Kind reminder about ...

Unconstrained optimization

Let $f : \mathbb{R}^p \to \mathbb{R} : \boldsymbol{x} \mapsto f(\boldsymbol{x})$ be a differentiable function, the optimization problem of minimizing f over its domain is usually written as

$$\min_{\boldsymbol{x} \in \mathbb{R}^p} f(\boldsymbol{x})$$

For this setting, any local minimizer $\boldsymbol{x}^*$ of f satisfies the following equality

$$\nabla f(\boldsymbol{x}^*) = 0 \tag{1}$$

where, for every $\boldsymbol{x} \in \mathbb{R}^p$,

$$\nabla f(\boldsymbol{x}) = \begin{pmatrix} \frac{\partial f(\boldsymbol{x})}{\partial x_1} \\ \frac{\partial f(\boldsymbol{x})}{\partial x_2} \\ \vdots \\ \frac{\partial f(\boldsymbol{x})}{\partial x_p} \end{pmatrix}$$

Note: if the objective function, f, is *convex*, any local optimum is also global, and eq. (1) is a necessary and sufficient condition for the global optimum. Well-known convex functions include affine functions and norms.

Norms

Let $\boldsymbol{x} \in \mathbb{R}^p$, the Euclidean norm (or ℓ_2-norm) is computed as

$$\|\boldsymbol{x}\|_2 = \sqrt{\boldsymbol{x}^T \boldsymbol{x}} = \sqrt{\sum_{i=1}^{p} x_i^2}.$$

Given a data matrix $\boldsymbol{A} \in \mathbb{R}^{N \times p}$ and vector $\boldsymbol{y} \in \mathbb{R}^N$, we may be interested to minimize the residual

$$\min_{\beta \in \mathbb{R}^p} \left\| \boldsymbol{A}\beta - \boldsymbol{y} \right\|_2 \tag{2}$$

As the *square function*, is continuous monotonically increasing, we can solve instead

$$\min_{\beta \in \mathbb{R}^p} \left\| \boldsymbol{A}\beta - \boldsymbol{y} \right\|_2^2 \tag{3}$$

and both minimizers of (2) and (3) are identical.

Polynomial approximation

Assume we dispose of a set of N points $(x, y) \in \mathbb{R}^2$, namely $\mathcal{T} := \{(x_i, y_i)\}_{i=1}^N$, we are interested in finding the best polynomial of degree p which fits the data the best, in the *least-squares* sense.

This consists in minimizing the sum of the squared residuals:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^{p+1}} \sum_{i=0}^{N} \left((\hat{f}(x_i; \boldsymbol{\beta}) - y_i \right)^2 \tag{4}$$

where the image of our polynomial model of degree p parametrized by $\boldsymbol{\beta} = (\beta_0, \beta_1, \cdots, \beta_p)^\top$ is given by

$$\hat{f}(x; \boldsymbol{\beta}) := \beta_0 + \beta_1 x + \ldots + \beta_p x^p = \sum_{k=0}^{p} \beta_k x^k$$

Since $\hat{f}(x; \boldsymbol{\beta})$ takes the form of a matrix product,

$$\hat{f}(x; \boldsymbol{\beta}) = \left(1, x, x^2, \cdots, x^p\right) \cdot \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix},$$

the approximation problem (4) can be reformulated as a linear inverse problem:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^{p+1}} \left\|\boldsymbol{V}\boldsymbol{\beta}-\boldsymbol{y}\right\|_2^2,$$

with $\boldsymbol{V}\in\mathbb{R}^{N\times(p+1)}$, the so-called *Vandermonde* matrix, defined as

$$\boldsymbol{V} := \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^d \\ 1 & x_2 & x_2^2 & \dots & x_2^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \dots & x_N^d \end{pmatrix}.$$

Accuracy, precision and recall, confusion matrix

Let us consider a random vector $(\boldsymbol{X}, Y)\in\mathbb{R}^p\times\mathcal{Y}$, with $\mathcal{Y}$ either $\mathbb{R}$ (for regression) or a finite set $\mathcal{Y}:=\{c_1,\dots,c_L\}$ (for classification).

Accuracy

Following the course notations, let us consider a *classifier* $\hat{G}$ mapping $\mathbb{R}^p$ to $\mathcal{Y}$, *i.e.*,

$$\hat{G}:\mathbb{R}^p\to\mathcal{Y},\ \boldsymbol{x}\to\hat{G}(\boldsymbol{x}).$$

The *overall accuracy* of $\hat{G}$ on $\mathbb{R}^p\times\mathcal{Y}$ is defined as the proportion of samples from $\mathbb{R}^p\times\mathcal{Y}$ for which it correctly predicts the target, *i.e.*, the proportion of observations $(\boldsymbol{x}, y)$ drawn from $(\boldsymbol{X}, Y)$, for which

$$\hat{G}(\boldsymbol{x}) = y.$$

Mathematically, it corresponds to the value $\mathrm{Acc}(\hat{G})\in[0,1]$, defined as

$$\mathrm{Acc}(\hat{G}) := \mathbb{P}\big(\hat{G}(\boldsymbol{X}) = Y\big).$$

The *empirical accuracy*, also called *training accuracy*, of a classifier $\hat{G}$ on a data set $\mathcal{T}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N\subseteq\mathbb{R}^p\times\mathcal{Y}$ is computed as the fraction of correctly classified training samples

$$\mathrm{Acc}^{\mathrm{emp}}(\mathcal{T}) := \tfrac{1}{N}\sum_{i=1}^{N} I\big(y_i=\hat{G}(\boldsymbol{x}_i)\big),$$

where $I(\boldsymbol{x})=1$ if $\boldsymbol{x}$ is true, 0 otherwise.

If N is large enough, the *overall accuracy* is often estimated thanks to the *empirical accuracy*.

Precision and recall

Consider a classification problem with two classes, *i.e.*, $\mathcal{Y}=\{c_1,c_2\}=\{False,\ True\}$. This problem is also called a binary problem.

The *precision* is the proportion of samples rightfully assigned to true among all samples classified as true, whereas the *recall* is the proportion of samples rightfully assigned to true among all true samples.

The *overall precision* and *recall* are computed, respectively, as

$$\mathrm{Pre}(\hat{G}) := \frac{\mathbb{P}(Y=True\ \&\ \hat{G}(\boldsymbol{X})=True)}{\mathbb{P}(\hat{G}(\boldsymbol{x})=True)} = \mathbb{P}(Y=True\mid\hat{G}(\boldsymbol{X})=True), \qquad \text{(precision)}$$

$$\mathrm{Rec}(\hat{G}) := \frac{\mathbb{P}(\hat{G}(\boldsymbol{X})=True\ \&\ Y=True)}{\mathbb{P}(Y=True)} = \mathbb{P}(\hat{G}(\boldsymbol{X})=True\mid Y=True), \qquad \text{(recall)}$$

The *empirical* counterparts are given by the relationships:

$$\mathrm{Pre}^{\mathrm{emp}}(\hat{G}) := \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \qquad \text{(empirical precision)}$$

$$\mathrm{Rec}^{\mathrm{emp}}(\hat{G}) := \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \qquad \text{(empirical recall)}$$

with the *true positive*, *false negative*, and *false positive* ratios as

$$\begin{aligned}
\mathrm{TP} &= \sum_{\{(\boldsymbol{x},True)\}\in\mathcal{T}} I(\hat{G}(\boldsymbol{x}) = True),\\
\mathrm{FN} &= \sum_{\{(\boldsymbol{x},True)\}\in\mathcal{T}} I(\hat{G}(\boldsymbol{x}) = False),\\
\mathrm{FP} &= \sum_{\{(\boldsymbol{x},False)\}\in\mathcal{T}} I(\hat{G}(\boldsymbol{x}) = True).
\end{aligned}$$

Confusion matrix

The *confusion matrix* of a classifier $\hat{G}$ based on a set $\mathcal{T}$ as written above is the matrix $\mathcal{C}(\hat{G}) \in \mathbb{N}^{L\times L}$ defined by

$$\mathcal{C}_{i,j}(\hat{G}) := \sum_{\{(x,y)\}\in\mathcal{T}} I(y = c_i,\, \hat{G}(\boldsymbol{x}) = c_j), \quad 1 \le i,j \le L,$$

In other words, $C_{i,j}(\hat{G})$ is the number of elements whose true class is c_i that are classified in class c_j.
Note that in binary case, the confusion matrix becomes

$$C = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix}.$$

Therefore, it contains all necessary information to compute precision and recall!