

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**

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

$\mathbb{D}_n = (\mathcal{X} \times \mathcal{Y})^n \subset \mathbb{D}$: **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

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

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

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

$p(\mathbf{x}, y) : \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$ (or $p(\mathbf{x}, y \mid \boldsymbol{\theta})$) : **joint probability density function (pdf)**, often parametrized by $\boldsymbol{\theta} \in \Theta$

Model and Learner

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

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

$\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_d) \in \Theta$: model **parameters**
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 defining a specific model class to which we restrict our learning task.

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

$\Lambda \subset \mathbb{R}^{foo}$: **hyperparameter space**

$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_{foo}) \in \Lambda$: **hyperparameter vector**

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

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

$\mathcal{L}(\boldsymbol{\theta})$ and $\ell(\boldsymbol{\theta}) = \log(\mathcal{L}(\boldsymbol{\theta}))$: likelihood and log-likelihood for parameter $\boldsymbol{\theta}$
These are based on a statistical model.

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

$y f(\mathbf{x})$ or $y^{(i)} f(\mathbf{x}^{(i)})$: **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

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

$\mathcal{R}_{\text{emp}} : \mathcal{H} \rightarrow \mathbb{R}$: 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, the **empirical risk** :

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

Since f is usually defined by **parameters** $\boldsymbol{\theta}$, this becomes:

$$\mathcal{R}_{\text{emp}} : \mathbb{R}^d \rightarrow \mathbb{R}$$
$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^n L(y^{(i)}, f(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}))$$

Learning then amounts to **empirical risk minimization** – figuring out which model f has the smallest summed loss:

$$\hat{f} = \arg \min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}).$$

Components of Learning

Learning = Hypothesis space + Risk + Optimization.

Hypothesis space : Defines (and restricts!) what kind of model f can be learned from the data.
Examples: linear functions, decision trees

Risk: Quantifies how well a model performs on a given data set. This allows us to rank candidate models in order to choose the best one.
Examples: squared error, negative (log-)likelihood

Optimization: Defines how to search for the best model, i.e., the model with the smallest risk, in the hypothesis space.
Examples: gradient descent, quadratic programming

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)
- Optimal constant model: $\hat{f}(\mathbf{x}) = \text{mean of } y \mid \mathbf{x}$

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
- Optimal constant model: $\hat{f}(\mathbf{x}) = \text{median of } y \mid \mathbf{x}$

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)
 $\mathcal{D} = ((\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(n)}, y^{(n)}))$ 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 $f(\mathbf{x}) = f_1(\mathbf{x}) - f_2(\mathbf{x})$.