



Geographical-XGBoost: a new ensemble model for spatially local regression based on gradient-boosted trees

George Grekousis^{1,2,3}

Received: 2 October 2024 / Accepted: 6 March 2025 / Published online: 7 April 2025

© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2025

Abstract

XGBoost is a widely used machine learning method known for exhibiting high accuracies in regression and classification tasks. Current efforts to create spatial versions of XGBoost do not alter the original XGBoost algorithm; nor do they implement any geographical modification. Here, we fill these gaps by introducing Geographical-XGBoost (G-XGBoost), which extends XGBoost in multiple ways. First, G-XGBoost creates local models by assigning spatial weights linked directly to the model's accuracy by focusing on the samples that cause the most error. Second, it applies ensemble architecture, by combining the global and local models, for training, validation, and prediction, which improves model accuracy. Third, G-XGBoost calculates local feature importance using spatial weights, something not proposed so far. We evaluate G-XGBoost on six benchmark datasets against three global regression models (Ordinary Least Squares, Random Forests, and XGBoost) and three spatially local regression models (Geographically Weighted Regression, Geographical Random Forests, and Geographically Weighted Random Forests). G-XGBoost outperforms all existing models improving R^2 by 17.44% and mean absolute error by 17.42%, on bootstrapped mean values. G-XGBoost is also an exploratory tool as it analyzes how feature importance varies in space, contributing thus to the need for interpretable and explainable AI, and therefore, it can further advance spatial machine learning application to geographical studies.

Keywords Spatial machine learning · Spatially local regression · XGBoost · Benchmark datasets

JEL Classification C14 · C18 · C21 · C63

✉ George Grekousis
geograik@gmail.com; graikousis@mail.sysu.edu.cn

¹ School of Geography and Planning, Sun Yat-sen University, Guangzhou, China

² Guangdong Key Laboratory for Urbanization and Geo-Simulation, Sun Yat-sen University, Guangzhou, China

³ Guangdong Provincial Engineering Research Center for Public Security and Disaster, Sun Yat-sen University, Guangzhou, China

1 Introduction

Machine learning (ML) has boomed in the last few years in nearly all scientific disciplines. However, most current ML methods cannot entirely handle the considerations posed by spatial data, which can be summarized as spatial dependence and spatial heterogeneity. Neglecting the spatial dimension negatively impacts learning and infers model inaccuracies, as the presence of spatial autocorrelation may distort the results of non-spatial approaches (i.e., overoptimistic fit of models or biased predictions) and, more importantly, does not allow for analysis of how complex nonlinear relationships vary across space (Grekousis 2020, Effati et al. 2015, Griffith et al. 2013, Griffith 2006, Griffith 2004).

Remarkably, around 80% of all data can have a geographic dimension, and much of these data can be georeferenced (VoPham et al. 2018). It is thus a big surprise that spatial ML (SML) lags behind typical aspatial ML. There is an on-going quest on how SML is different than non-spatial ML or geographic artificial intelligence, and a comprehensive discussion on the topic is provided by Credit (2024). SML is a set of georeferenced-data-driven techniques that attempt to handle spatial dependence and heterogeneity by incorporating spatial awareness through the inclusion of geography within the structure of the algorithms (Grekousis et al. 2022). In other words, an SML algorithm should be capable of space conceptualization by integrating location, distance, proximity, neighborhood, or spatial weights within its framework and preferable within its algorithmic steps. Merely including spatial data in the standard input matrix by neglecting the spatial properties (e.g., spatial dependence, spatial heterogeneity, scale) does not make an algorithm spatial.

Two broad modeling approaches exist in current SML for incorporating spatial properties into the learning process: (a) through the spatial observation matrix and (b) by modifying the learning algorithm itself (Nikparvar and Thill 2021). In the first approach, the properties of spatial data are developed in the spatial observation matrix, while the underlying ML algorithm remains unchanged. In the second approach, spatial properties are considered within the learning algorithm's representation or within its objective function. In other words, the original non-spatial ML algorithm is alternated to explicitly incorporate the spatial relationships.

Examples of SML methods in regression tasks are Geographically Weighted Regression (GWR) (Brunsdon et al. 1996), Geographical Random Forests (GRF) (Georganos et al. 2021), Geographically Weighted Random Forests (GRF-W) (Georganos et al. 2022), and several spatial econometric models, such as the Spatial Lag Model (SLM) and the Spatial Error Model (SEM) (Anselin 2009). GWR and typical spatial econometric models like SLM and SEM are parametric models. As such, they share common characteristics and limitations, including a strong reliance on statistical assumptions. They do, however, offer a solid way of interpreting results, which is one of the main reasons why they are still prevalent among geographers and other social scientists. On the other hand, data-driven ML

methods are nonparametric and can handle high-dimensional data and nonlinear relationships. The strength of ML methods is that they are data-driven and can extract complex structures and patterns from data, without having to make any assumptions as in traditional inferential statistics (Li 2022).

Though parametric methods offer instructive tools to analyze and interpret spatial data, nonparametric spatial methods should be integrated into spatial analysis. Only limited nonparametric ML methods (mainly tree-based techniques) have been extended to spatially local regression. As such, there is a critical need for novel SML methods that improve prediction accuracy and interpretability in the spatial analysis context (Credit 2024). Furthermore, SML methods that modify the original ML algorithm to explicitly model spatial properties have been significantly less explored, despite being considered more promising (Nikparvar and Thill 2021). Our motivation is to expand research in these directions. Here, we present a new SML algorithm named Geographical-XGBoost (G-XGBoost) that modifies the XGBoost (extreme gradient boosting trees) algorithm to become geographically aware through geographically varying models and as such belongs to the second modeling paradigm where spatial properties are integrated in the learning algorithm itself. The proposed model addresses spatial heterogeneity by assessing how feature importance varies locally. This characteristic also makes G-XGBoost an exploratory tool, thereby contributing to the demand for more explainable ML methods. Though the proposed algorithm is not explicitly designed to handle spatial dependence, it can indirectly address some of its effects by identifying spatial patterns in local feature importance.

1.1 Related work and research gaps

XGBoost is a supervised ML model based on classification and regression trees (CART) ensembles that has been widely recognized in numerous ML and data mining challenges and competitions (Chen et al. 2016). Unlike other ML methods such as fully connected neural networks, it provides high prediction accuracy, model stability, computational efficiency, and interpretability of results (Chen et al. 2023). However, only a few works have attempted to develop spatial versions of the XGBoost approach. For example, Kang et al. (2022) proposed an XGBoost composite model to predict tourism trends. A spatiotemporal framework was constructed based on spatial clustering and temporal data, and the factors were fed to the XGBoost algorithm for prediction.

Schimohr et al. (2023) introduced the ‘Geographically Weighted XGBoost’ algorithm for predicting bike-sharing trip counts. Though the proposed XGBoost model is named ‘Geographically Weighted,’ the process is a pipeline where non-spatial ordinary XGBoost models are used to make predictions, which are subsequently fed into a GWR model. Consequently, only the GWR handles spatial autocorrelation and optimal bandwidth selection. In this sense, the term ‘Geographically Weighted XGBoost’ is a pipeline that sequentially combines two models rather than a method that modifies the internal algorithmic structure of XGBoost for spatially local regression. More importantly, and according to the

authors of this approach, the geographically weighted XGBoost decreased the predictive accuracy of the XGBoost model, indicating that the general accuracy of predictions could not be further increased through the estimation of GWR (Schimohr et al. 2023). In a similar effort, Ye et al. (2023) also introduced a model with the same name (Geographically Weighted XGBoost) to analyze soil arsenic concentration in a three-step approach. First, XGBoost was used for feature selection. Second, the spatial weights were calculated using the GWR method. Lastly, data and spatial weights were used only as input variables on the XGBoost model. Like Schimohr et al. (2023), their model combined GWR with XGBoost with no other adjustment on the XGBoost algorithm and, as such, remained aspatial.

As shown above, current efforts do not alter the original XGBoost algorithm; nor do they implement any geographical modification. In addition, the works including GWR bring together all known limitations of the method, such as the linearity assumption and the many drawbacks of linear regression (O'Sullivan and Unwin 2010). Our approach overcomes this limitation by omitting GWR and instead creates a Geographical-XGBoost algorithm that analyzes data locally. Though spatial heterogeneity can be traced by global models (Li et al. 2022; Wang et al. 2020), local models allow for a different perspective on the data, as they estimate all model statistics and metrics at the local level that can be subsequently mapped and further analyzed spatially. Moreover, the bandwidth introduction in local models allows for further investigation of varying scales of analysis or even the development of multiscale models (i.e., bandwidth varies by location). This emphasizes the importance of the key concepts of adjacency and neighborhood in local modeling (O'Sullivan and Unwin 2010). On this note, there is currently no genuinely geographical local XGBoost model incorporating space conceptualization within its calculation, such as spatial weighting and bandwidth estimation, which is a significant literature gap.

This work fills this gap by introducing G-XGBoost, which is innovative in the following ways:

- First, it applies the concept of geographically varying models in XGBoost: that is, it creates local models that analyze data within a specified neighborhood through spatial weights. A significant difference with other SML methods is that the spatial weights are directly linked to the model's accuracy by focusing on the samples that cause the most error and not just for assigning some higher probability for selection reasons.
- Second, to gain information existing in the global model, G-XGBoost creates an ensemble of the global and local models by using both models (local + global) for training, validation, and prediction, further improving model accuracy. This concept has not been applied before in spatially local regression, as previous related methods were global or local.
- Third, G-XGBoost calculates local feature importance using spatial weights through the gain function, which has not been proposed before. As such, G-XGBoost is also a robust exploratory tool for tracing spatial heterogeneity apart from being a predictive tool, as it evaluates how the spatially weighted

feature importance varies locally. This adds to the model's interpretability, which is extremely helpful from the policy perspective.

G-XGBoost shares a similar concept to GRF and GWR, that of adopting a local regression analysis framework where the final model consists of several local sub-models, that will hopefully address spatial heterogeneity.

We compare G-XGBoost to six models—three non-spatial models (Ordinary Least Squares—OLS, Random Forests—RF, and XGBoost) and three spatially local regression models (GWR, GRF, and GRF-W)—over six benchmark datasets. Instead of comparing G-XGBoost model with global spatial models or pipeline approaches that integrate trivial XGBoost with GWR, we compared it directly to local baseline models such as GWR and GRF that have already gained wide acceptance through multiple citations and, as such, are well-established in the literature. In addition, as GWR and GRF are local models, they are directly compared with the proposed G-XGBoost. The results show that G-XGBoost outperforms all models in all experimental settings, making it a promising tool for geographical analysis.

2 G-XGBoost

2.1 Mathematical formulation

2.1.1 Global model

Suppose we have a given dataset $D = \{(x_i, y_i)\} (|D| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$ with n observations and m features with y_i the response variable and x_i the predictor value for the i observation, where $i = 1, \dots, n$. In the geographical context, the number of observations n corresponds to the number of spatial units in which data are aggregated. To fit the needs of modeling spatial data, we call the XGBoost model 'global XGBoost' to distinguish it from the proposed local model. Global XGBoost can be written in the following form (Xgboost 2024):

$$\hat{y}_{i-gl} = \hat{y}_i = \sum_{k=1}^K f_k(x_i), f_k \in F \quad (1)$$

where \hat{y}_{i-gl} is the predicted value for observation i , K is the number of trees, f_k is a function in the functional space F , and F is the set of all possible CARTs. The model is trained to find the best parameters θ that optimize the objective function

$$\text{obj}(\theta) = \sum_i^n l(y_i, \hat{y}_i) + \sum_{k=1}^K \omega(f_k) \quad (2)$$

where the first part of the equation is the training loss function l , given by Eq. (3), and the second part is the regularization term, where $\omega(f)$ is the complexity of the tree, given by Eq. (4). The loss function measures the capability of the model to

make predictions using the training data. The typical loss function for XGBoost is the mean squared error, given as

$$l = \frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2 \quad (3)$$

where y_i is the target and \hat{y}_i the prediction.

The regularization term in the objective function, given by Eq. (2), corresponds to the complexity of the tree $\omega(f)$, which quite commonly is defined as follows (though it can also take other forms)

$$\omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (4)$$

where w is the leaves' scores vector, T is the number of leaves, with γ (gamma) and λ (lambda) the hyperparameters to control complexity and the degree of regularization. Gamma is the minimum loss reduction required to make a split and controls the trees' depth by keeping them shallow (depending on the value set). The default value is zero, which means that there is no control over the gain function (see Eq. (18)). The regularization term assists in smoothing the final learned weights to avoid overfitting (Chen et al. 2016).

However, the tree ensemble model in Eq. (2) cannot be optimized using traditional approaches in the Euclidean space because the model is trained in an additive way. The objective function can thus be rewritten for the t -th iteration as

$$\text{obj}^t = \sum_i^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \omega(f_t) + \text{constant} \quad (5)$$

where $\hat{y}_i^{(t-1)}$ is the prediction at the previous step. In essence, the f_t that most improves the model is added to Eq. (2) in an additive way.

We can optimize the above objective function by taking the Taylor expansion of the loss function up to the second-order approximation

$$\text{obj}^t = \sum_i^n [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \omega(f_t) + \text{constant} \quad (6)$$

where g_i (aka Jacobian matrix) and h_i (aka Hessian matrix) are the first and second partial derivatives of the loss function

$$g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)}) \quad (7)$$

$$h_i = \partial_{\hat{y}_i^{(t-1)}}^2 l(y_i, \hat{y}_i^{(t-1)}) \quad (8)$$

To obtain a simplified objective function at step t , we can remove the constants in Eq. (6) to get

$$obj^j = \sum_i^n [g_i f_i(x_i) + \frac{1}{2} h_i f_i^2(x_i)] + \omega(f_i) \quad (9)$$

So, every new tree in the sequence is optimized through the above objective function, which has a significant advantage. Its value depends only on the first (g_i) and second (h_i) order gradients, which are easy to compute.

2.2 Local model

G-XGBoost extends Eq. (1) to train a spatially local model based only on a subset of the original data defined by the bandwidth value (b) and the use of spatial kernels to calculate the spatial weights (Fig. 1) (G-XGBoost function G_{XGB}). To differentiate spatial weights from the weights (w) used in decision tree leaves in Eq. (4), we use the symbol w_s . We introduce two types of spatial kernels, namely fixed and adaptive. The fixed kernel creates neighborhoods based on a distance threshold bandwidth value. The adaptive kernel defines the bandwidth as the number N of the nearest neighbors that will be used to create the neighborhood. In the case of a fixed kernel, each unit has a varying number of neighbors, whereas in the adaptive kernel,

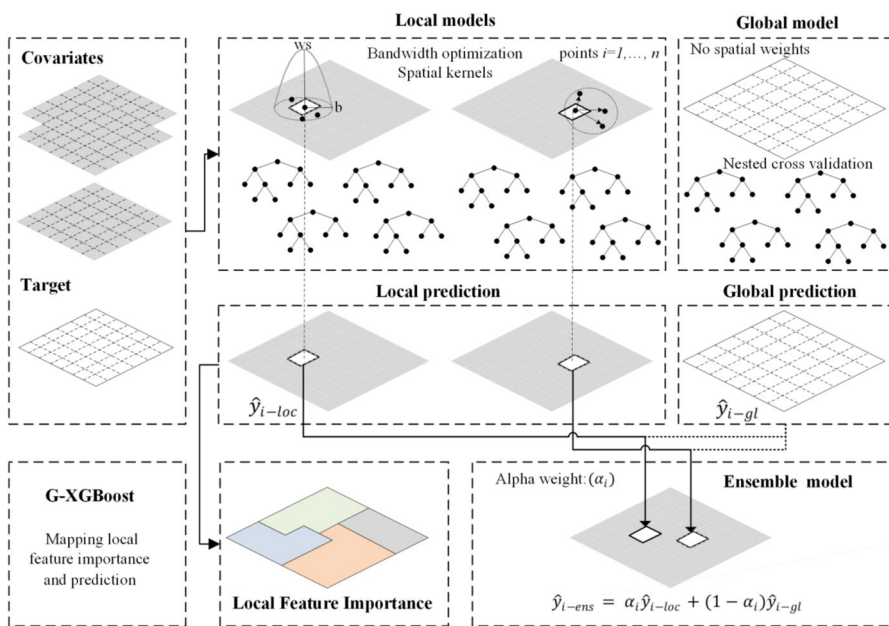


Fig. 1 G-XGBoost ensemble for spatially local regression. A different sub-model is built for every spatial unit (i), including only its neighboring units, through either a fixed or an adaptive spatial kernel. The optimal bandwidth value (either distance or number of nearest neighbors) is defined by minimizing the cross-validation criterion. Hyperparameters are selected using grid search through nested cross-validation of the global model. G-XGBoost results from the ensemble (\hat{y}_{i-ens}) of global (\hat{y}_{i-gl}) and local (\hat{y}_{i-loc}) models using the alpha weight (α_i) regularization hyperparameter. Feature importance is produced for the local models

every unit has the same number of neighbors. When the aerial size of the spatial units varies a lot (as in most real-world geographical problems), the adaptive kernel is preferred.

Many kernel functions can be used for the spatial weights. G-XGBoost applies the bi-square function given in Eq. (10)

$$ws_{ij} = \begin{cases} \left[1 - \left(\frac{d_{ij}}{b} \right)^2 \right] & \text{if } e_{i-\text{loc}} > e_{i-\text{gl}} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where d_{ij} is the Euclidean distance between the i -th and j -th points (centroids in case of polygons). In the case of a fixed kernel, b is the bandwidth value corresponding to the defined distance threshold. In the case of an adaptive kernel, b is the distance corresponding to the N -th nearest neighbor.

The bandwidth (b) defines the extent of the neighborhood of i that will be used to train and test the local models. As the bandwidth is a hyperparameter of the model, it should be tuned. In G-XGBoost (G-XGBoost function `optimize_bw`), and similar to other spatially local regression models like the GWR (Farber and Páez 2009), the optimal value of b is determined by minimizing the cross-validation (CV) criterion, given as

$$b = \operatorname{argmin} CV = \sum_{i=1}^n (y_i - \hat{y}_{\neq i}(b))^2 \quad (11)$$

where $\hat{y}_{\neq i}(b)$ is the fitted value of y_i by omitting the i -th observation. The bandwidth value can also be user-defined.

Based on the above, G-XGBoost extends Eq. (1) to train n local models (one for every single spatial unit) in the following form

$$\hat{y}_{i-\text{loc}} = \sum_{k=1}^K (u_i, v_i) f_k(ws_i, x_{ib}), f_k \in F \quad (12)$$

where $\hat{y}_{i-\text{loc}}$ is the prediction of the G-XGBoost local model calibrated on location i , with (u_i, v_i) being the coordinates of the centroid of the i th spatial unit. We call the spatial unit at location i the central point. ws_i is the spatial weights matrix of central point i to the spatial units x_{ib} within the spatial kernel, defined by a bi-square kernel weighting function, given by Eq. (10), with bandwidth value b .

The objective function to be optimized for G-XGBoost for the i local model referring to the i central point is given as

$$\text{obj}_i^t = \sum_i^n [(g_i)ws_i f_t(x_i) + \frac{1}{2}(h_i)ws_i f_t^2(x_i)] + \omega(f_t) \quad (13)$$

G-XGBoost creates local models and weighs the samples using spatial weights. To do so, it uses the sample weights functionality provided by XGBoost and replaces them with spatial weights. This is done by multiplying the spatial

weights by the gradients. The idea of this multiplication lies in the fact that h_i , the Hessian for data point i , is already used as a proxy for the importance of each sample in the learning process to focus on the samples that cause the most error. Therefore, according to Chen et al. (2016), sample weights, and in our case spatial weights, are multiplied to the first and second gradients to add extra weight to samples. G-XGBoost supports the integration of spatial weights, but it can also be applied with no spatial weights.

In addition, to avoid data leakage and overestimated accuracies, we exclude the central point during the training–testing phase for every local model. In this manner, the local XGBoost model is trained and tested only on the $y_{\neq i}(b)$ (data points within bandwidth b , omitting the i -th observation). For example, if the optimized adaptive bandwidth value is 40, the subset to be used in Eq. (13) would be only the 40 nearest neighbors around the central point, excluding the central point itself. This is achieved through the spatial weights matrix, where the central point gets zero weight with itself. Then, the trained model is used to predict $\hat{y}_{i-\text{loc}}$. We refer to the model residual for location i as ‘out-of-bag (OOB) error’ in Eq. (14).

$$OOB_i = y_i - \hat{y}_{i-\text{loc}}. \quad (14)$$

Though the term ‘OOB error’ is used in bagging approaches and not in boosting, we use it here to emphasize that the values of the central point are not used for training or testing (and thus are taken out of the bag), but only for evaluation of the prediction in unseen data. These OOB errors are calculated for all central points in turn, and various evaluation metrics such as the R^2_{oob} , given by Eq. (15), the OOB mean absolute error (MAE_{oob} ; given by Eq. (16), and the OOB root mean square error ($RMSE_{oob}$; given by Eq. (17)) are calculated.

$$R^2_{oob} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_{i-\text{loc}})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (15)$$

$$MAE_{oob} = \frac{\sum_{i=1}^n |y_i - \hat{y}_{i-\text{loc}}|}{n} \quad (16)$$

$$RMSE_{oob} = \sqrt{\sum_{i=1}^n \frac{(y_i - \hat{y}_{i-\text{loc}})^2}{n}} \quad (17)$$

where y_i is the actual value for observation i , $\hat{y}_{i-\text{loc}}$ is the predicted value for observation i from the local model, \bar{y} is the average value of the dependent variable, and n is the total sample size.

For every local model, G-XGBoost calculates the feature importance using the built-in function of XGBoost but integrating the spatial weights ws_i . The split is decided based on the gain function, given as

$$\text{Gain} = \frac{1}{2} \left[\frac{\left(\sum_{i \in I_L} w s_i g_i \right)^2}{\sum_{i \in I_L} w s_i h_i + \lambda} + \frac{\left(\sum_{i \in I_R} w s_i g_i \right)^2}{\sum_{i \in I_R} w s_i h_i + \lambda} - \frac{\left(\sum_{i \in I} w s_i g_i \right)^2}{\sum_{i \in I} w s_i h_i + \lambda} \right] - \gamma \quad (18)$$

where I_L and I_R are the observation (instance) sets of left and right nodes after the split, with γ and λ the regularization parameters. If the gain is positive, the split is kept; otherwise it is not. As such γ controls the depth of the tree and the model complexity. G-XGBoost applies as default the ‘gain’ built-in XGBoost importance type, which calculates the average gain across all splits where a feature was used on the spatially weighted observations. The ‘gain’ method reflects the contribution of the corresponding feature to the model in relation to all features’ contributions for each tree in the model. In G-XGBoost, the contribution is also affected by the spatial weight. A higher value of this metric implies that a feature is more important for generating a prediction. The average gain for every feature across all n observations can be computed to compare with the global XGBoost feature importance output and cross-check differences in feature rankings in the global and local models. The ‘weight,’ ‘cover,’ ‘total gain’ and ‘total cover’ importance types are also available in G-XGBoost through the XGBoost library (see XGBoost 2024).

Locally varying importance has also been implemented in GRF and applied in various works to date (Lotfata et al. 2024, Lotfata et al. 2023, Grekousis et al. 2022). However, G-XGBoost does not use the feature importance methods used in GRF, such as the permutation feature importance, or the increase in the Mean Squared Error (MSE). G-XGBoost integrates the idea of spatial weights to the calculation of feature importance by including them in the gain function. In contrast, GRF does not use spatial weights when determining feature importance. Therefore, the local feature importance in G-XGBoost is not the result of some unweighted permutation procedure or change in MSE but is spatially weighted, giving more importance to the observations and their features that contribute to splitting a tree node.

It should be highlighted that while G-XGBoost does not directly model spatial dependence, it can help identify potential spatial autocorrelation. This is accomplished through the use of spatial weights in calculating the local models and the local feature importance. In specific, the spatial variation in local feature importance indirectly reveals how spatial dependencies might influence relationships. For instance, a sudden change in the local importance of a variable within a specific area could indicate a localized spatial effect or the influence of neighboring observations. In addition, areas with poor model fit (i.e., high residuals, or low R-squared values) could be indicative of unmodeled spatial dependence or unobserved spatial patterns. Therefore, although G-XGBoost cannot be used for modeling spatial dependence, it can indirectly address some of its effects by identifying spatial patterns in local feature importance and residuals.

2.3 Ensemble model

G-XGBoost goes one step further and provides an ensemble solution by combining the predictions of the global and local models. We use the weighted averaging for the ensemble

$$\hat{y}_{i-\text{ens}} = \alpha_i \hat{y}_{i-\text{loc}} + (1 - \alpha_i) \hat{y}_{i-\text{gl}} \quad (19)$$

where $\hat{y}_{i-\text{ens}}$ is the prediction of the ensemble for the i location, and α_i a newly introduced hyperparameter called *alpha weight* (alias *alpha_wt*), which takes values between zero and one and controls spatial dependence (further explained below). *Alpha weight* (α_i) can be fixed across all local i modes or vary according to the residual errors e_i of the global and local model for the i -th spatial unit, as shown here

$$\alpha_i = \begin{cases} [0, 1] & \text{if } e_{i-\text{loc}} > e_{i-\text{gl}} \\ 1 & \text{otherwise} \end{cases} \quad (20)$$

where

$$e_{i-\text{gl}} = y_i - \hat{y}_{i-\text{gl}} \quad (21)$$

and

$$e_{i-\text{loc}} = y_i - \hat{y}_{i-\text{loc}} \quad (22)$$

The ensemble architecture estimates the prediction for a spatial unit i , by combining the predictions for i from two models: the global model and the local model. The global model ($\hat{y}_{i-\text{gl}}$) is calculated with the standard XGBoost algorithm given by Eq. (1). The local model ($\hat{y}_{i-\text{loc}}$) is calculated using the G-XGBoost algorithm given by Eq. (13). However, G-XGBoost incorporates the alpha weight hyperparameter. If alpha weight is zero, then G-XGBoost is calculated directly through Eq. (13). In other words, Eq. (13) is just a special case of the G-XGBoost model when the global model is not considered in the solution. If alpha weight is nonzero, we extend Eq. (13) by adding an additional term to the local model, the global one, with both terms being controlled by the alpha weight hyperparameter. The ensemble model's prediction for spatial unit i is the weighted average of the predictions from the local and global model for i given by Eq. (19). For example, if the *alpha weight* is set to 0.70, the prediction for the spatial unit i would be 70% from the local model and 30% from the global model.

Using Eq. (20), and varying alpha weight value, we favor the local model when it performs worse than the global one, exhibiting thus higher residual errors. In this case, we boost the overall performance with the addition of the global model by setting a value of α_i between zero and one. A value of zero for α_i indicates zero spatial dependence, and thus, the local model is not considered. Hence, the ensemble model drops to the global model. On the other hand, a value of one for α_i indicates high dependence and that the global model is not considered. In practice, a reasonable starting value of *alpha weight* is 0.5, indicating the existence of spatial dependence and the need to create an ensemble model expected to perform better than the global

or local one. Local Moran's I can be used to estimate spatial autocorrelation and the need for an *alpha weight* hyperparameter. In the case of no spatial dependence and no spatial heterogeneity, the ensemble will probably not be necessary, as the global model is expected to perform with high accuracy. *Alpha weight* should not be confused with the alpha hyperparameter of XGBoost, referring to L1 regularization term on weights (analogous to Lasso regression).

The above ensemble is created following the three basic criteria that a good ensemble should follow (Zhou 2012, p. 100). First, it should perform better than any contributing member. This can be tested directly by comparing the ensemble model's evaluation metrics with the ensemble's individual models. Second, it should offer diversity, which is the ability of the ensemble to predict values different from the individual ones, or in other words, to have a diversity in the errors on any given sample. Diversity also reflects the fact that different learners are able to learn different knowledge. In the case of G-XGBoost, this is achieved because the local and global models analyze different sets under different perspectives. Third, learners should be independent, such that the prediction of one learner does not influence the prediction of another in the ensemble. For example, one model can influence another if it attempts to correct the predictions made by it. Independence also increases ensemble diversity. The global and local models' ensembles in G-XGBoost are independent, as the prediction of one does not influence the prediction of the other. Although diversity and independence are crucial for a good ensemble, there is no strict threshold value that measures these two criteria (Zhou 2012). A more detailed analysis of the rationale of this ensemble from the spatial analysis perspective is given in the Closing remarks section.

2.4 G-XGBoost hyperparameter tuning through nested cross-validation

Hyperparameters' optimization may have a crucial impact on the performance of every method. There are three basic strategies for choosing parameters in ML benchmark studies (Weber et al. 2019). The first one is using the default values. While very simple, it does not guarantee an unbiased approach, as default values may vary across programming languages or different libraries in the same language. The second approach is to use hyperparameter settings based on previous knowledge. This method relies significantly on the researcher's choices, which could infer bias. Lastly, the third method uses automated methods such as grid search across ranges of values or random search. Grid search and random search can also be combined with one of the various cross-validation (CV) methods for more accurate results.

G-XGBoost uses grid search through nested CV for hyperparameter optimization. Nested CV can be used for both hyperparameters optimization and when comparing several models to select the most appropriate for a specific dataset. Typical non-nested CV, such as k-fold, used for both tuning and selecting a model, is likely to infer biased model performance. One reason is that it can underestimate the overfitting that results from the hyperparameter tuning procedure. In addition, it may infer data leakage, which is undesirable in ML. For this reason, nested CV is preferred when comparing different ML models (Cawley et al. 2010). Briefly, nested CV is composed of an inner

loop CV nested in an outer CV. A hold-out subset of the data (in the outer loop) is completely separated from the training dataset and any preparations performed in the inner loop. The inner loop is responsible for model selection/hyperparameter tuning. The best model of the inner loop is used for error estimation of the hold-out subset in the outer loop. Lastly, the error of all models in the outer loop is averaged, and the error estimation is assessed, for example, through standard deviation. To conclude, the above process applied in G-XGBoost aims to tune hyperparameters and estimate an unbiased generalization performance: in other words, to assess the model's predictive accuracy and its expected error in unseen data. Lastly, the cost time of tuning non-spatial hyperparameters is the same between G-XGBoost and XGBoost. The reason is that hyperparameter tuning occurs in the global model, and then, hyperparameter values are plugged into the local models.

2.5 G-XGBoost bandwidth hyperparameter tuning

G-XGBoost uses the CV criterion which is a common metric for determining the optimal neighborhood size required for model estimation (Farber and Páez 2007). The bandwidth value b that minimizes CV—the cross-validation score, given by Eq. (11), maximizes the model's predictive power. However, though G-XGBoost has enabled the multi-threading support offered by XGBoost for efficient multi-core parallel processing, the optimal bandwidth evaluation can be computationally expensive. For this reason, G-XGBoost enables bandwidth search through step size control.

Suppose we have a set of 1000 spatial units, and we need to optimize the optimal bandwidth value for a range of 100 nearest neighbors threshold values (i.e., testing a kernel from 30 to 130 nearest neighbors). In that case, the model should run 100 times for each spatial unit, which is $1000 \times 100 = 10,000$ times in total, significantly more than the global model, which runs only once. To overcome an exhaustive search (e.g., 30–100 number of nearest neighbors), the G-XGBoost allows for searching over varying bandwidth step sizes (e.g., for every ten neighbors) to minimize the CV criterion. Then, following a gradient-descent-like strategy, in the next iteration, the step size is reduced to search around the number of neighbors that minimized CV in the previous iteration. The process continues until the minimum CV value is reached or the step size drops to one (see Supplementary Note 1 for an example). This strategy significantly reduces time and leads to a near-optimal selection of the bandwidth value. G-XGBoost also provides an exhaustive search if the step size is set to one for the entire range of search values. Defining the optimal bandwidth value is unavoidable in the context of local modeling, and (similar to GWR and GRF) the additional computational cost is the trade-off between creating local spatial models with properly defined scale of analysis with the global models that do not apply the bandwidth concept.

2.6 G-XGBoost for prediction

G-XGBoost can predict unknown data using pre-trained local G-XGBoost models. Three criteria should be met for prediction: (i) we should have n spatial units with known centroid coordinates (up_i, vp_i) ; (ii) the spatial units are within

the same geographical boundaries as the trained local G-XGBoost models; and (iii) the spatial units should have the same predictors as the trained G-XGBoost model. To predict the values of the dependent variable, for every spatial unit i , G-XGBoost will pick the nearest local model and the *alpha weight* value of that model to fuse it with the global model and produce the outcome. The nearest local model is the one with the minimum Euclidean distance between spatial unit i with coordinates (up_i, vp_i) and the set of all central points (trained local models) with coordinates (u, v) . *Alpha weight* can be fixed for all n spatial units or varying by using the *alpha weight* value defined during the model's training phase by Eq. (20).

It should be mentioned that the prediction functionality of the G-XGBoost cannot be applied in completely different locations from those where the original local G-XGBoost models were trained, as spatial weights would be different, and thus, the model output would be inaccurate. For example, a G-XGBoost model trained and validated for Boston housing prices cannot be used to predict housing prices in New York City. This is because the spatial weights used to train the Boston model, which reflect the unique spatial relationships and influences within that specific area, will differ significantly from the spatial weights that would be relevant for the New York City housing market.

Consequently, G-XGBoost is applicable for prediction within the same geographic region where the original model was trained. This limitation stems from the inherent local nature of the model architecture, which is designed to capture the specific spatial characteristics of the training data through the use of spatial weights and the notion of central unit (with its associated centroid coordinates stored as an extra property in the local model which are used to retrieve the nearest local model during prediction). This is an additional difference between G-XGBoost to non-spatial ML models, such as standard XGBoost or RF, which are global approaches aiming to identify general patterns and relationships that can potentially be applied to new datasets, even if they originate from different geographic locations.

2.7 G-XGBoost library

G-XGBoost is compiled into a Python library named 'geoxgboost,' and can be found at <https://pypi.org/project/geoxgboost/>. Geoxgboost library has enabled parallelization through the XGBoost library for efficient multi-core parallel processing. Documentation of the library is available at <https://geoxgboost.readthedocs.io/en/latest/>. A detailed manual on how to use and apply G-XGBoost on a test dataset is available at <https://doi.org/10.6084/m9.figshare.28302887.v1> and at <https://github.com/geogreko/DemoGXGBoost>.

2.8 G-XGBoost concluding key points

The key points of G-XGBoost can be summarized as follows:

- (i) G-XGBoost is calibrated locally through spatial weighting, attempting in this way to address spatial heterogeneity and allow for further investigation of spatial dependence.
- (ii) The spatial weights are multiplied with the first and second gradients to focus on the samples that cause the most error.
- (iii) Two types of spatial kernels are introduced, namely fixed and adaptive.
- (iv) G-XGBoost estimates the optimal bandwidth values for the spatial kernels by minimizing the CV criterion. The algorithm allows for searching over varying bandwidth values (using step size) to avoid an exhaustive search that reduces computational cost significantly.
- (v) G-XGBoost allows for hyperparameter optimization through built-in functions.
- (vi) G-XGBoost assesses how the importance of the independent variables (on how they influence the dependent variable) varies locally (spatial heterogeneity) by integrating spatial weights in feature importance calculation.
- (vii) G-XGBoost is an ensemble model that averages global and local data knowledge through a new hyperparameter, significantly improving performance.

3 Experimental results

3.1 Experimental datasets

G-XGBoost was evaluated using six real-world (see Table 1) benchmark datasets that have been used in previous works on spatial analysis and regression (Ahmed et al. 2021; Baller et al. 2001; Grekousis 2021; Kalogirou et al. 2024; Liu et al. 2022; Pace and Gilley 1997; Pace and Barry 1997; Wiedemann et al. 2023). A complete presentation of each dataset can be found in Supplementary Note 2. The selection of the datasets is made so that the algorithm is tested across various dimensionality settings, scales of analysis, and thematic scopes. The algorithm was tested from simple low-dimensionality geographical sets containing 325 observations (spatial units) and three independent variables to relatively larger ones (at least from the spatial analysis perspective) with 5164 observations and four independent variables, as well as 3021 observations and 29 independent variables. Evaluating the algorithm across various dimensions is essential for its capacity to perform in various data settings, thus offering a generalized assessment of its performance. The idiosyncrasy of local geographic modeling lies in the fact that a model is applied for each spatial unit. Though a dataset of 1000 observations by 10 features nowadays is regarded as small, a local model should run 1000 times (for every observation), in contrast to the global model, which would run only once for the entire dataset.

The thematic scopes of the benchmark datasets cover typical problems in spatial analysis such as economic analysis (home value evaluation, household income), geography of health (mortality rate, cancer rate), and geography of crime (homicides). The selection of the datasets was also made so that the algorithm is tested across various analysis scales, from block group, one of the smallest geographical

Table 1 Real-world benchmark datasets ordered by complexity

Dataset	Dependent variable	Observations/Scale of analysis	Independent variables	Source
Greece income	Yearly mean household income	325 municipalities	3	https://cran.r-project.org/web/packages/SpatialML/SpatialML.pdf
Boston housing	Median value of owner-occupied homes	506 census tracts	13	http://lib.stat.cmu.edu/datasets/boston_corrected.txt
Atlantic mortality	Mortality rate	666 counties	5	https://www.dropbox.com/s/lrz6og0ld2m64df/Data_GWR_7z?dl=0&e=1
New York cancer	Total cancer incidents per 1000 people	973 block group	6	https://www.satscan.org/datasets/nyscancer/index.html
South crime	Homicide rate	1412 counties	7	https://geodacenter.github.io/data-and-lab/south/
California housing	Natural logarithm of median house values	5,164 block group	4	https://www.dcc.fc.up/~ltorgo/Regression/cal_housing.html

units for which the US Census Bureau publishes sample data, up to the census tract and the county level.

3.2 Baseline models and evaluation criteria

G-XGBoost was evaluated against OLS, GWR, RF, GRF, GRF-W, and XGBoost baseline models (based on the OOB errors of the local models) using three standard metrics, namely R^2 , given by Eq. (15), MAE given by Eq. (16), and RMSE given by Eq. (17). In general, it is unlikely that a single method will perform best across all metrics: Therefore, a good approach is to identify the method that consistently overperforms even if it lags on some metrics in some experiments (Weber et al. 2019). Similarly, tracing methods that consistently underperform are not a good choice, even if they exhibit good metric results in some experimental settings.

3.3 Experimental environments and results

Gradient boosting models were fit with the `xgboost` 2.0.3 package using the `scikit-learn` API, in Python Version 3.10 within a Conda environment under PyCharm IDE Community Edition 2023.3.3. To implement the RF, GRF, and GRF-W models, the `R` packages ‘`randomForest`’ (Liaw et al. 2022) and ‘`SpatialML`’ (Kalogirou et al. 2024) were applied. OLS models were computed using the `scikit-learn` 1.4.1 Python library, while the software `GWR4.0` was used for GWR.

The adaptive spatial kernel was applied for all spatial models (GWR, GRF, GRF-W, and G-XGBoost). Built-in functions were employed to define the optimal bandwidth value for GWR and GRF-W. However, GRF does not have such functionality, so the optimal value resulted from the GWR-W model. For the G-XGBoost model, the optimized bandwidth value minimizes the CV criterion (see Sect. 2).

Grid search was combined with user-defined values in the global and local XGBoost through nested CV with $k=5$ folds in the outer loop and $k=3$ folds in the inner loop. Multiple values were tested across a range of hyperparameters, such as `n_estimators`, `learning_rate`, `max_depth`, and `min_child_weight`. Stochastic gradient descent was applied through `column` and `row` subsampling and the hyperparameters `subsample` and `colsample_bytree`. In essence, stochastic gradient descent is a variation of gradient descent in which, at each iteration, a random subsample of rows or columns of the training set is drawn from the training sample without replacement. To avoid biased representation, such as extensive tuning of hyperparameters in the proposed method while using the default parameters for competing methods, we also tuned the hyperparameters of RF, GRF, and GRF-W.

3.4 G-XGBoost results

Three versions of G-XGBoost were compared with the baseline models (see Sect. 3.2). In practice, these three versions are realized by different choices in

the properties and hyperparameters of the G-XGBoost algorithm through the G-XGBoost library. Spatial weights were not used in the first version, labeled L-XGBoost. Local models were created and trained independently considering that all spatial units have the same weight. In the second version, spatial weights were calculated through spatial kernels for the local models (LW-XGBoost). The third version (G-XGBoost) used local models, spatial weights, and the ensemble of local and global models. The results of these three versions are demonstrated to better understand how each one of the key modifications to XGBoost, namely a) local models, b) spatial weights, and c) an ensemble of local and global models, improves the original XGBoost algorithm.

Global Moran's I index was calculated for all models to assess whether the residuals exhibit spatial autocorrelation. The null hypothesis states that the residuals are randomly distributed. P values smaller than a significance level (i.e., 1%) indicate statistically significant spatial autocorrelation of the residuals, which is undesirable, as it indicates model misspecification. However, as the objective was to test G-XGBoost over benchmark datasets, which have fixed variables, misspecified models were anticipated. Therefore, related issues, like omitted or irrelevant variables, were not attempted to be fixed. However, comparing Moran's I index statistical significance through p values across various local regression methods can potentially reveal whether, even in misspecified models, errors are less clustered with some methods compared to others.

OLS and GWR were evaluated using casual R^2 , MAE, and RMSE metrics (not OOB). Tree-based metrics are not evaluated on these metrics but on their OOB equivalents. A direct comparison of OLS and GWR is possible if we calculate the non-OOB metrics for tree-based algorithms based on the training dataset presented in Supplementary Table S1. Results show that OLS and GWR have lower R^2 and higher MAE and RMSE than tree-based models. However, overfitting using the training dataset for nonlinear models is likely. For this reason, we calculated the OOB metrics, the standard way of evaluating tree-based models, which are more reliable, as they are calculated on data not previously seen by the model (see Table 2). XGBoost (global) refers to the metrics values of the test set when trained with the hyperparameters produced from the nested CV approach. The generalization performance of the nested CV model is presented in Supplementary Table 2.

The results, summarized in Table 2, show that G-XGBoost improves the global and local versions of XGBoost and outperforms RF, GRF, and GRF-W. Indicatively, for the set 'Atlantic mortality,' we notice the G-XGBoost achieves an $R^2=67.78\%$, which is much higher than the local model with spatial weights (LW-XGBoost), with $R^2=61.94\%$, and the global model (XGBoost), with $R^2=56.86\%$. Similarly, G-XGBoost decreases substantially the MAE_{oob} and the RMSE_{oob}. G-XGBoost achieves better metrics for the 'South crime' set, which has a very low R^2 across all models, perhaps due to missing variables. All models, including L-XGBoost and LW-XGBoost, have R^2 ranging from 26.70% (GRF) to 33.38% (RF). G-XGBoost achieves $R^2=39.81\%$, an additional 6.43% from the highest R^2 achieved by RF. Likewise, for the 'New York cancer' set, G-XGBoost achieves $R^2=47.16\%$, much higher than the second-best XGBoost ($R^2=40.02\%$). Finally, G-XGBoost performs

Table 2 OOB evaluation results. Bold values indicate maximum values for R^2 oob, and minimum values for MAEoob and RMSEoob

Set:	Greece income					Boston housing				
	Model	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	
Set:	RF	71.69	1030.988	1563.575	0.00002	89.12	2.052	3.026	0.00000	
	GRF	73.77	1029.613	1504.996	0.00687	87.92	2.106	3.189	0.00005	
	GRF-W	72.72	1060.954	1535.065	0.04034	86.97	2.115	3.311	0.00821	
	XGBoost	76.57 (test)	1010.136 (test)	1383.209 (test)	N/A	87.33 (test)	2.0240 (test)	3.108 (test)	N/A	
	L-XGBoost	73.02	1046.645	1526.519	0.00224	89.49	1.949	2.975	0.01171	
	LW-XGBoost	74.32	1035.423	1488.990	0.06339	90.80	1.948	2.782	0.03337	
	G-XGBoost	79.22	874.702	1339.191	0.03557	93.67	1.580	2.308	0.00936	
New York cancer										
Set:	Model	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	
	RF	58.01	6.089	7.818	0.00000	30.65	7.738	11.258	0.00000	
	GRF	60.59	5.818	7.574	0.00000	35.67	7.522	10.843	0.00000	
	GRF-W	58.75	5.977	7.749	0.00000	36.33	7.530	10.787	0.00021	
	XGBoost	56.86 (test)	6.6790 (test)	8.590 (test)	N/A	40.02 (test)	7.404 (test)	10.186(test)	N/A	
	L-XGBoost	60.22	5.934	7.610	0.00000	34.69	7.590	10.925	0.00000	
	LW-XGBoost	61.94	5.783	7.444	0.00000	38.57	7.361	10.596	0.00060	
	G-XGBoost	67.78	5.155	6.849	0.00000	47.16	6.519	9.827	0.00000	
California housing										
Set:	Model	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	
	RF	33.38	4.134	5.743	0.00000	62.66	0.399	0.556	0.00000	
	GRF	31.93	4.130	5.805	0.00000	73.14	0.300	0.472	0.00000	
	GRF-W	26.70	4.247	6.024	0.00000	74.05	0.289	0.464	0.00000	

Table 2 (continued)

Set: Model	South crime			California housing			
	R^2 oob (%)	MAE oob	RMSE oob	Moran's I p -value	R^2 oob (%)	MAE oob	RMSE oob
XGBoost	33.19 (test)	4.370 (test)	6.454 (test)	N/A	63.68 (test)	0.409 (test)	0.570 (test)
L-XGBoost	31.59	4.158	5.820	0.00000	76.65	0.286	0.444
LW-XGBoost	32.27	4.095	5.791	0.00000	77.92	0.274	0.432
G-XGBoost	39.81	3.743	5.459	0.00000	83.79	0.228	0.370

Bold values in Moran's I indicate no spatial autocorrelation presence at the residuals (p values larger than 1%)

better in the ‘California housing’ set, a large dataset with 5,164 observations and four independent variables, showing that G-XGBoost can successfully handle small and large spatial datasets (from the spatial analysis perspective).

As datasets are different and the hyperparameters used vary, we cannot directly compare the G-XGBoost performance of one set to the G-XGBoost performance of another (e.g., NY cancer to Boston housing). However, we can compare how much the evaluation metrics R^2 , MEA, and RMSE were improved by using G-XGBoost and analyze these values across all datasets. In other words, we calculated the % change in R^2 , MAE, and RMSE of G-XGBoost to RF, GRF, GRF-W, XGBoost, L-XGBoost, LW-XGBoost, for all datasets. Moreover, we used nonparametric bootstrapping to calculate the 95% confidence intervals of the estimated improvement. Bootstrapping is a statistical method for estimating quantities and related confidence intervals (e.g., means) from a data sample by measuring those quantities when sampling from an approximating distribution (Grekousis 2019). In practice, for every set (% increase in R^2 , % decrease in MAE, and % decrease in RMSE), 1000 subsamples are created with replacement. Then, the mean value and the 95% confidence intervals (using all subsamples) are calculated. The Kernel probability density estimate of the bootstrapped means is also calculated to show how the improvement of G-XGBoost over the tree-based algorithms in R^2 , MAE, and RMSE values is distributed over the 1000 resampled values (see Fig. 2A). These values are also depicted in Fig. 2B as boxplots to provide a more straightforward comparison regarding key characteristics of the distributions such as the median, quartiles, and outliers. Results reveal that G-XGBoost achieved higher accuracies in all cases. On bootstrapped mean values, G-XGBoost improved R^2 by 17.45% (CI 13.94%, 21.91%), MAE by 17.42% (CI 15.33%, 20.76%), and RMSE by 14.53% (CI 12.08%, 17.38%). These values highlight that G-XGBoost significantly improved all evaluation metrics compared to the tree-based algorithms tested under the specific benchmark datasets.

Spatial autocorrelation of residuals is present in most models. Nevertheless, as we mentioned above, this is expected, as we used fixed benchmark datasets and did not attempt to add missing variables to fix model misspecifications. However, for the set ‘Boston housing’, the LW-XGBoost shows no spatial autocorrelation of the residuals at the 1% significance level (see Table 2). G-XGBoost could potentially remove spatial autocorrelation in the residuals if we tuned the *alpha weight* hyperparameter (we used the default values for all datasets). A similar result is observed in the set ‘Greece income’, where LW-XGBoost does not indicate spatial autocorrelation of the residuals at the 5% significance level, while G-XGBoost exhibits no spatial autocorrelation of the residuals at the 1% significance level. On the contrary, GRF does not manage to handle spatial autocorrelation, and GRF-W only addresses it for the ‘Greece income’ set. This shows that G-XGBoost and its versions can better handle spatial autocorrelation of residuals for an adequately specified model, probably because spatial weights are used to minimize the residuals during the node split of each tree.

Lastly, Fig. 3 presents, indicatively, the output of G-XGBoost for the NY and Atlantic sets. For example, mapping the primary factor of cancer per city block (NY) is significant in tracing the prevailing feature at every spatial unit (see Fig. 3A). Further analysis can integrate sociodemographic variables to study why

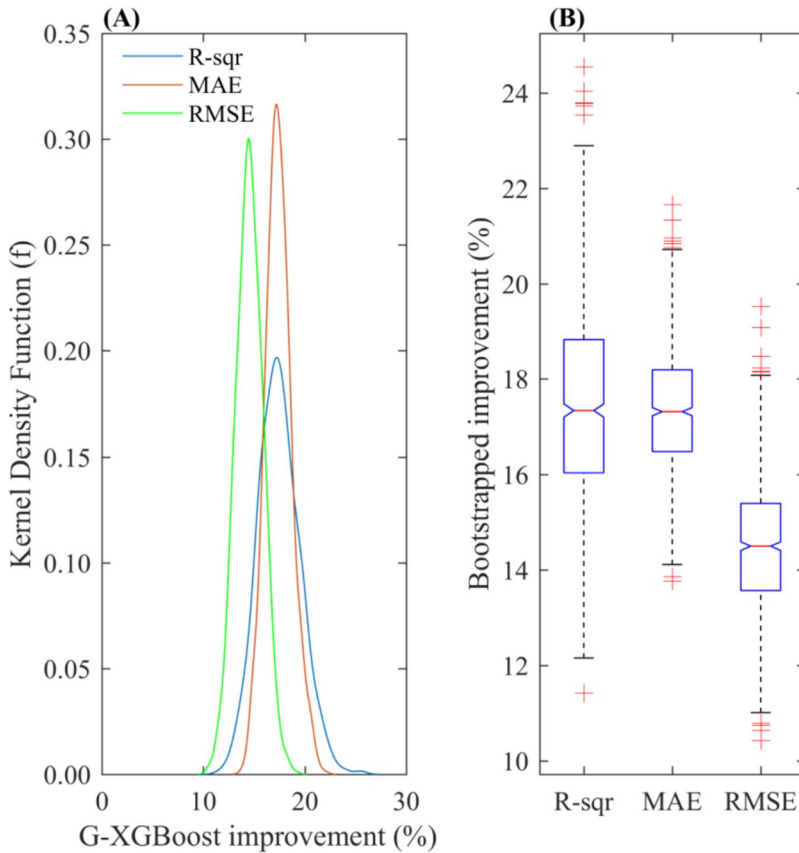


Fig. 2 G-XGBoost improvement (%) of OOB metrics over tree-based models through parametric bootstrapping. **A** Kernel probability density estimate of bootstrapped means and **B** boxplots of G-XGBoost improvement for R^2 , MAE, and RMSE, including median, 25th, and 75th percentiles (marked at upper and lower edges of boxes). Red crosses indicate outlier values

a factor prevails in a spatial unit. Mapping the local importance of a single feature (i.e., household size for the NY set) can assist in tracing spatial heterogeneity. For example, Fig. 3B shows how the importance of household size varies locally, getting higher values mainly in northern Manhattan. This provides a better understanding of how every feature influences the dependent variable compared to local coefficients in GWR, as importance is directly comparable across all features. Similar to GWR, mapping R^2 values for every local G-XGBoost model provides details on the local performance of the model (see Fig. 3C). Lastly, standardized residuals can be calculated and mapped for further analysis through local and global spatial autocorrelation metrics to identify whether the method has removed spatial autocorrelation of the residuals (see Fig. 3D).

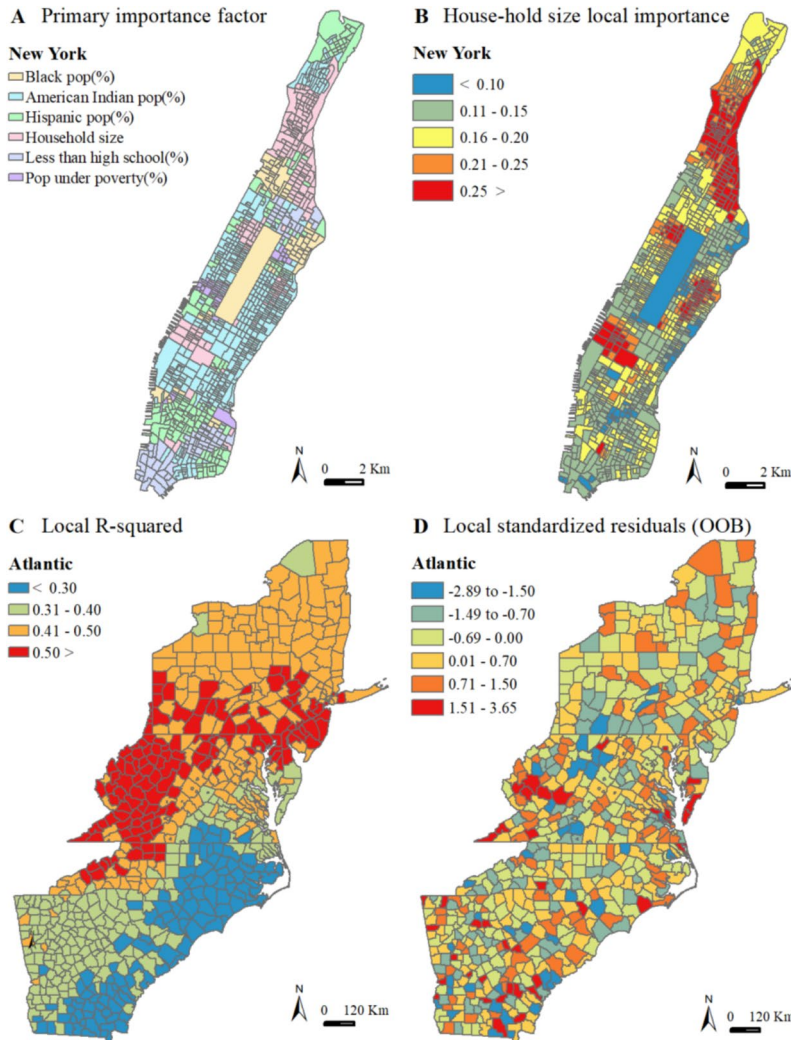


Fig. 3 G-XGBoost local outputs enhancing explainable ML and subsequent spatial analysis

4 Closing remarks

We propose a new ensemble model named G-XGBoost, which implements SML for spatially local regression and extends the well-established XGBoost algorithm to handle spatial heterogeneity. G-XGBoost was tested over six real-world benchmark datasets and outperformed all tested baseline models under all experimental settings.

From a technical perspective, G-XGBoost modifies the original XGBoost in four key ways. First, it employs multiple local models instead of a single global one. To do so, the concept of geographical weighting is applied through the integration of spatial weights via a bi-square spatial kernel function. Second, it allows for

optimizing the bandwidth value that defines the extent of the spatial kernel by minimizing the CV criterion. Third, it estimates the local feature importance by integrating spatial weights. Lastly, it creates an ensemble model that combines the knowledge of the global and local models, leading to increased predictive performance.

From a spatial perspective, G-XGBoost makes two contributions to the current literature. First, it is an exploratory tool, as it allows for analysis and mapping of how the importance of the independent variables varies, tracing thus spatial heterogeneity. Second, it is a predictive tool that, in contrast to other spatial ML models, mixes global with local knowledge in an ensemble structure.

Regarding the first contribution, the relationships between the dependent and independent variables are likely to vary in space due to spatial heterogeneity. G-XGBoost traces spatial heterogeneity through the local calculation of feature importance. This leads to a very convenient way to explore spatial heterogeneity by mapping the varying importance of every independent variable and tracing areas exhibiting different spatial patterns. The above approach is similar to how GWR traces spatial heterogeneity through mapping varying regression coefficients (Brunsdon et al. 1996).

Regarding the second contribution, there are two main reasons to move to an ensemble model. First, a properly defined ensemble model is expected to achieve improved predictive performance compared to the individual models from which it is constructed. The second reason has to do with the spatial dependence and heterogeneity issues that exist in spatial datasets. Specifically, XGBoost splits data based on the objective function, which minimizes the residuals. In consequence, it may not be able to generalize data if residuals are outside the boundaries of those in the training set. Like RF, XGBoost cannot extrapolate and predict to a range of values not initially trained. In this sense, XGBoost can only predict within the range of values of the explanatory variables initially trained. When a dataset is large, this is not likely to infer serious problems. However, for models with a relatively small training set, like the local models, this could lead to poor performance. Similarly, the presence of spatial autocorrelation may lead to variable values and residuals with little variance. As such, the local model will learn from a dataset with relatively low variability. However, when combined with the global dataset that includes information on how residual values are distributed across the entire dataset, this knowledge can be added to the model and improve predictive performance.

Future research could explore several pathways. G-XGBoost could be compared with spatial regression models like SLM or SEM, known for their ability to effectively handle spatial dependence. More importantly, research could investigate how G-XGBoost can directly account for spatial autocorrelation by incorporating spatial features such as spatial lag or eigenvector spatial filtering. In addition, a thorough comparison of local feature importance and local R^2 values between G-XGBoost and GRF could be conducted to test the robustness and effectiveness of these models across multiple datasets and experimental settings.

To conclude, the G-XGBoost ensemble of global local models assists in capturing spatial heterogeneity and anisotropic space, which reflects the spatially varying relationship among the variables studied. The global XGBoost model analyzes the

entire dataset, thus offering a full spectrum of data variance, but does not consider spatial dependence and heterogeneity at all. On the contrary, the local model is only trained on a relatively small subset of the dataset, and with the inclusion of spatial weights and bandwidth, it introduces geographical awareness, thus handling spatial heterogeneity while allowing for further investigation of spatial dependence presence. Thus, the two models contribute different aspects of data knowledge, making a highly diverse ensemble. We always run the risk of getting a model of low predictive performance. This can be due to low data availability or feature omission, leading to model misspecification. Yet even a model of low predictive accuracy can be beneficial if it allows for analysis of how feature importance varies in space. G-XGBoost offers this characteristic by calculating the feature importance for each spatial unit, making it a valuable exploratory tool in local decision-making and geographical analysis.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10109-025-00465-4>.

Funding No funding was received for conducting this study.

Data availability The data and codes that support the findings of this study are available with the identifier at <https://figshare.com/s/b049a9dbefd64fd827aa>

Declarations

Competing interest The authors have no competing interests to declare that are relevant to the content of this article.

References

- Anselin L (2009) Spatial regression. In: Fotheringham AS, Rogerson PA (eds) *The SAGE handbook of spatial analysis*. Sage, Thousand Oaks, pp 255–276
- Ahmed ZU, Sun K, Shelly M, Mu L (2021) Explainable artificial intelligence (XAI) for exploring spatial variability of lung and bronchus cancer (LBC) mortality rates in the contiguous USA. *Sci Rep* 11(1):24090. <https://doi.org/10.1038/s41598-021-03198-8>
- Baller RD, Anselin L, Messner SF, Deane G, Hawkins DF (2001) Structural covariates of US county homicide rates: Incorporating spatial effects. *Criminology* 39(3):561–588
- Brunsdon C, Fotheringham AS, Charlton ME (1996) Geographically weighted regression: a method for exploring spatial nonstationarity. *Geogr Anal* 28(4):281–298
- Cawley GC, Talbot NL (2010) On over-fitting in model selection and subsequent selection bias in performance evaluation. *J Machine Learn Res* 11:2079–2107
- Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp 785–794
- Chen Y, Zhang X, Grekousis G, Huang Y, Hua F, Pan Z, Liu Y (2023) Examining the importance of built and natural environment factors in predicting self-rated health in older adults: an extreme gradient boosting (XGBoost) approach. *J Clean Prod* 413:137432. <https://doi.org/10.1016/j.jclepro.2023.137432>
- Credit K (2024) Introduction to the special issue on spatial machine learning. *J Geogr Syst* 26:451–460. <https://doi.org/10.1007/s10109-024-00452-1>
- Effati M, Thill JC, Shabani S (2015) Geospatial and machine learning techniques for wicked social science problems: analysis of crash severity on a regional highway corridor. *J Geogr Syst* 17:107–135. <https://doi.org/10.1007/s10109-015-0210-x>

- Farber S, Páez A (2007) A systematic investigation of cross-validation in GWR model estimation: empirical analysis and Monte Carlo simulations. *J Geogr Syst* 9:371–396. <https://doi.org/10.1007/s10109-007-0051-3>
- Georganos S, Grippa T, Niang Gadiaga A, Linard C, Lennert M, Vanhuyse S, Mboga N, Wolff E, Kalogirou S (2021) Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling. *Geocarto Int* 36:121–136
- Georganos S, Kalogirou S (2022) A forest of forests: a spatially weighted and computationally efficient formulation of geographical random forests. *ISPRS Int J Geo Inf* 11(9):471. <https://doi.org/10.3390/ijgi11090471>
- Grekousis G (2019) Artificial neural networks and deep learning in urban geography: a systematic review and meta-analysis. *Comput Environ Urban Syst* 74:244–256. <https://doi.org/10.1016/j.compenvurbysys.2018.10.008>
- Grekousis G (2020) Spatial analysis methods and practice: describe–explore–explain through GIS. Cambridge University Press, New York
- Grekousis G (2021) Local fuzzy geographically weighted: a new method for geodemographic segmentation. *Int J Geogr Inf Sci* 35(1):152–174. <https://doi.org/10.1080/13658816.2020.1808221>
- Grekousis G, Feng Z, Marakakis I, Lu Y, Wang R (2022) Ranking the importance of demographic, socio-economic, and underlying health factors on US COVID-19 deaths: a geographical random forest approach. *Health Place* 74:102744. <https://doi.org/10.1016/j.healthplace.2022.102744>
- Griffith D (2004) Distributional properties of georeferenced random variables based on the eigenfunction spatial filter. *J Geogr Syst* 6:263–288. <https://doi.org/10.1007/s10109-004-0134-3>
- Griffith DA (2006) Hidden negative spatial autocorrelation. *J Geograph Syst* 8:335–355. <https://doi.org/10.1007/s10109-006-0034-9>
- Griffith DA, Fischer MM (2013) Constrained variants of the gravity model and spatial dependence: model specification and estimation issues. *J Geogr Syst* 15:291–317. <https://doi.org/10.1007/s10109-013-0182-7>
- Kalogirou S, Georganos S (2024) Spatial machine learning. SpatialML, R package (version April 2, 2024). Available online: <https://cran.r-project.org/web/packages/SpatialML/SpatialML.pdf> (accessed on 17 April, 2024)
- Kang J, Guo X, Fang L, Wang X, Fan Z (2022) Integration of Internet search data to predict tourism trends using spatial-temporal XGBoost composite model. *Int J Geogr Inf Sci* 36(2):236–252
- Li Z (2022) Extracting spatial effects from machine learning model using local interpretation method: an example of SHAP and XGBoost. *Comput Environ Urban Syst* 96:101845. <https://doi.org/10.1016/j.compenvurbysys.2022.101845>
- Liaw A (2022) Breiman and Cutler's random forests for classification and regression. Package 'randomForest' (version October 14, 2022). Available online: <https://cran.r-project.org/web/packages/randomForest/randomForest.pdf> (accessed on 17 April, 2024)
- Liu X, Kounadi O, Zurita-Milla R (2022) Incorporating spatial autocorrelation in machine learning models using spatial lag and eigenvector spatial filtering features. *ISPRS Int J Geo Inf* 11(4):242
- Lotfata A, Grekousis G (2023) Using geographical random forest models to explore spatial patterns in the neighborhood determinants of hypertension prevalence across Chicago, Illinois, USA. *EPB Urban Anal City Sci*. <https://doi.org/10.1177/23998083231153401>
- Lotfata A, Georganos S (2024) Spatial machine learning for predicting physical inactivity prevalence from socioecological determinants in Chicago, Illinois, USA. *J Geogr Syst* 26:461–481. <https://doi.org/10.1007/s10109-023-00415-y>
- Nikparvar B, Thill J-C (2021) Machine learning of spatial data. *ISPRS Int J Geo Inf* 10(9):600. <https://doi.org/10.3390/ijgi10090600>
- O'Sullivan D, Unwin D (2010) Geographic information analysis, 2nd edn. John Wiley & Sons, Hoboken
- Pace RK, Gilley OW (1997) Using the spatial configuration of the data to improve estimation. *J Real Estate Finan Econ* 14(3):333–340
- Pace RK, Barry R (1997) Sparse spatial autoregressions. *Statist Probab Lett* 33(3):291–297
- Schimohr K, Doeblér P, Scheiner J (2023) Prediction of bike-sharing trip counts: comparing parametric spatial regression models to a geographically weighted XGBoost algorithm. *Geogr Anal* 55(4):651–684
- VoPham T, Hart JE, Laden F, Chiang YY (2018) Emerging trends in geospatial artificial intelligence (geoAI): potential applications for environmental epidemiology. *Environ Health* 17(1):1–6

- Wang Y, Feng L, Li S, Ren F, Du Q (2020) A hybrid model considering spatial heterogeneity for landslide susceptibility mapping in Zhejiang Province China. *CATENA* 188:104425. <https://doi.org/10.1016/j.catena.2019.104425>
- Weber LM, Saelens W, Cannoodt R, Sonesson C, Hapfelmeier A, Gardner PP, Anne-Laure Boulesteix AL, Saeys Y, Robinson MD (2019) Essential guidelines for computational method benchmarking. *Genome Biol* 20:1–12. <https://doi.org/10.1186/s13059-019-1738-8>
- Wiedemann N, Martin H, Westerholt R (2023) Benchmarking regression models under spatial heterogeneity. In: 12th International Conference on Geographic Information Science (GIScience 2023)
- XGBoost (2024) xgboost Release 2.1.0-dev. Downloaded from <https://readthedocs.org/projects/xgboost/downloads/pdf/latest/> Accessed April 29, 2024
- Ye M, Zhu L, Li X, Ke Y, Huang Y, Chen B, Yu H, Li H, Feng H (2023) Estimation of the soil arsenic concentration using a geographically weighted XGBoost model based on hyperspectral data. *Sci Total Environ* 858:159798
- Zhou ZH (2012) Ensemble methods: foundations and algorithms. CRC Press, Boca Raton

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.