

ELEN0062 - Introduction to Machine Learning

Project 2 - Bias and variance analysis

October 29, 2026

The goal of this second assignment is to help you better understand the important notions of bias and variance. The first part is purely theoretical, while the second part requires to perform some experiments with scikit-learn. You should hand in a *brief* report giving your developments, observations and conclusions along with the scripts you have implemented to answer the questions of the second part. The project must be carried out by groups of at most *three students* and submitted on Gradescope¹ before *November 20, 23:59*. There will be two projects to submit to: one for your python scripts and one for your report.

Use of generative AI. The use of generative AI tools is permitted, but it is neither encouraged nor required. The analytical derivations of Section 1, the design and implementation of the experimental protocol, and the discussion of the results are expected to be your own work, as they are the core of this project. If used, generative AI tools should be limited to assistance with programming tasks and language revision. Regardless of whether such tools are used, you are expected to fully understand and be able to explain every aspect of your submission. Your report must include a short AI-use declaration stating which tools were used, for which tasks, to what extent, and how their outputs were verified. If you did not use any generative AI tool, state this explicitly. Students may be asked to explain any part of their code, methodology, or report. Failure to disclose the use of generative AI tools, or any other external sources, may be considered academic fraud.

1 Analytical derivations

Let us consider a regression problem with a single input variable $x \in [0, 1]$, where the output is given by:

$$y = h(x) + \epsilon, \quad \text{with } h(x) = ax + b,$$

$a, b \in \mathbb{R}$, and $\epsilon \in \mathbb{R}$ drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$, independently of x .

We study a very simple learning algorithm, called the *regressogram*. The input interval $[0, 1]$ is partitioned into K bins of equal width $\Delta = 1/K$, $B_j = [(j-1)\Delta, j\Delta)$, $j = 1, \dots, K$, and the prediction $\hat{f}_{LS}(x)$ at a point x is the average of the outputs y^i of the learning sample examples whose input x^i falls into the same bin as x . This model can be seen as a regression tree whose structure (i.e., its splits) is fixed in advance, independently of the learning sample, and K plays the role of the tree complexity.

To simplify the analysis, we assume that the inputs $\{x^1, \dots, x^N\}$ of the learning sample are **fixed**, so that only the outputs y^i are random, and that they are spread so regularly over $[0, 1]$ that²:

- any interval of length $\ell \leq \Delta$ included in $[0, 1]$ contains exactly $N\ell$ inputs. In particular, each bin contains exactly $m = N\Delta = N/K$ inputs (we assume that N/K is an integer);
- the average of the inputs falling into any interval of length Δ is equal to the middle point of this interval.

¹<https://www.gradescope.com>, Entry code: JN44ND.

²This is an idealization of, e.g., the regular grid $x^i = (i - 1/2)/N$, $i = 1, \dots, N$, when N is large.

- (1.1) Let x_0 be a point that belongs to bin B_j , whose center is denoted $c_j = (j-1/2)\Delta$. Give an analytical expression of the Bayes model $h_B(x_0)$, the residual error $\text{noise}(x_0)$, the squared bias $\text{bias}^2(x_0)$, and the variance $\text{variance}(x_0)$ of the regressogram at x_0 , as a function of a , b , σ , N , K , x_0 , and c_j .
- (1.2) Assuming that test inputs x are drawn uniformly in $[0, 1]$, derive the *mean* (over x) of the residual error, squared bias and variance, and show that the mean expected generalization error is equal to:

$$E_x\{E_{LS}\{E_{y|x}\{(y - \hat{f}_{LS}(x))^2\}\}\} = \sigma^2 + \frac{a^2}{12K^2} + \frac{\sigma^2 K}{N}.$$

- (1.3) Discuss the impact of K , N , σ , and a on the mean squared bias and the mean variance.
- (1.4) Treating K as a continuous variable, derive the value K^* that minimizes the mean expected generalization error.
- (1.5) Discuss how K^* depends on N , σ , and a . How does the excess error with respect to the residual error decrease with N when $K = K^*$?

2 Empirical analysis

In this part, we move to binary classification. Rather than studying the misclassification error, for which there is no simple additive bias-variance decomposition, we will study the quality of the *class probability estimates* $\hat{f}_{LS}(\mathbf{x}) \in [0, 1]$ of $P(y = 1|\mathbf{x})$ produced by the learning algorithms (as given by the `predict_proba` method in scikit-learn). Encoding the classes as $y \in \{0, 1\}$, the quality of these estimates is measured by the squared error $(y - \hat{f}_{LS}(\mathbf{x}))^2$ (also called the *Brier score*), to which the bias-variance decomposition seen in the course directly applies.

Data. We consider the following synthetic problem:

- the input vector is $\mathbf{x} = (x_1, x_2)$, where x_1 and x_2 are drawn independently and uniformly in $[-1, 1]$;
- given $\mathbf{x}$, the output $y \in \{0, 1\}$ is drawn with

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(-\beta(x_2 - \sin(\pi x_1)))},$$

where $\beta > 0$ controls the noise level of the problem.

Unless we ask you to study the impact of these parameters, you can use the following default values: $N = 250$ for the size of the learning sample and $\beta = 5$.

Note: in the following questions, we do not tell you precisely how to set all other parameter values or ranges. It is your responsibility to choose them wisely to illustrate the expected behaviors. Some of the experiments may take time depending on your computer. Be patient.

- (2.1) Show that, for the squared error, the Bayes model is $h_B(\mathbf{x}) = P(y = 1|\mathbf{x})$ and that the residual error at $\mathbf{x}$ is $P(y = 1|\mathbf{x})(1 - P(y = 1|\mathbf{x}))$. Where in the input space is the residual error the largest? How does β affect it?
- (2.2) Using all the information you have about the problem, describe a protocol to estimate the residual error, the squared bias, the variance, and the expected error at a given point $\mathbf{x}$, for a given learning algorithm. Explain also how to estimate the *mean* values of the same quantities over the input space.
- (2.3) Implement this protocol and apply it to neural networks with one hidden layer³, k NN, and decision trees, with hyper-parameter values of your choice. For each method, display, as maps over the (x_1, x_2) plane, the squared bias, the variance, and the average model, and compare the latter with the Bayes model. Where are bias and variance concentrated? Comment these results.

³Use `MLPClassifier` in scikit-learn with `solver="lbfgs"`, a very small l2 penalty (e.g., `alpha=1e-5`), and a large enough `max_iter` (e.g., 2000), so that the networks are trained (almost) to convergence. Do not fix the `random_state` parameter to the same value for all learning samples: the random initialization of the weights is part of the learning algorithm.

(2.4) Estimate the *mean* values of the expected error, the residual error, the squared bias, and the variance for the same three methods as a function of:

- their main complexity parameter: the number of neurons in the hidden layer for neural networks, k for k NN, and the maximum depth for decision trees;
- the size N of the learning sample;
- the noise level β .

Comment your results and support your observations with the appropriate plots.

(2.5) For one of the three methods of your choice, study how the optimal value of its complexity parameter evolves with N and β , and relate your observations to your answer to questions (1.4) and (1.5).

(2.6) One generic method to reduce variance is bagging (for “bootstrap aggregating”), which grows several models from bootstrap samples drawn from the original LS and then averages their predictions. Apply this bagging⁴ idea on all three methods and evaluate its impact on bias and variance for increasing values of the number of models that are averaged. In particular, discuss the interest of applying bagging in combination with each method.

(2.7) Let us finally come back to the misclassification error. For k NN and decision trees, estimate the mean expected misclassification error of the classifier obtained by predicting class 1 when $\hat{f}_{LS}(\mathbf{x}) > 0.5$, as a function of the complexity parameter, and compare it with the error of the Bayes classifier. Compare the resulting curves with those obtained for the squared error in (2.4). Is the optimal complexity the same for both losses? Discuss.

⁴You can use its implementation in scikit-learn.