I2ML :: BASICS

Data

$\mathcal{X} \subset \mathbb{R}^p$: p-dimensional **feature / input space**
Usually we assume $\mathcal{X} \equiv \mathbb{R}^p$, but sometimes, dimensions may be bounded (e.g., for categorical or non-negative features.)

$\mathcal{Y} \subset \mathbb{R}^g$: **target space**
e.g.: $\mathcal{Y} = \mathbb{R}$, $\mathcal{Y} = \{0,1\}$, $\mathcal{Y} = \{-1,+1\}$, $\mathcal{Y} = \{1,\dots,g\}$ with g classes

$\mathbf{x} = (x_1,\dots,x_p)^T \in \mathcal{X}$: **feature vector**

$y \in \mathcal{Y}$: **target / label / output**

$\mathbb{D}_n = (\mathcal{X} \times \mathcal{Y})^n \subset \mathbb{D}$: **set of all finite data sets of size** n

$\mathbb{D} = \bigcup_{n \in \mathbb{N}} (\mathcal{X} \times \mathcal{Y})^n$: **set of all finite data sets**

$\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \dots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right) \in \mathbb{D}_n$: **data set** with n observations

$\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}} \subset \mathcal{D}$: **data for training and testing**
(often: $\mathcal{D} = \mathcal{D}_{\text{train}} \,\dot\cup\, \mathcal{D}_{\text{test}}$)

$\left(\mathbf{x}^{(i)}, y^{(i)}\right) \in \mathcal{X} \times \mathcal{Y}$: i -th **observation** or **instance**

$\mathbb{P}_{xy}$: **joint probability distribution on** $\mathcal{X} \times \mathcal{Y}$

$\pi_k = \mathbb{P}(y = k)$: **prior probability** for class k
In case of binary labels we might abbreviate: $\pi = \mathbb{P}(y = 1)$.

Model and Learner

Model / hypothesis: $f : \mathcal{X} \to \mathbb{R}^g$, $\mathbf{x} \mapsto f(\mathbf{x})$ is a function that maps feature vectors to predictions, often parametrized by $\boldsymbol{\theta} \in \Theta$ (then we write $f_{\boldsymbol{\theta}}$, or, equivalently, $f(\mathbf{x} \mid \boldsymbol{\theta})$).

$\Theta \subset \mathbb{R}^d$: **parameter space**

$\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_d) \in \Theta$: model **parameters**
Some models may traditionally use different symbols.

$\mathcal{H} = \{f : \mathcal{X} \to \mathbb{R}^g \mid f \text{ belongs to a certain functional family}\}$:
hypothesis space – set of functions defining a specific model class to which we restrict our
learning task

Learner $\mathcal{I} : \mathbb{D} \times \boldsymbol{\Lambda} \to \mathcal{H}$ takes a training set $\mathcal{D}_{\text{train}} \in \mathbb{D}$ and produces a model $f : \mathcal{X} \to \mathbb{R}^g$, its hyperparameters set to $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$.
For a parametrized model this can be adapted to $\mathcal{I} : \mathbb{D} \times \boldsymbol{\Lambda} \to \Theta$

$\boldsymbol{\Lambda} = \boldsymbol{\Lambda}_1 \times \boldsymbol{\Lambda}_2 \times ... \times \boldsymbol{\Lambda}_\ell \subset \mathbb{R}^\ell$, where $\boldsymbol{\Lambda}_j = (a_j, b_j)$, $a_j, b_j \in \mathbb{R}$, $j = 1, 2, ..., \ell$: **hyperparameter space**

$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_\ell) \in \boldsymbol{\Lambda}$: **model hyperparameters**

$\pi_k(\mathbf{x}) = \mathbb{P}(y = k \mid \mathbf{x}) \in [0,1]$: **posterior probability** for class k, given $\mathbf{x}$
In case of binary labels we might abbreviate: $\pi(\mathbf{x}) = \mathbb{P}(y = 1 \mid \mathbf{x})$.

$h(\mathbf{x}) : \mathbb{R}^g \to \mathcal{Y}$: **prediction function** for classification that maps class scores / posterior probabilities to discrete classes

$\epsilon = y - f(\mathbf{x})$ or $\epsilon^{(i)} = y^{(i)} - f\left(\mathbf{x}^{(i)}\right)$: (i-th) **residual** in regression

$yf(\mathbf{x})$ or $y^{(i)}f\left(\mathbf{x}^{(i)}\right)$: **margin** for (i-th) observation in binary classification (with $\mathcal{Y} = \{-1,1\}$).

$\hat{y}$, $\hat{f}$, $\hat{h}$, $\hat{\pi}_k(\mathbf{x})$, $\hat{\pi}(\mathbf{x})$ and $\hat{\boldsymbol{\theta}}$
The hat symbol denotes **learned** functions and parameters.

Loss and Risk

$L : \mathcal{Y} \times \mathbb{R}^g \to \mathbb{R}$: **loss function**
$L(y, f(\mathbf{x}))$ quantifies the "quality" of the prediction $f(\mathbf{x})$ for a single observation $\mathbf{x}$.

$\mathcal{R}_{\text{emp}} : \mathcal{H} \to \mathbb{R}$: **empirical risk**
The ability of a model f to reproduce the association between $\mathbf{x}$ and y that is present in the data $\mathcal{D}$ can be measured by the summed loss:

$$\mathcal{R}_{\text{emp}}(f) = \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right)$$

Learning then amounts to **empirical risk minimization** – figuring out which model $\hat{f}$ has the smallest summed loss.
Since f is usually defined by **parameters** $\boldsymbol{\theta}$, this becomes:

$$\hat{f} = \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})) = \arg\min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right),$$

where $\mathcal{R}_{\text{emp}} : \Theta \to \mathbb{R}$.

Components of Learning

Learning = Hypothesis space + Risk + Optimization
$= \mathcal{H} + \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) + \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$

Regression Losses

Basic idea (L2 loss / squared error):
$L(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$ or $L(y, f(\mathbf{x})) = 0.5(y - f(\mathbf{x}))^2$
Convex and differentiable
Tries to reduce large residuals (loss scaling quadratically)
$\hat{f}(\mathbf{x})$ = mean of $y|x$

Basic idea (L1 loss / absolute error):
$L(y, f(\mathbf{x})) = |y - f(\mathbf{x})|$
Convex and more robust
Non-differentiable for $y = f(\mathbf{x})$, optimization becomes harder
$\hat{f}(\mathbf{x})$ = median of $y|x$

Classification Losses

tbd

Classification

Assume we are given a **classification problem:**

$\mathbf{x} \in \mathcal{X}$	feature vector
$y \in \mathcal{Y} = \{1,\dots,g\}$	*categorical* output variable (label)
$\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \dots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right)$	observations of $\mathbf{x}$ and y

Classification usually means to construct g discriminant functions:

$f_1(\mathbf{x}), \dots, f_g(\mathbf{x})$, so that we choose our class as
$h(\mathbf{x}) = \arg\max_{k \in \{1,\dots,g\}} f_k(\mathbf{x})$

Linear Classifier: If the functions $f_k(\mathbf{x})$ can be specified as linear functions, we will call the classifier a *linear classifier*.

Binary classification: If only 2 classes exist, we can use a single discriminant function $f(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$.