

# I2ML :: BASICS

## Data

$\mathcal{X} \subseteq \mathbb{R}^p$  :  $p$ -dimensional **feature space** / input space  
Categorical features are encoded suitably, e.g., one-hot encoding

$\mathcal{Y}$  : **target space**  
e.g.:  $\mathcal{Y} = \mathbb{R}$  for regression,  $\mathcal{Y} = \{0, 1\}$  or  $\mathcal{Y} = \{-1, +1\}$  for binary classification,  $\mathcal{Y} = \{1, \dots, g\}$  for multi-class classification with  $g$  classes

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

$y \in \mathcal{Y}$  : **target variable** / output variable  
Concrete variable samples are called labels

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

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

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

$\mathcal{D} = ((\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(n)}, y^{(n)})) \in \mathbb{D}_n$  : **data set** with  $n$  observations. This is an  $n$ -tuple, i.e., a family indexed by  $\{1, \dots, n\}$ .  
We write  $\mathcal{D}_n$  if we want to emphasize the size of the data set

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

$o_k^{(i)} = (\overbrace{0, 0, \dots}^{k-1}, \overbrace{1, 0, 0, \dots}^{n-k}) \in \{0, 1\}^n$  : **one-hot encoding** for  $i$ -th observation, belonging to class  $k$  (egtl. brauch ich ne matrix?)

$\mathbb{P}_{xy}$  : **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:  $f : \mathcal{X} \rightarrow \mathbb{R}^g$ ,  $\mathbf{x} \mapsto f(\mathbf{x})$  is a function that maps feature vectors to predictions, often parametrized by  $\theta \in \Theta$   
(then we write  $f_\theta$ , or, equivalently,  $f_\theta(\mathbf{x}|\theta)$ ).

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

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

$\mathcal{H} = \{f : \mathcal{X} \rightarrow \mathbb{R}^g \mid f \text{ belongs to a certain functional family} \}$  :  
**Hypothesis space** – set of functions to which we restrict learning

**Learner** / Inducer  $\mathcal{I} : \mathbb{D} \times \Lambda \rightarrow \mathcal{H}$  takes a training set  $\mathcal{D}_{\text{train}} \in \mathbb{D}$ , produces model  $f : \mathcal{X} \rightarrow \mathbb{R}^g$ , with hyperparam. configuration  $\lambda \in \Lambda$ .  
For a parametrized model this can be adapted to  $\mathcal{I} : \mathbb{D} \times \Lambda \rightarrow \Theta$   
Alternative notation:  $\mathcal{I}_\lambda : \mathbb{D} \rightarrow \Theta$

$\Lambda = \Lambda_1 \times \Lambda_2 \times \dots \times \Lambda_\ell \subseteq \mathbb{R}^\ell$ : **hyperparameter space**  
 $\Lambda_j$  can e.g., be  $\mathbb{R}$ , intervals in  $\mathbb{R}$  or intervals in  $\mathbb{N}$

$\lambda = (\lambda_1, \lambda_2, \dots, \lambda_\ell) \in \Lambda$  : **hyperparameter configuration**

$\pi_k(\mathbf{x}) : \mathcal{X} \rightarrow [0, 1]$  **probability prediction** for class  $k$ , shall approximate  $\mathbb{P}(y = k \mid \mathbf{x})$  (in a binary case we might abbreviate to  $\pi(\mathbf{x})$ , approximating  $\mathbb{P}(y = 1 \mid \mathbf{x})$ ).

$f_k(\mathbf{x}) : \mathcal{X} \rightarrow \mathbb{R}$ : **scoring** / discriminant **function** for class  $k$  (in a binary case we might abbreviate to  $f(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$ )

$h(\mathbf{x}) : \mathbb{R}^g \rightarrow \mathcal{Y}$  : **hard label function** for classification that maps class scores / probabilities to discrete classes. Typically created by  $h(\mathbf{x}) = \arg \max_{k \in \{1, \dots, g\}} f_k(\mathbf{x})$  or  $h(\mathbf{x}) = \arg \max_{k \in \{1, \dots, g\}} \pi_k(\mathbf{x})$

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

$yf(\mathbf{x})$  or  $y^{(i)}f(\mathbf{x}^{(i)})$ : **margin** for ( $i$ -th) observation in binary classification

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

## Loss and Risk

$L : \mathcal{Y} \times \mathbb{R}^g \rightarrow \mathbb{R}_0^+$  : **loss function**: Quantifies "quality"  $L(y, f(\mathbf{x}))$  of prediction  $f(\mathbf{x})$  (or  $L(y, \pi(\mathbf{x}))$  of prediction  $\pi(\mathbf{x})$ ) for true  $y$ .

$\mathcal{R}_{\text{emp}} : \mathcal{H} \rightarrow \mathbb{R}$  : **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:

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

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**  $\theta$ , this becomes:

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

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

## Regression Losses

**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)  
► Optimal constant model:  $\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n y^{(i)} = \bar{y}$   
► **Optimal model for unrestricted  $\mathcal{H}$ :**  $\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \mathbb{E}(y^{(i)} \mid \mathbf{x}^{(i)})$

**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  
► Optimal constant model:  $\hat{f}(\mathbf{x}) = \text{med}(y^{(i)})$   
► **Optimal model for unrestricted  $\mathcal{H}$ :**  $\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \text{med}(y^{(i)} \mid \mathbf{x}^{(i)})$

## Classification Losses

**Brier score (binary case)**  
 $L(y, \pi(\mathbf{x})) = (\pi(\mathbf{x}) - y)^2$  for  $\mathcal{Y} = \{0, 1\}$

**Log-loss / Bernoulli loss / binomial loss (binary case)**  
For  $\mathcal{Y} = \{0, 1\}$ :  $L(y, \pi(\mathbf{x})) = -y \log(\pi(\mathbf{x})) - (1 - y) \log(1 - \pi(\mathbf{x}))$   
For  $\mathcal{Y} = \{-1, +1\}$ :  $L(y, \pi(\mathbf{x})) = \log(1 + (\frac{\pi(\mathbf{x})}{1 - \pi(\mathbf{x})})^{-y})$   
**Assuming a logit-link  $\pi(\mathbf{x}) = \exp(f(\mathbf{x})) / (1 + \exp(f(\mathbf{x})))$ :**  
For  $\mathcal{Y} = \{0, 1\}$ :  $L(y, f(\mathbf{x})) = -y \cdot f(\mathbf{x}) + \log(1 + \exp(f(\mathbf{x})))$   
For  $\mathcal{Y} = \{-1, +1\}$ :  $L(y, f(\mathbf{x})) = \log(1 + \exp(-y \cdot f(\mathbf{x})))$

**0-1-loss (binary case)**  
 $L(y, h(\mathbf{x})) = \mathbb{I}(y \neq h(\mathbf{x}))$

**Brier score (multi-class case)**  
 $L(y, \pi(\mathbf{x})) = \sum_{k=1}^g (\pi_k(\mathbf{x}) - o_k)^2$

**Log-loss (multi-class case)**  
 $L(y, \pi(\mathbf{x})) = - \sum_{k=1}^g o_k \log(\pi_k(\mathbf{x}))$

## Components of Learning

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