Generalization, Overfitting, AIC, ...

Yashar Ahmadian

February 21, 2014

First some context. In **supervised learning** in general, the goal is to learn or infer an (initially) unknown function $f : x \mapsto y$, from a set of training data in the form of T "input/output" pairs $\{(x_\mu, y_\mu)\}_{\mu=1:T}$.[1] More generally, you try to infer the conditional distribution $\rho(y|x)$ from this training set; the reason is that in general your outputs contains some noise (or stated better, trial-to-trial variability, not eliminated by controlling x), and therefore the y's are not given by a deterministic (and smooth) function of x. The $\rho(y|x)$ thus captures/formalizes the "data generating process," and I will call it that.

One simple special case is that of **additive noise**, where

$$y = f(x) + \varepsilon \tag{1}$$

and the noise, ε, is some zero-mean random variable independent of x. It is usually assumed that this noise changes "fast" so that it is independent from trial to trial, and that trials are statistically equivalent, so that ε is identically distributed in all trials, with some distribution $p_\varepsilon(\varepsilon)$. In this setting, $\rho(y|x) = p_\varepsilon(y - f(x))$. And given the iid assumption, $\rho(\{y_\mu\}_{\mu=1:T} \,|\, \{x_\mu\}_{\mu=1:T}) = \prod_{\mu=1}^{T} p_\varepsilon(y_\mu - f(x_\mu))$.

1 Linear regression

In **linear regression** (a special case of supervised learning) based on a set of N fixed/untrained (and generally nonlinear) features $\varphi_k(x)$ (for $k = 1 : N$), you model the unknown function as a linear combination of those nonlinear features, i.e. you assume

$$y(x) = \overrightarrow{\varphi}(x) \cdot \overrightarrow{w} \tag{2}$$

where $\overrightarrow{\varphi}(x) = (\varphi_1(x), \varphi_2(x), \ldots, \varphi_N(x))$, and try to learn/infer the weights $\overrightarrow{w}$ from the training set $\{(x_\mu, y_\mu)\}_{\mu=1:T}$. A standard example is polynomial curve-fitting (of degree $N-1$) in which case x is a scalar, and $\varphi_k(x) = x^{k-1}$.

In linear regression with square-loss **regularization** (aka **ridge regression**) we fit the weights by finding the $\overrightarrow{w}$ that minimizes the cost function

$$\hat{\overrightarrow{w}} := \arg\min_{\overrightarrow{w}} \left[\mathrm{RSS}(\overrightarrow{w}) + \lambda^2 \|\overrightarrow{w}\|^2\right] \tag{3}$$

$$\mathrm{RSS}(\overrightarrow{w}) := \sum_{\mu=1}^{N} (y_\mu - \overrightarrow{\varphi}(x_n) \cdot \overrightarrow{w})^2. \tag{4}$$

[1] y will be assumed a scalar throuout, but no need to assume x is a scalar. In general it's some vector, but what is important in the current discussion is not the space in which x lives in , but rather the feature space where the $\overrightarrow{\varphi}(x)$ live in (see below); that's an d dimensional space.

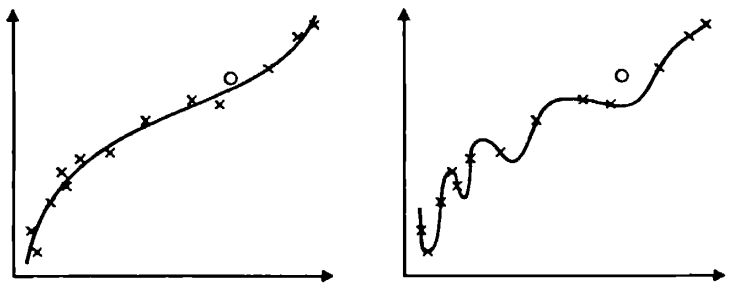

FIGURE 6 13 (a) A good fit to noisy data. (b) Overfitting of the same data: the fit is perfect on the "training set" (x's), but is likely to be poor on a "test set" represented by the circle.

Figure 1: From J. Hertz *et al.*, "Introduction to the theory of neural computation", page 147.

The second term in the cost function is called the regularization penalty. Thus when $\lambda = 0$ (i.e. no regularization), you fit by minimizing the sum of squared residuals/deviations, $\mathrm{RSS}(\overrightarrow{w})$.[2]

Defining a T dimensional vector $\mathbf{y} = (y_\mu)_{\mu=1:T}$, and a $T \times N$ "design matrix" Φ with elements

$$\Phi_{\mu,k} = \varphi_k(x_\mu), \tag{5}$$

I can rewrite $\mathrm{RSS}(\overrightarrow{w})$ as

$$\mathrm{RSS}(\overrightarrow{w}) = \|\mathbf{y} - \Phi\overrightarrow{w}\|^2, \tag{6}$$

where for any vector $\|\mathbf{v}\|^2 := \mathbf{v}^\dagger\mathbf{v} = \sum_\mu v_\mu^2$ (and $\dagger$ means the transpose).

To do the minimization in Eq. (3) you take the gradient of the quadratic cost, $\mathrm{RSS}(\overrightarrow{w}) + \frac{1}{2}\lambda^2\|\overrightarrow{w}\|^2$, and set it to zero, obtaining

$$\hat{\overrightarrow{w}} = (\Phi^\dagger\Phi + \lambda^2\mathbf{1})^{-1}\Phi^\dagger\mathbf{y}. \tag{7}$$

(In the limit $\lambda \to 0^+$ the matrix hitting $\mathbf{y}$ is nothing but the pseudo-inverse of Φ.)

For a new x, your estimate of (or prediction for) $y(x)$ is thus

$$\hat{y}(x) = \overrightarrow{\varphi}(x) \cdot \hat{\overrightarrow{w}}. \tag{8}$$

Similarly, on the old training set's x_μ's your predictions are $\hat{y}_\mu = \overrightarrow{\varphi}(x_\mu) \cdot \hat{\overrightarrow{w}}$, or in vector notation

$$\hat{\mathbf{y}} := \Phi\hat{\overrightarrow{w}} \tag{9}$$

Using Eq. (7) we can write this as

$$\hat{\mathbf{y}} = \mathcal{P}_\lambda\,\mathbf{y} \tag{10}$$

where I defined

$$\mathcal{P}_\lambda := \Phi(\Phi^\dagger\Phi + \lambda^2\mathbf{1})^{-1}\Phi^\dagger. \tag{11}$$

$\mathcal{P}_\lambda$ is a positive semi-definite (and in particular symmetric) matrix.

[2] people use RSS = residual sum of squares, though SSR probably would have made more sense.

1.1 Bayesian interpretation

One can give a Bayesian interpretation to what is happening here. In probablisitc modeling in general your model for the "data generating process" (i.e. for $\rho(y|x)$) should not only state how it accounts for the systematics in the data, but also how it accounts for the noise (or more precisely the **trial-to-trial variability** = all contributions to y that are not determined or fixed by fixing/controlling x, and thus the features $\phi_k(x)$). In parametric modeling, you assume a form $\rho(y|x,\theta)$ parametrized by a set of parameters θ. In the Bayesian setting you normally also impose a prior distribution, $p_\theta(\theta)$, on the parameters.

Ridge regression can then be interpreted as follows. You model noise as **additive gaussian noise** added to an underlying function of the form $f(x) = \overrightarrow{\varphi}(x) \cdot \overrightarrow{w}$, i.e.

$$y = f(x) + \varepsilon = \overrightarrow{\varphi}(x) \cdot \overrightarrow{w} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2). \tag{12}$$

In other words you are assuming the form

$$\rho(y|x,\theta) = \frac{1}{\sqrt{2\pi\sigma_\varepsilon^2}} e^{-\frac{(y-f(x))^2}{2\sigma_\varepsilon^2}} = \frac{1}{\sqrt{2\pi\sigma_\varepsilon^2}} e^{-\frac{(y-\overrightarrow{\varphi}(x)\cdot\overrightarrow{w})^2}{2\sigma_\varepsilon^2}} \tag{13}$$

for the data generating process. Here the model parameters, θ are

$$\theta = \{\overrightarrow{w}, \sigma_\varepsilon\}\,. \tag{14}$$

Given the iid noise assumption, the **likelihood** of the training set data is given by

$$\rho(y_{1:T}|x_{1:T},\theta) = \prod_{\mu=1}^{T} \rho(y_\mu|x_\mu,\theta) = \frac{1}{(2\pi\sigma_\varepsilon^2)^{T/2}} e^{-\frac{\|\mathbf{y}-\Phi\overrightarrow{w}\|^2}{2\sigma_\varepsilon^2}}, \tag{15}$$

or

$$-\log\rho(y_{1:T}|x_{1:T},\theta) = \frac{T}{2}\log(2\pi\sigma_\varepsilon^2) + \frac{1}{2\sigma_\varepsilon^2}\|\mathbf{y}-\Phi\overrightarrow{w}\|^2. \tag{16}$$

The l.h.s. is the **negative log-likelihood**.

As you know, one cheap Bayesian way of learning/training is the maximum *a posteriori* or **MAP** method, where the MAP estimate of the parameters, $\hat{\theta}$, is the one that maximizes the **posterior distribution** ($\sim$ probablity) of parameters. The latter is given by Bayes' formula:

$$p(\theta|y_{1:T},x_{1:T}) \propto \rho(y_{1:T}|x_{1:T},\theta)\; p_\theta(\theta) \tag{17}$$

and

$$\hat{\theta}_{\text{MAP}} := \arg\max_\theta p(\theta|y_{1:T},x_{1:T}) = \arg\max_\theta \rho(y_{1:T}|x_{1:T},\theta)\; p_\theta(\theta) \tag{18}$$

$$= \arg\min_\theta \Big[-\log\rho(y_{1:T}|x_{1:T},\theta) - \log p_\theta(\theta)\Big] \tag{19}$$

Regularization *a la* ridge regression corresponds to the case in which the Bayesian model imposes an isotropic Gaussian prior on $\overrightarrow{w}$, (and say a uniform prior on σ^2), i.e. when

$$p_\theta(\theta) \propto \frac{1}{(2\pi\sigma_w^2)^{N/2}} e^{-\frac{\|\overrightarrow{w}\|^2}{2\sigma_w^2}} \tag{20}$$

where the prior std σ_w reflects the Bayesian model's prior expectation of the scale/size of the components of $\overrightarrow{w}$. In this case, using Eq. (16) we obtain[3]

$$\hat{\theta}_{\rm MAP} \;=\; \arg\min_{\overrightarrow{w},\sigma_\varepsilon}\Big[\frac{1}{2\sigma_\varepsilon^2}\|\mathbf{y}-\Phi\overrightarrow{w}\|^2+\frac{1}{2\sigma_w^2}\|\overrightarrow{w}\|^2+\frac{N}{2}\log\left(2\pi\sigma_w^2\right)+\frac{T}{2}\log\left(2\pi\sigma_\varepsilon^2\right)\Big]. \quad (21)$$

In the (utopian) case that σ_ε is known to the model, only $\overrightarrow{w}$ is fit, and multiplying the cost by $2\sigma_\varepsilon^2$, we obtain

$$\hat{\overrightarrow{w}}_{\rm MAP}=\arg\min_{\overrightarrow{w}}\Big[\|\mathbf{y}-\Phi\overrightarrow{w}\|^2+\lambda^2\|\overrightarrow{w}\|^2\Big], \quad (22)$$

where

$$\lambda:=\frac{\sigma_\varepsilon}{\sigma_w}. \quad (23)$$

i.e. we recover the ridge regression problem, Eq. (3). [4]

1.2 "Feature subspace," SVD and $\mathcal{P}_\lambda$

Let us go back to the form Eq. (10) for the in-sample predictor $\hat{\mathbf{y}}$. In the $\lambda\to 0$ limit, $\mathcal{P}_\lambda$ given by Eq. (11) is in fact an orthogonal **projection**, projecting vectors onto the subspace of $\mathbb{R}^T$ spanned by the columns of Φ; i.e. the subspace of all functions (evaluated on the training x_μ's) that can be written as linear combinations of the features, $\phi_k(x)$. So I will refer to it as the **feature subspace**. One way of seeing this fact is by using the very useful singular value decomposition

$$\Phi=USV^\dagger \quad (25)$$

where U, S and V are $T\times r$, $r\times r$ and $N\times r$ matrices, and $r\leq\max(N,T)$ is the rank of Φ. S is the diagonal matrix of the (positive) s.v.'s, and U and V have orthonormal columns, such that

$$U^\dagger U=\mathbf{1}_{r\times r},\qquad\qquad V^\dagger V=\mathbf{1}_{r\times r}. \quad (26)$$

Using this we obtain

$$\mathcal{P}_\lambda=U\frac{S^2}{S^2+\lambda^2}U^\dagger. \quad (27)$$

and for $\lambda\to 0$

$$\mathcal{P}_0=UU^\dagger. \quad (28)$$

Using Eq. (26) we see that $\mathcal{P}_0^2=\mathcal{P}_0$, which means (given its symmetry) that $\mathcal{P}_0$ is a projection matrix, projecting onto the column space of U, which is the column space of Φ (column space = space spanned the columns; aka image).

[3] note that at this stage the **hyper-parameter** σ_w which is not a parameter of the model's data generating process (i.e. the likelihood function), is not fit/optimized at this stage.

[4] When (as in reality) σ_ε is not known, one possibility is to perform the joint optimization in $\overrightarrow{w}$ and σ_ε greedily for $\overrightarrow{w}$ and then for σ_ε and then iterate back and forth. I.e. first initialize a value for σ_ε and fit $\overrightarrow{w}$ by solving Eq. (22) (with the corresponding λ), and then substitute the result $\hat{\overrightarrow{w}}_{\rm MAP}$ (which is explicitly given by Eq. (7)) in the full cost and minimize for σ_ε. This yields the estimate

$$\hat{\sigma_\epsilon}^2=\frac{1}{T}\|\mathbf{y}-\Phi\hat{\overrightarrow{w}}\|^2. \quad (24)$$

One then plugs this back into Eq. (23) and solves Eq. (22) again, and ... As we will see, Eq. (23) tends to underestimate σ_ϵ^2 by a multiplicative factor, bounded from below by $(1-\frac{N}{T})$.

Note that the (empirical) $N \times N$ covariance matrix of the features on the training set, $\{\varphi_\mu\}$, is given by $\frac{1}{T}\Phi^\dagger\Phi$ (we can assume w.l.o.g. that the sample mean of features and $\{y_\mu\}$ has been removed). This means that $\frac{1}{T}S_{ii}^2$ is the i-th eigenvalue of the feature covariance (if $r < N$, then the remaining $N - r$ of said eigenvalues are 0), i.e. the variance of the i-th **principle component** of the features. Given this connection, I will later on use the notation $s_i^2 := \frac{1}{T}S_{ii}^2$, where s_i^2 are the principal variances of $\overrightarrow{\varphi}_\mu$.

2 Generalization error

(Warning to Bayesians: this section has a frequentist flavor.) Let us now assume the point of view of an imaginary all-knowing observer who knows the true data generating process, and evaluates the prediction performance of the worldly (i.e. real) observer, i.e. its generalization/prediction error, in the long-run. So I will assume that the data are generated by some underlying $f(x)$ with additive iid noise, i.e.

$$y = f(x) + \varepsilon, \qquad \varepsilon \sim p_\varepsilon(\cdot) \tag{29}$$

where the noise distribution $p_\varepsilon(\cdot)$ has zero mean and variance σ_0^2 (in general σ_0^2 is unknown to the real observer and is thus different from the σ_ε^2 introduced in the last section, which was a parameter to be inferred/learned by the real observer). In particular, for the training set we have

$$y_\mu = f(x_\mu) + \varepsilon_\mu, \qquad \varepsilon_\mu \overset{\text{iid}}{\sim} p_\varepsilon(\cdot) \tag{30}$$

or in vector form

$$\mathbf{y} = \mathbf{f} + \boldsymbol{\varepsilon}, \tag{31}$$

where I introduced the notation $\mathbf{f} := \{f(x_\mu)\}_{\mu=1:N}$ and $\boldsymbol{\varepsilon} := \{\varepsilon_\mu\}_{\mu=1:N}$.

What we ultimately really care about (i.e. the true measure of performance) is the average prediction/generalization/test error of our predictor Eq. (8) on a new x, which is sampled from the "natural" distribution of x, $\rho(x)$, i.e.

$$\text{Generalization Error} := \langle (y - \hat{y}(x))^2 \rangle_{\delta,x} = \langle (y - \overrightarrow{\varphi}(x) \cdot \hat{\overrightarrow{w}})^2 \rangle_{\delta,x} \tag{32}$$

where the average is taken over both the noise, δ,[5] and the x, with distributions $p_\varepsilon(\delta)$ and $\rho(x)$, respectively. [6]

Unfortunately, the natural distribution of x is usually unknown (even to God). So let us do something simpler, and look at the "in-sample prediction error": the average generalization/prediction error on the training set x's.[7] So let us generate a new set of outputs on the training set inputs, x_μ. I will denote these new y's by u_μ.[8] We have

$$u_\mu = f(x_\mu) + \delta_\mu \qquad \delta_\mu \overset{\text{iid}}{\sim} p_\varepsilon(\cdot) \tag{33}$$

[5] this is the noise of the new observation, and is different and independent from the ε_μ's in the training set; so I've called it δ to be clear.

[6] Note that the Generalization Error, as defined, depends on the fixed ε_μ's, but not on the averaged out δ (it only depends on the latter's *distribution*).

[7] In other words, we use the *empirical* distribution of x's, provided by the training set $\{x_\mu\}$, as a surrogate for the unknown $\rho(x)$.

[8] If it helps, you can alternatively think that now we are looking at the test-set part of data, unseen during training, as in cross-validation. x_μ can be thought of as controlled "conditions" or "stimuli" in the experiment which are the same between the training and test sets, but the responses/outputs (u_μ's vs. y_μ's) are different due to neural variability.

or in vector form

$$\mathbf{u} = \mathbf{f} + \boldsymbol{\delta} \qquad\qquad \delta_\mu \overset{\text{iid}}{\sim} p_\varepsilon(\cdot) \tag{34}$$

where $\boldsymbol{\delta} := \{\delta_\mu\}_{\mu=1:N}$ are the **test set noise** variables. The "**in-sample prediction error**" is defined as

$$\begin{aligned} \text{PE}(\boldsymbol{\varepsilon}) &:= \left\langle \frac{1}{T}\sum_{\mu=1}^{T}(u_\mu - \hat{y}_\mu)^2 \right\rangle_{\boldsymbol{\delta}} && (35)\\ &= \frac{1}{T}\langle \|\mathbf{u} - \hat{\mathbf{y}}\|^2 \rangle_{\boldsymbol{\delta}} && (36) \end{aligned}$$

where $\hat{\mathbf{y}}$ is the trained model's prediction, given by Eq. (8). I have given PE the explicit arguments $\boldsymbol{\varepsilon}$ to emphasize that it depends on the specific realizations of the training set which gave rise to the estimate $\hat{\overrightarrow{w}}$ and hence the predictions $\hat{\mathbf{y}}$. Note that true generalization, in the sense of the generalization error Eq. (32) carries in it some element of extrapolation to outputs for new x's as well; by contrast, the in-sample prediction error does not involve extrapolation.

Now before massaging Eq. (36), let us see what would be the error incurred by the all-knowing observer[9] In that case, we have to replace $\hat{\mathbf{y}}$ with the true $\mathbf{f}$, yielding

$$\begin{aligned} \text{IE} &:= \frac{1}{T}\langle \|\mathbf{u} - \mathbf{f}\|^2 \rangle_{\boldsymbol{\delta}} && (37)\\ &= \frac{1}{T}\langle \|\boldsymbol{\delta}\|^2 \rangle_{\boldsymbol{\delta}} = \frac{1}{T}\sum_\mu \langle \delta_\mu^2 \rangle_{\boldsymbol{\delta}} = \text{Var}[\delta] && (38)\\ &= \sigma_0^2, && (39) \end{aligned}$$

for the **intrinsic error**.

Going back to Eq. (47), we can use Eqs. (8), (31) and (34), to rewrite the vector of predictor **deviation**s or residuals, $\mathbf{u} - \hat{\mathbf{y}}$, as

$$\begin{aligned} \mathbf{u} - \hat{\mathbf{y}} &= \mathbf{u} - \mathcal{P}_\lambda \mathbf{y} = \mathbf{f} + \boldsymbol{\delta} - \mathcal{P}_\lambda(\mathbf{f} + \boldsymbol{\varepsilon}) && (40)\\ &= \boldsymbol{\delta} + (\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f} - \mathcal{P}_\lambda \boldsymbol{\varepsilon} && (41) \end{aligned}$$

and obtain

$$\text{PE}(\boldsymbol{\varepsilon}) = \frac{1}{T}\langle \|\boldsymbol{\delta} + (\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f} - \mathcal{P}_\lambda \boldsymbol{\varepsilon}\|^2 \rangle_{\boldsymbol{\delta}}. \tag{42}$$

Before proceeding to simplify Eq. (42) further, let us analyze the different contributions to the deviations Eq. (41). The first term is the intrinsic deviation due to noise. The second term is the **systematic deviation** or **bias**:

$$\begin{aligned} \mathbf{b} &:= \langle \mathbf{u} - \hat{\mathbf{y}} \rangle_{\boldsymbol{\delta},\boldsymbol{\varepsilon}} && (43)\\ &= (\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f}. && (44) \end{aligned}$$

It is the average deviation; averaged over all sources of noise/variability, in both test-set ($\boldsymbol{\delta}$) or training-set ($\boldsymbol{\varepsilon}$). Finally, the last term is the contribution of *inevitable* "**noise-fitting**": it is (up to a sign) the contribution of the training-set noise to the predictor, $\hat{\mathbf{y}} = \mathcal{P}_\lambda \mathbf{y} = \mathcal{P}_\lambda \mathbf{f} + \mathcal{P}_\lambda \boldsymbol{\varepsilon}$,

[9] Who nevertheless has no knowledge of noise, but only of the underlying systematics, i.e. of $f(x)$.

and is thus the part that fluctuates from training-set to training-set; the mean square of its μ-th component is the predictor's **variance** at x_μ (variance from training-set to training-set, or more precisely between training-sets with fixed input set).

Going back to Eq. (42), let's use the identity

$$\|\boldsymbol{\delta}+(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}-\mathcal{P}_\lambda\boldsymbol{\varepsilon})\|^2 = \|\boldsymbol{\delta}\|^2+\|(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}\|^2+\|\mathcal{P}_\lambda\boldsymbol{\varepsilon}\|^2-2\mathbf{f}^\dagger(\mathbf{1}-\mathcal{P}_\lambda)^\dagger\mathcal{P}_\lambda\boldsymbol{\varepsilon}-2\boldsymbol{\delta}^\dagger\left[\mathcal{P}_\lambda\boldsymbol{\varepsilon}-(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}\right], \tag{45}$$

and the fact that the last term (linear in $\boldsymbol{\delta}$) vanishes after averaging over the zero-mean $\boldsymbol{\delta}$, to obtain

$$\mathrm{PE}(\boldsymbol{\varepsilon}) \;=\; \sigma_0^2+\frac{1}{T}\left[\|(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}\|^2+\|\mathcal{P}_\lambda\boldsymbol{\varepsilon}\|^2-2\mathbf{f}^\dagger(\mathbf{1}-\mathcal{P}_\lambda)^\dagger\mathcal{P}_\lambda\boldsymbol{\varepsilon}\right]. \tag{46}$$

To recap, this is the in-sample error for the observer fitting his model to the training set with the specific noise realizations $\boldsymbol{\varepsilon}$ hidden in his training set y_μ's. Depending on the specific realization s/he may get too lucky or too unlucky. To get a more concrete result, we can look at the average (training set) case. That is to say, let's average $\mathrm{PE}(\boldsymbol{\varepsilon})$ over $\boldsymbol{\varepsilon}$ as well. The last term in Eq. (46) vanishes after such averaging (in the no-regularization ($\lambda = 0$) case, the last term already vanishes before averaging over $\boldsymbol{\varepsilon}$, as in that case $\mathcal{P}_0^2 = \mathcal{P}_0 = \mathcal{P}_0^\dagger$) and we obtain

$$\overline{\mathrm{PE}} := \langle \mathrm{PE}(\boldsymbol{\varepsilon})\rangle_{\boldsymbol{\varepsilon}} = \sigma_0^2+\frac{1}{T}\|(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}\|^2+\frac{1}{T}\langle\|\mathcal{P}_\lambda\boldsymbol{\varepsilon}\|^2\rangle_{\boldsymbol{\varepsilon}}. \tag{47}$$

In words:

$$\begin{aligned}&\text{average in-sample prediction error} = \\ &\text{intrinsic error + in-sample bias squared + in-sample variance}\end{aligned} \tag{48}$$

Let us now compare this with the **training error**. The latter is given by plugging the estimate $\overrightarrow{w}$ into the $\mathrm{RSS}(\overrightarrow{w})$ formula, or what is the same (up to normalization by $\frac{1}{T}$), by replacing $\mathbf{u}$ in Eq. (36) with the training set output-vector $\mathbf{y}$ (and getting rid of the averaging over the now non-existant $\boldsymbol{\delta}$):

$$\mathrm{TE}(\boldsymbol{\varepsilon}) := \frac{1}{T}\|\mathbf{y}-\hat{\mathbf{y}}\|^2 \tag{49}$$

The massaging steps for this one, mirror exactly Eqs. (40)–(42) and Eq. (45), except $\boldsymbol{\delta}$ is replaced by $\boldsymbol{\varepsilon}$. Let us again look at the average-case training error, by averaging over the training-set noise, $\boldsymbol{\varepsilon}$. Getting rid of terms in Eq. (45) (after $\boldsymbol{\delta}\to\boldsymbol{\varepsilon}$) linear in $\boldsymbol{\varepsilon}$ which vanish upon averaging, we obtain

$$\overline{\mathrm{TE}} := \langle \mathrm{TE}(\boldsymbol{\varepsilon})\rangle_{\boldsymbol{\varepsilon}} = \sigma_0^2+\frac{1}{T}\|(\mathbf{1}-\mathcal{P}_\lambda)\mathbf{f}\|^2+\frac{1}{T}\left\langle\|\mathcal{P}_\lambda\boldsymbol{\varepsilon}\|^2\right\rangle_{\boldsymbol{\varepsilon}}-\frac{1}{T}\left\langle 2\boldsymbol{\varepsilon}^\dagger\mathcal{P}_\lambda\boldsymbol{\varepsilon}\right\rangle_{\boldsymbol{\varepsilon}}. \tag{50}$$

Comparing with Eq. (47), we obtain the relationship

$$\overline{\mathrm{PE}} \;=\; \overline{\mathrm{TE}}+\frac{2}{T}\left\langle\boldsymbol{\varepsilon}^\dagger\mathcal{P}_\lambda\boldsymbol{\varepsilon}\right\rangle_{\boldsymbol{\varepsilon}}, \tag{51}$$

$$\;=\; \overline{\mathrm{TE}}+\frac{2\sigma_0^2}{T}\,\mathrm{Tr}\,(\mathcal{P}_\lambda). \tag{52}$$

(Noting that $\boldsymbol{\varepsilon}\sim\mathcal{N}(0,\sigma_0^2\mathbf{1})$ you can derive the second line as a quick exercise.) The positive extra term (positive because $\mathcal{P}_\lambda$ is p.d.) measures the average-case seriousness of **over-fitting**:[10]

[10] Formally, we can take overfitting to mean that the fit parameters $\hat{\overrightarrow{w}}$ (see Eq. (7)) which yield very good (in the $\lambda = 0$ case, the best possible) training error, yield bad generalization error, but that there are choices of parameters, $\tilde{\overrightarrow{w}}$ that do worse than $\hat{\overrightarrow{w}}$ on the particular training set, but do better than $\hat{\overrightarrow{w}}$ in generalization.

i.e. how much over-optimistic it is to naively take the training error for the test error. As we will see, the training error is not only optimistic w.r.t. the prediction/test error, it is also optimistic w.r.t. the intrinsic error Eq. (39). The flip-side is that the PE is on average worse than the intrinsic error; you are fitting partly to noise in the training set, canceling some of its intrinsic contribution to error when it comes to the training set itself, but harming the prediction.

2.1 the no-regularization case

In the no-regularization case ($\lambda = 0$), Eq. (50) has a straightforward geometric interpretation, since in that case $\mathcal{P}_\lambda$ is the projection operator onto the column subspace of Φ, i.e. the subspace spanned by the feature vectors $\{\varphi_k(x_\mu)\}_{\mu=1:T}$; let's call this the **feature subspace**. This is a r (rank) dimensional subspace, and given the isotropic covariance of $\boldsymbol{\varepsilon}$ (and $\mathcal{P}_0^2 = \mathcal{P}_0$), we obtain $\left\langle \boldsymbol{\varepsilon}^\dagger \mathcal{P}_0 \boldsymbol{\varepsilon} \right\rangle_{\boldsymbol{\varepsilon}} = \left\langle \|\mathcal{P}_0 \boldsymbol{\varepsilon}\|^2 \right\rangle_{\boldsymbol{\varepsilon}} = r\sigma_0^2$ or

$$\overline{\mathrm{PE}} = \overline{\mathrm{TE}} + \frac{r}{T} 2\sigma_0^2, \qquad (\lambda = 0) \tag{53}$$

In cases in which the features have been chosen sensibly so as to be linearly independent (on the input set x_μ) and furthermore $T \geq N$, the rank is N, and we obtain

$$\overline{\mathrm{PE}} = \overline{\mathrm{TE}} + \frac{N}{T} 2\sigma_0^2, \qquad (\lambda = 0, r = N \leq T) \tag{54}$$

Note that when $T \gg N$ the difference between the prediction/test error and the training error is vanishingly small, and we can stop worrying about the over-fitting problem; Over-fitting is a finite/limited data (i.e. real life) problem.

Having the $\lambda = 0$ case in mind, it is worthwhile to re-derive Eq. (50) another time in a slightly different manner. Note that $\mathbf{y} - \hat{\mathbf{y}} = (\mathbf{1} - \mathcal{P}_\lambda)\mathbf{y} = (\mathbf{1} - \mathcal{P}_\lambda)(\mathbf{f} + \boldsymbol{\varepsilon})$. So, using Eq. (31), $\mathrm{TE}(\boldsymbol{\varepsilon}) = \|(\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f} + (\mathbf{1} - \mathcal{P}_\lambda)\boldsymbol{\varepsilon}\|^2/T$, and getting rid of terms linear in $\boldsymbol{\varepsilon}$ which vanish after $\langle\cdot\rangle_{\boldsymbol{\varepsilon}}$ (again for $\lambda = 0$ these actually vanish before averaging), we obtain

$$\overline{\mathrm{TE}} = \frac{1}{T}\|(\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f}\|^2 + \frac{1}{T}\left\langle \|(\mathbf{1} - \mathcal{P}_\lambda)\boldsymbol{\varepsilon}\|^2 \right\rangle_{\boldsymbol{\varepsilon}}. \tag{55}$$

(expanding the last term here we recover the first and the last two terms in Eq. (50)).
Interpretation: in linear regression with $\lambda = 0$, the model cavalierly captures *all* the variations in $\mathbf{y}$ that fall in the feature subspace; and "*all*" includes both systematic variations (changes of y due to variations in x) and noisy variations; this is expressed by the expression $\hat{\mathbf{y}} = \mathcal{P}_0\mathbf{y} = \mathcal{P}_0\mathbf{f} + \mathcal{P}_0\boldsymbol{\varepsilon}$ for the predictor. Everything orthogonal to that subpace (including systemtics and noise) is not captured by the fit model. $\mathbf{1} - \mathcal{P}_0$ is the projection onto the orthogonal complement of the feature subspace, and applying it to $\mathbf{f}$ and $\boldsymbol{\varepsilon}$ yields the systematic and the noise-induced/fluctuating deviations, respectively, which contribute the first and the second terms in Eq. (55), respectively.

When $\lambda = 0$, the second term in Eq. (55) is given by $\left(1 - \frac{N}{T}\right)\sigma_0^2$. Furthermore, in the special case that the true function is indeed captured by the model, i.e. $f(x) = \overrightarrow{\varphi}(x) \cdot \overrightarrow{w}_0$ for some *true* set of weights $\overrightarrow{w}_0$, we have $\mathbf{f} = \Phi\overrightarrow{w}_0$ and since $P_0\Phi = \Phi$, the first term in Eq. (55) vanishes when $\lambda = 0$, and we obtain

$$\overline{\mathrm{TE}} = \left(1 - \frac{N}{T}\right)\sigma_0^2, \qquad (\lambda = 0, r = N \leq T, \text{ zero-bias model}). \tag{56}$$

Thus the average training error underestimates the intrinsic error by $-100\frac{N}{T}$ %. A direct consequence of this is that is that the prediction error is $+100\frac{N}{T}$ more than the intrinsic error:

$$\overline{\mathrm{PE}} = \left(1 + \frac{N}{T}\right)\sigma_0^2, \qquad (\lambda = 0, r = N \leq T, \text{ zero-bias model}). \tag{57}$$

2.2 The effect of regularization

For the general λ case, I note the following identities, which can be obtained by using Eq. (27) (you can work this out as an exercise):

$$\text{predictor variance} = \frac{1}{T}\left\langle \|\mathcal{P}_\lambda \boldsymbol{\varepsilon}\|^2 \right\rangle_{\boldsymbol{\varepsilon}} = \frac{\sigma_0^2}{T}\,\mathrm{Tr}(\mathcal{P}_\lambda^\dagger \mathcal{P}_\lambda) = \frac{\sigma_0^2}{T}\sum_{i=1}^{r} \frac{s_i^4}{\left(s_i^2 + \frac{\lambda^2}{T}\right)^2} \leq \sigma_0^2 \frac{r}{T}, \tag{58}$$

$$\text{average optimisim} = \overline{\mathrm{PE}} - \overline{\mathrm{TE}} = 2\frac{\sigma_0^2}{T}\,\mathrm{Tr}(\mathcal{P}_\lambda) = 2\sigma_0^2 \frac{1}{T}\sum_{i=1}^{r} \frac{s_i^2}{s_i^2 + \frac{\lambda^2}{T}} \leq 2\sigma_0^2 \frac{r}{T} \tag{59}$$

$$\text{predictor squared bias} = \frac{1}{T}\|(\mathbf{1} - \mathcal{P}_\lambda)\mathbf{f}\|^2 = \frac{1}{T}\left[\sum_{i=1}^{r} \frac{\lambda^4 |f_i|^2}{\left(T s_i^2 + \lambda^2\right)^2} + \sum_{i=r+1}^{T} |f_i|^2\right] \geq \frac{1}{T}\sum_{i=r+1}^{T} |f_i|^2 \tag{60}$$

where s_i^2 ($i \leq r$) where defined at the end of 1.2 (for $i > r$ they vanish); they are the (nonzero) eigenvalues of $\frac{1}{T}\Phi^\dagger\Phi$, i.e. (assuming φ's are w.l.o.g. mean removed) the variances of the principle components of the features (in particular, they have a finite limit as $T \to \infty$). I also defined $f_i := \mathbf{u}_i^\dagger \mathbf{f}$, where for $i \leq r$, $\mathbf{u}_i$ is the i-th column of U, and for $r < i \leq T$, $\mathbf{u}_i$ furnish some orthonormal completion of the columns of U in the full T-dimensional space.

When $\lambda = 0$, or $T \gg \frac{\lambda^2}{\min s_i^2}$, for l.h.s.'s in Eqs. (58)–(60) we obtain the r.h.s. of the inequalities therein, i.e. $\frac{r\sigma_0^2}{T}$, $\frac{2r\sigma_0^2}{T}$, and $\frac{1}{T}\sum_{i=r+1}^{T}|f_i|^2$ for the three lines, respectively. Thus we see clearly what regularization does: by increasing λ you trade off some deterioration on the bias front with some improvement in the contribution of variance/fluctuation; you have a smaller tendency to over-fit, but higher tendency to miss some of the systematics, as you've become more rigid and less flexible.

The art of learning can arguably be summarized in finding (more) intelligent methods for regularization (of complex models), such that you remain as flexible to complexities of systematics (what you care about), but as rigid w.r.t. noise or nuisance, as possible. In one way or another, this always relies on some sort of *prior* knowledge on the nature of noise/nuisance and the systematic variations, and on exploiting the differences between the two.

Also ote how when T is sufficiently large the regularization doesn't have any effect (it is not needed either), i.e. it's as if $\lambda = 0$;[11] regularization is a solution to over-fitting, which ceases to be an issue given sufficiently large amounts of data, and it automatically stops influencing things for large T.

[11] Since λ can be written as $\frac{\sigma_0}{\sigma_w}$ where $\sigma_0 =: \sigma_\varepsilon$ is the strength of noise in y, and σ_w is the prior expecation of weight size, we see that T is "sufficiently large" roughly when $T \gg \left(\frac{\sigma_\varepsilon}{\sigma_w \sigma_\varphi}\right)^2$, wherer $\sigma_\varphi^2 := \min s_i^2$.

3 Model selection, AIC and VC

Often in practice, we need to choose between M different models, $\rho(y|x,\theta_m)$ ($m \in 1:M$) that we are intend to fit to the data (each with its own parameter set θ_m) and deciding which one does a better (or the best) job. From the point of view of the last section, we would like to choose the model that yields the smallest generalization error. One way to achieve this is **cross-validation**, which is probably the best thing to do at least when we are not too data-limited: you train on (the bigger) part of the data and then test on the (smaller) remainder, choosing the model with the best performance on the test part.

Average squared deviations which have been our notion of error so far are just one possible choice, and our discussion of subsection 1.1 suggests that it's a natural one for the case of additive gaussian noise (in the data generating process), as it is the negative log-likelihood in that case. On the other hand, maximizing likelihood (or minimizing the negative log-likelihood) is a standard way of fitting models, and thus conversely, the negative log-likelihood can serve as a general notion of error. However it is the negative log-likelihood of the training set that is minimized by fitting, and this provides only a notion of training error.

But inspired by Eq. (54), we can come up with a simple heuristic to go from our estimate of the training error (the minimum of negative log-likelihood) to a rough estimate of generalization/prediction error. Note that in the gaussian case, the negative log-likelihood Eq. (16) (up to an additive constant involving σ_ε) is proportional to the training error. More precisely the second term in Eq. (16) is $\frac{T}{2\sigma_\varepsilon^2}$ times $\mathrm{TE}(\boldsymbol{\varepsilon})$ defined in Eq. (49). Multiplying Eq. (54) by this factor (after cheating by identifying an estimate of $\hat{\sigma}_\varepsilon$ with the true σ_0) we obtain

$$\frac{T}{2\sigma_0^2}\overline{\mathrm{PE}} = \overline{\mathrm{NLL}} + N. \tag{61}$$

The r.h.s. (without the bar on NLL) is thus heuristically generalized to general models, beyond the additive gaussian noise assumption, and used as a criterion for model-selection which is known as the **Akaike information criterion**. [12] Thus according to AIC, one chooses the model with the smallest

$$\mathrm{NLL}_m + N_m, \tag{62}$$

(with m indexing the M competing models). We see that more complex models with more parameters are penalized by the last term, N_m. But note that the NLL_m hides in it the systematic bias contribution, and more flexible/complex models do better in that.

But the number of parameters is not always the best notion of complexity, optimism (difference between test and training error), or (over-fitting) capacity. Furthermore, as noted in footnote 12, the deeper justification for the general usage of the AIC is **asymptotic** (large T). The **VC dimension** (a characteristic of a model) provides one solution to these deficiencies. In general, it is not the number of trained parameters, and it yields a general ("uniform") probablistic upper bound on the difference of the training error and test/generalization error (i.e. replacing N_m is Eq. (62)), but which is **non-asymptotic** and holds with arbitrarily high probability irrespective of the true data generating process/distrubtion (i.e. "uniformly") [13]. The

[12] To be fair Akaike and others have shown that in some more rigorous sense, such a relationship holds **asymptotically** (i.e. when $T \gg 1$ –large dataset) between the training set error and the predictor error (which now are not the squared deviation, but are provided by the negative log-likelihood of a data in training and test).

[13] in the context in which it is defined, which in its original formulation is binary classification, and not regression which was discussed here; although there are generalizations of it for regression. More precisely, VC provides a family of probabilistic upper bounds with progressively and arbitrarily higher confidence/probability of holding.

problem with the VC dimension (as Stefano would have told you) is that in many cases of interest it yields an upper bound which is too loose and hence hugely (pessimistically) over-estimates the generalization error.