Generalisation in theory and practice

Christos Dimitrakakis

September 20, 2024

Outline

Classification

The classifier as a decision rule

A decision rule $\pi(a|x)$ generates a decision $a \in [m]$. It is the conditional probability of a given x.

A note on conditional probabilities

Even though normally conditional probabilities are defined as $P(A|B) = P(A \cap B)/P(B)$, the probability of the decision a is undefined without a given x. So it's better to think if $\pi(a|x)$ as a collection of distributions on a, one for each value of x.

Deterministic predictions given a model $P(y|x)$

Here, we pick the most likely class:

$$\pi(a|x_t) = \mathbb{I}\left\{a = \arg\max_y P(y|x_t)\right\}$$

Deterministic predictions given a model $P(y|x)$

Here, we randomly select a class according to our model:

$$\pi(a|x_t) = P(y_t = a|x_t)$$

Accuracy as a classification metric

The accuracy of a single decision

$$U(a_t, y_t) = \mathbb{I}\{a_t = y_t\} = \begin{cases} 1, & \text{if } a_t = y_t \\ 0, & \text{otherwise} \end{cases}$$

The accuracy on the training set

$$U(\pi, D) \triangleq \frac{1}{T} \sum_{t=1}^{T} \sum_{a=1}^{m} \pi(y_t | x_t)$$

The expected accuracy of a decision rule

If $(x, y) \sim P$, the accuracy U of a stochastic decision rule π under the distribution P is the probability it predicts correctly

$$U(\pi, P) \triangleq \int_X dP(x) \sum_{y=1}^{m} P(y|x) \pi(y|x)$$

Regression

The regressor as a decision rule

A decision rule π generates a decision $a \in \mathbb{R}^m$.

- For randomised rules, $\pi(a|x)$ is the conditional density of a given x.
- For deterministic rules $\pi(x)$ is the prediction for x.

Mean-Squared Error on a Dataset

The mean-square error is simply the squared difference in predicted versus actual values:

$$\frac{1}{T}\sum_{t=1}^{T}[y_t - \pi(x_t)]^2$$

Expected MSE

If $(x, y) \sim P$, the expected MSE of a deterministic decision rule $\pi : X \to \mathbb{R}$ is

$$\int_X \int_Y dP(x, y)[y - \pi(x)]^2.$$

Training and overfitting

Training data

- $D = ((x_t, y_t) : t = 1, \ldots, T)$.
- $x_t \in X$, $y_t \in Y$.

Assumption: The data is generated i.i.d.

- $(x_t, y_t) \sim P$ for all t (identical)
- $D \sim P^T$ (independent)

The optimal decision rule for P

$$\max_\pi U(\pi, P) = \max_\pi \int_{X \times Y} dP(x, y) \sum_a \pi(a|x) U(a, y)$$

The optimal decision rule for D

$$\max_\pi U(\pi, D) = \max_\pi \sum_{(x,y) \in D} \sum_a \pi(a|x) U(a, y)$$

The Train/Validation/Test methodology

Main idea
Use each piece of data once to make decisions and measure

Training set
Use to decide low-level model parameters

Validation set
Use to decide between:

- different hyperparameters (e.g. K in nearest neighbours)
- model (e.g. neural networks versus kNN)

Test set
Use to measure the final quality of a model

Cross-validation (XV)

Idea

- Use XV to select hyperparameters instead of a single train/valid test.

Methodology

- Split training set D in k different subsets
- At iteration i
- Use the i-th subset for validation
- Use all the remaining $k-1$ subsets for training
- Average results on validation sets

Bootstrapping

Idea

- How to take into account variability?
- Resample the data and repeat your calculations for each resample

Boostrap samples

- Input: Data D, of size T
- For t in $\{1, \ldots, T\}$

– Select i uniformly in $[T]$ – Add the i-th point to D_b

- Return D_b

The wrong way to do XV for subset selection

1. Screen the predictors: find a subset of "good" predictors that show fairly strong (univariate) correlation with the class labels.
2. Using just this subset of predictors, build a multivariate classifier.
3. Use cross-validation to estimate the unknown tuning parameters and to estimate the prediction error of the final model.

Is this a correct application of cross-validation?

Consider a scenario with $N = 50$ samples in two equal-sized classes, and $p = 5000$ quantitative predictors (standard Gaussian) that are independent of the class labels. The true (test) error rate of any classifier is 50%.

The right way to do XV for feature selection

1. Divide the samples into K cross-validation folds (groups) at random.
2. For each fold $k = 1, 2, \ldots, K$
3. Find a subset of “good” predictors that show fairly strong (univariate) correlation with the class labels, using all of the samples except those in fold k.
4. Using just this subset of predictors, build a multivariate classifier, using all of the samples except those in fold k.
5. Use the classifier to predict the class labels for the samples in fold k.

Learning and generalisation

How well can decision rule perform?

Estimation theory view

- Bias: The expected difference between the estimated value and the unknown parameter
- Variance: The expected difference between the estimated value and the unknown parameter

Learning theory view

- Approximation ability: How well a class of rules can approximate the optimal one.
- Statistical error: How easy it is to choose the best rule in the class.

The bias/variance trade-off

- Dataset $D \sim P$.
- Predictor $f_D(x)$
- Target function $y = f(x) + \epsilon$
- $\mathbb{E}\,\epsilon = 0$ zero-mean noise with variance $\sigma^2 = \mathbb{V}(\epsilon)$

MSE decomposition

$$\mathbb{E}[(f - f_D)^2] = \mathbb{V}(f_D) + \mathbb{B}(f_D)^2 + \sigma^2$$

Variance

How sensitive the estimator is to the data

$$\mathbb{V}(f_D) = \mathbb{E}[(f_D - \mathbb{E}(f_D))^2]$$

Bias

What is the expected deviation from the true function

$$\mathbb{B}(f_D) \triangleq \mathbb{E}[(f_D - f)]$$

Example: mean estimation

- Data $D = y_1, \ldots, y_T$ with $\mathbb{E}[y_t] = \mu$.
- Goal: estimate μ with some estimator f_D to minimise
- MSE: $\mathbb{E}[(y - f_D)^2]$, the expected square difference between new samples our guess.

Optimal estimate

To minimise the MSE, we use $f^* = \mu$. This gives us two ideas:

Empirical mean estimator:

- $f_D = \sum_{t=1}^{T} x_t / T$.
- $\mathbb{V}(f_D) = \mathbb{E}[f_D - \mu] = 1/\sqrt{T}$
- $\mathbb{B}(f_D) = 0$.

Laplace mean estimator:

- $f_D = \sum_{t=1}^{T} (\lambda + x_t) / T$.
- $\mathbb{V}(f_D) = \mathbb{E}[f_D - \mu] = \frac{1}{1+\sqrt{T}}$
- $\mathbb{B}(f_D) = O(1/T)$.

A proof of the bias/variance trade-off

- RV's $y_t \sim P$, $\mathbb{E}[y_t] = \mu$, $y_t = \mu + \epsilon_t$.
- Estimator f_D, $D = y_1, \ldots, y_{t-1}$.

$$
\begin{aligned}
\mathbb{E}[(f_D - y_t)^2] &= \mathbb{E}[f_D^2] - 2\,\mathbb{E}[f_D y_t] + \mathbb{E}[y_t^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D y_t] + \mathbb{E}[y_t^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D]\,\mathbb{E}[y_t] + \mathbb{E}[y_t^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D]\mu + \mathbb{E}[y_t^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D]\mu + \mathbb{E}[(\mu + \epsilon_t)^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D]\mu + \mathbb{E}[\mu^2 + 2\mu\epsilon_t + \epsilon_t^2] \\
&= \mathbb{V}[f_D] + \mathbb{E}[f_D]^2 - 2\,\mathbb{E}[f_D]\mu + \mu^2 + \sigma^2 \\
&= \mathbb{V}[f_D] + (\mathbb{E}[f_D] - \mu)^2 + \sigma^2 \\
&= \mathbb{V}(f_D) + \mathbb{B}(f_D)^2 + \sigma^2
\end{aligned}
$$

Generalisation error

Regret decomposition

Let the optimal rule be $\pi^* \in \Pi$, the best approximate rule be $\hat{\pi}^* \in \Pi$ and our rule be $\hat{\pi} \in \hat{\Pi}$. We call the difference between the performance of π^* and $\hat{\pi}$ our regret:

$$\underbrace{U(\pi^*, P) - U(\hat{\pi}, P)}_{\text{regret}} = \underbrace{U(\pi^*, P) - U(\hat{\pi}^*, P)}_{\text{approximation error}} + \underbrace{U(\hat{\pi}^*, P) - U(\hat{\pi}, P)}_{\text{estimation error}}$$

We can bound the regret by bounding each term separately.

- The approximation error tells us how expressive our class of rules is, i.e. how much we lose by looking at a restricted class $\hat{\Pi}$ of rules. It is similar to estimator bias.
- The statistical error tells us how well the empirical performance on D approximates the true performance. It is similar to estimator variance.
- As a rule of thumb, the larger our class, the better the possible approximation but the higher the statistical error.

Approximation error

- Our model limits us to a set of decision rules $\hat{\Pi} \subset \Pi$.
- The most we could do is find the best rule in $\hat{\Pi}$.
- This still leaves a gap:

$$\Delta \triangleq \max_{\pi \in \Pi} U(\pi, P) - \max_{\hat{\pi} \in \hat{\Pi}} U(\pi, P)$$

The gap can be characterised in some cases.

Example: ϵ-net on Lipschitz $U(\cdot, P)$.

- Assume $U(\pi, P)$ is a Lipschitz function of π for all P, i.e. $|U(\pi, P) - U(\pi', P)| \leq L d(\pi, \pi')$ for some metric d.
- Let $\hat{\Pi}$ be an ϵ-net on Π, i.e. $\max_{\pi \in \Pi} \min_{\pi' \in \hat{\Pi}} d(\pi, \pi') = \epsilon$.
- Then $\Delta \leq L\epsilon$.

Estimation error

- First, let us bound $U(\hat{\pi}^*, P) - U(\hat{\pi}, P)$ by making an assumption.
- Then, we can prove that our assumption holds with high probability.

Lemma

Let $f, g : S \to \mathbb{R}$. If $\|f - g\|_\infty \leq \epsilon$ and $f(x) \geq f(z)$, while $g(y) \geq g(z)$, for all z, i.e. x, y maximise f, g respectively

$$f(x) - f(y) \leq 2\epsilon.$$

This holds as: $f(x) - f(y) \leq g(x) + \epsilon - f(y) \leq g(y) + \epsilon - f(y) \leq 2\epsilon$.

Corollary

If $|U(\pi, P) - U(\pi, D)| \leq \epsilon$ for all π then

$$U(\hat{\pi}^*, P) - U(\hat{\pi}, P) \leq 2\epsilon$$

- Let us now prove that, with high probability, $|U(\pi, P) - U(\pi, D)| \leq \epsilon$.

Bounding the estimation error

For any fixed rule $\pi \in \Pi$ and utility function $U : \Pi \times X^T \to [0, 1]$,

$$P^T(|U(\pi, D) - U(\pi, P)| \geq \epsilon) \leq 2 \exp(-2T\epsilon^2).$$

This is a direct application of Hoeffding's inequality[1]. Taking the union bound over the set $\hat{\Pi}$ gives:

$$P^T(\exists \pi \in \hat{\Pi} : |U(\pi, D) - U(\pi, P)| \geq \epsilon) \leq 2|\hat{\Pi}| \exp(-2T\epsilon^2).$$

Setting the right side equal to δ and re-arranging,

$$P^T\left(\max_{\pi \in \hat{\Pi}} |U(\pi, D) - U(\pi, P)| \geq \sqrt{\frac{\ln(2|\hat{\Pi}|/\delta)}{2T}}\right) \leq \delta.$$

Example: ϵ-net.

In a n dimensional space we require $|\hat{\Pi}| = O(\epsilon^{-n})$. This means that our statistical error is $O(\sqrt{n \ln(1/\epsilon\delta)/T})$.

[1]See Hoeffding's inequality in the confidence intervals presentation

The finite hypothesis algorithm

- Input: a finite set of rules $\hat{\Pi}$, data D, utility U
- Return $\hat{\pi} \in \arg\max_{\pi \in \hat{\Pi}} U(\pi, D)$.

Regret of the finite hypothesis algorithm.

With probability $1 - \delta$

$$U(\hat{\pi}, P) \geq U(\hat{\pi}^*, P) - \sqrt{2 \ln(2|\hat{\Pi}|/\delta)/T} \tag{1}$$

$$U(\pi^*, P) - U(\hat{\pi}, P) \leq \Delta + \sqrt{2 \ln(2|\hat{\Pi}|/\delta)/T} \tag{2}$$

Examples

- ML estimation: $U(\theta, D) = P_\theta(D)$ is the data likelihood.
- Accuracy, etc: $U(\pi, D)$.

VC Dimension

Here we consider sets Π of deterministic rules $\pi : X \to \{0,1\}$.

Shattering

If a $S \subset X$ can with $|S| = m$, can be assigned any labelling $y_1, \ldots, y_m$ by a $\pi \in \Pi$, then we say Π shatters S.

The VC dimension

This is the largest-size set S that Π can shatter.

Example: Perceptrons on $\mathbb{R}^2$

This class has VC dimension 3 on the plane.

Binary classification

Learning algorithm λ

- Takes data $D = \{(x_t, y_t)\}$ as input
- Generates deterministic decision rules $\pi : X \to \{0, 1\}$,

The loss of a rule π.

- Assume an existing concept class $\pi^* \in \Pi$
- Distribution $x_t \sim P$ is i.i.d. and $x_1, \ldots, x_T \sim P^T$.
- The loss under distribution P is

$$L(\pi) = P(\{x : \pi(x) \neq \pi^*(x)\})$$

Realisable PAC learner

- $\lambda : (X \times Y)^* \to \Pi$ is (ϵ, δ)-PAC, if for any P and $\epsilon, \delta > 0$, and any concept $\pi^* \in \Pi$, there is T such that
$$P^T(\{D : L[\lambda(D)] > \epsilon\}) < \delta, \qquad D = (\{x_t, \pi^*(x_t)\}), x_t \sim P.$$