Simple validation, cross-validation and leave-one-out

Estimating the risk directly from the data

MATH-412 - Statistical Machine Learning

Simple validation

Can we use the data to obtain an unbiased estimate of the risk of a learnt decision function ?

Simple validation

1. Split the original data set D in a new training set L and a validation set V.

$$L = \{(x_1, y_1), \ldots, (x_{n'}, y_{n'})\} \quad \text{and} \quad V = \{(x_{n'+1}, y_{n'+1}), \ldots, (x_n, y_n)\}$$

2. Learn a decision function $\widehat{f}_L$ using only L
3. Estimate the risk with the validation set V

$$\widehat{\mathcal{R}}_V^{\text{val}}(\widehat{f}_L) = \frac{1}{|V|} \sum_{i \in V} \ell\Big(\widehat{f}_L(x_i), y_i \Big)$$

We have $\mathbb{E}[\widehat{\mathcal{R}}_V^{\text{val}}(\widehat{f}_L) | L] = \mathcal{R}(\widehat{f}_L)$, so that $\widehat{\mathcal{R}}_V^{\text{val}}(\widehat{f}_L)$ is an unbiased estimator of $\mathcal{R}(\widehat{f}_L)$.

K-fold cross-validation

Partition D in blocks of (almost) equal size :

B_1	B_2	B_3	V	B_5

For each block

- Use the block $V = B_k$ as validation data and the rest $L = D \backslash B_k$ as training set.
- Estimate the validation error

$$\widehat{\mathcal{R}}^{\text{val}}_{B_k}(\widehat{f}_{D\backslash B_k}) = \frac{1}{|B_k|} \sum_{i \in B_k} \ell(\widehat{f}_{D\backslash B_k}(x_i), y_i).$$

Then compute the CV risk estimate as the average $\widehat{\mathcal{R}}^{K-\text{fold}} = \frac{1}{K} \sum_{k=1}^{K} \widehat{\mathcal{R}}^{\text{val}}_{B_k}(\widehat{f}_{D\backslash B_k}).$

Note that we have

$$\mathbb{E}[\widehat{\mathcal{R}}^{K-\text{fold}}] = \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}[\mathcal{R}(\widehat{f}_{D\backslash B_k})] \approx \mathbb{E}[\mathcal{R}(\widehat{f}_{n'})]$$

where $n' = n - |B_1|$ if $\lfloor n/K \rfloor \leq |B_k| \leq \lceil n/K \rceil$ and $\widehat{f}_{n'}$ is a decision function trained with a subset of size n' of D.

Leave-one-out cross validation

- Consists in removing a single point from the training set at a time and use it for validation

$$L = D_{-i} = \{(x_1,y_1),\dots,(x_{i-1},y_{i-1}),(x_{i+1},y_{i+1}),\dots,(x_n,y_n)\} \quad \text{and} \quad V = \{(x_i,y_i)\}$$

- ... and to average over the choice of that point :

$$\widehat{\mathcal{R}}^{LOO} = \frac{1}{n}\sum_{i=1}^{n} \widehat{\mathcal{R}}^{\text{val}}_{\{(x_i,y_i)\}}(\widehat{f}_{D_{-i}}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\widehat{f}_{D_{-i}}(x_i), y_i).$$

- The LOO error can sometimes be computed in closed form.
  E.g. for the ordinary least square linear regression estimate $\widehat{\boldsymbol{w}} = (\boldsymbol{X}^\top\boldsymbol{X})^\dagger\boldsymbol{X}\boldsymbol{y}$.

$$\widehat{\mathcal{R}}^{LOO}(\widehat{\boldsymbol{w}}) = \frac{1}{n}\sum_{i=1}^{n}\frac{(\widehat{\boldsymbol{w}}^\top\mathbf{x}_i - y_i)^2}{(1-h_{ii})^2} \qquad \text{with} \quad h_{ii} = \boldsymbol{H}_{ii} = \mathbf{x}_i^\top(\boldsymbol{X}^\top\boldsymbol{X})^\dagger\mathbf{x}_i.$$

h_{ii} is called the i^{th} leverage score.

(Cross)-Validation for hyperparameter & model selection

Let $\big(\widehat{f}^{(\lambda)}_{D\setminus B_k}\big)_k$ the CV decision functions all learned with the hyperparameter(s) λ.

An optimal hyperparameter is estimated via

$$\widehat{\lambda}_{\text{CV}} = \arg\min_{\lambda} \widehat{\mathcal{R}}^{K-\text{fold}}(\lambda) \qquad \text{with} \qquad \widehat{\mathcal{R}}^{K-\text{fold}}(\lambda) = \frac{1}{K}\sum_{k=1}^{K} \widehat{\mathcal{R}}^{\text{val}}_{B_k}\big(\widehat{f}^{(\lambda)}_{D\setminus B_k}\big).$$

- In practice, this optimization in often done via grid search because the objective is noisy and thus typically locally non-smooth and non-convex.
- For *regularization coefficients*, grids uniform on the log-scale are recommended : e.g., $\log_{10}(\lambda) \in \{-6, -5.5, \dots, 1.5, 2\}$.
- This is can be done similarly with simple validation and LOOCV.

Comments on cross-validation

How to choose K ?

- Difficult theoretical problem
- In practice $K = 5$ or $K = 10$.

Performance of the *decision function* $\widehat{f}$ *vs* performance of the *learning scheme* $\mathscr{A}$

Two natural questions :

- How well will my *decision function* $\widehat{f}$ perform on future data ?

$$\mathcal{R}(\widehat{f}) \quad \rightarrow \quad \text{simple validation / LOO}$$

- If $\widehat{f}_D = \mathscr{A}(D)$, how well does my *learning scheme* $\mathscr{A}$ perform ?

$$\mathbb{E}_D\big[\mathcal{R}(\widehat{f}_D)\big] \quad \rightarrow \quad \text{cross validation}$$

- However, even in the perspective of producing a single decision function, for hyperparameter optimization or model selection, cross-validation will be more robust than simple validation.

Final decision function

How to build a final decision function given $\widehat{\lambda}_{\text{CV}} = \arg\min_{\lambda} \frac{1}{K} \sum_{k=1}^{K} \widehat{\mathcal{R}}^{\text{val}}_{B_k}(\widehat{f}^{(\lambda)}_{D \setminus B_k}) \quad ?$

Solution 1 : Retrain. $\widehat{f} = \widehat{f}_D^{(\widehat{\lambda}_{\text{CV}})}$ re-learned with all of the data D.

- PRO : A single decision function from all the data.
- CON : $\widehat{\lambda}_{\text{CV}}$ is optimized for other decision functions and for a sample size of $n' = |D \setminus B_k| < n$.

$\Rightarrow$ Appropriate for LOOCV and large K (i.e., $|B_k|$ small).

Solution 2 : Ensembling. $\widehat{f} = \frac{1}{K} \sum_k \widehat{f}_{D \setminus B_k}$ is just the average of the fold decision functions.

- PROs : No retraining + if the risk $\mathcal{R}$ is convex then $\mathcal{R}(\widehat{f}) \leq \frac{1}{K} \sum_k \mathcal{R}(\widehat{f}_{D \setminus B_k})$ which is precisely estimated by $\widehat{\mathcal{R}}^{K-\text{fold}}$.
- CON : Requires several decision functions at test time (unless they are linear in the parameters in which case one just needs to average the parameters).

Nested-cross validation

If the number and/or dimensions of the hyperparameters is large, or if many models are considered, overfitting at the validation level (e.g. in CV) is possible.

It becomes necessary to keep a test set for final evaluation.

Simple validation : Training (e.g. 80%) + Validation (e.g. 10%) + Test (e.g. 10%)

Cross-validation with simple test : The data set D is split into a CV set C and a test set T

Nested CV : Use multiple splits to have $D = C_k \cup T_k$ and apply CV to each C_k.

Data imbalance in classification : Proportions of each class should be kept in all sets.

Remark on time series data :

- It is fine to have dependence *within* each Training, Validation or Test set.
- There should be **no** dependence *across* these sets. This requires to throw away *buffer data* at the interface between these sets.

Summary and additional remarks

- Simple validation is sufficient if a lot of data is available, and the only option if the data distribution drifts over time (the validation/test sets have to be the most recent data)
- Cross-validation remains the most standard procedure for small data sets ($n < 500$) especially if the number of parameters is large compared to n.
- LOO is often too computationally expensive but recommended if it is closed form and the goal is evaluate a single decision function $\widehat{f}$ (vs not the learning scheme $\mathscr{A}$)
- A separate test set is needed if many hyperparameters/models are optimized/selected.