Machine Learning

Yuzhe Zhang

March 2025

0.1 Introduction

In this document, I aim to demonstrate the mathematical principles underlying machine learning algorithms. The document is to record the study path of my Ph.D. in University of Otago.

Chapter 1

Feedforward Network

1.1 Introduction

In this chapter, we derive the formula for the feedforward network.

The feedforward network is the foundation of neural networks, which are considered the backbone of deep learning. The core idea behind many machine learning algorithms can be traced back to linear regression:

$$\hat{y} = ax + b$$

In linear regression, the goal is to find the parameter a that minimizes an error function. There are many ways to define this error function, such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and others, which we will describe in detail later. The most commonly used one is MSE:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{1.1}$$

This equation calculates the average squared distance between the true values and the predicted values. In linear regression, the optimal parameter a minimizes the MSE. This process is referred to as training the model. Once the model is trained, it can make predictions using new input x and the learned parameter a.

In a feedforward network (FFN), the goal is similar to that of linear regression: use a dataset to train a model by finding parameters that minimize the error function. After training, the FFN can make predictions on unseen data.

To construct and train an FFN, several elements and steps are required. First, we consider the input data (which are assumed to have predictive value), the weights (which act similarly to the coefficient a in linear regression), and the bias b, which allows the network to approximate functions that do not pass through the origin:

$$a = Wx + b \tag{1.2}$$

Here, W is the weight matrix and b is the bias vector. At this point, the FFN resembles a linear regression model. The key difference is that FFNs contain multiple layers, each with neurons and non-linear activation functions.

Let us consider a simple two-layer FFN that predicts the next day's S&P 500 index using the past two days' S&P 500 index and gold prices. The input data would look like:

Day	**S&P 500**	**Gold Price**
Day 1	3000	800
Day 2	3100	790

Table 1.1: Simple simulated data.

1. The input training data x is flattened into a single vector since FFNs do not incorporate time structure. In this case, the input vector is in $\mathbb{R}^4$.

2. The algorithm initializes weights for each element in the input vector. The weight matrix W has 5 columns (4 for the input features and 1 for the bias), and a number of rows equal to the number of neurons in the next layer.

The FFN consists of multiple layers, neurons (perceptrons), and activation functions. Unlike linear regression, FFNs can model non-linear relationships. This is achieved by applying a non-linear activation function to each neuron's output:

1. In the first layer, compute the activation function output: $\sigma(Wx + b)$. If there are 3 neurons in the first layer, then $W \in \mathbb{R}^{3\times 5}$.

2. The output of the first layer becomes the input to the next layer. This process is repeated until the final prediction is generated. The non-linearity of the activation function allows the network to model complex relationships.