Regression Modeling for Monitoring Organochlorine Pesticide Residues

Serge Olszewski[1], Iryna Lurie[2,3], Volodymyr Lytvynenko[3], Violetta Demchenko[4], Mariia Voronenko[3], Natalia Kornilovska[3] and Oleg Boskin[3]

[1] *Taras Shevchenko National University of Kyiv, Kyiv, Ukraine*
[2] *Ben-Gurion University of Negev, Beer Sheva, Israel*
[3] *Kherson National Technical University, Kherson, Ukraine*
[4] *Kundiiev Institute of Occupational Health of the National Academy of Medical Sciences of Ukraine, Kyiv, Ukraine*

Abstract
The importance of investigating organochlorine pesticide residues (OCPs) in the environment is vital for understanding their local and global impacts on ecosystems and human health. The primary aim of this study was to identify and assess robust and trustworthy methodologies for creating predictive models based on limited statistical samples from monitoring data. For this purpose, we used experimental data illustrating the spatial and temporal fluctuations of various pesticides concentrations across French provinces. For regression tasks, we implemented regression algorithms like eXtreme Gradient Boosting (XGBoost), Categorical Boosting (CatBoost), and Light Gradient Boosting Machine (LightGBM). To evaluate the predictive performance of XGBoost, CatBoost, and LightGBM, we utilized the root means square error (RMSE), coefficient of determination (CD), and mean absolute error (MAE). The results showed that the XGBoost regression showed the best results with a score of 83% to 93% on the examined data. This study proposes regular and rigorous monitoring strategies that include investigations of OCPs and phthalates for the Loskop Dam and similar water systems worldwide.

Keywords 1
Mass Spectra, organochlorine pesticide residues, Fréchet Distance, Decomposition. Machine learning, eXtreme Gradient Boosting, Categorical Boosting, Light Gradient Boosting Machine

1. Introduction

Polychlorinated biphenyls (PCBs) and organochlorine pesticides (OCPs), synthetic organic pollutants, have been recognized as significant contaminants for a substantial duration. These substances, both PCBs and OCPs, exhibit hydrophobic and lipophilic properties, resulting in their persistence in the environment over extended periods. OCPs were primarily used as agricultural pesticides until their usage was curtailed due to severe adverse effects. There have been reports of ecological risks as well, including biomagnification - a phenomenon where these pollutants accumulate and magnify within the food chain in marine ecosystems. OCPs, due to their long-term persistence in the environment and harmful effects on human health and the environment, are

MoMLeT+DS 2023: 5th International Workshop on Modern Machine Learning Technologies and Data Science, June, 3-4, 2023, Lviv-Leiden, Ukraine
EMAIL: olszewski.serge@gmail.com (S. Olszewski); lurieira@gmail.com (I. Lurie); immun56@gmail.com (V. Lytvynenko); chemioh@ukr.net (V. Demchenko); mary_voronenko@i.ua (M.Voronenko); pypsiknata2015@gmail.com (Natalia Kornilovska); bbb@i.ua (Oleg Boskin)
ORCID: 0000-0003-4499-8485 (S. Olszewski); 0000-0001-8100-1846 (I. Lurie); 0000-0002-1536-5542 (V. Lytvynenko); 0000-0001-6239-0882 (V. Demchenko); 0000–0002–5392–5125 (M. Voronenko); 0000-0002-8331-8027 (N. Kornilovska); 0000-0001-7391-0986 (O. Boskin)

considered suitable only for restricted use. Nevertheless, in developing nations, these substances are extensively utilized in agriculture for the control of pests.

Even in countries with a sufficiently high level of development of productive forces focused on the agricultural sector, it is impossible to abandon the use of toxic agricultural products.

In Ukraine alone, the range of pesticides is about 268 names, and their tonnage reaches 36 thousand tons, while the need is 40 thousand. A number of assortment of pesticides by the criteria of toxicity, persistence in the environment, migration, bioconcentration, and actual contamination of objects refers to the 1-2 class of hazard.The leading component of methods to control and limit the harmful effects of persistent organic pollutants (OCPs) on the environment is comprehensive monitoring of the distribution of their residual concentration in space and the evolution of this distribution over time. However, the experimental results of such monitoring accumulated to date are fragmentary, non-systematic, and highly discrete. Moreover, the sample sizes are statistically small, and the measurement results contain a tangible stochastic component.

This nature of the accumulated data does not allow us to obtain reliable estimates of the pollution level in the intervals between the control points of OCPs concentration. Moreover, it is difficult to judge the mechanisms of OCPs migration based on monitoring results since these mechanisms are determined by gradients and the rate of concentration change rather than by its absolute value. And obtaining these characteristics directly from experimental data with small sample size and high statistical error significantly increases the dispersion of the results and reduces their reliability.

Thus, the need to use adequate and reliable predictive models built on statistically small samples of monitoring data is evident and urgent.

To address this problem, this study compares the performance of three machine learning algorithms for regression problems, called CATBoost, LightGBM regression, and XGBoost for the spatial and temporal distribution of organochlorine pesticides. For this reason, one of the main goals of this article was to compare and evaluate different regression models based on model evaluation metrics. Hence, the primary contributions of this study are as follows:

- We have extensively reviewed papers on airborne pesticide migration problems and the use of machine-learning methods to solve pesticide monitoring problems.
- We evaluated three regression-based machine-learning methods for estimating pesticide distribution.

The rest of this paper is organized as follows. In the second section, we present the related work. The third section details the data employed, methodologies including CATBoost, LightGBM regression, and XGBoost, as well as the metrics used to gauge the quality of the derived models. In the fourth section, we provide the outcomes of the methodologies elucidated in the previous section. Finally, the fifth section concludes the paper.

2. Review of Literature

The fastest and least controlled mechanism of the formation of spatial and temporal distributions of OCPS is their transport by air masses. In order to track the formation and development of the distribution of organochlorine pesticide residues (OCPs) through air migration, the authors in reference [1] scrutinized the outcomes of passive air sampling in diverse areas like urban, suburban, coastal, and agricultural from April 2009 to January 2010 in Tamil Nadu, southern India. Compounds like dichlorodiphenyltrichloroethane, dichlorodiphenyldichloroethylene, heptachlor, and murex were primarily detected during the monsoon season. The presence of prohibited pesticides such as aldrin, dieldrin, and heptachlor in the air signals their unlawful usage. Moreover, murex, a pesticide not registered in India, was identified in the air for the first time. This gathered information can provide significant insights for the future handling of atmospheric OCPs, but without the development of nonlinear regression models, the management procedure appears to be highly challenging [1].

The investigation of persistent organic pollutants (OCPs) in tropical and subtropical urban areas with low latitudes is crucial for comprehending their local and global influences on ecosystems and human health. Despite having studies on OCPs levels in water, soil, and sediments, the analysis of distribution trends, seasonality, and sources of OCPs in urban regions of Nepal is still limited.

The conclusions drawn from the rather labor-intensive experimental studies are purely qualitative in nature. For example, the movement distances of OCPs suggest that high precipitation levels in tropical climates are insufficient to remove OCPs and that Nepal may be an important source region for OCPs [2]. At the same time, building a nonlinear regression model based on these monitoring data would allow a quantitative assessment of the feasibility of additional measures of artificial flushing of the region and possible remediation of pollution effects. Concentrations of banned organochlorine pesticides and a number of currently used pesticides in samples from the first four years, roughly overlapping 2005, 2006, 2007, and 2008, show distinct spatial and temporal patterns. Although the wide variety of sampling site types helps characterize the entire global variability of pesticide concentrations, it also greatly increases the number of sites required for reliable regional differentiation [3]. However, in this case, too, the data are provided in raw form, without any attempt to investigate the possibility of constructing reliable approximations on their basis.

In [4], to improve monitoring efficiency, the authors studied the possibility of using butter as a sampling matrix to reflect the regional and global distribution of PCBs and individual organochlorine pesticides/metabolites in the air. This is because persistent organic pollutants (OCPs) are concentrated in milk fat. Dairy fat concentrations are regulated by feed intake, which in turn is controlled mainly by atmospheric deposition. Therefore, butter is sensitive to local, regional and global spatial and temporal atmospheric trends of many OCPs and can thus serve as a helpful sampling medium for monitoring purposes. However, to improve quantitative information derived from air concentrations, it is necessary to understand the mechanisms of the influence of climatic factors on the processes of transfer of OCPs from air to milk [4], which is also problematic without the construction of nonlinear regression models.

The most potent factor of OCPs migration is water resources. In this regard, the paper's authors investigated the level of OCP contamination in the urbanized river network of Shanghai with high river densities. The task of assessing the environmental and health risk of OCPs in river networks is complicated by the pressure of high population density. The main objective of the research was to establish the relationship between OCP residues and determine their environmental and human health impacts. Without building reliable predictive models of the spatial and temporal distribution of OCPs, the solution to this problem is incomplete. However, methods for constructing such models on an array of experimental data were not included in [5].

The aim of the study in reference [6] was to evaluate the degree of pesticide contamination on the coast of Karachi, Pakistan, with a focus on nine different OCPs considered to be highly toxic. Spatial analysis revealed that Creek Avenue and Channa Creek sites were the most severely affected areas in terms of pesticide pollution. As such, it is of utmost importance to strictly supervise and tightly regulate the reckless and illegal usage of OCPs to prevent seawater contamination, thereby ensuring the wellbeing of the marine ecosystem. However, the execution of such controls via automated systems involves the establishment of formalized data models, an aspect that hasn't received much focus yet.

In another study [7], the residual concentrations of 11 organochlorine pesticides (OCPs) were determined at nine sampling points in the surface waters of the Juxi Valley during spring and autumn, aiming to evaluate their pollution levels and potential risks. It was apparent from the study's results that the current sampling guidelines do not necessarily guide the construction of nonlinear regression models of the explored spatial and temporal distributions of OCPs. This was indicated by the rather small size of the data sampling, although, in comparison to air sampling, surface water sampling is considerably less labour-intensive.

On a more promising note regarding the development of nonlinear regression models, a study [8] investigated the status and shifts in Organochlorine Pesticide Pollution (OCPs) in Honghu Lake situated in the Jianggang Plain of central China. To comprehend and evaluate the risks posed by OCPs to the ecosystem of Lake Honghu, 30 surface water samples, 15 surface sediment samples, and a sediment core were gathered in January and July 2005

However, despite the goal of the work to obtain predictive estimates, the dimensionality of the surface water sample array is insufficient to build adequate mathematical models for different time slices of the spatial distributions of OCPs.

The study in reference [9] showcases information on the levels of organochlorine pesticides found in precipitation samples gathered between 1997 and 2003 at seven sites within the Integrated

Atmospheric Deposition Networks in the Great Lakes region. Notably, the 28-day volume-weighted average concentrations of several pesticides, such as hexachlorocyclohexane (HCH), endosulfan, hexachlorobenzene, and chlordane, displayed noteworthy seasonal variations. However, mathematical models for these trends were not proposed by the authors.

Organochlorine pesticides (OCPs) and phthalates are among the most significant anthropogenic environmental pollutants because of their prevalence, persistence, and potential to cause adverse effects in organisms. The studies presented in [10] aimed at monitoring pollution levels of OCPs and phthalates in South Africa, especially in the Oliphant's catchment area, are limited and limited to short-term monitoring. After reviewing the results of this study, the authors of this paper propose regular and rigorous monitoring strategies that include investigations of OCPs and phthalates for the Loskop Dam and similar water systems around the world. However, the proposed approach mainly involves intensifying sampling procedures and increasing monitoring time intervals. While regulation of data structure and sampling dimensionality oriented to constructing adequate predictive models based on these data is not foreseen.

The study outlined in reference [11] investigates the concentrations and distribution of organochlorine pesticides (OCPs) across various tissues of freshwater fish species - silver carp (Hypophthalmichthys molitrix) and bighead carp (Aristichthys Nobilis) - collected from Poyang Lake, the largest freshwater lake in China. However, the authors primarily concentrated on studying the OCPs distributions within the tissues of the biological subjects themselves. Furthermore, the creation of spatial distributions of OCPs, associated with fish migration between different habitats, is represented by a very limited number of samples. Thus, an important mechanism of OCPs transfer together with biota was also not covered by the task of building nonlinear regression models.

As a result of the analysis of the approaches used in various areas of environmental monitoring for the spatial and temporal distributions of the residual concentration of OCPs, it can be clearly concluded that the monitoring tasks are focused on obtaining a series of static slices of the already existing state of the environment and are not adapted to constructing nonlinear regression models of the evolution of these distributions. Problem statement. This paper proposes a comparative analysis of modern machine learning regression methods for their effectiveness in constructing predictive models of the spatiotemporal evolution of OCPs in the environment.

3. Materials and Methods

3.1. Data structure

Construction of a nonlinear regression model was carried out on experimental data describing the spatial and temporal distribution of concentrations of various pesticides in the provinces of France, presented in [12]. Such substances as Chlorpyrifos, Folpet, Lindane, PBO, Pendimethalin, and Tebuconazole were considered. The experiment consisted in the construction of nonlinear regression models for six arrays [8×8×12] of elements. Each array described points of a four-dimensional hypersurface reflecting the spatial and temporal distribution of the concentration of the respective substance. The total number of concentration values was 768. The input for each pesticide type was an array of three independent variables and one dependent variable. The training sample consisted of 537 objects, and the test sample consisted of 231 objects. An example of a 3D directional grid for such a hypersurface is shown in Fig. 1.

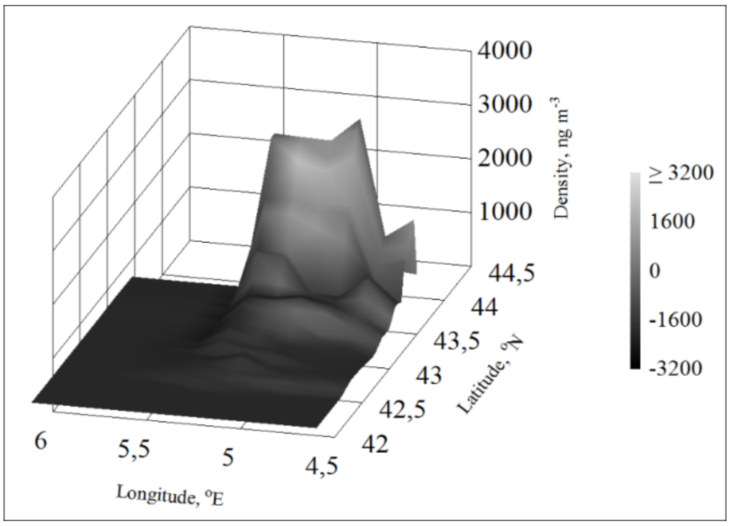

Figure 1: 3D directional cross section for the hypersurface of the spatiotemporal distribution of chlorpyrifos concentration in French provinces

3.2. Ensemble algorithms

XGBoost, CatBoost, and LightGBM are well-known gradient-boosting algorithms commonly employed in machine learning tasks. As prominent ensemble gradient-boosting methods, they can prove efficient in addressing regression challenges. Here is a general description of each of these algorithms for regression tasks.

XGBoost (eXtreme Gradient Boosting):

XGBoost offers powerful capabilities for regression, providing high prediction accuracy. It uses gradient boosting with an ensemble of decision trees. XGBoost has flexible parameters that allow for model optimization and control over tree complexity. It also supports regularization to prevent overfitting. XGBoost has built-in features for handling missing values and categorical features.

CatBoost (Categorical Boosting):

CatBoost is a gradient-boosting algorithm that effectively solves regression problems, considering the characteristics of categorical features. It automatically handles categorical variables without the need for prior encoding. CatBoost provides high prediction accuracy and offers capabilities for fine-tuning the model. It supports regularization and automatic parameter selection, making the model optimization process easier.

LightGBM (Light Gradient Boosting Machine):

LightGBM is a fast and efficient gradient-boosting algorithm that demonstrates excellent performance in regression tasks. It uses optimized tree-building methods, including leaf-wise growth, which leads to faster training speed and lower memory usage. LightGBM has built-in support for categorical features and provides parameters for model optimization. It can handle large datasets and achieve high prediction accuracy in regression tasks.

In general, all three gradient boosting algorithms - XGBoost, CatBoost, and LightGBM - offer powerful capabilities for solving regression problems. Here are some common characteristics of these algorithms:

High prediction accuracy: XGBoost, CatBoost, and LightGBM exhibit high prediction accuracy in regression tasks. They can capture complex dependencies between input features and the target variable, resulting in accurate predictions.

Gradient boosting: All three algorithms are based on the gradient boosting method, which builds an ensemble of weak models (decision trees) and combines them into a strong model. This improves the predictive power and generalization ability of the model.

Regularization: XGBoost, CatBoost, and LightGBM offer regularization methods to mitigate overfitting. Regularization allows for controlling the complexity of the trees and prevents overfitting.

Handling categorical features: CatBoost and LightGBM have built-in support for categorical features, making it easier to work with such data types. They automatically handle categorical variables without the need for manual encoding.

High performance: XGBoost, CatBoost, and LightGBM are designed for efficiency and can handle large datasets. They are optimized for fast model training and provide parallelization options, which accelerate the learning process.

The choice between XGBoost, CatBoost, and LightGBM for solving regression problems may depend on the data characteristics, performance requirements, and the presence of categorical features. It is recommended to conduct comparative studies and parameter tuning for each algorithm on your specific dataset to determine which one demonstrates the best performance and accuracy in your particular case.

3.2.1. Algorithm eXtreme Gradient Boosting

XGBoost (eXtreme Gradient Boosting) is an ensemble algorithm for machine learning based on decision trees using gradient boosting techniques. Boosting based on decision trees is a relatively well-known and very effective machine-learning technique. XGBoost is widely used in data processing to achieve the most accurate results for various machine-learning purposes, especially regarding small to medium-sized structured or tabular data [13].

Boosting is an ensemble method where new models are progressively introduced to rectify errors committed by existing models. The models are incorporated sequentially until no further enhancement is achieved [14].

In many practical applications, Gradient Boosting aims to minimize the objective function. With each iteration, we assign the base learning object to the negative gradient of the loss function, multiply our prediction by a constant, and append it to the value from the preceding iteration. Essentially, fitting a base learner to a negative gradient at each iteration conducts a gradient descent on the loss function [13]. These negative gradients are often called pseudo-residuals as they indirectly aid in minimizing the objective function. XGBoost works by sequentially training a set of weak models called base learners. Each base learner aims to reduce the error of the previous model using gradient descent. In this case, learners are added to the ensemble with weights corresponding to their effectiveness.

$$F = \{f_{1,}f_{2,}f_{3}\cdots f_{m}\} \tag{1}$$

$$\hat{Y}_{l} = \sum_{t=1}^{m} f_{t}(x_{i}) \tag{2}$$

First, consider (1) the initial set of learners as the base set, and then (2) this will serve as the ultimate prediction. Following that, it is necessary to choose a cost-reducing function.

$$o = \{x_1, x_2, x_3, x_4 \ldots x_n\} \tag{3}$$

$$L^{<t>} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i^{<t-1>} + f_t(x_i)) + \Omega(f_t) \tag{4}$$

At each iteration, we get $f_t(x_i)$ by fitting the base trainer to a negative gradient of the loss function with respect to the previous iteration [13]. In this algorithm, we examine several base learners or functions and choose the one that minimizes the loss. This approach has several disadvantages:

1. Learning various basic learning functions
2. Calculation of the value of the loss function of all these essential training functions.

XGBoost uses Taylor's theorem to approximate the value of the loss function for the base learner $f_t(x_i)$ to calculate the exact loss for the various possible base learners.

Taylor's theorem:

$$f(a+h)=f(a)+f'(a)h+\frac{1}{2}f''(a)h^2+\cdots+f^n(a)\frac{h^n}{n!} \quad (5)$$

$$a=\hat{y}_i^{<t-1>} \quad (6)$$

$$h=f_t(x_i) \quad (7)$$

$$f(a)=\ell(y_i,\hat{y}_i^{<t-1>}) \quad (8)$$

$$L^{<t>}=\sum_{i=1}^{n}\ell(y_i,\hat{y}_i^{<t-1>})+\left(\frac{\delta\ell(y_i,\hat{y}_i^{<t-1>})}{\delta\hat{y}_i^{<t-1>}}\right)f_t(x_i)+\left(\frac{\delta^2\ell(y_i,\hat{y}_i^{<t-1>})}{\delta\hat{y}_i^{<t-1>}}\right)f_t(x_i)^2+\ell(y_i,\hat{y}_i^{<t-1>}) \quad (9)$$

$$L^{<t>}=\sum_{i=1}^{n}(C+g_if_t(x_i)+h_if_t(x_i))+\Omega(f_t) \quad (10)$$

$$L^{<t>}=\sum_{i=1}^{n}(g_if_t(x_i)+h_if_t(x_i))+\Omega(f_t) \quad (11)$$

XGBoost uses Taylor's second-order derivative theorem, assuming that the approximation at this stage will be sufficient.

C is constant regardless of any chosen $f_t(x_i)$. g_i - is the first-order derivative of the loss of the previous iteration with respect to the predictions of the previous iteration. h_i - is the second-order derivative of the previous iteration's loss with respect to the previous iteration's predictions [13].

So the algorithm can calculate g_i and h_i before it starts learning the different base learners since it will just be a matter of multiplication.

$$\Omega(f_t)=\gamma K+\frac{1}{2}\lambda\sum_{j=1}^{K}\omega_j^2 \quad (12)$$

$$L^{<t>}=\sum_{j=1}^{K}\left[\left(\sum_{i\in I_j}h_i\right)\omega_i+\frac{1}{2}\left(\sum_{i\in I_j}h_i+\lambda\right)\omega_j^2\right]+\gamma K \quad (13)$$

Let f_t has K nodes in the decision tree. Then I_j is the set of instances of node j. ω_i – is the prediction for node j. For each node j

$$\frac{dL^{<t>}}{d\omega_j^*}=0 \quad (14)$$

$$0=\sum_{i\in I_j}h_i+\omega_j^*\left(\sum_{i\in I_j}h_i+\lambda\right) \quad (15)$$

$$\omega_j^*=\frac{-\sum_{i\in I_j}g_i}{\sum_{i\in I_j}h_i+\lambda} \quad (16)$$

Now let's substitute in $f_t(x_i)$ and consider prediction:

$$L^{<t>}=-\frac{1}{2}\sum_{j=1}^{K}\frac{(-\sum_{i\in I_j}g_i)^2}{\sum_{i\in I_j}h_i+\lambda}+\gamma K \quad (17)$$

Using Taylor's theorem, it becomes feasible to compute the loss function for a node in a tree. However, when dealing with numerous nodes, manually exploring all potential tree structures becomes impractical. Instead, XGBoost constructs an entire tree by selectively determining splits that yield maximum loss reduction. By applying specific partition criteria, the nodes are conditionally divided into the left (v) and right (R) branches (18).

Consequently, instances are allocated to the respective nodes based on the splitting outcome [13]. At this stage, the loss reduction can be calculated, and the partition that offers the most significant loss reduction can be chosen.

$$L = \frac{1}{2}\left[\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_l} h_i + \lambda} + \frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_I} h_i + \lambda}\right] - \gamma \quad (18)$$

The pseudocode depicted in Figure 2 illustrates the XGBoost algorithm for regression problems. In a practical implementation, additional optimizations and handling of specific cases can be added.

```
Ensemble initialization:
1. Set input data: X (feature matrix) and y (target variable vector)
2. Set the number of base learners ($n_{estimators}$)
3. Initialize ensemble predictions with zero values: $y_{pred} = 0$
For each base learner i from 1 to $n_{estimators}$:
4. Calculate residuals: $residuals = y - y_{pred}$
5. Learn the base learner on residuals and features $X$
6.Get the forecast of the current base learner:
$y_{pred_i} = basic_{learner.predict}(X)$
7. Update ensemble predictions: $y_{pred} = y_{pred} + learning_{rate} * y_{pred_i}$
Conclusion:
8. Return ensemble predictions: $y_{pred}$
```

Figure 2: Pseudocode of the XGBoost algorithm

In this pseudocode $learning_{rate}$ is a parameter that controls the contribution of each base learner to the ensemble's final prediction. The smaller the value $learning_{rate}$, the less the influence of each base student. Note that this pseudo-code represents a basic iteration of the XGBoost algorithm and does not include regularization, choice of the optimal number of base learners, and other optimizations that might be applied in a real implementation of the XGBoost algorithm.

3.2.2. CatBoost algorithm

The main idea of the CatBoost algorithm for processing non-categorical features in regression problems is to use gradient boosting and apply regularization to obtain more accurate and generalizing models. Although the CatBoost algorithm is designed specifically for working with categorical features, it is also effective when using non-categorical features. Here are a few key features [15]:

Missing Value Handling: CatBoost has built-in missing value handling, which allows data to be modelled with missing values and makes it suitable for dealing with non-categorical features where missing values can be common.

Regularization: CatBoost applies various regularization techniques, such as L2 regularization and random weight dropout, to prevent model overfitting and increase its generalization ability. This is especially important when working with non-categorical features, where there may be more noise or redundancy.

Optimization of deprivation function: CatBoost uses optimized algorithms to find the optimal parameter settings for models based on gradient descent. This allows you to adapt the model to non-categorical manifestations and improve the accuracy of prediction.

Automatic selection of optimal hyperparameters: CatBoost offers an automatic selection of optimal hyperparameter values using the GridSearch algorithm and other fitting methods. This helps to find the best model settings using non-categorical features.

In general, the main idea of the CatBoost algorithm in processing non-categorical features in a regression problem is to apply regularization, Optimization of deprivation function, and automatic selection of optimal hyperparameters to create optimal accurate and generalizing models.

Advantages of Cat Boost:

- Reliability - simplified setup of hyperparameters (number of trees, learning rate, regularity, tree depth, etc.), which allows you to create more generalized models.

- Automatic Feature Processing - CatBoost converts categorical features to numbers using various statistics and combination, allowing you to use CatBoost without any explicit pre-processing.
- Simplicity in the table - The algorithm has a very convenient API for Python and R.
- Performance - with CatBoost, you can get fast and high-quality results that are not inferior to common machine learning algorithms.

Gradient boosting iteratively builds a sequence of approximations F^t taking into account deprivation function $L(y_i, F^t)$ that have two input values, the *i*-th final output value y_i, and the t-th function F^t, that evaluates y_i. The estimates of y_i can be improved on the found other function $F^t = F^{t-1} + a \cdot h^t$, where α is the step size and the function h^t is the base predictor selected from the population of H functions to calculate losses.

$$h^t = \arg\min EL(y, F^{t-1} + h) \tag{19}$$

$$h^t = \arg_{h\in H} \min E\left(\frac{\delta Ly}{\delta F^{t-1}} - h\right)^2 \approx \arg_{h\in H} \min \frac{1}{n}\left(\frac{\delta Ly}{\delta F^{t-1}} - h^2\right) \tag{20}$$

CatBoost uses approximation of functions by means of the Taylor series with some refinements of the gradient boost technique. Let there be a data set D of *n* instances, each of which has *m* feature sets in vector *x* and values in vector *y*.

$$D = x_k, y_k \left(|D| == n, x_k \in \mathbb{R}^m, y_k, \in \mathbb{R}\right) \tag{21}$$

One of the most common feature processing methods in CatBoost is one-step coding, but it is effective for a small number of features. To solve this problem, features are grouped into categories according to target statistics. Mathematically, the target score of the *i*-th categorical variable of the *k*-th element *D* can be defined as follows:

$$\hat{x}_k^i = \frac{\sum_{x_j \in D_k} 1_{\{x_j^i = x_k^i\}} \cdot y_j + ap}{\sum_{x_j \in D_k} 1_{\{x_j^i = x_k^i\}} \cdot + a}; if\ D_k = \{x_j : \sigma(j) < \sigma(i)\}; (a > 0) \tag{22}$$

The function of of indicator $1_{\{x_j^i = x_k^i\}}$. is equal to 1 when the *i*-th component of the input vector x_j is equal to the *i*-th element of the input vector x_k. *k* is used as the *k*-th element according to the order we put on *D* with the random permutation σ, and i takes integer values from 1 to *k-1*. Options *a* and *p* required to prevent overflow in the equation [15].

Value condition $D_k = \{x_j : \sigma(j) < \sigma(i)\}$ controls value exception y_k to determine values for x_i when encoding value x_k^i. This method also uses the past data of a particular example to calculate its target statistics [16]. When using target statistics, the gradients of the loss functions L with respect to function F^{t-1} re also random variables, since we use a random permutation $\sigma(k)$ to select elements of D_k to encode of categorical variables that affect the value of F^{t-1} In the case of obtaining $\frac{\delta Ly}{\delta F^{t-1}}$ the distribution of the gradient may be biased with the condition of encoding value x_k^i. This conditional bias results in changes in the score value for h^t, and this is a negative impact on the results obtained when estimating F^{t-1} in data that was not used in use [16].

The ability of F^{t-1} to generalize, known as prediction bias, can have an impact. To address this, CatBoost introduces n auxiliary models and utilizes a random permutation of training instances. However, implementing this approach can be challenging due to data limitations and memory costs. In order to avoid such errors, CatBoost employs a method where a single decision tree structure is used for all models. The algorithm utilizes the same D_k, which defines an ordered target statistic, and evaluates if h^t is the optimal decision tree to minimize the expected loss using the complete data set D. Residual values are calculated using permutations $\delta_1 \ldots \delta_n$ which are then utilized to obtain F^{t-1} and h^t. This approach helps reduce the variance in gradient estimates and prevent prediction bias.

```
Ensemble initialization:
1. Set input data: X (feature matrix) and y (target variable vector)
2. Set the number of base learners ($n_{estimators}$)
3. Initialize ensemble predictions with zero values: $y_{pred} = 0$
For each base learner i from 1 to $n_{estimators}$:
    4. Calculate residuals: $residuals = y - y_{pred}$
    5. Learn the base learner $i$ on residuals and feature $X$ CatBoostRegressor
    6. Get the forecast of the current base learner:
    $y_{pred_i} = basic_{learner.predict}(X)$
    7. Update ensemble predictions: $y_{pred} = y_{pred} + learning_{rate} * y_{pred_i}$
Conclusion:
8. Return ensemble predictions: $y_{pred}$
```
Figure 3: Pseudocode of the CatBoost algorithm for solving regression problems

In this pseudocode $learning_{rate}$ as in the XGBoos algorithm, is a parameter that determines the contribution of each base learner to the final prediction of the ensemble.

The CatBoostRegressor algorithm from the CatBoost library provides the ability to solve regression problems. It automatically handles categorical features, works with various data types, and offers a number of optimizations for efficient model training. For some problems, using CatBoost in regression may require additional parameter tweaks and optimizations to achieve better model performance and accuracy.

3.2.3. LightGBM algorithm

Light Gradient Boosted Machine (LightGBM) is an open source implementation of gradient boosting. LightGBM combines 2 main ideas of GOSS and EFB. GOSS means Gradient-based One-Side Sampling. To preserve the accuracy of the information gain of the estimate, it is more appropriate to leave instances with large gradients and discard instances with small gradients. This approach contributes to a more accurate estimate than a uniform sample [17]. EFB means Exclusive Feature Bundling. Since in practice the mass of all features is often quite sparse, the idea of EFB tends to reduce the number of effective features. In such a feature space, most features almost never take on non-zero values at the same time.

The main idea of the LightGBM algorithm is to develop efficient and fast gradient boosting for solving regression problems. It is an optimized version of gradient boosting that has a number of key features:

1. Building trees vertically: Unlike traditional gradient boosting, which builds trees horizontally (in series), LightGBM builds trees vertically (parallel). This allows you to speed up the learning process and achieve high performance.

2. Leaf-wise tree growth: LightGBM uses a leaf-wise tree growth algorithm in which each split node is selected with the largest gradient gain. This allows you to model dependencies that are more complex and improves the quality of predictions.

3. Histogram-Based Optimization: LightGBM uses histograms to efficiently compute gradients and compress histogram. This reduces the amount of memory required to store data and speeds up calculations.

4. Accounting for categorical features: LightGBM automatically processes categorical features without converting them first. It applies unique algorithms for encoding categorical values and allows you to use them directly in the model.

5. Parallel Processing Support: LightGBM supports parallel data processing and model training. This allows you to use all available processor cores and speeds up the learning process.

The LightGBM algorithm is based on the idea of using efficient optimization and parallel processing to achieve high performance and accuracy when solving regression problems. It is widely used in various fields where fast and accurate prediction of numerical values is required.

Benefits of Light GBM:

1) Can work with large amounts of data with significantly reduced training time.

2) The possibility of parallel learning.

3) Uses much less memory.

4) High learning rate and efficiency due to histogram algorithms.

5) High accuracy of boosting results, since Light GBM builds quite thorny trees, following the split-by-leaf approach rather than by levels.

All leaves in the decision tree are split at the same time. This is necessary to optimize flows and control the complexity of the model. Leaves have different information gain, which shows the expected decrease in entropy, which can be defined as follows [18]:

$$IG(B,V) = En(B) - \sum_{v \in V} \frac{|B_v|}{B} En(B_v) \tag{23}$$

$$En(B) = \sum_{d=1}^{D} -p_d \log_2 p_d \tag{24}$$

Where $En(B)$ is the information entropy of collection B, p_d is the relation of B to category *d*, *D* is the number of categories, *v* is the value of the attribute *V*, and B_v is the subset of B for which the attribute has the value *v*. The sheet-growth method is more efficient, as it only splits the sheet that has the most information gain on the same layer. The GOSS method ranks the training instances based on the absolute values of their gradients in descending order. Next, it keeps the first a×100% of instances with larger gradients and we get a subset of instances of A, and then, for the rest of the set A^c, consisting of (1-a)×100% of instances with smaller gradients, we randomly choose a subset of B with size $b \times |A^c|$ Finally, we split the instances according to the estimate of the gain in variance $\tilde{V}_j(d)$ ver the subsets $A \cup B$:

$$\tilde{V}_j(d) = \frac{1}{n}\left(\frac{\left(\sum_{x_i \in A_l} g_i + \frac{1-a}{b}\sum_{x_i \in B_l} g_i\right)^2}{n_l^j(d)} + \frac{\left(\sum_{x_i \in A_r} g_i + \frac{1-a}{b}\sum_{x_i \in B_r} g_i\right)^2}{n_l^j(d)} \right) \tag{25}$$

where $A_l = \{x_i \in A: x_{ij} \le d\}$, $A_r = \{x_i \in A: x_{ij} > d\}$, $B_l = \{x_i \in B: x_{ij} \le d\}$, $B_l = \{x_i \in B: x_{ij} > d\}$. The coefficient $\frac{1-a}{b}$ normalizes the sum of the gradients over B to the size of A^c . The coefficient $\frac{1-a}{b}$ is used to normalize the sum of gradients over B to the size of A^c . Thus, an estimate of $\tilde{V}_j(d)$ over a smaller subset of instances is used instead of an exact estimate of $V_j(d)$ over all instances to determine the split point. Therefore, the computational cost can be significantly reduced. The GOSS approximation error looks like this:

$$\varepsilon(d) \le C_{a,b}^2 \ln\frac{1}{\delta} \cdot \max\left\{\frac{1}{n_l^j(d)}, \frac{1}{n_r^j(d)}\right\} + 2DC_{a,b}\sqrt{\frac{\ln\frac{1}{\delta}}{n}}, \tag{26}$$

where

$$\varepsilon(d) = \left|\tilde{V}_j(d) - V_j(d)\right|,\ g_l^{-j}(d) = \frac{\sum_{x_{i \in (A \cup A^C)_l}} |g_i|}{n_l^j(d)},\ g_r^{-j}(d) = \frac{\sum_{x_{i \in (A \cup A^C)_r}} |g_i|}{n_r^j(d)},\ C_{a,b}^2 = \frac{1-a}{\sqrt{b}} \max_{x_i \in A^C} |g_i|,\ D = \max(d_l^{-J}(d), g_r^{-j}(d)).$$

Here's a pseudocode representation of the LightGBM algorithm for regression (Fig.4):

```
Inputs:
- $training_{data}$: Dataset containing the input features and target values for training
- $num_{iterations}$: Number of boosting iterations (trees) to build
- $learning_{rate}$ learning_rate: Step size for updating the model at each iteration
- $max_{depth}$: Maximum depth of each tree
- $num_{leaves}$ num_leaves: Maximum number of leaves in each tree
Procedure:
$LightGBM_{Regression}(training_{data}, num_{iteratios}, learning_{rate}, max_{depth}, num_{leaves})$
   Step 1: Initialize the model
   Initialize the ensemble model as an empty list of trees
   Step 2: Recomputed gradients and Hessians
   Compute the initial gradients and Hessians for the target values
   Step 3: Boosting iterations
   for iteration = 1 to $num\_iterations$ do:
      Step 4: Build a new tree
      Construct a new decision tree based on the gradients and Hessians
      Step 5: Update the model
      Update the model by adding the new tree to the ensemble
            Step 6: Update the gradients and Hessians
      Update the gradients and Hessians by computing the residuals
            Step 7: Early stopping (optional)
      If a stopping criterion is met, such as no improvement in validation error, stop the iterations
            Step 8: Predictions
   Predict the target values by aggregating the predictions of all trees in the ensemble
            Step 9: Return the predictions
   Return the predicted target values
End Procedure
```

Figure 4: Pseudocode of the LightBoost algorithm for solving regression problems

This pseudocode provides a high-level overview of the LightGBM algo-rithm for regression. The actual implementation may involve additional optimizations and techniques for efficiency and performance.

3.3. Measuring error

The root mean square error (RMSE), coefficient of determination (R^2) and mean absolute error (MAE) are used to compare the prediction performance XGBoost (eXtreme Gradient Boosting), CatBoost (Categorical Boosting), LightGBM (Light Gradient Boosting Machine) These error measures are expressed as follows, where y_j and $\hat{y}_j$ are the actual response and the predicted response of observation j, and $\bar{y}$ is the average of all actual responses [23].

RMSE measures the root-mean-square difference between estimated values and actual values and is a risk function corresponding to the expected value of the squared error loss.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{n}} \tag{27}$$

The accuracy of the models obtained was based on the formula of the determination coefficient (R^2 - statistics):

$$R^2 = \frac{\sum_{i=1}^{n} (\hat{y} - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{28}$$

where $\bar{y}$ average value, $\hat{y}_i$ – output of the model.

The determination coefficient characterizes the fraction of the variance of the resultant variable Y, an explanation of the regression, in the overall variance of the resultant variable *Y*. Accordingly, the magnitude 1-R^2 characterizes the fraction of the variance of the variable Y caused by the influence of other factors not taken into account in the model. MAE is a measure of error between paired observations expressing the same phenomenon and is calculated using a formula.

$$MAE = \frac{1}{n} \sum_{j=1}^{n} |y_i - \hat{y}_i| \tag{29}$$

4. Results and Discussion

Table 1
Metrics for LightGBM

Pesticides	RMSE	R^2	MAE
Chlorpyrifos	219.63	0.6067544	56.96
Pendimethalin	0.61	0.7247481	0.23
PBO	0.02	0.81	0.02
Lindane	0.09	0.82	0.05
Folpet	17.56	0.91	6.76
Tebuconazole	0.02	0.84	0.01

Table 2
Metrics for CATGBM

Pesticides	RMSE	R^2	MAE
Chlorpyrifos	274.13	0.44	80.10
Pendimethalin	0.76	0.62	0.29
PBO	0.01	0.77	0.01
Lindane	0.13	0.66	0.07
Folpet	36.52	0.62	14.94
Tebuconazole	0.03	0.49	0.02

Table 3
Metrics for XGGBM

Pesticides	RMSE	R^2	MAE
Chlorpyrifos	126.9035	0.83	31.81
Pendimethalin	0.5295898	0.78	0.19
PBO	0.005016078	0.88	0.02
Lindane	0.06687168	0.91	0.04
Folpet	14.69492	0.93	5.21
Tebuconazole	0.01596549	0.85	0.02

5. Conclusions and Future Work

Comparing the obtained models based on the RMSE from Tables 1-3 while considering that the lower the RMSE, the higher the accuracy. From these tables, we can conclude that XGBoost has the lowest RMSE value for all pesticide species; hence, it works well. On the other hand, regression of the reference vector had the highest value for all pesticide species, so the model did not work well for the data sets studied.

Based on R^2 - we know that the higher the R^2 higher the accuracy. Tables 1-3 show the values of each model. From these tables, we can conclude that XGBoost has a slightly higher R^2 value than CatBoost algorithm, so they both perform well. The reference vector regression had the lowest value, so the model did not work well for the examined datasets. Based on MAE - we know that the lower the MAE, the higher the accuracy. Tables 1-3 show the MAE values for each model for the six pesticide species. From these tables, we can conclude that the XGBoost regression has the lowest MAE value; hence it works well for this task. On the other hand, the reference CatBoost had the highest value, so the model did not work for the data set.

As the results of the experiments have shown, XGBoost is an efficient algorithm with good performance. It offers a wide range of features, supports regularization, and has flexible parameters for model optimization. The LightGBM algorithm demonstrates high efficiency, fast training time, and low memory usage. CatBoost performs well and provides built-in support for categorical feather waver; in our research, based on the accuracy results presented in tables 1-3, we would prefer the XGBoost algorithm for solving our regression task.

The comparison of accuracy between LightGBM and XGBoost in solving regression problems may depend on the specific dataset and model parameters. Both algorithms generally exhibit high accuracy in regression tasks, but the results can vary depending on the data characteristics.

The main objective of this paper was to find and evaluate adequate and reliable methods for constructing predictive models built on statistically small samples of monitoring data. Therefore our task was to compare and evaluate different regression models based on model evaluation indicators. Experimental data describing the spatial and temporal distributions of concentrations of different pesticides across the French provinces were used as a dataset. The pesticides used were Chlorpyrifos, Folpet, Lindane, PBO, Pendimethalin, and Tebuconazole.

In the study, we were able to explore different regression algorithms such as k LightGBM regression, CatBoost XGBoost regression and applied these algorithms to the dataset. The study was conducted in the R environment.

From this study on our dataset, the XGBoost regression showed the best results with an R^2 score of 83% to 93% on the examined data. In contrast, the reference vector regression showed the lowest results.

6. References

[1] S. Srimurali, S. Govindaraj, S. Krishna Kumar, R. Babu Rajendran, Distribution of organochlorine pesticides in atmospheric air of Tamilnadu, southern India.//Int. J. Environ. Sci. Technol, 2015, 12, pp.1957-1964. doi 10.1007/sl3762-014-055p-3.

[2] B. Pokhrel, P. Gong, Xi. Wang, S. Nath Khanal, J. Ren, Ch. Wang, Sh. Gao, T. Yao, Atmospheric organochlorine pesticides and polychlorinated biphenyls in urban areas of Nepal: spatial variation, sources, temporal trends, and long-range transport potential//Atmos. Chem. Phys., 2018, 18, pp. 1325-1336. doi.org/10.5194/acp-18-1325-2018.

[3] C. Shunthirasingham, C. E. Oyiliagu, X. Cao, T.Gouin, F.Wania, S.-C. Lee, D. C. Muir, Spatial and temporal pattern of pesticides in the global atmosphere.//Journal of Environmental Monitoring, 2010, 12(9), 1650. doi:10.1039/c0em00134a

[4] O. I.Kalantzi, R. E.Alcock, P. A. Johnston, D.Santillo, R. L.Stringer, G. O.Thomas, K. C. Jones, The Global Distribution of PCBs and Organochlorine Pesticides in Butter.// Environmental Science & Technology, 2001, 35(6), pp. 1013–1018. doi:10.1021/es0002464

[5] C. Chen, W. Zou, S. Chen, K.Zhang, L. Ma, Ecological and health risk assessment of organochlorine pesticides in an urbanized river network of Shanghai, China.// Environmental Sciences Europe, 32(1), 2020. doi:10.1186/s12302-020-00322-9

[6] R. Majeed, S. U. Fatima, M. A. Khan, M. A. Khan, S. Sh. Shaukat, Occurrence and distribution of organochlorine pesticides in Karachi coastal water//International Journal Of Biology And Biotechnology, 2020, 17(3), pp. 503-512.

[7] Z.Liu, G. Zheng, Z. Liu, Organochlorine Pesticides in Surface Water of Jiuxi Valley, China: Distribution, Source Analysis, and Risk Evaluation.// Journal of Chemistry, 2020, pp.1–8. doi:10.1155/2020/5101936

[8] L. Yuana, Sh. Qia, X. Wud, Ch. Wua, X. Xinga, X. Gongf, Spatial and temporal variations of organochlorine pesticides (OCPs) in water and sediments from Honghu Lake, China//Journal of Geochemical Exploration, 2013, 132, pp.181–187 [http://dx.doi.Org/10.1016/j.gexplo.2013.07.002]

[9] P. Sun, S. Backus, P. Blanchard, R. Hites, Temporal and Spatial Trends of Organochlorine Pesticides in Great Lakes Precipitation//Environ. Sci. Technol., 2006, v. 40, no. 7, pp. 2135-2141.

[10] T. Ansara-Ross, V. Wepener, P. van den Brink, M. Ross, Pesticides in South African fresh waters. // African Journal of Aquatic Science, 2012, 37(1), pp.1–16. doi:10.2989/16085914.2012.666336

[11] Z.Zhao, Y. Wang, L. Zhang, Y. Cai, Y. Chen, Bioaccumulation and tissue distribution of organochlorine pesticides (OCPs) in freshwater fishes: a case study performed in Poyang Lake, China's largest lake.// Environmental Science and Pollution Research, 2014, 21(14), pp.8740–8749. doi:10.1007/s11356-014-2805-z

[12] M. Desert, S. Ravier, G. Gille, A. Quinapallo, A. Armengaucl, et al., Spatial and temporal distribution of current-use pesticides in ambient air of Provence-Alpes-Cote-d'Azur Region and Corsica, France. //Atmospheric Environment, Elsevier, 2018, 192, pp.241-256. 10.1016/j.atmoscnv.2018.08.054.hal-01865350

[13] T. Chen, C. Guestrin, XGBoost: A Scalable Tree Boosting System. arXiv:1603.02754v3 [cs.LG] 10 Jun 2016.

[14] A. I. Ahmed Osman, A. N. Ahmed, M. F. Chow, Y. F. Huang, A. El-Shafieef, Extreme gradient boosting (Xgboost) model to predict the groundwater levels in Selangor Malaysia. Ain Shams Engineering Journal, Volume 12, Issue 2, June 2021, pp. 1545-1556.

[15] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, CatBoost: unbiased boosting with categorical features. 1706.09516v5 [cs.LG] 20 Jan 2019.

[16] T. Hu, T. Song, Research on XGboost academic forecasting and analysis modelling. J. Phys.: Conf. Ser. 1324 012091.

[17] J. T. Hancock, T. M. Khoshgoftaar, CatBoost for big data: an interdisciplinary review. Journal of Big Data volume 7, Article number: 94, 2020.

[18] G. Ke, Q. Meng, Th. Finley, T.Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree.

[19] O. Eyecioglu, B. Hangun, K. Kayisli, Performance Comparison of Different Machine Learning Algorithms on the Prediction of Wind Turbine Power Generation Conference, Proceedings of the 8th International Conference on Renewable Energy Research and Applications (ICRERA), 2019, pp. 922-926, doi:10.1109/icrera47325.2019.8996541.