Bias - variance decomposition

Bishop - Pattern Regognition and ML (Book)

SSS, SBD - Understanding ML (Book) (UML)

In previous classes, we have studied the no-free lunch theorem, which essentially states that no learning can suceed in all tasks (there is no universal learner).

More concretly, though informal: for any learning algorithm A and training set S of size m, there there exists a distribution D such that

$$\exists \text{ a function } h^* \text{ with } L_D(h^*) = 0$$

but for $S \sim D^m$:

$$L_D(A(S)) \gtrsim \frac{1}{2} - \underbrace{f(|\mathcal{X}|, m)}_{\text{'some function depending on the size of domain set } \mathcal{X} \text{ and } m.}$$

We have also seen that the no-free lunch implies the following (stated informally here) corollary (corollary 5.2 in UML):

Corollary

Let $\mathcal{X}$ be an infinite domain set and let $\mathcal{H}$ be set of all functions mapping $\mathcal{X}$ to a particular target set. Then, $\mathcal{H}$ is not PAC learnable.

↳ any algorithm that chooses its output from hypotheses in $\mathcal{H}$, e.g. the ERM predictor, will fail on some learning task.

Question: How one can escape the hazards foreseen by the theorem by making use of prior knowledge about an specific learning task?

The prior knowledge can be expressed by restricting the hypothesis class

Question How one can choose a "good" hypothesis class ?

Ideally, one would like

☺ Class including hypothesis with no error at all (in the PAC sensing) or at least that the smallest error achievable by a hypothesis from this class in indeed really small (in the agnostic setting)

However, there is a tradeoff:

☹ One cannot simply choose the richest class, the class of all function over a given domain.

⇒ Bias - complexity tradeoff
(or bias-variance)

The bias-variance tradeoff is analyzed through an error decomposition. This discussion can be made in a fairly good degree of generality (see, e.g., 5.2 in UML).

Neverthelless, in order to fix the ideas, we will focus on a particular set:

Least squares regression
Bias-variance decomposition

Suppose we are given a training set

$$S=\{(\vec{x}_1,y_1),\dots,(\vec{x}_m,y_m)\} \; ; \; \vec{x}_j \in \mathbb{R}^d,\; y_j \in \mathbb{R}$$

where each pair is sampled iid from a distribution $\mathcal{D}(\vec{x},y)$.

Given a $\vec{x}$ outside S, the goal is to construct an estimator $h_S(\vec{x})$ to predict the respective label.

The prediction error on $\vec{x}$ is quantified by

$$\ell_S(\vec{x},y) = \left(h_S(\vec{x}) - y \right)^2$$

Consider now the expected prediction, which is known to be optimal for the square loss

$$\bar{h}(\vec{x}) = \mathbb{E}[y|\vec{x}] = \int dy \; y \, \mathcal{D}(y|\vec{x})$$

Let us now consider the expectation of the prediction error

$$\mathbb{E}_{\vec{x},y|S}\left[\ell_S(\vec{x},y)\right] =$$

$$= \mathbb{E}_{\vec{x},y|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x}) + \bar{h}(\vec{x}) - y \right)^2\right]$$

$$= \mathbb{E}_{\vec{x},y|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x}) \right)^2\right]$$

$$+ \mathbb{E}_{\vec{x},y|S}\left[\left(h(\vec{x}) - y \right)^2\right]$$

$$+ 2\,\mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x})\right)\underbrace{\left(\bar{h}(\vec{x}) - \mathbb{E}_{y|\vec{x}}[y]\right)}_{= 0}\right]$$

Then

$$\mathbb{E}_{\vec{x},y|S}\left[\ell_S(\vec{x},y)\right] =$$

$$= \underline{\underline{\mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x})\right)^2\right]}} + \underline{\underline{\mathbb{E}_{\vec{x},y}\left[\left(\bar{h}(\vec{x}) - y\right)^2\right]}}$$

Variance of the predictor $h_S(\vec{x})$ over the "mean" (optimal) preditor

↳ We will focus on this contribution

Intrinsic noise on the data. It does not depend on the preditor

↳ the minimum achievable value of the expected loss

The estimator $h_S(\vec{x})$ can be thought as a parametric model $h_S(\vec{x};\vec{w})$, where $\vec{w}$ represents a set of parameters, e.g., the weights of a neural network.

Following a Bayesian point of view, one

would quantify the uncertainty of the predictor (given the data set S) through a posterior over $\vec{w}$.

A frequentist perspective, on the other hand, involves making a point estimate based on S. The uncertainty of this estimator is interpreted as follows:

- Suppose a large number of data sets of size m drawn from $\mathcal{D}(\vec{x}, y)$ is available. For any given data set S, one runs the learning algorithm and predicts a function $h_S(\vec{x}; \vec{w})$

- Different data sets from the ensemble will provide different estimator functions and different values of the square loss. The performance of a particular learning algorithm is then quantified by taking the average over thin ensemble of data sets.

↳ A remark or criticism on the frequentist perspective: If one has access to a very large number of data sets, why do not one just merge them in a huge single data set in order to obtain a better predictor?

↳ And then quantify the (small, one would hope) uncertainty through prior knowledge on $\vec{w}$ (Bayesian perspective).

For the illustrative purpose of this class, we will not stick to possible criticism to the frequentist point of view. We will then consider the quantity:

$$\mathbb{E}_S\left[h_S(\vec{x})\right],$$

add and subtract what is inside the expectation of the first term, the one representing

$$\underline{\underline{\mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x})\right)^2\right]}} =$$

$$= \mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \mathbb{E}_S\left[h_S(\vec{x})\right] + \mathbb{E}_S\left[h_S(\vec{x})\right] - \bar{h}(\vec{x})\right)^2\right]$$

$$= \mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \mathbb{E}_S\left[h_S(\vec{x})\right]\right)^2\right]$$

$$+ \mathbb{E}_{\vec{x}}\left[\left(\underline{\mathbb{E}_S\left[h_S(\vec{x})\right] - \bar{h}(\vec{x})}\right)^2\right] \quad \text{does not depend on } S$$

$$+ 2\mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \mathbb{E}_S\left[h_S(\vec{x})\right]\right)\left(\mathbb{E}_S\left[h_S(\vec{x})\right] - \bar{h}(\vec{x})\right)\right]$$

Taking the expectation over S, i.e., $\mathbb{E}_S$, the cross term goes to zero, a

$$\mathbb{E}_S \underline{\underline{\mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \bar{h}(\vec{x})\right)^2\right]}} =$$

$$\mathbb{E}_S \mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \mathbb{E}_S\left[h_S(\vec{x})\right]\right)^2\right]$$

$$+ \mathbb{E}_{\vec{x}}\left[\left(\mathbb{E}_S\left[h_S(\vec{x})\right] - \bar{h}(\vec{x})\right)^2\right]$$

Including the intrinsic noise contribution, we have

$$\mathbb{E}_S \mathbb{E}_{\vec{x},y|S}\left[\ell_S(\vec{x},y)\right] =$$

$$= \underbrace{\mathbb{E}_{\vec{x}}\left[\left(\mathbb{E}_S\left[h_S(\vec{x})\right] - \bar{h}(\vec{x})\right)^2\right]}_{\text{“}(\text{bias})^2\text{”}}$$

$$+ \underbrace{\mathbb{E}_S \mathbb{E}_{\vec{x}|S}\left[\left(h_S(\vec{x}) - \mathbb{E}_S\left[h_S(\vec{x})\right]\right)^2\right]}_{\text{variance}}$$

$$+ \underbrace{\mathbb{E}_{\vec{x},y}\left[\left(\bar{h}(\vec{x}) - y\right)^2\right]}_{\text{noise}}$$

Bias: represents the extent to which the average prediction over all data sets differs from the desired regression function.

Variance: measures the extent to which the solutions for individual data sets vary around their average: measures the sensitivity of $h_S(\vec{x})$ to a particular data set.

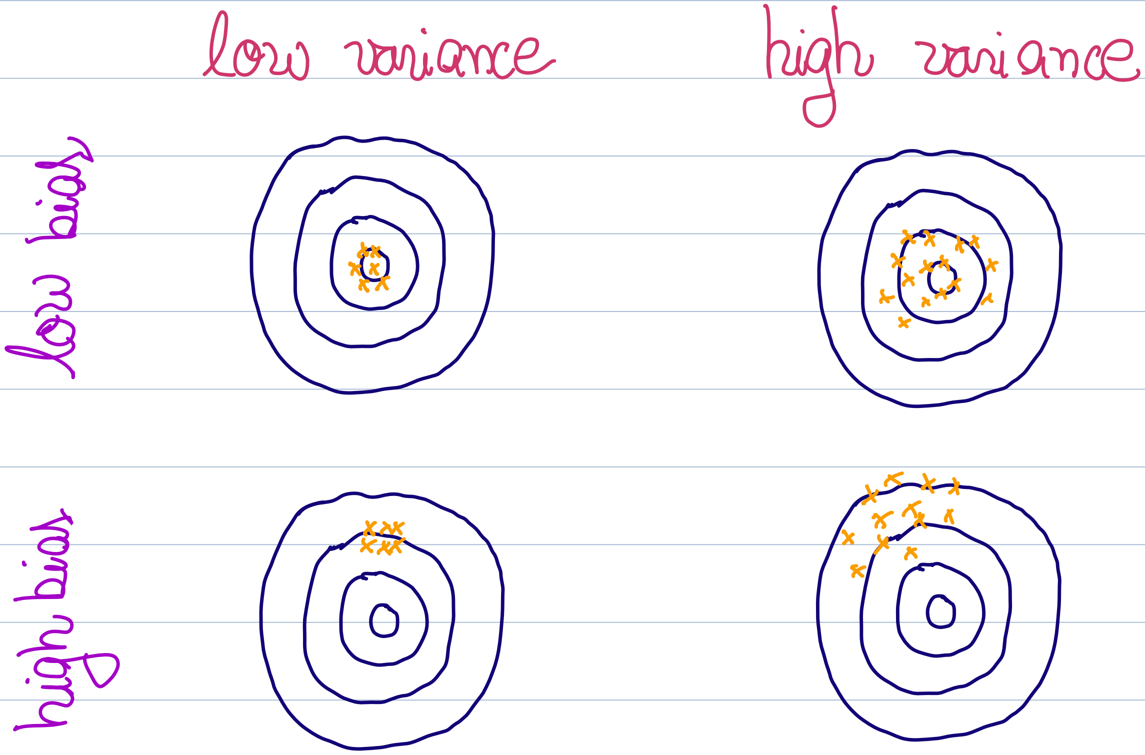
There is a trade-off between bias and variance. Essentially:

- Very flexible models have low bias and high variance.

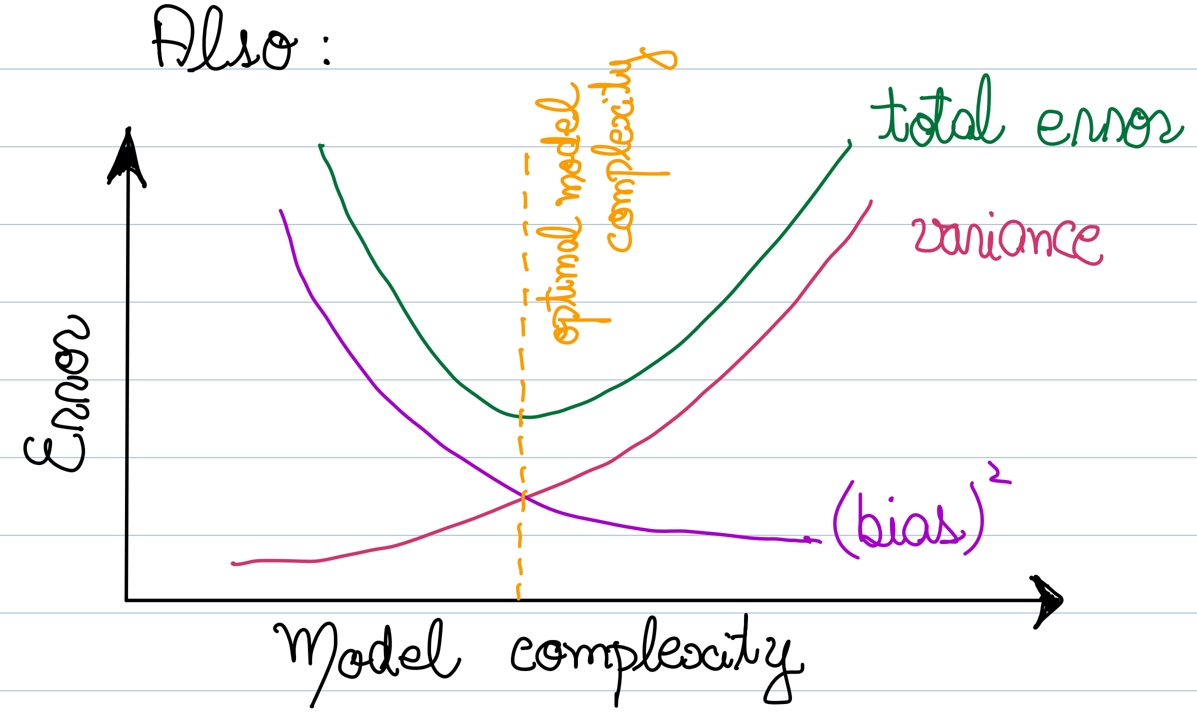
- Relatively rigid models have high bias and low variance.

↳ think about trying to fit highly non-linear data with a linear model: the variance will be low, but the bias will be high since the model will make many mistakes on the training data.

A picture:



Also:



The model with best prediction capability is then the one that leads to the best balance between bias and variance.

↳ observe that this is similar to the concept of overfitting and underfitting

Although the bias variance decomposition my provide interesting insights into the model complexity from the frequentist perspective, it is of limited practical value, as one does not have an ensamble of data sets.

If one had a large number of independent training sets of a given size, it would be better to combine them into a single large training set, which would reduce the level of overfitting, for a given model complexity.

A step further would be a Bayesian treatment of linear basis function models. In this particular case, the over-fitting associated with maximum likelihood can be avoided by marginalizing over the model parameter. In this class, we will not enter on such detais. Those interested can check Chapter 3 on Bishop's book.

Double-descent

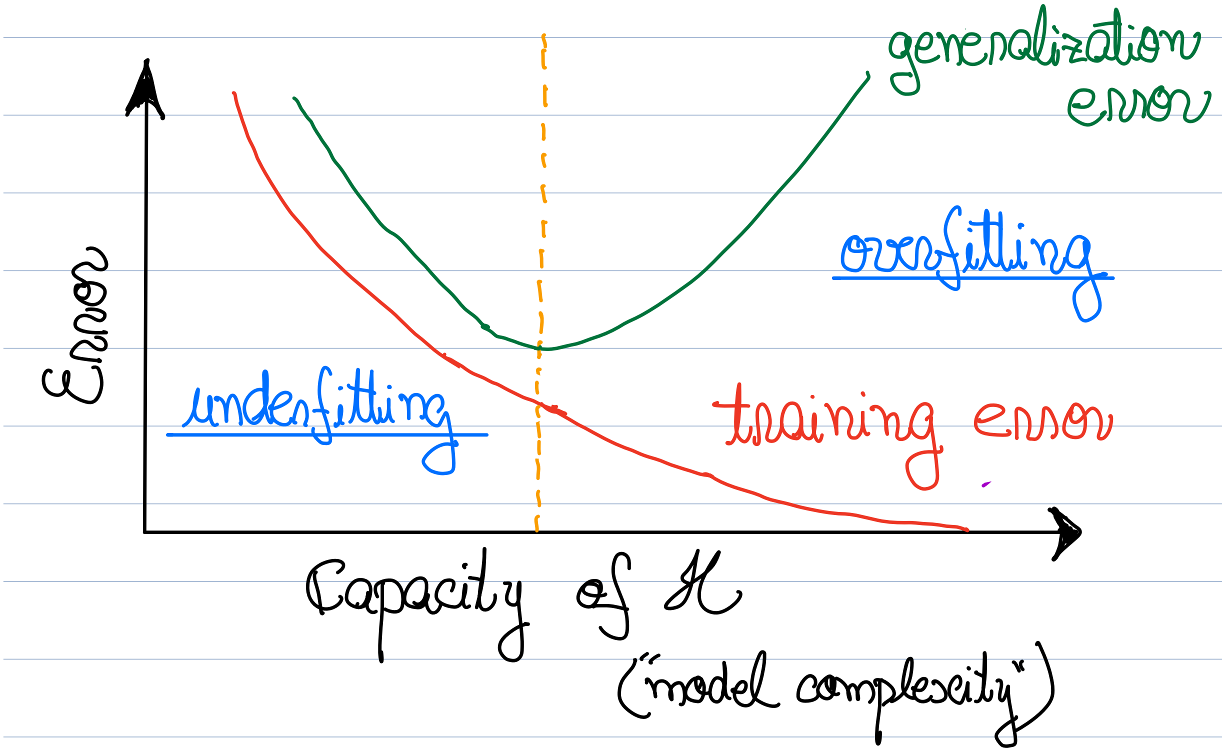
Belkin et al (2018)

Reconciling modern machine learning practice and the bias-variance trade-off.

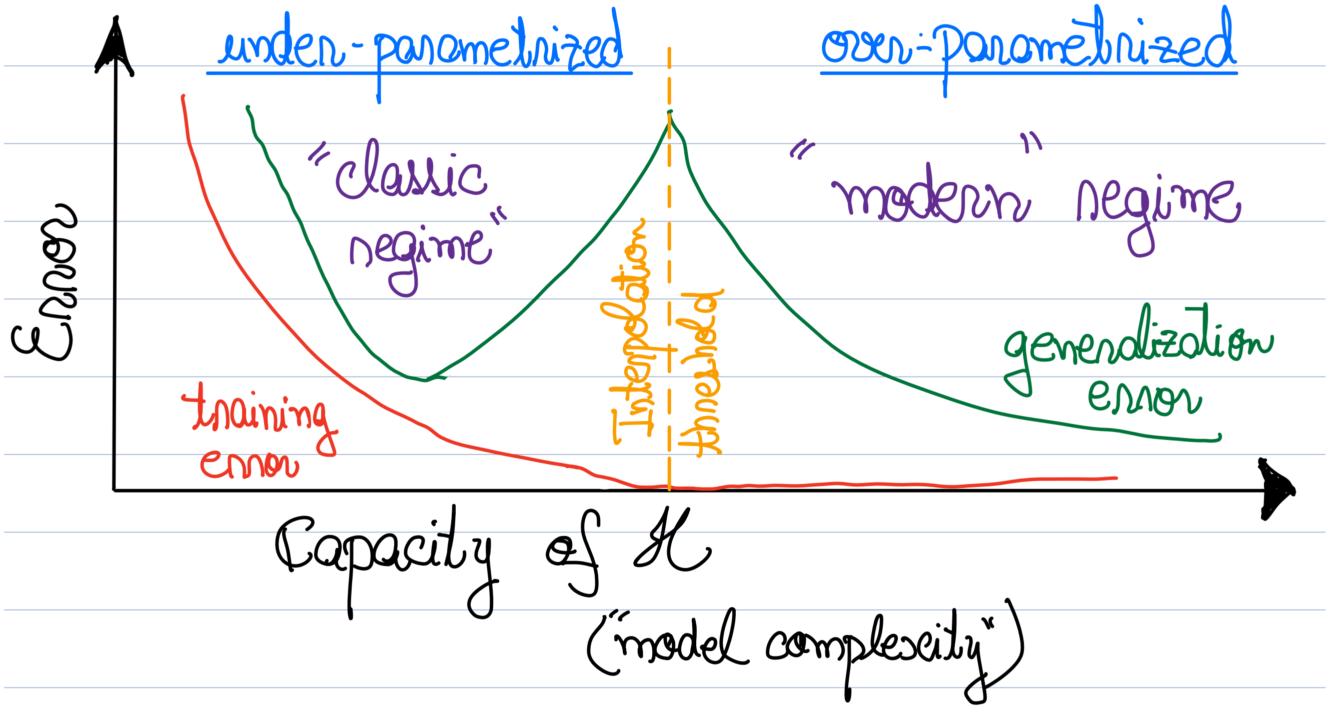
The picture we painted before can be represented, in practice, by the U-shaped curve below.

Let $\mathcal{H}$ be the function class from which a predictor h_S is chosen.

For example $\mathcal{H}$ can be the class of linear classifiers or the class of l-layers neural networks.



However, in real world applications there are plenty of evidence for the following picture: (Belkin et al 2018)



Understanding systematically under which conditions and why the "modern" regime appears is probably the most important problem in machine learning theory!

It is important to remark that regardless the catchy name double-descent has been recently popularized since Belkin et al 2018, this two-fold descent has not been historically overlooked.

For a "prehistory" on double-descent, see:

Loog et al 2020, PNAS 117(20) 10625
"A brief prehistory of double descent"

and references therein.