Dive into Deep Learning Book

Notes Based on the book Dive into Deep Learning at [d2l.ai](d2l.ai) & supplemental reading orvideos

Krish Katariya

Last updated: **February 25, 2024**

Contents

1. Introduction

Machine Learning is the study of algorithms that can learn from experience, typically in the form of observational data or interactions with an environment, its performance improves.

Core components that follow us around, no matter what kind of machine learning we tackle:
1. The *data* that we can learn from
2. A *model* of how to transform the data
3. An *objective function* that quantifies how well (or badly) the model is doing
4. An *algorithm* to adjust the model's parameters to optimize the objective function

Note 1.1

Data

Each data point/instance consists of a set of attributes called *features* (or covaries/inputs)/

In supervised learning, our goal is the predict the value of a special attribute, called the *label/target* which is not part of the model's input.

In cases when every example is characterized by the same number of numerical features, we say that the inputs are fixed-length vectors and we call the constant length of the vectors the *dimensionality* of the data. However, often we have to work with varying lengths of data.

By **model**, we denote the computational machinery for ingesting data of one type, and spitting out predictions of a possibly different type. Specifically, we are interested in *statistical models* that can be estimated from data.

Deep learning is differentiated from the classical approaches principally by the set of power models that it focuses on. We often chain many successive transformations of the data from top to bottom.

Note 1.2

In order to develop a formal mathematical system for learning machines, we need to have objective functions to see how good or bad our models our. We often define objective functions so that lower is better by convention, which is why we call them *loss functions*.

When trying to predict numerical values, we can often use *squared error* as a loss function, but for classification, the most common objective function is to minimize error rate (% of wrong predictions). **During optimization, we think of the loss as a function of the model's parameters, and treat the training dataset as a constant**. We will learn the best values of our model's parameters by minimize the loss incurred on a set consisting of some number of examples collected for training.

We typically want to split the available data into a *training dataset* and a *test set* which is held for evaluation. When a model performs well on the training set but fails to generalize unseen data, we say that it is **overfitting** on training data.

Now, once we have a data source, a model, and a well defined objective function, we need an algorithm capable of searching for the best possible parameters for minimizing the loss function. Popular optimization algorithms for deep learning are based on an approach known as **gradient descent**

Definition 1.1

Gradient Descent: an optimization algorithm where the method check to see (at each step) for each parameter, how that training set would change if you perturbed that parameter by just a small amount. It then updates the parameter in a direction that lowers the loss.

Neural Networks and other complicated models often have many hundreds or thousands of inputs, so using calculus/statistics to find the minimum is often not feasible. Instead, we start at any old input and find the slope of the function where you are, and shift towards the direction that minimizes the loss.

However, there are many local minimums we can land in (see below)