CWM: AI/ML with python --- Optimization Notes by Yangchen Pan

Mathematical Optimization

It refers to the selection of a best element with regard to some criterion or objective, from some set of available alternatives.

Assume we have an objective function $c(w) : \mathbb{R}^d \rightarrow \mathbb{R}$ that we want to minimize:

$$\min_w c(w)$$

In machine learning, the function we minimize is often called the **cost function**.

We already saw some cost functions like the summation of cross-entropy loss or squared loss over a set of traininig points. Note that in the Maximum Likelihood Estimation (MLE) problem, we aim to maximize the likelihood function. This is equivalent to minimizing the negative log-likelihood function.

The objective is to solve the problem $\min_w c(w)$. The basic idea is to iteratively find a direction Δw such that moving along this direction decreases the function value $c(w)$. By doing this iteratively, we aim to gradually decrease the function value and hopefully reach a stationary point, indicating a local optimum where further decreases are not possible.

Gradient Descent

One of the fundamental approaches for minimizing the cost function is called **gradient descent**.

Here, we use $\nabla c(w)$ to denote the gradient vector with respect to w, defined as:

$$\nabla c(w) = \left(\frac{\partial c(w)}{\partial w_1}, \frac{\partial c(w)}{\partial w_2}, \ldots, \frac{\partial c(w)}{\partial w_d} \right)$$

To see how gradient descent works, let's perform a first-order Taylor expansion around the point $w + \Delta w$:

$$c(w + \Delta w) \approx c(w) + \nabla c(w)^\top \Delta w$$

From this expansion, it is evident that if the vectors $\nabla c(w)$ and Δw form an acute angle, then $c(w + \Delta w) \geq c(w)$; if they form an obtuse angle, then $c(w + \Delta w) \leq c(w)$. Therefore, the steepest ascent/descent direction is achieved when $\Delta w \propto \pm \nabla c(w)$. The learning rate α is crucial because the Taylor approximation may be highly inaccurate if we move too far away from w.

In summary, to minimize $c(w)$, we use the update rule:

$$w \leftarrow w - \alpha \nabla c(w)$$

where α is called the learning rate.

Stochastic Gradient Descent

In many machine learning problems, the loss function is a summation over many data points. As a computationally efficient alternative, a stochastic gradient can be used. This involves randomly sampling one data point or a mini-batch of data points to estimate the gradient, rather than computing the gradient using the entire dataset. This approach is known as **Stochastic Gradient Descent (SGD)**.

Summary

- **Objective Function**: $c(w)$ to minimize.
- **Gradient**: $\nabla c(w)$ provides the direction of the steepest ascent.
- **Gradient Descent Update Rule**: $w \leftarrow w - \alpha \nabla c(w)$.
- **Learning Rate**: α controls the step size.
- **Stochastic Gradient Descent**: Uses random samples to estimate the gradient, improving computational efficiency.