

Machine Learning

Bias-Complexity Trade-off

Fabio Vandin

November 10th, 2023

Our Goal in Learning

Given:

- training set: $S = ((x_1, y_1), \dots, (x_m, y_m))$
- loss function: $\ell(h, (x, y))$

Want: a function \hat{h} such that $L_{\mathcal{D}}(\hat{h})$ is small

We can pick: the learning algorithm A that given S will produce $\hat{h} = A(S)$

Our Goal in Learning

Given:

- training set: $S = ((x_1, y_1), \dots, (x_m, y_m))$
- loss function: $\ell(h, (x, y))$

Want: a function \hat{h} such that $L_{\mathcal{D}}(\hat{h})$ is *small*

We can pick: the learning algorithm A , that given S will produce $\hat{h} = A(S)$

Note: A comprises:

- the hypothesis set \mathcal{H}
- the procedure to pick $\hat{h} = A(S)$ from \mathcal{H}

Question: is there a *universal learner*, i.e., an (implementable) algorithm A that predicts the best \hat{h} for any distribution \mathcal{D} ?

The No Free Lunch Theorem

The following answers the previous question for some specific settings.

Theorem (No-Free Lunch)

Let A be any learning algorithm for the task of binary classification with respect to the 0-1 loss over a domain \mathcal{X} . Let m be any number smaller than $|\mathcal{X}|/2$, representing a training set size. Then, there exists a distribution \mathcal{D} over $\mathcal{X} \times \{0, 1\}$ such that:

- there exists a function $f: \mathcal{X} \rightarrow \{0, 1\}$ with $L_{\mathcal{D}}(f) = 0$
- with probability of at least $1/7$ over the choice of $S \sim \mathcal{D}^m$ we have that $L_{\mathcal{D}}(A(S)) \geq 1/8$.

Note: there are similar results for other learning tasks.

No Free Lunch and Prior Knowledge

Corollary

Let \mathcal{X} be an infinite domain set and let \mathcal{H} be the set of all functions from \mathcal{X} to $\{0, 1\}$. Then, \mathcal{H} is not PAC learnable.

What's the implication?

We need to use our prior knowledge about \mathcal{D} to pick a *good* hypothesis set.

How do we choose \mathcal{H} ?

- we would like \mathcal{H} to be *large*, so that it may contain a function h with small $L_{\mathcal{D}}(h)$
- no free lunch $\Rightarrow \mathcal{H}$ cannot be too large!

Error Decomposition

training set

hypothesis class

Let h_S be an ERM h hypothesis.

$$L_D(h_S) = \underbrace{L_D(h_S) - \min_{h \in \mathcal{H}} L_D(h)}_{\substack{\text{estimation error} \\ \mathcal{E}_{\text{est}}}} + \underbrace{\min_{h \in \mathcal{H}} L_D(h)}_{\substack{\text{approximation error} \\ \mathcal{E}_{\text{app}}}}$$

Error Decomposition

Let h_S be an $\text{ERM}_{\mathcal{H}}$ hypothesis.

Then

$$L_{\mathcal{D}}(h_S) = \epsilon_{\text{app}} + \epsilon_{\text{est}}$$

where

- $\epsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$ (approximation error)
- $\epsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$ (estimation error)

Approximation error: $\epsilon_{\text{app}} = \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$

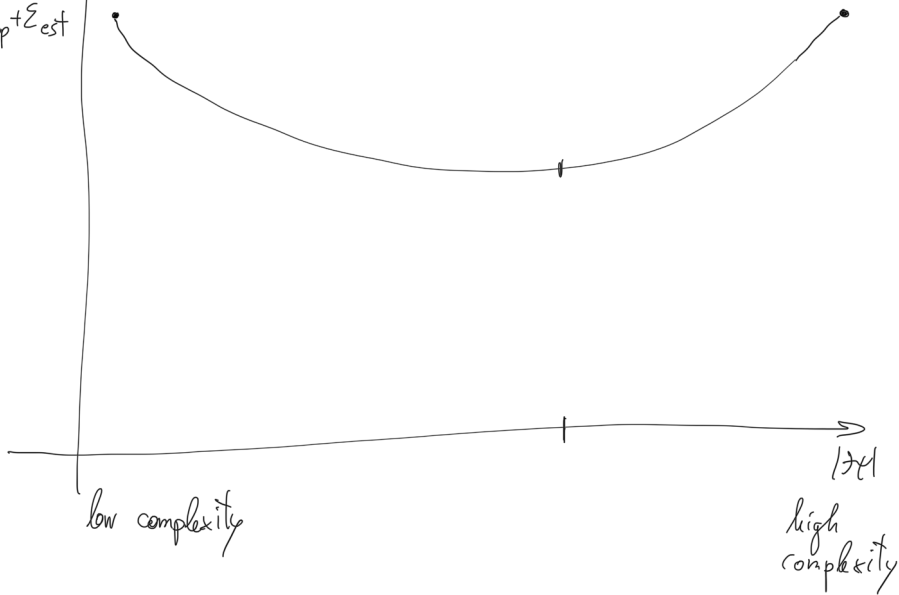
- derives from our choice of \mathcal{H}
- once we have chosen $\mathcal{H} \Rightarrow \epsilon_{\text{app}}$ is unavoidable!
- to decrease it, chose a “larger” \mathcal{H}

Estimation error: $\epsilon_{\text{est}} = L_{\mathcal{D}}(h_S) - \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h)$

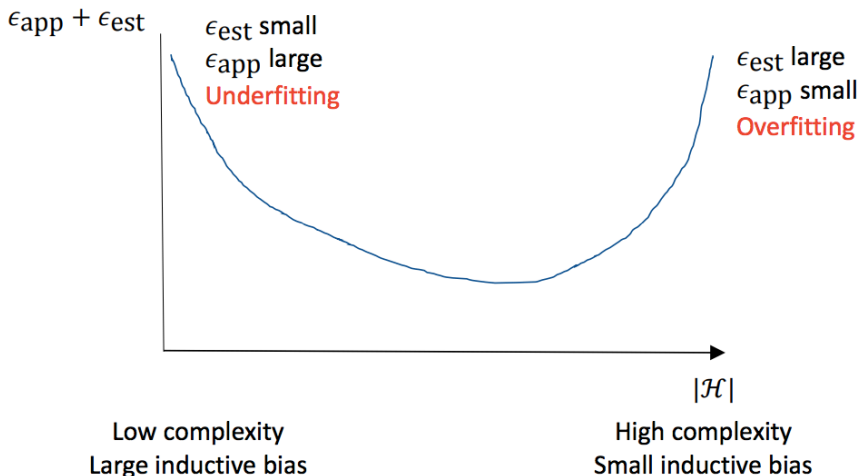
- derives from our inability to choose (with ERM) the best hypothesis in \mathcal{H}
- could be avoided if had chosen the best hypothesis!
- to decrease, we need a low number of hypotheses in \mathcal{H} so that training error is good estimate of generalization error for all of them \Rightarrow need a “small” \mathcal{H}

Complexity of \mathcal{H} and Error Decomposition

$$\angle_{\Phi} (h_s)$$
$$\varepsilon_{\text{app}} + \varepsilon_{\text{est}}$$



Complexity of \mathcal{H} and Error Decomposition



Estimating $L_{\mathcal{D}}(h_S)$

How can we estimate the generalization error $L_{\mathcal{D}}(h)$ for a function h , for example $h_S \in \text{ERM}_{\mathcal{H}}$?

We can use a **test set**: new set of samples not used for picking h_S (=the training set).

Notes:

- the test must not be looked at until we have picked our **final** hypothesis!
- in practice: we have 1 set of samples and we split it in *training set* and *test set*.

Bibliography

[UML] Chapter 5