Noise reduction in Differentially Private Learning

By

Zhanliang Huang

A thesis submitted to
the University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
University of Birmingham
January 2023

ABSTRACT

Privacy protection is a rising concern that gained increasing attention over the past decade due to the widespread of machine learning applications. Differential privacy (DP) is an emerging notion of privacy that provides a rigorous information-theoretic privacy guarantee. While DP is a very useful privacy guarantee many practitioners aim to achieve, DP algorithms typically require more data samples to perform well. Moreover, the magnitude of the injected noise from DP can scale with the dimensionality, hence further increasing the difficulty of private learning in high dimensions. In addition, high-dimensional learning also incurs problem it-self known as ‘the curse of dimensionality’.

The notion of compressed learning that aims to achieve a low-dimensional representation of the high-dimensional data samples has shined some light on this problem. However, although the notion of random projection has pre-existed for a long time, little is known about randomly compressed models in the DP framework. In this thesis, we develop theories that quantify the effect of random projections on the learning performance, and the excess error that privacy incurred. The theory developed in this thesis demonstrates the interplay between generalisation performance, random projection and differential privacy. We quantify the effect of random projection and the effect of DP implementation on the generalisation performance of learning algorithms. We show that random projection can reduce the magnitude of the required noise injection in DP algorithms while also exploiting structure, which results in a reduced dimensionality dependence on generalisation guarantees. Finally, we also introduce a novel machine ensemble with known in-built structure-exploiting capability, which utilises the privacy budget efficiently and is able to use the assistance of unlabelled data samples to boost accuracy performance without the compression of data samples.

DEDICATION

Dedicated to all of my family and my cat.

ACKNOWLEDGMENTS

I would like to follow the tradition and use this opportunity to thank the people who has helped me along the way, this work would not have been possible without them.

I would like to give my special thanks to my partner Hannah and my family, for their love, patience and relentless support throughout. Thank you Hannah for your accompany and the joy you brings, it would be a more difficult and stressful journey without you.

I would like to express my deepest appreciation to my supervisors Prof. Ata Kabán and Dr. Yunwen Lei who have been providing valuable guidance and advice throughout all stages of this thesis. Thank you for the time and patience to read through my work and providing long lists of improvements that I would struggle to come up myself. I have enjoyed the fun and interactive discussions between us and thank you for the invaluable insights and supports. I would also like to extend my gratitude to my RSMG group member Dr. Tom Chothia for his practical suggestions and helpful advice to improve my work. I would also like to thank all of those that I had the privilege to meet during the course of this work, for interesting and often informative discussions. In particular, I would like to thank Dr. Henry Reeve for his constructive criticism and sharing his extensive knowledge in my first year. Thanks should also go to Wu, for his helpful advice on programming and software assistance.

Finally, I would like to thank the School of Computer Science at the University of Birmingham for their financial support that allowed me to focus on my research, as well as the School Research committee for funding to conferences. This work would surely be impossible without their generous financial support.

Zhanliang Huang

January 2023

"One, remember to look up at the stars and not down at your feet.

Two, never give up work. Work gives you meaning and purpose and life is empty without it.

Three, if you are lucky enough to find love, remember it is there and don't throw it away."

– Stephen Hawking

Contents

List of Figures

List of Tables

Chapter One

Introduction

1.1 Motivation

The term ‘Machine Learning’ (ML) refers to the study of pattern recognition and making predictions based on previously observed data, which is the analogue of human ‘learning’ from past ‘experience’. ML algorithms typically aim to output a good-performing predicting function, $\hat{h}$, from a function class $\mathcal{H}$ that is tailored to solve a specific task. As pointed out by the ‘No-Free-Lunch Theorem’, there does not exist a model that works perfectly for every problem [137]. ML can be divided into sub-branches as ‘supervised learning’, ‘unsupervised learning’, ‘semi-supervised learning’, ‘active learning’, etc, depending on what kind of information is available to the learner during the training phase (picking a good predictor $\hat{h}$ from $\mathcal{H}$). Since the underlying data distribution, $\mathcal{D}$ is usually unknown, the training is done by observing a finite set S of identically and independently distributed (i.i.d.) sample points from $\mathcal{D}$.

Due to the increasingly wide applications of ML algorithms nowadays, concerns about data privacy have rise a considerable amount of attention over the past decade. The observations used to train ML models may contain sensitive information that the data subject does not want to disclose to the public or any third parties. For example, medical records and financial records contain confidential information but they are very useful for medical/financial analysis. This creates a conflict between the data subject and the data collector. Intuitive methods to protect data privacy such as anonymizing names or only publishing ‘summarised statistics’ has proven unsafe from attacks, even if we only care about the privacy of a small subset of a very large sample set [45, 125, 103]. A particularly innovative and popular approach to a solution of learning privately is *differential privacy* (DP), which has been initially introduced by Dwork [50] and has gained much attention in the field. Differential privacy provides a rigorous mathematical definition of privacy that we can implement in ML algorithms and provide a privacy guarantee. In simple words, DP is a promise, made by the data holder to the data subject: ‘You will not be affected, adversely or otherwise, by allowing your data to be used in any study or analysis, no matter what other studies, data sets, or information sources, are available’ [48]. DP algorithms allow their outputs to be published and shared across multiple parties to carry out useful analyses or studies, due to the properties of DP guarantee. The key to implementing DP in a learning algorithm is by injecting controlled randomisation into key steps of the algorithm. However, while these randomisations are controlled with a carefully designed distribution, there are some issues that it brings: 1) Repeated queries to the private samples reduce privacy guarantee meaning that we can only query the sample set a limited number of times. 2) Variance of randomisation depends on the sensitivity of the query, hence queries that are very dependent on a particular sample are very expensive in privacy (e.g. the range of samples). 3) The variance of the randomisation also depends on the number of samples n, which means that the variance can be large if n is small and overwhelm the true signal.

Although these are risks we have to cope with in all DP applications, the issues become even worst in high-dimensional learning, which is the setting when the data samples have a large number of attributes. Consider the following two common scenarios of high-dimensional learning:

1) The data samples are cheap and easy to collect, hence the number of samples n and the dimensionality d are both large. In this case, DP is less likely to significantly worsen the accuracy performance of the learning algorithm. However, the time and space complexity of large-scale high-dimensional problems is a common concern, and DP can add further

time and space complexity to the algorithm due to the added randomisation process.

2) Another scenario is when the data sample is high-dimensional but expensive/difficult to collect. This is especially the case for private and sensitive data, as people are less likely to participate in data collection if it is confidential. In this case, the number of samples n can be less than the dimensionality d, which is a difficult learning situation due to the lack of samples. It is known that the number of samples required for training grows exponentially with the dimension [5, 67]. DP algorithms, however, typically require more samples than their non-private versions to achieve similar accuracy performance due to the added randomisation [55, 34].

Unfortunately, the price of the DP guarantee is higher for high-dimensional problems as it worsens the issue with high-dimensional data samples in both scenarios discussed above. As an example, injecting a noise vector into the key computations of the algorithm is a common step for DP algorithms. While this noise vector has a Euclidean norm very close to zero at small dimensions, the Euclidean norm of the noise vector at higher dimensions scales with order $\sqrt{d}$. [120]

While there exist many dimensionality reduction techniques that aim to reduce the dimension of the data, data-independent methods will be more suitable in the privacy setting because we do not need to 'privatize' the dimensionality reduction process. *Random projection* (RP), in particular, is a very useful dimensionality reduction technique if the sample set has a simple structure (low-dimensional structure). Real high-dimensional data sets such as medical imaging and signal processing have a low-dimensional structure but they are embedded in a higher dimension. Random projection provides a simple, efficient and non-adaptive approach to achieve a low dimensional representation of the data set. The non-adaptive approach means that the dimensionality reduction does not depend on the data samples S, hence privacy bleach is not a concern when publishing the random projection step as part of our learning algorithm. By generating a suitable random matrix as a random projector, we can transform the high-dimensional data samples into a low-dimensional subspace such that, all Euclidean distances between the sample points are preserved with high probability [94, 89].

This fantastic property allows us to perform distance-related learning algorithms such as Nearest Neighbours with the projected samples only, which is commonly referred to as 'compressed learning' in the literature. In other words, if the data samples are contained in a low-dimensional structure, then we can learn a model in the low dimension that is as good as we could get using the original data. For private learning, this can be especially useful because we can also potentially reduce the dimensionality of the noise vector, which can reduce the magnitude of the noisy vector and improve accuracy.
Although existing work has studied the condition where random projection can be applied [43, 89, 18] without ruining utility, it is still unclear how random projection affects learning performance, especially in the privacy setting. In this thesis, we study this problem by analysing the effect of compressed learning on the generalisation guarantee of learning algorithms. We focus on random projection as a data-compressing technique and analyse its effect on both private and non-private learning. It is intuitive that DP models are likely to be sub-optimal as they can never do better than non-private models due to the added randomisation for privacy guarantees. Despite randomisation has been seen as a technique towards increasing the diversity of model selection in ML, e.g. random forest [26] and extreme random trees [59], and to increase learning efficiency as for Stochastic Gradient Methods [19], the randomness invoked in DP algorithm is typically where the key computations lie and hence usually decreases accuracy performance. Nevertheless, it is very important and useful to understand the excess error that privacy brings, as well as the trade-off between privacy and accuracy.

1.2 Research questions

In this thesis, we are interested in the interplay among random projection, privacy guarantee and generalisation guarantee of learning models. Research questions that we would like to investigate include:

- To what extent does differential privacy affect the generalisation performance of private classification? What are the key factors that control the randomisation required of DP algorithms?

- To what extent does random projection affect the generalisation performance of non-parametric classification compared to high-dimensional models? What type of data geometry is able to preserve the accuracy performance for compressed models?

- Does random projection affect the generalisation of private models? What is the achievable error convergence rate with respect to the number of samples n for compressed private models?

1.3 Contributions of the thesis

The main contributions of this thesis are the theoretical results of compressed classification models and differentially private predictors. We start with the generalisation guarantee of non-parametric algorithms with random projection in Chapter 4. We prove the generalisation error guarantee of the famous k-Nearest Neighbour (kNN) algorithm under compressed learning with random projections, which has not been studied before except in the $k = 1$ case. We introduce the histogram classifier which is a non-parametric classifier that can be seen as an efficient approximation of kNN. We show that the histogram classifier can achieve the optimal convergence rate but can also further improve in the realizable case. Furthermore, we show that random projection can improve the dimensionality dependence of its generalisation bound, where we have also investigated the variations of data geometry that allowed the histogram classifier to learn well with randomly projected samples. We proposed a modified histogram classifier that uses a neighbour search in sparse regions to avoid over-fitting in higher dimensions.

In Chapter 5, we introduce approaches to modify the kNN algorithm and the histogram classifier so that the trained model satisfies differential privacy. We analyse the generalisation performance of the private classifiers in expectation over the randomness of drawing an i.i.d. sample set from the data distribution and the randomness injected for privacy. We show that the privacy classifier achieves the same generalisation error convergence with respect to the number of samples. In particular, if the level of privacy guarantee approaches zero, then we can recover the exact same rate of convergence as we have observed from the non-private models. Furthermore, we show that random projection can improve the dimensionality dependence of their generalisation convergence similar to the non-private models in Chapter 4. More interestingly for the histogram classifier, random projection reduces the dimensionality of the injected noise by training with compressed samples in a lower-dimension space. Hence decreasing the magnitude of the norm of the injected noise vector.

In Chapter 6 we turn our focus to gradient descent methods with randomly projected (compressed) gradients (CompSGD) which can be used to train parametric models such as logistic regression. By using random projection to project the gradient vector, we can perform each gradient update in a lower dimension while keeping track in the original dimension by 'lifting' the updated vector back to the original space. We show that using only the projected gradient we can achieve 'almost optimal' optimisation convergence compared to classic SGD (up to log factors). Furthermore, we give the first analysis of the generalisation performance of CompSGD by proving the uniform stability bound. We further extend the optimisation and generalisation analysis to two variants of CompSGD that uses batch gradient and randomly sampled mini-batch gradients respectively. The result obtained for the variants matches the same convergence rate for non-projected methods up to logarithmic factors.

In Chapter 7 we study the compressed gradient method in the differentially private setting. We prove the first optimisation and stability bound of gradient descent with randomly projected gradients. From this, we obtain the first generalisation results for DP compressed gradient descent. We also extend the analysis to DP-compressed SGD with mini-batches and observed that DP-CompGD has a better optimisation convergence despite the generalisation convergence being the same with mini-batch. In the case where the level of privacy guarantee approaches zero, we recover the error bound obtained in Chapter 6 for the non-private models. Furthermore, we show that the dimensionality dependence induced from DP is reduced by compressing the gradients at each gradient update, as randomized noise can now be injected at a lower dimension.

In Chapter 8 we investigate an alternative method of noise-efficient private learning besides compressing the data samples using random projection. We proposed a new framework that builds DP ensemble classifiers using a semi-randomized tree construction. We invoke a median-based splitting criterion to build trees with a balanced structure, which can balance out added noise in each region of the sample space. We show that by using a geometric-scaling privacy budget allocation technique we can improve the accuracy of the median computation, which further improves accuracy performance. We show that our framework naturally extends to semi-supervised learning, and hence takes advantage of unlabelled data samples that are potentially cheaper to collect than labelled samples. We demonstrate the effect of having an additional unlabelled sample on the final accuracy performance, the improvement in accuracy is particularly significant for smaller sample sets or when a strong privacy guarantee is required. Empirically we show that our proposed method significantly improved the accuracy performance of DP classifiers with a similar structure.

The rest of the thesis is organised as follows. Chapter 2 provides a brief summary of the main background knowledge, including key definitions and results of learning theory, random projections and differential privacy. Chapter 3 summarizes the background literature and the state-of-the-art related to random projections and differentially private learning. The last main chapter, Chapter 9 will summarize the results presented in this thesis and discuss the open problems left for future research. Finally, the rest of the chapters (4, 5, 6, 7 and 8) are described in the previous section.

1.4 Publications from this thesis

Zhanliang Huang, Yunwen Lei, and Ata Kabán. “*Noise-efficient Learning of Differentially Private Partitioning Machine Ensembles: Noise Reduction in Private Forests.*” In European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases. 2022.

Zhanliang Huang, Yunwen Lei, and Ata Kabán. “*Optimisation and Learning with Randomly Compressed Gradient Updates.*” Neural Computation (Journal), Volumne 35, Issue 7, July 2023.

Zhanliang Huang, Ata Kabán and Henry Reeve. “*Efficient differentially private learning with projected histograms*” Data Mining and Knowledge Discovery (Journal) - *Submitted.*

Chapter Two

Mathematical Background

In this chapter, we give a short summary of the mathematical backgrounds and tools that provide the foundation of our research. Many of these ideas are well-known and widely used in machine learning. The main topics are statistical learning theory, differential privacy and random matrix theory.

2.1 Statistical learning theory

Statistical learning theory is a framework that allows us to study the properties of learning algorithms using statistical tools, where the results are usually presented in the form of error bounds. There are various forms of error bounds that provides useful information about the properties of the learning algorithms such as how "good" an algorithm is or how "complex" the learning problem is. Here, the definition of "good" and "complex" can be interpreted differently using different notions of measurements. In this section, we will be mainly looking at generalization bounds and the techniques we can use to derive tight error bounds. Many works exist to date on providing error bounds on various problems using the fundamental concepts and techniques presented in this section repeatedly.

2.1.1 Generalization bounds

Generalization bounds provide important information on how well the learning model generalizes to new/unseen samples, typically on the performance in the worst-case scenario and the rate of convergence with respect to the size of the training set. Models that perform well on the training set but badly on new test points are said to be *over-fitted.* Since the testing error of a model is arguably the most important accuracy measure, this makes the study of generalization bounds important and meaningful. To formalize the idea of generalization errors, we consider the following general setting of supervised learning: Let $\mathcal{X}$ be a feature space containing d features (or attributes) and $\mathcal{Y}$ denote the label space (we usually let $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{0, 1\}$ for binary classification). For some arbitrary unknown probability distribution $\mathcal{D}$ over some sample space $\mathcal{Z} \subseteq \mathcal{X} \times \mathcal{Y}$, where each $Z \in \mathcal{Z}$ consists of d attributes and a label, we denote by $S = \{Z_1, \ldots, Z_n\}$ a finite sample set of size $|S| = n$ drawn independently and identically distributed (i.i.d.) from $\mathcal{D}$. Let $\mathcal{H}$ denote an arbitrary hypothesis class. A learning algorithm A receives as input a training sample set S and outputs a predictor $\hat{h}$. For a sample $Z \in \mathcal{Z}$, we quantify the quality of a hypothesis $\hat{h} \in \mathcal{H}$ with respect to Z using a *loss function* $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}_+$, where $\mathbb{R}_+$ is the set of non-negative reals. Examples of common loss functions include the 0-1 loss, squared loss, logistic loss, hinge loss, etc. For a given loss function ℓ, we aim to minimize the generalization loss with respect to $\mathcal{D}$ defined as:

$$L_{\mathcal{D}}(h) = \mathbb{E}_{Z \sim \mathcal{D}}[\ell(h, Z)]. \tag{2.1}$$

Since in most cases the distribution $\mathcal{D}$ is unknown, a strategy is to minimize its empirical analogue using the finite sample set S defined as:

$$L_S(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h, Z_i). \tag{2.2}$$

$L_S(h)$ is often referred to as the *empirical risk* or the *training error* of h and $L_{\mathcal{D}}(h)$ is often referred to as the *true risk* or the *generalization error* of the h. The learning process via minimizing the empirical risk is called the *Empirical Risk Minimization* (ERM). We now demonstrate the theory of PAC (Probably Approximately Correct) learning which shows that if the hypothesis $\hat{h}$ is the output of an ERM, then its true risk is also small with high probability (given also that $\mathcal{H}$ satisfies certain conditions). The definition of PAC learning is given as follows.

Definition 1 (Agnostic PAC Learnability [115]). *A hypothesis class $\mathcal{H}$ is agnostic PAC learnable with respect to a set $\mathcal{Z}$ and a loss function $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}_+$, if there exists a function $n_{\mathcal{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm with the following property: For every $\epsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over $\mathcal{Z}$, when running the learning algorithm on $n \geq n_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by $\mathcal{D}$, the algorithm returns $h \in \mathcal{H}$ such that, with probability of at least $1 - \delta$ (over the choice of the n training examples),*

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon.$$

The number $n_{\mathcal{H}}(\epsilon, \delta)$ here is often referred to as the *sample complexity*, which is the smallest number of samples required to guarantee (2.2) holds. We also denote $h^* \in \mathcal{H}$ as the best performing hypothesis in $\mathcal{H}$:

$$h^* = \underset{h \in \mathcal{H}}{\arg\min}\ L_{\mathcal{D}}(h). \tag{2.3}$$

Note: The definition provided here is agnostic PAC learnability. PAC learnability has the additional assumption that there exists a 'perfect' hypothesis in the class which has zero error (i.e. $L_{\mathcal{D}}(h^*) = 0$). However, the two classes are equivalent from the result of VC theory [115].

In machine learning, one of the typical tasks is to choose a hypothesis from a hypothesis class with the best performance, and one way to accomplish this is to consider the empirical risk of the hypothesis. However as mentioned before, a model that performs well on the training set may not necessarily generalize well to testing samples due to over-fitting. The following result from PAC theory ensures that over-fitting does not happen if we have a sufficiently large training set.

Theorem 2.1 ([115])**.** *Let $\mathcal{H}$ be a finite hypothesis class, let $\mathcal{Z}$ be a domain, and let $\ell : \mathcal{H} \times \mathcal{Z} \to [0,1]$ be a loss function. Then, the hypothesis class is agnostically PAC learnable using the ERM algorithm with sample complexity*

$$n_{\mathcal{H}}(\epsilon, \delta) \leq \left\lceil \frac{2\log(2|\mathcal{H}|/\delta)}{\epsilon^2} \right\rceil .$$

This is a strong and useful result as it ensures that our hypothesis output from the ERM will generalize well on unseen sample points, and it provides useful information on how large the training set needs to be. The drawback is that the sample complexity grows logarithmically with the size of the hypothesis class which may be problematic for large classes and not applicable to non-finite classes, this issue can be addressed by using the notion of uniform convergence and VC dimensions [115].
One of the cause of over-fitting is that the sample set we obtained has a distribution that is far from its true underlying distribution. i.e. the sample set is a bad representation of the true underlying distribution. Hence if we have a more representative sample set then the chance of over-fitting will be relatively smaller, this is the intuition of uniform convergence is formalised as the following:

Definition 2 (ϵ-representative sample [115])**.** *A training set S is called ϵ-representative (w.r.t. domain $\mathcal{Z}$, hypothesis class $\mathcal{H}$, loss function ℓ, and distribution $\mathcal{D}$) if*

$$\forall h \in \mathcal{H}, \quad |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon. \tag{2.4}$$

If we have a representative sample set, then the approximation of the underlying distribution calculated with the given sample set wouldn't be too far off the true distribution. Hence any hypothesis that performs well on this approximation well enough should also have a good performance on the true distribution. This argument is stated formally in the following lemma.

Lemma 2.2 ([115])**.** *Assume that a training set S is $\epsilon/2$-representative. Then, any output of $ERM_{\mathcal{H}}(S)$, namely, any $\hat{h} \in \arg\min_{h\in\mathcal{H}} L_S(h)$, satisfies*

$$L_{\mathcal{D}}(\hat{h}) \leq \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon. \tag{2.5}$$

By the law of large numbers, if we have a sufficiently large sample set, we should be fairly confident that our sample set is a representative sample of the underlying distribution. The question is, how large will be sufficiently large? And for a sample set of given size n, how confident are we that the sample set is a representative sample set? This confidence level is formalised by the uniform convergence notion:

Definition 3 (Uniform convergence [115])**.** *We say that a hypothesis class $\mathcal{H}$ has the uniform convergence property if there exists a function $n_{\mathcal{H}}^{UC} : (0,1)^2 \to \mathbb{N}$ such that for every $\epsilon, \delta \in (0,1)$ and for every probability distribution $\mathcal{D}$ over Z, if S is a sample of $n \geq n_{\mathcal{H}}^{UC}$ examples drawn i.i.d. according to $\mathcal{D}$, then, with probability of at least $1-\delta$, S is ϵ-representative.*

From here we can conclude that if a hypothesis class $\mathcal{H}$ has the uniform convergence property then it also satisfies the PAC learnability (but with a different ϵ parameter) and hence we can provide error bounds of the hypothesis with a high probability. The number of samples required to guarantee ϵ-representative with high probability is given by the function $n_{\mathcal{H}}^{UC}$ which depends on the hypothesis class $\mathcal{H}$.

2.1.2 Stability and generalization

The stability of a learning algorithm has close relations to its generalization performance. Intuitively, a learning algorithm A is considered to be a stable algorithm if a small change in the inputs does not affect the output of the algorithm by too much. Recall that we are interested to output a hypothesis $\hat{h} \in \mathcal{H}$ that minimizes the empirical risk in the context of ERM. As seen in the previous section we often analyse the generalization risk of $\hat{h}$ by comparing it to the risk of the best-performing hypothesis h^*. Hence we can considering the *excess generalization error*: $L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*)$, which is the difference between the true (generalization) risk of the output hypothesis and the best hypothesis in $\mathcal{H}$. The expected excess generalization error can be decomposed into two separate terms:

$$\mathbb{E}_{S,A}[L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*)] = \mathbb{E}_{S,A}[L_{\mathcal{D}}(\hat{h}) - L_S(\hat{h})] + \mathbb{E}_{S,A}[L_S(\hat{h}) - L_S(h^*)], \tag{2.6}$$

where the expectation is taken over the random draws of samples in S and randomness in algorithm A. The first term of the decomposition is called the estimation error due to sampling of S and the second term is the optimization error induced by minimizing the empirical risk. The stability properties of the algorithms have a strong connection to their generalization, as stability allows us to understand the scale of the estimation error. The notion of stability that we will employ is *uniform stability.* Uniform stability is a widespread stability measure that drives powerful analysis [51, 66].

Definition 4 (Uniform stability)**.** *An algorithm A is ϵ-uniformly stable if for all $S, S' \in \mathcal{Z}^n$ that differ by at most one example, we have*

$$\sup_{Z} \mathbb{E}_A \left[|\ell(A(S), Z) - \ell(A(S'), Z)| \right] \leq \epsilon. \tag{2.7}$$

If A is uniformly stable then we denote by ϵ_{stab} the infimum over all ϵ in which (2.7) *holds.*

Throughout this thesis, we will consider stability in the setting where we have two neighbouring sample sets S, S' differing in one point. The following powerful theorem connects uniform stability and generalization:

Theorem 2.3 ([116])**.** *If algorithm A is ϵ_{stab}-uniformly stable, then*

$$|\mathbb{E}_{S,A} \left[L_{\mathcal{D}}(A(S)) - L_S(A(S)) \right]| \leq \epsilon_{stab}. \tag{2.8}$$

2.2 Properties of functions

In the theoretical analysis of this thesis, we will be required to exploit some key properties of functions that implies useful results. In this section, we present some important properties of functions that we will use in many parts of our theory.

Definition 5 (Lipschitzness). *Let $\mathcal{X} \subseteq \mathbb{R}^d$. A function $f : \mathcal{X} \to \mathbb{R}^m$ is L-Lipschitz with respect to the norm $\|\cdot\|$ over $\mathcal{X}$ if $\forall x, x' \in \mathcal{X}$ we have for $L \geq 0$ that*

$$\|f(x) - f(x')\| \leq L\|x - x'\|. \tag{2.9}$$

Lipschitzness is an important property that we will use in the majority of the chapters throughout this thesis. Lipschitzness of a function f guarantees that the difference between function values is bounded by a constant multiple of the distance of their inputs. Note that the metric norm in the domain and codomain space can be different in the definition of Lipschitzness. For our purpose, we are mainly interested with respect to the ℓ_2 norm unless otherwise specified.

Definition 6 (Convexity). *Let $\mathcal{X} \subseteq \mathbb{R}^d$. A differentiable function $f : \mathcal{X} \to \mathbb{R}$ is convex over $\mathcal{X}$ if $\forall x, x' \in \mathcal{X}$ we have that*

$$f(x) \geq f(x') + \langle x - x', \nabla f(x') \rangle. \tag{2.10}$$

Definition 7 (Smoothness). *Let $\mathcal{X} \subseteq \mathbb{R}^d$. A differentiable function $f : \mathcal{X} \to \mathbb{R}$ is μ-smooth on $\mathcal{X} \subseteq \mathbb{R}^d$ if $\forall x, x' \in \mathcal{X}$ we have*

$$\|\nabla f(x) - \nabla f(x')\| \leq \mu\|x - x'\|. \tag{2.11}$$

Note that smoothness of f is also equivalent to Lipschitzness of the first-order gradient function of f. In the case where a function f is both convex and smooth, we can derive some very useful inequalities.

Theorem 2.4 ([104]). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex and μ-smooth function. We have $\forall x, y \in \mathbb{R}^d$:*

1. *(upper bound)* $f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \frac{\mu}{2}\|x - y\|^2$;
2. *(co-coercivity)* $\frac{1}{\mu}\|\nabla f(x) - \nabla f(y)\|^2 \leq \langle \nabla f(x) - \nabla f(y), x - y \rangle$;
3. *(lower bound)* $f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle + \frac{1}{2\mu}\|\nabla f(x) - \nabla f(y)\|^2$.

2.3 Differential privacy

In this section, we give a brief introduction to the notion of differential privacy (DP) and its key properties. We also give a short summary of the key mechanisms and composition theorems that we will be using in later chapters. These results are considered as the building bricks of differentially private algorithms. Many complex DP algorithms consist of a combination of standard mechanisms to obtain powerful algorithms [48, 120, 122, 55, 140, 134].

Suppose that we want to perform a research study on a confidential database. In the case of perfect privacy, the output of any research should not compromise the privacy of any individual. However, this has been proved impossible if we want to collect any useful data at all because any meaningful conclusion from the research will compromise a certain amount of privacy. Differential privacy is a notion that weakens the perfect privacy requirement but still provides a very strong semantic guarantee of privacy. It has the idea that the participation of any individual in the data set should not cause additional compromise in privacy. That is to say that, the outcome of a differentially private algorithm should not be much different with or without any particular sample. This is a useful guarantee as it provides individuals with reasonable deniability of their participation in a data set and it preserves the utility of the data. Hence this allows the output of a DP learning algorithm to be shared across multiple parties without concern of leaking the information of a particular individual. The definition is given formally as follows: We first define the distance between two sample sets S,S' to be the Hamming distance denoted as $\|S - S\|_H$, which equals the number of points to be added or removed from S until $S = S'$. In the case where $\|S - S\|_H \leq 1$, we say that S, S' are *neighbouring* sets. To distinguish between ϵ, δ from PAC learning, we use $0 < \epsilon_p, 0 \leq \delta_p < 1$ in this section as privacy parameters.

Definition 8 (Differential privacy [47])**.** *For $\epsilon_p > 0$ and $0 \leq \delta_p < 1$, a randomized algorithm A is said to satisfy (ϵ_p, δ_p)-differential privacy if for all measurable subsets $B \subset Range(A)$ and for all $S, S' \subseteq \mathcal{Z}$ such that $\|S - S'\|_H \leq 1$, we have*

$$\mathbb{P}[A(S) \in B] \leq \exp(\epsilon_p)\mathbb{P}[A(S') \in B] + \delta_p, \tag{2.12}$$

where the probability space is over the coin flips of the algorithm A. In particular if $\delta_p = 0$, then we say that A is ϵ_p-differentially private.

We note that some other literature on differential privacy considers the case where S and S' (with the same size) differ by at most one sample point. In contrast, we consider adding/removing a sample point here, which is the setting considered in most of the related works. The two settings only differ by a constant factor in the sensitivity analysis. The notion of (ϵ_p, δ_p)-differential privacy is a weakening of ϵ_p-differential privacy. Roughly, ϵ_p-differential privacy says that for any similar data sets S, S', it is very unlikely that the output of the differentially private algorithm $A(S)$ and $A(S')$ will be much different to each other. In contrast, (ϵ_p, δ_p)-differential privacy ensures that for all neighbouring sets S, S', the absolute value of the privacy loss will be bounded by ϵ_p with probability at least $1 - \delta_p$.

An immediate result from the definition is that DP algorithms are immune to *post-processing.* i.e. If A is ϵ-DP, then the composition $f \circ A$ is also ϵ_p-DP for an arbitrary mapping f [48].

Theorem 2.5 (Post-processing [48])**.** *Let $\mathcal{M} : \mathcal{Z}^n \to R$ be a randomized algorithm that is (ϵ_p, δ_p)-differentially private. Let $f : R \to R'$ be an arbitrary randomized mapping. Then*

$f \circ \mathcal{M} : \mathcal{Z}^n \to R'$ is (ϵ_p, δ_p)-differentially private.

A typical way of achieving differential privacy is by introducing randomized noise in one or more steps of the algorithm. To add a suitable amount of noise into our algorithm, we need to consider the *sensitivity*:

Definition 9 (Global ℓ_1-sensitivity [47]). *The global ℓ_1-sensitivity $\Delta(A)$ of a function (non-private algorithm) $A : \mathcal{Z}^n \cup \mathcal{Z}^{n-1} \to \mathbb{R}^r$ is defined as*

$$\Delta(A) = \max_{S,S' \subset \mathcal{Z} : \|S - S'\|_H = 1} \|A(S) - A(S')\|_1. \tag{2.13}$$

We also define its analogue ℓ_2-sensitivity as

$$\Delta_2(A) = \max_{S,S' \subset \mathcal{Z} : \|S - S\|_H = 1} \|A(S) - A(S')\|_2. \tag{2.14}$$

The global sensitivity captures the maximum difference in the output when we use a neighbouring data set that differs by at most one point. There are other notions of sensitivity, such as the local sensitivity that considers the particular data set S. Local sensitivity can be significantly smaller than global sensitivity in many cases. However, local sensitivity in itself does not guarantee differential privacy. Next, we present two well-known mechanisms for private algorithm design - the Laplace and Exponential mechanisms.

Definition 10 (Laplace mechanism of [48]). *Given any function $A : \mathcal{Z}^n \to \mathbb{R}^r$, the Laplace mechanism M is defined as*

$$M(S, A, \epsilon_p) = A(S) + \beta, \tag{2.15}$$

where β is a r-dimensional noise vector with entries drawn i.i.d. from Lap$(\Delta(A)/\epsilon_p)$.

The Laplace mechanism provides randomisation in the algorithm by adding a randomly generated noise vector in the output of a function f. The expected magnitude of the noise vector mainly depends on the sensitivity of f and the privacy budget ϵ_p.

Definition 11 (Exponential mechanism of [99])**.** *Let $\mathcal{R}$ be an arbitrary set of output candidates. Given a utility function $u : \mathcal{X}^n \times \mathcal{R} \to \mathbb{R}$ that computes the quality of a candidate $r \in \mathcal{R}$, the exponential mechanism $\mathcal{M}(S, u, \mathbb{R})$ selects and outputs one of these with probability proportional to the following*

$$\mathbb{P}[\mathcal{M}(S, u, \mathbb{R}) = r] \propto \exp\left(\frac{\epsilon_p u(S, r)}{2\Delta(u)}\right). \tag{2.16}$$

In contrast to the Laplace mechanism, the Exponential mechanism works by randomly selecting a candidate as the output with probability proportional to the "utility" of the candidate. It is important how we decide to quantify the "utility" of a candidate based on the problem for the exponential mechanism to work well. As a simple example, if we are trying to publish the most deadly disease in 2022, then the "utility" of a given disease should be proportional to the death rate of the disease.

It is very well-known that the Laplace and the Exponential mechanisms both satisfy ϵ_p-DP. The reader can refer to [48] for detailed proof of the privacy guarantee.

Besides the Laplace and Exponential mechanism which both guarantees ϵ_p-DP, another popular private mechanism is the *Gaussian mechanism*. The Gaussian mechanism works similarly to the Laplace mechanism but adds Gaussian noise instead of Laplacian noise, with an aim to reduce added noise in comparison to Laplace.

Definition 12 (Gaussian mechanism [50])**.** *Let $A : \mathcal{Z}^n \to \mathbb{R}^m$. The algorithm with input $S \in \mathcal{Z}^n$ outputs $A(S) + \beta$ where $\beta \sim N(0, 2\Delta_2(f)^2 \log(2/\delta_p)/\epsilon_p^2 I_{m\times m})$ is (ϵ_p, δ_p)-differentially private. Here $I_{m\times m}$ denotes the identity matrix in $\mathbb{R}^{m\times m}$.*

The Gaussian mechanism only guarantees (ϵ_p, δ_p)-DP which is a weaker guarantee compared to Laplace, however, it can reduce added noise in the case where we do not require pure differential privacy.

With these private mechanisms on hand, the following composition theorems allow us to combine multiple private mechanisms to develop more complex private algorithms.

Theorem 2.6 (Sequential composition [99])**.** *For a sequence of queries $\{f_i\}_{i=1}^N$ on a data set S each satisfying $(\{\epsilon_i\}_{i=1}^N, \{\delta_i\}_{i=1}^N)$-differential privacy. Then the output sequence $\{f_i(S)\}_{i=1}^N$ of all queries satisfies $\left(\sum_{i=1}^N \epsilon_i, \sum_{i=1}^N \delta_i\right)$-differential privacy.*

Theorem 2.7 (Parallel composition [100])**.** *For disjoint subsets $\{S_i\}_{i=1}^N \subset S$, consider a query f applied on each of the subset S_i while satisfying (ϵ_p, δ_p)-DP. Then the output sequence $\{f(S_i)\}_{i=1}^N$ satisfies (ϵ_p, δ_p)-DP.*

Sequential composition states that the more queries we send to the original data set, the less privacy guarantee we have. Parallel composition says that we do not lose the independent privacy guarantees if we query disjoint subsets independently.

In the case where we query the private data over N adaptive interactions, adaptive interaction means the attacker sees the previous query output before making the next query, the following composition theorem provides a stronger result on the accumulated privacy loss.

Theorem 2.8 (Strong Composition [49])**.** *Let $\epsilon_p, \delta_p, \delta_p' > 0$ and $\epsilon_p \leq 1$. A mechanism that permits N adaptive interactions with mechanisms that preserves (ϵ_p, δ_p)-differential privacy ensures $(\epsilon_p\sqrt{2N\log(1/\delta_p')} + 2k\epsilon_p^2, N\delta_p + \delta_p')$-differential privacy.*

Note that the strong composition only provides (ϵ_p, δ_p)-DP, hence it usually works together with the Gaussian mechanism to provide strong privacy guarantees over multiple composites of the Gaussian mechanism.

Relation between differential privacy and stability

Note that the intuition of stability in algorithms is very similar to the intuition of differential privacy: we do not want the output of the algorithm to change by too much when we perturb the input by a little. Note that if an algorithm satisfies differential privacy then the outputs of two similar sample sets will likely to be the same. Hence the term $\ell(A(S), Z)) - \ell(A(S'), Z)$ will be small with a high probability because the output hypothesis $A(S)$ and $A(S')$ will be the same with a high probability. This shows that a differentially private algorithm is likely to be a stable algorithm too. On the other hand, [34] has shown that ERM algorithms with a large regularization parameter require less additive noise to guarantee differential privacy. This matches with our intuition - a stable algorithm is less sensitive to input changes and hence less noise is required to guarantee privacy.

2.4 Random matrix theory

In this section, we briefly summarise some key results in random matrix theory. In particular, we focus on the theoretical guarantees of random projections and the necessary conditions. We will present the classic Johnson-Lindenstrauss lemma and its variations to discuss their properties.

2.4.1 Random projections

Random projection has the idea that projecting a high dimensional data set onto much lower dimensions by a linear mapping in a way that preserves most of the structures of the original data set with high probability. Many theoretical works on ML such as error bounds have the problem that it scales with the dimension of the data set, which can be a problem when the dimension is large. Random projection provides a useful method of dimensionality reduction, the Johnson-Lindenstrauss lemma provides a precise statement that the structure of the projected data set is preserved with high probability and hence will not affect the accuracy by too much for distance-based algorithms.
Let $\mathcal{X} = \mathbb{R}^d$. Suppose we have two vectors $X_1, X_2 \in \mathcal{X}$ and a linear map $\Phi : \mathbb{R}^d \to \mathbb{R}^m$, then we say that Φ does not distort the distance between X_1 and X_2 if

$$\frac{\|\Phi(X_1) - \Phi(X_2)\|}{\|X_1 - X_2\|} \approx 1. \tag{2.17}$$

Equivalently we want a map Φ which satisfies the property that for any $X \in \mathbb{R}^d$, $\frac{\|\Phi(X)\|}{\|X\|}$ is close to 1. For a sample set $S \subset \mathcal{X}$, the Johnson-Lindenstrauss lemma gives a sufficient condition where the linear map does not distort the pairwise distance for all points in S:

Lemma 2.9 (Johnson-Lindenstrauss [94]). *Let S be a set of n points in $\mathbb{R}^d$, for any $0 < \epsilon_\Phi < 1$, let m be a positive integer that satisfies:*

$$m \geq \frac{24}{3\epsilon_\Phi^2 - 2\epsilon_\Phi^3} \log n. \tag{2.18}$$

Then there exists a map $\Phi : \mathbb{R}^d \to \mathbb{R}^m$ such that for all $X_1, X_2 \in S$

$$(1 - \epsilon_\Phi)\|X_1 - X_2\|^2 \leq \|\Phi(X_1) - \Phi(X_2)\|^2 \leq (1 + \epsilon_\Phi)\|X_1 - X_2\|^2. \tag{2.19}$$

Furthermore, this map can be found in polynomial time.

A key fact to note is that the random projection does not depend on the dimension of the original data set d and it even works for infinite dimensions. It only depends on the size of our data set and the level of accuracy we want to preserve. While this is a powerful result, there are two major limitations of this standard form of JL-lemma. Firstly, the projection dimension only depends on the distortion parameter and the number of samples, which means that the structure of the data is lost in the condition of the projection dimension. Secondly, since it depends on the number of samples, this is invalid if we have an infinite number of points. If we add a test point to our analysis, and since the test point is arbitrary, we will have a projection set of infinite size (assuming our sample space is non-finite). Hence we can not simply apply the JL lemma to our algorithm and perform the analysis. To address these issues, we employ the following generalized JL-lemma that uses the *Gaussian width* to measure the complexity of a set.

Definition 13 (Gaussian width). *The Gaussian width of a set $\mathcal{C} \subseteq \mathbb{R}^d$ is defined as*

$$w(\mathcal{C}) = \mathbb{E}_g \left[\sup_{X \in \mathcal{C}} \langle g, X \rangle \right], \quad \textit{where } g \sim N(0, I_d) \tag{2.20}$$

and I_d is the $d \times d$ identity matrix.

The Gaussian width is capable of capturing the complexity of the structure of a set. Hence we no longer require the set S to be finite, examples of the Gaussian width of some common sets are given in Tab. 2.1 [76].

Sets	Gaussian width
Unit ℓ_1 ball in $\mathbb{R}^d$	$\mathcal{O}(\sqrt{\log d})$
ℓ_p ball in $\mathbb{R}^d$ for $1 < p \leq \infty$	$\mathcal{O}(d^{1-1/p})$
Probability simplex in $\mathbb{R}^d$	$\mathcal{O}(\sqrt{\log d})$
Convex hull of n vectors with bounded ℓ_2-norm of c	$\mathcal{O}(c\sqrt{\log n})$
m-dimensional subspace of $\mathbb{R}^d$	$\mathcal{O}(\sqrt{m})$
$d_1 \times d_2$ matrices of rank at most r and unit Frobenius norm	$\mathcal{O}(\sqrt{r(d_1+d_2)})$
$d \times d$ matrices with unit nuclear norm	$\mathcal{O}(\sqrt{d})$

Table 2.1: Examples of Gaussian width of sets

Using the notion of Gaussian width, we have the following generalised JL-lemma. We specify the linear map Φ as a random matrix and we will denote $\mathbb{S}^d$ as the d-dimensional unit sphere.

Theorem 2.10 (Deviation of random matrices [89]). *Let Φ be an isotropic, sub-Gaussian random matrix, and D be a bounded subset of $\mathbb{R}^d$. For any $u \geq 0$ the event*

$$\sup_{X \in D \cap \mathbb{S}^{d-1}} \left| \|\Phi X\|_2 - \sqrt{m} \right| \leq CK^2 \left[w(D \cap \mathbb{S}^{d-1}) + u \right] \tag{2.21}$$

holds with probability at least $1 - \exp(-u^2)$. Here C is an absolute constant and K is the bounded Orlicz norm of the rows of Φ for the Orlicz function $\psi_2(x) = \exp(x^2) - 1$.

This result from [89] allows an alternative version of the JL-lemma. Suppose we want to project the points from the sample space $\mathcal{X}$ and preserve their pairwise distances. We take $D' = \mathcal{X} - \mathcal{X}$, and construct D by removing the zero vector and project all remaining vectors onto $\mathbb{S}^{d-1}$ by normalization. Now the set D contains all (normalized) pairwise distances of the points in $\mathcal{X}$. Since D belongs to the unit sphere we can apply Theorem 2.10 and obtain

$$\sup_{X \in D} \left| \frac{1}{\sqrt{m}} \|\Phi X\|_2 - 1 \right| \leq \frac{CK^2 \left[w(D) + u \right]}{\sqrt{m}} \tag{2.22}$$

with probability at least $1 - \exp(-u^2)$. Since elements of D is of the form $\frac{(X-X')}{\|X-X'\|_2}$, this is equivalent to: For all $X, X' \in \mathcal{X}$

$$(1-\epsilon_\Phi)\|X-X'\|_2 \leq \frac{1}{\sqrt{m}}\|\Phi(X-X')\|_2 \leq (1+\epsilon_\Phi)\|X-X'\|_2 \tag{2.23}$$

with probability at least $1-\delta_\Phi$, where $\epsilon_\Phi = \frac{CK^2\left[w(D)+\sqrt{\log(1/\delta_\Phi)}\right]}{\sqrt{m}}$.
Hence if we have $m \gg \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2$ then the projection preserves the pairwise distances of the points in $\mathcal{X}$ with probability at least $1-\delta_\Phi$.

Theorem 2.11 (Generalized Johnson-Lindenstrauss [89])**.** *Let Φ be an isotropic, subgaussian random matrix, and S be a subset of $\mathbb{R}^d$. Let $0 < \epsilon_\Phi < 1$ denote the distortion parameter, $0 < \delta_\Phi < 1$ and D be the set of normalized pairwise distances of S. Let*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2, \tag{2.24}$$

where C is an absolute constant, K is the bounded Orlicz norm of the rows of Φ and $w(D)$ is the Gaussian width of D. Then for all $X, X' \in S \subseteq \mathbb{R}^d$, the mapping $X \to \Phi X/\sqrt{m}$ preserves pairwise distances of D with probability at least $1-\delta_\Phi$:

$$(1-\epsilon_\Phi)\|X-X'\|_2 \leq \frac{1}{\sqrt{m}}\|\Phi(X-X')\|_2 \leq (1+\epsilon_\Phi)\|X-X'\|_2. \tag{2.25}$$

By using this generalized version, we can now perform random projections on sets of infinite size. Moreover, since Gaussian width is a generalized way of quantifying the complexity of the data set rather than just taking the cardinality of the set, we can have a more informative bound on the condition required for random projection. For instance, if the set D is an m-dimensional linear subspace embedded in the d-dimensional ambient space then its Gaussian width is proportional to m rather than d.

Theorem 2.12 (Gordon's Theorem [63])**.** *Let $m, d \in \mathbb{N}$, let $\Phi \in \mathbb{R}^{m\times d}$ be a random matrix with independent $\mathcal{N}(0, 1/m)$ entries. Let $D \subset \mathbb{S}^{d-1}$ be a subset of the unit sphere* *in d dimensions. If $m = \Theta(w(D)^2/\beta^2)$, then*

$$\mathbb{E}_\Phi \left[\sup_{\mathbf{x}\in B} \left| \|\Phi\mathbf{x}\|^2 - 1 \right| \right] \leq \beta, \tag{2.26}$$

where $w(D)$ is the Gaussian width of B and the expectation $\mathbb{E}_\Phi[\cdot]$ is over the randomness in Φ.

Gordon's Theorem is a key result to bound the expected norm of projected points with respect to the norm of original points. We note that the projection dimension must increase as β decreases, implying we need to project onto a higher dimension if we wish to decrease the distortion.

Chapter Three

Literature Review

This chapter reviews a summary of the related works of this thesis. We divide the literature into sub-categories as follows

- Section 3.1 covers some background literature in differentially private learning.
- Section 3.2 mostly covers the literature review related to Chapter 4 and 5;
- Section 3.3 mostly covers a literature review related to Chapter 6 and 7;
- Section 3.4 mostly covers a literature review related to Chapter 8.

We note that since the materials of different chapters are related, this division of sub-categories is not precise and will contain overlapping literature in different sections.

3.1 Differentially private learning

Differential privacy is originally introduced by Dwork in [50] and has been granted increasing attention over the past decade. Fundamental DP mechanisms such as the Laplace mechanism [50], the exponential mechanism [99] and the Gaussian mechanism [48] have then been introduced. The majority of the DP learning algorithms today use one or more of these DP mechanisms as a crucial step to guarantee privacy. Composition theorems were introduced by McSherry in [99] and [100] to compose multiple DP algorithms which allowed more complex DP algorithm design. Advanced composition theorems have been developed in [48] and extended by [74] to allow privacy loss aggregation to be reduced over adaptive compositions of queries. More recently, the work of [8] extends the composition theorem and considers the randomness of sub-sampling in privacy guarantee, which can provide a better privacy guarantee if the samples are randomly selected from a uniform distribution. An adaptive form of sensitivity measure is introduced by [106] that aimed to reduce noise required for privacy by analysing the sensitivity of functions dependent on training samples. Other variants of DP such as local differential privacy [142], Renyi DP [101] and zero-concentrated DP [29] have slightly different privacy definitions from the classic setting we consider in this thesis.

Many well-known learning algorithms such as kNN [64], decision trees [54, 71], empirical risk minimization [34, 132, 79, 131], linear models [33, 105], online learning [87] and unsupervised learning [123, 135] are studied in a differentially private setting. Refer to [62] for a review of machine learning algorithms with DP.

3.2 Non-parametric learning and random projections

Non-parametric learning algorithms are broad research topics that have long lasted and gained great popularity in theoretical study and practical applications. In particular, the k-Nearest Neighbour algorithm is a very well-known non-parametric algorithm introduced initially by [53] for over decades and still remains popular. The generalisation guarantees of kNN were studied by [41] for the case of $k = 1$ and proved that the 1-NN converges to twice the Bayes error as $n \to \infty$. After that, there is a vast amount of research on the kNN algorithm in both practical and theoretical fields. New techniques such as weighted kNN, ranked kNN, radius-based NN, etc, have been developed, see for example [15] for a survey on NN methods. Despite the success of the kNN algorithm, it is also well-known that kNN struggles with high-dimensional data for both sample complexity and run-time complexity due to the 'curse of dimensionality', which is a series of problems incurred by high-dimensional data [130]. Furthermore, from the result of [5], the curse of dimensionality is indeed a common problem for non-parametric learning algorithms including the histogram classifier that we will analyse in later chapters. The histogram is also a very common technique applied to different learning tasks. More commonly known in density estimation problems [57, 129, 83] and regression problems [65, 107] than classification in machine learning. In particular, [65] shows that random projection can also be used to boost the diversity of constructing histograms and achieved great regression performance by ensemble learning.

Random projection has been a widely used data compression technique to reduce dimensions, for example in image compressing [17, 60, 30], classification [114, 31, 73, 42], regression [23, 95, 128], clustering [52, 24, 143] and image registration [126]. Furthermore, random projection has been studied theoretically to quantify the class of sample structures in which RP can provide strong and useful guarantees. The well-known Johnson-Lindentrauss (JL) lemma [43] stated the projection dimension requirement where RP preserves pair-wise distance is dependent on a logarithmic factor of the number of samples. Further research by [80, 96, 85, 46] has improved and extended the result of classic JL-lemma. In particular, [89] uses the Gaussian width to measure the complexity of the input samples rather than the size which captures the geometric complexity, [37, 9] extends the guarantee to manifold data structures.

In relation to differentially private learning, Blocki *et al* [18] showed how the Johnson-Lindenstrauss transform could be used to publish privatized graphs. Another work by Kenthapadi *et al* [77] showed a method of computing pairwise distances by applying random projection first. They showed that by using this technique, the overall noise added to the distance matrix is smaller. Gursoy *et al*[64] proposed a private version of kNN that satisfies ϵ-differential privacy. First, they have proposed a private version of the radius-based neighbour (rN) algorithm which is an approximate alternative to the classic kNN. They have proposed an algorithm which converts kNN to rN, which allows kNN to be implemented in the DP setting using the private rN algorithm they have introduced. The work of [146] proposed the idea that we should only use a random subset of all samples to reduce privacy loss aggregation over multiple neighbour queries, however, this approach reduces the strength of the final privacy guarantee. Private histograms have been studied for the density estimation problem by [136] and histogram counting queries in [68]. Recently, Berrett *et al* [14] has shown the minimax rate of a similar histogram classifier under the local differential privacy setting, which is different from the setting we consider here. Besides modifying the learning algorithm to guarantee privacy, another approach for non-parametric private learning is by using a privatized data set. Works by [77, 124, 138] focus on private database release that can be used for statistical analysis. However, when using this privatized data approach to obtain DP, we lack knowledge of the utility guarantees of non-parametric classification.

3.3 Gradient descent methods and stability

The concept of stability has existed for over thirty years [44]. The introduction of uniform stability in [22] provides a modern stability framework that a portion of our work relies on. They have shown that the optimiser for regularized empirical loss minimization (ERM) is uniformly stable under some regularity assumptions. Further work by [116] studied the relation between stability, uniform convergence and learnability where they identify stability as the key condition for learnability. The notion of uniform stability has then been extended to study randomized algorithms in [51], where they derived stability bounds when randomness occurs in the algorithm. It has been shown in [36] that a fast iterative optimisation algorithm can not be too stable, and vice versa.

Gradient descent methods (GDM) has been one of the most popular methods of optimisation and is a very common approach to optimize neural networks, so there are a vast amount of variants of gradient descent [113, 20]. Besides the classic batch gradient descent ([75, 119]) which uses all samples in each iteration, mini-batch GDMs (e.g. [69, 78, 88, 40]) and stochastic GDMs (typically known as SGD, see [19, 66, 86, 12]) are the two most commonly used types of GDMs which only uses only a small subset or a single sample to estimate the gradient in each iteration. SGD has proven to have dimensionality-free error convergence (e.g. [66, 86]) and parallelisation (e.g. [147, 112]), which has made SGD a favourable optimisation algorithm for complex high-dimensional problems. In our case, we are interested in the generalisation performance and the stability bounds for compressed SGD in some learning procedures when it receives slightly different inputs. The optimisation performance has been analysed in [117] to achieve optimal convergence for its optimisation error with the appropriate choice of the learning rate parameter. The uniform stability of SGD is studied by [66] on convex, strongly convex and non-convex settings, which together provide the generalisation performance of SGD. Recently, [86] has derived generalisation bounds for SGD under a relaxed smoothness assumption where they have shown optimal bounds without smoothness with an appropriate choice of parameters. Mini-batch GDMs can have convergence bound which depends on the size of the chosen mini-batch, as demonstrated in [40].

In relation to differential privacy, this notion of privacy indeed has strong relations with stability and is discussed in [48], where they have stated that DP algorithms are also stable randomized algorithms. Several private SGD algorithms have been developed in the past decade [120, 1, 134] to train ML algorithms. In particular, the work of [120] provides DP guarantee by using noisy gradients, which adds an independent noise vector to each gradient computed. DP-SGD has been analysed in [134] which demonstrates a dimensionality dependence in its error bound due to the added noise. A DP-SGD approach by [28] uses Johnson-Lindenstrauss projections to estimate the norm of gradient, making SGD training more efficient in the private setting. The algorithm we are interested in is the compressed SGD proposed by [76] and reviewed in Section 6.2, where randomly compressed low-dimensional gradients are used for reducing the communication cost in transmitting the gradients. It uses the random projection technique [94, 43, 63] which is a popular dimensionality reduction tool that has been applied to many learning algorithms in various non-private contexts [118, 73] and in DP-related learning [141, 77]. Other works with the goal of reducing communication costs use some sparsified/quantized version of the gradient (e.g. [2, 3, 133, 1, 121]). The general idea used is to apply some encoding process on the gradient before transmission, hence reducing communication but can induce additional loss.

3.4 Private learning ensembles

Random forest [26] is one of the most commonly used learning ensembles today that has dated back over two decades ago. Random forest is a combination of multiple decision trees, which are non-parametric classifiers (or for regression) of their own. Non-private decision trees can have a variety of splitting rule that defines their structure, such as information gain [110], gain ratio [109] and Gini index [27]. Besides these greedy approaches that aim to compute the optimal split at each level, there are also trees that aim to take advantage of randomness to increase the diversity of trees [26] and save privacy budget. Random trees typically perform quite badly on their own, however, performance can be boosted when using a large combination of random trees [97].

Tree-based methods are certainly one of the most popular research topics in private learning, and they also divide into two main categories as conventional trees. Either approach with random trees (e.g. [71]) or approach with greedy trees (e.g. [58]). In [54] the authors proposed the use of local sensitivity [106] to reduce the randomness in greedy trees. The idea has been extended in [56] to improve private random trees using smooth sensitivity, which utilizes an upper bound on the local sensitivity. More recently, [139] proposed a greedy approach that takes advantage of a notion of smooth sensitivity with the exponential mechanism for both the Gini index and label prediction. Another DP forest algorithm by [111] considers (ϵ, δ)-differential privacy which is a weaker notion of the pure ϵ-differential privacy that we are concerned with here. As discussed in [56], it is possible to obtain high utility while guaranteeing pure differential privacy. Other differentially private algorithms such as [102] consider private data release – a different problem setting from what we study here.

The construction of a tree structure using a median split has long lasted in spatial trees [13] and private spatial decomposition [39]. For classification problems it was initially used by [25], it then gradually gained attention and was analyzed theoretically by [16].

A similar idea is also used in spatial decomposition such as kd-trees [39]. More recently, [81] extended the idea to a median-splitting random forest in the non-private setting. The authors demonstrated a random forest using a median-based splitting along with its theoretical analysis. Furthermore, recent work by [38] proposed a private random forest using median splits. However, their method uses a greedy approach to compute each split attribute and does not extend to semi-supervised learning. For private semi-supervised classifiers, the random forest by [71] can be extended to take advantage of unlabelled data [70]. Furthermore, work by [93] took advantage of unlabelled samples by providing predicted labels using a kNN classifier and then training a linear predictor using the predicted set. However, both methods apply only when the privacy of the features is not a concern. Other semi-supervised methods [108] make extra assumptions on the data set and differ from our setting here.

Chapter Four

Compressed Non-parametric Algorithms

In this chapter, we discuss two classic non-parametric algorithms - the k nearest neighbour (kNN) algorithm and the histogram predictor. We present the corresponding generalization analysis for each algorithm and find that both algorithms suffer from the “curse of dimensionality", i.e. their generalization errors increase rapidly as the dimension of the data increase. We show that this issue can be addressed with random projections and we provide excess generalization error bounds for projected case. For the simplicity of analysis, we assume that $\mathcal{X} = [0,1]^d$ and $\mathcal{Y} = \{0,1\}$. We can normalize our data points to satisfy this condition provided that the feature range is bounded. Let $\mathcal{D}$ be an unknown distribution over $\mathcal{X} \times \mathcal{Y}$, denote $\eta(X) = \mathbb{P}[Y=1|X]$ the conditional probability function (the regression function). Throughout this chapter, we will use the 0-1 loss function to measure the error of a hypothesis h

$$\ell_{0-1}(h(X),Y) = \mathbb{1}_{\{h(X)\neq Y\}}, \tag{4.1}$$

and the generalization loss with respect to the distribution $\mathcal{D}$ with the 0-1 loss is defined as

$$L_{\mathcal{D}}(h) = \mathop{\mathbb{E}}_{X,Y}[\ell_{0-1}(h(X),Y)] = \mathop{\mathbb{P}}_{X,Y}[h(X)\neq Y] \tag{4.2}$$

Recall that we use $\hat{h}$ to denote the output hypothesis of a learning algorithm, and we use $L_{\mathcal{D}}(f^*)$ to denote the Bayes optimal error or the "irreducible error". We provide a summary of the main results in this Chapter in Table 4.1 to provide a quick overview of the error convergence rates.

	kNN	Histogram	Comp-kNN	Comp-Histogram
Realizable	$\mathcal{O}\left((L\sqrt{d})^d n^{-1}\right)$ (Thm. 4.5)	$\mathcal{O}\left((L\sqrt{d})^d n^{-1}\right)$ (Thm. 4.16)	$\mathcal{O}\left((L\sqrt{d})^m n^{-1}\right)$ (Thm. 4.9)	$\mathcal{O}\left((L\sqrt{d})^m n^{-1}\right)$ (Thm. 4.21)
General	$\mathcal{O}\left(L\sqrt{d}n^{-\frac{1}{d+3}}\right)$ (Thm. 4.2)	$\mathcal{O}\left(L\sqrt{d}n^{-\frac{1}{d+2}}\right)$ (Thm. 4.12)	$\mathcal{O}\left(\left(\frac{L\sqrt{d}(1+\epsilon_\Phi)}{1-\epsilon_\Phi}\right) n^{-\frac{1}{m+3}}\right)$ (Thm. 4.8)	$\mathcal{O}\left(\left(\frac{L\sqrt{d}(1+\epsilon_\Phi)}{1-\epsilon_\Phi}\right) n^{-\frac{1}{m+2}}\right)$ (Thm. 4.20)

Table 4.1: A summary of the generalisation results, where d is the dimension and m is the projection dimension. Other parameters are as defined in previous Chapters. Note here we only consider the error convergence rate, in the general case the error will converge to the Bayes error which is non-zero. 'Comp' means 'compressed' denoting the algorithm with random projections.

4.1 The k-nearest neighbour algorithm

The k-nearest neighbour (k-NN) is a classic non-parametric algorithm that is widely used. It is simple to implement and very intuitive to understand while achieving great accuracy in classification and regression tasks. The intuition is that data points that are close together will likely to have the same labels. Hence for a given test point we look at the labels of k nearby points and make a prediction based on the information observed.

For example, for sample space $\mathcal{X} = [0,1]^d$ with the Euclidean metric, we will use the Euclidean distance $\|X - X'\|_2 = \sqrt{\sum_{i=1}^d (x_i - x_i')^2}$. Now suppose we have a training sample of size n, then for each X, let $\pi_1(X), \ldots, \pi_n(X)$ be orderings of $\{1, \ldots, n\}$ according to their corresponding distance to X. i.e. rearranged with the closest distance first. For an integer k, we have the kNN rule as follows:

1. Take a training sample of size n as inputs.

2. For every point $X \in \mathcal{X}$, output the majority label among $\{Y_{\pi_i(X)} : i \leq k\}$.

Theorem 4.1 (Generalization of kNN [115, Thm.19.5]). *Let $\mathcal{X} = [0,1]^d, \mathcal{Y} = \{0,1\}$, and $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$ for which the conditional probability function, $\eta(X) =$*

$\mathbb{P}[Y=1|X]$, is a L-Lipschitz function. Let $\hat{h}$ denote the result of applying the $k-NN$ rule to an i.i.d. sample set $S \sim \mathcal{D}^n$, where $k \geq 10$. Let f^* be the Bayes optimal hypothesis. Then,

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + (6L\sqrt{d}+k)n^{-1/(d+1)}.$$

Remark 1. *The L-Lipschitzness of the conditional probability function is key for its generalization analysis because kNN is a distance-based learning algorithm. The Lipschitz assumption guarantees that if the samples are close enough to each other, then their label is likely to be the same, hence we can predict the label of an unseen sample by looking at its "neighbours".*

From this generalization bound we can observe that the first term decreases as we increase k. However, increasing k will cause an increase in the latter complexity term. Hence k is the parameter that controls the bias-complexity trade-off in the k-NN algorithm. Theorem 4.1 shows that with sufficiently large n, the expected generalization error of k-NN is arbitrarily close to the Bayes optimal error using adaptive k increasing with n. In particular, we can find that the optimal choice of k that balances out both terms as shown in the following theorem:

Theorem 4.2 (Error convergence rate of kNN). *Let $k = \mathcal{O}(n^{2/(d+3)})$ for the kNN algorithm, then by the same condition as in Theorem 4.1 and assuming $L_{\mathcal{D}}(f^*) \neq 0$, we have*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] = L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(d+3)}}\right). \tag{4.3}$$

Proof. The proof of this theorem is based on modifying the proof of Thm. 4.1 and optimising the choice of k in the proof. By following the same steps in the proof of Thm. 4.1, we can obtain the following bound:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + 3LD\sqrt{d} + \frac{2k}{D^d n}, \tag{4.4}$$

where D is the maximum distance of a sample X and its k-th neighbour. From this inequality we can obtain Thm. 4.1 with $D = 2n^{-1/(d+1)}$. However, by considering the partial derivatives of (4.4) with respect to k and D, we found that the optimal choice of k and D that balance out all three terms are

$$D^* = \mathcal{O}\left(\frac{k}{n}\right)^{\frac{1}{d+1}}, \tag{4.5}$$

$$\begin{aligned} k^* &= \arg\min_k \left(\frac{1}{\sqrt{k}} + \left(\frac{k}{n}\right)^{\frac{1}{d+1}} + \left(\frac{k}{n}\right)^{\frac{1}{d+1}}\right) \\ &= \mathcal{O}\left(\frac{1}{n^{\frac{2}{d+3}}}\right). \end{aligned} \tag{4.6}$$

Substitute the choice of parameter above will give the desired result. □

However, we observe that in practical situations k is better chosen by cross-validation and tends to be smaller than the parameter proposed by its generalization bound. One reason for this is because Thm. 4.1 is an upper bound which may imply parameters that under-fits the data distribution.
The steps of the proof for Thm. 4.1 can be found in [115]. The outline of the proof falls into two key steps: 1) if the k-neighbours are “far away” then we assume the prediction is unlikely to be accurate and we just bound the probability of this case happening; 2) for the case where the k-neighbours are “close enough” within a bounded distance then we evaluate the error for this case. The following lemma bounds the probability of drawing a sample within a sparse region (i.e. regions where only a few sample is observed), hence implying that the k-neighbours are far away. Note that the “sparsity" here is controlled by the parameter k, and the probability increases as k increases.

Lemma 4.3 ([115])**.** *Let $C_1, \ldots, C_r$ be a collection of subsets of some domain set $\mathcal{X}$. Let S be a sequence of n points sampled i.i.d. according to some probability distribution $\mathcal{D}$ over $\mathcal{X}$. Then, for every $k \geq 2$, we have*

$$\mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S| < k} \mathbb{P}[C_i]\right] \leq \frac{2rk}{n}.$$

For the second case, we evaluate the error using the following auxiliary lemma, which bounds the empirical misclassification error by comparison with the underlying regression function.

Lemma 4.4 ([115]). *Let $k \geq 10$ and let $Y_1, \ldots, Y_k$ be independent Bernoulli random variables with $\mathbb{P}[Y_i = 1] = p_i$. Denote $p = \frac{1}{k}\sum_i p_i$ and $p' = \frac{1}{k}\sum_{i=1}^k Y_i$. We have*

$$\mathop{\mathbb{E}}_{Y_1,\ldots,Y_k} \mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p'>1/2]}] \leq \left(1 + \sqrt{\frac{8}{k}}\right) \mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p>1/2]}].$$

Lemma 4.3 and Lemma 4.4 will be useful in analysis for later chapters. For completeness, the proof of these lemmas is provided in the Appendix.

In the case where the Bayes error is zero ($L_{\mathcal{D}}(f^*) = 0$), the kNN algorithm can converge quicker by carefully tunning the hyper-parameters in the proof. The result is as follows.

Theorem 4.5. *Let $\hat{h}$ denote the result of applying the kNN rule to an i.i.d. sample set $S \sim \mathcal{D}^n$. Assume that η is a L-Lipschitz function and $L_{\mathcal{D}}(f^*) = 0$. Then for any fixed $k \geq 2$, we have*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \mathcal{O}\left(\frac{(2L\sqrt{d})^d}{n}\right) \tag{4.7}$$

Proof. The idea is to only look for neighbours which are close enough where we can guarantee the label is correct by the Lipschitz condition. Hence we only need to bound the probability where this close neighbourhood has no sample points at all. A detailed proof is deferred to the appendix A.1. □

4.1.1 Compressed kNN with random projections

In this section, we show that we can use the kNN algorithm in the compressed learning setting, where the algorithm only has access to the projected points. The Johnson-Lindentrauss (JL) lemma [43] says that if the projection dimension is greater than a constant multiple of $\log n$, then all pairwise distances are preserved. This suggests that if the condition of the JL-lemma is satisfied, then the error of the compressed kNN classifier will only change by a minor factor. However, there are two major limitations of the classical JL-lemma. Firstly, the projection dimension only depends on a distortion parameter and the number of samples, which means that the data structure is not exploited in the condition of the projection dimension. Secondly, since it only applies to finite sets, this is invalid if we have an infinite number of points. Hence we can not simply apply the JL lemma to our algorithm and perform the analysis. We use the generalization of the JL-lemma by Liaw *et al* [89] (Thm. 2.11) to address these issues.

By using this generalized version, we can now perform random projections on sets of infinite size. Moreover, since Gaussian width is a generalized way of quantifying the complexity of the data set rather than just taking the cardinality of the set, we can have a more informative bound on the condition required for random projection. For instance, if the set P is a m-dimensional linear subspace embedded in the d-dimensional ambient space then $w^2(P)$ is proportional to m rather than d. This also implies that if P is intrinsically d-dimensional then the error bound we derive here (Thm. 4.7) recovers, up to a constant factor, the corresponding error bound of the previous section (Thm. 4.1). Applying this result to our compressed learning setting, we found that with a small increase of the approximation error due to distortion, we reduce the dimension dependence of our error bound as the following theorem states. We will adapt some proof techniques from the generalization bound proved by [115] for the classic kNN algorithm and compare the result between them. Recall that Φ is an isotropic, sub-Gaussian $m \times d$ random matrix as in Thm. 2.11, and (X, Y) is a sample in S with features X and label Y. Before the training stage, we first randomly project the sample set $S \subset \mathbb{R}^d$ onto a lower dimensional space $\mathbb{R}^m$, we denote by S_Φ the set of projected sample points in $\mathbb{R}^m$:

$$S_\Phi = \{(\Phi X, Y) : (X, Y) \in S\}. \tag{4.8}$$

We will use the following supplementary lemma in our main proof, the proof of the lemma is deferred to the appendix.

Lemma 4.6. *For any fixed $p, p' \in [0,1]$ and $Y' \in \{0,1\}$. We have*

$$\mathop{\mathbb{P}}_{Y \sim p}[Y \neq Y'] \leq \mathop{\mathbb{P}}_{Y \sim p'}[Y \neq Y'] + |p - p'|.$$

Our main result for the compressed kNN is as follows.

Theorem 4.7 (Generalization of compressed kNN)**.** *Assume the conditional probability function, η, is a L-Lipschitz function. Let $\hat{h}_\Phi$ denote the result of applying the kNN rule to a projected sample set S_Φ, where $k \geq 10$. Let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. For any $0 < \epsilon_\Phi, \delta_\Phi < 1$, let m be a positive integer that satisfies:*

$$m \geq \Omega\left(\epsilon_\Phi^{-2} K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2\right). \tag{4.9}$$

Then we have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{kL(1+\epsilon_\Phi)\sqrt{d}}{n^{1/(m+1)}(1-\epsilon_\Phi)}\right). \tag{4.10}$$

Proof. Let $\pi_j(X)$ denote the index of the j-th nearest neighbour of the sample X in S. Fix some $b > 0$ and let $C_1, \ldots, C_r$ be the cover of the projected set $\Phi\mathcal{X} = \{\Phi X : X \in \mathcal{X}\}$ using axis-aligned boxes of side lengths of $1/b$ each. Let D_Φ denote the diameter of the projected set. For each $\Phi X, \Phi X' \in \Phi\mathcal{X}$ in the same box we have $\|\Phi X - \Phi X'\|_2 \leq D_\Phi / b$. Otherwise $\|\Phi X - \Phi X'\|_2 \leq D_\Phi$. Hence the expected generalization error is

$$\begin{aligned}\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] = \mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S_\Phi| < k} \mathbb{P}[C_i]\right] \cdot \mathop{\mathbb{P}}_{S,(X,Y)}\left[\hat{h}_\Phi(\Phi X) \neq Y \mid \forall j \in [k], \|\Phi X - \Phi X_{\pi_j(\Phi X)}\| \leq D_\Phi\right] \\ + \mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S_\Phi| \geq k} \mathbb{P}[C_i]\right] \cdot \mathop{\mathbb{P}}_{S,(X,Y)}\left[\hat{h}_\Phi(\Phi X) \neq Y \mid \forall j \in [k], \|\Phi X - \Phi X_{\pi_j(\Phi X)}\| \leq D_\Phi / b\right].\end{aligned}$$

By the law of total probability and Lemma 4.3 we obtain:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \frac{2rk}{n} + \max_{i:|C_i \cap S_\Phi| \geq k}\left\{\mathop{\mathbb{P}}_{S,(X,Y)}\left[\hat{h}_\Phi(\Phi X) \neq Y \mid \forall j \in [k], \|\Phi X - \Phi X_{\pi_j(\Phi X)}\| \leq D_\Phi / b\right]\right\}. \tag{4.11}$$

Note that the second term has expectation over the features and labels, we want to decompose the expectation separately for $X's$ and $Y's$. Fix $X_1, \ldots, X_n, X$, let $p = \frac{1}{k}\sum_{i=1}^{k} \eta(X_{\pi_i(\Phi X)})$ denote the average conditional probability function value of the k nearest neighbours in the projected space. W.l.o.g. (by symmetry) assume that $p \leq 1/2$. Hence by Lemma 4.4 we have:

$$\mathop{\mathbb{E}}_{Y_1,\ldots,Y_k}\left[\mathop{\mathbb{P}}_{Y\sim p}[\hat{h}_\Phi(\Phi X) \neq Y]\right] \leq \left(1+\sqrt{\frac{8}{k}}\right)\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p>1/2]}]. \tag{4.12}$$

Note that for $p \leq 1/2$ we have $\mathop{\mathbb{P}}_{Y\sim p}[\mathbb{1}_{[p>1/2]} \neq Y] = p = \min\{p, 1-p\}$, and hence by lemma 4.6:

we have

$$\min\{p, 1-p\} \leq \min\{\eta(X), 1-\eta(X)\} + |p-\eta(X)|.$$

Hence we obtain:

$$\begin{aligned}\mathop{\mathbb{E}}_{Y_1,\ldots,Y_j}\left[\mathop{\mathbb{P}}_{Y\sim p}[\hat{h}_\Phi(X) \neq Y]\right] &\leq \left(1+\sqrt{\frac{8}{k}}\right)\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p>1/2]}] \\ &\leq \left(1+\sqrt{\frac{8}{k}}\right)\left(\min\{\eta(X), 1-\eta(X)\} + |p-\eta(X)|\right).\end{aligned}$$

Combining inequality 4.12 and Lemma 4.6, we have

$$\begin{aligned}\mathop{\mathbb{E}}_{Y_1,\ldots,Y_j}\mathop{\mathbb{P}}_{Y\sim\eta(X)}[\hat{h}_\Phi(X) \neq Y] &\leq \mathop{\mathbb{E}}_{Y_1,\ldots,Y_j}\mathop{\mathbb{P}}_{Y\sim p}[\hat{h}_\Phi(X) \neq Y] + |p-\eta(X)| \\ &\leq \left(1+\sqrt{\frac{8}{k}}\right)\left(\min\{\eta(X), 1-\eta(X)\} + |p-\eta(X)|\right) + |p-\eta(X)| \\ &= \left(1+\sqrt{\frac{8}{k}}\right)\min\{\eta(X), 1-\eta(X)\} + \left(2+\sqrt{\frac{8}{k}}\right)|p-\eta(X)| \\ \text{(because } k \geq 10) \quad &\leq \left(1+\sqrt{\frac{8}{k}}\right)\min\{\eta(X), 1-\eta(X)\} + 3|p-\eta(X)|.\end{aligned}$$

Since p is just an average of the function value of η for the k neighbours of X, hence by using the fact that η is L-Lipschitz we have

$$|p - \eta(X)| \leq L\|X - X_{\pi_k(\Phi X)}\|_2. \tag{4.13}$$

Now applying the generalized JL-lemma (theorem 2.11) we have

$$\|X - X_{\pi_k(\Phi X)}\|_2 \leq \frac{\|\Phi X - \Phi X_{\pi_k(\Phi X)}\|}{(1-\epsilon_\Phi)} \leq \frac{D_\Phi}{b(1-\epsilon_\Phi)}, \tag{4.14}$$

where D_Φ is the diameter of the projected sample set. Since our original sample space is $[0,1]^d$ hence $\|X\|_2^2 \leq d$, by the JL-lemma we have $D_\Phi \leq (1+\epsilon_\Phi)\sqrt{d}$. Note that the error of the Bayes optimal classifier is

$$L_{\mathcal{D}}(f^*) = \mathop{\mathbb{E}}_{X}[\min\{\eta(X), 1-\eta(X)\}].$$

Combining our results and taking expectations over the randomness of the features we obtain:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \frac{3L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \frac{2rk}{n}. \tag{4.15}$$

In particular, since $r = b^d$ letting $b = O(n^{-\frac{1}{m+1}})$ results

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{kL(1+\epsilon_\Phi)\sqrt{d}}{n^{1/(m+1)}(1-\epsilon_\Phi)}\right). \tag{4.16}$$

□

By a similar analysis as in theorem 4.2, we obtain the following immediate result:

Theorem 4.8. *Let $k = \mathcal{O}(n^{2/(m+3)})$ for the kNN algorithm, then by the same condition as in Theorem 4.7 and assuming $L_{\mathcal{D}}(f^*) \neq 0$, we have*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] = L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(m+3)}}\right). \tag{4.17}$$

Proof. Can be easily verified using similar procedure in theorem 4.2 and theorem 4.7. □

Remark 2. *The special case where $k = 1$ is done in [73] and Theorem 4.7 provides the result for general k of the compressed kNN. The 1-NN algorithm with projected data sets converges to two times the Bayes error as for the classical NN algorithm. In contrast, Theorem 4.7 implies that the compressed kNN will converge to the Bayes error given that k increases with n.*

Remark 3. *The dimension dependence $\sqrt{d}$ that remains in Theorem 4.7 comes from the maximum norm of the samples in the original space. This can be removed if given the feature domain is bounded and we can scale the samples beforehand so that the norm of every sample has norm ≤ 1. The resulting error bound will then be independent of the original dimension.*

From Theorem 4.7 we see that as k and n approach to infinity, the error of the compressed kNN classifier converges to the Bayes optimal error as the regular kNN. The dimensionality dependence has been reduced to m instead of d, where m is dependent on the Gaussian width of the feature space. As an example, if the samples are a subset of the ℓ_1-ball, then the dimensionality dependence will be of order $\log d$ instead of d. The trade-off for a better dimensionality dependence is the additional multiplicative factor on ϵ_Φ. However ϵ_Φ can usually kept quite small if the intrinsic dimension of the data is small.

Theorem 4.9. *Let $\hat{h}_\Phi$ be the result of applying the kNN rule on the training set S_Φ. Assume that $\eta(X)$ is an L-Lipschitz function and let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. Then for any $0 < \delta_\Phi < 1$, $0 < \epsilon_\Phi \leq 1/2$ let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)} \right]^2 . \tag{4.18}$$

Then for any $k \geq 2$, we have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \mathcal{O}\left(\frac{L^m d^{m/2}}{n}\right). \tag{4.19}$$

Proof. See appendix A.1. □

We observed a similar result for the compressed kNN in the realisable case. The rate for error convergence is significantly improved for the realisable case as in the non-compressed setting. We also observe a dimensionality reduction in the error term. However, since in the realisable case the dimensionality dependence is in the exponential of the Lipschitz constant and the original dimension d only, we do not obtain an improvement in the rate with respect to the sample set size n.

4.2 Histogram Classifier

In this section, we introduce the histogram classifier and analyse its properties. The histogram classifier is a non-parametric classifier with similar properties to the kNN classifier. The main advantage of the histogram classifier over kNN is that it is much more computationally efficient because it does not need to calculate the pairwise distances in training sets. Furthermore, histogram only takes a small amount of memory to store and kNN needs to memorise all training data points, which is problematic when the training data gets large. The histogram classifier is a computationally lighter alternative to the kNN algorithm for large data sets, and it can achieve the optimal generalization convergence rate as we will prove in later sections.

We now formally describe the histogram classifier for binary classification as follows. Let $\mathcal{X} = [0,1]^d$ be the input domain[1] (or feature space), and $\mathcal{Y} = \{0,1\}$ the label set. For some positive integer $b \in \mathbb{Z}_+$, we partition the feature space by dividing the interval $[0,1]$ into b evenly spaced intervals for each dimension. This creates $r := b^d$ hyper-cube cells each having side lengths $1/b$. We denote these cells by $C_1, \ldots, C_r$. Let S be a sample

[1]E.g. by transforming the data into the unit hypercube.

set drawn i.i.d. from an unknown distribution $\mathcal{D}$, and assume w.l.o.g. that $|S| = n \in \mathbb{N}$. Each observation in S has the form $Z_i = (X_i, Y_i) \in \mathcal{X} \times \mathcal{Y} = \mathcal{Z}$ for all $i \in [n]$ where $[n] = \{1, \ldots, n\}$ is the indexing set. For an unseen point X, let $i(X)$ denote the index of the cell that contains X, i.e. $X \in C_{i(X)}$. We define the histogram classifier based on this setting as follows

$$\hat{h}(X) = \mathbb{1}_{\{\hat{\eta}(X) \geq \frac{1}{2}\}}, \quad \hat{\eta}(X) = \frac{\sum_{i\in[n]} \mathbb{1}_{\{X_i \in C_{i(X)}\}} Y_i}{\sum_{i\in[n]} \mathbb{1}_{\{X_i \in C_{i(X)}\}}}, \tag{4.20}$$

where $\mathbb{1}_{\{\cdot\}}$ is the indicator function that outputs 1 if the condition in $\{\cdot\}$ is satisfied and 0 otherwise. We note that for any cell C_j, we have $\hat{\eta}(X) = \hat{\eta}(X')$ for all $X, X' \in C_j$. Hence we denote for simplicity $\hat{\eta}_j = \hat{\eta}(X)$ for any X in C_j. Note that $\hat{\eta}_j$ is not dependent on X but on the cell index j only. The detailed procedure of the histogram predictor is described in Alg. 1.

Algorithm 1 Histogram Predictor

```
 1: procedure TrainHistogram($S$)
 2:     for each $i$ in the index set $[n] = \{1, \ldots, n\}$ do
 3:         record label $Y_i$ in cell $C_{i(X)}$
 4:     end for
 5:     for each non-empty cell $C_j$ do
 6:         compute $\hat{\eta}_j$ and set label of $C_j$ to $\mathbb{1}_{\{\hat{\eta}(X) \geq \frac{1}{2}\}}$ cf. (4.20)
 7:     end for
 8: end procedure
 9: procedure HistogramPredict($X$)
10:     if $C_{i(X)}$ is non-empty then
11:         return label of $C_{i(X)}$
12:     else
13:         make a random prediction
14:     end if
15: end procedure
```

Note that for a fixed b, we obtain a finite hypothesis class $\mathcal{H}_b$ of size 2^r (each cell either has label 1 or 0), where each hypothesis $h \in \mathcal{H}_b$ is defined by the labelling in each cell. This allows us to use techniques from finite hypothesis classes as we will demonstrate in later sections. From equation (4.20) we see the classification rule is based on the ratio $\hat{\eta}$, ideally we would prefer a large number of points in each cell so that this empirical ratio is estimated accurately. However with a finite number of points that usually requires us to choose a small b, which may under-fit the data. Hence the parameter b controls the bias-variance trade-off in the histogram classifier.

For a fixed b, i.e. in the class $\mathcal{H}_b$, the optimal histogram classifier is denoted by h^*_b. We omit the index when b is clear from the context. The classification rule of $h^* = h^*_b$ is defined as follows

$$h^*(X) = \mathbb{1}_{\{\eta^*_b(X)\geq\frac{1}{2}\}}, \quad \eta^*_b(X) = \mathop{\mathbb{E}}_{X'\in C_{i(X)}}[\eta(X')] = \mathbb{E}_{X'}\left[\mathbb{P}[Y=1|X'\in C_{i(X)}]\right]. \quad (4.21)$$

4.3 Generalization of the histogram classifier

In this section, we present the generalization bound of the histogram classifier. We show that it is universally consistent, i.e. it converges to the Bayes error as $n\to\infty$ for any distribution $\mathcal{D}$, and we show that under the Lipschitz assumption of the conditional probability function η, it attains the optimal convergence rate of non-parametric classification.

In the analysis of this section, recall that $L_{\mathcal{D}}(f^*)$ denotes the Bayes error (or the irreducible error). Our goal is to show that the histogram classifier converges to the Bayes error with optimal rates (under the same condition). Recall $\eta(X) := \mathbb{P}[Y=1|X]$ denotes the regression function. We assume that $\eta:\mathcal{X}\to\mathcal{Y}$ is L-Lipschitz with respect to the Euclidean norm over $\mathcal{X}$. This assumption is believed to hold in many practical cases, as the Lipschitz constant L can be large and our convergence analysis will still hold. However, a larger Lipschitz constant means that the samples in different classes are less distinguishable, i.e. the samples from different classes are likely to overlap to a larger extent. This implies that the learning problem is more difficult. If the Lipschitz condition does not hold, then there are samples in $\mathcal{X}$ where the label distribution of these samples is different despite the samples being extremely close. That would mean that any distance-based similarity between the points can fail and makes the problem very difficult to solve, which is a comparatively rare situation given the success of distance-based learning algorithms like kNN.

In this section, we prove the generalization error of the histogram classifier using two different approaches. We will discuss the main advantages and disadvantages of each approach and explain how they work. While we are using different approaches for generalization analysis, the general setting and assumptions we make will be identical.

4.3.1 Generalization via finite hypothesis class

In this section, we give an approach to bound the generalization error of $\hat{h}$ using the fact that $\mathcal{H}_b$ is a finite hypothesis class, which implies that it has PAC learnability (see e.g. [115] for existing theoretical results with finite hypothesis class). As pointed out previously, the parameter b controls the bias-variance trade-off of the histogram. As b increases, we obtain finer cells of the histogram which better fits the training set but we run the risk of over-fitting. Furthermore, the size of the hypothesis class $\mathcal{H}_b$ increases as b increases due to the increasing number of cells, which gives a more complex learning problem. The following generalization error bound demonstrates this argument:

Theorem 4.10. *Let $\hat{h}$ be the histogram classifier as defined in* (4.20)*, and assume that $\eta(X)$ is a L-Lipschitz function. Then we have, for any fixed $b \in \mathbb{N}$:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2(b^d+1)}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \tag{4.22}$$

Proof. There are $r = b^d$ cells in total, and for each cell, we can either choose the label to be 1 or 0. Hence the class of hypothesis defined by the cells have size 2^r. Let h^* be the hypothesis in class $\mathcal{H}_b$ with minimal generalization error. Then by the generalization bound of agnostic PAC learning for finite hypothesis class (Thm. 2.1) we have for any $\epsilon > 0$:

$$\mathop{\mathbb{P}}_{S}[L_{\mathcal{D}}(\hat{h}) > L_{\mathcal{D}}(h^*) + \epsilon] \leq \frac{2^{r+1}}{e^{n\epsilon^2/2}}. \tag{4.23}$$

Rearranging this we see that for $1 > \delta > 0$, we have with probability at least $1 - \delta$:

$$\begin{aligned} L_{\mathcal{D}}(\hat{h}) &\le L_{\mathcal{D}}(h^*) + \sqrt{\frac{2\log(2|\mathcal{H}_b|/\delta)}{n}} \\ &\le L_{\mathcal{D}}(h^*) + \sqrt{\frac{2\log(2|\mathcal{H}_b|)}{n}} + \sqrt{\frac{2\log(1/\delta)}{n}}. \end{aligned} \tag{4.24}$$

Rearranging, for $\epsilon > 0$,

$$\mathop{\mathbb{P}}_{S}\left[L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*) - \sqrt{\frac{2\log(2|\mathcal{H}_b|)}{n}} > \epsilon\right] \le \exp\left(-\epsilon^2 n/2\right), \tag{4.25}$$

which implies that:

$$\begin{aligned} \mathop{\mathbb{E}}_{S}\left[L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*) - \sqrt{\frac{2\log(2|\mathcal{H}_b|)}{n}}\right] &= \int_0^\infty \mathop{\mathbb{P}}_{S}\left[L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*) - \sqrt{\frac{2\log(2|\mathcal{H}_b|)}{n}} > \epsilon\right] \mathrm{d}\epsilon \\ &\le \int_0^\infty \exp\left(-\epsilon^2 n/2\right) \mathrm{d}\epsilon \\ &= \int_0^\infty \frac{\exp\left(-u^2\right)}{\sqrt{n/2}} \mathrm{d}u \\ &= \frac{\sqrt{\pi}}{\sqrt{2n}}, \end{aligned}$$

where the third equality is by using integration with a substitution $u = \sqrt{n/2}\epsilon$.

From the above we obtain

$$\begin{aligned} \mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] &\le L_{\mathcal{D}}(h^*) + \sqrt{\frac{2\log(2^{r+1})}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \\ &\le L_{\mathcal{D}}(h^*) + \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \end{aligned} \tag{4.26}$$

Now we evaluate the error gap between $h^* = h_b^*$ and f^*. Recall that

$$h^*(X) = \mathbb{1}_{[\eta_b^*(X) \ge 1/2]} \text{ where } \eta^*(X) = \mathbb{E}_{X'}\left[\mathbb{P}[Y = 1 | X' \in C_{i(X)}]\right].$$

This means that the probability of h^* giving the wrong prediction on a sample point $X' \in C_{i(X)}$ is the probability that Y differ from the majority label in the hypercube $C_{i(X)}$

given $X' \in C_{i(X)}$. Explicitly, this is $\min\{\eta^*(X), 1-\eta^*(X)\}$. Hence we have

$$\begin{aligned} L_{\mathcal{D}}(h^*) &= \mathop{\mathbb{E}}_{X}[\min\{\eta^*(X), 1-\eta^*(X)\}] \\ &\leq L_{\mathcal{D}}(f^*) + \mathop{\mathbb{E}}_{X}[|\eta(X) - \eta^*(X)|]. \end{aligned} \tag{4.27}$$

The second term is the expectation of the difference between $\mathbb{P}[Y=1|X]$ and $\mathbb{P}[Y=1|X']$ where $X' \in C_{i(X)}$. We apply the L-Lipschitz assumption of η and use that fact that for any X, X' in the same cell, $\|X - X'\|_2 \leq \frac{\sqrt{d}}{b}$. Hence we obtain the following:

$$\begin{aligned} L_{\mathcal{D}}(h^*) &\leq L_{\mathcal{D}}(f^*) + L\mathop{\mathbb{E}}_{X}[\|X - X'\|_2] \\ &\leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b}. \end{aligned} \tag{4.28}$$

Combining results in equation (4.26) and (4.28) we conclude that:

$$\begin{aligned} \mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] &\leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \\ &\leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2(r+1)}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}, \end{aligned} \tag{4.29}$$

where $r = b^d$. □

From the proof of Theorem 4.10 we also have the following high probability version as an immediate result.

Theorem 4.11. *Let $\hat{h}$ be the histogram classifier defined as in* (4.20)*, and assume that $\eta(X)$ is a L-Lipschitz function. Then, for any fixed $b \in \mathbb{N}$ and $0 < \delta < 1$, we have with probability at least $1-\delta$:*

$$L_{\mathcal{D}}(\hat{h}) \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2(b^d+1)}{n}} + \sqrt{\frac{2\log(1/\delta)}{n}}. \tag{4.30}$$

Proof. Combine equation 4.24 and 4.28. □

Remark 4. *From the generalization analysis in Theorem 4.10 we immediately observe* *that the histogram classifier is universally consistent. Indeed, as we increase the parameter b and n tends to infinity we get that $L_{\mathcal{D}}(\hat{h})$ converges to the irreducible error $L_{\mathcal{D}}(f^*)$.*

Remark 5. *The parameter b for the histogram classifier controls the trade-off between the approximation term and the estimation term, which should be chosen carefully for best performance in practice. This has a similar role as the parameter k for the kNN algorithm and can be chosen by cross-validation in practice. Increasing b here has a similar effect as decreasing k in the kNN algorithm as the histogram will have access to fewer samples in each cell and vice versa.*

So far we have derived a generalization bound on our classifier $\hat{h}$ relative to the Bayes error. Since the parameter b is important both in practice and in theory to control the bias-variance trade-off, it is important to choose a suitable parameter b. While in practice we can always choose b by validation or similar techniques, we can also compute the parameter b that minimises its excess risk bound. The parameter suggested by the theoretical result will give insights into the optimal order of b, and give us the convergence rate for histogram classification, as captured in the following theorem.

Theorem 4.12 (Error convergence of histogram)**.** *Let $\hat{h}$ be the histogram classifier defined in* (4.20)*; assume that $\eta(X)$ is a L-Lipschitz function. Choosing $b = \left(\frac{2L^2 n}{d}\right)^{\frac{1}{d+2}}$, then we have:*

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h})] \leq L_{\mathcal{D}}(f^*) + \frac{Ld^{\frac{d+3}{2d+4}}}{n^{\frac{1}{d+2}}} + L\sqrt{\frac{2}{d^{\frac{d}{d+2}}}\frac{1}{n^{\frac{1}{d+2}}}} + \frac{\sqrt{\pi}+2L}{\sqrt{2n}} \tag{4.31}$$

$$= L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{\frac{1}{d+2}}}\right). \tag{4.32}$$

Proof. By the result of Theorem 4.10 we are trying to find b^* such that:

$$b^* = \arg\min_b \left\{ L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2(b^d+1)}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \right\}. \tag{4.33}$$

After dropping the terms that have no dependence on b, this is equivalent to:

$$b^* = \arg\min_b \left\{ \frac{\sqrt{d}L}{b} + \sqrt{\frac{2b^d}{n}} \right\}. \tag{4.34}$$

To find such b, we can take the derivative of (4.34) with respect to b and equate it to zero to find the minimum. Hence we have

$$\sqrt{\frac{2}{n}} \frac{d}{2} b^{d/2-1} - \frac{L\sqrt{d}}{b^2} = 0. \tag{4.35}$$

Since by algorithm construction $1/b$ is the width of each cell and $b \neq 0$, by solving the equation for b, we obtain: $b = \left(\frac{2L^2n}{d}\right)^{\frac{1}{d+2}}$. We substitute this into (4.10) and obtain:

$$\begin{aligned}\mathbb{E}_S[L_\mathcal{D}(\hat{h})] &\leq L_\mathcal{D}(f^*) + L\sqrt{d}\left(\frac{d}{2L^2n}\right)^{\frac{1}{d+2}} + \sqrt{\frac{2}{n}}\sqrt{\left(\frac{2L^2n}{d}\right)^{\frac{d}{d+2}} + 1} + \sqrt{\frac{\pi}{2n}} \\ &\leq L_\mathcal{D}(f^*) + \frac{Ld^{\frac{d+3}{2d+4}}}{n^{\frac{1}{d+2}}} + L\sqrt{\frac{2}{d^{\frac{d}{d+2}}} \frac{1}{n^{\frac{1}{d+2}}}} + \frac{\sqrt{\pi} + 2L}{\sqrt{2n}}. \end{aligned} \tag{4.36}$$

□

Remark 6. *The rate of order $\mathcal{O}(n^{-1/(d+2)})$ we obtained from Theorem 4.12 is optimal for non-parametric classifiers under the Lipschitz η assumption, up to constant factors [5]. Our setting only assumes the Lipschitzness of the conditional probability function η. The optimal rate under the same setting for non-parametric classification is known to be $\mathcal{O}(n^{-1/(d+2)})$ and can be found in [5, Thm. 4.1].*

Despite Theorem 4.12 have already achieved optimal convergence rate under the same condition, the result does not provide any information on whether faster error convergence is possible for easier learning problems. More precisely, can the convergence rate be quicker when the Bayes error is zero or very small? We investigate this question in the following section.

4.3.2 Generalization by case analysis

In this section, we consider an alternative approach to analysing the histogram classifier. As we shall see, this will turn out to have an advantage in the special case when the Bayes error of the learning problem is zero or very small, while being suboptimal in the general regime.

Due to the nature of the histogram classifier, we can derive a generalization bound by an approach of splitting the cells of the histogram into two groups. The intuition is that we consider the first case as we have an insufficient number of points in a cell hence we assume that a bad prediction is likely to be made in this case, and vice versa for the second case. We use a hyper-parameter k to determine how many points in a cell will be considered "sufficient".
Let us fix some $k \in \mathbb{N}$, and consider the case where the cell contains more than k sample points and the case where a cell contains less than k sample points. We have the following Theorem:

Theorem 4.13. *Let $\hat{h}, \hat{\eta}$ be as defined in* (4.20)*, and assume that $\eta(X)$ is an L-Lipschitz function. Then for any fixed $10 \leq k \in \mathbb{N}, b \in \mathbb{N}$, we have:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \left(1 + \sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \frac{3\sqrt{d}L}{b} + \frac{2kb^d}{n}. \tag{4.37}$$

We will use Hoeffding's inequality in our proof.

Lemma 4.14 (Hoeffding's Inequality)**.** *Let $\theta_1, \ldots, \theta_n$ be a sequence of independent random variables and let $\bar{\theta} = \frac{1}{n}\sum_{i=1}^{n} \theta_i$. Assume that $\mathbb{E}[\bar{\theta}] = \mu$ and $\mathbb{P}[a \leq \theta_i \leq b] = 1$ for every i. Then, for any $\epsilon > 0$*

$$\mathbb{P}\left[\left|\frac{1}{n}\sum_{i=1}^{n} \theta_i - \mu\right| > \epsilon\right] \leq 2\exp\left(-2n\epsilon^2/(b-a)^2\right). \tag{4.38}$$

Proof of Theorem 4.13. Let $\hat{h}$ be our output hypothesis, let $Z_i = (X_i, Y_i), i \in [n]$ be the set of points in S, and let (X,Y) be the additional point we have drawn from $\mathcal{X}\times\mathcal{Y}$. We have:

$$\begin{aligned}
\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] &= \mathop{\mathbb{E}}_{X_1,\dots,X_n,X}\left[\mathop{\mathbb{E}}_{Y_1,\dots,Y_n,Y}[L_{\mathcal{D}}(\hat{h})|X_1,\dots,X_n,X]\right] \\
&= \mathop{\mathbb{E}}_{X_1,\dots,X_n}\left[\sum_{i:|C_i\cap S|<k}\mathbb{P}[C_i]\right]\cdot \mathop{\mathbb{E}}_{X_1,\dots,X_n,X}\left[\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{h}(X)\neq Y|\ |C_{i(X)}\cap S|<k]\right] \\
&+ \mathop{\mathbb{E}}_{X_1,\dots,X_n}\left[\sum_{i:|C_i\cap S|\geq k}\mathbb{P}[C_i]\right]\cdot \mathop{\mathbb{E}}_{X_1,\dots,X_n,X}\left[\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{h}(X)\neq Y|\ |C_{i(X)}\cap S|\geq k]\right] \\
&\leq \frac{2rk}{n} + \mathop{\mathbb{E}}_{X_1,\dots,X_n,X}\left[\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{h}(X)\neq Y|\ |C_{i(X)}\cap S|\geq k]\right],
\end{aligned} \tag{4.39}$$

where the last inequality follows from Lemma 4.3 and using the trivial bound of probabilities ≤ 1.

Let $\hat{Y} := \hat{h}(X)$, and fix S and X such that $|C_{i(X)}\cap S|\geq k$. Let $p=p(X)=\frac{\sum_{i\in[n]}\mathbb{1}_{[X_i\in C_{i(X)}]}\eta(X_i)}{\sum_{i\in[n]}\mathbb{1}_{[X_i\in C_{i(X)}]}}$ and $|C_{i(X)}|=\sum_{i\in[n]}\mathbb{1}_{[X_i\in C_{i(X)}]}$. Note that from lemma 4.6 we have

$$\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{Y}\neq Y] \leq \mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim p\\ i\in[n]}}[\hat{Y}\neq Y] + |p-\eta(X)|. \tag{4.40}$$

Since we fixed X, so $|C_{i(X)}|$ is also fixed, we apply Lemma 4.4 and have

$$\begin{aligned}
\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim p\\ i\in[n]}}[\hat{Y}\neq Y] &\leq \left(1+\sqrt{\frac{8}{|C_{i(X)}|}}\right)\mathop{\mathbb{P}}_{Y\sim p}[Y\neq \mathbb{1}_{[p\geq 1/2]}] \\
&\leq \left(1+\sqrt{\frac{8}{k}}\right)\mathop{\mathbb{P}}_{Y\sim p}[Y\neq \mathbb{1}_{[p\geq 1/2]}].
\end{aligned} \tag{4.41}$$

W.l.o.g. assume $p\leq 1/2$, we also have

$$\mathop{\mathbb{P}}_{Y\sim p}[Y\neq \mathbb{1}_{[p\geq 1/2]}] = p = \min\{p,1-p\} \leq \min\{\eta(X),1-\eta(X)\} + |p-\eta(X)| \tag{4.42}$$

Taking expectations of (4.40) and combining our results we have

$$\begin{aligned}\mathbb{E}_S[L_{\mathcal{D}}(\hat{h})] &\leq \frac{2rk}{n} + \left(1+\sqrt{\frac{8}{k}}\right)[L_{\mathcal{D}}(f^*) + \mathbb{E}[|p-\eta(X)|]] + \mathbb{E}[|p-\eta(X)|] \\ &\leq \frac{2rk}{n} + \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + 3\mathbb{E}[|p-\eta(X)|]. \qquad (4.43)\end{aligned}$$

Now consider $\mathbb{E}[|p-\eta(X)|]$, we can use the assumption that $\eta(X)$ is L-Lipschitz here and conclude $\mathbb{E}[|p-\eta(X)|] \leq L\sqrt{d}/b$. Hence we conclude that:

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h})] \leq \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \frac{3L\sqrt{d}}{b} + \frac{2kb^d}{n}. \qquad (4.44)$$

□

Unfortunately, optimising the choice of k and b over the result obtained in Thm. 4.13 leads to a suboptimal rate for nonzero Bayes error, given in the following theorem for completeness. However, in Theorem 4.16 the advantage in zero (or small) Bayes error case will become apparent.

Theorem 4.15. *Let $\hat{h}, \hat{\eta}$ be as defined in* (4.20)*, and assume that $\eta(X)$ is a L-Lipschitz function. Then we have:*

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h})] \leq L_{\mathcal{D}}(f^*) + \frac{11L + 4Ld}{4L^{\frac{3}{d+3}} n^{\frac{1}{d+3}} d^{\frac{d}{d+3}}} \qquad (4.45)$$

Proof. To optimise our result in Theorem 4.13 we want to work out the following, given any fixed b, we have:

$$\begin{aligned} k_b^* &= \arg\min_k \left\{ \frac{2kb^d}{n} + \frac{\sqrt{\pi}}{\sqrt{2k}} + \frac{L}{b} \right\}, \\ b^* &= \arg\min_b \left\{ \frac{2k_b^* b^d}{n} + \frac{\sqrt{\pi}}{\sqrt{2k_b^*}} + \frac{L}{b} \right\}. \end{aligned} \qquad (4.46)$$

Notice that our bound holds for all $k \geq 2$ and all $b \geq 1$. Due to the complexity of computing and representing the optimal k and b in this case, we will try to work out an approximate of k^* and b^* instead.

Let us approximate k^* first, taking the derivative of (4.46) (w.r.t. k) and equating to zero we have

$$\frac{2b^d}{n} - \frac{\sqrt{\pi}}{(2k)^{3/2}} = 0 \tag{4.47}$$

Hence

$$\begin{aligned}
&\Rightarrow 2b^d(2k)^{3/2} = n\sqrt{\pi} \\
&\Rightarrow 32b^{2d}k^3 = n^2\pi \\
&\Rightarrow k = \left(\frac{n^2\pi}{32b^{2d}}\right)^{1/3} \approx \left(\frac{n}{b^d}\right)^{2/3} \cdot \frac{6}{13} = \tilde{k}.
\end{aligned} \tag{4.48}$$

Substitute $\tilde{k}$ into the second equation of (4.46) we get

$$\begin{aligned}
\frac{\frac{12}{13}\left(\frac{n}{b^d}\right)^{2/3} b^d}{n} + \frac{\sqrt{\pi}}{\sqrt{\frac{12}{13}\left(\frac{n}{b^d}\right)^{2/3}}} + \frac{L}{b} &= \frac{12n^{2/3}b^{d/3}}{13n} + \frac{\sqrt{\pi}}{\sqrt{\frac{12}{13}}\left(\frac{n}{b^d}\right)^{1/3}} + \frac{L}{b} \\
&= \frac{12b^{d/3}}{13n^{1/3}} + \frac{\sqrt{13\pi}b^{d/3}}{\sqrt{12}n^{1/3}} + \frac{L}{b} \\
&= \frac{b^{d/3}}{n^{1/3}}\left(\frac{12}{13} + \sqrt{\frac{13\pi}{12}}\right) + \frac{L}{b} \\
&\approx \frac{11b^{d/3}}{4n^{1/3}} + \frac{L}{b}.
\end{aligned} \tag{4.49}$$

Now we find an approximate value for b^* by finding

$$\tilde{b} = \arg\min_b \frac{11b^{d/3}}{4n^{1/3}} + \frac{L}{b}. \tag{4.50}$$

Differentiating w.r.t. b and equating zero we have:

$$\begin{aligned}
&\frac{11db^{\frac{d-3}{3}}}{12n^{1/3}} - \frac{L}{b^2} = 0 \\
\Rightarrow &11db^{\frac{d+3}{3}} = 12Ln^{1/3} \\
\Rightarrow &b = \left(\frac{12Ln^{1/3}}{11d}\right)^{\frac{3}{d+3}} \approx \left(\frac{L^3 n}{d^3}\right)^{\frac{1}{d+3}} = \tilde{b}.
\end{aligned} \tag{4.51}$$

We substitute our results back to (4.37) and obtain the following:

$$\begin{aligned}
\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] &\leq L_{\mathcal{D}}(f^*) + \frac{11b^{d/3}}{4n^{1/3}} + \frac{L}{b} \\
&\lessapprox L_{\mathcal{D}}(f^*) + \frac{11\left(\frac{L^3 n}{d^3}\right)^{\frac{d}{3(d+3)}}}{4n^{1/3}} + \frac{L}{\left(\frac{L^3 n}{d^3}\right)^{\frac{1}{d+3}}} \\
&= L_{\mathcal{D}}(f^*) + \frac{11L^{\frac{d}{d+3}}}{4n^{\frac{1}{(d+3)}} d^{\frac{d}{d+3}}} + \frac{Ld^{\frac{3}{d+3}}}{L^{\frac{3}{d+3}} n^{\frac{1}{d+3}}} \\
&= L_{\mathcal{D}}(f^*) + \frac{11L + 4Ld}{4L^{\frac{3}{d+3}} n^{\frac{1}{d+3}} d^{\frac{d}{d+3}}}. \qquad (4.52)
\end{aligned}$$

□

Remark 7. *The generalization bound we obtained using the case analysis approach gives sub-optimal rates of $\mathcal{O}(n^{-1/(d+3)})$, while we know the optimal rate is $\mathcal{O}(n^{-1/(d+2)})$ from Theorem 4.12. This is the artefact of this proof method, as we have to choose $k = \mathcal{O}(n^{2/(d+3)})$ to balance the terms. However, we note that one of the terms is multiplied by the irreducible error $L_{\mathcal{D}}(f^*)$ in Thm. 4.13, so in the case where $L_{\mathcal{D}}(f^*)$ is sufficiently small then the analysis here will provide a better bound than Thm. 4.12.*

In the special case where the Bayes error is zero, which is often called the "realizable" setting in literature, we can obtain a fast (order n^{-1}) convergence rate.

Theorem 4.16 (Error convergence of histogram, realizable case)**.** *Let $\hat{h}, \hat{\eta}$ be as defined in* (4.20)*, assume that $\eta(X)$ is an L-Lipschitz function and the Bayes error $L_{\mathcal{D}}(f^*) = 0$. If we choose $b = 2L\sqrt{d}$, we have:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \mathcal{O}\left(\frac{(2L\sqrt{d})^d}{n}\right). \qquad (4.53)$$

Proof. We recall the result from the step of the main theorem, referring to equation (4.43). Since the Bayes error is zero we have from eq. (4.43):

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \frac{2rk}{n} + 3\mathbb{E}[|p - \eta(X)|]. \qquad (4.54)$$

Now we choose the parameter $b = 2L\sqrt{d}$ and consider $\mathbb{E}[|p - \eta(X)|]$

$$\begin{aligned}\mathbb{E}[|p - \eta(X)|] &\le \mathbb{E}\left[\max_{X' \in C_{i(X)}} |\eta(X') - \eta(X)|\right] \\ &\le \mathbb{E}[L\|X' - X\|_2] \\ &\le \frac{L\sqrt{d}}{b} = \frac{1}{2}.\end{aligned} \tag{4.55}$$

Since the Bayes error is zero so $\max\limits_{X' \in C_{i(X)}} |\eta(X') - \eta(X)|$ can only be either zero or one. This implies that $\mathbb{E}[\max\limits_{X' \in C_{i(X)}} |\eta(X') - \eta(X)|] = 0$ hence $\mathbb{E}[|p - \eta(X)|] = 0$. We conclude that by choosing $k = 2$, recalling the choice of b, and that $r = b^d$, we get:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \le \frac{2rk}{n} = \frac{2^{d+2}(L\sqrt{d})^d}{n} = \mathcal{O}\left(\frac{(2L\sqrt{d})^d}{n}\right). \tag{4.56}$$

□

Theorem 4.16 describes the generalization loss in the case where the Bayes error is zero, which gives a fast rate of $\mathcal{O}(1/n)$. However, even if the Bayes error is non-zero, as long as the Bayes error is close enough to zero, we are also likely to obtain a tighter bound than Thm. 4.10. We formalise this statement in the following theorem by combining with the results in Thm. 4.10, we obtain the following generalization error guarantee for the histogram classifier:

Theorem 4.17 (Generalization of histogram)**.** *Let $\hat{h}, \hat{\eta}$ be as defined in* (4.20)*, and assume that $\eta(X)$ is an L-Lipschitz function. We have for all b,*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \le \min\left\{L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{4(b^d+1)+\pi}{2n}}, 2L_{\mathcal{D}}(f^*) + \frac{3L\sqrt{d}}{b} + \frac{20b^d}{n}\right\}. \tag{4.57}$$

Proof. From the result of Thm. 4.13, take $k \ge 10$. Then take the minimum between this and result from Thm. 4.10. □

Theorem 4.15 already provides the convergence rate and the choice of b where the first expression is the minima in equation (4.57). For the another case, we have the following result as an immediate corollary of Thm. 4.17.

Corollary 4.18. *Let $\hat{h}, \hat{\eta}$ be as defined in* (4.20)*, and assume that $\eta(X)$ is an L-Lipschitz function. Then we have for $b = \mathcal{O}(n^{1/(d+1)})$,*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq 2L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{\frac{1}{d+1}}}\right). \tag{4.58}$$

Note that this corollary shows the histogram classifier converges to 2 times the Bayes error at a faster rate of $\mathcal{O}(n^{-1/(d+1)})$ compared to $\mathcal{O}(n^{-1/(d+2)})$ in Theorem 4.12 even when the Bayes error is zero. While the analysis of Thm. 4.12 provides the optimal rate in the general setting, this corollary shows that we can obtain a faster convergence if the Bayes error is sufficiently small. For example, if $L_{\mathcal{D}}(f^*) < 1 \times 10^{-100}$, then a faster error convergence towards 2×10^{-100} is usually more useful than a slower convergence to 1×10^{-100}.

The following section will exploit a neat property of random projection-based dimensionality reduction to get better bounds whenever the data support has a low complexity structure, even if it is not known a-priori.

4.4 Learning with compressed histograms

In this section, we study the performance of the histogram classifier in the compressed setting, i.e. when the histogram classifier has only access to the projected sample points. Recall that Φ is a linear map that satisfies the condition of the JL-lemma (Thm. 2.11) and S_Φ is the set of projected sample points in $\mathbb{R}^m$ as defined in previous sections. The training of the histogram is then performed with S_Φ only without seeing the original sample set S. For any fixed cell width parameter b (in the projected space), the optimal classifier in the hypothesis class in the projected space $\mathcal{H}_b^\Phi$ is denoted by h_Φ^* defined in a similar way as h^* in the original space.

$$h^*_\Phi(X) = \mathbb{1}_{\{\eta^*_\Phi(X) \geq \frac{1}{2}\}}, \quad \eta^*_\Phi(X) = \mathop{\mathbb{E}}_{\Phi X' \in C_{i(\Phi X)}} [\eta(X')] = \mathbb{E}_{X'} \left[\mathbb{P}[Y = 1 | \Phi X' \in C_{i(\Phi X)}]\right]. \tag{4.59}$$

We also denote the ERM output histogram learned with the compressed data by $\hat{h}_\Phi$.

$$\hat{h}_\Phi(X) = \mathbb{1}_{\{\hat{\eta}_\Phi(X) \geq \frac{1}{2}\}}, \quad \hat{\eta}_\Phi(X) = \frac{\sum_{i \in [n]} \mathbb{1}_{\{\Phi X_i \in C_{i(\Phi X)}\}} Y_i}{\sum_{i \in [n]} \mathbb{1}_{\{\Phi X_i \in C_{i(\Phi X)}\}}}. \tag{4.60}$$

4.4.1 Histogram classifier with random projection

From the analysis of the generalization bound in the previous result, we see that there is a bad dependence on the dimension. As the dimension d increases, one of the terms grows exponentially with d. However, natural data sources often have a lower intrinsic dimension (in some sense) even if their feature representation is d-dimensional. The bounds presented in the previous section are unable to take advantage of this. This naturally inspires us to use a dimensionality reduction tool such as random projection as a solution. We will use the generalized JL-lemma (Thm. 2.11) for our analysis which allowed us to capture the low dimensional structure of the sample set. Applying the result to our compressed learning setting, we found that with a smaller increase of the approximation error due to distortion, we reduce the dimension dependence on our error bound as the following theorem states.

Theorem 4.19. *Let S be a set of n samples drawn i.i.d. from distribution $\mathcal{D}$, and let $\hat{h}_\Phi$, S_Φ as defined in* (4.60)*. Assume that $\eta(X)$ is an L-Lipschitz function and let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. Then for any $b \in \mathbb{N}$, $0 < \epsilon_\Phi < 1$, let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2. \tag{4.61}$$

Then we have with probability at least $1 - \delta_\Phi$*:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq L_{\mathcal{D}}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \tag{4.62}$$

Proof. First we note that the argument for finite hypothesis class also holds in the compressed histogram setting because the number of hypotheses is still finite for any fixed parameter b. Specifically, the number of cells in the compressed histogram is reduced from b^d to b^m, hence reducing the size of the hypothesis class. By the same procedure as the derivation of equation 4.26 in the proof of Thm. 4.10 we have

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq L_{\mathcal{D}}(h^*_\Phi) + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \tag{4.63}$$

We have

$$\begin{aligned}
L_{\mathcal{D}}(h^*_\Phi) &= \mathop{\mathbb{P}}_{(X,Y)\sim\mathcal{D}}[h^*_\Phi(X) \neq Y] = \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{Y\sim\eta(X)}[\mathbb{1}_{\eta^*_\Phi(X)\geq 1/2} \neq Y]\right] && (4.64)\\
(\textit{Lemma } 4.6) \quad &\leq \mathop{\mathbb{E}}_{X}\left[\mathop{\mathbb{E}}_{Y\sim\eta^*_\Phi(X)}[\mathbb{1}_{\eta^*_\Phi(X)\geq 1/2} \neq Y]\right] + \mathbb{E}_X[|\eta^*_\Phi(X) - \eta(X)|]\\
&= \mathop{\mathbb{E}}_{X}[\min\{\eta^*_\Phi(X), 1-\eta^*_\Phi(X)\}] + \mathbb{E}_X[|\eta^*_\Phi(X) - \eta(X)|]\\
(\textit{Lemma } 4.6) \quad &\leq \mathop{\mathbb{E}}_{X}[\min\{\eta(X), 1-\eta(X)\}] + 2\mathbb{E}_X[|\eta^*_\Phi(X) - \eta(X)|]\\
&\leq L_{\mathcal{D}}(f^*) + 2\mathbb{E}_X\left[\max_{\Phi X' \in C_{i(\Phi X)}} |\eta(X) - \eta(X')|\right]\\
&\leq L_{\mathcal{D}}(f^*) + 2L\mathop{\mathbb{E}}_{X}\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right],
\end{aligned}$$

where the second-to-last line is because η^*_Φ is the average of $\eta(X')$ over all X' such that $\Phi X'$ is in the same cell as ΦX, and the last line follows from the Lipschitz condition of η. Substituting this into equation 4.63 we obtain:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq L_{\mathcal{D}}(f^*) + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + 2L\mathop{\mathbb{E}}_{X}\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right]. \tag{4.65}$$

Since we have that ΦX and $\Phi X'$ are in the same cell, by construction this implies that $\|\Phi X - \Phi X'\|_\infty \leq \frac{D_\Phi}{b}$, where $D_\Phi = \max_{X,X'\in\mathcal{X}} \|\Phi X - \Phi X'\|_2$ denotes the diameter of the projected sample set. We also have from the JL-lemma that

$$\|X - X'\|_2 \leq \frac{\|\Phi X - \Phi X'\|_2}{1 - \epsilon_\Phi} \leq \frac{D_\Phi}{b(1 - \epsilon_\Phi)}, \tag{4.66}$$

Since $\mathcal{X} = [0,1]^d$, we have $\|X\|^2 \leq d$ for all $X \in \mathcal{X}$. Applying the JL-lemma we have $D_\Phi \leq (1 + \epsilon_\Phi)\sqrt{d}$. Hence we can conclude with high probability:

$$\mathop{\mathbb{E}}_S[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m + 2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \tag{4.67}$$

□

We have shown that the dimensionality dependence of the generalization bound can be reduced to m if the sample set has a simple structure. Since the error bound grows exponentially with respect to the dimension, this implies a significant decrease in its value if m is small compared with d. The cost of this dimensionality reduction is the extra multiplication factor that came from the projection distortion (one that depends on ϵ_Φ). However, this factor is usually much less significant compared to the reduction of the dimensional-dependent term. Since we only incur a multiplicative factor due to projection distortion in the generalization bound of the compressed histogram, by a similar choice of the parameter b, we obtain the 'optimal' convergence rate of $\mathcal{O}(n^{-\frac{1}{m+2}})$ with respect to the projection dimension m (up to distortion factors of ϵ_Φ) as an immediate result.

Theorem 4.20. *Let $\hat{h}_\Phi$, $\Phi, \epsilon_\Phi, \delta_\Phi, m$ be as described in Thm. 4.19. Assume that $\eta(X)$ is L-Lipschitz. If $b = \mathcal{O}(n^{-\frac{1}{m+2}})$ then we have with probability at least $1 - 2\delta_\Phi$:*

$$\mathop{\mathbb{E}}_S[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \mathcal{O}\left(\frac{L(1+\epsilon_\Phi)\sqrt{d}}{n^{\frac{1}{m+2}}(1-\epsilon_\Phi)}\right). \tag{4.68}$$

Proof. Use theorem 4.19 and verify with the choice of b. □

Just as in the non-projected histogram case, we can obtain a better convergence rate if the Bayes error is zero $L_\mathcal{D}(f^*) = 0$. We show in the following theorem that, as long as we choose a small enough bin-width (or large enough b) for each cell and the projection parameters $\epsilon_\Phi, \delta_\Phi$ are not too large ($\geq 1/2$), then the error of the compressed histogram converges with rate of order $\mathcal{O}(1/n)$.

Theorem 4.21. *Let $\hat{h}_\Phi$ be the ERM histogram classifier using the training set S_Φ. Assume that $\eta(X)$ is an L-Lipschitz function and let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. Then for any $0 < \delta_\Phi < 1$, $0 < \epsilon_\Phi \leq 1/2$ let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2. \tag{4.69}$$

Then for any $b > 3L\sqrt{md}$, we have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \mathcal{O}\left(\frac{L^m d^{m/2}}{n}\right). \tag{4.70}$$

Proof. Note that by a similar argument from the proof of Thm 4.13, for any positive integer k we have

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \frac{2b^m k}{n} + \mathop{\mathbb{E}}_{X_1,\ldots,X_n,X}\left[\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{h}_\Phi(X) \neq Y|\ |C_{i(\Phi X)} \cap S| \geq k]\right]. \tag{4.71}$$

Denote $\hat{h}_\Phi(X) = \hat{Y}$, fix S and X with $|C_{i(\Phi X)} \cap S| \geq k$. By Lemma 4.6 we have

$$\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta(X)\\ i\in[n]}}[\hat{Y} \neq Y] \leq \mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta^*_\Phi\\ i\in[n]}}[\hat{Y} \neq Y] + |\eta^*_\Phi - \eta(X)|. \tag{4.72}$$

Since X is fixed, the number of points in $C_{i(\Phi X)}$ is also fixed, we apply Lemma 4.4 and we have

$$\mathop{\mathbb{P}}_{\substack{Y_i\sim\eta(X_i),Y\sim\eta^*_\Phi\\ i\in[n]}}[\hat{Y} \neq Y] \leq \left(1 + \sqrt{\frac{8}{k}}\right) \mathop{\mathbb{P}}_{Y\sim\eta^*_\Phi}[Y \neq \mathbb{1}_{[\eta^*_\Phi \geq 1/2]}]. \tag{4.73}$$

We also have

$$\mathop{\mathbb{P}}_{Y\sim\eta^*_\Phi}[Y \neq \mathbb{1}_{[\eta^*_\Phi \geq 1/2]}] = \min\{\eta^*_\Phi, 1-\eta^*_\Phi\} \leq \min\{\eta(X), 1-\eta(X)\} + |\eta^*_\Phi(X) - \eta(X)|.$$

Taking expectations and combining our results we have

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \left(1+\sqrt{\frac{8}{k}}\right) L_{\mathcal{D}}(f^*) + \frac{2b^m k}{n} + 3\mathbb{E}[|\eta^*_\Phi(X) - \eta(X)|]. \tag{4.74}$$

Since the Bayes error is zero $L_{\mathcal{D}}(f^*) = 0$, we eliminate the first term and choose $k = 10$ to be a constant.

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \frac{20b^m}{n} + 3\mathbb{E}[|\eta^*_\Phi(X) - \eta(X)|] \leq \frac{20b^m}{n} + 3L\mathop{\mathbb{E}}_{X}\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right]. \tag{4.75}$$

followed by the Lipschitz condition of η. Since each $\Phi X'$ is in the same box as ΦX, by setting this means that $\|\Phi X' - \Phi X\|_\infty \leq \frac{D_\Phi}{b}$, by Theorem 2.11 we have

$$\|X - X'\|_2 \leq \frac{\|\Phi X - \Phi X'\|_2}{1-\epsilon_\Phi} \leq \frac{D_\Phi}{b(1-\epsilon_\Phi)} \tag{4.76}$$

Note that our original input space is $[0,1]^d$, and $\|X\|_2^2 \leq d$. By the Johnson-Lindenstrauss lemma we have $\|\Phi X\|_2 \leq (1+\epsilon_\Phi)\sqrt{d}$. Hence $D_\Phi \leq (1+\epsilon_\Phi)\sqrt{d}$ and we have

$$\max_{\Phi X' \in C_{i(\Phi X)}} |\eta(X) - \eta(X')| \leq \frac{L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)}. \tag{4.77}$$

Now note by that choosing

$$b \geq 3L\sqrt{d} \text{ and } \epsilon_\Phi < 1/2, \tag{4.78}$$

we have $\max_{\Phi X' \in C_{i(\Phi X)}} |\eta(X) - \eta(X')| < 1$. However since the Bayes error is zero, the difference $|\eta(X) - \eta(X')|$ can only be zero or one. Hence we can conclude that $\max_{\Phi X' \in C_{i(\Phi X)}} |\eta(X) - \eta(X')| = 0$. Finally, substituting the choice of b into equation (4.75), we have with high probability:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \mathcal{O}\left(\frac{L^m d^{m/2}}{n}\right). \tag{4.79}$$

□

4.4.2 Random projection with non-linear spaces

In this section, we consider different variations of the data geometry that allow us to obtain good results with random projection on the histogram classifier. Although the Gaussian width captures the structure of the sample set, it is also likely in practical situations that while the majority of samples lie on a low-dimensional subspace but a small fraction of the points can attain a high-dimensional structure. We consider the following general setting: Assume that there exists a low-dimensional subspace P such that for some $0 < \gamma < 1, 0 < \xi \leq 1$, $(1-\gamma)$ fraction of samples is contained in P while the rest γ fraction of the sample set S is ξ distance away from subspace. Examples of these types of structures are demonstrated in Fig. 4.1 and 4.2 where we have generated data sets in a 3-dimensional space but have a structure similar to a 2-dimensional plane. In this section, we show that it is sufficient to project onto the lower dimension based on the geometry of the $(1-\gamma)$ fraction of points that have a lower dimensional structure provided that γ and ξ are not large simultaneously.

We first show the result for the case when γ is small, but ξ can be arbitrary:

Theorem 4.22 (Small fraction of complex points). *Let $\hat{h}_\Phi$ be the ERM histogram classifier using the training set S_Φ. Let S_1 be the $(1-\gamma)$ fraction of points in S that lie within a linear subspace P and S_2 be the γ fraction of points that are not on the subspace. Assume that $\eta(X)$ is a L-Lipschitz function and let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S_1. Then for any $b \in \mathbb{N}$, $0 < \epsilon_\Phi, \delta_\Phi < 1$, let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)} \right]^2 . \tag{4.80}$$

Then we have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_{S}[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + 2\gamma(1-\gamma). \tag{4.81}$$

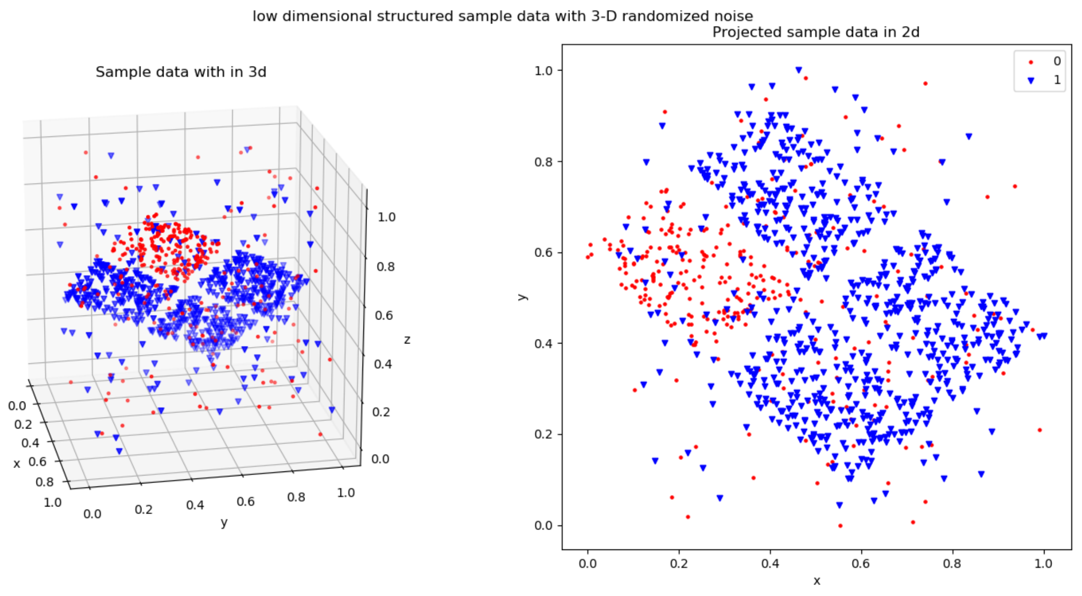

Figure 4.1: 2d data embedded in 3d where $\gamma = 0.1, \xi = 1$. In this case, the majority of the sample points lie on a plane P and a small fraction (γ) of points can be anywhere in the space.

In particular, if $\gamma \leq 1/b$ *we have*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_{\Phi})] \leq L_{\mathcal{D}}(f^*) + \frac{2L(1+\epsilon_{\Phi})\sqrt{d}}{b(1-\epsilon_{\Phi})} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{2}{b}. \tag{4.82}$$

Proof. By the same argument as in the proof of Thm. 4.19, starting with equation (4.65), we have

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h}_{\Phi})] \leq L_{\mathcal{D}}(f^*) + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + 2L\mathop{\mathbb{E}}_{X}\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right]. \tag{4.83}$$

Note that this already implies the impact of the complex points can be quantified by the distance of projected points in the original space, and we know this can be bounded effectively as long as they lie on the subspace. We will consider two cases of X and X'. Let D_{Φ} denote the diameter of the projected sample set:

1. Both X, X' lies in the subspace: We have by the generalized JL-lemma

$$\sup \|X - X'\|_2 \leq \frac{\|\Phi X' - \Phi X\|_2}{1-\epsilon_{\Phi}} \leq \frac{D_{\Phi}}{b(1-\epsilon_{\Phi})}. \tag{4.84}$$

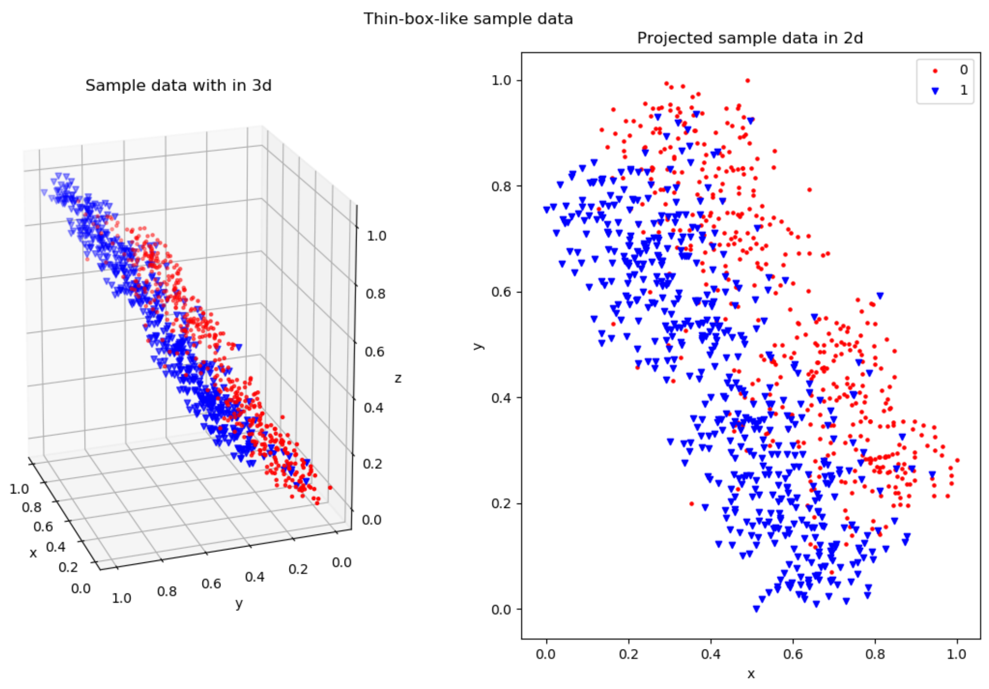

Figure 4.2: 2d data embedded in 3d where $\gamma = 1, \xi = 0.1$. In this case, instead of a plane containing the sample points, every sample point is "near" some plane P but not necessarily on P.

2. For another case where one or both of X, X' are not on the subspace, we bound the excess error by the probability that this case occurs

$$\mathbb{P}[X, X' \notin P] = 2\gamma(1-\gamma) + \gamma^2 = 2\gamma - \gamma^2. \tag{4.85}$$

Note that our original input space is $[0,1]^d$, hence $\|X\|_2^2 \leq d$. By Thm. 2.11 we have $\|\Phi X\|_2 \leq (1+\epsilon_\Phi)\sqrt{d}$. Hence $D_\Phi \leq (1+\epsilon_\Phi)\sqrt{d}$.

Hence we conclude that with high probability:

$$\mathop{\mathbb{E}}_S[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + 2\gamma(1-\gamma). \tag{4.86}$$

In particular, if $\gamma \leq 1/b$ we have

$$\mathop{\mathbb{E}}_S[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L\sqrt{d(1+\epsilon_\Phi)}}{b\sqrt{1-\epsilon_\Phi}} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{2}{b}. \tag{4.87}$$

□

Theorem 4.22 shows that if $\gamma \leq 1/b$, then the rate of convergence of the compressed histogram is not affected (up to a constant factor) by the γ fraction of points that have a high-dimensional structure. The next case we will consider is when ξ is small, i.e. all points are in bounded distance away from a subspace P. However, we do not restrict the number of points that are not contained in P, as long as they are all close enough. We consider the complexity of such geometry as follows.

Lemma 4.23. *Let S be a finite sample set in $\mathbb{R}^d$ such that there exists a linear subspace P that satisfies the following. For every point $X \in S$, the ℓ_∞ distance of X to the nearest point in P is at most ξ. Then the Gaussian width of S, $w(S)$ is at most*

$$w(S) = w(P) + \xi \frac{\sqrt{2}\Gamma((d+1)/2)}{\Gamma(d/2)}. \tag{4.88}$$

Proof. Note that for every $X \in S$, we can write X as $X' + U$ where $X' \in P$ and U is an orthogonal vector to P. Hence by the definition of Gaussian width we have

$$\begin{aligned} w(S) &= \mathbb{E}[\sup_{X \in S}\langle g, X\rangle] = \mathbb{E}[\sup_{X \in S}\langle g, X'\rangle] + \mathbb{E}[\sup_{U}\langle g, U\rangle] \\ &\leq w(P) + \xi \mathbb{E}[\sup \|g\|] \\ &= w(P) + \xi \frac{\sqrt{2}\Gamma((d+1)/2)}{\Gamma(d/2)}, \end{aligned}$$

where Γ is the Gamma function. □

Using Lemma 4.23, we have the following result directly implied from Theorem 4.19.

Corollary 4.24 (linear subspaces with small shifts)**.** *Suppose there exists a linear subspace P such that every $X \in S$ has distance at most ξ from P. Assume that $\eta(X)$ is a L-Lipschitz function, then for any $b \in \mathbb{N}$, $0 < \epsilon_\Phi, \delta_\Phi < 1$. Let b be a positive integer that* *satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(P) + \sqrt{\log(1/\delta_\Phi)} + \xi \frac{\sqrt{2}\Gamma((d+1)/2)}{\Gamma(d/2)} \right]^2. \tag{4.89}$$

We have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_S [L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m + 2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}}. \tag{4.90}$$

4.4.3 Projection with manifolds

Besides linear subspaces and their variations, manifold structured data sets are also commonly seen in practice. In this section, we study the compressed learning of histogram classifier when the training sample has a manifold structure or is 'almost' a manifold structure, in the sense that points are allowed to be shifted a small distance from the actual manifold. We use the following form of the JL-lemma applied to manifolds, refer to [9] for the definitions of the parameters used.

Theorem 4.25 ([9]). *Let M be a compact d_M-dimensional Riemannian submanifold of $\mathbb{R}^d$ having condition number $1/\tau$, volume V, and geodesic covering regularity R. Fix $0 < \epsilon_\Phi < 1$ and $0 < \delta_\Phi < 1$. Let Φ be random orthoprojector from $\mathbb{R}^d$ to $\mathbb{R}^m$ with*

$$m = \mathcal{O}\left(\epsilon_\Phi^{-2} d_M \log(dVR\tau^{-1}\epsilon_\Phi^{-1}) \log(1/\delta_\Phi)\right). \tag{4.91}$$

If $m \leq d$, then with probability at least $1 - \delta_\Phi$ the following statement holds: For pair of points $X, X' \in M$,

$$(1-\epsilon_\Phi)\sqrt{\frac{m}{d}} \leq \frac{\|\Phi X - \Phi X'\|_2}{\|X - X'\|_2} \leq (1+\epsilon_\Phi)\sqrt{\frac{m}{d}}. \tag{4.92}$$

Theorem 4.26 (Manifold data). *Assume that there exists a manifold M satisfying the conditions in Thm. 4.25 and every point in S has a distance less than ξ to M. In particular, for $0 \leq \gamma \leq 1$, there are $(1-\gamma)$ fraction of points that lies exactly on M. Assume that $\eta(X)$ is a L-Lipschitz function. Then for any $b \in \mathbb{N}$, $0 < \epsilon_\Phi, \delta_\Phi < 1$, let m be a* *positive integer that satisfies:*

$$m = \mathcal{O}\left(\frac{d_M \log(dVR\tau^{-1}\epsilon_\Phi^{-1}) \log(1/\delta_\Phi)}{\epsilon_\Phi^2}\right). \tag{4.93}$$

Then we have with probability at least $1 - \delta_\Phi$*:*

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq L_{\mathcal{D}}(f^*) + \frac{2L\sqrt{d}(1+\epsilon_\Phi)}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + +2(1+\gamma)L\sqrt{d}\xi. \tag{4.94}$$

Proof. Following the lines of the proof as in Thm. 4.19 we can start from equation (4.65) as follows:

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq L_{\mathcal{D}}(f^*) + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + 2L\mathbb{E}_X\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right]. \tag{4.95}$$

We need to consider three cases of X and X'. Let D_Φ denote the diameter of the projected sample set:

1. Both X and X' lie in the manifold: We have by Thm. 4.25 that

$$\sup \|X - X'\|_2 \leq \frac{\|\Phi X' - \Phi X\|_2}{1-\epsilon_\Phi} \leq \frac{D_\Phi}{b(1-\epsilon_\Phi)}. \tag{4.96}$$

2. Only one of X, X' lies in the manifold, w.l.o.g. assume $X' \notin M$. Let U denote the normal vector from the manifold to X' and $X'_0 \in M$ be the point closest to X', i.e. $X' = X'_0 + U$. Fix X, X', we have

$$\begin{aligned} \|X - X'\|_2 \leq \|X - X'_0\|_2 + \|U\|_2 &\leq \frac{\|\Phi X - \Phi X'_0\|_2}{1-\epsilon_\Phi} + \|U\|_2 \\ &\leq \frac{\|\Phi X - \Phi X'\|_2}{1-\epsilon_\Phi} + \sqrt{d}\xi \\ &\leq \frac{D_\Phi}{b(1-\epsilon_\Phi)} + \sqrt{d}\xi. \end{aligned}$$

3. Both X and X' are not in the manifold. Let U, U' denote the normal vector from the manifold to X, X' respectively. Furthermore, let $X_0, X'_0 \in M$ be the point closest

to X, X' respectively. Fix X, X', we have

$$\begin{aligned}\|X - X'\|_2 \leq \|X_0 - X_0'\|_2 + \|U\|_2 + \|U'\|_2 &\leq \frac{\|\Phi X_0 - \Phi X_0'\|_2}{1-\epsilon_\Phi} + \|U\|_2 + \|U'\|_2 \\ &\leq \frac{\|\Phi X - \Phi X'\|_2}{1-\epsilon_\Phi} + 2\sqrt{d}\xi \\ &\leq \frac{D_\Phi}{b(1-\epsilon_\Phi)} + 2\sqrt{d}\xi.\end{aligned}$$

Combining the three cases and note the probability for any $X \in S$ to lie on the manifold M is $1-\gamma$, we have

$$\mathop{\mathbb{E}}_X\left[\max_{\Phi X' \in C_{i(\Phi X)}} \|X - X'\|_2\right] \leq \frac{D_\Phi}{b(1-\epsilon_\Phi)} + (1+\gamma)\sqrt{d}\xi.$$

Note that our original input space is $[0,1]^d$, hence $\|X\|_2^2 \leq d$, by the Johnson-Lindenstrauss lemma we have $D_\Phi \leq (1+\epsilon_\Phi)\sqrt{d}$.

Hence we conclude that with high probability:

$$\mathop{\mathbb{E}}_S[L_\mathcal{D}(\hat{h}_\Phi)] \leq L_\mathcal{D}(f^*) + \frac{2L\sqrt{d}(1+\epsilon_\Phi)}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + +2(1+\gamma)L\sqrt{d}\xi. \quad (4.97)$$

□

4.5 Histogram with neighbour search

A potential issue with the classical histogram classifier is that many cells of the histogram can be empty (no training points) in higher dimensions because the number of cells $r = b^d$ grows exponentially w.r.t. the dimensionality. Random projection discussed in the last section solves this issue to an extent. However, the projection dimension may remain high after projection because the data set has a high dimensional structure. e.g. projection from a 1000000 dimension to a 1000 dimension is still too large ($r \geq 2^{1000}$). The excessive number of cells can lead to over-fitting at higher dimensions and hence performs poorly on unseen sample points.

In this section, we present an approach to solving this problem using neighbour search through the trained cells. The key idea is that if we observe an unseen sample in a cell C_i without any training samples, we seek the nearest cell in our histogram that contains at least one training sample, and we use the label of the neighbour cell as the label for the new cell C_i. We limit the distance radius for the neighbour search so that it will not search for far away neighbours where their labels may not be relevant for the unseen sample. The new prediction algorithm is outlined in Alg. 2 (training procedure will be the same). We set a default distance bound equal to the dimensionality d of the samples. The intuition is to allow the cell to look through its adjacent cells and corner cells. For example, a cell with index $[1, 1]$ in two dimensions can search neighbours in $\{[0, 0], [0, 1], [1, 0], [2, 1], [1, 2], [2, 2]\}$.

Algorithm 2 Histogram with neighbour search

```
1: procedure NeighbourPredict($X$)
2:     if $C_{i(X)}$ is labelled then
3:         return $label(C_{i(X)})$
4:     else
5:         finds the nearest cell $C_j$ of $C_{i(X)}$ within distance $d$ (dimension of $X$)
6:         if neighbour exists then
7:             set $label(C_{i(X)}) := label(C_j)$
8:         else
9:             Assign a random label to $label(C_{i(X)})$
10:        end if
11:        return $label(C_{i(X)})$
12:    end if
13: end procedure
```

Note that the time complexity of Alg. 2 is never greater than the neighbour search for the kNN algorithm because we only have to search through non-empty cells within the bounded distance. The number of non-empty cells is never greater than the number of samples since each cell contains at least one sample. Furthermore, empty cells require at most one neighbour search because it memorised the label after the first time it searched for its neighbour. Hence the maximum number of neighbour searches required is bounded by the number of empty cells after training and independent of the number of testing points. Moreover, since we imposed the bounded distance search, this can be done more efficiently by searching through a smaller subset of cell indices.

The bounded distance search condition is to avoid searching for "neighbouring cells" which are far away. Based on our Lipschitzness assumption of η, intuitively we expect samples to be more similar if they are close and vice versa (although this is not guaranteed). Hence searching for cells that are far away is unlikely to give informative predictions. We now show this method provides the same generalisation guarantee as the vanilla histogram.

Theorem 4.27. *Let $\hat{h}$ be the histogram classifier as defined in* (4.20) *but with the prediction rule defined in Alg. 2. Assume that $\eta(X)$ is a L-Lipschitz function. Then we have, for any fixed $b \in \mathbb{N}$:*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2(b^d+1)}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \tag{4.98}$$

Proof. We note that the only change caused by comparing with the classical histogram is that the distance between the training points and an arbitrary test point has increased, as we are allowed to look at labels in neighbour cells. Since the size of the hypothesis class $\mathcal{H}_b$ did not change and we still have the Lipschitz condition on η, we have the same result from a finite hypothesis class argument as in Thm. 4.10:

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \tag{4.99}$$

Furthermore, note the best hypothesis $h^* \in \mathcal{H}_b$ is independent of the training set S, hence there is no change to the generalisation error of h^* and therefore we obtain the result. $\square$

Remark 8. *Note that this neighbour search technique actually reduces the size of the hypothesis class by considering only a subset of $\mathcal{H}_b$. However, this is invisible in our bound in Thm. 4.27 because this reduction of size depends on the training set S (and the underlying distribution $\mathcal{D}$), which in the worst case, can have the same size as $\mathcal{H}_b$. Hence although in practise this approach will likely to reduce the hypothesis class, we can not improve our upper bound here as we do not make assumptions on the structure of S and*

$\mathcal{D}$.

Remark 9. *The analysis of Theorem 4.27 is the same as the original analysis for the classical histograms. It is trivial to show the same analysis on other results of the histogram predictor, including the compressed histogram analysis. Hence we are free to use the theoretical guarantees of the classical histogram.*

4.6 Empirical Illustration

In this section, we will analyse the practical performance of the Histogram predictor and the kNN predictor. We will use the histogram variant described in Alg. 2 that uses the neighbour search for best performance. We will use a variety of data sets, both synthetic and real, and have various sizes and dimensions. To demonstrate the results implied by our theory, we will generate data sets that match the condition specified by our theoretical result and analyse the outcome.

4.6.1 Performance of kNN and Histogram on different parameters

The first step for training the kNN and the histogram classifier is to choose the appropriate parameters for the algorithm. This parameter will be the number of nearest neighbours k for kNN and the cell width parameter b for the histogram. As discussed in previous sections, these parameters control the bias-variance trade-off for their generalization bounds. For the kNN algorithm, choosing a small k may lead to over-fitting in the training set, and a large k may lead to under-fitting. For the histogram, larger b can lead to over-fitting and smaller b may lead to under-fitting. Hence it is important to tune these parameters in practice to obtain the best possible performance on accuracy. The choice of parameters for the algorithms varies a lot depending on the sample set and the learning problem, and there is no fixed parameter that will work on all problems. For demonstration purposes, we generated a data set with 10 dimensions consisting of 10000 data points, where 70% of the samples are used for training and the rest to compute the validation error that allows us to choose the best parameter. The result is plotted in Fig 4.3. In Fig. 4.3, we

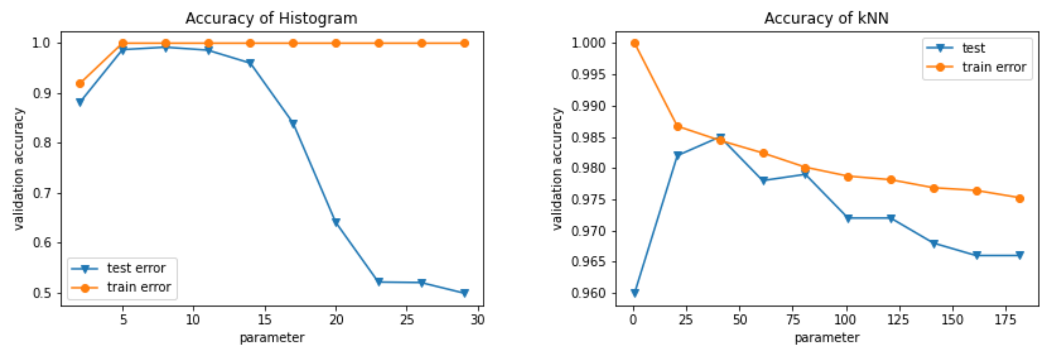

Figure 4.3: Variation of parameters for Histogram and kNN, cell width parameter b for Histogram (left) and k neighbours for kNN (right).

allow the parameters k and b to be chosen in a wide range. We note that in both plots, the validation accuracy starts at a lower level and starts to increase as the parameter increase. After reaching a peak in the accuracy for the best parameter, any further increase in the parameter will lead to a decrease in accuracy. However, although the shape of their curve looks similar (increase then decrease w.r.t. parameter), the reason for the shape is different. For kNN, the model is over-fitted for a small value of k, the evidence from Fig. 4.3 (right) is that the training error starts very high but the validation accuracy is relatively low. As k increases for large values, both validation and training error decrease which implies that the model is under-fitted, a plot for a larger value of k is plotted in Fig. 4.4. On the other hand, the histogram starts with a lower accuracy for both training and testing accuracy, implying that the model is under-fitted. As b increases to larger values, the testing accuracy decreases quickly while the training accuracy remains high, hence implying that the model is over-fitted. We also note that the parameter range for the histogram $(0-30)$ is much smaller than the parameter range for kNN $(0-200)$, which shows that k is less sensitive for kNN than b is for the histogram. i.e. a small change in the parameter b for histogram can lead to a bigger change in accuracy compared to a change in k for kNN. Hence we need to choose b more carefully and try out consecutive choices so that we don't miss the best-performing parameter. However, the benefit of the histogram is that it doesn't need a huge amount of trials to find the best-performing b before noticing the accuracy is decreasing because the range is relatively small. The

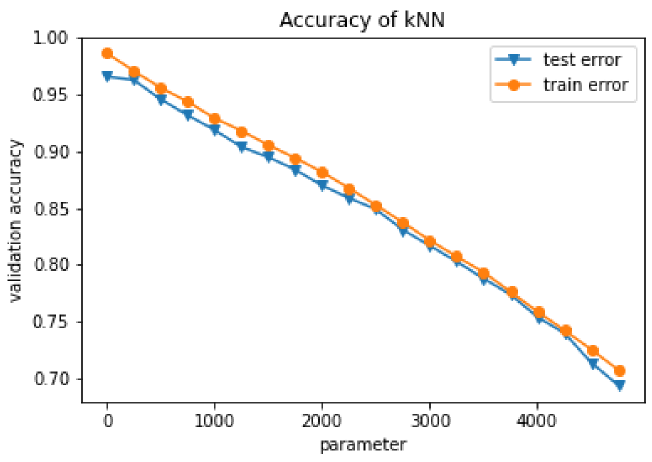

Figure 4.4: kNN parameter variations in a wide range.

minimum possible choice of b is 2 and usually, we can identify the best b within $[2, 30]$. The kNN algorithm, on the other hand, is more complicated to try out every possible k before noticing the accuracy is clearly decreasing, but if we try out values of k in large interval steps we can miss the best parameter as illustrated in Fig. 4.4. Furthermore, due to the inefficient computational time of the kNN prediction rule, it is also computationally heavy to test out a large number of k parameters.

4.6.2 Distortion of projection on low-dimensional data set

Recall that in the compressed learning setting, the learning algorithm only has access to the compressed data set, i.e. the projected data set using a random matrix projector. We have proved the generalization guarantee for compressed kNN in Thm. 4.7 and for the compressed histogram in Thm. 4.19. The condition of the bound depends on the intrinsic dimension of the sample space $\mathcal{X}$ as specified in Thm. 2.11. We have discussed how we have reduced the variance term of their generalization bounds by learning in the compressed setting with the trade-off of a small factor of distortion parameter ϵ_Φ. In this section, we generate a data set in 5 dimensions but embedded in a 100 dimensional space. i.e. the features vector has 100 attributes but its intrinsic dimension is 5. The data set has $10K$ sample points and we project this data set onto a 10 dimensional subspace using by Gaussian random matrix as in Thm 2.11. We compare the accuracy result between the original classifier and the compressed classifier, the result is demonstrated in Fig. 4.5. The distortion of the projection in general depends a lot on the intrinsic dimension of

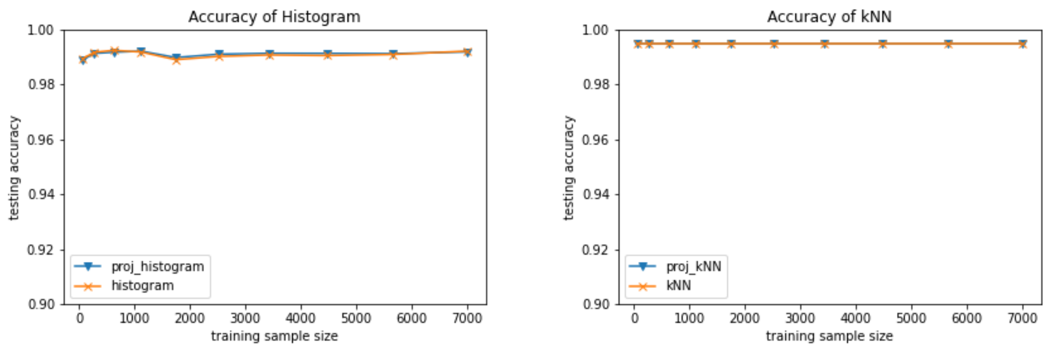

Figure 4.5: Comparison of testing accuracy for kNN and Histogram with and without compressed learning, where a 5-d data set is embedded in 100-d and then projected onto a 10-d subspace. Experiments are repeated 10 times and error bars are omitted here due to small standard deviation.

the data set and the projection dimension [73, 76]. The former can be difficult to find or estimate in practical situations. In this demonstration, we have generated the data set with 5 intrinsic dimensions and projected it onto a 10 dimensional subspace to minimise the distortion. From Fig. 4.5 we see that indeed the accuracy of the compressed kNN and histogram is almost identical to the accuracy of their high-dimensional analogues. This shows that the distortion due to random projection is minimal if the condition for Theorem 4.19 and Thm 4.7 is fully satisfied.

However, things can be a lot less satisfactory if the conditions for projection are not satisfied. If the data set does not have a low-dimensional structure, and we applied random projection to project the data to a low dimension, then the distortion of the data can be large and significantly affects the accuracy of the classification. To demonstrate this case, we generated a data set in 30 dimensions and embedded in a 50-dimensional space. There are $10K$ sample points and we perform a random projection to project the data onto a 5-dimensional subspace. The result for the testing accuracy of kNN and histogram is plotted in Fig. 4.6.

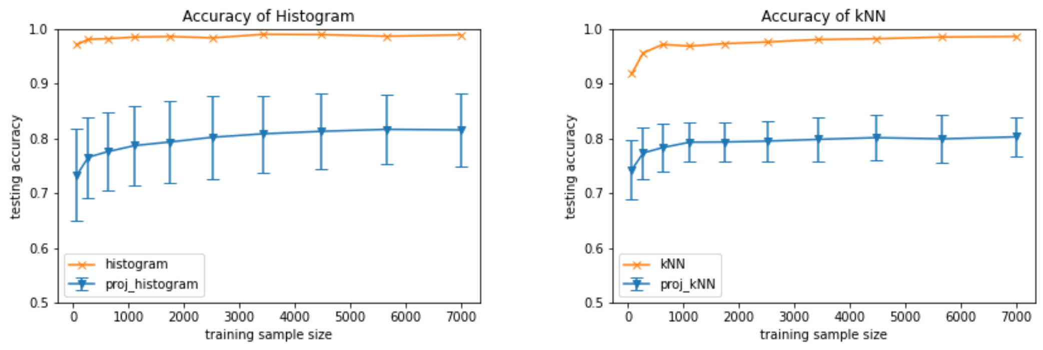

Figure 4.6: Comparison of accuracy for kNN and Histogram with and without compressed learning (bad projection case), where a 30-d data set is embedded in 50-d and then projected onto a 5-d subspace. Experiments are repeated 10 times and error bars represent one standard deviation away from the mean.

4.6.3 Run-time Analysis

Unlike the accuracy performance of kNN and histogram, which only differ by a small difference, the computational cost between the histogram and kNN is very significant. The kNN algorithm requires computing the pair-wise distances between the test sample and the training samples, hence the classic kNN has no training cost. That said, we still need to choose k for the best performance of kNN, which is done by validation in many cases. However, validation requires computing the validation error for each potential candidate of k, which adds complexity to its training phase. Furthermore, due to the slow performance of the classic kNN that is based on a brute force search for the nearest neighbours, spatial algorithms like KD-trees are implemented to aid the neighbour search of kNN, which also adds computational complexity to its training. Although learning with projected sample sets as noted in Thm. 4.7 can significantly improve its efficiency, it is still computationally heavy as the number of samples increases. On the other hand, histograms are much more computationally efficient than kNN for both training and testing due to their structure.

In this section, we analyse the computational cost of kNN and the histogram and compare them with their counterpart in the compressed learning setting. The histogram algorithm we used here is Alg. 2, we note that this is less efficient than the classic histogram in Alg. 1 and hence time complexity can be further improved if we use Alg. 1 instead. However, Alg. 1 usually requires larger data sets to achieve high performance hence we will only use Alg. 2 here for demonstration. We generate a data set with $10K$ sample points in 5 dimensions but embedded in a 50 dimensional space. The result for the histogram is plotted in Fig. 4.7 and kNN in Fig. 4.8. From Fig. 4.7, note that the training time of the

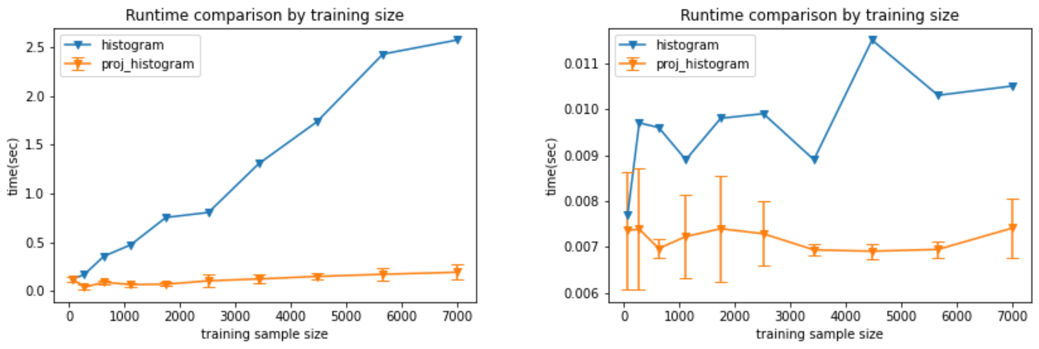

Figure 4.7: Runtime comparison for Histogram with compressed learning; training time (left plot) and testing time (right plot). Experiments are repeated 20 times and error bars indicate one standard deviation away from the mean.

histogram will be affected by the number of training samples as we need to record the label of each training sample. The testing time for the histogram is not affected much by random projection as we observed from Fig. 4.7 (right). The reason is that for the testing phase of the histogram we only need to find the index of the cell that contains the sample and record its label, hence this takes constant time. The testing time of the histogram is improved by random projection because we need to find the nearest cell index for the minority points in a sparse region (Alg. 2), and this process will be quicker in a lower dimension. From Fig 4.8 we observe that the run-time of kNN is significantly improved

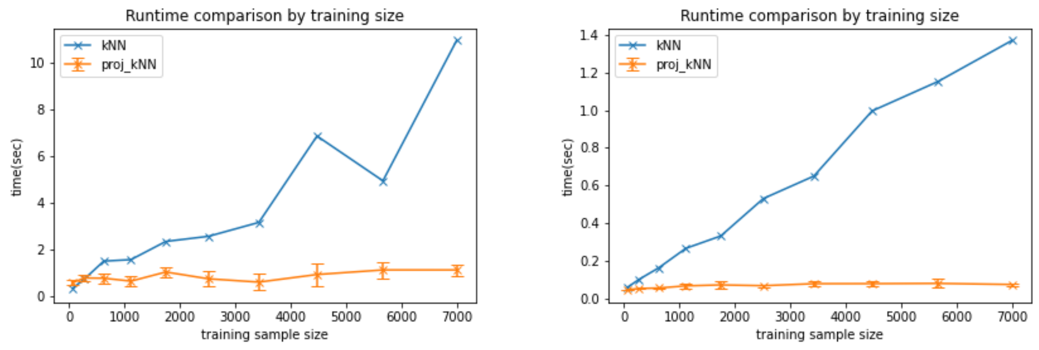

Figure 4.8: Run-time comparison for kNN with compressed learning; training time (left plot) and testing time (right plot).

using random projection. Both training and testing time is reduced by at least 80% when the algorithm is trained with the compressed samples, despite almost identical accuracies (Fig. 4.5). We note that neither training nor testing time is affected by the change in training size because they only depend on the size of the testing set and the validation set. We plotted the difference in the computational complexity between kNN and the

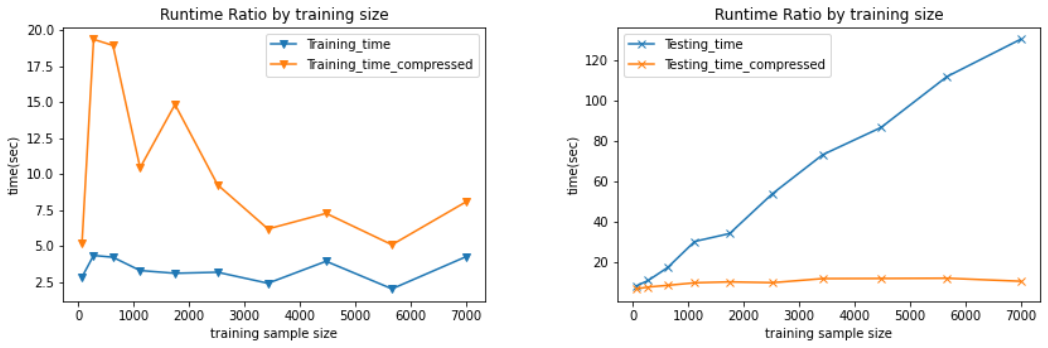

Figure 4.9: Run-time ratio of kNN over Histogram; training time (left plot) and testing time (right plot). The ratio is calculated by time elapsed by kNN over the time elapsed by the histogram.

histogram in Fig. 4.9. We observe that the histogram can be 5 times faster in the training phase than kNN in the non-projected case and up to 20 times faster in the projected case. This difference becomes more significant in the testing complexity: the histogram is nearly 10 times faster than kNN in the projected case and up to 120+ times faster in the non-projected case. The difference between the time elapsed in the general case can be bigger or smaller than the result demonstrated depending on the size and structure of the training set and the testing set. However, as pointed out in the discussion of Alg. 2, it can never be worse than the kNN in the absolute worst case, but it can be significantly faster than kNN in most scenarios.

4.6.4 Additional experiments

To further illustrate the performance of compressed kNN and histogram, we perform additional experiments with real high-dimensional data sets. We will use five data sets with dimension from 2400 to 100000 with different sizes, the details are listed in Table 4.2. We use a kd-tree structure for neighbour search inside both algorithms to reduce run-time.

Since we do not know the true intrinsic dimension of each data set, for simplicity we will

Data set	Attributes	Proj_d	Size	Train size	Validation size	Testing size
Dorothea	100000	100	1150	800	350	N/A
Dexter	20000	100	600	300	300	N/A
Aligned	2400	100	24016	16811	4803	2402
Grouped	2400	100	24016	16811	4803	2402
Flocking	2400	100	24016	16811	4803	2402

Table 4.2: Information of data set used, there is no separate testing set for Dorothea and Dexter, hence they are noted as N/A.

use a random projection to randomly project all samples to a 100-d subspace before training each classifier. We divided all samples into three groups, a training set (70%) for training, a validation set (20%) for choosing the parameter and a testing set (10%) to computer the error. For experiments with Dorothea and Dexter, due to the limited number of available samples, we will not further divide all samples to three groups. Instead we will only carry out testing with the validation set and only the validation error is recorded. All experiments are repeated 20 times.

	Dorothea	Dexter	Aligned	Grouped	Flocking
kNN	0.902(±0.001)	0.673(±0.032)	1(±0)	0.999(±0)	0.998(±0.001)
Histogram	0.892(±0.007)	0.674(±0.025)	0.994(±0.01)	0.992(±0.01)	0.999(±0.001)
Decision Tree	0.898(±0.005)	0.593(±0.026)	0.792(±0.069)	0.769(±0.079)	0.803(±0.093)

Table 4.3: Comparison of average classification accuracy using randomly projected data sets to $m = 100$ dimensions. The values in brackets represent one standard deviation. The decision tree used here is based on the Gini index splitting.

From Table 4.3 we find that Histogram and kNN have the best overall accuracy performance. There are slight variations between them, and kNN seems to perform a little better on the Dorothea data set, but these differences have not been found statistically significant. On the other hand, decision trees perform worse, most likely because random projection change the original features used for creating label-dependent splits into branches. The strength of random projection is that it preserves the distance structure, and by implication it preserves the similarities which both Histograms and kNN classifiers rely on. So it is perhaps unsuprising that random projection works best with these classifiers.

	Dorothea	Dexter	Aligned	Grouped	Flocking
kNN	0.21(±0.03)	0.35(±0.13)	29.75(±3.67)	27.76(±6.53)	26.99(±5.36)
Histogram	0.15(±0.03)	0.30(±0.1)	6.80(±3.49)	7.05(±2.96)	8.19(±1.95)
Decision Tree	0.40(±0.07)	0.40(±0.07)	11.50(±4.12)	13.46(±11.036)	16.97(±6.83)

Table 4.4: Average training time (seconds) of projected kNN and projected histogram. The values in brackets represent one standard deviation.

Table 4.4 shows the average time elapsed for training, including the time used for parameter tuning. We note that the histogram has better training time on average, compared to the other two algorithms over all data sets tested. Furthermore, the difference becomes more significant on the larger data sets (Aligned, Grouped and Flocking).

	Dorothea	Dexter	Aligned	Grouped	Flocking
kNN	0.014(±0.0004)	0.025(±0.001)	1.47(±0.30)	1.67(±1.30)	1.61(±0.38)
Histogram	0.002(±0.0007)	0.02(±0.004)	0.45(±0.18)	0.59(±0.26)	0.641(±0.15)
Decision Tree	0.0004(±0.0001)	0.0004(±0.0001)	0.001(±0.0001)	0.001(±0.0001)	0.001(±0.0001)

Table 4.5: Average testing time (seconds) of projected kNN and projected histogram. The values in brackets represent one standard deviation.

From Table 4.5 we observe that the histogram classifier has faster testing times on average compared to the kNN classifier, and again the differences are more significant for larger data sets. Decision tree has the best testing time over all data sets, however it lacks in terms of accuracy and training efficiency, as shown in Table 4.3 and Table 4.4. Furthermore, recall that testing efficiency for the histogram tends to gradually improve over time as explained in discussion of Alg. 2, and we can use the more efficient prediction rule of Alg. 1 for large-scale data sets.

Summary

We discussed how the 'curse of dimensionality' has affected the generalization convergence of kNN and presented the kNN algorithm in the compressed learning setting. We then proved the generalization error bound of the compressed kNN and showed that as long as the data samples has a simple structure, random projection can reduce the dimension of the samples without affecting the performance of kNN. We have pursued two different approaches to provide a generalization bound on the histogram classifier. We found that as n increases, after the optimisation with b^*, the bound we obtained with the finite hypothesis approach achieved the optimal rate of order $n^{-\frac{1}{d+2}}$ when the Bayes error is nonzero. The case splitting approach achieved the optimal rate of order $n^{-\frac{1}{d+1}}$ towards twice the Bayes error. In particular, a convergence rate of n^{-1} can be achieved if the Bayes error is zero.

We presented the compressed histogram algorithm and proved its generalization guarantee in a similar way as the compressed kNN. Moreover, we extended the result of the projected histogram to other variants of low-dimensional stuctures other than linear subspace. We found that as long as the majority of the data samples are within close distance to a linear hyperplane, then random projection does not affect the rate of error convergence for the histogram.

Finally, we illustrated the findings of our theory using high-dimensional data sets empirically. We showed that when the data samples has a low-dimensional structure then learning with projected samples has almost the exact same performance as predictors in high dimension. However, in the case that the data set indeed has a high-dimensional structure, then random projection can throw away important information of the samples during projection which can significantly reduces the accuracy of compressed learners. Furthermore, we showed that the difference of the run-time complexity between histogram and kNN is significant, in both training and testing. Histogram classifiers have a much better time and space complexity compared to kNN while achieving good accuracy level, the run-time difference is especially large when the data set is large or the dimensionality is high.

Chapter Five

Differentially Private Non-parametric Algorithms

In this Chapter, we extend the classic kNN classifier and histogram classifier to the differentially private setting. We will impose the Laplace mechanism to guarantee privacy for both kNN and histogram algorithms by analysing the sensitivity of their estimators. In the differentially private setting, it is expected that the algorithm will incur additional error due to the randomization by the privacy mechanism. However, it is unclear how large this magnitude of excess error will be comparing to their non-private analogues. Furthermore, does random projection affect the interplay between privacy and generalization guarantee? If so, how it is that privacy affects generalization and to what extend? In this Chapter, we will study this question for the first time in the context of kNN and histogram classification. A summary of our generalization results in the private setting is presented in Table. 5.1 for both projected and non-projected algorithms.

5.1 Differentially Private kNN

The kNN algorithm is known to achieve good accuracy in many classification problems and is widely used to complete learning tasks. In this section, we introduce the DP-kNN algorithm using the Laplace mechanism which guarantees ϵ_p-DP for the classical kNN algorithm. The DP guarantee comes from the addition of randomness by injecting

	Private kNN	NoisyReg
Projected	$\mathcal{O}\left(\left(\frac{(1+\epsilon_\Phi)N}{(1-\epsilon_\Phi)\epsilon_p}\right)n^{-\frac{1}{m+3}}\right)$ (Thm. 5.4)	$\mathcal{O}\left(\left(\frac{(1+\epsilon_\Phi)}{(1-\epsilon_\Phi)\epsilon_p}\right)n^{-\frac{1}{m+2}}\right)$ (Thm. 5.10)
Non-projected	$\mathcal{O}\left(N\epsilon_p^{-1}n^{-\frac{1}{d+3}}\right)$ (Thm. 5.3)	$\mathcal{O}\left(\epsilon_p^{-1}n^{-\frac{1}{d+2}}\right)$ (Thm. 5.7)

Table 5.1: A summary of the generalisation results, where d is the dimension m is the projection dimension. Privacy parameter is denoted by ϵ_p and N is the number of testing samples. We have omitted other smaller factors here for simplicity, refer to the corresponding theorem for a detailed view.

noise drawn from the Laplace distribution. Given a testing set S_{test} of size N, note that the k neighbours of each testing sample can overlap. i.e. the intersection of the set of neighbours for two different testing sample can be non-empty. Due to this restriction, we are required to distribute the total privacy budget ϵ_p to N equal portions and use each portion on one testing sample (one query). The prediction is then made using the noisy empirical regression estimator of kNN (line 8 of Alg. 3). The algorithm is outlined in Alg. 3.

Algorithm 3 Differentially Private kNN by Laplace Mechanism

```
1: Training set $S$, privacy parameter $\epsilon_p$ and testing set $S_{test}$ of size $N$.
2: procedure DP-KNN($S, \epsilon_p, S_{test}$)
3:     Given $N$ testing points $S_{test} = \{X_1, \dots, X_N\}$
4:     for each $i$ of the index set $[N] = \{1, \dots, N\}$ do
5:         find the $k$ nearest neighbour of $X := S_{test}[i]$
6:         choose $\alpha, \beta$ randomly from $Lap\left(\frac{N}{\epsilon_p}\right)$
7:     end for
8:     Record $\hat{\eta}(X) = \frac{\sum_{i=1}^k Y_i + \alpha}{k+\alpha+\beta}$, where $\sum_{i=1}^k Y_i$ is the sum of the labels of the $k$ neighbours of $X$.
9:     Predict 1 if $\hat{\eta}(X) \geq 1/2$, else 0.
10: end procedure
```

Remark 10. *Note that Alg. 3 splits up the privacy budget into N equal portions, where N is the number of testing points. This requires us to know the total number of testing samples beforehand, which may not be possible in practical situations. One way to get around this is to use an infinite series that sums to one (e.g. a geometric series) and have a decreasing privacy budget when we increase the number of predictions. However this can also lead to very small privacy budgets for each query and hence poor accuracy performance.*

Our first result is the privacy guarantee of DP-kNN.

Theorem 5.1. *The DP-kNN algorithm described in Alg.3 satisfies ϵ_p-DP.*

Proof. Given one testing point X, the global sensitivity of the number of counts in each label class is clearly 1, as adding or removing one training sample changes the count by at most 1. By the Laplace mechanism adding Laplace noise with variance parameter $1/\epsilon_p$ guarantees ϵ_p-DP for one prediction query. For N testing queries, since the neighbours of N testing samples may overlap, we need to use the sequential composition theorem 2.6 over N queries, which guarantees ϵ_p-DP with scale parameter N/ϵ_p. □

To analyse the generalisation error of the DP-kNN, we need to analyse the effect of the added noise on its predictions. Note that the Laplace distribution we use here has mean value 0 and scale parameter N/ϵ_p as required by the Laplace mechanism (10) to guarantee ϵ_p-DP, hence the noise drawn is from a fixed Laplace distribution. We can analyse the magnitude of the noise added for each prediction query. The following lemma bounds the error of the noisy empirical regression estimator of kNN with the true regression function. We prove the result for a general scale of $2/\epsilon_p$ and we can obtain the required result for kNN by dividing ϵ_p by N.

Lemma 5.2. *Let $k \geq 1$ and let $Y_1, \ldots, Y_k$ be independent Bernoulli random variables with $\mathbb{P}[Y_i = 1] = p_i$. Denote $p = \frac{1}{k}\sum_{i=1}^{k} p_i$ and $p' = \frac{\sum_{i=1}^{k} Y_i + \alpha}{k+\alpha+\beta}$ where α and β are independent Laplace noises with scale $2/\epsilon_p$. Then we have:*

$$\mathop{\mathbb{E}}_{Y_1,\ldots,Y_k,\alpha,\beta} \mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p' \geq 1/2]}] \leq p + \frac{2}{\sqrt{ek}} + \frac{5}{4k\epsilon_p}. \tag{5.1}$$

Proof. W.l.o.g. (by symmetry) assume that $p \leq 1/2$. Then, $\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p>1/2]}] = p$. Let

$Y' = \mathbb{1}_{[p'>1/2]}$. We have

$$\begin{aligned}
\mathop{\mathbb{E}}_{Y_1,\ldots,Y_k} \mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p'>1/2]}] - p &= \mathop{\mathbb{E}}_{Y_1,\ldots,Y_k} \left[\mathop{\mathbb{P}}_{Y\sim p}[Y=0]\mathbb{P}[p'>1/2] + \mathop{\mathbb{P}}_{Y\sim p}[Y=1]\mathbb{P}[p'\leq 1/2] \right] - p \\
&= \mathop{\mathbb{E}}_{Y_1,\ldots,Y_k} [(1-p)\mathbb{P}[p'>1/2] + p(1-\mathbb{P}[p'>1/2])] - p \\
&= \mathop{\mathbb{E}}_{Y_1,\ldots,Y_k} [(1-2p)\mathbb{P}[p'>1/2] + p] - p \\
&= \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k} [p'>1/2](1-2p).
\end{aligned} \tag{5.2}$$

Note that

$$\mathop{\mathbb{P}}_{Y_1,\ldots,Y_k}[p'>1/2] = \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k}\left[\frac{\sum_{i=1}^k Y_i + \alpha}{k+\alpha+\beta} > \frac{1}{2}\right] = \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k}\left[2\sum_{i=1}^k Y_i > k+\beta-\alpha\right]. \tag{5.3}$$

For any $c<0$, we have:

$$\begin{aligned}
\mathop{\mathbb{P}}_{Y_1,\ldots,Y_k,\alpha,\beta}\left[2\sum_{i=1}^k Y_i > k+\beta-\alpha\right] &\leq \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k,\alpha,\beta}\left[2\sum_{i=1}^k Y_i > k+\beta-\alpha \middle| \beta-\alpha>c\right]\mathbb{P}[\beta-\alpha>c] \\
&\quad + \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k,\alpha,\beta}\left[2\sum_{i=1}^k Y_i > k+\beta-\alpha \middle| \beta-\alpha<c\right]\mathbb{P}[\beta-\alpha<c] \\
&\leq \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k,\alpha,\beta}\left[2\sum_{i=1}^k Y_i > k+\beta-\alpha \middle| \beta-\alpha>c\right] + \mathbb{P}[\beta-\alpha<c] \\
&\leq \mathop{\mathbb{P}}_{Y_1,\ldots,Y_k}\left[2\sum_{i=1}^k Y_i > k+c\right] + \mathbb{P}[\beta-\alpha<c].
\end{aligned}$$

Now we can use Hoeffding's inequality on Y_i's:

$$\mathop{\mathbb{P}}_{Y_1,\ldots,Y_k}\left[\sum_{i=1}^k Y_i - pk > \frac{k+c}{2} - pk\right] \leq \exp\left(-2kt^2\right), \quad \text{where } t = \frac{(1-2p)k+c}{2k}.$$

Evaluating the exponential term and choosing $c = -ks/2$ ($s = 1-2p$), we have:

$$\exp\left(-2kt^2\right) \leq \exp\left(-\frac{s^2k}{8}\right). \tag{5.4}$$

We want to bound the quantity $(1-2p)\exp(-2kt^2)$, for simplicity we denote $s = 1-2p$ and f as:

$$f = s\exp\left(-\frac{s^2k}{8}\right). \tag{5.5}$$

By differentiating and equal to zero we find that

$$s^* = \arg\max_s f(s) = \frac{2}{k^{1/2}}. \tag{5.6}$$

Hence

$$f(s) \le \max_s f(s) = f(s^*) = \frac{2}{\sqrt{k}}\exp\left(-\frac{(4/k)k}{8}\right) = \frac{2}{\sqrt{ek}}.$$

We can also calculate the second term $\mathbb{P}[\beta - \alpha < c]$ by working out the probability density function of two i.i.d. Laplacian random variables. Note that the density of the difference of two independent random variable X, Y is given by:

$$f_{X-Y}(z) = \int_{-\infty}^{\infty} f_X(x)f_Y(x-z)dx. \tag{5.7}$$

For two i.i.d. Laplacian random variables X, Y with scale b and $z > 0$ we have:

$$\begin{aligned}
f_{X-Y}(z) &= \int_{-\infty}^{\infty} \frac{1}{2b}\exp(-|x|/b)\frac{1}{2b}\exp(-|x-z|/b)\ dx \\
&= \frac{1}{4b^2}\left[\int_{-\infty}^{0} \exp(x/b)\exp((x-z)/b) + \int_0^z \exp(-x/b)\exp((x-z)/b)\right. \\
&\qquad \left. + \int_z^{\infty} \exp(-x/b)\exp(-(x-z)/b)\right] \\
&= \frac{1}{4b^2}\left[\frac{b}{2}\exp(-z/b) + z\exp(-z/b) + \frac{b}{2}\exp(-z/b)\right] \\
&= \frac{1}{4b^2}\left[b\exp(-z/b) + z\exp(-z/b)\right].
\end{aligned}$$

Since from the symmetry of the Laplace distribution we also get symmetry of the difference between two Laplacian random variables about the origin. We have that

$$f_{X-Y}(z) = \frac{1}{4b^2}\left[b\exp(-|z|/b) + |z|\exp(-|z|/b)\right] \quad \text{for } -\infty < z < \infty. \tag{5.8}$$

Using this with our two Laplacian random variables α, β from a Laplace distribution with 0 mean and scale $2/\epsilon_p$ and $c < 0$, we have

$$\begin{aligned}
\mathop{\mathbb{P}}_{\alpha,\beta}[\beta - \alpha < c] &= \frac{\epsilon_p^2}{16} \int_{-\infty}^{c} \frac{2}{\epsilon_p} \exp(-|c|\epsilon_p/2) + |c| \exp(-|c|\epsilon_p/2)\ dc \\
&= \frac{\epsilon_p^2}{16} \left[\frac{4}{\epsilon_p^2} \exp(c\epsilon_p/2) + \left(\frac{2}{\epsilon_p} - c \right) \frac{2}{\epsilon_p} \exp(c\epsilon_p/2) \right] \\
&= \left(\frac{1}{2} - \frac{c\epsilon_p}{8} \right) \exp\left(\frac{c\epsilon_p}{2} \right) \qquad (c < 0).
\end{aligned}$$

Substituting $c = -ks/2$ we have:

$$(1-2p)\mathbb{P}[\beta - \alpha < c] = \left(\frac{s}{2} + \frac{s^2 k\epsilon_p}{16} \right) \exp\left(-\frac{sk\epsilon_p}{4} \right) = g(s). \tag{5.9}$$

By differentiation and equate to zero we found that:

$$s^{**} = \arg\max_s g(s) = \frac{4\sqrt{2}}{k\epsilon_p}. \tag{5.10}$$

Therefore:

$$g(s) \le \max_s g(s) = g(s^{**}) = \left(\frac{2\sqrt{2}}{k\epsilon_p} + \frac{2}{k\epsilon_p} \right) \exp(-\sqrt{2}) \le \frac{5}{4k\epsilon_p}.$$

Combining the two terms we conclude:

$$\mathop{\mathbb{P}}_{Y_1,\dots,Y_k}[p' > 1/2](1-2p) \le \frac{2}{\sqrt{ek}} + \frac{5}{4k\epsilon_p}.$$

Hence we conclude:

$$\mathop{\mathbb{E}}_{Y_1,\dots,Y_n} \mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p' \ge 1/2]}] \le p + \frac{2}{\sqrt{ek}} + \frac{5}{4k\epsilon_p}. \tag{5.11}$$

□

Remark 11. *The result of Lemma 5.2 bounds the deviation of the empirical noisy esti-*

mator p' with its expected value (with respect to the labels and the noise) p as stated in the Lemma. The three terms in the upper bound of the Lemma accounts for both empirical estimation error and additional error due to privacy. In particular, the final term quantifies the error due to added noise for privacy, we can also find the analogue of the first two terms for the non-private case in Lemma 4.4.

Using Lemma 5.2, we can now prove the generalisation of the private kNN using a similar analysis approach as for its non-private analogue because Lemma 5.2 has quantified the excess error due to privacy.

Theorem 5.3 (Generalisation of Private kNN). *Let $\mathcal{X} = [0,1]^d, \mathcal{Y} = \{0,1\}$, and $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$ for which the conditional probability function, η, is a L-Lipschitz function. Let $\hat{h}_p$ denote the result of applying the DP k-NN rule to a sample $S \sim \mathcal{D}^n$, where $k \geq 10$ and the number of test instance is N. Then for any $0 < \epsilon_p$ we have*

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{n^{1/(d+1)}} + \frac{2k}{n^{1/(d+1)}} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.12}$$

In particular, if $k = \mathcal{O}(n^{2/(d+2)})$, we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(d+3)}} + \frac{N}{\epsilon_p n^{2/(d+3)}}\right). \tag{5.13}$$

Proof. Let $\pi_j(X)$ denote the index of the j-th nearest neighbour of the sample X in S. Fix some $b > 0$ and let $C_1, \ldots, C_r$ be the cover of the feature space $\mathcal{X}$ using axis-aligned boxes of side lengths of $1/b$ each. For each $X, X' \in \mathcal{X}$ in the same box we have $\|X - X'\|_2 \leq \sqrt{d}/b$. Otherwise $\|X - X'\|_2 \leq \sqrt{d}$. Hence the expected generalisation error is

$$\begin{aligned}\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] = &\mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S| < k} \mathbb{P}[C_i]\right] \cdot \mathop{\mathbb{P}}_{S,A,(X,Y)}\left[\hat{h}_p(X) \neq Y \mid \forall j \in [k], \|X - X_{\pi_j(X)}\| \leq \sqrt{d}\right] \\ &+ \mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S| \geq k} \mathbb{P}[C_i]\right] \cdot \mathop{\mathbb{P}}_{S,A,(X,Y)}\left[\hat{h}_p(X) \neq Y \mid \forall j \in [k], \|X - X_{\pi_j(X)}\| \leq \sqrt{d}/b\right].\end{aligned}$$

By the law of total probability and Lemma 4.3 we obtain:

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \le \frac{2rk}{n} + \max_{i:|C_i\cap S|\ge k} \mathop{\mathbb{P}}_{S,A,(X,Y)} \left[\hat{h}_p(X) \neq Y \mid |C_{i(X)} \cap S| \ge k\right], \tag{5.14}$$

where $\pi_j(X)$ denotes the index of the j-th nearest neighbour of the sample X in S. Let $p = \frac{1}{k}\sum_{i=1}^{k} \eta(X_{\pi_i(X)})$ and $p' = \frac{\sum_{i=1}^{k} Y_i + \alpha}{k+\alpha+\beta}$ where α, β are the Laplace noises from $Lap(0, N/\epsilon_p)$. From Lemma 5.2 we have

$$\mathop{\mathbb{E}}_{Y_1,\dots,Y_j} \mathop{\mathbb{P}}_{Y\sim\eta(X)}[\hat{h}_p(X) \neq Y] - p \le \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.15}$$

Note that for $p \le 1/2$ we have $\mathop{\mathbb{P}}_{Y\sim p}[\mathbb{1}_{[p>1/2]} \neq Y] = p = \min\{p, 1-p\}$, and hence

$$\min\{p, 1-p\} \le \min\{\eta(X), 1-\eta(X)\} + |p - \eta(X)|$$

by a similar argument as in the proof of the claim above. Note that the error of the Bayes optimal classifier is

$$L_{\mathcal{D}}(f^*) = \mathop{\mathbb{E}}_{X}[\min\{\eta(X), 1-\eta(X)\}].$$

Hence we obtain:

$$\mathop{\mathbb{E}}_{A,Y_1,\dots,Y_j}\left[\mathop{\mathbb{P}}_{Y\sim p}[\hat{h}_p(X) \neq Y]\right] \le L_{\mathcal{D}}(f^*) + \mathbb{E}[|p - \eta(X)|] + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.16}$$

Using the fact that η is L-Lipschitz we have

$$|p - \eta(X)| \le L \sup \|X - X'\|_2 \le \frac{L\sqrt{d}}{b}. \tag{5.17}$$

Combining equations (5.14),(5.15),(5.1) we obtain

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \le L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \frac{2rk}{n} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.18}$$

Finally, letting $b = n^{1/(d+1)}$ we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{n^{1/(d+1)}} + \frac{2k}{n^{1/(d+1)}} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.19}$$

By optimizing k, b simultaneously, we found that with $b = n^{1/(d+3)}$ and $k = n^{2/(d+3)}$, we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(d+3)}} + \frac{N}{\epsilon_p n^{2/(d+3)}}\right). \tag{5.20}$$

□

5.1.1 DP compressed kNN

Learning with compressed data sets is beneficial in the non-private setting of kNN as we have discussed in Section 4.1.1. Random projection can reduce the dimension dependence of its generalisation error bound which can improve the rate of convergence and computational efficiency of the kNN algorithm.

In this section, we introduce the kNN algorithm in compressed learning under the differential privacy constraint. We show that the benefits of random projection for classic kNN also apply in the privacy setting. Recall that for a sample set S and a random projection matrix Φ, we denote the projected sample set as $S_\Phi = \{(\Phi X, Y)|(X, Y) \in S\}$.

Theorem 5.4 (Private kNN with RP). *Let $\hat{h}_{\Phi p}$ denote the result of applying the k-NN rule to the projected sample set S_Φ, where $k \geq 10$ and the number of test instances is N. Assume that η is L-Lipschitz function. Let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. Then for any $\epsilon_p > 0$ and $0 < \epsilon_\Phi, \delta_\Phi < 1$, let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2. \tag{5.21}$$

Then we have with probability at least $1 - \delta_\Phi$:

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_{\Phi p})] \leq L_{\mathcal{D}}(f^*) + \frac{L(1+\epsilon_\Phi)\sqrt{d}}{(1-\epsilon_\Phi)n^{1/(m+1)}} + \frac{2k}{n^{1/(m+1)}} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.22}$$

In particular, if $k = \mathcal{O}(n^{2/(m+3)})$, *we have*

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_{\Phi p})] \leq L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L(1+\epsilon_\Phi)\sqrt{d}}{(1-\epsilon_\Phi)n^{1/(m+3)}} + \frac{N}{\epsilon_p n^{2/(m+3)}}\right). \tag{5.23}$$

Proof. From a similar analysis as in Theorem 4.7 we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_{\Phi p})] \leq \frac{2rk}{n} + \max_{i:|C_i \cap S| \geq k}\left\{\mathop{\mathbb{P}}_{S,A,(X,Y)}\left[\hat{h}_{\Phi p}(X) \neq Y \mid \forall j \in [k], \|\Phi X - \Phi X_{\pi_j(\Phi X)}\| \leq D_\Phi/b\right]\right\},$$

where $\pi_j(\Phi X)$ denotes the index of the j-th nearest neighbour of the sample ΦX in S. Let $p = \frac{1}{k}\sum_{i=1}^k \eta(X_{\pi_i(\Phi X)})$ and $p' = \frac{\sum_{i=1}^k Y_i + \alpha}{k+\alpha+\beta}$ where α, β are the Laplace noises from $Lap(0, N/\epsilon_p)$. W.l.o.g. assume that $p \leq 1/2$. From Lemma 5.2 we have

$$\mathop{\mathbb{E}}_{A,Y_1,\ldots,Y_j}\mathop{\mathbb{P}}_{Y\sim\eta(X)}[\hat{h}_{\Phi p}(X) \neq Y] - p \leq \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.24}$$

Note that for $p \leq 1/2$ we have $\mathop{\mathbb{P}}_{Y\sim p}[\mathbb{1}_{[p>1/2]} \neq Y] = p = \min\{p, 1-p\}$, and hence by Lemma 4.6 we have

$$\min\{p, 1-p\} \leq \min\{\eta(X), 1-\eta(X)\} + |p - \eta(X)|.$$

Note that the error of the Bayes optimal classifier is

$$L_{\mathcal{D}}(h^*) = \mathop{\mathbb{E}}_{X}[\min\{\eta(X), 1-\eta(X)\}].$$

Combining our results we obtain:

$$\mathop{\mathbb{E}}_{A,Y_1,\ldots,Y_j}\left[\mathop{\mathbb{P}}_{Y\sim p}[\hat{h}_{\Phi p}(X) \neq Y]\right] \leq L_{\mathcal{D}}(f^*) + \mathbb{E}[|p - \eta(X)|] + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}.$$

Using the fact that η is L-Lipschitz we have

$$|p - \eta(X)| \leq L \sup \|X - X'\|_2. \tag{5.25}$$

Now applying the JL-lemma we have

$$\sup \|X - X'\|_2 \leq \sup \frac{\|\Phi X - \Phi X'\|}{1 - \epsilon_\Phi} \leq \frac{D_\Phi}{b(1 - \epsilon_\Phi)}, \tag{5.26}$$

where D_Φ is the diameter of the projected sample set and $1/b$ is the width of the covering boxes. Since our original sample space is $[0,1]^d$ hence $\|X\|_2^2 \leq d$. By the JL-lemma we have $D_\Phi \leq (1 + \epsilon_\Phi)\sqrt{d}$. Combining our results we obtain:

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq L_\mathcal{D}(h^*) + \frac{L(1 + \epsilon_\Phi)\sqrt{d}}{b(1 - \epsilon_\Phi)} + \frac{2b^m k}{n} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.27}$$

Finally, letting $b = n^{1/(m+1)}$ we have

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq L_\mathcal{D}(f^*) + \frac{L(1 + \epsilon_\Phi)\sqrt{d}}{(1 - \epsilon_\Phi)n^{1/(m+1)}} + \frac{2k}{n^{1/(m+1)}} + \frac{2}{\sqrt{ek}} + \frac{5N}{4k\epsilon_p}. \tag{5.28}$$

By optimizing k, b simultaneously, we found that with $b = n^{1/(m+3)}$ and $k = n^{2/(m+3)}$, we have

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq L_\mathcal{D}(f^*) + \mathcal{O}\left(\frac{L(1 + \epsilon_\Phi)\sqrt{d}}{(1 - \epsilon_\Phi)n^{1/(m+3)}} + \frac{N}{\epsilon_p n^{2/(m+3)}}\right). \tag{5.29}$$

□

We observe from theorem 5.4 that the dimensionality in the exponential is reduced to the projection dimension instead. The cost is the additional factor due to projection distortion as we observed similarly for the non-private setting. Reduction in dimensionality did not seem to help with the privacy term (last term) as there is no dependence on d or m. However, we should note that for the classifier to converge to the Bayes error, we must choose the parameter k to increase as n increases. Furthermore, recall from theorem 4.2 that the theoretical optimal k is dependent on the dimension d. Hence a reduction in dimension in theorem 5.4 actually improves the privacy term by allowing a larger choice of k in its denominator, although this is not immediate from the theorem.

5.2 Differentially Private Histogram Classifier

In this section, we present two private algorithms and two private classifiers induced by the algorithms. Both algorithms add noise from the Laplace distribution; we write $Lap(\sigma)$ to denote the Laplace distribution with mean 0 and scale σ (the mean is always zero in our analysis).

5.2.1 Private Classifier by NoisyReg

Ideally, if the sample set we have is large, we will expect to add less noise since the sample set with a large size will be more stable in the sense that changing one sample will not causes a significant change in the output.

We note that the histogram classifier can be completely described by a rule that finds the cell index of a point X, which has been previously denoted by $i(X)$, and the regression estimator in each cell $\hat{\eta}$. Hence to guarantee privacy of the histogram, we simply need to guarantee privacy for its regression estimator because $i(X)$ is not dependent on the sample set S. We will impose the Laplace mechanism in our approach and guarantee DP for the histogram. This is formally described in Algorithm 4.

Algorithm 4 NoisyReg

```
1: Set $\hat{\eta}(S) = (\hat{\eta}_1, \dots, \hat{\eta}_r)$ where $\hat{\eta}_j$ is the fraction of points in class "1" in cell $j$
2: Let $n_j$ be the number of sample points in cell $j$ in the training sample $S$
3: Set parameter $\epsilon_p$.
4: procedure NoisyReg($S, \epsilon_p$)
5:     for each $j$ in the index set $[r] = \{1, \dots, r\}$ do
6:         calculate $n_j$
7:         if $n_j \geq 1$ then
8:             Set $\Sigma = \frac{1}{n_j \epsilon_p}$
9:         else
10:            Set $\Sigma = \frac{1}{\epsilon_p}$
11:        end if
12:        draw $\beta_j$ from $Lap(\Sigma)$
13:    end for
14:    Output $\hat{\eta}(S) + (\beta_1, \dots, \beta_r)$
15: end procedure
```

Our first claim is that Algorithm 4 satisfies ϵ_p-differential privacy.

Theorem 5.5. *The mechanism **NoisyReg** defined in Algorithm 4 satisfies ϵ_p-differential privacy.*

The proof of the privacy guarantee follows directly from the definition of the Laplace mechanism and the parallel composition theorem. Hence we just need to evaluate its sensitivity.

Theorem 5.6. *The global sensitivity of $\hat{\eta}(S)$ in Alg. 4 is bounded by $1/n_j$ for the j-th cell.*

Proof. First we consider the case where $n_i > 1$. Let k_i be the number of points in cell i with label 1. Let S be a sample set of size n, we consider the effect of changing one labelled point on the value of $\hat{\eta}_i$. We distinguish 5 possible cases, as follows.

1. If we change a point in cell i with different label but the point remains in cell i, the change to $\hat{\eta}_i$ is :
$$\left|\frac{k_i}{n_i} - \frac{k_i \pm 1}{n_i}\right| = \left|\frac{1}{n_i}\right|. \tag{5.30}$$

2. If we add a point to cell i with label 1:
$$\left|\frac{k_i}{n_i} - \frac{k_i + 1}{n_i + 1}\right| = \left|\frac{k_i - n_i}{n_i(n_i + 1)}\right|. \tag{5.31}$$

3. If we add a point to cell i with label 0:
$$\left|\frac{k_i}{n_i} - \frac{k_i}{n_i + 1}\right| = \left|\frac{k_i}{n_i(n_i + 1)}\right|. \tag{5.32}$$

4. If we remove a point in cell i with label 1:
$$\left|\frac{k_i}{n_i} - \frac{k_i - 1}{n_i - 1}\right| = \left|\frac{k_i - n_i}{n_i(n_i - 1)}\right|. \tag{5.33}$$

5. If we remove a point in cell i with label 0:

$$\left|\frac{k_i}{n_i} - \frac{k_i}{n_i - 1}\right| = \left|\frac{k_i}{n_i(n_i-1)}\right|. \tag{5.34}$$

In all cases, the sensitivity is upper bounded by $\frac{1}{n_i}$. If $n_i = 0$ then the sensitivity is clearly 1 since adding a sample with arbitrary label cause the regression estimator to be either 0 or 1. □

From the output of **NoisyReg**, we can define the corresponding private classifier. For an unknown test point X, we find the cell index $C_{i(X)}$ where X belongs and we set the private classifier as follows.

$$\hat{h}_p(X) = \begin{cases} 1 & \text{if } NoisyReg(S, \epsilon_p)_{C_{i(X)}} \geq 1/2, \\ 0 & \text{otherwise.} \end{cases} \tag{5.35}$$

By the property of differential privacy, privacy will be preserved for post-processing [48]. Hence our output classifier is also ϵ_p-DP.

Theorem 5.7 (Generalisation of NoisyReg). *Let $\hat{h}_p$ be the private and non-private classifier respectively. Assume that $\eta(X)$ is an L-Lipschitz function and let ϵ_p denote the privacy budget. Then for any $b \in \mathbb{N}$, we have:*

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2b^d+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{b^d}{e\epsilon_p n}. \tag{5.36}$$

In particular, if $b = \mathcal{O}(n^{1/(d+2)})$, we have

$$\mathbb{E}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] = L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(d+2)}} + \frac{1}{\epsilon_p n^{2/(d+2)}}\right). \tag{5.37}$$

Note that in the case where we take ϵ_p to infinity (meaning 0 privacy) we get back to our original bound on the non-private classifier (Thm 4.12).

Proof. Fix X and consider the possible error for one particular cell, say cell j and recall it has sensitivity bounded by $\frac{1}{n_j}$. For simplicity, assume that $\hat{\eta}_j < 0.5$, as the opposite case follows by symmetry. For any fixed $\hat{\eta}_j$, the probability that the noise b_j will cause $\hat{\eta}_j$ to pass the threshold 0.5 is

$$\begin{aligned}
\mathbb{P}_A\left[\hat{\eta}_j + b_j \geq \frac{1}{2} \,|\hat{\eta}_j < \frac{1}{2}\right] &= \int_{0.5-\hat{\eta}_j}^{\infty} \frac{\epsilon_p n_j}{2} \exp\left(-\frac{|x|\epsilon_p n_j}{2}\right) dx && (5.38)\\
&= \frac{1}{2} - \int_{0}^{0.5-\hat{\eta}_j} \frac{\epsilon_p n_j}{4} \exp\left(-\frac{|x|\epsilon_p n_j}{2}\right) dx && (5.39)\\
&= \frac{1}{2} \exp\left(-\frac{|2\hat{\eta}_j - 1|\epsilon_p n_j}{2}\right). && (5.40)
\end{aligned}$$

For the case where $\hat{\eta}_j > 0.5$ we have a similar proof by symmetry. Hence combining the two cases we have for all $X \in C_j$, j is fixed,

$$\begin{aligned}
\mathbb{P}_A\left[\hat{h}_p(X) \neq \hat{h}(X) | X \in C_j\right] &= \mathbb{P}\left[\hat{\eta}_j + b_j \geq \frac{1}{2} \,|\hat{\eta}_j < \frac{1}{2}\right] \mathbb{P}\left[\hat{\eta}_j < \frac{1}{2}\right]\\
&\quad + \mathbb{P}\left[\hat{\eta}_j + b_j \geq \frac{1}{2} \,|\hat{\eta}_j \geq \frac{1}{2}\right] \mathbb{P}\left[\hat{\eta}_j \geq \frac{1}{2}\right]\\
&= \frac{1}{2} \exp\left(-\frac{|2\hat{\eta}_j - 1|\epsilon_p n_j}{2}\right) \left(\mathbb{P}\left[\hat{\eta}_j < \frac{1}{2}\right] + \mathbb{P}\left[\hat{\eta}_j \geq \frac{1}{2}\right]\right)\\
&= \frac{1}{2} \exp\left(-\frac{|2\hat{\eta}_j - 1|\epsilon_p n_j}{2}\right). && (5.41)
\end{aligned}$$

Now, suppose the worst case, i.e. that the added noise has flipped the label prediction for the cell C_j; that is, for all $X \in C_j, \hat{h}_p(X) \neq \hat{h}(X)$. Denote by $S_j = \{X \in S | X \in C_j\}$ the set of samples contained in C_j. Then the maximum empirical error difference over all $X \in C_j$ is as follows:

$$\begin{aligned}
L_{S_j}(\hat{h}_p) - L_{S_j}(\hat{h}) &\leq \left| \frac{1}{n_j} \sum_{Z \in S_j} \mathbb{1}_{\{\hat{h}(X) \neq Y\}} - \frac{1}{n_j} \sum_{Z \in S_j} \mathbb{1}_{\{\hat{h}(X) = Y\}} \right|\\
&= \left| \frac{1}{n_j} \sum_{Z \in S_j} \mathbb{1}_{\{Y=1\}} - \frac{1}{n_j} \sum_{Z \in S_j} \mathbb{1}_{\{Y=0\}} \right|\\
&= |\hat{\eta}_j - (1 - \hat{\eta}_j)| = |1 - 2\hat{\eta}_j|, && (5.42)
\end{aligned}$$

Now combining eq. (5.42) with eq.(5.41), we have

$$\begin{aligned}\mathbb{E}_A\left[L_{S_j}(\hat{h}_p) - L_{S_j}(\hat{h}) | \hat{h}_p(X) \neq \hat{h}(X)\right]&\mathbb{P}_A[\hat{h}_p(X) \neq \hat{h}(X) | X \in C_j] \\ &\leq |1 - 2\hat{\eta}_j| \mathbb{P}_A[\hat{h}_p(X) \neq \hat{h}(X) | X \in C_j] \\ &= \frac{|1 - 2\hat{\eta}_j|}{2} \exp\left(-\frac{|2\hat{\eta}_j - 1|\epsilon_p n_j}{2}\right). \qquad (5.43)\end{aligned}$$

Note that the this is a random quantity depending on the cell j, hence we need to sum over all possibilities of j. Also note that the probability of getting a sample in S_j is n_j/n. Hence the total excess empirical error of $\hat{h}_p$ is bounded by

$$\begin{aligned}|L_S(\hat{h}_p) - L_S(\hat{h})| &\leq \sum_{j=1}^{r} \frac{|1 - 2\hat{\eta}_j| n_j}{2n} \exp\left(-\frac{|2\hat{\eta}_j - 1|\epsilon_p n_j}{2}\right) \\ &\leq \sum_{j=1}^{r} \max_a \left\{\frac{a n_j}{2n} \exp\left(-\frac{a\epsilon_p n_j}{2}\right)\right\} \\ &\leq \sum_{j=1}^{r} \frac{n_j}{2n}\left(\frac{2}{e\epsilon_p n_j}\right) \\ &= \sum_{j=1}^{r} \frac{1}{e\epsilon_p n} \\ &= \frac{r}{e\epsilon_p n}, \qquad (5.44)\end{aligned}$$

where the second to third line is due to the inequality $\max_a\{a\exp(-am)\} \leq 1/em$. Now consider the following error decomposition:

$$\begin{aligned}L_{\mathcal{D}}(\hat{h}_p) - L_{\mathcal{D}}(h^*) &= (L_{\mathcal{D}}(\hat{h}_p) - L_S(\hat{h}_p)) + (L_S(\hat{h}_p) - L_S(\hat{h})) + (L_S(\hat{h}) - L_{\mathcal{D}}(h^*)) \\ &\leq (L_{\mathcal{D}}(\hat{h}_p) - L_S(\hat{h}_p)) + (L_S(\hat{h}_p) - L_S(\hat{h})) + (L_S(h^*) - L_{\mathcal{D}}(h^*)) \\ &\leq (L_{\mathcal{D}}(\hat{h}_p) - L_S(\hat{h}_p)) + (L_S(h^*) - L_{\mathcal{D}}(h^*)) + \frac{r}{e\epsilon_p n} \qquad (5.45)\end{aligned}$$

where the second line is because $\hat{h}$ is an empirical risk minimiser and the last line follows from equation (5.44). The other two terms are due to the estimation of the empirical risk to the generalization risk. Recall that for a finite hypothesis class $\mathcal{H}$, we have the following uniform convergence property over all $h \in \mathcal{H}$ (see e.g. [115], section 4.2):

$$\mathbb{P}_S \left[|L_S(h) - L_{\mathcal{D}}(h)| > \epsilon \right] \leq 2|\mathcal{H}| \exp(-2n\epsilon^2), \forall \epsilon > 0 \tag{5.46}$$

Hence by substituting $\epsilon = \epsilon/2$ and applying this result to both $\hat{h}_p$ and $\hat{h}^*$, we can use the same argument as in proof of Thm. 4.10, start from equation (4.23) and obtain

$$(L_{\mathcal{D}}(\hat{h}_p) - L_S(\hat{h}_p)) + (L_S(h^*) - L_{\mathcal{D}}(h^*)) \leq \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} \tag{5.47}$$

Finally, combining with inequality (5.45), we obtain

$$\begin{aligned} \mathbb{E}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] &\leq L_{\mathcal{D}}(h^*) + \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{r}{e\epsilon_p n} && (5.48) \\ &\leq L_{\mathcal{D}}(f^*) + \frac{L\sqrt{d}}{b} + \sqrt{\frac{2r+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{r}{e\epsilon_p n} && (5.49) \end{aligned}$$

where the last line follows from equation (4.28).

In particular, if $b = \mathcal{O}(n^{1/(d+2)})$, we have

$$\mathbb{E}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] = L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L\sqrt{d}}{n^{1/(d+2)}} + \frac{1}{\epsilon_p n^{2/(d+2)}} \right). \tag{5.50}$$

□

From result of theorem 5.7 and theorem 4.10 we immediately obtain the following corollary on the excess risk due to privacy.

Corollary 5.8 (Expected excess risk of NoisyReg). *Let $\hat{h}_p$ be the DP histogram from Alg. 4, and $\hat{h}$ the non-private histogram classifier from Alg. 1. For any privacy parameter $\epsilon_p > 0$ and any positive integer b, the excess error of the DP-histogram due to privacy is bounded by*

$$\mathbb{E}_{S,A}\left[|L_{\mathcal{D}}(\hat{h}_p) - L_{\mathcal{D}}(\hat{h})| \right] \leq \frac{b^d}{e\epsilon_p n}, \tag{5.51}$$

In particular, if $b = \mathcal{O}(n^{1/(d+2)})$, we have

$$\mathop{\mathbb{E}}_{S,A}[|L_{\mathcal{D}}(\hat{h}_p) - L_{\mathcal{D}}(\hat{h})|] \leq \mathcal{O}\left(\frac{1}{\epsilon_p n^{2/(d+2)}}\right). \tag{5.52}$$

Similarly to the non-private case, we can achieve a significantly better rate in the realizable setting.

Theorem 5.9. *Let $\hat{h}_p$ be the private classifier in Alg. 4. Assume that $\eta(X)$ is an L-Lipschitz function and $L_{\mathcal{D}}(f^*) = 0$. For any $0 < \epsilon_p$, we have:*

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq \mathcal{O}\left(\frac{(L\sqrt{d})^d}{n} + \frac{(L\sqrt{d})^d}{n\epsilon_p}\right). \tag{5.53}$$

Proof. Recall from Thm. 4.16 that for $L_{\mathcal{D}}(f^*) = 0$, if $b = 2L\sqrt{d}$ then we have for any positive integer k

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \frac{2b^d k}{n}. \tag{5.54}$$

We substitute in this result with theorem 5.8 and obtain the following

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq \frac{2kb^d}{n} + \frac{b^d}{e\epsilon_p n}. \tag{5.55}$$

Substitute in our choice of b (same argument as in Thm. 4.16) and letting $k = 10$, we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_p)] \leq \frac{20(2L\sqrt{d})^d}{n} + \frac{(2L\sqrt{d})^d}{e\epsilon_p n} = \mathcal{O}\left(\frac{(L\sqrt{d})^d}{n} + \frac{(L\sqrt{d})^d}{n\epsilon_p}\right). \tag{5.56}$$

□

Thm. 5.9 quantified the loss of the classifier from NoisyReg and its error rate convergence with respect to the sample set size and the privacy parameter. The convergence rate of Thm. 5.9 is almost the same to the result of Thm. 4.16 (up to constant factors and privacy parameter).

Remark 12. *By comparing Thm. 5.9 and Thm. 4.16, we note that the second term of Thm. 5.9 represents the excess error induced by privacy compared to the non-private classifier. In particular, as the privacy parameter approaches infinity $\epsilon_p \to \infty$, the second term will vanish and we obtain the exact same error convergence as in Thm. 4.16.*

5.3 DP Compressed Histogram Classifiers

From the result of Section 4.4, we see that random projection can improve the algorithm by reducing the dimension of the sample set whenever $w(S)$ is smaller than the intrinsic dimension of S. We now ask the question: Does random projection help with privacy as well? It turns out that the answer to this is yes. By reducing the dimension of the data set before running the private algorithm, we can improve the noise level required to guarantee privacy as we will be working on a lower-dimensional space. Recall that S_Φ denotes the projected sample set from S and $\hat{h}_\Phi$ denotes the ERM output classifier using the compressed data set S_Φ. We prove the accuracy guarantee of the private classifier with **NoisyReg** under the compressed learning setting as shown below.

Theorem 5.10. *Let $\hat{h}_{\Phi p}$ denote the private histogram obtained after applying the private mechanism **NoisyReg** (Alg. 4) on the regression estimator of $\hat{h}_\Phi$. Assume that $\eta(X)$ is an L-Lipschitz function and denote the privacy budget by ϵ_p. Then for any $b \in \mathbb{N}$, $0 < \epsilon_\Phi, \delta_\Phi < 1$, and privacy parameter $\epsilon_p > 0$, let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)} \right]^2 . \tag{5.57}$$

Then we have with probability at least $1 - \delta$:

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq L_\mathcal{D}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m + 2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{b^m}{e\epsilon_p n}. \tag{5.58}$$

In particular, if $b = \mathcal{O}(n^{1/(m+2)})$, we have

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_{\Phi p})] \leq L_{\mathcal{D}}(f^*) + \mathcal{O}\left(\frac{L(1+\epsilon_\Phi)\sqrt{d}}{(1-\epsilon_\Phi)n^{1/(m+2)}} + \frac{1}{\epsilon_p n^{2/(m+2)}}\right). \tag{5.59}$$

Proof. Note that

$$L_{\mathcal{D}}(\hat{h}_{\Phi p}) = L_{\mathcal{D}}(\hat{h}_\Phi) + (L_{\mathcal{D}}(\hat{h}_{\Phi p}) - L_{\mathcal{D}}(\hat{h}_\Phi)). \tag{5.60}$$

We can bound the first term by Thm. 4.19. For the second term, since $\hat{h}_\Phi$ is an ERM classifier in lower dimension, we can use the excess error bound of Thm. 5.8 and obtain

$$\mathbb{E}_{S,A}[|L_{\mathcal{D}}(\hat{h}_{\Phi p}) - L_{\mathcal{D}}(\hat{h}_\Phi)|] \leq \frac{b^m}{e\epsilon_p n}. \tag{5.61}$$

Hence combining with Thm. 4.19 we obtain:

$$\mathop{\mathbb{E}}_{S,A}[L_{\mathcal{D}}(\hat{h}_{\Phi p})] \leq L_{\mathcal{D}}(f^*) + \frac{2L(1+\epsilon_\Phi)\sqrt{d}}{b(1-\epsilon_\Phi)} + \sqrt{\frac{2b^m+2}{n}} + \frac{\sqrt{\pi}}{\sqrt{2n}} + \frac{b^m}{e\epsilon_p n}. \tag{5.62}$$

$\square$

Theorem 5.10 implies that, if the condition for the projection dimension m is satisfied, then we can reduce the dimension dependence in the bound at the price of a small multiplicative factor that depends on ϵ_Φ. However, what does this mean for privacy? In general, our optimal choice of the parameter b from the error bound is of order $n^{\frac{1}{d+2}}$. Hence by reducing the dimension dependence the new optimal choice of b becomes of order $n^{\frac{1}{m+2}}$ – which is smaller – this can be verified easily by using a similar analysis as in Cor. 4.12. Hence, in the projected setting, we have a significantly smaller number of cells to classify. Since our private algorithm adds noise to each cell, this means that in the projected setting we add less noise into our model. In turn, reducing the amount of random noise that we add will improve the accuracy, as the private classifier now becomes more similar to the non-private algorithm.

Finally, we extend the result to the realisable case for faster convergence, as follows.

Theorem 5.11 (Generalisation of DP compressive histogram in realisable case)**.** *Let $\hat{h}_{\Phi p}$ be the private histogram obtained on RP-ed data, let $\epsilon_p > 0$, $b \in \mathbb{N}$, $0 < \epsilon_\Phi < 1/2, 0 < \delta_\Phi < 1$, and let m be the same as in Thm. 5.10. In addition assume $L_\mathcal{D}(f^*) = 0$. Then, for $b \geq 3L\sqrt{md}$, we have with probability with probability at least $1 - \delta_\Phi$:*

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq \mathcal{O}\left(\frac{(L\sqrt{d})^m}{n} + \frac{(L\sqrt{d})^m}{\epsilon_p n}\right). \tag{5.63}$$

Proof. Note that

$$L_\mathcal{D}(\hat{h}_{\Phi p}) = L_\mathcal{D}(\hat{h}_\Phi) + (L_\mathcal{D}(\hat{h}_{\Phi p}) - L_\mathcal{D}(\hat{h}_\Phi)). \tag{5.64}$$

Recall from the proof of Thm. 4.21 that for $L_\mathcal{D}(f^*) = 0$, if $b = 3L\sqrt{d}$ and $\epsilon_\Phi < 1/2$ then we have for any positive integer k

$$\mathop{\mathbb{E}}_{S}[L_\mathcal{D}(\hat{h}_\Phi)] \leq \frac{2b^m k}{n} \leq \mathcal{O}\left(\frac{L^m (d)^{m/2} k}{n}\right). \tag{5.65}$$

We substitute in this result with theorem 5.8 and choosing $k = 10$, we obtain the following

$$\mathop{\mathbb{E}}_{S,A}[L_\mathcal{D}(\hat{h}_{\Phi p})] \leq \mathcal{O}\left(\frac{L^m (d)^{m/2}}{n} + \frac{(L\sqrt{d})^m}{\epsilon_p n}\right). \tag{5.66}$$

□

Similarly to the non-private case, Thm. 5.11 shows that as long as the distortion parameter is not too big ($\leq 1/2$), then with the appropriate choice of b we can significantly improve the rate of convergence. Note, as before, that realisability is only required of the uncompressed high dimensional problem for this improved rate.

5.4 Empirical analysis

In this section, we analyse the performance of the DP-algorithms presented in this chapter experimentally. We focus on the accuracy performance of NoisyReg histograms (Alg. 4) with DP-kNN (Alg. 3). Since both algorithm guarantees privacy by adding noise, the time complexity comparison between them will be very similar to the non-private case which is analysed in Chapter 4.

Note that since the DP-kNN algorithm requires to know the number of testing instances before making predictions, validation process of choosing the the best parameter k will requires further splitting of privacy budget and lead to inaccurate result. Hence in all of our experiments, we will use a default value of k to avoid extensive split of the privacy budget. By testing for a range of potential values of k, we found that $k = 2N$ is a good choice in general, where N is the total number of testing samples. For k values smaller than this, the noise added for privacy seem to be overwhelming and lead to inaccurate predictions. On the other hand, larger values of k can reduce the variance of added noise, but also considers the label of far away samples which can lead to under-fitting, which is especially likely if N is large.

5.4.1 Varying the privacy parameter

We first analyse the accuracy performance of NoisyReg and DP-kNN under different privacy levels. We will use two synthetic data sets generated by Gaussian distributed clusters, each with a size of 3000. For the NoisyReg we will use 80% of the samples for training, 10% for validation and the rest for testing. For the DP-kNN we will use 90% of the samples for training and the rest for testings, where a default value of $k = 2N$ is used. All experiments are repeated 20 times and the results are plotted in Fig. 5.1.

From Fig. 5.1 it is clear that the NoisyReg has a significant advantage in accuracy performance over DP-kNN in all privacy levels, except in the 20-d case with $\epsilon_p = 0.01$ where the added noise has overwhelmed the signal in both algorithms. For the easier 10-d prob-

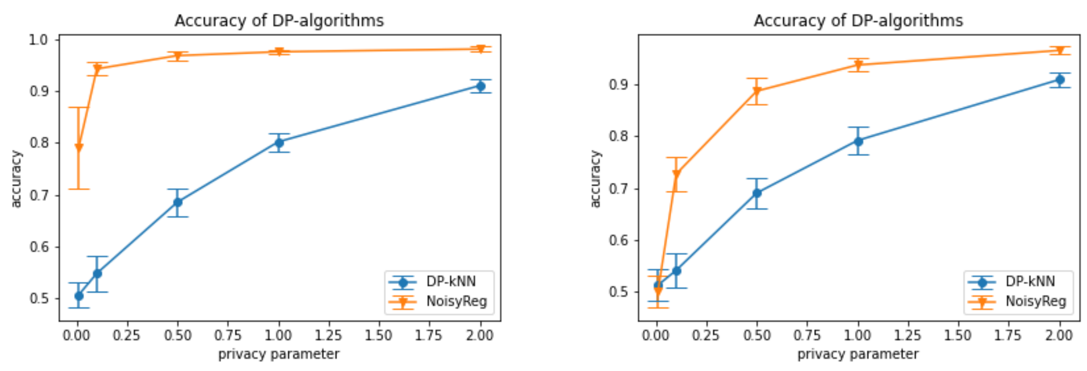

Figure 5.1: Accuracy performance with varying privacy parameter. Left plot uses a 10-d data set and the right plot uses a 20-d data set.

lem, NoisyReg achieves good performance with a low privacy budget but DP-kNN has a very similar convergence curve in both 10-d and 20-d experiments. The intuition for this observation is that the choice of k is fixed for both set of samples, when the problem is easy kNN should adapt with a smaller value of k for fast convergence. However, in the privacy setting, this may not be a good idea not only because choosing k by validation is an expensive task, but also because choosing a smaller k will result in increased variance of added noise. This matches the observation from our theory (Thm. 5.3) which showed that decreasing k will increase the excess error induced by privacy but can also decrease the error of the bias term. From the slower convergence of the accuracy in the 20-d experiment, we can see that the 'curse of dimensionality' indeed further increases the complexity of the problem in the private setting, especially when the privacy budget is small (≤ 1).

5.4.2 Comparison of private non-parametric algorithms

To analyse the practical performance of NoisyReg and DP-kNN, we tested both algorithms using a variety of real data sets. The size of the data sets range from 747 to 30000 and the experiment setting is identical to the previous section (5.4.1), a brief description of each data set used is presented in Tab. 5.2.

We compare the accuracy performance of NoisyReg and DP-kNN using the data sets listed in Tab. 5.2, we will also include a version of DP decision trees in our comparison,

	Size	Attributes	Problem description
Adults	32561	14	Predict whether income exceeds \$50K/yr based on census data
Bank	4520	16	predict if a client will subscribe a term deposit
Banknotes	1372	4	classify genuine and forged banknote-like specimens
Transfusion	747	4	whether he/she donated blood in March 2007
Claves	10787	16	predict clave-direction according to the partido-alto-based paradigm
Credit Card	30000	24	predict existence of default payment
Mushroom	8124	7	classify poisonous mushrooms
Occupancy	8142	7	Occupancy detection of rooms
Synthetic	3000	10/20	normal distributed clusters

Table 5.2: Data set information

namely, random decision trees (RDT) by [71]. Decision tree is another non-parametric classification algorithm that is widely used and RDT is one of the benchmark algorithms in the DP setting, hence we use RDT as a representative for comparison with DP decision trees. The accuracy results for the three algorithms are presented in Tab. 5.3. From

	NoisyReg	DP-kNN	RDT
Adults	**79.37% (±0.31)**	69.10% (±0.46)	76.16% (±0.13)
Bank	86.31% (±1.1)	79.55% (±1.47)	**88.21% (±1.31)**
Banknotes	**97.71% (±0.35)**	80.00% (±3.5)	70.59% (±7.89)
Transfusion	**79.86% (±0.4)**	70.80% (±3.61)	78.05% (±3.98)
Claves	**79.07% (±1.2)**	76.78% (±0.74)	73.02% (±0.82)
Credit card	72.00% (±0.47)	69.92% (±0.6)	**78.28% (±0.14)**
Mushroom	**97.88% (±0.63)**	72.73% (±1.19)	93.89% (±1.71)
Occupancy	**98.74% (±0.2)**	79.27% (±1.45)	82.87% (±5.03)

Table 5.3: Accuracy comparison of DP non-parametric algorithms, value in bracket represents the standard deviation value over 20 runs. Privacy parameter ϵ_p is fixed to 2 over all runs.

Tab. 5.3, we observe that NoisyReg achieved the highest accuracy over majority of the data sets and second place for the rest. Despite DP-kNN has achieved better accuracies on some experiments (Banknotes and Claves), it has not achieved best performance in any of the date sets. In particular, NoisyReg has better performance over DP-kNN in all data sets tested. However, despite NoisyReg can perform very well on many data sets and make a significant improvement in accuracy (e.g. in Banknotes), NoisyReg can under-perform in some cases (e.g. Credit card). The structure of the DP histogram is quite data-dependent (boxy shape with even widths). Hence different data structure can significantly affects the performance of the DP histogram.

Remark 13. *Since the NoisyReg algorithm works by adding adaptive noise into each of the cells depending on the number of samples, hence cells with a small number of samples will be added a large value of noise (in average). This makes the NoisyReg algorithm more suitable for training samples that are more evenly distributed over the space, because that reduces the chance of a cell receives only a few samples and balances out the noise.*

Summary

We have presented a kNN algorithm in the differentially private setting and analysed its generalisation errors in comparison to the non-private case and the Bayes optimal classifier. We found that the DP-kNN suffers from over-splitting the privacy budget ϵ_p when the number of testing samples becomes large. The generalisation results for DP-kNN can be improved with compressed learning in a similar way as in Section 4.1.1. However, it does not solve the problem of over-splitting the privacy budget. We have also presented the NoisyReg algorithm which achieves differential privacy for the histogram classifier, unlike the DP-kNN, does not split up the privacy budget as the number of testing samples increases. We have proved that the NoisyReg approach achieves the optimal convergence rate with respect to n and d (same rate as in the non-private case). We have also proved that a better convergence rate can be achieved if the Bayes error is zero for the histogram classifier, as we have observed similarly in the non-private case. Furthermore, we have shown that by learning in the compressed setting, we reduced the dimensionality dependence in the generalisation bound of the private histogram along with the dimension of the added noise. Finally, we have analysed the presented algorithms experimentally with a variety of data sets. Empirically we found that NoisyReg achieves good classification performance over DP-kNN and private RDT.

Chapter Six

Generalization of Projected Gradient Descent Methods

Stochastic gradient descent (SGD) is a popular optimisation algorithm that has gained much attention for decades [21]. For instance, it is well-known in convex optimisation that SGD can optimise convex functions over a convex domain with guaranteed convergence rates [144]. Furthermore, it is known that the error bound of SGD can be dimension-independent which makes it favorable for high-dimensional optimisation [66, 32, 36, 86, 84, 91].

More recently, in the context of distributed optimisation, there is an increasing interest in sketching methods in SGD, where a sketch of the gradient is transmitted to the server instead of the original full gradient in order to reduce communication costs [2, 1, 133, 121]. A particularly useful and innovative recent approach proposed by [76] is compressed SGD (CompSGD), which employs random projection for this purpose. The authors showed that, contrary to previous approaches (where sketching comes at the expense of an increase of the variance of the gradient), compression by random projection is able to exploit the geometry of the parameter space imposed by regularisation to make the approach lossless. In other words, compressed SGD can achieve the same convergence rate as classical SGD up to logarithmic factors (in expectation). Furthermore, the authors also demonstrated empirically that the run time of CompSGD is almost the same as that of classical SGD.

Hence, one can use the lower dimensional (compressed) gradient for almost 'zero cost' in its performance. Furthermore, compressed SGD lends itself to applications in privacy-related optimisation, as we only need to add noise in the low dimensional space of compressed gradients [76], which then reduces the dimensionality of the injected noise.

The existing theoretical analysis of CompSGD only provides its optimisation convergence rate [76] in specific non-private settings and has not considered differential privacy. Moreover, an analysis of optimisation can only guarantee the performance of models on training examples. However, the object of primary interest in machine learning is the generalisation error, or population risk, of the learnt models. It is therefore imperative to find out to what extent the use of compressed gradients would affect the generalisation guarantees of learning algorithms. A positive finding on this question will provide a solid theoretical footing for applications in large-scale distributed and federated learning with low communication cost, such as the systems described in [98], and applications in differentially private learning.

In this chapter, we set out to study these questions for the first time, starting with convex problems. These have a fundamental role in both learning and optimisation [10], and apply naturally to a variety of high dimensional sparsity-based models [72, 90, 127, 98] and structure learning [6, 61]. Insights from the study of convex problems are indispensable to advance our understanding further. We will consider differentially private settings in both optimisation and learning problems. To tackle the latter, following a line of research on SGD [66, 86], we will leverage the fundamental concept of algorithmic stability to study the generalisation performance of CompSGD. This will shed light on the effects of gradient update compression in algorithm-dependent bounds while the analysis itself is independent on the particular form of predictors.

We provide a rigorous analysis of CompSGD [76] in terms of optimisation convergence rates, as well as generalisation convergence rates. These quantify the effect of random compression of gradient updates. As a key ingredient, we employ a stability-based analysis, providing the first stability and generalisation guarantees for SGD with low-dimensional gradients. We consider both smooth and non-smooth problems, with and without privacy constraints. Furthermore, we also give the first optimisation and stability convergence analysis for two variants of CompSGD in both private and non-private settings. Results for the private setting are presented in the next chapter but we summarize both setting here for a complete overview of CompSGD. Our main contributions and findings are summarized as follows:

1. We prove the first uniform stability bounds of CompSGD for both smooth and non-smooth problems. Based on this, we show that CompSGD can achieve the same population risk bounds as regular SGD up to logarithmic factors. Our bound of the order $\widetilde{O}(1/\sqrt{n})$ is optimal up to a logarithmic factor, where n is the sample size. Here we use $\widetilde{O}$ to hide logarithmic factors.

2. We prove the first optimisation bounds of batch and mini-batch compressed gradient descent and show the convergence can be quicker with a larger step size in the smooth case.

3. We further extend our stability analysis to batch and mini-batch variants of CompSGD and show that these variants can achieve the exact same population risk bounds as CompSGD with fewer iterations.

4. We prove the first optimisation bound for CompSGD in the differentially private setting and show that the dimensionality of the injected noise can be significantly reduced from $\mathcal{O}(d)$ to $\mathcal{O}(\log(d))$, where d is the dimension.

5. Finally, by our stability analysis in the differentially private setting, we also prove the first generalisation bound of DP-CompSGD and show the same generalisation convergence also holds while the dimensionality is reduced.

6.1 Preliminaries

We describe some slight changes of notation in this section as well as newly introduced notation. Recall the following general setting of supervised learning. Let $\ell : \mathbb{R}^d \times \mathcal{Z} \to \mathbb{R}$ be a loss function that quantifies the quality of outputs for a hypothesis represented by the parameter vector $\mathbf{w} \in \mathcal{C}$, where $\mathcal{C}$ is the parameter set, assumed to be a convex set (analogue to $\mathcal{H}$ in Chapter 4 and Chapter 5). Given some loss function ℓ, we aim to find a $\mathbf{w} \in \mathcal{C}$ that minimises the risk (expected loss) defined as $L_{\mathcal{D}}(\mathbf{w}) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(\mathbf{w}, z)]$. We also define the empirical analogue as

$$L_S(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{w}, z_i). \tag{6.1}$$

Given a finite sample set $S \subset \mathcal{Z}$, we run an iterative gradient-based optimisation algorithm to minimise (6.1) over the parameter set $\mathcal{C}$, such as the Stochastic Gradient Descent (SGD), where at each step we update our weight vector $\mathbf{w} \in \mathcal{C}$ using a sample-based estimate of the gradient of ℓ.

As in the work of [76], we shall make use of a geometric measure of complexity of the parameter set $\mathcal{C}$, namely the *Gaussian width.*

Recall that we use $\|\cdot\|$ to denote the Euclidean norm $\|\cdot\|_2$. We denote by A the optimisation algorithm and note its randomised nature. Expectations $\mathbb{E}[\cdot]$ are taken with respect to the random sampling of S and the randomness in the algorithm A, unless otherwise specified. In the context of stability analysis, we say two sets S, S' with size n are *neighbouring sets* if they differ by at most one sample. For a set $\mathcal{C}$, we define its diameter as $\|\mathcal{C}\| = \sup_{\mathbf{w},\mathbf{w}'\in\mathcal{C}} \|\mathbf{w} - \mathbf{w}'\|$. Given a linear transform $\Phi \in \mathbb{R}^{m\times d}$, we define the projected set $\Phi\mathcal{C} := \{\Phi\mathbf{w} : \mathbf{w} \in \mathcal{C}\}$. Furthermore, we define the orthogonal projection operator Π in the usual way as follows: for a set $\mathcal{C}$ and vector $\mathbf{w}$, the projection of $\mathbf{w}$ onto $\mathcal{C}$ is denoted by $\Pi_{\mathcal{C}}\mathbf{w}$; this is the vector $\mathbf{w}' \in \mathcal{C}$ such that $\mathbf{w}'$ has a minimal distance to $\mathbf{w}$. We use the notation $B \asymp \widetilde{B}$ if there exist universal constants $c_1, c_2 > 0$ such that $c_1\widetilde{B} < B \leq c_2\widetilde{B}$.

6.2 Optimization with Compressed Gradient Updates

In this section, we briefly review the stochastic gradient descent with compressed gradient updates proposed by [76]. As outlined in Algorithm 5, we first compress both the weight vector $\mathbf{w}$ and the gradient to a lower-dimensional space using a random projection matrix Φ, where the projection dimension depends on the Gaussian width of the convex set $\mathcal{C}$. Then we take a step to the opposite direction of the gradient, followed by an orthogonal projection onto $\Phi\mathcal{C}$ using the projection operator Π (line 6 in Alg. 5). After each update in the lower dimension, we will then 'lift' the low dimensional parameter vector back into the higher dimensional parameter space. In our analysis, we will use the average output model throughout the paper, which is defined as $\bar{\mathbf{w}}_T = (\sum_{t=1}^T \eta_t \mathbf{w}_t)/\sum_{t=1}^T \eta_t$ where η_t and $\mathbf{w}_t$ is the parameter and output of the algorithm at each iteration as defined in Alg. 5.

Algorithm 5 CompSGD [76]

```
1: Inputs: Sample set $S$ of $n$ points in $\mathbb{R}^d$, convex set $\mathcal{C}$, learning rate parameters $\{\eta_t\}$,
   and projection dimension parameters $\{\beta_t\}$.
2: initialize $\mathbf{w}$ as any point in $\mathcal{C}$.
3: for $t = 1$ to $T$ do
4:     $m_t = \mathcal{O}(\min\{d, \omega(\mathcal{C})^2/\beta_t^2\})$
5:     Let $\Phi_t \in \mathbb{R}^{m_t \times d}$ be an i.i.d. random projection matrix
6:     set $\nabla\ell(\mathbf{w}_t, z_{i_t})$ as the gradient where $i_t$ is a uniformly chosen index from $[n]$
7:     set $\theta_t = \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}_t, z_{i_t}))$
8:     pick $\mathbf{w}_{t+1}$ as any element from the set $\{\mathbf{w} \in \mathcal{C} : \Phi_t\mathbf{w} = \theta_t\}$.
9: end for
```

A key result of [76] showed that the convergence of SGD with compressed gradients is the same as that of regular SGD up to logarithmic factors. In further experimental analysis, they also demonstrated the run time of using low-dimensional gradients is almost as quick as regular SGD. Hence we can perform SGD with low-dimensional gradients at 'almost' zero cost.

Their result only addresses the optimisation performance. From the perspective of machine learning, it is intriguing to know what can we say about generalisation. One of our main results will establish that the stability and generalization convergence of compressed SGD is also the same as that of regular SGD up to logarithmic factors. A major benefit of using compressed gradients is that we can reduce the dimension of the noise in the differentially private setting, as we only need to add noise in the lower dimensional space. We will also show that this dimensionality reduction effect is captured in its optimization convergence analysis.

Throughout this chapter, we will assume that the loss function ℓ is convex and Lipschitz in its first argument.

We note that, despite the popular use of SGD on non-convex problems, the theoretical analysis in the non-convex case is still very limited (e.g. privacy-related applications [131]). There are currently no generalization analysis of SGD with compressed gradients at all, to the best of our knowledge. Therefore our aim is to provide the first insights in compressed gradient descent methods that yield low generalization error. This will eventually lead to new insights in privacy applications and complex non-convex problems.

	Optimization error			
	Smooth case		Non-smooth case	
	Convergence rate	Step size	Convergence rate	Step size
Classic SGD	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ ([117, Thm. 2])	$\eta_t = \frac{\eta}{\sqrt{t}}$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ ([117, Thm.2])	$\eta_t = \frac{\eta}{\sqrt{t}}$
CompSGD	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ ([76])	$\eta_t = \frac{\eta}{\sqrt{t}}$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ ([76])	$\eta_t = \frac{\eta}{\sqrt{t}}$
CompGD	$\mathcal{O}\left(\frac{\log T}{T}\right)$ (Thm. 6.3)	$\eta_t = \eta$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ (Thm. 6.5)	$\eta_t = \frac{\eta}{\sqrt{t}}$
CompMinibatch	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ (Thm. 6.7)	$\eta_t = \frac{\eta}{\sqrt{t}}$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right)$ (Thm. 6.7)	$\eta_t = \frac{\eta}{\sqrt{t}}$
DP-CompGD	$\mathcal{O}\left(\frac{\log T\sqrt{m_T}}{T} + \frac{T\sqrt{m_T}\log(1/\delta)}{n^2\epsilon^2}\right)$ (Thm. 7.2)	$\eta_t = \frac{1}{\sqrt{m_T}}$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}} + \frac{\sqrt{m_T\log(1/\delta)}}{n\epsilon}\right)$ (Thm. 7.1)	$\eta_t = \frac{1}{\sqrt{t(1+m_T\sigma^2)}}$
DP-CompMinibatch	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}} + \frac{\log T\sqrt{m_T\log(T/n\delta)}}{n\epsilon}\right)$ (Thm. 7.4)	$\eta_t = \frac{1}{\sqrt{t(1+m_T\sigma^2)}}$	$\mathcal{O}\left(\frac{\log T}{\sqrt{T}} + \frac{\log T\sqrt{m_T\log(T/n\delta)}}{n\epsilon}\right)$ (Thm. 7.4)	$\eta_t = \frac{1}{\sqrt{t(1+m_T\sigma^2)}}$

Table 6.1: A summary of optimization error bounds for algorithms with randomly compressed gradients. Assumptions made for simplicity: The loss function is 1-Lipschitz, and the diameter of the parameter set $\mathcal{C}$ is 1. Refer to the indicated theorems for results with general parameters. For the differentially private algorithms (last two rows), m_T denotes the maximum projection dimension used and σ^2 is the variance of the noise added for privacy.

	Generalization error			
	Smooth case		Non-smooth case	
	Convergence rate	Parameters	Convergence rate	Parameters
Classic SGD	$\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ ([66, Prop.5.4])	$\eta_t = \frac{\eta}{\sqrt{T}}, T \asymp n$	$\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ ([86, Thm.7])	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$
CompSGD	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.1)	$\eta_t = \frac{\eta}{\sqrt{t}}, T \asymp n$	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.2)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$
CompGD	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.4)	$\eta_t = \eta, T \asymp \sqrt{n}$	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.6)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$
CompMinibatch	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.8)	$\eta_t = \frac{\eta}{\sqrt{t}}, T \asymp n$	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}}\right)$ (Thm. 6.9)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$
DP-CompGD	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}} + \frac{\log(1/\delta)}{n^{3/2}\epsilon^2}\right)$ (Thm. 7.3)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}} + \frac{\log(1/\delta)}{n^{3/2}\epsilon^2}\right)$ (Thm. 7.3)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$
DP-CompMinibatch	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}} + \frac{\log(T/n\delta)}{n^{3/2}\epsilon^2}\right)$ (Thm. 7.5)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$	$\mathcal{O}\left(\frac{\log n}{\sqrt{n}} + \frac{\log(T/n\delta)}{n^{3/2}\epsilon^2}\right)$ (Thm. 7.5)	$\eta_t = \frac{\eta}{T^{3/4}}, T \asymp n^2$

Table 6.2: A summary of the generalization error bounds for algorithms with randomly compressed gradients. Assumptions made for simplicity: The loss function is 1-Lipschitz and the diameter of the parameter set $\mathcal{C}$ is 1. Refer to the indicated theorems for results with general parameters. For the differentially private (DP) algorithms (last two rows), m_T denotes the maximum projection dimension used and σ^2 is the variance of the added noise.

6.3 Generalization of Comp-SGD

6.3.1 Generalization bound under smoothness assumption

In this section, we assume that the loss function ℓ is convex, L-Lipschitz, and also μ-smooth. These assumptions are common and critical for convex optimization problems as they lead to bounds on the divergence or expansiveness of the gradient updates when the algorithm is run on neighbouring sample sets [66]. Examples of common loss functions that satisfy these assumptions include logistic loss, Huber loss and exponential loss (assuming bounded input samples). Later we also provide analysis without the smoothness assumption, which applies to e.g. the hinge loss.

However, the bottleneck here is to account for the effects of random compression that operate at each iteration in Alg 5. This is a form of sketching, which creates a perturbation that was not present in classic SGD. It is not at all obvious as to whether this sketched parameter update rule is sufficiently well behaved, since at each iteration the updated parameter vector $\mathbf{w}_{t+1}$ depends on the random matrix of the previous iteration, Φ_t, and these perturbations accumulate from iteration to iteration. Fortunately, it turns out that the Gaussian width defined in (2.20) allows us to estimate an appropriate projection dimension based on the structural complexity of the parameter set, such that the parameters learned by CompSGD from two neighbouring sets still do not diverge too much. This allows us to carry out a useful stability analysis similar to that of classic SGD, while incurring just an extra log factor.

Theorem 6.1 (Stability and generalization of CompSGD under smoothness)**.** *Assume that the loss function ℓ is convex, μ-smooth and L-Lipschitz on $\mathbf{w} \in \mathcal{C}$, for every $z \in \mathcal{Z}$. Suppose that we run the compressed SGD with step sizes $\eta_t = \frac{\eta}{\sqrt{t}} \leq 2/\mu$ for $T \asymp n$ iterations. Let $\beta_t = \frac{1}{t+1}$, $\eta_t = \frac{\eta}{\sqrt{t}}$ for some absolute constant η.*

1. *Then, the CompSGD algorithm (both $\mathbf{w}_T$ and $\bar{\mathbf{w}}_T$) is ϵ_{stab}-uniformly stable with $\epsilon_{stab} = \mathcal{O}\big(L^2 \log(n)/\sqrt{n}\big)$.*

2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ of CompSGD satisfies the following generalization bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}(L^2 \log(n)/\sqrt{n}). \tag{6.2}$$

Proof. See Section 6.5.1. □

Remark 14. *For the vanilla SGD, excess risk bounds of the order $O(1/\sqrt{n})$ were established for SGD with $\eta_t \asymp 1/\sqrt{n}$ and $T \asymp n$ [66]. Theorem 6.1 shows that CompSGD is able to achieve the same generalization bounds (up to a log factor) by updating with compressed stochastic gradient.*

6.3.2 Generalization bound without smoothness

Smoothness assumption is commonly used in the analysis for regular SGD since it simplifies the analysis for both optimization and generalization. However, in practice we often encounter learning problems with non-smooth loss functions (e.g. the hinge loss). Recently, [86, 134] showed for SGD without the smoothness assumption (relaxed Höldercontinuous assumption) enjoys stability and generalization bounds (up to constant factors) similar to that with the smooth assumption by choosing an appropriate choice of parameters. We will adapt parts of their technique here to prove the generalization convergence for CompSGD in the non-smooth case. We show that we can obtain the same convergence as in the smooth case for CompSGD up to log factors by choosing suitable projection and learning parameters.

Theorem 6.2 (Stability and generalization of CompSGD without smoothness)**.** *Assume that the loss function ℓ is convex and L-Lipschitz over the convex set $\mathcal{C}$. Suppose that we run the compressed SGD with step sizes $\eta_t = \frac{\eta}{T^{3/4}}$ for some absolute constant η for T steps. Furthermore we let $\beta_t = \frac{1}{t+1}$ and $T \asymp n^2$.*

1. *The compressed SGD is ϵ_{stab}-uniformly stable with $\epsilon_{stab} = \mathcal{O}\big(L^2\sqrt{\log(n)}/\sqrt{n}\big)$.*
2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ of CompSGD satisfies the following generalization bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}}\right). \tag{6.3}$$

Proof. See Section 6.5.2. □

From Theorem 6.2 we have chosen a relatively smaller step size parameter $\eta_t = \eta/T^{3/4}$ comparing to $\eta_t = \eta/\sqrt{t}$ in the smooth case. The choice of η_t here is needed for our result to have the same convergence as in the smooth case. An intuitive explanation for the smaller step size parameter is that the problem is harder without the smoothness. Hence, we need to take more careful steps towards the optima.

Remark 15. *The choice of the step size parameter in the non-smooth case matches with the choice for classical SGD in the same setting [86]. Hence Theorem 6.2 shows that CompSGD achieves the same generalization bounds in the non-smooth setting with the same parameters (up to logarithmic factors).*

6.4 Variants of CompSGD

In this section, we present the convergence guarantee of the variants of the compressed SGD - batch gradient descent that uses the full gradient (Section 6.4.1) and mini-batch gradient descent that uses the gradient of the mini-batch (Section 6.4.2). These variants of SGD are also widely used in practice, especially in cases where we can compute the full gradient easily to make more informative updates in each iteration. In the differentially private setting it is also desirable to use batch gradient instead of stochastic gradient due to the random noise injected and to limit the number of iterations required. In this section, we show that the risk bounds of these variants are the same as the rates for compressed SGD. Hence, we can choose an appropriate method suited to our needs without affecting its generalization.

6.4.1 Compressed gradient descent

The first variant we will analyze is the classical batch gradient descent. Batch gradient descent utilizes the most information from the sample set S at each iteration to make accurate updates. Hence we can usually converge close to the optima using much less iterations compared to SGD which can be beneficial in many cases (e.g. private optimizations). The detail is outlined in Algorithm 6.

Algorithm 6 Compressed Gradient Descent (CompGD)

```
1: Inputs: Sample set $S$ of $n$ points, convex set $\mathcal{C}$, learning rate parameters $\{\eta_t\}$, and
   projection dimension parameters $\{\beta_t\}$.
2: initialize $\mathbf{w}$ as any point in $\mathcal{C}$.
3: for $t = 1$ to $T$ do
4:     Set $m_t = \mathcal{O}(\min\{d, \omega(\mathcal{C})^2/\beta_t^2\})$
5:     Let $\Phi_t \in \mathbb{R}^{m_t \times d}$ be an i.i.d. random projection matrix
6:     compute $\hat{g}_t = \frac{1}{n}\sum_{z\in S} \nabla \ell(\mathbf{w}_t, z)$ as the gradient
7:     set $\theta_t = \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\hat{g}_t)$
8:     pick $\mathbf{w}_{t+1}$ to be any element from the set $\{\mathbf{w} \in \mathcal{C} : \Phi_t\mathbf{w} = \theta_t\}$.
9: end for
10: Output: $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_T$
```

Batch gradient descent is a special case where we can use a constant step size η to obtain a quicker convergence rate in its optimization bound. We show that this is also the case for compressed batch gradient descent.

Theorem 6.3 (Optimization of CompGD, smooth case). *Let ℓ be convex function over a convex set $\mathcal{C}$, and satisfy L-Lipschitz condition and μ-smooth condition. Then with $\eta_t = \eta \leq 1/(2\mu)$ for some absolute constant η and $\beta_t = \frac{1}{t+1}$, the compressed gradient descent satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 \log(T)}{T}\right). \tag{6.4}$$

Proof. See Section 6.5.3. □

We now study the risk bounds of CompGD. Note that here we have $O(1/T)$ for its optimization bound which is better than SGD. We also note that we have chosen a constant step size η for batch gradient because each gradient update is accurate enough to take larger steps. Hence, we only required $T \asymp \sqrt{n}$ to achieve the same generalization convergence compared to SGD.

Theorem 6.4 (Stability and generalization of CompGD, smooth case). *Assume the loss function ℓ is convex, μ-smooth and L-Lipschitz over the convex set $\mathcal{C}$. Suppose we run the CompGD with $\eta_t = \eta \leq 1/(2\mu)$ for $T \asymp \sqrt{n}$ steps where η is an absolute constant and $\beta_t = 1/(t+1)$.*

1. *Then CompGD satisfies uniform stability with $\epsilon_{stab} = \mathcal{O}\left(L^2/\sqrt{n}\right)$.*

2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ of compGD satisfies the following excess risk bound*

$$\mathbb{E}[L_\mathcal{D}(\bar{\mathbf{w}}_T)] = L_\mathcal{D}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}}\right). \tag{6.5}$$

Proof. See Section 6.5.3. □

While we can obtain a faster convergence rate for batch gradient descent in optimization, the stability guarantee of CompGD is same as CompSGD. This is also the case for classic SGD, since the samples we use for $\mathbf{w}_{t+1}, \mathbf{w}'_{t+1}$ will differ in one point every iteration with probability 1. Hence we do not obtain an improvement in the expected excess generalization risk. However, the smoothness case will allow us to choose a larger learning rate compared with the non-smooth case.

Theorem 6.5 (Optimization of CompGD, non-smooth case). *Let ℓ be a convex function over a convex set $\mathcal{C}$ and satisfy L-Lipschitz condition. Then with $\eta_t = \eta/\sqrt{t}$ for some absolute constant η and $\beta_t = \frac{1}{t+1}$, the compressed gradient descent satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(T)}{\sqrt{T}}\right). \tag{6.6}$$

Proof. See Section 6.5.3. □

Theorem 6.6 (Stability and generalization of CompGD, non-smooth case). *Assume the loss function ℓ is convex and L-Lipschitz over the convex set $\mathcal{C}$. Suppose we run the CompGD with $\eta_t = \eta/T^{3/4}$ for $T \asymp n^2$ steps where η is an absolute constant and $\beta_t = 1/(t+1)$.*

1. *Then CompGD satisfies uniform stability with $\epsilon_{stab} = \mathcal{O}\left(L^2 \log(n)/\sqrt{n}\right)$.*
2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ of compGD satisfies the following excess risk bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}}\right). \tag{6.7}$$

Proof. See Section 6.5.3. □

6.4.2 Compressed SGD with Mini-batch

In this section, we study the stability and generalization of CompSGD with a mini-batch strategy. Mini-batch SGD is considered as a semi-stochastic version of gradient descent and is widely used in various applications [82, 145]. Different sampling techniques may be used to sample a mini-batch depending on the application and preference. Here we will use the following sampling method for the CompSGD with mini-batch: For each iteration we sample a mini-batch B_t of size b from the sample set S without replacements. Then we will sample a fresh batch from S at the next iteration so that the sampling set S is consistent.

Algorithm 7 Compressed SGD with Mini-batch

```
1: Inputs: Sample set $S$ of $n$ points in $\mathbb{R}^d$, batch size $b$, convex set $\mathcal{C}$, learning rate
   parameters $\{\eta_t\}$, and projection dimension parameters $\{\beta_t\}$.
2: initialize $\mathbf{w}$ as any point in $\mathcal{C}$.
3: **for** $t = 1$ to $T$ **do**
4:     Set $m_t = \mathcal{O}(\min\{d, \omega(\mathcal{C})^2/\beta_t^2\})$
5:     Let $\Phi_t \in \mathbb{R}^{m_t \times d}$ be an i.i.d. random projection matrix
6:     Sample a mini-batch $B_t$ of size $b$ uniformly from $S$
7:     compute $\hat{g}_t = \frac{1}{b}\sum_{z \in B_t} \nabla \ell(\mathbf{w}_t, z)$ as the gradient of the mini-batch
8:     set $\theta_t = \Pi_{\Phi_t \mathcal{C}}(\Phi_t \mathbf{w}_t - \eta_t \Phi_t \hat{g}_t)$
9:     pick $\mathbf{w}_{t+1}$ to be any element from the set $\{\mathbf{w} \in \mathcal{C} : \Phi_t \mathbf{w} = \theta_t\}$.
10: **end for**
11: Output: $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_T$
```

Theorem 6.7 (Optimization of Mini-batch SGD). *Let ℓ be a convex function over a convex set $\mathcal{C}$, and satisfy L-Lipschitz condition. Then with $\eta_t = \frac{\eta}{\sqrt{t}}$ and $\beta_t = \frac{1}{t+1}$, the compressed SGD with minibatch satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(T)}{\sqrt{T}}\right). \tag{6.8}$$

Proof. See Section 6.5.4. □

Theorem 6.8 (Stability and generalization of Mini-batch SGD, smooth case). *Assume that the loss function ℓ is μ-smooth, convex and L-Lipschitz for every z. Suppose that we run the compressed SGD with mini-batch of size b and step sizes $\eta_t \le 2/\mu$ for T iterations.*

1. *Then with $\eta_t = \eta/\sqrt{t}$ for some absolute constant η, $\beta_t = \frac{1}{t+1}$ and $T \asymp n$, the compressed SGD with mini-batch is ϵ_{stab}-uniformly stable with $\epsilon_{stab} = \mathcal{O}\left(L^2 \log(n)/\sqrt{n}\right)$.*

2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ satisfies the following excess risk bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}}\right). \tag{6.9}$$

Proof. See Section 6.5.4. □

We note that the optimization and stability of the mini-batch achieves the same convergence rate as SGD, which is optimal up to logarithmic factors. The main advantage of mini-batch is to perform stochastic gradient descent while preventing over-randomized convergence to the optima. In the best case we will obtain the same convergence as for batch gradient descent in Section 6.4.1.

Theorem 6.9 (Stability and generalization of Mini-batch SGD, non-smooth case)**.** *Assume that the loss function ℓ is convex and L-Lipschitz for every z. Suppose that we run the compressed SGD with mini-batch of size b and step sizes $\eta_t = \eta/T^{3/4}$ for some absolute constant η, $\beta_t = \frac{1}{t+1}$ and $T \asymp n^2$.*

1. *The compressed SGD with mini-batch is ϵ_{stab}-uniformly stable with*

$$\epsilon_{stab} = \mathcal{O}\left(L^2 \log(n)/\sqrt{n}\right).$$

2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ satisfies the following excess risk bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}}\right). \tag{6.10}$$

Proof. See Section 6.5.4. □

6.5 Proofs

In this section, we present the proofs of the key theorems in Section 6.3 and Section 6.4. Proofs for intermediate results and proofs for Section 6.4.1 and Section 6.4.2 are deferred to the Appendix. For simplicity of the proof, we first denote our gradient update at iteration t as follows:

A gradient update in compressed SGD (lines 4-8 in Alg. 5) is a map $G : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ that takes a parameter vector $\mathbf{w}_t$ and a training point z as inputs, and it outputs the updated parameter vector $G(\mathbf{w}_t, z)$, defined as the following

$$G(\mathbf{w}_t, z) = \underset{\mathbf{w}\in\mathcal{C}}{\arg\min}\{\|\mathbf{w}\|_1 : \Phi_t\mathbf{w} = \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}_t, z))\}, \tag{6.11}$$

where Φ_t is a RP matrix, and η_t is the step size parameter. We drop the dependence of z when it is clear from the context and just write $G(\mathbf{w})$ for simplicity. We remark that our analysis does not require $\mathbf{w}$ to have the minimum ℓ_1-norm property; this is included only to break ties so that the map G is well defined. Indeed, one can pick the updated element as described in Alg 5.

6.5.1 Proof of stability & generalization under smoothness

To prove the stability and generalization of CompSGD, we first establish some important properties of CompSGD. A key idea in stability analysis is to control the extent to which a sequence of updates starting from neighbouring sample sets diverge, in each iteration - in our case, one update corresponds to one run of lines 4-8 in Alg. 5. The algorithm is more stable if the divergence is smaller. The following result shows how the divergence between two gradient updates in CompSGD is controlled by the projection parameter β_t.

Lemma 6.10 (Distortion from RP). *Let $\mathbf{w}_{t+1}, \mathbf{w}'_{t+1} \in \mathcal{C}$ be the parameter vectors at iteration $t+1$ of Alg. 5 when run on two neighbouring sample sets S and S'. For any choices of training points z_{i_t} and z'_{i_t}, we have*

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \le \|(\mathbf{w}_t - \mathbf{w}'_t) - (\eta_t\nabla\ell(\mathbf{w}_t, z_{i_t}) - \eta_t\nabla\ell(\mathbf{w}'_t, z'_{i_t}))\|^2.$$

Since Lemma 6.10 implies that the divergence of gradient update with projected gradient is upper bounded by the divergence of regular gradient update (up to the $1-\beta_t$ factor), we can make use of some property about classical SGD. We will achieve this using the concepts of expansivity and boundedness introduced in [66].

Definition 14 (Well behaved gradient update). *We say the gradient update $G(\mathbf{w})$ is α-*

expansive if, for all $\mathbf{v}, \mathbf{w} \in \mathcal{C}$ *we have* $\|G(\mathbf{v}) - G(\mathbf{w})\| \leq \alpha \|\mathbf{v} - \mathbf{w}\|$. *We say the gradient update* $G(\mathbf{w})$ *is* γ*-bounded if* $\sup_{\mathbf{w} \in \mathcal{C}} \|\mathbf{w} - G(\mathbf{w})\| \leq \gamma$.

We now prove that, for the type of loss functions considered, the updating rule of Alg. 5 is well behaved despite the distortion created by random compression. First we show that, at any iteration t the CompSGD updating rule $\mathbf{w}_{t+1} = G(\mathbf{w}_t)$ has limited expansiveness whenever the same training point is chosen for gradient estimation (Lemma. 6.11). Secondly, the CompSGD update is bounded whenever different training points are chosen for gradient estimation (Lemma. 6.12).

Lemma 6.11 (Limited expansiveness)**.** *Assume that* ℓ *is convex and* μ*-smooth. Fix any* $t \in \mathbb{N}$*, and let* $\mathbf{w}_t, \mathbf{w}'_t \in \mathcal{C}$ *be the parameter vectors at the* t*-th iteration of Alg. 5 when run on two neighbouring sample sets* S *and* S'*. If* $z_{i_t} = z'_{i_t}$ *i.e. the same training point is chosen to estimate the gradient at the* t*-th iteration, then the updating rule of compressed SGD (Alg. 5) is* $\frac{1}{\sqrt{1-\beta_t}}$*-expansive for* $\eta_t \leq 2/\mu$ *– that is, we have* $\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| \leq \frac{1}{\sqrt{1-\beta_t}} \|\mathbf{w}_t - \mathbf{w}'_t\|$.

Lemma 6.12 (Boundedness)**.** *Assume that* ℓ *is* L*-Lipschitz. Fix any* $t \in \mathbb{N}$*, and let* $\mathbf{w}_t \mathcal{C}$ *be the parameter vectors at the* t*-th iteration of Alg. 5 when run on* S*. Then the updating rule of the compressed SGD (Alg. 5) is* $\frac{\eta_t L}{\sqrt{1-\beta_t}}$*-bounded – that is, we have* $\|\mathbf{w}_{t+1} - \mathbf{w}_t\| \leq \frac{\eta_t L}{\sqrt{1-\beta_t}}$.

With these core properties recorded, we now prove the stability guarantee of the CompSGD under the smoothness setting. The basic idea in the proof is that we note with probability $1 - 1/n$ the sample we select from S and S' will be identical, which allows us to use the expansiveness of CompSGD. We can then bound the low probability case with the γ-bounded property and put the two cases together to obtain our result.

Proof of Theorem 6.1. Let S and S' be two neighbouring sample sets of size n that differ in one single sample point. Denote $G_t := G(\cdot, z_{i_t})$ and $G'_t := G(\cdot, z'_{i_t})$, with $t \in [T], i_t \in [n]$, the gradient updates induced by running the compressed SGD on the neighbouring sample sets S and S', respectively. Let $\delta_T = \|\mathbf{w}_T - \mathbf{w}'_T\|$, and fix a sample point z. By the Lipschitz condition,

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}'_T, z)|] \leq L\mathbb{E}[\delta_T]. \tag{6.12}$$

Observe that, at iteration t, with probability $1 - 1/n$, the example z_{i_t} and z'_{i_t} selected from both S and S' is the same. In this case we have $G_t = G'_t$, and we use the limited expansiveness property of the update G_t from Lemma 6.11. With the remaining probability $1/n$, we have $z_{i_t} \neq z'_{i_t}$, in which case we use the boundedness property of both updates G_t and G'_t cf. Lemma 6.12. By the linearity of expectation, and the triangle inequality, this yields the following

$$\mathbb{E}[\delta_{t+1}] \leq \left(\frac{1 - 1/n}{\sqrt{1 - \beta_t}}\right) \mathbb{E}[\delta_t] + \frac{1}{n}\left(\mathbb{E}[\delta_t] + \frac{2\eta_t L}{\sqrt{1 - \beta_t}}\right). \tag{6.13}$$

Now it remains to solve this recursive sequence. We multiply both sides by $\prod_{j=1}^{t-1} \sqrt{1 - \beta_j}$,

$$\left[\prod_{j=1}^{t} \sqrt{1 - \beta_j}\right] \mathbb{E}[\delta_{t+1}] \leq \left[\prod_{j=1}^{t-1} \sqrt{1 - \beta_j}\right] \mathbb{E}[\delta_t] + \frac{2\eta_t L}{n} \prod_{j=1}^{t-1} \sqrt{1 - \beta_j} \tag{6.14}$$

and sum up the T iterates

$$\left[\prod_{j=1}^{T-1} \sqrt{1 - \beta_j}\right] \mathbb{E}[\delta_T] \leq \sum_{t=1}^{T-1} \frac{2\eta_t L}{n} \prod_{j=1}^{t-1} \sqrt{1 - \beta_j}. \tag{6.15}$$

Rearranging, we have:

$$\mathbb{E}[\delta_T] \leq \frac{2L}{n} \sum_{t=1}^{T-1} \eta_t \prod_{j=t}^{T-1} (1 - \beta_j)^{-1/2}. \tag{6.16}$$

In particular, with the choice $\beta_j = \frac{1}{j+1}$, we have $\prod_{j=t}^{T-1}(1 - \beta_j)^{-1/2} = \frac{\sqrt{T}}{\sqrt{t}}$. Furthermore, choosing $\eta_t = \frac{\eta}{\sqrt{t}}$ with some absolute constant η, we have

$$\mathbb{E}[\delta_T] = \frac{2\eta L\sqrt{T}}{n} \sum_{t=1}^{T-1} \frac{1}{t} = \mathcal{O}\left(\frac{L\sqrt{T}\log(T)}{n}\right), \tag{6.17}$$

where we exploited the fact that the growth rate of the partial sum of a harmonic series is just logarithmic. Finally, we take $T \asymp n$ and plug it back into (6.12) to conclude our stability bound. □

The generalization bound in Theorem 6.1 is a direct consequence of stability, combined with the optimization bound from [76, Thm.2.3] and Theorem 2.3, using the strategy discussed in Section 2.1.2.

6.5.2 Proof of stability & generalization without smoothness

We first prove the optimization bound of CompSGD in the non-smooth case.

Theorem 6.13 (Optimization of CompSGD with small step size). *Let ℓ be a convex and L-Lipschitz function over a convex set $\mathcal{C}$. Then with $\eta_t = \frac{\eta}{T^{3/4}}$ and $\beta_t = \frac{1}{t+1}$, CompSGD satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 \log(T) + L^2}{T^{1/4}}\right). \tag{6.18}$$

Proof for Theorem 6.13. We apply Thm A.6 with $\mathbf{w} = \mathbf{w}^*$ and have that for all $t \geq 1$:

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 &= \|\mathbf{w}_t - \mathbf{w}^*\|^2 + 2\eta_t\langle \nabla\ell(\mathbf{w}_t; z_{i_t}), \mathbf{w}^* - \mathbf{w}_t\rangle + \eta_t^2\|\nabla\ell(\mathbf{w}_t)\|^2 \\ &\leq \|\mathbf{w}_t - \mathbf{w}^*\|^2 + 2\eta_t(\ell(\mathbf{w}^*; z_{i_t}) - \ell(\mathbf{w}_t; z_{i_t})) + \eta_t^2 L^2. \end{aligned} \tag{6.19}$$

Rearranging we have:

$$2\eta_t(\ell(\mathbf{w}_t; z_{i_t}) - \ell(\mathbf{w}^*; z_{i_t})) \leq \|\mathbf{w}_t - \mathbf{w}^*\|^2 - \|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 + \beta_t\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 + \eta_t^2 L^2.$$

We take expectation on both sides, and sum over the T iterates,

$$2\sum_{t=1}^{T} \eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)] \leq \|\mathbf{w}_1 - \mathbf{w}^*\|^2 + \sum_{t=1}^{T}\beta_t\mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2] + L^2\sum_{t=1}^{T}\eta_t^2.$$

By choosing $\beta_t = 1/(t+1)$ and using the bound $\|\mathbf{w}_t - \mathbf{w}^*\|^2 \le \|\mathcal{C}\|^2$, we obtain

$$2\sum_{t=1}^{T} \eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\|\mathcal{C}\|^2 + \|\mathcal{C}\|^2 \sum_{t=1}^{T} \frac{1}{t+1} + L^2 \sum_{t=1}^{T} \eta_t^2\right)$$
$$= \mathcal{O}\left(\|\mathcal{C}\|^2 \log(T) + L^2 \sum_{t=1}^{T} \eta_t^2\right). \tag{6.20}$$

Finally, choosing $\eta_t = \frac{\eta}{T^{3/4}}$ we obtain our result as

$$\left(\sum_{t=1}^{T} \eta_t\right)^{-1} \sum_{t=1}^{T} \eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 \log(T)}{T^{1/4}} + \frac{L^2}{T^{3/4}}\right). \tag{6.21}$$

□

One can find from the proof of the optimization bound that we can use the parameters $\eta = \mathcal{O}(1/\sqrt{t})$ and $T \asymp n$ to yield convergence of order of $\mathcal{O}(1/\sqrt{n})$. However, we need to balance stability and optimization so that we obtain the best generalization convergence overall by choosing a smaller learning rate.

Proof of Theorem 6.2. In the non-smooth setting, we no longer have all the properties of CompSGD proved for the smooth case. However we still have the core result (Lemma 6.10) and note that the probability that we pick a different sample at an iteration is $1/n$ as in the smooth case. For the case where the selected sample is identical, we can make use of the convexity of ℓ and obtain the same convergence rate by carefully choosing the learning rate η_t.

Let S and S' be two neighbouring sample sets of size n that differ in one single sample. Let $G(\mathbf{w}_t) = \mathbf{w}_{t+1}$ denote the gradient update and let $G_1, \dots, G_T$ and $G'_1, \dots, G'_T$ be the updates induced by running the compressed SGD on S and S' for T iterates, respectively. Let $\delta_T = \|\mathbf{w}_T - \mathbf{w}'_T\|$, by the Lipschitz condition,

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}'_T, z)|] \le L\mathbb{E}[\delta_T]. \tag{6.22}$$

If at iteration t, the sample we selected is the same $G_t = G'_t$, then from Lemma 6.10 we have the following (short notations $\nabla\ell(\mathbf{w}_t, z_{i_t}) = \nabla\ell(\mathbf{w}_t)$ for simplicity)

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 &\le \|(\mathbf{w}_t - \mathbf{w}'_t) - \eta_t(\nabla\ell(\mathbf{w}_t, z_{i_t}) - \nabla\ell(\mathbf{w}'_t, z_{i_t}))\|^2 \\ &= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - 2\eta_t\langle\nabla\ell(\mathbf{w}_t) - \nabla\ell(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t\rangle + \eta_t^2\|\nabla\ell(\mathbf{w}_t) - \nabla\ell(\mathbf{w}'_t)\|^2.\end{aligned}$$

From the convexity of ℓ we have that $\langle\nabla\ell(\mathbf{w}_t) - \nabla\ell(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t\rangle \ge 0$ and from Lipschitzness of ℓ we also have $\|\nabla\ell(\mathbf{w}_t) - \nabla\ell(\mathbf{w}'_t)\| \le 2L$. Hence we obtain the following bound

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \le \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + 4L^2\eta_t^2. \tag{6.23}$$

For the case where $G_t \neq G'_t$, we use the inequality $(a+b)^2 \le (1+p)a^2 + (1+1/p)b^2, \forall p > 0$ to obtain

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 &\le \|(\mathbf{w}_t - \mathbf{w}'_t) - \eta_t(\nabla\ell(\mathbf{w}_t, z_{i_t}) - \nabla\ell(\mathbf{w}'_t, z_{i'_t}))\|^2 \\ &\le (1+p)\|\mathbf{w}_t - \mathbf{w}'_t\|^2 + 4(1+1/p)L^2\eta_t^2,\end{aligned}$$

where we have again used the Lipschitz condition of ℓ. Combining the two cases we have

$$\begin{aligned}(1-\beta_t)\mathbb{E}[\delta_{t+1}^2] &\le \left(1 - \frac{1}{n}\right)(\mathbb{E}[\delta_t^2] + 4L^2\eta_t^2) + \frac{1}{n}\left((1+p)\mathbb{E}[\delta_t^2] + 4(1+1/p)L^2\eta_t^2\right) \\ &\le (1+p/n)\mathbb{E}[\delta_t^2] + \left(4 + \frac{4(1+1/p)}{n}\right)L^2\eta_t^2. \end{aligned}\tag{6.24}$$

Denoting $\Delta_t = \left[\prod_{j=1}^{t-1}(1-\beta_j)\right](1+p/n)^{-t}\mathbb{E}[\delta_t^2]$ and multiplying both sides by $\left[\prod_{j=1}^{t-1}(1-\beta_j)\right](1+p/n)^{-(t+1)}$, we obtain

$$\Delta_{t+1} \le \Delta_t + \left[\prod_{j=1}^{t-1}(1-\beta_j)\right](1+p/n)^{-(t+1)}\left(4 + \frac{4(1+1/p)}{n}\right)L^2\eta_t^2. \tag{6.25}$$

Choosing $p = n/T$ and summing over T iterates we have

$$\Delta_T \le \sum_{t=1}^{T-1}\left[\prod_{j=1}^{t-1}(1-\beta_j)\right](1+1/T)^{-(t+1)}\left(4 + \frac{4(1+T/n)}{n}\right)L^2\eta_t^2. \tag{6.26}$$

By rearranging and choosing $\beta_j = \frac{1}{j+1}$, we get $\prod_{j=1}^{t-1}(1-\beta_j) = 1/t$. Hence we have

$$\mathbb{E}[\delta_T^2] \le \sum_{t=1}^{T-1} \frac{T}{t}\Big(1+\frac{1}{T}\Big)^{T-(t+1)} \left(4+\frac{4(1+T/n)}{n}\right) L^2\eta_t^2 = \mathcal{O}\left(L^2 T \sum_{t=1}^{T-1} \frac{(1+T/n^2)}{t}\eta_t^2\right), \tag{6.27}$$

where we noted that the factor $(1+1/T)^{T-(t+1)} = 1+\mathcal{O}((T-(t+1)/T) = \mathcal{O}(1)$. Using Eq. (6.22) and (6.27), letting $\eta_t = \frac{\eta}{T^{3/4}}$ for some absolute constant η and $T \asymp n^2$, we obtain our result by taking the square-root from both sides

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}'_T, z)|] = \mathcal{O}\left(\frac{L^2\sqrt{\log(n)}}{\sqrt{n}}\right). \tag{6.28}$$

The generalization result then follows as a direct consequence by combining it with Thm 6.13 and applying Thm. 2.3. □

Proof of Lemma 6.10. The proof of Lemma 6.10 follows similar derivation as for Theorem A.6 using Gordon's Theorem (Thm. 2.12), except that we are bounding the distortion between two gradient updates rather than a fixed point $\mathbf{w} \in \mathcal{C}$.

We start by replacing $\mathbf{w}$ with $\mathbf{w}'_{t+1}$ from equation (A.13), we have

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}'_{t+1}\|^2 &\le \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_{t+1}-\mathbf{w}'_{t+1})\|^2\right] \\ &= \mathbb{E}_{\Phi_t}\left[\left\|\prod_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}_t, z_{i_t}) - \prod_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}'_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}'_t, z'_{i_t}))\right\|^2\right] \\ &\le \mathbb{E}_{\Phi_t}\left[\left\|(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}_t, z_{i_t}) - (\Phi_t\mathbf{w}'_t - \eta_t\Phi_t\nabla\ell(\mathbf{w}'_t, z'_{i_t}))\right\|^2\right] \\ &= \|(\mathbf{w}_t-\mathbf{w}'_t) - (\eta_t\nabla\ell(\mathbf{w}_t, z_{i_t}) - \eta_t\nabla\ell(\mathbf{w}'_t, z'_{i_t}))\|^2,\end{aligned} \tag{6.29}$$

where we have used the fact that the projection map $\Pi_{\Phi\mathcal{C}}$ is contractive in the second step, i.e. distance between two points will not be larger after projection onto $\Phi\mathcal{C}$; and the final step follows since Φ_t is independent from all the remaining variables, $\mathbf{w}_t, \mathbf{w}'_t, \eta_t, z_{i_t}, z'_{i_t}$. □

Remark 16. *Similar to Thm. A.6, there are no requirements on the gradient used ($\nabla\ell$)*

being stochastic. Hence the same result will hold for batch (∇L_S) and mini-batch gradients (∇L_B).

Proof of Lemma 6.11. Since we assumed $z_{i_t} = z'_{i_t}$ in the runs of compSGD, we use the shorthand $\nabla \ell(\mathbf{w}_t)$ and $\nabla \ell(\mathbf{w}'_t)$ for $\nabla \ell(\mathbf{w}_t, z_{i_t})$ and $\nabla \ell(\mathbf{w}'_t, z_{i_t})$ as respectively. Denote $\mathbf{w}_t - \mathbf{w}'_t$ by Δ_t. From Lemma 6.10 we have

$$(1-\beta_t)\|\Delta_{t+1}\|^2 \le \|\Delta_t\|^2 - 2\eta_t\langle \nabla \ell(\mathbf{w}_t) - \nabla \ell(\mathbf{w}'_t), \Delta_t\rangle + \eta_t^2\|\nabla \ell(\mathbf{w}_t) - \nabla \ell(\mathbf{w}'_t)\|^2.$$

By Part 2 of Lemma 2.4 (co-coercivity), the second term on the r.h.s. is further bounded as

$$\langle \nabla \ell(\mathbf{w}_t) - \nabla \ell(\mathbf{w}'_t), \Delta_t\rangle \ge \frac{1}{\mu}\|\nabla \ell(\mathbf{w}_t) - \nabla \ell(\mathbf{w}'_t)\|^2. \tag{6.30}$$

Hence, we have

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \le \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - \left(\frac{2\eta_t}{\mu} - \eta_t^2\right)\|\nabla \ell(\mathbf{w}_t) - \nabla \ell(\mathbf{w}'_t)\|^2. \tag{6.31}$$

Setting $\eta_t \le 2/\mu$ eliminates the last term in (6.31), and the result follows. □

Proof of Lemma 6.12. By Theorem A.6, we have that $(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \le \|\mathbf{w}_t - \eta_t\nabla \ell(\mathbf{w}_t, z) - \mathbf{w}_t\|^2$. Hence, $\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \le \frac{\eta_t^2}{1-\beta_t}\|\nabla \ell(\mathbf{w}_t, z)\|^2 \le \frac{\eta_t^2 L^2}{1-\beta_t}$, where the last inequality is a consequence of the L-Lipschitz assumption on ℓ. □

6.5.3 Proofs for Compressed Gradient Descent

Proof of Theorem 6.3. By Thm. A.6 with $\mathbf{w} = \mathbf{w}^*_S$ we have

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}^*_S\|^2 &\le \|\mathbf{w}_t - \eta_t\nabla L_S(\mathbf{w}_t) - \mathbf{w}^*_S\|^2 \\ &= \|\mathbf{w}_t - \mathbf{w}^*_S\|^2 + 2\eta_t\langle \nabla L_S(\mathbf{w}_t), \mathbf{w}^*_S - \mathbf{w}_t\rangle + \eta_t^2\|\nabla L_S(\mathbf{w}_t)\|^2 \\ &\le \|\mathbf{w}_t - \mathbf{w}^*_S\|^2 + 2\eta_t(L_S(\mathbf{w}^*_S) - L_S(\mathbf{w}_t)) + \eta_t^2\|\nabla L_S(\mathbf{w}_t)\|^2, \quad (6.32)\end{aligned}$$

where the last line follows from the convexity of ℓ. From the smoothness we also have

$$\begin{aligned}
L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*) &\geq \langle \mathbf{w}_t - \mathbf{w}_S^*, \nabla L_S(\mathbf{w}_S^*) \rangle + \frac{1}{2\mu} \|\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_S^*)\|^2 \\
&\geq \frac{1}{2\mu} \|\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_S^*)\|^2,
\end{aligned}$$

where we have used $\langle \mathbf{w}_t - \mathbf{w}_S^*, \nabla L_S(\mathbf{w}_S^*) \rangle \geq 0$ by convexity and $\mathbf{w}_S^*$ is a minimizer of $L_S(\mathbf{w})$. Applying this property we have

$$\|\nabla L_S(\mathbf{w}_t)\|^2 = \|\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_S^*)\|^2 \leq 2\mu(L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)). \tag{6.33}$$

Substituting equation (6.33) into (6.32) we have

$$\begin{aligned}
(6.32) &\leq \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 + 2\eta_t(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) + \eta_t^2 \|\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_S^*)\|^2 \\
&\leq \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 + 2\eta_t(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) + 2\eta_t^2 \mu(L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)) \\
&\leq \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 + \frac{\eta_t}{2}(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) \text{ (assuming } \eta_t \leq 1/(2\mu)).
\end{aligned} \tag{6.34}$$

Rearranging, we have:

$$\frac{\eta_t}{2}(L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)) \leq \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 - \|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2 + \beta_t \|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2.$$

Taking expectation and summing over T iterates and choosing $\beta_t = 1/(t+1)$ we have:

$$\sum_{t=1}^{T} \frac{\eta_t}{2} \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)] \leq \|\mathbf{w}_1 - \mathbf{w}_S^*\|^2 + \sum_{t=1}^{T} \beta_t \mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2] = \mathcal{O}\left(\|\mathcal{C}\|^2 + \|\mathcal{C}\|^2 \log(T)\right),$$

where we have used $\mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|]^2 \leq \|\mathcal{C}\|^2$. Finally, for $\eta_t = \eta$ being an absolute constant we have

$$\left(\sum_{t=1}^{T} \eta_t\right)^{-1} \sum_{t=1}^{T} \eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 \log(T)}{T}\right). \tag{6.35}$$

The proof is completed. □

Proof of Theorem 6.4. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. W.l.o.g. we assume that the sample where S, S' differs is at index j: we denote z_j, z_j' for the sample in S and S' respectively. Fix a sample z, by the Lipschitz condition we get that

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}_T', z)|] \leq L\mathbb{E}[\delta_T], \tag{6.36}$$

where $\delta_T = \|\mathbf{w}_T - \mathbf{w}_T'\|$. At iteration t, we have from Lemma 6.10

$$\sqrt{1-\beta_t}\|\mathbf{w}_{t+1} - \mathbf{w}_{t+1}'\| \leq \|(\mathbf{w}_t - \mathbf{w}_t') - \eta_t(\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_t'))\|. \tag{6.37}$$

Note that since S, S' only differ on the j-th point, we have $S \cup \{z_j'\} = S' \cup \{z_j\}$.

$$\begin{aligned}(6.37) &= \left\|\mathbf{w}_t - \mathbf{w}_t' - \frac{\eta_t}{n}\sum_{z \in S\cup\{z_j'\}}(\nabla\ell(\mathbf{w}_t, z) - \nabla\ell(\mathbf{w}_t', z)) + \frac{\eta_t}{n}(\nabla\ell(\mathbf{w}_t, z_j') - \nabla\ell(\mathbf{w}_t', z_j))\right\| \\ &\leq \|\mathbf{w}_t - \mathbf{w}_t'\| + \frac{\eta_t}{n}\left\|\nabla\ell(\mathbf{w}_t, z_j) - \nabla\ell(\mathbf{w}_t', z_j')\right\| \leq \|\mathbf{w}_t - \mathbf{w}_t'\| + \frac{2L\eta_t}{n},\end{aligned} \tag{6.38}$$

where we have used the non-expansitivity of gradient update rule (Lemma A.5) and the sub-additivity of the norm on the second step. The last inequality is by applying the L-Lipschitz condition of ℓ. Therefore we have the recursion

$$\|\mathbf{w}_{t+1} - \mathbf{w}_{t+1}'\| \leq \frac{\|\mathbf{w}_t - \mathbf{w}_t'\|}{\sqrt{1-\beta_t}} + \frac{2L\eta_t}{n\sqrt{1-\beta_t}}. \tag{6.39}$$

By the same argument as in proof of Theorem 6.1 starting with equation (6.14), we have:

$$\mathbb{E}[\delta_T] \leq \frac{2L}{n}\sum_{t=1}^{T-1}\eta_t\prod_{j=t}^{T-1}(1-\beta_j)^{-1/2}. \tag{6.40}$$

By letting $\beta_t = 1/(t+1)$, $\eta_t = \eta$ for some absolute constant η and $T \asymp \sqrt{n}$, we have

$$\mathbb{E}[\delta_T] \leq \frac{2L\eta\sqrt{T}}{n}\sum_{t=1}^{T-1}\frac{1}{\sqrt{t}} = \mathcal{O}\left(\frac{L\eta}{\sqrt{n}}\right). \tag{6.41}$$

$\square$

Proof of Theorem 6.5. For simplicity, let ∇L_t denote the gradient $\nabla L_S(\mathbf{w}_t)$ at iteration t. By Jensen's inequality we have

$$\begin{aligned} L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*) &= L_S\left(\left(\sum_{t=1}^T \eta_t\right)^{-1} \sum_{t=1}^T \eta_t \mathbf{w}_t\right) - L_S(\mathbf{w}^*) \\ &\leq \left(\sum_{t=1}^T \eta_t\right)^{-1} \sum_{t=1}^T \eta_t (L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)) \\ &\leq \left(\sum_{t=1}^T \eta_t\right)^{-1} \sum_{t=1}^T \eta_t \langle \nabla L_t, \mathbf{w}_t - \mathbf{w}^* \rangle, \end{aligned} \tag{6.42}$$

where the last inequality is by the convexity of ℓ.

To bound the terms above, note that we have for all $t \geq 1$:

$$\begin{aligned} (1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 &\leq \|(\mathbf{w}_t - \mathbf{w}^*) - \eta_t \nabla L_t\|^2 \\ &= \|\mathbf{w}_t - \mathbf{w}^*\|^2 + 2\eta_t \langle \nabla L_t, \mathbf{w}^* - \mathbf{w}_t \rangle + 4\eta_t^2 L^2. \end{aligned}$$

Rearranging, we have:

$$2\eta_t \langle \nabla L_t, \mathbf{w}_t - \mathbf{w}^* \rangle \leq \|\mathbf{w}_t - \mathbf{w}^*\|^2 - \|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 + \beta_t \|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 + 4\eta_t^2 L^2. \tag{6.43}$$

Taking expectation and summing over T iterates we have:

$$\begin{aligned} 2\sum_{t=1}^T \eta_t \mathbb{E}[\langle \nabla L_t, \mathbf{w}_t - \mathbf{w}^* \rangle] &\leq \|\mathbf{w}_1 - \mathbf{w}^*\|^2 + \sum_{t=1}^T \beta_t \mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2] + 4L^2 \sum_{t=1}^T \eta_t^2 \\ &= \mathcal{O}\left(\|\mathcal{C}\| + \|\mathcal{C}\|^2 \sum_{t=1}^T \beta_t + L^2 \sum_{t=1}^T \eta_t^2\right) \\ &= \mathcal{O}\left(\|\mathcal{C}\|^2 \log(T) + L^2 \sum_{t=1}^T \eta_t^2\right), \end{aligned} \tag{6.44}$$

where we have used $\mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}^*\|]^2 \leq \|\mathcal{C}\|^2$. Hence for $\eta_t = \eta/\sqrt{t}$ we have

$$\left(\sum_{t=1}^T \eta_t\right)^{-1} \sum_{t=1}^T \eta_t \mathbb{E}[\langle \nabla L_t, \mathbf{w}_t - \mathbf{w}^* \rangle] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 \log(T)}{\sqrt{T}} + \frac{L^2}{\sqrt{T}} \sum_{t=1}^T \frac{1}{t}\right). \tag{6.45}$$

Finally, combining the above inequalities we have

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T) - L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(T)}{\sqrt{T}}\right). \tag{6.46}$$

□

Proof of Theorem 6.6. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. W.l.o.g. assume that they differ on the j-th point denoted z_j, z_j' for S and S', respectively. Fix a sample z, by the Lipschitz condition we get

$$\mathbb{E}[|\ell(\mathbf{w}_T) - \ell(\mathbf{w}_T')|] \leq L\mathbb{E}[\delta_T], \tag{6.47}$$

where $\delta_T = \|\mathbf{w}_T - \mathbf{w}_T'\|$.

Since after each update, $\mathbf{w}_t \in \mathcal{C}$ for all t, by Lemma 6.10 we have (here we will denote $\nabla L_S(\mathbf{w}_t)$ by ∇L_t and $\nabla L_{S'}(\mathbf{w}_t')$ by $\nabla L_t'$)

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}_{t+1}'\|^2 &\leq \|(\mathbf{w}_t - \mathbf{w}_t') - \eta_t(\nabla L_S(\mathbf{w}_t) - \nabla L_{S'}(\mathbf{w}_t'))\|^2 \\ &\leq \|\mathbf{w}_t - \mathbf{w}_t'\|^2 + 2\eta_t\langle \mathbf{w}_t' - \mathbf{w}_t, \nabla L_t - \nabla L_t'\rangle + 4\eta_t^2 L^2, \end{aligned} \tag{6.48}$$

where the last line follows from the Lipschitz assumption.

Since S, S' only differs on the j-th point, we have

$$\begin{aligned}\langle \mathbf{w}_t' - \mathbf{w}_t, \nabla L_t - \nabla L_t'\rangle &= \langle \mathbf{w}_t' - \mathbf{w}_t, \nabla L_{S\cup\{z_j'\}}(\mathbf{w}_t) - \nabla L_{S'\cup\{z_j\}}(\mathbf{w}_t')\rangle \\ &\qquad + \frac{1}{n}\langle \mathbf{w}_t' - \mathbf{w}_t, \nabla\ell(\mathbf{w}_t', z_j) - \nabla\ell(\mathbf{w}_t, z_j')\rangle \\ &\leq \frac{1}{n}\langle \mathbf{w}_t' - \mathbf{w}_t, \nabla\ell(\mathbf{w}_t', z_j) - \nabla\ell(\mathbf{w}_t, z_j')\rangle \\ &\leq \frac{2L}{n}\|\mathbf{w}_t - \mathbf{w}_t'\|, \end{aligned} \tag{6.49}$$

where the second line holds because the first term is negative by the convexity of L_S, and the last line follows from the Lipschitz condition.

We substitute the inequality (6.49) into equation (6.48) and obtain:

$$(1-\beta_t)\delta_{t+1}^2 = \delta_t^2 + 4L\eta_t\left(\frac{\delta_t}{n} + L\eta_t\right). \tag{6.50}$$

By multiplying both sides by $\prod_{j=1}^{t-1}(1-\beta_j)$, we have

$$\left[\prod_{j=1}^{t}(1-\beta_j)\right]\delta_{t+1}^2 \le \left[\prod_{j=1}^{t-1}(1-\beta_j)\right]\delta_t^2 + \frac{4\eta_t L\delta_t}{n}\prod_{j=1}^{t-1}(1-\beta_j) + 4\eta_t^2L^2\left[\prod_{j=1}^{t-1}(1-\beta_j)\right]. \tag{6.51}$$

By summing over T iterates we have:

$$\left[\prod_{j=1}^{T-1}(1-\beta_j)\right]\delta_T^2 \le \sum_{t=1}^{T-1}\frac{4\eta_t L\delta_t}{n}\prod_{j=1}^{t-1}(1-\beta_j) + \sum_{t=1}^{T-1}4\eta_t^2L^2\left[\prod_{j=1}^{t-1}(1-\beta_j)\right]. \tag{6.52}$$

Taking $\beta_t = 1/t+1$ and rearranging we have:

$$\delta_T^2 \le \frac{4LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t\delta_t}{t} + 4L^2T\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}. \tag{6.53}$$

The rest of the proof follows from the same procedure as in proof of Thm. 7.3, starting with equation (7.18). □

6.5.4 Proofs for Compressed Mini-batch SGD

Proof of Theorem 6.7. By Thm.A.6 with $\mathbf{w} = \mathbf{w}^*$, we have for all $t \ge 1$:

$$\begin{aligned}
(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2 &\le \|(\mathbf{w}_t-\mathbf{w}^*) - \frac{\eta_t}{b}\sum_{z\in B_t}\nabla\ell(\mathbf{w}_t, z_{i_t})\|^2 \\
&= \|\mathbf{w}_t-\mathbf{w}^*\|^2 + 2\eta_t\Big\langle\frac{1}{b}\sum_{z\in B_t}\nabla\ell(\mathbf{w}_t,z), \mathbf{w}^*-\mathbf{w}_t\Big\rangle + \eta_t^2\|\frac{1}{b}\sum_{z\in B_t}\nabla\ell(\mathbf{w}_t,z)\|^2 \\
&\le \|\mathbf{w}_t-\mathbf{w}^*\|^2 + \frac{2\eta_t}{b}\sum_{z\in B_t}(\ell(\mathbf{w}^*,z)-\ell(\mathbf{w}_t,z)) + \eta_t^2L^2.
\end{aligned}$$

Rearranging the above inequality gives

$$\frac{2\eta_t}{b}\sum_{z\in B_t}(\ell(\mathbf{w}_t)-\ell(\mathbf{w}^*)) \le \|\mathbf{w}_t-\mathbf{w}^*\|^2-\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2+\beta_t\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2+\eta_t^2L^2.$$

Since each minibatch B_t is drawn uniformly from the sample, we note that $\mathbb{E}_A[L_{B_t}(\mathbf{w})] = \mathbb{E}_A[L_S(\mathbf{w})]$. Hence, setting $\beta_t = 1/(t+1)$, taking expectation and summing over T iterates give

$$\begin{aligned}
2\sum_{t=1}^{T}\eta_t\mathbb{E}_{S,A}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)] &\le \|\mathbf{w}_0-\mathbf{w}^*\|^2+\sum_{t=1}^{T}\beta_t\mathbb{E}[\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2]+L^2\sum_{t=1}^{T}\eta_t^2\\
&= \mathcal{O}\left(\|\mathcal{C}\|^2+\|\mathcal{C}\|^2\sum_{t=1}^{T}\frac{1}{t+1}+L^2\sum_{t=1}^{T}\eta_t^2\right)\\
&= \mathcal{O}\left(\|\mathcal{C}\|^2\log(T)+L^2\sum_{t=1}^{T}\eta_t^2\right),
\end{aligned}$$

where we have used $\mathbb{E}[\|\mathbf{w}_{t+1}-\mathbf{w}^*\|]^2 \le \|\mathcal{C}\|^2$. Finally, choosing $\eta_t = \frac{\eta}{\sqrt{t}}$ we have

$$\left(\sum_{t=1}^{T}\eta_t\right)^{-1}\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)] = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2\log(T)}{\sqrt{T}}+\frac{L^2\log(T)}{\sqrt{T}}\right). \tag{6.54}$$

The proof is completed. □

Proof of Theorem 6.8. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. Denote the gradient updates by $G_1,\ldots,G_T$ and $G'_1,\ldots,G'_T$ induced by running the compressed SGD on S and S', respectively. Let $\delta_T = \|\mathbf{w}_T-\mathbf{w}'_T\|$. Observe that at step t, with probability $1-b/n$, the minibatch B_t, B'_t selected is the same in both S and S'. In this case we have $G_t = G'_t$ and we use the expansivity of the update G_t by a similar proof as Lemma 6.11. With probability b/n the selected minibatch B_t is different in which case assume they differ by the j-th point and we have from Lemma 6.10

$$\sqrt{1-\beta_t}\|\mathbf{w}_{t+1}-\mathbf{w}'_{t+1}\| \le \|(\mathbf{w}_t-\mathbf{w}'_t)-\eta_t(\nabla L_{B_t}(\mathbf{w}_t)-\nabla L_{B'_t}(\mathbf{w}'_t))\|. \tag{$*$}$$

Note that since B_t, B'_t only differ on the j-th point, we have $B_t \cup \{z'_j\} = B'_t \cup \{z_j\}$.

$$\begin{aligned}(*) &= \left\| (\mathbf{w}_t - \mathbf{w}'_t) - \frac{\eta_t}{b} \sum_{z \in B_t \cup \{z'_j\}} (\nabla \ell(\mathbf{w}_t, z) + \nabla \ell(\mathbf{w}'_t, z)) - \frac{\eta_t}{b}(\nabla \ell(\mathbf{w}_t, z'_j) - \nabla \ell(\mathbf{w}'_t, z_j)) \right\| \\ &\leq \|\mathbf{w}_t - \mathbf{w}'_t\| + \frac{\eta_t}{b} \left\|\nabla \ell(\mathbf{w}_t, z_j) - \nabla \ell(\mathbf{w}'_t, z'_j)\right\| \leq \|\mathbf{w}_t - \mathbf{w}'_t\| + \frac{2L\eta_t}{b}, \end{aligned} \tag{6.55}$$

where we have used the non-expansitivity of gradient update (Lemma A.5) and the sub-additivity of the norm on the second step. The last inequality is by applying the L-Lipschitz condition of ℓ. Hence, combining the two cases and by the linearity of expectation we have the following:

$$\mathbb{E}[\delta_{t+1}] \leq \left(\frac{1 - b/n}{\sqrt{1-\beta_t}} \right) \mathbb{E}[\delta_t] + \frac{b}{n\sqrt{1-\beta_t}} \left(\mathbb{E}[\delta_t] + \frac{2L\eta_t}{b} \right). \tag{6.56}$$

The rest of the proof for stability then follows by the same argument as in proof of Theorem 6.1 starting with equation (6.14). □

Proof of Theorem 6.9. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. Let $G(\mathbf{w}_t) = \mathbf{w}_{t+1}$ denote the gradient update and let $G_1, \dots, G_T$ and $G'_1, \dots, G'_T$ be the updates induced by running the compressed SGD on S and S' for T iterates, respectively. Let $\delta_T = \|\mathbf{w}_T - \mathbf{w}'_T\|$, by the Lipschitz condition,

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}'_T, z)|] \leq L\mathbb{E}[\delta_T]. \tag{6.57}$$

If at iteration t, the mini-batch B_t, B'_t we selected is the same, i.e. $G_t = G'_t$, then from Lemma 6.10 we have the following

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 &\leq \|(\mathbf{w}_t - \mathbf{w}'_t) - \eta_t(\nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t))\|^2 \\ &= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - 2\eta_t \langle \nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t \rangle + \eta_t^2 \|\nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t)\|^2. \end{aligned}$$

From the convexity of ℓ we have that $\langle \nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t \rangle \geq 0$ and from Lipschitzness of ℓ we also have $\|\nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t)\| \leq 2L$. Hence we obtain the following bound

$$(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}'_{t+1}\|^2 \le \|\mathbf{w}_t-\mathbf{w}'_t\|^2 + 4L^2\eta_t^2. \tag{6.58}$$

For the case where $G_t \neq G'_t$, note that B_t and B'_t differ by a single sample, hence we have

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}'_{t+1}\|^2 &\le \|(\mathbf{w}_t-\mathbf{w}'_t)-\eta_t(\nabla L_{B_t}(\mathbf{w}_t)-\nabla \ell_{B'_t}(\mathbf{w}'_t))\|^2\\ &\le \|\mathbf{w}_t-\mathbf{w}'_t\|^2 - 2\eta_t\langle \nabla L_{B_t}(\mathbf{w}_t)-\nabla\ell_{B'_t}(\mathbf{w}'_t), \mathbf{w}_t-\mathbf{w}'_t\rangle + \eta_t^2\|\nabla L_{B_t}(\mathbf{w}_t)-\nabla\ell_{B'_t}(\mathbf{w}'_t)\|^2\\ &\le \|\mathbf{w}_t-\mathbf{w}'_t\|^2 + \frac{4L\eta_t}{b}\|\mathbf{w}_t-\mathbf{w}'_t\| + 4L^2\eta_t^2, \end{aligned} \tag{6.59}$$

where the last inequality (6.59) follows by the same derivation as for equation (6.49), replacing S, S' with B_t, B'_t respectively. Combining the two cases we have

$$\begin{aligned}(1-\beta_t)\mathbb{E}[\delta_{t+1}^2] &\le \left(1-\frac{b}{n}\right)(\mathbb{E}[\delta_t^2]+4L^2\eta_t^2) + \frac{b}{n}\left(\mathbb{E}[\delta_t^2] + \frac{4L\eta_t\mathbb{E}[\delta_t]}{b} + 4L^2\eta_t^2\right)\\ &= \mathbb{E}[\delta_t^2] + 4L\eta_t\left(\frac{\mathbb{E}[\delta_t]}{n} + L\eta_t\right). \end{aligned} \tag{6.60}$$

The rest of the proof then follows by the same procedure as the proof for Theorem 7.3 starting from equation (6.50). □

Summary

We presented a rigorous analysis on the stability and generalization guarantee of SGD with compressed gradients. We have proved the first optimization error guarantee for two variants of CompSGD and show that they achieve the same optimization convergence as CompSGD in both smooth and non-smooth case. In particular, CompGD can achieve the same optimization convergence with larger step size parameter and fewer iterations. Furthermore, from our stability and generalization analysis we showed that we can obtain the same generalization convergence with compressed gradients in both smooth and non-smooth settings. We also show that the same approach can be extended to variants of SGD with similar guarantees. In particular, we require fewer iterations for these variants to obtain the same bounds.

Chapter Seven

Reducing the Noise of Differentially Private Gradient Descent

In this Chapter, we investigate the question of what is the effect of the projected gradients in the private setting. It is known that SGD in the DP setting will incur an additional term which will scale with the dimension [134]. Hence it is very interesting to find out whether the reduction on dimension will indeed reduce the dimension dependence in the error due to privacy. Furthermore, what is the interplay between privacy, dimensionality and optimisation performance? In this Chapter, we study this question for the first time. We first consider the classic gradient descent with projected gradients in the private setting in the following section.

7.1 Differentially Private Compressed Gradient Descent

In this section, we demonstrate and analyse the DP-SGD with compressed gradients. To guarantee differential privacy we impose the Gaussian mechanism to add Gaussian noise to the gradient updates. For classical DP gradient updates, we need to add noises in the original d-dimensional space which could be of very large size if d is large. We impose the compressed gradient updates in the private setting to add noise in a much lower dimension instead. The algorithm is introduced in the appendix of [76]. However, no convergence analysis has been done for the private setting. The detailed algorithm is outlined as in

Algorithm 8.

The algorithm uses a standard application of the *Gaussian mechanism* [47] to guarantee (ϵ_p, δ_p)-differential privacy. The main idea of the mechanism is to perturb the gradient update at each iteration by injecting noise. Similar approach has also been taken in [120] in the classical case (without projections).

Algorithm 8 Differentially private CompGD (DP-CompGD)

```
 1: Inputs: Sample set $S = \{z_1, \ldots, z_n\}$, privacy parameters $(\epsilon_p, \delta_p)$, step size parameters
    $\{\eta_t\}$ and projection parameters $\{\beta_t\}$.
 2: initialize $\mathbf{w}_1$ as any point in $\mathcal{C}$
 3: for $t = 1$ to $T$ do
 4:     set $m_t = \min\{d, \omega(\mathcal{C})^2/\beta_t^2\}$
 5:     choose projection matrix $\Phi_t \in \mathbb{R}^{m_t \times d}$ with i.i.d. entries from $\mathcal{N}(0, 1/m_t)$
 6:     set $\sigma^2 = \frac{32L^2T\log(1/\delta_p)}{n^2\epsilon_p^2}$
 7:     set $s_t = \frac{\|\nabla L_S(\mathbf{w}_t)\|}{\|\Phi_t \nabla L_S(\mathbf{w}_t)\|}$
 8:     set $\theta_t = \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t(s_t\Phi_t\nabla L_S(\mathbf{w}_t) + \mathbf{e}))$ where $\mathbf{e} \sim \mathcal{N}(0, \sigma^2 I_{m_t})$
 9:     pick $\mathbf{w}_{t+1}$ to be any element from the set $\{\mathbf{w} \in \mathcal{C} : \Phi_t\mathbf{w} = \theta_t\}$
10: end for
11: Output: $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_T$
```

We note that the variance of the injected noise σ^2 here depends not only on the privacy parameters (ϵ_p, δ_p), but also on the number of iterations T and the number of samples n. The dependence on T follows from the iterative property of gradient descent as we need to query the sample set once every epoch. The dependence on n follows from the use of the gradient $\nabla L_S(\mathbf{w}_t)$ for our algorithm. We remark that it is more preferable to use the full gradient in the privacy setting as compared to SGD because we can reduce the variance of the noise. Since differential privacy requires that the sensitivity of the gradient is bounded uniformly, we are required to set the normalization factor s_t as in the algorithm to guarantee this property. Other methods such as gradient clipping as in [35] also works similarly to bound the sensitivity uniformly.

One of the main challenge in the differentially private setting is to analyse the effect of the extra normalization factor s_t on the gradient updates, which will depend on the random projector Φ_t and the gradient ∇L_S. The random projection Φ_t only depends on the Gaussian width of $\mathcal{C}$ (and distortion parameter β_t), meaning we have no guarantee on the distortion of the projected gradients $\Phi_t\nabla L_S$, hence making the convergence of the projected gradient updates difficult to analyse. We overcome this difficulty by using the properties of Gaussian random matrices and show that the convergence of DP-CompGD is almost the same as high-dimension SGD (without projection) while reducing the dimensionality of the noise and the gradient. Hence we required the random projection matrix Φ to be Gaussian distributed in the private case as stated in Alg. 8 which is an additional restriction compared to the non-private case (e.g. Alg. 6).

Theorem 7.1 (Non-smooth case)**.** *Assume that the loss function ℓ is convex and L-Lipschitz over the convex set $\mathcal{C}$. Suppose we run the DP compressed GD with step sizes $\eta_t = \|\mathcal{C}\|/\sqrt{t(L^2 + m_T\sigma^2)}$. For privacy parameters ϵ_p, δ_p we let $\sigma^2 = \mathcal{O}(L^2 \log(1/\delta_p)T/(\epsilon_p^2 n^2))$ and $\beta_t = 1/(t+1)$. Then we have that the private compressed GD satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|L}{\sqrt{T}} + \frac{\log T\|\mathcal{C}\|L\sqrt{m_T\log(1/\delta_p)}}{n\epsilon_p}\right), \tag{7.1}$$

where $m_T = \max_{t\in[n]} m_t \leq d$.

Proof. (*sketch*) The proof of the optimization guarantee for the DP CompGD follows a similar logic as in the non-private setting. However, we must consider carefully the additional normalization factor s_t which is dependent on Φ, and the magnitude of the added noise in each iteration. For the former we use the properties of Gaussian random matrices to show a key relationship between inner products on the high-dimensional vector and the projected vector (normalized by s_t). This allowed us to relate projected vectors to high-dimensional vectors and use techniques from classic GD. For the magnitude of the noise we can use the Lipschitz assumption on the high-dimensional gradients together with properties of the added Gaussian noise with variance σ^2. The detailed proof is in Section 7.3.1. □

We note that the second term in (7.1) has the privacy parameter ϵ_p in the denominator, which implies that the second term will vanish as ϵ_p tends to infinity (zero privacy). In that case we recover the same convergence rate as in the non-private case. We also note that m_T is dependent on the Gaussian width of the constraint set $\mathcal{C}$. This captures the dimensionality reduction from d to $\omega(\mathcal{C})$, which can be much smaller if the set has a low dimensional structure. e.g. if $\mathcal{C}$ is the ℓ_1-ball, then we have $\omega(\mathcal{C}) = \mathcal{O}(\sqrt{\log d})$.

In the case of smooth ℓ, we can obtain a faster convergence in optimization just as we have observed for the non-private case.

Theorem 7.2 (Smooth case). *Assume that the loss function ℓ is convex, μ-smooth and L-Lipschitz over the convex set $\mathcal{C}$. Suppose that we run the DP compressed GD with step sizes $\eta_t = \frac{\|\mathcal{C}\|}{L\sqrt{m_T}} \leq 1/(4\mu)$. For privacy parameters ϵ_p, δ_p we let $\sigma^2 = \mathcal{O}(\log(1/\delta_p)L^2T/(\epsilon_p^2 n^2))$ and $\beta_t = 1/(t+1)$. Then we have that the private compressed GD satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{L\|\mathcal{C}\| \log T\sqrt{m_T}}{T} + \frac{LT\sqrt{m_T}\log(1/\delta_p)}{n^2\epsilon_p^2}\right),$$

where $m_T = \max_{t\in[n]} m_t \leq d$.

Proof. See Section 7.3.1. □

Similar to the non-smooth case, we note that when the privacy parameters ϵ_p, δ_p converge to infinity, we will recover the same convergence bound as in the non-private case for compressed gradient descent.

Remark 17. *Notice here that we have specified the learning rate η_t needed for the optimization error bounds in Thm. 7.1 and Thm. 7.2. The only necessary dependence in η_t is t, σ^2, and m_T for the dimensionality dependence. Other constants can be chosen freely without affecting the result (up to constant factors) in a similar way as previous results (as in e.g. Thm. 7.2).*

Theorem 7.3 (Stability and generalization of Compressed Private GD). *Assume that the loss function ℓ is convex and L-Lipschitz for every $\mathbf{w} \in \mathcal{C}, z \in \mathcal{Z}$.*

1. *Then, for $\beta_t = \frac{1}{t+1}$, $\eta_t = \frac{\eta}{T^{3/4}}$ for some absolute constant η and $T \asymp n^2$, the differentially private CompGD is ϵ_{stab}-uniformly stable with $\epsilon_{stab} = \mathcal{O}\left(\frac{L\log(n)}{\sqrt{n}}\right)$.*

2. *Moreover, the weighted average output $\bar{\mathbf{w}}_T$ satisfies the following excess risk bound*

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}} + \frac{\log(1/\delta_p)}{n^3\sqrt{n}\epsilon_p^2}\sum_{t=1}^{n^2} m_t\right). \tag{7.2}$$

Proof. See Section 7.3.1. □

Theorem 7.3 shows that the generalization convergence of DP-CompGD achieves the same rate as CompGD up to privacy and constant factors. In particular, we recover the same guarantee as for CompGD (theorem 6.6) when $\epsilon_p \to \infty$. Moreover, note that the dimensionality dependence in the privacy error term (last term) is dependent on the projection dimension used for each iteration. Hence if the projection dimension is small then we can reduce the error incurred due to privacy, as compared to high-dimensional DP-SGD [134]. We omit the generalization in the smooth case here since there is no gain compared to the non-smooth setting.

7.2 Differentially Private Compressed SGD with Mini-batch

While we prefer using large batch gradients in the private setting to reduce the global sensitivity of the gradient and improve optimization with fewer iterations, private mini-batch SGD can be useful for large sample sets. Moreover, the mini-batch act as a trade-off parameter between computational complexity and accuracy of gradient updates. In this section, we present the differentially private CompSGD algorithm using mini-batch gradient updates, the algorithm is outlined in Alg. 9.

Note that the mini-batch version of DP-CompSGD induces an extra log factor in its variance σ^2, which will lead to an extra multiplicative log factor in the optimization and generalization bounds. This is a trade-off in privacy from using a smaller batch of samples in each gradient update (instead of the full batch as in Alg. 8). Fortunately, we are still able to obtain the same convergence guarantees for the mini-batch as in Section 7.1 when

Algorithm 9 Differentially private CompSGD with mini-batch

```
1: Inputs: Sample set $S = \{z_1, \dots, z_n\}$, batch size $b$, privacy parameters $(\epsilon_p, \delta_p)$, step
   size parameters $\{\eta_t\}$ and projection parameters $\{\beta_t\}$.
2: initialize $\mathbf{w}_1$ as any point in $\mathcal{C}$
3: for $t = 1$ to $T$ do
4:     set $m_t = \min\{d, \omega(\mathcal{C})^2/\beta_t^2\}$
5:     choose projection matrix $\Phi_t \in \mathbb{R}^{m_t \times d}$ with i.i.d. entries from $\mathcal{N}(0, 1/m_t)$
6:     Sample a mini-batch $B_t$ of size $b$ uniformly from $S$
7:     set $\sigma^2 = \frac{16^2 LT \log(1/\delta_p) \log(2.5Tb/(\delta_p n))}{n^2 \epsilon_p^2}$
8:     set $s_t = \frac{\|\nabla L_{B_t}(\mathbf{w}_t)\|}{\|\Phi_t \nabla L_{B_t}(\mathbf{w}_t)\|}$
9:     set $\theta_t = \Pi_{\Phi_t \mathcal{C}}(\Phi_t \mathbf{w}_t - \eta_t(s_t \Phi_t \nabla L_{B_t}(\mathbf{w}_t) + \mathbf{e}))$ where $\mathbf{e} \sim \mathcal{N}(0, \sigma^2 I_{m_t})$
10:    pick $\mathbf{w}_{t+1}$ to be any element from the set $\{\mathbf{w} \in \mathcal{C} : \Phi_t \mathbf{w} = \theta_t\}$
11: end for
12: Output: $\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_T$
```

ϵ_p tends to infinity.

Theorem 7.4 (Optimization). *Assume that the loss function ℓ is convex and L-Lipschitz over the convex set $\mathcal{C}$. Suppose we run the DP CompGD with step sizes $\eta_t = \frac{\|\mathcal{C}\|}{\sqrt{t(L^2 + m_T \sigma^2)}}$. For privacy parameters $0 \le \epsilon_p, \delta_p \le 1$ and $\beta_t = 1/(t+1)$, the private compressed SGD with mini-batch satisfies*

$$\mathbb{E}[L_S(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*) + \mathcal{O}\left(\frac{\log T \|\mathcal{C}\| L}{\sqrt{T}} + \frac{\log T \|\mathcal{C}\| L \sqrt{m_T \log(1/\delta_p) \log(4Tb/(\delta_p n))}}{n \epsilon_p} \right).$$

Similar to the non-private case for the mini-batch variance, there is no improvement with the additional smoothness condition, we obtain the same convergence for both cases. The rate we obtained here is same as the result obtained with non-projected gradients in [12] with a key difference: the dimensionality dependence $\sqrt{d}$ is replaced with the maximum projection dimension $\sqrt{m_T}$. Hence the reduction of dimensionality here comes for 'free' compared with using non-projected gradients.

Theorem 7.5 (Stability and generalization of DP-CompSGD with minibatch). *Assume that the loss function ℓ is convex and L-Lipschitz for every $\mathbf{w} \in \mathcal{C}, z \in \mathcal{Z}$.*

1. *Then, for $\beta_t = \frac{1}{t+1}$, $\eta_t = \frac{\eta}{T^{3/4}}$ for some absolute constant η and $T \asymp n^2$, the differentially private minibatch CompSGD is ϵ_{stab}-uniformly stable with $\epsilon_{stab} = \mathcal{O}\left(\frac{L \log(n)}{\sqrt{n}}\right)$.*

2. Moreover, the weighted average output $\bar{\mathbf{w}}_T$ satisfies the following excess risk bound

$$\mathbb{E}[L_{\mathcal{D}}(\bar{\mathbf{w}}_T)] = L_{\mathcal{D}}(\mathbf{w}^*)+\mathcal{O}\left(\frac{(\|\mathcal{C}\|^2 + L^2)\log(n)}{\sqrt{n}} + \frac{L^2\log(4Tb/(\delta_p n))\log(1/\delta_p)}{n^3\sqrt{n}\epsilon_p^2}\sum_{t=1}^{n^2} m_t\right).$$

Remark 18. *Note that the generalization bound obtained here is tight, as we required at least $\mathcal{O}(1/\sqrt{n})$ even in the non-private case. The convergence rate will be almost the same (up to* log *factors) as the non-private case if the privacy parameter is not too small - $\epsilon_p \asymp 1/\sqrt{n}$.*

7.3 Proofs for Differentially Private SGD

7.3.1 Proofs for DP-CompGD

Before proving the utility guarantee of the algorithm, we require the following lemma as part of our proof, the proof of the lemma is deferred to later on in this section to focus on the main theorem.

Lemma 7.6. *For $\mathbf{w}_t$ in DP compressed SGD algorithm, we have for all t,*

$$\mathbb{E}[\langle\Phi_t(\mathbf{w}-\mathbf{w}_t), s_t\Phi_t\nabla\ell(\mathbf{w}_t)\rangle] = \langle\mathbf{w}-\mathbf{w}_t, \nabla\ell(\mathbf{w}_t)\rangle C_{m_t},$$

where $C_{m_t} = \sqrt{\frac{2}{m_t}}\frac{\Gamma((m_t+1)/2)}{\Gamma(m_t/2)} \in \left[\sqrt{\frac{m_t}{m_t+1}}, 1\right]$, and $\Gamma(\cdot)$ is the gamma function.

Lemma 7.6 shows that the inner product between the projected weight vector and the normalized projected gradient vector is almost identical to their inner product before projection. This is a key observation that allows us to prove a similar convergence result with projected gradients. Using Lemma 7.6, we now show the following optimization bound for the differentially private compressed gradient descent.

Proof of Theorem 7.1. Since after each update, $\mathbf{w}_t \in \mathcal{C}$ for all t, by equation (A.13) in the proof of Thm. A.6 we have

$$(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}\|^2 \le \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_{t+1}-\mathbf{w})\|^2\right]. \tag{7.3}$$

Hence we have (here we will denote $\nabla L_S(\mathbf{w}_t)$ by ∇L_t)

$$\begin{aligned}
(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}\|^2 &\le \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t-\eta_t(s_t\Phi_t\nabla L_t+\mathbf{e}))-\Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w})\|^2\right]\\
&\le \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Phi_t\mathbf{w}_t-\eta_t(s_t\Phi_t\nabla L_t+\mathbf{e})-\Phi_t\mathbf{w}\|^2\right]\\
&= \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Phi_t\mathbf{w}_t-\Phi_t\mathbf{w}\|^2\right]+2\eta_t\mathbb{E}_{\Phi_t,\mathbf{e}}[\langle\Phi_t\mathbf{w}-\Phi_t\mathbf{w}_t, s_t\Phi_t\nabla L_t+\mathbf{e}\rangle]\\
&\quad+\eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|s_t\Phi_t\nabla L_t+\mathbf{e}\|^2]\\
&= \|\mathbf{w}_t-\mathbf{w}\|^2+2C_{m_t}\eta_t\langle\mathbf{w}-\mathbf{w}_t,\nabla L_t\rangle+\eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|s_t\Phi_t\nabla L_t\|^2]\\
&\quad+2\eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\langle s_t\Phi_t\nabla L_t,\mathbf{e}\rangle]+\eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|\mathbf{e}\|^2]\\
&= \|\mathbf{w}_t-\mathbf{w}\|^2+2C_{m_t}\eta_t\langle\mathbf{w}-\mathbf{w}_t,\nabla L_t\rangle+\eta_t^2(L^2+m_t\sigma^2)\\
&\le \|\mathbf{w}_t-\mathbf{w}\|^2+2C_{m_t}\eta_t(L_S(\mathbf{w})-L_S(\mathbf{w}_t))+\eta_t^2(L^2+m_t\sigma^2), \quad (7.4)
\end{aligned}$$

where we have used Lemma 7.6 between the fourth and fifth line and the convexity of ℓ in the last step. Also note that since $\mathbf{e}$ is i.i.d. fresh Gaussian noise, the expectation of $\mathbf{e}$ is 0. Hence the expected inner product with $\mathbf{e}$ is also zero. Rearranging the last inequality and let $\mathbf{w}=\mathbf{w}^*$ we have:

$$2C_{m_t}\eta_t(L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)) \le \|\mathbf{w}_t-\mathbf{w}^*\|^2-(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2+\eta_t^2(L^2+m_t\sigma^2). \tag{7.5}$$

Taking expectation and summing over T iterations we have

$$2\sum_{t=1}^{T}C_{m_t}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)] \le \|\mathbf{w}_1-\mathbf{w}^*\|^2+\sum_{t=1}^{T}\beta_t\mathbb{E}[\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2]+\sum_{t=1}^{T}\eta_t^2(L^2+m_t\sigma^2).$$

Choosing $\beta_t = 1/(t+1)$ we obtain that

$$\frac{\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)]}{\sum_{t=1}^{T}\eta_t} = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2+\log T\|\mathcal{C}\|^2+(L^2+m_T\sigma^2)\sum_{t=1}^{T}\eta_t^2}{\sum_{t=1}^{T}\eta_t}\right), \tag{7.6}$$

where we have used $m_T = \max_{t\in[T]} m_t$.

Finally, let $\eta_t = \|\mathcal{C}\|/\sqrt{t(L^2 + m_T\sigma^2)}$ and $\sigma^2 = \mathcal{O}(TL^2 \log(1/\delta_p)/(\epsilon_p^2 n^2))$ we have

$$\begin{aligned}\frac{\sum_{t=1}^T \eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)]}{\sum_{t=1}^T \eta_t} &= \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|\sqrt{L^2 + m_T\sigma^2}}{\sqrt{T}}\right)\\ &\le \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|L}{\sqrt{T}} + \frac{\log T\|\mathcal{C}\|L\sqrt{m_T T\log(1/\delta_p)}}{n\epsilon_p\sqrt{T}}\right)\\ &= \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|L}{\sqrt{T}} + \frac{\log T\|\mathcal{C}\|L\sqrt{m_T \log(1/\delta_p)}}{n\epsilon_p}\right). \quad (7.7)\end{aligned}$$

The proof is completed. □

Proof of Theorem 7.2. Since after each update, $\mathbf{w}_t \in \mathcal{C}$ for all t, by equation (6.29) in the proof of lemma 6.10 (replacing $\mathbf{w}'_{t+1}$ with $\mathbf{w}$) we have

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}\|^2 \le \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_{t+1} - \mathbf{w})\|^2\right]. \quad (7.8)$$

Hence we have

$$\begin{aligned}&(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}\|^2 \le \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Phi_t(\mathbf{w}_{t+1} - \mathbf{w})\|^2\right]\\ &= \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t(s_t\Phi_t\nabla L_t + \mathbf{e})) - \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w})\|^2\right]\\ &\le \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Phi_t\mathbf{w}_t - \eta_t(s_t\Phi_t\nabla L_t + \mathbf{e}) - \Phi_t\mathbf{w}\|^2\right]\\ &= \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Phi_t\mathbf{w}_t - \Phi_t\mathbf{w}\|^2\right] + \mathbb{E}_{\Phi_t,\mathbf{e}}[\langle\Phi_t\mathbf{w} - \Phi_t\mathbf{w}_t, s_t\Phi_t\nabla L_t + \mathbf{e}\rangle] + \eta^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|s_t\Phi_t\nabla L_t + \mathbf{e}\|^2]\\ &= \|\mathbf{w}_t - \mathbf{w}\|^2 + 2C_{m_t}\eta_t\langle\mathbf{w} - \mathbf{w}_t, \nabla L_t\rangle + \eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|s_t\Phi_t\nabla L_t\|^2]\\ &\qquad\qquad + 2\eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\langle s_t\Phi_t\nabla L_t, \mathbf{e}\rangle] + \eta_t^2\mathbb{E}_{\Phi_t,\mathbf{e}}[\|\mathbf{e}\|^2]\\ &= \|\mathbf{w}_t - \mathbf{w}\|^2 + 2C_{m_t}\eta_t\langle\mathbf{w} - \mathbf{w}_t, \nabla L_t\rangle + \eta_t^2(\|\nabla L_t\|^2 + m_t\sigma^2)\\ &\le \|\mathbf{w}_t - \mathbf{w}\|^2 + 2C_{m_t}\eta_t(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) + \eta_t^2(\|\nabla L_t\|^2 + m_t\sigma^2), \quad (7.9)\end{aligned}$$

where we have used Lemma 7.6 between the fourth and fifth step and convexity of ℓ in the second to last step. Also note that since $\mathbf{e}$ is i.i.d. Gaussian noise, the expectation of $\mathbf{e}$ is 0. Hence the expected inner product with $\mathbf{e}$ is also zero. Now we substitute $\mathbf{w} = \mathbf{w}_S^*$.

Since $\mathbf{w}_S^*$ is an minimiser we have $\nabla L_S(\mathbf{w}_S^*) = 0$. Hence by smoothness we have

$$\|\nabla L_S(\mathbf{w}_t)\|^2 = \|\nabla L_S(\mathbf{w}_t) - \nabla L_S(\mathbf{w}_S^*)\|^2 \le 2\mu(L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)). \tag{7.10}$$

Substituting equation (7.10) into (7.9) we have

$$\begin{aligned}(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2 &\le \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 + (\eta_t - 2\eta_t^2\mu)(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) + \eta_t^2 m_t \sigma^2 \\ \text{(assuming } \eta_t \le 1/(4\mu)) \ &\le \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 + \frac{\eta_t}{2}(L_S(\mathbf{w}_S^*) - L_S(\mathbf{w}_t)) + \eta_t^2 m_t \sigma^2. \qquad (7.11)\end{aligned}$$

Rearranging we have:

$$\frac{\eta_t}{2}(L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)) \le \|\mathbf{w}_t - \mathbf{w}_S^*\|^2 - \|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2 + \beta_t\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2 + \eta_t^2 m_t \sigma^2.$$

Taking expectation and summing over T iterates and choosing $\beta_t = 1/(t+1)$, we have:

$$\begin{aligned}\sum_{t=1}^{T} \frac{\eta_t}{2}\mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)] &\le \|\mathbf{w}_1 - \mathbf{w}_S^*\|^2 + \sum_{t=1}^{T}\beta_t \mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|^2] + \sum_{t=1}^{T}\eta_t^2 m_t \sigma^2 \\ &= \mathcal{O}\left(\|\mathcal{C}\| + \|\mathcal{C}\|^2 \sum_{t=1}^{T}\beta_t + \sum_{t=1}^{T}\eta_t^2 m_t \sigma^2\right) \\ &= \mathcal{O}\left(\|\mathcal{C}\| + \|\mathcal{C}\|^2 \log(T) + \frac{\log(1/\delta_p)T}{n^2\epsilon_p^2}\sum_{t=1}^{T}\eta_t^2 m_t\right), \qquad (7.12)\end{aligned}$$

where we have used $\mathbb{E}[\|\mathbf{w}_{t+1} - \mathbf{w}_S^*\|]^2 \le \|\mathcal{C}\|^2$. Finally, for $\eta_t = \frac{\|\mathcal{C}\|}{L\sqrt{m_T}}$ we have

$$\left(\sum_{t=1}^{T}\eta_t\right)^{-1}\sum_{t=1}^{T}\eta_t \mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}_S^*)] = \mathcal{O}\left(\frac{L\|\mathcal{C}\|\log T\sqrt{m_T}}{T} + \frac{LT\sqrt{m_T}\log(1/\delta_p)}{n^2\epsilon_p^2}\right).$$

The proof is completed. □

Proof of Theorem 7.3. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. W.l.o.g. assume that they differ on the j-th point denoted z_j, z_j' for

S and S', respectively. Fix a sample z, by the Lipschitz condition we get

$$\mathbb{E}[|\ell(\mathbf{w}_T) - \ell(\mathbf{w}'_T)|] \leq L\mathbb{E}[\delta_T], \tag{7.13}$$

where $\delta_T = \|\mathbf{w}_T - \mathbf{w}'_T\|$.

Since after each update, $\mathbf{w}_t \in \mathcal{C}$ for all t, by equation (6.29) in the proof of Lemma 6.10 we have

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \leq \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_{t+1} - \mathbf{w}'_{t+1})\|^2\right]. \tag{$**$}$$

Hence we have (here we will denote $\nabla L_S(\mathbf{w}_t)$ by ∇L_t and $\nabla L_{S'}(\mathbf{w}'_t)$ by $\nabla L'_t$)

$$\begin{aligned}
(**) &= \mathbb{E}_{\Phi_t,\mathbf{e}}\left[\|\Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t(s_t\Phi_t\nabla L_t + \mathbf{e})) - \Pi_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}'_t - \eta_t(s'_t\Phi_t\nabla L'_t + \mathbf{e}))\|^2\right] \\
&\leq \mathbb{E}_{\Phi_t}\left[\|\Phi_t\mathbf{w}_t - \eta_t(s_t\Phi_t\nabla L_t) - (\Phi_t\mathbf{w}'_t - \eta_t(s'_t\Phi_t\nabla L'_t))\|^2\right] \\
&= \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_t - \mathbf{w}'_t)\|^2\right] + 2\eta_t\mathbb{E}_{\Phi_t}[\langle\Phi_t(\mathbf{w}'_t - \mathbf{w}_t), \Phi_t(s_t\nabla L_t - s'_t\nabla L'_t)\rangle] \\
&\quad + \eta_t^2\mathbb{E}_{\Phi_t}[\|\Phi_t(s_t\nabla L_t - s'_t\nabla L'_t)\|^2] \\
&= \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_t - \mathbf{w}'_t)\|^2\right] + 2\eta_t\mathbb{E}_{\Phi_t}[\langle\Phi_t(\mathbf{w}'_t - \mathbf{w}_t), \Phi_t s_t\nabla L_t\rangle - \langle\Phi_t(\mathbf{w}'_t - \mathbf{w}_t), \Phi_t s'_t\nabla L'_t\rangle] \\
&\quad + \eta_t^2\mathbb{E}_{\Phi_t}[\|\Phi_t(s_t\nabla L_t - s'_t\nabla L'_t)\|^2] \\
&= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + 2C_{m_t}\eta_t\langle\mathbf{w}'_t - \mathbf{w}_t, \nabla L_t - \nabla L'_t\rangle + \eta_t^2\mathbb{E}_{\Phi_t}[\|\Phi_t(s_t\nabla L_t - s'_t\nabla L'_t)\|^2] \qquad (7.14) \\
&= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + 2C_{m_t}\eta_t\langle\mathbf{w}'_t - \mathbf{w}_t, \nabla L_t - \nabla L'_t\rangle + 4\eta_t^2L^2, \qquad (7.15)
\end{aligned}$$

where the second-to-last line (7.14) follows by applying Lemma 7.6 twice, the last line (7.15) follows from the Lipschitz assumption.

Since S, S' only differs on the j-th point, we have (C_{m_t} omitted here since $C_{m_t} \leq 1$)

$$\begin{aligned}
\langle\mathbf{w}'_t - \mathbf{w}_t, \nabla L_t - \nabla L'_t\rangle &= \langle\mathbf{w}'_t - \mathbf{w}_t, \nabla L_{S\cup\{z'_j\}}(\mathbf{w}_t) - \nabla L_{S'\cup\{z_j\}}(\mathbf{w}'_t)\rangle \\
&\qquad + \frac{1}{n}\langle\mathbf{w}'_t - \mathbf{w}_t, \nabla\ell(\mathbf{w}'_t, z_j) - \nabla\ell(\mathbf{w}_t, z'_j)\rangle \\
&\leq \frac{1}{n}\langle\mathbf{w}'_t - \mathbf{w}_t, \nabla\ell(\mathbf{w}'_t, z_j) - \nabla\ell(\mathbf{w}_t, z'_j)\rangle \\
&\leq \frac{2L}{n}\|\mathbf{w}_t - \mathbf{w}'_t\|, \qquad (7.16)
\end{aligned}$$

where the second line holds because the first term is negative by the convexity of L_S, and the last line follows from the Lipschitz condition.

We substitute the inequality (7.16) into equation (7.15) and multiply both sides by $\prod_{j=1}^{t-1}(1-\beta_j)$. It then follows that

$$\left[\prod_{j=1}^{t}(1-\beta_j)\right]\delta_{t+1}^2 \le \left[\prod_{j=1}^{t-1}(1-\beta_j)\right]\delta_t^2 + \frac{4\eta_t L\delta_t}{n}\prod_{j=1}^{t-1}(1-\beta_j) + 4\eta_t^2 L^2\left[\prod_{j=1}^{t-1}(1-\beta_j)\right].$$

By summing over T iterates we have:

$$\left[\prod_{j=1}^{T-1}(1-\beta_j)\right]\delta_T^2 \le \sum_{t=1}^{T-1}\frac{4\eta_t L\delta_t}{n}\prod_{j=1}^{t-1}(1-\beta_j) + \sum_{t=1}^{T-1}4\eta_t^2 L^2\left[\prod_{j=1}^{t-1}(1-\beta_j)\right]. \tag{7.17}$$

Taking $\beta_t = 1/t+1$ and rearranging we have:

$$\delta_T^2 \le \frac{4LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t\delta_t}{t} + 4L^2T\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}. \tag{7.18}$$

Claim: The following inequality holds for all T:

$$\delta_T \le 2L\sqrt{T}\sqrt{\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}} + \left(\frac{2LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t}\right). \tag{7.19}$$

We prove this claim by induction: The base case $T=0$ clearly holds as the right hand side is always positive. For the inductive step, if $\delta_T \le \max_{t\in[T]}\delta_t$, then by the inductive hypothesis we have

$$\delta_T \le \delta_{T-1} \le 2L\sqrt{T}\sqrt{\sum_{t=1}^{T-2}\frac{\eta_t^2}{t}} + \left(\frac{2LT}{n}\sum_{t=1}^{T-2}\frac{\eta_t}{t}\right) \le 2L\sqrt{T}\sqrt{\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}} + \left(\frac{2LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t}\right). \tag{7.20}$$

For the other case where $\delta_T > \max_{t\in[T]}\delta_t$, we have from (7.18):

$$\delta_T^2 \le \frac{4LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t\delta_t}{t} + 4L^2T\sum_{t=1}^{T-1}\frac{\eta_t^2}{t} \le \frac{4LT\delta_T}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t} + 4L^2T\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}. \tag{7.21}$$

Which after rearranging is equivalent to:

$$\left(\delta_T - \frac{2LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t}\right)^2 \leq \left(\frac{2LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t}\right)^2 + 4L^2T\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}. \tag{7.22}$$

Taking square-root from both sides, we get the result by the sub-additivity of square-roots. The inductive step is completed. Finally using the choice $\eta_t = \mathcal{O}(1/T^{3/4})$ and $T \asymp n^2$ together with our proved claim, we have:

$$\begin{aligned}\mathbb{E}[\delta_T] &\leq 2L\sqrt{T}\sqrt{\sum_{t=1}^{T-1}\frac{\eta_t^2}{t}} + \left(\frac{2LT}{n}\sum_{t=1}^{T-1}\frac{\eta_t}{t}\right)\\ &= \mathcal{O}\left(\frac{L\sqrt{\log T}}{T^{1/4}} + \frac{LT^{1/4}\log T}{n}\right) = \mathcal{O}\left(\frac{L\log n}{\sqrt{n}}\right).\end{aligned}$$

For the excess risk bound we have from Thm. 7.1

$$\frac{\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)]}{\sum_{t=1}^{T}\eta_t} = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2 + \log T\|\mathcal{C}\|^2 + \sum_{t=1}^{T}\eta_t^2(L^2 + m_t\sigma^2)}{\sum_{t=1}^{T}\eta_t}\right). \tag{7.23}$$

Using the choice of $\eta_t = \eta/T^{3/4}$ and $T \asymp n^2$ we have

$$\frac{\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t) - L_S(\mathbf{w}^*)]}{\sum_{t=1}^{T}\eta_t} = \mathcal{O}\left(\frac{\log n(\|\mathcal{C}\|^2 + L^2)}{\sqrt{n}} + \frac{\log(1/\delta_p)\sum_{t=1}^{n^2}m_t}{\sqrt{n}n^3\epsilon_p^2}\right). \tag{7.24}$$

Combining with the stability bound we obtain our final result. □

Theorem 7.7. *The output of DP-CompGD in Alg. 8 satisfies (ϵ_p, δ_p)-differential privacy.*

Proof. The proof follows similar procedure as in [11] (Thm. 2.1) with a standard application of the Gaussian mechanism. Note that the norm of the projected gradient $\Phi_t\nabla L_S(\mathbf{w}_t)$ is normalized by the normalization factor s_t (line 7 of Alg. 8). Hence the norm $\|s_t\Phi_t\nabla L_S(\mathbf{w}_t)\|$ is upper bounded by the Lipschitz constant L which implies a global sensitivity of $2L$. The privacy guarantee then follows directly by applying the Gaussian mechanism with the strong composition theorem (Thm. 2.8) over T iterations of SGD. □

Definition 15 (Chi-distribution)**.** *The probability density function of chi-distribution is*

$$f(x;k) = \begin{cases} \frac{x^{k-1}e^{-x^2/2}}{2^{k/2-1}\Gamma\left(\frac{k}{2}\right)}, & \textit{if } x \geq 0, \\ 0, & \textit{otherwise}, \end{cases} \tag{7.25}$$

where $\Gamma(z)$ is the gamma function. It is known that the expected value of the chi-distribution is $\frac{\sqrt{2}\Gamma((k+1)/2)}{\Gamma(k/2)}$.

Proof of Lemma 7.6. Note that $\mathbf{e}$ is independent from the rest of parameters and $\mathbb{E}[\mathbf{e}] = 0$. Denoting $\nabla L_S(\mathbf{w}_t)$ by ∇L_t, we have

$$\begin{aligned} \mathbb{E}_{\Phi_t,\mathbf{e}}[\langle \Phi_t(\mathbf{w}-\mathbf{w}_t), s_t\Phi_t\nabla L_t + \mathbf{e}\rangle] &= \mathbb{E}_{\Phi_t,\mathbf{e}}[\langle \Phi_t(\mathbf{w}-\mathbf{w}_t), s_t\Phi_t\nabla L_t\rangle + \langle \Phi_t\mathbf{w} - \Phi\mathbf{w}_t, \mathbf{e}\rangle] \\ &= \mathbb{E}_{\Phi_t}[\langle \Phi_t(\mathbf{w}-\mathbf{w}_t), s_t\Phi_t\nabla L_t\rangle]. \end{aligned} \tag{7.26}$$

For simplicity let us denote $(\mathbf{w}-\mathbf{w}_t)$ by $\mathbf{v}$. To bound the above quantity, we first consider two special cases of ∇L_t: 1. ∇L_t is a scale multiple of $\mathbf{v}$; 2. ∇L_t is perpendicular to $\mathbf{v}$. For the first case, $\nabla L_t = c\mathbf{v}$ for some constant c. We have

$$\begin{aligned} \|\nabla L_t\|\mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t\mathbf{v}, \frac{\Phi_t\nabla L_t}{\|\Phi_t\nabla L_t\|}\right\rangle\right] &= sign(c)\|c\mathbf{v}\|\mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t\mathbf{v}, \frac{\Phi_t\mathbf{v}}{\|\Phi_t\mathbf{v}\|}\right\rangle\right] \\ &= sign(c)\|c\mathbf{v}\|\mathbb{E}_{\Phi_t}\left[\|\Phi_t\mathbf{v}\|\right] \\ &= sign(c)\frac{|c|\|\mathbf{v}\|^2}{\sqrt{m_t}}\mathbb{E}_{\Phi_t}\left[\frac{\|\Phi_t\mathbf{v}\|\sqrt{m_t}}{\|\mathbf{v}\|}\right]. \end{aligned} \tag{7.27}$$

Since Φ_t's entries are randomly drawn from distribution $\mathcal{N}(0, 1/m_t)$, this implies that $\Phi_t(\mathbf{v}/\|\mathbf{v}\|)\sqrt{m_t} \sim \mathcal{N}(0, I_{m_t})$. Hence the norm $\|\Phi_t(\mathbf{v}/\|\mathbf{v}\|)\sqrt{m_t}\|$ is Chi-distributed with m_t degrees of freedom. The expectation of a Chi-distributed random variable is

$$\mathbb{E}_{\Phi_t}\left[\frac{\|\Phi_t\mathbf{v}\|\sqrt{m_t}}{\|\mathbf{v}\|}\right] = \frac{\sqrt{2}\Gamma((m_t+1)/2)}{\Gamma(m_t/2)}.$$

With $C_{m_t} = \frac{\sqrt{2}\Gamma((m_t+1)/2)}{\Gamma(m_t/2)\sqrt{m_t}}$ we get from equation (7.27) that

$$\|\nabla L_t\| \mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] = C_{m_t} c \|\mathbf{v}\|^2. \tag{7.28}$$

The second special case of interest is when ∇L_t is perpendicular to $\mathbf{v}$. Note that this condition implies that $\Phi_t \nabla L_t$ is independent to $\Phi_t \mathbf{v}$. Indeed, if we consider their covariance:

$$\mathrm{cov}_{\Phi_t}(\Phi_t \mathbf{v}, \Phi_t \nabla L_t) = \mathbb{E}_{\Phi_t}[\langle \Phi_t \mathbf{v}, \Phi_t \nabla L_t\rangle] = \langle \mathbf{v}, \nabla L_t\rangle, \tag{7.29}$$

which equals to zero when $\mathbf{v}$ is perpendicular to ∇L_t. Hence we have

$$\|\nabla L_t\| \mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] = \|\nabla L_t\| \langle \mathbb{E}_{\Phi_t}\left[\Phi_t \mathbf{v}\right], \mathbb{E}_{\Phi_t}\left[\Phi_t \nabla L_t\right]\rangle = 0. \tag{7.30}$$

Now for any vector $\mathbf{v}$, we can write $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$ where $\mathbf{v}_1$ is a vector perpendicular to ∇L_t and $\mathbf{v}_2$ is a scalar multiple of ∇L_t. Hence we have

$$\begin{aligned}
\mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] &= \mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}_1, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] + \mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}_2, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] \\
&= \mathbb{E}_{\Phi_t}\left[\left\langle \Phi_t \mathbf{v}_2, \frac{\Phi_t \nabla L_t}{\|\Phi_t \nabla L_t\|}\right\rangle\right] = \left\langle \mathbf{v}_2, \frac{\nabla L_t}{\|\nabla L_t\|}\right\rangle C_{m_t} \qquad (7.31) \\
&= \left\langle \mathbf{v}, \frac{\nabla L_t}{\|\nabla L_t\|}\right\rangle C_{m_t}, \qquad (7.32)
\end{aligned}$$

where (7.31) used that $\mathbf{v}_2$ is a scalar multiple of ∇L_t with scalar multiple $c = \pm 1$ being sufficient to consider (since otherwise we can divide and multiply with $\|\mathbf{v}_2\|$), and the last equality holds because $\mathbf{v}_1$ is perpendicular to ∇L_t. Multiply both sides of eq. (7.31) by $\|\nabla L_t\|$, we conclude for all $\mathbf{v} = \mathbf{w} - \mathbf{w}_t$ that

$$\mathbb{E}_{\Phi_t}[\langle \Phi_t(\mathbf{w} - \mathbf{w}_t), s_t \Phi_t \nabla L_t\rangle] = \langle (\mathbf{w} - \mathbf{w}_t), \nabla L_t\rangle \cdot C_{m_t}. \tag{7.33}$$

□

7.3.2 Proofs of DP-CompSGD with Minibatch

In the case where we only use a random subset of the whole sample set in each iterate, the sensitivity will increase due to a smaller sample set. However, we can apply the following result to strengthen our privacy guarantee:

Theorem 7.8 (Amplification by subsampling [8])**.** *Let $\mathcal{X}$ be a data domain and $M : \mathcal{X}^n \to \mathcal{X}^b$ be a procedure such that $M(S)$ returns a random subset of b records sampled uniformly without replacement from S. Let A be an (ϵ_p, δ_p)-DP algorithm. Then $A \circ S$ satisfies $(\epsilon'_p, (b/n)\delta_p)$-DP with $\epsilon'_p = \log(1 + (b/n)(e^{\epsilon_p} - 1))$.*

Theorem 7.9. *The output of DP-CompMiniBatch in Alg. 9 satisfies (ϵ_p, δ_p)-differential privacy.*

Proof. The proof follows the same idea as for DP-CompGD. Note that since we only use a random subset of S for each iteration (of size b), we can apply Thm. 7.8 to obtain a stronger privacy guarantee relative to the size of the subsample at each iteration. The privacy guarantee then follows similarly by the strong composition over T iterations. □

Proof of Theorem 7.4. We start with the same procedure as in the derivation for the optimization of DP-CompGD in equation (7.4). Note that we can replace the batch gradient ∇L_S with the minibatch gradient ∇L_{B_t} without affecting the derivation of the inequality

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}\|^2 \leq \|\mathbf{w}_t - \mathbf{w}\|^2 + 2C_{m_t}\eta_t(L_{B_t}(\mathbf{w}) - L_{B_t}(\mathbf{w}_t)) + \eta_t^2(L^2 + m_t\sigma^2).$$

Rearranging the inequality and letting $\mathbf{w} = \mathbf{w}^*$ we have:

$$2C_{m_t}\eta_t(L_{B_t}(\mathbf{w}_t) - L_{B_t}(\mathbf{w}^*)) \leq \|\mathbf{w}_t - \mathbf{w}^*\|^2 - (1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}^*\|^2 + \eta_t^2(L^2 + m_t\sigma^2). \quad (7.34)$$

Since B_t is a random subset drawn uniformly from S, we have $\mathbb{E}_A[L_{B_t}(\mathbf{w})] = L_S(\mathbf{w})$.

Hence by taking expectation and summing over T iterations we have

$$2C_{m_t}\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)] \le \|\mathbf{w}_1-\mathbf{w}^*\|^2+\sum_{t=1}^{T}\beta_t\mathbb{E}[\|\mathbf{w}_{t+1}-\mathbf{w}^*\|^2]+\sum_{t=1}^{T}\eta_t^2(L^2+m_t\sigma^2).$$

Choosing $\beta_t = 1/(t+1)$ we obtain that

$$\frac{\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)]}{\sum_{t=1}^{T}\eta_t} = \mathcal{O}\left(\frac{\|\mathcal{C}\|^2+\log T\|\mathcal{C}\|^2+\sum_{t=1}^{T}\eta_t^2(L^2+m_t\sigma^2)}{\sum_{t=1}^{T}\eta_t}\right). \quad (7.35)$$

Finally, note that $m_T = \max_{t\in[T]} m_t$.

Letting $\eta_t = \|\mathcal{C}\|/\sqrt{t(L^2+m_T\sigma^2)}$ and $\sigma^2 = \mathcal{O}(TL^2\log(1/\delta_p)\log(4Tb/(\delta_p n)/(\epsilon_p^2 n^2))$ we have

$$\begin{aligned}\frac{\sum_{t=1}^{T}\eta_t\mathbb{E}[L_S(\mathbf{w}_t)-L_S(\mathbf{w}^*)]}{\sum_{t=1}^{T}\eta_t} &= \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|\sqrt{L^2+m_T\sigma^2}}{\sqrt{T}}\right)\\ &\le \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|L}{\sqrt{T}}+\frac{\log T\|\mathcal{C}\|L\sqrt{m_T T\log(1/\delta_p)\log(4Tb/(\delta_p n)}}{n\epsilon_p\sqrt{T}}\right)\\ &= \mathcal{O}\left(\frac{\log T\|\mathcal{C}\|L}{\sqrt{T}}+\frac{\log T\|\mathcal{C}\|L\sqrt{m_T\log(1/\delta_p)\log(4Tb/(\delta_p n)}}{n\epsilon_p}\right).\end{aligned}$$

□

Proof of Theorem 7.5. Let S and S' be two neighbouring sample sets of size n that differ in one single sample. Let $G(\mathbf{w}_t) = \mathbf{w}_{t+1}$ denote the gradient update and let $G_1, \dots, G_T$ and $G'_1, \dots, G'_T$ be the updates induced by running the compressed SGD on S and S' for T iterates, respectively. Let $\delta_T = \|\mathbf{w}_T - \mathbf{w}'_T\|$, by the Lipschitz condition we have

$$\mathbb{E}[|\ell(\mathbf{w}_T, z) - \ell(\mathbf{w}'_T, z)|] \le L\mathbb{E}[\delta_T]. \quad (7.36)$$

If at iteration t, the mini-batch B_t, B'_t we selected is the same, i.e. $G_t = G'_t$, then by the same derivation for equation (7.15) we have the following bound

$$(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}'_{t+1}\|^2 \le \|\mathbf{w}_t-\mathbf{w}'_t\|^2 - 2\eta_t\langle\nabla L_{B_t}(\mathbf{w}_t)-\nabla L_{B_t}(\mathbf{w}'_t), \mathbf{w}_t-\mathbf{w}'_t\rangle + 4\eta_t^2L^2.$$

From the convexity of ℓ we have that $\langle \nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B_t}(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t \rangle \geq 0$. Hence we obtain the following bound

$$(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \leq \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + 4L^2\eta_t^2. \tag{7.37}$$

For the case where $G_t \neq G'_t$, we use the fact that B_t and B'_t differ by a single sample. Hence we have

$$\begin{aligned}
&(1-\beta_t)\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 \\
&\qquad \leq \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - 2\eta_t\langle \nabla L_{B_t}(\mathbf{w}_t) - \nabla L_{B'_t}(\mathbf{w}'_t), \mathbf{w}_t - \mathbf{w}'_t\rangle + 4\eta_t^2 L^2 \\
&\qquad \leq \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + \frac{4L\eta_t}{b}\|\mathbf{w}_t - \mathbf{w}'_t\| + 4L^2\eta_t^2, \qquad (7.38)
\end{aligned}$$

where the last inequality (7.38) follows by the same derivation as for equation (6.49), replacing S, S' with B_t, B'_t respectively. Combining the two cases we have

$$\begin{aligned}
(1-\beta_t)\mathbb{E}[\delta_{t+1}^2] &\leq \left(1 - \frac{b}{n}\right)\left(\mathbb{E}[\delta_t^2] + 4L^2\eta_t^2\right) + \frac{b}{n}\left(\mathbb{E}[\delta_t^2] + \frac{4L\eta_t\mathbb{E}[\delta_t]}{b} + 4L^2\eta_t^2\right) \\
&= \mathbb{E}[\delta_t^2] + 4L\eta_t\left(\frac{\mathbb{E}[\delta_t]}{n} + L\eta_t\right). \qquad (7.39)
\end{aligned}$$

The rest of the stability proof then follows by the same procedure as the proof for Thm. 7.3 starting from equation (6.50). □

7.4 Empirical illustration

To illustrate the theory for compressed SGD, particularly in the private setting, and to show how the dimensionality of the noise affects the error, we generated some example data sets with a low dimensional structure. We run the private gradient descent (without projection) and the private CompGD algorithm on these data sets with a fixed size $N = 3000$ and dimension ranging from $d = 30$ to $d = 1000$. The data sets are generated randomly in the unit sphere at high dimensions, labels are then assigned to each sample using a low dimensional weight vector $\mathbf{w}$ as follows: For each sample z we compute the sigmoid function value with z and $\mathbf{w}$ and assign a label $\in \{+1, -1\}$ depending on their sigmoid value. We then run the algorithm on this data set with the optimizing function ℓ being the logistic loss function. After running on epochs ranging from 1 to 10 we then compute the test error using an independent test set of size $N = 100$. For the private CompGD, we will perform the projection of the gradient using a randomly generated Gaussian matrix and project it onto a $m = d/10$ dimension. Each experiment is repeated 10 times and the average is calculated and plotted. The error bars in the plots indicate one standard deviation from the mean. To maintain consistency between the two algorithms so that they only differ by the projection step, we fixed the privacy parameters to the standard setting $\epsilon_p = 2$ and $\delta_p = 0.01$ for both algorithms. With this restricted setting, the experiment result may not be state-of-the-art. However, our goal is to illustrate the effect of the projection step in private compressed gradient descent implied by our risk bounds in Section 7.1.

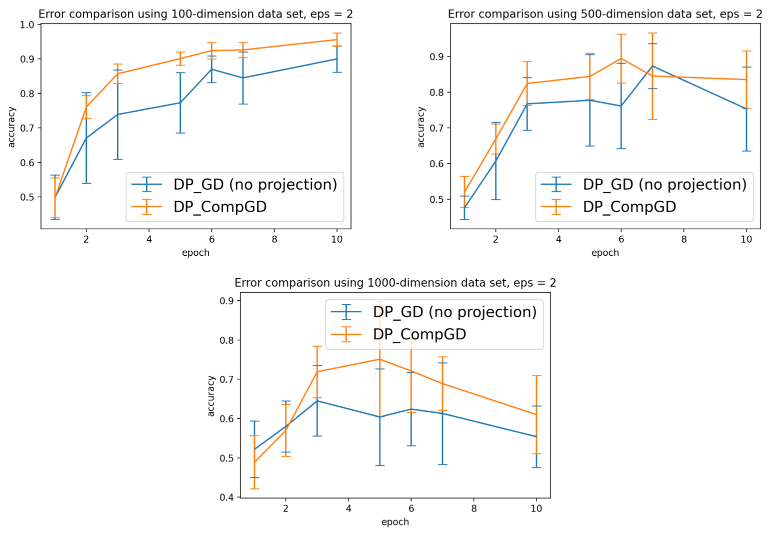

Figure 7.1: Comparison of DP gradient descent with and without projection using $d = 100, 500, 1000$ (left to right) respectively. The dimension of the labelling weight vector $\mathbf{w}$ is 1. We can observe from the plot that despite the simple structure of the labelling vector, the problem is difficult at very high dimensions due to the excessive amount of injected noise. We also observe a significant improvement of DP-CompGD over DP-GD over all three sets of experiments.

We note from the experiments that for higher dimensions (≥ 500), the accuracy of both algorithms starts to decay after ≈ 6 epochs. This is expected from the theory because the variance of the injected noise is increasing as the number of iterations increases and

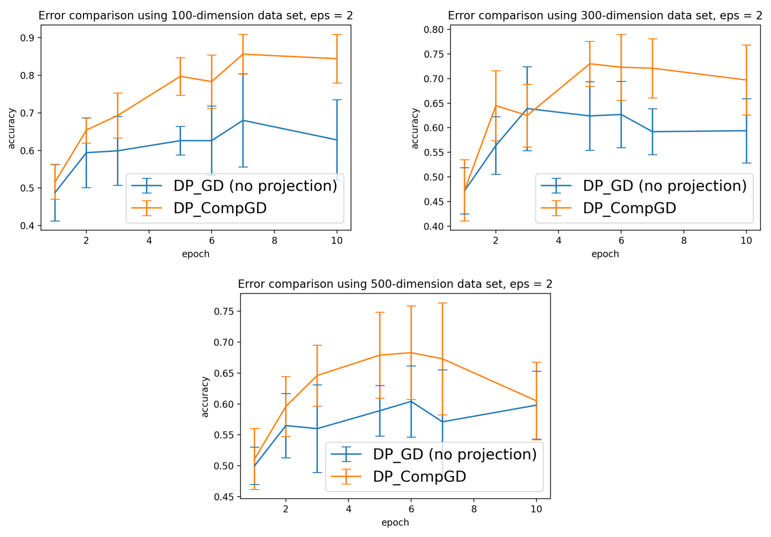

Figure 7.2: Comparison of DP gradient descent with and without projection using $d = 100, 300, 500$ (left to right) respectively. The dimension of the labelling weight vector $\mathbf{w}$ is 3. We observe that the average error for this set of experiments is higher compared to Fig. 7.1, this is due to a more complex problem generated using a 3-dimensional labelling vector. Similar to Fig. 7.1 we observe that DP-CompGD has a significant improvement in accuracy compared with DP-GD without projections.

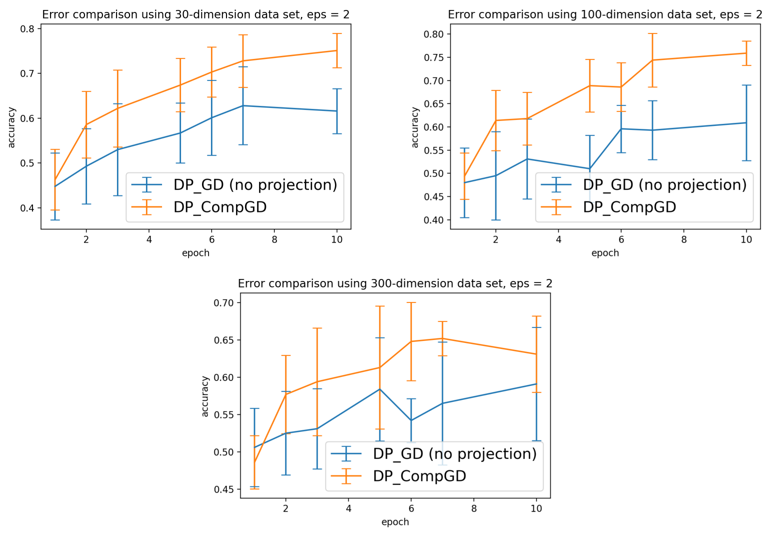

Figure 7.3: Comparison of DP gradient descent with and without projection using $d = 30, 100, 300$ (left to right) respectively. The dimension of the labelling weight vector $\mathbf{w}$ is 10. We start at dimension $d = 30$ here since this is a more complex problem and we do not have sufficient sample points to learn effectively. Although the average accuracy of both algorithms is relatively lower than Fig. 7.1 and 7.2, we can still observe the improvement of accuracy of DP-CompGD over DP-GD similarly.

we have fixed the number of training samples. Hence as described in Alg. 8, the variance σ^2 increases as T increases. This effect is more noticeable in higher dimensions because the amount of noise injected is greater for higher dimensions. Hence, for private SGD algorithms, it is preferred to run fewer epochs and have a large training sample to minimize the effect of injected noise. Since we can not run an excessive number of iterations in the private setting, it is very advantageous to reach a good accuracy level with fewer iterations. The DP-CompGD has this advantage over DP-GD due to the reduction of noise.

The results from our experiments meet the expectation from our theory, where the reduction of dimensionality of the noise reduces the 'privacy term' in its optimization bound, hence improving the accuracy of the outputted model.

Summary

We have introduced two different algorithms of gradient descent with compressed gradients in the DP setting. Both algorithms provides the same level of privacy guarantee but the CompGD has a smaller variance due to the increased number of samples in each iteration. Furthermore, since full batch gradient is used for the CompGD, less iterations are required to converge which will also reduce the variance of the added noise.

We have developed the first optimization and generalization error bounds for both algorithms. In particular, the bound showed that CompGD benefits for convergence with fewer iteration in the case of smooth and convex loss functions, although their generalization convergence are the same. Furthermore, we found that by using the compressed gradients instead of full gradient, we have reduced the dimensionality dependence of the added noise, hence resulting a reduced dimensionality dependence in the generalization bounds of both algorithms. In the case where privacy guarantee vanishes ($\epsilon_p \to \infty$), the error rate convergence recovers to the non-private bound we obtained in Chapter 6. A natural question is whether we can perform a similar analysis and use the compressed gradient to reduce the noise for other variants of differentially private SGD. Extending the analysis to other private SGD algorithms to investigate the improvement of noise level will be an interesting open research problem. Finally, we have demonstrated empirically the difference of accuracy performance between gradient descent with projected gradients and vanilla GD. We observed that CompGD has improved accuracy performance over vanilla GD in the private setting while reducing the dimension of the gradient.

Chapter Eight

Noise-efficient Learning of Private Ensembles

8.1 Preliminaries

While DP is a useful privacy guarantee for learning algorithms, we often require a larger training set to achieve good performance due to the added noise. However, large labelled training set is expensive and sometimes impractical to obtain in reality, especially for a privacy-concerning data set (e.g. HIV positive data). On the other hand, unlabelled data can be less privacy-sensitive in many cases and much more cost-effective to obtain in a large scale comparing to accurately labelled data. Due to the fore-mentioned reasons, it is extremely valuable if we can extract information from unlabelled data and improve the performance of the learning algorithm in the private setting.

In this Chapter, we present a framework that builds machine ensembles in both supervised and semi-supervised setting. The framework takes advantage of unlabelled data to reduce the noise requirement and hence it only requires a small number of labelled samples to achieve high performance. Our framework invokes constructing a tree structure that uses a density-informed splitting criterion to create balanced leaves and naturally extends to semi-supervised learning with different privacy settings. Current private tree-based algorithms in the literature either use a greedy-decision approach, or a random-tree approach.

Methods with greedy approaches take the route of classical decision tree construction [27], and compute the optimal splits at each node privately. Popular splitting techniques such as the Gini index [139, 54] or the information gain criterion [58] are applied in conjunction with a privatized algorithm. The drawback for this approach is that it cannot be naturally extended to semi-supervised learning as they greedily estimate the optimal split using the labels. On the other hand, random tree approaches construct the tree by randomized splits at each node [71, 56, 59]. Randomization is beneficial from the privacy perspective as it is data-independent and leaks zero information about individuals in the data set. However, a fully randomized split creates high variance and requires a large ensemble of trees to perform well. We cannot afford a large ensemble due to the privacy constraint. Furthermore, random-tree approaches do not take advantage of the unlabelled data as the splits are chosen fully randomly. Since these approaches do not naturally extend to the semi-supervised setting, we need to assign labels to the unlabelled set using a trained model if we want to make any use of the unlabelled data [70, 93]. While this method can help to improve accuracy in some cases, it requires the predicted labels to be accurate for the outputting data to be useful, which can not be guaranteed in general. Furthermore, since the output data contains the original features of the unlabelled set, it can only be applied where we do not need the privacy of the features at all.

Instead of the previous approaches, our approach uses a semi-greedy median splitting criterion that uses the features to make formative splits. Median splitting has been used to build trees mostly in spatial decomposition where we partition data sets to allow quick access to different parts of the data [13, 39]. However it also can be used for classification and regression problems with good utility as shown in [25, 81] - even though classical decision tree methods are better in general without privacy. A main intuition of median split is that it creates density-balanced nodes, the concept matches with the density-based dissimilarities in the work of Aryal *et al* [4] – that states two points are more similar if they lie in a sparse region than two other points in a dense region with equal geometric distance. Each leaf comes with a similar amount of sample points hence we avoid empty leaves and the noise level of each leaf is balanced. To achieve high utility, we also employ several techniques to optimize each step of the privatized model as follows. 1) We use a geometric-scaling privacy budget allocation strategy to ensure accurate splits at each level. 2) We use a random sampling technique to compute the private median effectively. 3) We use disjoint subsets to create ensembles for both labelled and unlabelled sets to reduce noise effects. While these techniques pre-existed in isolation as parts of other algorithms, the combination appears novel and it leads to a novel framework that achieves high performance in both supervised and semi-supervised setting.

8.2 A density-informed private ensemble

In this section, we describe the procedures and the key steps of our tree construction. Overall the algorithm is broken down into the following steps:

1. Decide the parameters. (number of trees and maximum depth)
2. At each tree node, uniformly randomly select an attribute to split on.
3. Calculate a private median for the selected attribute using the exponential mechanism.
4. When reaching a leaf node, use the Laplace mechanism to store the privatized counts for each class.
5. For a test point, summarise the counts obtained for each tree and output the class of the label majority.

Note that at each split we randomly select an attribute over the entire range, this avoids the greedy computation of optimal attribute using the labels as in [38, 139] and allows us to construct the tree using only an unlabelled sample later on. Furthermore, random selection of the features can improve the diversity of each tree while protecting privacy. After selection of a splitting attribute, we compute privately the median of the values at the selected attribute. This splitting method allows the feature space to be partitioned into even density regions. A key property of median splits is that it only depends on the features of the data, not the labels. Hence it does not overfit the data set easily, which may concern classic decision trees. Due to this property, we do not require any pruning process that many trees require to avoid over-fitting. Other similar techniques involve the centred random forest described in [81] and replacement of the median with the mean. However in the private setting, median is usually less sensitive to outliers compared to the mean and less noise is required to guarantee privacy.

Algorithm 10 BuildTree

```
1: Inputs: Sample set $S$, maximum depth $k$, privacy budget $\epsilon_p$.
2: procedure BuildTree($S, k, \epsilon_p$)
3:     if k $\leq 0$ or $|S| \leq 10$ then
4:         Return a Leaf node
5:     end if
6:     LB, RB $=$ $PrivateSplit(S, \epsilon_p)$
7:     BuildTree(LB, $k-1$, $\epsilon_p$), BuildTree(RB, $k-1$, $\epsilon_p$) # build subtrees
8:     Return a decision node that holds the split criteria and left/right branch.
9: end procedure
10: procedure PrivateMedian($V, \epsilon_p$)
11:     $a = \min V, b = \max V$
12:     $\mathcal{R} = \{$ set of random i.i.d. draws from $Uniform(a, b)\}$;
13:     for each $r$ in $\mathcal{R}$ do
14:         computes the quality score $u(V, r)$
15:     end for
16:     Return $\tilde{r} \in \mathcal{R}$ with exponential mechanism with budget $\epsilon_p$.
17: end procedure
18: procedure PrivateSplit($S, \epsilon_p$)
19:     Choose a splitting dimension $i$ uniformly from data dimension $[d]$
20:     $V$ = sorted{set of values in dimension $i$}
21:     Choose private median $p$ by $PrivateMedian(V, \epsilon_p)$
22:     for each sample $X$ in $S$ do
23:         if $X_i \leq p$ then
24:             add to left branch
25:         else
26:             add to right branch
27:         end if
28:     end for
29:     Return LeftBranch, RightBranch
30: end procedure
```

As one of the most crucial steps in our private tree construction, being able to compute the median accurately while preserving privacy is crucial for the final performance of the tree. A demonstration of the effect of the median estimation on accuracy is shown in Fig. 8.2.

There are different methods of private median computation as discussed in [39], the most common approach may be using the exponential mechanism. However, direct application of the exponential mechanism considers many redundant potential values that are far away from the median. Furthermore, these values can be assigned with high probability due to their large share of the marginal distribution. We instead randomly sample a finite number of values from a uniform distribution over the possible range of values. In this way we significantly reduce the complexity of the problem by only considering a fixed number of possible outputs, hence improving the computational efficiency and resulting in a more accurate estimate.

We employ the exponential mechanism with a utility measure function that uses the

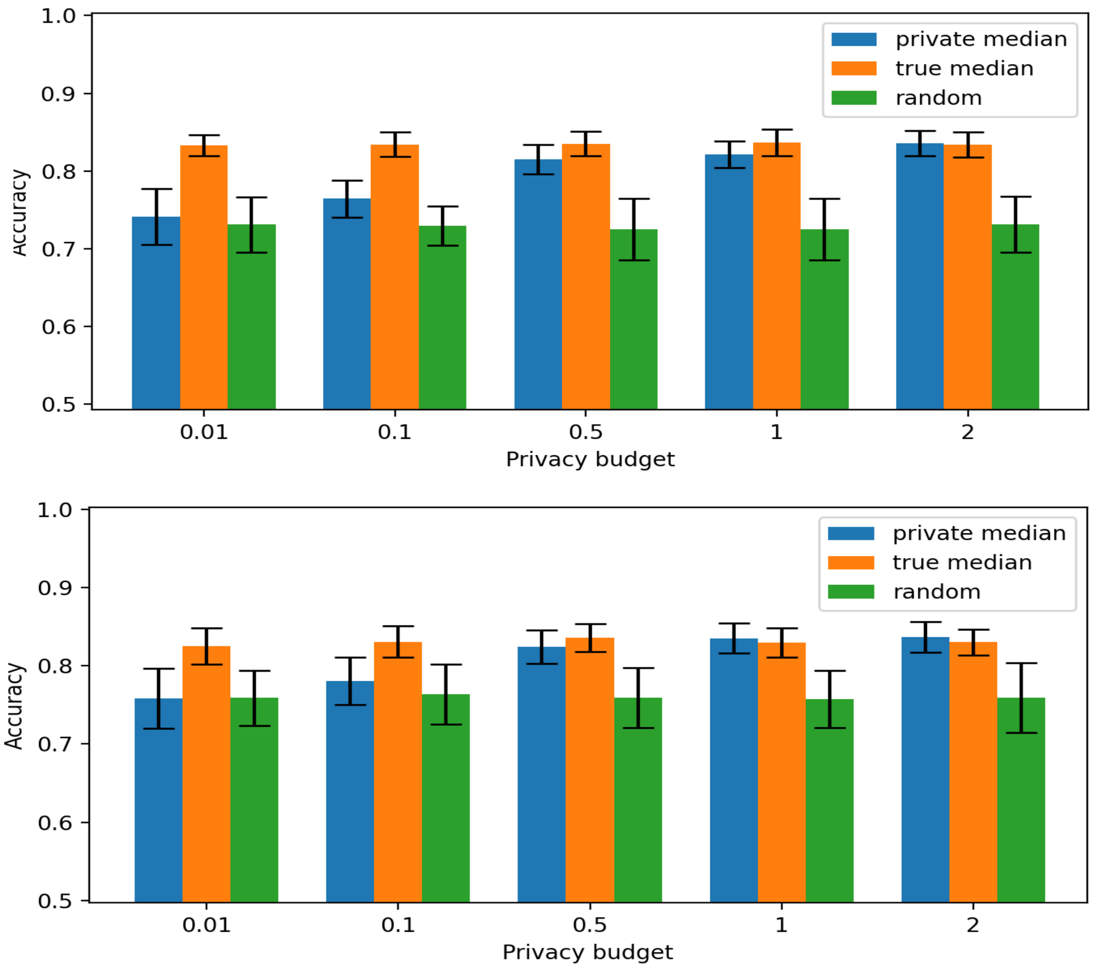

Figure 8.1: Effect of median estimation on accuracy on synthetic data sets of dimension 5 (top) and 10 (bottom). We compare the accuracy of Alg. 10 with its variation that uses the true median and random splitting. We observe that accuracy increases as we make better estimates of the true median due to larger privacy budgets.

rank of a potential output $r \in \mathcal{R}$: Let ϵ_s denote the desired privacy parameter for tree construction. Denote a subset of the data as $S_i \subset S$ and let V denote the set of the j-th values for points in S_i. Then the utility of a potential output $r \in \mathcal{R}$ over a randomly selected attribute j is defined as

$$u(V, r) = -|rank_j(r) - |V|/2|, \tag{8.1}$$

where $rank_j(r)$ denotes the number of points in V that is $\leq r$.
This utility function assigns negative weights (quality score) to all values except for the true median, which will have weight zero. Values $r \in \mathcal{R}$ will have decreasing utilities the further away they are from the median. For categorical variables we use the same utility function, except that we let $\mathcal{R}$ to range over all categories for attribute j, and we define $rank_j(r)$ to be the number of points in V that is $= r$. Note that the sensitivity of u is 1/2: adding a data point in V increases $|V|/2$ by 1/2 and $rank_j(r)$ either increases by 1 or remains the same; removing a data point has a similar effect. Now by the exponential mechanism, for any given ϵ_p we can guarantee ϵ_p-DP by outputting $r \in \mathcal{R}$ with probability

$$\mathbb{P}[\mathcal{M}(V, u, \mathcal{R}) = r] \propto \exp\left(\epsilon_p u(V, r)\right), \tag{8.2}$$

where the actual probability will be obtained through dividing the sum of proportional probabilities over all $r \in \mathcal{R}$.

We note that we have not query the sample labels in our construction of the tree, by partitioning the sample set into N disjoint sets and distributing the labels to the leaves of the tree (privately), we obtain a private supervised ensemble model for classification and regression tasks. The algorithm is outlined in Alg. 11.

8.2.1 Privacy analysis

There are two steps in Alg. 10 and Alg. 11 where we have used a privacy mechanism: (i) *PrivateSplit*, and (ii) *DistributeLabels*. In this section, we analyze the privacy guarantee

Algorithm 11 Supervised Private Ensemble

```
1: Inputs: labelled set $S$, size of ensemble $N$, maximum depth $k$, privacy budget $\epsilon_p$
2: procedure SupervisedEnsemble($S, N, k, \epsilon_p$)
3:     Split the total budget $\epsilon_p$ into $\epsilon_s, \epsilon_l$. # even split by default
4:     Partition $S$ into $N$ disjoint subsets $\{S_i\}_{i=1}^N$.
5:     for $i$ in $1, \ldots, N$ do
6:         Tree $i$ = $BuildTree(S_i, k, \epsilon_s)$
7:         DistributeLabels(Tree $i$, $S_i$, $\epsilon_l$) and add Tree $i$ to ensemble
8:     end for
9: end procedure
10: procedure DistributeLabels(Tree, $S, \epsilon_p$)
11:     for each sample $X$ in $S$ do
12:         find the corresponding leaf of $X$ and record the label of $X$
13:     end for
14:     for each leaf in the Tree do
15:         for each label class do
16:             add $noise \sim Lap(1/\epsilon_p)$ to the label count.
17:         end for
18:     end for
19: end procedure
```

of each mechanism. For the *PrivateSplit* procedure we note that the only computation required to query the data set is private median estimation. The splitting process afterwards only partitions the data set by the split condition, which guarantees the same privacy by the post-processing property of DP [48]. To guarantee ϵ_s-DP over the whole sequence of splits along the tree construction, we need to split ϵ_s into a sequence of privacy budgets

$$\epsilon_0 + \epsilon_1 + \cdots + \epsilon_{k-1} = \sum_{i=0}^{k-1} \epsilon_i = \epsilon_s, \tag{8.3}$$

where ϵ_i is the privacy budget for splits at depth i, and k is the maximum depth. A node with depth k corresponds to a leaf and hence no split is required. For splits at the same depth (say depth i), we can assign ϵ_i budget to each split because the data set held at the nodes of the same depth are disjoint. Therefore we have ϵ_i-DP guaranteed simultaneously by the parallel composition theorem. Furthermore, by the sequential composition theorem we sum all splits at different depths and together and obtain $(\epsilon_0 + \cdots + \epsilon_{k-1}) = (\epsilon_s)$-DP. Now for *DistributeLabels*, we will use the Laplace mechanism to output a private count of the classes while guaranteeing ϵ_l-DP, where ϵ_l is the desired privacy parameter for leaf construction. We note that the sensitivity of the class count is 1 as adding or removing a point changes the count by at most 1. Hence by the Laplace mechanism, it suffices to add random noise drawn from $Lap(1/\epsilon_l)$ to achieve ϵ_l-DP. Moreover, since the data set in each leaf is disjoint from each other, by guaranteeing ϵ_l-DP for each leaf we can obtain ϵ_l-DP for all leaves simultaneously by parallel composition. Overall, we have shown that for any given ϵ_s and ϵ_l, the ensemble construction in Alg. 11 achieves $(\epsilon_s + \epsilon_l)$-DP.

8.2.2 Privacy budget allocation

In this section we discuss our strategy of privacy budget allocation for the construction of ensembles in Alg. 11. For a total privacy budget ϵ_p, since we have partitioned the sample set S into N disjoint subsets $\{S_i\}_{i=1}^N$, we can allocate the whole privacy budget ϵ_p to every tree by the parallel composition theorem. We further split ϵ_p to $\epsilon_p = \epsilon_s + \epsilon_l$ for privacy budget used in node splits and label predictions, respectively. We will use an equal share between the two as default since both procedures are important to its final performance. i.e. $\epsilon_s = \epsilon_l = \epsilon/2$. We now discuss the budget allocation of ϵ_s along the nodes as follows. As a general intuition, the optimal budget allocation should depend on the difficulty of performing the private median computation. Based on this idea, we found that the privacy budget allocation follows a geometrically-scaling sequence along the depths of the nodes. Let $r, r' \in \mathcal{R}$ be two any potential outputs drawn uniformly from $[a, b], a, b \in \mathcal{R}$. The private estimation of the median is easier if we can distinguish the utility of r, r' and output the better option with higher probability. This means we want the utility difference $|u(r) - u(r')|$ to be large which is calculated by the number of values between r and r'. We observe that the expected difference between two randomly chosen points from a uniform distribution is equal to $|a - b|/3$ (we cannot make any assumption on the values of the input samples). This observation implies that for every point added or removed, the probability that $|u(r) - u(r')|$ will change due to the added/removed point being equal to $1/3$. Since any parent node is expected to receive 2 times the number of samples compared to its child nodes, the median estimation problem will be $2 \times (1/3)$ times in difficulty compared to its child node. Hence for any node at depth i that received ϵ_i budget, we assign $\epsilon_{i+1} = (3/2)\epsilon_i$ privacy budget to its child nodes at depth $i+1$. Furthermore, we must full-fill the condition that the sum of privacy budgets over all depths is equal to ϵ_s. Hence we scale each ϵ_i by a constant C so that we have $\sum_{i=0}^{k-1} \epsilon_i = \epsilon_s$, W.l.o.g. we assume $\epsilon_0 = C\epsilon_s$.

$$\epsilon_s = \sum_{i=0}^{k-1} \epsilon_i = \epsilon_0 + (3/2)\epsilon_0 + \cdots + (3/2)^{k-1}\epsilon_0 \tag{8.4}$$

$$= C\epsilon_s(1 + (3/2) + \cdots + (3/2)^{k-1}) = C\epsilon_s \left(\frac{(3/2)^k - 1}{(3/2) - 1}\right), \tag{8.5}$$

which implies

$$\epsilon_i = C\epsilon_s \left(\frac{3}{2}\right)^i \text{ , where } \quad C = \frac{1}{2(3/2)^k - 2}. \tag{8.6}$$

To illustrate the effectiveness of our privacy budget allocation strategy, we run simulations of Alg. 10 to analyse the quality of the estimated median by comparing our allocation strategy with uniform allocation which is the baseline method. We run our experiments on a synthetic data set generated from a normal distribution and we kept all other parameters identical except the allocation strategy.

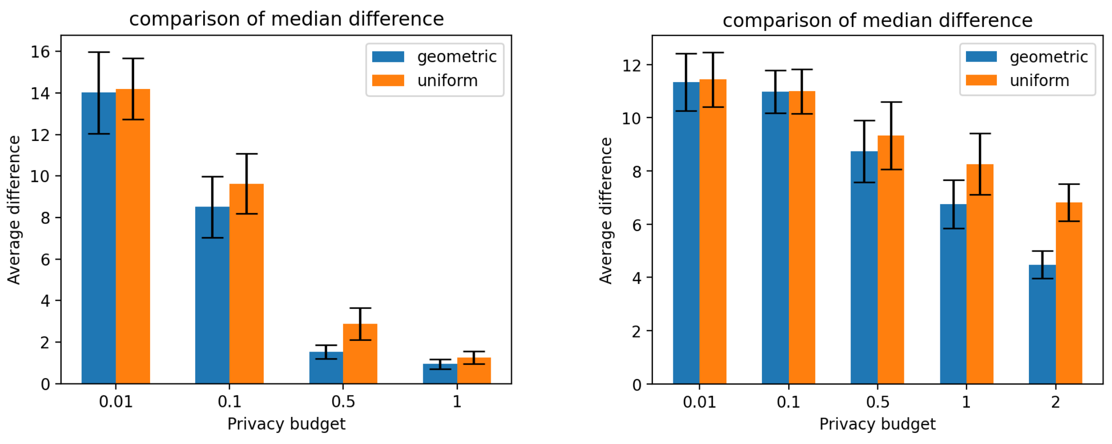

Figure 8.2: Comparison between geometric allocation and uniform allocation at different privacy levels with maximum depth 5 (left plot) and 10 (right plot). Error bars indicates (± 1) standard deviation from the mean and each experiment is repeated 50 times. We observe that our strategy achieves smaller average distance to the true median over all set of experiments. This shows that the proposed allocation strategy has a significant effect on the median estimation while guaranteeing the same level of privacy.

8.3 Differentially Private Semi-Supervised Ensembles

In this section we present our framework of private semi-supervised learning that creates private ensembles in the following private settings: (I) both privacy of the features and labels are required; (II) only privacy for labels is required. For the first case, existing work can only train on a labelled set and could not take advantage of a separate unlabelled set. In contrast, the method of our framework can build the tree using the unlabelled set only, as a result we can assign all privacy budget to the label predictions and reduce the noise. Moreover, we significantly reduce the number of labelled samples needed while achieving a good accuracy level. We present the procedures of the private ensembles in Alg. 12 and 13, Alg. 12 applies to both private settings (I) and (II) where Alg. 13 is applicable only in setting (II). For setting (II) in Alg. 12 we use ∞-privacy to build trees meaning that we can compute the true median for each split, and no partitioning of the unlabelled set is required.

Algorithm 12 Semi-Supervised Private Ensemble

```
 1: Inputs: labelled set $S$, unlabelled set $D$, maximum depth $k$, size of ensemble $N$,
    privacy budget $\epsilon_p$
 2: procedure SS-Ensemble($S, D, k, N, \epsilon_p$)
 3:     if need privacy for features and labels then
 4:         Partition $S, D$ into $N$ disjoint portions: $\{S_i\}_i, \{D_i\}_i$
 5:         for $i$ in range $1, \dots, N$ do
 6:             Tree $i$ = BuildTree($D_i$, $k$, $\epsilon_p$)
 7:             $DistributeLabels(Tree\ i, S_i, \epsilon_p)$ and add Tree $i$ to ensemble
 8:         end for
 9:     else #privacy for labels only
10:         $S' = S$ with labels removed
11:         Partition $S$ into $N$ disjoint portions: $\{S_i\}_i$
12:         for $i$ in range $1, \dots, N$ do
13:             Tree $i$ = BuildTree($S' \cup D$, $k$, $\infty$)
14:             $DistributeLabels(Tree\ i, S_i, \epsilon_p)$ and add Tree $i$ to ensemble
15:         end for
16:     end if
17:     Return ensemble
18: end procedure
```

In setting (II) we can perform computations with the features as many times as needed and release the output without privacy concern. A transductive approach can be applied in this case to take further advantage of the unlabelled data [93]. The transductive approach trains a small ensemble using a labelled set and then predicts labels for each sample in the unlabelled set using the trained ensemble. The newly-labelled set can then be used to train a larger ensemble. Our framework also takes advantage of this approach in the privacy for label-only setting. A draw-back of this technique is that the newly-labelled set can have noisy labels due to inaccurate predictions which can decrease the accuracy of the final model.

Algorithm 13 Private Transductive Ensemble

```
1: Inputs: labelled set $S$, unlabelled set $D$, maximum depth $k$, size of first ensemble $N1$,
   size of second ensemble $N2$, privacy budget $\epsilon_p$
2: procedure TransductiveEnsemble($S, D, k, N1, N2, \epsilon_p$)
3:     Partition $S$ into $N1$ disjoint portions: $\{S_i\}_i$
4:     for $i$ in range $1, \ldots, N1$ do
5:         Tree $i$ = BuildTree($S' \cup D$, $k$, $\infty$)
6:         $DistributeLabels(Tree\ i, S_i, \epsilon_p)$ and add Tree $i$ to ensemble1
7:     end for
8:     Assign labels to samples in $D$ using ensemble1 and denote by $D_l$
9:     for $i$ in range $1, \ldots, N2$ do
10:        Tree $i$ = BuildTree($S' \cup D$, $k$, $\infty$)
11:        $DistributeLabels(Tree\ i, D_l, \epsilon_p)$ and add Tree $i$ to ensemble2
12:    end for
13:    Return $ensemble1 \cup ensemble2$
14: end procedure
```

8.4 Experimental Analysis

To illustrate the performance of the proposed algorithm, we perform a series of experiments using synthetic data sets as well as real data sets from the UCI [7]. The synthetic data sets are generated by forming normally distributed clusters with random centers using the python package *sklearn.datasets.make_ classification*. We generate three synthetic data sets each with 3000 samples with $5, 10$ and 15 attributes, each data set contains 2 classes. The UCI data sets covers a wide range of real data sets with size range from 150 to 32561 and dimensions range from 4 to 33. We use 90% of the data for training and the 10% rest for testing across all of our experiments. Each data set is randomly shuffled before training. Each experiment is repeated on the same data set 50 times where an average and standard deviation of the prediction accuracy is calculated.

8.4.1 Scaling with the privacy parameter

In this section, we demonstrate that our approach is able to take advantage of additional unlabelled data. As discussed in section 8.3, being able to construct the tree without labelled data is a great advantage that classical decision tree algorithms lack. We examine the effect of using unlabelled data under different privacy budgets (from 0.1 to 2). We generated data sets using the tool *sklearn.datasets.make_ classification* that generates random clusters with a standard normal distribution, we generate data sets with $5, 10$ and 15 attributes and examine the performance on each. We fix the depth to equal the dimension of the data set and size of ensemble fixed to 20 to keep consistency. For all experiments we use labelled training sets of size from $100 - 5000$ and a fixed unlabelled set of size 10000. All experiments are repeated 20 times and we plot the average accuracy with error bars indicating one standard deviation from the mean. We note from Fig 8.3,

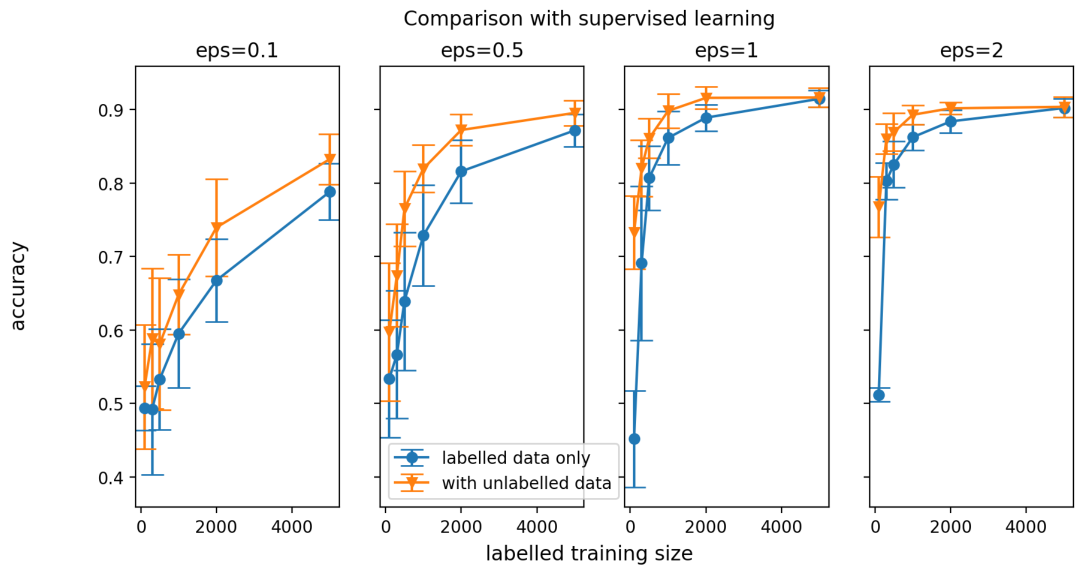

Figure 8.3: Comparison of private ensemble with and without unlabelled data using Synthetic 5 dimensional data set.

8.4, 8.5 that the aid of unlabelled data is most significant at lower privacy budgets. In view of labelled training sizes, the accuracy with unlabelled data is significantly better than using labelled data only when the labelled set is small. As we increase the labelled

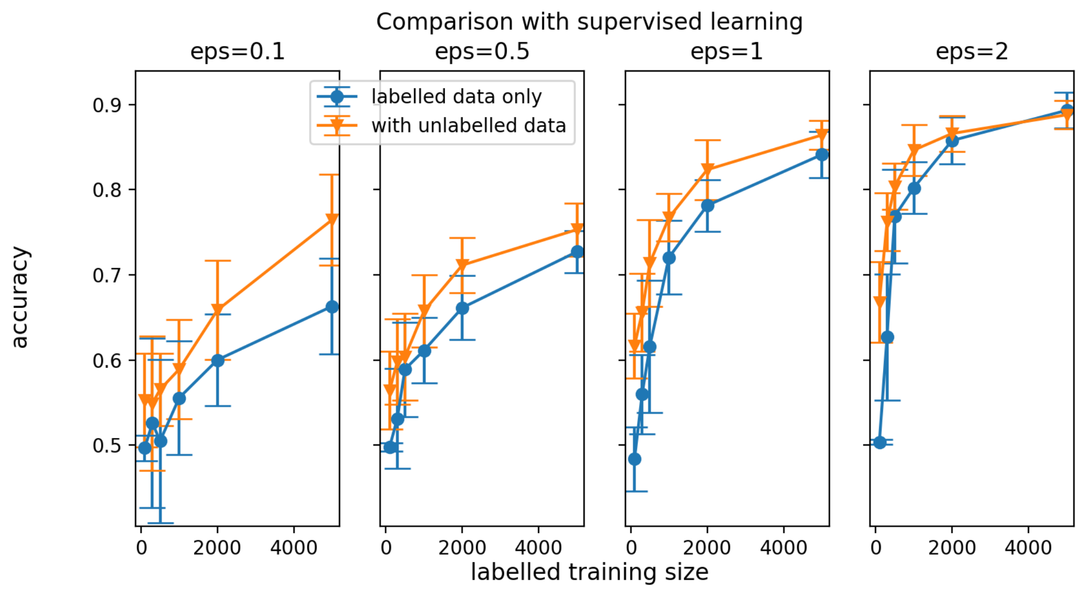

Figure 8.4: Comparison of private ensemble with and without unlabelled data using Synthetic 10 dimensional data set.

training size, the two algorithms perform more similarly until there is minimal difference between them (at $\epsilon_p = 2$ with 5000 labelled samples). This observation indeed matches with our intuition, as we can always do accurate computation with sufficient labelled samples. However in cases where we do not have sufficiently many labelled samples or if we have a restrictive small privacy budget, then addition of unlabelled samples can significantly boost the accuracy of the algorithm while guaranteeing the same amount of privacy.

8.4.2 Varying the parameters

We demonstrate the effect of parameters (N trees and max-depth) on the accuracy in Fig. 8.6 with three UCI data sets using the supervised ensemble (Alg. 11). We see the accuracy is high with small N and decreases as we add more trees into the forest, as we expected, since under privacy constraints each tree works on a disjoint subset of the data, leading to weak learning of the individual trees. From the plots we see that most of the accuracy curves decline after 10 trees, hence we have set $N = 10$ as our default. Furthermore, we see that the choice of max-depth$=d$ is a reasonable default as it reaches high accuracy across data sets. We also experimented with deeper ($2d$) trees (features to

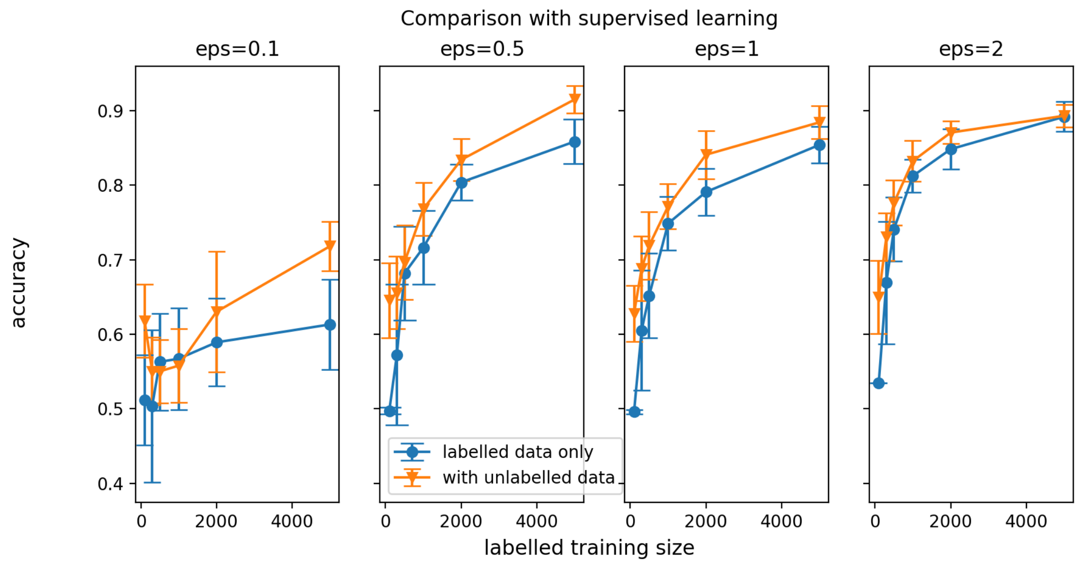

Figure 8.5: Comparison of private ensemble with and without unlabelled data using Synthetic 15 dimensional data set.

split on are sampled with replacement) and see in Fig. 1 that this may win in some cases. However, we must be cautious in general, as this increases the complexity of the function class and we run the risk of over-fitting as the leaf nodes become too small.

8.4.3 Comparison with other supervised algorithms

We analyse the prediction accuracy in the supervised setting (no unlabelled set) by comparing our Supervised Private Ensemble (SPE) with three state-of-the-art differentially private tree-based algorithms: Smooth random trees (SRT) by [56], Random Decision Trees (RDT) by [71] and a version of greedy decision trees (MaxTree) by [92]. In the experiments we build 10 trees with maximum depth d as a default value where d is the dimension. For competitors we used parameters recommended by the authors. For a non-private reference we use a random forest classifier with 100 trees as a benchmark for the best performance achievable on the particular data set without privacy constraints. We perform our experiments over a wide range of UCI data sets and three synthetic data sets, the size of each data set and the number of attributes is described in Table 8.1.

To further evaluate if the reported result is statistically significant, we perform a Mann-Whitney U test (Wilcoxon rank sum test) at a 95% confidence level. From Table 8.1

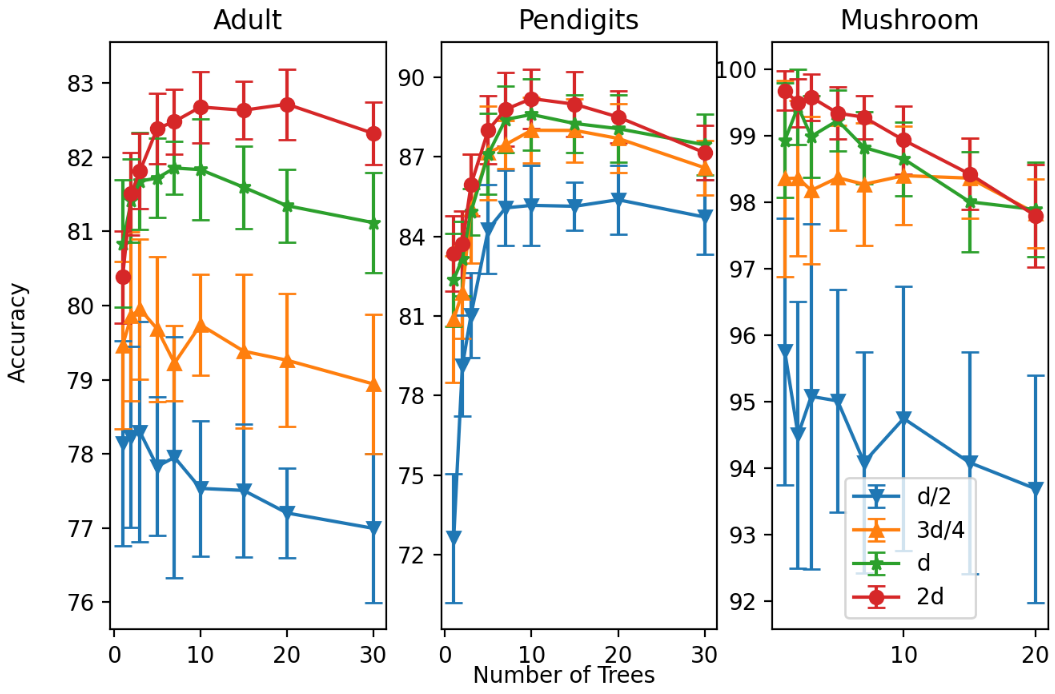

Figure 8.6: Effect of the number of trees (x-axis) and maximum depth (legend) on the accuracy, where d denotes the dimension of the data. Error bars indicates ± 1 standard deviation from the mean and ϵ_p is fixed to 2.

we see that our method achieves higher accuracy for the majority of the data sets with a statistical significant difference. The largest improvement is more than 25% better on accuracy for the *Robot* data set. Out of 22 total data sets tested, the only data set where our method is significantly worse is the *Nursery* data set.

8.4.4 Comparisons in private semi-supervised learning

For analysis of our framework in the semi-supervised setting, we perform our experiments with a reduced number of labelled samples and a separate unlabelled set. The data sets will be the same as our analysis in Section 8.4.3 except *iris* and *wine* which has less than 200 points and are too small in the semi-supervised learning. For every other data set, we only use 20% of the training set as the labelled training set, the other 80% will be used as an unlabelled set with their labels removed. We also fix the privacy budget to 2 to keep consistency. We compare the methods of our framework in semi-supervised learning with two state-of-the-art competitors - Semi-supervised RDT (SSRDT) by [70]

	Size	Attributes	RF	SPE	SRT	RDT	MaxTree
Adults	32561	14	85.47%	**82.05**% $\star$	76.89%	76.16%	81.80%
Bank	4520	16	89.39%	88.24%	**88.73**%	88.21%	88.31%
Banknotes	1372	4	99.38%	**93.54**% $\star$	77.39%	70.59%	86.93%
Blood Transfusion	747	4	74.99%	78.00%	75.71%	**78.05**%	75.68%
Car	1728	7	97.88%	**72.71**% $\star$	71.16%	69.38%	71.42%
Claves	10787	16	80.82%	**73.84**% $\star$	70.80%	73.02%	73.22%
Credit Card	30000	24	82.48%	**80.22**% $\star$	77.80%	78.28%	78.61%
Dry Bean	13611	17	92.61%	**90.74**% $\star$	84.42%	75.28%	89.99%
GammaTele	19020	10	87.99%	**82.34**% $\star$	71.62%	66.84%	78.58%
Iris	150	4	95.33%	**81.87**%	80.53%	81.20%	31.07%
Letter	20000	17	96.13%	**67.86**% $\star$	47.61%	40.25%	62.99%
Mushroom	8124	7	100.00%	**99.15**% $\star$	98.36%	93.89%	97.51%
Nursery	12960	8	98.56%	79.19%	80.64%	65.57%	**88.29**% $\star$
Occupancy	8142	7	99.71%	**98.19**% $\star$	90.26%	82.87%	97.07%
Pendigits	7494	16	99.22%	**91.72**% $\star$	89.03%	81.61%	47.12%
Robot	5456	4	99.46%	**87.43**% $\star$	61.50%	56.86%	47.70%
Student	648	33	78.58%	65.29%	60.77%	64.28%	**65.60**%
Wine	178	4	100.00%	73.00%	72.56%	**74.11**%	36.00%
Syn5d	3000	5	92.49%	**90.41**% $\star$	86.52%	79.78%	88.34%
Syn10d	3000	10	94.52%	**87.64**%	80.55%	78.57%	86.71%
Syn15d	3000	15	94.08%	**87.11**%	80.67%	80.24%	86.83%

Table 8.1: Comparisons with other private tree ensemble methods in supervised learning. The privacy budget is fixed to 2 over all data sets. The best result is highlighted with **bold**. We use the symbol $\star$ to indicate if the best result is statistically significant compared to others.

and Transductive Output Perturbation (TOP) by [93], where SSRDT is a tree-structured non-parametric method and TOP is a combination of kNN and linear predictors. Both competitors work in the case where feature privacy is not required, hence we compare them with our second setting of Alg. 12 (DPE-2) and our Private Transductive Ensemble (PTE) as they are in the same setting. We also include the first setting of Alg. 12 in our comparison however the result can be worse since it guarantees stronger privacy (both features and labels). From Table 8.2 we observe that our method SSPE-2 and PTE has better results over SSRDT and TOP for the majority of the data sets tested, in which most of them are statistically significant. Note that the results are in general worse than the figures in the supervised case since we only have access to 20% of the labels. For SSPE-1, despite it guarantees the same level of privacy for both the features and labels, the result has shown its performance remains on a same level (no significant difference) compared to the competitors which only guarantee privacy for the labels. Moreover, it has significant improvements over SSRDT and TOP on some data sets despite the additional

	SSPE - 1	SSPE - 2	PTE	SSRDT	TOP
Adults	**80.53** % $\star$	**80.88%** $\star$	**80.29%**$\star$	75.97%	76.18%
Bank	87.97%	88.02%	**88.84%**	88.58%	88.44%
Banknotes	53.86%	**89.67%**	**90.41%**$\star$	88.52%	55.30%
Blood Transfusion	76.16%	75.84%	76.11%	75.68%	**77.36%**
Car	70.23%	**72.55%** $\star$	**71.11%**	70.40%	33.88%
Claves	67.88%	67.70%	57.68%	**74.16%**$\star$	9.49%
Credit Card	**78.79** % $\star$	**78.38%** $\star$	**77.89%**	77.82%	77.80%
Dry Bean	**86.84**% $\star$	**87.76%** $\star$	**88.61%**$\star$	81.41%	72.30%
GammaTele	**74.77** % $\star$	**77.02%**$\star$	**74.11%** $\star$	65.50%	64.83%
Letter	42.27%	**53.80%**	**56.91%** $\star$	16.72%	53.42%
Mushroom	90.09%	**95.96%** $\star$	**95.46%** $\star$	92.69%	51.65%
Nursery	**74.31** % $\star$	**77.17%** $\star$	**76.37%** $\star$	66.30%	50.66%
Occupancy	**96.14** % $\star$	94.20%	92.77%	**95.16%** $\star$	79.08%
Pendigits	72.04%	**85.22%** $\star$	**87.00%** $\star$	70.10%	84.26%
Robot	**77.30**% $\star$	**87.01%** $\star$	**82.56%** $\star$	64.33%	61.43%
Student	64.77%	**65.14%**	**65.05%**	63.05%	64.98%
Syn5d	86.82%	**91.59%** $\star$	**91.54%** $\star$	90.09%	90.29%
Syn10d	86.30%	90.51%	91.11%	83.93%	**91.31%**
Syn15d	76.40%	85.94%	**86.45%**	79.59%	86.29%

Table 8.2: Comparison with other private semi-supervised methods. **Bold** indicates if the result is better than its competitors and $\star$ indicates if the difference is significant. For SSPE-1, we use *underline* to indicate when it is not significantly worse than SSRDT and TOP.

added noise as shown in Table 8.2. Hence we conclude that the methods in our framework achieve a significant improvement over the current state-of-the-art in utility performance and privacy guarantee.

Summary

We have proposed a framework of differentially private classification for supervised and semi-supervised learning with high utility. We devise a new combination of techniques to build a novel private machine ensemble for supervised learning and naturally extends to semi-supervised settings. We proposed a novel privacy budget allocation scheme that increases the usage efficiency of the available privacy budget and improves the accuracy of the computation. Our experiment analysis over a wide range of data sets show that our method provides significantly better performance than the current state-of-the-art. In the semi-supervised setting, we proposed private ensembles that can be trained efficiently using a small number of labelled samples while achieving high utility, which allows us to reduce labelling efforts in data sets with sensitive information. In particular, we proposed the first semi-supervised private ensemble that is applicable in both privacy settings where the privacy of the features are required or excluded, empirically we found that our method provides high performance in both settings.

Chapter Nine

Conclusions, Discussions and Open Problems

In this Chapter, we will summaries our findings and results presented in this thesis and the conclusions we make from our results. We will also discuss the problems that is not solved or only solved partially which is left as open research problems for the future.

9.1 Compressed non-parametric algorithms

In Chapter 4 we have studied two non-parametric algorithms, kNN and histogram, in their classical form and in the compressed learning setting where algorithms have only access to the randomly projected samples. Our main contributions in Chapter 4 includes the generalization analysis of kNN with randomly projected samples, the generalization analysis of histogram using two different approaches, and the effect of sample set structure variations on the generalization performance.

Using the two different approaches to prove the generalization of the histogram classifier, we were able to derive error bounds adaptive to the Bayes error. While the first approach gives the optimal convergence rate of $\mathcal{O}(n^{-\frac{1}{d+2}})$ in the general setting, we derived a faster convergence of $\mathcal{O}(n^{-\frac{1}{d+1}})$ towards twice the Bayes error, which is beneficial when the Bayes error is small. In particular, in the case where the Bayes error is zero, we obtained an error bound with convergence rate of $1/n$ by choosing the appropriate cell-width parameter b. Since the Lipschitz assumption on η creates a margin between different classes in the realizable case, our choice of b guarantees that all samples in the same cell will have the same class labels, which significantly simplifies the problem. We have extended the analysis to the compressed histogram, which randomly projects samples to a lower dimensional space prior training. Using the tool from [89] we showed that as long as the samples are contained in a low-dimensional structure, the generalization error of compressed histogram has a significantly reduced dimension dependence with a small multiplicative trade-off due to projection distortion. The dimensionality reduction for the histogram allows a larger choice of b without over-fitting as will happen in high-dimensions, which will reduce the bias term of its generalization error. We also showed that the compressed histogram can be applied on a variety of data structures other than linear sub-spaces, we have demonstrated variations of sub-spaces and manifolds and showed that the generalization convergence of the histogram is not significantly affected.

From our experiments we have showed that the accuracy between compressed classifiers and classical classifiers are almost identical when the sample set has a simple structure, hence allowing learning to be more feasible for high-dimensional problems. Through experimental evaluation, we have also showed that the histogram classifier runs significantly faster in both training and testing. Moreover, the time complexity of both algorithm has improved with compressed learning, kNN in particular has the most significant improvement due to its high-complexity in the non-compressed setting.

We have analysed a variety of low-dimensional structures that does not significantly disturb the accuracy performance of the histogram while reducing the dimension. However, these variants has not been analysed for the kNN algorithm in the compressed learning setting. Although intuitively, we should expect the similar guarantees also holds for the kNN due to its similarities with the histogram, it is left for future work to prove these guarantees. Furthermore, the variants of linear sub-spaces we cover in this thesis is not exhaustive, for example in the setting such that all sample points are within bounded distance to a manifold. Exploring other common low-dimensional structures in real applications that can be used in compressed learning while preserving the accuracy performance will be an interesting research topic in the future.

9.2 DP non-parametric algorithms

In Chapter 5 we have studied the DP-kNN and the histogram classifier in the DP setting, which is achieved using NoisyReg (Alg. 4). Both DP-kNN and NoisyReg provides ϵ_p-DP and works similarly by adding noise to the neighbour labels of a testing sample. Our main contribution of this Chapter is the generalization error guarantee of DP-kNN and NoisyReg in non-projected setting and randomly projected setting. We showed that as the privacy parameter ϵ_p approaches infinity, the result we obtain in this Chapter recovers to the non-private setting in Chapter 4.

By analysing the generalization error of DP-kNN, we pointed out how the privacy of kNN depends on the number of testing samples and showed that as the number of testing queries increase, the injected noise will overwhelm the true signal and lead to random predictions. In contrast, since the samples in different cells are disjoint, the number of testing samples does not affect the privacy of NoisyReg as a result of the parallel composition theorem. Furthermore, private histograms also has a run-time complexity advantage over DP-kNN for similar reasons as demonstrated in Chapter 4, making private histogram more favourable in large-scale classification in the DP setting. We also noted that the generalization error bound of the NoisyReg is tight as it achieves the same optimal convergence rate compared to its non-private analogue up to a linear dependence of the privacy parameter ϵ_p. In the realizable case we obtained 'almost optimal' convergence rate of NoisyReg which is the same as the non-private case up to privacy parameter and log factors. Unfortunately, it is unknown whether the extra log factor in the private case can be removed and obtain a tighter bound.

Similarly as in the non-private setting, we studied both private algorithms in the compressed setting. Learning in the compressed learning has the additional advantage of reducing the magnitude of added noise in the private case. From Thm. 5.10 we have showed that random projection also reduces the dimensionality dependence of the excess error due to privacy.

Furthermore, experimentally we have compared the presented algorithms with private decision trees. Unfortunately, DP-kNN is forced to choose a large value of k to overcome the added noise in the DP setting, which caused a slow convergence of accuracy performance as demonstrated in our experiments. From our experiments with real data sets, we have found that NoisyReg achieves best performance in majority of the data sets and adapts better in lower privacy levels.

Since NoisyReg has similar properties to its non-private analogue, we have not carried out extensive research of NoisyReg in the compressed learning setting. A natural extension of this Chapter is to extend the results of DP algorithms in the compressed setting to quantify the generalization error in the private case. It will also be interesting to see the performance of the private algorithms on practical high-dimensional problems.

9.3 Compressed gradient descent methods

In Chapter 6 we have studied gradient descent methods (GDMs) with randomly projected gradients. We have presented the CompSGD algorithm by [76] and two of its variants. Our main contributions in Chapter 6 is the stability and generalization analysis of the compressed gradient descent methods. Our analysis covers convex and Lipschitz loss functions that are both smooth and non-smooth.

Our analysis of the generalization error for GDMs with projected gradients divided to two main parts. The first part analyse the optimization performance of the algorithm and the convergence is presented with respect to the number of iterations T. Our analysis in optimization has shown that GDMs with randomly projected gradients achieves optimal convergence error rate as the vanilla SGD. In particular, for CompGD that uses the full batch gradient descent, we can achieve a faster optimization convergence in the smooth case by choosing a constant step size. Otherwise, our theory suggests that that all GDMs with projected gradients achieves the same optimization convergence in both smooth and non-smooth case and we should choose a step size that decays with respect to square-root of the current iteration for best convergence. The second part of our analysis has provided a rigorous stability analysis of the GDMs, which analyses the difference in accuracy performance if we alter one sample in the training set. We have proved the first stability bounds for GDMs with randomly projected gradients. We have showed that despite we have to randomly project the gradient at each iteration, which intuitively implies that difference in training samples can be magnified due to random compression, we can prove the stability bound by bounding the deviation due to random projection. Furthermore, by using the uniform stability bound and the optimization error bound from the first part, we have proved the generalization error bounds for the compressed GDMs. Our theory shows that the generalization error convergence is 'almost optimal' (up to log factors) comparing to the GDMs without compression, hence the dimensionality reduction of the gradient comes for almost 'free' without losing much generalization performance. We also showed that generalization convergence of GDMs achieves the same rate in both smooth and non-smooth case, but the smooth case benefits from larger step sizes which leads to fewer iterations required. In particular, the CompGD algorithm can choose a constant step size and achieves the same convergence rate as CompSGD with only $T \asymp \sqrt{n}$ iterations compared to $T \asymp n$. For the non-smooth case, we required to choose a smaller step size and approach to the optima more carefully, which has lead to more iterations required for convergence. Unfortunately, the fast convergence benefit of CompGD no longer exits in the non-smooth case as it also requires to take smaller steps.

A very interesting and natural research direction in the future is to investigate the properties of GDMs with randomly compressed gradients with more relaxed assumptions. For example, the relaxed Hölder-continouity condition instead of Lipschitznes and smoothness, which has been studied in [86] for vanilla SGD. A more challenging relaxation is the convexity of the loss function. Despite the widely use and success of gradient descent on non-convex problems, the theoretical analysis for SGD in the non-convex case is fairly limited compared to the classic convex setting. Investigating the guarantee of GDMs with random projection in the non-convex case will be an interesting and challenging research topic in the future.

9.4 DP compressed gradient descent

In Chapter 7 we have extended the work in Chapter 6 in the differentially private setting. Using the reduced dimensionality of randomly projected gradients, we can inject noise in a lower-dimensional space during gradient descent to guarantee privacy. Our main contribution of Chapter 7 is the optimization and generalization error guarantee for compressed GDMs in the DP setting.

Although SGD is known for dimensionality-independent error guarantee and suitable for high-dimensional learning. SGD in the DP setting however, introduces a dependence of the dimensionality of the gradient due to injected noise. As the dimensionality increases, the magnitude of the noise also increases which can ruin the signal of the gradient. We have introduced two approaches for randomly compressed gradient descents in the DP setting, we have shown that the full batch gradient approach has a smaller variance of the added noise due to a larger batch in each iteration. We have proved the first optimization error bound for DP compressed gradient descent and showed that dimensionality dependence can be reduced using compressed gradients. Furthermore, our result implies that the full-batch version benefits from fewer iterations for optimization convergence, which can further reduce the variance of the added noise due to fewer iterations. Our dimension dependence depends on the number of iterations and the Gaussian width of the constraint set. In the worst case of dimensionality dependence, our result recovers the guarantee in the non-compressed case where the dimensionality dependence is d. From the privacy perspective, as the privacy guarantee approaches zero ($\epsilon_p \to \infty$), our result in the DP setting recovers to the non-DP result in Chapter 6 up to constant factors.

Our analysis for DP compressed GDMs requires the random matrix to come from a Gaussian distribution, it is unclear whether this condition can be lifted or relaxed to the sub-Gaussian case. An intuition implied by our analysis is that the random projection matrix needs to preserve the inner product value with projected gradients so that deviation of the projected norm is bounded in the private setting. Another interesting research direction is to further investigate the affect of randomly projected gradients on other variants of DP-SGD, which can vary depending on the privacy mechanism design.

9.5 DP learning ensembles

In Chapter 8 we have introduced a learning framework that builds ensembles, we build weak learners with a tree structure based on a median-split. Our main contributions in this Chapter is the learning algorithms we introduced in both supervised and semi-supervised setting, we have proposed techniques to maximize the utility of available privacy budget which result in more accurate classification results.

We have proposed to build private trees that uses a median-based splits instead of classical approaches such as information gain and Gini index. We showed that by using the median split, we can take advantage of unlabelled data to build private trees and only use the limited number of private labels to make the final label prediction. This approach reduces the need of a large labelled training set for differentially private learning which is more cost-effective. We have proposed a random selection of splitting attribute to increase the diversity of a single tree and saving the privacy budget for more crucial computations. We have used a geometrically-scaling budget allocation scheme to guarantee median-splits along the tree will be computed accurately and privately. By splitting the training samples into disjoint subsets, we guaranteed the same level of privacy for all weak learners simultaneously, and hence the ensemble learner. Experimentally we have found that a small size of ensemble ($10-20$ trees) will be sufficient to achieve good accuracy performance, which is fairly computationally cheap to train.

We have carried out extensive experimental analysis to compare with other similar private classifiers in both supervised and semi-supervised case. By testing with a wide range of data sets, we have shown that our approach achieves the best accuracy performance in majority of the data sets, and in some cases, very close to the performance of Random Forest in the non-private setting. In particular, most of the accuracy improvements are statistically significant with 95% confidence. In the semi-supervised case, we have introduced three approaches for building private tree ensembles. Two of the methods guarantees privacy for the labels only and another can guarantee privacy for both labels and features. Our results has shown that our methods achieves the best accuracy performance in most cases. In particular, the SSPE-1 algorithm that guarantees stronger privacy has performed at least as good as its competitors that guarantees weaker privacy in many data sets.

Appendix One

Missing Proofs

A.1 Proofs of Compressed Non-parametric algorithms

Lemma A.1 (Lemma 4.3 restated)**.** *Let $C_1, \ldots, C_r$ be a collection of subsets of some domain set $\mathcal{X}$. Let S be a sequence of n points sampled i.i.d. according to some probability distribution $\mathcal{D}$ over $\mathcal{X}$. Then, for every $k \geq 2$,*

$$\mathop{\mathbb{E}}_{S}\left[\sum_{i:|C_i \cap S|<k} \mathbb{P}[C_i]\right] \leq \frac{2rk}{n}.$$

Proof. By the linearilty of expectation, we have:

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}\left[\sum_{i:|C_i \cap S|<k} \mathbb{P}[C_i]\right] = \sum_{i=1}^{r} \mathbb{P}[C_i] \mathop{\mathbb{E}}_{S\sim\mathcal{D}^m}[\mathbb{1}_{|C_i \cap S|<k}] = \sum_{i=1}^{r} \mathbb{P}[C_i] \mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}[|C_i \cap S| < k]. \quad \text{(A.1)}$$

Fix some i and suppose that $k < \mathbb{P}[C_i]n/2$, we have

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}[|C_i \cap S| < k] \leq \mathop{\mathbb{P}}_{S\sim\mathcal{D}^m}[|C_i \cap S| < \mathbb{P}[C_i]n/2].$$

Note that $\mathop{\mathbb{E}}_{S}[|C_i \cap S|] = \mathbb{P}[C_i]n$. So setting $\mu = \mathbb{P}[C_i]n$ and $\delta = 1/2$, then applying the

Chernoff's bound, we have

$$\underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < k] \le \underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < \mu(1-\delta)] \le e^{-\frac{1/4\mu}{2}} = e^{-\frac{\mathbb{P}[C_i]n}{8}}.$$

Combining this result with (A.1) and using the fact that $\max_a ae^{-an} \le \frac{1}{ne}$, we obtain

$$\begin{aligned}
\mathbb{P}[C_i]\underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < k] &\le \max_i \mathbb{P}[C_i]\underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < k] \\
&\le \max_i \mathbb{P}[C_i]e^{-\frac{\mathbb{P}[C_i]n}{8}} \\
&\le \max_i \frac{8}{ne} = \frac{8}{ne}.
\end{aligned}$$

Hence we have for $k < \mathbb{P}[C_i]n/2$,

$$\underset{S\sim\mathcal{D}^n}{\mathbb{E}}\left[\sum_{i:|C_i\cap S|<k} \mathbb{P}[C_i]\right] = \sum_{i=1}^{r} \frac{8}{ne} = \frac{8r}{ne} \le \frac{2rk}{n} \quad \text{for k} \ge 2.$$

Now for the case of $k \ge \mathbb{P}[C_i]n/2$, we have

$$\begin{aligned}
\underset{S\sim\mathcal{D}^n}{\mathbb{E}}\left[\sum_{i:|C_i\cap S|<k} \mathbb{P}[C_i]\right] &= \sum_{i=1}^{r} \mathbb{P}[C_i]\underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < k] \\
&\le r \max_i \mathbb{P}[C_i]\underset{S\sim\mathcal{D}^n}{\mathbb{P}}[|C_i \cap S| < k] \\
&\le r\mathbb{P}[C_i] \\
&\le \frac{2rk}{n}.
\end{aligned}$$

This concludes the proof. □

Lemma A.2 (Lemma 4.4 restated). *Let $k \ge 10$ and let $Y_1, \dots, Y_k$ be independent Bernoulli random variables with $\mathbb{P}[Y_i = 1] = p_i$. Denote $p = \frac{1}{k}\sum_i p_i$ and $p' = \frac{1}{k}\sum_{i=1}^{k} Y_i$. Show that*

$$\underset{Y_1,\dots,Y_k}{\mathbb{E}}\underset{Y\sim p}{\mathbb{P}}[Y \ne \mathbb{1}_{[p'>1/2]}] \le \left(1 + \sqrt{\frac{8}{k}}\right)\underset{Y\sim p}{\mathbb{P}}[Y \ne \mathbb{1}_{[p>1/2]}].$$

Proof. W.L.O.G. assume that $p \le 1/2$. Then, $\underset{Y\sim p}{\mathbb{P}}[Y \ne \mathbb{1}_{[p>1/2]}] = p$.

$$\begin{aligned}\mathop{\mathbb{E}}_{Y_1,\dots,Y_k}\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p'>1/2]}] - p &= \mathop{\mathbb{E}}_{Y_1,\dots,Y_k}\left[\mathop{\mathbb{P}}_{Y\sim p}[Y=0]\mathbb{P}[p'>1/2] + \mathop{\mathbb{P}}_{Y\sim p}[Y=1]\mathbb{P}[p'\leq 1/2]\right] - p \\ &= \mathop{\mathbb{E}}_{Y_1,\dots,Y_k}[(1-p)\mathbb{P}[p'>1/2] + p(1-\mathbb{P}[p'>1/2])] - p \\ &= \mathop{\mathbb{E}}_{Y_1,\dots,Y_k}[(1-2p)\mathbb{P}[p'>1/2] + p] - p \\ &= \mathop{\mathbb{P}}_{Y_1,\dots,Y_k}[p'>1/2](1-2p).\end{aligned} \tag{A.2}$$

Now we put an upper bound on $\mathop{\mathbb{P}}_{Y_1,\dots,Y_k}[p'>1/2]$: Let $\delta = \frac{1}{2p} - 1$, we have by Chernoff's bound:

$$\mathop{\mathbb{P}}_{Y_1,\dots,Y_k}[p'>1/2] = \mathop{\mathbb{P}}_{Y_1,\dots,Y_k}[p'>(1+\delta)p] \leq e^{-kph(1/2p-1)}, \tag{A.3}$$

where $h(\delta) = (1+\delta)\log(1+\delta) - \delta$. By combining (A.2) and (A.3) we get

$$\mathop{\mathbb{E}}_{Y_1,\dots,Y_k}\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p'>1/2]}] - p \leq (1-2p)e^{-kph(1/2p-1)} = (1-2p)e^{-kp+\frac{k}{2}(\log(2p)+1)} \leq \sqrt{\frac{8}{k}}p.$$

Hence

$$\mathop{\mathbb{E}}_{Y_1,\dots,Y_k}\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p'>1/2]}] \leq \sqrt{\frac{8}{k}}p + p = \left(1+\sqrt{\frac{8}{k}}\right)\mathop{\mathbb{P}}_{Y\sim p}[Y \neq \mathbb{1}_{[p>1/2]}].$$

□

Theorem A.3 (Theorem 4.5 restated)**.** *Let $\hat{h}$ denote the result of applying the kNN rule to an i.i.d. sample set $S \sim \mathcal{D}^n$. Assume that η is a L-Lipschitz function and $L_{\mathcal{D}}(f^*) = 0$. Then for any fixed $k \geq 2$, we have*

$$\mathop{\mathbb{E}}_{S}[L_{\mathcal{D}}(\hat{h})] \leq \mathcal{O}\left(\frac{(2L\sqrt{d})^d}{n}\right) \tag{A.4}$$

Proof. The proof follows the same procedure as for the realisable case of the histogram classifier in theorem 4.16. We first follow the proof of theorem 4.13 and noting that instead of number of points in each cell for the histogram ($|C_{i(x)}|$), we have a fixed k for kNN that indicates the search neighbourhood of an unseen sample. Furthermore, the condition $|C_{i(X)} \cap S| \geq k$ is equivalent to the k neighbours in kNN is within bounded distance of the cell width parameter $1/b$ in the histogram classifier. Explicitly, this is

$$\underset{\substack{Y_i,Y \\ i\in[n]}}{\mathbb{P}}[\hat{h}(X) \neq Y| \; |C_{i(X)} \cap S| \geq k] = \underset{\substack{Y_i,Y \\ i\in[n]}}{\mathbb{P}}[\hat{h}(X) \neq Y | \forall j \in [k], \|X - X_{\pi_j(X)}\| \leq \sqrt{d}/b] \quad \text{(A.5)}$$

This allows us to derive equation (4.43) by the same procedure, and we obtain

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h})] \leq \frac{2rk}{n} + 3\mathbb{E}[|p - \eta(X)|], \quad \text{(A.6)}$$

where we have eliminated the term with $L_{\mathcal{D}}(f^*)$ using the assumption that $L_{\mathcal{D}}(f^*) = 0$ and r here is a hyper-parameter controlling the diameter of the neighbourhood of an unseen sample. The rest of the proof follows by fixing any $k \geq 2$ and use the same argument as in theorem 4.16 to tune r. □

Theorem A.4 (Theorem 4.9 restated)**.** *Let $\hat{h}_\Phi$ be the result of applying the kNN rule on the training set S_Φ. Assume that $\eta(X)$ is an L-Lipschitz function and let $w(D)$ be the Gaussian width of the set of normalised pairwise distances of S. Then for any $0 < \delta_\Phi < 1$, $0 < \epsilon_\Phi \leq 1/2$ let m be a positive integer that satisfies:*

$$m \geq \epsilon_\Phi^{-2} C K^4 \left[w(D) + \sqrt{\log(1/\delta_\Phi)}\right]^2. \quad \text{(A.7)}$$

Then for any $k \geq 2$, we have with probability at least $1 - \delta_\Phi$:

$$\mathbb{E}_S[L_{\mathcal{D}}(\hat{h}_\Phi)] \leq \mathcal{O}\left(\frac{L^m d^{m/2}}{n}\right). \quad \text{(A.8)}$$

Proof. By the same reasoning as in theorem 4.5, we can adapt the proof of the projected histogram classifier in the realisable case and obtain the same result. □

A.2 Proofs of CompSGD

In this section, we present the missing proofs for Section 6.3 and the proofs of our results in Section 6.4.1 and Section 6.4.2. We first state the preliminary theorems that we will use during some steps of our analysis.

Lemma A.5 (Non-expansitivity of convex-smooth gradient updates)**.** *Let f be a loss function that is convex and μ-smooth. For gradient updates of the form $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t, z_{i_t})$, assume that two gradient updates $\mathbf{w}_{t+1}, \mathbf{w}'_{t+1}$ uses the same samples for update, i.e. $\mathbf{w}'_{t+1} = \mathbf{w}'_t - \eta_t \nabla f(\mathbf{w}'_t, z_{i_t})$. Then we have $\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| \leq \|\mathbf{w}_t - \mathbf{w}'_t\|$ for all $\eta_t \leq 1/(2\mu)$.*

Proof. We have

$$\begin{aligned}
\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|^2 &= \|\mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t, z_{i_t}) - (\mathbf{w}'_t - \eta_t \nabla f(\mathbf{w}'_t, z_{i_t}))\|^2 \\
&= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - 2\eta_t \langle \mathbf{w}_t - \mathbf{w}'_t, \nabla f(\mathbf{w}_t, z_{i_t}) - \nabla f(\mathbf{w}'_t, z_{i_t})\rangle \\
&\qquad + \eta_t^2 \|\nabla f(\mathbf{w}_t, z_{i_t}) - \nabla f(\mathbf{w}'_t, z_{i_t})\|^2 \\
&\leq \|\mathbf{w}_t - \mathbf{w}'_t\|^2 - 2\eta_t \frac{1}{\mu} \|\nabla f(\mathbf{w}_t, z_{i_t}) - \nabla f(\mathbf{w}'_t, z_{i_t})\| \\
&\qquad + \eta_t^2 \|\nabla f(\mathbf{w}_t, z_{i_t}) - \nabla f(\mathbf{w}'_t, z_{i_t})\|^2 \\
&= \|\mathbf{w}_t - \mathbf{w}'_t\|^2 + (\eta_t^2 - \frac{2\eta_t}{\mu}) \|\nabla f(\mathbf{w}_t, z_{i_t}) - \nabla f(\mathbf{w}'_t, z_{i_t})\|^2, \qquad \text{(A.9)}
\end{aligned}$$

where the last inequality is by applying the co-coercivity of convex and smooth function. Now, by assuming that $\eta_t \leq \frac{1}{2\mu}$ we eliminated the last term, we conclude our result $\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| \leq \|\mathbf{w}_t - \mathbf{w}'_t\|$. □

Remark 19. *From the proof of Lemma A.5, we note we do not require the gradient to be stochastic. Hence non-expansitivity holds for batch and mini-batch gradients using the same argument, as long as the condition for η_t holds.*

The following result bounds the norm of the gradient update in CompSGD with a fixed point $\mathbf{w} \in \mathcal{C}$ which will be useful for the analysis of optimisation step. We provide the proof here for completeness.

Theorem A.6 ([76])**.** *In Algorithm 5 CompSGD, for any $t \in [T]$, we have for all $\mathbf{w} \in \mathcal{C}$*

$$(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}\|^2 \leq \|\mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t, z) - \mathbf{w}\|^2. \tag{A.10}$$

Remark 20. *Note that from the proof of Theorem A.6, there are no requirements on the gradient used (∇f) being stochastic. Hence the same result will hold for batch (∇F_S) and mini-batch gradients (∇F_B).*

Proof of Theorem A.6. At iteration t, fix a RP matrix Φ_t. Define the normalizing map $u : \mathbb{R}^d \to \mathbb{R}^d$ as $u(\mathbf{w}) = \frac{\mathbf{w}}{\|\mathbf{w}\|}$. Let $\mathbf{w} \in \mathcal{C}$ be any vector. To simplify notation, note that $\mathbf{w}_{t+1} - \mathbf{w} \in \mathcal{C} + \mathcal{C}$ (the Minkowski sum) and denote $\mathcal{C}' = \{u(\mathbf{w}) \mid \mathbf{w} \in \mathcal{C} + \mathcal{C}\}$. Since $u(\mathbf{w}_{t+1} - \mathbf{w}) \in \mathcal{C}'$, we have

$$|\|\Phi_t u(\mathbf{w}_{t+1}-\mathbf{w})\|^2 - 1| \leq \sup_{\mathbf{w}\in\mathcal{C}'} |\|\Phi_t \mathbf{w}\|^2 - 1|. \tag{A.11}$$

Eq. (A.11) holds for all Φ_t. Taking expectation with respect to Φ_t, Gordon's theorem implies

$$\mathbb{E}_{\Phi_t}\left[|\|\Phi_t u(\mathbf{w}_{t+1}-\mathbf{w})\|^2 - 1|\right] \leq \mathbb{E}_{\Phi_t}\left[\sup_{\mathbf{w}\in\mathcal{C}'} |\|\Phi_t \mathbf{w}\|^2 - 1|\right] \leq \beta_t. \tag{A.12}$$

The above inequality can be rearranged as

$$\begin{aligned}
&(1-\beta_t) \leq \mathbb{E}_{\Phi_t}\left[|\|\Phi_t u(\mathbf{w}_{t+1}-\mathbf{w})\|^2|\right] \leq (1+\beta_t), \\
\Rightarrow &(1-\beta_t)\|\mathbf{w}_{t+1}-\mathbf{w}\|^2 \leq \mathbb{E}_{\Phi_t}\left[\|\Phi_t(\mathbf{w}_{t+1}-\mathbf{w})\|^2\right]. \qquad (A.13)
\end{aligned}$$

Hence we obtain

$$\begin{aligned}
(\text{A.13}) &= \mathbb{E}_{\Phi_t}\left[\left\| \prod_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla f(\mathbf{w}_t, z_{i_t}) - \prod_{\Phi_t\mathcal{C}}(\Phi_t\mathbf{w})\right\|^2\right] \\
&\leq \mathbb{E}_{\Phi_t}\left[\|(\Phi_t\mathbf{w}_t - \eta_t\Phi_t\nabla f(\mathbf{w}_t, z_{i_t}) - (\Phi_t\mathbf{w})\|^2\right] \\
&= \|(\mathbf{w}_t - \eta_t\nabla f(\mathbf{w}_t, z_{i_t})) - \mathbf{w}\|^2, \qquad (\text{A.14})
\end{aligned}$$

where we have used the fact that the projection map $\Pi_{\Phi\mathcal{C}}$ is contractive in the second step, i.e. distance between two points will not be larger after projection onto $\Phi\mathcal{C}$; and the final step follows since Φ_t is independent from all the remaining variables, $\mathbf{w}_t, \mathbf{w}, \eta_t, z_{i_t}$. $\square$

References

[1] Naman Agarwal et al. “cpSGD: Communication-efficient and differentially-private distributed SGD”. In: *arXiv preprint arXiv:1805.10559* (2018).

[2] Dan Alistarh et al. “QSGD: Communication-efficient SGD via gradient quantization and encoding”. In: *Advances in Neural Information Processing Systems* 30 (2017).

[3] Dan Alistarh et al. “The convergence of sparsified gradient methods”. In: *In Advances in Neural Information Processing Systems* (2018), pp. 5973–5983.

[4] Sunil Aryal et al. “Data-dependent dissimilarity measure: an effective alternative to geometric distance measures”. In: *Knowledge and information systems* 63 (Nov. 2017), pp. 479–506.

[5] Jean-Yves Audibert and Alexandre B. Tsybakov. “Fast Learning Rates for Plug-in Classifiers”. In: *The Annals of Statistics* 35.2 (2007), pp. 608–633.

[6] Francis Bach et al. “Structured Sparsity through Convex Optimization”. In: *Statistical Science* 27.4 (2012), pp. 450–468.

[7] K. Bache and M. Lichman. *UCI Machine Learning Repository*. 2013. URL: http://archive.ics.uci.edu/ml.

[8] Borja Balle, Gilles Barthe, and Marco Gaboardi. “Privacy amplification by subsampling: Tight analyses via couplings and divergences”. In: *Advances in Neural Information Processing Systems* 31 (2018).

[9] Richard G Baraniuk and Michael B Wakin. “Random projections of smooth manifolds”. In: *Foundations of computational mathematics* 9.1 (2009), pp. 51–77.

[10] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. “Convexity, Classification, and Risk Bounds”. In: *Journal of the American Statistical Association* 101.473 (2006), pp. 138–156.

[11] Raef Bassily, Adam Smith, and Abhradeep Thakurta. “Private empirical risk minimization: Efficient algorithms and tight error bounds”. In: *IEEE 55th annual symposium on foundations of computer science*. IEEE. 2014, pp. 464–473.

[12] Raef Bassily et al. “Stability of stochastic gradient descent on nonsmooth convex losses”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4381–4391.

[13] JL Beniley. “Multidimensional binary seareh trees used for assoeiative searehing”. In: *ACM Communications* 18.9 (1975), pp. 509–517.

[14] Thomas Berrett and Cristina Butucea. “Classification under local differential privacy”. In: *arXiv preprint arXiv:1912.04629* (2019).

[15] Nitin Bhatia et al. “Survey of nearest neighbor techniques”. In: *arXiv preprint arXiv:1007.0085* (2010).

[16] Gérard Biau. “Analysis of a random forests model”. In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 1063–1095.

[17] Ella Bingham and Heikki Mannila. “Random projection in dimensionality reduction: applications to image and text data”. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 2001, pp. 245–250.

[18] Jeremiah Blocki et al. “The johnson-lindenstrauss transform itself preserves differential privacy”. In: *53rd Annual Symposium on Foundations of Computer Science*. IEEE. 2012, pp. 410–419.

[19] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT*. Springer, 2010, pp. 177–186.

[20] Léon Bottou. "Stochastic gradient descent tricks". In: *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436.

[21] Léon Bottou, Frank E Curtis, and Jorge Nocedal. "Optimization methods for large-scale machine learning". In: *Siam Review* 60.2 (2018), pp. 223–311.

[22] Olivier Bousquet and André Elisseeff. "Stability and generalization". In: *The Journal of Machine Learning Research* 2 (2002), pp. 499–526.

[23] Christos Boutsidis and Petros Drineas. "Random projections for the nonnegative least-squares problem". In: *Linear algebra and its applications* 431.5-7 (2009), pp. 760–771.

[24] Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. "Random projections for k-means clustering". In: *Advances in Neural Information Processing Systems* 23 (2010).

[25] Leo Breiman. "Consistency for a simple model of random forests". In: (2004).

[26] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[27] Leo Breiman et al. *Classification and regression trees*. Routledge, 2017.

[28] Zhiqi Bu et al. "Fast and memory efficient differentially private-sgd via JL projections". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 19680–19691.

[29] Mark Bun and Thomas Steinke. "Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds". In: *Theory of Cryptography*. Ed. by Martin Hirt and Adam Smith. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 635–658.

[30] Emmanuel J Candes and Justin K Romberg. "Signal recovery from random projections". In: *Computational Imaging III*. Vol. 5674. SPIE. 2005, pp. 76–86.

[31] Timothy I Cannings and Richard J Samworth. "Random-projection ensemble classification". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79.4 (2017), pp. 959–1035.

[32] Zachary Charles and Dimitris Papailiopoulos. “Stability and generalization of learning algorithms that converge to global optima”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 745–754.

[33] Kamalika Chaudhuri and Claire Monteleoni. “Privacy-preserving logistic regression”. In: *Proceedings of the 21st International Conference on Neural Information Processing Systems*. 2008, pp. 289–296.

[34] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. “Differentially private empirical risk minimization.” In: *Journal of Machine Learning Research* 12.3 (2011).

[35] Xiangyi Chen, Steven Z Wu, and Mingyi Hong. “Understanding gradient clipping in private SGD: A geometric perspective”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 13773–13782.

[36] Yuansi Chen, Chi Jin, and Bin Yu. “Stability and convergence trade-off of iterative optimization algorithms”. In: *arXiv preprint arXiv:1804.01619* (2018).

[37] Kenneth L Clarkson. “Tighter bounds for random projections of manifolds”. In: *Proceedings of the twenty-fourth annual symposium on Computational geometry*. 2008, pp. 39–48.

[38] Shorya Consul and Sinead A. William. “Differentially Private Random Forests for Regression and Classification”. In: *Association for the Advancement of Artificial Intelligence* (2021).

[39] Graham Cormode et al. “Differentially private spatial decompositions”. In: *2012 IEEE 28th International Conference on Data Engineering*. IEEE. 2012, pp. 20–31.

[40] Andrew Cotter et al. “Better mini-batch algorithms via accelerated gradient methods”. In: *Advances in Neural Information Processing Systems* 24 (2011).

[41] T. Cover and P. Hart. “Nearest neighbor pattern classification”. In: *IEEE Transactions on Information Theory* 13.1 (1967), pp. 21–27. DOI: 10.1109/TIT.1967.1053964.

[42] Sanjoy Dasgupta and Yoav Freund. “Random projection trees and low dimensional manifolds”. In: *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 2008, pp. 537–546.

[43] Sanjoy Dasgupta and Anupam Gupta. “An elementary proof of a theorem of Johnson and Lindenstrauss”. In: *Random Structures & Algorithms* 22.1 (2003), pp. 60–65.

[44] Luc Devroye and Terry Wagner. “Distribution-free performance bounds for potential function rules”. In: *IEEE Transactions on Information Theory* 25.5 (1979), pp. 601–604.

[45] Irit Dinur and Kobbi Nissim. “Revealing information while preserving privacy”. In: *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 2003, pp. 202–210.

[46] Sjoerd Dirksen. “Dimensionality reduction with subgaussian matrices: a unified theory”. In: *Foundations of Computational Mathematics* 16.5 (2016), pp. 1367–1396.

[47] Cynthia Dwork. “Differential Privacy”. In: *Automata, Languages and Programming* (2006), pp. 1–12.

[48] Cynthia Dwork and Aaron Roth. “The algorithmic foundations of differential privacy.” In: *Foundations and Trends in Theoretical Computer Science* 9.3-4 (2014), pp. 211–407.

[49] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. “Boosting and differential privacy”. In: *51st Annual Symposium on Foundations of Computer Science*. IEEE. 2010, pp. 51–60.

[50] Cynthia Dwork et al. “Calibrating noise to sensitivity in private data analysis”. In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284.

[51] Andre Elisseeff, Theodoros Evgeniou, and Massimiliano Pontil. “Stability of Randomized Learning Algorithms”. In: *Journal of Machine Learning Research* 6.55-79 (2005).

[52] Xiaoli Z Fern and Carla E Brodley. “Random projection for high dimensional data clustering: A cluster ensemble approach”. In: *Proceedings of the 20th International Conference on Machine Learning.* 2003, pp. 186–193.

[53] Evelyn Fix and Joseph Lawson Hodges. “Discriminatory analysis. Nonparametric discrimination: Consistency properties”. In: *International Statistical Review/Revue Internationale de Statistique* 57.3 (1989), pp. 238–247.

[54] Sam Fletcher and Md Zahidul Islam. “A Differentially Private Decision Forest”. In: *Proceedings of the 13-th Australasian Data Mining Conference.* Vol. 15. 2015, pp. 99–108.

[55] Sam Fletcher and Md Zahidul Islam. “Decision tree classification with differential privacy: A survey.” In: *ACM computing surveys* 52.4 (2019).

[56] Sam Fletcher and Md Zahidul Islam. “Differentially private random decision forests using smooth sensitivity”. In: *Expert Systems with Applications* 78 (2017), pp. 16–31.

[57] David Freedman and Persi Diaconis. “On the histogram as a density estimator: L 2 theory”. In: *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* 57.4 (1981), pp. 453–476.

[58] Arik Friedman and Assaf Schuster. “Data mining with differential privacy”. In: *Proceedings of the 16th SIGKDD Conference on Knowledge Discovery and Data Mining.* 2010, pp. 493–502.

[59] Pierre Geurts, Damien Ernst, and Louis Wehenkel. “Extremely randomized trees”. In: *Machine Learning* 63.1 (2006), pp. 3–42.

[60] Navin Goel, George Bebis, and Ara Nefian. “Face recognition experiments with random projection”. In: *Biometric Technology for Human Identification II.* Vol. 5779. SPIE. 2005, pp. 426–437.

[61] André R Gonçalves et al. “Multi-task sparse structure learning”. In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management.* 2014, pp. 451–460.

[62] Maoguo Gong et al. “A Survey on Differentially Private Machine Learning”. In: *IEEE Computational Intelligence Magazine* 15.2 (2020), pp. 49–64.

[63] Yehoram Gordon. “On Milman’s inequality and random subspaces which escape through a mesh in $\mathbb{R}^n$”. In: *Geometric aspects of functional analysis*. Springer, 1988, pp. 84–106.

[64] Mehmet Emre Gursoy et al. “Differentially private nearest neighbor classification”. In: *Data Mining and Knowledge Discovery* 31.5 (2017), pp. 1544–1575.

[65] Hanyuan Hang et al. “Histogram Transform Ensembles for Large-scale Regression.” In: *Journal of Machine Learning Research* 22 (2021), pp. 95–1.

[66] Moritz Hardt, Ben Recht, and Yoram Singer. “Train faster, generalize better: Stability of stochastic gradient descent”. In: *International conference on machine learning*. PMLR. 2016, pp. 1225–1234.

[67] Trevor Hastie et al. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer, 2009.

[68] Michael Hay et al. “Boosting the accuracy of differentially-private histograms through consistency”. In: *arXiv preprint arXiv:0904.0942* (2009).

[69] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. “Neural networks for machine learning lecture 6a overview of mini-batch gradient descent”. In: *Cited on* 14.8 (2012), p. 2.

[70] Geetha Jagannathan, Claire Monteleoni, and Krishnan Pillaipakkamnatt. “A semi-supervised learning approach to differential privacy”. In: *International Conference on Data Mining Workshops*. IEEE. 2013, pp. 841–848.

[71] Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca N Wright. “A practical differentially private random decision tree classifier”. In: *International Conference on Data Mining Workshops*. IEEE. 2009, pp. 114–121.

[72] Martin Jaggi. “Sparse convex optimization methods for machine learning”. PhD thesis. ETH Zurich, 2011.

[73] Ata Kabán. “A new look at nearest neighbours: Identifying benign input geometries via random projections”. In: *Asian Conference on Machine Learning*. PMLR. 2016, pp. 65–80.

[74] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. “The composition theorem for differential privacy”. In: *International Conference on Machine Learning*. PMLR. 2015, pp. 1376–1385.

[75] Zoi Kaoudi et al. “A cost-based optimizer for gradient descent optimization”. In: *ACM International Conference on Management of Data*. 2017, pp. 977–992.

[76] Shiva Prasad Kasiviswanathan. “SGD with low-dimensional gradients with applications to private and distributed learning”. In: *Uncertainty in Artificial Intelligence*. PMLR. 2021, pp. 1905–1915.

[77] Krishnaram Kenthapadi et al. “Privacy via the johnson-lindenstrauss transform”. In: *arXiv preprint arXiv:1204.2606* (2012).

[78] Sarit Khirirat, Hamid Reza Feyzmahdavian, and Mikael Johansson. “Mini-batch gradient descent: Faster convergence under data sparsity”. In: *Annual Conference on Decision and Control (CDC)*. IEEE. 2017, pp. 2880–2887.

[79] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. “Private convex empirical risk minimization and high-dimensional regression”. In: *Conference on Learning Theory*. JMLR Workshop and Conference Proceedings. 2012, pp. 25–1.

[80] B Klartag and Shahar Mendelson. “Empirical processes and random projections”. In: *Journal of Functional Analysis* 225.1 (2005), pp. 229–245.

[81] Jason Klusowski. “Sharp Analysis of a Simple Model for Random Forests”. In: *International Conference on Artificial Intelligence and Statistics*. 2021, pp. 757–765.

[82] Jakub Konečný et al. “Mini-batch semi-stochastic gradient descent in the proximal setting”. In: *IEEE Journal of Selected Topics in Signal Processing* 10.2 (2015), pp. 242–255.

[83] Petri Kontkanen and Petri Myllymäki. "MDL histogram density estimation". In: *Artificial Intelligence and Statistics*. PMLR. 2007, pp. 219–226.

[84] Ilja Kuzborskij and Christoph Lampert. "Data-dependent stability of stochastic gradient descent". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 2815–2824.

[85] Kasper Green Larsen and Jelani Nelson. "Optimality of the Johnson-Lindenstrauss lemma". In: *Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 633–638.

[86] Yunwen Lei and Yiming Ying. "Fine-grained analysis of stability and generalization for stochastic gradient descent". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 5809–5819.

[87] Chencheng Li et al. "Differentially private distributed online learning". In: *IEEE Transactions on Knowledge and Data Engineering* 30.8 (2018), pp. 1440–1453.

[88] Mu Li et al. "Efficient mini-batch training for stochastic optimization". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2014, pp. 661–670.

[89] Christopher Liaw et al. "A simple tool for bounding the deviation of random matrices on geometric sets". In: *Geometric Aspects of Functional Analysis: Israel Seminar (GAFA) 2014–2016*. Springer. 2017, pp. 277–299.

[90] Jun Liu, Jianhui Chen, and Jieping Ye. "Large-Scale Sparse Logistic Regression". In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. Paris, France: Association for Computing Machinery, 2009, pp. 547–556.

[91] Tongliang Liu et al. "Algorithmic stability and hypothesis complexity". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2159–2167.

[92] Xiaoqian Liu et al. "Differentially private classification with decision tree ensemble". In: *Applied Soft Computing* 62 (2018), pp. 807–816.

[93] Xu Long and Jun Sakuma. "Differentially Private Semi-Supervised Classification". In: *International Conference on Smart Computing (SMARTCOMP)*. IEEE. 2017, pp. 1–6.

[94] Michael Mahoney. "The Johnson-Lindenstrauss Lemma". In: *Lecture notes on Alg orithms for Modern Massive Data Set Analysis* (2009).

[95] Odalric Maillard and Rémi Munos. "Linear regression with random projections". In: *Journal of Machine Learning Research* 13.1 (2012), pp. 2735–2772.

[96] Jiří Matoušek. "On variants of the Johnson–Lindenstrauss lemma". In: *Random Structures & Algorithms* 33.2 (2008), pp. 142–156.

[97] J. Maudes et al. "Random feature weights for decision tree ensemble construction". In: *Information Fusion* 13.1 (2012), pp. 20–30.

[98] Chandresh Kumar Maurya and Durga Toshniwal. "Large-scale distributed sparse class-imbalance learning". In: *Information Sciences* 456 (2018), pp. 1–12.

[99] Frank McSherry and Kunal Talwar. "Mechanism design via differential privacy". In: *Symposium on Foundations of Computer Science (FOCS'07)*. IEEE. 2007, pp. 94–103.

[100] Frank D McSherry. "Privacy integrated queries: an extensible platform for privacy-preserving data analysis". In: *Proceedings of the ACM SIGMOD International Conference on Management of data*. 2009, pp. 19–30.

[101] Ilya Mironov. "Rényi Differential Privacy". In: *Computer Security Foundations Symposium (CSF)*. 2017, pp. 263–275. DOI: 10.1109/CSF.2017.11.

[102] Noman Mohammed et al. "Differentially private data release for data mining". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011, pp. 493–501.

[103] Arvind Narayanan and Vitaly Shmatikov. "Robust de-anonymization of large sparse datasets". In: *Symposium on Security and Privacy*. IEEE. 2008, pp. 111–125.

[104] Yurii Nesterov. *Introductory Lectures on Convex Optimization.* Vol. 87. Springer Science & Business Media, 2003.

[105] Huy Lê Nguyễn, Jonathan Ullman, and Lydia Zakynthinou. "Efficient Private Algorithms for Learning Halfspaces". In: (2019).

[106] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. "Smooth sensitivity and sampling in private data analysis". In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing.* 2007, pp. 75–84.

[107] Andrew Nobel. "Histogram regression estimation using data-dependent partitions". In: *The Annals of Statistics* 24.3 (1996), pp. 1084–1105.

[108] Anh T Pham and Jing Xi. "Differentially private semi-supervised learning with known class priors". In: *International Conference on Big Data.* IEEE. 2018, pp. 801–810.

[109] J. Ross Quinlan. "Improved use of continuous attributes in C4. 5." In: *Journal of artificial intelligence research* 4 (1996), pp. 77–90.

[110] J. Ross. Quinlan. *C4. 5: programs for machine learning.* Elsevier, 1993.

[111] Santu Rana, Sunil Kumar Gupta, and Svetha Venkatesh. "Differentially private random forest with high utility". In: *International Conference on Data Mining.* IEEE. 2015, pp. 955–960.

[112] Benjamin Recht et al. "Hogwild!: A lock-free approach to parallelizing stochastic gradient descent". In: *Advances in Neural Information Processing Systems* 24 (2011).

[113] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[114] Alon Schclar and Lior Rokach. "Random projection ensemble classifiers". In: *International Conference on Enterprise Information Systems.* Springer. 2009, pp. 309–316.

[115] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[116] Shai Shalev-Shwartz et al. "Learnability, stability and uniform convergence". In: *The Journal of Machine Learning Research* 11 (2010), pp. 2635–2670.

[117] Ohad Shamir and Tong Zhang. "Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes". In: *International conference on machine learning*. PMLR. 2013, pp. 71–79.

[118] Mehrdad Showkatbakhsh, Can Karakus, and Suhas Diggavi. "Privacy-utility trade-off of linear regression under random projections and additive noise". In: *International Symposium on Information Theory (ISIT)*. IEEE. 2018, pp. 186–190.

[119] Zhongwei Si, Shaoguo Wen, and Bing Dong. "NOMA codebook optimization by batch gradient descent". In: *IEEE Access* 7 (2019), pp. 117274–117281.

[120] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. "Stochastic gradient descent with differentially private updates". In: *Global Conference on Signal and Information Processing*. IEEE. 2013, pp. 245–248.

[121] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. "Sparsified sgd with memory". In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 4447–4458.

[122] Dong Su, Jianneng Cao, and Ninghui Li. "Differentially private projected histograms of multi-attribute data for classification". In: *arXiv preprint arXiv:1504.05997* (2015).

[123] Dong Su et al. "Differentially Private K-Means Clustering". In: New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450339353. DOI: 10.1145/2857705.2857708. URL: https://doi.org/10.1145/2857705.2857708.

[124] Dong Su et al. "PrivPfC: Differentially private data publication for classification". In: *The VLDB Journal* 27.2 (2018), pp. 201–223.

[125] Latanya Sweeney. "Weaving technology and policy together to maintain confidentiality". In: *The Journal of Law, Medicine & Ethics* 25.2-3 (1997), pp. 98–110.

[126] Zoltán Szabó and András Lőrincz. “Distributed high dimensional information theoretical image registration via random projections”. In: *Digital Signal Processing* 22.6 (2012), pp. 894–902.

[127] Kean Ming Tan et al. “A convex formulation for high-dimensional sparse sliced inverse regression”. In: *Biometrika* 105.4 (Oct. 2018), pp. 769–782.

[128] Gian-Andrea Thanei, Christina Heinze, and Nicolai Meinshausen. “Random projections for large-scale regression”. In: *Big and complex data analysis*. Springer, 2017, pp. 51–68.

[129] J Van Ryzin. “A histogram method of density estimation”. In: *Communications in Statistics-Theory and Methods* 2.6 (1973), pp. 493–506.

[130] Michel Verleysen and Damien François. “The curse of dimensionality in data mining and time series prediction”. In: *International work-conference on artificial neural networks*. Springer. 2005, pp. 758–770.

[131] Di Wang, Changyou Chen, and Jinhui Xu. “Differentially private empirical risk minimization with non-convex loss functions”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6526–6535.

[132] Di Wang, Minwei Ye, and Jinhui Xu. “Differentially private empirical risk minimization revisited: Faster and more general”. In: *Advances in Neural Information Processing Systems* 30 (2017).

[133] Hongyi Wang et al. “Atomo: Communication-efficient learning via atomic sparsification”. In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 9850–9861.

[134] Puyu Wang et al. “Differentially private SGD with non-smooth losses”. In: *Applied and Computational Harmonic Analysis* (2022), pp. 306–336.

[135] Yining Wang, Yu-Xiang Wang, and Aarti Singh. “Differentially private subspace clustering”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/051e4e127b92f5d98d3c79b195f2b291-Paper.pdf.

[136] Larry Wasserman and Shuheng Zhou. “A statistical framework for differential privacy”. In: *Journal of the American Statistical Association* 105.489 (2010), pp. 375–389.

[137] David H Wolpert and William G Macready. “No free lunch theorems for optimization”. In: *IEEE transactions on evolutionary computation* 1.1 (1997), pp. 67–82.

[138] Yonghui Xiao et al. “DPCube: Differentially private histogram release through multidimensional partitioning”. In: *arXiv preprint arXiv:1202.5358* (2012).

[139] Bangzhou Xin et al. “Differentially Private Greedy Decision Forest”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, pp. 2672–2676.

[140] Ping Xiong, Tian-Qing Zhu, and Xiao-Feng Wang. “A survey on differential privacy and applications”. In: *Jisuanji Xuebao/Chinese Journal of Computers* 37.1 (2014), pp. 101–122.

[141] Chugui Xu et al. “DPPro: Differentially private high-dimensional data release via random projection”. In: *IEEE Transactions on Information Forensics and Security* 12.12 (2017), pp. 3081–3093.

[142] Mengmeng Yang et al. “Local differential privacy and its applications: A comprehensive survey”. In: *arXiv preprint arXiv:2008.03686* (2020).

[143] Mao Ye et al. “Fuzzy-means and cluster ensemble with random projection for big data clustering”. In: *Mathematical Problems in Engineering* (2016).

[144] Tong Zhang. “Solving large scale linear prediction problems using stochastic gradient descent algorithms”. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 116.

[145] Peilin Zhao and Tong Zhang. “Accelerating minibatch stochastic gradient descent using stratified sampling”. In: *arXiv preprint arXiv:1405.3080* (2014).

[146] Yuqing Zhu et al. “Private-knn: Practical differential privacy for computer vision”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 11854–11862.

[147] Martin Zinkevich et al. “Parallelized stochastic gradient descent”. In: *Advances in Neural Information Processing Systems* 23 (2010).

Websites consulted

- Wikipedia – www.wikipedia.org
- WolframAlpha – www.wolframalpha.com
- Scikit-learn – https://scikit-learn.org
- SciPy – https://scipy.org
- UCI Machine Learning Repository – https://archive.ics.uci.edu/ml/index.php