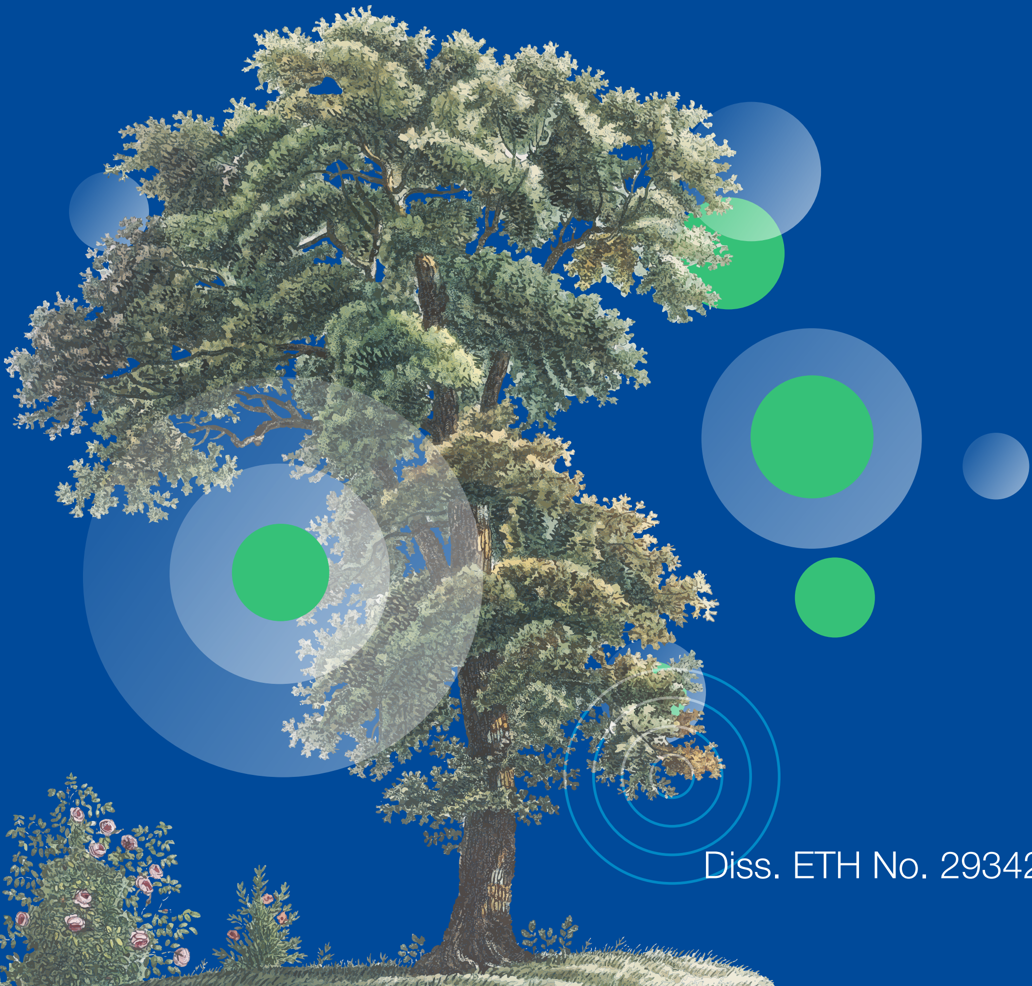
David Dao

Advancing Algorithms and Applications for Data Valuation in Machine Learning



Diss. ETH No. 29342

DISS. ETH NO. 29342

ADVANCING ALGORITHMS AND APPLICATIONS FOR DATA VALUATION IN MACHINE LEARNING

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by

David Dao
Master of Science in Computer Science, TU Munich

born on 12 June 1991
citizen of Germany

accepted on the recommendation of

Prof. Dr. Ce Zhang (ETH Zurich), examiner
Prof. Dr. Gustavo Alonso (ETH Zurich), co-examiner
Dr. Theodoros Rekatsinas (Apple), co-examiner
Dr. Matteo Interlandi (Microsoft GSL Redmond), co-examiner

2023

DAVID DAO

ADVANCING ALGORITHMS AND APPLICATIONS FOR DATA VALUATION IN MACHINE LEARNING

This doctoral thesis is based on previously published research which is the result of cooperative efforts with Attila Steinegger, Björn Lütjens, Bo Li, Boxin Wang, Ce Zhang, Costas J. Spanos, Dawn Song, Frances Ann Hubis, Gyri Reiersen, Konstantin Klemmer, Nezihe Merve Gürel, Nick Hynes, Raymond Cheng, Ruoxi Jia, Xiaoxiang Zhu and many other co-authors and collaborators.

Although this dissertation carries my name, the research presented here would not have been achievable without their invaluable contributions.

To the brave Vietnamese boat people,
and the compassionate crew of the Cap Anamur.
Thank you for a new home.

ABSTRACT

"How much is my data worth?" is an increasingly common question posed by organizations and individuals alike. An answer to this question could allow, for instance, fairly distributing profits among multiple data contributors, determining prospective compensation when data breaches happen. This dissertation takes a first step toward *data valuation* by presenting a principled framework utilizing the Shapley value, a popular notion of value which originated in cooperative game theory.

First, we show that the Shapley value defines a unique payoff scheme that satisfies many desiderata for the notion of data value. However, the Shapley value often requires *exponential* time to compute. To meet this challenge, we propose efficient algorithms for approximating the Shapley value with provable error bounds for general machine learning (ML) utilities. Alongside its theoretical robustness, our empirical findings indicate that the Shapley value aligns with people's intuitive understanding of data value.
Second, we present a family of efficient algorithms for computing the exact Shapley values for *K*NN classification and regression. We demonstrate that both the exact algorithm and the approximate algorithm for *K*NN Shapley can scale to millions of data points, making them suitable for valuing data in common ML datasets.
Lastly, we explore the practical challenges that data marketplaces are facing focusing on two main concerns: Training machine learning models on private data and curating specialized and complex datasets. To study and address these challenges, we demonstrate a decentralized design of a marketplace for private data and incentivize the creation of a real-world ecological dataset benchmark.

ZUSAMMENFASSUNG

"Wie viel sind meine Daten wert?" ist eine immer häufiger gestellte Frage von Organisationen und Einzelpersonen. Eine Antwort auf diese Frage könnte beispielsweise dazu beitragen, Gewinne fair unter mehreren Datenlieferanten aufzuteilen und potenzielle Entschädigungen bei Datenverstößen festzulegen. Diese Dissertation unternimmt einen ersten Schritt in Richtung *Datenwertanalyse*, indem sie einen prinzipientreuen Rahmen auf der Grundlage des Shapley-Werts präsentiert, einer populären Wertvorstellung, die ihren Ursprung in der kooperativen Spieltheorie hat.
Zuerst zeigen wir, dass der Shapley-Wert ein einzigartiges Auszahlungsschema definiert, das viele Desiderate für den Begriff des Datenwerts erfüllt. Der Shapley-Wert erfordert jedoch häufig exponentielle Zeit für die Berechnung. Um dieser Herausforderung zu begegnen, schlagen wir effiziente Algorithmen für die Approximation des Shapley-Werts mit nachweisbaren Fehlergrenzen für allgemeine maschinelle Lernfunktionen (ML) vor. Neben seiner theoretischen Robustheit zeigen unsere empirischen Ergebnisse, dass der Shapley-Wert mit einem intuitiven Verständnis für den Datenwert übereinstimmt. Zweitens präsentieren wir eine Familie von effizienten Algorithmen zur Berechnung der exakten Shapley-Werte für KNN-Klassifikation und Regression. Wir demonstrieren, dass sowohl der exakte Algorithmus als auch der approximative Algorithmus für KNN-Shapley auf Millionen von Datenpunkten skaliert werden können, was sie für die Bewertung von Daten in gängigen ML-Datensätzen geeignet macht. Schließlich untersuchen wir die praktischen Herausforderungen, denen sich Datenmarktplätze gegenübersehen, und konzentrieren uns dabei auf zwei Hauptanliegen: Das Trainieren von maschinellen Lernmodellen auf privaten Daten und das Kuratieren von spezialisierten und komplexen Datensätzen. Um diese Herausforderungen anzugehen, zeigen wir ein dezentrales Design eines Marktplatzes für private Daten und motivieren die Erstellung eines realen ökologischen Datenbenchmark.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Ce, for giving me the unique opportunity to work on problems that spark my curiosity alongside an amazing group of people. I am truly grateful for your support and the scientific freedom and opportunity you provided me over the years to chase after my dreams, even when they sounded a bit wild. Thank you for your trust, guidance and mentorship. In that spirit, I want to thank all my members of my research group - the DS3Lab. It has been a remarkable journey witnessing the group's growth from its early days. I'd like to extend special thanks to the initial DS3-wave, with whom I spent the most time - Johannes, for being a fantastic friend since our master's studies in Munich; Bojan and Frances, for our unforgettable adventures; Josh, Susie and Zhikuan, for being among the kindest people I've ever met; Maurice and Leo, for the countless laughs we shared; Luka, Cedric, Nora, Merve, Shaoduo, and Jiawei, for our unforgettable retreats and daily ML moments; and, of course, all the newer members who have become invaluable members of our academic family.

I would like to express my heartfelt gratitude to the wonderful Systems Group, including Gustavo, Mothy, Ana, Theo, Ankit, Simonetta, Nadia, Jena, and the numerous kind-hearted members who have been a part of this group over the years, making it truly exceptional. Additionally, I want to extend my warmest appreciation to Gustavo, Theo and Matteo for being part on my defense committee.

During my PhD I was fortunate to work with many inspiring collaborators. I would like to extend my gratitude to Dawn Song, for hosting me in Berkeley and at Oasis Labs, as well as for the countless enlightening conversations over the years; Ruoxi Jia, for her tremendous kindness and support; Robert Chang, for welcoming me at Stanford Medical School and guiding me through the daily operations of a hospital; Xiaoxiang Zhu and Dava Newman, for their exceptional dedication in supporting ambitious research ideas and local communities; Kristy Deiner, Tom Crowther, Attila Steinegger, Björn Lütjens, Lucas Czech for their passion and dedication to our planet's biodiversity, and my early academic mentors, Alexis Stamatakis, Shantanu Singh, and Anne Carpenter, for their unwavering support and guidance even after more than a decade; and my rockstar students Lasse, Thomas, Nils, Ghjulia, Marc, Gyri, Kenza, Simona, Iveta, Mina, Levin, Catherine, Ming, Luca, Florian, Nino, and Chris.

Additionally, I would like to thank the inspiring Climate Change AI, particularly Priya, David, and Lynn, for creating and welcoming me into the AI and climate change community.

Last, but certainly not least, I have been incredibly blessed to be surrounded by the best friends and family anyone could ask for. Sharfy, Simge, Simeon, Yumna, Reuven, Facundo, Shirley, Angela, Niru, Frances, Marius, Charlotte, Johannes & Johannes, Alex & Alex, Melissa, Lucas, Theodoris, Alexnick, Tal, Frederik, Yifei, Marie-Claire, Chip, Lindsey, Christy, Hala, Paula, Camille, Guido, Anna, Zeynep, Özgür, Lucas, Björn, Judith, Lucy, the Global Shapers community, Team GainForest, and many others have enriched my life. Mom, Dad, and Anna - thank you for your unwavering support and encouragement. It is impossible to express the significance of these individuals in words, but I am confident that each of them is well aware of the impact they have had on my life.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1

MOTIVATION AND RESEARCH QUESTIONS

Measure what is measurable, and make measurable what is not so.

— Galileo Galilei

Throughout our history, humanity has seen value in collecting, storing, and exchanging data. From the development of the first writing systems in Mesopotamia around 5,000 years ago, that has allowed us to record, store and share stories and observations from our ancestors, to the invention of modern computer systems in the last century, that enabled us to process and derive data-driven insights, products and decisions, datasets have played a crucial role for scientists, companies and governments. In recent decades, the development of mature machine learning (ML) systems have facilitated the rise of successful data-driven applications, including e-commerce (Ballestar, Grau-Carles, and Sainz, 2019; Rao et al., 2020), online advertising (Lacerda et al., 2006; Perlich et al., 2014; Rafieian and Yoganarasimhan, 2021), personalized medicine (Carpenter et al., 2006; Bray et al., 2016; McQuin et al., 2018; Bunne et al., 2021), sustainability (Rolf et al., 2021; Rolnick et al., 2022; Beery et al., 2022), ride-hailing services (Huval et al., 2015; Bojarski et al., 2016; Grigorescu et al., 2020), and social media (Chancellor, Baumer, and De Choudhury, 2019). Consequently, these applications collect large amounts of, often sensitive and personal, data from their users in order to provide their goods and services. In 2022 alone, by predicting targeted advertisement from its user data, Google and Facebook were able to jointly generate over 330 billion dollars in advertising revenue[1]. Advocates of the data economy often emphasize that *"data is the new oil"* to underscore its importance (Delacroix and Lawrence, 2019). Naturally, given the economic significance of data, it raises the question of whether it is possible to quantify and compute the worth of a single data point.

[1] `https://dweb.link/ipfs/bafkreidszsuulthvfi3u42ym273rgiw5xzgi3ar3unvuloiqm4qjwzdr4i`

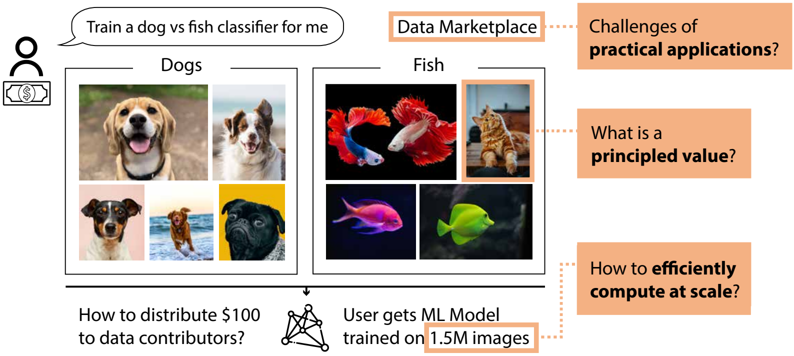

Train a dog vs fish classifier for me
Data Marketplace
Challenges of **practical applications**?
Dogs
Fish
What is a **principled value**?
How to **efficiently compute at scale**?
How to distribute $100 to data contributors?
User gets ML Model trained on 1.5M images

FIGURE 1.1: Data market example scenario. A customer requests a dog vs fish classifier and chooses his data sources from various data contributors through a marketplace. How should he distribute his payment of 100$ in an equitable (addressing low data quality) and scalable (millions of images) way?

Organizations and individuals alike pose an increasingly common question: *"How much is my data worth?"*

On the one hand, this question is of great interest to data contributors. More and more initiatives in consumer and healthcare markets propose that individuals should be compensated for the personal data they generate. For example, emerging data markets (Richter and Slowinski, 2019; McConaghy, 2022) already enable users to sell their data. An answer to this question could allow, for instance, fairly distributing profits among multiple data contributors or determining prospective compensation when data breaches happen (Scaria et al., 2018).

Additionally, from the perspective of ML, data consumers would like to understand which data points to choose from. Alternatively, if they already have an existing dataset, they would like to understand the value and importance of their training examples, relative to other training examples, in order to explaining model predictions (Lundberg and S.-I. Lee, 2017). Data importance can help ML practitioners in determining which data point to include or exclude to improve the model's utility with profound impact on a range of applications including interpretability, robustness, and data

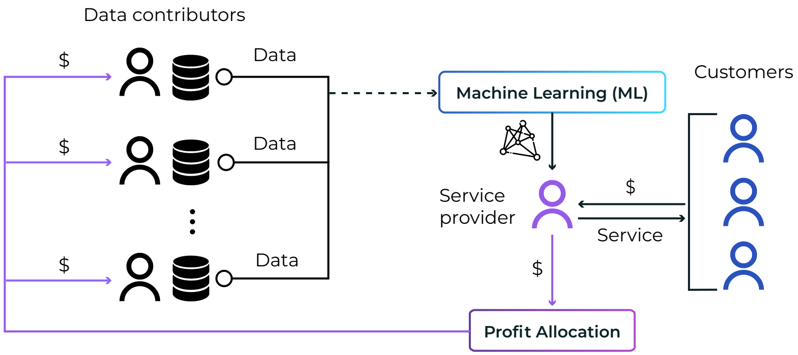


FIGURE 1.2: Fundamental problem of data valuation. Data contributors provide data points that are being used for machine learning training. A service provider than uses the resulting model to sell a product to customers. The customers pay out the service provider. How should the service provider allocate the profit to the data contributors?

acquisition among others.

Surprisingly, unlike the clear quantifiable value of a barrel of oil (with which data is often compared to), or the goods and services that are directly derived from data, setting a price tag and valuation on an individual data point itself is a complex and challenging task. A fair valuation for data is not easily defined as many aspects of data values have not been previously explored in this context.

MOTIVATING EXAMPLE To illustrate this, envision a data marketplace where data contributors can sell data to train a classifier (see Figure 1.2).

Problem of Data Valuation

> **Given a data set D provided by many data contributors and a model owner who is willing to pay \$$X$ to train a machine learning (ML) model over D, how should we distribute \$$X$ to each data point to reflect its "value"?**

Intuitively, a value concept should satisfy some key characteristics of data. For instance, the value of a data source is *combinatorial* and depends on the presence of other data sources. Furthermore, the value of a data point *depends on the outcome* of the ML model and the specific task. Also, since data is non-rival (allowing to be trained for multiple models), the idea of value should be *cumulative*. Moreover, data contributors might provide low-quality data (e.g., in Figure 1.1, a data contributor mislabeled a fish as a cat). How can we ensure that the value is *equitable* for all data contributors? In addition to meeting these requirements, a data marketplace must also be scalable to handle millions of data points as well as tackle real-world challenges, such as privacy concerns and data processing.

RESEARCH QUESTIONS In this dissertation we break down this scenario into three high-level questions that guide our research agenda and address different aspects, including desired theoretical guarantees and properties, scalability, and real-world challenges.

Research Question 1

> *What is a principled framework to address data valuation in the context of supervised machine learning?*

We aim to study the question of data valuation first from a theoretical perspective. By developing a principled framework, it will help us to identify desired properties that data valuation methods should possess and establish theoretical guarantees that ensure the reliability and accuracy of our methods.

Research Question 2

> *Can we efficiently compute the value of millions of individual data points?*

Machine learning has achieved great success in recent years, largely due to the ability to work with increasingly large data sets and faster computation.

As a result, it is crucial to develop efficient algorithms for data evaluation that can handle real-world training data sets, which can contain millions of data points.

Research Question 3

> *What are the challenges when building practical applications that utilize and benefit from data markets?*

To ensure that a data marketplace with data valuation for real-world data is beneficial, it is important to carefully consider the specific environment and needs of the application. This includes taking into account factors such as privacy, incentives for data contribution and curation, and other considerations that are relevant to the application's real-world context.

1.1 Scientific Contributions

We provide three main contributions that address the above raised questions in this dissertation. More detailed descriptions of each contribution can be found in the beginning of each respective chapter.

CONTRIBUTION 1 We provide a principled framework for data valuation based on the Shapley value. We show that this framework uniquely satisfies many desired properties. In order to estimate the Shapley value in diverse scenarios, we developed methods considering general machine learning utilities. Alongside its theoretical robustness, our empirical findings show that the Shapley value aligns with people's intuitive understanding of data value.

CONTRIBUTION 2 We present an efficient algorithm for computing the exact Shapley values for unweighted KNN classification. We demonstrate that both the exact algorithm and the approximate algorithm utilizing locality sensitive hashing (LSH) can scale to millions of data points, making them suitable for real-world data valuation. Furthermore, we expand our algorithms to address additional scenarios, such as (1) weighted *K*NN classifiers, (2) data points clustered by distinct data curators, and (3) valuing computation and data jointly.

CONTRIBUTION 3 We explore the practical challenges faced by data marketplaces, with a primary focus on two key concerns: training machine learning models using private data and curating specialized, complex datasets. We introduce two applications *Sterling* and *ReforesTree*, each tackling these challenges. Sterling demonstrate a decentralized design of a task-driven marketplace for private data, while ReforesTree incentivizes the creation of a real-world ecological dataset benchmark.

1.2 Organization of the Thesis

This thesis is organized as follows: We start by providing relevant background in Chapter 2 and a comprehensive overview of related work in data valuation. In Chapter 3, we then proceed by introducing a principled framework for data valuation based on the Shapley value. We examine some general properties of machine learning models and analyze their implications for calculating the Shapley value. In Chapter 4, we study Shapley-based data valuation for K Nearest Neighbor methods to provide exact algorithms that can scale to millions of data points. In Chapter 5 we study potential applications of data marketplaces and their corresponding challenges. We conclude this thesis with Chapter 6 by summarizing the contributions and limitations of this thesis, along with outlining potential future work.

1.3 Author's Publications

This dissertation is largely based on four publications presented in the order of appearance of this thesis (* co-first authorship):

- Ruoxi Jia*, **David Dao***, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Guerel, Bo Li, Ce Zhang, Dawn Song, Costas J. Spanos.

"Towards Efficient Data Valuation Based on the Shapley Value". *AISTATS* 2019.

- Ruoxi Jia, **David Dao**, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas J. Spanos, Dawn Song. "Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms". *VLDB* 2019.
- Nick Hynes, **David Dao**, David Yan, Raymond Cheng, Dawn Song "A demonstration of Sterling: A Privacy-Preserving Data Marketplace". *VLDB Demo* 2018.
- Gyri Reiersen, **David Dao**, Björn Lütjens, Konstantin Klemmer, Kenza Amara, Attila Steinegger, Ce Zhang, Xiaoxiang Zhu. "ReforesTree: A Dataset for Estimating Tropical Forest Carbon Stock with Deep Learning and Aerial Imagery". *AAAI* 2022.

Further published works, which are outside the scope of this thesis in chronological order:

- Bojan Karlaš, **David Dao**, Matteo Interlandi, ..., Wentao Wu, Ce Zhang. "Data Debugging with Shapley Importance over End-to-End Machine Learning Pipelines". Submitted to *NeurIPS* 2023
- Alexandre Lacoste, ... **David Dao**, ... Yoshua Bengio, Stefano Ermon, Xiaoxiang Zhu "GEO-Bench: Toward Foundation Models for Earth Monitoring". Submitted to *ICCV* 2023
- Ghislain Fourny, **David Dao**, ... Ce Zhang, Gustavo Alonso. "RumbleML: program the lakehouse with JSONiq". Arxiv. 2022
- Leonel Aguilar, **David Dao**, ... Wentao Wu and Ce Zhang. "Ease.ML: A Lifecycle Management System for Machine Learning". *Conference on Innovative Data Systems Research* 2021.
- Ruoxi Jia, ... **David Dao**, ... Bo Li and Dawn Song "Scalability vs. Utility: Do We Have To Sacrifice One for the Other in Data Importance Quantification?". *CVPR* 2021.
- Laure Berti-Equille, **David Dao**, Stefano Ermon, Bedharta Goswami. "Challenges in KDD and ML for Sustainable Development". *KDD* 2021.
- Simona Santamaria*, **David Dao***, Björn Lütjens*, Ce Zhang "True-Branch: Robust Deep Learning-Based Verification of Forest Conservation Projects". *ICLR CCAI Workshop* 2020.

2 BACKGROUND

If you cannot explain something in simple terms, you don't understand it.

— Richard Feynman

In this chapter, we give a high-level background on some of the relevant topics, with the aim of providing readers with a common foundation on which to approach the content of this thesis. In the following sections, we will give:

1. A primer of the basic principles of supervised machine learning, explaining the relevant terminology and dependencies for data valuation
2. Foundational concepts of cooperative game theory which provide a starting point for a principled framework and discussion on data valuation
3. A comprehensive overview of related work in data valuation and beyond.

2.1 Background on Machine Learning

The goal of this section is to emphasize the connection between a machine learning algorithm, the specific task being performed and its training dataset. Unlike traditional computer programs that require manual feature engineering, ML models are trained on data and examples. In recent years, machine learning has allowed us to tackle many complex tasks with great success such as computer vision (Simonyan and Zisserman, 2014; Krizhevsky, Sutskever, and G. E. Hinton, 2017; K. He et al., 2016; Szegedy et al., 2016a; Russakovsky et al., 2015; Tan, Pang, and Le, 2020; Dosovitskiy et al., 2021), natural language processing, (LeCun, Bengio, and G. Hinton, 2015; Vaswani et al., 2017; Bommasani et al., 2021; OpenAI, 2023; T. B. Brown et al., 2020) speech recognition (G. Hinton et al., 2012; Amodei et al., 2015; Devlin et al., 2018; Raffel et al., 2020), and reinforcement learning (Mnih et al., 2013; Horgan et al., 2018; Kapturowski et al., 2019; Schrittwieser et al., 2019). This accomplishment is due to a number of factors, including the explosion of large amounts of available data, advancements in hardware infrastructure and computing power, and the development of novel algorithms in deep learning.

(Mitchell, 1997) provided a widely recognized definition of machine learning that highlights the link between the learning algorithm, the task and the data: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"*. Based on the type of data that the learning algorithms are allowed to experience during the learning process, one can thus broadly classify machine learning algorithms into unsupervised or supervised learning. In unsupervised learning, algorithms are trained on a dataset that contains many features, and they aim to learn useful properties or structures that can be utilized to solve a particular task. On the other hand, in supervised learning, each data point, besides the features, is also associated with a label or target. The term "supervised learning" originates from the concept that the labels or targets provided can be interpreted as a supervision signal to the machine learning system, guiding it on what to do.

2.1.1 *Classification and Regression Tasks*

In this thesis we will be focusing mainly on supervised learning. Given a feature space $\mathcal{X}$, a label space $\mathcal{Y}$, n samples, and a labeled dataset $D := \{(z_i)\}_{i\in[n]}$ with data points (or examples) $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. Each data point is a pair of an instance x_i and its corresponding label y_i. Typically the feature space is a subset of the real numbers $\mathcal{X} \subset \mathbb{R}^d$ and each instance x_i is a d-dimensional vector where each dimension corresponds to a feature (e.g a pixel in an image) whilst the label space $\mathcal{Y}$ depends on the specific task T and can be either discrete or continuous. For example, if task T is a *classification task* (e.g. recognize objects in images) on C number of distinct classes, we usually assume that the label space is discrete and $\mathcal{Y} = \{1, 2, \ldots, C\}$. The ML algorithm can then be asked to produce a predictor $f : \mathbb{R}^n \to \{1, \ldots, C\}$ that predicts the correct class given x. On the other hand, in a *regression task* (e.g. predict future stock prices) the label space is continuous and usually takes the form $\mathcal{Y} = \mathbb{R}$. A machine learning algorithm can then for instance be asked to produce a predictor $f : \mathbb{R}^n \to \mathbb{R}$ which predicts the correct numerical value given the sample.

2.1.2 *Performance measure*

In order to assess a machine learning algorithm, we need to design a quantitative number of how well it is doing. For that, we need a performance measure. Often, this can be i.e. a loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ that quantifies the cost of each possible prediction for a given true outcome. It is important to note that choosing an appropriate measure is not a straightforward process and depends strongly on the specific task being performed and the desired behaviour of the system. For instance, in classification, if the model owner is interested in an accuracy, we commonly choose the equivalent *error rate* as a performance measure defined as $L(\hat{y}, y) = \mathbb{1}[\hat{y} = y]$ where $\hat{y}$ is a predicted label from a machine learning model given an example x and y is its true label. The error rate (also called 0-1 loss) is 0 if the model correctly classifies a sample and 1 if it is not. In regression tasks and for continuous label spaces $\mathcal{Y}$, we typically use the mean squared error (MSE) given by $L(\hat{y}, y) = (\hat{y} - y)^2$. On the other hand, if the model owner's task is to make fair predictions that do not discriminate against any sensitive group and prevent algorithmic bias (e.g. for credit scores), accuracy or mean squared error may not be the most appropriate performance measure and fairness metrics such as equalized odds or demographic parity may be more relevant. In many cases, the right performance metric is inherently hard to define such as in large language models where incorporating human feedback has been invaluable. As we will see in the next chapters, choosing a suitable performance measure is not only crucial for training a model to perform the desired task, but *the specific properties of a chosen performance measure can also lead to vastly different solutions for data valuation*.

2.1.3 *Validation and Test Set*

One of the central challenges of machine learning is to train a predictor $f : \mathcal{X} \to \mathcal{Y}$ that can *generalize* to unseen examples, meaning it can perform well on previously unobserved samples and not just the data it was trained on. Supervised learning turns the task of finding this predictor into an optimization problem called empirical risk minimization (ERM) using the available data set and a given loss function. The risk is equal to the expected value of a loss function and since the number of training examples is finite, we can't directly compute the risk. Instead, we use the empirical risk as a proxy of the risk by averaging the loss function over the available data points. Give a set of labeled data points and a class of functions $\mathcal{F} \subseteq \mathcal{X} \to \mathcal{Y}$, we can find a predictor f by solving

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), y_i) \tag{2.1}$$

ML practitioners then typically divide their dataset into a training set, a validation set and a test set. The training loss is used for training the model, while the validation loss is used to choose the best possible hyperparameters for the training. Lastly, the test loss is used for evaluating the overall performance of the predictor. As the value of data is often associated to improving the learning performance, such as validation or test accuracy, of a predictor, this introduces *a close link between the value of a data point in the training set and the available data in the validation and test set*.

2.2 A Primer on Cooperative Game Theory

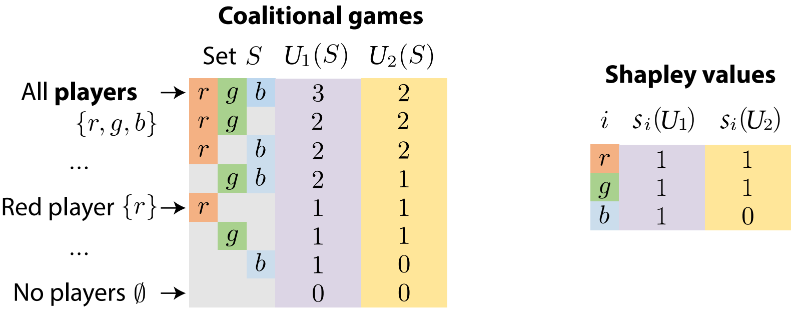
Game theory, introduced by von Neumann and Morgenstern's Theory of Games and Economic Behavior (Neumann and Morgenstern, 1947), is the study of mathematical models to analyze strategic interactions among rational players and has applications in a variety of fields, including economics, political science, psychology, and biology. While non-cooperative game theory (Nash, 1951) studies conflicting interactions and competition between players, cooperative game theory instead focuses on studying cooperation between players. In this thesis, we will restrict ourselves to cooperative game theory as a natural way to analyze the behaviors of coalitions formed by game players.

2.2.1 *Cooperative game setting*

Formally, a cooperative (or coalition) game (see Figure 2.1) is defined by a pair (I, U), where $I = \{1, \ldots, N\}$ denotes the *set of all players* and $U : 2^N \rightarrow \mathbb{R}$ is the *utility function*, which maps each possible coalition to a real number to the usefulness of the subset to the coalition. 2^N represents the power set of N, i.e., the collection of all subsets of N, including the empty set and N itself. To make these concepts more concrete, consider a hypothetical company with N employees that generates profit based on the set of employees who choose to work on a given day. Let $U(S)$ be the profit generated by a set of employees S. The natural question that arises is how to fairly compensate the employees for their contribution to the company's profit?

2.2.2 *The Shapley value*

Since any arbitrary mapping between player sets and outcomes is possible, how to assign individual compensations can be unclear. The Shapley value (Shapley, 1953) is a classic method in cooperative game theory to distribute the total gains generated by the coalition of all players, and has been applied to problems in various domains, ranging from economics (Gul, 1989), counter-terrorism (T. Michalak et al., 2013; Lindelauf, Hamers, and Husslage, 2013), environmental science (Petrosjan and Zaccour, 2003), to ML (Cohen, Ruppin, and Dror, 2005). Continuing with our example of the hypothetical company, let us now assume that we have information about the profit generated by all possible subsets of employees, for instance,



Coalitional games

	Set S			$U_1(S)$	$U_2(S)$
All **players** →	r	g	b	3	2
$\{r, g, b\}$	r	g		2	2
	r		b	2	2
…		g	b	2	1
Red player $\{r\}$→	r			1	1
		g		1	1
…			b	1	0
No players $\emptyset$ →				0	0

Shapley values

i	$s_i(U_1)$	$s_i(U_2)$
r	1	1
g	1	1
b	1	0

FIGURE 2.1: Example of the Shapley values of two coalitional games. There are three players: r,g and b. Each game is defined through its utility U_i.

through past balance sheets. Shapley values assign a payout to an individual i by calculating a weighted average of the profit increase when i works with group S versus when i does not work with group S (the marginal contribution). Averaging this difference over all possible subsets S to which i does not belong $S \subseteq I \setminus \{i\}$, we arrive at the definition of the Shapley value:

$$\overbrace{s_i(U)}^{i\text{'s Shapley value}} = \sum_{S \subseteq I \setminus \{i\}} \underbrace{\frac{1}{\binom{N-1}{|S|}}}_{S\text{'s weight}} \overbrace{\left[U(S \cup \{i\}) - U(S)\right]}^{i\text{'s marginal contribution}} \tag{2.2}$$

The formula in (2.2) can also be stated in the equivalent form:

$$s_i = \frac{1}{N!} \sum_{\pi \in \Pi(S)} \left[U(P_i^{\pi} \cup \{i\}) - U(P_i^{\pi})\right] \tag{2.3}$$

where $\pi \in \Pi(S)$ is a permutation of individuals and P_i^{π} is the set of individuals which precede individual i in π. Intuitively, imagine all individuals to arrive in a random order, and that every individual i receives their marginal contribution with respect to the existing current group. If we average these contributions over all the possible orders of individuals, we obtain s_i.

2.2.3 *Properties of the Shapley Value*

Although, there exist other methods within cooperative game theory for valuing the contributions of players, such as the core and nucleolus, the Shapley values are the only method that satisfies some very basic and desirable properties. Typically these properties are given as four axioms, where the Shapley values *uniquely* satisfies all four axioms:

Efficiency: The sum of the Shapley values of all players equals the value of the grand coalition and thus all the gain is distributed among the players:

$$U(I) = \sum_{i \in I} s_i \tag{2.4}$$

Symmetry: Two players who are identical with respect to what they contribute to a utility should have the same value:

$$U(S \cup \{i\}) = U(S \cup \{j\}), \forall S \subseteq I \setminus \{i, j\}, \text{ then } s_i = s_j \tag{2.5}$$

Null effects: Players with zero marginal contributions to all subsets of the dataset receive zero payoff:

$$s_i = 0 \text{ if } U(S \cup \{i\}) = U(S) \text{ for all } S \subseteq I \setminus \{i\} \tag{2.6}$$

Linearity: The values under multiple utilities sum up to the value under a utility that is the sum of all these utilities:

$$s(U_1, i) + s(U_2, i) = s(U_1 + U_2, i) \text{ for } i \in I \tag{2.7}$$

Alternative axiomatizations have been put forward and, i.e. (Young, 1985) showed that it is possible to replace the null player axiom (2.6) and the linearity axiom (2.7) by the monotonicity axiom.

Monotonicity: A payout to a player depends only on his marginal contributions and monotonically:

$$\text{if } \forall S \subseteq I \setminus \{i\} : U_1(S \cup \{i\})) - U_1(S) \geq U_2(S \cup \{i\}) - U_2(S)$$

$$\text{then } s_i(U_1) \geq s_i(U_2)$$

Below we will explore further how these game theory properties can be translated to machine learning and specifically the problem of valuing data.

2.3 Related Work

2.3.1 *Data Valuation*

There have been several proposed strategies for valuing data, which has inspired or build upon the work presented in this dissertation. For clarity, we broadly classify the various strategies into seperate categories:

(a) Leave-one-out and influence functions

(b) Principled cooperative game theory-based approaches; these notably include strategies using the Shapley value, the (least) core, and the Banzhaf value

(c) Reinforcement learning

(d) Deep Neural Networks

(e) Data-centered

In the following, we review related work and group them according to these categories (see Table 2.1 for an overview). For a more in-depth and technical analysis of these strategies, interested readers can refer to the study by (Sim, X. Xu, and Low, 2022). This work thoroughly evaluates existing data valuation approaches in terms of their properties and desired outcomes, while also highlighting the current challenges and future research directions. For completeness, we have also included our work in Table 2.1.

LEAVE-ONE-OUT AND INFLUENCE FUNCTIONS Leave-one-out is a simple approach to data valuation that calculates the value of each sample as its marginal utility, i.e., the performance change caused by excluding that sample from training (Sim, X. Xu, and Low, 2022; Jia, F. Wu, et al., 2021). However, this method has several fundamental problems including that it is prohibitively expensive to compute for large datasets or complex models, as

Category	Reference	Year	Utility type	Proposed Method
Cooperative Game Theory	(Jia, Dao, B. Wang, Hubis, Hynes, et al., 2019)	2019	task-dependent	Shapley value (in Chapter 3)
	(Ghorbani and Zou, 2019)	2019	task-dependent	Shapley value
	(Frye, Rowat, and Feige, 2020)	2019	task-dependent	Asymmetric Shapley value
	(Ohrimenko, Tople, and Tschiatschek, 2019)	2019	task-dependent	Robust Shapley value
	(Ghorbani, Kim, and Zou, 2020)	2020	task-dependent	Distributional Shapley value
	(Jia, Dao, B. Wang, Hubis, Gurel, et al., 2020)	2020	task-dependent and learning-agnostic	KNN-Shapley (in Chapter 4)
	(Sim, Y. Zhang, et al., 2020)	2020	task-dependent	Shapley value and information gain
	(Okhrati and Lipani, 2021)	2021	task-dependent	Owen Sampling for Shapley value
	(Yan and Procaccia, 2021)	2021	task-dependent	Least core
	(X. Xu, L. Lyu, et al., 2021)	2021	task-dependent	Cosine gradient Shapley value
	(Kwon and Zou, 2022)	2022	task-dependent	Beta Shapley
	(Schoch, H. Xu, and Ji, 2022)	2022	task-dependent	CS-Shapley
	(J. T. Wang and Jia, 2023a)	2023	task-dependent	Banzhaf value
Leave-One-Out and Influence Functions	(Koh and Liang, 2017)	2017	task-dependent	Influence functions
Reinforcement Learning	(Yoon, Arık, and Pfister, 2019)	2019	task-dependent and learning-agnostic	Data Valuation using Reinforcement Learning (DVLR)
Deep Neural Networks	(Z. Wu, Shu, and Low, 2022)	2022	learning-agnostic	Data valuation at initialization (DaVinz)
Data-Centered	(X. Xu, Z. Wu, et al., 2021)	2021	learning-agnostic	Robust volume
	(Just et al., 2023)	2023	learning-agnostic	Learning-agnostic data valuation (LAVA)
Energy-Based	(Bian et al., 2022)	2022	task-dependent	Variational Index

Table 2.1: Overview of proposed data valuation strategies in literature. The publications related to this dissertation is marked with a reference to the corresponding chapter.

retraining models on every single data subset to exactly compute the leave-one-out error scales linearly with the number of samples (Just et al., 2023). To address the computational expense of computing the leave-one-out error for large datasets or complex models, (Koh and Liang, 2017) propose using influence functions to efficiently estimate the leave-one-out and identify the training points most responsible for a model's given predictions. However, this approach inherits some of the other fundamental problems with leave-one-out, such as inaccurate utility estimations. In contrast, other strategies, such as cooperative game theoretic ones, may not have these issues (Sim, X. Xu, and Low, 2022).

(Koh, Ang, et al., 2019) extend their research to study the use of influence functions in measuring the effects of large groups of samples, rather than just individual points. This allows for more efficient computation of influence functions, particularly in the context of data valuation. The authors find that their predicted group effect correlates well with the actual effect in practice, despite some large errors. However, they caution that this correlation may not hold in general and may be due to some unique properties of real-world datasets.

In the context of data valuation, it is important to note that influence functions, like most other data valuation strategies, depend upon the choice of learning algorithm. While influence functions are well-defined for linear models due to their convexity, deep learning models often use non-convex loss functions, making their influence functions more complex. To investigate this, (Basu, Pope, and Feizi, 2021) conducted an empirical study on the use of influence functions in a deep learning context. Their findings suggest that influence functions can be accurate for shallow networks, but their performance degrades in deeper networks. Furthermore, the accuracy of the influence estimates may depend on the specific test points examined and how the network was trained.

SHAPLEY VALUE Concurrent to the work in this dissertation, Ghorbani et al. (Ghorbani and Zou, 2019) developed two heuristics to accelerate the estimation of the Shapley value for complex learning algorithms, such as neural networks. One is to truncate the calculation of the marginal contributions as the change in performance by adding only one more training point becomes smaller and smaller. Another is to use one-step gradient to approximate the marginal contribution. The authors also demonstrate the use of the approximate Shapley value for outlier identification and informed acquisition of new training data. However, their algorithms do not provide any guarantees on the approximation error, thus limiting its viability for practical data valuation. It should be noted that Data Shapley is sensitive to changes in input and lacks stability in terms of "Lipschitzness" with respect to data contributors, as pointed out by (Sim, X. Xu, and Low, 2022). This implies that the Data Shapley must be recalculated when new data contributors are introduced or removed.

The work of (Ghorbani, Kim, and Zou, 2020) address this lack of stability and need for recomputing by proposing distributional Shapley. Unlike the conventional Shapley value that defines the value of a point based on a fixed dataset, distributional Shapley defines the value of a point in the context of an underlying data distribution. This approach guarantees stability even when there are minor variations in the data or underlying distribution. The authors also proposed an efficient sampling-based algorithm to compute the distributional Shapley value. This algorithm employs a separate regression model to predict distributional Shapley values for unseen data and provides robust approximation guarantees.

(Kwon, Rivas, and Zou, 2021) improved the distributional Shapley value by creating practical and manageable analytic expressions for its estimation in linear regression, binary classification, and non-parametric density estimation. These expressions enabled the development of new algorithms that accurately and efficiently estimate the distributional Shapley value, enhancing its practicality and applicability. By reducing the computational complexity associated with existing algorithms, the proposed methodology broadens the scope of the distributional Shapley value and improves its efficiency.

To overcome data replication issues, (Ohrimenko, Tople, and Tschiatschek, 2019) proposed a modified version of the Shapley value that guarantees robustness to data replication. However, their approach comes at a cost: it sacrifices the efficiency axiom (2.4).

To reduce the exponential complexity of the Shapley value computation, (Okhrati and Lipani, 2021) suggest using a multilinear extension technique from game theory. Their proposed Owen sampling algorithm provides a computationally efficient approach for estimating Shapley values, surpassing the Castro sampling method (Castro, Gómez, and Tejada, 2009) by reducing the estimator's variance and generating more precise and efficient estimates. Importantly, their technique can be applied to all learning algorithms.

Aside from its high computational complexity, the Shapley value has another limitation. It cannot accommodate causal knowledge, as emphasized by (Frye, Rowat, and Feige, 2020). To address this limitation, the authors propose a less restrictive framework called Asymmetric Shapley Values (ASVS), which can incorporate causal knowledge. This new framework is built on a rigorous set of axioms, enabling the integration and respect of any causal structure present in the data. The ASVS approach does not require a complete causal graph underlying the data, making it more practical in real-world applications. The authors demonstrate the effectiveness of ASVS when causal knowledge is present, providing empirical evidence of its potential.

In their work, (Sim, Y. Zhang, et al., 2020) present a data valuation approach that employs separate ML models as rewards based on the Shapley value and information gain on model parameters given the data. The authors establish various conditions for incentives, such as Shapley fairness, stability, individual rationality, and group welfare, which are appropriate for their model reward scheme's freely replicable nature. Their reward scheme also includes an adjustable parameter that allows a trade-off between these conditions while maintaining fairness since they cannot all be achieved simultaneously. Notably, the proposed data valuation method does not require any assumptions about the distribution of test samples.

The authors of (Schoch, H. Xu, and Ji, 2022) introduce CS-Shapley, a new valuation method that builds upon the Shapley value to differentiate between training samples that belong to either the in-class or out-of-class distribution. Value functions based on predictive accuracy cannot distinguish between samples that benefit their own class accuracy and those that harm it. CS-Shapley is able to overcome this limitation by decomposing the value function into two functions that emphasize in-class accuracy and introduce a discount for out-of-class accuracy. The proposed approach yields a single function that satisfies two desirable properties for evaluating data values in classification, and empirical results demonstrate its effectiveness and transferability to various models and applications.

The cosine gradient Shapley value, introduced by (X. Xu, L. Lyu, et al., 2021), provides a fair evaluation of each agent's parameter update in federated learning. This method does not require a separate test set and is therefore validation-free. The authors also propose a gradient reward mechanism during training to ensure fairness and demonstrate its effectiveness in experiments.

(Kwon and Zou, 2022) introduce Beta Shapley, a generalization of Data Shapley by relaxing the efficiency axiom. Instead, the authors suggest that ranking is sufficient in most machine learning settings. They consider all semi-values that satisfy all other axioms except the efficiency axiom and use the Beta function to choose weights for the weighted sums of marginal utilities over all data subsets. This results in the Beta Shapley method, for which the authors provide efficient algorithms for estimation.

LEAST CORE (Yan and Procaccia, 2021) challenge the use of the Shapley value as the standard criterion in data valuation and propose the core, another game theoretic solution concept that satisfies group fairness, as an alternative. They demonstrate that a relaxation of the core, called the least core, can be approximated using a Monte Carlo algorithm, making it computationally tractable. The authors also show empirically that the approximate least core can outperform the approximate Shapley value in certain settings. (Tianhao Wang, Yang, and Jia, 2022) introduce a general framework to enhance the efficiency of strategies that use sampling-based Shapley value or least core estimation heuristics. Their proposed approach involves learning to estimate the performance of a learning algorithm on unseen data combinations, known as data utility learning. This significantly reduces the need for retraining and leads to cheaper Shapley value and least core estimations. The authors derive theoretical bounds on the error of data utility learning and demonstrate empirically that it improves the accuracy of these estimations.

BANZHAF VALUE In the context of noisy models, (J. T. Wang and Jia, 2023a) investigate the robustness of data valuation notions. They find that randomness introduced by stochastic gradient descent and other factors can lead to inconsistent data value rankings between model runs, particularly for high variance data subsets, when using data value notions such as the Shapley value. To address this issue, the authors propose a concept called safety margin as a robustness measure for data value notions. They show that the Banzhaf value, another solution concept from game theory, achieves the largest safety margin among all semi-values, including the Shapley value. The authors also develop an efficient algorithm for Monte Carlo approximation of the Banzhaf value and highlight it as a promising alternative data value notion.

REINFORCEMENT LEARNING Most data valuation strategies separate the training of the learning algorithm from the data valuation process. However, (Yoon, Arık, and Pfister, 2019) proposed a different approach called data valuation reinforcement learning (DVRL) that combines learning of data values and the predictor model. This joint learning method uses a data value estimator to estimate the values of individual samples and select the most valuable ones for training the predictor model. By incorporating data valuation into the training procedure, the strategy enables the predictor model to optimize for high-value samples, leading to better model performance and data valuation estimates. The use of reinforcement learning in DVRL is motivated by the non-differentiability of the sample selections in the data value estimator, which makes gradient-based methods impractical.

DEEP NEURAL NETWORKS (Z. Wu, Shu, and Low, 2022) present a new approach for efficient data valuation with large and complex deep neural networks (DNNs) that is both validation-based and training-free. The authors develop a domain-aware generalization bound for DNNs, called DaVinz, which can characterize the performance of the model without the need for training. This generalization bound is used as the scoring function in the data valuation process, while alternative techniques such as the Shapley value are employed as the utility or valuation function. By using DaVinz, it is possible to evaluate data valuation methods with DNNs in a more cost-effective manner, while avoiding uncertainties arising from hyperparameter selection and other handcrafted design choices in model training. Compared to other training-free methods, such as the validation-free robust-volume based approach proposed by (X. Xu, Z. Wu, et al., 2021), DaVinz is well-suited for high-dimensional input and can retain useful information from the validation dataset. This is particularly beneficial in scenarios where data consumers prefer datasets that result in better performance as measured on a validation dataset.

DATA-CENTERED (Just et al., 2023) propose a novel framework for LAVA that can efficiently estimate data values independent of downstream learning algorithms, removing the dependency on design choices of the learning

algorithm in the utility function. They use a proxy for the validation performance associated with a training set, based on the class-wise Wasserstein distance between the two datasets, which measures the minimum cost of transforming one probability distribution to another. The authors show that the class-wise Wasserstein distance provides an upper bound on the validation performance for any given model under certain Lipschitz conditions. They then use the calibrated gradients of the optimal transport distance to value individual samples, which can be obtained for free in off-the-shelf solvers for Wasserstein distance. The authors demonstrate that their framework can significantly improve the performance of various use cases related to detecting low-quality data while being orders of magnitude faster than state-of-the-art methods. However, the authors note that LAVA may not be suitable for tasks that aim for goals beyond accuracy, such as fairness or equitability.
(X. Xu, Z. Wu, et al., 2021) suggest using data diversity via robust volume of the data matrix as a means of quantifying data value. This method is agnostic to both task and learning algorithm and is based solely on the trace of the feature matrix inner product, making it validation-free. The method also has theoretical guarantees on replication robustness, which discourages data contributors from duplicating data. However, the method suffers from the curse of dimensionality, making it less effective in high-dimensional spaces. Additionally, this approach may disregard valuable information in the validation set and cannot detect labeling errors. The Shapley value is still used to measure the value of individual samples.

ENERGY-BASED The valuation of data is often performed using cooperative game theory, with solution concepts such as the Shapley value, least core, or Banzhaf value. However, (Bian et al., 2022) have recently proposed a novel approach that generalizes these concepts and connects them with energy-based models. By using mean-field variational inference, the authors can recover classical game-theoretic valuation criteria through a fixed-point iteration for maximizing the evidence lower bound objective. They also introduce the Variational Index, which is defined as the lowest conceivable decoupling error among a trajectory of variational valuations obtained by running the fixed-point iteration for multiple steps. This index satisfies three important game-theoretic axioms, including null player, marginalism, and symmetry. Finally, the authors show that the Variational Index has a lower decoupling error and better valuation performance than other variational valuations in some scenarios.

2.3.2 *Further related work*

MORE DATA PRICING MODEL Before withdrawn by Microsoft, the Azure Data Marketplace adopted a subscription model that gave users access to a certain number of result pages per month (Koutris et al., 2015), SkyFi sells access to satellite imagery, Xignite sells financial datasets and prices data based on the data type, size, query frequency, etc. There is rich literature on query-based pricing (Koutris et al., 2015; Koutris et al., 2013; Koutris et al., 2012; Deep, Koutris, and Bidasaria, 2017; Lin and Kifer, 2014; C. Li and Miklau, 2012; Upadhyaya, Balazinska, and Suciu, 2016), aimed at design pricing schemes for fine-grained queries over a dataset. In query-based pricing, a seller can assign prices to a few views and the price for any queries purchased by a buyer is automatically derived from the explicit prices over the views.

Koutris et al. (Koutris et al., 2015) identified two important properties that the pricing function must satisfy, namely, arbitrage-freeness and discount-freeness. The arbitrage-freeness indicates that whenever query Q_1 discloses more information than query Q_2, we want to ensure that the price of Q_1 is higher than Q_2; otherwise, the data buyer has an arbitrage opportunity to purchase the desired information at a lower price. The discount-freeness requires that the prices offer no additional discounts than the ones specified by the data seller. The authors further proved the uniqueness of the pricing function with the two properties, and established a dichotomy on the complexity of the query pricing problem when all views are selection queries.

Li et al. (C. Li and Miklau, 2012) proposed additional criteria for data pricing, including non-disclosiveness (preventing the buyers from inferring unpaid query answers by analyzing the publicly available prices of queries)

and regret-freeness (ensuring that the price of asking a sequence of queries in multiple interactions is not higher than asking them all-at-once), and investigated the class of pricing functions that meet these criteria.
Zheng et al. (Zheng et al., 2019) studied how data uncertainty should affect the price of data, and proposed a data pricing framework for mobile crowd-sensed data. Recent work on query-based pricing focuses on enabling efficient pricing over a wider range of queries, overcoming the issues such as double-charging arising from building practical data marketplaces (Koutris et al., 2013; Deep, Koutris, and Bidasaria, 2017; Upadhyaya, Balazinska, and Suciu, 2016), and compensating data owners for their privacy loss (C. Li, D. Y. Li, et al., 2017).

ML MODEL PRICING Due to the increasing pervasiveness of ML-based analytics, there is an emerging interest in studying the cost of acquiring data for ML. Chen et al. (L. Chen, Koutris, and A. Kumar, 2018; L. Chen, Koutris, and A. Kumar, 2017) proposed a formal framework to price ML model instances, wherein an optimization problem was formulated to find the arbitrage-free price that maximizes the revenue of a seller. The model price can be also used for pricing its training dataset. While the interaction between data analytics and economics has been extensively studied in the context of both relational database queries and ML, few works have dived into the vital problem of allocating revenues among data owners. (Koutris et al., 2012) presented a technique for fair revenue sharing when multiple sellers are involved in a relational query. By contrast, data valuation focuses on the revenue allocation for ML models. Raskar et al (Raskar et al., 2019) presented a taxonomy of data valuation problems for data markets and discussed challenges associated with data sharing. Specifically, the paper discussed intrinsic (e.g., data quality) vs. extrinsic (e.g., demand-supply) factors of data valuation as well as goal-dependent vs. goal-independent depending on whether or not there is a specific well-defined goal for the data purchase.

EFFICIENT ESTIMATION OF SHAPLEY VALUES. Originated from game theory, the Shapley value, in its most general form, can be #P-complete to compute (X. Deng and Papadimitriou, 1994). Efficiently estimating Shapley value has been studied extensively for decades. For bounded utility functions, (Maleki et al., 2013) described a sampling-based approach that requires $\mathcal{O}(N \log N)$ samples to achieve a desired approximation error in l_∞ norm and $\mathcal{O}(N^2 \log N)$ in l_2 norm. Bachrach et al. (Bachrach et al., 2008) also leveraged a similar approach but focused on the case where the utility function has binary outputs. By taking into account special properties of the utility function, one can derive more efficient approximation algorithms. For instance, (Fatima, Wooldridge, and Jennings, 2008) proposed a probabilistic approximation algorithm with $\mathcal{O}(N)$ complexity for weighted voting games. (T. P. Michalak et al., 2013) showed that for specific network games, the exact Shapley value can be computed efficiently.

VALUE OF PERSONAL DATA. The game-theoretic analysis of the value of personal data has been explored in (Chessa and Loiseau, 2017; Kleinberg, Papadimitriou, and Raghavan, 2001), which proposed a fair compensation mechanism based on the Shapley value. They derived the Shapley value under simple data utility models abstracted from network games or recommendation systems, while data valuation focuses on more complex utility functions derived from ML applications. In our case, the Shapley value no longer has closed-form expressions. We develop novel and efficient approximation algorithms to overcome this hurdle.

SHAPLEY VALUE FOR FEATURE ATTRIBUTION. Using the Shapley value in the context of ML is not new. For instance, the Shapley value has been applied to feature selection (Lundberg and S.-I. Lee, 2017; Cohen, Ruppin, and Dror, 2005; X. Sun et al., 2012; Mokdad et al., 2015; Sasikala, Balamurugan, and Geetha, 2015). While their contributions have inspired this work, many assumptions made for feature "valuation" do not hold for data valuation. As we will see, by studying the Shapley value tailored to data valuation, we can develop novel algorithms that are more efficient than the previous approaches (Maleki et al., 2013).

DATA IMPORTANCE There exist various methods to rank the importance of training data, which can also potentially be used for data valuation. For instance, influence functions (Koh and Liang, 2017), beyond their use for data valuation, is primarily used as an approximate the change of the model performance after removing a training point for smooth parametric ML models, a variant (Sharchilev et al., 2018) for non-parametric ones. Ogawa et al. (Ogawa, Suzuki, and Takeuchi, 2013) proposed rules to identify and remove the least influential data when training support vector machines (SVM) to reduce the computation cost. One can also construct coresets—weighted data subsets—such that models trained on these coresets are provably competitive with models trained on the full dataset (Dasgupta et al., 2009). However, unlike the Shapley value, these approaches do not satisfy the efficiency, fairness, and linearity properties simultaneously. Nevertheless, as the Shapley value becomes increasingly prevalent in data importance, its appropriateness for feature selection is called into question by (Fryer, Strümke, and Nguyen, 2021). The authors advise caution against the magical thinking that presenting the abstract general axioms of the Shapley value as "favorable and fair" may introduce. Additionally, the authors argue that the four axioms of efficiency, null player, symmetry, and additivity do not guarantee that the Shapley value is well-suited for feature selection and may even imply the opposite in some cases.

3

A PRINCIPLED FRAMEWORK FOR DATA VALUATION BASED ON THE SHAPLEY VALUE

> *I consider myself a mathematician. I never, never in my life took a course in economics.*
>
> — Lloyd Shapley

3.1 Introduction

Machine learning (ML) and data analytics is an increasingly common practice in science and business and with today's data-centric development pattern of ML applications, high-quality data has become increasingly valuable. For many large organizations, the data for building an ML model are often provided by multiple data contributors. For instance, a large internet enterprises analyze various users' data to improve product design, customer retention, and initiatives that help them earn revenue. On the other hand, particularly for smaller organizations and entities, acquiring necessary high-quality data to build accurate ML models can often be a non-trivial task. However, even though a single small organization may have limited data, it is still possible to develop better, high-quality ML models by training them on aggregated data from a combination of multiple parties.

ML MODELS ARE BUILT COLLABORATIVELY. Collaborative ML approaches which leverages data from multiple organizations and working together to develop models are gaining popularity (Richter and Slowinski, 2019; Scaria et al., 2018). These approaches are more accurate and robust than those developed using a single data source and allows organizations to tap into the collective knowledge and expertise of different entities, leading to improved results. Examples of collaborative ML can be seen in precision agriculture, a farmer with limited land area and sensors can combine his collected data with the other farmers to improve the modeling of the effect of various influences (e.g., weather, pest) on his crop yield (Claver, 2019) Such data sharing also benefits other application domains, including real estate in which a property agency can pool together its limited transactional data with that of the other agencies to improve the prediction of property prices (Conway, 2018). In healthcare, a hospital or healthcare firm whose data diversity and quantity are limited due to its small patient base can draw on data from other patients, hospitals and firms to improve the prediction of some disease progression (e.g., glaucoma) (Center for Open Data Enterprise, 2019). This collaboration can be encouraged by a government agency, such as the National Institute of Health in the United States. Moreover, in many real-world applications, the datasets that support queries and ML are often contributed by *multiple individuals* without their knowledge. One example is that complex ML tasks such as foundation models (e.g Stable Diffusion (Rombach et al., 2021), GPT (OpenAI, 2023; T. B. Brown et al., 2020)) training often relies on massive crowdsourcing efforts i.e. CommonCrawl, LAION-5B (Schuhmann et al., 2022). A recent public movement among writers and artists has emerged to advocate for the fair distribution of revenue generated from queries and machine learning models among the various contributors of data.

A FAIR DISTRIBUTION OF VALUE. Our motivation for valuing data first arose by a system we were building at the time together with one of the largest hospital in the US. In the system (called Sterling, further described in Chapter 5), patients submit part of their medical records onto a “data market,” and analysts pay a certain amount of money to train a ML model on patients’ data. Naturally, the question arises how to distribute the payment from analysts back to the patients, or specifically "how much is the patient’s data worth?". Answering the question of how to distribute payments from analysts back to patients in a data market has significant implications. One of the most important benefits is the fair distribution of profits among multiple data contributors. In many data markets, multiple individuals or entities contribute data, and it is essential to ensure that each party receives equitable share of the profits generated from the data.

Another potential benefit is the ability to determine prospective compensation when data breaches happen. If a data breach occurs, patients may be entitled to compensation for any harm caused by the breach. Having a clear understanding of how payments should be distributed among data contributors can help to determine how compensation should be divided among those affected.

DESIDERATA FOR DATA VALUATION. Recalling our toy example in Figure 1.1, we believe a comprehensive framework for data valuation should have desired properties (see Figure 3.1):
1) Data points are often only valuable in (large) *combinations* with other data points within the set. 2) The overall profit is typically *task-dependent* (for instance, a classifier for rare diseases may yield greater returns than one for distinguishing between dogs and fish). 3) The quality of individual data points can vary, necessitating *equitable* valuations. Lastly, 4) since data points can be frequently repurposed for multiple tasks, any generated value should be *cumulative*. Note that some desiderata are derived from the specifics of ML-based utilities (e.g. combinatorial and cumulative). Therefore, we aim to find a valuation strategy with the following challenging desiderata:

Challenge 1

> *What is a principled framework for data valuation that is combinatorial, task-dependent, cumulative and equitable?*

In this chapter, we propose a natural way of tackling the data valuation problem by adopting a game-theoretic viewpoint, where each data contributor is modeled as a player in a coalitional game and the usefulness of data from any subset of contributors is characterized via a utility function. As we have seen in Chapter 2 the Shapley value (2.2) defines a unique profit allocation scheme that satisfies a set of properties with appealing real-world interpretations for machine learning, such as rationality (through satisfying axiom 2.4), fairness (through axioms 2.6 and 2.5) and decentralizability (through axiom 2.7). We demonstrate that these axioms can also be connected to the desired properties mentioned above.

	Assumptions	**Techniques**	**Complexity**		**Approximation**
			incrementally trainable models	**otherwise**	
Existing	Bounded utility	Permutation sampling	$\mathcal{O}(N \log N)$ model training and $\mathcal{O}(N^2 \log N)$ eval	$\mathcal{O}(N^2 \log N)$ model training and eval	(ϵ, δ)
Application -agnostic	Bounded utility	Group testing	$\mathcal{O}(N(\log N)^2)$ model training and eval	$\mathcal{O}(N(\log N)^2)$ model training and eval	(ϵ, δ)
	Monotone utility & sparse value	Compressive permutation sampling	$\mathcal{O}(\log\log N)$ model training and $\mathcal{O}(N \log\log N)$ eval	$\mathcal{O}(N \log\log N)$ model training and eval	(ϵ, δ)
ML-specific	Stable learning	Uniform division	$\mathcal{O}(1)$ computation		$(\epsilon, 0)$
	Smooth utility	Influence function	$\mathcal{O}(N)$ optimization routines		Heuristic

TABLE 3.1: Summary of Technical Results. N is the number of data points.

EXPONENTIAL COMPUTE TIME. Despite the desirable properties of the Shapley value, computing the Shapley value is known to be expensive; the number of utility function evaluations required by the exact Shapley value calculation grows exponentially in the number of players. Even worse, for ML tasks, evaluating the utility function itself (e.g., testing accuracy) is already computationally expensive, as it requires training a model. Due to the computational challenge, the application of the Shapley value to data valuation has thus far been limited to stylized examples, in which the underlying utility function of the game is simple and the resulting Shapley value can be represented as a closed-form expression (Chessa and Loiseau, 2017; Kleinberg, Papadimitriou, and Raghavan, 2001). The state-of-the-art method to estimate the Shapley value for a black-box utility function is based on Monte Carlo simulations (Maleki et al., 2013), which still requires evaluating ML models for $\mathcal{O}(N^2 \log N)$ many times in order to compute the Shapley value of N data points and is thus clearly impracticable. Thus, we would like to address another challenge:

Challenge 2

> *How can we efficiently estimate the Shapley value for general utilties while achieving the same performance guarantee as the state-of-the-art method?*

3.1.1 *Contributions*

The contributions in this chapter are products of joint work with many collaborators and have been previously published in AISTATS under

Ruoxi Jia*, **David Dao***, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Guerel, Bo Li, Ce Zhang, Dawn Song, Costas J. Spanos. "Towards Efficient Data Valuation Based on the Shapley Value". *AISTATS*. 2019.

CONTRIBUTION 1: A PRINCIPLED FRAMEWORK FOR DATA VALUATION. Addressing the first challenge, we present a principled framework for data valuation grounded in the Shapley value. This framework uniquely tackles our desired criteria for data valuation. It is combinatorial, considering all potential subsets; task-dependent, as it relies on a specific utility function tasked by the service provider; cumulative, due to the linearity property of the Shapley value; and equitable, thanks to Shapley fairness. The principled framework enables us value a data point by calculating its respective Shapley values of a cooperative game in which the data contributors serve as players.

CONTRIBUTION 2: ALGORITHMS FOR EFFICIENT ESTIMATION OF THE SHAPLEY VALUE. To efficiently estimate the Shapley value, we first address *Challenge 2* from a *theoretical* perspective and provide following contributions:

C2.1 Approximation for Bounded Utilities. In the most general case we only make a bounded utility assumption. We show that, to approximate the Shapley value of N data points with provable error guarantees, it is possible to design an algorithm with $\mathcal{O}(N(\log N)^2)$ model evaluations based on group testing (Du, F. K. Hwang, and F. Hwang, 2000). The idea is to share the information we get from a single utility evaluation across all data points, as opposed to treating different data points independently as in existing approaches. In cases where the utility function cannot be efficiently evaluated for new data points in an incremental manner, and when the number of training data points is large, our group testing-based approach has been shown to require significantly fewer model evaluations than the state-of-the-art sampling-based approach (Maleki et al., 2013).

C2.2 Approximation for Monotone Utilities and Sparse Value. Moreover, if it is reasonable to assume that the utility function is monotone and the Shapley value is "sparse" in the sense that only few data points have significant values, then we are able to further reduce the number of model training to $\mathcal{O}(\log \log N)$, when the model can be incrementally maintained.

Although there may be theoretical improvements, retraining models multiple times can still be impractical for large datasets and ML models. Thus we introduce two *practical* Shapley value estimation algorithms specific to ML tasks by introducing various assumptions on the utility function:

C2.3 Uniform Value Division for Stable Algorithms. For stable learning algorithms, such as many norm regularized models (Bousquet and Elisseeff, 2002), the model only changes slightly when the training data is changed slightly. Intuitively, stable algorithms are not sensitive to individual training points and therefore all training points should have very similar values. Our theoretical results show that uniform data value division is a fairly good approximation to the Shapley value for stable learning algorithms.

C2.4 Using the Influence Function Heuristic. For a smooth utility function one can use influence functions (Koh and Liang, 2017) as a proxy to the Shapley value by assuming (1) the value of a data point depends only on its marginal contribution to the data subset that contains all other points, and (2) the influence function is a good *local* approximation to the change of the utility caused by adding one more training data point. Whether this heuristic is acceptable is application-specific.

It is worth noting that the algorithms in (C2.1) and (C2.2) are agnostic to the context wherein the Shapley value is computed; hence, they are also useful for the applications beyond data valuation. Furthermore for algorithms (C2.3) and (C2.4), the efficiency does not come for free. (C2.3) relies on the stability of a learning algorithm, which is difficult to prove for complex ML models, such as deep neural networks. The compromise that we have to make in (C2.4) is that the resulting Shapley value estimates no longer have provable guarantees on the approximation error. Filling the gap between theoretical soundness and practicality is important future work.

Table 3.1 summarizes the contributions of this chapter. In the rest of the chapter, we will elaborate on the idea and analysis of these algorithms, and further use them to compute the data values for various benchmark datasets.

3.1.2 *Overview*

In the following, we first introduce our principled framework for data valuation based on the Shapley value in Section 3.2. Next we introduce a repository of efficient approximation algorithms by studying general properties of ML-based utilities in Section 3.3. We evaluate our proposed algorithms through a set of experiments in Section 3.4. Lastly, we discuss our findings in Section 3.5 and provide a short summary in Section 3.6.

3.2 A Principled Framework for Data Valuation

ML AS A COOPERATIVE GAME Consider two types of agents that interact in a data marketplace: the sellers (or data contributors) and the buyer (or data consumers). Sellers provide training data instances, each of which is a pair of a feature vector and the corresponding label. The buyer is interested in analyzing the training dataset aggregated from various sellers and producing an ML model, which can predict the labels for unseen features. The buyer pays a certain amount of money which depends on the utility of the ML model. Our goal is to distribute the payment fairly between the sellers. A natural way to tackle the question of profit allocation is to view ML as a cooperative game and model each seller as a player. This game-theoretic viewpoint allows us to formally characterize the "power" of each seller and

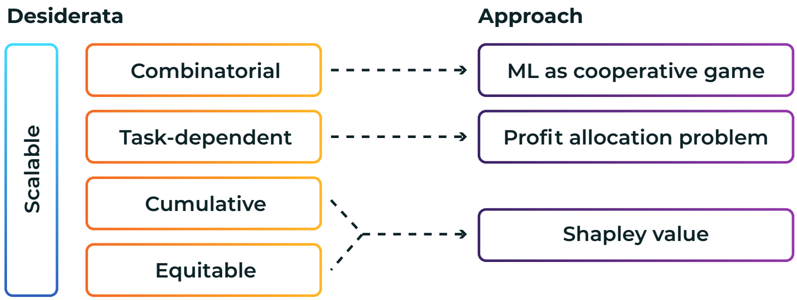


FIGURE 3.1: Overview of our data valuation framework. A data valuation framework should be combinatorial, task-dependent, cumulative and equitable. The Shapley value provides a uniquely maps all desiderata. Tackling the last desiderata, scalability, will be the main focus of the next chapters.

in turn determine their deserved share of the revenue. Let us first identify the necessary components of the cooperative game (see Figure 3.2):

Data D**.** Both the number M of individual sellers and the total number of data points of the aggregated dataset N will affect the cost of calculating the data valuation. For ease of exposition, we will assume that every seller contributes one data instance in the training set; therefore $M = N$. This is usually not the case as data contributors can, in general, contribute multiple data points. Later in Chapter 4, we will discuss the extension to the case where a seller contributes multiple data instances.

Learning Algorithm A**.** The worth of data is not static and can fluctuate depending on the properties of the learning algorithm $A(\cdot)$ that maps a dataset D onto a ML model. The choice of a learning algorithm can also impact the subsequent utility function U, which in return will have effects on computing the value.

Utility U**.** The utility $U_A(D) \triangleq U_m(A(D))$ provides a higher performance score for more desirable models. One intuitive U is through training a learning algorithm A and evaluating it through a performance measure U_m. However, it is also possible to formulate $U(\cdot)$ in a *learning-agnostic* manner

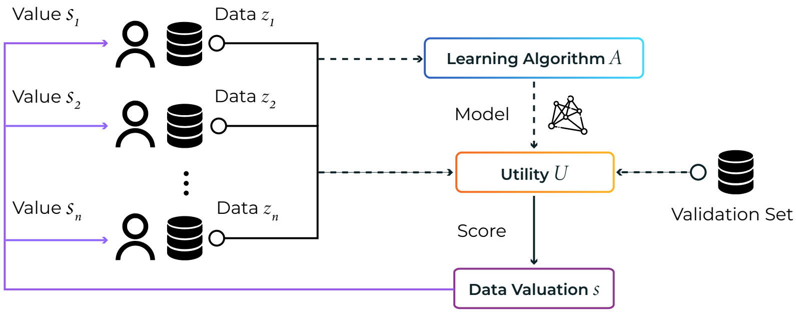


FIGURE 3.2: An overview of the individual components of data valuation and their relationships. Data valuation aims to map a value s to data z given a learning algorithm, a utility (usually validation accuracy), and a data valuation strategy (we will focus on the Shapley value in this dissertation).

by evaluating directly on dataset D i.e. $U(D)$. For the scope of this chapter, we will restrict ourselves to the case of *task-dependent* data valuation. Thus, for the rest of this chapter, we will leave out the dependency on A in cases where the learning algorithm is self-evident, and use $U(\cdot)$ instead.

SHAPLEY-BASED DATA VALUATION. Consider a dataset $D = \{z_i\}_{i=1}^N$ containing data from N users. Let $U(S)$ be the utility function, representing the value calculated by the additive aggregation of $\{z_i\}_{i \in S}$ and $S \subseteq I = \{1, \cdots, N\}$. Note that for an ML task, we can write the utility function $U(S) \triangleq U_m(A(S))$, where $A(\cdot)$ is the learning algorithm and $U_m(\cdot)$ is a task-specific utility. Without loss of generality, we assume throughout the next Chapters $U_m(\dot)$ to be the validation loss and $U(\emptyset) = 0$. Our goal is to partition $U_{\text{tot}} \triangleq U(I)$, the utility of the entire dataset, to the individual users; more formally, we want to find a function that assigns to user i a number $s(U, i)$ for a given utility function U. We further suppress the dependency on U when the utility is self-evident and use s_i to represent the value allocated to user i.

Let us briefly recall the definition of the Shapley Value (Equation 2.2) from Chapter 2. Given a utility function $U(\cdot)$, the Shapley value for user i

is defined as the average marginal contribution of z_i to all possible subsets of $D = \{z_i\}_{i \in I}$ formed by other users:

$$s_i = \sum_{S \subseteq I \setminus \{i\}} \frac{1}{N\binom{N-1}{|S|}} \left[U(S \cup \{i\}) - U(S)\right]$$

SHAPLEY PROPERTIES FOR DATA VALUATION. The importance of the Shapley value stems from the fact that it is the *unique* value division scheme that satisfies the following desirable properties discussed in Chapter 2. To make it easier for the reader, we will provide the properties below and explain their practical implications for data valuation:

1. Efficiency: The value of the entire dataset is completely distributed among all users, i.e., $U(I) = \sum_{i \in I} s_i$.
2. Symmetry: Two data contributors who are identical with respect to what they contribute to a dataset's utility should have the same value. That is, if user i and j are equivalent in the sense that $U(S \cup \{i\}) = U(S \cup \{j\}), \forall S \subseteq I \setminus \{i,j\}$, then $s_i = s_j$.
3. Null property: Users with zero marginal contributions to all subsets of the dataset receive zero payoff, i.e., $s_i = 0$ if $U(S \cup \{i\}) = 0$ for all $S \subseteq I \setminus \{i\}$.
4. Linearity: The values under multiple utilities sum up to the value under a utility that is the sum of all these utilities: $s(U,i) + s(V,i) = s(U+V,i)$ for $i \in I$.

The *efficiency* axiom in Equation (2.4) states that any rational group of data contributors would expect to distribute the full yield of their coalition. The *symmetry* in Equation (2.5) and *null* axioms in Equation (2.6) are "fairness" properties. They requires that the names of the users play no role in determining the value, which should be sensitive only to how the utility function responds to the presence of a user's data. The *linearity* axiom in Equation (2.7) facilitates efficient value calculation when data is used for multiple applications, each of which is associated with a specific utility function. Furthermore, the properties of the Shapley value map to the desiderata for data valuation (see Figure 3.1):

1. Combinatorial: The Shapley value accounts for the interaction and combination of different data contributors by considering all possible coalitions and their utilities. This ensures that the value calculation reflects the joint contribution of data points in various configurations.
2. Task-dependent: The Shapley value relies on a utility function that quantifies the effectiveness of the data for a specific task. By adopting different utility functions for different tasks, the Shapley value remains flexible to incorporate task-specific requirements in the data valuation process.
3. Cumulative: The linearity axiom facilitates efficient value calculation when data is used across multiple applications with distinct utility functions. It allows decomposing a given utility function into a sum of utility functions and computing utility shares separately, thereby supporting the accumulation of value from various applications.
4. Equitable: The fairness properties, i.e., symmetry and null axiom, ensure that the data valuation process is equitable. The symmetry property guarantees that the value assigned to a data point is solely based on its impact on the utility function, regardless of the contributor's identity. The null property ensures that a data contributor with no impact on the utility function receives no value, which upholds fairness in the valuation process.

The fact that the Shapley value uniquely possesses these properties, combined with its flexibility to support different utility functions, leads us to employ the Shapley value to attribute the total gains generated from a dataset to each user.

3.3 Efficient Shapley Value Estimation

The challenge in adopting the Shapley value lies in its computational cost. Evaluating the exact Shapley value using Equation (2.2) involves computing the marginal utility of every user to every coalition, which is $\mathcal{O}(2^N)$. Even worse, in many ML tasks, evaluating utility *per se* (e.g., validation accuracy) is computationally expensive as it requires training an ML model. In this section, we present various efficient algorithms for approximating the Shapley value.

Definition 1. *We say that $\hat{s} \in \mathbb{R}^N$ is a (ϵ, δ)-approximation to the true Shapley value $s = [s_1, \cdots, s_N]^T \in \mathbb{R}^N$ with respect to l_p-norm if*

$$P_{\hat{s}}[||\hat{s}_i - s_i||_p \leq \epsilon] \geq 1 - \delta$$

Throughout this chapter, we will measure the approximation error in terms of l_2 norm.

3.3.1 *Baseline: Permutation Sampling*

We start by describing a baseline algorithm (Maleki, 2015) that approximates the Shapley value for any bounded utility functions with provable guarantees. The central idea behind the baseline algorithm is to regard the Shapley value definition in Equation (2.3):

$$s_i = \frac{1}{N!} \sum_{\pi \in \Pi(D)} \left[U(P_i^\pi \cup \{i\}) - U(P_i^\pi) \right]$$

as the expectation of a training instance's marginal contribution over a random permutation and then use the sample mean to approximate it. More specifically, let π be a random permutation of I and each permutation has a probability of $1/N!$. Consider the random variable

$$\phi_i = U(P_i^\pi \cup \{i\}) - U(P_i^\pi)$$

By (2.3), the Shapley value s_i is equal to $\mathbb{E}[\phi_i]$. Thus,

$$\hat{s}_i = \frac{1}{T} \sum_{t=1}^{T} U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t}) \tag{3.1}$$

is a consistent estimator of s_i, where π_t be t-th sample permutation uniformly drawn from all possible permutations $\Pi(I)$. Let r be the range of utility differences ϕ_i. By applying the Hoeffding's inequality, (Maleki et al.,

2013) shows that for general, bounded utility functions, the number of permutations T needed to achieve an (ϵ, δ)-approximation is

$$T = \frac{2r^2 N}{\epsilon^2} \log \frac{2N}{\delta} \tag{3.2}$$

For each permutation, the utility function is evaluated N times in order to compute the marginal contribution for all N users; therefore, the number of utility evaluations involved in the baseline approach is

$$m_{\text{eval}} = NT = \mathcal{O}(N^2 \log N)$$

We present a formal proof in Appendix A.1 and the pseudocode of our baseline in Algorithm 1. Typically, a substantial part of computational costs associated with the utility evaluation lies in $A(\cdot)$. Hence, it is useful to examine the efficiency of an approximation algorithm in terms of the number of model training required. In general, one utility evaluation would need to re-train a model. Particularly, when $A(\cdot)$ is incrementally trainable, one pass over the entire training set allows us to evaluate ϕ_i for all $i = 1, \cdots, N$. Hence, in this case, the number of model training needed to achieve an (ϵ, δ)-approximation is $m_{\text{eval}} = \mathcal{O}(N \log N)$.

Algorithm 1: Baseline: Permutation Sampling-Based Approach

```
input  : Training set - $D = \{(x_i, y_i)\}_{i=1}^N$
         Utility function $U(\cdot)$ with range $r$
         Approximation error parameters $\epsilon, \delta$
output : The Shapley value of each training point - $\hat{s} \in \mathbb{R}^N$
$T \leftarrow \frac{2r^2}{\epsilon^2} \log \frac{2N}{\delta}$
for $t \leftarrow 1$ to $T$ do
    $\pi_t \leftarrow$ GenerateUniformRandomPermutation$(D)$;
    $\phi_i^t \leftarrow U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t})$ for $i = 1, \ldots, N$;
end
$\hat{s}_i = \frac{1}{T} \sum_{t=1}^{T} \phi_i^t$ for $i = 1, \ldots, N$;
```

3.3.2 *Group Testing-Based Approach*

We now describe an algorithm that makes the same assumption of bounded utility as the baseline algorithm, but requires significantly fewer utility evaluations than the baseline.

Our proposed approximation algorithm is inspired by previous work applying the group testing theory to feature selection (Zhou et al., 2014). Recall that group testing is a combinatorial search paradigm (Du, F. K. Hwang, and F. Hwang, 2000), in which one wants to determine whether each item in a set is "good" or "defective" by performing a sequence of tests. The result of a test may be positive, indicating that at least one of the items of that subset is defective, or negative, indicating that all items in that subset are good. Each test is performed on a pool of different items and the number of tests can be made significantly smaller than the number of items by smartly distributing items into pools. Hence, the group testing is particularly useful when testing an individual item's quality is expensive.

Analogously, we can think of Shapley value calculation as a group testing problem with continuous quality measure. Each user's data is an "item" and the data utility corresponds to the item's quality. Each "test" in our scenario corresponds to evaluating the utility of a subset of users and is expensive. Drawing on the idea of group testing, we hope to recover the utility of all user subsets from a small amount of customized tests.

Let T be the total number of tests. At test t, a random set of users is drawn from I and we evaluate the utility of the selected set of users. If we model the appearance of user i and j's data in a test as Boolean random variables β_i and β_j, respectively, then the difference between the utility of user i and that of user j is

$$(\beta_i - \beta_j)U(\beta_1, \cdots, \beta_N) \tag{3.3}$$

where $U(\beta_1, \cdots, \beta_N)$ is the utility evaluated on the users with the Boolean appearance random variable equal to 1.

Using the definition of the Shapley value in Equation (2.2), one can derive the following formula of the Shapley value difference between any pair of users.

Lemma 1. *For any $i, j \in I$, the difference in Shapley values between i and j is*

$$s_i - s_j = \frac{1}{N-1} \sum_{S \subseteq I \setminus \{i,j\}} \frac{U(S \cup \{i\}) - U(S \cup \{j\})}{\binom{N-2}{|S|}} \tag{3.4}$$

The key idea of the proposed algorithm is to smartly design the sampling distribution of $\beta_1, \cdots, \beta_N$ such that the expectation of Equation (3.3) mirrors the Shapley difference in Equation (4.5). This will enable us to calculate the Shapely differences from the test results with a high-probability error bound. The following Lemma states that if we can estimate the Shapley differences between all data pairs up to $(\epsilon/\sqrt{N}, \delta/N)$, then we will be able to recover the Shapley value with the approximation error (ϵ, δ). The formal proof of this lemma can be found in the Appendix A.2.

Lemma 2. *Suppose that C_{ij} is an $(\epsilon/(2\sqrt{N}), \delta/(N(N-1)))$-approximation to $s_i - s_j$. Then, any solutions to the feasibility problem*

$$\sum_{i=1}^{N} \hat{s}_i = U_{tot} \tag{3.5}$$

$$|(\hat{s}_i - \hat{s}_j) - C_{i,j}| \leq \epsilon/(2\sqrt{N}) \quad \forall i, j \in \{1, \ldots, N\} \tag{3.6}$$

is an (ϵ, δ)-approximation to s with respect to l_2-norm.

The formal proof can be found in the Appendix A.3. Algorithm 2 presents the pseudo-code of the group testing-based algorithm, which first estimates the Shapley differences and then derives the Shapley value from the Shapley differences by solving a feasibility problem.
The following theorem provides a lower bound on the number of tests T needed to achieve an (ϵ, δ)-approximation.

Algorithm 2: Group Testing Based Shapley value Estimation.

```
input  : Training set - $D = \{(x_i, y_i)\}_{i=1}^N$
         Utility function $U(\cdot)$
         The number of tests - $T$
output : The estimated Shapley value of each training point - $\hat{s} \in \mathbb{R}^N$
$Z \leftarrow 2\sum_{k=1}^{N-1} \frac{1}{k}$;
$q(k) \leftarrow \frac{1}{Z}(\frac{1}{k} + \frac{1}{N-k})$ for $k = 1, \cdots, N-1$;
Initialize $\beta_{ti} \leftarrow 0$, $t = 1, ..., T, i = 1, ..., N$;
for $t = 1$ to $T$ do
    Draw $k_t \sim q(k)$;
    Uniformly sample a length-$k_t$ sequence $S$ from $\{1, \cdots, N\}$ ;
    $\beta_{ti} \leftarrow 1$ for all $i \in S$;
    $u_t \leftarrow U(S)$;
end
$\Delta U_{ij} \leftarrow \frac{Z}{T}\sum_{t=1}^{T} u_t(\beta_{ti} - \beta_{tj})$ for $i = 1, .., N$, $j = 1, ..., N$ and $j \geq i$ ;
Find $\hat{s}$ by solving the feasibility problem
  $\sum_{i=1}^N \hat{s}_i = U(I), |(\hat{s}_i - \hat{s}_j) - \Delta U_{i,j}| \leq \epsilon/(2\sqrt{N}), \forall i, j \in \{1, \cdots, N\}$;
```

Theorem 1. *Algorithm 2 returns an (ϵ, δ)-approximation to the Shapley value with respect to l_2-norm if the number of tests T satisfies*

$$T \geq 8 \log \frac{N(N-1)}{2\delta} / \left((1 - q_{tot}^2) h\left(\frac{\epsilon}{Zr\sqrt{N}(1 - q_{tot}^2)}\right)\right)$$

where

$$q_{tot} = \frac{N-2}{N} q(1) + \sum_{k=2}^{N-1} q(k)[1 + \frac{2k(k-N)}{N(N-1)}]$$

$$h(u) = (1+u)\log(1+u) - u$$

$$Z = 2\sum_{k=1}^{N-1} \frac{1}{k}$$

and r is the range of the utility function.[1]

Using the Taylor expansion of h, it can be proved that when N is large, T is $\mathcal{O}(N(\log N)^2)$. Since only one utility evaluation is required for a single test, the number of utility evaluations is at most

$$m_{eval} = \mathcal{O}(N(\log N)^2)$$

On the other hand, in the baseline approach, the number of utility evaluations is $\mathcal{O}(N^2 \log N)$. Hence, the group testing requires significantly fewer model evaluations than the baseline. We refer to Appendix A.4 for the formal proof.

3.3.3 *Exploiting the Sparsity of Values*

We now present an algorithm inspired by our empirical observations of the Shapley value for large datasets. This algorithm can produce an (ϵ, δ)-approximation to the Shapley value with only $\mathcal{O}(N \log(N) \log(\log(N)))$ utility evaluations.

Figure 3.3 illustrates the distribution of the Shapley value of the MNIST dataset, from which we observed that the Shapley value is "approximately sparse"—most of values are concentrated around its mean and only a few data points have significant values. In the literature, the "approximate sparsity" of a vector s is characterized by a small error of its best K-term approximation:

$$\sigma_K(s) = \inf\{\|s - z\|_1, \text{ z is } K\text{-sparse}\} \tag{3.7}$$

This observation opens up a vast collection of tools from compressive sensing for the purpose of calculating the Shapley value.

[1] Wang and Jia (J. T. Wang and Jia, 2023b) has since provided an improved version of this result with a more efficient lower bound that saves a factor of 8 and improves q_{tot}^2 to q_{tot}.

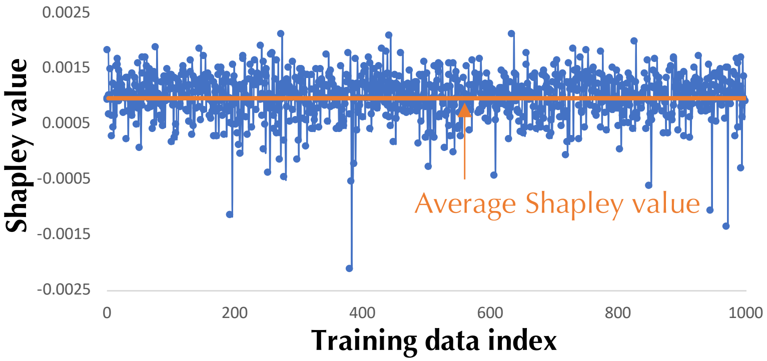


FIGURE 3.3: The distribution of the Shapley value of a size-1000 training set randomly sampled from MNIST. $\sigma_{367}(s)/(\sum_{i=1}^{N} s_i) = 0.5$. The utility function is the validation accuracy.

Compressive sensing studies the problem of recovering a sparse signal s with far fewer measurements $y = As$ than the length of the signal. A sufficient condition for recovery is that the measurement matrix $A \in \mathbb{R}^{M \times N}$ satisfies a key property, the *Restricted Isometry Property (RIP)*. In order to ensure that A satisfies this property, we simply choose A to be a random Bernoulli matrix. The results in random matrix theory imply that A satisfies RIP with high probability. Define the kth restricted isometry constant δ_k for a matrix A as

$$\delta_k(A) = \min\{\delta : \forall s, \|s\|_0 \le k, \\ (1-\delta)\|s\|_2^2 \le \|As\|_2^2 \le (1+\delta)\|s\|_2^2\} \tag{3.8}$$

It has been shown in (Rauhut, 2010) that every k-sparse vector s can be recovered by solving a convex optimization problem

$$\min_{s \in \mathbb{R}^N} \|s\|_1, \quad \text{s.t. } As = y \tag{3.9}$$

if $\delta_{2s}(A) < 1/3$. This result can also be generalized to noisy measurements (Candes, Romberg, and Tao, 2006). Drawing on the ideas of compressed sensing, we present Algorithm 3, termed compressive permutation sampling.

Algorithm 3: Compressive Permutation Sampling.

```
input  : Training set - $D = \{(x_i, y_i)\}_{i=1}^N$
         Utility function $U(\cdot)$
         The number of measurements - $M$
         The number of permutations - $T$
output : The Shapley value of each training point - $\hat{s} \in \mathbb{R}^N$
Sample a Bernoulli matrix $A$, where $A_{m,i} \in \{-1/\sqrt{M}, 1/\sqrt{M}\}$ with
  equal probability;
for $t \leftarrow 1$ to $T$ do
    $\pi_t \leftarrow$ GenerateUniformRandomPermutation$(D)$;
    $\phi_i^t \leftarrow U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t})$ for $i = 1, \ldots, N$;
    for $m \leftarrow 1$ to $M$ do
        $\hat{y}_{m,t} \leftarrow \sum_{i=1}^N A_{m,i}\phi_i^t$;
    end
end
$\bar{y}_m = \frac{1}{T}\sum_{t=1}^T \hat{y}_{m,t}$ for $m = 1, \ldots, M$;
$\bar{s} = U(D)/N$;
$\Delta s^* \leftarrow \text{argmin}_{\Delta s \in \mathbb{R}^N} \|\Delta s\|_1$, s.t. $\|A(\bar{s} + \Delta s) - \bar{y}\|_2 \leq \epsilon$;
$\hat{s} = \bar{s} + \Delta s^*$;
```

Theorem 2. *Suppose that $U(\cdot)$ is monotone. There exists some constant C' such that if*

$$M \geq C'(K \log(N/(2K)) + \log(2/\delta))$$

$$\text{and } T \geq \frac{2r^2}{\epsilon^2} \log \frac{4M}{\delta}$$

except for an event of probability no more than δ, the output of Algorithm 3 obeys

$$\|\hat{s} - s\|_2 \leq C_{1,K}\epsilon + C_{2,K}\frac{\sigma_K(s)}{\sqrt{K}} \tag{3.10}$$

for some constants $C_{1,K}$ and $C_{2,K}$.

Therefore, the number of utility evaluations (and model training) required for achieving the approximation error guarantee in Theorem 2 is

$$m_{eval} = NT = \mathcal{O}(N \log(\log(N)))$$

Particularly, when the utility function is defined with respect to an incrementally trainable model, only $\log\log(N)$ full model training is needed for achieving the error guarantee. The formal proof is provided for the reader in Appendix A.5.

3.3.4 *Stable Learning Algorithms*

We say a learning algorithm is *stable* if the model learned by the algorithm is insensitive to the removal of an arbitrary point in the training dataset (Bousquet and Elisseeff, 2002).

Definition 2. *An algorithm G has uniform stability γ with respect to the loss function l if*

$$\|l(G(S),\cdot) - l(G(S^{\setminus i}),\cdot)\|_\infty \leq \gamma \textit{ for all } i \in \{1,\cdots,|S|\}$$

where S denotes the training set and $S^{\setminus i}$ denotes the one by removing ith element of S

Indeed, a broad variety of learning algorithms are stable, including all learning algorithms with Tikhonov regularization. Stable learning algorithms are appealing as they enjoy provable generalization error bounds (Bousquet and Elisseeff, 2002). Assume that the model is trained via a stable learning algorithm and training data's utility is measured in terms of the validation loss. Due to the inherent insensitivity of a stable learning algorithm to the training data, we expect that the Shapley value of each training point is similar to one another. The following theorem confirms our intuition and provides an upper bound on the Shapley value difference between any pair of training data points.

Theorem 3. *For a learning algorithm $A(\cdot)$ with uniform stability $\beta = C_{stab}/|S|$, where $|S|$ is the size of the training set and C_{stab} is some constant. Let the utility of D be*

$$U(D) = M - L_{val}(A(D), D_{val})$$

where

$$L_{val}(A(D), D_{val}) = \frac{1}{N}\sum_{i=1}^{N} l(A(D), z_{val,i}) \text{ and } 0 \le l(\cdot,\cdot) \le M$$

Then

$$s_i - s_j \le 2C_{stab}\frac{1+\log(N-1)}{N-1}$$

and the Shapley difference vanishes as $N \to \infty$.

By Lemma 2, if

$$2C_{\text{stab}}\frac{1+\log(N-1)}{N-1} \le \epsilon/(2\sqrt{N})$$

then uniformly assigning U_{tot}/N to each data contributor provides an $(\epsilon, 0)$-approximation to the Shapley value in constant time $\mathcal{O}(1)$. We refer to Appendix A.6 for the formal proof.

3.3.5 *Heuristic Based on Influence Functions*

Computing the Shapley value involves evaluating the change in utility of all possible sets of data points after adding one more point. A plain way to evaluate the difference requires training a large number of models on different subsets of data. Koh et al. (Koh and Liang, 2017) show that influence functions can be used as an efficient approximation of parameter changes after adding or removing one point. Therefore, the need for re-training models is circumvented. Assume that model parameters are obtained by solving an empirical risk minimization problem

$$\hat{\theta}^m = \text{argmin}_\theta \frac{1}{m} \sum_{i=1}^{m} L(z_i, \theta)$$

Applying the result in (Koh and Liang, 2017), we can approximate the parameters learned after adding z by using the relation

$$\hat{\theta}_z^{m+1} = \hat{\theta}^m - \frac{1}{m} H_{\hat{\theta}^m}^{-1} \nabla_\theta L(z, \hat{\theta}^m)$$

where $H_{\hat{\theta}^m} = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta^2 L(z_i, \hat{\theta}^m)$ is the Hessian. The parameter change after removing z can be approximated similarly, except for replacing the $-$ by $+$ in the above formula. The efficiency of the baseline permutation sampling method can be significantly improved by combining it with influence functions. Moreover, we can employ a more sophisticated sampling scheme to reduce the variance of the result. Indeed, we can re-write the Shapley value as

$$s_i = \frac{1}{N} \sum_{k=1}^{N} \mathbb{E}[X_i^k]$$

where $X_i^k = U(S \cup \{i\}) - U(S)$ is the marginal contribution of user i to a size-k subset that is randomly selected with probability $1/\binom{N-1}{k}$. This suggests that stratified sampling can be used to approximate the Shapley value, which customizes the number of samples for estimating each expectation term according to the variance of X_i^k.

Largest-S Approximation. One practical heuristic of using influence functions is to consider a single subset S for computing s_i, namely, $I \setminus \{i\}$. With this heuristic, we can simply take a trained model on the whole dataset, and calculate the influence function for each data point. For logistic regression models, the first and second derivations enjoy closed-form expressions and the change in parameters after removing one point $z = (x, y)$ can be approximated by

$$-\Big(\sum_{i=1}^{N} \sigma(x_i^T \hat{\theta}^N)\sigma(-x_i^T \hat{\theta}^N) x_i x_i^T\Big)^{-1} \sigma(-y x_i^T \hat{\theta}^N) y x$$

where $\sigma(u) = 1/(1 + \exp(-u))$ and $y \in \{-1, 1\}$. Unfortunately, the fact that largest-S influence only considers a single subset makes it impossible to satisfy the *efficiency* and *linearity* properties simultaneously as shown below.

Theorem 4. *Consider the value attribution scheme that assigns the value $\hat{s}(U, i) = C_U[U(S \cup \{i\}) - U(S)]$ to user i where $|S| = N - 1$ and C_U is a constant such that $\sum_{i=1}^{N} \hat{s}(U, i) = U(I)$. Consider two utility functions $U(\cdot)$ and $V(\cdot)$. Then,*

$$\hat{s}(U + V, i) \neq \hat{s}(U, i) + \hat{s}(V, i)$$

unless

$$V(I)[\sum_{i=1}^{N} U(S \cup \{i\}) - U(S)] = U(I)[\sum_{i=1}^{N} V(S \cup \{i\}) - V(S)]$$

We refer to the Appendix A.7 for the formal proof. In the experimental section, we empirically validate this approach. On the `iris` dataset we show that this heuristic can produce a value that is correlated with the true Shapley value.

3.4 Experimental Results

COMPARING APPROXIMATION ACCURACY. We first compare the proposed approximation methods that only require mild assumptions on the ML models (e.g., bounded or differentiable utility), including

(a) the permutation sampling baseline

(b) the group testing-based method

(c) using influence functions to approximate all marginal contributions

(d) approximating the Shapley value with only the influence function to the largest subset.

The last two methods are hereinafter referred to as *all-S influence* and *largest-S influence*, respectively.
We use a small-scale dataset, `iris`, and use (a) to estimate the true Shapley value for a regularized logistic regression up to $\epsilon = 1/N$. Figure 3.4 (a) shows that the approximations produced by (a)-(c) are closest to each other. The result of the largest-S influence is correlated with that of the other techniques, although it cannot recover the true Shapley value.

RUNTIME COMPARISON. We implement the Shapley value calculation techniques on a machine with 16 cores (Intel Xeon CPU E5-2620 v4 @ 2.10GHz) and compare the runtime of different techniques on a two-class `dog-vs-fish` dataset (Koh and Liang, 2017) of size 900 constructed from the ImageNet dataset. To evaluate the runtime for training sizes above 900, we concatenate duplicate copies of the dog-vs-fish dataset. For each training data point, we first pre-compute the 2048-dimensional inception features and then train a logistic regression using the stochastic gradient descent for 150 epochs. The utility function is the negative testing loss of the logistic regression model.

For the largest-S influence and the all-S influence, we use the method in (Koh and Liang, 2017) to compute the influence function. The runtime of different techniques in logarithmic scale is displayed in Figure 3.4 (b). We can see that the group testing-based method outperforms the permutation sampling baseline by several orders of magnitude for a large number of data points. By exploiting influence function heuristics and the stratified sampling trick in Section 3.3.5, the computational costs can be further reduced. Due to the fact that the largest-S influence heuristic only focuses on the marginal contribution of each training data point to a single subset, it is much more efficient than the permutation sampling, group testing and the all-S influence, which compute the marginal contributions to a large number of subsets.

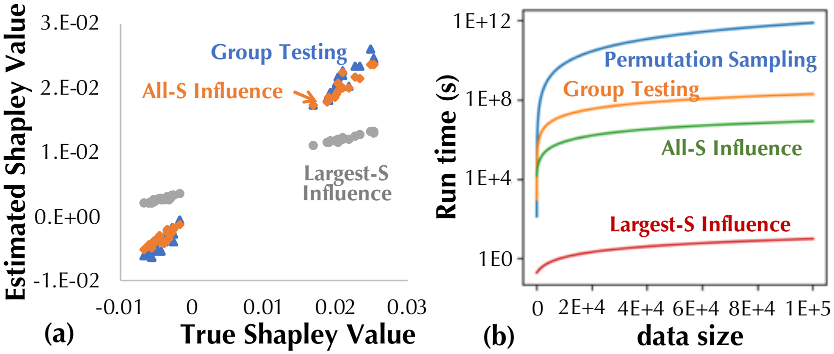


FIGURE 3.4: Consider the Shapley value approximation methods that do not rely on specific assumptions on the underlying learning algorithms and compare the (a) data values produced by them for training a logistic regression model and (b) their runtime.

APPROXIMATION UNDER SPARSITY ASSUMPTIONS. When it is plausible to assume the Shapley value of a training set is sparse, we could employ the idea of compressive sensing to recover the Shapley value with fewer samples. Figure 3.5 compares the sample efficiency of the baseline permutation sampling and the compressive permutation sampling method on a size-1000 dataset sampled randomly from MNIST. For a given approximation error, the compressive permutation requires significantly fewer samples and model valuations than the baseline approach. The superiority of the compressive permutation becomes less evident at the large sample regime.

STABLE LEARNING ALGORITHMS. Our theoretical result in Section 3.3.4 shows that the Shapley value of training data tends to be uniform for a stable learning algorithm, which has a small stability parameter β. We empirically validate this result by training a ridge regression on the diabetes dataset and varying the strength of its regularization term. In (Bousquet and Elisseeff, 2002), it is shown that the stability parameter β of the ridge regression $\min_\theta \frac{1}{N}\sum_{i=1}^{N} l(\theta, z_i) + \lambda\|\theta\|^2$ is proportional to σ^2/λ, where σ is the Lipschitz constant of the loss function with respect to the model parameter θ and equal to $2|x_i^T\theta - y_i| \cdot |x_i|$. When the model fits the training

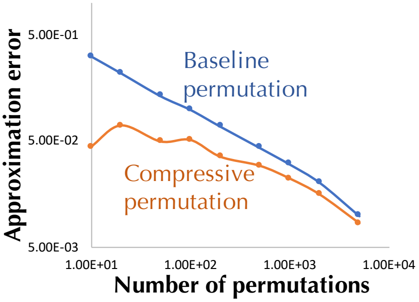


Figure 3.5: Comparison of approximation errors with different numbers of permutations for the baseline permutation sampling and the compressive permutation sampling method.

data well, the change in σ is small; therefore, applying more regularization leads to a more stable learning algorithm, which has lower variance in the training data values as illustrated in the shaded area of Figure 3.6. On the other hand, if the model no longer fits the data well due to excessive regularization, then σ will dominate the stability parameter. In this case, since σ increases with the regularization strength, β and thereby the variance of the Shapley value also increase. Note that the variance of the Shapley value is identical to the approximation error of a uniform value division scheme.

VALUE FOR PRIVACY-PRESERVING DATA. Differential privacy (Dwork, 2008) has emerged as a standard privacy notation and is often achieved by adding noise that has a magnitude proportional to the desired privacy level. On the other hand, noise diminishes the usefulness of data and thereby degrades the value of data. We construct a training set using the MNIST, and divide the training dataset into two halves, one half containing normal images and the other half containing noisy ones. The testing accuracy on normal images is used as the utility function. Figure 3.6 (b) illustrates a clear tradeoff between privacy and data value - the Shapley value decreases as data becomes noisier.

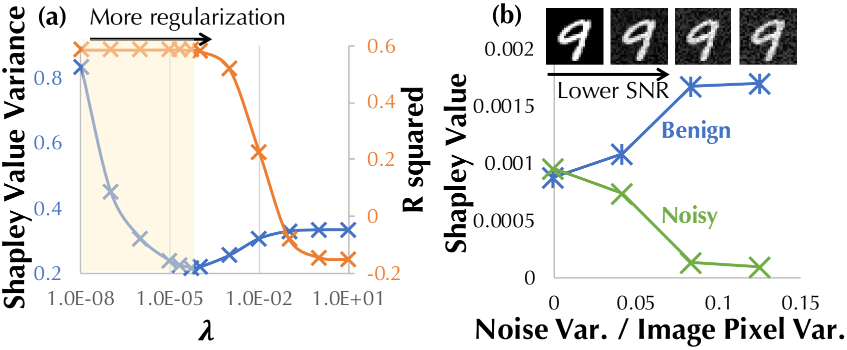


FIGURE 3.6: (a) Variance of data values for a ridge regression with different regularization strength (λ). (b) Tradeoff between data value and privacy.

VALUE FOR ADVERSARIAL EXAMPLES. Mixing adversarial examples with benign examples in the training dataset, or adversarial training, is an effective method to improve the adversarial robustness of a model. In practice, we measure the robustness in terms of the testing accuracy on a dataset containing adversarial examples. We expect that the adversarial examples in the training dataset become more valuable as more adversarial examples are added into the testing dataset. Based on the MNIST, we construct a training dataset that contains both benign and adversarial examples and synthesize testing datasets with different adversarial-benign mixing ratios. Two popular attack algorithms, namely, Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy, 2014) and the Carlini and Wagner (CW) attack (Carlini and Wagner, 2017) are used to generate adversarial examples. Figure 3.7 (a, b) compares the average Shapley value for adversarial examples and for benign examples in the training dataset. The negative testing loss for logistic regression is used as the utility function. We see that the Shapley value of adversarial examples increases as the testing data becomes more adversarial and contrariwise for benign examples. This is consistent with our expectation. In addition, the adversarial examples in the training set are more valuable if they are generated from the same attack algorithm for testing adversarial examples.

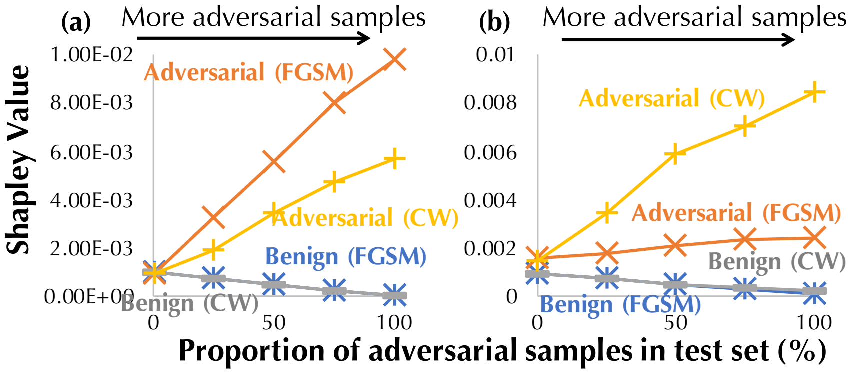


FIGURE 3.7: (a, b) Comparison of Shapley value of benign and adversarial examples. FGSM and CW are different attack algorithms used for generating adversarial examples in the testing dataset: (a) (resp. (b)) is trained on Benign + FGSM (resp. CW) adversarial examples.

3.5 Discussion

IMPLICATIONS OF TASK-SPECIFIC DATA VALUATION Since the Shapley value depends on the utility function associated with the game, data dividends based on the Shapley value are contingent on the definition of model usefulness in specific ML tasks. The task-specific nature of our data valuation framework offers clear advantages—it allows to accommodate the variability of a data point's utility from one application to another and assess its worth accordingly. Moreover, it enables the data buyer to defend against *data poisoning attacks*, wherein the attacker intentionally contributes adversarial training data points crafted specifically to degrade the performance of the ML model. In our framework, the "bad" training points will naturally have low Shapley values because they contribute little to boosting the performance of the model.

Having the data values dependent on the ML task, on the other hand, may raise some concerns about whether the data values may inherit the flaws of the ML models as to which the values are computed: if the ML model is biased towards a subpopulation with specific sensitive attributes (e.g., gender, race), will the data values reflect the same bias? Indeed, these concerns can be addressed by designing proper utility functions that devalue the unwanted properties of ML models. For instance, even if the ML model may be biased towards specific subpopulation, the buyer and data contributors can agree on a utility function that gives lower score to unfair models and compute the data values with respect to the concordant utility function. In this case, the training points will be appraised partially according to how much they contribute to improving the model fairness and the resulting data values would not be affected by the bias of the underlying model. Moreover, there is a venerable line of works studying algorithms to help improve fairness (Zemel et al., 2013; Woodworth et al., 2017; Hardt, Price, Srebro, et al., 2016). These algorithms can also be applied to resolve the potential bias in value assignments. For instance, before providing the data to the data buyer, data contributors can preprocess the training data so that the "sanitized" data removes the information correlated with sensitive attributes (Zemel et al., 2013). However, to ensure that the data values are accurately computed according to an appropriate utility function that the buyer and the data contributors agree on or that the models are trained with proper fairness criteria, it is necessary to develop systems that can support transparent machine learning processes.

3.6 Summary

ML has opened up exciting opportunities to tackle a wide variety of problems; nevertheless, very few works have attempted to understand the value of data used for training models. A principled way of data valuation is the key to stimulating data exchange, enabling the development of more sophisticated and robust ML models. In this Chapter, we adopt the Shapley value, a classic concept from cooperative game theory, for data valuation and provide a principled framework. The Shapley value has many unique properties appealing to data valuation. However, the lack of efficient methods to compute the Shapley value has prevented it from being adopted in the past. We develop a repertoire of techniques for estimating the Shapley value in different scenarios under general ML utilities. In addition to its theoretical soundness, we empirically demonstrated that the Shapley value coincides with people's intuition of data value. For instance, noisy images tend to have lower Shapley values than the high-fidelity ones; the training data whose distribution is closer to the validation data distribution tends to have higher Shapley values. These empirical results further back up the use of the Shapley value for data valuation.

4

SCALABLE DATA VALUATION FOR NEAREST NEIGHBOR ALGORITHMS

Any sufficiently advanced technology is indistinguishable from magic.

— Arthur C. Clarke

4.1 Introduction

SCALING DATA VALUATION TO MILLIONS OF DATA POINTS. Intuitively, the Shapley value measures the marginal improvement of utility attributed to the data point z_i, averaged over all possible subsets of data points. Calculating exact Shapley values requires exponentially many utility evaluations. Thus, when N grows to millions or even billions in a realistic ML-based data valuation setting, how can we design more efficient algorithms? This poses a radical challenge to using the Shapley value for practical data valuation for machine learning:

Challenge 1

> *Can we efficiently compute the Shapley value at scale for millions or even billions of data points?*

This scale is rare to the previous applications of the Shapley value but is not uncommon for real-world data valuation tasks tailored to ML models. In Chapter 3, we examined some general properties of machine learning models, such as the boundedness of the utility functions, stability of the learning algorithms, and analyzed their implications for calculating the Shapley value. The algorithms presented were only capable of producing (ϵ, δ)-approximations to the Shapley value. When the desired level of approximation error is small, these algorithms entail significant computational costs, making them unsuitable for handling large-scale data.

To tackle this challenge, we focus on a specific family of ML models which restrict the class of utility functions $U(\cdot)$ that we consider. Specifically, we study K-nearest neighbors (KNN) classifiers (Dudani, 1976), a simple yet popular supervised learning method used in image recognition (Hays and Efros, 2015), recommendation systems (Adeniyi, Wei, and Yongquan, 2016), healthcare (C. Li, S. Zhang, et al., 2012), etc.

Given a validation set, $\mathcal{D}_{val} = \{z'_1, ..., z'_{N_{\text{val}}}\}$, we focus on a natural utility function, called the *KNN utility*, which, intuitively, measures the boost of the likelihood that KNN assigns the correct label to each validation data point. When $K = 1$, this utility is the same as the validation accuracy. The most surprising result is that for unweighted KNN classifiers and regressors, the Shapley value of all N data points can be computed, *exactly*, in $O(N \log N)$ time – an exponential improvement on computational complexity! We show that for (ϵ, δ)-approximation, we are able to develop an approximate KNN Shapley algorithm based on Locality Sensitive Hashing (LSH) with only *sublinear* complexity $O(N^{h(\epsilon,K)} \log N)$ when ϵ is not too small and K is not too large. We empirically evaluate our algorithms on up to 10 million data points and even our *exact* algorithm is up to three orders of magnitude faster than the baseline approximation algorithm. The LSH-based approximation algorithm can accelerate the value calculation process even further.

EXTENDING THE DATA VALUATION FRAMEWORK. Furthermore, in Chapter 3, we presented data valuation as a cooperative game in which a data contributor supplies only one data point, and the service provider manages data processing and computation, thereby solely allocating profit to data contributors. Nonetheless, in many real-world situations, a data provider might contribute multiple data points. For example, in social media platforms such as Twitter, Facebook or Instagram, the users are data contributors that generate multiple data by posting messages, im-

	Exact	**Approximate**
Baseline	$2^N N \log N$	$\frac{N^2}{\epsilon^2} \log N \log \frac{N}{\delta}$
Unweighted KNN classifier	$N \log N$	$N^{h(\epsilon,K)} \log N \log \frac{K^*}{\delta}$
Unweighted KNN regression	$N \log N$	—
Weighted KNN	N^K	$\frac{N}{\epsilon^2} \log K \log \frac{K}{\delta}$
Multiple-data-per-curator KNN	M^K	$\frac{N}{\epsilon^2} \log K \log \frac{K}{\delta}$

FIGURE 4.1: Time complexity for computing the Shapley value for KNN models. N is the total number of training data points. M is the number of data contributors. $h(\epsilon, K) < 1$ if $K^* = \max\{1/\epsilon, K\} < C$ for some dataset-dependent constant C.

ages, and videos linked to their account. This data is used to develop e.g. sentiment analysis algorithms to understand user preferences. Moreover, service providers may also outsource the machine learning development to a dedicated data analyst or to a data science competition (e.g. Kaggle). This introduces the challenge of how to fairly value not just data but also computation and introduces. Expanding on our result for unweighted KNN, we would like to study if our practical algorithms are applicable for these variations.

Challenge 2

> *Can we extend data valuation problems according to whether data contributors are valued separately or in tandem with a data analyst and whether each data contributor contribute a single data instance or multiple ones?*

4.1.1 *Contributions*

The contributions presented in this chapter are the results of joint work of many co-authors and have been previously published in VLDB under

Ruoxi Jia, **David Dao**, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas J. Spanos, Dawn Song. "Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms". *VLDB*. 2019.

The contribution of this work is a collection of novel algorithms for efficient data valuation within the above scope. Figure 4.1 summarizes our technical results. Specifically, we made following contributions:

CONTRIBUTION 1: DATA VALUATION FOR KNN CLASSIFIERS The main challenge of adopting the Shapley value for data valuation is its computational complexity — for general, bounded utility functions, calculating the Shapley value requires $O(2^N)$ utility evaluations for N data points. Even getting an (ϵ, δ)-approximation (error bounded by ϵ with probability at least $1 - \delta$) for all data points requires $O(N \log N)$ utility evaluations using state-of-the-art methods (See Section 4.2.1). For the KNN utility, each utility evaluation requires to sort the training data, which has asymptotic complexity $O(N \log N)$.

C1.1 Exact Computation We first propose a novel algorithm specifically designed for KNN classifiers. We observe that the KNN utility satisfies what we call the *piecewise utility difference* property: the difference in the marginal contribution of two data points z_i and z_j over has a "piecewise form" (See Section 4.2.2):

$$U(S \cup \{z_i\}) - U(S \cup \{z_j\}) = \sum_{t=1}^{T} C_{i,j}^{(t)} \mathbb{1}[S \in \mathcal{S}_t], \forall S \in \mathcal{D} \backslash \{z_i, z_j\}$$

where $\mathcal{S}_t \subseteq 2^{\mathcal{D} \backslash \{z_i, z_j\}}$ and $C_{i,j}^{(t)} \in \mathbb{R}$. This combinatorial structure allows us to design a very efficient algorithm that only has $O(N \log N)$ complexity for *exact* computation of Shapley values on all N data points. This is an exponential improvement over the $O(2^N N \log N)$ baseline!

C1.2 Sublinear Approximation The exact computation requires to sort the entire training set for each test point, thus becoming time-consuming for large and high-dimensional datasets. Moreover, in some applications such as document retrieval, test points could arrive sequentially and the values of each training point needs to get updated and accumulated on the fly, which makes it impossible to complete sorting offline. Thus, we investigate whether higher efficiency can be achieved by finding approximate Shapley values instead. We study the problem of getting (ϵ, δ)-approximation of the Shapley values for the KNN utility. This happens to be reducible to the problem of answering approximate $\max\{K, 1/\epsilon\}$-nearest neighbor queries with probability $1 - \delta$. We designed a novel algorithm by taking advantage of LSH, which only requires $O(N^{h(\epsilon,K)} \log N)$ computation where $h(\epsilon, K)$ is dataset-dependent and typically less than 1 when ϵ is not too small and K is not too large.

Limitation of LSH The $h(\epsilon, K)$ term monotonically increases with $\max\{\frac{1}{\epsilon}, K\}$. In experiments, we found that the LSH can handle mild error requirements (e.g., $\epsilon = 0.1$) but appears to be less efficient than the exact calculation algorithm for stringent error requirements. Moreover, we can extend the exact algorithm to cope with KNN regressors and other scenarios detailed in Contribution 2; however, the application of the LSH-based approximation is still confined to the classification case.

To our best knowledge, the above results are one of the very first studies of efficient Shapley value evaluation designed specifically for utilities arising from ML applications.

contribution 2: extensions Our second contribution is to extend our results to different settings beyond a standard KNN classifier and the KNN utility (Section 4.3). The connection between different settings are illustrated in Figure 4.2, where each vertical layer represents a different slicing to the data valuation problem. In some of these scenarios, we successfully designed algorithms that are as efficient as the one for KNN classifiers. In some other cases, including weighted KNN and the multiple-data-per-curator setup, the exact computation algorithm is less practical although being improved exponentially. Thus, we introduce an improved MC approximation for KNN that is able to improve the state-of-the-art.

C2.1 Unweighted *K*NN regressors. We introduce an efficient algorithm for unweighted *K*NN regressors, extending our findings beyond classification tasks.

C2.2 Weighted *K*NN classifiers and regressors. We develop an algorithmic framework for weighted *K*NN classifiers and regressors, broadening our approach to handle varying importance of data points.

C2.3 Data/analytics joint valuation. We introduce composite games that extends to a "data analyst" player that provides ML analytics and provide a system that attaches value to both the analyst and data curators.

C2.4 Multiple data per contributor. We extend to scenarios where one "data curator" contributes multiple data points *and* has the freedom to delete all data points at the same time.

C2.5: Improved Monte Carlo Approximation for *K*NN. To further improve the efficiency in the less efficient cases, we strengthen the sample complexity bound of the state-of-the-art approximation algorithm, achieving an $\mathcal{O}(N \log^2 N / \log^2 K)$ complexity improvement over the state-of-the-art. Our algorithm requires in total $\mathcal{O}(N/\epsilon^2 \log^2 K)$ computation and is often practical for reasonable ϵ.

EXPERIMENTS We implement our algorithms and evaluate them on datasets up to ten million data points. We observe that our exact Shapley value calculation algorithm can provide up to three orders of magnitude speed-up over the state-of-the-art Monte Carlo approximation approach. With the LSH-based approximation method, we can accelerate the Shapley value calculation even further by allowing approximation errors. The actual performance improvement of the LSH-based method over the exact algorithm depends the dataset as well as the error requirements. For instance, on a *10M subset* of the Yahoo Flickr Creative Commons 100M dataset, we observe that the LSH-based method can bring another 4.6×

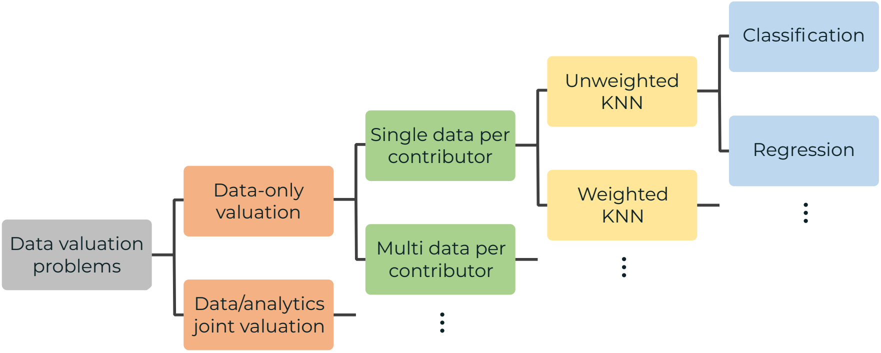


Figure 4.2: Taxonomy of data valuation problems. Each parent node has two children. However, due to the size of the figure, we have omitted most of the branches and represented them as dots.

speed-up. Moreover, to our best knowledge, this work is also one of the first to evaluate data valuation at scale.

4.1.2 *Overview*

The rest of this Chapter is organized as follows. We present our efficient algorithms for *K*NN classifiers in in Section 4.2. We discuss the extensions in Section 4.3 and propose a Monte Carlo approximation algorithm in Section 4.5, which significantly boosts the efficiency for the extensions that have less practical exact algorithms. We evaluate our approach in Section 4.6. Lastly, we conclude with a discussion in Section 4.7 and a summary of the Chapter in Section 4.8.

4.2 Valuing Data for *K*NN Classifiers

In this section, we present an algorithm that can calculate the exact Shapley value for *K*NN classifiers in quasi-linear time. Further, we exhibit an approximate algorithm based on LSH that could achieve sublinear complexity.

4.2.1 *A Baseline Algorithm for KNN Shapley Computation*

Despite the desirable properties of the Shapley value, efficient algorithms to compute it *exactly* have not yet been developed. For large datasets, it is therefore common to approximate the Shapley value through Monte Carlo (MC) sampling (Maleki, 2015) as introduced in Chapter 3. We will use MC sampling as a baseline for our subsequent studies. Consider the random variable

$$\phi_i = U(P_i^\pi \cup \{i\}) - U(P_i^\pi)$$

where π be a random permutation of I and each permutation has a probability of $1/N!$. According to Equation (2.3), $s_i = \mathbb{E}[\phi_i]$ and we can estimate s_i by the sample mean. The number of permutations needed to achieve an (ϵ, δ)-approximation is

$$(2r^2 N/\epsilon^2) \log(2N/\delta)$$

where r is the range of the utility function. We refer the reader to Chapter 3 for a detailed discussion.

Take the KNN classifier as an example and assume that $U(\cdot)$ represents the validation accuracy of the classifier. Then, evaluating $U(S)$ needs to sort the training data in S according to their distances to the validation point, which has $\mathcal{O}(|S| \log |S|)$ complexity. Since on average $|S| = N/2$, the asymptotic complexity of calculating the Shapley value for a KNN classifier via the baseline algorithm is

$$\mathcal{O}(N^2 \log^2 N)$$

which is prohibitive for large-scale datasets. In the following sections, we will show that leveraging the locality of KNN models allows for the development of much more efficient algorithms for computing the Shapley value.

4.2.2 *Exact Shapley Value Calculation*

KNN algorithms are popular supervised learning methods, widely adopted in a multitude of applications such as computer vision, information retrieval, etc. Suppose the dataset D consisting of pairs (x_1, y_1), (x_2, y_2), ..., (x_N, y_N) taking values in $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the label space. Depending on whether the nearest neighbor algorithm is used for classification or regression, $\mathcal{Y}$ is either discrete or continuous.

The training phase of KNN consists only of storing the features and labels in D. The testing phase is aimed at finding the label for a given query (or test) feature. This is done by searching for the K training features most similar to the query feature and assigning a label to the query according to the labels of its K nearest neighbors. Given a single validation point x_{val} with the label y_{val}, the simplest, unweighted version of a KNN classifier first finds the top-K training points $(x_{\alpha_1}, \cdots, x_{\alpha_K})$ that are most similar to x_{val} and outputs the probability of x_{val} taking the label y_{val} as $P[x_{\text{val}} \to y_{\text{val}}] = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[y_{\alpha_k} = y_{\text{val}}]$, where α_k is the index of the kth nearest neighbor.

One natural way to define the utility of a KNN classifier is by the likelihood of the right label:

$$U(S) = \frac{1}{K} \sum_{k=1}^{\min\{K, |S|\}} \mathbb{1}[y_{\alpha_k(S)} = y_{\text{val}}] \tag{4.1}$$

where $\alpha_k(S)$ represents the index of the training feature that is kth closest to x_{val} among the training examples in S. Specifically, $\alpha_k(I)$ is abbreviated to α_k.

Using this utility function, we can derive an efficient *and exact* way of computing the Shapley value.

Theorem 5. *Consider the utility function in (4.1). Then, the Shapley value of each training point can be calculated recursively as follows:*

$$s_{\alpha_N} = \frac{\mathbb{1}[y_{\alpha_N} = y_{val}]}{N} \tag{4.2}$$

$$s_{\alpha_i} = s_{\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{val}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{val}]}{K} \frac{\min\{K, i\}}{i} \tag{4.3}$$

Note that the above result for a single validation point can be readily extended to the multiple-validation-point case, in which the utility function is defined by

$$U(S) = \frac{1}{N_{\text{val}}} \sum_{j=1}^{N_{\text{val}}} \frac{1}{K} \sum_{k=1}^{\min\{K,|S|\}} \mathbb{1}[y_{\alpha_k^{(j)}(S)} = y_{\text{val},j}] \tag{4.4}$$

where $\alpha_k^{(j)}(S)$ is the index of the kth nearest neighbor in S to $x_{\text{val},j}$. By the additivity property, the Shapley value for multiple test points is the average of the Shapley value for every single test point. The pseudo-code for calculating the Shapley value for an unweighted KNN classifier is presented in Algorithm 4. The computational complexity is only

$$\mathcal{O}(N \log N N_{\text{val}})$$

for N training data points and N_{val} validation data points—this is simply to sort N_{val} arrays of N numbers!

PROOF IDEA FOR 1NN. Before, we introduce the formal proof below, we want to illustrate the proof idea for $K = 1$ in Figure 4.3. Given an ordered list of training data points and one validation data point, we analyze the difference in utility between two neighboring training data points. Consider Case 1, where two data points i and j have different labels $y_i \neq y_j$. If there is no data point in $S \subseteq I \setminus \{i, j\}$ that ranks higher (is more similar) than i and j (Case 1.1) then the utility difference between i and j will be ± 1, because i or j will be used by the 1NN algorithm to make a prediction. However, if the subset S contains a point l ranked lower than i or j (Case 1.2) then the utility difference will be 0, because neither i nor j, but only l,

Algorithm 4: Exact algorithm for calculating the Shapley value for an unweighted KNN classifier.

```
input  : Training data $D = \{(x_i, y_i)\}_{i=1}^{N}$
         Validation data $D_{\text{val}} = \{(x_{\text{val},i}, y_{\text{val},i})\}_{i=1}^{N_{\text{val}}}$
output : The Shapley value $\{s_i\}_{i=1}^{N}$
for $j \leftarrow 1$ to $N_{val}$ do
    $(\alpha_1, ..., \alpha_N) \leftarrow$ Indices of training data in an ascending order using
     $d(\cdot, x_{\text{val}})$;
    $s_{j,\alpha_N} \leftarrow \frac{\mathbb{1}[y_{\alpha_N} = y_{\text{val}}]}{N}$;
    for $i \leftarrow N-1$ to $1$ do
        $s_{j,\alpha_i} \leftarrow s_{j,\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{\text{val},j}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{\text{val},j}]}{K} \frac{\min\{K,i\}}{i}$;
    end
end
for $i \leftarrow 1$ to $N$ do
    $s_i \leftarrow \frac{1}{N_{\text{val}}} \sum_{j=1}^{N_{\text{val}}} s_{j,i}$;
end
```

will be used for prediction, i.e., $U(S \cup \{i\}) = U(S) = U(S \cup \{j\})$. If i and j have identical labels (Case 2), the utility difference is 0 for any S. Therefore, we can simply calculate the Shapley value difference between i and j by counting how many subsets S fall into Case 1.1.

FORMAL PROOF. The formal proof of Theorem 5 relies on lemma 1 from Chapter 3 which states that the difference in the utility gain induced by either point i or point j translates linearly to the difference in the respective Shapley values.

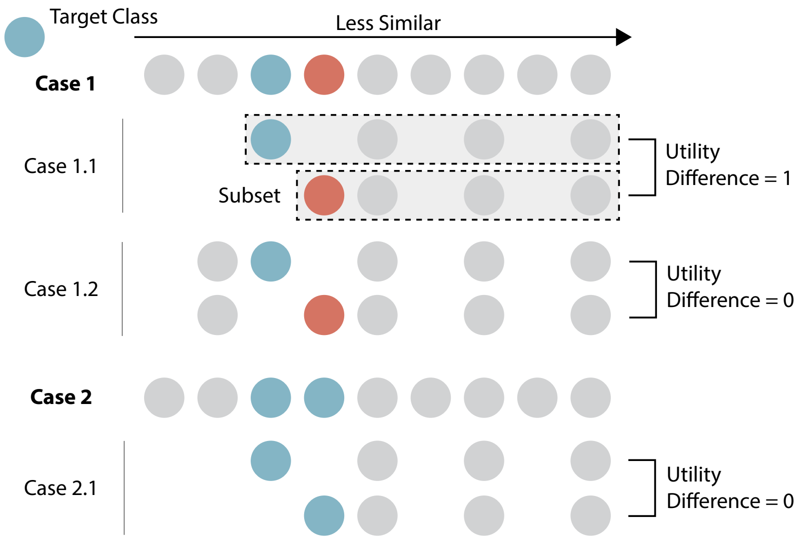


FIGURE 4.3: Illustration of the proof idea for 1NN. There are two distinct classes of data points: red and blue. The nodes colored in grey can belong to either the red or blue class.

Lemma 3. *For any $i, j \in I$, the difference in Shapley values between i and j is*

$$s_i - s_j = \frac{1}{N-1} \sum_{S \subseteq I \setminus \{i,j\}} \frac{U(S \cup \{i\}) - U(S \cup \{j\})}{\binom{N-2}{|S|}} \tag{4.5}$$

Proof of Theorem 5. W.l.o.g., we assume that $x_1, \ldots, x_n$ are sorted according to their similarity to x_{val}, that is, $x_i = x_{\alpha_i}$. For any given subset $S \subseteq I \setminus \{i, i+1\}$ of size k, we split the subset into two disjoint sets S_1 and S_2 such that $S = S_1 \cup S_2$ and $|S_1| + |S_2| = |S| = k$. Given two neighboring points with indices $i, i+1 \in I$, we constrain S_1 and S_2 to $S_1 \subseteq \{1, ..., i-1\}$ and $S_2 \subseteq \{i+2, ..., N\}$.

Let s_i be the Shapley value of data point x_i. By Lemma 1, we can draw conclusions about the Shapley value difference $s_i - s_{i+1}$ by inspecting the utility difference $U(S \cup \{i\}) - U(S \cup \{i+1\})$ for any $S \subseteq I \setminus \{i, i+1\}$. We analyze $U(S \cup \{i\}) - U(S \cup \{i+1\})$ by considering the following cases.

(1) $|S_1| \geq K$.
In this case, we know that $i, i+1 > K$ and therefore $U(S \cup \{i\}) = U(S \cup \{i+1\}) = U(S)$, hence $U(S \cup \{i\}) - U(S \cup \{i+1\}) = 0$.

(2) $|S_1| < K$.
In this case, we know that $i \leq K$ and therefore $U(S \cup \{i\}) - U(S)$ might be nonzero. Note that including a point i into S can only expel the Kth nearest neighbor from the original set of K nearest neighbors. Thus, $U(S \cup \{i\}) - U(S) = \frac{1}{K}(\mathbb{1}[y_i = y_{\text{val}}] - \mathbb{1}[y_K = y_{\text{val}}])$. The same hold for the inclusion of point $i+1$: $U(S \cup \{i+1\}) - U(S) = \frac{1}{K}(\mathbb{1}[y_{i+1} = y_{\text{val}}] - \mathbb{1}[y_K = y_{\text{val}}])$. Combining the two equations, we have

$$\phi_{i,K} \triangleq U(S \cup \{i\}) - U(S \cup \{i+1\}) = \frac{\mathbb{1}[y_i = y_{\text{val}}] - \mathbb{1}[y_{i+1} = y_{\text{val}}]}{K}$$

Combining the two cases discussed above and applying Lemma 1, we have

$$\begin{aligned}
&s_i - s_{i+1} \\
=&\frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{\substack{S_1 \subseteq \{1,\ldots,i-1\},\\ S_2 \subseteq \{i+2,\ldots,N\}:\\ |S_1|+|S_2|=k, |S_1|<K}} \phi_{i,K} \\
=&\phi_{i,K} \times \frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{m=0}^{\min(K-1,k)}\binom{i-1}{m}\binom{N-i-1}{k-m} \qquad (4.6)
\end{aligned}$$

The sum of binomial coefficients in (4.6) can be simplified as follows:

$$\begin{aligned}
&\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{m=0}^{\min\{K-1,k\}}\binom{i-1}{m}\binom{N-i-1}{k-m} && (4.7)\\
=&\sum_{m=0}^{\min\{K-1,i-1\}}\sum_{k'=0}^{N-i-1}\frac{\binom{i-1}{m}\binom{N-i-1}{k'}}{\binom{N-2}{m+k'}} && (4.8)\\
=&\frac{\min\{K,i\}(N-1)}{i} && (4.9)
\end{aligned}$$

where the first equality is due to the exchange of the inner and outer summation and the second one is by taking $v = N-i-1$ and $u = i-1$ in the binomial identity $\sum_{j=0}^{v} \frac{\binom{u}{i}\binom{v}{j}}{\binom{u+v}{i+j}} = \frac{u+v+1}{u+1}$.

Therefore, we have the following recursion

$$s_i - s_{i+1} = \frac{\mathbb{1}[y_i = y_{\text{val}}] - \mathbb{1}[y_{i+1} = y_{\text{val}}]}{K} \frac{\min\{K, i\}}{i} \tag{4.10}$$

Now, we analyze the formula for s_N, the starting point of the recursion. Since x_N is farthest to x_{val} among all training points, x_N results in non-zero marginal utility only when it is added to the subsets of size smaller than K. Hence, s_N can be written as

$$s_N = \frac{1}{N} \sum_{k=0}^{K-1} \frac{1}{\binom{N-1}{k}} \sum_{|S|=k, S\subseteq I\setminus\{N\}} U(S \cup N) - U(S) \tag{4.11}$$

$$= \frac{1}{N} \sum_{k=0}^{K-1} \frac{1}{\binom{N-1}{k}} \sum_{|S|=k, S\subseteq I\setminus\{N\}} \frac{\mathbb{1}[y_N = y_{\text{val}}]}{K} \tag{4.12}$$

$$= \frac{\mathbb{1}[y_N = y_{\text{val}}]}{N} \tag{4.13}$$

□

4.2.3 *LSH-based Approximation*

The exact calculation of the KNN Shapley value for a query instance requires to sort the entire training dataset, and has computation complexity $\mathcal{O}(N_{\text{val}}(Nd + N\log(N)))$, where d is the feature dimension. Thus, the exact method becomes expensive for large and high-dimensional datasets. We now present a sublinear algorithm to approximate the KNN Shapley value for classification tasks.

The key to boosting efficiency is to realize that only $\mathcal{O}(1/\epsilon)$ nearest neighbors are needed to estimate the KNN Shapley value with up to ϵ error. Therefore, we can avert the need of sorting the entire database for every new query point.

Algorithm 5: LSH-based algorithm for estimating the Shapley value for an unweighted KNN classifier.

```
input  : Training data $D = \{(x_i, y_i)\}_{i=1}^{N}$
         Validation data $D_{\mathrm{val}} = \{(x_{\mathrm{val},i}, y_{\mathrm{val},i})\}_{i=1}^{N_{\mathrm{val}}}$
         Hash tables $H$
output : The estimated Shapley value of each training point$\{\hat{s}_i\}_{i=1}^{N}$
$K^* \leftarrow \max(K, \lceil 1/\epsilon \rceil)$
$\hat{s}_{i,j} \leftarrow 0$;
for $j \leftarrow 1$ to $N_{val}$ do
    $(\alpha_1, ..., \alpha_{K^*}) \leftarrow$ Indices of LSH$(H, D, x_{\mathrm{val}})$ candidates in ascending
     order using $d(\cdot, x_{\mathrm{val}})$;
    for $i \leftarrow K^* - 1$ to $1$ do
        $\hat{s}_{j,\alpha_i} \leftarrow \hat{s}_{j,\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{\mathrm{val},j}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{\mathrm{val},j}]}{K} \frac{\min\{K,i\}}{i}$;
    end
end
for $i \leftarrow 1$ to $N$ do
    $\hat{s}_i \leftarrow \frac{1}{N_{\mathrm{val}}} \sum_{j=1}^{N_{\mathrm{val}}} \hat{s}_{j,i}$;
end
```

Theorem 6. *Consider the utility function defined in Equation (4.1). Consider $\{\hat{s}_i\}_{i=1}^{N}$ defined recursively by*

$$\hat{s}_{\alpha_i} = 0 \qquad \text{if } i \geq K^* \tag{4.14}$$

$$\hat{s}_{\alpha_i} = \hat{s}_{\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{val}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{val}]}{K} \frac{\min\{K, i\}}{i} \qquad \text{if } i \leq K^* - 1 \tag{4.15}$$

where $K^* = \max\{K, \lceil 1/\epsilon \rceil\}$ for some $\epsilon > 0$. Then, $[\hat{s}_{\alpha_1}, \ldots, \hat{s}_{\alpha_N}]$ is an $(\epsilon, 0)$-approximation to the true Shapley value $[s_{\alpha_1}, \ldots, s_{\alpha_N}]$ and $\hat{s}_i - \hat{s}_{i+1} = s_i - s_{i+1}$ for $i \leq K^* - 1$.

Theorem 6 indicates that we only need to find $\max\{K, \lceil 1/\epsilon \rceil\} (\triangleq K^*)$ nearest neighbors to obtain an $(\epsilon, 0)$-approximation. Moreover, since $\hat{s}_i - \hat{s}_{i+1} = s_i - s_{i+1}$ for $i \leq K^* - 1$, the approximation retains the original value rank for K^* nearest neighbors.

A common technique used to search for the nearest neighbors is locality sensitive hashing (LSH) (Charikar, 2002). In LSH, every training instance x is converted into codes in each hash table by using a series of hash functions $h_j(x)$, $j = 1, \dots, m$. Each hash function is designed to preserve the relative distance between different training instances; similar instances have the same hashed value with high probability. Various hash functions have been proposed to approximate KNN under different distance metrics (Charikar, 2002; Datar et al., 2004). We will focus on the distance measured in l_2 norm; in that case, a commonly used hash function is $h(x) = \left\lfloor \frac{w^T x + b}{r} \right\rfloor$, where w is a vector with entries sampled from a p-stable distribution, and b is uniformly chosen from the range $[0, r]$. It is shown in (Datar et al., 2004):

$$P[h(x_i) = h(x_{\text{val}})] = f_h(\|x_i - x_{\text{val}}\|_2) \tag{4.16}$$

where the function $f_h(c) = \int_0^r \frac{1}{c} f_2(\frac{z}{c})(1 - \frac{z}{r}) dz$ is a monotonically decreasing with c. Here, f_2 is the probability density function of the absolute value of a 2-stable random variable.

We now present a theorem which relates the success rate of finding approximate nearest neighbors to the intrinsic property of the dataset and the parameters of LSH.

Theorem 7. *LSH can find the exact K nearest neighbors with probability $1 - \delta$ and*

$$\mathcal{O}(d \log(N) N^{g(C_K)} \log \frac{K}{\delta}) \textit{ time complexity}$$

$$\mathcal{O}(Nd + N^{g(C_K)+1} \log \frac{K}{\delta}) \textit{ space complexity}$$

$$\mathcal{O}(N^{g(C_K)} \log \frac{K}{\delta}) \textit{ hash tables}$$

where $g(C_K) = \log f_h(1/C_K)/\log f_h(1)$ *is monotonically decreasing.* $C_K = D_{mean}/D_K$, *where* D_{mean} *is the expected distance of a random training instance to a query* x_{val} *and* D_K *is the expected distance between* x_{val} *to its Kth nearest neighbor denoted by* $x_{\alpha_i}(x_{val})$, *i.e.,*

$$D_{mean} = \mathbb{E}_{x,x_{val}}[D(x, x_{val})] \tag{4.17}$$

$$D_K = \mathbb{E}_{x_{val}}[D(x_{\alpha_i}(x_{val}), x_{val}] \tag{4.18}$$

The above theorem essentially extends the 1NN hardness analysis in Theorem 3.1 of (J. He, S. Kumar, and S.-F. Chang, 2012) to KNN. C_K measures the ratio between the distance from a query instance to a random training instance and that to its Kth nearest neighbor. We will hereinafter refer to C_K as *Kth relative contrast*. Intuitively, C_K signifies the difficulty of finding the Kth nearest neighbor. A smaller C_K implies that some random training instances are likely to have the same hashed value as the Kth nearest neighbor, thus entailing a high computational cost to differentiate the true nearest neighbors from the false positives. Theorem 7 shows that among the datasets of the same size, the one with higher relative contrast will need lower time and space complexity and fewer hash tables to approximate the K nearest neighbors. Combining Theorem 6 and Theorem 7, we obtain the following theorem that explicates the tradeoff between KNN Shapley value approximation errors and computational complexity.

Theorem 8. *Consider the utility function defined in Equation* (4.4). *Let* $\hat{x}_{\alpha_k^{(j)}}$ *denote the kth closest training point to* $x_{val,j}$ *output by LSH with*

$$\mathcal{O}(N_{val} d \log(N) N^{g(C_{K^*})} \log \frac{N_{val}K^*}{\delta}) \text{ time complexity}$$

$$\mathcal{O}(Nd + N^{g(C_{K^*})+1} \log \frac{N_{val}K^*}{\delta}) \text{ space complexity}$$

$$\mathcal{O}(N^{g(C_{K^*})} \log \frac{N_{val}K^*}{\delta}) \text{ hash tables}$$

where $K^* = \max(K, \lceil 1/\epsilon \rceil)$. *Suppose that* $\{\hat{s}_i\}_{i=1}^N$ *is computed via*

$$\hat{s}_i = \frac{1}{N_{val}} \sum_{j=1}^{N_{val}} \hat{s}_{i,j}$$

and $\hat{s}_{i,j}$ $(j = 1, \ldots, N_{val})$ *are defined recursively by*

$$\hat{s}_{\alpha_i^{(j)},j} = 0 \qquad \textit{if } i \geq K^* \tag{4.19}$$

$$\hat{s}_{\alpha_i^{(j)},j} = \hat{s}_{\alpha_{i+1}^{(j)},j} + \frac{\mathbb{1}[\hat{y}_{\alpha_i^{(j)}} = y_{val,j}] - \mathbb{1}[\hat{y}_{\alpha_{i+1}^{(j)}} = y_{val,j}]}{K} \frac{\min\{K, i\}}{i} \qquad \textit{if } i \leq K^* - 1 \tag{4.20}$$

where $\hat{y}_{\alpha_i^{(j)}}$ *and* $y_{val,j}$ *are the labels associated with* $\hat{x}_{\alpha_i^{(j)}}$ *and* $x_{val,j}$*, respectively. Let the true Shapley value of* $\hat{x}_{\alpha_k}$ *be denoted by* s_{α_i}. *Then,* $[\hat{s}_{\alpha_1}, \ldots, \hat{s}_{\alpha_N}]$ *is an* (ϵ, δ)*-approximation to the true Shapley value* $[s_{\alpha_1}, \ldots, s_{\alpha_N}]$.

The gist of the LSH-based approximation is to focus only on the Shapley value of the retrieved nearest neighbors and neglect the values of the rest of the training points since their values are small enough. For a error requirement ϵ not too small such that $C_{K^*} > 1$, the LSH-based approximation has sublinear time complexity, thus enjoying higher efficiency than the exact algorithm, especially for large training datasets. We provide the pseudocode for the LSH-based approximation in Algorithm 5.

A note on choosing LSH. The question on how to efficiently retrieve nearest neighbors to a query in large-scale databases has been studied extensively in the past decade. Various techniques, such as the kd-tree (Mount and Arya, 1998), LSH (Datar et al., 2004), have been proposed to find approximate nearest neighbors. Although all of these techniques can potentially help improve the efficiency of the data valuation algorithms for KNN, we focus on LSH in this chapter, as it was experimentally shown to achieve large speedup over several tree-based data structures (Datar et al., 2004; Gionis, Indyk, Motwani, et al., 1999; Har-Peled, Indyk, and Motwani, 2012).

4.3 Valuing Data for *K*NN Regressions and Extensions

In this section, we extend the exact algorithm for unweighted *K*NN to other settings. Specifically, as illustrated by Figure 4.2, whether the underlying ML model is a weighted *K*NN or unweighted; and whether the model solves a regression or a classification task. We will discuss the valuation algorithm for each of the above settings.

4.3.1 *Unweighted KNN Regression*

For regression tasks, we define the utility function by the negative mean square error of an unweighted *K*NN regressor:

$$U(S) = -\left(\frac{1}{K}\sum_{k=1}^{\min\{K,|S|\}} y_{\alpha_k(S)} - y_{\text{val}}\right)^2 \tag{4.21}$$

Using similar proof techniques to Theorem 5, we provide an efficient iterative procedure to compute the n *exact* Shapley value for unweighted *K*NN regression. The derivation of the theorem requires to analyze the utility difference between two adjacent training points, similar to *K*NN classification, which we provided in the Appendix B.4.1.

Theorem 9. *Consider the KNN regression utility function in (4.21). Then, the Shapley value of each training point can be calculated recursively as follows:*

$$s_{\alpha_N} = -\frac{K-1}{NK} y_{\alpha_N}\left[\frac{1}{K}y_{\alpha_N} - 2y_{val} + \frac{1}{N-1}\sum_{l\in I\setminus\{N\}} y_{\alpha_l}\right] - \frac{1}{N}\left[\frac{1}{K}y_{\alpha_N} - y_{val}\right]^2 \tag{4.22}$$

$$s_{\alpha_i} = s_{\alpha_{i+1}} + \frac{1}{K}(y_{\alpha_{i+1}} - y_{\alpha_i})\frac{\min\{K,i\}}{i}\left(\frac{1}{K}\sum_{l=1}^{N} A_i^{(l)} y_{\alpha_l} - 2y_{val}\right) \tag{4.23}$$

where

$$A_i^{(l)} = \begin{cases} \frac{\min\{K-1,i-1\}}{i-1} & \textit{if } 1 \leq l \leq i-1 \\ 1 & \textit{if } l \in \{i, i+1\} \\ \frac{\min\{K,l-1\}\min\{K-1,l-2\}i}{(l-1)(l-2)\min\{K,i\}} & \textit{if } i+2 \leq l \leq N \end{cases} \tag{4.24}$$

According to (4.23), two adjacent training points will have the same Shapley value if they have the same label. Otherwise, their Shapley value difference will depend on three terms:

1. their difference in the labels $y_{\alpha_{i+1}} - y_{\alpha_i}$
2. the rank of their distances to the test point $\frac{\min(K,i)}{i}$
3. the goodness of fit term $\frac{1}{K}\sum_{l=1}^{N} A_i^{(l)} y_{\alpha_l} - 2y_{\text{val}}$ of a "weighted" KNN regression model in which $A_i^{(l)}$ stands for the weight.

By simple algebraic operations, it can be obtained that y_{α_i} and $y_{\alpha_{i+1}}$ are weighted highest among all training points; therefore, the third term can be roughly thought of as how much error y_{α_i} and $y_{\alpha_{i+1}}$ induce for predicting y_{val}. If the goodness of fit term represents a positive error and $y_{\alpha_i} > y_{\alpha_{i+1}}$, then adding $(x_{\alpha_i}, y_{\alpha_i})$ into the training dataset will even enlarge the positive prediction error. Thus, $(x_{\alpha_i}, y_{\alpha_i})$ is less valuable than $(x_{\alpha_{i+1}}, y_{\alpha_{i+1}})$ in terms of the Shapley value. Similar intuition about the interaction between the first and third term can be established when $y_{\alpha_i} < y_{\alpha_{i+1}}$. Moreover, the training points closer to the validation test point are more influential to the prediction result; this phenomenon is captured by the second term. In summary, the Shapley value difference between two adjacent training points is large when their labels differ largely, their distances to the validation test point are small, and their presence in the training set leads to large prediction errors.

We present the pseudo-code for calculating exact Shapley values for KNN regressors in Algorithm 6. Similar to unweighted KNN classifiers (Algorithm 4) the computational complexity is only

$$\mathcal{O}(N \log N N_{\text{val}})$$

for N training data points and N_{val} validation data points due to sorting N_{val} arrays of N numbers.

Algorithm 6: Exact algorithm for calculating the Shapley value for an unweighted KNN regressor.

```
input  : Training data $D = \{(x_i, y_i)\}_{i=1}^N$
         Validation data $D_{\text{val}} = \{(x_{\text{val},i}, y_{\text{val},i})\}_{i=1}^{N_{\text{val}}}$
output: The Shapley value $\{s_i\}_{i=1}^N$
for $j \leftarrow 1$ to $N_{val}$ do
    $(\alpha_1, ..., \alpha_N) \leftarrow$ Indices of training data in an ascending order using
     $d(\cdot, x_{\text{val}})$;
    $C \leftarrow \frac{1}{N-1}\sum_{l\in I\setminus\{N\}} y_{\alpha_l}$;
    $s_{j,\alpha_N} = -\frac{K-1}{NK} y_{\alpha_N}\left[\frac{1}{K}y_{\alpha_N} - 2y_{\text{val}} + C\right] - \frac{1}{N}\left[\frac{1}{K}y_{\alpha_N} - y_{\text{val}}\right]^2$;
    for $i \leftarrow N-1$ to $1$ do
        $A_i^{(l)} \leftarrow$ Calculate weight according to (4.43);
        $s_{j,\alpha_i} \leftarrow s_{\alpha_{i+1}} + \frac{1}{K}(y_{\alpha_{i+1}} - y_{\alpha_i})\frac{\min\{K,i\}}{i}(\frac{1}{K}\sum_{l=1}^N A_i^{(l)} y_{\alpha_l} - 2y_{\text{val}})$;
    end
end
for $i \leftarrow 1$ to $N$ do
    $s_i \leftarrow \frac{1}{N_{\text{val}}}\sum_{j=1}^{N_{\text{val}}} s_{j,i}$;
end
```

4.3.2 *Weighted KNN*

A weighted KNN estimate produced by a training set S can be expressed as

$$\hat{y}(S) = \sum_{k=1}^{\min\{K,|S|\}} w_{\alpha_k(S)} y_{\alpha_k} \tag{4.25}$$

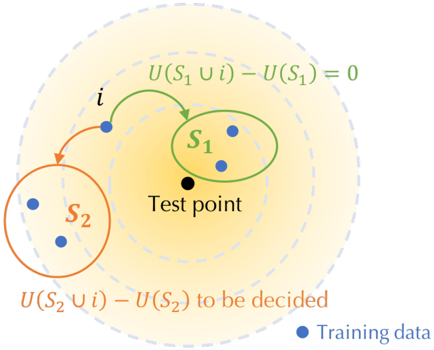

$U(S_1 \cup i) - U(S_1) = 0$
i
S_1
Test point
S_2
$U(S_2 \cup i) - U(S_2)$ to be decided
● Training data

FIGURE 4.4: Illustration of the idea to compute the Shapley value for weighted KNN.

where $w_{\alpha_k(S)}$ is the weight associated with the kth nearest neighbor of the test point in S. The weight assigned to a neighbor in the weighted KNN estimate often varies with the neighbor-to-test distance so that the evidence from more nearby neighbors are weighted more heavily (Dudani, 1976). Correspondingly, we define the utility function associated with weighted KNN classification as

$$U(S) = \sum_{k=1}^{\min\{K,|S|\}} w_{\alpha_k(S)} \mathbb{1}[y_{\alpha_k(S)} = y_{\text{val}}] \tag{4.26}$$

and for regression tasks as

$$U(S) = -\left(\sum_{k=1}^{\min\{K,|S|\}} w_{\alpha_k(S)} y_{\alpha_k(S)} - y_{\text{val}} \right)^2. \tag{4.27}$$

For weighted KNN classification and regression, the Shapley value can no longer be computed exactly in $\mathcal{O}(N \log(N))$ time. Theorem 10 shows that it is however possible to compute the exact Shapley value for weighted KNN in $\mathcal{O}(N^K)$ time.

To see the reason, we consider the 1NN example depicted in Figure 4.3 and the following inverse-distance-based weighting function:

$$w_{\alpha_k(S)} = \frac{1/d(x_{\alpha_k(S)}, x_{\text{val}})}{\sum_{k'=1}^{K} 1/d(x_{\alpha_{k'}(S)}, x_{\text{val}})} \tag{4.28}$$

We can see that for Case 1.1 and Case 2 in Figure 4.3, since i and j might have different distances to x_{val}, $w_{\alpha_k(S\cup\{i\})} \neq w_{\alpha_k(S\cup\{j\})}$ for all $k = 1, \ldots, K$. Thus, unlike the unweighted 1NN classification case, the utility difference between i and j is no longer a constant; instead, it hinges on the distance from every element in S to x_{val} and we can no longer just rely on counting Case 1.2.

Despite the difficulty in analyzing the utility difference in the weighted KNN case, we show that it is possible to compute the Shapley value in $\mathcal{O}(N^K)$ time. The theorem applies the definition in Equation (2.2) to calculating the Shapley value and relies on the following idea to circumvent the exponential complexity as illustrated by the toy example in Figure 4.4: When applying Equation (2.2) to KNN, we only need to focus on the sets S whose utility might be affected by the addition of ith training instance. Since there are only N^K possible distinctive combinations for K nearest neighbors, the number of distinct utility values for all $S \subseteq I$ is upper bounded by N^K, in contrast to 2^N for general utility functions.

Theorem 10. *Consider the utility function in (4.26) or (4.27) with some weights $w_{\alpha_k(S)}$. Let $B_k(i) = \{S : |S| = k, i \notin S, S \subseteq I\}$, for $i = 1, \ldots, N$ and $k = 0, \ldots, K$. Let $r(\cdot)$ be a function that maps the set of training data to their ranks of similarity to x_{val}. Then, the Shapley value of each training point can be calculated recursively as follows:*

$$s_{\alpha_N} = \frac{1}{N} \sum_{k=0}^{K-1} \frac{1}{\binom{N-1}{k}} \sum_{S \in B_k(\alpha_N)} \left[U(S \cup \{\alpha_N\}) - U(S) \right] \tag{4.29}$$

$$s_{\alpha_{i+1}} = s_{\alpha_i} + \frac{1}{N-1} \sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \sum_{S \in D_{i,k}} A_{i,k} \tag{4.30}$$

where

$$D_{i,k} = \begin{cases} B_k(\alpha_i) \cap B_k(\alpha_{i+1}), 0 \leq k \leq K-2 \\ B_{K-1}(\alpha_i) \cap B_{K-1}(\alpha_{i+1}, K-1 \leq k \leq N-2 \end{cases} \tag{4.31}$$

and

$$A_{i,k} = \begin{cases} 1, 0 \leq k \leq K-2 \\ \binom{N-\max r(S\cup\{\alpha_i,\alpha_{i+1}\})}{k-K+1}, K-1 \leq k \leq N-2 \end{cases} \tag{4.32}$$

Note that $|B_k(i)| \leq \binom{N-1}{k}$. Thus, the complexity for computing the weighted KNN Shapley value is at most

$$N(N-1) \times \binom{N-1}{K-1} \leq (\frac{e}{K-1})^{K-1} N^{K+1} \tag{4.33}$$

We provide the full proof in the Appendix.

4.4 Valuing Data and Computation

In this section, we categorize a data valuation problem according to whether data contributors are valued in tandem with a data analyst; whether each data contributor provides a single data instance or multiple ones.

4.4.1 *Shapley Value Computation in the Composite Game*

Oftentimes, the buyer may outsource data analytics to a third party, which we call the analyst. The analyst analyzes the training dataset aggregated from different sellers and returns an ML model to the buyer. In this process, the analyst contributes various computation efforts, which may include intellectual property pertaining to data anlytics, usage of computing infrastructure, among others. Here, we want to address the problem of appraising both sellers (data contributors) and analysts (computation contributors) within a unified game-theoretic framework.

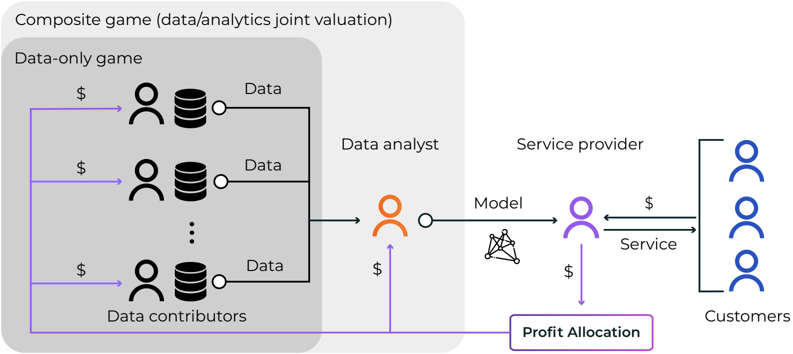


FIGURE 4.5: A description of the composite game in data valuation to jointly value data and analytics. A composite game includes the data-only game and simply extends it with an additional (critical) player, the data analyst.

Firstly, we extend the game-theoretic framework for data valuation to model the interplay between data and computation as shown in Figure 4.5. The resultant game is termed a *composite game*. By contrast, the game discussed previously which involves only the sellers is termed a *data-only game*. In the composite game, there are $M+1$ players, consisting of M sellers denoted by I^s and one analyst denoted by C. We can express the utility function U_c associated with the game in terms of the utility function U in the data-only game as follows. Since in the case of outsourced analytics, both contributions from data sellers and data analysts are necessary for building models, the value of a set $S \subseteq I^s \cup \{C\}$ in the composite game is zero if S only contains the sellers or the analyst; otherwise, it is equal to U evaluated on all the sellers in S. Formally, we define the utility function U_c by

$$U_c(S) = \begin{cases} 0, \text{ if } S = \{C\} \text{ or } S \subseteq I^s \\ U(S \setminus \{C\}), \text{ otherwise} \end{cases} \tag{4.34}$$

The goal in the composite game is to allocate $U_c(\{I^s, C\})$ to the individual sellers and the analyst. $s(U_c, i)$ and $s(U_c, C)$ represent the value received by seller i and the analyst, respectively. We suppress the dependency of s on the utility function whenever it is self-evident, denoting the value allocated to seller i and the analyst by s_i and s_c, respectively.

We show that one can compute the Shapley value for both the sellers and the analyst with the same computational complexity as the one needed for the data-only game. In the composite game, we simply extend our core proof idea as shown in Figure 4.3, by adding a new higher-level case that considers a set with and without the analyst C. We can see that the Shapley value differences are non-zero only for *Case 1.1 with the analyst*. Thus, by counting all these cases, similarly to the previous proofs, we can derive our theorems. The procedures to compute the Shapley value for unweighted KNN classifiers and regressors is shown below. We are providing the theorems for calculating the Shapley values for weighted KNN in the composite game setup, in the Appendix.

4.4.1.1 *Unweighted KNN classification*

Theorem 11. *Consider the utility function U_c in (4.34), where $U(\cdot)$ is the KNN classification performance measure in (4.1). Then, the Shapley value of each training point and the computation contributor can be calculated recursively as follows:*

$$s_{\alpha_N} = \frac{K+1}{2(N+1)N}\mathbb{1}[y_{\alpha_N} = y_{val}] \tag{4.35}$$

$$s_{\alpha_i} = s_{\alpha_{i+1}} + \frac{\mathbb{1}[y_{\alpha_i} = y_{val}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{val}]}{K} \cdot \frac{\min\{i,K\}(\min\{i,K\}+1)}{2i(i+1)} \tag{4.36}$$

$$s_C = U(I) - \sum_{i=1}^{N} s_i \tag{4.37}$$

Comparing $s(U,i)$ in Theorem 5 and $s(U_c,i)$ in the above theorem, we have

$$\frac{s(U_c,\alpha_N)}{s(U,\alpha_N)} = \frac{\min\{N,K\}+1}{2(N+1)} \tag{4.38}$$

$$\frac{s(U_c,\alpha_i) - s(U_c,\alpha_{i+1})}{s(U,\alpha_i) - s(U,\alpha_{i+1})} = \frac{\min\{i,K\}+1}{2(i+1)} \tag{4.39}$$

Note that the right-hand side of (4.38) and (4.39) are at most $1/2$ for all $i = 1, \dots, N-1$; thus, each seller will receive a much smaller share of the total revenue in the composite game than that in the data-only game. Moreover, the analyst obtains at least one half of the total revenue in the composite game setup.

4.4.1.2 *Unweighted KNN Regression*

Theorem 12. *Consider the utility function in (4.34), where $U(\cdot)$ is the KNN regression performance measure in (4.21). Then, the Shapley value of each training point and the computation contributor can be calculated recursively as follows:*

$$\begin{aligned} s_{\alpha_N} = &-\frac{1}{K(N+1)} y_{\alpha_N} \Bigg[\frac{(K+2)(K-1)}{2N} \big(\frac{1}{K} y_{\alpha_N} - 2y_{val} \big) \\ &+ \frac{2(K-1)(K+1)}{3N(N-1)} \sum_{l \in I \setminus \{\alpha_N\}} y_l \Bigg] - \frac{1}{N(N+1)} \Big[\frac{1}{K} y_{\alpha_k(N)} - y_{val} \Big]^2 \end{aligned} \tag{4.40}$$

$$s_{\alpha_i} = s_{\alpha_{i+1}} + \frac{1}{K}(y_{\alpha_{i+1}} - y_{\alpha_i}) B_i \cdot \frac{\min\{K,i\}}{i} \big(\frac{1}{K} \sum_{l=1}^{N} A_i^{(l)} y_{\alpha_l} - 2y_{val} \big) \tag{4.41}$$

$$s_C = U(I) - \sum_{i=1}^{N} s_i \tag{4.42}$$

where

$$A_i^{(l)} = \begin{cases} \frac{2}{3} \frac{\min\{K-1, i-1\}}{(i-1)} & \text{if } 1 \le l \le i-1 \\ \frac{1}{2} & \text{if } l \in \{i, i+1\} \\ \frac{2}{3} \frac{\min\{K+1,l\} \min\{K,l-1\} \min\{K-1,l-2\} i}{(l-1)(l-2) \min\{K,i\} B_i} & \text{if } i+2 \le l \le N \end{cases} \tag{4.43}$$

and $B_i = \frac{\min\{K+1, i+1\}}{i+1}$

Not surprisingly, two adjacent training points will have the same Shapley value if they have the same label. Otherwise, their Shapley value differences, as in KNN Shapley regression depend on the three terms, their difference in the labels, the rank of their distances to the test point $\frac{\min(K,i)}{i}$ and the goodness of fit term $\frac{1}{K} \sum_{l=1}^{N} A_i^{(l)} y_{\alpha_l} - 2y_{\text{val}}$ of a "weighted" KNN regression model in which $A_i^{(l)}$ stands for the weight.

The difference to the *data-only* game for both, KNN classification and regression, is that the *composite game* "reweights" the data-only Shapley values due to the adjusted utility function U_C and the addition of an additional powerful analyst C. Therefore both algorithms to calculate the Shapley values for composite games benefit from the same runtime as it would be for a data-only games.

4.4.2 *Multiple Data Per Contributor*

Now, we investigate the method to compute the Shapley value when each seller provides more than one data instance. The goal is to fairly value individual sellers for their individual training points.

NOTATION Following the previous notations, we still use $I = \{1, \ldots, N\}$ to denote the set of all training instances and use I^s to denote the set of all sellers, i.e., $I^s = \{1, \ldots, M\}$. The number of training instances owned by jth seller is N_j. We denote the ith training point contributed by jth seller as $x_j^{(i)}$. Without loss of generality, we assume that every seller's data is sorted such that

$$d(x_j^{(1)}, x_{\text{val}}) \leq \ldots \leq d(x_j^{(N_j)}, x_{\text{val}})$$

Let $h(i)$ denote the owner of ith training instance. With slight abuse of notations, we denote the owners of a set S of training instance as $h(S)$, where $S \subseteq I$, and denote the training instances from the set of sellers $\tilde{S} \subseteq I^s$ by $h^{-1}(\tilde{S})$. Let

$$\mathcal{N}(S) = \{\alpha_1(S), \ldots,\ \alpha_{\min\{K,|S|\}}(S)\}$$

be a function that maps a set of training instances to its K-nearest neighbors. Let

$$\mathcal{A} = \{S : \tilde{S} \subseteq I^s, |\tilde{S}| \leq K, S = \mathcal{N}(h^{-1}(\tilde{S}))\}$$

be the collection of all possible K-nearest neighbors formed by sellers; $|\tilde{S}| \leq K$ because the top K instances cannot belong to more than K sellers.

DATA-ONLY GAME The next theorem shows that we can compute the Shapley value of each seller for the *data-only game* with $\mathcal{O}(M^K)$.

Theorem 13. *Consider the utility functions (4.1), (4.21), (4.26) or (4.27). Let*

$$\mathcal{A}^{\backslash j} = \{S : S \in \mathcal{A}, j \notin h(S)\}$$

be the set of top-K elements that do not contain seller j's data,

$$\mathcal{D}(\tilde{S}) = \{S : S \in \mathcal{A}, h(S) = \tilde{S}\}$$

be the set of top-K elements of the data from the set $\tilde{S}$ of sellers, and

$$G(S,j) = \{j' : d(x_{j'}^{(1)}, x_{val}) \geq \max_{x \in S} d(x, x_{val}),\ S \in \mathcal{A}^{\backslash j}, j' \in I^s \setminus \{h(S), j\}\}$$

be the set of sellers that do not affect the K-nearest neighbors when added into the sellers $h(S)$ and S does not include seller j's data. Then, the Shapley value of seller j can be represented as

$$s_j = \frac{1}{M} \sum_{S \in \mathcal{A}^{\backslash j}} \sum_{k=0}^{|G(S,j)|} \frac{\binom{|G(S,j)|}{k}}{\binom{M-1}{|h(S)|+k}} [U(\mathcal{D}(h(S) \cup \{j\})) - U(S)] \tag{4.44}$$

COMPOSITE GAME Furthermore, as shown previously, we can adapt our data-only game to the *composite game*. The following theorem demonstrates that it is possible to calculate the Shapley value of every seller and the analyst for the composite game in $\mathcal{O}(M^K)$.

Theorem 14. *Consider the utility functions (4.1), (4.21), (4.26) or (4.27). Let*

$$\mathcal{A}^{\backslash j} = \{S : S \in \mathcal{A}, j \notin h(S)\}$$

be the set of top-K elements that do not contain seller j's data,

$$\mathcal{D}(\tilde{S}) = \{S : S \in \mathcal{A}, h(S) = \tilde{S}\}$$

be the set of top-K elements of the data from the set $\tilde{S}$ of sellers, and

$$G(S,j) = \{j' : d(x_{j'}^{(1)}, x_{val}) \geq \max_{x \in S} d(x, x_{val}),\ S \in \mathcal{A}^{\backslash j}, j' \in I^s \setminus \{h(S), j\}\}$$

be the set of sellers that do not affect the K-nearest neighbors when added into the sellers $h(S)$ and S does not include seller j's data. Then, the Shapley value of seller j can be represented as

$$s_j = \frac{1}{M+1} \sum_{S \in \mathcal{A}^{\backslash j}} \sum_{k=0}^{|G(S,j)|} \frac{\binom{|G(S,j)|}{k}}{\binom{M}{|h(S)|+k+1}} \left[U(\mathcal{D}(h(S) \cup \{j\})) - U(S)\right] \tag{4.45}$$

and the Shapley value of the computation contributor is

$$s_C = U(I) - \sum_{i=1}^{M} s_i \tag{4.46}$$

COMPLEXITY FOR 1NN. In the 1NN case, even though each seller can provision multiple instances, the utility function only depends on the point that is nearest to a query point in each seller's data. Thus, for 1NN, the problem of computing the multi-data-per-seller KNN Shapley value reduces to the the single-data-per-seller case; thus, the corresponding computational complexity is $\mathcal{O}(M \log M)$.

4.5 Improved MC Approximation

As discussed previously, the Shapley value for unweighted KNN classification and regression can be computed exactly with $\mathcal{O}(N \log N)$ complexity.

However, for the variants including the weighted KNN and multiple-data-per-seller KNN, the complexity to compute the exact Shapley value is $\mathcal{O}(N^K)$ and $\mathcal{O}(M^K)$, respectively, which are clearly not scalable. We propose a more efficient way to evaluate the Shapley value up to provable approximation errors, which modifies the existing MC algorithm presented in Section 3.3.1. By exploiting the locality property of the KNN-type algorithms, we propose a tighter upper bound on the number of permutations for a given approximation error and exhibit a novel implementation of the algorithm using efficient data structures.

The existing sample complexity bound is based on Hoeffding's inequality, which bounds the number of permutations needed in terms of the range of utility difference ϕ_i. This bound is not always optimal as it depends on the extremal values that a random variable can take and thus accounts for the worst case. For KNN, the utility does not change after adding training instance i for many subsets; therefore, the variance of ϕ_i is much smaller than its range. This inspires us to use Bennett's inequality, which bounds the sample complexity in terms of the variance of a random variable and often results in a much tighter bound than Hoeffding's inequality.

Theorem 15. *Given the range $[-r, r]$ of the utility difference ϕ_i, an error bound ϵ, and a confidence $1-\delta$, the sample size required such that*

$$P[\|\hat{s} - s\|_\infty \geq \epsilon] \leq \delta$$

is $T \geq T^*$. T^* is the solution of

$$\sum_{i=1}^{N} \exp(-T^*(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})) = \delta/2. \tag{4.47}$$

where $h(u) = (1+u)\log(1+u) - u$ and

$$q_i = \begin{cases} 0,\ i = 1, \dots, K \\ \frac{i-K}{i},\ i = K+1, \dots, N \end{cases} \tag{4.48}$$

Given ϵ, δ, and r, the required permutation size T^* derived from Bennett's bound can be computed numerically. For general utility functions the range r of the utility difference is twice the range of the utility function, while for the special case of the unweighted KNN classifier, $r = \frac{1}{K}$.

Although determining exact T^* requires numerical calculation, we can nevertheless gain insights into the relationship between N, ϵ, δ and T^* through some approximation. We leave the detailed derivation to Appendix B.6, but it is often reasonable to use the following $\tilde{T}$ as an approximation of T^*:

$$\tilde{T} \geq \frac{r^2}{\epsilon^2} \log \frac{2K}{\delta} \tag{4.49}$$

The sample complexity bound derived above does not change with N. On the one hand, a larger training data size implies more unknown Shapley values to be estimated, thus requiring more random permutations. On the other hand, the variance of the Shapley value across all training data decreases with the training data size, because an increasing proportion of training points makes insignificant contributions to the query result and results in small Shapley values. These two opposite driving forces make the required permutation size about the same across all training data sizes.

The algorithm for the improved MC approximation is provided in Algorithm 7. We use a max-heap to organize the KNN. Since inserting any training data to the heap costs $\mathcal{O}(\log K)$, incrementally updating the KNN in a permutation costs $\mathcal{O}(N \log K)$. Using the bound on the number of permutations in (4.49), we can show that the total time complexity for our improved MC algorithm is

$$\mathcal{O}(\frac{N}{\epsilon^2} \log K \log \frac{K}{\delta}) \tag{4.50}$$

Algorithm 7: Improved MC Approach

```
input  : Training set - $D = \{(x_i, y_i)\}_{i=1}^N$
         Utility function $U(\cdot)$
         The number of measurements - $M$
         The number of permutations - $T$
output : The Shapley value of each training point - $\hat{s} \in \mathbb{R}^N$
for $t \leftarrow 1$ to $T$ do
    $\pi_t \leftarrow$ GenerateUniformRandomPermutation$(D)$;
    Initialize a length-$K$ max-heap $H$ to maintain the $K$NN;
    for $i \leftarrow 1$ to $N$ do
        Insert $\pi_{t,i}$ to $H$;
        if $H$ changes then
            $\phi^t_{\pi_{t,i}} \leftarrow U(\pi_{t,1:i}) - U(\pi_{t,1:i-1})$;
        else
            $\phi^t_{\pi_{t,i}} \leftarrow \phi^t_{\pi_{t,i-1}}$;
        end
    end
end
$\hat{s}_i = \frac{1}{T}\sum_{t=1}^{T} \phi^t_i$ for $i = 1, \ldots, N$;
```

4.6 Experiments

We evaluate the proposed approaches to computing the Shapley value of training data for various nearest neighbor algorithms.

4.6.1 *Experimental Setup*

DATASETS We used the following popular benchmark datasets of different sizes:

(1) `dog-fish` (Koh and Liang, 2017) contains the features of dog and cat images extracted from ImageNet, with 900 training examples and 300 test examples for each class. The features have 2048 dimensions, generated by the state-of-the-art Inception v3 network (Szegedy et al., 2016b) with all but the top layer.

(2) `MNIST` (LeCun and Cortes, 2010) is a handwritten digit dataset with 60000 training images and 10000 test images. We extracted 1024-dimensional features via a convolutional network.

(3) The `CIFAR-10` dataset consists of 60000 32×32 color images in 10 classes, with 6000 images per class. The deep features have 2048 dimensions and were extracted via the ResNet-50 (K. He et al., 2016).

(4) `ImageNet` (J. Deng et al., 2009) is an image dataset with more than 1 million images organized according to the WordNet hierarchy. We chose 1000 classes which have in total around 1 million images and extracted 2048-dimensional deep features by the ResNet-50 network.

(5) Yahoo Flickr Creative Commons 100M that consists of 99.2 million photos. We randomly chose a 10-million subset (referred to as `Yahoo10m` hereinafter) for our experiment, and used the deep features extracted by (Amato et al., 2016).

PARAMETER SELECTION FOR LSH The three main parameters that affect the performance of the LSH are

m - the number of projections per hash value

h - the number of hash tables

r - the width of the projection

Decreasing r decreases the probability of collision for any two points, which is equivalent to increasing m. Since a smaller m will lead to better efficiency, we would like to set r as small as possible. However, decreasing r below a certain threshold increases the quantity $g(C_K)$, thereby requiring us to increase h. Following (Datar et al., 2004), we performed grid search to find the optimal value of r which we used in our experiments. Following (Gionis, Indyk, Motwani, et al., 1999), we set $m = \alpha \log N / \log(f_h(D_{\text{mean}})^{-1})$. For a given value of m, it is easy to find the optimal value of h which will

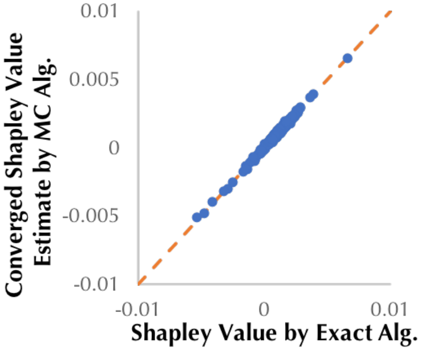


Figure 4.6: The Shapley value produced by the exact algorithm and the baseline MC approximation algorithm.

guarantee that the Shapley value approximation error is no more than a user-specified threshold. We tried a few values for α and reported the m that leads to lowest runtime. For all experiments pertaining to the LSH, we divided the dataset into two disjoint parts: one for selecting the parameters, and another for testing the performance of LSH for computing the Shapley value.

4.6.2 *Experimental Results*

4.6.2.1 *Unweighted KNN Classifier*

correctness We first empirically validate our theoretical result. We randomly selected 1000 training points and 100 test points from MNIST. We computed the Shapley value of each training point with respect to the KNN utility using the exact algorithm and the baseline MC method. Figure 4.6 shows that the MC estimate of the Shapley value for each training point converges to the result of the exact algorithm.

performance We validated the hypothesis that our exact algorithm and the LSH-based method outperform the baseline MC method. We take the approximation error $\epsilon = 0.1$ and $\delta = 0.1$ for both MC and LSH-based approximations. We bootstrapped the MNIST dataset to synthesize training datasets of various sizes. The three Shapley value calculation methods were implemented on a machine with 2.6 GHz Intel Core i7 CPU.

The runtime of the three methods for different datasets is illustrated in Figure 4.7 (a). The proposed exact algorithm is faster than the baseline approximation by several orders magnitude and it produces the exact Shapley value. By circumventing the computational complexity of sorting a large array, the LSH-based approximation can significantly outperform the exact algorithm, especially when the training size is large.

Figure 4.7 (b) sheds light on the increasing performance gap between the LSH-based approximation and the exact method with respect to the training size. The relative contrast of these bootstrapped datasets grows with the number of training points, thus requiring fewer hash tables and less time to search for approximate nearest neighbors. We also tested the approximation approach proposed in Chapter 3, which achieves the-start-of-the-art performance for ML models that cannot be incrementally maintained. However, for models that have efficient incremental training algorithms, like KNN, it is less efficient than the baseline approximation, and the experiment for 1000 training points did not finish in 4 hours.

Using a machine with the Intel Xeon E5-2690 CPU and 256 GB RAM, we benchmarked the runtime of the exact and the LSH-based approximation algorithm on three popular datasets, including CIFAR-10, ImageNet, and Yahoo10m. For each dataset, we randomly selected 100 test points, computed the Shapley value of all training points with respect to each test point, and reported the average runtime across all test points.

The results for $K = 1$ are reported in Figure 4.8. We can see that the LSH-based method can bring a 3×-5× speed-up compared with the exact algorithm. The performance of LSH depends heavily on the dataset, especially in terms of its relative contrast. This effect will be thoroughly studied in the sequel. We compare the prediction accuracy of KNN ($K = 1, 2, 5$) with the commonly used logistic regression and the result is illustrated in

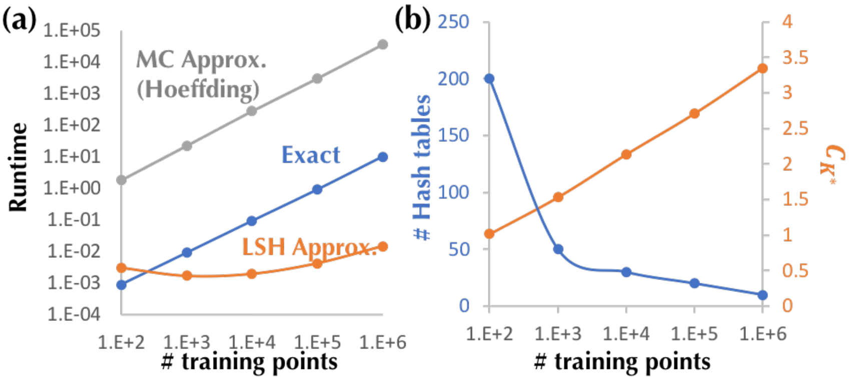


FIGURE 4.7: Performance of unweighted KNN classification in the single-data-per-seller case.

Dataset	Size	Estimated	Runtime	Runtime
		Contrast	(Exact)	(LSH)
`CIFAR-10`	6E+4	1.2802	0.78s	0.23s
`ImageNet`	1E+6	1.2163	11.34s	2.74s
`Yahoo10m`	1E+7	1.3456	203.43s	44.13s

FIGURE 4.8: Average runtime of the exact and the LSH-based approximation algorithm for computing the unweighted KNN Shapley value for a single test point. We take $\epsilon, \delta = 0.1$ and $K = 1$.

Figure 4.9. We can see that KNN achieves comparable prediction power to logistic regression when using features extracted via deep neural networks. The runtime of the exact and the LSH-based approximation for $K = 2, 5$ is similar to the $K = 1$ case in Figure 4.8, so we will leave their corresponding results to Appendix B.1.1.

EFFECT OF RELATIVE CONTRAST ON THE LSH-BASED METHOD Our theoretical result suggests that the K^*th relative contrast where $K^* = \max\{K, \lceil 1/\epsilon \rceil\}$ determines the complexity of the LSH-based approximation. We verified the effect of relative contrast by experimenting on three datasets, namely, `dog-fish`, `deep` and `gist`. `deep` and `gist` were constructed

Dataset	1NN	2NN	5NN	Logistic Regression
CIFAR-10	81%	83%	80%	**87%**
ImageNet	77%	73%	**84%**	82%
Yahoo10m	90%	96%	**98%**	96%

FIGURE 4.9: Comparison of prediction accuracy of KNN vs. logistic regression on deep features.

by extracting the deep features and gist features (Siagian and Itti, 2007) from MNIST, respectively. All of these datasets were normalized such that $D_{\text{mean}} = 1$.

Figure 4.10 (a) shows that the relative contrast of each dataset decreases as K^* increases. In this experiment, we take $\epsilon = 0.01$ and $K = 2$, so the corresponding $K^* = 1/\epsilon = 100$. At this value of K^*, the relative contrast is in the following order: deep (1.57) $>$ gist (1.48) $>$ dog-fish (1.17).
From Figure 4.10 (b) and (c), we see that the number of hash tables and the number of returned points required to meet the ϵ error tolerance for the three datasets follow the reversed order of their relative contrast, as predicted by Theorem 8. Therefore, the LSH-based approximation will be less efficient if the K in the nearest neighbor algorithm is very large or the desired error ϵ is small.
Figure 4.10 (d) shows that the LSH-based method can better approximate the true Shapley value as the recall of the underlying nearest neighbor retrieval gets higher. For the datasets with high relative contrast, e.g., deep and gist, a moderate value of recall (~ 0.7) can already lead to an approximation error below the desired threshold. On the other hand, dog-fish, which has low relative contrast, will need fairly accurate nearest neighbor retrieval (recall ~ 1) to obtain a tolerable approximation error.

The reason for the different retrieval accuracy requirements is that for the dataset with higher relative contrast, even if the retrieval of the nearest neighbors is inaccurate, the rank of the erroneous elements in the retrieved set may still be close to that of the missed true nearest neighbors. Thus,

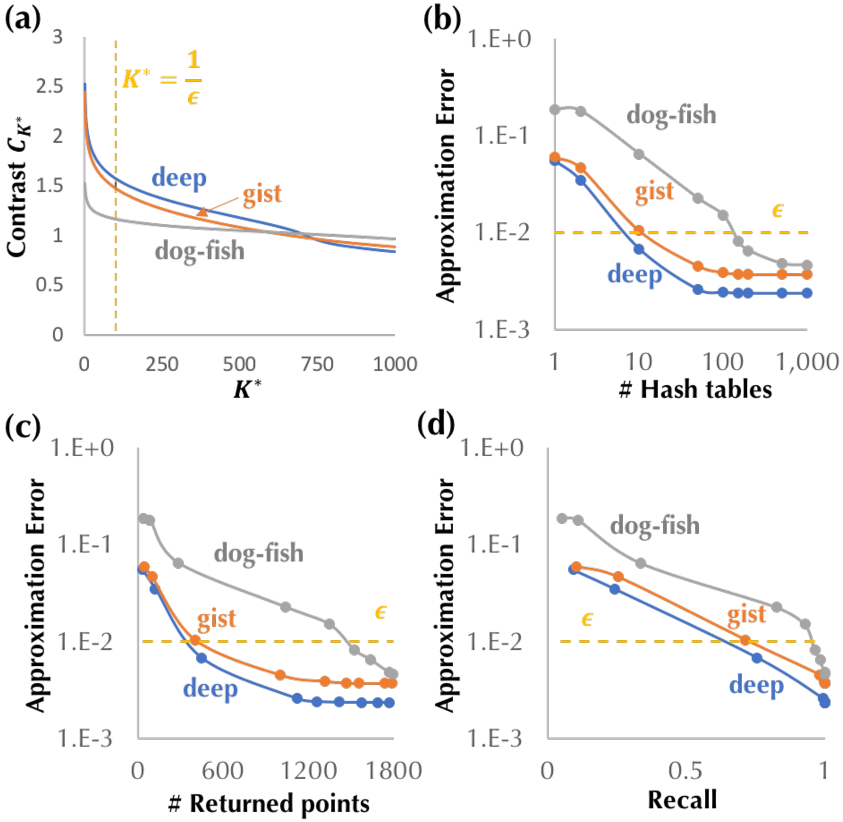


FIGURE 4.10: Performance of LSH on three datasets: `deep`, `gist`, `dog-fish`. (a) Relative contrast C_{K^*} vs. K^*. (b), (c) and (d) illustrate the trend of the Shapley value approximation error for different number of hash tables, returned points and recalls.

these erroneous elements will have only little impacts on Shapley value approximation errors.

SIMULATION OF THE THEORETICAL BOUND OF LSH According to Theorem 8, the complexity of the LSH-based approximation is dominated by the exponent $g(C_{K^*})$, where $K^* = \min\{K, 1/\epsilon\}$ and $g(\cdot)$ depends on the width r of the p-stable distribution used for LSH. We computed C_{K^*}

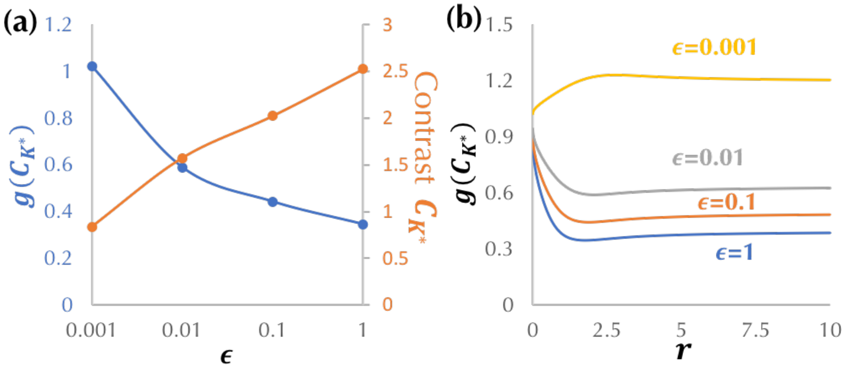


FIGURE 4.11: (a) The exponent $g(C_{K^*})$ in the complexity bound of the LSH-based method and the relative contrast C_{K^*} computed for different ϵ. K is fixed to 1. (b) $g(C_{K^*})$ vs. the projection width r of the LSH.

and $g(C_{K^*})$ for $\epsilon \in \{0.001, 0.01, 0.1, 1\}$ and let $K = 1$ in this simulation. The orange line in Figure 4.11 (a) shows that a larger ϵ induces a larger value of relative contrast C_{K^*}, rendering the underlying nearest neighbor retrieval problem of the LSH-based approximation method easier. In particular, C_{K^*} is greater than 1 for all epsilons considered except for $\epsilon = 0.001$. Recall that $g(C_K) = \log f_h(1/C_K)/\log f_h(1)$; thus, $g(C_{K^*})$ will exhibit different trends for the epsilons with $C_{K^*} > 1$ and the ones with $C_{K^*} < 1$, as shown in Figure 4.11 (b). Moreover, Figure 4.11 (b) shows that the value of $g(C_{K^*})$ is more or less insensitive to r after a certain point. For ϵ that is not too small, we can choose r to be the value at which $g(C_{K^*})$ is minimized. It does not make sense to use the LSH-based approximation if the desired error ϵ is too small to have the corresponding $g(C_{K^*})$ less than one, since its complexity is theoretically higher than the exact algorithm. The blue line in Figure 4.11 (a) illustrates the exponent $g(C_{K^*})$ as a function of ϵ when r is chosen to minimize $g(C_{K^*})$. We observe that $g(C_{K^*})$ is always below 1 except when $\epsilon = 0.001$.

4.6.2.2 *Evaluation of Other Extensions*

We introduced the extensions of the exact Shapley value calculation algorithm to the settings beyond unweighted KNN classification. Some of these settings require polynomial time to compute the exact Shapley value, which is impractical for large-scale datasets. For those settings, we need to resort to the MC approximation method. We first compare the sample complexity of different MC methods, including the baseline and our improved MC method (Section 4.5). Then, we demonstrate data values computed in various settings.

SAMPLE COMPLEXITY FOR MC METHODS The time complexity of the MC-based Shapley value approximation algorithms is largely dependent on the number of permutations. Figure 4.12 compares the permutation sizes used in the following three methods against the actual permutation size needed to achieve a given approximation error (marked as "ground truth" in the figure):

(1) "Hoeffding", which is the baseline approach and uses the Hoeffding's inequality to decide the number of permutations

(2) "Bennett", which is our proposed approach and exploits Bennett's inequality to derive the permutation size;

(3) "Heuristic", which terminates MC simulations when the change of the Shapley value estimates in the two consecutive iterations is below a certain value, which we set to $\epsilon/50$ in this experiment.

We notice that the ground truth requirement for the permutation size decreases at first and remains constant when the training data size is large enough. From Figure 4.12, the bound based on the Hoeffding's inequality is too loose to correctly predict the correct trend of the required permutation size. By contrast, our bound based on Bennett's inequality exhibits the correct trend of permutation size with respect to training data size. In terms of runtime, our improved MC method based on Bennett's inequality is more than $2\times$ faster than the baseline method when the training size is above 1 million. Moreover, using the aforementioned heuristic, we were able to

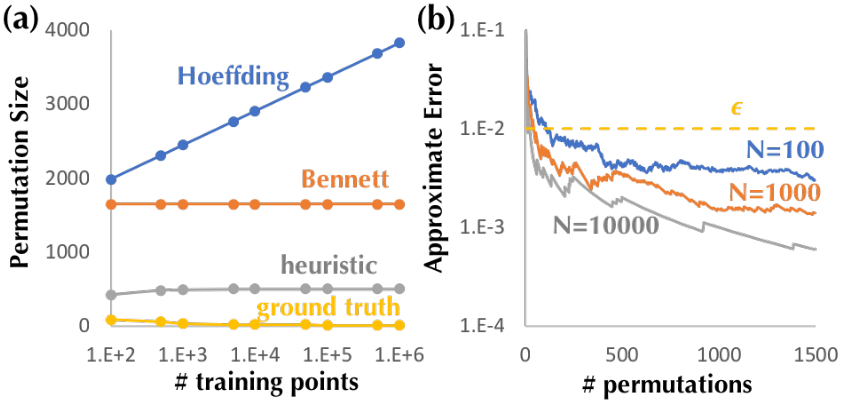


Figure 4.12: Comparison of the required permutation sizes for different number of training points derived from the Hoeffding's inequality (baseline), Bennett's inequality and the heuristic method against the ground truth.

terminate the MC approximation algorithm even earlier while satisfying the requirement of the approximation error.

PERFORMANCE We conducted experiments on the `dog-fish` dataset to compare the runtime of the exact algorithm and our improved MC method. We took $\epsilon = 0.01$ and $\delta = 0.01$ in the approximation algorithm and used the heuristic to decide the stopping iteration.

Figure 4.13 compares the runtime of the exact algorithm and our improved MC approximation for weighted KNN classification. In the first plot, we fixed $K = 3$ and varied the number of training points. In the second plot, we set the training size to be 100 and changed K. We can see that the runtime of the exact algorithm exhibits polynomial and exponential growth with respect to the training size and K, respectively. By contrast, the runtime of the approximation algorithm increases slightly with the number of training points and remains unchanged for different values of K.

Figure 4.14 compares the runtime of the exact algorithm and the MC approximation for the unweighted KNN classification when each seller can

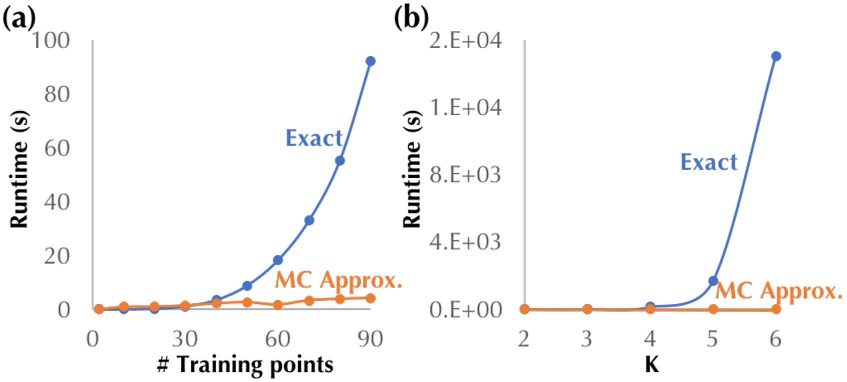


FIGURE 4.13: Performance of the weighted KNN classification.

own multiple data instances. To generate Figure 4.14 (a), we set $K = 2$ and varied the number of sellers. We kept the total number of training instances of all sellers constant and randomly assigned the same number of training instances to each seller. We can see that the exact calculation of the Shapley value in the multi-data-per-seller case has polynomial time complexity, while the runtime of the approximation algorithm barely changes with the number of sellers. Since the training data in our approximation algorithm were sequentially inserted into a heap, the complexity of the approximation algorithm is mainly determined by the total number of training data held by all sellers. Moreover, as we kept the total number of training points constant, the approximation algorithm appears invariant over the number of sellers.

Figure 4.14 (b) shows that the runtime of exact algorithm increases with K, while the approximation algorithm's runtime is not sensitive to K. To summarize, the approximation algorithm is preferable to the exact algorithm when the number of sellers and K are large.

UNWEIGHTED VS. WEIGHTED KNN SHAPLEY VALUE We constructed an unweighted KNN classifier using the `dog-fish`. Figure 4.15 (a) illustrates the training points with top KNN Shapley values with respect to a specific test image.

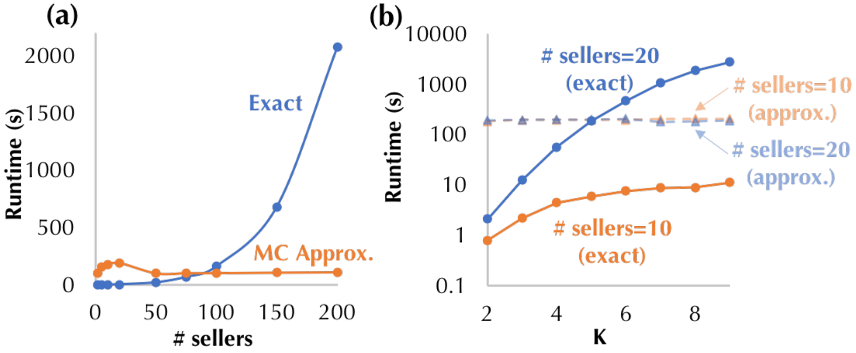


FIGURE 4.14: Performance of the KNN classification in the multi-data-per-seller case.

We see that the returned images are semantically correlated with the test one. We further trained a weighted KNN on the same training set using the weight function that weighs each nearest neighbor inversely proportional to the distance to a given test point; and compared the Shapley value with the ones obtained from the unweighted KNN classifier. We computed the average Shapley value across all test images for each training point and demonstrated the result in Figure 4.15 (b). Every point in the figure represents the Shapley values of a training point under the two classifiers. We can see that the unweighted KNN Shapley value is close to the weighted one. This is because in the high-dimensional feature space, the distances from the retrieved nearest neighbors to the query point are large, in which case the weights tend to be small and uniform. Another observation from Figure 4.15 (b) is that the KNN Shapley value assigns more values to dog images than fish images.

Figure 4.15 (c) plots the distribution of the number test examples with regard to the number of their top-K neighbors in the training set are with a label inconsistent with the true label of the test example. We see that most of the nearest neighbors with inconsistent labels belong to the fish class. In other words, the fish training images are more close to the dog images in the test set than the dog training images to the test fish. Thus, the fish training images are more susceptible to mislead the predictions and should

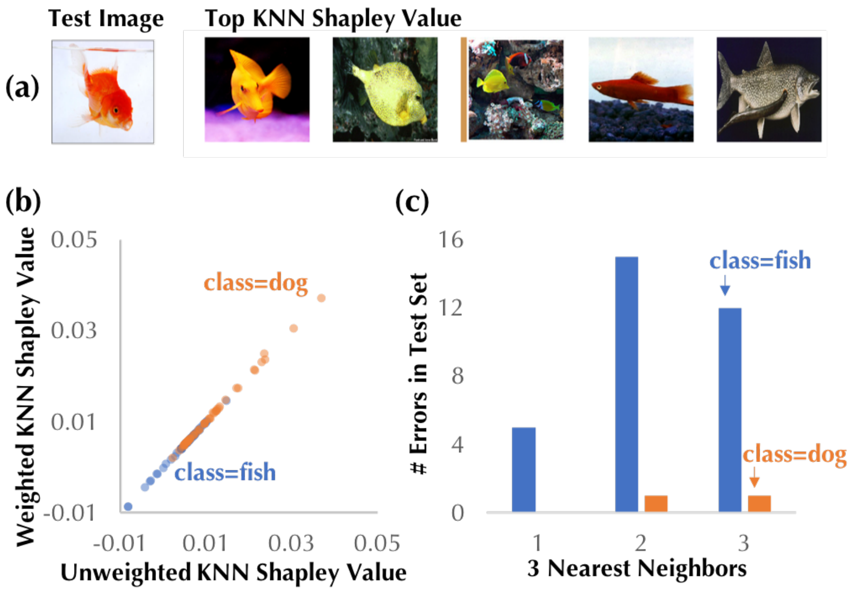


FIGURE 4.15: Data valuation on DOG-FISH dataset ($K = 3$). (a) top valued data points; (b) unweighted vs. weighted KNN Shapley value on the whole test set; (c) Per-class top-K neighbors labeled inconsistently with the misclassified test example.

have lower values. This intuitively explains why the KNN Shapley value places a higher importance on the dog images.

DATA-ONLY VS. COMPOSITE GAME We introduced two game-theoretic models for distributing the gains from an ML model and would like to understand how the shares of the analyst and the data contributors differ in the two models. We constructed an unweighted KNN classifier with $K = 10$ on the `dog-fish` dataset and compute the Shapley value of each player in the data-only and the composite game. Recall that the total utility of both games is defined as the average validation accuracy trained on the full set of training data.

Figure 4.16 (a) shows that the Shapley value for the analyst increases with the total utility. Therefore, under the composite game formulation, the analyst has huge incentive to train a good ML model as the values assigned to the analyst gets larger with a better ML model. In addition, in the composite game formulation, the analyst has exclusive control over the computational resources and the data only creates value when it is analyzed with computational modules, the analyst should take the greatest share of the utility extracted from the ML model. This intuition is reflected in Figure 4.16 (a).

Figure 4.16 (b) demonstrates that the Shapley value of the data contributors in the composite game is correlated with that in the data-only game, although the actual value is much smaller.

Figure 4.16 (c) exhibits the trend of the Shapley value of the analyst and data contributors as more data contributors participate in a data transaction. The Shapley value of the analyst gets larger with more data contributors, while the average value obtained by each data contributor decreases in both composite and data-only games.

Figure 4.16 (d) zooms into the change of the maximum and minimum value among all data contributors in the data-only game setting (the result in the composite game setting is similar). We can see that both the maximum and minimum value decreases at the beginning; as more data contributors are involved in a data transaction, the minimum value demonstrates a small increment. The points with lowest values tend to hurt the ML model performance when they are added into the training set. With more data contributors and more training points, the negative impacts of these "outliers" can get mitigated.

4.7 Discussion

REMARKS We summarize several takeaways from our experimental evaluation.

(1) For unweighted KNN classifiers, the LSH-based approximation is more preferable than the exact algorithm when a moderate amount of approximation error can be tolerated and K is relatively small. Otherwise, it is recommended to use the exact algorithm as a default approach for data valuation.

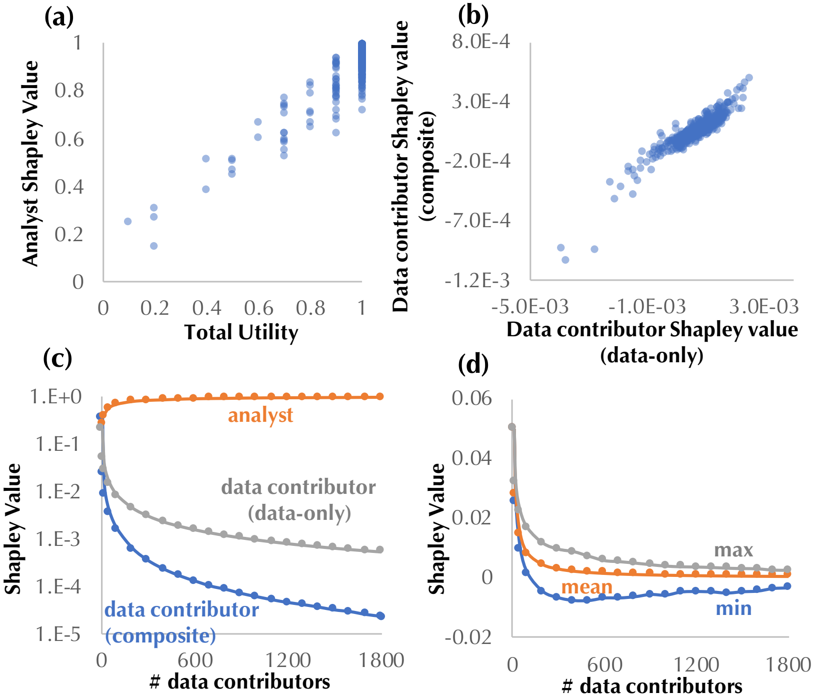


FIGURE 4.16: (a) The Shapley value of the analyst in the composite game vs. total utility obtained from the ML model; (b) the correlation between the data contributors' Shapley value in the composite game with that in the data-only game; (c) The Shapley value of all players in the two games for different number of data contributors; (d) The mean, maximum, minimum of the data contributors' Shapley values in the data-only game.

(2) For weighted KNN regressors or classifiers, computing the exact Shapley value has $\mathcal{O}(N^K)$ complexity, thus not scalable for large datasets and large K. Hence, it is recommended to adopt the Monte Carlo method in Algorithm 7
(3) Setting up a data valuation problem as a data-only game or a composite game presents no difference in computational complexity. In both setups, the relative contribution of each data contributor decreases with the number of data contributors. In the composite game setup, the data analyst will

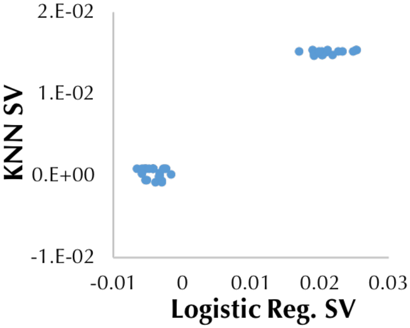


Figure 4.17: Comparison of the Shapley value for a logistic regression and a KNN trained on the `Iris` dataset.

account for the largest proportion of the total profit generated by the model. Moreover, using the heuristic based on the change of Shapley value estimates in two consecutive iterations to decide the termination point of the algorithm is much more efficient than using the theoretical bounds, such as Hoeffding or Bennett.

computing the shapley value for models beyond Knn The efficient algorithms presented in this chapter are possible only because of the "locality" property of KNN. However, given many previous empirical results showing that a KNN classifier can often achieve a classification accuracy that is comparable with classifiers such as SVMs and logistic regression given sufficient memory, we could use the KNN Shapley value as a proxy for other classifiers.

We compute the Shapley value for a logistic regression classifier and a KNN classifier trained on the same dataset namely `Iris`, and the result shows that the Shapley values under these two classifiers are indeed correlated (see Figure 4.17). The only caveat is that KNN Shapley value does not distinguish between neighboring data points that have the same label. If this caveat is acceptable, we believe that the KNN Shapley value provides an efficient way to approximately assess the relative contribution of different data points for other classifiers as well.

Moreover, for calculating the Shapley value for general deep neural networks, we can take the deep features (i.e., the input to the last softmax layer) and corresponding labels, and train a KNN classifier on the deep features. We calibrate K such that the resulting KNN mimics the performance of the original deep net and then employ the techniques presented in this chapter to calculate a surrogate for the Shapley value under the deep net.

4.8 Summary

We previously introduced the Shapley value as a valuable economic concept for quantifying the value of data. However - even with improved estimation algorithms - its practical application has been limited so far by the challenge of dealing with exponential computational complexity. This issue becomes particularly acute in real-world data valuation settings that involve enormous datasets with billions of data points. In this chapter, we focus on one popular family of ML models relying on K-nearest neighbors (KNN) and provide an efficient algorithm to calculate the *exact* Shapley values for unweighted KNN classification. We show that the exact algorithm and the approximate algorithm using LSH can scale to millions of data points and is thus suitable for the above mentioned challenge. We then extend our algorithms to other scenarios such as (1) weighed KNN classifiers, (2) different data points are clustered by different *data curators*, and (3) there are *data analysts* providing computation who also requires proper valuation. *Some* of these extensions, although also being improved exponentially, are less practical for exact computation (e.g., $O(N^K)$ complexity for weighted KNN). We thus propose a Monte Carlo approximation algorithm, which is $O(N(\log N)^2/(\log K)^2)$ times more efficient than the baseline approximation algorithm.

5

VALUING CONTRIBUTORS IN PRIVATE DATA MARKETS AND DATA CURATION

All theory, dear friend, is gray, but the golden tree of life springs ever green

— Johann Wolfgang von Goethe

5.1 Introduction

Machine learning (ML) systems benefit from large quantities of diverse training data. In recent years, numerous initiatives have attempted to build data marketplaces, as a way for individuals and organizations to share, buy, and sell data (McConaghy, 2022; Azcoitia and Laoutaris, 2022). Nevertheless, multiple aspects of the design of such a marketplace remain unclear, including the transfer of data and payments, the handling of potentially private and sensitive data, the potential need for data curation and cleaning, and the applicability of impactful real-world use cases. In this Chapter, we we focus on two settings (see Figure 5.1): First, a data marketplace tailored for private data in healthcare, and second, a mutually beneficial curated dataset for forest carbon.

DATA MARKETPLACE FOR PRIVATE DATA. First, let us revisit the introductory toy example from Chapter 1. Imagine that instead of training a dog and fish classifier, the data consumer is aiming to develop a machine learning model for healthcare purposes that requires sensitive information, such as patient healthcare records. Additionally, the data consumer is reluctant to disclose or share his model weights for training purposes (e.g. due to IP concerns). How will the exchange of private data be managed in this context? Collecting high-quality data sets, especially for sensitive data, is

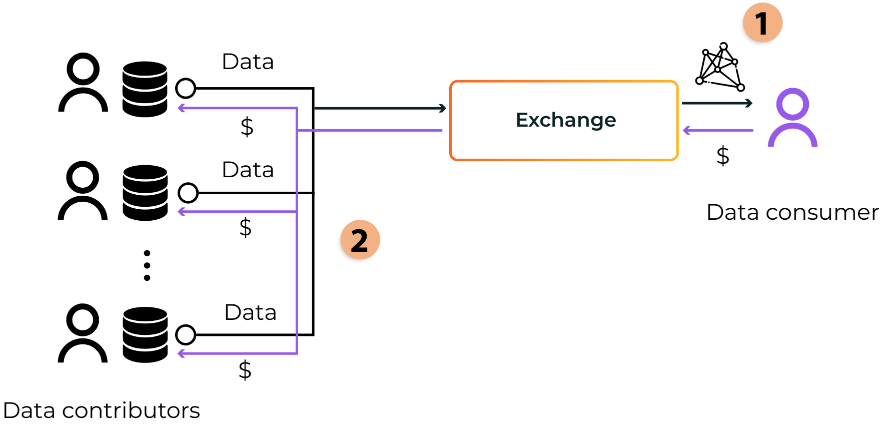


Figure 5.1: We will study two use cases of a data marketplace to guide our implementation: 1) How can we train an ML model without revealing the data to the data consumer, nor the model parameters to the data contributors? 2) How can we incentivize and curate data contributions effectively?

challenged by data regulatory and ethical privacy requirements. Thus, the development and implementation of such platforms for private data are not without challenges which include, but are not restricted to:

Data privacy and security. Protecting sensitive information while enabling data exchange is a critical concern for data marketplaces. E.g. in the context of healthcare, providing access to patient data can enable the training of ML models that enable more accurate diagnoses. However, sharing this sensitive data can potentially expose patients' private information, which could lead to ethical issues, privacy breaches and non-compliance with data protection regulations such as HIPAA in the United States or GDPR in the European Union.

Valuing data ahead of time. As discussed in previous chapters, one important component of a data marketplace involves the creation of robust, efficient, and task-specific data valuation methods to accurately determine the value of individual data points. However, a data consumer would often like to assess ahead of time if the data is of value, which might not be possible in the case of sensitive data. Thus it is unclear how to integrate data valuation into a data marketplace for private data.

Data governance and ownership. An important question in the design of data marketplaces is the question of who owns the data and who can use it. Addressing these issues can help ensure that that sensitive data usage aligns with the values and preferences of the data contributors, while also preventing unauthorized or unethical practices from any data consumer. For instance, a patient may choose to allow their health records to be only used for research on disease classification, but they might prohibit its use for any commercial purposes, such as targeted advertising.

In this Chapter, we want to address the above mentioned challenges by studying the question:

Challenge 1

> *What are the key design considerations required to develop a data marketplace that effectively handles private data while maintaining privacy and governance?*

INCENTIVIZING REAL-WORLD DATA CURATION. In a data marketplace, the mutual benefit of data contributors and consumers is an important aspect that drives the success and sustainability of the ecosystem. This is particularly relevant in data collection and curation scenarios, where datasets do not readily exist, and their creation depends upon connecting them to real-world use cases and benefits. Providing payments and incentives to data contributors, as well as offering compelling reasons for data consumers to support the growth of the dataset, is therefore important to understand. Additionally, data payments and market mechanisms can offer novel opportunities to involve local communities and individuals, who are traditionally not part of the machine learning development process, to benefit from the existing data economy. Consequently, our aim is to incentivize an impactful real-world example that highlights the mutual benefits for data contributors and consumers as well as challenges when working with the resulting data.

Challenge 2

> *How can we effectively incentivize data contributions, engage local communities, and provide mutual benefits for data contributors and consumers in real-world machine learning applications?*

5.1.1 *Contributions*

CONTRIBUTION 1 To address the first challenge, we propose Sterling, a data marketplace for private datasets. Our approach combines blockchain smart contracts, trusted execution environments (e.g., Intel SGX (Anati et al., 2013), Sanctum (Lebedev, Hogan, and Devadas, 2018), Keystone (D. Lee et al., 2020)), and differential privacy, to offer strong security and privacy guarantees for user data and machine learning models. Smart contracts allow the enforcement of data contributors' constraints on how their data is used. For example, they can require analytics performed on their data to be differentially private. Smart contracts also enable users to define payments and rewards. By leveraging privacy-preserving smart contracts running in trusted execution environments, we can compute analytics and train machine learning models while keeping all data and models private. Sterling thus enables mutually distrusting parties to collaboratively train privacy-preserving machine learning models, compensating parties while keeping their data private. Specifically we make the following technical contributions:

C1.1 Smart contract framework We present a framework supporting generic data contributor and data consumer smart contracts which uphold their creators' interests.

C1.2 Contributor-defined terms of usage We provide a mechanism for data contributors to control the use of their data through automatic verification of data consumer contracts, allowing contributors to express constraints such as pricing and differential privacy.

C1.3 Privacy-preserving ML training Sterling enables privacy-preserving distribution and use of data. We achieve this by preventing data contributors and data consumers from directly accessing each other's respective private data and models.

C1.4 Use case for medical diagnosis We provide a concrete demonstration of the aforementioned contributions by applying them to the task of medical diagnosis in ophthalmology.

Sterling is the result of joint work with my co-authors and has been previously presented as a VLDB demo.

Nick Hynes, **David Dao**, David Yan, Raymond Cheng, Dawn Song "A demonstration of Sterling: A Privacy-Preserving Data Marketplace". *VLDB Demo*. 2018.

CONTRIBUTION 2 To study the second challenge, we present a real-world use case with mutual benefit: ReforesTree, a benchmark dataset for forest carbon stock prediction that encompasses 6 (agro-)forestry carbon offsetting sites and more than 4463 individual tree measurements which has been collected on the ground by 18 community members. We incentivized the creation of this specific dataset by compensating local community members with 1$ for every 3 trees collected. ReforesTree proved useful for ML practitioners in accurately estimating carbon stocks and detecting overestimation in existing satellite-based estimations, emphasizing the need for continued collection and a mutual benefit of data contributors and consumers. We provide four technical contributions:

C2.1 Data contributor framework We present a low-cost framework for collecting tree measurements, utilizing mobile applications and low-cost drones for data entry in remote regions, and ensuring data quality and consistency throughout the process. The framework enables local and Indigenous communities, who conduct the field work to receive payments for data collection efforts.

C2.2 Data processing pipeline We develop a robust data processing pipeline that extracts the GPS coordinates the collected field and drone images and transforming it into a format suitable for machine learning algorithms.

C2.3 Satellite-based evaluation Surprisingly, based on the ReforesTree dataset, our analysis shows that existing forest carbon estimates from satellite imagery can overestimate above-ground biomass by up to 10-times for tropical reforestation projects.

C2.4 ML models for carbon stock estimation We show that a deep learning-based end-to-end model using individual tree detection from low cost RGB-only drone imagery is accurately estimating forest carbon stock within official carbon offsetting certification standards. Additionally, our baseline ML model outperforms state-of-the-art satellite-based forest biomass and carbon stock estimates for this type of small-scale, tropical agro-forestry sites.

ReforesTree is the result of joint work co-led by the author and has been previously published at AAAI.

Gyri Reiersen, **David Dao**, Björn Lütjens, Konstantin Klemmer, Kenza Amara, Attila Steinegger, Ce Zhang, Xiaoxiang Zhu. "ReforesTree: A Dataset for Estimating Tropical Forest Carbon Stock with Deep Learning and Aerial Imagery". *AAAI*. 2022.

5.1.2 *Overview*

In this chapter, we first present Sterling, a data marketplace for private data, and outline our method in Sterling for automatically enforcing data contributor constraints (Section 5.2.2), the resulting data economy (Section 5.2.3), and our implementation of a comprehensive privacy-preserving machine learning pipeline (Section 5.2.4). Next, we discuss the Sterling demonstration use case, which involves training a disease prediction model using electronic health records (Section 5.2.5).

Following that, we introduce ReforesTree, a benchmark dataset for carbon estimation created through field measurements by local communities. We describe the mutual advantages of this dataset for both data contributors and users (Section 5.3.1), the essential collaboration with data contributors (Section 5.3.2), and the necessary data processing to make the dataset ready for analysis (Section 5.3.5). Finally, we conclude with experiments that demonstrate the value of such a benchmark dataset in the development of ML models (Section 5.3.6) and provide an overall summary (Section 5.4).

5.2 Sterling: A Data Marketplace for Private Data

There have been several attempts at creating distributed AI and data marketplaces for public datasets, some of which are implemented as *smart contracts* on distributed ledgers known as blockchains. Although smart contracts enable reaching consensus on the result of a computation, current mechanisms for verifying correctness requires public disclosure of contract inputs and state. This poses a difficulty for data marketplaces since any user of the blockchain can directly view and copy the data and models. Furthermore, even in the benign case, there is no way to ensure that data are not used in a manner that conflicts with its contributor's constraints (e.g., using biometric data to train ad-serving models).

To address this question, we demonstrate Sterling, a decentralized marketplace for private data. Sterling enables privacy-preserving distribution and use of data by using *privacy-preserving smart contracts* which run on a permissionless blockchain. These smart contracts, written by data contributors and consumers, immutably and irrevocably represent the interests of their creators. In particular, we provide a mechanism for data contributors to control the use of their data through automatic verification of data consumer contracts, allowing contributors to express constraints such as pricing and differential privacy.

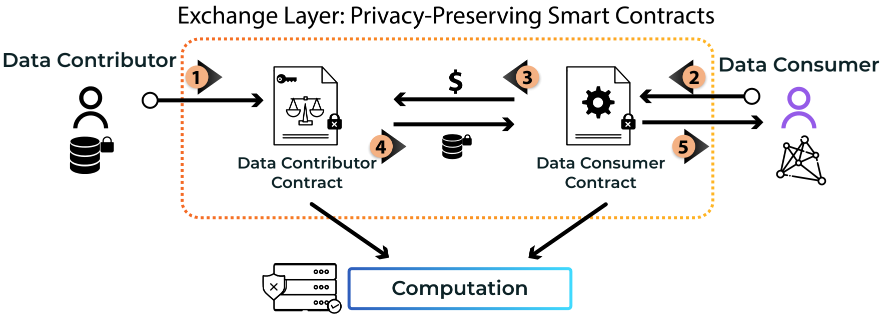

Exchange Layer: Privacy-Preserving Smart Contracts
Data Contributor
Data Consumer
1
2
3
4
5
$
Data Contributor Contract
Data Consumer Contract
Computation

FIGURE 5.2: Diagram of the interaction between data producers and consumers in the Sterling marketplace. The economic and privacy interests of each party is mediated and enforced using privacy-preserving smart contracts. The circled numbers refer to steps of the workflow described in Section 5.2.1.

5.2.1 *The Sterling Marketplace*

Let us revisit our motivating example of a medical researcher wishing to train a predictive model of disease. Currently, this would require a lengthy process of negotiating with hospitals for data (Rajkomar et al., 2018). Obtaining a truly representative dataset may require collaborations with clinics across the globe. Instead Sterling, a privacy-preserving data marketplace, allows individuals to provide their EHR data for direct use by researchers and organizations. Thus, individuals can realize the economic value of their data without compromising privacy. Notably, this application is unsuitable for marketplaces built using public smart contracts as leakage of a *single* record can compromise an individual's privacy.

Generally, we seek to provide the following workflow (Figure 5.2):

1. A data contributor, U_d, uploads encrypted data to a centralized or decentralized storage service (e.g., AWS, IPFS, Swarm). U_d publishes a smart contract C_d containing the address of the data and, optionally, constraints like payment or privacy requirements. U_d provisions C_d with a data decryption key which is privately stored by the contract.

2. A data consumer, U_c, desiring to use provided data writes a smart contract C_c which satisfies the constraints of C_d.
3. U_c invokes C_c which sends a signed request, attesting to its identity, to C_d.
4. C_d automatically verifies that C_c satisfies the constraints and securely returns a data decryption key.
5. C_c performs its computation on the decrypted data and U_d is compensated according to the terms of use.

Enabling a secure protocol like the above is challenging due to attacks on data/model privacy and we will explain the following each component in detail.

5.2.2 *Automatically Enforcing Terms of Use*

A primary contribution of the Sterling marketplace is the ability for data contributors to impose *terms of use*, or constraints, on the use of their data. In our system, the privacy-preserving smart contracts are programmed in a general-purpose language (e.g., Rust, JavaScript). Thus, data contributors can encode flexible requirements within the Sterling framework. Perhaps the simplest term of use is requiring payment for each use of the data. Recalling the motivating example, a more nuanced term might be that a consumer contract bearing the cryptographic signature of a hospital receives the EHR data for free.

By executing the contracts on a blockchain, Sterling ensures correct autonomous execution of smart contracts. The high availability and immutability of the blockchain ensures that consensus on the correct enforcement of constraints is achieved. Indeed, Byzantine fault tolerance guarantees that if a consumer contract is run on constrained data, then the consumer contract has been verified by a majority of participants to satisfy the constraints. Sterling runs on the Oasis blockchain platform which extends Ekiden (Cheng et al., 2018) and provides privacy-preserving smart contracts.

5.2.2.1 *Terms of Use for Training ML Models*

In our initial system, we focus on the constraints of payment and differential privacy (Dwork, Rothblum, and Vadhan, 2010) of models trained on the data. In both cases, our approach relies on static analysis to ensure that the data consumer contract satisfies the constraints of data contributor contract. For ensuring differential privacy, we provide functionality for training differentially private ML models like logistic regression and neural networks using stochastic gradient descent (Abadi et al., 2016). We use techniques from Optio (Near et al., 2018) to perform privacy-aware type checking of a consumer contract's model definition, so to ensure that it satisfies differential privacy.

We further describe differential privacy constraints in Section 5.2.4.2. Since Sterling supports flexible logic (which includes calls to other contracts), a data contributor can straightforwardly create additional, custom constraints within the general framework.

5.2.3 *Data Economics*

In general, the data economy is governed by the terms of use set by data contributors. Since data contributors are free to create additional contributor contracts, they can re-share data under modified terms of use–for instance, lowering the price to reflect other contributors' actions in the marketplace.

A main challenge of working with private data is that the consumer is unable to determine ahead of time that the data are of value. In a benign case, a data contributor may simply offer poor documentation. An adversarial contributor, however, may attempt to defraud buyers by submitting random noise or even plausible fake data. Thus it would be advantageous–indeed essential–for a data consumer contract to automatically determine the value of the data it receives. Generally, appraising data requires domain specific knowledge of what constitutes *good* data. In the section to follow, we present as examples techniques usable in machine learning applications.

Assuming that data consumers are able to verify the utility of data, the economics of the market ensure that the objectives of contributors and consumers are aligned. For example, an adversary might submit fake data with the constraint that payment be made upfront, but no rational data consumer would use the data without first verifying its utility. Conversely, an honest data contributor would not want their data to be used without payment, so they might require that a data consumer contract not reveal the results of its computation until a payment is made. Since each party's terms are immutably and irrevocably encoded in a privacy-preserving smart contract, Sterling guarantees that all parties requirements are fulfilled.

INTEGRATING DATA VALUATION INTO STERLING For the specific use case of machine learning, we draw on techniques from *data valuation* introduced in the previous Chapters and adapt them for use on the blockchain. Programs written as smart contracts on blockchains are computationally expensive in runtime and gas costs (a miner's fee that you have to pay for the computation) to evaluate as each node within the network has to execute the program. Therefore data valuation needs to be highly scalable. While none of the baseline algorithms discussed in Chapter 3 are able to be implemented or deployed directly into a blockchain-based smart contract, the KNN data valuation presented in Chapter 4 and its extensions are the only family of valuation algorithms that can be supported directly within a decentralized execution model. Implemented as smart contract, we observe an average of 1300ms execution time for a single KNN data valuation iteration on the iris dataset and an average gas cost of 24 million gas.

5.2.4 *Privacy-Preserving Machine Learning*

To protect the contents of data and models and ensure their fair use, we must guarantee that the complete machine learning pipeline remains privacy-preserving–from data loading to evaluation of the trained model. To this end, we use the unique combination of trusted execution environments and differential privacy.

5.2.4.1 *Trusted Execution Environments*

In Sterling, trusted execution environments (TEEs) (D. Lee et al., 2020) can serve as the foundation for secure computation. The ML pipeline begins with the TEE remotely attesting to the veracity of the consumer smart contract. Once verified, the TEE runs the smart contract while keeping the program state safe from external observation or manipulation. The consumer contract is then able to obtain encrypted data via the contributor smart contract, decrypt it, and use it to directly update the model parameters, inside the TEE. Even with the overhead of memory encryption and privacy-preserving context switches, this approach is significantly more efficient than direct cryptographic methods like homomorphic encryption or secure multi-party computation. Indeed, machine learning in TEEs has performance comparable to non-private CPU-based training (Hynes, Cheng, and D. Song, 2018).

The TEE threat model does not include side-channel attacks, however. We address this using *data-oblivious* implementations of common training algorithms (Ohrimenko, Schuster, et al., 2016) which do not depend on the values of input data. Moreover, the threat model does not aim to protect the *host* from the computation. For example, since an TEE can directly access host RAM, a malicious smart contract could probe host memory for sensitive information like private keys. To counter such an attack, we sandbox the smart contracts by running them within a WebAssembly interpreter which provides complete memory isolation and limits the resources available to the computation. As a side benefit, compiling to Wasm enables Sterling to operate non-privately on other Wasm-enabled blockchains like the Kovan testnet[1].

Having established a secure way to operate on ML models, we now turn to ensuring that the model does not learn the exact values of the training data.

[1] https://kovan-testnet.github.io/website/

5.2.4.2 *Differential Privacy*

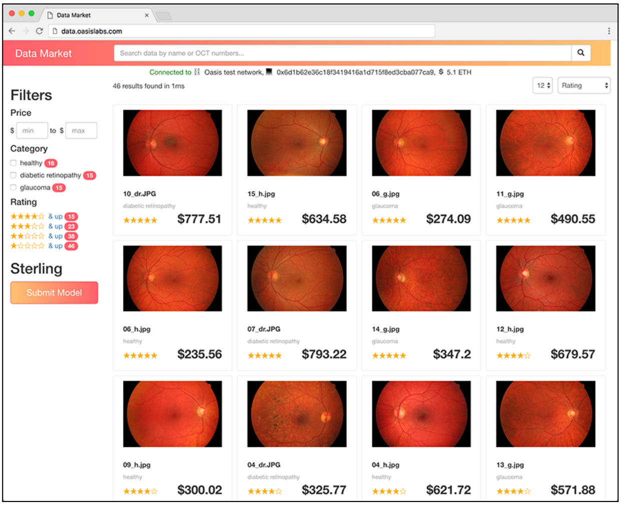
Even if data and model parameters are secured within a TEE, naive implementations of machine learning algorithms can memorize and later reveal training data (Carlini, C. Liu, et al., 2018).

Differential privacy (DP), in essence, provides strong theoretical guarantee that the risk to a data contributor's privacy is not significantly increased by the use of the data. In other words, applying a DP *mechanism* ensures that the results of analyzing the data are relatively insensitive to the exact values of any particular contributor's data. A simple and intuitive DP mechanism is the addition of noise to the model's gradients during training. The trade-off between privacy and precision is controlled by the *privacy budget*. An important element of DP is that making queries of the data (e.g., through model training or inference) "spends" the privacy budget.

The Sterling framework allows the data contributor contract to specify the differential privacy parameters as terms of use. We make the novel contribution of an automatic tracker for privacy budget expenditure which does not require trust assumptions (c.f. (McSherry, 2009)): the privacy requirements of every consumer request is automatically determined by analyzing its computation graph (Near et al., 2018). When the consumer contract uses the data, the contributor smart contract's privacy budget is correspondingly reduced; when the budget reaches zero, the contract ceases to yield data and consumer contracts admit no further queries. Sterling permits an economy to develop around privacy budget by allowing contributors to require payment in proportion to privacy usage, perhaps using principles from the literature (Hsu et al., 2014; Fleischer and Y. Lyu, 2012).

5.2.4.3 *Efficient private machine learning*

Given the growing scale of both data and ML models, the ability to quickly train complex models is a desirable property. For our benchmarks we train a multi-layer perceptron on MNIST using hidden layers of size 500 and 300 and ReLU activations; these are followed by a softmax and cross-entropy



Connected to Oasis test network, 0x6d1b62e36c18f3419416a1d715f8ed3cba077ca9, $ 5.1 ETH

46 results found in 1ms

10_dr.JPG diabetic retinopathy ★★★★★ $777.51
15_h.jpg healthy ★★★★★ $634.58
06_g.jpg glaucoma ★★★★★ $274.09
11_g.jpg glaucoma ★★★★★ $490.55
06_h.jpg healthy ★★★★★ $235.56
07_dr.JPG diabetic retinopathy ★★★★★ $793.22
14_g.jpg glaucoma ★★★★★ $347.2
12_h.jpg healthy ★★★★☆ $679.57
09_h.jpg healthy ★★★★☆ $300.02
04_dr.JPG diabetic retinopathy ★★★★☆ $325.77
04_h.jpg healthy ★★★★☆ $621.72
13_g.jpg glaucoma ★★★★☆ $571.88

FIGURE 5.3: Data contributors upload encrypted data and list it on Sterling. Data consumers can browse and purchase those data which satisfy their requirements.

loss. We train using SGD with a batch size of 32 and learning rate of 0.1. The classification accuracy on the test set reaches 98.7% after ten epochs of training.

5.2.5 *Use Case: Medical Data*

To demonstrate the utility of the Sterling data marketplace, we implement the disease modeling scenario described in the beginning of Section 5.2.1. The concrete application is diagnosis of diabetic retinopathy from fundus (back of eye) images. Examples of which are shown in Figure 5.3).

We simulate multiple data contributors by splitting a public dataset of fundus images among several data contributors. Each contributor contract will offer a randomly sized partition of the data and have its own privacy and payment requirements. Demo participants then assume the role of medical researchers and design consumer smart contracts, through our web interface, which train and evaluate privacy-preserving models on contributors' data. As a basis for customization, we provide several examples of models including logistic regression and deep neural networks.

In this setting of medical diagnosis, we provide a walkthrough which highlights the key features of Sterling. Namely:

1. the ability of a data contributor to specify a rich set of constraints, like payment and privacy, on privately shared data,
2. the ability of a data consumers to, via a web interface (shown in Figure 5.3), browse the marketplace, assemble a custom dataset, and create contracts which satisfy the constraints of all selected contributors,
3. efficient, secure training of differentially private ML models, and automatic appraisal of training data and the resulting model.

To yield insight into the otherwise opaque blockchain operations, we develop a blockchain explorer that displays events like pending transactions and model training progress (shown in the Appendix Figure C.2).

5.2.6 *Discussion & Summary*

The goal of demonstrating Sterling is to propose an end-to-end design of a data marketplace for private data that help us study the big picture, while providing us on the insights and dependencies on the roles of data valuation in respect to other components. Sterling is only possible through a set of opinionated assumptions including: privacy-preserving smart contracts and efficient compute requirements. Lastly, the demonstration highlights that, in addition to data valuation and data economics, a significant amount of work is necessary to make data marketplaces secure and viable for private data. This includes trusted computation, secure data transfer, enforcement of terms of service, and the design of appropriate incentives beyond valuation to ensure the participation of reliable actors.

In summary, Sterling is a data marketplace demonstration based on privacy-preserving smart contracts, which allows participants to exchange and use private data without revealing the data or the analytics performed thereon to untrusted parties. These interactions are mediated through novel data contributor and consumer smart contracts; each automatically enforces the terms-of-use set by its creator. Upon this generic platform, we build a market for privacy-preserving machine learning data and models. Models are kept from leaking the training data by automatic verification of differential privacy. In this way Sterling enables applications including credit scoring, smart home automation, and medical diagnosis.

5.3 ReforesTree: A Curated Dataset for ML-based Carbon Estimation

In this section, we will focus on a specific practical scenario for data payments: Machine learning for Monitoring, Reporting and Verification (MRV) of forests. In this setting, data payments can offer mutual advantages for both data contributors and consumers, with particular emphasis on the role of data payments to empower historically underserved communities.

ReforesTree offers an initial dataset derived from this approach. The dataset comprises six tropical agroforestry reforestation project sites, featuring individual tree crown bounding boxes for over 4,600 trees, along with their corresponding diameter at breast height (DBH), species, species group, aboveground biomass (AGB), and carbon stock. This ground truth field data is combined with cost-effective, high-resolution RGB drone imagery, which can be utilized to train new models for carbon offsetting protocols and to assess existing models. Data payments incentivized the acquisition of both high-resolution RGB imagery and field measurements. Additionally, we present supplementary work that quantifies the extent of carbon stock overestimation, demonstrating the value of ReforesTree in enhancing existing models.

5.3.1 *The Setting*

The deterioration of the natural world is unparalleled in human history and a key driver of the current climate crisis and global extinction (IPCC, 2021; Ceballos and Ehrlich, 2018; Exposito-Alonso et al., 2022). In the past twenty years, we have lost forest area equivalent to the size of Europe, accounting for more than 7% of global anthropogenic emissions (Hansen et al., 2013; IPCC, 2019). Reducing deforestation, restoring ecosystems, and natural sequestrating of carbon are therefore of uttermost importance and urgency. A current approach to finance the needed restoration of forest ecosystems are carbon offsets. The carbon offsetting market is expected to grow 100-fold until 2050 due to high demand and available capital (Blaufelder et al., 2021; Ecosystem Marketplace, 2021). However, an obstacle is the limited supply of offsetting projects, as forest owners lack upfront capital and market access (Kreibich and Hermwille, 2021).

manual forest inventory The standardized forest carbon stock inventory consists of manually measuring and registering sample trees of a project site. Tree metrics such as diameter at breast height (DBH), height, and species are then put through scientifically developed regression models called allometric equations to calculate the aboveground biomass (AGB) as seen in Figure 5.4. The total biomass of a forest is the total AGB added with the below-ground biomass (BGB), calculated using a root-to-shoot ratio specific to the forest type and region (H. Ma et al., 2021). The procedure how to calculate the correct amount of carbon offsets (CO_2e) to be certified for a project is standardized through (Pearson, Walker, and S. Brown, 2005) as shown in Figure 5.4. The (CO_2e), also known as the baseline forest carbon stock, is equivalent of the total biomass divided by two. Despite being prone to error propagation (Petrokofsky et al., 2012; Malhi et al., 2004) and shown to systematically overestimate carbon stock (Badgley et al., 2021), this is currently the standardized forest inventory method for certification of forestry projects.

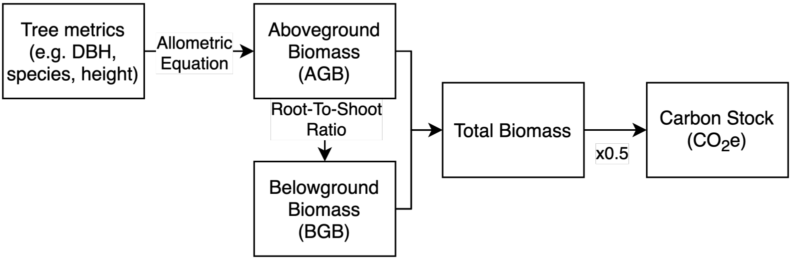

Tree metrics (e.g. DBH, species, height)
Allometric Equation
Aboveground Biomass (AGB)
Root-To-Shoot Ratio
Belowground Biomass (BGB)
Total Biomass
x0.5
Carbon Stock (CO_2e)

FIGURE 5.4: The standard procedure for calculating the correct amount of carbon offsets to be certified for a reforestation project. The tree metrics are collected from manual forest inventory.

ML FOR FOREST CARBON ESTIMATION Accurately estimating forest carbon stock, especially for small-scale carbon offset projects below 10,000 ha, presents several challenges, such as high variance of species and occlusion of individual tree crowns (White et al., 2018; Global Forest Watch, 2019). There are many promising approaches, such as hyper-spectral species classification (Schiefer et al., 2020), lidar-based height measurements (Ganz, Käber, and Adler, 2019) and individual tree crown segmentation across sites (Ben G. Weinstein et al., 2020). In recent years, remote sensing and ML have been used to estimate biomass (Narine, Popescu, and Malambo, 2020; Dubayah et al., 2022) based on drone and satellite data, to automate parts of the certification process of forestry carbon offsetting projects (Kellner et al., 2019). We may soon have mapped every tree on earth (Hanan and Anchang, 2020), enabling forest above-ground biomass and carbon to be estimated at scale (Dubayah et al., 2022; Saatchi et al., 2011; Santoro et al., 2021).

Recent research has however shown that the current manual forest carbon stock practices systematically overestimate forestry carbon offsetting projects (Badgley et al., 2021; West et al., 2020), unless they are properly calibrated and transparently validated. This even applies to the latest generation satellite programs such as GEDI (Dubayah et al., 2022; Silva et al., 2021). One reason is that these applications have been developed mainly on datasets from boreal and temperate forests, which are not suitable for other

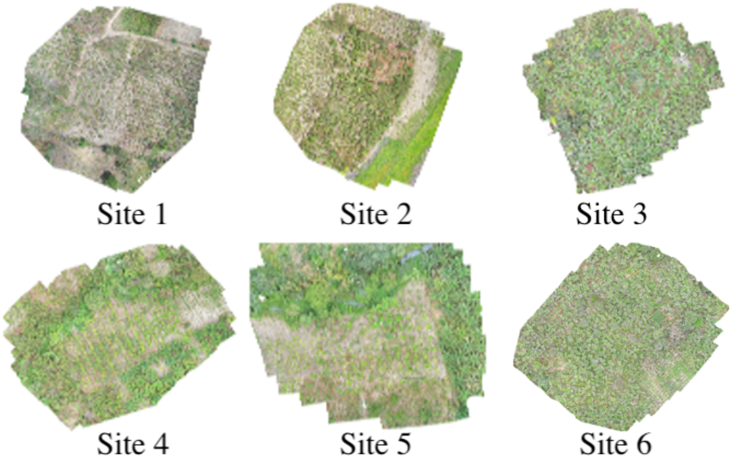


Figure 5.5: Drone imagery of each site of the ReforesTree dataset with a resolution of 2cm/px.

types of ecosystems. To the best of our knowledge, there is no publicly available dataset of tropical forests with both aerial imagery and ground truth field measurements, and very little available data of that kind from the Global South in general, putting these regions at a disadvantage when competing in the global carbon emission market. There is thus need for higher-quality carbon offsetting data to achieve more transparency and accountability in the MRV of the forest carbon stock (Haya et al., 2020).

role of data payments for fair data Thus, high-quality data on tropical forests are in high demand. But ground forest measurements are hard to sustain and the people who make them are extremely disadvantaged compared to those who use them (R. A. F. d. Lima et al., 2022). Data payments provide a new approach to forest data that focuses on the needs of local data contributors, and ensure that they are rewarded properly. Data consumers incentivize the collection and curation of geographically-balanced gold-standard open datasets of small-scale forest plots of currently underrepresented forest ecosystems in the Global South. At the same time, data marketplaces can provide significant economic opportunity for local and Indigenous communities by rewarding them as data contributors.

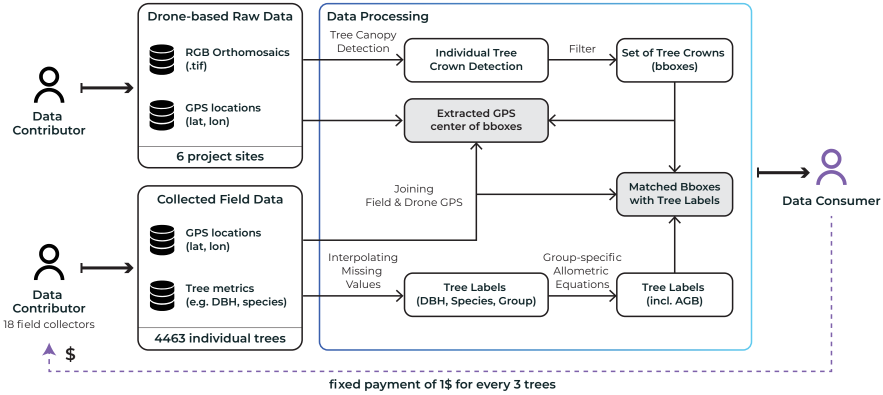

Data Contributor
Drone-based Raw Data
RGB Orthomosaics (.tif)
GPS locations (lat, lon)
6 project sites
Data Contributor
18 field collectors
$
Collected Field Data
GPS locations (lat, lon)
Tree metrics (e.g. DBH, species)
4463 individual trees
Data Processing
Tree Canopy Detection
Individual Tree Crown Detection
Filter
Set of Tree Crowns (bboxes)
Extracted GPS center of bboxes
Joining Field & Drone GPS
Matched Bboxes with Tree Labels
Interpolating Missing Values
Tree Labels (DBH, Species, Group)
Group-specific Allometric Equations
Tree Labels (incl. AGB)
Data Consumer
fixed payment of 1$ for every 3 trees

FIGURE 5.6: The raw data and its subsequent data processing pipeline for the ReforesTree dataset, resulting in labels matched to bounding boxes per tree. Data consumers reward data contributors upon receiving the processed data points.

5.3.2 *Community-centered data collection*

With our approach, we want to establish a compromise between cost and scope. Using small RGB drones and smart phones for the data collection allows our approach to be employed cheaply, hence offering potential to be adapted for small-scale projects worldwide. However, this comes with certain limitations, such as more difficulty in measuring aspects such as tree height from the drone data, compared to, e. g., LIDAR-based technology. For the ground-level field data, we developed a mobile tree mapper app [2] that can estimate biomass based on species and diameter of each individual tree. Both the species and the diameter are estimated from a single photo of the tree trunk, making it easy and fast to collect data on many individual trees. As these estimates are based on ground-level data, they can be used as ground-truth and accurately matched with satellite image based data. Where logistically feasible, we further collect aerial images using RGB drones, operated by experts in the local projects we are collaborating with.

[2] The app is openly available at: https://play.google.com/store/apps/details?id=com.sprinteins.treeapp

The data collection is implemented in collaboration with local and Indigenous communities, who conduct the field work. Their expert knowledge on the local flora can also be leveraged to manually corrected the tree species estimated by TreeMapper, if needed. To streamline our scenario, we will exclusively concentrate on fixed payouts in this context, as opposed to dynamic pricing facilitated by data valuation in Chapter 3. Through conversations with collaborators and local community members during our work with ReforesTree, we learned that Shapley-based data valuation, despite its desirable properties, can be challenging to interpret or explain to data contributors in practice. We consider tackling this issue to be a crucial area for future research. Thus, we determine a fixed payment rate of 1$ for every 3 trees collected, based on local hourly wages in each region for field-based data. Drone data has been collected through a local operator at a fixed rate of 5̃00$ per site.

SOCIAL IMPACT Recognizing the true costs of forest data origination is critical to empower an equitable benchmark (R. A. d. Lima et al., 2022). Rewarding for data collection has the potential to provide an important additional funding source to frontline communities. For instance, the average monthly salary of forest rangers helping us to collect data in Ecuador is $400 per month. By contributing to ReforesTree, rangers have experienced an immediate financial improvement. Additionally, the benchmark incentivizes local upskilling through the frequent use of drone monitoring, a skill that empowers communities to monitor and protect larger forest areas.

The resulting **ReforesTree** dataset consists of six agro-forestry sites in the central coastal region of Ecuador. The sites are of dry tropical forest type and eligible for carbon offsetting certification with forest inventory done and drone imagery captured in 2020. See Table 5.1 for information on each site.

Site no.	No. of Trees	No. of Species	Site Area	total AGB	total CO2e
1	743	18	0.53	8	5
2	929	22	0.47	15	9
3	789	20	0.51	10	6
4	484	12	0.56	5	3
5	872	14	0.62	15	9
6	846	16	0.48	12	7
total	4463	28	3.17	66	40

Table 5.1: Overview of the six project sites in Ecuador, as gathered in field measurements. Each tree includes a diameter and species measurement. The resulting significant aboveground biomass (AGB) is measured in metric tons and area in hectares.

5.3.3 *Forest Inventory Data and Drone Imagery*

Field measurements were done by hand for all live trees and bushes within the site boundaries and include GPS location, species, and diameter at breast height (DBH) per tree. Drone imagery was captured by an RGB camera from a Mavic 2 Pro drone with a resolution of 2cm per pixel. Each site is around 0.5 ha, mainly containing banana trees (Musaceae) and cacao plants (Cacao), planted in 2016-2019.

$$AGB_{fruit} = 0.1466 * DBH^{2.223} \tag{5.1}$$

$$AGB_{musacea} = 0.030 * DBH^{2.13} \tag{5.2}$$

$$AGB_{cacao} = 0.1208 * DBH^{1.98} \tag{5.3}$$

$$AGB_{timber} = 21.3 - 6.95 * DBH + 0.74 * DBH^{2} \tag{5.4}$$

The aboveground biomass (AGB) is calculated using published allometric equations for tropical agro-forestry, namely Equation (5.1) for fruit trees, including citrus fruits (Segura, Kanninen, and Suárez, 2006), Equation (5.2) for banana trees (Van Noordwijk et al., 2002), Equation (5.3) for cacao

(Yuliasmara, Wibawa, and Prawoto, 2009), and Equation (5.4) for shade trees (timber) (S. Brown and Iverson, 1992). These are commonly used in global certification standards. The carbon stock is calculated through the standard forest inventory methodology using a root-to-shoot ratio of 22%, which is standard for dry tropical reforestation sites (L. Ma et al., 2019).

5.3.4 *Limitations*

gps noise When collecting the field data the collectors recorded the gps coordinates, latitude and longitude, of each tree on their phone. It turns out that the coordinates are very noisy as seen in Figure 5.7 and therefore makes the mapping of the drone imagery with individual trees challenging.

unbalanced diameter and species measurements The dataset is unbalanced with regards to species, of which 43% is cacao and 32% is banana. Additionally, due to all of the trees being planted between 2016-2019, many of the trees have a similar size (e.g. diameter) and half of the trees have diameters between 7-10cm.

5.3.5 *Data Processing and Method*

The collected raw data undergoes multiple processing steps, shown in Figure 5.6, due to the challenges of data curation. The goal of this data processing is to have a ML ready dataset that consists of matched drone images of individual trees with the trees labels, such as the carbon stock.

Tree Crown Detection. Initially the RGB orthomosaics are cut into 4000×4000 tiles and sent through DeepForest, a python package for predicting individual tree crowns from RGB imagery (Ben G Weinstein et al., 2019), fine-tuned on some manually labelled bounding boxes from the sites.

White Filter and Manual Labeling. By visual inspection, several of the bounding boxes were of poor quality such as not being of a tree, too small

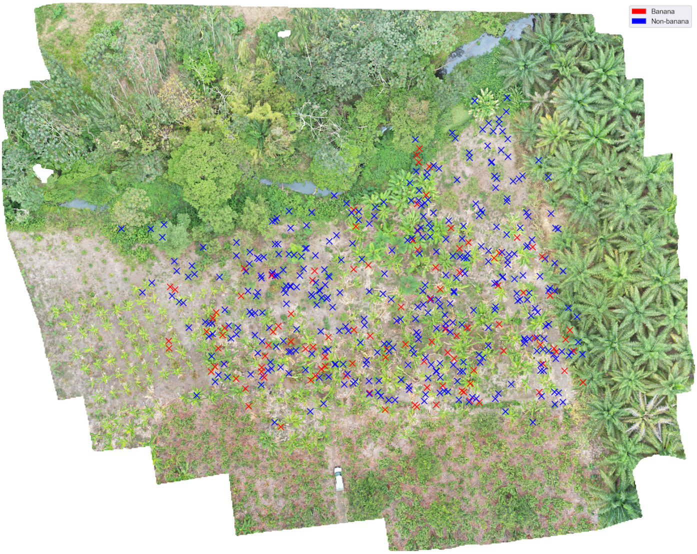


FIGURE 5.7: A plot of site no. 5 and the raw GPS coordinates (latitudes and longitudes) of the field data collected. Blue crosses represent *non-banana* and red crosses represents *banana*. As we can see the noise of the GPS data is significant, and it is difficult to recognize any pattern of trees which they belong to.

(zoomed in on a leaf) or large (several trees), or being on the edge of the drone imagery and therefore largely consist of white pixels. Thus, the bounding boxes containing more than 80% white were filtered out, e.g. bounding boxes lying on the border of the drone imagery, and manually labeled to banana and non-banana, due to the easily recognizable characteristics of banana trees, resulting in clear bounding boxes of all trees as shown in Figure 5.8.

GPS Matching. To fuse the tree information extracted from the ground measurements with the bounding boxes of the trees detected, we map between the two closest GPS positions (center of bounding box from drone imagery and GPS location of tree from field data) that has the similar labels.

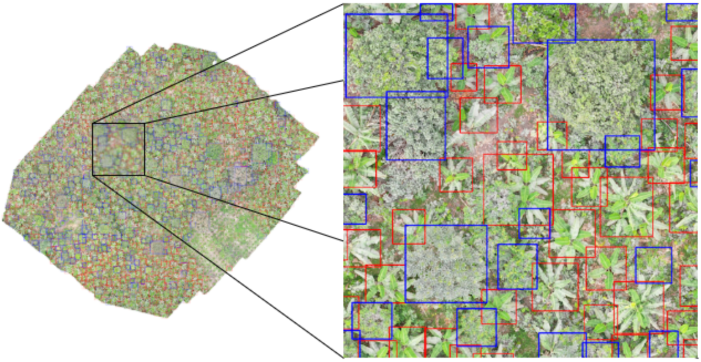


FIGURE 5.8: Bounding box annotations per tree, as a result of fine-tuned DeepForest tree crown detection and manual cleaning. Red boxes represent banana trees and blue boxes represent other species.

5.3.6 *Experiments*

We demonstrate the usefulness of the ReforesTree benchmark for data consumers in two sets of experiments. First, we train an end-to-end ML model (Baseline CNN Model) that can accurately predict AGB from aerial imagery. Second, we benchmark existing and widely used satellite-based predictions.

5.3.6.1 *Baseline CNN Model*

With a dataset of matched bounding boxes and tree labels, we fine-tuned a basic pre-trained CNN, ResNet18 (K. He et al., 2015) with a mean-square-error loss to estimate individual tree AGB from aerial imagery. The results (shown in Table 5.2 with cross validation on 6 sites) were satisfying despite the simple baseline model, and proves that the individual tree estimation from drone imagery has potential. Fourteen images were identified as being larger than the expected crown size of a tree, and they were center cropped at 800×800. To preserve the crown size information, the smaller images were zero-padded up to 800×800, before all images were resized to fit the network architecture. The dataset is unbalanced with regards to species, of which 43% is cacao and 32% is banana. Additionally, due to the trees being planted between 2016-2019, many of the trees have similar size (e.g. DBH) and half of the trees have DBH between 7-10cm. The training dataset consisted of equal number of samples of species and DBH, and from the different project sites.

5.3.6.2 *Benchmarking Satellite-Based Estimation*

With the emerging new biomass maps and forest stock estimation models, we used the **ReforesTree** dataset to benchmark these maps and compare with our baseline CNN model for AGB estimation. We compared the maps taken from Global Forest Watch (Global Forest Watch, 2019), Spawn (Spawn, Sullivan, and Lark, 2020), and Santoro (Santoro et al., 2021). The Global Forest Watch's Above-Ground Woody Biomass dataset is a global map of AGB and carbon density at 30m×30m resolution for the year 2000. It is based on more than 700,000 quality-filtered Geoscience Laser Altimeter System (GLAS) LIDAR observations using machine learning models based on allometric equations for the different regions and vegetation types. The second dataset from Spawn (Spawn, Sullivan, and Lark, 2020) is a 300m×300m harmonized map based on overlayed input maps. The input maps were allocated in proportion to the relative spatial extent of each vegetation type using ancillary maps of tree cover and landcover, and a rule-based decision schema. The last, and most recent 100m×100m dataset from Santoro (Santoro et al., 2021) is obtained by spaceborne SAR (ALOS PALSAR, Envisat ASAR), optical (Landsat-7), LIDAR (ICESAT) and auxiliary datasets with multiple estimation procedures with a set of biomass expansion and conversion factors following approaches to extend ground estimates of wood density and stem-to-total biomass expansion factors. To benchmark the low-resolution satellite-based maps, we fitted it to the high-resolution drone imagery overlapping the GPS coordinates. The calculation of the total AGB was done in five steps, illustrated in Figure 5.9:

1. Cropping the low-resolution satellite map with a padding around the polygon of the site to reduce computation intensity (Satellite Raw)

SITE NO.	FIELD DATA	GFW 2019	SPAWN 2020	SANTORO 2021	BASELINE (OURS)
1	8	90	84	14	7
2	15	99	102	12	8
3	10	25	33	19	15
4	5	9	82	12	9
5	15	78	76	15	11
6	12	30	35	16	15
TOT.	66	331	413	89	65

TABLE 5.2: The benchmark results from comparing different models for estimating AGB with the forest inventory of the ReforesTree sites. All numbers are given as AGB in kg. GFW is (Global Forest Watch, 2019), Spawn is (Spawn, Sullivan, and Lark, 2020), Santoro is (Santoro et al., 2021). All of these three are satellite-based. Lastly, the baseline CNN is our drone-based model.

2. Linearly interpolating the values for this map and resize the map with the same high-resolution pixel resolution as the drone imagery (Satellite Interpolated)
3. Cropping the map further fitting with the GPS locations (max/min) of the drone imagery
4. Filtering out the site area by removing all pixels in the satellite-based map, that are outside of the drone imagery, coloured white (Satellite Filtered)
5. Lastly, multiplying the AGB mean density of the filtered map with the project site area to get the total AGB

As seen in Table 5.2, all of the available global AGB maps have a tendency to overestimate the ground truth measurements up to a factor of ten. These are not encouraging results showing that these maps are far from being accurate enough to be used in remote sensing of forest carbon stock at a small scale, as is the case for the **ReforesTree** dataset.

Our baseline model, on the other hand, has a slight tendency of underestimating the biomass. The model has an evident advantage, to be trained on

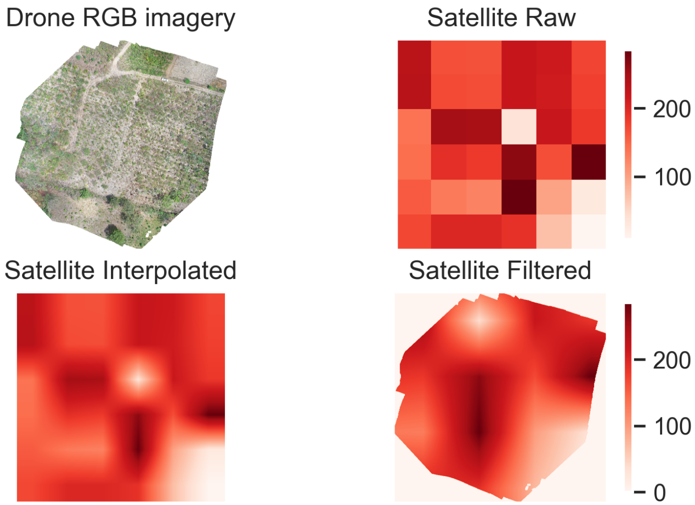


FIGURE 5.9: This figure represents the different steps in the benchmark analysis and how we calculated the total AGB amount from the satellite-based maps for the ReforesTree sites. This is taken from site no. 0. The values represented in the image is AGB density (tons/ha).

the dataset, but these initial results show promise for the individual tree estimation approach using drone imagery for forest carbon inventory.

5.3.7 *Discussion & Summary*

We introduce the **ReforesTree** benchmark dataset in hopes of encouraging the fellow machine learning community to take on the challenge of developing low-cost, scalable, trustworthy and accurate solutions for monitoring, verification and reporting of tropical reforestation inventory. We also present an outlined methodology for creating an annotated machine learning dataset from field data and drone imagery, and train a baseline CNN model for individual tree aboveground biomass estimation. This methodology includes a data processing pipeline leveraging a fine-tuned tree crown detection algorithm to fuse drone imagery and field-based data measurement.

The ReforesTree benchmark dataset, along with the baseline CNN model, can be found at `https://zenodo.org/record/6813783`. As tropical forest data is often gathered by individuals from lower-income communities, providing payouts to local data collectors has demonstrated potential for creating a positive social impact while also benefiting the remote sensing community. To further this effort, we are in the process of expanding our data collection to include more sites covering a broader geographical range of tropical forest areas. An enhanced benchmark dataset is currently under development, featuring 45,141 data points from an additional 20 sites and contributions from 77 local individuals.

5.4 Summary

In this chapter, our goal was to examine the real-world challenges that data marketplaces may encounter, with a specific focus on two primary issues: Training ML on private data and curating unique and demanding datasets. We presented two data marketplace applications, **Sterling** and **ReforesTree**, each addressing these challenges. Sterling proposes a design for an opinionated marketplace for private data, while ReforesTree encourages the curation of an ecological dataset in difficult conditions. Both applications emphasize the importance of looking beyond data valuation when studying data marketplaces.

To ensure the success of data marketplaces and data valuation, it is crucial to take a comprehensive view of the marketplace, addressing key questions such as the mutual benefits for data contributors and consumers, the privacy and security of data exchange, and the overall design of the marketplace. Although Sterling and ReforesTree have demonstrated potential and received initial support from the community (Sterling has led to follow-up projects within the Oasis Blockchain (D. Lee et al., 2020), and ReforesTree serves as a benchmark dataset within TorchGeo (Stewart et al., 2022) and was able to receive funding to curate a 10x larger dataset), we believe that there is still considerable future work needed to make such marketplaces effective and resilient for real-world applications.

6

CONCLUSION

I never think about the future - it comes soon enough.
— Albert Einstein

In this Chapter, we summarize our findings and its potential impact. We conclude by offering insights into prospective future research paths, emphasizing the opportunities and constraints associated with data valuation.

6.1 Impact

DATA MARKETS The research presented in this dissertation has inspired various follow-up studies that build on the idea of Shapley-based (or generally cooperative game theory-based) data valuation, as valuation is a fundamental component of markets and incentive-based machine learning. A short overview of data valuation methods is provided in Section 2.3, while a comprehensive list can be found in (Sim, X. Xu, and Low, 2022). The work presented in this dissertation has also impacted the growing political movements of data as labor (Arrieta-Ibarra et al., 2018; Posner and Weyl, 2019) and data dignity (Delacroix and Lawrence, 2019), as it offers a principled approach to address data values within a machine learning context.

DATA DEBUGGING A recent observation that has gained significant attention is the notion that the quality of a machine learning model is frequently a reflection of the quality of the underlying training data. Consequently, the most practical and efficient approach to enhancing machine learning model performance is often to improve the quality of the data. Specifically *data debugging*, the process of discovering and repairing data errors in order to improve the quality of data, has received recent interest in leveraging the

Shapley value. (Ghorbani, Kim, and Zou, 2020) demonstrated that Shapley values can be used to assess the influence and significance of a data point. (Jia, F. Wu, et al., 2021) conducted various experiments showing that KNN Shapley values can act as efficient learning-agnostic heuristics for data importance beyond KNN models. (Karlaš et al., 2022) introduced *DataScope* [1], a system that effectively computes Shapley values of training examples across an end-to-end machine learning pipeline, showcasing its applications in data debugging for ML training. We anticipate that our exact algorithms for KNN Shapley will persist as efficient and practical heuristics for data debugging purposes.

OPEN-SOURCE SOFTWARE The research presented in this dissertation has been adopted and reimplemented in various open-source software projects. For instance, our work presented in Chapter 3 and Chapter 4 has been incorporated into pyDVL [2], while the the ReforesTree benchmark from Chapter 5 has been integrated into the widely-used TorchGeo [3] package (Stewart et al., 2022).

6.2 Future Work

FROM THE SHAPLEY VALUE TO MONETARY REWARD In Chapter 3 we have focused on the problem of attributing an ML utility to each data and computation contributor. In practice, the buyer pays a certain amount of money depending on the model utility and it is required to determine the share of each contributor in terms of monetary rewards. Thus, a remaining question is how to map the Shapley value, a share of the total model utility, to a share of the total revenue acquired from the buyer. A simple method for such mapping is to assume that the revenue is an affine function of the model utility, i.e., $R(S) = aU(S) + b$ where a and b are some constants which can be determined via market research. Due to the linearity property, we have $s(R, i) = as(U, i) + b$. Thus, we can apply the same affine function

[1] https://github.com/easeml/datascope
[2] https://github.com/appliedAI-Initiative/pyDVL
[3] https://github.com/microsoft/torchgeo

to the Shapley value to obtain the the monetary reward for each contributor. Nonetheless, in Chapter 5 and via conversations with collaborators and local community members in our work with ReforesTree, we found that Shapley-based valuation, despite its desirable properties, can be difficult to interpret or explain to data contributors in practice. As a result, data valuation is often better treated as internal information, and it is usually simpler to rely on fixed payouts.

TRUTHFUL REPORTING A concern related to the task-dependence of the data valuation scheme is that the buyer can be trusted to truthfully report the total worth of the model, which will then be split between different contributors. Suppose that the data buyer coincides with the analyst of the data and his contribution is valued together with data contributors using the composite game framework. Then, as shown in 4.16 (b), the buyer will get a larger share of the total worth if the model produced by the buyer is more performant. In other words, the buyer is incentivized to train a good model and truthfully report the total worth when his contribution is valued in tandem with the data contributors. Now, we turn to a different scenario where the data buyer does not participate in the model training process or his contribution is not valued in the composite game framework. In that case, the buyer is no longer incentivized to report truthfully and future work is necessary to build systems to ensure transparency of the training process.

LACK OF DATA DEMAND In Chapter 5, we discovered that data demand is not a certainty. Continuing the analogy of data as oil, unprocessed data can be likened to crude oil, which is not suitable for direct use in production. Just as we need an oil refinery to process crude oil, we must comprehend and process raw data to make it ready for analysis. However, even then, justifying demand can be challenging. In today's society, we have become accustomed to free data due to the targeted advertising business model. Consequently, users are not familiar with the concept of being paid or having to pay for data and services, and often data purchases happen passively through the means of another business model (e.g. data labeling). Future work is needed to further understand the data demand side.

TOWARDS A PRODUCTION-READY SYSTEM Real-world data markets hold significant potential to enhance data curation and decentralized machine learning research. However, future work is needed to understand how to transform data valuation into actionable incentives for users - and eventually develop a production-ready system. We believe will require interdisciplinary collaboration involving expertise from various fields, including economics, law, social sciences, and computer science. Designing robust, secure, and fair data markets can not only benefit researchers and businesses but also empower individuals to maintain control over their data and receive the rewards of their contributions.

A

APPENDIX: CHAPTER 3

A.1 Theoretical Results on the Baseline Permutation Sampling

Let π_t be a random permutation of $D = \{z_i\}_{i=1}^N$ and each permutation has a probability of $\frac{1}{N!}$. Let $\phi_i^t = U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t})$, we consider the following estimator of s_i:

$$\hat{s}_i = \frac{1}{T}\sum_{t=1}^{T} \phi_i^t$$

Given the range of the utility function r, an error bound ϵ, and a confidence $1-\delta$, the sample size required such that

$$P[\|\hat{s} - s\|_2 \geq \epsilon] \leq \delta$$

is

$$T \geq \frac{2r^2 N}{\epsilon^2} \log \frac{2N}{\delta}$$

Proof.

$$P[\max_{i=1,\cdots,N} |\hat{s}_i - s_i| \geq \epsilon] = P[\cup_{i=1,\cdots,N}\{|\hat{s}_i - s_i| \geq \epsilon\}] \leq \sum_{i=1}^{N} P[|\hat{s}_i - s_i| \geq \epsilon]$$

$$\leq 2N \exp\left(-\frac{2T\epsilon^2}{4r^2}\right)$$

The first inequality follows from the union bound and the second one is due to Hoeffding's inequality. Since $\|\hat{s}-s\|_2 \leq \sqrt{N}\|\hat{s}-s\|_\infty$, we have

$$P[\|\hat{s}-s\|_2 \geq \epsilon \leq P[\|\hat{s}-s\|_\infty \geq \epsilon/\sqrt{N}] \leq 2N\exp\Big(-\frac{2T\epsilon^2}{4Nr^2}\Big)$$

Setting $2N\exp(-\frac{T\epsilon^2}{2Nr^2}) \leq \delta$ yields

$$T \geq \frac{2r^2N}{\epsilon^2}\log\frac{2N}{\delta}$$

□

A.2 Proof of Lemma 1

Proof.

$$\begin{aligned}
s_i - s_j = &\sum_{S\subseteq I\setminus\{i\}} \frac{|S|!(N-|S|-1)!}{N!}[U(S\cup\{i\})-U(S)] \\
&- \sum_{S\subseteq I\setminus\{j\}} \frac{|S|!(N-|S|-1)!}{N!}[U(S\cup\{j\})-U(S)] \\
&= \sum_{S\subseteq I\setminus\{i,j\}} \frac{|S|!(N-|S|-1)!}{N!}[U(S\cup\{i\})-U(S\cup\{j\})] \\
&+ \sum_{S\in\{T|T\subseteq I, i\notin T, j\in T\}} \frac{|S|!(N-|S|-1)!}{N!}[U(S\cup\{i\})-U(S)] \\
&- \sum_{S\in\{T|T\subseteq I, i\in T, j\notin T\}} \frac{|S|!(N-|S|-1)!}{N!}\cdot[U(S\cup\{j\})-U(S)] \\
&= \sum_{S\subseteq I\setminus\{i,j\}} \frac{|S|!(N-|S|-1)!}{N!}[U(S\cup\{i\})-U(S\cup\{j\})] \\
&+ \sum_{S'\subseteq I\setminus\{i,j\}} \frac{(|S'|+1)!(N-|S'|-2)!}{N!}[U(S'\cup\{i\})-U(S'\cup\{j\})] \\
&= \sum_{S\subseteq I\setminus\{i,j\}} \Big(\frac{|S|!(N-|S|-1)!}{N!} + \frac{(|S|+1)!(N-|S|-2)!}{N!}\Big)
\end{aligned}$$

$$\cdot \left[U(S \cup \{i\}) - U(S \cup \{j\})\right]$$
$$= \frac{1}{N-1} \sum_{S \subseteq I \setminus \{i,j\}} \frac{1}{C_{N-2}^{|S|}} \left[U(S \cup \{i\}) - U(S \cup \{j\})\right].$$

□

Loosely speaking, the proof distinguishes subsets S which include neither i nor j (such that the subset utility $U(S)$ of the marginal contribution directly cancels) and subsets including either i or j. In the latter case, S can be partitioned to a mock subset S' by excluding the respective point from S such that a common sum over S' again eliminates all terms other than $U(S' \cup \{i\}) - U(S' \cup \{j\})$.

A.3 Proof of Lemma 2

Proof. Let $\epsilon' = \epsilon/(2\sqrt{N})$. Assume that $\hat{s}_i - s_i > \epsilon/\sqrt{N}$. Let $\hat{s}_i - s_i = c\epsilon'$ where $c > 2$.

Since $C_{i,j}$ is an $(\epsilon', \delta/(N(N-1)))$-approximation to $s_i - s_j$, we have that with probability at least $1 - \delta/(N(N-1))$,

$$|(s_i - s_j) - C_{i,j}| \leq \epsilon' \tag{A.1}$$

Moreover, the inequality (3.6) implies that

$$|(\hat{s}_i - \hat{s}_j) - C_{i,j}| \leq \epsilon'$$

Therefore,

$$|\hat{s}_i - s_i + s_j - \hat{s}_j| = |\hat{s}_i - \hat{s}_j - C_{i,j} - (s_i - s_j - C_{i,j})| \tag{A.2}$$
$$\leq |\hat{s}_i - \hat{s}_j - C_{i,j}| + |s_i - s_j - C_{i,j}| \tag{A.3}$$
$$\leq 2\epsilon' \tag{A.4}$$

with probability at least $1-\delta/(N(N-1))$. By the assumption that $\hat{s}_i - s_i = c\epsilon'$ and $c > 2$, we have

$$(c-2)\epsilon' \le \hat{s}_j - s_j \le (c+2)\epsilon' \tag{A.5}$$

which further implies that $\hat{s}_j - s_j > 0$ for some $j \neq i$. Thus, with probability $1-\delta/N$, we have $\hat{s}_j - s_j > 0$ for all $j \neq i$.

Then,

$$\sum_{j=1}^{N}(\hat{s}_j - s_j) = \sum_{j\neq i}(\hat{s}_j - s_j) + (\hat{s}_i - s_i) > 0 \tag{A.6}$$

Since $\sum_{j=1}^{N} s_j = U_{\text{tot}}$, it follows that $\sum_{j=1}^{N} \hat{s}_j > U_{\text{tot}}$, which contradicts with the fact that $\hat{s}_j$ $(j = 1, \ldots, N)$ is a solution to the feasibility problem (3.5) and (3.6).

The contradiction can be similarly established for $s_i - \hat{s}_i = c\epsilon'$. Therefore, we have that with probability at least $1-\delta/N$, $|s_i - \hat{s}_i| \le 2\epsilon'$ for some i. This in turn implies that with probability at least $1-\delta$, $\|\hat{s} - s\|_\infty \le 2\epsilon' = \epsilon/\sqrt{N}$. Moreover, since $\|\hat{s} - s\|_2 \le \sqrt{N}\|\hat{s} - s\|_\infty = \epsilon$, we have that $\|\hat{s} - s\|_2 \le \epsilon$ with probability at least $1-\delta$.

□

A.4 Proof of Theorem 1

We prove Theorem 1, which specifies a lower bound on the number of tests needed for achieving a certain approximation error. Before delving into the proof, we first present a lemma that is useful for establishing the bound in Theorem 1.

Lemma 4 (Bennett's inequality (Bennett, 1962)). *Given independent zero-mean random variables $X_1, \cdots, X_n$ satisfying the condition $|X_i| \le a$, let $\sigma^2 = \sum_{i=1}^{n} \sigma_i^2$ be the total variance. Then for any $t \ge 0$,*

$$P[S_n > t] \le \exp\left(-\frac{\sigma^2}{a^2} h\left(\frac{at}{\sigma^2}\right)\right)$$

where $h(u) = (1+u)\log(1+u) - u$.

We now restate Theorem 1 and proceed to the main proof.

Theorem. *Algorithm 1 returns an* (ϵ, δ)*-approximation to the Shapley value with respect to* l_2*-norm if the number of tests* T *satisfies* $T \geq 8\log\frac{N(N-1)}{2\delta}/\big((1-q_{tot}^2)h(\frac{\epsilon}{Zr\sqrt{N}(1-q_{tot}^2)})\big)$, *where* $q_{tot} = \frac{N-2}{N}q(1) + \sum_{k=2}^{N-1} q(k)[1 + \frac{2k(k-N)}{N(N-1)}]$, $h(u) = (1+u)\log(1+u) - u$, $Z = 2\sum_{k=1}^{N-1}\frac{1}{k}$, *and* r *is the range of the utility function.*

Proof. By Lemma 1, the difference in Shapley values between points i and j is given as

$$s_i - s_j = \frac{1}{N-1}\sum_{S\subseteq I\backslash\{i,j\}}\frac{1}{C_{N-2}^{|S|}}\Big[U(S\cup\{i\}) - U(S\cup\{j\})\Big]$$
$$= \frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{C_{N-2}^{k}}\sum_{S\subseteq I\backslash\{i,j\},|S|=k}\Big[U(S\cup\{i\}) - U(S\cup\{j\})\Big].$$

Let $\beta_1, \cdots, \beta_N$ denote N Boolean random variables drawn with the following sampler:

1. Sample the "length of the sequence" $\sum_{i=1}^{N}\beta_i = k \in \{1, 2, \cdots, N-1\}$, with probability $q(k)$.
2. Uniformly sample a length-k sequence from $\binom{N}{k}$ all possible length-k sequences

Then the probability of any given sequence $\beta_1, \cdots, \beta_N$ is

$$P[\beta_1, \cdots, \beta_N] = \frac{q(\sum_{i=1}^{N}\beta_i)}{C_N^{\sum_{i=1}^{N}\beta_i}}.$$

Now, we consider any two data points x_i and x_j where $i, j \in I = \{1, \cdots, N\}$ and their associated Boolean variables β_i and β_j, and analyze

$$\Delta = \beta_i U(\beta_1, \cdots, \beta_N) - \beta_j U(\beta_1, \cdots, \beta_N)$$

Consider the expectation of Δ where $\gamma_k = \frac{q(k+1)}{C_N^{k+1}}$. Obviously, only $\beta_i \neq \beta_j$ has non-zero contributions:

$$\begin{aligned}\mathbb{E}[\Delta] = \sum_{k=0}^{N-2} \gamma_k \sum_{S \subseteq I\setminus\{i,j\}, |S|=k} \big[&U(\beta_1, \cdots, \beta_{i-1}, 1, \beta_{i+1}, \cdots, \beta_{j-1}, 0, \beta_{j+1}, \cdots, \beta_N) \\ &- U(\beta_1, \cdots, \beta_{i-1}, 0, \beta_{i+1}, \cdots, \beta_{j-1}, 1, \beta_{j+1}, \cdots, \beta_N)\big] \\ = \sum_{k=0}^{N-2} \gamma_k \sum_{S \subseteq I\setminus\{i,j\}, |S|=k} &\big[U(S \cup \{i\}) - U(S \cup \{j\})\big]\end{aligned}$$

We would like to have $Z\mathbb{E}[\Delta] = s_i - s_j$

$$Z\frac{q(k+1)}{C_N^{k+1}} = \frac{1}{(N-1)C_{N-2}^k}$$

which yields

$$q(k+1) = \frac{N}{Z(k+1)(N-k-1)} = \frac{1}{Z}(\frac{1}{k+1} + \frac{1}{N-k-1})$$

for $k = 0, \cdots, N-2$. Equivalently,

$$q(k) = \frac{1}{Z}(\frac{1}{k} + \frac{1}{N-k})$$

for $k = 1, \cdots, N-1$. The value of Z is given by

$$Z = \sum_{k=1}^{N-1}(\frac{1}{k} + \frac{1}{N-k}) = 2\sum_{k=1}^{N-1}\frac{1}{k} \leq 2(\log(N-1)+1)$$

Now, $\mathbb{E}[Z\Delta] = s_i - s_j$. Assume that the utility function ranges from $[0, r]$; then, we know from (4.5) that $Z\Delta$ is random variable ranges in $[-Zr, Zr]$. Consider

$$\Delta := \beta_i U(\beta_1, \cdots, \beta_N) - \beta_j U(\beta_1, \cdots, \beta_N)$$

Note that $\Delta = 0$ when $\beta_i = \beta_j$. If $P[\beta_i = \beta_j]$ is large, then the variance of Δ will be much smaller than its range.

$$
\begin{aligned}
P[\beta_i = \beta_j] &= P[\beta_i = 1, \beta_j = 1] + P[\beta_i = 0, \beta_j = 0] \\
&= \left[\sum_{k=2}^{N-1} \frac{q(k)}{C_N^k} C_{N-2}^{k-2} \right] + \left[q(1) + \sum_{k=2}^{N-1} \frac{q(k)}{C_N^k} C_{N-2}^{k} \right] \\
&= \frac{N-2}{N} q(1) + \sum_{k=2}^{N-1} q(k) \left[1 + \frac{2k(k-N)}{N(N-1)} \right] \equiv q_{tot}
\end{aligned}
$$

Let $W = \mathbb{1}[\Delta \neq 0]$ be an indicator of whether or not $\Delta = 0$. Then, $P[W = 0] = q_{tot}$ and $P[W = 1] = 1 - q_{tot}$.

Now, we analyze the variance of Δ. By the law of total variance,

$$
\mathtt{Var}[\Delta] = \mathbb{E}[\mathtt{Var}[\Delta|W]] + \mathtt{Var}[\mathbb{E}[\Delta|W]]
$$

Recall $\Delta \in [-r, r]$. Then, the first term can be bounded by

$$
\begin{aligned}
\mathbb{E}[\mathtt{Var}[\Delta|W]] &= P[W = 0]\mathtt{Var}[\Delta|W = 0] + P[W = 1]\mathtt{Var}[\Delta|W = 1] \\
&= q_{tot}\mathtt{Var}[\Delta|\Delta = 0] + (1 - q_{tot})\mathtt{Var}[\Delta|\Delta \neq 0] \\
&= (1 - q_{tot})\mathtt{Var}[\Delta|\Delta \neq 0] \\
&\leq (1 - q_{tot})r^2
\end{aligned}
$$

where the last inequality follows from the fact that if a random variable is in the range $[m, M]$, then its variance is bounded by $\frac{(M-m)^2}{4}$.

The second term can be expressed as

$$
\begin{aligned}
\mathtt{Var}[\mathbb{E}[\Delta|W]] &= \mathbb{E}_W[(\mathbb{E}[\Delta|W] - \mathbb{E}[\Delta])^2] \\
&= P[W = 0](\mathbb{E}[\Delta|W = 0] - \mathbb{E}[\Delta])^2 + P[W = 1](\mathbb{E}[\Delta|W = 1] - \mathbb{E}[\Delta])^2 \\
&= q_{tot}(\mathbb{E}[\Delta|\Delta = 0] - \mathbb{E}[\Delta])^2 + (1 - q_{tot})(\mathbb{E}[\Delta|\Delta \neq 0] - \mathbb{E}[\Delta])^2 \\
&= q_{tot}(\mathbb{E}[\Delta])^2 + (1 - q_{tot})(\mathbb{E}[\Delta|\Delta \neq 0] - \mathbb{E}[\Delta])^2 \qquad (A.7)
\end{aligned}
$$

Note that

$$
\mathbb{E}[\Delta] = P[W = 0]\mathbb{E}[\Delta|\Delta = 0] + P[W = 1]\mathbb{E}[\Delta|\Delta \neq 0]
$$

$$= (1 - q_{tot})\mathbb{E}[\Delta|\Delta \neq 0] \tag{A.8}$$

Plugging (A.8) into (A.7), we obtain

$$\mathtt{Var}[\mathbb{E}[\Delta|W]] = (q_{tot}(1 - q_{tot})^2 + q_{tot}^2(1 - q_{tot}))(\mathbb{E}[\Delta|\Delta \neq 0])^2$$

Since $|\Delta| \leq r$, $(\mathbb{E}[\Delta|\Delta \neq 0])^2 \leq r^2$. Therefore,

$$\mathtt{Var}[\mathbb{E}[\Delta|W]] \leq q_{tot}(1 - q_{tot})r^2$$

It follows that

$$\mathtt{Var}[\Delta] \leq (1 - q_{tot}^2)r^2$$

Given T samples, the application of Bennett's inequality in Lemma 4 yields

$$P\Big[\sum_{t=1}^{T}(Z\Delta_t - \mathbb{E}[Z\Delta_t]) > \epsilon'\Big] \leq \exp\Big(- \frac{T(1 - q_{tot}^2)}{4} h(\frac{2\epsilon'}{TZr(1 - q_{tot}^2)})\Big)$$

By letting $\epsilon = \epsilon'/T$,

$$P\big[(Z\bar{\Delta} - \mathbb{E}[Z\Delta]) > \epsilon\big] \leq \exp\Big(- \frac{T(1 - q_{tot}^2)}{4} h(\frac{2\epsilon}{Zr(1 - q_{tot}^2)})\Big)$$

Therefore, the number of tests T we need in order to get an $(\epsilon/(2\sqrt{N}), \delta/(N(N-1)))$-approximation to the difference of two Shapley values for a single pair of data points is

$$T \geq \frac{4}{(1 - q_{tot}^2)h(\frac{\epsilon}{Z\sqrt{N}r(1-q_{tot}^2)})} \log \frac{N(N-1)}{\delta}$$

By union bound, the number of tests T for achieving $(\epsilon/(2\sqrt{N}), \delta/(N(N-1)))$-approximation to the difference of the Shapley values for all $N(N-1)/2$ pairs of data points is

$$T \geq \frac{8}{(1 - q_{tot}^2)h(\frac{\epsilon}{Z\sqrt{N}r(1-q_{tot}^2)})} \log \frac{N(N-1)}{2\delta}$$

By Lemma 2, we approximate the Shapley value up to (ϵ, δ) with $(\epsilon/(2\sqrt{N}), \delta/(N(N-1)))$ approximations to all $N(N-1)/2$ pairs of data points.

COMPLEXITY CALCULATION. It can be shown that $q_{\text{tot}} = 1 - \frac{2}{Z}$ and so

$$Z(1 - q_{\text{tot}}^2) = Z(1 - q_{\text{tot}})(1 + q_{\text{tot}}) = 2(1 + q_{\text{tot}}) \in [2, 4] \tag{A.9}$$

Therefore, as $N \to \infty$,

$$\frac{\epsilon}{Zr\sqrt{N}(1 - q_{\text{tot}}^2)} \to 0 \tag{A.10}$$

The Tayor series of $h(u)$ centered at 0 is $\frac{u^2}{2} + \cdots$. Thus, we have

$$\begin{aligned}
&\frac{8}{(1-q_{tot}^2)h(\frac{\epsilon}{Z\sqrt{N}r(1-q_{tot}^2)})} \log \frac{N(N-1)}{2\delta} && \text{(A.11)}\\
&= \mathcal{O}\bigg(\frac{\log N}{(1-q_{tot}^2)\frac{\epsilon^2}{Z^2Nr^2(1-q_{tot}^2)^2}}\bigg) && \text{(A.12)}\\
&= \mathcal{O}(NZ^2(1-q_{tot}^2)\log N) && \text{(A.13)}\\
&= \mathcal{O}(NZ\log N) && \text{(A.14)}
\end{aligned}$$

Since $Z \leq 2(\log(N-1)+1)$, we have $\mathcal{O}(NZ\log N) = \mathcal{O}(N(\log N)^2)$. □

A.5 Proof of Theorem 2

Theorem. *Suppose that $U(\cdot)$ is monotone. There exists some constant C' such that if $M \geq C'(K\log(N/(2K)) + \log(2/\delta))$ and $T \geq \frac{2r^2}{\epsilon^2}\log\frac{4M}{\delta}$, except for an event of probability no more than δ, the output of Algorithm 3 obeys*

$$\|\hat{s} - s\|_2 \leq C_{1,K}\epsilon + C_{2,K}\frac{\sigma_K(s)}{\sqrt{K}} \tag{A.15}$$

for some constants $C_{1,K}$ and $C_{2,K}$.

Proof. Due to the monotonicity of $U(\cdot)$, $\hat{y}_{m,t}$ can be lower bounded by $-\frac{1}{\sqrt{M}}\sum_{i=1}^{N} U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t}) = -\frac{1}{\sqrt{M}}U(\pi_t) \geq -\frac{r}{\sqrt{M}}$; the upper bound can be similarly analyzed. Thus, the range of $\hat{y}_{m,t}$ is $[-1/\sqrt{M}r, 1/\sqrt{M}r]$. Since $\mathbb{E}[\hat{y}_{m,t}] = \sum_{i=1}^{N} A_{m,i}\mathbb{E}[U(P_i^{\pi_t} \cup \{i\}) - U(P_i^{\pi_t})] = \sum_{i=1}^{N} A_{m,i}s_i$ for all $m = 1, \ldots, M$, an application of Hoeffding's bound gives

$$P[\|As - \bar{y}\|_2 \geq \epsilon] \leq P[\|As - \bar{y}\|_\infty \geq \frac{\epsilon}{\sqrt{M}}] \tag{A.16}$$

$$\leq \sum_{m=1}^{M} P[|A_m s - \bar{y}_m| \geq \frac{\epsilon}{\sqrt{M}}] \tag{A.17}$$

$$\leq 2M \exp(-\frac{\epsilon^2 T}{2r^2}) \tag{A.18}$$

Let $s = \Delta s + \bar{s}$. Thus, $P[\|A(\bar{s} + \Delta s) - \bar{y}\|_2 \leq \epsilon]$ holds with probability at least $\delta/2$ provided

$$T \geq \frac{2r^2}{\epsilon^2} \log \frac{4M}{\delta}. \tag{A.19}$$

By the random matrix theory, the restricted isometry constant of A satisfies $\delta_{2K} \leq C_\delta = 0.465$ with probability at least $1 - \delta/2$ if

$$M \geq CC_\delta^{-2}(2K \log(N/(2K)) + \log(2/\delta)) \tag{A.20}$$

where $C > 0$ is a universal constant.

Applying the Theorem 2.7 in (Rauhut, 2010), we obtain that the output of Algorithm 2 satisfies

$$\|\hat{s} - s\| = \|\Delta s^* - \Delta s\| \leq C_{1,K}\epsilon + C_{2,K}\frac{\sigma_K(s)}{\sqrt{K}} \tag{A.21}$$

with probability at least $1 - \delta$ provided that (A.19) holds and
$M \geq C'(K \log(N/(2K)) + \log(2/\delta))$ for some constant C'. □

A.6 Proof of Theorem 3

For the proof of Theorem 3 we need the following definition of a *stable utility function*.

Definition 3. *A utility function $U(\cdot)$ is called λ-stable if*

$$\max_{i,j \in I, S \subseteq I \setminus \{i,j\}} |U(S \cup \{i\}) - U(S \cup \{j\})| \leq \frac{\lambda}{|S|+1}$$

Then, Shapley values calculated from λ-stable utility functions have the following property. If $U(\cdot)$ is λ-stable, then for all $i, j \in I$ and $i \neq j$

$$s_i - s_j \leq \frac{\lambda(1 + \log(N-1))}{N-1}$$

Proof. By Lemma 1, we have

$$s_i - s_j \leq \frac{1}{N-1} \sum_{S \subseteq I \setminus \{i,j\}} \frac{1}{C_{N-2}^{|S|}} \frac{\lambda}{|S|+1} = \frac{1}{N-1} \sum_{|S|=0}^{N-2} \frac{\lambda}{|S|+1}$$

Recall the bound on the harmonic sequences

$$\sum_{k=1}^{N} \frac{1}{k} \leq 1 + \log(N)$$

which gives us

$$s_i - s_j \leq \frac{\lambda(1 + \log(N-1))}{N-1}$$

□

Then, we can prove Theorem 3.

Theorem. *For a learning algorithm $A(\cdot)$ with uniform stability $\beta = \frac{C_{stab}}{|S|}$, where $|S|$ is the size of the training set and C_{stab} is some constant. Let the utility of D be $U(D) = M - L_{val}(A(D), D_{val})$, where $L_{val}(A(D), D_{val}) =$*

$\frac{1}{N}\sum_{i=1}^{N} l(A(D), z_{val,i})$ and $0 \leq l(\cdot,\cdot) \leq M$. Then, $s_i - s_j \leq 2C_{stab}\frac{1+\log(N-1)}{N-1}$ and the Shapley difference vanishes as $N \to \infty$.

Proof. For any $i, j \in I$ and $i \neq j$,

$$\begin{aligned}
&|U(S \cup \{i\}) - U(S \cup \{j\})| \\
&= |\frac{1}{N}\sum_{i=1}^{N}[l(A(S \cup \{i\}), z_{\text{val},i}) - l(A(S \cup \{j\}), z_{\text{val},i})]| \\
&\leq \frac{1}{N}\sum_{i=1}^{N}|l(A(S \cup \{i\}), z_{\text{val},i}) - l(A(S), z_{\text{val},i})| \\
&+ |l(A(S), z_{\text{val},i}) - l(A(S \cup \{j\}), z_{\text{val},i})| \\
&\leq \frac{1}{N}\sum_{i=1}^{N}\frac{2C_{\text{stab}}}{|S|+1} = \frac{2C_{\text{stab}}}{|S|+1}
\end{aligned}$$

Combining the above inequality with Proposition A.6 proves the theorem. □

A.7 Proof of Theorem 4

Theorem. *Consider the value attribution scheme that assign the value $\hat{s}(U,i) = C_U[U(S \cup \{i\}) - U(S)]$ to user i where $|S| = N - 1$ and C_U is a constant such that $\sum_{i=1}^{N}\hat{s}(U,i) = U(I)$. Consider two utility functions $U(\cdot)$ and $V(\cdot)$. Then, $\hat{s}(U+V,i) \neq \hat{s}(U,i) + \hat{s}(V,i)$ unless $V(I)[\sum_{i=1}^{N} U(S \cup \{i\}) - U(S)] = U(I)[\sum_{i=1}^{N} V(S \cup \{i\}) - V(S)]$.*

Proof. Consider two utility functions $U(\cdot)$ and $V(\cdot)$. The values attributed to user i under these two utility functions are given by

$$\hat{s}(U,i) = C_U[U(S \cup \{i\}) - U(S)]$$

and

$$\hat{s}(V,i) = C_V[V(S \cup \{i\}) - V(S)]$$

where C_U and C_V are constants such that $\sum_{i=1}^{N} \hat{s}(U,i) = U(I)$ and $\sum_{i=1}^{N} \hat{s}(V,i) = V(I)$. Now, we consider the value under the utility function $W(S) = U(S) + V(S)$:

$$\hat{s}(U+V,i) = C_W[U(S \cup \{i\}) - U(S) + V(S \cup \{i\}) - V(S)]$$

where

$$C_W = \frac{U(I) + V(I)}{\sum_{i=1}^{N}[U(S \cup \{i\}) - U(S) + V(S \cup \{i\}) - V(S)]}$$

Then, $\hat{s}(U+V,i) = \hat{s}(U,i) + \hat{s}(V,i)$ if and only if $C_U = C_V = C_W$, which is equivalent to

$$V(I)[\sum_{i=1}^{N} U(S \cup \{i\}) - U(S)] = U(I)[\sum_{i=1}^{N} V(S \cup \{i\}) - V(S)]$$

□

B

APPENDIX: CHAPTER 4

B.1 Additional Experiments

B.1.1 *Runtime Comparision for Computing the Unweighted KNN SV*

For each dataset, we randomly selected 100 test points, computed the SV of all training points with respect to each test point, and reported the average runtime across all test points. The results for $K = 2, 5$ are presented in Figure B.1. We can see that the LSH-based method can bring a 3×-5× speed-up compared with the exact algorithm.

Dataset	**Size**	**C**	**K=2**		**K=5**	
			Exact	**LSH**	**Exact**	**LSH**
`CIFAR-10`	6E+4	1.2802	0.83s	0.25s	0.82s	0.26s
`ImageNet`	1E+6	1.2163	12.71s	3.29s	12.57s	3.25s
`Yahoo10m`	1E+7	1.3456	198.73s	41.83s	200.06s	39.20s

Figure B.1: Average runtime of the exact and the LSH-based approximation algorithm for computing the unweighted *KNN* SV for a single test point. We take $\epsilon, \delta = 0.1$ and $K = 2, 5$.

B.2 Proof of Theorem 6

Proof. We first observe that if the true Shapley value $|s_{\alpha_i}| \leq \min(\frac{1}{i}, \frac{1}{K})$, then $|s_i| \leq \epsilon$ for $i \geq i^* = \max(K, \lceil 1/\epsilon \rceil)$. Hence, when $i \geq i^*$, the approximation error is given by

$$|\hat{s}_{\alpha_i} - s_{\alpha_i}| = |s_{\alpha_i}| \leq \epsilon. \tag{B.1}$$

When $i \leq i^* - 1$, $\hat{s}_{\alpha_i}$ and s_{α_i} follow the same recursion, i.e.,

$$\hat{s}_{\alpha_i} - \hat{s}_{\alpha_{i+1}} = s_{\alpha_i} - s_{\alpha_{i+1}} = \frac{\mathbb{1}[y_{\alpha_i} = y_{\text{test}}] - \mathbb{1}[y_{\alpha_{i+1}} = y_{\text{test}}]}{K} \frac{\min(K-1, i-1)+1}{i}. \tag{B.2}$$

As a result, we have

$$|\hat{s}_{\alpha_i} - s_{\alpha_i}| = |\hat{s}_{\alpha_{i+1}} - s_{\alpha_{i+1}}| = \cdots = |\hat{s}_{\alpha_{i^*}} - s_{\alpha_{i^*}}| \leq \epsilon \tag{B.3}$$

To sum up, $|\hat{s}_{\alpha_i} - s_{\alpha_i}| \leq \epsilon$ for all $i = 1, \ldots, N$, provided that $|s_{\alpha_i}| \leq \min(\frac{1}{i}, \frac{1}{K})$. In the following, we will prove that the aforementioned condition is satisfied.

We can convert the recursive expression of the KNN Shapley value in Theorem 5 to a non-recursive one:

$$s_{\alpha_N} = \frac{\mathbb{1}[y_{\alpha_N} = y_{\text{test}}]}{N} \tag{B.4}$$

$$s_{\alpha_i} = \frac{\mathbb{1}[y_{\alpha_i} = y_{\text{test}}]}{i} - \sum_{j=i+1}^{N} \frac{\mathbb{1}[y_{\alpha_j} = y_{\text{test}}]}{j(j-1)} \text{ for } i \geq K \tag{B.5}$$

$$s_{\alpha_i} = \frac{\mathbb{1}[y_{\alpha_i} = y_{\text{test}}]}{K} - \sum_{j=K+1}^{N} \frac{\mathbb{1}[y_{\alpha_j} = y_{\text{test}}]}{j(j-1)} \text{ for } i \leq K-1 \tag{B.6}$$

We examine the bound on the absolute value of the Shapley value in three cases: (1) $i = N$, (2) $i \geq K$, and (3) $i \leq K-1$.

Case (1). It is easy to verify that $|s_{\alpha_N}| \leq \frac{1}{N}$.

Case (2). We can bound the second term in (B.5) by

$$0 \leq \sum_{j=i+1}^{N} \frac{\mathbb{1}[y_{\alpha_j} = y_{\text{test}}]}{j(j-1)} \leq \sum_{j=i+1}^{N} \frac{1}{j(j-1)} = \sum_{j=i+1}^{N} (\frac{1}{j-1} - \frac{1}{j}) = \frac{1}{i} - \frac{1}{N} \tag{B.7}$$

Thus, s_{α_i} can be bounded by

$$-(\frac{1}{i} - \frac{1}{N}) \leq s_{\alpha_i} \leq \frac{1}{i}, \tag{B.8}$$

which yields the bound on the absolute value of s_{α_i}:

$$|s_{\alpha_i}| \leq \frac{1}{i}. \tag{B.9}$$

Case (3). The absolute value of s_{α_i} for $i \leq K-1$ can be bounded using a similar technique as in Case (2). By (B.6), we have

$$-(\frac{1}{K} - \frac{1}{N}) \leq s_{\alpha_i} \leq \frac{1}{K} \tag{B.10}$$

Therefore, $|s_{\alpha_i}| \leq 1/K$.

Summarizing the results in Case (1), (2), and (3), we obtain $|s_{\alpha_i}| \leq \min(1/i, 1/K)$ for $i = 1, \ldots, N$.

□

B.3 Proof of Theorem 7

Proof. For the hashing function $h(x) = \left\lfloor \frac{w^T x + b}{t} \right\rfloor$, (Datar et al., 2004) have shown that

$$P(h(x_i) = h(x_{\text{test}})) = f_h(\|x_i - x_{\text{test}}\|_p) \tag{B.11}$$

where the function $f_h(a) = \int_0^t \frac{1}{a} f_p(\frac{z}{a}(1 - \frac{z}{t})dz$ is monotonically decreasing with a. f_p is the probability density function of the absolute value of a p-stable random variable.

Suppose the data are normalized by a factor such that $D_{\text{mean}} = 1$. Since such a normalization does not change the nearest neighbor search results, $D_k = 1/C_k$ for $k = 1, \ldots, K$. Denote the probability for one random test point x_{test} and a random training point to have the same code with one hash function by p_{rand} and the probability for x_{test} and its k-nearest neighbor to have the same code by p_{nn}^k. According to (B.11),

$$p_{\text{rand}} = f_h(1) \tag{B.12}$$

and

$$p_{\text{nn},k} = f_h(1/C_k) \tag{B.13}$$

because the expected distance between x_{test} and a random training point is $D_{\text{mean}} = 1$, and the expected distance between x_{test} and its k-nearest neighbor is $1/C_k$.

Let E_k denote the event that the k-nearest neighbor of x_{test} is included by one of the hash tables. Then, the probability of the inclusion of all K nearest neighbors is

$$P(E_1, \ldots, E_K) = 1 - P(\cup_{k=1}^{K} \bar{E}_k) \tag{B.14}$$

$$\geq 1 - \sum_{k=1}^{K} P(\bar{E}_k). \tag{B.15}$$

We want to make sure that $P(E_1, \ldots, E_K) \geq 1 - \delta$, so it suffices to let $P(\bar{E}_k) \leq \delta/K$ for all $k = 1, \ldots, K$.

Suppose there are m hash bits in one table and l hash tables in LSH. The probability that the true k-nearest neighbor has the same code as the query in one hash table is $p_{\text{nn},k}^{m}$. Hence, the probability that the true k-nearest neighbor is missed by l hash tables is $P(\bar{E}_k) = (1 - p_{\text{nn},k}^{m})^l$. In order to ensure $P(\bar{E}_k) \leq \delta/K$, we need

$$l \geq \frac{\log \frac{\delta}{K}}{\log(1 - p_{\text{nn},k}^{m})} \tag{B.16}$$

The RHS is upper bounded by $\frac{-\log \frac{\delta}{K}}{p_{\text{nn},k}^{m}} = p_{\text{nn},k}^{-m} \log \frac{K}{\delta}$. Therefore, it suffices to ensure

$$l \geq p_{\text{nn},k}^{-m} \log \frac{K}{\delta} \tag{B.17}$$

Note that $p_{\text{nn},k} = p_{\text{rand}}^{\frac{\log p_{\text{nn},k}}{\log p_{\text{rand}}}}$ and we can choose $Np_{\text{rand}}^m = \mathcal{O}(1)$, i.e., $m = \mathcal{O}(\frac{\log N}{\log p_{\text{rand}}^{-1}})$, as discussed in (Gionis, Indyk, Motwani, et al., 1999). Hence,

$$p_{\text{nn},k}^m = p_{\text{rand}}^{m\frac{\log p_{\text{nn},k}}{\log p_{\text{rand}}}} = \mathcal{O}((\frac{1}{N})^{\frac{\log p_{\text{nn},k}}{\log p_{\text{rand}}}}) = \mathcal{O}(N^{-g(C_k)}) \tag{B.18}$$

where $g(C_k) = \frac{\log p_{\text{nn},k}}{\log p_{\text{rand}}} = \frac{\log f_h(1/C_k)}{\log f_h(1)}$. Plugging (B.18) into (B.16), we obtain

$$l \geq \mathcal{O}(N^{g(C_k)} \log \frac{K}{\delta}) \tag{B.19}$$

In order to guarantee $P(\bar{E}_k) \leq \delta/K$ for all $k = 1, \cdots, K$, the number of hash tables needed is

$$\mathcal{O}(N^{g(C_K)} \log \frac{K}{\delta}) \tag{B.20}$$

□

B.4 Detailed Algorithms and Proofs for the Extensions

B.4.1 *Unweighted KNN Regression*

For regression tasks, we define the utility function by the negative mean square error of an unweighted KNN regressor:

$$U(S) = -\left(\frac{1}{K}\sum_{k=1}^{\min\{K,|S|\}} y_{\alpha_k(S)} - y_{\text{test}}\right)^2 \tag{B.21}$$

The following theorem provides a simple iterative procedure to compute the SV for unweighted KNN regression. The derivation of the theorem requires to analyze the utility difference between two adjacent training points, similar to KNN classification.

Proof of Theorem 9. W.l.o.g., we assume that $x_1, \ldots, x_n$ are sorted according to their similarity to x_{test}, that is, $x_i = x_{\alpha_i}$. We split a subset $S \subseteq I \setminus \{i, i+1\}$

into two disjoint sets S_1 and S_2 such that $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$. Given two neighboring points with indices $i, i+1 \in I$, we constrain S_1 and S_2 to $S_1 \subseteq \{1, \ldots, i-1\}$ and $S_2 \subseteq \{i+2, \ldots, N\}$.

We analyze the difference between s_i and s_{i+1} by considering the following cases:

Case 1. Consider the case $|S_1| \geq K$. We know that $i > K$ and therefore $U(S \cup \{i\}) = U(S \cup \{i+1\}) = U(S)$. From Lemma 1, it follows that

$$\begin{aligned} s_i - s_{i+1} &= \frac{1}{N-1} \sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \sum_{\substack{S_1 \subseteq \{1,\ldots,i-1\},\\ S_2 \subseteq \{i+2,\ldots,N\}:\\ |S_1|+|S_2|=k, |S_1| \geq K}} \left[U(S \cup \{i\}) - U(S \cup \{i+1\}) \right] \\ &= 0. \end{aligned}$$

Case 2. Consider the case $|S_1| < K$. The difference between $U(S \cup \{i\})$ and $U(S \cup \{i+1\})$ can be expressed as

$$\begin{aligned} &U(S \cup \{i\}) - U(S \cup \{i+1\}) \\ =&\left(\frac{1}{K} \sum_{j=1}^{K} y_{\alpha_j(S \cup \{i+1\})} - y_{\text{test}}\right)^2 - \left(\frac{1}{K} \sum_{j=1}^{K} y_{\alpha_j(S \cup \{i\})} - y_{\text{test}}\right)^2 \\ =&\frac{1}{K}(y_{i+1} - y_i) \cdot \left(\frac{1}{K}(y_{i+1} + y_i) - 2y_{\text{test}} + \frac{2}{K} \sum_{j=1,\ldots,K-1} y_{\alpha_j(S)} \right) \end{aligned}$$

By Lemma 1, the Shapley difference between i and $i+1$ is

$$\begin{aligned} s_i - s_{i+1} = &\frac{1}{K}(y_{i+1} - y_i) \\ \cdot &\underbrace{\left(\frac{1}{N-1} \sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \sum_{\substack{S_1 \subseteq \{1,\ldots,i-1\},\\ S_2 \subseteq \{i+2,\ldots,N\}:\\ |S_1|+|S_2|=k, |S_1| \leq K-1}} \left(\frac{1}{K}(y_{i+1} + y_i) - 2y_{\text{test}}\right) \right.}_{U_1} \end{aligned}$$

$$\underbrace{+\frac{2}{K}\frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{\substack{S_1\subseteq\{1,\ldots,i-1\},\\ S_2\subseteq\{i+2,\ldots,N\}:\\ |S_1|+|S_2|=k,|S_1|\leq K-1}}\sum_{j=1,\ldots,K-1} y_{\alpha_j(S)}\Big)}_{U_2}$$

We firstly simplify U_1. Note that $\frac{1}{K}(y_{i+1}+y_i)-2y_{\text{test}}$ does not depend on the summation; as a result, we have

$$\begin{aligned} U_1 =&(\frac{1}{K}(y_{i+1}+y_i)-2y_{\text{test}})\frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\Bigg(\sum_{\substack{S_1\subseteq\{1,\ldots,i-1\},\\ S_2\subseteq\{i+2,\ldots,N\}:\\ |S_1|+|S_2|=k,|S_1|\leq K-1}} 1\Bigg) \\ =&(\frac{1}{K}(y_{i+1}+y_i)-2y_{\text{test}})\frac{1}{N-1}\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{m=0}^{\min(K-1,k)}\binom{i-1}{m}\binom{N-i-1}{k-m} \end{aligned} \tag{B.22}$$

The sum of binomial coefficients in (B.22) can be further simplified as follows:

$$\begin{aligned} &\sum_{k=0}^{N-2}\frac{1}{\binom{N-2}{k}}\sum_{m=0}^{\min(K-1,k)}\binom{i-1}{m}\binom{N-i-1}{k-m} \\ =&\sum_{m=0}^{\min(K-1,i-1)}\sum_{k=0}^{N-i-1}\frac{\binom{i-1}{m}\binom{N-i-1}{k}}{\binom{N-2}{m+k}} \\ =&\sum_{m=0}^{\min(K-1,i-1)}\frac{N-1}{i} \\ =&\min(K,i)\frac{N-1}{i} \end{aligned}$$

where the second equality follows from the binomial coefficient identity $\sum_{j=0}^{M}\frac{\binom{N}{i}\binom{M}{j}}{\binom{N+M}{i+j}}=\frac{M+N+1}{N+1}$. Hence,

$$U_1=(\frac{1}{K}(y_{i+1}+y_i)-2y_{\text{test}})\frac{\min(K,i)}{i}$$

Then, we analyze U_2. We let

$$\sum_{\substack{S_1\subseteq\{1,\dots,i-1\},\\ S_2\subseteq\{i+2,\dots,N\}:\\ |S_1|+|S_2|=k,|S_1|\le K-1}} \sum_{j=1,\dots,K-1} y_{\alpha_j(S)} = \sum_{l\in I\setminus\{i,i+1\}} c_l y_l \tag{B.23}$$

where c_l counts the number of occurrences of y_l in the left-hand side expression and

$$c_l = \begin{cases} \sum_{m=0}^{\min(K-2,k-1)} \binom{i-2}{m}\binom{N-i-1}{k-m-1} \text{ if } l \in \{1,\dots,i-1\} \\ \sum_{m=0}^{\min(K-2,k-1)} \binom{l-3}{m}\binom{N-l}{k-m-1} \text{ if } l \in \{i+2,\dots,N\} \end{cases} \tag{B.24}$$

Plugging in (B.23) and (B.24) into U_2 yields

$$\begin{aligned} U_2 = {} & \frac{2}{K(N-1)} \sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \Bigg[\sum_{l\in\{1,\dots,i-1\}} \sum_{m=0}^{\min(K-2,k-1)} \binom{i-2}{m}\binom{N-i-1}{k-m-1} y_l \\ & + \sum_{l\in\{i+2,\dots,N\}} \sum_{m=0}^{\min(K-2,k-1)} \binom{l-3}{m}\binom{N-l}{k-m-1} y_l \Bigg] \\ = {} & \frac{2}{K(N-1)} \Bigg[\sum_{l\in\{1,\dots,i-1\}} y_l \Bigg] \cdot \underbrace{\Bigg[\sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \sum_{m=0}^{\min(K-2,k-1)} \binom{i-2}{m}\binom{N-i-1}{k-m-1} \Bigg]}_{U_{21}} \\ & + \frac{2}{K(N-1)} \Bigg[\sum_{l\in\{i+2,\dots,N\}} y_l \cdot \underbrace{\sum_{k=0}^{N-2} \frac{1}{\binom{N-2}{k}} \sum_{m=0}^{\min(K-2,k-1)} \binom{l-3}{m}\binom{N-l}{k-m-1}}_{U_{22}} \Bigg] \end{aligned} \tag{B.25}$$

Using the binomial coefficient identity $\sum_{j=0}^{M} \frac{\binom{N}{i}\binom{M}{j}}{\binom{N+M+1}{i+j+1}} = \frac{(i+1)(M+N+2)}{(N+2)(N+1)}$, we obtain

$$\begin{aligned} U_{21} & = \sum_{m=0}^{\min(K-2,i-2)} \sum_{k=0}^{N-i-1} \frac{\binom{i-2}{m}\binom{N-i-1}{k}}{\binom{N-2}{k+m+1}} \\ & = \sum_{m=0}^{\min(K-2,i-2)} \frac{N-1}{(i-1)i}(m+1) \end{aligned}$$

$$= \frac{N-1}{(i-1)i} \frac{\min(K,i)\min(K-1,i-1)}{2} \tag{B.26}$$

and

$$\begin{aligned} U_{22} &= \sum_{m=0}^{\min(K-2,l-3)} \sum_{k=0}^{N-l} \frac{\binom{l-3}{m}\binom{N-l}{k}}{\binom{N-2}{k+m+1}} \\ &= \sum_{m=0}^{\min(K-2,l-3)} \frac{N-1}{(l-1)(l-2)}(m+1) \\ &= \frac{N-1}{(l-1)(l-2)} \frac{\min(K,l-1)\min(K-1,l-2)}{2} \end{aligned} \tag{B.27}$$

Now, we plug (B.26) and (B.27) into the expression of U_2 in (B.25). Rearranging (B.25) gives us

$$\begin{aligned} U_2 = &\frac{1}{K} \sum_{l \in \{1,\ldots,i-1\}} y_l \frac{\min(K,i)\min(K-1,i-1)}{(i-1)i} \\ &+ \frac{1}{K} \sum_{l \in \{i+2,\ldots,N\}} y_l \frac{\min(K,l-1)\min(K-1,l-2)}{(l-1)(l-2)} \end{aligned}$$

Therefore, we have

$$\begin{aligned} &s_i - s_{i+1} \\ &= \frac{1}{K}(y_{i+1} - y_i)(U_1 + U_2) \\ &= \frac{1}{K}(y_{i+1} - y_i) \cdot \Bigg[\Big(\frac{1}{K}(y_{i+1} + y_i) - 2y_{\text{test}} \Big) \frac{\min(K-1,i-1)+1}{i} \\ &+ \frac{1}{K} \sum_{l \in \{1,\ldots,i-1\}} y_l \frac{\min(K,i)\min(K-1,i-1)}{(i-1)i} \\ &+ \frac{1}{K} \sum_{l \in \{i+2,\ldots,N\}} y_l \frac{\min(K,l-1)\min(K-1,l-2)}{(l-1)(l-2)} \Bigg] \end{aligned}$$

Now, we analyze the formula for s_N, the starting point of the recursion. Since x_N is farthest to x_{test} among all training points, x_N results in non-zero marginal utility only when it is added to a set of size smaller than K. Hence, given $\gamma = \frac{U(\{N\})}{N}$, s_N can be written as

$$\begin{aligned}
s_N &= \frac{1}{N}\sum_{k=0}^{K-1}\frac{1}{\binom{N-1}{k}}\sum_{|S|=k, S\subseteq I\setminus\{N\},} U(S\cup\{N\}) - U(S) \\
&= \frac{1}{N}\sum_{k=1}^{K-1}\frac{1}{\binom{N-1}{k}}\sum_{|S|=k, S\subseteq I\setminus\{N\}}\left[(\frac{1}{K}\sum_{i\in S} y_i - y_{\text{test}})^2 - (\frac{1}{K}\sum_{i\in S\cup\{N\}} y_i - y_{\text{test}})^2\right] + \gamma \\
&= \frac{1}{N}\sum_{k=1}^{K-1}\frac{1}{\binom{N-1}{k}}\sum_{|S|=k, S\subseteq I\setminus\{N\}}\left[(-\frac{1}{K}y_N)\cdot(\frac{2}{K}\sum_{i\in S} y_i + \frac{1}{K}y_N - 2y_{\text{test}})\right] + \gamma \\
&= -\frac{K-1}{NK}y_N(\frac{1}{K}y_N - 2y_{\text{test}}) - \frac{2}{NK^2}y_N\sum_{k=1}^{K-1}\frac{\binom{N-2}{k-1}}{\binom{N-1}{k}}\sum_{l\in I\setminus\{N\}} y_l + \gamma \\
&= -\frac{1}{N}y_N\left[\frac{K-1}{K}(\frac{1}{K}y_N - 2y_{\text{test}}) + \frac{2}{K^2}(\sum_{l\in I\setminus\{N\}} y_l)\sum_{k=1}^{K-1}\frac{k}{N-1}\right] + \gamma \\
&= -\frac{K-1}{NK}y_N\left[\frac{1}{K}y_N - 2y_{\text{test}} + \frac{1}{N-1}\sum_{l\in I\setminus\{N\}} y_l\right] + \gamma
\end{aligned}$$

□

B.4.2 *Proof of Theorem 10*

Proof of Theorem 10. Without loss of generality, we assume that the training points are sorted according to their distance to x_{test}, such that $d(x_1, x_{\text{test}}) \leq \ldots \leq d(x_N, x_{\text{test}})$.

We start by analyzing the SV for x_N. Since the farthest training point does not affect the utility of S unless $|S| \leq K-1$, we have

$$s_N = \frac{1}{N}\sum_{k=0}^{K-1}\frac{1}{\binom{N-1}{k}}\sum_{|S|=k, S\subseteq I\setminus\{N\}}\left[U(S\cup\{N\}) - U(S)\right]$$

For $i \leq N-1$, the application of Lemma 1 yields

$$s_i - s_{i+1} = \frac{1}{N-1} \sum_{k=0}^{N-2} \sum_{|S|=k, S \subseteq I \setminus \{i,i+1\}} \frac{1}{\binom{N-2}{k}} \cdot \left[U(S \cup \{i\}) - U(S \cup \{i+1\}) \right] \tag{B.28}$$

Recall that for KNN utility functions, $U(S)$ only depends on the K training points closest to x_{test}. Therefore, we can also write $s_i - s_{i+1}$ as follows:

$$s_i - s_{i+1} = \frac{1}{N-1} \sum_{k'=0}^{K-1} \sum_{S' \in B_{k'}(i) \cap B_{k'}(i+1)} M_{i,i+1}^{k'} \left[U(S' \cup \{i\}) - U(S' \cup \{i+1\}) \right] \tag{B.29}$$

which can be computed in at most $\sum_{k'=0}^{K-1} \binom{N-2}{k'} \sim \mathcal{O}(N^K)$, in contrast to $\mathcal{O}(2^{N-2})$ with (B.28). Our goal is thus to find $M_{i,i+1}^{k'}$ such that the right-hand sides of (B.29) and (B.28) are equal. More specifically, for each $S' \in B_{k'}(i) \cap B_{k'}(i+1)$, we want to count the number of $S \subseteq I \setminus \{i, i+1\}$ such that $|S| = k$, and $U(S \cup \{i\}) = U(S' \cup \{i\})$ and $U(S \cup \{i+1\}) = U(S' \cup \{i+1\})$; denoting the count by $C_{i,i+1}^{k,k'}$, we have

$$M_{i,i+1}^{k'} = \sum_{k=0}^{N-2} C_{i,i+1}^{k,k'} / \binom{N-2}{k}. \tag{B.30}$$

When $k' \leq K-2$, only $S = S'$ satisfies $U(S \cup \{i\}) = U(S' \cup \{i\})$ and $U(S \cup \{i+1\}) = U(S' \cup \{i+1\})$. Therefore,

$$C_{i,i+1}^{k,k'} = \begin{cases} 1 \text{ if } k' \leq K-2 \text{ and } k = k' \\ 0 \text{ otherwise} \end{cases} \tag{B.31}$$

When $k' = K-1$, there will be multiple subsets S of $I \setminus \{i, i+1\}$ that obey $U(S \cup \{i\}) = U(S' \cup \{i\})$ and $U(S \cup \{i+1\}) = U(S' \cup \{i+1\})$. Let r denote the index of the training point that is farthest to x_{test} among $S \cup \{i, i+1\}$, i.e., $r = \max S \cup \{i, i+1\}$. Note that adding any training points with indices larger than r into $S' \cup \{i\}$ or $S' \cup \{i+1\}$ would not affect their utility. Hence,

$$C^{k,k'}_{i,i+1} = \begin{cases} \binom{N-r}{k-K+1} \text{ if } k' = K-1, k \geq k' \\ 0 \text{ otherwise} \end{cases} \tag{B.32}$$

Combining (B.29), (B.30), (B.31), and (B.32) yields the recursion in (4.29) and (4.30). □

B.4.3 *Valuing Computation*

B.4.3.1 *Weighted KNN*

Theorem 16. *Consider the utility function in (4.34), where $U(\cdot)$ is the weighted KNN performance measure in (4.26) or (4.27) with some weights $w_{\alpha_k(S)}$. Let $B_k(i) = \{S : |S| = k, i \notin S, S \subseteq I\}$, for $i = 1, \ldots, N$ and $k = 0, \ldots, K$. Let $r(\cdot)$ be a function that maps the set of training data to their ranks in terms of similarity to x_{test}. Then, the SV of each training point and the computation contributor can be calculated recursively as follows:*

$$s_{\alpha_N} = \frac{1}{N+1} \sum_{k=0}^{K-1} \frac{1}{\binom{N}{k+1}} \sum_{S \in B_k(\alpha_N)} U(S \cup \{\alpha_N\}) - U(S) \tag{B.33}$$

$$\begin{aligned} s_{\alpha_{i+1}} &= s_{\alpha_i} + \frac{1}{N} \sum_{k=0}^{K-2} \frac{1}{\binom{N-1}{k+1}} \sum_{S \in B_k(\alpha_i) \cap B_k(\alpha_{i+1})} U(S \cup \{\alpha_i\}) - U(S \cup \{\alpha_{i+1}\}) \\ &+ \frac{1}{N} \sum_{k=K-1}^{N-2} \frac{1}{\binom{N-1}{k+1}} \sum_{S \in B_{K-1}(\alpha_i) \cap B_{K-1}(\alpha_{i+1})} \binom{N - \max r(S \cup \{\alpha_i, \alpha_{i+1}\})}{k-K+1} \\ &\cdot U(S \cup \{\alpha_i\}) - U(S \cup \{\alpha_{i+1}\}) \end{aligned} \tag{B.34}$$

$$s_C = U(I) - \sum_{i=1}^{N} s_i \tag{B.35}$$

B.5 Proof of Theorem 15

Proof. We will use Bennett's inequality to derive the approximation error associated with the estimator in (3.1). Bennett's inequality provides an upper bound on the deviation of the empirical mean from the true mean in terms of the variance of the underlying random variable. Thus, we first provide an upper bound on the variance of ϕ_i for $i = 1, \dots, N$.

Let the range of ϕ_i for $i = 1, \dots, N$ be denoted by $[-r, r]$. Further, let $q_i = P[\phi_i = 0]$. Let W_i be an indicator of whether or not $\phi_i = 0$, i.e., $W_i = \mathbb{1}[\phi_i \neq 0]$; thus $P[W_i = 0] = q_i$ and $P[W_i = 1] = 1 - q_i$.

We analyze the variance of ϕ_i. By the law of total variance,

$$\texttt{Var}[\phi_i] = \mathbb{E}[\texttt{Var}[\phi_i|W_i]] + \texttt{Var}[\mathbb{E}[\phi_i|W_i]] \tag{B.36}$$

Recall $\phi_i \in [-r, r]$. Then, the first term can be bounded by

$$\begin{aligned}
&\mathbb{E}[\texttt{Var}[\phi_i|W_i]] \\
&= P[W_i = 0]\texttt{Var}[\phi_i|W_i = 0] + P[W_i = 1]\texttt{Var}[\phi_i|W_i = 1] && \text{(B.37)} \\
&= q_i\texttt{Var}[\phi_i|\phi_i = 0] + (1 - q_i)\texttt{Var}[\phi_i|\phi_i \neq 0] && \text{(B.38)} \\
&= (1 - q_i)\texttt{Var}[\phi_i|\phi_i \neq 0] && \text{(B.39)} \\
&\leq (1 - q_i)r^2 && \text{(B.40)}
\end{aligned}$$

where the last inequality follows from the fact that if a random variable is in the range $[m, M]$, then its variance is bounded by $\frac{(M-m)^2}{4}$.

The second term can be expressed as

$$\begin{aligned}
&\texttt{Var}[\mathbb{E}[\phi_i|W_i]] \\
&= \mathbb{E}_{W_i}[(\mathbb{E}[\phi_i|W_i] - \mathbb{E}[\phi_i])^2] && \text{(B.41)} \\
&= P[W_i = 0](\mathbb{E}[\phi_i|W_i = 0] - \mathbb{E}[\phi_i])^2 \\
&\quad + P[W_i = 1](\mathbb{E}[\phi_i|W_i = 1] - \mathbb{E}[\phi_i])^2 && \text{(B.42)} \\
&= q_i(\mathbb{E}[\phi_i|\phi_i = 0] - \mathbb{E}[\phi_i])^2 + (1 - q_i)(\mathbb{E}[\phi_i|\phi_i \neq 0] - \mathbb{E}[\phi_i])^2 && \text{(B.43)} \\
&= q_i(\mathbb{E}[\phi_i])^2 + (1 - q_i)(\mathbb{E}[\phi_i|\phi_i \neq 0] - \mathbb{E}[\phi_i])^2 && \text{(B.44)}
\end{aligned}$$

Note that

$$\mathbb{E}[\phi_i] = P[W_i = 0]\mathbb{E}[\phi_i|\phi_i = 0] + P[W_i = 1]\mathbb{E}[\phi_i|\phi_i \neq 0] \tag{B.45}$$

$$= (1 - q_i)\mathbb{E}[\phi_i|\phi_i \neq 0] \tag{B.46}$$

Plugging (B.46) into (B.41), we obtain

$$\texttt{Var}[\mathbb{E}[\phi_i|W]] = (q_i(1-q_i)^2 + q_i^2(1-q_i))(\mathbb{E}[\phi_i|\phi_i \neq 0])^2 \tag{B.47}$$

Since $|\phi_i| \leq r$, $(\mathbb{E}[\phi_i|\phi_i \neq 0])^2 \leq r^2$. Therefore,

$$\texttt{Var}[\mathbb{E}[\phi_i|W]] \leq q_i(1-q_i)r^2 \tag{B.48}$$

It follows that

$$\texttt{Var}[\phi_i] \leq (1 - q_i^2)r^2 \tag{B.49}$$

Therefore, we can upper bound the variance of ϕ_i in terms of the probability that $\phi_=0$. Now, let us compute $P[\phi_i = 0]$ for $i = 1, \ldots, N$.

Without loss of generality, we assume that x_i are sorted according to their distance to the test point x_{test} in an ascending order.

When $i \leq K$, then whatever place x_i appears in the permutation π, adding x_i to the set of points preceding i in the permutation will always potentially lead to a non-zero utility change. Therefore, we know that $q_i \geq 0$ and

$$\texttt{Var}[\phi_i] \leq r^2 \equiv \sigma_i^2 \text{ for } i = 1, \ldots, K \tag{B.50}$$

When $i \geq K + 1$, adding x_i to P_i^ϕ may lead to zero utility change. More specifically, if there are no less than K elements in $\{x_1, \ldots, x_{i-1}\}$ appearing in P_i^ϕ, then adding i would not change the K nearest neighbors of P_i^ϕ and thus ϕ_i. Let the position of x_i in the permutation pi be denoted by k. Note that if there are at least K elements in $\{x_1, \ldots, x_{i-1}\}$ appearing before x_i in the permutation, then x_i must at least locate in order $K + 1$ in the permutation, i.e., $k \geq K + 1$.

The number of permutations such that x_i is in the kth slot and there are at least K elements appearing before x_i is

$$\sum_{m=K}^{\min\{i-1,k-1\}} \binom{k-1}{m}\binom{N-k}{i-1-m}(i-1)!(N-i)! \tag{B.51}$$

Thus, the probability that ϕ_i is zero is lower bounded by

$$q_i^* = \frac{\sum_{k=K+1}^{N}\sum_{m=K}^{\min\{i-1,k-1\}}\binom{k-1}{m}\binom{N-k}{i-1-m}(i-1)!(N-i)!}{N!} \tag{B.52}$$

$$= \frac{\sum_{k=K+1}^{N}\sum_{m=K}^{\min\{i-1,k-1\}}\binom{k-1}{m}\binom{N-k}{i-1-m}}{\binom{N-1}{i-1}N} \tag{B.53}$$

$$= \frac{i-K}{i} \tag{B.54}$$

By (B.49), we have

$$\texttt{Var}[\phi_i] \leq (1-q_i^{*2})r^2 \text{ for } i = K+1,\dots,N \tag{B.55}$$

By Bennett's inequality, we can bound the approximation error associated with $\hat{s}_i$ by

$$P[|\hat{s}_i - s_i| > \epsilon] \leq 2\exp(-\frac{T\sigma_i^2}{r^2}h(\frac{r\epsilon}{\sigma_i^2})) \tag{B.56}$$

By the union bound, if $P[|\hat{s}_i - s_i| > \epsilon] \leq \delta_i$ for all $i = 1,\dots,N$ and $\sum_{i=1}^N \delta_i = \delta$, then we have

$$P[\max_i |\hat{s}_i - s_i| > \epsilon] = P[\cup_{i=1,\dots,N]\{|\hat{s}_i - s_i}| > \epsilon\}] \leq \sum_{i=1}^N P[|\hat{s}_i - s_i| > \epsilon]$$
$$\leq \sum_{i=1}^N \delta_i = \delta \tag{B.57}$$

Thus, to ensure that $P[\max_i |\hat{s}_i - s_i| > \epsilon] \leq \delta$, we only need to choose T such that

$$2\exp(-\frac{T\sigma_i^2}{r^2}h(\frac{r\epsilon}{\sigma_i^2})) \leq \delta_i \tag{B.58}$$

which yields

$$T \geq \frac{r^2}{\sigma_i^2 h(\frac{r\epsilon}{\sigma_i^2})}\log\frac{2}{\delta_i} \tag{B.59}$$

Since

$$\frac{r^2}{\sigma_i^2 h(\frac{r\epsilon}{\sigma_i^2})} \leq \frac{1}{(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})} \tag{B.60}$$

it suffices to let

$$T \geq \frac{\log\frac{2}{\delta_i}}{(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})} \tag{B.61}$$

for all $i = 1, \dots, N$. Therefore, we would like to choose $\{\delta_i\}_{i=1}^N$ such that $\max_{i=1,\dots,N} T_i^*$ is minimized. We can do this by letting

$$\frac{\log\frac{2}{\delta_i}}{(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})} = T^* \tag{B.62}$$

which gives us

$$\delta_i = 2\exp(-T^*(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})) \tag{B.63}$$

Since $\sum_{i=1}^N \delta_i = \delta$, we get

$$\sum_{i=1}^N \exp(-T^*(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})) = \delta/2 \tag{B.64}$$

and the value of T^* can be solved numerically.

□

B.6 Derivation for the Improved MC Approximation

Because $\log(1+u) > \frac{2x}{2+x}$ (Topsok, 2006), we have $h(u) > \frac{x^2}{2+x}$. Thus, $(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r})) > \frac{\epsilon^2}{2(1-q_i^2)r+\epsilon r}$. Furthermore, by the definition of q_i, $(1-q_i)^2 = 1$ for $i = 1, \ldots, K$ and decreases approximately with the speed $2K/i$ otherwise. Thus, the lower bound of $(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r}))$ increases linearly with i when $i \geq K+1$. Letting $x = \exp(-T^*)$, we can rewrite (4.47) as $\sum_{i=1}^N x^{(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r}))} = \delta/2$. In light of the above analysis, $x^{(1-q_i^2)h(\frac{\epsilon}{(1-q_i^2)r}))}$ will have significant values when $i \geq K$ and is comparatively negligible otherwise. Therefore, we can derive an approximate solution $\tilde{T}$ to T^* by solving the following equation

$$K \exp(-\tilde{T}h(\frac{\epsilon}{r})) = \delta/2. \tag{B.65}$$

which gives us

$$\tilde{T} = \frac{1}{h(\epsilon/r)} \log \frac{2K}{\delta} \tag{B.66}$$

Due to the inequality $h(u) \leq u^2$, we can obtain the following lower bound on $\tilde{T}$:

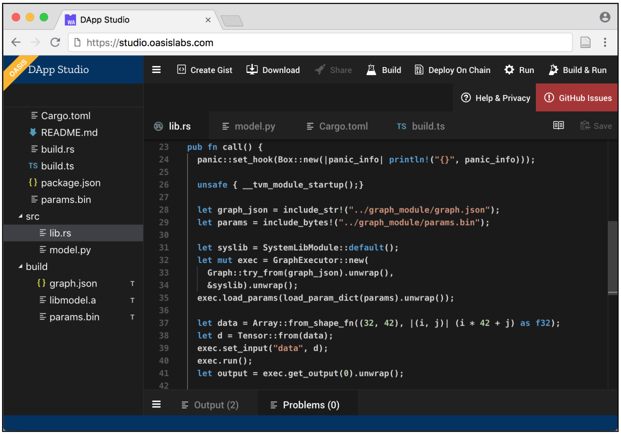
$$\tilde{T} \geq \frac{r^2}{\epsilon^2} \log \frac{2K}{\delta} \tag{B.67}$$

C APPENDIX: CHAPTER 5

C.1 Sterling Runtime

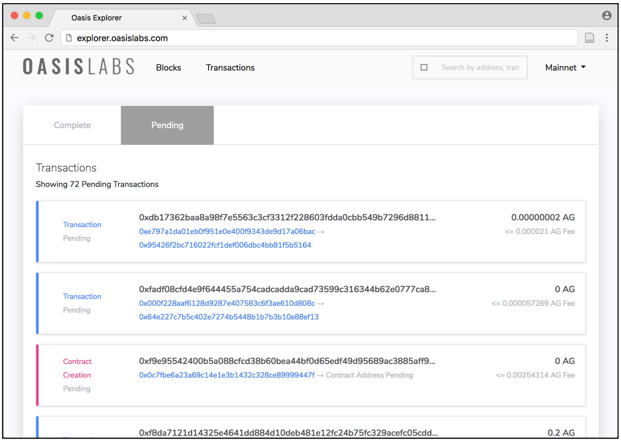
		Time (ms/batch)	
Context	Threads	Training	Inference
CPU (Hynes, Cheng, and D. Song, 2018)	1	28.15	16.20
	8	12.565	2.84
Sterling (ours)	1	38.19	16.27
GPU	24k	3.72	0.19

Table C.1: Performance of running a deep neural network on the CPU, GPU, and Sterling (i.e. WASM in SGX) execution contexts. Comparing single-threaded CPU to Sterling reveals that privacy-preservation introduces minimal overhead.



```
pub fn call() {
  panic::set_hook(Box::new(|panic_info| println!("{}", panic_info)));

  unsafe { __tvm_module_startup();}

  let graph_json = include_str!("../graph_module/graph.json");
  let params = include_bytes!("../graph_module/params.bin");

  let syslib = SystemLibModule::default();
  let mut exec = GraphExecutor::new(
    Graph::try_from(graph_json).unwrap(),
    &syslib).unwrap();
  exec.load_params(load_param_dict(params).unwrap());

  let data = Array::from_shape_fn((32, 42), |(i, j)| (i * 42 + j) as f32);
  let d = Tensor::from(data);
  exec.set_input("data", d);
  exec.run();
  let output = exec.get_output(0).unwrap();

```

FIGURE C.1: Web IDE for creating and editing smart contracts. From here, data providers can specify precise constraints on the use of their data; and data consumers can design machine learning models which use the data.



Transactions

Showing 72 Pending Transactions

Transaction Pending | 0xdb17362baa8a98f7e5563c3cf3312f228603fdda0cbb549b7296d8811... | 0.00000002 AG
0xe797a1da01eb0f951e0e400f9343de9d17a06bac → | <= 0.000021 AG Fee
0x95426f2bc716022fcf1def006dbc4bb81f5b5164

Transaction Pending | 0xfadf08cfd4e9f644455a754cadcadda9cad73599c316344b62e0777ca8... | 0 AG
0x000f228aaf6128d9287e407583c6f3ae610d808c → | <= 0.000057269 AG Fee
0x64e227c7b5c402e7274b5448b1b7b3b10e88ef13

Contract Creation Pending | 0xf9e95542400b5a088cfcd38b60bea44bf0d65edf49d95689ac3885aff9... | 0 AG
0x0c7fbe6a23a69c14e1e3b1432c328ce89999447f → Contract Address Pending | <= 0.00254314 AG Fee

0xf8da7121d14325e4641dd884d10deb481e12fc24b75fc329acefc05cdd... | 0.2 AG

FIGURE C.2: Blockchain explorer which provides visibility into Sterling transactions. Here, we see two requests for data and the creation of a new provider contract.

BIBLIOGRAPHY

Ballestar, Marıa Teresa, Pilar Grau-Carles, and Jorge Sainz (2019). “Predicting customer quality in e-commerce social networks: a machine learning approach”. In: *Review of Managerial Science* 13, 589.

Rao, Susie Xi, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang (2020). “xFraud: Explainable Fraud Transaction Detection on Heterogeneous Graphs”. In: *CoRR* abs/2011.12193.

Lacerda, Anisio, Marco Cristo, Marcos André Gonçalves, Weiguo Fan, Nivio Ziviani, and Berthier Ribeiro-Neto (2006). “Learning to advertise”. In: *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 549.

Perlich, Claudia, Brian Dalessandro, Troy Raeder, Ori Stitelman, and Foster Provost (2014). “Machine learning for targeted display advertising: Transfer learning in action”. In: *Machine learning* 95.1, 103.

Rafieian, Omid and Hema Yoganarasimhan (2021). “Targeting and privacy in mobile advertising”. In: *Marketing Science* 40.2, 193.

Carpenter, Anne E, Thouis R Jones, Michael R Lamprecht, Colin Clarke, In Han Kang, Ola Friman, David A Guertin, Joo Han Chang, Robert A Lindquist, Jason Moffat, et al. (2006). “CellProfiler: image analysis software for identifying and quantifying cell phenotypes”. In: *Genome biology* 7, 1.

Bray, Mark-Anthony, Shantanu Singh, Han Han, Chadwick T Davis, Blake Borgeson, Cathy Hartland, Maria Kost-Alimova, Sigrun M Gustafsdottir, Christopher C Gibson, and Anne E Carpenter (2016). “Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes”. In: *Nature protocols* 11.9, 1757.

McQuin, Claire, Allen Goodman, Vasiliy Chernyshev, Lee Kamentsky, Beth A Cimini, Kyle W Karhohs, Minh Doan, Liya Ding, Susanne M Rafelski, Derek Thirstrup, et al. (2018). "CellProfiler 3.0: Next-generation image processing for biology". In: *PLoS biology* 16.7, e2005970.

Bunne, Charlotte, Stefan G Stark, Gabriele Gut, Jacobo Sarabia del Castillo, Kjong-Van Lehmann, Lucas Pelkmans, Andreas Krause, and Gunnar Rätsch (2021). "Learning single-cell perturbation responses using neural optimal transport". In: *bioRxiv*, 2021.

Rolf, Esther, Jonathan Proctor, Tamma Carleton, Ian Bolliger, Vaishaal Shankar, Miyabi Ishihara, Benjamin Recht, and Solomon Hsiang (2021). "A generalizable and accessible approach to machine learning with global satellite imagery". In: *Nature communications* 12.1, 4392.

Rolnick, David, Priya L Donti, Lynn H Kaack, Kelly Kochanski, Alexandre Lacoste, Kris Sankaran, Andrew Slavin Ross, Nikola Milojevic-Dupont, Natasha Jaques, Anna Waldman-Brown, et al. (2022). "Tackling climate change with machine learning". In: *ACM Computing Surveys (CSUR)* 55.2, 1.

Beery, Sara, Guanhang Wu, Trevor Edwards, Filip Pavetic, Bo Majewski, Shreyasee Mukherjee, Stanley Chan, John Morgan, Vivek Rathod, and Jonathan Huang (2022). "The Auto Arborist Dataset: A Large-Scale Benchmark for Multiview Urban Forest Monitoring Under Domain Shift". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 21294.

Huval, Brody, Tao Wang, Sameep Tandon, Jeff Kiske, Will Song, Joel Pazhayampallil, Mykhaylo Andriluka, Pranav Rajpurkar, Toki Migimatsu, Royce Cheng-Yue, et al. (2015). "An empirical evaluation of deep learning on highway driving". In: *arXiv preprint arXiv:1504.01716*.

Bojarski, Mariusz, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. (2016). "End to end learning for self-driving cars". In: *arXiv preprint arXiv:1604.07316*.

Grigorescu, Sorin, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu (2020). "A survey of deep learning techniques for autonomous driving". In: *Journal of Field Robotics* 37.3, 362.

Chancellor, Stevie, Eric PS Baumer, and Munmun De Choudhury (2019). "Who is the" human" in human-centered machine learning: The case of predicting mental health from social media". In: *Proceedings of the ACM on Human-Computer Interaction* 3.CSCW, 1.

Delacroix, Sylvie and Neil D Lawrence (2019). "Bottom-up data Trusts: disturbing the 'one size fits all'approach to data governance". In: *International data privacy law* 9.4, 236.

Richter, H. and P. R. Slowinski (2019). "The Data Sharing Economy: On the Emergence of New Intermediaries". In: *IIC-International Review of Intellectual Property and Competition Law* 50.1, 4.

McConaghy, Trent (2022). "Ocean Protocol: Tools for the Web3 Data Economy". In: *Handbook on Blockchain*. Ed. by Duc A. Tran, My T. Thai, and Bhaskar Krishnamachari. Springer Optimization and Its Applications. Springer, 505.

Scaria, E., A. Berghmans, M. Pont, C. Arnaut, and S. Leconte (2018). *Study on data sharing between companies in Europe*. Tech. rep. EU Publications.

Lundberg, Scott M and Su-In Lee (2017). "A unified approach to interpreting model predictions". In: *Advances in Neural Information Processing Systems*, 4768.

Simonyan, Karen and Andrew Zisserman (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. cite arxiv:1409.1556.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2017). "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6, 84.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016a). “Rethinking the Inception Architecture for Computer Vision”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei (2015). “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision* 115.3, 211.

Tan, Mingxing, Ruoming Pang, and Quoc V. Le (2020). “EfficientDet: Scalable and Efficient Object Detection”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10778.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *International Conference on Learning Representations*.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *Nature* 521.7553, 436.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). “Attention Is All You Need”. In: *CoRR* abs/1706.03762.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. (2021). “On the Opportunities and Risks of Foundation Models”. In: *CoRR* abs/2108.07258.

OpenAI (2023). *GPT-4 Technical Report*.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei (2020). “Language Models are Few-Shot Learners”. In: *CoRR* abs/2005.14165.

Hinton, Geoffrey, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury (2012). “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups”. In: *IEEE Signal Processing Magazine* 29.6, 82.

Amodei, Dario, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu (2015). *Deep Speech 2: End-to-End Speech Recognition in English and Mandarin*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu (2020). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller (2013). "Playing Atari with Deep Reinforcement Learning". In: *CoRR* abs/1312.5602.

Horgan, Dan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver (2018). "Distributed Prioritized Experience Replay". In: *CoRR* abs/1803.00933.

Kapturowski, Steven, Georg Ostrovski, Will Dabney, John Quan, and Remi Munos (2019). "Recurrent Experience Replay in Distributed Reinforcement Learning". In: *International Conference on Learning Representations*.

Schrittwieser, Julian, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver (2019). "Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model". In: *CoRR* abs/1911.08265.

Mitchell, Tom M (1997). *Machine learning*. Vol. 1. 9. McGraw-hill New York.

Neumann, J. von and O. Morgenstern (1947). *Theory of games and economic behavior*. Princeton University Press.

Nash, J.F. (1951). "Non-cooperative Games". In: *Annals of Mathematics* 54.2, 286.

Shapley, Lloyd S (1953). "A value for n-person games". In: *Contributions to the Theory of Games* 2.28, 307.

Gul, Faruk (1989). "Bargaining foundations of Shapley value". In: *Econometrica: Journal of the Econometric Society*, 81.

Michalak, Tomasz, Talal Rahwan, Piotr L Szczepanski, Oscar Skibski, Ramasuri Narayanam, Michael Wooldridge, and Nicholas R Jennings (2013). "Computational analysis of connectivity games with applications to the investigation of terrorist networks". In.

Lindelauf, RHA, HJM Hamers, and BGM Husslage (2013). “Cooperative game theoretic centrality analysis of terrorist networks: The cases of jemaah islamiyah and al qaeda”. In: *European Journal of Operational Research* 229.1, 230.

Petrosjan, Leon and Georges Zaccour (2003). “Time-consistent Shapley value allocation of pollution cost reduction”. In: *Journal of economic dynamics and control* 27.3, 381.

Cohen, Shay, Eytan Ruppin, and Gideon Dror (2005). “Feature selection based on the Shapley value”. In: *In other words* 1, 98Eqr.

Young, H Peyton (1985). “Monotonic solutions of cooperative games”. In: *International Journal of Game Theory* 14.2, 65.

Sim, Rachael Hwee Ling, Xinyi Xu, and Bryan Kian Hsiang Low (2022). “Data Valuation in Machine Learning: "Ingredients", Strategies, and Open Challenges”. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. Thirty-First International Joint Conference on Artificial Intelligence {IJCAI-22}. Vienna, Austria: International Joint Conferences on Artificial Intelligence Organization, 5607.

Jia, Ruoxi, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gurel, Bo Li, Ce Zhang, Dawn Song, and Costas Spanos (2019). “Towards Efficient Data Valuation Based on the Shapley Value”.

Ghorbani, Amirata and James Zou (2019). “Data Shapley: Equitable Valuation of Data for Machine Learning”. In: *arXiv preprint arXiv:1904.02868*, 10.

Frye, Christopher, Colin Rowat, and Ilya Feige (2020). “Asymmetric Shapley Values: Incorporating Causal Knowledge into Model-Agnostic Explainability”.

Ohrimenko, Olga, Shruti Tople, and Sebastian Tschiatschek (2019). *Collaborative Machine Learning Markets with Data-Replication-Robust Payments*. URL: http://arxiv.org/abs/1911.09052 (visited on 03/07/2023). preprint.

Ghorbani, Amirata, Michael P. Kim, and James Zou (2020). *A Distributional Framework for Data Valuation*. URL: `http://arxiv.org/abs/2002.12334` (visited on 10/10/2022). preprint.

Jia, Ruoxi, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas J. Spanos, and Dawn Song (2020). *Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms*. URL: `http://arxiv.org/abs/1908.08619` (visited on 10/10/2022). preprint.

Sim, Rachael Hwee Ling, Yehong Zhang, Mun Choon Chan, and Bryan Kian Hsiang Low (2020). *Collaborative Machine Learning with Incentive-Aware Model Rewards*. URL: `http://arxiv.org/abs/2010.12797` (visited on 10/10/2022). preprint.

Okhrati, Ramin and Aldo Lipani (2021). "A Multilinear Sampling Algorithm to Estimate Shapley Values". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2020 25th International Conference on Pattern Recognition (ICPR). Milan, Italy: IEEE, 7992.

Yan, Tom and Ariel D. Procaccia (2021). "If You Like Shapley Then You'll Love the Core". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.6, 5751.

Xu, Xinyi, Lingjuan Lyu, Xingjun Ma, Chenglin Miao, Chuan Sheng Foo, and Bryan Kian Hsiang Low (2021). "Gradient-Driven Rewards to Guarantee Fairness in Collaborative Machine Learning". In: 14.

Kwon, Yongchan and James Zou (2022). *Beta Shapley: A Unified and Noise-reduced Data Valuation Framework for Machine Learning*. URL: `http://arxiv.org/abs/2110.14049` (visited on 10/10/2022). preprint.

Schoch, Stephanie, Haifeng Xu, and Yangfeng Ji (2022). *CS-Shapley: Class-wise Shapley Values for Data Valuation in Classification*. URL: `http://arxiv.org/abs/2211.06800` (visited on 03/07/2023). preprint.

Wang, Jiachen T. and Ruoxi Jia (2023a). *Data Banzhaf: A Robust Data Valuation Framework for Machine Learning*. URL: `http://arxiv.org/abs/2205.15466` (visited on 03/07/2023). preprint.

Koh, Pang Wei and Percy Liang (2017). “Understanding Black-box Predictions via Influence Functions”. In: *International Conference on Machine Learning*, 1885.

Yoon, Jinsung, Sercan Ö Arık, and Tomas Pfister (2019). “Data Valuation Using Reinforcement Learning”.

Wu, Zhaoxuan, Yao Shu, and Bryan Kian Hsiang Low (2022). “DAVINZ: Data Valuation Using Deep Neural Networks at Initialization”. In: 27.

Xu, Xinyi, Zhaoxuan Wu, Chuan Sheng Foo, and Bryan Kian Hsiang Low (2021). “Validation Free and Replication Robust Volume-based Data Valuation”. In: 12.

Just, Hoang Anh, Feiyang Kang, Jiachen T Wang, Yi Zeng, Myeongseob Ko, Ming Jin, and Ruoxi Jia (2023). “LAVA: Data Valuation without Pre-Specified Learning Algorithms”. In.

Bian, Yatao, Yu Rong, Tingyang Xu, Jiaxiang Wu, Andreas Krause, and Junzhou Huang (2022). “Energy-Based Learning for Cooperative Games, with Applications to Valuation Problems in Machine Learning”.

Jia, Ruoxi, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song (2021). *Scalability vs. Utility: Do We Have to Sacrifice One for the Other in Data Importance Quantification?* URL: http://arxiv.org/abs/1911.07128 (visited on 10/10/2022). preprint.

Koh, Pang Wei, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang (2019). *On the Accuracy of Influence Functions for Measuring Group Effects*. URL: http://arxiv.org/abs/1905.13289 (visited on 10/10/2022). preprint.

Basu, Samyadeep, Phillip Pope, and Soheil Feizi (2021). “Influence Functions in Deep Learning Are Fragile”.

Kwon, Yongchan, Manuel A. Rivas, and James Zou (2021). *Efficient Computation and Analysis of Distributional Shapley Values*. URL: http://arxiv.org/abs/2007.01357 (visited on 10/10/2022). preprint.

Castro, Javier, Daniel Gómez, and Juan Tejada (2009). "Polynomial Calculation of the Shapley Value Based on Sampling". In: *Computers & Operations Research* 36.5, 1726.

Wang, Tianhao, Yu Yang, and Ruoxi Jia (2022). *Improving Cooperative Game Theory-based Data Valuation via Data Utility Learning*. URL: `http://arxiv.org/abs/2107.06336` (visited on 10/12/2022). preprint.

Koutris, Paraschos, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu (2015). "Query-based data pricing". In: *Journal of the ACM (JACM)* 62.5, 43.

– (2013). "Toward practical query pricing with QueryMarket". In: *proceedings of the 2013 ACM SIGMOD international conference on management of data*. ACM, 613.

– (2012). "Querymarket demonstration: Pricing for online data markets". In: *PVLDB* 5.12, 1962.

Deep, Shaleen, Paraschos Koutris, and Yash Bidasaria (2017). "QIRANA demonstration: real time scalable query pricing". In: *PVLDB* 10.12, 1949.

Lin, Bing-Rong and Daniel Kifer (2014). "On arbitrage-free pricing for general data queries". In: *PVLDB* 7.9, 757.

Li, Chao and Gerome Miklau (2012). "Pricing Aggregate Queries in a Data Marketplace." In: *WebDB*, 19.

Upadhyaya, Prasang, Magdalena Balazinska, and Dan Suciu (2016). "Price-optimal querying with data apis". In: *PVLDB* 9.14, 1695.

Zheng, Zhenzhe, Yanqing Peng, Fan Wu, Shaojie Tang, and Guihai Chen (2019). "ARETE: On Designing Joint Online Pricing and Reward Sharing Mechanisms for Mobile Data Markets". In: *IEEE Transactions on Mobile Computing*.

Li, Chao, Daniel Yang Li, Gerome Miklau, and Dan Suciu (2017). "A theory of pricing private data". In: *Communications of the ACM* 60.12, 79.

Chen, Lingjiao, Paraschos Koutris, and Arun Kumar (2018). “Model-based Pricing for Machine Learning in a Data Marketplace”. In: *arXiv preprint arXiv:1805.11450*.

– (2017). “Model-based Pricing: Do Not Pay for More than What You Learn!” In: *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning*. ACM, 1.

Raskar, Ramesh, Praneeth Vepakomma, Tristan Swedish, and Aalekh Sharan (2019). “Data Markets to support AI for All: Pricing, Valuation and Governance”. In: *arXiv preprint arXiv:1905.06462*.

Deng, Xiaotie and Christos H Papadimitriou (1994). “On the complexity of cooperative solution concepts”. In: *Mathematics of Operations Research* 19.2, 257.

Maleki, Sasan, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers (2013). “Bounding the estimation error of sampling-based Shapley value approximation”. In: *arXiv preprint arXiv:1306.4265*.

Bachrach, Yoram, Evangelos Markakis, Ariel D Procaccia, Jeffrey S Rosenschein, and Amin Saberi (2008). “Approximating power indices”. In: *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems, 943.

Fatima, Shaheen S, Michael Wooldridge, and Nicholas R Jennings (2008). “A linear approximation method for the Shapley value”. In: *Artificial Intelligence* 172.14, 1673.

Michalak, Tomasz P, Karthik V Aadithya, Piotr L Szczepanski, Balaraman Ravindran, and Nicholas R Jennings (2013). “Efficient computation of the Shapley value for game-theoretic network centrality”. In: *Journal of Artificial Intelligence Research* 46, 607.

Chessa, Michela and Patrick Loiseau (2017). “A cooperative game-theoretic approach to quantify the value of personal data in networks”. In: *Proceedings of the 12th workshop on the Economics of Networks, Systems and Computation*. ACM, 9.

Kleinberg, Jon, Christos H Papadimitriou, and Prabhakar Raghavan (2001). "On the value of private information". In: *Proceedings of the 8th conference on Theoretical aspects of rationality and knowledge*. Morgan Kaufmann Publishers Inc., 249.

Sun, Xin, Yanheng Liu, Jin Li, Jianqi Zhu, Xuejie Liu, and Huiling Chen (2012). "Using cooperative game theory to optimize the feature selection problem". In: *Neurocomputing* 97, 86.

Mokdad, Fatiha, Djamel Bouchaffra, Nabil Zerrouki, and Azzedine Touazi (2015). "Determination of an optimal feature selection method based on maximum Shapley value". In: *Intelligent Systems Design and Applications (ISDA), 2015 15th International Conference on*. IEEE, 116.

Sasikala, S, S Appavu alias Balamurugan, and S Geetha (2015). "A novel feature selection technique for improved survivability diagnosis of breast cancer". In: *Procedia Computer Science* 50, 16.

Sharchilev, Boris, Yury Ustinovsky, Pavel Serdyukov, and Maarten de Rijke (2018). "Finding Influential Training Samples for Gradient Boosted Decision Trees". In: *arXiv preprint arXiv:1802.06640*.

Ogawa, Kohei, Yoshiki Suzuki, and Ichiro Takeuchi (2013). "Safe screening of non-support vectors in pathwise SVM computation". In: *International Conference on Machine Learning*, 1382.

Dasgupta, Anirban, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W Mahoney (2009). "Sampling algorithms and coresets for \ell_p regression". In: *SIAM Journal on Computing* 38.5, 2060.

Fryer, Daniel, Inga Strümke, and Hien Nguyen (2021). *Shapley Values for Feature Selection: The Good, the Bad, and the Axioms*. URL: `http://arxiv.org/abs/2102.10936` (visited on 10/10/2022). preprint.

Claver, H. (2019). *Data sharing key for AI in agriculture*. Future Farming.

Conway, J. (2018). "Artificial Intelligence and Machine Learning: Current Applications in Real Estate". PhD thesis. Massachusetts Institute of Technology.

Center for Open Data Enterprise (2019). *Sharing and Utilizing Health Data for AI Applications*. Roundtable Report.

Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer (2021). *High-Resolution Image Synthesis with Latent Diffusion Models*.

Schuhmann, Christoph, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev (2022). *LAION-5B: An open large-scale dataset for training next generation image-text models*.

Du, Dingzhu, Frank K Hwang, and Frank Hwang (2000). *Combinatorial group testing and its applications*. Vol. 12. World Scientific.

Bousquet, Olivier and André Elisseeff (2002). "Stability and generalization". In: *Journal of machine learning research* 2.Mar, 499.

Maleki, Sasan (2015). "Addressing the computational issues of the Shapley value with applications in the smart grid". PhD thesis. University of Southampton.

Zhou, Yingbo, Utkarsh Porwal, Ce Zhang, Hung Q Ngo, XuanLong Nguyen, Christopher Ré, and Venu Govindaraju (2014). "Parallel feature selection inspired by group testing". In: *Advances in Neural Information Processing Systems*, 3554.

Wang, Jiachen T. and Ruoxi Jia (2023b). *A Note on "Towards Efficient Data Valuation Based on the Shapley Value"*.

Rauhut, Holger (2010). "Compressive sensing and structured random matrices". In: *Theoretical foundations and numerical methods for sparse recovery* 9, 1.

Candes, Emmanuel J, Justin K Romberg, and Terence Tao (2006). "Stable signal recovery from incomplete and inaccurate measurements". In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 59.8, 1207.

Dwork, Cynthia (2008). "Differential privacy: A survey of results". In: *International Conference on Theory and Applications of Models of Computation*. Springer, 1.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy (2014). "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572*.

Carlini, Nicholas and David Wagner (2017). "Towards evaluating the robustness of neural networks". In: *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 39.

Zemel, Rich, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork (2013). "Learning fair representations". In: *International Conference on Machine Learning*, 325.

Woodworth, Blake, Suriya Gunasekar, Mesrob I Ohannessian, and Nathan Srebro (2017). "Learning non-discriminatory predictors". In: *arXiv preprint arXiv:1702.06081*.

Hardt, Moritz, Eric Price, Nati Srebro, et al. (2016). "Equality of opportunity in supervised learning". In: *Advances in neural information processing systems*, 3315.

Dudani, Sahibsingh A (1976). "The distance-weighted k-nearest-neighbor rule". In: *IEEE Transactions on Systems, Man, and Cybernetics* 4, 325.

Hays, James and Alexei A Efros (2015). "Large-scale image geolocalization". In: *Multimodal Location Estimation of Videos and Images*. Springer, 41.

Adeniyi, DA, Z Wei, and Y Yongquan (2016). "Automated web usage data mining and recommendation system using K-Nearest Neighbor (KNN) classification method". In: *Applied Computing and Informatics* 12.1, 90.

Li, Chao, Shuheng Zhang, Huan Zhang, Lifang Pang, Kinman Lam, Chun Hui, and Su Zhang (2012). "Using the K-nearest neighbor algorithm for the classification of lymph node metastasis in gastric cancer". In: *Computational and mathematical methods in medicine* 2012.

Charikar, Moses S (2002). “Similarity estimation techniques from rounding algorithms”. In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, 380.

Datar, Mayur, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni (2004). “Locality-sensitive hashing scheme based on p-stable distributions”. In: *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 253.

He, Junfeng, Sanjiv Kumar, and Shih-Fu Chang (2012). “On the difficulty of nearest neighbor search”. In: *Proceedings of the 29th International Coference on International Conference on Machine Learning*. Omnipress, 41.

Mount, David M and Sunil Arya (1998). “ANN: library for approximate nearest neighbour searching”. In.

Gionis, Aristides, Piotr Indyk, Rajeev Motwani, et al. (1999). “Similarity search in high dimensions via hashing”. In: *Vldb*. Vol. 99. 6, 518.

Har-Peled, Sariel, Piotr Indyk, and Rajeev Motwani (2012). “Approximate nearest neighbor: Towards removing the curse of dimensionality”. In: *Theory of computing* 8.1, 321.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016b). “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818.

LeCun, Yann and Corinna Cortes (2010). “MNIST handwritten digit database”. In.

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*.

Amato, Giuseppe, Fabrizio Falchi, Claudio Gennaro, and Fausto Rabitti (2016). “YFCC100M-HNfc6: a large-scale deep features benchmark for similarity search”. In: *International Conference on Similarity Search and Applications*. Springer, 196.

Siagian, Christian and Laurent Itti (2007). “Rapid biologically-inspired scene classification using features shared with visual attention”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.2, 300.

Azcoitia, Santiago Andrés and Nikolaos Laoutaris (2022). *A Survey of Data Marketplaces and Their Business Models*.

Anati, Ittai, Shay Gueron, Simon Johnson, and Vincent Scarlata (2013). “Innovative technology for CPU based attestation and sealing”. In: *HASP*. Vol. 13.

Lebedev, Ilia A., Kyle Hogan, and Srinivas Devadas (2018). “Secure Boot and Remote Attestation in the Sanctum Processor”. In: *IACR*, 427.

Lee, Dayeol, David Kohlbrenner, Shweta Shinde, Krste Asanović, and Dawn Song (2020). “Keystone: An open framework for architecting trusted execution environments”. In: *Proceedings of the Fifteenth European Conference on Computer Systems*, 1.

Rajkomar, Alvin, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Peter J Liu, Xiaobing Liu, Mimi Sun, Patrik Sundberg, Hector Yee, et al. (2018). “Scalable and accurate deep learning for electronic health records”. In: *arXiv:1801.07860*.

Cheng, Raymond, Fan Zhang, Jernej Kos, Warren He, Nicholas Hynes, Noah M. Johnson, Ari Juels, Andrew Miller, and Dawn Song (2018). “Ekiden: A Platform for Confidentiality-Preserving, Trustworthy, and Performant Smart Contract Execution”. In: *arXiv:1804.05141*.

Dwork, Cynthia, Guy N. Rothblum, and Salil P. Vadhan (2010). “Boosting and Differential Privacy”. In: *FOCS*, 51.

Abadi, Martın, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang (2016). “Deep Learning with Differential Privacy”. In: *CCS*, 308.

Near, Joe, Lun Wang, Noah Johnson, Mu Zhang, and Dawn Song (2018). “Optio: Differential Privacy for Machine Learning Pipelines, Statically and Automatically”.

Hynes, Nick, Raymond Cheng, and Dawn Song (2018). "Efficient Deep Learning on Multi-Source Private Data". In: *arXiv 1801.07860*.

Ohrimenko, Olga, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa (2016). "Oblivious Multi-Party Machine Learning on Trusted Processors". In: *USENIX Security*, 619.

Carlini, Nicholas, Chang Liu, Jernej Kos, Úlfar Erlingsson, and Dawn Song (2018). "The Secret Sharer: Measuring Unintended Neural Network Memorization & Extracting Secrets". In: *arXiv:1802.08232*.

McSherry, Frank (2009). "Privacy integrated queries: an extensible platform for privacy-preserving data analysis". In: *SIGMOD*, 19.

Hsu, Justin, Marco Gaboardi, Andreas Haeberlen, Sanjeev Khanna, Arjun Narayan, Benjamin C. Pierce, and Aaron Roth (2014). "Differential Privacy: An Economic Method for Choosing Epsilon". In: *CSF*, 398.

Fleischer, Lisa and Yu-Han Lyu (2012). "Approximately optimal auctions for selling privacy when costs are correlated with data". In: *EC*, 568.

IPCC (2021). "Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change". In: ed. by V. Masson-Delmotte, P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, K. Leitzell M. Huang, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, O. Yelekçi, R. Yu, and B. Zhou. Cambridge University Press.

Ceballos, G. and P. Ehrlich (2018). "The misunderstood sixth mass extinction". In: *Science* 360, 1080.2.

Exposito-Alonso, Moises, Tom R. Booker, Lucas Czech, Lauren Gillespie, Shannon Hateley, Christopher C. Kyriazis, Patricia L. M. Lang, Laura Leventhal, David Nogues-Bravo, Veronica Pagowski, Megan Ruffley, Jeffrey P. Spence, Sebastian E. Toro Arana, Clemens L. Weiß, and Erin Zess (2022). "Genetic diversity loss in the Anthropocene". In: *Science* 377.6613, 1431.

Hansen, M. C., P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R.G. Townshend (2013). "High-resolution global maps of 21st-century forest cover change". In: *Science* 342.6160, 850.

IPCC (2019). "2019: Summary for Policymakers". In: *Climate Change and Land: an IPCC special report on climate change, desertification, land degradation, sustainable land management, food security, and greenhouse gas fluxes in terrestrial ecosystems*. Ed. by P.R. Shukla, J. Skea, E. Calvo Buendia, V. Masson-Delmotte, H.-O. Pörtner, D. C. Roberts, P. Zhai, R. Slade, S. Connors, R. van Diemen, M. Ferrat, E. Haughey, S. Luz, S. Neogi, M. Pathak, J. Petzold, J. Portugal Pereira, P. Vyas, E. Huntley, K. Kissick, M. Belkacemi, and J. Malley, 7.

Blaufelder, Christopher, Cindy Levy, Peter Mannion, Dickon Pinner, and Jop Weterings (2021). *McKinsey&Co: A Blueprint for Scaling Voluntary Carbon Markets to Meet the Climate Challenge*. Accessed 31.05.2021.

Ecosystem Marketplace (2021). *State of the Voluntary Carbon Markets 2021, `https://www.ecosystemmarketplace.com/publications/state-of-the-voluntary-carbon-markets-2021`*.

Kreibich, Nicolas and Lukas Hermwille (2021). "Caught in between: credibility and feasibility of the voluntary carbon market post-2020". In: *Climate Policy* 21.7, 939.

Ma, Haozhi, Lidong Mo, Daniel S. Maynard Thomas W. Crowther, Johan van den Hoogen, Benjamin D. Stocker, César Terrer, and Constantin M. Zohner (2021). "The global distribution and environmental drivers of aboveground versus belowground plant biomass". In: *Nature Ecology & Evolution* 5 (8), 1110.

Pearson, T., S. Walker, and S. Brown (2005). *Sourcebook for BioCarbon Fund Projects*. Accessed 15.09.2021 URL: https://winrock.org/document/sourcebook-for-land-use-land-use-change-and-forestry-projects/.

Petrokofsky, Gillian, Hideki Kanamaru, Frédéric Achard, Scott J. Goetz, Hans Joosten, Peter Holmgren, Aleksi Lehtonen, Mary C. S. Menton, Andrew S. Pullin, and Martin Wattenbach (2012). “Comparison of methods for measuring and assessing carbon stocks and carbon stock changes in terrestrial carbon pools. How do the accuracy and precision of current methods compare? A systematic review protocol”. In: *Environmental Evidence* 1 (1), 6.

Malhi, Y., O. L. Phillips, Jerome Chave, Richard Condit, Salomon Aguilar, Andres Hernandez, Suzanne Lao, and Rolando Perez (2004). “Error propagation and scaling for tropical forest biomass estimates”. In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 359.1443, 409.

Badgley, Grayson, Jeremy Freeman, Joseph J. Hamman, Barbara Haya, Anna T. Trugman, William R.L. Anderegg, and Danny Cullenward (2021). “Systematic over-crediting in California’s forest carbon offsets program”. In: *bioRxiv*.

White, Alisa E., David A. Lutz, Richard B. Howarth, and José R. Soto (2018). “Small-scale forestry and carbon offset markets: An empirical study of Vermont Current Use forest landowner willingness to accept carbon credit programs”. In: *PLOS ONE* 13.8, 1.

Global Forest Watch (2019). *Aboveground Live Woody Biomass Density*. Dataset Accessed: 30.11.2021.

Schiefer, Felix, Teja Kattenborn, Annett Frick, Julian Frey, Peter Schall, Barbara Koch, and Sebastian Schmidtlein (2020). “Mapping forest tree species in high resolution UAV-based RGB-imagery by means of convolutional neural networks”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 170, 205.

Ganz, Selina, Yannek Käber, and Petra Adler (2019). “Measuring Tree Height with Remote Sensing—A Comparison of Photogrammetric and LiDAR Data with Different Field Measurements”. In: *Forests* 10, 694.

Weinstein, Ben G., Sergio Marconi, Stephanie A. Bohlman, Alina Zare, and Ethan P. White (2020). “Cross-site learning in deep learning RGB tree crown detection”. In: *Ecological Informatics* 56, 101061.

Narine, Lana L., Sorin C. Popescu, and Lonesome Malambo (2020). “Using ICESat-2 to Estimate and Map Forest Aboveground Biomass: A First Example”. In: *Remote Sensing* 12.11.

Dubayah, Ralph, John Armston, Sean P Healey, Jamis M Bruening, Paul L Patterson, James R Kellner, Laura Duncanson, Svetlana Saarela, Göran Ståhl, Zhiqiang Yang, Hao Tang, J Bryan Blair, Lola Fatoyinbo, Scott Goetz, Steven Hancock, Matthew Hansen, Michelle Hofton, George Hurtt, and Scott Luthcke (2022). “GEDI launches a new era of biomass inference from space”. In: *Environmental Research Letters* 17.9, 095001.

Kellner, James R., John Armston, Markus Birrer, K. C. Cushman, Laura Duncanson, Christoph Eck, Christoph Falleger, Benedikt Imbach, Kamil Král, Martin Krůček, Jan Trochta, Tomáš Vrška, and Carlo Zgraggen (2019). “New Opportunities for Forest Remote Sensing Through Ultra-High-Density Drone Lidar”. In: *Surveys in Geophysics* 40.4, 959.

Hanan, Niall P. and Julius Y. Anchang (2020). “Satellites could soon map every tree on Earth”. In: *Nature* 587 (7832).

Saatchi, Sassan S., Nancy L. Harris, Sandra Brown, Michael Lefsky, Edward T. A. Mitchard, William Salas, Brian R. Zutta, Wolfgang Buermann, Simon L. Lewis, Stephen Hagen, Silvia Petrova, Lee White, Miles Silman, and Alexandra Morel (2011). “Benchmark map of forest carbon stocks in tropical regions across three continents”. In: *Proceedings of the National Academy of Sciences* 108.24, 9899.

Santoro, M., O. Cartus, N. Carvalhais, D. M. A. Rozendaal, V. Avitabile, A. Araza, S. de Bruin, M. Herold, S. Quegan, P. Rodrıguez-Veiga, H. Balzter, J. Carreiras, D. Schepaschenko, M. Korets, M. Shimada, T. Itoh, Á. Moreno Martınez, J. Cavlovic, R. Cazzolla Gatti, P. da Conceição Bispo, N. Dewnath, N. Labrière, J. Liang, J. Lindsell, E. T. A. Mitchard, A. Morel, A. M. Pacheco Pascagaza, C. M. Ryan, F. Slik, G. Vaglio Laurin, H. Verbeeck, A. Wijaya, and S. Willcock (2021). “The global forest above-ground biomass pool for 2010 estimated from high-resolution satellite observations”. In: *Earth System Science Data* 13.8, 3927.

West, Thales A. P., Jan Börner, Erin O. Sills, and Andreas Kontoleon (2020). "Overstated carbon emission reductions from voluntary REDD+ projects in the Brazilian Amazon". In: *Proceedings of the National Academy of Sciences* 117.39, 24188.

Silva, Carlos Alberto, Laura Duncanson, Steven Hancock, Amy Neuenschwander, Nathan Thomas, Michelle Hofton, Lola Fatoyinbo, Marc Simard, Charles Z. Marshak, John Armston, Scott Lutchke, and Ralph Dubayah (2021). "Fusing simulated GEDI, ICESat-2 and NISAR data for regional aboveground biomass mapping". In: *Remote Sensing of Environment* 253, 112234.

Haya, Barbara, Danny Cullenward, Aaron L. Strong, Emily Grubert, Robert Heilmayr, Deborah A. Sivas, and Michael Wara (2020). "Managing uncertainty in carbon offsets: insights from California's standardized approach". In: *Climate Policy* 20.9, 1112.

Lima, Renato A. F. de, Oliver L. Phillips, Alvaro Duque, J. Sebastian Tello, Stuart J. Davies, Alexandre Adalardo de Oliveira, Sandra Muller, Euridice N. Honorio Coronado, Emilio Vilanova, Aida Cuni-Sanchez, Timothy R. Baker, Casey M. Ryan, Agustina Malizia, Simon L. Lewis, Hans ter Steege, Joice Ferreira, Beatriz Schwantes Marimon, Hong Truong Luu, Gerard Imani, Luzmila Arroyo, Cecilia Blundo, David Kenfack, Moses N. Sainge, Bonaventure Sonké, and Rodolfo Vásquez (2022). "Making forest data fair and open". In: *Nature Ecology &amp Evolution* 6.6, 656.

Lima, Renato AF de, Oliver L Phillips, Alvaro Duque, J Sebastian Tello, Stuart J Davies, Alexandre Adalardo de Oliveira, Sandra Muller, Euridice N Honorio Coronado, Emilio Vilanova, Aida Cuni-Sanchez, et al. (2022). "Making forest data fair and open". In: *Nature Ecology & Evolution*, 1.

Segura, Milena, Markku Kanninen, and Damaris Suárez (2006). "Allometric models for estimating aboveground biomass of shade trees and coffee bushes grown together". In: *Agroforestry Systems* 68 (2), 143.

Van Noordwijk, Meine, Subekti Rahayu, Kurniatun Hairiah, Y. Wulan, Ai Farida, and Bruno Verbist (2002). "Carbon stock assessment for a forest-to-coffee conversion landscape in Sumber-Jaya (Lampung, Indonesia):

from allometric equations to land use change analysis". In: *Science in China* 45.

Yuliasmara, Fitria, Aris Wibawa, and Adi Prawoto (2009). "Carbon stock in different ages and plantation system of cocoa: allometric approach". In: *Pelita Perkebunan (a Coffee and Cocoa Research Journal)* 26.

Brown, S. and Louis Iverson (1992). "Biomass estimates for tropical forest". In: *World Res. Rev.* 4, 366.

Ma, Lei, Yu Liu, Xueliang Zhang, Yuanxin Ye, Gaofei Yin, and Brian Alan Johnson (2019). "Deep learning in remote sensing applications: A meta-analysis and review". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 152, 166.

Weinstein, Ben G, Sergio Marconi, Stephanie Bohlman, Alina Zare, and Ethan White (2019). "Individual tree-crown detection in RGB imagery using semi-supervised deep learning neural networks". In: *Remote Sensing* 11.11, 1309.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). *Deep Residual Learning for Image Recognition*.

Spawn, S.A., C.C. Sullivan, and T.J. et al. Lark (2020). "Harmonized global maps of above and belowground biomass carbon density in the year 2010". In: *Sci Data* 7, 112.

Stewart, Adam J., Caleb Robinson, Isaac A. Corley, Anthony Ortiz, Juan M. Lavista Ferres, and Arindam Banerjee (2022). "TorchGeo: Deep Learning With Geospatial Data". In: *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*. SIGSPATIAL '22. Seattle, Washington: Association for Computing Machinery, 1.

Arrieta-Ibarra, Imanol, Leonard Goff, Diego Jiménez-Hernández, Jaron Lanier, and E. Glen Weyl (2018). "Should We Treat Data as Labor? Moving beyond "Free"". In: *AEA Papers and Proceedings* 108, 38.

Posner, Eric A and E Glen Weyl (2019). "5. Data as Labor". In: *Radical Markets*. Princeton University Press, 205.

Karlaš, Bojan, David Dao, Matteo Interlandi, Bo Li, Sebastian Schelter, Wentao Wu, and Ce Zhang (2022). *Data Debugging with Shapley Importance over End-to-End Machine Learning Pipelines*.

Bennett, George (1962). "Probability inequalities for the sum of independent random variables". In: *Journal of the American Statistical Association* 57.297, 33.

Topsok, Flemming (2006). "Some bounds for the logarithmic function". In: *Inequality theory and applications* 4, 137.