1
Introduction

Machine learning is starting to take over decision-making in many aspects of our life, including:

(a) keeping us safe on our daily commute in self-driving cars,
(b) making accurate diagnoses based on our symptoms and medical history,
(c) pricing and trading complex securities, and
(d) discovering new science, such as the genetic basis for various diseases.

But the startling truth is that these algorithms work without any sort of provable guarantees on their behavior. When they are faced with an optimization problem, do they actually find the best solution, or even a pretty good one? When they posit a probabilistic model, can they incorporate new evidence and sample from the true posterior distribution? Machine learning works amazingly well in practice, but that doesn't mean we understand *why* it works so well.

If you've taken traditional algorithms courses, the usual way you've been exposed to thinking about algorithms is through worst-case analysis. When you have a sorting algorithm, you measure its running time based on how many operations it takes on the worst possible input. That's a convenient type of bound to have, because it means you can say meaningful things about how long your algorithm takes without ever worrying about the types of inputs you usually give it.

But what makes analyzing machine learning algorithms, especially modern ones, so challenging is that the types of problems they are trying to solve really are *NP*-hard on worst-case inputs. When you cast the problem of finding the parameters that best fit your data as an optimization problem, there are instances where it is *NP*-hard to find a good fit. When you posit a probabilistic model and want to use it to perform inference, there are instances where that is *NP*-hard as well.

https://doi.org/10.1017/9781316882177.002 Published online by Cambridge University Press

In this book, we will approach the problem of giving provable guarantees for machine learning by trying to find more realistic models for our data. In many applications, there are reasonable assumptions we can make, based on the context in which the problem came up, that can get us around these worst-case impediments and allow us to rigorously analyze heuristics that are used in practice, as well as design fundamentally new ways of solving some of the central, recurring problems in machine learning.

To take a step back, the idea of moving beyond worst-case analysis is an idea that is as old[1] as theoretical computer science itself [95]. In fact, there are many different flavors of what it means to understand the behavior of algorithms on "typical" instances, including:

(a) probabilistic models for your input, or even hybrid models that combine elements of worst-case and average-case analysis like semi-random models [38, 71] or smoothed analysis [39, 130];
(b) ways to measure the complexity of your problem and ask for algorithms that are fast on simple inputs, as in parameterized complexity [66]; and
(c) notions of stability that attempt to articulate what instances of your problem have meaningful answers and are the ones you actually want to solve [20, 32].

This is by no means an exhaustive list of topics or references. Regardless, in this book, we will approach machine learning problems armed with these sorts of insights about ways to get around intractability.

Ultimately, we hope that theoretical computer science and machine learning have a lot left to teach each other. Understanding why heuristics like expectation- maximization or gradient descent on a nonconvex function work so well in practice is a grand challenge for theoretical computer science. But to make progress on these questions, we need to understand what types of models and assumptions make sense in the context of machine learning. On the other hand, if we make progress on these hard problems and develop new insights about *why* heuristics work so well, we can hope to engineer them better. We can even hope to discover totally new ways to solve some of the important problems in machine learning, especially by leveraging modern tools in our algorithmic toolkit.

In this book, we will cover the following topics:

(a) Nonnegative matrix factorization
(b) Topic modeling

[1] After all, heuristics performing well on real life inputs are old as well (long predating modern machine learning), hence so is the need to explain them.

https://doi.org/10.1017/9781316882177.002 Published online by Cambridge University Press

(c) Tensor decompositions
(d) Sparse recovery
(e) Sparse coding
(f) Learning mixtures models
(g) Matrix completion

I hope more chapters will be added in later versions as the field develops and makes new discoveries.

https://doi.org/10.1017/9781316882177.002 Published online by Cambridge University Press