ELEN0062 - Introduction to Machine Learning
Project 2 - Bias and variance analysis

PARING MADANASSONO POUEZI (S228407) and DUY VU DINH (S2401627)

1 ANALYTICAL DERIVATIONS

(1.1) **Show that the expected generalization error of the k-Nearest Neighbours (kNN) algorithm at some point x can be decomposed as follows**

$$\mathbb{E}_{LS}\{\mathbb{E}_{y|x}\{(y-\hat{y}(x;LS,k))^2\}\} = \sigma^2 + \left[f(x) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right]^2 + \frac{\sigma^2}{k} \qquad (1)$$

where $\hat{y}(x;LS,k)$ is the prediction of the kNN method for a point x given a learning sample LS (of size N), $x_{(l)}$ denotes the l-th nearest neighbours of x in LS and k is the number of neighbours.

According to the lectures, it has been demonstrated that generally the expected generalization error could be analyzed:

$$\mathbb{E}_{LS}\{\mathbb{E}_{y|x}\{(y-\hat{y})^2\}\} = \mathbb{E}_y\{(y-\mathbb{E}_y\{y\})^2\} + (\mathbb{E}_y\{y\} - \mathbb{E}_{LS}\{\hat{y}\})^2 + \mathbb{E}_{LS}\{(\hat{y}-\mathbb{E}_{LS}\{\hat{y}\})^2\}$$

This error can be rewritten as:

$$\mathbb{E}_{LS}\{\mathbb{E}_{y|x}\{(y-\hat{y})^2\}\} = var_y\{y\} + bias^2 + var_{LS}\{\hat{y}\}$$

where:

- $(\mathbb{E}_y\{(y-\mathbb{E}_y\{y\})^2\} = var_y\{y\})$ is the residual error due to inherent noise in the data,
- $((\mathbb{E}_y\{y\} - \mathbb{E}_{LS}\{\hat{y}\}) = bias)$ is the bias,
- and $(\mathbb{E}_{LS}\{(\hat{y}-\mathbb{E}_{LS}\{\hat{y}\})^2\} = var_{LS}\{\hat{y}\})$ is the estimator variance which quantifies how much the estimation varies from one learning sample to another.

For this problem, these terms could be transformed:

- The residual error:
$$\mathbb{E}_y\{(y-\mathbb{E}_y\{y\})^2\} = var_y\{y\} = \sigma^2$$
- The bias:
$$(\mathbb{E}_y\{y\} - \mathbb{E}_{LS}\{\hat{y}\}) = bias = \left[f(x) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right]$$
- The estimator variance:
$$\mathbb{E}_{LS}\{(\hat{y}-\mathbb{E}_{LS}\{\hat{y}\})^2\} = var_{LS}\{\hat{y}\} = var\left(\frac{1}{k}\sum_{l=1}^{k} x_{(l)}\right) = \frac{1}{k^2}\sum_{l=1}^{k} var(x_{(l)}) = \frac{1}{k^2}\sum_{l=1}^{k}\sigma^2 = \frac{\sigma^2}{k}$$

Therefore, the expected generalization error at a point x can be explicitly written as:

$$\mathbb{E}_{LS}\{\mathbb{E}_{y|x}\{(y-\hat{y}(x;LS,k))^2\}\} = \sigma^2 + \left[f(x) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right]^2 + \frac{\sigma^2}{k}$$

(1.2) **Let us now assume that the problem is unidimensional, i.e., $x \in \mathbb{R}$, that $f(x) = x^2$, and that for a given N, the inputs of the training examples are defined as follows:**

$$\{x_1, \dots, x_N\} = \{0\} \cup \left\{\frac{i}{N'}, i = 1, \dots, N'\right\} \cup \left\{-\frac{i}{N'}, i = 1, \dots, N'\right\} \tag{2}$$

with $N' > 1$ an arbitrary integer value. The $N = 2N' + 1$ points thus form an uniform grid in $[-1; +1]$. Using the result in Sec. (1.1), express analytically the bias and variance terms of the kNN method at $x = 0$, as a function of k, N and σ. You can assume that k only takes the values $k = 2k' + 1$ with $k' > 0$ an integer value chosen in $\{0, 1, \dots, N'\}$.

The squared bias term:

$$bias^2 = \left[f(x) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right]^2$$

At $x = 0$, $f(0) = 0^2 = 0$ and $f(x_{(l)}) = (x_{(l)})^2$. Then:

$$bias^2 = \left[f(0) - \frac{1}{k}\sum_{l=1}^{k} f(x_{(l)})\right]^2 = \left[0 - \frac{1}{k}\sum_{l=1}^{k}(x_{(l)})^2\right]^2 = \left[\frac{1}{k}\sum_{l=1}^{k}(x_{(l)})^2\right]^2$$

According to Eq. 2, the input points are symmetrically arranged around $x = 0$. The nearest neighbors to $x = 0$ are: 0, $\pm\frac{1}{N'}$, $\pm\frac{2}{N'}$, ..., $\pm\frac{k'}{N'}$, with $k = 2k' + 1$. Substituting the symmetric structure of the neighbors, the bias simplifies to:

$$bias^2 = \left[\frac{1}{k}\sum_{i=1}^{k'}\left(\frac{-i}{N'}\right)^2 + \frac{1}{k}\sum_{i=1}^{k'}\left(\frac{i}{N'}\right)^2\right]^2 = \left[\frac{1}{k}\sum_{i=1}^{k'}2\left(\frac{i}{N'}\right)^2\right]^2 = \left[2 \cdot \frac{1}{k} \cdot \frac{1}{(N')^2} \cdot \sum_{i=1}^{k'} i^2\right]^2$$

Using the formula for $\sum_{i=1}^{k'} i^2 = \frac{k'(k'+1)(2k'+1)}{6} = \frac{\left(\frac{k-1}{2}\right)\left(\frac{k+1}{2}\right)k}{6}$ with $k' = \frac{k-1}{2}$ and substituting $N' = \frac{N-1}{2}$, we can rewrite the bias in terms of N.

The squared bias becomes:

$$bias^2 = \left[\frac{1}{k} \cdot \frac{2}{\left(\frac{N-1}{2}\right)^2} \cdot \frac{\left(\frac{k-1}{2}\right)\left(\frac{k+1}{2}\right)k}{6}\right]^2 = \left[\frac{1}{k} \cdot \frac{8}{(N-1)^2} \cdot \frac{k(k^2-1)}{24}\right]^2 = \left[\frac{k^2-1}{3(N-1)^2}\right]^2$$

Therefore, the bias term for kNN at $x = 0$ in terms of N, k, and σ is:

$$bias^2 = \left[\frac{k^2-1}{3(N-1)^2}\right]^2$$

The estimator variance term for kNN at $x = 0$, expressed in terms of k and σ, as demonstrated at the previous question, is:

$$var_{LS}\{\hat{y}\} = \frac{\sigma^2}{k}$$

(1.3) **Discuss the impact of N, k, and σ on bias and variance. Are there some surprising or missing dependencies? If so, try and explain them.**

The impact of the parameters on bias:

- N: As N increases, the bias decreases. A larger N creates a denser grid of training points, enabling kNN to better approximate the true function. The bias decreases proportionally to $(N-1)^4$, highlighting the importance of larger datasets for reducing systematic error.
- k: Increasing k increases the bias. Larger k incorporates neighbors farther from the target point, resulting in less accurate local approximations of $f(x)$. The bias grows quadratically with k^2, making it crucial to choose k carefully.
- σ: Noise does not directly affect bias, as bias is determined by the systematic deviation of the model from the true function which is independent of the data randomness.

The impact of the parameters on variance:

- N: Variance is independent of N. This is because the variance is driven by the number of neighbors k, not the total number of points in the dataset.
- k: Increasing k reduces variance. Averaging the outputs of more neighbors smooths the prediction, reducing the impact of random fluctuations in individual neighbors' outputs. Variance decreases inversely with k, showcasing a clear trade-off with bias.
- σ: Variance increases with σ^2, as higher σ levels in the data lead to greater variability in the model's predictions.

Surprising dependencies:

- Small k: A small k reduces bias but increases variance. Predictions rely on fewer, closer neighbors, making the model more sensitive to noise and fluctuations in the data.
- Large k: A large k reduces variance but increases bias. The model oversmooths the data, losing its ability to capture fine details in the true function.
- Optimal k: The optimal k balances bias and variance, minimizing the total generalization error.

Missing dependencies:

- No direct dependency of bias on σ: While variance increases with σ^2, bias does not. Bias is determined by the systematic difference between the model's expected predictions and the true function.
- Variance not decreasing with N: In kNN case, variance remains unchanged for a fixed k, as predictions are based on the k-nearest neighbors, not the total dataset size. This is a limitation of kNN: even with large datasets, variance will not improve unless k is adjusted.
- Bias independent of dataset distribution details: Bias depends on the relative placement of neighbors around the target point, but not directly on the dataset's overall distribution. In non-uniform datasets, this assumption may lead to unexpected bias behaviors, as the formula does not account for uneven sampling in the input space.

(1.4) **For all combinations of $N \in \{25, 50\}$ and $\sigma \in \{0.0, 0.1, 0.2\}$, determine the value k^* of k that minimizes the expected generalization error at $x = 0$.**

To minimize the expected generalization error, we calculate its derivative with respect to k, denoted as $\mathbb{E}(k)$ for simplification, and solve for k^*, the value of k that minimizes the error. The expected generalization error function is given by:

$$\mathbb{E}(k) = \mathbb{E}_{LS}\{\mathbb{E}_{y|x}\{(y - \hat{y}(x; LS, k))^2\}\} = \sigma^2 + \left[\frac{k^2 - 1}{3(N-1)^2}\right]^2 + \frac{\sigma^2}{k}$$

Taking the derivative of Error(k) with respect to k:

$$\frac{d}{dk}\mathbb{E}(k) = \frac{d}{dk}\left[\sigma^2 + \left(\frac{k^2 - 1}{3(N-1)^2}\right)^2 + \frac{\sigma^2}{k}\right] = 0 + 2 \cdot \left(\frac{k^2 - 1}{3(N-1)^2}\right) \cdot \frac{2k}{3(N-1)^2} - \frac{\sigma^2}{k^2} = \frac{4k(k^2 - 1)}{9(N-1)^4} - \frac{\sigma^2}{k^2}$$

To find the extreme value:

$$\frac{d}{dk}\mathbb{E}(k) = 0 \tag{3}$$

Then, the equation becomes:

$$\frac{4k(k^2 - 1)}{9(N-1)^4} - \frac{\sigma^2}{k^2} = 0 \tag{4}$$

Finally:

$$k^3(k^2 - 1) = \frac{9\sigma^2(N-1)^4}{4} \tag{5}$$

This equation must be solved numerically for each N and σ. The values of k^* and E corresponding to each pair of N and σ are show in the Table 1.

Table 1. Values of k^* and E for different N and σ.

N	σ					
	0.0		0.1		0.2	
	k^*	E	k^*	E	k^*	E
25	1	0.000	6	0.012	8	0.046
50	1	0.000	11	0.011	14	0.044

(1.5) **Discuss the impact of N and σ on k^*.**

Based on Table 1, it can be observed that:

- Impact of N on k^*: As N increases from 25 to 50, k^* increases for a given σ. This is because a larger dataset provides a denser grid of training points, allowing the kNN algorithm to incorporate more neighbors without significantly increasing bias.
- Impact of σ on k^*: Higher σ leads to larger k^*. Noise increases the variance of predictions, and increasing k helps average out these fluctuations, reducing variance.
- Impact of N on generalization error (E): For a fixed σ, increasing N reduces the error E. This is expected because a larger dataset decreases bias, and with higher k^*, the variance is controlled.
- Impact of σ on generalization error (E): As σ increases, E increases due to the inherent noise in the data. Even with optimal k^*, the variance component of the error grows with σ^2.

Conclusions:

- Larger datasets (N): Larger datasets support higher k^*, allowing the model to incorporate more neighbors while maintaining low generalization error. Increasing N improves the balance between bias and variance, leading to lower E.
- Higher noise (σ): As noise increases, larger k^* is required to average out random fluctuations. However, even with optimal k^*, the generalization error increases due to the noise component.
- Trade-off between N and σ: For low noise levels ($\sigma \longrightarrow 0$), the optimal k^* is 1, as there is no need to average over neighbors. For higher noise, both N and k^* need to be sufficiently large to minimize E.

2 EMPIRICAL ANALYSIS

(2.1) **Explain why estimating the residual error term is very difficult in this setting**

Regarding the dataset, the wine quality is ranked from 0 to 10, which means it is ordinal data. The difficulty arises from the fact that an ordinal variable has discrete values that are labeled with. The values simply represent ordered categories. Ordinal data has categories with a defined order but no measurable or consistent distance between them, complicating residual estimation. There is no clear metric for residual error in ordinal data, making it difficult to define or interpret residuals.

(2.2) **Describe a protocol to nevertheless estimate variance, the expected error, as well as the sum of the bias and the residual error from a pool *P*. Since the residual error is constant, this protocol is sufficient to assess how method hyper-parameters affect biases and variances**

Dataset partitioning and sampling:

- Large pool of data: Start with a large dataset, denoted $P = \{(x_1, y_1), \dots, (x_{N_S}, y_{N_S})\}$, where N_S is significantly larger than the learning sample size N.
- Sampling from pool: For each experiment, randomly draw multiple training samples of fixed size N from P. These samples will be used to fit the model multiple times, allowing us to estimate the variability in predictions due to different training sets.

Model training and prediction:

- For each training sample LS_i (where i ranges from 1 to M, the number of samples drawn), train the model and make predictions on a separate test set T (a subset of P not used in any of the training samples).
- This setup ensures that predictions are evaluated on data that is not part of the training samples, providing an unbiased estimate of the model's error.

Estimating expected error:

- For each test point x_j in T, calculate the mean squared error (MSE) over all M models. This provides the expected error of the model's predictions:

$$\text{Expected Error} = \frac{1}{|T|} \sum_{j=1}^{|T|} \frac{1}{M} \sum_{i=1}^{M} (y_j - \hat{y}_i(x_j))^2$$

where $\hat{y}_i(x_j)$ is the prediction for x_j by the model trained on LS_i, and y_j is the true output for x_j.

Estimating variance:

- Average this variance across all test points $x_j \in T$ to get the overall variance estimate:

$$\text{Variance} = \frac{1}{|T|} \sum_{j=1}^{|T|} \frac{1}{M} \sum_{i=1}^{M} \left(\hat{y}_i(x_j) - \frac{1}{M} \sum_{i=1}^{M} \hat{y}_i(x_j) \right)^2$$

Estimating the sum of bias and residual error:

- The sum of bias and residual error at a test point x_j can be derived by subtracting the variance from the expected error:

$$\text{Bias + Residual Error} = \text{Expected Error} - \text{Variance}$$

- This estimate provides insight into how hyperparameters affect the model's bias and the residual error together.

(2.3) **Implement and use this protocol on the given dataset to estimate the expected error, variance, and the sum of bias and residual error, for Lasso regression, kNN, and regression trees. For all three methods, plot the evolution of the three quantities as a function of its main complexity parameter (respectively, λ, k, and maximum depth) on bias and variance. You can fix the learning sample size N to 250 for this experiment. Briefly discuss the different curves with respect to the theory.**

Figure 1 shows the evolution of the three quantities as a function of its main complexity parameter.

Decision tree (Figure 1a):

- Small *max_depth*: Expected error and variance are low, but the sum of bias and residual error is high due to underfitting and generalizing over large patterns.
- Increasing *max_depth*: As depth increases, variance rises significantly due to overfitting to noise, while the sum of bias and residual decreases slightly. Beyond a depth of 12, all error values stabilize with no significant changes.
- Optimal Depth: Achieved around $max\_depth = 2$, where the model balances capturing detail and avoiding overfitting.

kNN (Figure 1b):

- Compared to the decision tree, we observe opposite behaviors for kNN.
- Small k: Expected error and variance are high due to overfitting, as predictions are highly sensitive to noise, while the sum of bias and residual is moderate.
- Increasing k: Variance decreases significantly, then reduces gradually after $n\_neighbors = 5$ due to averaging over more neighbors, but the sum of bias and residual error rises slightly all over the increasing of *n_neighbors* as the model smooths local variations.
- Optimal k: Found between $k = 12$ and $k = 16$, where variance and bias reach a balance, consistent with the bias-variance trade-off.

Lasso (Figure 1c):

- Lasso regression obviously gives better results with small values of *alpha* (λ). In terms of the evolution of variance and bias, when λ increases, we observe that it overall behaves like kNN, but with a more stabilized and small variance, and a higher bias.
- Small λ: Expected error and bias + residual are low, while variance is higher due to the model's flexibility in capturing data patterns.
- Increasing λ: From $\lambda = 10^{-3}$ to $\lambda = 5 \times 10^{-2}$, the expected error remain stable. However, during that period, variance decreases slowly and the sum of bias and variance increases oppositely. Between $\lambda = 5 \times 10^{-2}$ and $\lambda = 10^{-3}$, the expected error rises dramatically along with the sum of bias and variance, while variance continues to reduce slowly. After that, all three errors remain almost unchanged.
- Optimal λ: Around 0.01, minimizing expected error while balancing bias and variance effectively.

(2.4) **For the same three methods, show the impact of the learning sample size on bias and variance. In the case of kNN and Lasso regression, choose one particular value of k and λ respectively. In the case of regression trees, compare fully grown trees with trees of fixed depth.**

Overall, Figure 2 shows that all methods reduce the expected error with the increase of the learning sample size N.

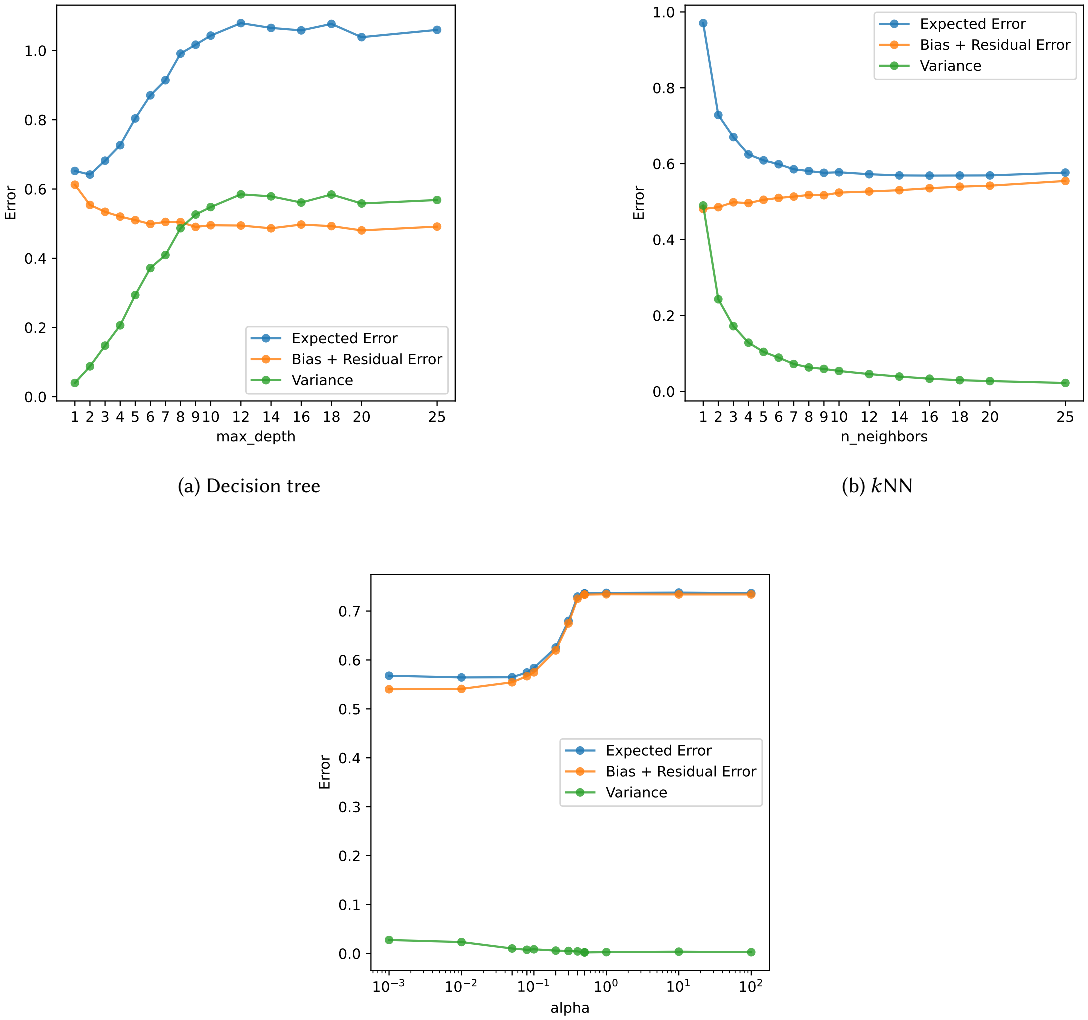
- kNN (Figure 2a): The expected error decreases steadily, with the variance decreasing significantly and the sum of bias and residual error starting at a low level and then gradually decreasing. When N is equal to the size of the learning samples, the variance reaches 0. kNN effectively balances variance and bias as N increases, using optimal data sets for improved generalization.
- Lasso (Figure 2b): Expected error decreases with N and stabilizes at a moderate level. Variance decreases sharply, becoming negligible for large datasets, while the sum of bias and residual error remains constant due to regularisation constraints. The Lasso mainly benefits from the variance reduction, but its regularisation limits its flexibility.
- Fixed-depth tree (Figure 2c): The expected error rate declines steadily with the size of the learning samples, with the variance approaching zero for large values of N. The sum of the bias and residual term error remains largely unchanged, as the fixed depth limits the model's capacity to capture complex relationships. Fixed-depth trees are stable and benefit from reduced variance but cannot reduce bias
- Fully-grown decision tree (Figure 2d): Expected error decreases more slowly and stabilizes at a higher level. Variance remains high despite larger N, while the sum of bias and residual error decreases slightly. However, when N is equal to the learning sample size, the variance decreases to 0 and the sum of bias and residual error increases, leading to an increase in expected error. Fully grown trees are highly flexible, prone to overfitting, and require very large data sets to mitigate their high variance.
- Fixed-depth trees exhibit lower error and variance, making them more robust and suitable for small to medium datasets. Fully grown trees, while flexible, suffer from persistent overfitting and require large datasets to generalize effectively. When N is equal to learning sample size, both variances come to 0, only the bias and residual error term contributes to the expected error.

(2.5) **Two so-called ensemble methods to address variance and bias are bagging ("bootstrap aggregating") and boosting. Both are based on combining weaker models to produce more accurate and stable results. Whereas bagging combines models in parallel, boosting does so in a sequential manner. Compute and discuss the evolution of bias and variance as the number of estimators increases, for both bagging and boosting, using a decision tree as your base learner. Discuss as well how the complexity of the base learner (e.g., the maximum depth) affects the ensembling results**

Regarding Figure 3, bagging and boosting show distinct effects on the evolution of errors—expected error, variance, and bias—as the number of estimators increases.

- For bagging, Figures 3a and 3c show that the primary effect is a significant reduction in variance, particularly when using high-variance base learners such as fully grown trees. This reduction leads to a notable decrease in the expected error, though the bias remains largely unchanged (approximately 0.5 in this experiment). With shallow trees, the variance starts lower and reduces modestly, resulting in diminishing improvements in the expected error.
- Boosting is highly effective at reducing bias for both types of trees. The term of bias and residual error decreases consistently until $n_estimator = 200$ (Figures 3b and 3d). However, boosting tends to increase variance as it sequentially fits the data and potentially overfits noise. Consequently, boosting reduces the expected error initially, but this improvement may plateau or even reverse at high estimator counts due to rising variance, especially for high-complexity base learners.
- Both fixed depth and fully grown trees, with their inherently high bias and low variance, benefit more from boosting. Boosting steadily reduces their bias, leading to substantial improvements in the expected error. However, in boosting, trees face a rapid increase in variance, which can offset gains from bias reduction and lead to overfitting, especially with a large number of estimators.
- Fully grown trees, on the other hand, start with low bias and high variance. They benefit most from bagging (Figure 3c), which dramatically reduces their variance while leaving their bias unchanged. This combination is particularly effective for improving expected error in bagging.

APPENDIX

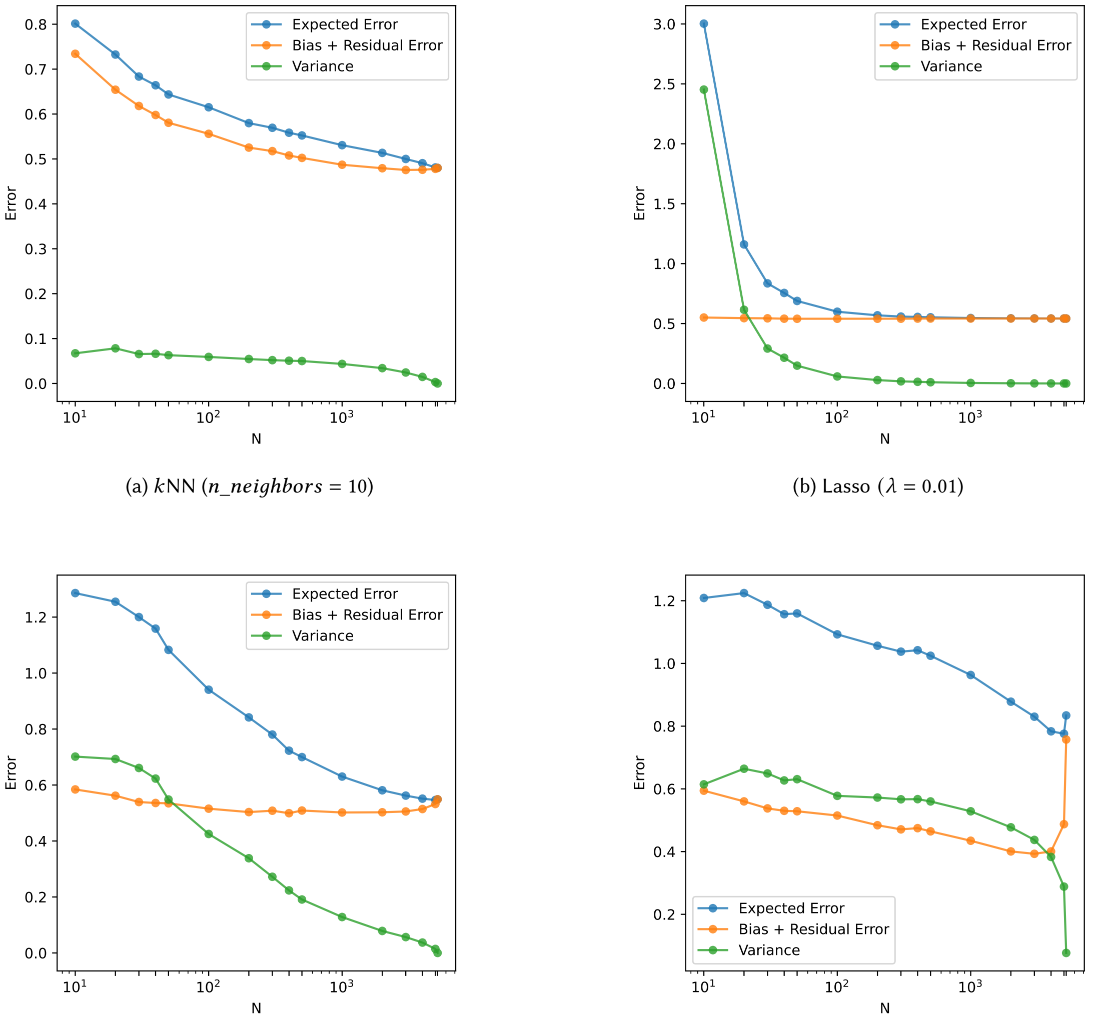


(a) Decision tree

(b) *k*NN

(c) Lasso

Fig. 1. Bias-variance complexity for question 2.3.



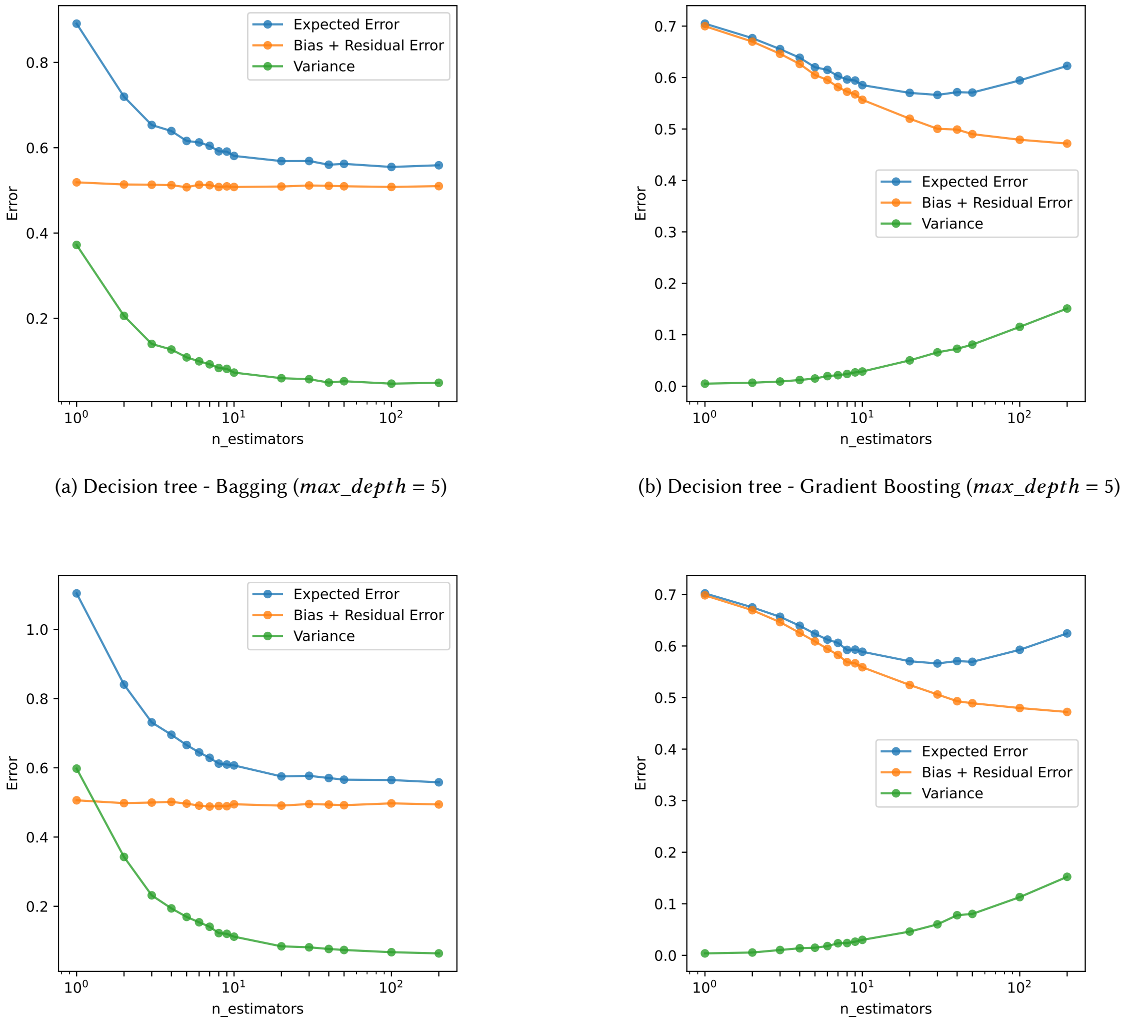
(a) kNN ($n\_neighbors = 10$)

(b) Lasso ($\lambda = 0.01$)

(c) Decision Tree ($max\_depth = 5$)

(d) Decision Tree ($max\_depth = None$)

Fig. 2. Bias-variance complexity for question 2.4.



(a) Decision tree - Bagging ($max\_depth = 5$)

(b) Decision tree - Gradient Boosting ($max\_depth = 5$)

(c) Decision tree - Bagging ($max\_depth = None$)

(d) Decision tree - Gradient Boosting ($max\_depth = None$)

Fig. 3. Bias-variance complexity for question 2.5.