Journal of Data Analysis and Information Processing, 2023, 11, 293-309
https://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

A Hybrid Spatial Dependence Model Based on Radial Basis Function Neural Networks (RBFNN) and Random Forest (RF)

Mamadou Hady Barry[1], Lawrence Nderu[2], Anthony Waititu Gichuhi[3]

[1]Department of Mathematics, Pan African University, Institute for Basic Sciences, Technology and Innovation (PAUSTI), Nairobi, Kenya
[2]Department of Computing School and Information Technology (SCIT), Jomo Kenyatta University of Agriculture and Technology (JKUAT), Nairobi, Kenya
[3]Department of Statistics and Actuarial Science, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
Email: hady.mamadou@students.jkuat.ac.ke

How to cite this paper: Barry, M.H., Nderu, L. and Gichuhi, A.W. (2023) A Hybrid Spatial Dependence Model Based on Radial Basis Function Neural Networks (RBFNN) and Random Forest (RF). *Journal of Data Analysis and Information Processing*, **11**, 293-309.
https://doi.org/10.4236/jdaip.2023.113015

Received: June 7, 2023
Accepted: July 24, 2023
Published: July 27, 2023

Abstract

The majority of spatial data reveal some degree of spatial dependence. The term "spatial dependence" refers to the tendency for phenomena to be more similar when they occur close together than when they occur far apart in space. This property is ignored in machine learning (ML) for spatial domains of application. Most classical machine learning algorithms are generally inappropriate unless modified in some way to account for it. In this study, we proposed an approach that aimed to improve a ML model to detect the dependence without incorporating any spatial features in the learning process. To detect this dependence while also improving performance, a hybrid model was used based on two representative algorithms. In addition, cross-validation method was used to make the model stable. Furthermore, global moran's I and local moran were used to capture the spatial dependence in the residuals. The results show that the HM has significant with a R2 of 99.91% performance compared to RBFNN and RF that have 74.22% and 82.26% as R2 respectively. With lower errors, the HM was able to achieve an average test error of 0.033% and a positive global moran's of 0.12. We concluded that as the R2 value increases, the models become weaker in terms of capturing the dependence.

Keywords

Spatial Data, Spatial Dependence, Hybrid Model, Machine Learning Algorithms

1. Introduction

The real world can be represented as its geographic location. Virtually every-

thing that occurs or exists happens "somewhere". It is crucial to know "where" something occurred or existed. Geographic location is crucial since all human actions require understanding of the earth [1]. If the information contained in a map or photograph can be expressed digitally, it is because a variety of specialized computer systems have been developed to process geographical information [2].

Nowadays, topographic surveys, satellites, and other methods collect vast amounts of data in computer format, frequently with geographical references [2]. Data regarding the position, attributes and associations features in space are often termed spatial data [3]. Therefore being referenced to a particular location in space creates unique characteristics (properties). These characteristics provide difficulties and present opportunities for information mining. For that reason, the exploration of these data requires implicit or explicit knowledge included in spatial science [4].

Machine learning (ML) techniques enable computers to learn within data, extract knowledge, and recognize structures from huge and high-dimensional datasets. They can be supervised, unsupervised, semisupervised or reinforcement [5]. These techniques have been frequently used across many fields [6]. The emergence of ML algorithms has also been applied in spatial data. The main challenge is to build an adaptable model based on the interaction between geographical data and ML management that can handle and evaluate complex spatial information.

According to the first property of spatial data, "everything is related to everything else, but near things are more related than distant things", knows as spatial dependence [7]. Consequently, we would anticipate that the majority of geographic events will have some sort of spatial autocorrelation (dependence). The existence of this geographical relationship contradicts the concept of identical and independent distribution (i.i.d.) upon which many non-spatial statistical methods are based. This is frequently the case in population data since people who have similar qualities tend to live in similar neighborhoods for a variety of reasons, including house prices, proximity to employers, and cultural considerations. For this reason, a ML algorithm may not adequately capture a significant occurrence [8].

One of the most common applications of ML is spatial prediction, which uses samples for training to predict unknown values in specific locations [9]. ML algorithms have been used as potential replacement for geostatistical interpolation techniques (ordinary kriging and regression kriging) and for spatial analysis techniques (spatial autoregressive and geographically weighted regression). Kriging and its multiple variants have been utilized as the Best Unbiased Linear Prediction approach since the 1960s, but the authors [10] [11] and [12] have proven that ML models to be superior to geostatistical interpolation techniques in terms of prediction.

According to [13], by incorporating geographical proximity (buffer distance) effects into a learning process, for spatial predictions, a ML model is able to gen-

erate a prediction comparable to ordinary and regression kriging. In the same ideas, [14] incorporated spatial lag and ESF features in a ML model whose showed fewer mistakes and lower spatial autocorrelation compared to a model without spatial features. For spatial ML prediction, the incorporation of spatial features is commonly used in a model to account for spatial dependence.

To assess spatial dependence in models, moran's I methods was quantified [14], and a new integration of weights matrix was developed to detect spatial dependence patterns among residuals [15]. Therefore, when analyzing spatial data, we need to be conscious of the specificity that dependence involves when we conduct ML. In fact, the explicit management of the spatial dependence might improve the performance of the ML model or provide important new insights into how a task is learned. However, failure to appropriately consider or include that property into the ML model can impact learning.

The exploration of the spatial dependence within the learning algorithm is still in its early stages. An approach is developed with the goal of illustrating and improving a ML model to identify the dependence. This study focuses on a hybrid model of ML that captures the spatial dependence through the residuals without incorporating any spatial features in the model. Two representative ML models are proposed for spatial data: RBFNN, a type of neural network used for prediction tasks when the current outcome is affected by the neighbors' states or by contextual information and RF algorithm, that is also appropriate for the case of spatially dependent samples.

The paper is structured as follows: Section 2 presents the mathematical methods of the models; Section 3 presents their application; Section 4 discusses the experimental results; Section 5 provides the conclusion.

2. Methods

2.1. Models Specification

The purpose of supervised learning is to infer a function or mapping from labeled training data. It involves defining the input vector $X = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m)$ (the features that will be used to train the model, which may include spatial indexing) and the output vector y (the prediction or label that the model is trying to predict). The parameters are then used to determine how our model will use these features and labels in order to make accurate predictions.

The following algorithms will be developed: RBFNN, RF and Hybrid model.

2.2. Radial Basis Function Neural Networks

Radial Basis Function (RBF)

In mathematics, RBF is a function $\varphi(\boldsymbol{x})$ whose value mainly depends on the distance between the input and some fixed point. In the case where c is the fixed point, the formula can be expressed as:

$$\varphi(\boldsymbol{x}) = f\left(\|\boldsymbol{x} - \boldsymbol{c}\|\right), \tag{2.1}$$

$f\left(\|\boldsymbol{x} - \boldsymbol{c}\|\right)$ is the distance function.

In the domain of neural network, RBF is used to develop a mathematical model called Radial Basis Function Networks. The model uses radial basis function as activation function [16]. The notion behind RBF is that a predicted target value of a particular item is likely to be nearly equal to the other neighbors of the predictor variables. Therefore, An RBF network places one or multiple RBF neurons in the coordinate space, depending on what the predictor factors indicate. The Euclidean distance in the space concerned is between each neutron, where distance is calculated from the centre of the neutrons [17].

Let denote $\boldsymbol{x} \in \mathbb{R}^n$ the input vector and $y : \mathbb{R}^n \to \mathbb{R}$ the output of the network given by a scalar function of the input vector. Then, the relation between the output and the input layers can be expressed as:

$$y = \sum_{j=1}^{M} w_{mj} \varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|\right) \tag{2.2}$$

In Equation (2.2), M is the number of the neurons in the hidden layer, c_j is the center vector for neuron j and w_{mj} the weight of neuron j in the linear output neuron. By including the bias $\left(w_{k0}\right)$ in Equation (2.2), the formula can be expressed as follows:

$$y_{rbf} = \sum_{j=1}^{M} w_{mj} F_j\left(X\right) + w_{k0}, \tag{2.3}$$

where $F_j\left(X\right) = \varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_j\right\|\right)$.

Generally, the radial basis function is taken to be Gaussian:

$$F_j\left(X\right) = \exp\left(-\beta_j \left\|\boldsymbol{x} - c_j\right\|^2\right) \tag{2.4}$$

The parameter β_j is given by: $\beta_j = \dfrac{1}{2\sigma_j^2}$, σ denotes the width of the basis function.

As shown in **Figure 1**, each input layer corresponds to the input vector space. The hidden layer processes the spatial data through the use of radial basis functions that is centered on a spot. The last is the linear output layer which is the summation of the value obtained from the hidden layer and multiplied by a weight related to the neuron.

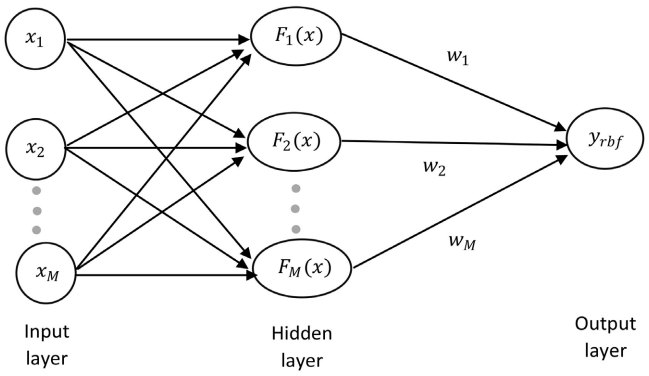

Figure 1. The architecture of RBF networks.

2.3. Random Forest

Random forest is a commonly-used machine learning algorithm based on the idea of constructing multiple decision trees. The use of multiple decision trees and random feature subsets allows the model to capture both linear and nonlinear relationships. Each decision tree works on a different sample and takes their majority vote for classification and average in case of regression [18].

Let consider Y_i the continuous response and $\boldsymbol{X}_i = \left(\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \cdots, \boldsymbol{x}_{im}\right)$ as input space vector. m defined the number of inputs and i the partitions of the feature space. The training set is expressed as:

$$D_n = \left\{\left(X_1, Y_1\right), \left(X_2, Y_2\right), \cdots, \left(X_m, Y_m\right)\right\}, X \in \mathbb{R}^m, Y \in \mathbb{R} \tag{2.5}$$

During the training process, the RF splits the input data at each node so that the parameters of the split functions can be improved to fit the D_n set. From the first step, the algorithm generates a number of decision trees. The decision tree has to determine the best split among all variables. This splitting process begins at the root and proceeds through each node, with each node applying its own split function to the new input X. This is done recursively until a terminal node (also known as tree leaves) is reached [19]. At the end of this process, a prediction function $f_b\left(\boldsymbol{x}\right)$ of each decision tree is constructed over D_n and calculated as follows:

$$f_b\left(\boldsymbol{x}\right) = \sum_{m=1}^{M} c_m I\left\{\boldsymbol{x}\right\}, \tag{2.6}$$

where $I_{\{x \in R_m\}}$ is the identity function that returns 1 if $\boldsymbol{x}$ is in the subset and 0 otherwise and c_m is the average of y.

The final prediction of the Random Forest model is the average of the predictions of all the decision trees in the forest. This can be written as:

$$y_{rf} = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b\left(\boldsymbol{x}\right), \tag{2.7}$$

where B is the number of decision trees in the forest and $f_b\left(\boldsymbol{x}\right)$ is the prediction of the m-th decision tree for the input space x.

In **Figure 2**, the input features are used to construct multiple decision trees, and the mean of all predicted decision trees is taken to obtain the prediction of the random forest.

2.4. Hybrid Model

The field of ML has rapidly evolved in recent years, with various models being developed and implemented to tackle different tasks. One such model that has gained popularity is the hybrid model. A hybrid model is a combination of ML models that are designed to solve a particular problem [20]. In this section, two distinct types of algorithms are used in the formulation of the hybrid model. These algorithms are RBFNN and RF.

RBFNN and RF prediction results are integrated as extra features within a Generalized Boosted Regression (gbm) technique to build a single strong model

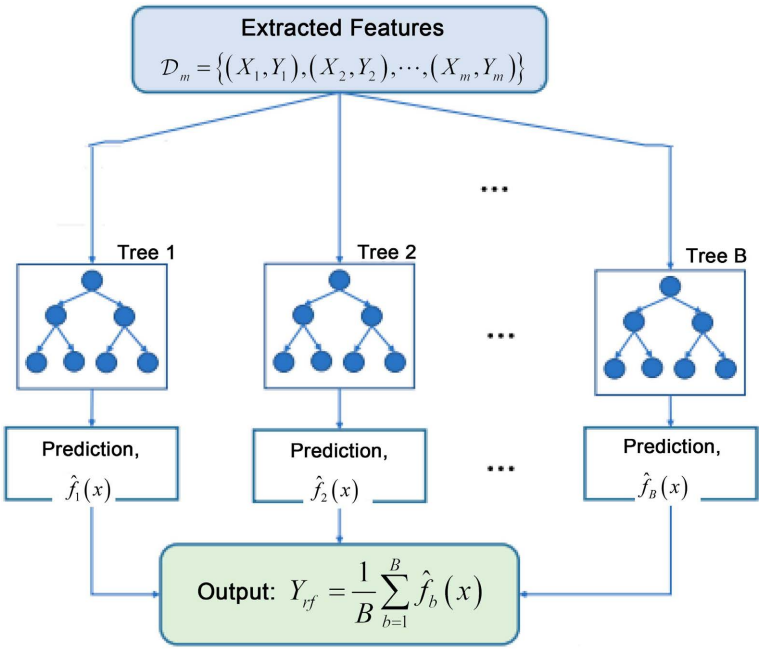

Figure 2. Random forest regression construction.

(strong learner). This combined gbm model can then be used for making predictions on new spatial data by following the same process of extracting RBFNN and RF predictions. One of the main benefits of this hybrid model is the ability to reduce errors. Errors arise when a model does not fit the training data set properly, resulting in poor performance. By incorporating the predictions from the two models as additional features, the gbm model can benefit from the different perspectives and strengths of the individual models, potentially leading to improved performance without losing the spatial dependence.

Let $\{\boldsymbol{z} : m = 1, 2, 3\}$ be a set of base learners:

$$\begin{aligned} \boldsymbol{z} &= \left\{ \boldsymbol{X}, y_{\boldsymbol{rbf}}, y_{\boldsymbol{rf}} \right\} \\ &= \left\{ \boldsymbol{X} + h_{RBFNN} + h_{RF} \right\}, \end{aligned} \tag{2.8}$$

where $\boldsymbol{X}$ is the matrix of the initial features, h_{RBFNN} and h_{RF} are the predicted values from RBFNN $\left(y_{\boldsymbol{rbf}}\right)$ and RF $\left(y_{\boldsymbol{rf}}\right)$ respectively.

Given a training set $\left\{(\boldsymbol{z}, y_i)\right\}_{i=1}^{n}$ of known $(y, \boldsymbol{z})$ values, the objective is to identify a prediction function $F(\boldsymbol{z})$ for predicting y^{HM} that maps $\boldsymbol{z}$ to y in such a way that, given the joint distribution of all $(y, \boldsymbol{z})$ values, the expected value of a certain loss function $\left(y; F(\boldsymbol{z})\right)$ is minimized.

Let L define the loss function:

$$L\left(y_i, F(\boldsymbol{z})\right), \tag{2.9}$$

Then the model is initialized

$$F_0(\boldsymbol{z}) = \arg\min_{F(\boldsymbol{z})} \sum_{i=1}^{N} L\left(y_i, F(\boldsymbol{z})\right) \tag{2.10}$$

The loss function is determined by the squared-error loss

$L\left(y_i, F\left(\boldsymbol{z}\right)\right) = \left(y_i - F\left(\boldsymbol{z}\right)\right)^2$.

For $m = 1$ to M do:

compute the pseudo residual by taking the loss function derivative with regard to the previous prediction $Fm-1$ *i.e.* $\left(F_0\left(\boldsymbol{z}\right)\right)$ and multiplied by −1.

$$\begin{aligned} r_{im} &= -\left[\frac{\partial L\left(y_i, F\left(\boldsymbol{z}_i\right)\right)}{\partial F\left(\boldsymbol{z}_i\right)}\right]_{F(\boldsymbol{z})=F_{m-1}(\boldsymbol{z})} \\ &= -\frac{\partial\left(y_i - F_{m-1}\right)^2}{\partial F_{m-1}} \\ &= 2\left(y_i - F_{m-1}\right) \\ &= \left(y_i - F_{m-1}\right) \end{aligned} \tag{2.11}$$

where m is the number of iteration.

Every iteration of the residuals, the model $F\left(\boldsymbol{z}\right)$ (also called week prediction model) is updated:

$$F_m\left(\boldsymbol{z}\right) = F_0\left(\boldsymbol{z}\right) + F_1\left(\boldsymbol{z}\right) + \cdots + F_M\left(\boldsymbol{z}\right) \tag{2.12}$$

Each model $\left(F_m\left(\boldsymbol{z}\right)\right)$ is constructed based on the residuals and is trained to minimize the remaining error. This process iterate until a low learning error is reached.

The weak models are combined to create a strong predictive model that captures the complex relationships between the predictors and the response. The final model is:

$$y^{HM} = \sum_{m=1}^{M} \alpha_m F_m\left(\boldsymbol{z}\right), \tag{2.13}$$

where $F_m\left(\boldsymbol{z}\right)$ corresponds to a weak prediction model and $0 \le \alpha_m \le 1$ stands for the weight of the weak model.

As we can see in **Figure 3**, the predicted values for each algorithms (h_{RBFNN} and h_{RF}) and the initial features are combined in the gbm function to obtain the hybrid model.

2.5. Cross-Validation

Validating the stability of a model is always necessary in machine learning. Cross-validation is a data re-sampling technique used to evaluate the true prediction error of models and tune model parameters to avoid overfitting [21]. In this case, we use the *k*-fold cross-validation. This technique splits the data into *k* subsets or folds, and the model is trained on $k - 1$ of these folds, while the remaining fold is used for testing. This process is repeated *k* times, with each fold being used once for testing, and the other k-1 folds used for training [22]. The idea is to assess the stability of each model by comparing the performance across different folds. It is expressed as:

$$\mathrm{CV}\left(k\right) = \frac{1}{k}\sum \mathcal{L}\left(y_i, \hat{f}_{-k}\left(x_i\right)\right), \tag{2.14}$$

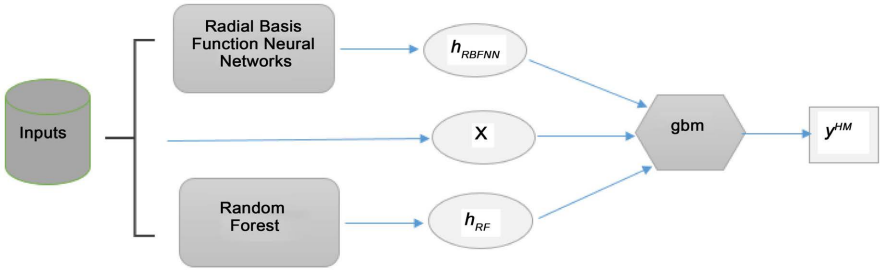

Figure 3. Hybrid model.

where $\mathrm{CV}(k)$ is the cross-validation estimate of the true error rate or performance metric, $\mathcal{L}$ is the loss or performance metric for the i^{th} fold, $\hat{f}_{-k}(x_i)$ is the model that was trained and k is the number of folds.

2.6. Spatial Autocorrelation

Spatial autocorrelation is a concept in statistics that refers to the degree to which nearby locations are similar to one another. In other words, it is a measure of the degree (that ranges from −1 to +1) to which the values of a variable in one location are similar to the values of the same variable in nearby locations. It can be positive or negative. Positive spatial autocorrelation occurs when nearby locations have similar values for a variable, while negative spatial auto-correlation occurs when nearby locations have dissimilar values for a variable. When there is no spatial auto-correlation, nearby locations have values for a variable that are unrelated to one another. In this study, we used Global Moran's I that is used to analyze the global spatial autocorrelation and Local Moran's I that evaluates the individual features and compares them to their neighbors and looks for clustering.

2.7. Models Evaluation

To evaluate the performance of the models, three criteria are used: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and R-squared (R^2), which are defined by

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{2.15}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{2.16}$$

$$\begin{aligned} R^2 &= 1 - \frac{\mathrm{sumsquaredresiduals}(\mathrm{SSR})}{\mathrm{totalsumofsquares}(\mathrm{SST})} \\ &= 1 - \frac{\sum\left(y_i - \hat{y}_i\right)^2}{\sum\left(y_i - \bar{y}\right)^2} \end{aligned} \tag{2.17}$$

3. Application

3.1. Dataset and Tools

A public spatial dataset is used in this study. The proposed HM model is applied

to California housing prices spatial data from Kaggle. This spatial dataset was originally published by Dr. Pace and Dr. Ronald Barry to build spatial auto-regressive models on 1990 California Census data. It contains information on the demographics (income, population, households) of the districts, as well as their location (latitude, longitude), and a general description of each district's homes (number of rooms, number of bedrooms, house value, ocean proximity). The dataset contains a total of 20.640 observations of housing prices with 10 features. Each observation consists of a single block in California. The attributes of the dataset are described in this **Table 1** and shows the data sample. **Figure 4** shows the distribution of each of the variables. The histograms and bar graph provide further details about the distribution of each feature. The dataset reveals that some features are skewed to the right, with Median House Value peaking on the far right. **Figure 5** displays the median home value dispersion across California by population and geographical area. We can observe that, on average the houses nearest to the ocean tend to have higher median house values. Typically, homes along the water cost more than homes inland. Therefore, it becomes sense to take the spatial data into consideration when predicting the price of household.

We used the open-source statistical programming language R. The algorithm implementations of the following packages have been used: caret (Short for Classification and Regression Training), random Forest (Classification and Regression with Random Forest), RSNNS (R Stuttgart Neural Network Simulator), gbm (Generalized Boosted Regression Modeling), spdep (Spatial Dependence) and tmap (Thematic Map Visualization).

3.2. Preprocessing

Real world data frequently possesses undesirable characteristics like inconsistent formats, missing values, unreadable formats etc. The spatial dataset contains 10 columns and 20,640 rows for different census tracts in California. This raw data needs to be processed in order for a machine learning algorithm to understand it and use it later for processing. This phase is known as preprocessing.

Modern ML techniques do well without feature selection, as models learn to identify useless features and focus on others [23] [24]. In this study, all features are included in the training process. The hybrid model (gbm method) is capable of evaluating the importance of features based on their contribution to the predictive performance of the model [25]. The target variable is the median house value that ranges from $14,999 to $500,001. The features are longitude, latitude, housing median age, total rooms, total bedrooms, population, households, median income and ocean proximity. Only ocean proximity is a categorical variable that has been transformed into numerical values. The variable total bedroom contains missing values that are imputed with a median value. Now, we can visualize dataset.

3.2.1. Normalization

Data normalization is used to organize data in a structured way so that it can be

Table 1. California housing prices description.

Variables	Descriptions
longitude	Longitude of the location of the house.
latitude	Latitude of the location of the house
median_income	Median age of the houses in the location
total_rooms	Total number of rooms in the houses in the location
total_bedrooms	Total number of bedrooms in the houses in the location
population	Total population in the location
households	Total number of households in the location
median_income	Median income of the households in the location
median_house_value	Median house value in the location
ocean_proximity	Proximity of the location to the ocean

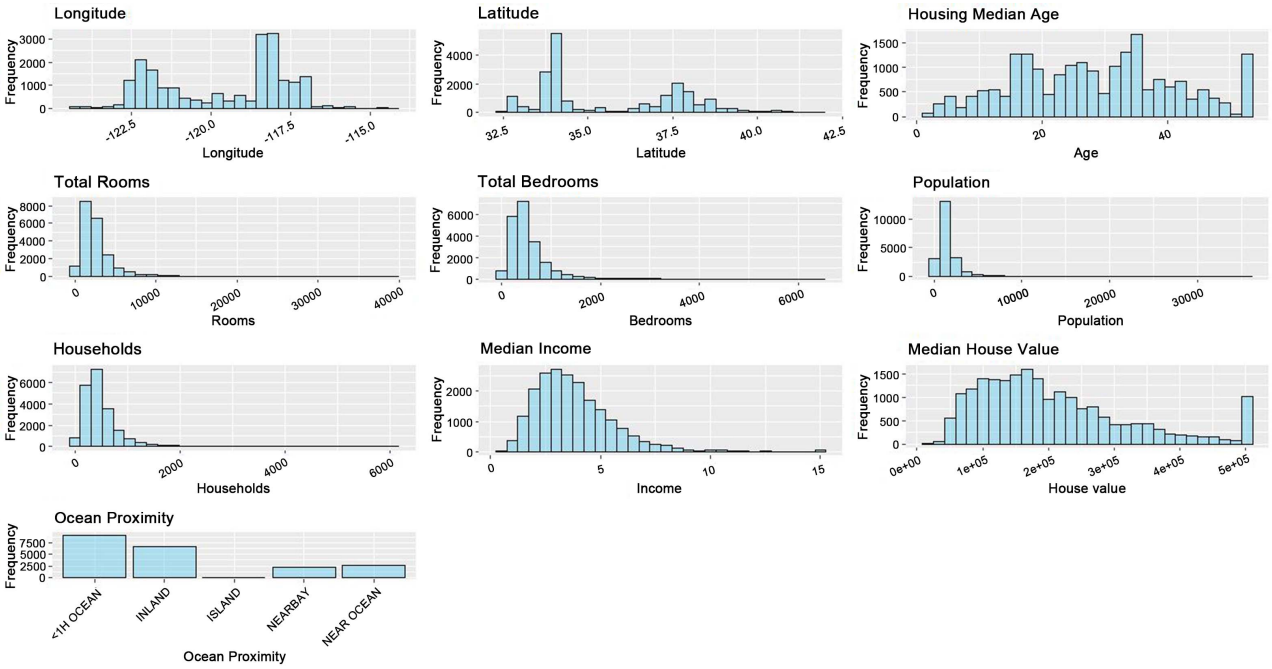

Figure 4. Distribution of variables.

on a similar scale. This process improves the performance and training stability of the model. Mathematically, it is given as follows:

$$X_n = \frac{X - X_{\min}}{X_{\max} - X_{\min}}, \tag{3.1}$$

where X is the normalized value, $X_{\min}$ is the minimum value of the feature and $X_{\max}$ is the maximum value of the feature.

3.2.2. Training and Testing Process

The training and testing process is a crucial step in the development of machine learning models. Over 20,640 observations, all the spatial data was divided into

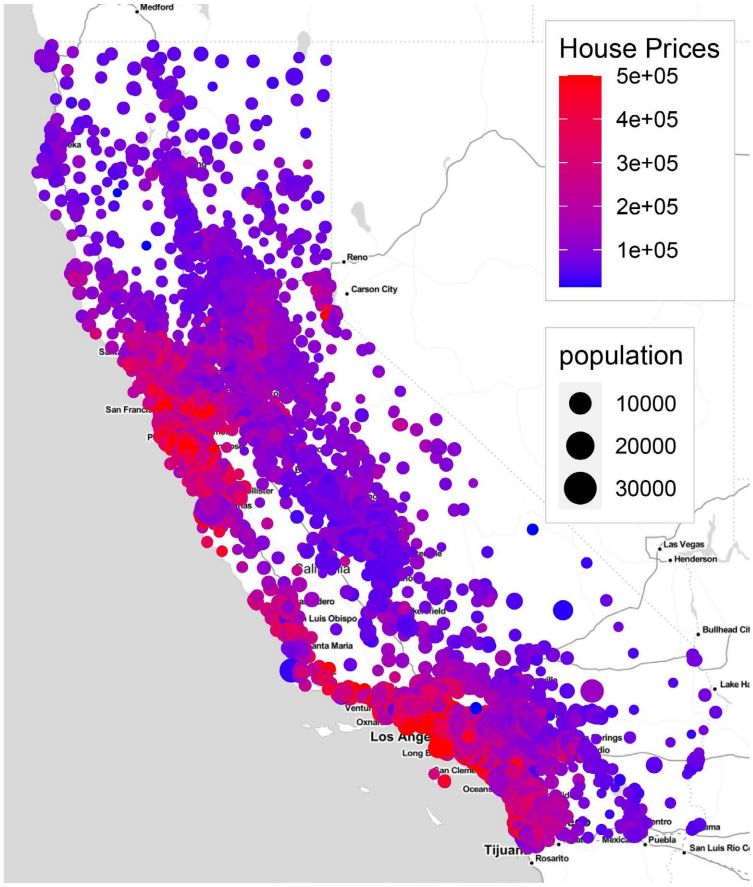

Figure 5. Distribution of house prices across the population in California.

two parts. The first part, the data is divided into 70% as training data. The second part, 30% as testing data. The former was used to build the previously mentioned models, while the latter was used to validate the models as showed in Table 2.

Table 2. The split description.

Original size	Training set	Testing set
20,640	14,447	6193

4. Experimental Results

Since it is necessary to quantify the results, in this step, the findings are extracted from the fit statement, which contains a list of stored values for each model. The RBFNN and RF algorithms are isolated and independent of each other during the training process. Next, we implemented a hybrid model to enhance the novelty and effectiveness of the work.

4.1. Results of Machine Learning Models

The training stage involves cross-validations and hyper-parameters adjustment to better connect spatial information. The rbf () function in the RSNNS package

is used to build a RBFFN model. The grid parameter values was defined for the number of hidden neurons (size = c(5, 10, 15, 20)) and maximum number of iterations (maxit) of 1000. For RF the number of variables randomly sampled (mtry) parameter was used with *c*(2, 5, 10) values and the number of trees to grow (*ntree* = 100). Then, the hybrid model was constructed using the predicted values from RBFNN and RF as additional features for training the gbm model. The number of iterations (n.trees), the maximum depth of each tree (interaction. depth) and the learning rate (shrinkage) were defined as 100, 2 and 0.8 respectively for the HM model. The distribution (for HM) was specified as gaussian.

Table 3 shows the results of all models. The proposed HM model outperforms all individual models. It has extremely low values for both RMSE and MAE, indicating that the predicted values align closely with the actual values. The high *R*2 value of 0.9991087 suggests that this model explains approximately 99.91% of the variance in the spatial data, indicating a very strong fit. The proposed model is significant compared to all other models used because we have combined the best performing RBFNN (*R*2 = 74.22%) and RF (*R*2 = 82.26%). On the other hand, while the RBFNN and RF models perform reasonably well, the RF model generally outperforms the RBFNN model in terms of RMSE, MAE, and *R*2 scores.

4.2. K-Fold Cross-Validation Results

To demonstrate the significance of our proposed approach, we performed a 5-fold cross-validation using all models on the used spatial dataset. Table 4 shows the performance metrics in terms of RMSE on the models on a cross-validation process, along with the corresponding test error. Analyzing the spatial data, it becomes evident that the HM model consistently achieves the lowest RMSE values across all folds and exhibits the lowest test error (which is the average RMSE across all folds within a model). These findings strongly suggest that the HM model is the most accurate model for the given task. The RF model also demonstrates reasonably good performance, showcasing lower RMSE values compared to the RBFNN model. However, the RBFNN model exhibits slightly higher RMSE values, indicating a comparatively lesser accuracy in predicting the target values.

4.3. Spatial Autocorrelation Evaluation

In this section, we measure the spatial autocorrelation Moran's I in the residuals to validate the HM model. In the case, we created a spatial weights matrix with the knn2nb function based on the k-nearest neighbors (k = 3) of the location coordinates (latitude and longitude) of the observations. In Table 5, the global Moran's I statistic of residuals results are describe for each models.

The Global Moran's I value represents the spatial autocorrelation, specifically the degree of spatial clustering or similarity, observed in the residuals. The RBFNN model exhibits a Moran's I value of 0.42, indicating a relatively strong

Table 3. Results obtained from all models on the testing set.

Models	Selected values	RMSE	MAE	R2
RBFNN	size = 20 maxit = 1000	0.1281168	0.0860183	0.7421976
RF	mtry = 5 ntree = 100	0.09955914	0.06559628	0.8225986
HM	n.trees = 100 interaction.depth = 2 shrinkage = 0.8	0.007307837	0.005467172	0.9991087

Table 4. 5-fold cross validation results.

Models-Outer Folds-RMSE						
Models	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test Error
RBFNN	0.1342258	0.1309185	0.1287859	0.1300938	0.1323121	0.131267
RF	0.09943762	0.10357063	0.10007843	0.10119896	0.10344938	0.101547
HM	0.03306925	0.03533340	0.03441766	0.03331788	0.03350357	0.033928

Table 5. Degree of global spatial autocorrelation.

	RBFNN	RF	HM
Moran's I of residuals	0.42	0.21	0.12

positive spatial autocorrelation pattern in the residuals. The RF model shows a lower Moran's I value of 0.21, suggesting a relatively weaker spatial autocorrelation compared to the RBFNN model. In contrast, the HM model demonstrates the lowest Moran's I value of 0.12, indicating the lowest spatial autocorrelation among the two other models. In summary, the comparison of the Moran's I values of the models shows the reduction of the spatial autocorrelation in the residuals. It clearly indicates that the performance of a model influences the way in which the model captures the spatial autocorrelation in the residuals.

The study also looked for the spatial association around each individual location. **Figures 6-8** show the maps of local moran's statistic (Ii). A positive Ii value implies that the unit is surrounded by units with similar values. It appears from the maps, all models used in this study captured the neighborhood, which means that they take into account the dependencies in the residuals according to the blue points in the figures. The hybrid model figure shows the li has a small range (−7.32 to 14.55) compared to others. While the model (HM) demonstrates high performance, it does not fully account for the spatial structure in the data, despite containing some dependencies.

This suggests that the models successfully captured the spatial property, as evidenced by the robust performance evaluation using RMSE for the HM model

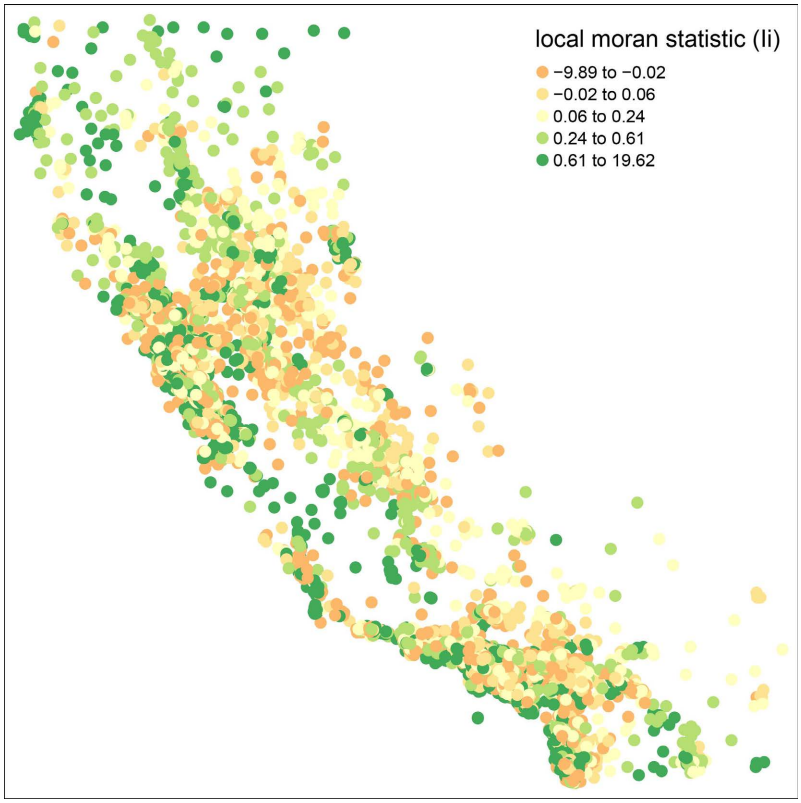

Figure 6. RBFNN model.

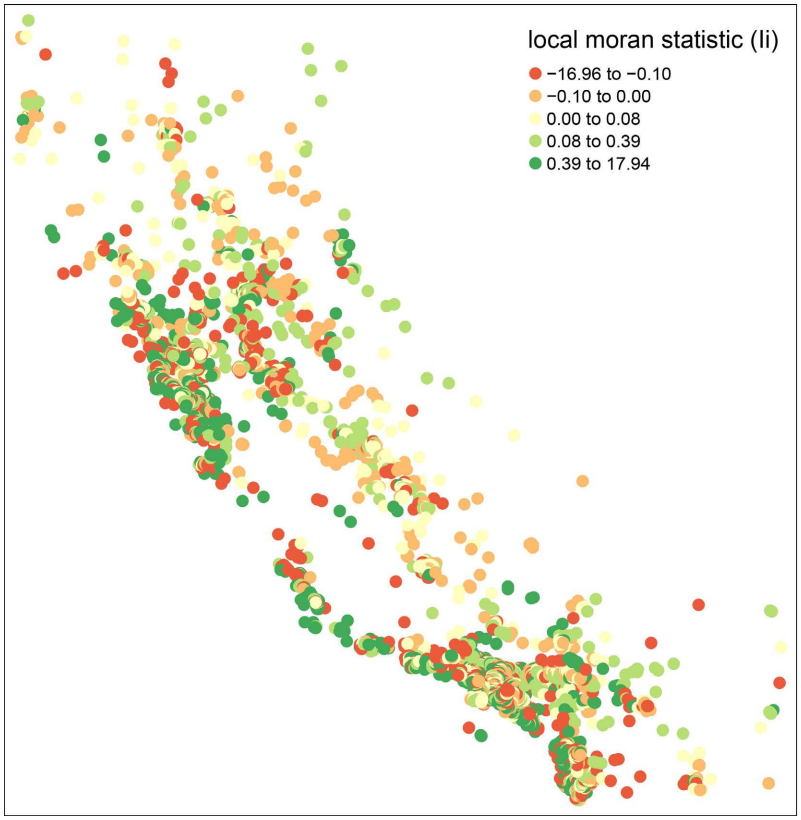

Figure 7. RF model.

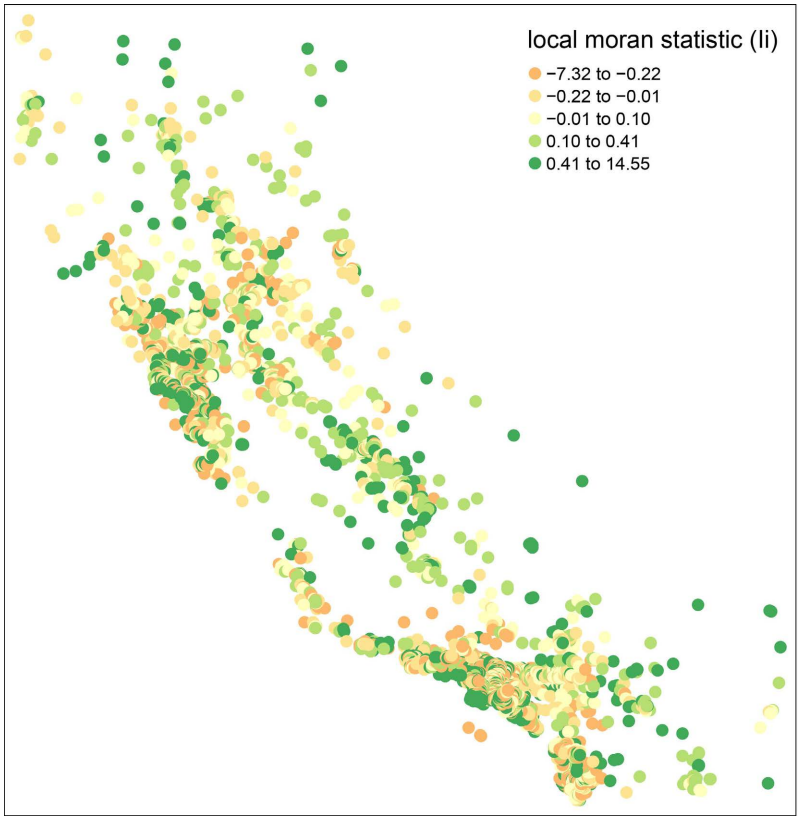

Figure 8. HM model.

and moderate performance for the other two models. These results align with our expectations, demonstrating that a specific ML model can effectively process spatial information without incorporating explicit spatial features during the learning process. Additionally, it highlights the model's ability to capture spatial dependencies and improve accuracy.

5. Conclusion

This study proposed a hybrid approach for spatial dependence detection using machine learning (ML) without incorporating any spatial features in the learning process. The hybrid model (HM) was developed by combining two models, Radial basis function neural networks (RBFNN) and Random forest (RF) to achieve high accuracy and efficiency. Both models (RBFNN and RF) perform well and can detect the dependence because of their ensemble architecture. Combining them, they further achieved 99.91% of performance. This significant performance improvement observed can be attributed to the utilization of the boosting technique (Generalized Boosted Regression), which identifies errors for each model. In conclusion, the individual models were able to capture a greater amount of spatial information, including spatial dependencies as measured by global Moran and local Moran, despite having lower R2 values compared to the HM model. The HM model, on the other hand, exhibited a high R2 but showed weak positive spatial dependence.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Fazal, S. (2008) GIS Basics. New Age International, New Delhi.

[2] Bernhardsen, T. (2002) Geographic Information Systems: An Introduction. John Wiley & Sons, Hoboken.

[3] Guptill, S.C. and Morrison, J.L. (2013) Elements of Spatial Data Quality. Elsevier, Amsterdam.

[4] Sneha, N.S. and Pushpa (2014) Clustering and Noise Detection for Geographic Knowledge Discovery. *International Journal of Advances in Engineering & Technology*, **7**, 845-855.

[5] Sarker, I.H. (2021) Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, **2**, Article No. 160. https://doi.org/10.1007/s42979-021-00592-x

[6] Gilardi, N. and Bengio, S. (2000) Local Machine Learning Models for Spatial Data Analysis. *Journal of Geographic Information and Decision Analysis*, **4**, 11-28.

[7] Miller, H.J. (2004) Tobler's First Law and Spatial Analysis. *Annals of the Association of American Geographers*, **94**, 284-289. https://doi.org/10.1111/j.1467-8306.2004.09402005.x

[8] Andersson, M. and Gråsjö, U. (2009) Spatial Dependence and the Representation of Space in Empirical Models. *The Annals of Regional Science*, **43**, 159-180. https://doi.org/10.1007/s00168-008-0211-5

[9] Shekhar, S., Jiang, Z., Ali, R.Y., Eftelioglu, E., Tang, X., Gunturi, V.M.V. and Zhou, X. (2015) Spatiotemporal Data Mining: A Computational Perspective. *ISPRS International Journal of Geo-Information*, **4**, 2306-2338. https://doi.org/10.3390/ijgi4042306

[10] Pereira, G.W., Valente, D.S.M., de Queiroz, D.M., de Freitas Coelho, A.L., Costa, M.M. and Grift, T. (2022) *Smart-Map*: An Open-Source QGIS Plugin for Digital Mapping Using Machine Learning Techniques and Ordinary Kriging. *Agronomy*, **12**, Article No. 1350. https://doi.org/10.3390/agronomy12061350

[11] Wang, Z., Shi, W., Zhou, W., Li, X. and Yue, T. (2020) Comparison of Additive and Isometric Log-Ratio Transformations Combined with Machine Learning and Regression Kriging Models for Mapping Soil Particle Size Fractions. *Geoderma*, **365**, Article ID: 114214. https://doi.org/10.1016/j.geoderma.2020.114214

[12] Takoutsing, B. and Heuvelink, G.B.M. (2022) Comparing the Prediction Performance, Uncertainty Quantification and Extrapolation Potential of Regression Kriging and Random Forest While Accounting for Soil Measurement Errors. *Geoderma*, **428**, Article ID: 116192. https://doi.org/10.1016/j.geoderma.2022.116192

[13] Hengl, T., Nussbaum, M., Wright, M.N., Heuvelink, G.B.M. and Gräler, B. (2018) Random Forest as a Generic Framework for Predictive Modeling of Spatial and Spatio-Temporal Variables. *PeerJ*, **6**, e5518. https://doi.org/10.7717/peerj.5518

[14] Liu, X., Kounadi, O. and Zurita-Milla, R. (2022) Incorporating Spatial Autocorrelation in Machine Learning Models Using Spatial Lag and Eigenvector Spatial Filtering Features. *ISPRS International Journal of Geo-Information*, **11**, Article No. 242. https://doi.org/10.3390/ijgi11040242

[15] Dubé, J. and Legros, D. (2013) A Spatio-Temporal Measure of Spatial Dependence:

An Example Using Real Estate Data. *Papers in Regional Science*, **92**, 19-30.

[16] Kumar, S., Pai, P.S. and Rao, B.R.S. (2012) Radial-Basis-Function-Network-Based Prediction of Performance and Emission Characteristics in a Bio Diesel Engine Run on WCO Ester. *Advances in Artificial Intelligence*, **2012**, Article ID: 610487. https://doi.org/10.1155/2012/610487

[17] Montazer, G.A., Giveki, D., Karami, M. and Rastegar, H. (2018) Radial Basis Function Neural Networks: A Review. *Computer Reviews Journal*, **1**, 52-74.

[18] Schonlau, M. and Zou, R.Y. (2020) The Random Forest Algorithm for Statistical Learning. *The Stata Journal*, **20**, 3-29. https://doi.org/10.1177/1536867X20909688

[19] Li, Y., Zou, C., Berecibar, M., Nanini-Maury, E., Chan, J.C.-W., Van den Bossche, P., Van Mierlo, J. and Omar, N. (2018) Random Forest Regression for Online Capacity Estimation of Lithium-Ion Batteries. *Applied Energy*, **232**, 197-210. https://doi.org/10.1016/j.apenergy.2018.09.182

[20] Al Mamun, A., Sohel, M., Mohammad, N., Sunny, M.S.H., Dipta, D.R. and Hossain, E. (2020) A Comprehensive Review of the Load Forecasting Techniques Using Single and Hybrid Predictive Models. *IEEE Access*, **8**, 134911-134939. https://doi.org/10.1109/ACCESS.2020.3010702

[21] Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L. and Ridella, S. (2012) The 'K' in K-Fold Cross Validation. 20*th European Symposium on Artificial Neural Networks*, *Computational Intelligence and Machine Learning* (*ESANN*), Bruges, 25 to 27 April 2012, 441-446.

[22] Berrar, D. (2019) Cross-Validation. In: Ranganathan, S., Gribskov, M., Nakai, K. and Christian Schönbach, C., Eds., *Reference Module in Life Sciences Encyclopedia of Bioinformatics and Computational Biology*, Vol. 1, Elsevier, Amsterdam, 542-545. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

[23] Laradji, I.H., Alshayeb, M. and Ghouti, L. (2015) Software Defect Prediction Using Ensemble Learning on Selected Features. *Information and Software Technology*, **58**, 388-402. https://doi.org/10.1016/j.infsof.2014.07.005

[24] Xu, Z., Huang, G., Weinberger, K.Q. and Zheng, A.X. (2014) Gradient Boosted Feature Selection. *Proceedings of the* 20*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, 24-27 August 2014, 522-531.

[25] Friedman, J.H. (2001) Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, **29**, 1189-1232. https://doi.org/10.1214/aos/1013203451