ELEN0062 - Introduction to machine learning
Project 2 - Bias and variance analysis

François Rozet (s161024)
Adrien Schoffeniels (s162843)

November 16, 2019

1 Bayes model and residual error in classification

(a) Using the conditional probability definition, the Bayes model becomes

$$\begin{aligned} h_b(x_0, x_1) &= \text{argmax}_y \, P_y(y \mid x_0, x_1) \\ &= \text{argmax}_y \, \frac{P_y(y)}{p_{x_0,x_1}(x_0, x_1)} p_{x_0,x_1}(x_0, x_1 \mid y) \\ &= \text{argmax}_y \, P_y(y) \, p_{x_0,x_1}(x_0, x_1 \mid y) \end{aligned}$$

where p represents a density probability and P a probability. But we know that

$$x_0 = r \cos\alpha \text{ and } x_1 = r \sin\alpha$$

with $r \in \mathbb{R}$ and $\alpha \in [0, 2\pi]$. Therefore,

$$p_{x_0,x_1}(x_0, x_1 \mid y) = \sum_i \frac{p_{r,\alpha}\left(r_i, \alpha_i \mid y\right)}{\left|\det J\left(r_i, \alpha_i\right)\right|}$$

where J is the Jacobian matrix of the transformation $(r, \alpha) \mapsto (x_0, x_1)$, i.e

$$J(r, \alpha) = \begin{pmatrix} \cos\alpha & -r\sin\alpha \\ \sin\alpha & r\cos\alpha \end{pmatrix}$$

and where all pairs (r_i, α_i) are solutions of

$$x_0 = r_i \cos\alpha_i \text{ and } x_1 = r_i \sin\alpha_i \, .$$

There exist two such pairs :

$$\left(\sqrt{{x_0}^2 + {x_1}^2}, \text{arctan2}(x_1, x_0)\right) \text{ and } \left(-\sqrt{{x_0}^2 + {x_1}^2}, \text{arctan2}(-x_1, -x_0)\right) .$$

Also,

$$\det J(r, \alpha) = \begin{vmatrix} \cos\alpha & -r\sin\alpha \\ \sin\alpha & r\cos\alpha \end{vmatrix} = r \, .$$

Hence,

$$p_{x_0,x_1}(x_0, x_1 \mid y) = \frac{1}{\bar{r}} \left[p_{r,\alpha}\left(\bar{r}, \bar{\alpha}_+ \mid y\right) + p_{r,\alpha}\left(-\bar{r}, \bar{\alpha}_- \mid y\right)\right]$$

where $\bar{r} = \sqrt{{x_0}^2 + {x_1}^2}$ and $\bar{\alpha}_{\pm} = \arctan2(\pm x_1, \pm x_0)$.

But, from the statement, it is known that

$$y \sim \mathcal{U}\{-1, +1\} \qquad r \sim \mathcal{N}(R^y, \sigma^2) \qquad \alpha \sim \mathcal{U}(0, 2\pi)$$

which yield

$$\begin{aligned} P_y(y) &= \frac{1}{2} \\ p_\alpha(\alpha \mid y) &= p_\alpha(\alpha) = \frac{1}{2\pi} \\ p_r(r \mid y) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r - R^y)^2}{2\sigma^2}\right) \\ p_{r,\alpha}(r, \alpha \mid y) &= p_r(r \mid y)\, p_\alpha(\alpha \mid y)\,. \end{aligned}$$

Then, injecting all the above in the previous expression of $h_b(x_0, x_1)$, one obtains

$$\begin{aligned} h_b(x_0, x_1) &= \text{argmax}_y \frac{1}{4\pi} \frac{1}{\bar{r}} \frac{1}{\sqrt{2\pi\sigma^2}} \left[\exp\left(-\frac{(\bar{r} - R^y)^2}{2\sigma^2}\right) + \exp\left(-\frac{(\bar{r} + R^y)^2}{2\sigma^2}\right)\right] \\ &= \text{argmax}_y \exp\left(-\frac{(\bar{r} - R^y)^2}{2\sigma^2}\right) + \exp\left(-\frac{(\bar{r} + R^y)^2}{2\sigma^2}\right)\,. \end{aligned}$$

Therefore, $h_b(x_0, x_1) = 1$ if $\bar{r}$ is greater than z such that

$$\begin{aligned} &\exp\left(-\frac{(z - R^+)^2}{2\sigma^2}\right) + \exp\left(-\frac{(z + R^+)^2}{2\sigma^2}\right) \\ &\qquad = \exp\left(-\frac{(z - R^-)^2}{2\sigma^2}\right) + \exp\left(-\frac{(z + R^-)^2}{2\sigma^2}\right)\,. \end{aligned}$$

Supposing $R^+ > R^- \gg 0$, one can show that $z \in [R^-, R^+]$. However, since $\sigma^2 = 0.1$, the exponentials of $-\frac{(z+R^y)^2}{2\sigma^2}$ are negligible in front of the exponentials of $-\frac{(z-R^y)^2}{2\sigma^2}$ within the interval $[R^-, R^+]$.

Thus, using this approximation,

$$\begin{aligned} & \exp\left(-\frac{(z - R^+)^2}{2\sigma^2}\right) = \exp\left(-\frac{(z - R^-)^2}{2\sigma^2}\right) \\ \Leftrightarrow \quad & -\frac{(z - R^+)^2}{2\sigma^2} = -\frac{(z - R^-)^2}{2\sigma^2} \\ \Leftrightarrow \quad & (z - R^+)^2 = (z - R^-)^2 \\ \Leftrightarrow \quad & z^2 - 2zR^+ + {R^+}^2 = z^2 - 2zR^- + {R^-}^2 \\ \Rightarrow \quad & z = \frac{{R^+}^2 - {R^-}^2}{2(R^+ - R^-)} = \frac{R^- + R^+}{2}\,. \end{aligned}$$

Finally,

$$h_b(x_0, x_1) = \begin{cases} +1 & \text{if } \bar{r} \geq z = R^{\pm} \\ -1 & \text{else} \end{cases} \tag{1}$$

where $R^{\pm} = \frac{R^- + R^+}{2}$.

(b) Using the fact that

$$E\{1(x)\} = P(x)$$

the residual error becomes

$$\begin{aligned} E_{x_0,x_1,y}\{1(y \neq h_b(x_0,x_1))\} &= P(y \neq h_b(x_0,x_1)) \\ &= \sum_y P_y(y)\, P(y \neq h_b(x_0,x_1) \mid y) \\ &= P_y(-1)\, P(\bar{r} > R^{\pm} \mid -1) + P_y(1)\, P(\bar{r} \leq R^{\pm} \mid 1) \end{aligned}$$

But $\bar{r} = |r|$ and $P_y(-1) = P_y(1) = \frac{1}{2}$. Therefore,

$$\begin{aligned} &E_{x_0,x_1,y}\{1(y \neq h_b(x_0,x_1))\} \\ &\quad = \frac{1}{2}\left[1 - P(-R^{\pm} \leq r \leq R^{\pm} \mid -1)\right] + \frac{1}{2}P(-R^{\pm} \leq r \leq R^{\pm} \mid 1) \\ &\quad = \frac{1}{2} + \frac{1}{2}\int_{-R^{\pm}}^{R^{\pm}} [p_r(r \mid 1) - p_r(r \mid -1)]\,\mathrm{d}r \\ &\quad = \frac{1}{2} + \frac{1}{2}\int_{-R^{\pm}}^{R^{\pm}} \frac{1}{\sqrt{2\pi\sigma^2}}\left[\exp\left(-\frac{(r-R^{+})^2}{2\sigma^2}\right) - \exp\left(-\frac{(r-R^{-})^2}{2\sigma^2}\right)\right]\mathrm{d}r \end{aligned} \quad (2)$$

whose second term is a finite (Gaussian) integral computable numerically. Indeed, using Python's `scipy.integrate.quad` function,

$$E_{x_0,x_1,y}\{1(y \neq h_b(x_0,x_1))\} = 0.0569 \quad (3)$$

2 Bias and variance of the *kNN* algorithm

(a) As seen in the theoretical lectures,

$$\begin{aligned} &E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{(y-\hat{y}(\boldsymbol{x}))^2\right\}\right\} \\ &\quad = E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{(y - E_{y|\boldsymbol{x}}\{y\} + E_{y|\boldsymbol{x}}\{y\} - \hat{y}(\boldsymbol{x}))^2\right\}\right\} \\ &\quad = E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{(y - E_{y|\boldsymbol{x}}\{y\})^2\right\}\right\} + E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{(E_{y|\boldsymbol{x}}\{y\} - \hat{y}(\boldsymbol{x}))^2\right\}\right\} \\ &\quad + E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{2(y - E_{y|\boldsymbol{x}}\{y\})(E_{y|\boldsymbol{x}}\{y\} - \hat{y}(\boldsymbol{x}))\right\}\right\} \\ &\quad = E_{y|\boldsymbol{x}}\left\{(y - E_{y|\boldsymbol{x}}\{y\})^2\right\} + E_{LS}\left\{(E_{y|\boldsymbol{x}}\{y\} - \hat{y}(\boldsymbol{x}))^2\right\} + 0 \\ &\quad = V_{y|\boldsymbol{x}}\{y\} + E_{LS}\left\{(E_{y|\boldsymbol{x}}\{y\} - \hat{y}(\boldsymbol{x}))^2\right\} \\ &\quad = V_{y|\boldsymbol{x}}\{y\} + E_{LS}\left\{(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\} + E_{LS}\{\hat{y}(\boldsymbol{x})\} - \hat{y}(\boldsymbol{x}))^2\right\} \\ &\quad = V_{y|\boldsymbol{x}}\{y\} + E_{LS}\left\{(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\})^2\right\} + E_{LS}\left\{(E_{LS}\{\hat{y}(\boldsymbol{x})\} - \hat{y}(\boldsymbol{x}))^2\right\} \\ &\quad + E_{LS}\left\{2(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\})(E_{LS}\{\hat{y}(\boldsymbol{x})\} - \hat{y}(\boldsymbol{x}))\right\} \\ &\quad = V_{y|\boldsymbol{x}}\{y\} + \left(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\}\right)^2 + E_{LS}\left\{(\hat{y}(\boldsymbol{x}) - E_{LS}\{\hat{y}(\boldsymbol{x})\})^2\right\} + 0 \\ &\quad = V_{y|\boldsymbol{x}}\{y\} + \left(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\}\right)^2 + V_{LS}\{\hat{y}(\boldsymbol{x})\} \end{aligned}$$

where, knowing $y = f(\boldsymbol{x}) + \epsilon$,

$$\hat{y}(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}; LS, k) = \frac{1}{k}\sum_{l=1}^{k} f(\boldsymbol{x}_{(l)}) + \epsilon_l\,.$$

Keeping in mind that the expectation of a constant[1] is itself and its variance is 0,

$$\begin{aligned}
V_{y|\boldsymbol{x}}\{y\} &= V_{y|\boldsymbol{x}}\{f(\boldsymbol{x})\} + V_{y|\boldsymbol{x}}\{\epsilon\} = 0 + \sigma^2 \\
V_{LS}\{\hat{y}(\boldsymbol{x})\} &= \frac{1}{k^2}\sum_{l=1}^{k}\left(V_{LS}\left\{f(\boldsymbol{x}_{(l)})\right\} + V_{LS}\{\epsilon_l\}\right) = \frac{1}{k^2}\sum_{l=1}^{k}\left(0+\sigma^2\right) \\
E_{y|\boldsymbol{x}}\{y\} &= E_{y|\boldsymbol{x}}\{f(\boldsymbol{x})\} + E_{y|\boldsymbol{x}}\{\epsilon\} = f(\boldsymbol{x}) + 0 \\
E_{LS}\{\hat{y}(\boldsymbol{x})\} &= \frac{1}{k}\sum_{l=1}^{k}\left(E_{LS}\left\{f(\boldsymbol{x}_{(l)})\right\} + E_{LS}\{\epsilon_l\}\right) = \frac{1}{k}\sum_{l=1}^{k}\left(f(\boldsymbol{x}_{(l)}) + 0\right)
\end{aligned}$$

which trivially yields

$$E_{LS}\left\{E_{y|\boldsymbol{x}}\left\{(y-\hat{y}(\boldsymbol{x}))^2\right\}\right\} = \sigma^2 + \left[f(\boldsymbol{x}) - \frac{1}{k}\sum_{l=1}^{k} f(\boldsymbol{x}_{(l)})\right]^2 + \frac{\sigma^2}{k} \tag{4}$$

supposing $k \leq N$.

(b) The bias-variance decomposition (4) is composed of three parts :

1. σ^2, the *noise* of the class y knowing $\boldsymbol{x}$. This term is independent of the classification model and therefore of k. Indeed, σ^2 doesn't vary with k.
2. $f(\boldsymbol{x}) - \frac{1}{k}\sum_{l=1}^{k} f(\boldsymbol{x}_{(l)})$, the *bias* (squared in the decomposition) of the model. The right term is the mean of f over the k nearest neighbours of $\boldsymbol{x}$.

   When k increases, this mean tends towards the mean of f over all input values and reaches it for $k = N$. If in addition $N \to \infty$, the bias becomes the difference between $f(\boldsymbol{x})$ and its mean $E_{\boldsymbol{x}}\{f(\boldsymbol{x})\}$.
3. $\frac{\sigma^2}{k}$, the *variance* of the model. This term corresponds to the variance of k independent drawing of ϵ which obviously decreases when k increases.

3 Bias and variance estimation

(a) Let $LS = \{(\boldsymbol{x}_i, y_i) \mid i = 1, \ldots, N\}$ (with $\boldsymbol{x}_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$) denote a given learning sample (of size N) and $\hat{y}(\boldsymbol{x})$ denote the function learned from LS by the given supervised algorithm. As seen before, it is known that

$$\begin{aligned}
\text{noise}(\boldsymbol{x}) &= V_{y|\boldsymbol{x}}\{y\} = E_{y|\boldsymbol{x}}\left\{\left(y - E_{y|\boldsymbol{x}}\{y\}\right)^2\right\} \\
\text{bias}^2(\boldsymbol{x}) &= \left(E_{y|\boldsymbol{x}}\{y\} - E_{LS}\{\hat{y}(\boldsymbol{x})\}\right)^2 \\
\text{variance}(\boldsymbol{x}) &= V_{LS}\{\hat{y}(\boldsymbol{x})\} = E_{LS}\left\{\left(\hat{y}(\boldsymbol{x}) - E_{LS}\{\hat{y}(\boldsymbol{x})\}\right)^2\right\}
\end{aligned}$$

Thus, in order to estimate the noise (residual error), bias and variance, it is necessary to approximate the expectations $E_{LS}\{\cdot\}$ and $E_{y|\boldsymbol{x}}\{\cdot\}$:

- $E_{LS}\{\cdot\}$ is approximated by the average of $\cdot$ over multiple learning samples.

[1] Evaluating the model at $\boldsymbol{x}$, all $\boldsymbol{x}_l$ are constant with respect to LS.

- $E_{y|\boldsymbol{x}}\{\cdot\}$ is approximated by the average of $\cdot$ over a (learning) sample in which $\boldsymbol{x_i} = \boldsymbol{x}$ for all i.

Therefore, for all $\boldsymbol{x}_0$ represented in LS, the protocol is

1. Select each pair $(\boldsymbol{x}_i, y_i)$ of LS such that $\boldsymbol{x}_i = \boldsymbol{x}_0$.
2. Compute the mean and variance (using Bessel's correction) of the selected classes (y_i), respectively estimating $E_{y|\boldsymbol{x}_0}\{y\}$ and the noise $V_{y|\boldsymbol{x}_0}\{y\}$.
3. Divide LS in p (by default 20) disjoint subsets and train the supervised algorithm on each of them.
4. Compute the mean and variance (using Bessel's correction) of the p learned function evaluated at $\boldsymbol{x}_0$ $(\hat{y}(\boldsymbol{x}_0))$, respectively estimating $E_{LS}\{\hat{y}(\boldsymbol{x})\}$ and the variance $V_{LS}\{\hat{y}(\boldsymbol{x})\}$.
5. Estimate the squared-bias by squaring the difference of the two means computed previously.
6. The expected error is computed as the sum of the noise, squared-bias and variance.

It should be noted that if $\boldsymbol{x}_0$ is represented only once in LS, the noise is undefined (NaN for convenience) because of Bessel's correction.

(b) To estimate the mean values of the noise, squared-bias, variance and expected error one can apply the above protocol (3(a)) to all $\boldsymbol{x}_0$ represented in LS and then compute the average values of obtained noises[2], squared-biases, variances and expected errors.

(c) Yes, they are still appropriate. Indeed, these protocols are specifically designed to work with a finite given learning sample. The main drawback of such protocol is that it cannot compute the bias-variance decomposition of points which are not represented in the learning sample. However, for a large enough learning set (compared to the discrete domain size) it is quite likely that most of the domain is represented.

(d) The two chosen methods are the *Ridge* (linear) and the *K-Neighbors* (non-linear) regression. Both regressors are implemented within the `scikit` library. The results of the two methods are respectively influenced by α, the regularization strength, and k, the number of considered neighbors.

After generating a random learning set LS[3] of size $N = 10^4$ without irrelevant variables ($q = 0$) using the `make_data` generator, both methods were put to test.

[2]For the noise, one shouldn't consider undefined values.

[3]Instead of $x \sim \mathcal{U}(-10, 10)$, the discretization $x \sim \mathcal{U}\{-10, -9.9, \ldots, 9.9, 10\}$ has been considered. Without this domain approximation, the noise wouldn't be estimable.

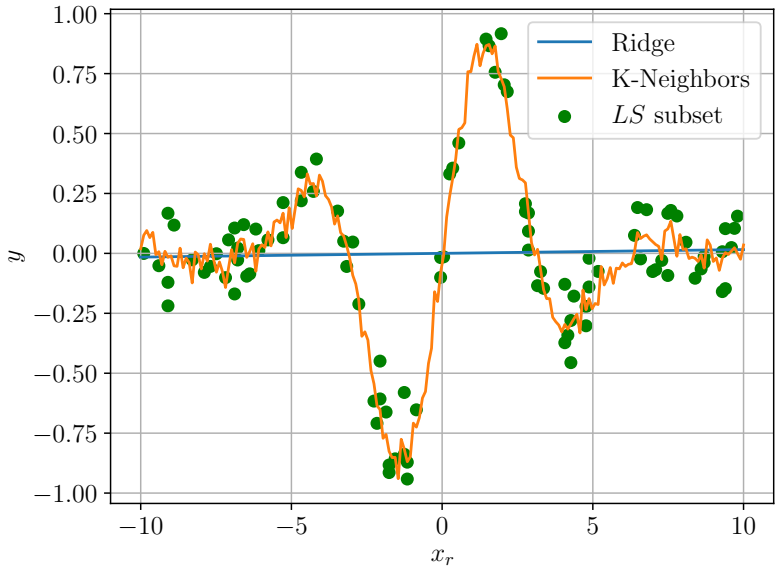


Figure 1 – Predictions of the Ridge ($\alpha = 1$) and K-Neighbors ($k = 5$) regressors trained with LS.

As one can see in Figure 1, if the K-Neighbors method predicts quite well the overall shape of $f(x_r)$, it is not at all the case of the Ridge method which predicts $\hat{y}(x_r) \simeq 0$. This behavior was predictable since the more the linear model is tilted, the worse it predicts the most extreme (near ± 10) points. Conversely, it doesn't predict that badly the center (near 0) points with an almost null slope.

Then, for both methods, a bias-variance decomposition (cf. Figures 2 and 3) has been performed using the proposed protocol 3(a) for each $\boldsymbol{x}$ represented in LS.

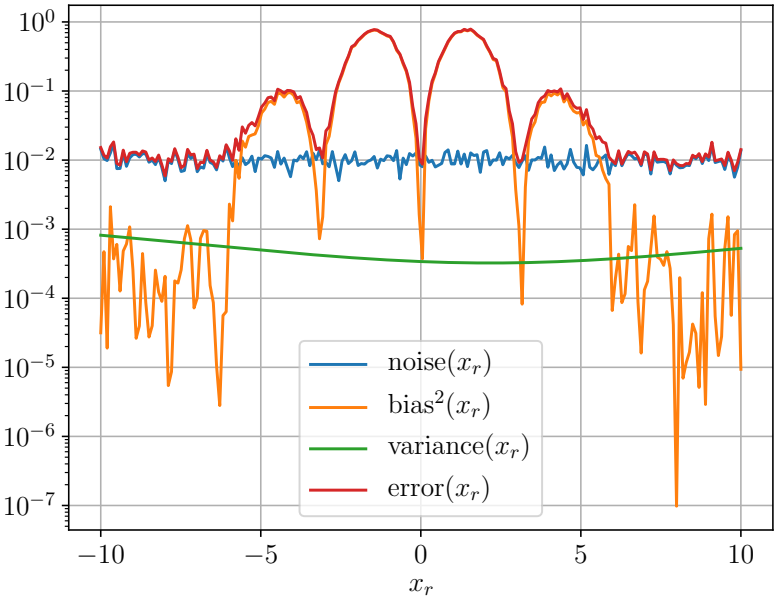


Figure 2 – Ridge regressor ($\alpha = 1$) trained with LS bias-variance decomposition as a function of x_r.

For the linear regression method (cf. Figure 2), as expected from the pace of its predictions, most of the error is concentrated around $x_r = 0$. Indeed, near the center, the (squared-)bias of the predictions is much greater due to the decreasing exponential and is, therefore, the main component of the global error.

One can also observe in Figure 2 that the variance is quite low which means that the trained regressor doesn't vary much on the $p = 20$ learning sets of size N/p. Yet, the extremities varies more than the center which is very likely the result of a (slightly) varying slope.

Finally, and this obviously hold for both methods, the estimated noise oscillates near the actual noise of the learning set, i.e. 0.1^2.

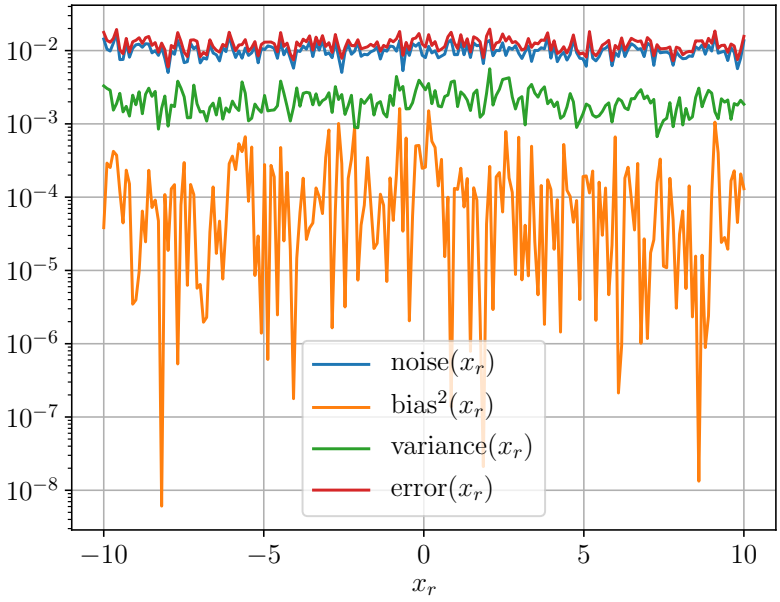


Figure 3 – K-Neighbors ($k = 5$) trained with LS bias-variance decomposition as a function of x_r.

For the non-linear regression method (cf. Figure 3), neither the bias nor the variance seem significantly dependent on x_r. Moreover, the variance seems to oscillate around the value 2×10^{-3}. This observation is supported by the theory[4] (cf. section 2) since the variance should equal $\frac{\text{noise}}{k} = \frac{0.1^2}{5}$. Concerning the bias, it is not surprising to obtain such low values since the predictions (cf. Figure 1) are quite accurate.

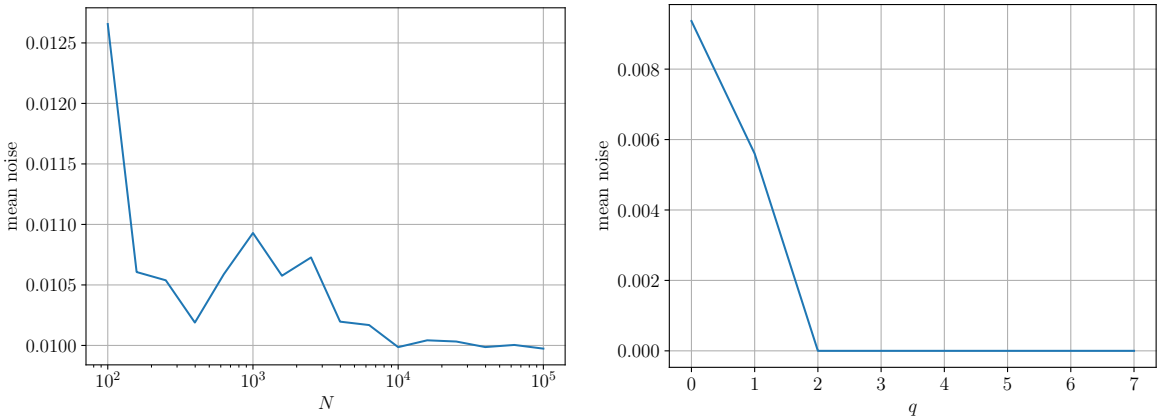
Because both bias and variance are lower than the noise, the later actually is the main component of the error. For this regression problem, the K-Neighbors ($k = 5$) regressor is preferable to the Ridge ($\alpha = 1$) one.

(e) To estimate the mean values of the noise, squared-bias, variance and expected error, the protocol 3(b) has been used. These values are dependent on the size of the

[4]It is not the exact same case as in section 2 since the input values $\boldsymbol{x}$ are not the same in each learning sample. However, because the size of each learning sample is far bigger (more than k times) than the size of the discrete domain, it is very likely that the k selected neighbors are the same; which is more or less the same assumption.

learning sample N, the number of irrelevant variables q and the complexity (or parameters) of the regressor (α or k). For both regression methods, the mean values have been computed as functions of N, q and the complexity (one at a time, while the two others are constant).

Starting by the noise (cf. Figure 4), one can observe that it tends towards its actual value (0.1^2) when N increases. However, when q increases, the size of the domain explodes exponentially and quickly surpasses N. Therefore from $q = 2$, there isn't anymore twice the same input $\boldsymbol{x}$ in the learning sample and all estimated noises are undefined and so do the mean noise. For convenience this undefined mean noise is represented as 0.



(a) $q = 0$ (b) $N = 10^3$

Figure 4 – Mean noise as functions of N and q.

It is not useful to plot the noise as a functions of the regressors complexity since it is independent of it, which is not the case of all other mean quantities (cf. 5).

For the linear regression method, one can see that the three parameters (N, q and α) barely have an influence on the overall accuracy, especially α that has literally none. Conversely, as N and q increases, the mean variance of the model respectively drops and rises.

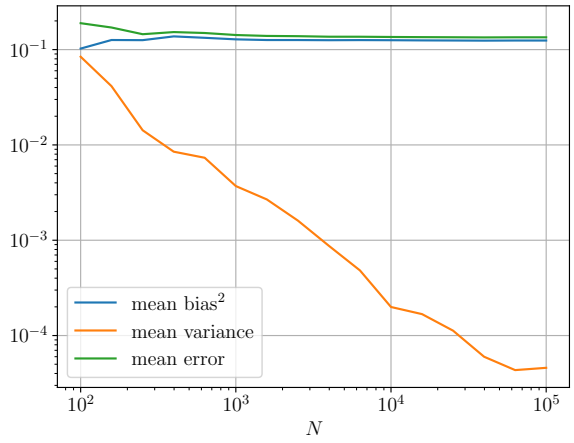
Indeed, the coefficients of the Ridge regressor are heavily influenced by the spacial distribution of the learning set depending both on N and q. Yet, the mean variance is too weak in comparison to the large mean squared-bias to be noticeable in the global mean error.

For the non-linear regression method, it can be observed that all three parameters (N, q and k) are determinant. As expected, when N increases, the K-Neighbors regressor improves its accuracy (up to a certain lower limit) as a result of the more homogeneous coverage of the domain which directly influences the bias (nearer neighbors) and the variance (cf. footnote 4).

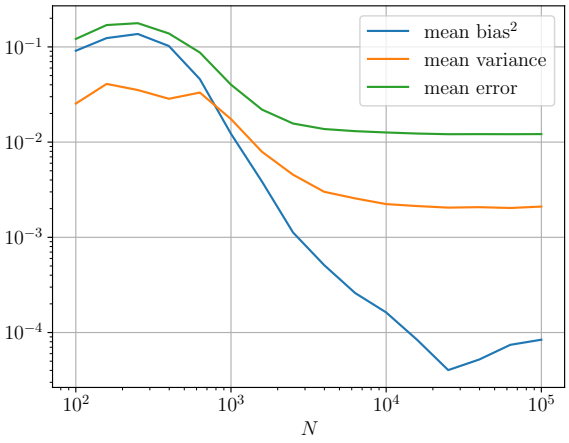
When q increases, the opposite effect occurs. Indeed, introducing irrelevant variables is misleading for the K-Neighbors algorithm since close neighbors in $q+1$ dimensions might be far away in the first one (the relevant one). Therefore, if each input $\boldsymbol{x}$

isn't represented multiple times within the learning set, the k nearest neighbors are selected more or less randomly regarding x_r and $\hat{y}(\boldsymbol{x})$ tends toward the mean of y, that is 0. Actually, one can draw a link between this behaviour and the predictions of the Ridge regressor. Indeed, the mean error and mean squared-bias of the K-Neighbors regressor for $q > 0$ and those of the Ridge regressor are very similar.

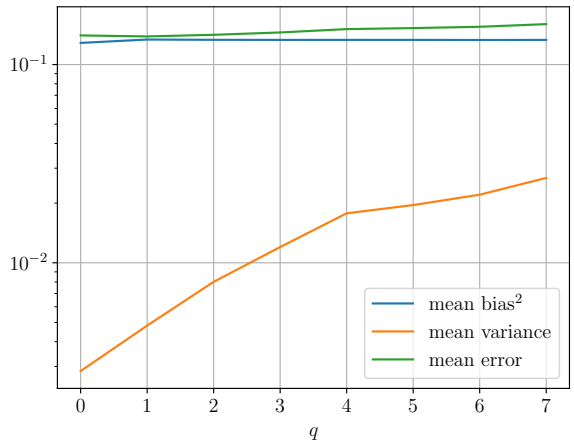
The exact same behaviour can be observed as k increases : $\hat{y}(\boldsymbol{x})$ tends toward the mean of y, i.e. 0. However, if the variance remains more or less constant when q grows, it is not the case with k whose rise makes the variance decrease.



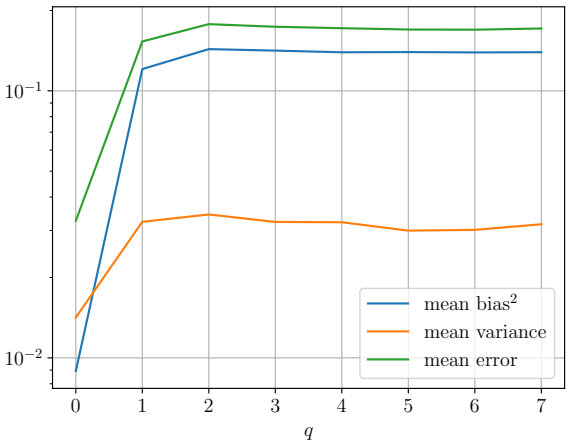
(a) Ridge regressor, $q = 0$, $\alpha = 1$



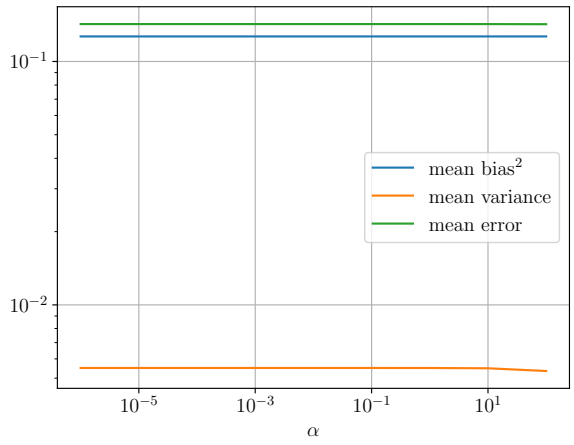
(b) K-Neighbors regressor, $q = 0$, $k = 5$



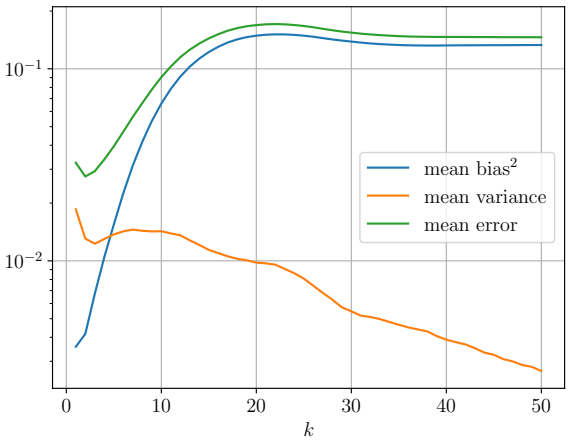
(c) Ridge regressor, $N = 10^3$, $\alpha = 1$



(d) K-Neighbors regressor, $N = 10^3$, $k = 5$



(e) Ridge regressor, $N = 10^3$, $q = 0$



(f) K-Neighbors regressor, $N = 10^3$, $q = 0$

Figure 5 – Mean squared-bias, variance and expected error as functions of N, q and the complexity of the regressors.