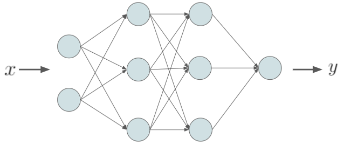
Introduction to Machine Learning

Examples of Hypothesis Spaces



Learning goals

- Recall that the hypothesis space is the set of all admissible functions
- Know the hypothesis space for: linear regression, separating hyperplanes, decision trees, ensembles, and neural networks

HYPOTHESIS SPACES

Recall, the **hypothesis space** is the set of all admissible functions, that we have to pick a certain element from during learning / risk minimization.

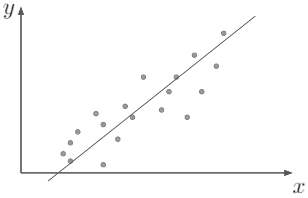
$$\mathcal{H} := \{f : \mathcal{X} \to \mathbb{R}^g \mid f \text{ has a specific form}\}.$$

Often f is parameterized by $\boldsymbol{\theta} \in \Theta$. We write $f(\mathbf{x}) = f(\mathbf{x} \mid \boldsymbol{\theta})$.

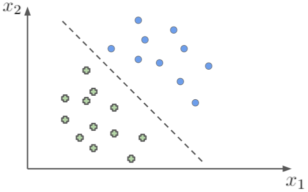
Note: If we are explicitly talking about hard classifiers outputting a discrete class, we write h instead of f.

LINEAR MODELS

- **Linear regression**: $f(\mathbf{x} \mid \theta_0, \boldsymbol{\theta}) = \mathbf{x}^T\boldsymbol{\theta} + \theta_0$ with $\boldsymbol{\theta} \in \mathbb{R}^p, \theta_0 \in \mathbb{R}$

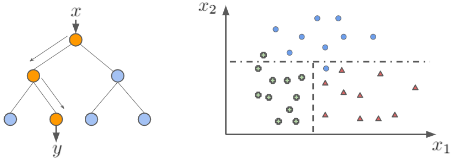


- **Separating hyperplanes**: $h(\mathbf{x} \mid \theta_0, \boldsymbol{\theta}) = \mathbb{I}[\mathbf{x}^T\boldsymbol{\theta} - \theta_0 > 0]$

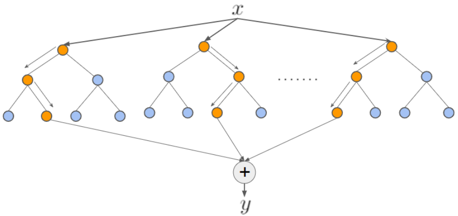


TREES AND ENSEMBLES

- **Decision trees**: $f(\mathbf{x}) = \sum_{i=1}^{m} c_i \mathbb{I}(\mathbf{x} \in Q_i)$ Where the Q_i are an axis-aligned, rectangular partitioning of the input space

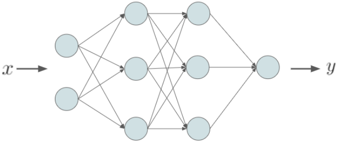


- **Simple Ensembles**: $f(\mathbf{x} \mid \beta^{[j]}) = \sum_{j=1}^{m} \beta^{[j]} b^{[j]}(\mathbf{x})$ Where the $b^{[j]}(\mathbf{x})$ come from some base learner space $\mathcal{B}$.



NEURAL NETWORKS

$$f(\mathbf{x}) = \tau \circ \phi \circ \sigma^{(h)} \circ \phi^{(h)} \circ \sigma^{(h-1)} \circ \phi^{(h-1)} \circ \ldots \circ \sigma^{(1)} \circ \phi^{(1)}(\mathbf{x})$$



- Consists of layers of simple computational "neurons"
- Each neuron in a given layer performs a two-step computation: an affine linear transformation of its inputs, then a scalar non-linear activation. We can write this in vector notation for a complete layer:

  $$\phi^{(j)}(\boldsymbol{z}) = \boldsymbol{W}_j^\top \boldsymbol{z} + \boldsymbol{b}_j$$

  The activation $\sigma^{(j)}$ applies componentwise the same non-linear function to its vector inputs (e.g. logistic or ReLU), while τ simply rescales for the final output (e.g. softmax). Usually, σ and τ contain no learnable parameters.