Multi-variate Modeling with Neural Networks

1 Notation

Let $\mathcal{X} \subseteq \mathbb{R}^{d_x}$ and $\mathcal{Y} \subseteq \mathbb{R}^{d_y}$ denote the input and output spaces, respectively. We consider a supervised learning setting with observations $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, where each pair $(\mathbf{x}_i, \mathbf{y}_i)$ is drawn i.i.d. from an unknown distribution $\mathbb{P}_{\mathbf{X},\mathbf{Y}}$. The input $\mathbf{x}_i \in \mathcal{X}$ is a feature vector of dimension d_x, and the corresponding response $\mathbf{y}_i \in \mathcal{Y}$ is a real-valued vector of dimension d_y.

We assume that the conditional distribution of $\mathbf{y} \mid \mathbf{x}$ is Gaussian with a mean function $f : \mathcal{X} \to \mathbb{R}^{d_y}$ and a covariance function $\Sigma : \mathcal{X} \to \mathbb{S}_{++}^{d_y}$, where $\mathbb{S}_{++}^{d_y}$ denotes the set of symmetric positive definite matrices of dimension $d_y \times d_y$. Specifically, the generative model is

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\epsilon}(\mathbf{x}), \quad \text{with} \quad \boldsymbol{\epsilon}(\mathbf{x}) \sim \mathcal{N}(\mathbf{0}, \Sigma(\mathbf{x})).$$

The functions f and Σ are unknown and must be learned from data. In what follows, we propose a parameterization of both components using neural networks, along with an appropriate training objective.

2 Problem Formulation

Our goal is to learn a flexible, heteroscedastic regression model that jointly estimates both the conditional mean and the input-dependent covariance of the target variable $\mathbf{y} \mid \mathbf{x}$. Let $f_\theta : \mathcal{X} \to \mathbb{R}^{d_y}$ be a neural network parameterized by θ, which approximates the mean function f. Simultaneously, let $g_\phi : \mathcal{X} \to \mathbb{R}^{d_y(d_y+1)/2}$ be another neural network parameterized by ϕ, which produces the parameters of a lower-triangular matrix $L(\mathbf{x})$ with positive diagonal entries, such that

$$\Sigma_\phi(\mathbf{x}) = L_\phi(\mathbf{x}) L_\phi(\mathbf{x})^\top.$$

This Cholesky-based parameterization guarantees that $\Sigma_\phi(\mathbf{x})$ is symmetric and positive definite for all $\mathbf{x}$, as required for a valid Gaussian covariance.

Under this parameterization, the model defines a conditional density

$$p_\theta(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{y}; f_\theta(\mathbf{x}), \Sigma_\phi(\mathbf{x})).$$

The parameters (θ, ϕ) are then optimized by maximizing the log-likelihood of the observed data under the model.

3 Loss Function

To train the model, we minimize the negative log-likelihood of the data under the Gaussian assumption. For a single data point $(\mathbf{x}_i, \mathbf{y}_i)$, the negative log-likelihood is given by

$$\mathcal{L}_i(\theta, \phi) = \frac{1}{2} \left[\log\det \Sigma_\phi(\mathbf{x}_i) + (\mathbf{y}_i - f_\theta(\mathbf{x}_i))^\top \Sigma_\phi(\mathbf{x}_i)^{-1} (\mathbf{y}_i - f_\theta(\mathbf{x}_i)) \right] + \frac{d_y}{2} \log(2\pi).$$

Aggregating over the dataset, we obtain the total loss

$$\mathcal{L}(\theta, \phi) = \frac{1}{n} \sum_{i=1}^n \mathcal{L}_i(\theta, \phi).$$

In practice, the constant term $(d_y/2)\log(2\pi)$ may be dropped, as it does not affect the optimization. The first term, $\log\det\Sigma_\phi(\mathbf{x}_i)$, penalizes model uncertainty that is too small (to avoid overconfident predictions), while the second term quantifies the Mahalanobis distance between the predicted and observed outputs that incorprate the covariance matrix between observed outcomes.

References