# LVRF: A Latent Variable Based Approach for Exploring Geographic Datasets

Deng, Liangdong; Mahara, Arpan; Adjouadi, Malek; and Rishe, Naphtali

**Abstract:** ***Geographic datasets are usually accompanied by spatial non-stationarity – a phenomenon that the relationship between features varies across space. Naturally, non-stationarity can be interpreted as the underlying rule that decides how data are generated and alters over space. Therefore, traditional machine learning algorithms are not suitable for handling non-stationary geographic datasets, as they only render a single global model. To solve this problem, researchers often adopt the multiple-local-model approach, which uses different models to account for different sub-regions of space. This approach has been proven efficient but not optimal, as it is inherently difficult to decide the size of sub-regions. Additionally, the fact that local models are only trained on a subset of data also limits their potential. This paper proposes an entirely different strategy that interprets non-stationarity as a lack of data and addresses it by introducing latent variables to the original dataset. Backpropagation is then used to find the best values for these latent variables. Experiments show that this method is at least as efficient as multiple-local-model-based approaches and has even greater potential.***

**Index Terms**: ***Back-propagation, Geographically Weighted Regression (GWR), Latent Variable, Machine Learning Algorithm, Non-stationary, Random Forest***

Manuscript received April 10, 2023.

This material is based in part upon work supported by the National Science Foundation under Grant Nos. MRI20 CNS-2018611, MRI CNS-1920182, by FDEP Grant C-2104, and DHS Grant E2055778.

Liangdong Deng, Arpan Mahara, and Naphtali Rishe are with the School of Computing and Information Sciences, Florida Internation University, USA. Correspondence email is liadeng@cs.fiu.edu.

Malek Adjouadi is with the Department Electrical and Computer Engineering, Florida Internation University, USA.

## 1. Introduction

Geographic data is defined as information that is implicitly or explicitly associated with a location on the surface of the Earth [1]. With advancements in remote sensing technologies and the widespread use of GPS-enabled devices, the number of available physical and human geography datasets has vastly increased in recent years [2]. These data are studied and utilized for social good, such as mitigating damages caused by natural disasters [3], discovering mineral resources [4], preventing crimes [5], improving traffic conditions [6], and many other scenarios.

However, when dealing with geographic datasets, researchers find that many traditional machine learning algorithms do not perform very well due to the presence of non-stationarity. In such data, the relationship between features does not necessarily remain the same everywhere, meaning the underlying model that governs the data changes over space. To address this issue, a natural solution is to replace the global model with many local models. Each local model is only responsible for describing a much smaller region within which the data is supposed to be relatively stationary. Most studies that have taken this approach (such as [7], [8] and [9]) have observed significantly better results compared to traditional algorithms, which are not specifically designed to handle non-stationarity.

These multiple-local-model based approaches all face similar challenges. First, the dataset used to train local models is only a subset of all available data. Previous research has shown that the accuracy of a model is strongly correlated with the amount of data used to train this model. There can be a

significant decrease in model performance if the training data size drops below a certain threshold [10]. Second, determining the size of sub-regions to which local models correspond is difficult. A larger size means more data can be used to train local models, but the region is more likely to exhibit non-stationary. Conversely, a smaller size implies the opposite. As a result, compromise is always necessary.

Our insight is that the source of non-stationarity can be explained as a lack of data, i.e., some dimensions of the data are not being collected. For example, a crime dataset could exhibit strong non-stationarity, as crime patterns in New York could be fundamentally different from those in Washington DC. Even within New York, it is hard to imagine that Brooklyn shares the same crime pattern as Manhattan. Ultimately, these differences are caused by various factors such as household income, population composition, culture, and the number of police officers per capita, among others. If one were able to collect data on every single aspect of an area, the dataset would ultimately become stationary. This theory is also in accordance with the fact that non-stationarity is quite often observed in human geography datasets but rarely found in physical geography data. Since physical geography data – which is generated by Earth's natural processes - has fewer determining factors and is usually simpler to collect, it is less prone to non-stationarity. In contrast, human geography data focuses on human activities and is much more complex. Even seemingly simple datasets can have countless deciding factors that are impossible to collect comprehensively. For example, house sale price data generally includes features of the house itself and its nearby areas, but other factors - such as school, traffic, population, and crime - are usually not included, even though they are important and would certainly affect the pricing model. The lack of these data would then be observed as non-stationarity in the dataset and would impact the final model in some way.

Based on this insight, we propose an entirely different strategy that addresses non-stationarity by introducing latent variables to the original dataset. These latent variables would account for all the missing factors that not collected by the original dataset but observable as non-stationarity. Theoretically, assuming we have unlimited calculating power, the optimal values of the latent variables could be easily found through a brute-force search of the entire vector space. However, this solution is obviously impossible due to the tremendous size of the vector space. Thus, inspired by neural networks, we use a back-propagation algorithm to find the optimal values of the latent variable. Experiments demonstrate that this new approach can build models as accurate as the state-of-the-art algorithms while offering the potential for further improvement.

## 2. Background and Study Area

### 2.1 Background

The first renowned method for exploring spatial non-stationarity, known as Geographically Weighted Regression (GWR), was proposed by Brunsdon, Fotheringham, and Charlton in 1996 [7]. The "main characteristic of GWR is that it allows regression coefficients to vary across space, and so the values of the parameters can vary between locations" [11]. The motivation for inventing GWR was that "a single global model cannot explain the relationship between some sets of variables" [7]. To address non-stationarity, GWR allows relationships between features and labels to differ across spaces. The basic idea of how GWR works is to learn a regression equation for every feature in the dataset, during which dependent and explanatory components are accounted for by examining neighboring data points. The neighbors contribute differently to this process according to their distance, which is why it is called a "weighted" regression. The closer a data point is, the more weight it is assigned. This design complies with Tobler's first law of geography, "everything is related to everything else, but near things are more related than distant things" [12]. Later, in 2002, Brunsdon further improved this algorithm to Semiparametric GWR (SGWR) [13], which allows some features to have fixed regression equations across space, while others can still be variable.

Due to the success of GWR, many later

studies followed this multiple-local-model design. One example is Multiscale GWR (MGWR), which was introduced in 2017 by Fotheringham, Yang, and Kang. This method "is similar in intent to Bayesian nonseparable spatially varying coefficients (SVC) models, although potentially providing a more flexible and scalable framework in which to examine multiscale processes" [9]. It improves upon GWR in a way that not only adapts to datasets on different levels of non-stationarity but also provides the necessary information to evaluate the scales of different processes. The latest research using this approach is Geographical Random Forest (GRF), proposed by Stefanos, Tais, et al. in 2019. It adopts Random Forests [14] as the base algorithm to create local models. The principle idea of this method is the "disaggregation of RF into geographical space in the form of local sub-models" [8], which is basically another version of the multiple-local-model approach.

In conclusion, all these methods are directly or indirectly based on the multiple-local-model approach and consequently suffer from the same problems mentioned in the previous section. In this work we propose a completely different approach with the goal of better understanding and accounting for the intrinsic nature of non-stationarity.

### 2.2 Study Area

We selected housing sales data from King County, US as the target study area (obtained from [15]). The dataset contains 21,613 records, with each record being a real estate transaction that occurred between May 2014 and May 2015, a period during which the housing market remained relatively stable in King County.

In this dataset, there are 20 features related to the house's location (latitude, longitude, zip code), its basic information (size, number of stories and rooms, garage, air conditioning), and transaction-related information (sale date and price). Some of the features have missing values. This is not a problem for our algorithm, which is based on the Random Forests algorithm and can handle missing values. However,

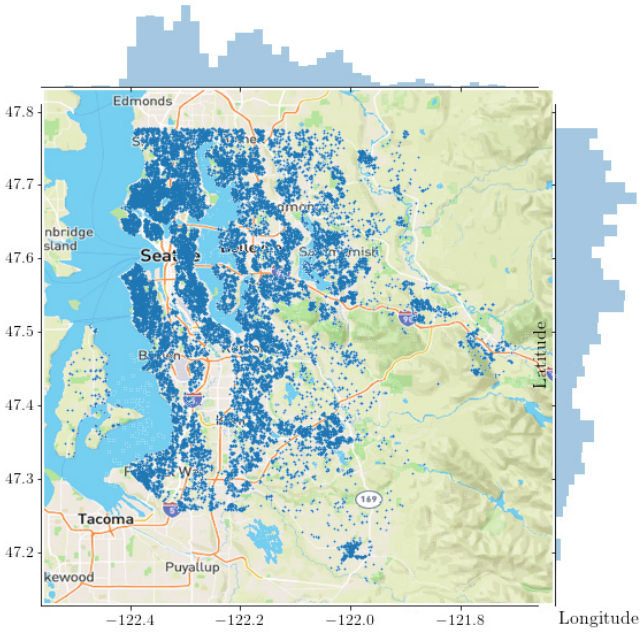


Figure 1: Distribution of the King County housing data.

some other algorithms we use for performance comparison are incapable of doing this. Therefore, during the data preparation stage, we fill in the missing values with the average value of that column.

The goal of this dataset is to build a predictive model that can estimate house sale prices, given the house's location and some of its basic information. It is a well-researched topic that has been studied for a long time. However, even state-of-the-art algorithms in this area still have ample room for improvement due to the complicated nature of this task. Additionally, it is a very typical human geography dataset in which data availability varies depending on the amount of human activity. Figure 1 shows the distribution of the dataset on the map. As depicted in the figure, the downtown area in Seattle is populated with data, with some areas left blank which are mostly parks or commercial zones. Rural regions have much less data scattered all over the place. The fact that this dataset is distributed extraordinarily unevenly across the space presents additional challenges when using the previously mentioned multiple-local-model approach, as local models which correspond to rural areas will have fewer train-

ing samples, leading to inaccurate results. In urban areas, overcrowded data points will only bring marginal improvement to models built for that area.

Another issue with this dataset is that the house sale price spans over a fairly large range with a long tail, as shown in figure 2, which is undesirable. To eliminate the tail, we convert Price to log(Price), which follows the normal distribution and is a much better target variable to deal with.

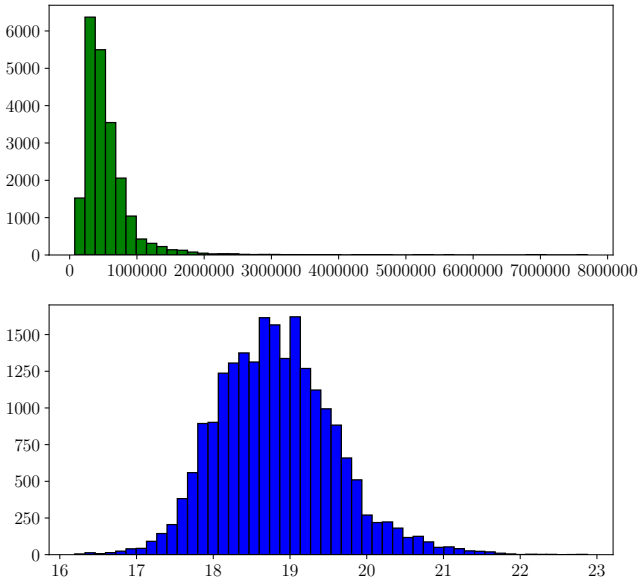


Figure 2: Distribution of Price vs. log(Price)

## 3. Latent Variable Random Forests

In this section, we provide a detailed description of the key designs of the Latent Variable Random Forests as follows.

### 3.1 Key Design of the Latent Variable

By introducing a new latent variable, we aim to use it to represent the hidden factors that cause non-stationarity. In our housing price model example, it would be a combination of various unknown factors that could affect how house prices should be modeled. For instance, the security level of a community obviously has an impact on house value. Although we don't have any information on which area is more secure and which is not, its influence on the sale price will be observable via non-stationarity. It is important to note that the target variable might be affected by multiple hidden factors such as security, traffic, nearby schools, and so on. But no matter how many hidden factors there are, they will influence the target variable together. It is impossible to know which factor has a larger impact. Fortunately, we don't need to care about that. Our primary focus is on how these hidden factors as a whole would affect the target variable we want to predict.

To better describe the problem, let $(f_1, f_2, ..., f_n)$ denote the features in the dataset and $t$ denote the target variable to be modeled and predicted. After adding a latent variable $lv$, the feature vector becomes $F(lv) = (f_1, f_2, ..., f_n, lv)$. Thus, the task is converted to finding the best $\vec{lv}$ that makes the model trained from $F(lv)$ (using a predetermined regular machine learning algorithm) achieve the highest accuracy.

The vector space $\vec{lv}$ is obviously unlimited. Thus we introduce a value range of $[0, 1]$ to $lv$ and define a minimum step interval of $0.01$. The reason why we limit the value range to $[0, 1]$ is that the value range of $lv$ actually doesn't play an important role in the final model. If $lv$ is multiplied by 2, the resulting model will still be the same. So, only the relative value matters and is what we should care about. Also, during the machine learning stage, all the features of the original dataset need to be standardized and normalized anyway, thus a standardized $lv$ will, in fact, benefit the entire procedure. For the minimum step, the smaller it is, the more fine-grained the final model would be. However, setting it too small will also considerably increase the calculation time and may not be worth the marginal return. So we recommend setting it to $0.01$ as a balance between speed and accuracy.

Theoretically, the value array of latent variable $\vec{lv}$ can be inferred by an exhaustive brute-force search of the entire vector space. The time complexity of doing so is as follows:

$$O(n) = (\frac{R}{S})^n * (T_{train} + T_{test}) \qquad (1)$$

where $n$ is the number of data points in the dataset, $R$ is the value range, $S$ is the step size,

$T_{train}$ and $T_{test}$ are the time needed for training and testing the model, respectively. Note that the value of $n$ is usually very large. Even for a very small dataset, $n$ will probably be greater than 1000. Thus, this brute-force method is completely impractical considering the amount of calculation needed.

### 3.2 Grid Based Latent Variable System

To solve the time complexity problem, we clearly need a smarter algorithm, for example, a heuristic search, which could greatly reduce the search space. But before that, let's examine the possibility of reducing the size of the potential vector space, which would greatly benefit the entire procedure even if a heuristic search is to be adopted.

Here we introduce a grid-based latent variable system. Let $(x_{min}, x_{max}, y_{min}, y_{max})$ denote the minimum bounding box that contains the entire dataset. A step size of $s$ will evenly divide the space into this many grids:

$$G(s) = \lceil \frac{x_{max} - x_{min}}{s} \rceil * \lceil \frac{y_{max} - y_{min}}{s} \rceil \quad (2)$$

For each intersection of the grid system, we assign an *Influence Center* (abbreviated as IC) to it. For a data point with a coordinate of $(x, y)$, we first determine which $grid$ it is located in. Then calculate its latent variable value from all the nearby ICs located at the four corners of $grid$. Here we use an inverse distance weighted method to combine the values from nearby ICs, in accordance with the idea that nearby ICs should have a stronger influence on the latent variable than remote ones. The detailed formula is as follows:

$$v(x, y) = \frac{\sum_{i=1}^{N} W(IC_i)V(IC_i)}{\sum_{i=1}^{N} W(IC_i)} \quad (3)$$

where $W(IC_i)$ is the weight for the $i$th influence center which equals the inverse of the Euclidean distance between the data point and the IC.

This design simulates how the hidden factors create non-stationarity in the dataset. No matter what hidden factors there are, as a general rule, they would affect nearby data points more than remote ones. Thus we simulate this procedure by introducing the concept of Influence Centers and making them impact nearby records in a similar way. Another benefit brought by this design is that now the search space is greatly reduced down to the number of ICs. Instead of finding the best values for all the records, we only need to optimize the values for ICs now, which is way less than the total number of records.

### 3.3 Random Forests as the Base Algorithm

Before proceeding, we still need to decide which base machine learning algorithm is to be used to train models. Here, our choice is the Random Forests [14] algorithm. As suggested in the name, Random Forests will create many randomly generated decision trees to perform the prediction task together. For classification tasks, the final result would be a majority vote of results from all the decision trees. For regression, this would be an average of all results. The core idea of RF is to create a bagging procedure where the variance of the model is decreased but the bias remains unchanged, thus generating a better result from sub-optimal models.

There are multiple reasons why we choose RF as our base algorithm. First, RF is based on decision trees which are naturally good at handling coordinates in geographic datasets. Then, Random Forests is among the top machine learning algorithms available and often shows exceedingly good results when handling spatial data, as proven by [16] and [17]. We will be able to inherit all of these advantages by using RF as the base algorithm.

### 3.4 Back Propagation

With a reduced search space, the time complexity is still massive as we are only replacing $(\frac{R}{S})^n$ in Formula 1 with $X^n$ ($X$ is the total number of influence centers) if a brute-force search is to be used. Thus we must find a way to further reduce the search space, i.e., a heuristic-search like method.

Here, inspired by the backpropagation algorithm in Neural Networks [18], we have de-

signed a backpropagation process to search for the best values for influence centers, as detailed in Algorithm $BackPropagation()$. In this function, a learning rate $\alpha$ is introduced, which determines how fast the backpropagation converges. A large value will cause $BackPropagation()$ to converge faster, but the generated result will be more likely to be coarse-grained and thus less than optimal. Conversely, a smaller value will converge slower but produce better results. Generally speaking, the best $\alpha$ value is recommended to be set to the smallest value within acceptable training time.

```
1  Function BackPropagation()
2      Initialize IC_Array
3      while IC_Array has not converged
        do
4          foreach IC in IC_Array do
5              foreach learn_rate in [α, -α]
                do
6                  IC_new = IC +
                    learn_rate
7                  if Trained model sees
                    improvement in
                    accuracy then
8                      IC = IC_new
9                  else
10                     continue
11                 end
12             end
13         end
14     end
15     return IC_Array
16 end
```

The *converge* condition in the *BackPropagation()* algorithm is a bit tricky. Ideally, if *IC_Array* remains the same after an iteration, the algorithm is considered converged as future iterations will produce the same results. However, this does not necessarily happen as *IC_Array* may always change slightly with pretty much the same results. So, we insert a process at the end of each iteration, which will evaluate the test accuracy under the current *IC_Array*. If the test accuracy does not improve for more than 5 iterations, we consider the algorithm converged and stop the backpropagation iteration. Although this extra calculation slows down the entire algorithm, it is worth the cost.

### 3.5 Prediction

The prediction process is relatively simple. After the *IC_Array* is returned by *BackPropagation()*, the final *Model* is trained from the original dataset plus the latent vector generated from *IC_Array*. When predicting an unknown observation, the latent variable is first calculated by using the inverse distance weighted method from Formula 3. Then, *Mode* is applied to get the final prediction result.

### 3.6 Assessment Measurements and Results

One thing that wasn't mentioned in the previous sections is that a proper assessment measurement must be chosen. This actually plays an important role in the algorithm, as the evaluation result generated by the measurement will be used to determine how the backpropagation process runs and guide it to generate a better result for each iteration. Some of the most commonly used measurements are [19]: mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). In our case, MAE is preferred as the other ones will penalize large errors and cause bias in our algorithm.

Now that the algorithm is complete, we have run LVRF on the King County housing dataset and achieved an MAE of 0.263. As a comparison, we also experimented with unmodified Random Forests on the same dataset and obtained a result of 0.289. This means that the learned latent variables were able to offset some of the non-stationarity and made it easier for the standard RF to generate a more accurate model. To compare with the others, we also evaluated the same dataset using two state-of-the-art algorithms, RFsp [20] and MGWR [9], which are specifically designed to handle geographic datasets and non-stationarity. The results for RFsp and MGWR were 0.261 and 0.272, respectively. These results suggest that the idea of using latent variables to capture hidden factors that cause non-stationarity is at

least as effective as the best results achieved using the multiple-local-model approach.

## 4. Conclusion

This paper presents LVRF, a machine-learning algorithm that can create predictive models for non-stationary geographic datasets. Unlike other algorithms, LVRF adopts a latent variable based approach, instead of the widely used multiple-local-model strategy. Experiments show that LVRF can build models as accurately as state-of-the-art algorithms while avoiding the common disadvantages of the multiple-local-model approach. First, LVRF establishes grid-based influence centers. The latent variable value of any data point is decided by the nearby influence centers using an inverse distance weighted method. Then it uses a backpropagation algorithm to train the values of the influence centers until they converge. To predict unknown observations, the data point's latent variable is calculated from the converged influence centers, and fed into the model with its other features.

The insight of LVRF is that the design of the influence center can mimic the hidden factors which affect nearby data points in different ways depending on the location. By learning these hidden factors with a backpropagation algorithm and then including them in the model creation stage, the impact brought by non-stationarity will be offset. This approach allows for a single global model to be used to describe the features plus the hidden factors.

It is also worth mentioning that, although Random Forests is selected as the base algorithm, LVRF is capable of using any other regular machine learning algorithm as the base algorithm. Doing so may bring advantages in certain scenarios when there is preknowledge regarding the dataset.

## References


[1] ISO/TC 211 committee, "ISO/TC 211," https://www.iso.org/committee/54904.html, 2011.

[2] C. Beath, I. Becerra-Fernandez, J. Ross, and J. Short, "Finding value in the information explosion," *MIT Sloan Management Review*, vol. 53, pp. 18–20, 06 2012.

[3] E. Spyrou and Y. Avrithis, "A region thesaurus approach for high-level concept detection in the natural disaster domain," vol. 4816, 12 2007, pp. 74–77.

[4] P. Partsinevelos and Z. Mitraka, "Change detection of surface mining activity and reclamation based on a machine learning approach of multitemporal landsat tm imagery," *Geocarto International*, vol. 28, pp. 1–20, 01 2012.

[5] A. Wheeler and W. Steenbeek, "Mapping the risk terrain for crime using machine learning," 01 2020.

[6] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: A machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 506–509. [Online]. Available: https://doi.org/10.1145/3019612.3019878

[7] C. Brunsdon, A. S. Fotheringham, and M. E. Charlton, "Geographically weighted regression: A method for exploring spatial nonstationarity," *Geographical Analysis*, vol. 28, no. 4, pp. 281–298, 1996. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1538-4632.1996.tb00936.x

[8] S. Georganos, T. Grippa, A. N. Gadiaga, C. Linard, M. Lennert, S. Vanhuysse, N. Mboga, E. Wolff, and S. Kalogirou, "Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling," *Geocarto International*, pp. 1–16, 2019.

[9] A. S. Fotheringham, W. Yang, and W. Kang, "Multiscale geographically weighted regression (mgwr)," *Annals of the American Association of Geographers*, vol. 107, no. 6, pp. 1247–1265, 2017.

[10] J. Morgan, R. Dougherty, A. Hilchie, and B. Carey, "Sample size and modeling accuracy with decision tree based data mining tools," *Acad Inf Manag Sci J*, vol. 6, 01 2003.

[11] J. Mateu, "Comments on: A general science-based framework for dynamical spatio-temporal models," *Test*, vol. 19, pp. 452–455, 11 2010.

[12] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, vol. 46, pp. 234–240, 1970. [Online]. Available: http://www.jstor.org/stable/143141

[13] A. Fotheringham, C. Brunsdon, and M. Charlton, "Geographically weighted regression: The analysis of spatially varying relationships," *John Wiley and Sons*, vol. 13, 01 2002.

[14] L. Breiman, "Random forests," in *Machine Learning*, 2001, pp. 5–32.

[15] Kaggle, "King County Housing Market data," https://www.kaggle.com/harlfoxem/housesalesprediction, 2016.

[16] A.-L. Boulesteix, S. Janitza, J. Kruppa, and I. R. König, "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics," *WIREs*

*Data Mining and Knowledge Discovery*, vol. 2, no. 6, pp. 493–507, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1072

[17] M. Nussbaum, K. Spiess, A. Baltensweiler, U. Grob, A. Keller, L. Greiner, M. E. Schaepman, and A. Papritz, "Evaluation of digital soil mapping approaches with large sets of environmental covariates," *SOIL*, vol. 4, no. 1, pp. 1–22, 2018. [Online]. Available: https://soil.copernicus.org/articles/4/1/2018/

[18] S. Putra and A. Wanto, "Analysis of artificial neural network accuracy using backpropagation algorithm in predicting process (forecasting)," *International Journal Of Information System & Technology (IJISTECH)*, vol. 1, pp. 34–42, 11 2017.

[19] J. Hernandez-Orallo, C. Ferri, N. Lachiche, A. Martínez-Usó, and M. Ramírez-Quintana, "Binarised regression tasks: methods and evaluation metrics," *Data Mining and Knowledge Discovery*, vol. 30, 11 2015.

[20] T. Hengl, M. Nussbaum, M. N. Wright, G. B. Heuvelink, and B. Gräler, "Random forest as a generic framework for predictive modeling of spatial and spatio-temporal variables," *PeerJ*, vol. 6, p. e5518, Aug. 2018. [Online]. Available: https://doi.org/10.7717/peerj.5518