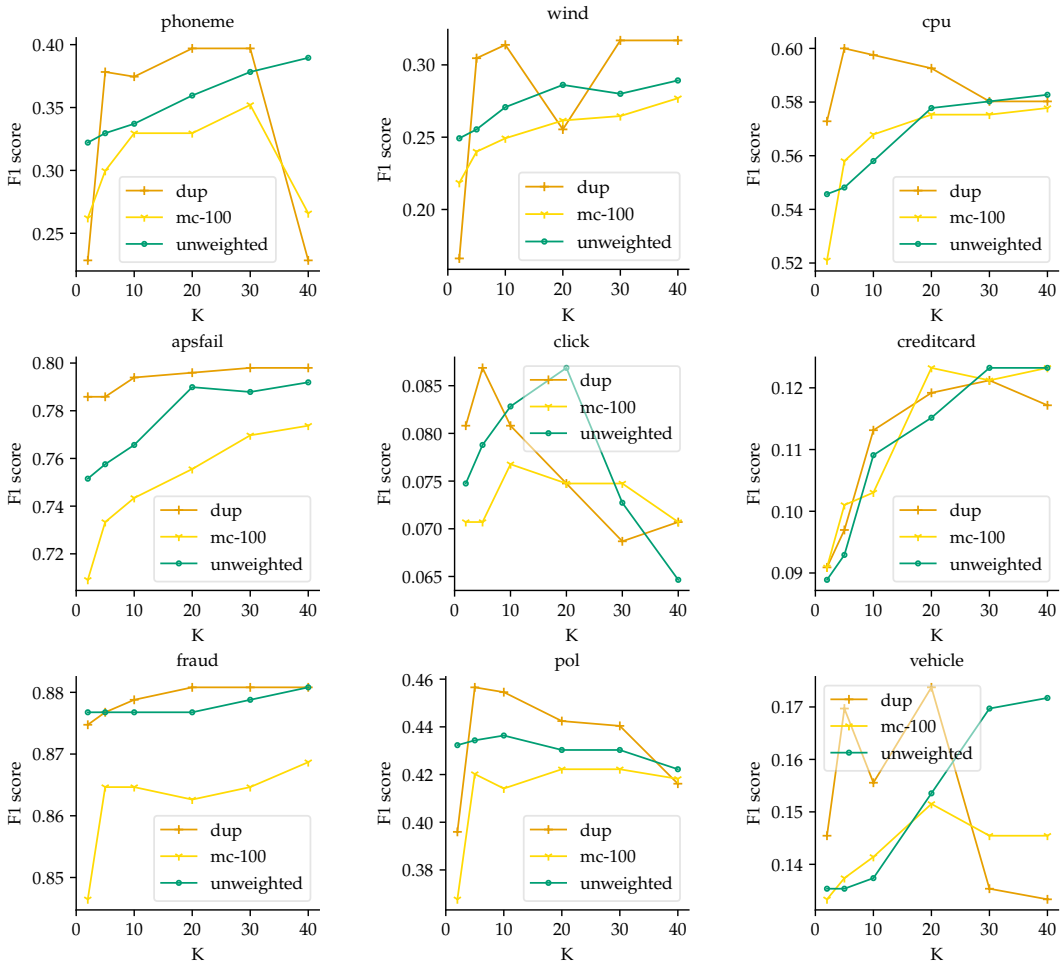


Figure A2: Effect of the parameter k on the performance of different methods.