MSc Mathematical Statistics Dissertation

Geographically Weighted Statistical Machine Learning Methods for Predicting Net Primary Productivity in the Eastern Sahel Region

By

Kopano Letsela

Supervisors

Dr Farai Mlambo (School of Statistics and Actuarial Science and Wits Business School)

Prof Elhadi Adam (School of Geography, Archaeological and Environmental Science)

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science.

June 9, 2025

Abstract

Sustainable land management and ecosystem resilience are essential for climate adaptation and resource conservation, particularly in regions susceptible to environmental degradation. This study applies the Geographically Weighted Statistical Machine Learning (GWSML) methods to predict Net Primary Productivity (NPP) in the eastern Sahel, a semi-arid region characterised by high climate variability, land degradation, and socio-economic vulnerability. By integrating Geographically Weighted Regression (GWR), Geographically Weighted Random Forests (GWRF), and Geographically Weighted Neural Networks (GWNN), the research addresses spatial heterogeneity and nonlinearity in environmental data, overcoming the limitations of traditional global models.

Using data from Niger, Chad, and Sudan, spanning 2019-2021, the models leverage spatially explicit climatic variables—rainfall, temperature, soil moisture, and elevation—to estimate NPP with high accuracy. The data were processed and analysed using Ordinary Kriging (OK) to handle missing data, followed by model calibration. Spatial autocorrelation in residuals was examined using Moran's I, and the evaluation was conducted using spatial regression and geographically weighted machine learning techniques. Model performance evaluation was carried out using key metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2). This evaluation involved a comparative analysis with global models, particularly Ordinary Least Squares (OLS), and traditional machine learning methods, including Random Forests (RF) and Neural Networks (NN).

The results demonstrate that machine learning methods, enhanced with geographical weighting, outperform traditional/global approaches by capturing localised variations and nonlinear dependencies. The best results produced for this study were an R^2 of 0.9360, RMSE of 0.0333, and MSE of 0.0012, all achieved by the GWNN model. The GWRF model yielded the best MAE of 0.0191, albeit with a lower R^2 of 0.9308 compared to GWNN. However, GWR produced better performance than global models, with an R^2 of 0.9207. The study results show that GWR, GWRF, and GWNN outperform global regression models in their ability to capture spatial vari-

ability. Concurrently, GWRF and GWNN significantly improve prediction accuracy, effectively capturing nonlinear relationships and spatial heterogeneity between NPP and its drivers.

The findings highlight the importance of spatially adaptive models for predicting ecological productivity and informing climate adaptation strategies. These models can help mitigate land degradation and promote sustainable agriculture in regions with spatial heterogeneity. Integrating these methods into ecological modelling promises improved outcomes for socio-economic stability, environmental sustainability, and food security in developing climate-vulnerable regions like the eastern Sahel.

Keywords: *Net Primary Productivity*, *Sustainable Land Management*, *Geographically Weighted Machine Learning*, *Eastern Sahel Region*, *Spatial Analysis*.

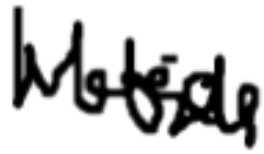
Declaration

I, Kopano Letsela, affirm that this dissertation, titled **"Geographically Weighted Statistical Machine Learning Methods for Predicting Net Primary Productivity in the Eastern Sahel Region"**, represents my original work. It is being submitted to fulfil the requirements for the Master of Science degree in Mathematical Statistics at the University of the Witwatersrand, Johannesburg.

The research was carried out under the guidance of Dr. Farai Mlambo (School of Statistics and Actuarial Science and Wits Business School) and Prof. Elhadi Adam (School of Geography, Archaeology, and Environmental Science). Neither this dissertation nor any portion of it has been submitted for another degree or qualification at this or any other institution.

All sources referenced in this work have been appropriately cited, and the data, analysis, and conclusions presented are the result of my independent efforts, except where explicitly stated otherwise.

Kopano Letsela
June 9, 2025



Acknowledgement

With profound gratitude, I extend my heartfelt thanks to my supervisors, Dr. Farai Mlambo (School of Statistics and Actuarial Science and Wits Business School) and Prof. Elhadi Adam (School of Geography, Archaeology, and Environmental Science). Their unwavering guidance, encouragement, and expertise were pivotal in the successful completion of this research.

I also wish to express my deep appreciation to the School of Statistics and Actuarial Science at the University of the Witwatersrand for providing me with the opportunity to undertake this study. Their support, particularly under the leadership of Dr. Charles Chimedza, played a crucial role in shaping this work.

I would like to extend my sincere thanks to the National Manpower Development Secretariat (NMDS) for funding my degree and supporting this academic journey.

Lastly, I am immensely grateful to my family and friends for their unwavering commitment, patience, and encouragement. Their sacrifices and belief in me have been a constant source of strength throughout this endeavour.

Kopano Letsela
June 9, 2025

Contents

List of Figures

List of Tables

List of Acronyms

AIC	Akaike Information Criterion
AICc	corrected Akaike Information Criterion
ANN	Artificial Neural Network
AVHRR	Advanced Very High Resolution Radiometer
CHIRPS	Climate Hazards Group InfraRed Precipitation with Station Data
CV	Cross-Validation
DAAC	Distributed Active Archive Center
DEM	Digital Elevation Model
ECMWF	European Centre for Medium-Range Weather Forecasts
ESA CCI	European Space Agency's Climate Change Initiative
GADM	Global Administrative Areas
GIMM-3G+	Global Inventory Modeling and Mapping Studies- 3rd Generation
GWNN	Geographically Weighted Neural Networks
GWR	Geographically Weighted Regression
GWRF	Geographically Weighted Random Forests
IncMSE	Increase in Mean Squared Error
LISA	Local Indicator of Spatial Autocorrelation
LST	Land Surface Temperature
MAE	Mean Absolute Error
MSE	Mean Squared Error
NASA	National Aeronautics and Space Administration
NDVI	Normalised Difference Vegetation Index
NPP	Net Primary Productivity
OK	Ordinary Kriging
OLS	Ordinary Least Squares
OOB	Out-Of-Bag
PDPs	Partial Dependence Plots
PReLU	Parametric Rectified Linear Unit
R-squared	Coefficient of Determination
RF	Random Forests
RGS	Random Grid Search
RMSE	Root Squared Mean Error
SRTM	Shuttle Radar Topography Mission
SWM	Spatial Weight Matrix
SWNN	Spatial Weighted Neural Networks
TINDVI	Time-Integrated Normalised Difference Vegetation Index
VIF	Variance Inflation Factor
WGS84	World Geodetic System 84
XAI	Explainable Artificial Intelligence

Chapter 1

Introduction

1.1 Background

Net Primary Productivity (NPP) serves as a vital metric for assessing global ecological changes, as it measures the amount of carbon absorbed by plants through photosynthesis and stored as biomass (Abdi et al., 2014; Li et al., 2003). This stored carbon, essential for producing food, fuel, and livestock feed, acts as the primary energy foundation for ecosystems and, by extension, for human societies that depend on these ecological systems (Abdi et al., 2014). It is also an essential indicator of agricultural output, carbon sequestration, and ecosystem health, making it crucial for addressing challenges related to climate change, food security, and mitigation efforts (Deb Burman, 2020).

The influence of NPP arises from various factors, including climate, soil nutrients, vegetation types, land use, and others (Abdi et al., 2014), with spatial variability apparent as environmental conditions differ across locations (Wang et al., 2023). Quantifying NPP within a spatial context presents a significant challenge at regional, landscape, and continental scales (Quan et al., 2005). Consequently, investigating the factors that influence NPP is key to understanding how ecosystems respond and adapt, offering a scientific foundation for safeguarding ecological systems and promoting sustainable development (Piao et al., 2010).

Although previous studies have offered valuable insights into the drivers of NPP, there remains a significant gap in understanding the marginal contributions of these factors to variations in NPP. Recent academic findings have highlighted the usefulness of both direct and indirect modelling approaches in predicting spatial variations in NPP, contributing to a deeper understanding of ecosystem productivity and its responses to environmental change (Deb Burman, 2020). Among

these, indirect models, particularly regression models, have been widely used to examine the spatial variability of NPP and the drivers affecting it (Lu et al., 2021). Notably, local regression methods such as Geographically Weighted Regression (GWR) have gained prominence, offering a nuanced understanding of NPP dynamics by allowing regression model parameters to vary across different regions (Li et al., 2022; Lu et al., 2021; Quan et al., 2005; Yang et al., 2023). This approach addresses the limitations of global models (Fotheringham et al., 2003).

Ongoing research has emphasised that variations in NPP exhibit significant nonlinear properties. This suggests that changes in NPP do not follow straightforward linear relationships but are instead characterised by substantial variability over time or in response to environmental factors (Xiao et al., 2023). To address these nonlinear relationships, integrating geographical weighting with machine learning methods like Random Forests (RF) and Artificial Neural Networks (ANN) can be effective. Such integrated approaches include Geographically Weighted Random Forests (GWRF) (Santos et al., 2019) and Geographically Weighted Neural Networks (GWNN) (Hagenauer and Helbich, 2022). These geographically weighted models are essential for uncovering spatial heterogeneity in the relationships between variables, which might be overlooked in broader analysis (Quan et al., 2005), offering a more nuanced understanding of the relationships between NPP and diverse environmental factors.

The Sahel is one of the least developed regions in Africa, characterised by generally low primary productivity and numerous conflicts among different social groups (Sakor, 2020). Geographically, it represents a vast area located in the northern section of Africa, stretching from the Atlantic Ocean to the coast of the Red Sea (Epule et al., 2022). This region is notably extensive, functioning as a transitional buffer between the Sahara and the southern savannahs (Rose, 2015). The Sahel is typically rocky, semi-arid, and barren, although variations in geographic conditions do occur. The region spans several countries.

Key features influencing this region include highly variable and unpredictable rainfall, a limited moisture budget, and extremely hot summers, with temperatures ranging from $33°C$ to $42°C$ (Pomposi et al., 2015; WMO, 2022). As a result, the Sahel region has been identified as a hotspot for land degradation, leading to biodiversity loss and food insecurity. Worsening climate change trends in Africa, particularly in the Sahel (WMO, 2022), will continue to exert pressure on existing resources, however, these threats can be mitigated through the implementation of sustainable resource utilisation practices (Mbow et al., 2021).

The Sahel is expected to experience increased aridity due to the region's anticipated rapid population growth. Consequently, there is likely to be a decline in agricultural production in the

region, along with an increase in food demand. As a result, greater attention should be given to the Sahel region, particularly the eastern part, which includes countries experiencing food insecurity and social instability (Sakor, 2020). Thus, this study seeks to explore how climatic data can be used to predict NPP in the eastern Sahel region in the future.

1.2 Statement of the Problem

The Sahel region, particularly its eastern part, faces challenges in accurately predicting NPP due to spatial heterogeneity and complex environmental dynamics. Traditional global models often fail to capture localised variations in vegetation productivity, resulting in suboptimal predictions and limited insights into sustainable land management practices.

Existing research on NPP prediction in the Sahel region relies on static modelling approaches that overlook spatially varying relationships between climatic factors and vegetation productivity (Higginbottom and Symeonakis, 2013). Combining GWR with machine learning techniques, such as GWNN and GWRF, presents an opportunity to enhance the accuracy and spatial detail of NPP predictions. However, their application in the Sahel region remains underexplored. This dissertation aims to apply and spatially adapt established statistical methodologies that leverage spatial information in the Sahel region, particularly the eastern part, to improve NPP prediction. This study aims to overcome the limitations of conventional modelling approaches by combining GWR with machine learning methods tailored to the region's unique characteristics. The goal is to generate actionable insights that can enhance sustainable ecosystem management and strengthen climate resilience.

1.3 Aims and Objectives

1.3.1 Aims

This study aims to explore the spatial variability of NPP in the eastern Sahel region by applying advanced spatial and machine learning techniques, including GWR, GWNN, and GWRF. By incorporating spatial variability into regression and machine learning models, the study aims to enhance the precision of NPP predictions, providing a deeper insight into the environmental drivers influencing NPP in this area.

1.3.2 Objectives

- Examine the spatial variability of NPP in the eastern Sahel and its relationship with key environmental factors, including temperature, rainfall, elevation, and soil moisture.
- Employ GWR to analyse the spatial dynamics of the relationships between NPP and its predictors across the eastern Sahel.
- Apply GWR in combination with machine learning extensions such as GWRF and GWNN to model NPP and account for complex nonlinear relationships and spatial heterogeneity.
- Evaluate and compare the predictive performance, computational efficiency, and scalability of GWR, GWRF, and GWNN, analysing their accuracy, execution time, and resource consumption across different dataset sizes.
- Investigate climate variability's impact on NPP changes in the eastern Sahel and assess the broader implications for land use, agricultural practices, and the development of climate adaptation strategies.

1.4 Limitations and Assumptions

This study was conducted under the assumption of spatial homogeneity in the eastern Sahel region, focusing on the quality and transferability of data for GW models. However, limitations such as potential spatial scale constraints, challenges in model complexity, issues with data availability, impacts of spatial autocorrelation, and difficulties in interpretation were noted. These factors affected the accuracy and generalisability of the results, emphasising the need for thorough consideration and interpretation of the findings.

1.5 Thesis Outline

The rest of this dissertation is organised as follows: Chapter 2 reviews the literature on GWR and its integration with machine learning techniques for predicting NPP, along with the theoretical foundations of the study. Chapter 3 outlines the research design, including the study area, data preparation, model implementation (GWR, GWRF, and GWNN), and evaluation framework. Chapter 4 presents the analysis and discussion, covering exploratory data analysis, model calibration, and a comparative evaluation of the models. Chapter 5 summarises the key findings, highlighting model performance, influential drivers of NPP, and implications for sustainable land management. Finally, Chapter 6 concludes the study, acknowledging limitations and offering recommendations for future directions.

Chapter 2

Literature Review

2.1 Introduction

This chapter begins by discussing the use of GWR in NPP modelling, highlighting its advantages compared to traditional regression models (Section 2.2). It then explores the advancements in machine learning-enhanced GWR (Section 2.3). Furthermore, the chapter reviews the theoretical background for the methods used in this study (Section 2.4).

2.2 GWR for Modelling NPP

NPP, when considered in a spatial context, continues to pose a significant challenge at regional, landscape, and continental scales. Various models have been introduced to estimate the spatial variation in NPP. Regression techniques are widely used empirical tools in NPP simulations.

Quan et al. (2005) employed the spatial lag model, OLS, and GWR to develop an NPP regression model, examining the relationship between Chinese forest NPP and environmental factors. Data from various regions in China were utilised for the study. The NPP data for 17 different types of Chinese forests (response variable) were collected and extracted from plot inventory data across the country. The predictor variables considered in the study included altitude, precipitation, temperature, and the annual Time-Integrated Normalised Difference Vegetation Index (TINDVI). The CRU-TS 2.0 global climate database was used to obtain precipitation and temperature data, while the NDVI information was sourced from the Advanced Very High-Resolution Radiometer (AVHRR) land Biosphere data, managed by the Goddard Distributed Active Archive Center (DAAC) (Quan et al., 2005).

The research team observed a notable improvement in model performance when using GWR for NPP simulation compared to OLS. The performance of the GWR, OLS, and spatial lag models was evaluated using two metrics: the R^2 and the corrected Akaike Information Criterion (AIC_c) value. In terms of these metrics, the GWR model performed best. While the spatial lag model performed better than OLS, it was still less effective than GWR. Quan et al. (2005) concluded that incorporating GWR in future regression analysis addressing spatial non-stationarity is crucial, particularly when interpolation is significant, as is the case in a large region like China.

The primary focus of this dissertation, inspired by the findings of Quan et al. (2005), is to explore the potential of GWR and its combination with machine learning models for more accurate estimation of NPP in large regions like the Sahel, where interpolation is highly relevant. Understanding how environmental and socioeconomic factors affect outcomes and improving upon the techniques used in earlier studies, such as those by Quan et al. (2005), are the driving motivations behind this research.

NPP holds significant value as an indicator that directly reflects the production capacity of vegetation and the accumulation of organic matter. Consequently, researchers are tasked with assessing the impact of environmental projects on NPP. Most NPP-related research has focused on simulating NPP using traditional or enhanced models, studying spatiotemporal changes in NPP characteristics, and analysing the effects of various factors on NPP (Yang et al., 2023).

The study by Yang et al. (2023) employed the GWR model to analyse the factors influencing NPP. The model was applied to datasets resampled into 8km × 8km pixels, with predictors including temperature, precipitation, elevation, and population density. According to the study, these factors significantly influenced NPP, with human activities having an increasing impact over time. The results showed an upward trend over three years (2000, 2011, 2021), demonstrating the effectiveness of the GWR model in accounting for spatial impacts and geographical locations.

NPP is a crucial metric for understanding ecosystem functioning and carbon cycling. Therefore, it is essential to evaluate how land use changes affect ecosystems. Lu et al. (2021) stated that regional ecosystem NPP was significantly impacted by urbanisation. The research conducted in Kunming City, China, divided the influence of urbanisation on NPP into a direct effect (NPPdir) and an indirect effect (NPFind) for the years 2009 and 2018. It used GWR to uncover the spatial variability of the Land Surface Temperature (LST) effects on NPPind.

This study compares GWR and OLS regression techniques to determine which is superior. Results show that GWR has a better fit and can better explain the spatial variability of LST's effect on NPPind, with higher R^2 values and lower AIC_c values in both 2009 and 2018. This suggests that GWR is a more accurate regression method. The research findings indicate that urbanisation has both positive and negative impacts on NPP, with LST playing a role in promoting urban NPP (Lu et al., 2021).

A similar study by Li et al. (2022), assessed the impacts of urbanisation on vegetation NPP in the Chengdu-Chongqing urban agglomeration. They scrutinised the spatial changes in NPP at the township scale, focusing on the effects of urbanisation factors such as land use changes, population growth, and economic development. Their findings underscored the significant local variation in the influence of the Comprehensive Urbanisation Level (UL) on NPP, thereby emphasising the importance of local examination of these driving effects. Furthermore, their study revealed that urbanisation negatively impacts NPP, underscoring the need for ecological governance and strategies to optimise urbanisation to mitigate these effects. These results are consistent with previous studies, further demonstrating the effectiveness of the GWR technique in such analysis.

2.3 Advancing GWR with Machine Learning for NPP Prediction

While GWR has proven effective in accounting for spatial heterogeneity in NPP predictions (Yang et al., 2023), its limitations in modelling complex nonlinear relationships and interactions among predictors remain a significant challenge (Wheeler and Tiefelsdorf, 2005). To date, no research has investigated the use of advanced machine learning techniques like GWNN and GWRF in the NPP prediction framework. These advanced methods offer the potential to address the identified gaps by integrating spatial weighting with nonlinear modelling and adaptive local regression. This dissertation builds on the foundational work of GWR in NPP prediction by introducing GWNN and GWRF, thereby advancing the methodological framework for large-scale regions such as the Sahel.

2.4 Theoretical Review

The theoretical review outlines and provides a mathematical explanation of the procedures and concepts employed in this study, including Ordinary Kriging (OK) for Imputation (Subsec-

tion 2.4.1), correlation analysis (Subsection 2.4.2), multicollinearity (Subsection 2.4.3), data transformation (Subsection 2.4.4), GWR (Subsection 2.4.5), GWRF (Subsection 2.4.6), GWNN (Subsection 2.4.7), Moran models (Subsection 2.4.8), performance evaluation metrics for prediction (Subsection 2.4.9), and stratified sampling for data splitting (Subsection 2.4.10).

2.4.1 Missingness: Ordinary Kriging (OK)

The imputation of missing data in geostatistical contexts is primarily conducted through interpolation methods, with kriging being a key technique (Bae et al., 2018; Chung et al., 2019). This is a logical strategy, as the task of filling in missing observations can be viewed as an inference problem that kriging can resolve. This approach is particularly suitable for this study's inherently geostatistical dataset.

The application under consideration includes four types of kriging: simple kriging, OK, simple co-kriging, and ordinary co-kriging, as described by Bae et al. (2018). The most commonly used method is OK. It serves as a linear unbiased estimator, with the property that the mean of the errors is zero. In the context of OK, the local mean is separated from the linear estimator by ensuring that the sum of the kriging weights equals one (Wackernagel and Wackernagel, 2003). This technique is generally preferred over simple kriging because it eliminates the need for prior knowledge about the mean or the stationarity of the mean across the area, often resulting in a lower mean squared error (Chung et al., 2019). Thus, this paper primarily considers the application of OK as an interpolation method to handle missing data.

The OK estimator is given by

$$\widehat{Z}_{\mathrm{OK}}\left(\boldsymbol{s}_0\right)=\sum_{i=1}^{n} w_i Z\left(\boldsymbol{s}_i\right) \text{ subject to } \sum_{i=1}^{n} w_i=1, \tag{2.1}$$

where:

- $Z\left(\boldsymbol{s}_i\right)$ is the random variable at the i^{th}location,
- w_i is the kriging weight for the observed value at the i^{th}location,
- $\boldsymbol{s}_0$ is the location of the prediction,
- n is the number of measured values.

2.4.2 Correlation

Correlation or correlation analysis refers to the association or relationship between multiple quantitative variables (Gogtay and Thatte, 2017). This analytical technique is essential for determining whether and how strongly linear correlations exist between variables. A correlation coefficient is a statistical indicator that indicates the strength of the association between two variables, and it is obtained from the correlation analysis procedure (Senthilnathan, 2019). Pearson's correlation coefficient is typically the most widely used. Equation 2.2 below provides its mathematical representation.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 (y_i - \bar{y})^2}}, \tag{2.2}$$

where:

- r represents the correlation coefficient,
- x_i represents the values of the predictor variable in the dataset,
- y_i represents the values of the target variable in the dataset,
- $\bar{x}$ represents the average of the predictor variable,
- $\bar{y}$ represents the average of the target variable.

The value of r ranges from -1 to 1. Table 2.1 shows how the coefficient values can be interpreted.

Table 2.1: Correlation Coefficient and Interpretation (Senthilnathan, 2019)

Correlation Coefficient	**Interpretation**
-1	Perfect Negative Linear Association
$-1 < r \leq -0.6$	Strong Negative Linear Association
$-0.6 < r \leq -0.4$	Moderate Negative Linear Association
$-0.4 < r < 0$	Weak Negative Linear Association
0	No Association
$0 < r < 0.4$	Weak Positive Linear Association
$0.4 \leq r < 0.6$	Moderate Positive Linear Association
$0.6 \leq r < 1$	Strong Positive Linear Association
1	Perfect Positive Linear Association

2.4.3 (Multi)Collinearity

In the context of multiple regression models, collinearity occurs when two or more predictor variables are correlated, indicating that one variable can be expressed as a linear combination of the others (Daoud, 2017). The Variance Inflation Factor (VIF) is a tool used to determine multicollinearity. The VIF formula, as provided by Thompson et al. (2017), is displayed below in Equation 2.3.

$$\text{VIF} = \frac{1}{1 - R_k^2}, \tag{2.3}$$

where R_k^2 is the multiple coefficients of determination of the k^{th} regressor, X_k, on the other regressors. The criteria for interpreting VIF are outlined in Table 2.2, as presented by Senthilnathan (2019).

Table 2.2: Interpretation of Multicollinearity using VIF Scores

VIF Score	**Multicollinearity Interpretation**
$VIF = 1$	Null Multicollinearity
$1 < VIF < 5$	Low Multicollinearity
$5 \leq VIF < \infty$	High Multicollinearity
$VIF \to \infty$	Perfect Multicollinearity

2.4.4 Transformation of Continuous Predictors

In statistical and machine learning analysis, continuous predictors often need to be transformed to ensure they contribute equally to the model, particularly when the predictors are measured on different scales (Ali et al., 2014). Two common methods for transforming continuous predictors are standardisation and min-max normalisation. In this study, standardisation is used for the transformation of continuous predictors. Standardisation is particularly useful when predictors have different units or ranges, ensuring that each variable contributes equally to the statistical and machine learning analysis (Ali et al., 2014).

For a given continuous predictor $x_j = \{x_{1,j}, \ldots, x_{n,j}\}$, standardisation transforms the original observations of a predictor so that the transformed values have a mean of 0 and a variance of 1. Thus the standardisation of the *i*-th observation $x_{i,j}$ is achieved using the following transformation:

$$z_{i,j} = \frac{x_{i,j} - \bar{x_j}}{s_j}, \tag{2.4}$$

where $\bar{x_j}$ and s_j are the sample mean and standard deviation, respectively, of the original observations of the *j*-th predictor.

2.4.5 Geographically Weighted Regression

The following section provides an in-depth look at GWR, beginning with an overview of GWR and its significance in various studies. This is followed by an examination of GWR's underlying theory, particularly its unique ability to address geographical heterogeneity. The discussion then transitions to the formulation and calibration of the GWR model.

2.4.5.1 Overview

Regression modelling requires careful consideration of spatial non-stationarity, which indicates that the relationship between variables varies throughout space within the study area (Brunsdon et al., 1996). Ignoring this can lead to biased parameter estimates, resulting in inaccurate predictions. Therefore, spatial non-stationarity must be considered and accounted for to enhance the accuracy and robustness of regression models in spatial data analysis. GWR is renowned as an extremely effective method for detecting, incorporating, and adjusting to spatial non-stationarity (Sulekan and Jamaludin, 2020).

GWR is a methodological approach used to account for spatial heterogeneity through the calibration of a multiple regression model. This approach enables the identification of different relationships that may exist at various locations within a given space (Sulekan and Jamaludin, 2020). Wheeler (2021) highlights that GWR possesses a unique capability to distinguish between spatially constant and varying processes, and it excels in accurately retrieving spatially varying relationships. Fotheringham et al. (2003) further emphasise its role in accounting for and examining spatial non-stationarity, producing more informative results regarding variation across space. Sulekan and Jamaludin (2020) also highlight the potential of visualising the variability in the original regression parameters to better understand the spatial patterns in the relationship between explanatory and target variables. These qualities make GWR a valuable method for understanding and analysing spatially varying relationships.

Developed primarily for spatial point data analysis, GWR provides the capability to interpolate values that are not explicitly present within the dataset and to address the misaligned data problem (Lin et al., 2011). GWR recognises and accommodates the spatial non-stationarity of relationships between variables by incorporating local contextual factors into regression analysis. By estimating separate models for different locations, GWR enables the investigation of

spatial non-stationarity and provides a more nuanced knowledge of how relationships between variables are modified by local contextual factors across space. GWR is extremely useful in epidemiology, particularly in infectious disease research and health policy or programme evaluation. The limitations of GWR include challenges associated with multicollinearity and the techniques used for determining goodness-of-fit measures.

2.4.5.2 Theory of GWR

GWR is a powerful exploratory tool primarily designed to highlight where non-stationarity occurs on the map, emphasising the points where locally weighted regression coefficients differ from their global counterparts (Bivand, 2017). GWR operates on the principle that it enables the estimation of parameters at any point within the study area by using a response variable in conjunction with one or more predictor variables recorded at specific locations (Charlton et al., 2009).

Tobler's first law of geography, which states that observations nearer to one another have a greater influence on parameter estimations than observations farther away, is consistent with the fundamental assumptions of GWR (Thapa and Estoque, 2012). To obtain a local linear regression estimate for each point in space, this technique uses distance-weighted subsampling of the data (LeSage, 2004). In GWR, the idea is to capture spatially varying relationships between variables and covariates more effectively than by using constant regression coefficients (Wheeler, 2021).

2.4.5.3 Formulation and Calibration of GWR Model

A variety of techniques are used in regression analysis to represent the association between one or more predictor variables and a target variable. The regression model assumes that there is no influence between observations, meaning they should be independent. However, this may not always hold true for data associated with spatial units. GWR addresses these problems by evaluating the localised associations between the target variable and predictor variables in various geographical areas (Sulekan and Jamaludin, 2020). This is achieved by assigning geographical weights to observations, with closer observations receiving higher weights.

Locally weighted regression forms the basis of GWR, a non-parametric technique that involves estimating regression parameters using data subsets near a model estimate point in variable space (Wheeler, 2021). GWR introduces an innovative approach by employing a subset of data

that is geographically near the model calibration location, as opposed to using variable space (Thapa and Estoque, 2012). Consider a global/traditional regression model:

$$y_i = \beta_0 + \sum_k \beta_k x_{ik} + \varepsilon_i, \tag{2.5}$$

where y_i denotes outcome of interest (target variable), x_{ik} denotes the k^{th} variable's value for observation i, β_k refers to the coefficients to be estimated, and ε_i is the error term for all $i = 1, \ldots, n$. The estimates for Equation 2.5 are given in Equation 2.6 below.

$$\hat{\beta} = \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}, \tag{2.6}$$

where $\hat{\beta}$ represents an estimate of β. Unlike conventional linear regression models that assume uniform coefficients across different spatial locations, GWR calculates regression coefficients in a localised manner at specific spatially referenced data points (Wheeler, 2021). The regression model that underlies GWR is presented in Equation 2.7 below.

$$y_i = \beta_0 \left(u_i, v_i\right) + \sum_k \beta_k \left(u_i, v_i\right) x_{ik} + \varepsilon_i, \tag{2.7}$$

where $\left(u_i, v_i\right)$ refers to the coordinates of the i^{th} point in the spatial domain, while $\beta_0 \left(u_i, v_i\right)$ signifies the intercept coefficient at that specific position. Additionally, $\beta_k \left(u_i, v_i\right)$ denotes the coefficient associated with the k^{th} predictor variable at the coordinates $\left(u_i, v_i\right)$. Equation 2.8 provides an expression for the estimated coefficients using weighted least squares.

$$\hat{\beta} \left(u_i, v_i\right) = \left(\boldsymbol{X}^T \boldsymbol{W} \left(u_i, v_i\right) \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{W} \left(u_i, v_i\right) \boldsymbol{y}, \tag{2.8}$$

where $\boldsymbol{y}$ is a $k \times 1$ explanatory variable vector, $\boldsymbol{X}$ is an $n \times k$ explanatory variable matrix, and $\boldsymbol{W} \left(u_i, v_i\right)$ is a $n \times n$ weight matrix specific to location i, ensuring that observations closer to i are assigned higher weights compared to those farther away. In $\boldsymbol{W} \left(u_i, v_i\right)$, the off-diagonal entries are zero, and the diagonal entries represent the geographic weight coefficients.

Within GWR, each attribute in the dataset is analysed using a regression equation to create a localised model aimed at understanding or predicting the variable or process involved. This is done by including the response and predictor variables of the attributes located near each feature. The determination of weights for the local models relies on a kernel, which acts as a distance decay function. This kernel effectively controls the rate at which weights decrease as the distance between elements increases. Table 2.3, as presented by Gollini et al. (2013), shows frequently used kernel functions in GWR, including Gaussian, exponential, boxcar, tricube, and bisquare kernels. Figure 2.1 illustrates a typical function. It reveals that data points distant from **X** will correspond to smaller weights, whereas those closer to **X** will have larger weights.

Additionally, a data point exactly at the regression point will be assigned the highest weight of one.

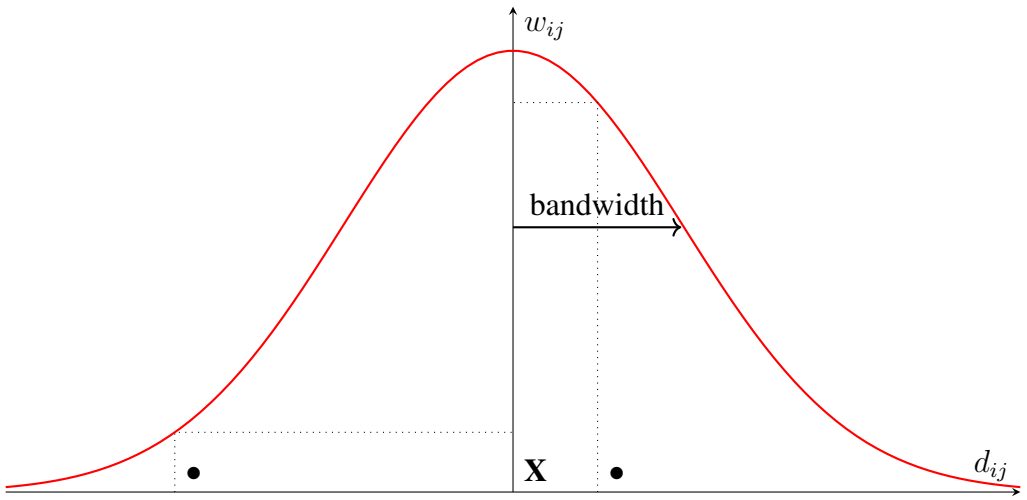


Figure 2.1: Spatial Kernel Plot

Where:

* **X**: regression point
* •: data point
* d_{ij}: distance between regression point *i* and data point *j*
* w_{ij}: weight of data point *j* at the regression point *i*
* bandwidth: labelled *b* in Table 2.3, is a key parameter that determines the size of the local neighbourhood around each data point

GWR results are relatively robust across different weighting functions, but they are sensitive to bandwidth selection. Choosing an appropriate bandwidth is essential for capturing meaningful spatial variations and ensuring reliable predictions in GWR analysis (Fotheringham et al., 2003). Thapa and Estoque (2012) highlight cross-validation (CV) and Akaike Information Criterion (AIC) as commonly used methods for selecting an appropriate bandwidth in GWR.

For bandwidth selection in GWR, CV can be used to evaluate the predictive accuracy of models with varying bandwidth values. The bandwidth that results in the best CV performance (e.g., lowest prediction error) is typically selected as the optimal bandwidth (Sulekan and Jamaludin, 2020; Thapa and Estoque, 2012). The goal is to minimise CV score, detailed as follows:

$$\mathrm{CV} = \sum_{i=1}^{n} \left[y_i - \hat{y}_{\neq i}(b) \right]^2 , \tag{2.9}$$

Table 2.3: Five Kernel Weighting Functions

Exponential	$w_{ij} = \exp\left(-\frac{\lvert d_{ij}\rvert}{b}\right)$
Gaussian	$w_{ij} = \exp\left(-\frac{1}{2}\left(\frac{d_{ij}}{b}\right)^2\right)$
Bi-square	$w_{ij} = \begin{cases} \left(1-(d_{ij}/b)^2\right)^2 & \text{if } \lvert d_{ij}\rvert < b, \\ 0 & \text{otherwise} \end{cases}$
Tri-cube	$w_{ij} = \begin{cases} \left(1-(\lvert d_{ij}\rvert/b)^3\right)^3 & \text{if } \lvert d_{ij}\rvert < b, \\ 0 & \text{otherwise} \end{cases}$
Box-car	$w_{ij} = \begin{cases} 1 & \text{if } \lvert d_{ij}\rvert < b, \\ 0 & \text{otherwise} \end{cases}$

where $\hat{y}_{\neq i}(b)$ is the predicted value of y_i obtained by excluding the observation for point i during calibration, and n is the total number of observations.

A similar strategy for selecting the bandwidth parameter in GWR requires a careful balance between model complexity and model fit. This balance is commonly referred to as a trade-off between goodness-of-fit and degrees of freedom (Sulekan and Jamaludin, 2020). One typical way to achieve this balance is to minimise the AIC, a metric that provides a measure of the relative quality of GWR (Charlton et al., 2009; Thapa and Estoque, 2012), which is defined as

$$\text{AIC} = 2n\log_e(\hat{\sigma}) + n\log_e(2\pi) + n\left\{\frac{n+\text{tr}(\boldsymbol{S})}{n-2-\text{tr}(\boldsymbol{S})}\right\}, \tag{2.10}$$

where $\hat{\sigma}$ is the error term's estimated standard deviation, $\text{tr}(\boldsymbol{S})$ is the hat matrix trace which maps $\hat{\boldsymbol{y}}$ onto $\boldsymbol{y}$, and n is the number of observations.

Fotheringham et al. (2003) discuss two main types of spatial kernels used in GWR: adaptive spatial kernels and fixed spatial kernels (Figure 2.2 right and left, respectively). Adaptive spatial kernels have the advantage of dynamically modifying the bandwidth based on the local spatial structure of the data, leading to more accurate and spatially adaptable estimates of local regression coefficients. Fixed spatial kernels offer a straightforward approach, as each data point is weighted by its distance from the regression point coefficients (Guo et al., 2008). In summary, while fixed kernels apply uniform weights within their bandwidth regardless of distance from regression points, adaptive kernels modify weights based on distances from them.

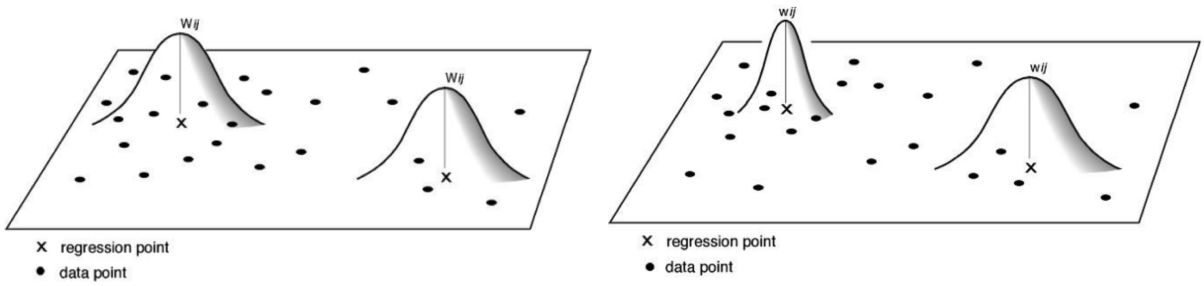


Figure 2.2: The Diagram Visually Depicts Both the Fixed (left) and the Adaptive (right) Weighting Scheme (Fotheringham et al., 2003)

2.4.6 Geographically Weighted Random Forests

Building on earlier discussions of GWR, this section revisits RF principles. It then explores GWRF, a hybrid model integrating GWR and RF for nuanced spatial data analysis.

2.4.6.1 Overview

GWRF combines the strengths of GWR and RF to tackle spatial heterogeneity in data. While GWR accommodates spatial variation by allowing parameters to differ across locations (Fotheringham et al., 2003), RF is a powerful ensemble learning algorithm employed for both regression and classification tasks (Breiman, 2001). GWRF extends RF by introducing localised models that adapt to spatial variation, making it well-suited for analysing high-dimensional, nonlinear, and multicollinear data (Luo et al., 2022).

2.4.6.2 Principles of Random Forests

RF are non-parametric, ensemble learning algorithms that use bagging and random feature selection to construct multiple decision trees (Breiman, 2001). Bagging refers to the process of generating a subset of training data for each decision tree (known as an In-Bag sample) by using sampling with replacement from the original training dataset. Conventionally, this subset comprises approximately two-thirds of the total training data. These subsets are used to grow individual trees. At each node of the tree, a randomly chosen subset of features is used to determine the split, thereby increasing the diversity of the model. Predictions are aggregated across all trees, with majority voting used for classification tasks and averaging used for regression tasks.

Out-of-bag (OOB) samples, representing approximately one-third of the data excluded during tree construction, are used for model validation and independent accuracy indicators (Lotfata

and Georganos, 2023). By using RF OOB, researchers can effectively assess the importance of explanatory variables (Breiman, 2001). Typical methods include Gini importance, Mean Squared Error (MSE), and permutation (Grömping, 2009). RF operates independently of any statistical assumptions regarding the data's distribution. Consequently, it is particularly effective in handling variables that exhibit nonlinear relationships.

2.4.6.3 Development and Formulation of GWRF

RF remains a global 'aspatial' concept, which may not adequately address the complexities of spatial heterogeneity (Wu et al., 2024). For this reason, Georganos et al. (2021) introduced GWRF to disaggregate RF within the geographical space by implementing local sub-models. The fundamental concept aligns with that of GWR (Fotheringham et al., 2003), as both methodologies emphasise the importance of locally calibrating the model rather than globally. To handle spatial heterogeneity, GWRF makes use of a spatial weights matrix (SWM) that is locally calibrated, taking into account only the observations nearby through a spatial kernel (Lotfata and Georganos, 2023). Consequently, a local basis is established and evaluated for each location using data from nearby spatial units. In essence, this indicates that for every training data point, an RF is calculated, which includes evaluations of feature importance, predictive capabilities, and performance metrics (Georganos et al., 2021), due to its non-parametric characteristics (Santos et al., 2019). The output of a GWRF model for a given location i can be expressed as:

$$Y_i = \frac{1}{B} \sum_{b=1}^{B} w_b(i) \cdot f_b(X_i) + \varepsilon_i, \tag{2.11}$$

where:

- Y_i is the output at data point i,
- B is the RF's total number of trees,
- $w_b(i)$ is the spatial weight for tree b at data point i,
- $f_b(X_i)$ is the prediction of the b-th tree for the input variables X_i,
- ε_i is the residual term.

2.4.6.4 Local Calibration and Spatial Weighting

Spatial weighting ($w_b(i)$ from Equation 2.11) is applied during the bagging stage, where each observation is assigned a sample probability based on a SWM that considers the distance between observations (Santos et al., 2019). In contrast to the RF model, numerous RF models

are built for distinct homogeneous sub-regions obtained from the spatial partitioning procedure, which accounts for the effect of geographic scene variety. For each RF model, the training input includes both the attributes of the target sample and those of its neighbouring samples in space, as depicted in Figure 2.3, which enhances the model's effectiveness in capturing spatial contextual features (Lan et al., 2023).

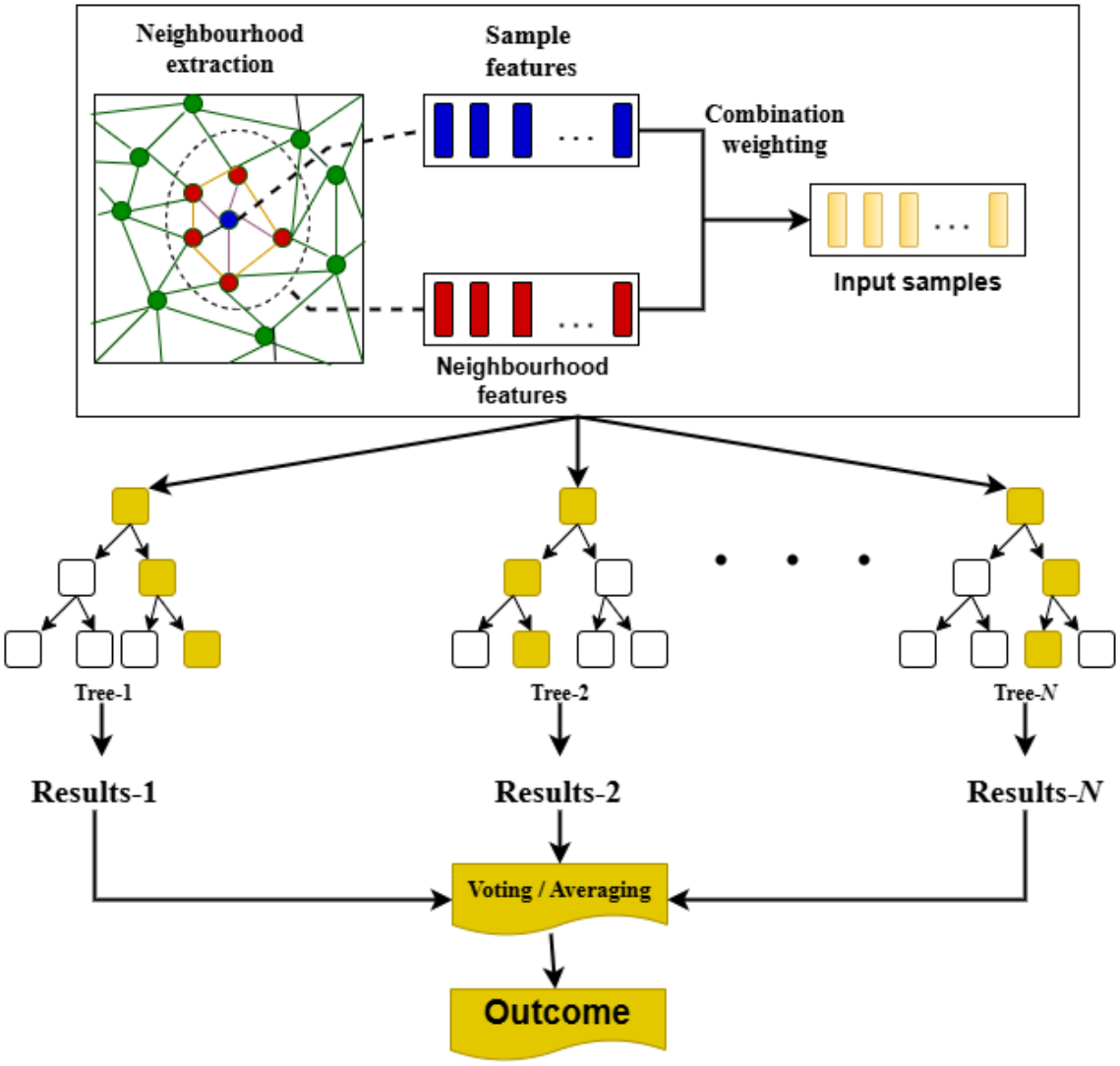


Figure 2.3: The Workflow of the GWRF Model (Sourced from Lan et al. (2023))

The success of GWRF lies in its local calibration process, which uses a defined neighbourhood or kernel to train models. Neighbourhoods are determined using either a distance threshold value (fixed kernel) or the number of closest neighbours (adaptive kernel) (Nduwayezu et al., 2023). The adaptive kernel is particularly useful for datasets with uneven spatial sampling densities, while the fixed bandwidth kernel is better suited for evenly distributed samples

(Georganos et al., 2021).

2.4.7 Geographically Weighted Neural Networks

This section reviews the fundamental concepts of GWNN models. A detailed explanation is provided for the hybrid model GWNN, which combines GWR and ANN.

2.4.7.1 Overview

GWNN represents an innovative approach that combines the spatial adaptability of geographically weighted techniques with the flexibility of ANNs (Hagenauer and Helbich, 2022). This method addresses limitations in traditional GWR, which presumes linear relationships between target and explanatory variables (Lu et al., 2017). In real-world scenarios, nonlinear associations are common, limiting GWR's ability to capture the complexity of these relationships. By integrating geographical weighting with the learning capabilities of ANNs, GWNN enables the model to identify complex, nonlinear relationships directly from the data while simultaneously adapting to spatial heterogeneity (Wang et al., 2022).

Hagenauer and Helbich (2022) demonstrated the effectiveness of GWNN using synthetic data and a real-world case study. GWNN outperformed traditional GWR in nonlinear relationships and high spatial variability scenarios. It accurately captured underlying patterns, providing superior predictions. The real-world case study confirmed GWNN's performance in practical applications, highlighting its robustness in handling nonlinear and spatially varying relationships, making it a valuable alternative to GWR for spatial data analysis.

2.4.7.2 Artificial Neural Network

The ANN comprises a series of interconnected nodes, modelled after the simplified structure of neurons in the human brain. NNs are used to address prediction and classification tasks involving nonlinear complexities (Sharma et al., 2017). These networks are organised into three key layers: the first layer is known as the 'input layer,' the last layer is referred to as the 'output layer,' and 'hidden layers' are located in between, where neuronal weights play a pivotal role in establishing and articulating the connections that link inputs to outputs (Dongare et al., 2012). Each layer consists of multiple interconnected nodes or neurons, with every layer serving to transform input values and establish neuronal weights (Wu and Feng, 2018). The activation function plays a crucial role in transforming input weights into output, incorporating the nonlinearity essential for resolving complex nonlinear functions (Sharma et al., 2017). ANNs are

known for high accuracy and strong learning ability. However, their interpretability is often compromised by the 'black box' nature of the interactions among neurons.

2.4.7.3 Model Formulation

The GWNN model extends traditional NN architecture by incorporating location-specific parameters to better capture spatial non-stationarity. GWNN adapts the standard feedforward neural network for geographically weighted applications. Consequently, the output Y_i at location i can be expressed through the following formulation:

$$Y_i = f\left(\sum_{j=1}^{n} w_j(i) \cdot \sigma\left(\sum_{k=1}^{m} w_{jk}(i) \cdot X_{ki} + b_{jk}(i)\right) + b_j(i)\right), \tag{2.12}$$

where:

- Y_i is the output at location i,
- f is the activation function,
- X_{ki} is the k-th input variable at location i,
- $w_j(i)$ and $b_j(i)$ are the location-specific weights and biases for the j-th neuron in the hidden layer,
- $w_{jk}(i)$ and $b_{jk}(i)$ are the location-specific weights and biases for the k-th neuron in the input layer connecting to the j-th neuron in the hidden layer.

2.4.7.4 Geographical Weighting

To demonstrate the complex relationship between spatial closeness and non-stationary weights, GWNN replaces kernel functions with a spatially weighted neural network (SWNN) (Dai et al., 2022). The SWNN utilises spatial distance for its input layer, spatial weights for its output layer, and employs hidden layers to analyse the relationship between the input and output (Dai et al., 2022; Du et al., 2020; Wang et al., 2022). The following is the expression for the non-stationary weights computation for location i:

$$\boldsymbol{W}\left(u_i, v_i\right) = SWNN\left(\left[d_{i1}^s, d_{i2}^s, ..., d_{in}^s\right]^T\right), \tag{2.13}$$

where, $[d_{i1}^s, d_{i2}^s, ..., d_{in}^s]$ represents the spatial proximity (e.g., spatial distance) from the observation point i to all n sample points. The GWNN-based estimation model is illustrated in Figure 2.4.

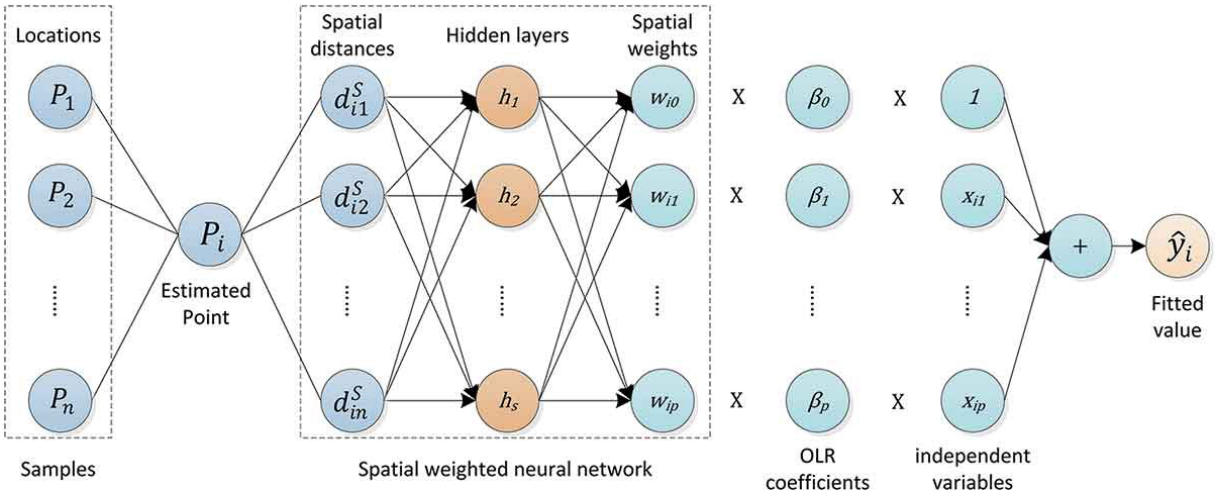


Figure 2.4: Network Structure for GWNN Model (Du et al., 2020)

The GWNN architecture, as delineated by Du et al. (2020) and Zhao et al. (2024), consists of an input layer, an output layer, and two network modules serving different roles. The number of input features, as established by the sample points used for estimation, is equal to the number of nodes in the input layer.

The SWNN's input layer is formed by computing the distance between the estimated point P_i and all training points in the dataset during the modelling process. Through the calculations performed by the SWNN, the output layer generates the spatial weights of P_i, with each weight linked to a designated predictor variable.

Subsequently, the non-stationary coefficients are derived by multiplying the spatial weights with the previously calculated OLS coefficients. The fitted value, denoted as $\hat{y}_i$, is then computed as the sum of the products of all non-stationary coefficients and their corresponding predictor variables. This comprehensive process allows for a robust and dynamic modelling procedure.

2.4.8 Moran Model

Moran's I serves as a prominent statistical measure for assessing spatial autocorrelation, providing insights into the degree of similarity between an object and its nearby entities (Bivand, 2008). The calculation of Moran's I involves determining the ratio of the product of a variable and its spatial lag to the product of the variable, adjusted according to the spatial weights used (Schabenberger and Gotway, 2017). The most commonly used models for measuring spatial clusters are local and global Moran's I (Kowe et al., 2022). Similar to a correlation coefficient, Moran's I value ranges from -1, which signifies perfect dispersion, to 1, representing perfect

correlation, with a value of 0 indicating the absence of any spatial pattern (Liu et al., 2023).

The global Moran's I formula, as presented by Schabenberger and Gotway (2017), is shown below.

$$I = \frac{n}{(n-1)S^2 w_{..}} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(Z\left(\mathbf{s}_i\right) - \bar{Z}\right) \left(Z\left(\mathbf{s}_j\right) - \bar{Z}\right), \tag{2.14}$$

where

$S^2 = \frac{1}{(n-1)} \sum_i^n \left(Z\left(\mathbf{s}_i\right) - \bar{Z}\right)^2$, an appropriate estimator for the variance of a sample, and $w_{..} = \sum_i^n \sum_j^n w_{ij}$.

The interpretation of the analysis's results is inherently tied to the null hypothesis, according to global Moran's I, an inferential statistic for spatial autocorrelation. The following hypothesis applies to the global Moran's I test:

H_0 : The spatial pattern is random $(I = E[I])$
H_a : The spatial pattern is uniform or clustered $(I \neq E[I])$.

One way to calculate the test statistic is as:

$$z\left(I\right) = \frac{I - E\left[I\right]}{\sqrt{Var\left(I\right)}}, \tag{2.15}$$

where $E[I] = \frac{-1}{n-1}$ and $Var(I) = E[I^2] - E[I]^2$.

The local Moran's I can be constructed as one of the n components that comprise the global test, as shown by Equation 2.16 below.

$$I\left(\mathbf{s}_i\right) = \frac{n}{(n-1)S^2} \left(Z\left(\mathbf{s}_i\right) - \bar{Z}\right) \sum_{j=1}^{n} w_{ij} \left(Z\left(\mathbf{s}_j\right) - \bar{Z}\right), \tag{2.16}$$

such that $\sum_{i=1}^{n} I\left(\mathbf{s}_i\right) = w_{..} I$.
The local Moran statistic, introduced by Anselin (1995), is a method for detecting local clusters and spatial outliers. By integrating local Moran's I with the positioning of each observation in the Moran scatterplot, it facilitates the categorisation of significant locations into high-low and low-high spatial outliers as well as high-high and low-low spatial clusters (Yang et al., 2023). The study identifies possible clustering within the residuals of the models using local Moran's I and examines the presence of residual spatial autocorrelation using global Moran's I.

2.4.9 Model Performance Metrics

The following performance metrics were employed to evaluate the predictive accuracy and overall effectiveness of the models applied in this study. A comparative analysis of these metrics was conducted to identify the model with superior performance.

In the following equations, X_i represents the predicted value for the i^{th} observation, while Y_i denotes the actual value for the i^{th} observation. The regression technique is used to estimate the X_i value corresponding to each Y_i value within the complete dataset (Chicco et al., 2021). The number of data points is denoted by n. Finally, the constant mean of the true values is displayed as follows:

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i \tag{2.17}$$

1. **Coefficient of Determination (R-squared or R^2)**
   The coefficient of determination is the ratio between the variance in the response variable that can be explained by the predictor variables (Di Bucchianico, 2008).

$$\mathrm{R}^2 = 1 - \frac{\sum_{i=1}^{n} (X_i - Y_i)^2}{\sum_{i=1}^{n} \left(\bar{Y} - Y_i\right)^2} \tag{2.18}$$

2. **Mean Square Error (MSE)**
   It evaluates the mean of the squared errors and is sometimes referred to as the mean squared deviation (MSD) of an estimator. It calculates the average squared difference between the actual value and the expected value (Bickel and Doksum, 2015).

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} (X_i - Y_i)^2 \tag{2.19}$$

3. **Root Mean Square Error (RMSE)**
   RMSE and MSE exhibit a monotonic relationship, with RMSE being the square root of MSE. This implies that the ranking of regression models based on MSE will be identical to that based on RMSE (Chicco et al., 2021).

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (X_i - Y_i)^2} \tag{2.20}$$

4. **Mean Absolute Error (MAE)**
   MAE is calculated by dividing the sum of the absolute error values (the differences between estimated and actual values) by the number of observations. This statistic is essential for evaluating the accuracy of the estimates compared to the measured values (Jamil

and Akhtar, 2017).

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|X_i - Y_i| \tag{2.21}$$

2.4.10 Stratified Sampling

Stratified random sampling is a sampling approach that categorises a population into smaller, defined subgroups called strata. In the process of stratification, these strata are established based on the shared attributes or characteristics of the individuals within the population (Levy and Lemeshow, 2013; Lohr, 2021). When the population is heterogeneous, stratified random sampling becomes more appropriate, as it enhances the precision of estimators by accounting for variability across distinct subgroups (Levy and Lemeshow, 2013). In the context of this study, stratified sampling is applied during the scalability assessment of the GW models (Subsection 3.9.2). The procedure for selecting a stratified random sample, as outlined by Lohr (2021), is as follows:

1. Divide the heterogeneous population of size N into k distinct subpopulations (strata) of sizes $N_i,\ \ i = 1, 2, \ldots, k$, such that sampling units within each subpopulation are homogeneous with respect to the characteristic under study.

2. Define the strata $N_1, N_2, \ldots, N_k$, where the total population size satisfies

$$\sum_{i=1}^{k} N_i = N$$

   The subpopulations (strata) are constructed to maximize heterogeneity between strata while ensuring homogeneity within each stratum.

3. From each stratum, select a sample of size n_i using simple random sampling. The combined samples from all strata form the complete stratified sample of size

$$n = \sum_{i=1}^{k} n_i$$

2.5 Summary of Chapter 2

This chapter reviewed the theoretical and methodological foundations for predicting NPP in the eastern Sahel, emphasising the role of spatially adaptive models. Traditional approaches, such as OLS, often fail to account for spatial heterogeneity, whereas GWR addresses this by allowing local variations in regression parameters. However, GWR's linearity assumption limits its ability to capture complex, nonlinear relationships. To overcome this, advanced techniques such as

GWRF and GWNN integrate machine learning with spatial weighting, improving predictive accuracy. The chapter also covered key concepts like spatial autocorrelation, model performance metrics, and data preprocessing methods, setting the stage for the methodological framework discussed in the next chapter.

Chapter 3

Methodology

3.1 Introduction

The fundamental focus of this research is to apply spatially informed statistical methodologies for forecasting NPP in the eastern Sahel. This study leveraged spatial information to address the challenges posed by spatial heterogeneity and complex environmental dynamics. To achieve this goal, the study integrated GWR with machine learning techniques, specifically GWRF and GWNN. This chapter outlines the methodological framework employed to achieve the study's objectives, including a description of the study area (Section 3.2), data acquisition processes (Section 3.3), data preprocessing steps (Section 3.4), exploratory data analysis (Section 3.5), model setup and implementation (Sections 3.6 and 3.7), spatial autocorrelation of model residuals (Section 3.8), and performance evaluation metrics for comparing the models in terms of predictive accuracy, computational efficiency, and spatial heterogeneity (Section 3.9). All analysis and visualisations were done with R version 4.4.1 (R Core Team, 2024).

3.2 Study Area

The Sahel is a semiarid ecological and geographic region in North-Central Africa, situated between the Sahara Desert to the north and the tropical savannas to the south. It functions as a transitional zone, commonly referred to as a steppe environment, and spans roughly 5,000 km from Senegal's Atlantic coast to Eritrea's Red Sea with an average width of about 350 km (Epule et al., 2022, 2018; Wu et al., 2022). Geographically, it lies between latitudes 10° and 20° north and encompasses 12 countries: The Gambia, Senegal, Mauritania, Burkina Faso, Mali, Niger, Nigeria, Sudan, Chad, South Sudan, Eritrea, and Ethiopia. (Rose, 2015).

The eastern Sahel region was selected as the case study for this research project (Figure 3.1). The eastern Sahel region covers approximately 6.3 million km^2 and is home to over 227 million people. In general, the eastern Sahel is characterised by its dry climate, with temperatures ranging between $25°$C and $42°$C and 200–800 mm of precipitation annually, primarily occurring from May to September. Despite the region's variable rainfall and frequent droughts, rainfed agriculture and livestock remain the primary sources of income for 80–90% of the population.

This choice was influenced by the insufficient academic focus on the eastern Sahel compared to the western Sahel, which has been the subject of considerable research regarding climate change adaptation strategies and climate stressors, including droughts, floods, and winds (Epule et al., 2022, 2018). While the western Sahel has attracted significant research attention following the catastrophic droughts of the 1970s-1980s (Epule et al., 2022), the eastern Sahel remains comparatively understudied, partly due to limited data availability and absence of historically catastrophic climate events (Epule et al., 2018). While the broader eastern Sahel comprises six countries: Niger, Sudan, Chad, South Sudan, Ethiopia, and Eritrea this study focuses only on three (Niger, Chad, Sudan) due to comprehensive data availability and the presence of a recovering rainfall deficit, which is critical for analysing spatial variability in NPP.

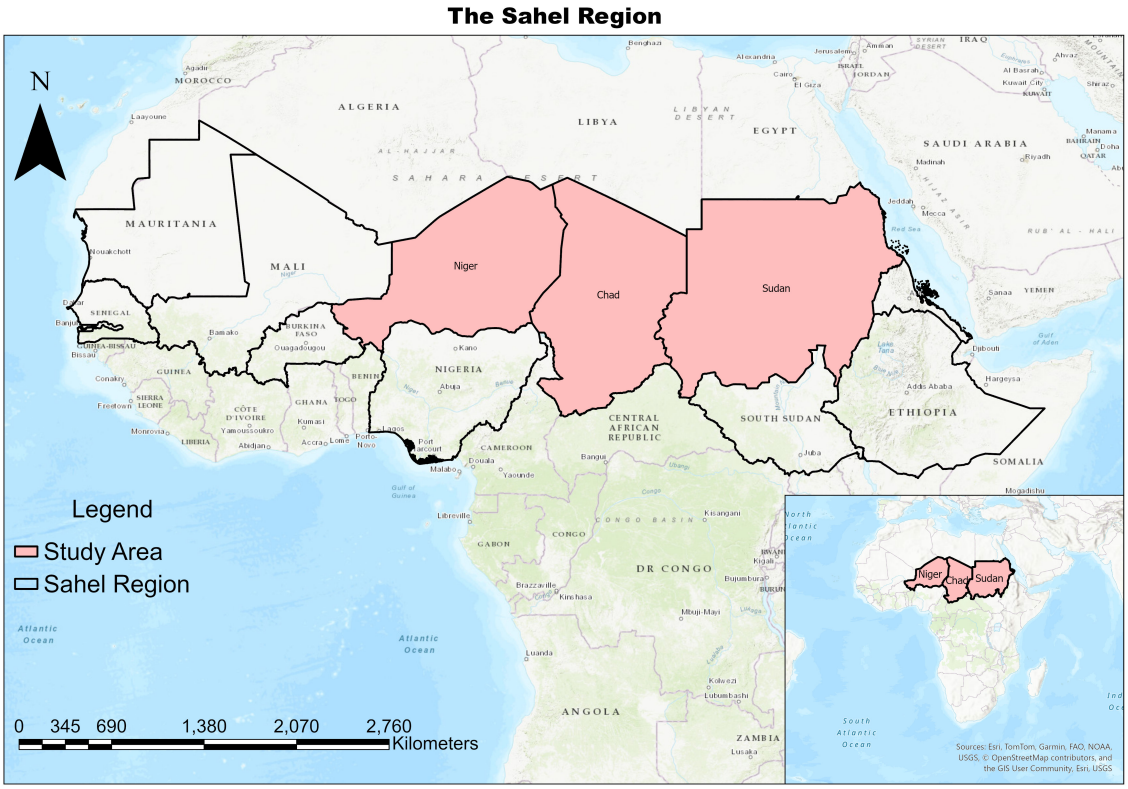


Figure 3.1: Study Area

3.3 Data Acquisition

This study used the NDVI to predict vegetation biomass, a common proxy for NPP (Xu et al., 2012). Thus, NDVI was used as the response variable, representing NPP. The predictor variables included rainfall, temperature, elevation, and soil moisture. These variables were selected due to their significant influence on vegetation growth and productivity in the Sahel region.

The computation of the NDVI was performed using cloud-free data. Global Inventory Modelling and Mapping Studies-3rd Generation V1.2 (GIMMS-3G+) data provided the NDVI dataset (Pinzon et al., 2023). Using data from the Advanced Very High-Resolution Radiometer (AVHRR), the NDVI was derived from corrected and calibrated measurements, with a spatial resolution of 0.0833 degrees (Pinzon et al., 2023), spanning from January 2019 to December 2021. The NDVI values were scaled by a factor of 10,000. To convert the NDVI values back to the standard range of -1 to 1, the values were divided by 10,000. To align with the coarser spatial resolution of the rainfall data, each monthly index was resampled to a grid size of approximately $\sim 5km$.

The Climate Hazards Group InfraRed Precipitation with Station data (CHIRPS) provided rainfall data in a gridded format for the study (Funk et al., 2015). The spatial resolution of the monthly data was approximately $\sim 5km$. Like the NDVI data, the CHIRPS data spanned from January 2019 to December 2021.

Soil moisture content data in volumetric units $(\mathbf{m^3 m^{-3}})$ for the topsoil surface were produced by the European Space Agency's Climate Change Initiative (ESA CCI) (Gruber et al., 2017, 2019). The monthly data were in grid format and had a spatial resolution of $25km$, spanning from January 2019 to December 2021. Subsequently, the spatial resolution of the data was adjusted to match the NDVI and rainfall data (approximately $\sim 5km$).

Monthly temperature data in Kelvin $(\mathbf{K})$ representing a height of $2m$ above the land surface, were used for the project. The data from January 2019 to December 2021 were acquired from the European Union's Copernicus online portal. The study made use of the latest data output from the European Centre for Medium-Range Weather Forecasts (ECMWF) reanalysis version 5 (ERA5) (Hersbach et al., 2020). Like the soil moisture data, the atmospheric temperature was resampled, transforming it from a $25km$ grid size to an approximate $\sim 5km$ grid size.

The digital elevation model (DEM) datasets were provided by the National Aeronautics and Space Administration's (NASA) Shuttle Radar Topography Mission (SRTM) (Jarvis et al., 2008). Since elevation remains constant over time, this dataset was used in its historical form

without requiring temporal updates. The data were resampled from a $1km$ grid size to an approximate $\sim 5km$ grid size.

Global Administrative Areas (GADM) data, obtained from the `geodata` package in R using the $gadm()$ function (Hijmans et al., 2024), provided administrative border shapefiles for each of these countries (Niger, Chad, and Sudan), which were used to define the study region. These shapefiles were merged into a single shapefile that represented the entire study region.

3.4 Data Preprocessing

This section outlines the data preprocessing steps, which included aggregating monthly data into a historical baseline, extracting raster data for specific locations, and imputing missing values through interpolation.

3.4.1 Data Aggregation

Since this study does not require time-series analysis, the monthly data were aggregated into a historical dataset. A stacking and averaging approach was used, where monthly values for each variable (NDVI, temperature, rainfall, and soil moisture) were combined and averaged across the entire period from January 2019 to December 2021. This method produced a single, averaged raster value for each variable at each location, effectively summarising the temporal data into historical values that represent typical conditions over the study period. The elevation data, being static, did not require aggregation.

3.4.2 Raster Data Extraction

The merged shapefile of the study area was used to extract the relevant raster data for each location. Utilising the *extract()* function from the `terra` package in R (Hijmans et al., 2022), values for NDVI, temperature, rainfall, soil moisture, and elevation were obtained from the respective raster layers. For each polygon representing a specific location within the shapefile, the mean value of each variable was computed, resulting in a collection of averaged environmental variables for every location in the study area.

3.4.3 Interpolation for Imputation

During the raster extraction process, some locations with missing values were identified. These missing values likely resulted from factors such as cloud cover, sensor errors, data gaps in

satellite imagery, and geographical limitations. Instead of excluding these points, the study treated locations with missing data as inference points. OK from the `gstat` package (Gräler et al., 2016; Pebesma, 2004) was applied to estimate values based on surrounding observations at these locations. This approach enabled the treatment of these points as new locations where predictions were made, providing estimated values and uncertainty measures for each location.

3.5 Exploratory Data Analysis (EDA)

Following the preparation of the complete dataset, an EDA was conducted to enhance understanding of the dataset. Descriptive statistics were calculated for all variables, and a correlation analysis (using *corrplot()* from the `corrplot` package in R (Wei and Simko, 2024)) was conducted to investigate linear relationships between NPP and its climatic predictors. Furthermore, the VIF was calculated to assess multicollinearity among the predictor variables, which could affect the model's accuracy and interpretation. This test was performed using the *vif()* function from the `car` package in R (Fox and Weisberg, 2019).

Additionally, the spatial properties of the dataset were analysed using mapping and visualisation techniques. The `tmap` package, specifically $tmap()$ function, was used to visualise spatial data thematically and display the spatial distributions of variables (Tennekes et al., 2023). This provided insights into the geographic variation of NPP and its environmental predictors, aiding in the identification of spatial patterns relevant to the modelling process. Due to the inconsistent value ranges of the variables, standardisation was applied, where all predictor variables were scaled to have a mean of 0 and a variance of 1 using the $scale()$ function from the R base before proceeding with model building.

3.6 Modelling with OLS and GWR

This section introduces the use of OLS regression as a global model and GWR to account for local spatial variations.

3.6.1 OLS Regression

First, a linear model was utilised to understand the global relationship between the variables within the study area. This involved constructing a regression model in R using the $lm()$ function, which stands for a linear model. Despite its simplicity, this standard ordinary regression function proved highly effective.

3.6.2 Fitting a GWR Model

The study undertook GWR in R using the `GWmodel` package (Lu et al., 2014), employing an adaptive spatial kernel approach where $bw.gwr()$ function was used to compute the optimal bandwidth for the model. The bandwidth was selected based on the $AICc$. The optimal bandwidth corresponds to the lowest $AICc$ score, confirming the best fit for the GWR model (Fotheringham et al., 2003). The Gaussian kernel was chosen as the geographical weighting function, specified through the $kernel = \text{“}gaussian\text{”}$ argument. Subsequently, the $gwr.basic()$ function was employed to fit the GWR model using the chosen spatial kernel, weighting scheme, and bandwidth. Additionally, the $F123.test$ option was used to determine whether the coefficients in the GWR model vary significantly across space (Leung et al., 2000).

3.7 Machine Learning Methods

This section describes GWRF and GWNN, the two machine learning techniques that will be used.

3.7.1 Modelling with GWRF

For each location, a GWRF model was constructed using data from neighbouring spatial units, similar to how GWR works. In this study, the optimal number of nearest neighbours was estimated using the adaptive spatial kernel technique, as the study consists of counties with irregular sizes of nearest neighbours. Before fitting the GWRF model, examples of Quiñones et al. (2021) and Grekousis et al. (2022) were used to optimise the hyperparameters of GWRF. Using the `caret` package in R (Kuhn and Max, 2008), a Random Grid Search (RGS) was employed to determine the global RF model's optimal parameters. The optimal hyperparameters were determined using a 10-fold CV approach, which assessed all possible combinations of hyperparameter values. For the global RF model, the optimal parameters (number of variables randomly sampled (mtry) and number of trees (ntree)) were kept fixed for GWRF. The model with the highest OOB R^2 was selected after models with varying bandwidth sizes were initialised. The study used `SpatialML` package (Kalogirou et al., 2022) in R.

3.7.2 Predictive Modeling using GWNN

The research implemented the methodology proposed by Hagenauer and Helbich (2022), utilising the `gwann` package in R to develop the GWNN model. To enhance the learning process, the input variables were initially transformed to achieve a mean of zero and a variance of one.

The Adam optimisation algorithm was then used to train the model, using the MSE as the loss function, a mini-batch of 50, and a learning rate of 0.001. Although an ANN can potentially have an infinite number of hidden layers, the network design employed in this study has a single hidden layer composed of six hidden neurons. Additionally, a bias neuron was introduced in both the input layer and the hidden layer. The activation function utilised for the hidden neurons was the hyperbolic tangent function.

The hyperparameters were maintained at constant values while RGS and adaptive bandwidth techniques were employed to identify an appropriate bandwidth. A 10-fold CV was used to assess the bandwidth search performance to avoid overfitting. A Gaussian kernel was used for geographical weighting, allowing for the consideration of all locations with differing levels of impact. Finally, the determination of the training iterations for the GWNN was also conducted using a 10-fold CV approach.

3.8 Spatial Autocorrelation in Residuals

To evaluate whether the models adequately account for spatial heterogeneity, local and global Moran's I were employed to calculate residual spatial autocorrelation for each model (GWR, GWRF, and GWNN) (Yuan et al., 2018) and to identify potential clustering in model residuals (Anselin, 1995). The study used the spatial autocorrelation functions available from the `spdep` package (Bivand and Wong, 2018; Roger Bivand, 2022).

3.9 Model Evaluation and Comparison

Two criteria were used to assess the model's performance: prediction accuracy and computational efficiency. CV and train-test splits were used to assess the predictive accuracy of local models (GWR, GWRF, and GWNN), and the impact of dataset size on both accuracy and computational efficiency was examined to determine each model's scalability. Additionally, a sensitivity analysis was conducted to evaluate how varying kernel functions and bandwidth settings affect the performance of each model.

3.9.1 Predictive Performance of GW-Models

Similar to other regression models, local models can be used to predict outcomes rather than to investigate spatial heterogeneity in the relationship between NPP and its variables. A 10-fold CV was conducted to assess the predictive performance of local models (i.e., GWR, GWRF, and

GWNN). Typically, CV statistics are more reliable indicators of a model's effectiveness when applied to unseen datasets (Quiñones et al., 2021). The original dataset was randomly divided into ten folds. In the model validation process, one fold is chosen to serve as the testing dataset. At the same time, the other nine folds are designated as the training dataset. This procedure is carried out ten times, guaranteeing that every fold is used as a testing dataset successively. Each time the models were trained, specifically during each fold, performance metrics such as MAE, RMSE, MSE, and R^2 were computed. Finally, the average of these metrics was subsequently reported across all 10 iterations.

An additional evaluation was performed for scatterplot visualisation, where the dataset (n = 939) was divided into 845 training data and 94 test data at random (i.e., 90:10 split). Similar to the approach of Lotfata et al. (2023), this study used 80:20 and 70:30 train-test splits to assess the model's prediction performance. The statistics showed that prediction accuracy with a 90:10 split was slightly higher. According to Helbich and Griffith (2016), the predictive performance of the 90:10 and 80:20 splits does not differ much. Since the literature on spatial statistics lacks a widely recognised train-test split ratio, the 90:10 split was chosen for this study. Global OLS, RF, and NN regression models were employed as benchmark methodologies.

3.9.2 Scalability and Computational Cost of GW-Models

To investigate the scalability of GWR, GWRF, and GWNN, this study evaluated how prediction accuracy and computational efficiency change as dataset size increases. The analysis involved training and assessing models using subsets representing 25%, 50%, and 75% of the original dataset. To ensure that the sampled data maintained the same distribution as the original dataset, stratified sampling was applied based on the categorical variable COUNTRY (Chad, Niger, and Sudan). Using the $initial\_split()$ function from the `rsample` package in R (Frick et al., 2024), each subset was generated while preserving the proportions of observations from each country.

Following the same steps from the full dataset (Subsection 3.9.1), the data were divided into training (90%) and testing (10%) sets for each subset. Models were implemented following the procedures described in Subsections 3.6.2, 3.7.1, and 3.7.2. The training time was recorded for each model to assess the effect of dataset size. Performance metrics—MSE, RMSE, MAE, and R^2—were also calculated and compared across different data sizes.

3.9.3 GW Models' Sensitivity on Spatial Weighting Parameters

This subsection further presents the sensitivity analysis conducted to evaluate how varying kernel functions and bandwidth settings influence the GWR, GWRF, and GWNN models' performance. Unlike the previous subsections, this analysis uses the full dataset ($n = 939$) without splitting it into training and testing sets. The goal is to assess how different configurations of kernel functions and bandwidths affect the models' predictive accuracy and their ability to account for spatial heterogeneity. Key performance metrics, including AIC, $RMSE$, and OOB R^2 scores, were used for comparisons across the three models. Lastly, visual representations, such as tables and graphs, were used to enhance understanding by illustrating trends and patterns observed in the dataset.

Chapter 4

Analysis and Discussion

4.1 Introduction

This chapter presents the analysis conducted to predict NPP in the eastern Sahel. It began with data preprocessing to handle missing values and prepare the dataset (Section 4.2.1), followed by exploratory data analysis (Section 4.2.2) to examine descriptive statistics, correlations, and multicollinearity. Spatial patterns of NPP and its predictors were then visualised through maps (Section 4.2.3). Next, OLS and GWR models were applied to assess spatial relationships (Section 4.3), followed by GWRF and GWNN models to capture nonlinear and complex spatial dynamics (Section 4.4). Model residuals were analysed for spatial autocorrelation (Section 4.5), and the chapter concluded with a comparison of model performance in terms of accuracy, efficiency, and sensitivity to spatial weighting (Section 4.6).

4.2 Data Preprocessing and EDA

4.2.1 Data Preprocessing

This subsection outlines how the dataset was prepared and cleaned to ensure its suitability for modelling and analysis. As mentioned in Subsections 3.4.1 and 3.4.2, the monthly climatic variables (temperature, rainfall, soil moisture, and NDVI) were aggregated from January 2019 to December 2021, using a stacking and averaging approach to compute a historical dataset for each location. Elevation data were also included but remained static due to its unchanging nature. The combined shapefile of Niger, Chad, and Sudan was used to extract relevant raster data, ensuring spatial consistency across variables.

The variable datasets were originally provided in various formats, including `.tif`, `.nc`, and `.nc4` (Table 4.1). To ensure ease of manipulation and consistency, all data were converted to the `.tif` format (RasterLayer). Subsequently, both shapefile and raster data were projected to the latest geographic coordinate system, World Geodetic System 84 (WGS84), to avoid spatial distortion and guarantee accurate analysis (Janssen, 2009).

The final dataset encompasses data from three countries, comprising 939 observations and eight variables. Among these, there is one response variable NDVI data, which is represented as NPP and four predictor variables: soil moisture (as Soil), rainfall (as Rainfall), temperature (as Temp), and elevation (as DEM) (Table 4.1). Additionally, two distinguishing variables are included: the ISO code for each country, with Niger coded as NER, Chad as TCD, and Sudan as SDN. The dataset also details the counties within each country, with Niger having 176 counties, Chad containing 427 counties, and Sudan comprising 336 counties (Figure 4.1), collectively resulting in 939 records. Each of these records includes a geometry column representing each county's polygons, constituting the dataset's final variable.

Table 4.1: Datasets and Data Sources for Study Parameters

Data	**Variables**	**Unit**	**Source**	**Format**	**Spatial Resolution**
Climate	Rainfall	$\mathbf{mm}$	CHIRPS	TIF file (`.tif`)	$5km$
	Temperature	$\mathbf{K}$	ECMWF	NetCDF (`.nc`)	$5km$
Soil	Soil Moisture	$\mathbf{m^3/m^3}$	ESACCI	NetCDF (`.nc`)	$5km$
Topography	Elevation	$\mathbf{m}$	SRTM	TIF file (`.tif`)	$5km$
Vegetation Indices	NDVI	-	AVHRR	NetCDF-4 (`.nc4`)	$5km$

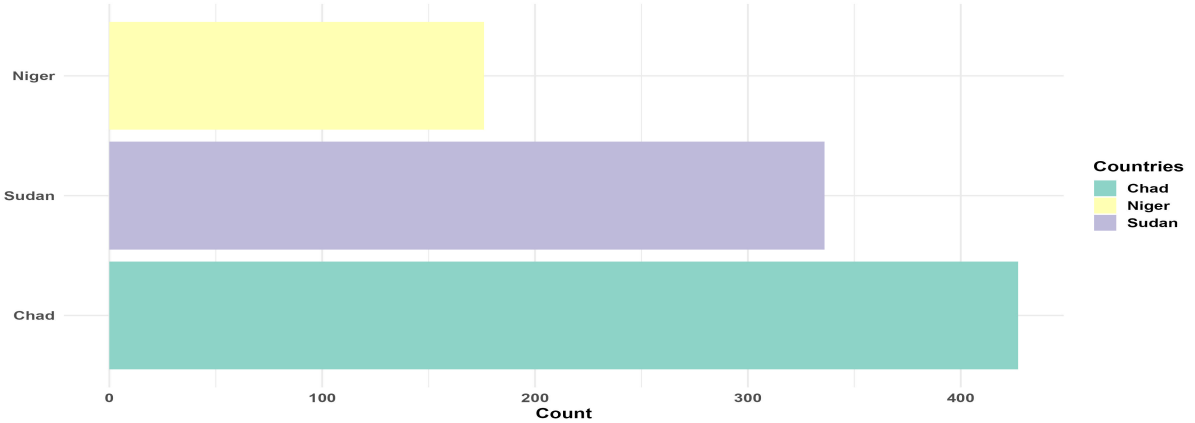


Figure 4.1: Observations Count for Each Country

4.2.1.1 Handling Missing Values Using OK

Missing values were detected in the dataset as outlined in Subsection 3.4.3. An analysis of the missing data was subsequently performed. The percentage of missing values for each variable was determined and is displayed in Table 4.2. This table indicates that several variables exhibit a significant number of missing entries. To address this issue, OK was implemented to estimate values for each variable, using the $krige()$ function from the `gstat` package in R.

The procedure began by extracting all variables that exhibited missing values. Each variable's data were then converted into a spatial points data frame (sp) to facilitate spatial analysis. Rows with missing values were removed to prepare a clean dataset. For each variable, a regular grid was created, encompassing the locations of the identified missing values. This grid allowed for the prediction of missing values.

An empirical variogram was computed for each variable using the $variogram()$ function and its clean data to assess spatial correlation. A semivariogram model, typically Gaussian, was fitted to this empirical data using the $fit.variogram()$ function to capture the underlying spatial relationships present for each variable. OK was then applied using the fitted semivariogram model to predict the values at the locations where data were missing. This interpolation method capitalised on the spatial correlation between neighbouring data points, ensuring that the missing values were estimated in a manner consistent with the observed spatial patterns.

Finally, the interpolated values for the missing points were extracted and used to update the dataset for each variable. This step ensured that the dataset was complete and maintained its spatial integrity.

Table 4.2: Missingness Percentage per Variable

Variables	**Missing Value** %
NDVI/NPP	0.96
Soil	62.19
DEM	12.99
Rainfall	0.00
Temp	23.86

Kriging results for each interpolated variable were mapped. Figures 4.2, 4.3, 4.4, and 4.5 show the interpolated map of NPP, Soil, DEM, and Temp, respectively, with associated error at each

prediction grid. Plots on the left provide a spatial distribution of the predicted values. Areas with yellow indicate higher predicted values, while areas with dark purple indicate lower predicted values. Plots on the right show kriging variance or prediction error, indicating the uncertainty associated with the prediction. Lower variance (dark purple) suggests more confidence in the predicted values, while higher values (yellow) indicate greater uncertainty.

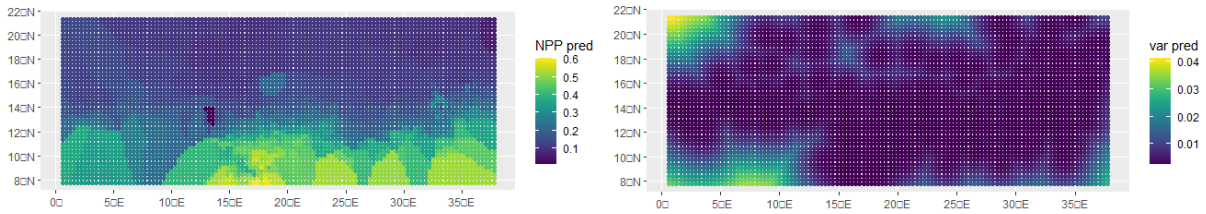


Figure 4.2: Plot for Predicted NPP and OK Error

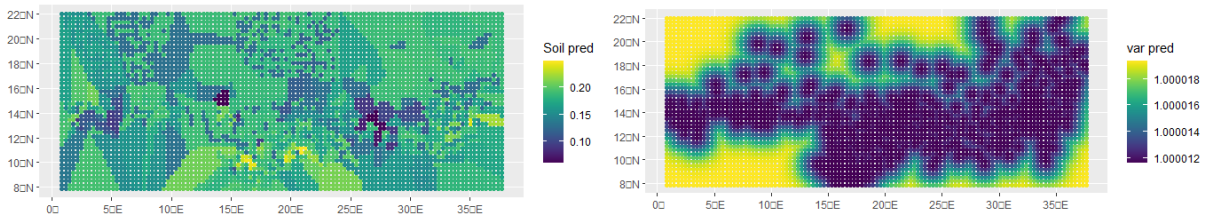


Figure 4.3: Plot for Predicted Soil and OK Error

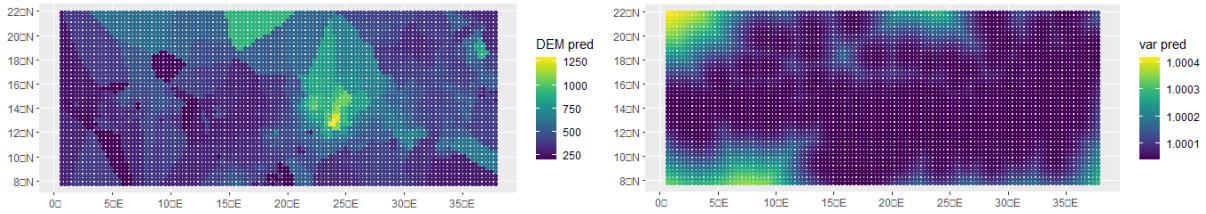


Figure 4.4: Plot for Predicted DEM and OK Error

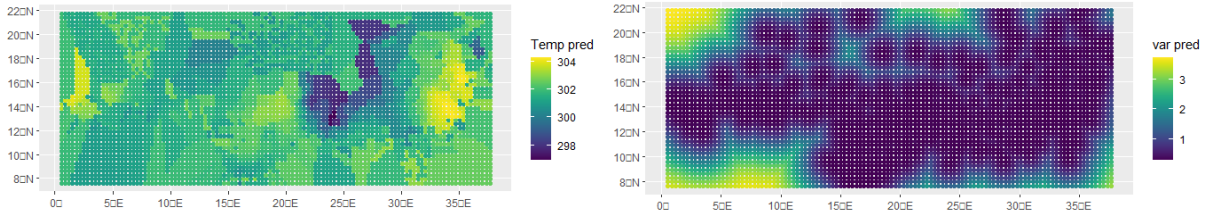


Figure 4.5: Plot for Predicted Temp and OK Error

4.2.2 Exploratory Data Analysis

This section presents the EDA carried out to understand the characteristics of the datasets and the relationships between variables. This included descriptive statistics for NPP and environmental variables, a correlation analysis to examine relationships, and a multicollinearity check.

4.2.2.1 Descriptive Statistics

An overview of the dataset utilised in the study is provided in this section, together with details on its characteristics and the relationships between the variables. Various graphical representations were employed to illustrate these properties and relationships, including boxplots, histograms, density curves, and scatterplots with regression trend lines.

Table 4.3 shows that the NPP data for the study area (2019–2021) range from 0.01 to 0.60. Its mean value is 0.27 along with standard deviation of 0.13. The statistical analysis of the NPP data reveals a slight positive skewness, as evidenced by a mean that is higher than the median, indicating the presence of higher productivity values. Rainfall, DEM, temperature, and soil moisture are the four factors that have a high correlation with NPP; Table 4.3 provides further information on their descriptive statistics.

Table 4.3: Summary Statistics of NPP-Related Features

	NPP	**Soil** $(\mathbf{m^3/m^3})$	**DEM** $(\mathbf{m})$	**Temp** $(\mathbf{K})$	**Rainfall** $(\mathbf{mm})$
Minimum	0.0069	0.0613	201.13	269.95	0.4362
Maximum	0.6016	0.2483	1297.68	304.28	117.6836
Mean	0.2722	0.169	447.12	301.77	50.8636
Median	0.2282	0.1809	411.36	301.70	44.4238
SD	0.1347	0.0311	160.6928	1.2591	30.498

The mean rainfall is 50.86 $\mathbf{mm}$ with a standard deviation of 30.50 $\mathbf{mm}$, and the range of rainfall values is 0.44 to 117.68 $\mathbf{mm}$. The mean of the DEM values is 447.12 $\mathbf{m}$, with a standard deviation of 160.69 $\mathbf{m}$ and the values range from 201.13 to 1297.68 $\mathbf{m}$. The temperature measurements recorded during the study ranged from 296.95 $\mathbf{K}$ to 304.28 $\mathbf{K}$, with an average of 301.77 $\mathbf{K}$ and a standard deviation of 1.26 $\mathbf{K}$. The average soil moisture value is 0.17, with a standard deviation of 0.03 and a range of 0.06 to 0.25.

The statistical distribution of NPP for each country is represented in Figure 4.6, showing that Chad has the highest median NPP and the greatest range, indicating more variability in its values. In contrast, Niger has the lowest median NPP and a narrow range, despite some low outliers indicating very low productivity on occasion. Sudan is in between, with a median NPP slightly higher than Niger and a moderate range, including one high outlier. This trend most likely reflects differences in environmental conditions, such as rainfall and soil quality, among the three countries.

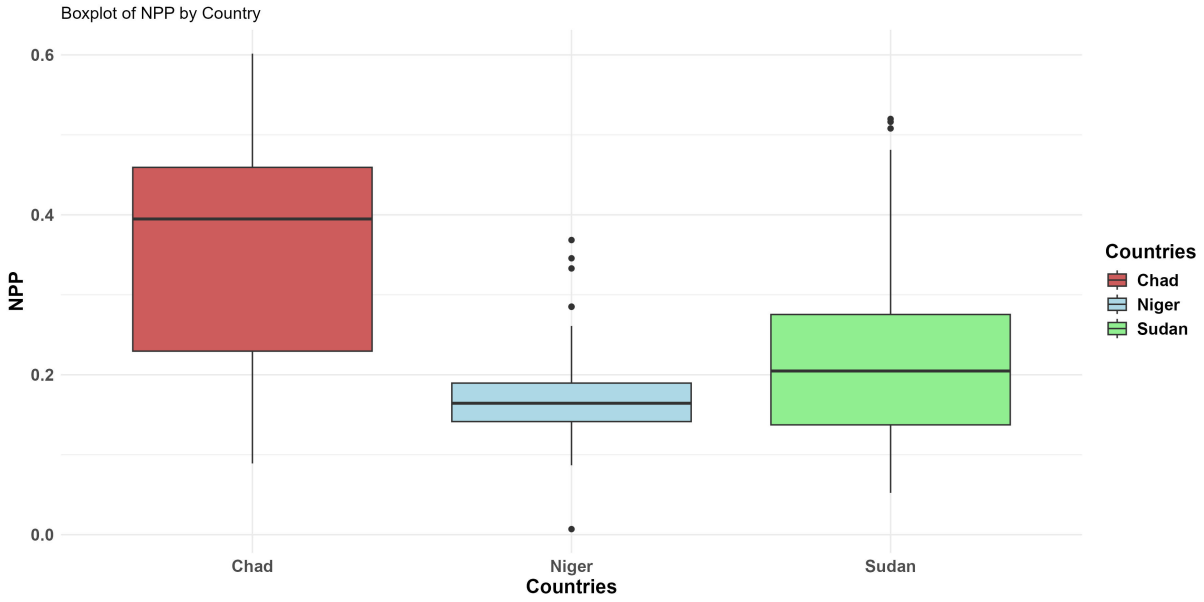


Figure 4.6: Spatial Distribution of NPP in Chad, Niger, and Sudan (2019–2021)

Figure 4.7 shows a histogram overlaid with a density curve representing NPP. The distribution of NPP appears to be bimodal, with two distinct peaks. The first and most notable peak appears at an NPP value of 0.2, indicating a high number of occurrences in this range. This indicates that a large number of data points are near this value. The second, smaller peak is seen around an NPP value of 0.4, showing another common range for NPP values, but less frequent than the first peak. As the density curve approaches both extremes, from 0 to 0.6, density decreases significantly, indicating fewer instances of extremely low or very high NPP levels.

Figure 4.8 displays the histogram for the response variables. The analysis of DEM data shows a left-skewed distribution, indicating that most elevation values are on the lower side, with a significant concentration around 500 meters. In contrast, the rainfall data have a reasonably normal distribution, though with a slight skew, and are centred around values between 40 and 60 millimetres. The soil moisture data, on the other hand, have a right-skewed distribution with a prominent peak at 0.2, indicating that the majority of data points are clustered around this

value. Finally, the temperature data reveal a slightly right-skewed distribution, with the majority of observations centred around 301 Kelvin. This skewness implies that, while the majority of temperature values are concentrated around this central point, some higher temperature values occur less frequently.

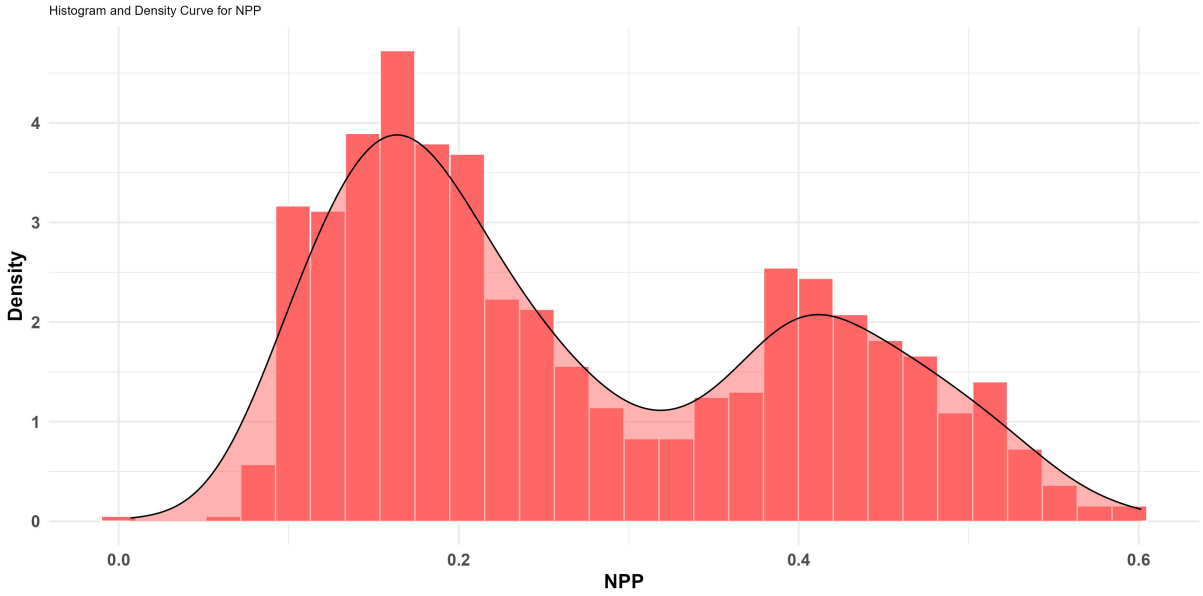


Figure 4.7: Histogram and Density Curve for NPP

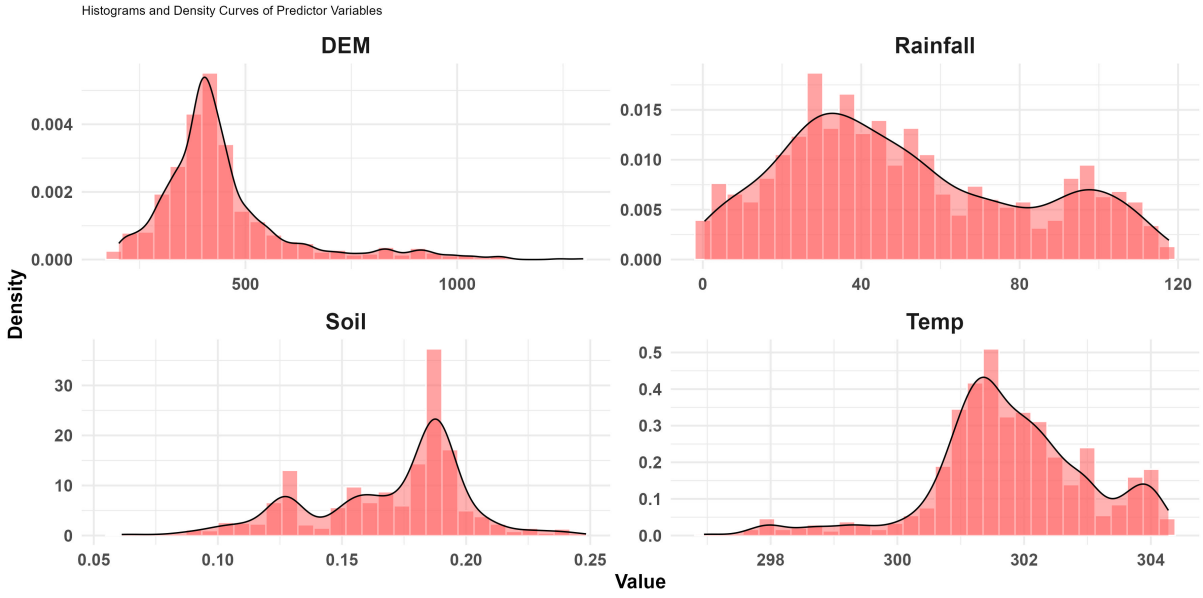


Figure 4.8: Histogram and Density Curve of Response Variables

Figures 4.9 and 4.10 provide further graphical representations. The scatterplots in Figure 4.9 show the relationship between NPP and four predictor variables: soil moisture, rainfall, DEM,

and temperature. Each scatterplot contains a linear regression trend line to aid in visualising these relationships.

For NPP versus DEM, there is a slight negative trend, indicating that as elevation increases, NPP tends to decrease. The data points are somewhat dispersed, with a concentration at lower elevations, indicating a weak relationship, as evidenced by the trend line's relatively flat slope.

In contrast, the scatterplot of NPP versus soil moisture demonstrates a positive relationship, with the trend line sloping upwards. This shows that when soil quality or type improves, NPP rises. The data points are more closely clustered around higher soil moisture values, demonstrating that improved soil conditions are related to increased productivity.

The relationship between NPP and rainfall is positive, with the scatterplot showing that as rainfall increases, NPP also tends to increase significantly. The data points are evenly distributed along the trend line, suggesting a strong relationship between these two variables and emphasising the significance of adequate rainfall for increased productivity.

Finally, the scatterplot for NPP vs temperature shows a weakly negative relationship, with the trend line sloping slightly downward. This shows that as temperature increases, NPP may decrease slightly. The data points are scattered, with some grouping at higher temperatures, suggesting that extreme temperatures may negatively influence productivity.

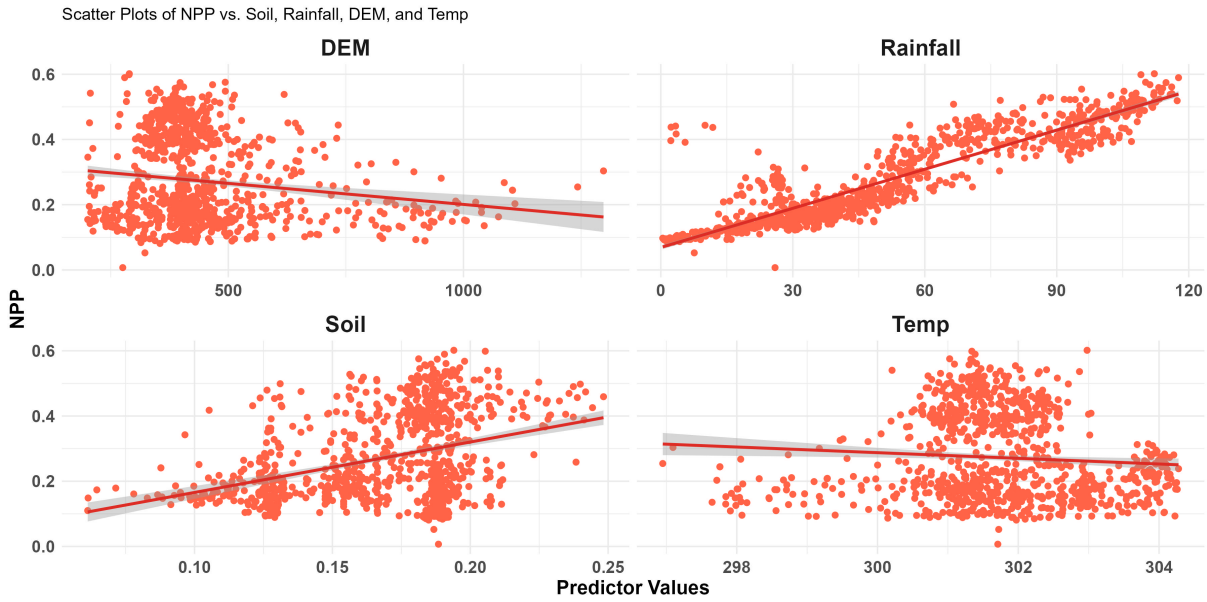


Figure 4.9: Scatter Plots of NPP vs. Soil, Rainfall, DEM, and Temp

Figure 4.10 illustrates the relationships between NPP and four predictor factors in Chad, Niger, and Sudan. Rainfall is the most important factor impacting NPP in all three countries, with a significant impact in Chad and Sudan. Temperature and elevation tend to have negative effects, particularly in Chad and Niger, while soil quality has a greater influence on NPP in Chad than in the other two nations.

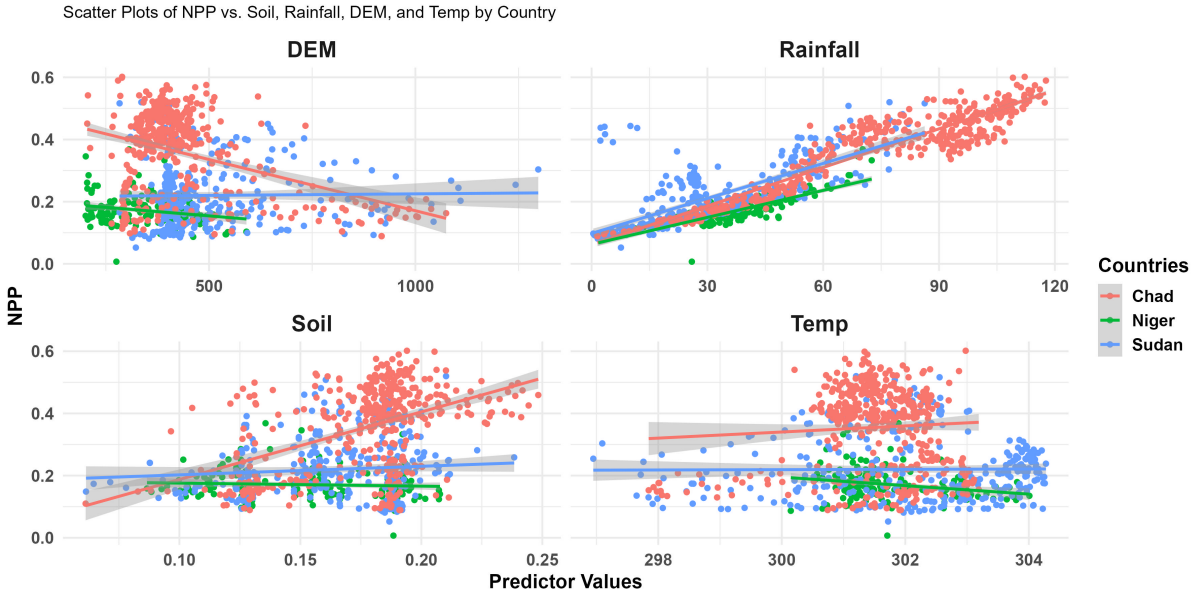


Figure 4.10: Scatter Plots of NPP vs. Soil, Rainfall, DEM, and Temp by Country

4.2.2.2 Correlation Analysis

The relationships between various environmental parameters (predictor variables) and NPP in the research region were investigated using a correlation analysis. The corrplot (Figure 4.11), which graphically depicts the Pearson correlation coefficients between the variables, provides the results.

The analysis shows a strong positive correlation between NPP and rainfall, with a coefficient of 0.90. This finding suggests that increased rainfall is linked to higher NPP, indicating that rainfall is a key factor influencing regional vegetation productivity. In contrast, the correlation between NPP and DEM is weakly negative at -0.15, indicating that elevation changes have a minor impact on productivity. Similarly, the relationship between NPP and temperature is weakly negative at -0.08, indicating minimal influence.

Soil moisture is positively correlated with NPP at a moderate level of 0.36, which suggests that improved soil conditions could facilitate greater productivity. Similarly, rainfall exhibits a moderate positive correlation of 0.30 with soil moisture, indicating that higher rainfall generally

contributes to increased soil moisture. Conversely, DEM shows a strong negative association with temperature (-0.61), implying that higher elevations are related to lower temperatures. The correlations between DEM and rainfall, and DEM and soil moisture are weakly negative, showing limited relationships.

Overall, the correlation analysis demonstrates the significant influence of rainfall and soil moisture on NPP, while also shedding light on the intricate dynamics between elevation, temperature, and these factors.

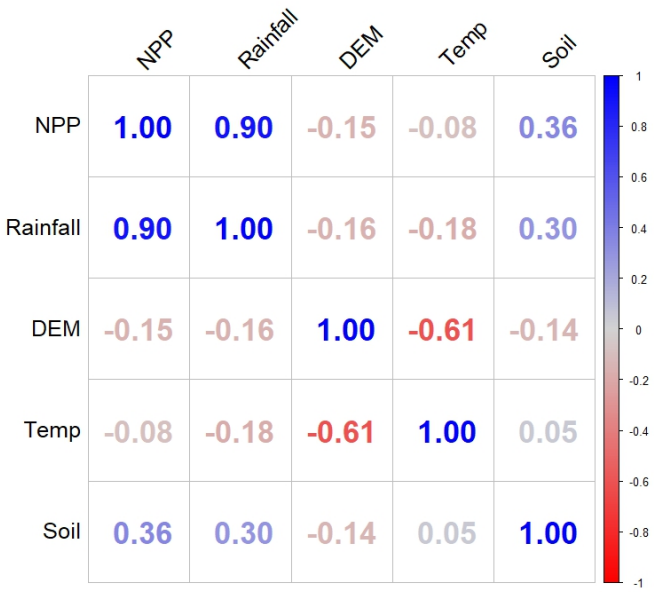


Figure 4.11: Relationship Between All Variables in the Dataset

4.2.2.3 Multicollinearity Check

Table 4.4: VIF Analysis for Predictor Variables

Variables	VIF
Soil	1.1099
DEM	1.8022
Rainfall	1.2696
Temp	1.8194

The VIF values derived from this analysis for the predictor variables are all recorded at less than 2 (Table 4.4), which is well below the generally accepted threshold of 2.5 (Craney and

Surles, 2002). This outcome suggests that the predictor variables (DEM, soil moisture, rainfall, and temperature) do not exhibit high levels of correlation. As a result, the regression coefficients for these variables are expected to be stable and reliable, which will support an accurate interpretation of their individual effects on NPP in the eastern Sahel Region.

4.2.3 Mapping Spatial Properties

This section visually represents the spatial distribution of NPP and the predictor variables across the study area through a series of refined choropleth maps and point data maps.

4.2.3.1 Choropleth Maps

Figure 4.12 illustrates that the southern regions within the study area are associated with higher NPP values (darker red), particularly in south Chad. This finding suggests enhanced productivity in these areas, likely resulting from favourable climatic conditions. Conversely, the northern regions exhibit lower NPP values (lighter shades), indicating the presence of harsher environmental conditions typical of arid and desert-like conditions.

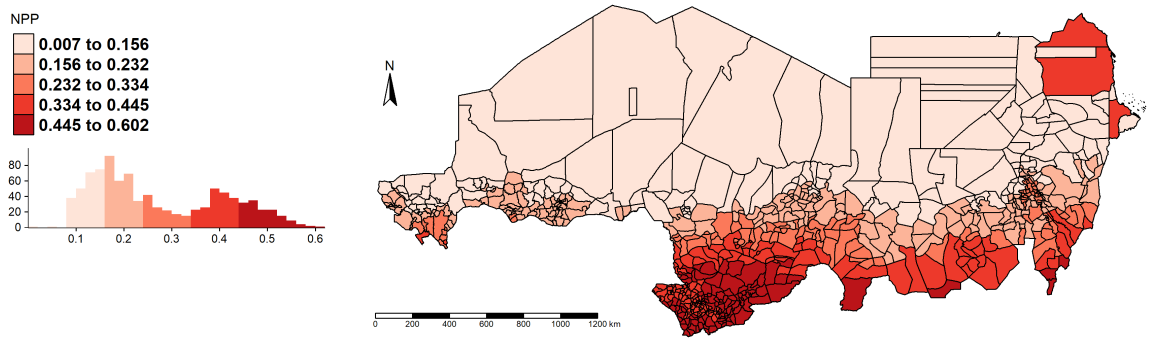


Figure 4.12: A Refined Choropleth Map of NPP

Figure 4.13 shows the spatial distribution of predictor variables, which offers a thorough visualisation of key environmental factors that influence NPP within the study area. The elevation map presents a gradient, showcasing higher elevations (785.63 to 1,297.68 meters) in the northern regions and lower elevations (201.13 to 346.47 meters) in the southern areas, significantly influencing local climate and biodiversity. Rainfall trends indicate that the southern regions receive a considerably greater amount of rainfall (87.02 to 117.68 $\mathbf{mm}$) than the northern areas

(0.4 to 21.2 $\mathbf{mm}$), which is a fundamental driver of NPP observed in the south. Moreover, soil moisture levels are higher in the southern regions (0.22 to 0.25), aligning with the increased rainfall and facilitating diverse plant life. Temperature gradients reveal warmer conditions in the south (303.20 to 304.28 $\mathbf{K}$), which further affects plant growth and the distribution of species.

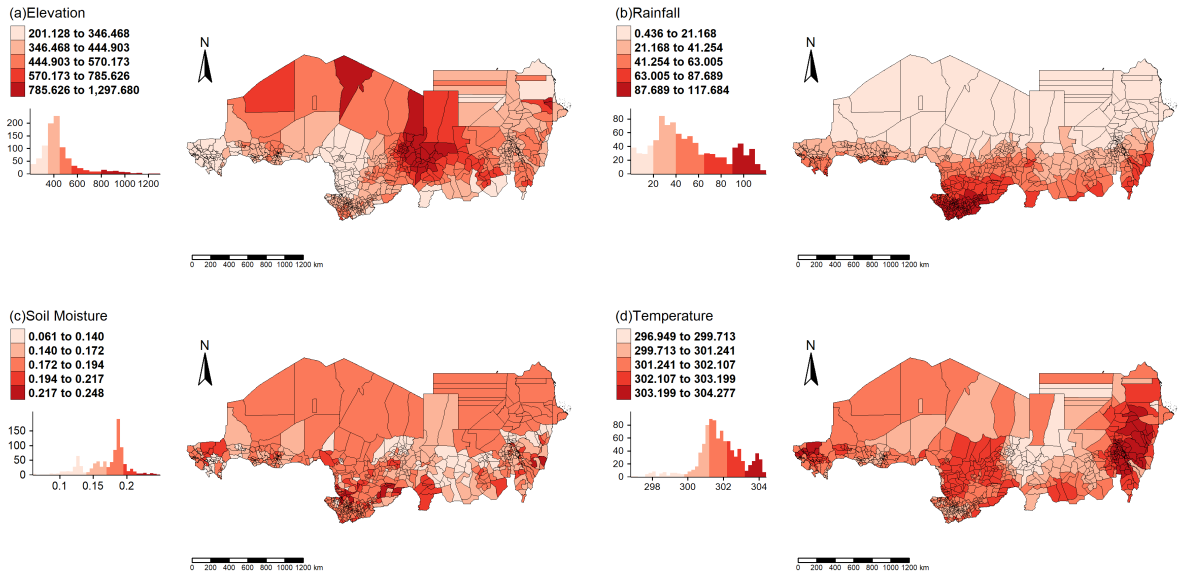


Figure 4.13: A Refined Choropleth Maps for Predictor Variables

The patterns illustrated in these maps correlate with the previously analysed NPP choropleth map (Figure 4.12). Areas characterised by high soil moisture, increased rainfall, and generally warmer temperatures in the southern areas correspond to higher NPP values. In contrast, the arid northern regions exhibit low NPP, indicative of high temperatures, low soil moisture, and limited rainfall.

The histogram located in the lower left corner in both Figures 4.12 and 4.13 offers a graphical depiction of the value distribution for each variable, namely NPP and predictors, throughout the mapped region. It illustrates the frequency of various ranges of NPP and predictor values, thereby revealing the number of areas corresponding to each productivity classification. Refer to Subsection 4.2.2.1 for further details.

4.2.3.2 Point-Based Maps

In addition to the choropleth maps, point-based centroid visualisations for NPP (Figure 4.14) and the predictor variables (Figure 4.15) provide a more granular view of spatial distribution. Each point denotes the value of the respective variable at a particular location, with shading from light yellow to dark brown indicating low and high values, respectively. For instance, the

NPP point data map (Figure 4.14) shows areas of high production in the southern regions, while the predictor variable maps (Figure 4.15) show the regional variations in temperature, elevation, rainfall, and soil moisture.

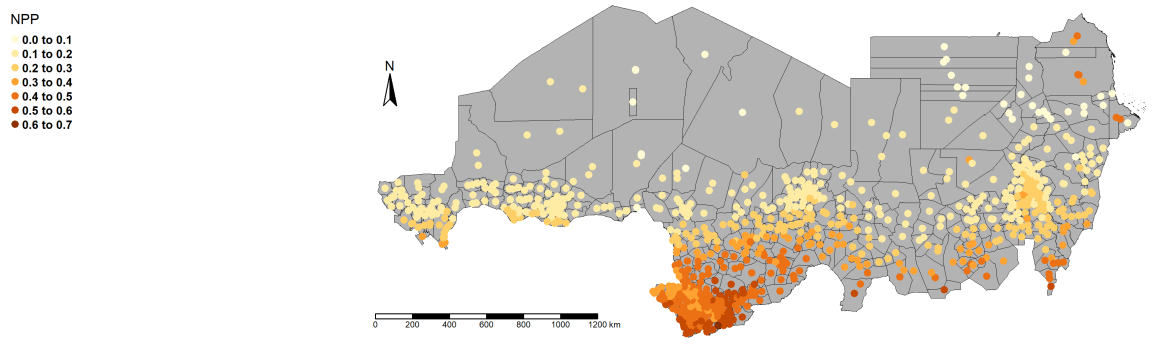


Figure 4.14: Color-Coded Point Map of NPP Observations Across the Study Area

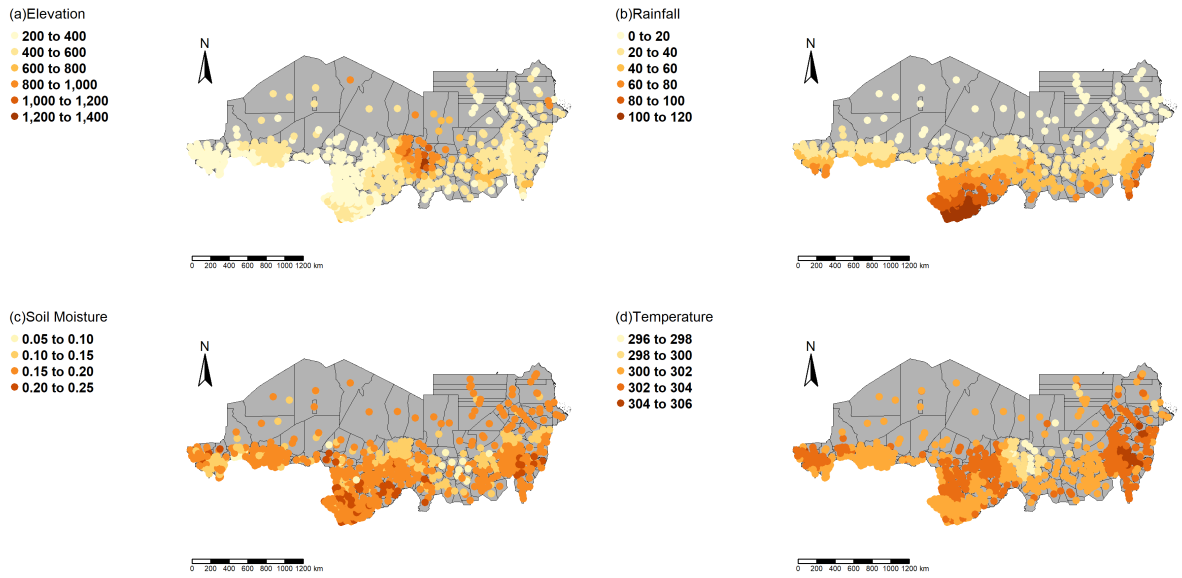


Figure 4.15: Color-Coded Point Maps of Predictor Variables Across the Study Area

4.3 OLS and GWR Analysis

This section presents the modelling results using OLS regression and GWR. It delves into model diagnostics, interpretation of results, and a comparison between the GWR and global regression model (OLS) to assess performance and spatial patterns.

4.3.1 OLS Regression

First, this subsection presents the fitting of an OLS linear regression model, evaluating its performance based on R^2 values, the significance of the predictor variables, and the strength of the relationships as demonstrated by the coefficient estimates shown in Table 4.5.

The results presented in Table 4.5 indicate that the OLS regression analysis reveals significant relationships between all predictor variables (DEM, Soil, Rainfall, Temp) and NPP, as evidenced by the low *p*-values. The coefficient estimates reveal that each variable positively influences NPP, with rainfall being the most influential. The adjusted R^2 value for this model is 0.8354, indicating that 83.54% of the variation can be explained by the model. Nonetheless, the overall model fit and individual coefficients may differ spatially when examining different parts of the study area. As a result, studying the model's standardised residuals can provide insights and lead to improvements for future models (Figures 4.16 and 4.17).

Table 4.5: OLS Results

Variable	**Coefficient**	**Std Error**	***t*-Statistic**	***p*-value**
Intercept	0.272211	0.001783	152.630	$<$ 2e-16
DEM	0.010649	0.002396	4.445	9.82e-06
Soil	0.012330	0.001880	6.559	8.96e-11
Rainfall	0.122999	0.002011	61.174	$<$ 2e-16
Temp	0.017131	0.002407	7.117	2.19e-12
Adjusted R^2	0.8354			

The study used a plot for residuals versus fitted values (Figure 4.16) to check for heteroskedasticity, which arises when the variability of the residuals is inconsistent across the levels of the fitted values. The distribution of the residuals appears to be random in relation to the horizontal line $(y = 0)$, signifying that the variance of the residuals is fairly consistent across the range of fitted values. This lack of a distinct trend in either increasing or decreasing spread implies that heteroskedasticity is not present.
The study also used a standardised residual histogram to check for normality. The histogram in Figure 4.17 suggests an approximately normal residual distribution, as it is symmetric about zero. However, some deviations, particularly in the tails, may indicate outliers.

Linear regression's implicit assumptions, such as linear relationships between x terms and y, identical normal distributions of error terms, and consistent relationships between $xk's$ and y ,

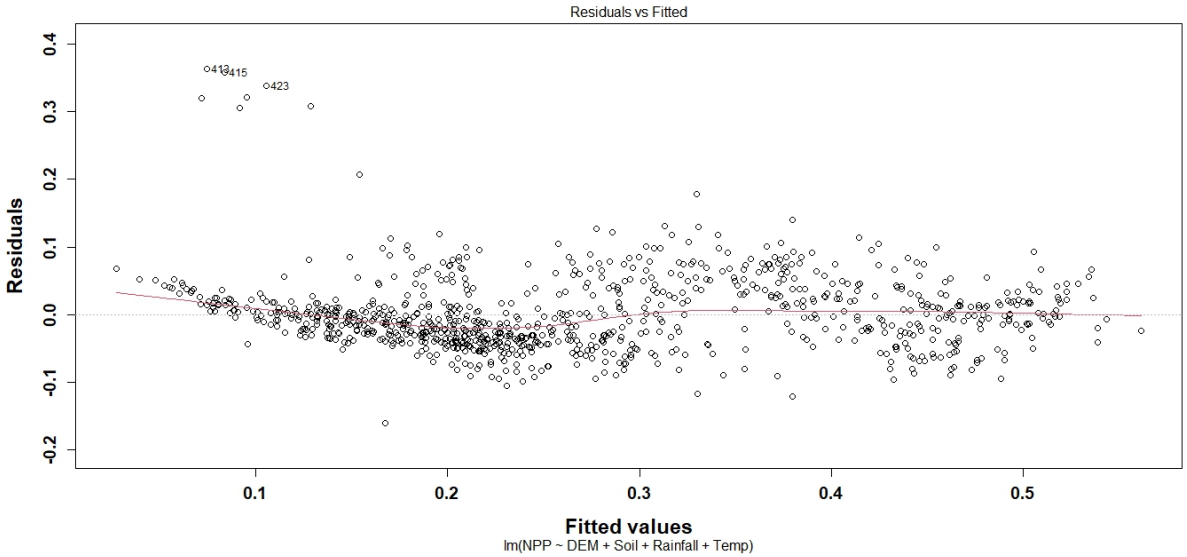


Figure 4.16: Plot for Residuals Versus Fitted Values

are often a result of wishful thinking rather than rigorous conclusions. To improve the reliability of predictions and enhance the theoretical understanding of the process under consideration, it is advisable to explore options beyond these assumptions.

The OLS regression model is typically employed to capture global relationships. Nevertheless, if spatial variations in these associations exist, the OLS model may misrepresent reality, assuming these relationships are invariant. To explore this potential misspecification, the study examined the OLS model for spatial residuals, resulting in a map (Figure 4.18) visually representing these residuals across various regions. The distribution is non-random, with light yellow and yellow areas signifying residuals close to zero, suggesting a good fit of the model in those areas. Conversely, red areas indicate higher residual values, which reflect a poorer fit of the OLS model.

Although the residual map reveals a problem with the OLS model, it does not indicate which parameters, if any, display spatial non-stationarity. An easy way to determine whether the relationships modelled by the global OLS approach are likely to remain consistent across space would be to run separate OLS models for each county or administrative region in Niger, Chad, and Sudan. However, this method would require higher-resolution data within each area, whereas the current data are only available in an aggregated form at the county level. As such, spatial models (GWR, GWRF, and GWNN), which allow relationships to vary over space, could provide a better fit (Subsections 4.3.2, 4.4.1, and 4.4.2).

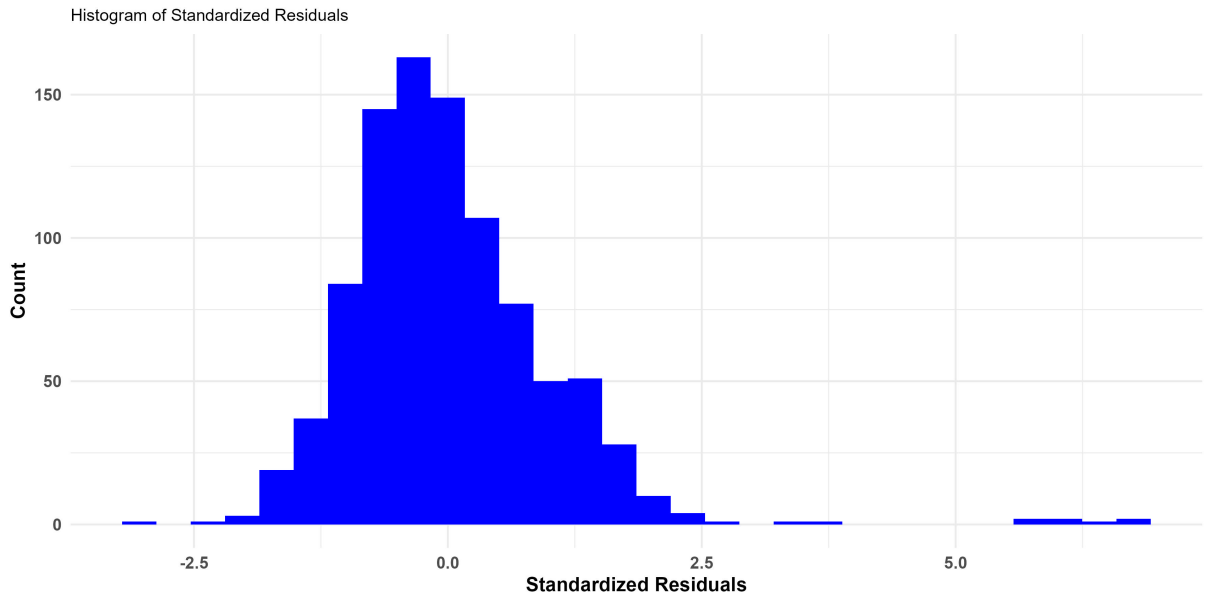


Figure 4.17: Histogram of Standardised Residuals

4.3.2 Geographically Weighted Regression

The limitations of the OLS regression model, which generates a global set of estimates, are addressed by GWR in this subsection. The primary purpose of GWR is to identify regions on the map that exhibit non-stationarity, indicating where the coefficients derived from locally weighted regression diverge from the global value (Bivand, 2017). As detailed in subsection 3.6.2, optimal bandwidth must be determined before fitting the GWR model. The results indicated that the optimal bandwidth is 20 (from AIC = -3479.35), indicating that although its size may differ, the nearest 20 observations will be used to calculate each locally weighted regression. The comparison of the GWR and OLS regression coefficient estimates, as presented in Table 4.6 and the accompanying maps in Figure 4.19, reveals significant insights into the spatial dynamics of the variables under consideration.

4.3.2.1 Summary of GWR Coefficient Estimates

The table below presents summary statistics of the GWR coefficient estimates alongside the global OLS coefficients. For intercept, the GWR coefficients range between 0.0304 and 0.4116, with a mean of 0.2582, while the global OLS coefficient is 0.2722, closely aligned with the mean and median of the GWR estimates. This suggests that while the global model predicts a single intercept value, the local model reveals spatial variation, showing that the intercept can vary significantly depending on location.

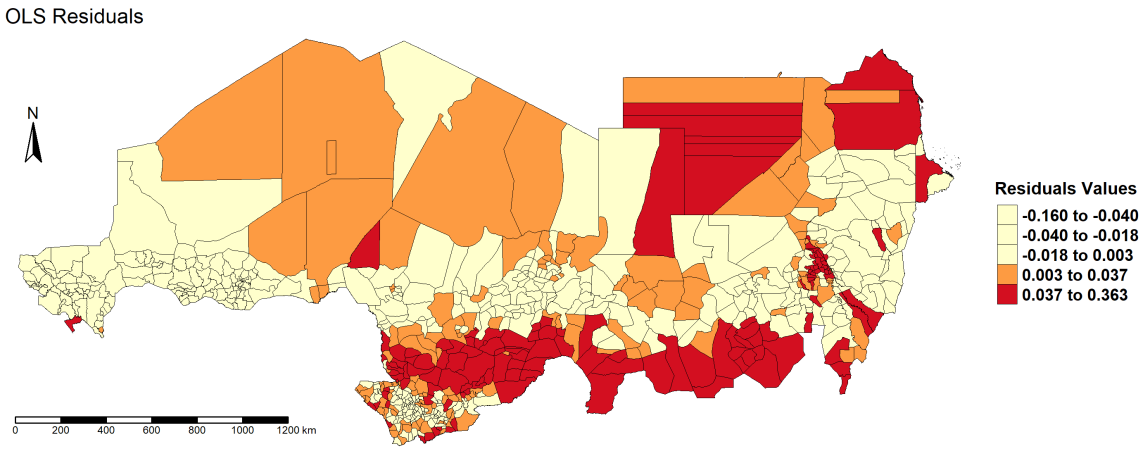


Figure 4.18: Map of OLS Residual Distribution

Regarding the predictor variables, the GWR estimates for DEM show a significant range, from -0.5019 to 0.1259, resulting in a mean of -0.0115. This shows that in some areas, DEM has a strong negative effect, while in others it has a positive effect. The global OLS estimate of 0.0106 suggests a negligible overall effect, which may mask critical local variation where elevation plays a more pronounced role. On average, the GWR model suggests a slightly negative relationship.

For the soil variable, the GWR estimates range from -0.0136 to 0.0183 with a mean of 0.0030. Although the global OLS suggests a positive relationship (0.0123), results from GWR show that soil moisture effects vary across locations, with some areas having a negative effect. GWR's median and mean values are smaller than the global OLS estimates, indicating that the OLS model may overestimate effects in some areas and obscure local variations, including potential negative influences.

Rainfall is an essential driver of productivity in the study area, and the GWR results reveal a broad range of coefficients from 0.0155 to 0.2352. The global OLS estimate of 0.1230 closely aligns with the mean of the GWR estimates (0.1280), suggesting that the overall effect of rainfall is consistently positive across the study area, with local coefficients varying significantly by location.

The GWR estimates for temperature range from -0.0350 to 0.1815, with a mean of 0.0260. The results of the GWR demonstrate that temperature has varying effects, both positive and negative, across the geographical scope of the study. The global coefficient (0.0171) is smaller than the mean of the GWR coefficients, indicating that the OLS may underestimate the positive

influence of temperature observed in specific areas, while also potentially obscuring negative effects in others.

The study further employed the F3 test to assess whether the coefficients of the explanatory variables in the GWR model exhibit significant spatial non-stationarity. The sample variance of the calculated model coefficients serves as the basis for the test (Brunsdon et al., 1996; Leung et al., 2000). Table 4.6 shows the p-values for each explanatory variable in the model resulting from the F3 test. The extremely low p-value results verify that the relationships between NPP and its predictor factors exhibit significant spatial non-stationarity. This means that the coefficients for these variables are not constant across the study area, and the relationships vary depending on the location. This justifies using GWR instead of a global OLS model, as it better accounts for spatial heterogeneity.

Overall, the GWR estimates show significant spatial variability across all variables, indicating that the local conditions have a substantial impact on the coefficients. The results in Table 4.6 show that OLS estimates provide a single average effect, which may mask important local variations. In cases like DEM, the GWR highlights areas with strong negative effects that the OLS does not capture.

Table 4.6: Coefficient Estimates from GWR and OLS Regressions

	Min.	**1st Qu.**	**Median**	**Mean**	**3rd Qu.**	**Max.**	**F3 Test (p-value)**	**Global OLS**
Intercept	0.0304	0.2182	0.2703	0.2582	0.2965	0.4116	1.50e-152	0.272211
DEM	-0.5019	-0.0209	-0.0001	-0.0115	0.0194	0.1259	4.37e-139	0.010649
Soil	-0.0136	0.0002	0.0027	0.0030	0.0063	0.0183	2.90e-10	0.012330
Rainfall	0.0155	0.0940	0.1279	0.1280	0.1583	0.2352	4.34e-146	0.122999
Temp	-0.0350	-0.0055	0.0103	0.0260	0.0522	0.1815	2.62e-183	0.017131

4.3.2.2 Spatial Distribution of GWR Coefficients

The principal strength of GWR models is their capacity to reveal spatial variations in the associations that exist between pairs of variables. Figure 4.19 maps the spatial variation of the GWR coefficient estimates. This visualisation illustrates how the relationships between different factors and NPP change across the study area.

The DEM coefficient map shows that the impact of elevation on NPP varies significantly across the study area. Parts of Sudan and Chad regions show a strong negative effect, with coefficients as low as -0.502, indicating that increased elevation reduces productivity in these areas. In contrast, light green areas (particularly in parts of Chad and Niger) exhibit weak positive or neutral effects, suggesting that elevations play a less critical role in influencing productivity in these areas. This spatial heterogeneity aligns with GWR summary statistics (Table 4.6), showing both negative and positive DEM coefficients.

The soil moisture coefficient map reveals a positive relationship between soil moisture and NPP in most areas. Green areas indicate areas where increased soil moisture significantly enhances productivity, aligning with the positive mean coefficient from Table 4.6. However, some regions (particularly in parts of Chad and Sudan) show weaker or slightly negative effects due to spatial variability, with a negative minimum soil moisture coefficient. This variability may be due to differences in soil type, land use practices, or other environmental factors that influence the availability and effectiveness of soil moisture.

The rainfall coefficient map demonstrates that rainfall is a significant determinant of NPP, especially in southern parts of Sudan and Chad, where the coefficients are strongly positive, with values as high as 0.235. These regions show a strong positive response to rainfall, indicating high productivity. In contrast, the northern regions exhibit a weaker relationship, suggesting that productivity in these areas is less dependent on rainfall. This spatial pattern mirrors the GWR summary in Table 4.6, which shows a broad range of rainfall coefficients but with a consistently positive trend.

The temperature coefficient map reveals that temperature effects on NPP vary across the study area, with some areas showing positive effects, such as the southern part of Chad and a small part of Sudan (green areas), suggesting that warmer temperatures promote productivity in these areas. Conversely, parts of the northern region show neutral or slightly negative effects, suggesting that higher temperatures may hinder productivity, especially in areas with limited water availability. The negative and positive range of coefficients indicate that temperature has both beneficial and detrimental effects depending on the location, which shows spatial variation, as also reflected in Table 4.6.

Overall, the maps reveal spatial variations in the strength of the relationships between changes in variables and NPP. The maps depict highly localised trends in the variation of these relationships with the response variable. The maps also highlight the spatial variability and non-stationarity present in the association between the target and predictor variables, as well as the alterations

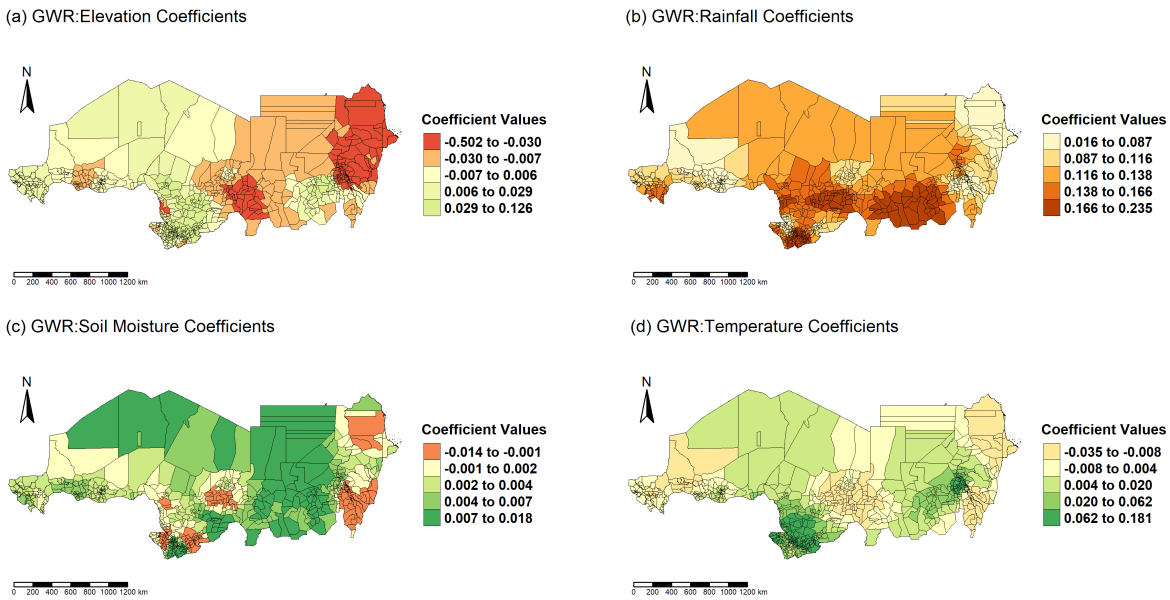


Figure 4.19: Maps of the GWR Coefficient Estimates

in the direction of the relationships between covariates and the response variable. This suggests that the causal process between NPP and the predictor variables may not be spatially consistent.

4.3.2.3 The R-Squared Map

This subsection examines the overall model fit by mapping the R^2 values for the calculated local regressions, as illustrated in Figure 4.20. The map reveals a robust in-sample model prediction with R^2 ranging from 0.887 to 0.988 in a relatively large part of the study area (i.e., Niger, Chad, and some parts of Sudan). This indicates that the GWR model accounts for a significant extent of the variance in NPP in these areas, suggesting that it accurately reflects the fundamental relationships present. In contrast, the northern-east part of Sudan shows poor prediction with R^2 values as low as 0.186. In these areas, the predictor variables explain only a small fraction of the variance in NPP, suggesting that the relationship between NPP and predictor variables may be complex, less consistent, or influenced by other unmeasured factors.

In conclusion, the R^2 map above is useful for evaluating the GWR model's performance across regions, highlighting effective areas and those needing improvement. However, GWR is limited by the linearity assumption of OLS regression (Grekousis, 2020), making it less effective for complex, nonlinear relationships. Regions with low R^2 values indicate this limitation. Future research should explore advanced spatial machine learning techniques like GWRF and GWNN to better capture nonlinear relationships and improve model accuracy.

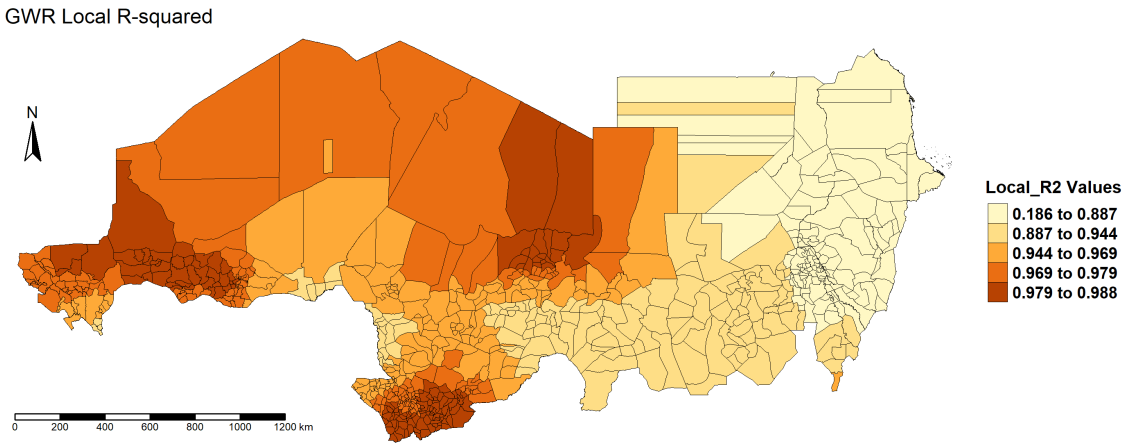


Figure 4.20: Map of R^2 Values from the GWR Model

4.4 Machine Learning Methods

This section explores the application of machine learning techniques to model the relationship between NPP and its drivers. Specifically, GWRF and GWNN were employed to capture potential nonlinear associations and spatial variations in these relationships. The analysis includes an examination of feature importance (for GWRF) and the weights associated with the network layers (for GWNN).

4.4.1 Geographically Weighted Random Forests

The GWRF model, characterised as a nonlinear, non-parametric approach, does not require multicollinearity consideration and is capable of analysing all predictor variables without any initial screening (Luo et al., 2021). The GWRF model was trained with the following variables, as described in Section 3.7.1: bandwidth = 24, "mtry" = 3, and "ntree" = 1,000. The data were used to train both the global and local RF models to investigate variations in feature importance caused by data distribution.

4.4.1.1 Nonlinear Associations

To investigate the nonlinear relationships between the predictor variables and NPP, partial dependence plots (PDPs) from the RF model were extracted, as shown in Figure 4.21. These plots present the expected response of the target variable in relation to the relevant input features,

thereby uncovering whether the relationship is linear, curvilinear, monotonic, or of an different nature (Friedman, 2001). The observations derived from Figure 4.21 suggest that the majority of associations are nonlinear and take on complex shapes. Although certain variables may present linear relationships, these are generally confined to specific ranges. A notable example is the positive linear relationship between DEM and NPP within the range of -1.5 to -0.5, where initially, as DEM increases, the average prediction of NPP increases slightly, reaches a peak, and then declines.

Within the range of -4 to 1, NPP and temperature exhibit a linear positive relationship, though the impact of temperature on NPP decreases beyond this threshold. Likewise, fluctuations can be observed in the soil moisture and NPP relationships. Furthermore, NPP seemed to be positively associated with rainfall. Collectively, these results demonstrate that practically all relationships are nonlinear, which emphasises the need to adopt nonlinear regression models, including GWRF, rather than relying on conventional linear regression.

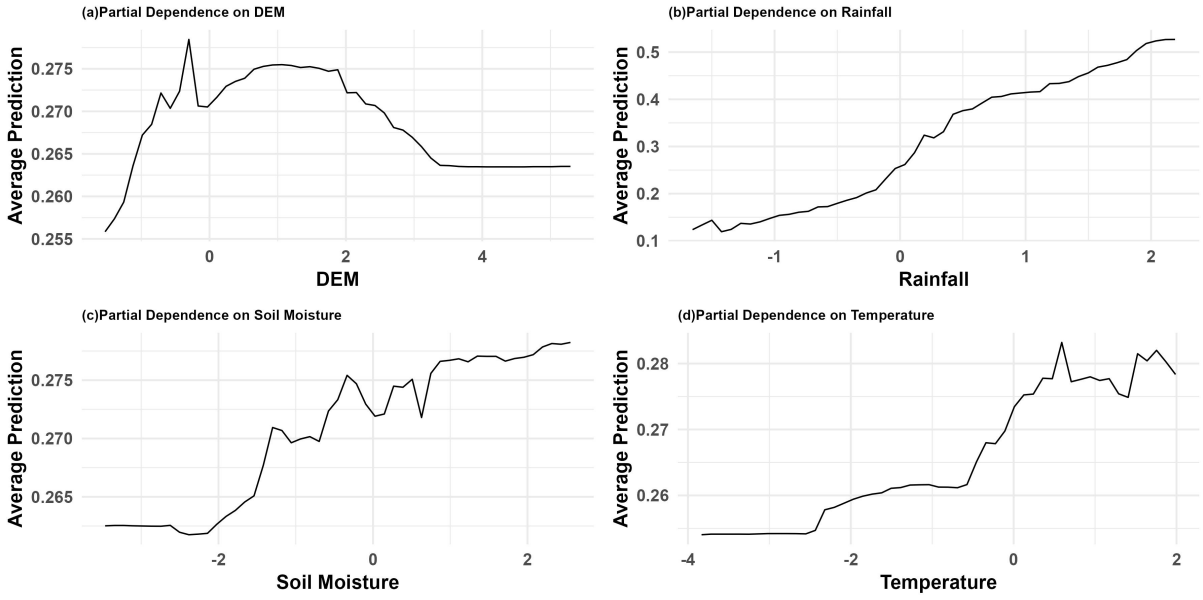


Figure 4.21: Visualising Nonlinear Effects of Covariates with PDPs

4.4.1.2 Feature Importance

In this analysis, the GWRF model was applied to investigate the contributions of covariates through global permutation-based feature importance. The local feature importance, when predictors are randomly permuted, is represented by the average increase in Mean Squared Error (IncMSE). This assessment was derived from the OOB error.

GWRF performs better (R^2 = 0.9376, OOB MSE = 0.001) than RF (R^2 = 0.8985, OOB MSE = 0.0018) (Table 4.7). Like other local metrics, GWRF relies on a subset of data (a neighbourhood defined by the bandwidth) to build nonlinear regressions for each county. Despite the relatively small size of these subsets, the total OOB MSE is less than that of the global model. This suggests that generating and analysing multiple locally defined RF can lead to more accurate predictions compared to a single global RF model. In the context of permutation-based feature importance, rainfall emerges as the predominant predictor by a significant margin in the RF model, followed by soil moisture and temperature, which have similar importance, with soil moisture slightly higher. DEM has the lowest importance among the four variables in the RF model. Figure 4.22 confirms these findings, showing rainfall as the most important global variable.

Table 4.7: Summary Results of RF and GWRF Models

Rank	**RF**		**GWRF**				
	Variables	Global Feature Importance	Variables	Local Feature Importance (IncMSE)			
				Min	Max	Mean	Std
1	Rainfall	14.5951	Rainfall	4.395e-05	0.3379	0.0175	0.0331
2	Soil Moisture	0.8620	DEM	7.335e-05	0.3216	0.0075	0.0212
3	Temperature	0.8408	Temperature	2.523e-05	0.0751	0.0056	0.0084
4	DEM	0.5900	Soil Moisture	1.661e-05	0.2066	0.0054	0.0173
R^2	0.8985		0.9376				
MSE	0.0018		0.001				

As shown in Table 4.7, the GWRF model demonstrates an average positive IncMSE, revealing that the majority of tracts with local significance to NPP experience a positive variation in local importance. Higher values reflect a greater degree of importance. Furthermore, the GWRF model's determinants are ranked differently in terms of relevance than the global RF model. For example, the RF model (greater IncMSE) placed soil moisture second, whereas the GWRF model (lower IncMSE) placed it fourth. Likewise, the RF model ranks DEM fourth, while the GWRF model ranks it second (Table 4.7). It is important to highlight that the mean value of the entire local feature importance distribution determines the GWRF approach's ranking. Furthermore, Figure 4.23 provides the visualisation of the computed average local effects of each explanatory variable on NPP in the GWRF model, the same as observed in Table 4.7, both

showing the same outcome, with rainfall remaining the most influential variable.

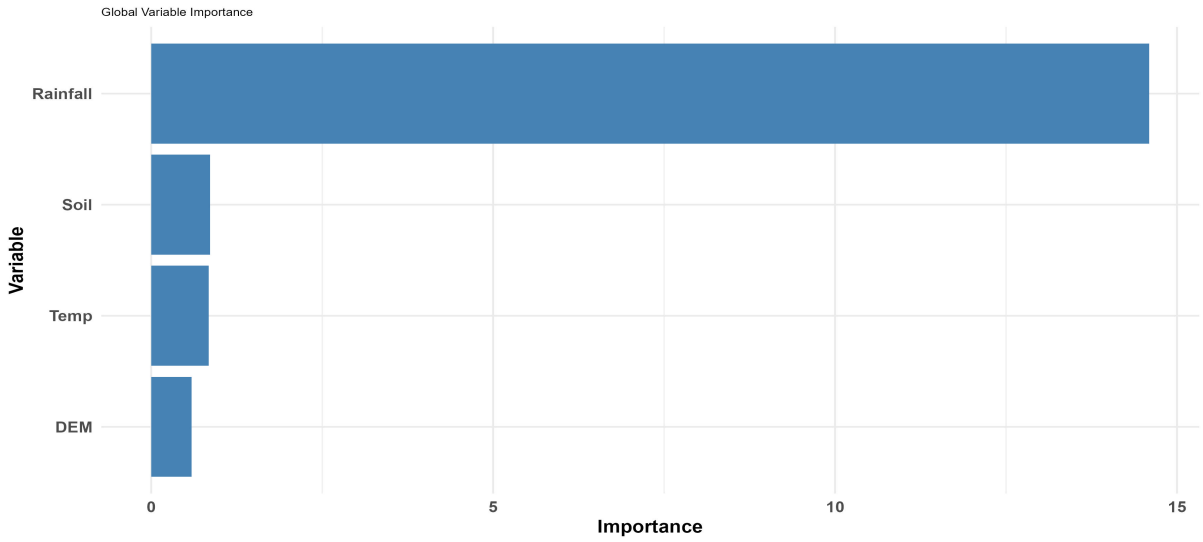


Figure 4.22: Global Feature Importance (IncMSE)

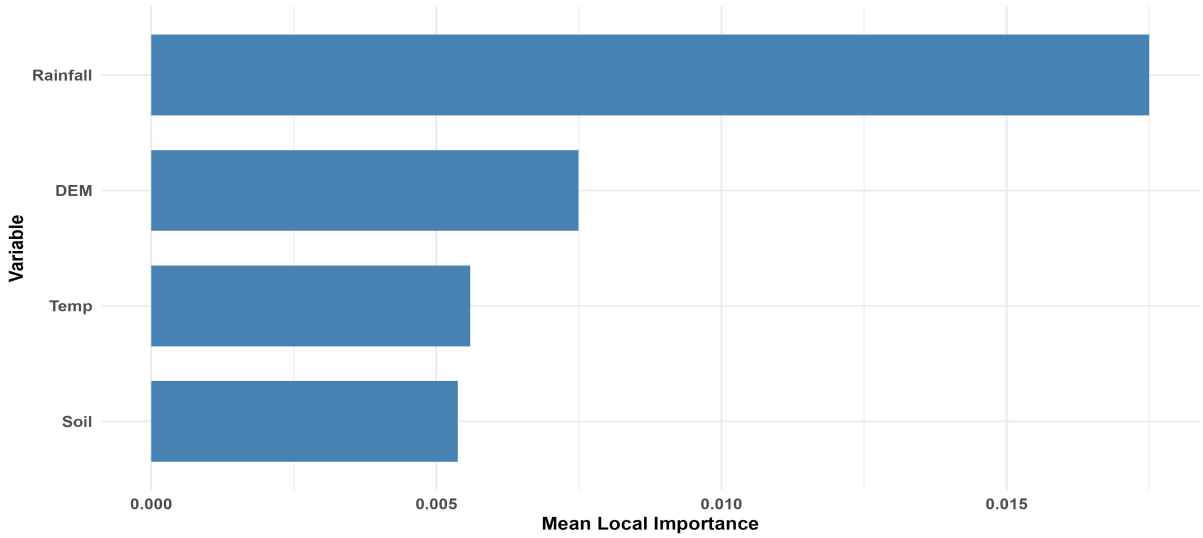


Figure 4.23: Mean Local Variable Contribution of Four Factors to NPP (GWRF)

Furthermore, the local version of GWRF enables the mapping of each variable's importance at individual spatial units (Figure 4.24). While the improvement in OOB MSE compared to the global RF appears minor, it significantly influences the accuracy of predictions and enhances the comprehension of the local factors affecting NPP. This level of spatial insight is absent in the global RF, which does not account for spatial variation. Therefore, it is crucial to focus on how variable importance changes across space rather than merely comparing the overall importance ranking from the summarised GWRF model (Table 4.7).

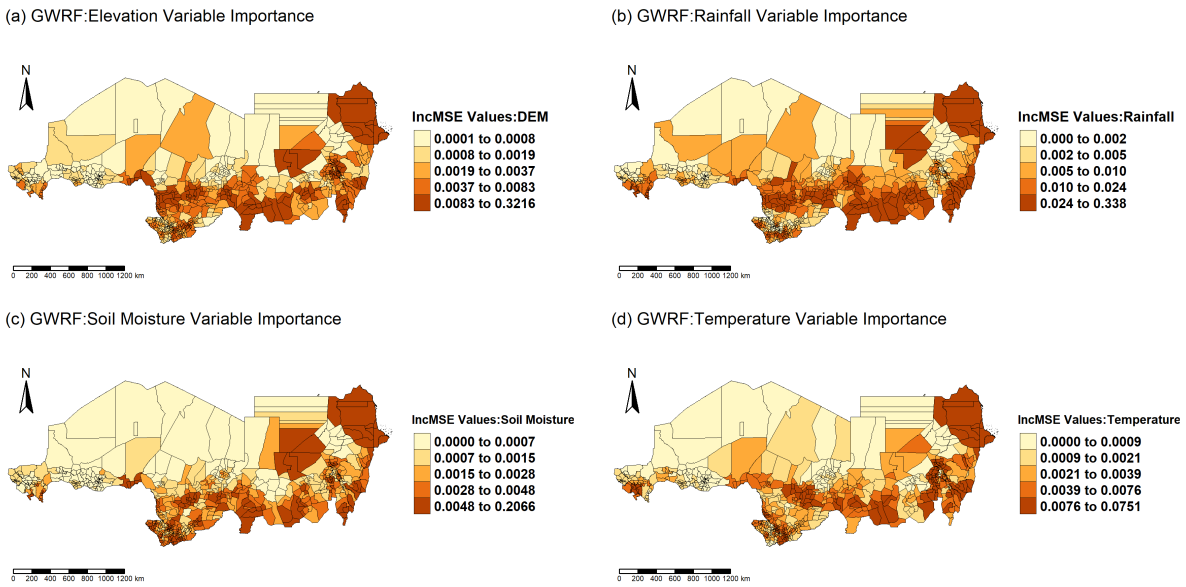


Figure 4.24: Local Feature Importance Maps

The DEM exhibits its highest importance in the southern region of Chad and the northeastern part of Sudan. Overall, DEM has the most considerable influence on the majority of Sudan. Soil moisture, on the other hand, shows varied importance across the region, with notable importance in most parts of Sudan and some in southern Chad. Rainfall is a significant factor affecting NPP across both the southern and northeastern regions, which cover both Sudan and Chad. The temperature map has lower overall IncMSE values than the other variables, implying that temperature plays a more consistent and less influential role throughout the study area. However, its importance is mostly concentrated in the southern and central regions.

Overall, Figure 4.24 demonstrates the importance of DEM, soil moisture, rainfall, and temperature determinants in a few southern and northeastern neighbourhoods on NPP. This framework aids in recognising the combinations of variables that have a significant relationship with NPP across various neighbourhoods, thereby serving as a crucial resource for policymakers in creating targeted place-based initiatives.

4.4.1.3 R-squared Distribution

The local fitting performance of the GWRF model was assessed by mapping the local R^2 values. The distribution of local R^2 values for the GWRF across the study area is displayed in Figure 4.25. This figure categorises the R^2 values into five distinct ranges ($\leq$ 0.2, (0.2, 0.4], (0.4, 0.6], (0.6, 0.8], $>$ 0.8), and the corresponding percentages of counties within these ranges are detailed in Table 4.8. From the map, it is evident that the GWRF model performs well in

most regions, especially in the study area's south, west, and southeast, where local R^2 values exceed 0.6, with several areas showing R^2 values greater than 0.8. These regions illustrate the remarkable capability of the GWRF model in accurately predicting NPP. In contrast, certain northern regions (particularly in Sudan), parts of southern Chad, and central Niger exhibit lower R^2 values (≤ 0.4), indicating less accuracy in those areas.

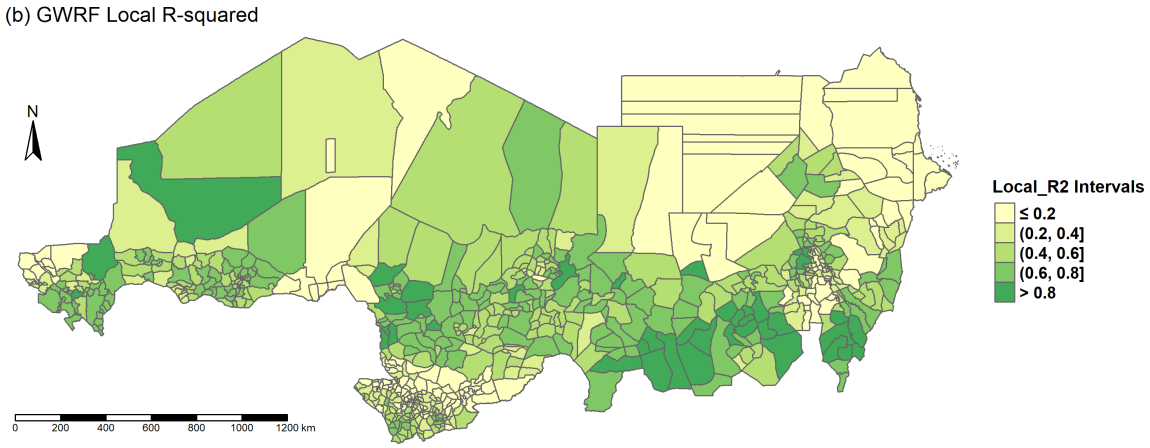


Figure 4.25: Map of R^2 Values from the GWRF Model

Table 4.8: Local R^2 Statistics for GWRF

The Local Value of R^2	**% of counties**
≤ 0.2	22.04
(0.2, 0.4]	16.30
(0.4, 0.6]	27.69
(0.6, 0.8]	27.26
> 0.8	6.71

Table 4.8 describes the statistics of the local R^2 values for the GWRF model. In 22.04 % of counties, the R^2 values are below 0.2, suggesting lower model accuracy in those areas. However, in 61.99 % of the counties, the R^2 exceeds 0.4, showing good prediction power in most areas. This indicates that the proposed local model, GWRF, achieved better results in evaluating the correlation between the predictors and NPP in most of the study areas.

4.4.2 Geographically Weighted Neural Network

In a manner similar to the GWR model, which allows for the visualisation of estimated coefficients on maps to examine the spatial variation of relationships (Sub-section 4.3.2.2), the connection weights of the GWNN that link the hidden and output layers can also be represented on maps. According to Subsection 3.7.2, the optimal value for the bandwidth "bw" was determined to be 4. To demonstrate, a bandwidth of 4 means that the distance to the fourth neighbour is equal to the kernel bandwidth for parameter estimation at a specified location. It is essential to understand that a small bandwidth does not inherently indicate a 'very local' context, as the data density influences the bandwidth. For instance, in cases of sparse data, a small bandwidth will extend over a larger geographic area than it would in instances of dense data. In addition, the iteration that achieved the best average performance across all folds was 10424. The minimum RMSE value of 0.0430 served as the basis for determining both the number of iterations and the bandwidth.

4.4.2.1 Input to Hidden Layer Weights

The heatmap shown in Figure 4.26 illustrates the degree of connections between the input layer and the hidden layer, thereby indicating how each input feature influences the activation of the model's hidden neurons.

A review of the soil feature reveals that the weights are typically near zero, demonstrating that this feature contributes very little to the activations of the hidden neurons. Specifically, Hidden 4 reveals a small negative weight of -0.111, suggesting a slight inverse relationship with this feature. On the other hand, rainfall has substantial positive weights with Hidden 1 (0.65) and Hidden 3 (0.662), signifying that an increase in rainfall significantly enhances the activations of these hidden neurons, underscoring the relevance of rainfall in the predictive task. However, Hidden 4 has a strong negative weight with rainfall (-0.875), suggesting an inverse influence.

The temperature feature demonstrates varied effects across hidden neurons. It has a pronounced positive impact on Hidden 1, recorded at 0.599, which highlights its importance in this context. Weaker positive influences are also seen in Hidden 4 (0.27) and Hidden 5 (0.109). In contrast, Hidden 2, 3, and 6 present negative weights, suggesting that an increase in temperature could negatively influence these neurons. Moreover, DEM exhibits the highest positive weight with Hidden 4 (0.655), indicating a strong influence of DEM on this particular hidden layer. Conversely, Hidden 2, 5, and 6 display smaller and negative weights, suggesting that the impact of elevation is less significant and may even result in reduced activation of these neurons. A notable observation regarding bias is the presence of strong positive weights associated with

Hidden neurons 2, 5, and 6, which indicates a considerable and persistent influence on these specific hidden neurons.

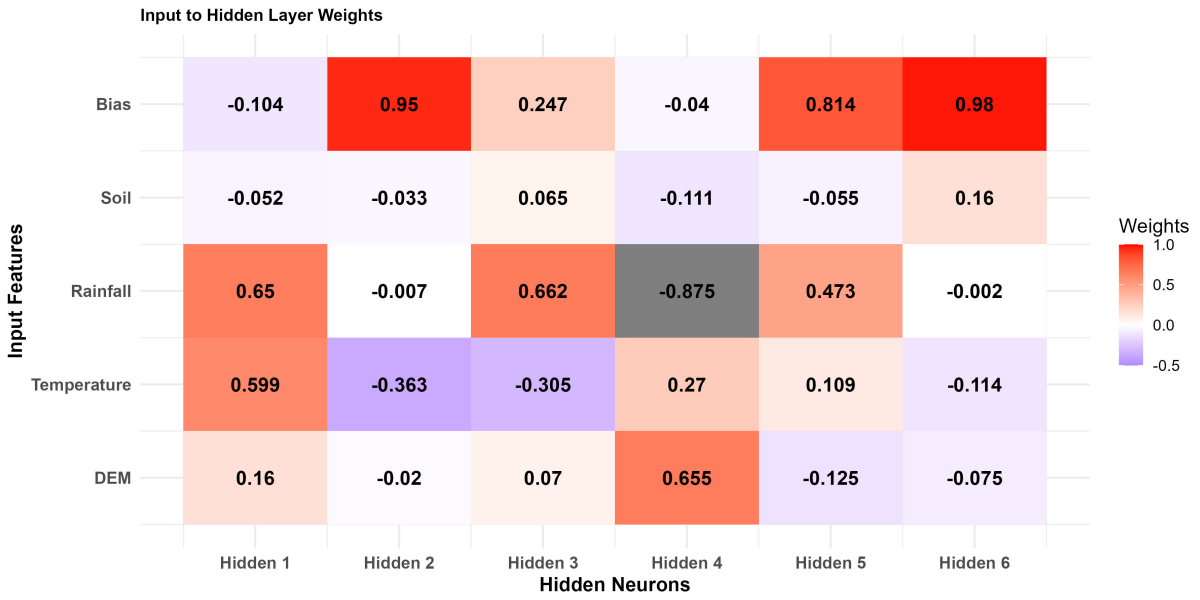


Figure 4.26: Input to Hidden Layer Weights

Overall, a thorough examination of these weights reveals how the hidden layer alters the input features, which in turn impacts the final predictions of the model. For example, the importance of rainfall and DEM in some neurons shows that these features greatly impact the NN's spatial predictions.

4.4.2.2 Hidden to Output Layer Weights

Within the framework of GWNN, the output generated by each hidden neuron is designated to a particular geographic location. This configuration enables the measurement of the spatial distance that separates the observations from the positions of the output neurons. In Figure 4.27, the coefficient weights for each observation are depicted, facilitating the examination of the connection strengths between hidden and output neurons, where the total number of output neurons matches the overall number of observations. Each map serves to represent the output of a hidden neuron, which results from a nonlinear transformation applied to a linear combination of the input variables.

Hidden 1 to 3 maps have primarily positive connection weights, with differing intensities across various locations. The central and southern regions, in particular, show higher weights, which

imply advantageous conditions for NPP in these areas. Conversely, Hidden 4 reveals a combination of both negative and positive weights, with the negative weights predominantly located in southern Chad and central Sudan. This could represent unfavourable conditions restricting productivity, such as high temperatures or low soil quality. Hidden 5 is positively weighted in central Sudan, while Hidden 6 demonstrates a more consistent and balanced influence across the map. The regions with higher weights in these neurons suggest inverse effects on the output across the geographic space.

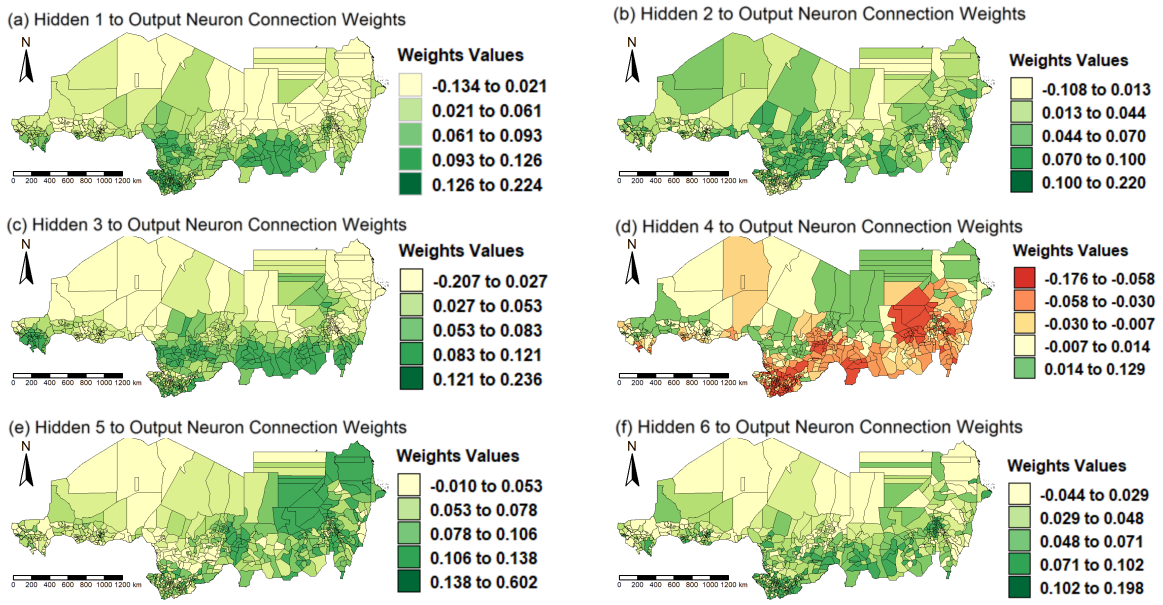


Figure 4.27: GWNN Connection Weights (Hidden to Output Neurons)

The study further produced a heatmap to show the aggregated weights that visualise the overall impact of each hidden neuron on the output layer (Figure 4.28). The heatmap illustrates that Hidden 5 plays the most crucial role in the output layer's prediction. This means that the features or patterns recorded by Hidden 5 have the strongest alignment with the target variable, potentially uncovering significant predictors in the data that drive the models' output. Following Hidden 5 in importance are Hidden neurons 1, 2, 3, and 6. In contrast, Hidden 4 has the lowest aggregate weight, suggesting it either failed to capture meaningful patterns or that those patterns are unnecessary for the final output calculations.

According to the GWNN maps (Figure 4.27), the model effectively captures spatial relationships within the dataset and demonstrates adaptability to the varying environmental conditions present in the study area. Each map of the hidden neurons reveals location-specific weighting, suggesting that the GWNN has learned detailed spatial patterns that could influence NPP. However, despite this evidence of spatial learning, the GWNN's complexity presents considerable

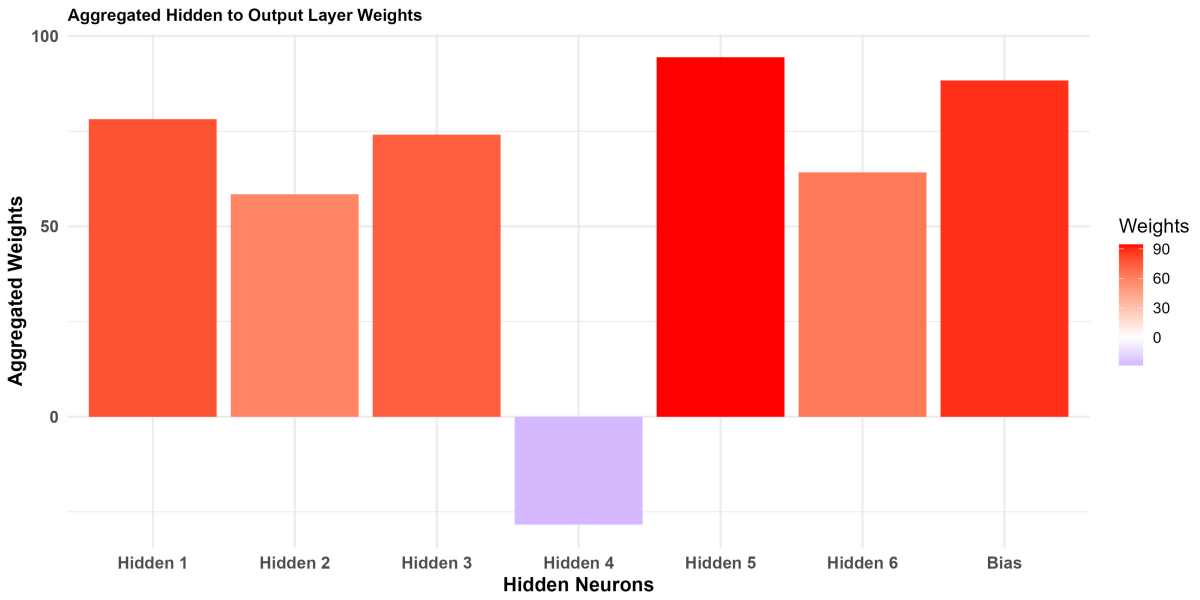


Figure 4.28: Aggregated Hidden to Output Layer Weights

interpretation challenges. The nonlinear transformations and intricate interactions occurring within the hidden layers of the network limit transparency, making it difficult to trace the impact of specific variables or hidden neurons on the outcome. Unlike traditional models, the GWNN does not provide easily interpretable coefficients, and as the number of neurons increases, interpretability decreases even further.

To sum up, while GWNN maps show geographic adaptation, they fail to provide meaningful causal insights. These maps are useful for visualising spatial patterns, but their application in exploratory spatial data analysis is limited due to NN's 'black-box' nature. As a result, GWNN provides a trade-off: it effectively captures complicated spatial connections while lacking the interpretability required for in-depth, actionable research.

4.5 Residual Spatial autocorrelation

To enhance the understanding of how GWR, GWRF, and GWNN addressed geographical heterogeneity, the research applied local Moran's I to examine spatial autocorrelation, uncover potential clustering in the residuals, and visualise the distribution of local residuals (Figure 4.18). The research also employed global Moran's I to assess the overall spatial autocorrelation of residuals throughout the entire study area. The results of the global Moran's I analysis are presented in Table 4.9.

The findings in Table 4.9 illustrate that the GWR model has a global Moran's I of 0.1750, along with a *p*-value that is highly significant, measured at 2.2e-05, indicating that the residuals exhibit considerable spatial autocorrelation. This indicates that the residuals tend to cluster together, revealing that the GWR model may not fully capture the spatial patterns present in the data. In contrast, the GWRF model shows a negative Moran's I value of -0.0352, which is statistically insignificant (*p*-value = 0.9958), signifying a lack of spatial correlation among residuals, indicating a marked improvement in capturing spatial heterogeneity compared to GWR. Similarly, the GWNN model displays a negligible Moran's I value of -0.0004 and an insignificant *p*-value (0.4810), indicating that the GWNN model also effectively addresses spatial structure, with residuals behaving randomly in space.

Overall, the residuals of GWR are more clustered than those of the GWRF and GWNN models. However, GWRF outperforms GWNN, as reflected in its lower Moran's I value, indicating that the errors associated with GWRF are less clustered.

Table 4.9: Global Moran's I for Residuals of GWR, GWRF, and GWNN

Model	**Global Moran's I**	***p*-value**
GWR	0.1750	2.2e-05
GWRF	-0.0352	0.9958
GWNN	-0.0004	0.4810

The accompanying spatial distribution in Figure 4.29 further illuminates the finding in Table 4.9. The residuals of the GWR models exhibit a distinct clustering of high positive and low negative values, indicating possible areas of concern where the model may be underperforming. Moreover, the residuals of the GWRF model show an even distribution with broader ranges in values compared to GWR. The GWNN model outperforms both GWR and GWRF, as seen in its residual map, where the residuals are tightly clustered around zero.

The residual maps illustrate the variations in residual autocorrelation that occur across various spatial locations. Nonetheless, it remains unclear whether these variations represent clusters of high or low values. The right column of Figure 4.29 depicts the local Moran's I cluster of residuals. The presence of large and significant high-high and low-low clusters is seen in the GWR model. In contrast, GWRF shows fewer and less pronounced high-high and low-low clusters than GWR. The insignificant clusters suggest better performance in accounting for spatial dependencies across residuals. The GWNN local Moran's I shows that the residuals are nearly entirely insignificant, indicating spatial randomness.

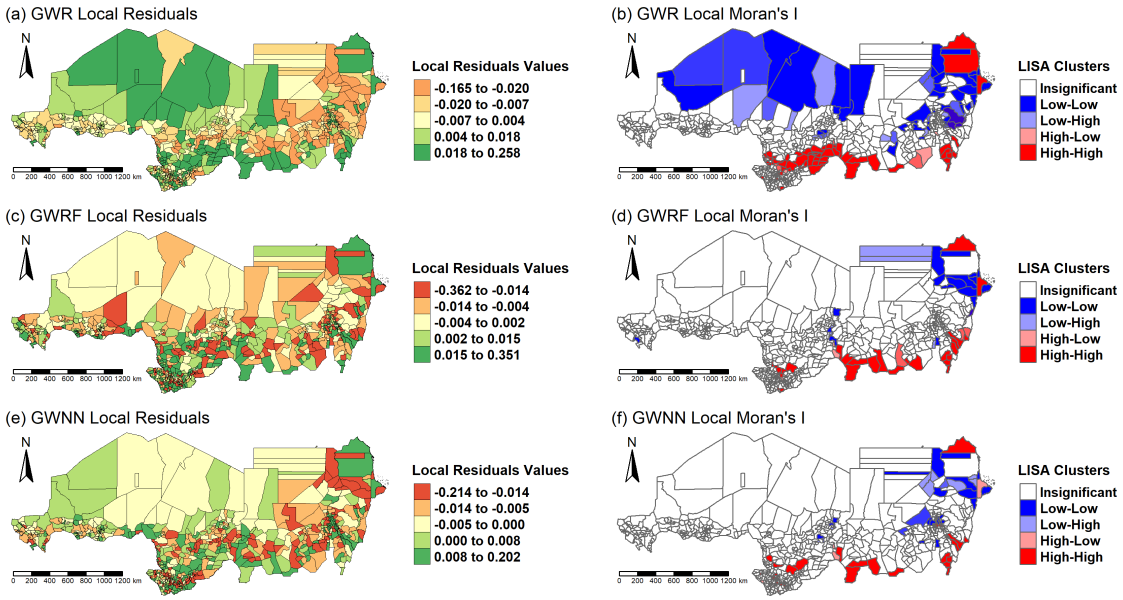


Figure 4.29: Residual Mapping and Spatial Clustering (Local Moran's I) for each Model

Figure 4.29 generally demonstrates that both GWNN and GWRF effectively account for spatial heterogeneity. The residuals for these two models are mostly randomly distributed and lack significant geographic clustering. It is crucial to recognise that the GWRF residuals are obtained through pseudo-hold-out predictions of the out-of-bag sample, instead of being calculated from a direct fit to the training data (Lotfata and Georganos, 2023).

4.6 Comparative Analysis

The findings in this section are presented as follows: Subsection 4.6.1 discusses the predictive performance of GW-models (GWR, GWRF, and GWNN) alongside global models (OLS, RF, and NN), Subsection 4.6.2 details the outcomes of the scalability experiment for GW-models, and Subsection 4.6.3 presents the sensitivity analysis of GW-models, focusing on the impact of varying kernel functions and bandwidth settings on model performance

4.6.1 Predictive Performance

The findings from the validation analysis, which employed a range of performance indicators, are displayed in Table 4.10. The MSE results suggest that the GWNN exhibits the best performance, with the smallest MSE value of 0.0012, closely followed by the GWRF (0.0013). The OLS model shows the highest MSE at 0.0030, indicating the poorest performance in this metric. Similarly, the RMSE further reinforces this pattern, where GWNN also leads with an RMSE

of 0.0333, while OLS shows a significantly higher RMSE of O.0542, further highlighting its relatively poor predictive accuracy compared to the other models.

When considering MAE, GWRF stands out with the lowest MAE of 0.0191, demonstrating that it has the smallest average magnitude of errors. GWNN follows with an MAE of 0.0205, while OLS has the highest MAE at 0.0392, reflecting more substantial average errors in its predictions. In terms of R^2, the GWNN and GWRF models showed comparable results, although the GWNN model slightly outperformed the GWRF model according to the independent validation method used in this study. The GWNN model indicates a higher level of variability (R^2 = 0.9360) compared to the GWRF model (R^2 of 0.9308). GWR follows with an R^2 of 0.9207. OLS, marking the lowest R^2 at 0.8378, shows that it explains the least variance among the models.

In summary, the results indicate that the geographically weighted models (GWNN, GWRF, and GWR) significantly outperform global models such as OLS, RF, and NN across all metrics. This suggests that incorporating geographical weighting enhances predictive performance, especially in capturing the nuances of local variations within the dataset.

Table 4.10: Comparison of Models' Performance on the Test Dataset using Cross-validation

	OLS	**RF**	**NN**	**GWR**	**GWRF**	**GWNN**
MSE	0.0030	0.0018	0.0023	0.0015	0.0013	**0.0012**
RMSE	0.0542	0.0429	0.0474	0.0371	0.0337	**0.0333**
MAE	0.0392	0.0270	0.0324	0.0243	**0.0191**	0.0205
R^2	0.8378	0.9008	0.8755	0.9207	0.9308	**0.9360**

An additional evaluation was performed using scatterplot visualisation, where the dataset (n = 939) was randomly split into 845 training data and 94 test data. Figure 4.30 presents the comparison of the predicted versus observed NPP values using different models: OLS, RF, NN, GWR, GWRF, and GWNN.

The results indicate a consistent improvement in explaining NPP variability (R^2) and reducing RMSE when transitioning from OLS (R^2 = 0.84, RMSE = 0.06) to NN (R^2 = 0.85, RMSE = 0.06) and RF (R^2 = 0.88, RMSE = 0.05). Further improvements were observed in geographically weighted models, where GWR and GWRF both achieved an R^2 of 0.92, with an RMSE of 0.04, indicating substantial accuracy in local spatial predictions. Finally, the GWNN model

performed the best, capturing the highest variability in NPP (R^2 = 0.93) and achieving the lowest RMSE (0.04).

The findings also align with Table 4.10 by highlighting the superiority of geographically weighted statistical machine learning model in capturing spatial heterogeneity and improving prediction accuracy for NPP compared to global models

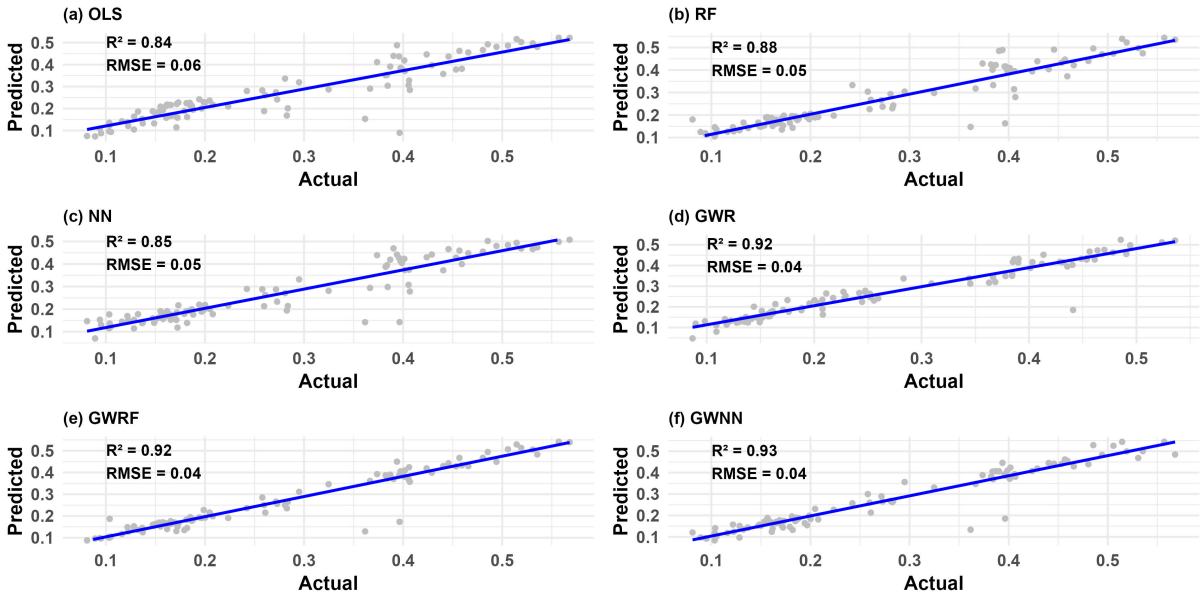


Figure 4.30: The Scatterplots of Actual and Predicted NPP in 94 test samples for the OLS, RF, NN, GWR, GWRF, and GWNN Models Respectively

4.6.2 Accuracy and Efficiency Across Dataset Sizes

This subsection presents the results of the scalability experiment, which evaluates the performance of the GWR, GWRF, and GWNN models across different dataset sizes (25%, 50%, and 75% of the full dataset). The analysis focuses on training time, accuracy metrics (MSE, RMSE, MAE, and R^2), and the capability of each model to maintain efficiency with an increasing dataset size.

Before conducting the scalability analysis, stratified sampling was applied to generate subsets as outlined in Subsection 3.9.2 to ensure that the sampled data maintained the same distribution as the original dataset. The variable "COUNTRY" was used as the stratification variable, ensuring that each subset preserved the original proportions of Chad (45.5%), Niger (18.7%), and Sudan (35.8%).

- For the 25% subset, the dataset was split so that 234 observations (25% of 939 total records) were selected while maintaining the original proportions of Chad, Niger, and Sudan.
- For the 50% subset, 470 observations were selected, ensuring the same proportional representation.
- For the 75% subset, 705 observations were selected while preserving the dataset's overall distribution.

Each subset was then used for the scalability and computational efficiency analysis to evaluate the performance of GWR, GWRF, and GWNN across different data sizes (Table 4.11). This approach ensured that the models were tested on representative samples, preventing bias due to uneven distribution.

Table 4.11: Model Performance Across Different Dataset Sizes

Model	**Data Size**	**Training Time (s)**	**MSE**	**RMSE**	**MAE**	R^2
GWR	**25%**	1.2346	0.0016	0.0396	0.0279	0.9358
GWR	**50%**	2.2205	0.0029	0.0537	0.0356	0.8471
GWR	**75%**	6.5252	0.0019	0.0436	0.0282	0.8846
GWRF	**25%**	9.8232	0.0006	0.0253	0.0193	0.9743
GWRF	**50%**	20.0893	0.0018	0.0424	0.0252	0.9054
GWRF	**75%**	44.0103	0.0012	0.0328	0.0208	0.9353
GWNN	**25%**	15.9712	0.0020	0.0451	0.0315	0.9142
GWNN	**50%**	44.9275	0.0021	0.0457	0.0313	0.9039
GWNN	**75%**	72.5231	0.0009	0.0298	0.0207	0.9467

Table 4.11 key observations:

1. Training time:
    - GWR demonstrates the fastest training times across all dataset sizes, with training times ranging from 1.2346 seconds for 25% of the data to 6.5252 seconds for 75% of the data. This indicates that GWR is computationally efficient and scales well with increasing data size.

- GWRF shows a significant increase in training time as the dataset size grows, from 9.8232 seconds for 25% of the data to 44.0103 seconds for 75% of the data. This suggests that GWRF, while accurate, is more computationally intensive compared to GWR.
- GWNN has the longest training times, ranging from 15.9712 seconds for 25% of the data to 72.5231 seconds for 75% of the data. This highlights the computational cost associated with neural network-based models, especially as the dataset size increases.

2. Model Accuracy:

- GWRF model consistently attains the minimum MSE and RMSE values for all dataset sizes, highlighting its outstanding predictive performance. For example, at 25% of the data, GWRF produces an RMSE of 0.0253 and an MSE of 0.0006, which are significantly lower than those of GWR and GWNN. This trend continues at 50% and 75% of the data, with GWRF maintaining the lowest error metrics.
- GWNN shows competitive accuracy, particularly at larger dataset sizes. At 75% of the data, GWNN achieves an MSE of 0.0009 and an RMSE of 0.0298, which are close to GWRF's performance. This suggests that GWNN benefits from larger datasets, improving its predictive capability.
- GWR performs well in terms of accuracy but is generally outperformed by GWRF and GWNN, especially at larger dataset sizes. For example, at 75% of the data, GWR has an MSE of 0.0019 and an RMSE of 0.0436, which are higher than those of GWRF and GWNN.

3. Coefficient of Determination (R^2):

- GWRF consistently achieves the highest R^2 values, indicating the best fit to the data. At 25% of the data, GWRF achieves an R^2 of 0.9743, which is significantly higher than GWR (0.9358) and GWNN (0.9142). This trend continues at 50% and 75% of the data, with GWRF maintaining the highest R^2 values.
- GWNN shows improvement in R^2 as the dataset size increases, reaching 0.9467 at 75% of the data, which is close to GWRF's performance. This indicates that GWNN benefits from larger datasets, improving its ability to explain the variance in the data.
- GWR has lower R^2 values compared to GWRF and GWNN, particularly at larger dataset sizes. For example, at 75% of the data, GWR has an R^2 of 0.8846, which is lower than both GWRF and GWNN. However, it shows the best performance with smaller datasets, similar to GWRF, particularly at 25% of the data.

4. Efficiency vs. Accuracy Trade-off:

    - GWR is the most efficient model in terms of training time but sacrifices some predictive accuracy, particularly at larger dataset sizes. This makes GWR a suitable choice for scenarios where computational efficiency is a priority, and moderate accuracy is acceptable.
    - GWRF offers the best balance between accuracy and computational cost. While it requires more training time than GWR, it consistently delivers the highest accuracy across all dataset sizes, making it the preferred choice for applications where predictive performance is critical.
    - GWNN is the most computationally expensive model but shows competitive accuracy, especially at larger dataset sizes. This makes GWNN a viable option for scenarios where computational resources are not a constraint and high accuracy is required.

The scalability experiment reveals that GWRF is the most accurate model across all dataset sizes, achieving the lowest error metrics and highest R^2 values. However, it comes at the cost of increased computational time. GWR is the most efficient model but is generally less accurate than GWRF and GWNN, particularly at larger dataset sizes. GWNN shows competitive accuracy, especially with larger datasets, but requires the most computational resources. The choice of model depends on the application's specific requirements, balancing the trade-off between accuracy and computational efficiency.

4.6.3 Kernel and Bandwidth Effects on Model Sensitivity

This subsection synthesises the findings from analysing three distinct spatially weighted modelling approaches: GWR, GWRF, and GWNN. Across all three models, selecting appropriate kernel functions and bandwidths emerged as a critical determinant of model performance.

In GWR, the analysis of AIC and R^2 (Table 4.12), coupled with graphical exploration of AIC_c across bandwidth ranges for both adaptive (Figure 4.31) and fixed (Figure 4.32) kernel function, highlighted the importance of balancing model fit and complexity. The estimation of the GWR model begins with determining the optimal bandwidth, as presented in Table 4.12. Based on the model's goodness of fit, modelling using the Fixed Gaussian kernel produced the smallest AIC value (-3983.574) and the highest R^2 value (0.9803) compared to other kernel weighting functions, resulting in an optimal bandwidth of 0.5880.

Note that the fixed bandwidth of 0.5880, for instance, indicates that the weighting function searches within a radius of 0.5880 units and includes all observations within this fixed distance for each local regression (Guo et al., 2008). An adaptive bandwidth translates to a proportion of the total dataset being used in each local model, effectively adjusting the spatial influence based on the local data density (Fotheringham et al., 2003). For example, an optimal bandwidth of 20, in the context of the eastern Sahel County data (which has 939 records), means that the local regression at any given point should be calibrated using the 20 nearest neighbours. This represents approximately 2.13% (calculated as 20/939 = 0.02129) of the total dataset.

Table 4.12: Comparison of Kernel Weighting Functions under GWR

Kernel Function	**Kernel Weighting Funtion**	**Bandwidth**	AIC	R^2
Adaptive	Gaussian	20	-3479.35	0.9374
	Bisquare	52	-3668.61	0.9608
	Tricube	52	-3659.087	0.9591
	Exponential	20	-3376.508	0.9310
	Boxcar	28	-3572.505	0.9504
Fixed	**Gaussian**	**0.5880**	**-3983.574**	**0.9803**
	Bisquare	3.5775	-3435.311	0.9402
	Tricube	3.0835	-3538.134	0.9673
	Exponential	0.5110	-3892.182	0.9800
	Boxcar	2.5929	-3335.8	0.9248

To gain a more nuanced understanding, Figures 4.31 and 4.32 complement the information in Table 4.12 by visualising the AIC_c landscape. As shown in Figure 4.31, the GWR model with a bandwidth of 52 and the Adaptive Bisquare have the best performance. Different kernels show smooth, consistent trends as $AICc$ increases with bandwidth. This indicates that for the study's dataset, the selection of smaller bandwidths, particularly ranging from 20 to 100, leads to smoother performance.

To further investigate the behaviour of fixed kernel functions, the $AICc$ graph for this category is examined (Figure 4.32). Figure 4.32 reveals a crucial aspect of fixed kernel models: their sensitivity to bandwidth, particularly at very small bandwidth values. A steep decline in $AICc$ is observed as bandwidth increases from near zero, highlighting the importance of selecting an

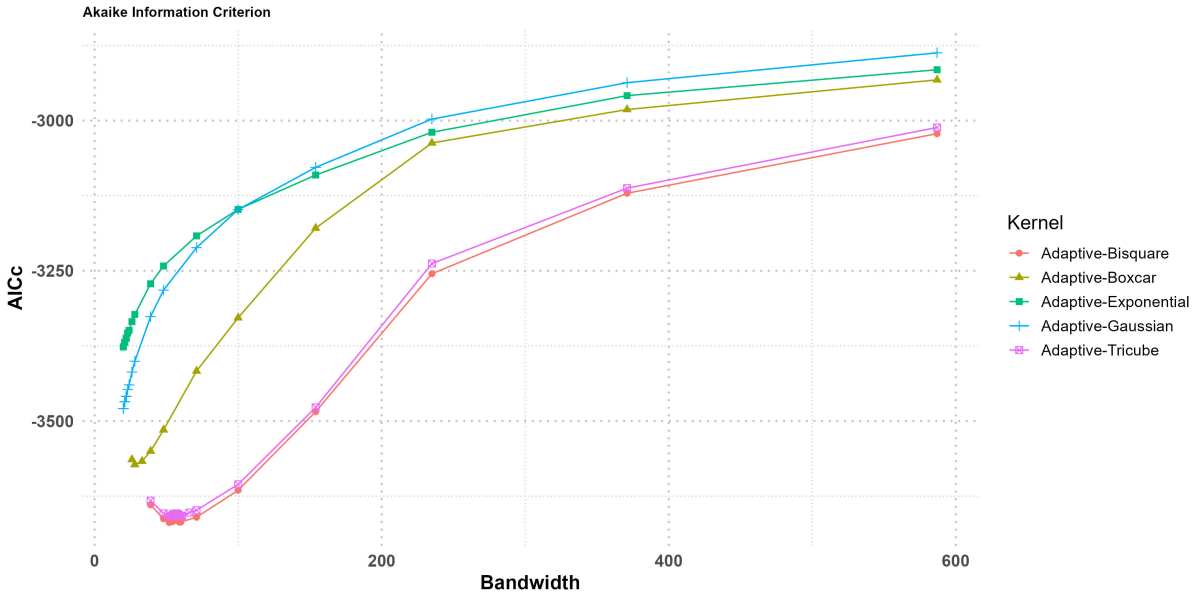


Figure 4.31: AIC of GWR with Different Bandwidths and Kernel Weighting Functions (Adaptive)

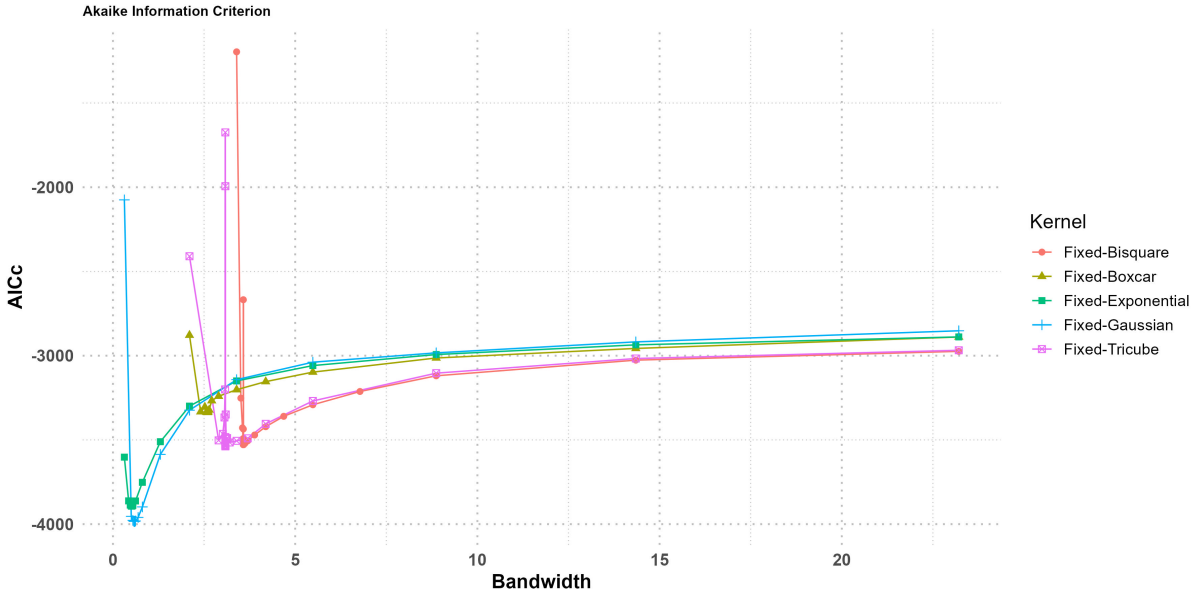


Figure 4.32: AIC of GWR with Different Bandwidths and Kernel Weighting Functions (Fixed)

appropriate bandwidth. Figure 4.32 also demonstrates that after this initial drop, the $AICc$ values stabilise, becoming less sensitive to further bandwidth increases. This stabilisation region is where the bandwidth values reported in Table 4.12 for the fixed kernels likely fall. Lastly, Figure 4.32 also provides visual confirmation of the superior performance of the Fixed Gaussian kernel, as it consistently exhibits the lowest $AICc$ values across a broad range of bandwidths after the initial steep decline.

Similar to Table 4.12, Table 4.13 displays the results of the GWNN's optimal bandwidth achieved through various kernels. The best weight is determined by the lowest $RMSE$ recorded (Hagenauer and Helbich, 2022). Table 4.13 indicates that the Fixed Exponential kernel function exhibits the lowest $RMSE$ value (0.0277) and highest R^2 value (0.9535), resulting in an optimum bandwidth of 0.3997. This suggests that for this specific model structure and dataset, a fixed bandwidth approach with an appropriate kernel function might be advantageous. Table 4.13 also highlights the sensitivity of GWNN to kernel choice, with certain kernels like Bisquare, Tricube, and Boxcar exhibiting comparatively higher $RMSE$ values.

Table 4.13: Comparison of Kernel Weighting Functions under GWNN

Kernel Function	**Kernel Weighting Funtion**	**Bandwidth**	$RMSE$	R^2
Adaptive	Gaussian	4	0.0430	0.9301
	Bisquare	8	0.0325	0.9372
	Tricube	8	0.0343	0.9315
	Exponential	6	0.0433	0.9264
	Boxcar	8	0.0344	0.9309
Fixed	Gaussian	0.5995	0.0292	0.9484
	Bisquare	3.5859	0.0442	0.8823
	Tricube	3.5859	0.0444	0.8809
	Exponential	**0.3997**	**0.0277**	**0.9535**
	Boxcar	2.9392	0.0510	0.8435

Finally, Figure 4.33 illustrates the OOB R^2 for GWRF models using adaptive and fixed spatial kernels across different bandwidths when predicting NPP in the Sahel region. The adaptive kernel consistently outperforms the fixed kernel, reaching a peak R^2 of approximately 0.94 at a bandwidth of 24–30, indicating that incorporating more neighbouring observations initially enhances predictive accuracy. However, beyond this range, fluctuations suggest that excessive smoothing reduces sensitivity to local variations. In contrast, the fixed kernel maintains a lower and more stable R^2 around 0.90–0.91, showing that a rigid spatial influence may not effectively capture spatial heterogeneity. The slight decline in performance at larger bandwidths suggests that increasing spatial windows dilutes localised effects.

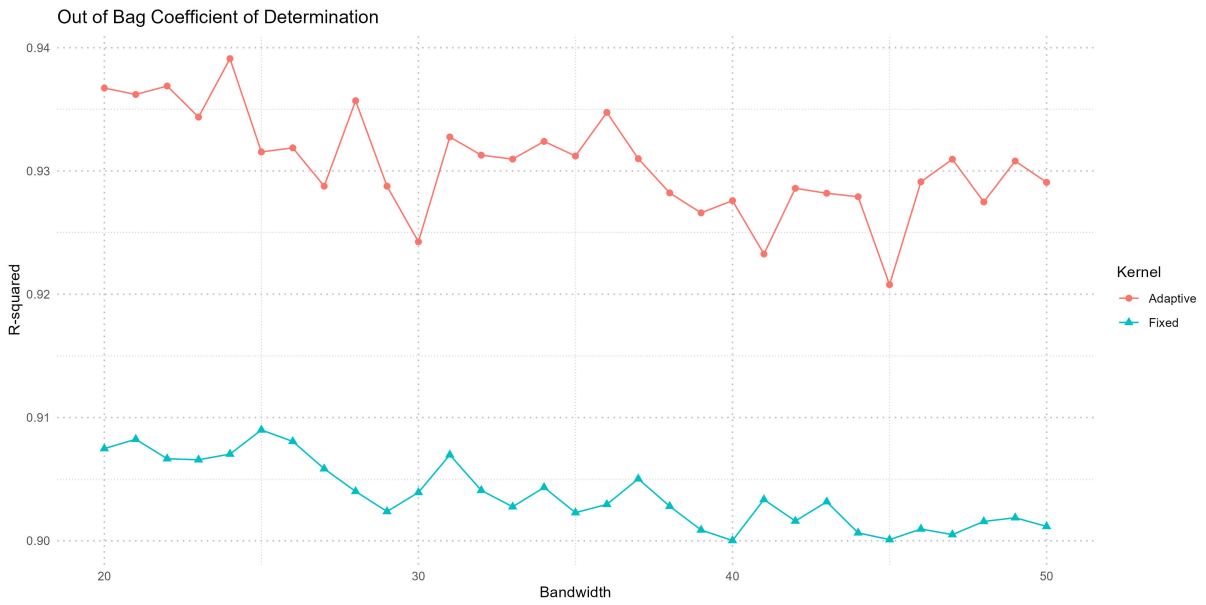


Figure 4.33: OOB R^2 of GWRF with Different Bandwidths and Spatial Kernels

Collectively, the analysis across GWR, GWRF, and GWNN underscores the importance of careful consideration of kernel function and bandwidth selection in spatially weighted models. While specific optimal configurations varied across the three approaches, some general trends emerged. Adaptive kernels often proved advantageous in GWRF, highlighting their flexibility in capturing spatial heterogeneity. Fixed kernels, particularly Gaussian and Exponential, frequently performed strongly in both GWR and GWNN. The choice of performance metric (AIC, OOB R^2, and $RMSE$) also played a crucial role in guiding model selection and optimisation. Ultimately, the most effective approach involves a comprehensive evaluation of various kernel functions and bandwidths, tailored to the specific characteristics of the dataset and the research question at hand.

Chapter 5

Summary

This section provides a summary of the findings and interpretations discussed in Chapter 4 to assess and better understand the effectiveness of the three geographically weighted models: GWR, GWRF, and GWNN in predicting NPP across the eastern Sahel. A conclusion on the overall model performance and the broader implications of this study can then be drawn.

5.1 Traditional vs. GW Models: A Performance Comparison

The results indicate that spatially adaptive models like GWR, GWRF, and GWNN outperformed traditional global models such as OLS, RF, and NN (Table 4.10 and Figure 4.30). OLS served as a baseline model for this study. While it provided a broad understanding of the linear relationships between predictor variables (e.g., rainfall, temperature, soil moisture, and elevation) and NPP, it failed to account for spatial heterogeneity. This limitation was evident in the high residuals visualised in Figure 4.18 and relatively low adjusted R^2 values (0.8354). These findings align with previous research, which highlights the inability of global models to capture local variations in environmental data (Higginbottom and Symeonakis, 2013; Quan et al., 2005).

The study continued with the GWR model, which effectively tackled the spatial non-stationarity inherent in the dataset by permitting model parameters to vary across geographic locations. The spatial distribution of GWR coefficients revealed significant regional differences in the relationship between predictor variables and NPP. For example, rainfall and soil moisture showed stronger correlations with NPP in Sudan compared to Chad and Niger (Figure 4.19). These results are consistent with studies by Yang et al. (2023), which emphasise the importance of accounting for local environmental conditions in ecological modelling. However, GWR's effectiveness was limited when dealing with complex, nonlinear relationships among variables. This

prompted the exploration of advanced machine learning models such as GWRF and GWNN.

The GWRF model demonstrated superior predictive performance compared to both OLS and GWR. By integrating the strengths of RF with geographical weighting, GWRF captured nonlinear relationships and spatial heterogeneity simultaneously. The feature importance analysis highlighted the varying contributions of rainfall, soil moisture, temperature, and elevation to NPP prediction. Soil moisture and rainfall emerged as the most influential predictors, with their impacts fluctuating across different regions (Figures 4.22 to 4.24). The local R^2 values (0.9376) for GWRF were significantly higher than those for GWR, suggesting that GWRF offers a more precise depiction of the spatial dynamics of NPP. This finding corroborates the work of Georganos et al. (2021) and Santos et al. (2019), who demonstrated the efficacy of GWRF in handling high-dimensional and nonlinear data.

Lastly, GWNN provided the most robust predictions among all the models. Its ability to capture complex, nonlinear interactions between variables while accounting for spatial variability makes it an invaluable tool for ecological modelling. The spatial weight matrices derived from GWNN revealed nuanced patterns of influence for each predictor variable, offering deeper insights into the spatial dynamics of NPP. Notably, GWNN outperformed GWRF in regions with highly heterogeneous environmental conditions, such as the transition zones between arid and semi-arid areas. These outcomes align with the research conducted by Hagenauer and Helbich (2022), who highlighted the advantages of GWNN in capturing both spatial and nonlinear complexities.

5.2 Drivers of NPP

This study identified several key factors significantly influencing NPP in the eastern Sahel region. Rainfall consistently emerged as the most critical predictor of NPP across all models, highlighting the crucial role of water availability in sustaining vegetation growth. The relationship between temperature and NPP proved more complex and nonlinear. While moderate temperatures correlated with higher productivity, extreme heat exhibited a negative impact. Soil moisture also played a significant role, particularly in areas characterised by high rainfall variability. Its influence, however, varied spatially, reflecting the intricate interplay between soil properties and vegetation growth. Finally, while elevation's impact on NPP was less pronounced compared to the other factors, it remained relevant in specific highland areas.

5.3 Regional Variations and Spatial Patterns

Analysis of spatial patterns revealed clear regional variations in the drivers of NPP. The highest predicted NPP values were found in Sudan, largely attributable to abundant rainfall. Conversely, arid conditions and limited water availability contributed to lower NPP predictions in areas of Chad and Niger. These results underscore the importance of developing region-specific strategies for natural resource management and sustainable land use practices. The spatially explicit outputs from the GWRF and GWNN models offer valuable information for policymakers and land managers seeking to design targeted interventions.

5.4 Sustainable Land Management Strategies

This study's findings offer significant contributions to sustainable land management in the eastern Sahel. The improved accuracy and spatial detail of NPP predictions provided by the advanced models applied here can enhance decision-making for agriculture, forestry, and biodiversity conservation. Specifically, the identification of high-productivity areas can optimise agricultural practices and improve food security, while understanding NPP's spatial variability can support efforts to mitigate land degradation and combat desertification.

Chapter 6

Conclusion, Limitations, Recommendations, and Future Directions

6.1 Conclusion

This study offers a comprehensive framework for predicting NPP in the eastern Sahel, employing geographically weighted statistical machine learning. The applied models, which incorporate spatial heterogeneity and nonlinear relationships, represent a substantial advancement over traditional methods, promising to enhance understanding of ecosystem productivity and support sustainable development in this vulnerable African region. This research contributes to ecological modelling by demonstrating the value of integrating geographical weighting with machine learning, underscoring the importance of spatially explicit models for understanding and managing complex ecological systems. These findings have practical implications for sustainable land management, enabling policymakers and land managers to leverage insights from the GWRF and GWNN models for informed decisions regarding resource allocation, agricultural planning, and climate adaptation.

6.2 Limitations

Despite its contributions, this study has several limitations that must be noted. Firstly, the quality and resolution of input data, including climate and vegetation data, varied and were aggregated to a 5 km resolution. This may have affected model performance and limited finer-scale analysis. Additionally, data imputation using OK, while effective, may introduce uncertainty in

missing values.

Secondly, the computational demands of the GWRF and GWNN models, particularly GWNN with its multiple local neural networks, are significant. This limits their applicability in resource-constrained settings and for large-scale applications. Lastly, the study focused on a relatively short time frame (2019–2021), which may not fully capture long-term climate trends or the effects of climate variability. This limited temporal generalisation and robustness for assessing climate change impacts on NPP trends.

6.3 Recommentations

Based on the study's findings, several recommendations are proposed to enhance the application of geographically weighted machine learning models in ecological research and land management.

Firstly, research institutions and environmental organisations should adopt advanced spatial machine learning methods like GWRF and GWNN to better capture spatial variability and non-linear relationships in ecological systems. Future studies should also explore hybrid approaches that combine these models to maximise predictive accuracy and model robustness. Secondly, improving data quality is essential. Efforts should focus on acquiring high-resolution environmental datasets, establishing long-term monitoring systems for key variables (such as rainfall and soil moisture), and integrating diverse data sources, including remote sensing, ground observations, and socio-economic indicators, to provide a comprehensive understanding of NPP drivers.

Thirdly, spatial modelling outputs should be leveraged to implement region-specific land management interventions. Policymakers and land managers should use these insights to guide targeted strategies, such as improving irrigation, enhancing soil conservation, and promoting drought-resistant crops tailored to local ecological conditions. Moreover, there is a strong need to build local capacity in the application of geographically weighted models. Training programmes, workshops, and knowledge-sharing platforms should be established to equip researchers, decision-makers, and land managers in the Sahel with the necessary skills to utilise these advanced tools effectively.

Finally, policymakers are encouraged to integrate spatially explicit NPP predictions into planning and decision-making processes, ensuring that resources are allocated efficiently and sus-

tainable land management practices are prioritised to mitigate land degradation and support ecosystem resilience.

6.4 Future Work

While this study has made significant contributions to the field of ecological modelling, several avenues for future research can build on the findings and address the limitations of the current study.

6.4.1 Expanded Variables

Future studies should incorporate additional environmental and socio-economic factors, such as land cover changes, human activities, and soil nutrient levels, to gain a more holistic understanding of NPP dynamics.

6.4.2 Integrate Spatial-Temporal Approaches

Integrating spatial-temporal machine learning approaches into geographically weighted frameworks could simultaneously capture spatial and temporal variations, offering a more holistic view of dynamic ecological processes. Extending GWR to support dynamic spatial relationships over time would allow for the analysis of evolving spatial patterns and the investigation of shifting environmental drivers, which is particularly relevant for studies that span multiple seasons or years.

6.4.3 Model Validation and Generalisability

Robustness and reliability of predictions should also be a priority. Future work should focus on model validation using independent datasets from various regions and periods and conducting cross-regional comparisons to evaluate the generalisability of GW models across different ecological contexts.

6.4.4 Incorporating Uncertainty Modelling

Accounting for uncertainties in GWR predictions is an essential consideration for robust decision-making. Future research can explore methodologies to quantify and incorporate uncertainties into GWR models (such as Bayesian Geographically Weighted Regression (BGWR)), providing users with a more comprehensive understanding of the reliability of spatial predictions.

6.4.5 Integration of Explainable AI

One avenue for future exploration lies in integrating Explainable Artificial Intelligence (XAI) techniques with machine learning-based GWR. As complex machine learning models are often regarded as 'black-boxes,' incorporating methods to interpret and explain model predictions can enhance the transparency and trustworthiness of GWR models. This is particularly crucial in applications where stakeholders require a clear understanding of the factors influencing spatial relationships.

6.4.6 Scalability and Computational Efficiency

With the increasing availability of big geospatial data, improving geographically weighted models' scalability and computational efficiency is vital. This includes developing optimised algorithms, leveraging distributed computing, and applying advanced data preprocessing techniques to enable the analysis of large, high-resolution datasets without compromising performance.

Bibliography

Abdi, A., Seaquist, J., Tenenbaum, D., Eklundh, L., and Ardö, J. (2014). The supply and demand of net primary production in the sahel. *Environmental Research Letters*, 9(9):094003. 1

Ali, P. J. M., Faraj, R. H., Koya, E., Ali, P. J. M., and Faraj, R. H. (2014). Data normalization and standardization: a technical report. *Mach Learn Tech Rep*, 1(1):1–6. 10

Anselin, L. (1995). Local indicators of spatial association—lisa. *Geographical analysis*, 27(2):93–115. 22, 32

Bae, B., Kim, H., Lim, H., Liu, Y., Han, L. D., and Freeze, P. B. (2018). Missing data imputation for traffic flow speed using spatio-temporal cokriging. *Transportation Research Part C: Emerging Technologies*, 88:124–139. 8

Bickel, P. J. and Doksum, K. A. (2015). *Mathematical statistics: basic ideas and selected topics, volumes I-II package*. Chapman and Hall/CRC. 23

Bivand, R. (2008). Applied spatial data analysis with r. 21

Bivand, R. (2017). Geographically weighted regression. *CRAN Task View: Analysis of Spatial Data*. 12, 50

Bivand, R. and Wong, D. W. S. (2018). Comparing implementations of global and local indicators of spatial association. *TEST*, 27(3):716–748. 32

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32. 16, 17

Brunsdon, C., Fotheringham, A. S., and Charlton, M. E. (1996). Geographically weighted regression: a method for exploring spatial nonstationarity. *Geographical analysis*, 28(4):281–298. 11, 52

Charlton, M., Fotheringham, S., and Brunsdon, C. (2009). Geographically weighted regression. *White paper. National Centre for Geocomputation. National University of Ireland Maynooth*, 2. 12, 15

Chicco, D., Warrens, M. J., and Jurman, G. (2021). The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *Peerj computer science*, 7:e623. 23

Chung, S. Y., Venkatramanan, S., Elzain, H. E., Selvam, S., and Prasanna, M. (2019). Supplement of missing data in groundwater-level variations of peak type using geostatistical methods. *GIS and geostatistical techniques for groundwater science*, 33. 8

Craney, T. A. and Surles, J. G. (2002). Model-dependent variance inflation factor cutoff values. *Quality engineering*, 14(3):391–403. 44

Dai, Z., Wu, S., Wang, Y., Zhou, H., Zhang, F., Huang, B., and Du, Z. (2022). Geographically convolutional neural network weighted regression: A method for modeling spatially non-stationary relationships based on a global spatial proximity grid. *International Journal of Geographical Information Science*, 36(11):2248–2269. 20

Daoud, J. I. (2017). Multicollinearity and regression analysis. In *Journal of Physics: Conference Series*, volume 949, page 012009. IOP Publishing. 10

Deb Burman, P. K. (2020). Estimation of net primary productivity: An introduction to different approaches. In *Spatial Modeling in Forest Resources Management: Rural Livelihood and Sustainable Development*, pages 33–69. Springer. 1

Di Bucchianico, A. (2008). Coefficient of determination (r 2). *Encyclopedia of statistics in quality and reliability*. 23

Dongare, A., Kharde, R., Kachare, A. D., et al. (2012). Introduction to artificial neural network. *International Journal of Engineering and Innovative Technology (IJEIT)*, 2(1):189–194. 19

Du, Z., Wang, Z., Wu, S., Zhang, F., and Liu, R. (2020). Geographically neural network weighted regression for the accurate estimation of spatial non-stationarity. *International Journal of Geographical Information Science*, 34(7):1353–1377. v, 20, 21

Epule, T. E., Chehbouni, A., and Dhiba, D. (2022). Recent climate change adaptation strategies in the sahel. *The Nature, Causes, Effects and Mitigation of Climate Change on the Environment*, page 263. 2, 26, 27

Epule, T. E., Ford, J., and Lwasa, S. (2018). Climate change stressors in the sahel. *GeoJournal*, 83. 26, 27

Fotheringham, A. S., Brunsdon, C., and Charlton, M. (2003). *Geographically weighted regression: the analysis of spatially varying relationships*. John Wiley & Sons. v, 2, 11, 14, 15, 16, 17, 31, 72

Fox, J. and Weisberg, S. (2019). *An R Companion to Applied Regression*. Sage, Thousand Oaks CA, third edition. 30

Frick, H., Chow, F., Kuhn, M., Mahoney, M., Silge, J., and Wickham, H. (2024). *rsample: General Resampling Infrastructure*. R package version 1.2.1, https://github.com/tidymodels/rsample. 33

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232. 56

Funk, C., Peterson, P., Landsfeld, M., Pedreros, D., Verdin, J., Shukla, S., Husak, G., Rowland, J., Harrison, L., Hoell, A., et al. (2015). The climate hazards infrared precipitation with stations—a new environmental record for monitoring extremes. *Scientific data*, 2(1):1–21. 28

Georganos, S., Grippa, T., Niang Gadiaga, A., Linard, C., Lennert, M., Vanhuysse, S., Mboga, N., Wolff, E., and 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 International*, 36(2):121–136. 17, 19, 77

Gogtay, N. J. and Thatte, U. M. (2017). Principles of correlation analysis. *Journal of the Association of Physicians of India*, 65(3):78–81. 9

Gollini, I., Lu, B., Charlton, M., Brunsdon, C., and Harris, P. (2013). Gwmodel: an r package for exploring spatial heterogeneity using geographically weighted models. *arXiv preprint arXiv:1306.0413*. 13

Grekousis, G. (2020). *Spatial analysis methods and practice: describe–explore–explain through GIS*. Cambridge University Press. 54

Grekousis, G., Feng, Z., Marakakis, I., Lu, Y., and Wang, R. (2022). Ranking the importance of demographic, socioeconomic, and underlying health factors on us covid-19 deaths: A geographical random forest approach. *Health & Place*, 74:102744. 31

Grömping, U. (2009). Variable importance assessment in regression: linear regression versus random forest. *The American Statistician*, 63(4):308–319. 17

Gruber, A., Dorigo, W. A., Crow, W., and Wagner, W. (2017). Triple collocation-based merging of satellite soil moisture retrievals. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):6780–6792. 28

Gruber, A., Scanlon, T., van der Schalie, R., Wagner, W., and Dorigo, W. (2019). Evolution of the esa cci soil moisture climate data records and their underlying merging methodology. *Earth System Science Data*, 11(2):717–739. 28

Gräler, B., Pebesma, E., and Heuvelink, G. (2016). Spatio-Temporal Interpolation using gstat. *The R Journal*, 8(1):204–218. 30

Guo, L., Ma, Z., and Zhang, L. (2008). Comparison of bandwidth selection in application of geographically weighted regression: a case study. *Canadian Journal of Forest Research*, 38(9):2526–2534. 15, 72

Hagenauer, J. and Helbich, M. (2022). A geographically weighted artificial neural network. *International Journal of Geographical Information Science*, 36(2):215–235. 2, 19, 31, 74, 77

Helbich, M. and Griffith, D. A. (2016). Spatially varying coefficient models in real estate: Eigenvector spatial filtering and alternative approaches. *Computers, Environment and Urban Systems*, 57:1–11. 33

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., et al. (2020). The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049. 28

Higginbottom, T. and Symeonakis, I. (2013). Modelling precipitation and anthropogenic influence on vegetation productivity trends in the african sahel. http://www. earsel. org/symposia/2013-symposium-Matera/index. php. 3, 76

Hijmans, R. J., Barbosa, M., Ghosh, A., and Mandel, A. (2024). *geodata: Download Geographic Data*. R package version 0.6-2. 29

Hijmans, R. J., Bivand, R., Forner, K., Ooms, J., Pebesma, E., and Sumner, M. D. (2022). Package 'terra'. *Maintainer: Vienna, Austria*. 29

Jamil, B. and Akhtar, N. (2017). Comparative analysis of diffuse solar radiation models based on sky-clearness index and sunshine period for humid-subtropical climatic region of india: A case study. *Renewable and Sustainable Energy Reviews*, 78:329–355. 23

Janssen, V. (2009). Understanding coordinate reference systems, datums and transformations. 36

Jarvis, A., Reuter, H., Nelson, A., and Guevara, E. (2008). Hole-filled seamless srtm data v4. tech. rep., international centre for tropical agriculture (ciat). *Cali, Columbia*. 28

Kalogirou, S., Georganos, S., and Kalogirou, M. S. (2022). Package 'spatialml'. *R Cran*. 31

Kowe, P., Dube, T., Mushore, T. D., Ncube, A., Nyenda, T., Mutowo, G., Chinembiri, T. S., Traore, M., and Kizilirmak, G. (2022). Impacts of the spatial configuration of built-up areas and urban vegetation on land surface temperature using spectral and local spatial autocorrelation indices. *Remote Sensing Letters*, 13(12):1222–1235. 21

Kuhn and Max (2008). Building predictive models in r using the caret package. *Journal of Statistical Software*, 28(5):1–26. 31

Lan, Q., Tang, J., Mei, X., Yang, X., Liu, Q., and Xu, Q. (2023). Hazard assessment of rainfall–induced landslide considering the synergistic effect of natural factors and human activities. *Sustainability*, 15(9):7699. v, 18

LeSage, J. P. (2004). A family of geographically weighted regression models. In *Advances in spatial econometrics: methodology, tools and applications*, pages 241–264. Springer. 12

Leung, Y., Mei, C.-L., and Zhang, W.-X. (2000). Statistical tests for spatial nonstationarity based on the geographically weighted regression model. *Environment and planning A*, 32(1):9–32. 31, 52

Levy, P. S. and Lemeshow, S. (2013). *Sampling of populations: methods and applications*. John Wiley & Sons. 24

Li, G., Ren, H., Li, Y., and Liu, J. (2003). Review and tendency of net primary productivity study. *Ecol. Sci*, 22:360–365. 1

Li, J., Bi, M., and Wei, G. (2022). Investigating the impacts of urbanization on vegetation net primary productivity: a case study of chengdu–chongqing urban agglomeration from the perspective of townships. *Land*, 11(11):2077. 2, 7

Lin, J., Cromley, R., and Zhang, C. (2011). Using geographically weighted regression to solve the areal interpolation problem. *Annals of GIS*, 17(1):1–14. 11

Liu, R., Jiao, L., Liu, Y., and Wang, Y. (2023). Multi-scale spatial analysis of satellite-retrieved surface evapotranspiration in beijing, a rapidly urbanizing region under continental monsoon climate. *Environmental Science and Pollution Research*, 30(8):20402–20414. 22

Lohr, S. L. (2021). *Sampling: design and analysis*. Chapman and Hall/CRC. 24

Lotfata, A. and Georganos, S. (2023). Spatial machine learning for predicting physical inactivity prevalence from socioecological determinants in chicago, illinois, usa. *Journal of Geographical Systems*, pages 1–21. 16, 17, 66

Lotfata, A., Grekousis, G., and Wang, R. (2023). Using geographical random forest models to explore spatial patterns in the neighborhood determinants of hypertension prevalence across chicago, illinois, usa. *Environment and Planning B: Urban Analytics and City Science*, 50(9):2376–2393. 33

Lu, B., Brunsdon, C., Charlton, M., and Harris, P. (2017). Geographically weighted regression with parameter-specific distance metrics. *International Journal of Geographical Information Science*, 31(5):982–998. 19

Lu, B., Harris, P., Charlton, M., and Brunsdon, C. (2014). The gwmodel r package: further topics for exploring spatial heterogeneity using geographically weighted models. *Geo-spatial Information Science*, 17(2):85–101. 31

Lu, X.-Y., Chen, X., Zhao, X.-L., Lv, D.-J., and Zhang, Y. (2021). Assessing the impact of land surface temperature on urban net primary productivity increment based on geographically weighted regression model. *Scientific Reports*, 11(1):22282. 2, 6, 7

Luo, Y., Yan, J., and McClure, S. (2021). Distribution of the environmental and socioeconomic risk factors on covid-19 death rate across continental usa: a spatial nonlinear analysis. *Environmental Science and Pollution Research*, 28(6):6587–6599. 55

Luo, Y., Yan, J., McClure, S. C., and Li, F. (2022). Socioeconomic and environmental factors of poverty in china using geographically weighted random forest regression model. *Environmental Science and Pollution Research*, pages 1–13. 16

Mbow, C., Halle, M., El Fadel, R., and Thiaw, I. (2021). Land resources opportunities for a growing prosperity in the sahel. *Current Opinion in Environmental Sustainability*, 48:85–92. 2

Nduwayezu, G., Zhao, P.-x., Kagoyire, C., Eklund, L., Bizimana, J. P., Pilesjo, P., and Mansourian, A. (2023). Understanding the spatial non-stationarity in the relationships between malaria incidence and environmental risk factors using geographically weighted random forest: A case study in rwanda. 18

Pebesma, E. J. (2004). Multivariable geostatistics in s: the gstat package. *Computers & geosciences*, 30(7):683–691. 30

Piao, S., Ciais, P., Huang, Y., Shen, Z., Peng, S., Li, J., Zhou, L., Liu, H., Ma, Y., Ding, Y., et al. (2010). The impacts of climate change on water resources and agriculture in china. *Nature*, 467(7311):43–51. 1

Pinzon, J., Pak, E., Tucker, C., Bhatt, U., Frost, G., and Macander, M. (2023). Global vegetation greenness (ndvi) from avhrr gimms-3g+, 1981–2022, ornl daac, oak ridge, tennessee, usa. 28

Pomposi, C., Kushnir, Y., and Giannini, A. (2015). Moisture budget analysis of sst-driven decadal sahel precipitation variability in the twentieth century. *Climate Dynamics*, 44:3303–3321. 2

Quan, Ni, J., and Tenhunen, Wang, J. (2005). Application of a geographically-weighted regression analysis to estimate net primary production of chinese forest ecosystems. *Global ecology and biogeography*, 14(4):379–393. 1, 2, 5, 6, 76

Quiñones, S., Goyal, A., and Ahmed, Z. U. (2021). Geographically weighted machine learning model for untangling spatial heterogeneity of type 2 diabetes mellitus (t2d) prevalence in the usa. *Scientific reports*, 11(1):6955. 31, 33

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. 26

Roger Bivand (2022). R packages for analyzing spatial data: A comparative case study with areal data. *Geographical Analysis*, 54(3):488–518. 32

Rose, R. M. (2015). The impact of climate change on human security in the sahel region of africa. *Donnish Journal of African Studies and Development*, 1(2):9–14. 2, 26

Sakor, B. Z. (2020). Is demography a threat to peace and security in the sahel? 2, 3

Santos, F., Graw, V., and Bonilla, S. (2019). A geographically weighted random forest approach for evaluate forest change drivers in the northern ecuadorian amazon. *PloS one*, 14(12):e0226224. 2, 17, 77

Schabenberger, O. and Gotway, C. A. (2017). *Statistical methods for spatial data analysis*. Chapman and Hall/CRC. 21, 22

Senthilnathan, S. (2019). Usefulness of correlation analysis. *Available at SSRN 3416918*. vii, 9, 10

Sharma, S., Sharma, S., and Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316. 19

Sulekan, A. and Jamaludin, S. S. S. (2020). Review on geographically weighted regression (gwr) approach in spatial analysis. *Malaysian Journal of Fundamental and Applied Sciences*, 16(2):173–177. 11, 12, 14, 15

Tennekes, M., Nowosad, J., Gombin, J., Jeworutzki, S., Russell, K., Zijdeman, R., Clouse, J., Lovelace, R., and Muenchow, J. (2023). Package 'tmap'. 30

Thapa, R. B. and Estoque, R. C. (2012). Geographically weighted regression in geospatial analysis. In *Progress in geospatial analysis*, pages 85–96. Springer. 12, 13, 14, 15

Thompson, C. G., Kim, R. S., Aloe, A. M., and Becker, B. J. (2017). Extracting the variance inflation factor and other multicollinearity diagnostics from typical regression results. *Basic and Applied Social Psychology*, 39(2):81–90. 10

Wackernagel, H. and Wackernagel, H. (2003). Ordinary kriging. *Multivariate geostatistics: an introduction with applications*, pages 79–88. 8

Wang, G., Peng, W., Zhang, L., and Zhang, J. (2023). Quantifying the impacts of natural and human factors on changes in npp using an optimal parameters-based geographical detector. *Ecological Indicators*, 155:111018. 1

Wang, Z., Wang, Y., Wu, S., and Du, Z. (2022). House price valuation model based on geographically neural network weighted regression: The case study of shenzhen, china. *ISPRS International Journal of Geo-Information*, 11(8):450. 19, 20

Wei, T. and Simko, V. (2024). *R package 'corrplot': Visualization of a Correlation Matrix*. (Version 0.95). 30

Wheeler, D. and Tiefelsdorf, M. (2005). Multicollinearity and correlation among local regression coefficients in geographically weighted regression. *Journal of Geographical Systems*, 7(2):161–187. 7

Wheeler, D. C. (2021). Geographically weighted regression. In *Handbook of regional science*, pages 1895–1921. Springer. 11, 12, 13

WMO (2022). State of the climate in africa 2021. pages WMO–No.1300. 2

Wu, D., Zhang, Y., and Xiang, Q. (2024). Geographically weighted random forests for macro-level crash frequency prediction. *Accident Analysis & Prevention*, 194:107370. 17

Wu, S., Gao, X., Lei, J., Zhou, N., Guo, Z., and Shang, B. (2022). Ecological environment quality evaluation of the sahel region in africa based on remote sensing ecological index. *Journal of Arid Land*, 14(1):14–33. 26

Wu, Y.-c. and Feng, J.-w. (2018). Development and application of artificial neural network. *Wireless Personal Communications*, 102:1645–1656. 19

Xiao, X., Wang, Q., Guan, Q., Zhang, Z., Yan, Y., Mi, J., and Yang, E. (2023). Quantifying the nonlinear response of vegetation greening to driving factors in longnan of china based on machine learning algorithm. *Ecological Indicators*, 151:110277. 2

Xu, C., Li, Y., Hu, J., Yang, X., Sheng, S., and Liu, M. (2012). Evaluating the difference between the normalized difference vegetation index and net primary productivity as the indicators of vegetation vigor assessment at landscape scale. *Environmental monitoring and assessment*, 184:1275–1286. 28

Yang, C., Zhai, G., Fu, M., and Sun, C. (2023). Spatiotemporal characteristics and influencing factors of net primary production from 2000 to 2021 in china. *Environmental Science and Pollution Research*, 30(39):91084–91094. 2, 6, 7, 22, 76

Yuan, Y., Cave, M., and Zhang, C. (2018). Using local moran's i to identify contamination hotspots of rare earth elements in urban soils of london. *Applied geochemistry*, 88:167–178. 32

Zhao, Z., Xu, Z., Hu, C., Wang, K., and Ding, X. (2024). Geographically weighted neural network considering spatial heterogeneity for landslide susceptibility mapping: A case study of yichang city, china. *Catena*, 234:107590. 21

Appendix A

Supplementary Material

A.1 Conference Contributions and Publications

A.1.1 Conference Presentation

An earlier version of this work was presented at the Wits Global Climate Finance Conference 2024, held at the Wits Fintech Hub, Johannesburg, South Africa, on 29 November 2024. This international academic conference, themed "Shaping Sustainable Finance for a Greener World", brought together experts from academia, policy, and industry. The session facilitated academic feedback and interdisciplinary discussion relevant to sustainable climate finance and spatial environmental modelling.

A.1.2 Manuscript in Preparation

The findings of this dissertation form the basis of a manuscript currently in preparation for submission to the International Journal of Data Science and Analytics (JDSA), under the Regular Stream. The article, titled "Geographically Weighted Statistical Machine Learning Methods for Predicting Net Primary Productivity in the Eastern Sahel Region", is in its final stages, with the Results and Discussion sections in progress. This work contributes to the field of spatial machine learning and environmental analytics and aligns with JDSA's thematic focus on applied data science and sustainability.

Appendix B

R Code

The R code used in this study is given below.

Raster Extraction

```
#  use the extract() function from the terra package
#  We first convert RasterLayer and spatial objects into SpatRaster
#  and SpatVector formats, respectively.

# import shapefile
study_area <- st_read("C:/Users/Public/Sahel","Country")

# Convert to SpatVector
s <- vect(study_area)

# Convert to SpatRaster
r <- rast(study_area)

# Now you can use the extract() function from terra

####
# 1. Soil Data
so = raster("merged_soil.tif")
```

```r
## Convert to SpatRaster
r_soil <- rast(so)

# Now you can use the extract() function from terra
soil_values = terra::extract(x = r_soil, y = s, fun = mean)

# Add extracted values to the original SpatVector's data
study_area$Soil <- soil_values[,2]

################################################################################
################################################################################
# 2. Temperature
t = raster("merged_temp.tif")
# # Convert to SpatRaster
r_temp <- rast(t)

# Now you can use the extract() function from terra
temp_values = terra::extract(x = r_temp, y = s, fun = mean)

# Add extracted values to the original SpatVector's data
s$Temp <- temp_values[,2]
study_area$Temp <- temp_values[,2]

################################################################################
################################################################################
# 3. DEM
d = raster("R_dem.tif")
# # Convert to SpatRaster
r_dem <- rast(d)
```

```
# Now you can use the extract() function from terra
dem_values = terra::extract(x = r_dem, y = s, fun = mean)

# Add extracted values to the original SpatVector's data
s$DEM <- dem_values[,2]
study_area$DEM <- dem_values[,2]

#############################################################################
#############################################################################
# 4. Rainfall
r = raster("merged_RF.tif")

# # Convert to SpatRaster
r_rain <- rast(r)

# Now you can use the extract() function from terra
rain_values = terra::extract(x = r_rain, y = s, fun = mean)

# Add extracted values to the original SpatVector's data
s$Rainfall <- rain_values[,2]
study_area$Rainfall <- rain_values[,2]

#############################################################################
#############################################################################
# 5. NDVI

# # Convert to SpatRaster
r_ndvi <- rast(n)

# Now you can use the extract() function from terra
ndvi_values = terra::extract(x = r_ndvi, y = s, fun = mean)
```

```
# Add extracted values to the original SpatVector's data
s$NPP <- ndvi_values[,2]
study_area$NPP <- ndvi_values[,2]

#Save datasets
st_write(study_area, "updated_polygon.shp")
writeVector(s, "updated_polygons.shp", overwrite = TRUE)
```

Handling Missing Values using Ordinary Kriging

```
################################################################################
################################################################################
################################################################################
#Load data
Output <- st_read(".","updated_polygons")

### 1. ORDINARY KRIGING FOR NPP DATA

# Calculate the centroid of each polygon
centroids <- st_centroid(Output)

# Extract the coordinates
coords <- st_coordinates(centroids)
coords <- as.data.frame(coords)

# Drop geometry column
df = st_drop_geometry(Output)
summary(is.na(df)) #Check Missing value

# Extract data for NPP since it has missing values
```

```
NPP <- as.data.frame(df[,3])
names(NPP)[1]<-"NPP"

# Convert the NDVI/NPP values to Standard Range by Dividing by 10000
NPP$NPP <- ifelse(!is.na(NPP$NPP), NPP$NPP / 10000, NA)

# Convert to spatial point
SPDF_NPP <- SpatialPointsDataFrame(coords = coords, data=NPP)

# Since the CRS has not yet be filled,
# re-projection is done before any further analysis
proj4string(SPDF_NPP) =CRS("+proj=longlat +datum=WGS84")

# Extract coordinates of points with missing NPP values
NA_coords_NPP <- coordinates(SPDF_NPP[is.na(SPDF_NPP$NPP), ])

# Create a data frame with the same number of rows as NA_coords_NPP
NPP_missing <- data.frame(NPP = rep(NA, nrow(NA_coords_NPP)))

# Create SpatialPointsDataFrame for missing points
missing_points_NPP <- SpatialPointsDataFrame(
                      coords = NA_coords_NPP,
                      data = NPP_missing,
                      proj4string = CRS("+proj=longlat +datum=WGS84"))

# Remove Rows with missing NPP values
NPP_clean <- SPDF_NPP[!is.na(SPDF_NPP$NPP), ]
df.NPP <- as.data.frame(NPP_clean)

# Create a grid for interpolation, including missing values
samp <- sp::spsample(x = NPP_clean, n = 5000, type = "regular")
```

```
proj4string(samp) <- CRS("+proj=longlat +datum=WGS84")
grid <- rbind(samp, missing_points_NPP)
gridded(samp) <- TRUE


#calculate appropriate transformation parameters
#using powerTransform() function of car package
powerTransform(df.NPP$NPP)
df.NPP$NPP.bc<-bcPower(df.NPP$NPP, 0.2222236)

# define x & y variables to coordinates
coordinates(df.NPP) = ~X+Y
proj4string(df.NPP) <- CRS("+proj=longlat +datum=WGS84")

# Variogram Modeling
# Generate an empirical variogram based on the data
v<-variogram(NPP~ 1, locations = NPP_clean, cloud=F)
plot(v)

# Determine the sill
psill = (max(v$gamma) + median(v$gamma))/2

#Determine the range
coord.2 <- as.data.frame(df.NPP@coords)
l = sqrt((diff(range(coord.2$X)))^2 + (diff(range(coord.2$Y)))^2)
range = 0.1 * l

# Determine the nugget effect
nugget <- min(v$gamma)
```

```
# Fit an estimated semivariogram to the data
m<-vgm(0.02,"Gau",1000,0.003)
m.f<-fit.variogram(v, m)
m.f
plot(v, pl=F,
     model=m.f,
     col="black",
     cex=0.9,
     lwd=0.5,
     lty=1,
     pch=19,
     main="Variogram and Fitted Model\n Box-Cox Transformed",
     xlab="Distance (m)",
     ylab="Semivariance")


# Perform Ordinary Kriging
OK<-krige(NPP~1,
          locations = NPP_clean,         # Data frame
          newdata=samp,              # Prediction grid
          model = m.f, nmax = 10)
summary(OK)

# Perform idw
idw = gstat::idw(formula = NPP~1, locations = NPP_clean, newdata = samp, idp = 1)
summary(idw)

# Cross-validation for IDW
idw_cv <- gstat::gstat.cv(gstat::gstat(formula = NPP ~ 1, locations = NPP_clean,
                                       set = list(idp = 1)))
summary(idw_cv)
```

```
# Cross-validation for Kriging
krig_cv <- gstat::gstat.cv(gstat::gstat(formula = NPP ~ 1, locations = NPP_clean,
                                        model = m.f, nmax = 10))
summary(krig_cv)

# Extract data for predicted values for both kriging and idw
data = data.frame(actual = NPP_clean$NPP,
                  idw = idw_cv$var1.pred,
                  krig = krig_cv$var1.pred)

# RMSE calculation
i = sqrt(mean((data$actual - data$idw)^2, na.rm = TRUE))
k = sqrt(mean((data$actual - data$krig)^2, na.rm = TRUE))


# Extract the interpolated values for missing points
NPP_interpolated <- OK$var1.pred[1:nrow(missing_points_NPP)]

SPDF_NPP$NPP[is.na(SPDF_NPP$NPP)] <- NPP_interpolated
df.1<-as.data.frame(SPDF_NPP)
head(df.1)

# Replace the Old NPP value with the interpolated NPP value
Output$NPP <- df.1[,1]

# Plot the results
st = st_as_sf(OK)
p1 = ggplot() +
           geom_sf(data = st,
          aes(color = var1.pred)) +
```

```
            viridis::scale_color_viridis(name = "NPP pred")
p2 = ggplot() +
  geom_sf(data = st,
          aes(color = var1.var)) +
  viridis::scale_color_viridis(name = "var pred")
grid.arrange(p1,p2, ncol = 2)



################################################################################
################################################################################
### 2. ORDINARY KRIGING FOR DEM DATA

# Calculate the centroid of each polygon
centroids <- st_centroid(Output)

# Extract the coordinates
coords <- st_coordinates(centroids)
coords <- as.data.frame(coords)

# Drop geometry column
df = st_drop_geometry(Output)
summary(is.na(df)) #Check Missing value

# Extract data for NPP since it has missing values
DEM <- as.data.frame(df[,5])
names(DEM)[1]<-"DEM"


# Convert to spatial point
SPDF_DEM <- SpatialPointsDataFrame(coords = coords, data=DEM)
```

```
# Since the CRS has not yet be filled,
# re-projection is done before any further analysis
proj4string(SPDF_DEM) =CRS("+proj=longlat +datum=WGS84")

# Extract coordinates of points with missing NPP values
NA_coords_DEM <- coordinates(SPDF_DEM[is.na(SPDF_DEM$DEM), ])

# Create a data frame with the same number of rows as NA_coords_NPP
DEM_missing <- data.frame(DEM = rep(NA, nrow(NA_coords_DEM)))

# Create SpatialPointsDataFrame for missing points
missing_points_DEM <- SpatialPointsDataFrame(
                        coords = NA_coords_DEM,
                        data = DEM_missing,
                        proj4string = CRS("+proj=longlat +datum=WGS84"))

# Remove Rows with missing NPP values
DEM_clean <- SPDF_DEM[!is.na(SPDF_DEM$DEM), ]
df.DEM <- as.data.frame(DEM_clean)

# Create a grid for interpolation, including missing values
samp <- sp::spsample(x = DEM_clean, n = 5000, type = "regular")
proj4string(samp) <- CRS("+proj=longlat +datum=WGS84")
grid <- rbind(samp, missing_points_DEM)
gridded(samp) <- TRUE


#calculate appropriate transformation parameters
#using powerTransform() function of car package
powerTransform(df.DEM$DEM)
```

```
df.DEM$DEM.bc<-bcPower(df.DEM$DEM, -0.5174856)

# define x & y variables to coordinates
coordinates(df.DEM) = ~X+Y
proj4string(df.DEM) <- CRS("+proj=longlat +datum=WGS84")

# Variogram Modeling
# Generate an empirical variogram based on the data
v<-variogram(DEM.bc~ 1, locations = df.DEM, cloud=F)
plot(v)

# Determine the sill
psill = (max(v$gamma) + median(v$gamma))/2

#Determine the range
coord.2 <- as.data.frame(df.DEM@coords)
l = sqrt((diff(range(coord.2$X)))^2 + (diff(range(coord.2$Y)))^2)
range = 0.1 * l

# Determine the nugget effect
nugget <- min(v$gamma)


# Fit an estimated semivariogram to the data
m<-vgm(psill = psill,model = "Gau",range = 800, nugget = nugget)
m.f<-fit.variogram(v, m)
m.f
plot(v, pl=F,
     model=m.f,
     col="black",
     cex=0.9,
```

```
     lwd=0.5,
     lty=1,
     pch=19,
     main="Variogram and Fitted Model\n Box-Cox Transformed",
     xlab="Distance (m)",
     ylab="Semivariance")


# Perform Ordinary Kriging
OK<-krige(DEM.bc~1,
          locations = df.DEM,         # Data frame
          newdata=samp,             # Prediction grid
          model = m.f,
          nmax = 10)
summary(OK)

# Perform idw
idw_DEM = gstat::idw(formula = DEM.bc~1,
                     locations = df.DEM,
                     newdata = samp, idp = 1)
summary(idw_DEM)

# Cross-validation for IDW
idw_cv_DEM <- gstat::gstat.cv(gstat::gstat(formula = DEM.bc ~ 1,
                                           locations = df.DEM,
                                           set = list(idp = 1)))
summary(idw_cv)

# Cross-validation for Kriging
krig_cv_DEM <- gstat::gstat.cv(gstat::gstat(formula = DEM.bc ~ 1,
                                model = m.f,
```

```
                                    nmax = 10))
summary(krig_cv)

# Extract data for predicted values for both kriging and idw
data = data.frame(actual = df.DEM$DEM.bc, idw = idw_cv_DEM$var1.pred,
                  krig = krig_cv_DEM$var1.pred)

# RMSE calculation
i = sqrt(mean((data$actual - data$idw)^2, na.rm = TRUE))
k = sqrt(mean((data$actual - data$krig)^2, na.rm = TRUE))

r2_krig = 1 - (sum((data$actual - data$krig)^2, na.rm = TRUE) /
       sum((data$actual - mean(data$actual, na.rm = TRUE))^2, na.rm = TRUE))

r2_idw = 1 - (sum((data$actual - data$idw)^2, na.rm = TRUE) /
                 sum((data$actual - mean(data$actual, na.rm = TRUE))^2,
                 na.rm = TRUE))

# Back Transformation
k1<-1/ -0.5174856
OK$OK.pred <-((OK$var1.pred * (-0.5174856) +1)^k1)
OK$OK.var <-((OK$var1.var * (-0.5174856) +1)^k1)
summary(OK)

# Extract the interpolated values for missing points
DEM_inter <- OK$OK.pred[1:nrow(missing_points_DEM)]

SPDF_DEM$DEM[is.na(SPDF_DEM$DEM)] <- DEM_inter
df.1<-as.data.frame(SPDF_DEM)
head(df.1)
```

```
# Replace the Old NPP value with the interpolated NPP value
Output$DEM <- df.1[,1]

# Plot the results
st = st_as_sf(OK)
p1 = ggplot() +
  geom_sf(data = st,
          aes(color = OK.pred)) +
  viridis::scale_color_viridis(name = "DEM pred")
p2 = ggplot() +
  geom_sf(data = st,
          aes(color = OK.var)) +
  viridis::scale_color_viridis(name = "var pred")
grid.arrange(p1,p2, ncol = 2)



################################################################################
################################################################################
### 3. ORDINARY KRIGING FOR Temp DATA

# Calculate the centroid of each polygon
centroids <- st_centroid(Output)

# Extract the coordinates
coords <- st_coordinates(centroids)
coords <- as.data.frame(coords)

# Drop geometry column
df = st_drop_geometry(Output)
summary(is.na(df)) #Check Missing value
```

```r
# Extract data for NPP since it has missing values
temp <- as.data.frame(df[,6])
names(temp)[1]<-"temp"


# Convert to spatial point
SPDF_temp <- SpatialPointsDataFrame(coords = coords, data=temp)

# Since the CRS has not yet be filled,
# re-projection is done before any further analysis
proj4string(SPDF_temp) =CRS("+proj=longlat +datum=WGS84")

# Extract coordinates of points with missing NPP values
NA_coords_temp <- coordinates(SPDF_temp[is.na(SPDF_temp$temp), ])

# Create a data frame with the same number of rows as NA_coords_NPP
temp_missing <- data.frame(temp = rep(NA, nrow(NA_coords_temp)))

# Create SpatialPointsDataFrame for missing points
missing_points_temp <- SpatialPointsDataFrame(
          coords = NA_coords_temp,
          data = temp_missing,
          proj4string =  CRS("+proj=longlat +datum=WGS84"))

# Remove Rows with missing NPP values
temp_clean <- SPDF_temp[!is.na(SPDF_temp$temp), ]
df.temp <- as.data.frame(temp_clean)

# Create a grid for interpolation, including missing values
samp <- sp::spsample(x = temp_clean, n = 5000, type = "regular")
proj4string(samp) <- CRS("+proj=longlat +datum=WGS84")
```

```
grid <- rbind(samp, missing_points_temp)
gridded(samp) <- TRUE


#calculate appropriate transformation parameters
#using powerTransform() function of car package
powerTransform(df.temp$temp)
df.temp$temp.bc<-bcPower(df.temp$temp,  34.16617)

# define x & y variables to coordinates
coordinates(df.temp) = ~X+Y
proj4string(df.temp) <- CRS("+proj=longlat +datum=WGS84")

# Variogram Modeling
# Generate an empirical variogram based on the data
v<-variogram(temp~ 1, locations = df.temp, cloud=F)
plot(v)

# Determine the sill
psill = (max(v$gamma) + median(v$gamma))/2

#Determine the range
coord.2 <- as.data.frame(df.DEM@coords)
l = sqrt((diff(range(coord.2$X)))^2 + (diff(range(coord.2$Y)))^2)
range = 0.1 * l

# Determine the nugget effect
nugget <- min(v$gamma)


# Fit an estimated semivariogram to the data
```

```
m<-vgm(psill = psill,model = "Gau",range = 1000, nugget = nugget)
m.f<-fit.variogram(v, m)
m.f
plot(v, pl=F,
     model=m.f,
     col="black",
     cex=0.9,
     lwd=0.5,
     lty=1,
     pch=19,
     main="Variogram and Fitted Model\n Box-Cox Transformed",
     xlab="Distance (m)",
     ylab="Semivariance")


# Perform Ordinary Kriging
OK<-krige(temp~1,
          locations = df.temp,         # Data frame
          newdata=samp,              # Prediction grid
          model = m.f,
          nmax = 10)
summary(OK)

# Perform idw
idw_temp = gstat::idw(formula = temp~1,
                                locations = df.temp,
                                newdata = samp,
                                idp = 1)
summary(idw_temp)

# Cross-validation for IDW
```

```
idw_cv <- gstat::gstat.cv(gstat::gstat(formula = temp ~ 1,
                                       locations = df.temp,
                                       set = list(idp = 1)))
summary(idw_cv)

# Cross-validation for Kriging
krig_cv <- gstat::gstat.cv(gstat::gstat(formula = temp ~ 1,
                                        locations = df.temp,
                                        model = m.f,
                                        nmax = 10))
summary(krig_cv)

# Extract data for predicted values for both kriging and idw
data = data.frame(actual = df.temp$temp, idw = idw_cv$var1.pred,
                  krig = krig_cv$var1.pred)

# RMSE calculation
i = sqrt(mean((data$actual - data$idw)^2, na.rm = TRUE))
k = sqrt(mean((data$actual - data$krig)^2, na.rm = TRUE))

r2_krig = 1 - (sum((data$actual - data$krig)^2, na.rm = TRUE) /
                 sum((data$actual - mean(data$actual, na.rm = TRUE))^2,
                 na.rm = TRUE))

r2_idw = 1 - (sum((data$actual - data$idw)^2, na.rm = TRUE) /
                sum((data$actual - mean(data$actual, na.rm = TRUE))^2,
                na.rm = TRUE))

# Back Transformation
k1<-1/ -0.6219563
OK$OK.pred <-((OK$var1.pred * (-0.6219563) +1)^k1)
```

```
OK$OK.var <-((OK$var1.var * (-0.6219563) +1)^k1)
summary(OK)

# Extract the interpolated values for missing points
temp_inter <- OK$var1.pred[1:nrow(missing_points_temp)]
summary(temp_inter)

SPDF_temp$temp[is.na(SPDF_temp$temp)] <- temp_inter
df.1<-as.data.frame(SPDF_temp)
head(df.1)

# Replace the Old NPP value with the interpolated NPP value
Output$Temp <- df.1[,1]



# Plot the results
st = st_as_sf(OK)
p1 = ggplot() +
  geom_sf(data = st,
          aes(color = var1.pred)) +
  viridis::scale_color_viridis(name = "Temp pred")
p2 = ggplot() +
  geom_sf(data = st,
          aes(color = var1.var)) +
  viridis::scale_color_viridis(name = "var pred")
grid.arrange(p1,p2, ncol = 2)



################################################################################
################################################################################
```

```
### 4. ORDINARY KRIGING FOR SOIL DATA

# Calculate the centroid of each polygon
centroids <- st_centroid(Output)

# Extract the coordinates
coords <- st_coordinates(centroids)
coords <- as.data.frame(coords)

# Drop geometry column
df = st_drop_geometry(Output)
summary(is.na(df)) #Check Missing value

# Extract data for NPP since it has missing values
DEM <- as.data.frame(df[,7])
names(DEM)[1]<-"DEM"



# Convert to spatial point
SPDF_DEM <- SpatialPointsDataFrame(coords = coords, data=DEM)

# Since the CRS has not yet be filled, re-projection is
#done before any further analysis
proj4string(SPDF_DEM) =CRS("+proj=longlat +datum=WGS84")

# Extract coordinates of points with missing NPP values
NA_coords_DEM <- coordinates(SPDF_DEM[is.na(SPDF_DEM$DEM), ])

# Create a data frame with the same number of rows as NA_coords_NPP
DEM_missing <- data.frame(DEM = rep(NA, nrow(NA_coords_DEM)))
```

```
# Create SpatialPointsDataFrame for missing points
missing_points_DEM <- SpatialPointsDataFrame(coords = NA_coords_DEM,
                          data = DEM_missing,
                          proj4string = CRS("+proj=longlat +datum=WGS84"))

# Remove Rows with missing NPP values
DEM_clean <- SPDF_DEM[!is.na(SPDF_DEM$DEM), ]
df.DEM <- as.data.frame(DEM_clean)

# Create a grid for interpolation, including missing values
samp <- sp::spsample(x = DEM_clean, n = 5000, type = "regular")
proj4string(samp) <- CRS("+proj=longlat +datum=WGS84")
grid <- rbind(samp, missing_points_DEM)
gridded(samp) <- TRUE


#calculate appropriate transformation parameters
#using powerTransform() function of car package
powerTransform(df.DEM$DEM)
df.DEM$DEM.bc<-bcPower(df.DEM$DEM, 2.247778)

# define x & y variables to coordinates
coordinates(df.DEM) = ~X+Y
proj4string(df.DEM) <- CRS("+proj=longlat +datum=WGS84")

# Variogram Modeling
# Generate an empirical variogram based on the data
v<-variogram(DEM.bc~ 1, locations = df.DEM, cloud=F)
plot(v)

# Determine the sill
```

```
psill = (max(v$gamma) + median(v$gamma))/2

#Determine the range
coord.2 <- as.data.frame(df.DEM@coords)
l = sqrt((diff(range(coord.2$X)))^2 + (diff(range(coord.2$Y)))^2)
range = 0.1 * l

# Determine the nugget effect
nugget <- min(v$gamma)



# Fit an estimated semivariogram to the data
m<-vgm(psill = psill,model = "Gau",range = 800, nugget = nugget)
m.f<-fit.variogram(v, m)
m.f
plot(v, pl=F,
     model=m.f,
     col="black",
     cex=0.9,
     lwd=0.5,
     lty=1,
     pch=19,
     main="Variogram and Fitted Model\n Box-Cox Transformed ",
     xlab="Distance (m)",
     ylab="Semivariance")


# Perform Ordinary Kriging
OK<-krige(DEM.bc~1,
          locations = df.DEM,         # Data frame
          newdata=samp,             # Prediction grid
```

```
            model = m.f,
            nmax = 10)
summary(OK)

# Perform idw
idw_DEM = gstat::idw(formula = DEM.bc~1,
                                locations = df.DEM,
                                newdata = samp,
                                idp = 1)
summary(idw_DEM)

# Cross-validation for IDW
idw_cv_DEM <- gstat::gstat.cv(gstat::gstat(formula = DEM ~ 1,
                                           locations = df.DEM,
                                           set = list(idp = 1)))
summary(idw_cv)

# Cross-validation for Kriging
krig_cv_DEM <- gstat::gstat.cv(gstat::gstat(formula = DEM ~ 1,
                                            locations = df.DEM,
                                            model = m.f,
                                            nmax = 10))
summary(krig_cv)

# Extract data for predicted values for both kriging and idw
data = data.frame(actual = df.DEM$DEM, idw = idw_cv_DEM$var1.pred,
                  krig = krig_cv_DEM$var1.pred)

# RMSE calculation
i = sqrt(mean((data$actual - data$idw)^2, na.rm = TRUE))
k = sqrt(mean((data$actual - data$krig)^2, na.rm = TRUE))
```

```
r2_krig = 1 - (sum((data$actual - data$krig)^2, na.rm = TRUE) /
                 sum((data$actual - mean(data$actual, na.rm = TRUE))^2,
                 na.rm = TRUE))

r2_idw = 1 - (sum((data$actual - data$idw)^2, na.rm = TRUE) /
                sum((data$actual - mean(data$actual, na.rm = TRUE))^2,
                na.rm = TRUE))

# Back Transformation
k1<-1/ 2.247778
OK$OK.pred <-((OK$var1.pred * (2.247778) +1)^k1)
OK$OK.var <-((OK$var1.var * (2.247778) +1)^k1)
summary(OK)

# Extract the interpolated values for missing points
DEM_inter <- OK$OK.pred[1:nrow(missing_points_DEM)]
summary(DEM_inter)

SPDF_DEM$DEM[is.na(SPDF_DEM$DEM)] <- DEM_inter
df.1<-as.data.frame(SPDF_DEM)
head(df.1)

# Replace the Old NPP value with the interpolated NPP value
Output$Soil <- df.1[,1]

# Plot the results
st = st_as_sf(OK)
p1 = ggplot() +
  geom_sf(data = st,
          aes(color = OK.pred)) +
```

```
  viridis::scale_color_viridis(name = "Soil pred")
p2 = ggplot() +
  geom_sf(data = st,
          aes(color = OK.var)) +
  viridis::scale_color_viridis(name = "var pred")
grid.arrange(p1,p2, ncol = 2)
```

EDA

```
################################EXPLONATORY DATA ANALYSIS#############################
#######################################################################################
#######################################################################################
library(ggplot2)
library(tidyr)
library(dplyr)
library(forcats)
library(RColorBrewer)
library(corrplot)

Output <- st_read("C:/Users/Public/Sahel","Complete")
df = st_drop_geometry(Output)

# Create plot: Country Distribution of Observations
p1 <- ggplot(Output, aes(y = fct_infreq(factor(COUNTRY)), fill = COUNTRY)) +
  geom_bar(stat = "count") +
  xlab("Count") +
  ylab(NULL) +  # Remove y-axis label
  scale_fill_manual("Countries", values = brewer.pal(11, "Set3")) +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
```

```
    axis.title.x = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 14, face = "bold"),
    legend.text = element_text(size = 12, face = "bold"))

ggsave("country.jpg", plot = p1, width = 12, height = 6)

# Scatter Plots of NPP vs. Soil, Rainfall, DEM, and Temp
p2 = Output %>%
  pivot_longer(cols = c(Soil, Rainfall, DEM, Temp), names_to = "Variable",
  values_to = "Value") %>%
  ggplot(aes(x = Value, y = NPP)) +
  # specify point characteristics
  geom_point(col = "tomato", shape = 20, size = 2.7) +
  xlab("Value") + ylab("NPP") +
  # specify a trend line and a theme/style
  geom_smooth(method = "lm", colour = "#DE2D26") +
  facet_wrap(~Variable, scales = "free_x", ncol = 2) +
  theme_minimal() +
  labs(title = "Scatter Plots of NPP vs. Soil, Rainfall, DEM, and Temp",
       x = "Predictor Values",
       y = "NPP") +
  theme(
    plot.title = element_text(size = 12),
    strip.text = element_text(size = 16, face = "bold"),  # Facet labels
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 14, face = "bold"),
    legend.text = element_text(size = 12, face = "bold")
  )
```

```r
ggsave("Scatterplot_data.jpg", plot = p2, width = 12, height = 6)

# Create faceted scatter plot with trend lines
p3 <- Output %>%
  pivot_longer(cols = c(Soil, Rainfall, DEM, Temp),
               names_to = "Variable",
               values_to = "Value") %>%
  ggplot(mapping = aes(x = Value, y = NPP, colour = COUNTRY)) +
  geom_point() +  # Slight transparency for overlap
  geom_smooth(method = "lm") +  # Trend line only
  facet_wrap(~ Variable, scales = "free_x", ncol = 2) +
  theme_minimal(base_size = 14) +  # Set base font size
  labs(
    title = "Scatter Plots of NPP vs. Soil, Rainfall, DEM, and Temp by Country",
    x = "Predictor Values",
    y = "NPP",
    colour = "Countries"
  ) +
  theme(
    plot.title = element_text(size = 12),
    strip.text = element_text(size = 16, face = "bold"),  # Facet labels
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 14, face = "bold"),
    legend.text = element_text(size = 12, face = "bold")
  )

ggsave("Scatterplot_country.jpg", plot = p3, width = 12, height = 6)

p4 <- ggplot(Output, aes(x = COUNTRY, y = NPP, fill = COUNTRY)) +
  geom_boxplot() +
```

```
  theme_minimal() +
  labs(title = "Boxplot of NPP by Country",
       x = "Countries",
       y = "NPP",
       fill = "Countries") +
  scale_fill_manual(values = c("Chad" = "indianred", "Niger" = "lightblue",
  "Sudan" = "lightgreen")) +
  theme(legend.position = "right")+
  theme(
    plot.title = element_text(size = 12),
    strip.text = element_text(size = 16, face = "bold"),  # Facet labels
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 14, face = "bold"),
    legend.text = element_text(size = 12, face = "bold")
  )

ggsave("Descriptive_stat.jpg", plot = p4, width = 12, height = 6)

# Plot histograms with density curves for NPP
p5 = ggplot() +
  geom_histogram(data = df, aes(x = NPP, y=..density..),
                 bins = 30, fill="#FF6666", col="white")+
  geom_density(data = df, aes(x = NPP), alpha=.5, fill="#FF6666") +
  labs(title = "Histogram and Density Curve for NPP",
       x = "NPP",
       y = "Density") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 9),
    strip.text = element_text(size = 16),
```

```
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))
ggsave("Hist_and_Den_NPP.jpg", plot = p5, width = 12, height = 6, dpi = 300)

# Plot histograms with density curves for each predictor variable

df_long <- df %>%
  pivot_longer(cols = c(DEM, Rainfall, Soil, Temp),
               names_to = "Variable",
               values_to = "Value")

p6 = ggplot(df_long, aes(x = Value)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "#FF6666",
                 color = "white", alpha = 0.6) +
  geom_density(alpha=.5, fill="#FF6666") +
  facet_wrap(~ Variable, scales = "free", ncol = 2) +
  labs(title = "Histograms and Density Curves of Predictor Variables",
       x = "Value",
       y = "Density") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 9),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

ggsave("Hist_and_Den_response.jpg", plot = p6, width = 12, height = 6, dpi = 300)

# Step 1: Create correlation matrix
Cor_matrix <- df %>%
  select_if(is.numeric) %>%
```

```
  cor(use = "complete.obs") %>%
  round(3)

# Step 2: Generate the correlation plot
dev.new()
corrplot(Cor_matrix,
         method = "number",
         type = "full",
         tl.col = "black",
         tl.cex = 1.6,
         tl.srt = 45,
         number.cex = 2.2,
         number.font = 2,
         number.col = "black",
         col = colorRampPalette(c("red", "lightgrey", "blue"))(200))
dev.off()

##############################################################################
##############################################################################
### 4.4 Mapping Spatial Properties

# 1. Net Primary Productivity
NPP_map <- tm_shape(Output)+
  tm_fill("NPP", title = "NPP", palette = "Reds",
          style = "kmeans", legend.hist = T)+
  tm_layout(frame = F, legend.outside = T,
            legend.hist.width = 1,
            legend.format = list(digits = 3),
            legend.outside.position = c("left", "top"),
            legend.text.size = 1,
            legend.title.size = 1,
```

```
            legend.text.fontface = "bold")+
  tm_borders(col = "black") +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom"))

tmap_save(NPP_map, filename = "NPP.png", width = 12, height = 6)

## 2. Predictor variavles

p1 <- tm_shape(Output) +
  tm_fill("DEM", palette = "Reds", style = "kmeans", legend.hist = T ,
  n = 5, title="(a)Elevation")+
  tm_layout(frame = F, legend.outside = T,
            legend.hist.width = 1,
            legend.format = list(digits = 3),
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.7,
            legend.title.size = 1,
            legend.text.fontface = "bold")+
  tm_borders(col = "black", lwd = 0.2) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom"))

p2 <- tm_shape(Output) +
  tm_fill("Rainfall", palette = "Reds", style = "kmeans", legend.hist = T ,
  n = 5, title="(b)Rainfall")+
  tm_layout(frame = F, legend.outside = T,
            legend.hist.width = 1,
            legend.format = list(digits = 3),
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.7,
```

```
            legend.title.size = 1,
            legend.text.fontface = "bold")+
  tm_borders(col = "black", lwd = 0.2) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom"))

p3 <- tm_shape(Output) +
  tm_fill("Soil", palette = "Reds", style = "kmeans",
  n = 5, legend.hist = T , title="(c)Soil Moisture")+
  tm_layout(frame = F, legend.outside = T,
            legend.hist.width = 1,
            legend.format = list(digits = 3),
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.7,
            legend.title.size = 1,
            legend.text.fontface = "bold")+
  tm_borders(col = "black", lwd = 0.2) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom"))

p4 <- tm_shape(Output) +
  tm_fill("Temp", palette = "Reds", style = "kmeans", legend.hist = T ,
  n = 5, title="(d)Temperature")+
  tm_layout(frame = F, legend.outside = T,
            legend.hist.width = 1,
            legend.format = list(digits = 3),
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.7,
            legend.title.size = 1,
            legend.text.fontface = "bold")+
  tm_borders(col = "black", lwd = 0.2) +
```

```
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom"))


combined_map <- tmap_arrange(p1,p2,p3,p4, nrow = 2)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "tmap_kmeans.png", width = 12, height = 6)


## 4.4.2 Point-Based Maps

# 1. NPP
dot_NPP <- tm_shape(Output) +
  tm_fill("grey70", n = 5) +
  tm_layout(frame = F, legend.outside = T,
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.7,
            legend.title.size = 1,
            legend.text.fontface = "bold") +
  tm_borders("black", lwd = 0.5) +
  # the points layer
  tm_shape(st_centroid(Output)) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom")) +
  tm_dots("NPP", size = 0.2, title = "NPP", shape = 19)


tmap_save(dot_NPP, filename = "tm_dots_NPP.png", width = 12, height = 6)
```

```
# 2. Predictor Variables

p1 <- tm_shape(Output) +
  tm_fill("grey70") +
  tm_layout(frame = F, legend.outside = T,
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold") +
  tm_borders("black", lwd = 0.5) +
  # the points layer
  tm_shape(st_centroid(Output)) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom")) +
  tm_dots("DEM", size = 0.2, title = "(a)Elevation", shape = 19)

p2 <- tm_shape(Output) +
  tm_fill("grey70") +
  tm_layout(frame = F, legend.outside = T,
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold") +
  tm_borders("black", lwd = 0.5) +
  # the points layer
  tm_shape(st_centroid(Output)) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom")) +
  tm_dots("Rainfall", size = 0.2, title = "(b)Rainfall", shape = 19)

p3 <- tm_shape(Output) +
```

```
  tm_fill("grey70") +
  tm_layout(frame = F, legend.outside = T,
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold") +
  tm_borders("black", lwd = 0.5) +
  # the points layer
  tm_shape(st_centroid(Output)) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom")) +
  tm_dots("Soil", size = 0.2, title = "(c)Soil Moisture", shape = 19)

p4 <- tm_shape(Output) +
  tm_fill("grey70") +
  tm_layout(frame = F, legend.outside = T,
            legend.outside.position = c("left", "top"),
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold") +
  tm_borders("black", lwd = 0.5) +
  # the points layer
  tm_shape(st_centroid(Output)) +
  tm_compass(position = c(0.005, 0.69)) +
  tm_scale_bar(position = c("left", "bottom")) +
  tm_dots("Temp", size = 0.2, title = "(d)Temperature", shape = 19)

combined_map <- tmap_arrange(p1,p2,p3,p4, nrow = 2)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "tm_dots_Predictors.png",
```

```
width = 12, height = 6)
```

OLS

```
Output <- st_read("C:/Users/Public/Sahel","Complete")
df = st_drop_geometry(Output)

#Standardize only the predictor variables
df[, 4:7] = scale(df[, 4:7])

# runs a linear model
model <- lm(NPP ~ DEM + Soil + Rainfall + Temp, data = df)

summary(model)

dev.new()
# Plot only the first diagnostic plot: Residuals vs Fitted
par(
  cex.main = 1.6,        # Title size
  font.main = 2,         # Title bold
  cex.lab = 1.4,         # Axis labels size
  font.lab = 2,          # Axis labels bold
  cex.axis = 1.2,        # Axis ticks size
  font.axis = 2          # Axis ticks bold
)

plot(model, which = 1)  # Plot only plot #1: Residuals vs Fitted


# we can use the par function if we want to plot them in a 2x2 frame
par(mfrow=c(2,2),mar = c(2, 2, 2, 2))
plot(model)
```

```
# determine studentised residuals and attach to OA.Census
# Studentized residuals are essentially residuals divided by
#their estimated standard deviation
# They help identify outliers in the dataset
s.resids <- rstudent(model)
df.resids = cbind(df, s.resids)

# Histogram of standardized residuals
hist_plot <- ggplot(df.resids, aes(s.resids)) +
  geom_histogram(bins = 30, fill = "blue") +
  theme_minimal(base_size = 14)+
  labs(title = "Histogram of Standardized Residuals",
       x = "Standardized Residuals", y = "Count")+
  theme(
    plot.title = element_text(size = 12),
    strip.text = element_text(size = 16, face = "bold"),  # Facet labels
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12, face = "bold"),
    legend.title = element_text(size = 14, face = "bold"),
    legend.text = element_text(size = 12, face = "bold")
  )

ggsave("Hist_residuals.jpg", plot = hist_plot, width = 12, height = 6)

# Mapping the residuals
resids<-residuals(model)

map.resids <- cbind(Output, resids)
# we need to rename the column header from the resids file
#in this case its the 6th column of map.resids
```

```
names(map.resids)[8] <- "resids"


# maps the residuals using the quickmap function from tmap
qtm(map.resids, fill = "resids")

# Or tmap

tm_shape(map.resids) +
  tm_polygons('resids',palette = "YlOrRd",
              title = "OLS residuals") +
  tm_layout(legend.position = c("left", "top"), legend.outside = T, frame = F) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


# OR

map1 <- tm_shape(map.resids) +
  tm_fill("resids", n = 5, style = "quantile", palette = "YlOrRd",
  title = "Residuals Values")  +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "OLS Residuals",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
```

```
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

tmap_save(map1, filename = "map_residuals.png", width = 12, height = 6)
```

GWR

```
##############################################################################
#####################################GWR#######################################

Output.scaled <- st_read("C:/Users/Public/Sahel","Scaled_data")

# There are 2 basic steps to any GW approach:

#1.Determine the optimal kernel bandwidth
#2.Fit the GW model.

#The GWmodel package has tools for determining the optimal GWR bandwidth,
#but requires the data to be transformed to sp format from sf:

# convert to sp
Output.sp = as(Output.scaled, "Spatial")

# determine the kernel bandwidth
# AIC is preferred to CV because it reflects model parsimony and
# its use tends to avoid over-fitting GWR models
bw <- bw.gwr(NPP ~ DEM + Soil + Rainfall + Temp,
             approach = "AIC",
             kernel = "gaussian",
```

```
              adaptive = T,
              data=Output.sp)

# Now create the GWR model with adaptive bandwidth stored in bw,
# and have a look at the outputs:

# fit the GWR model
m.gwr <- gwr.basic(NPP ~ DEM + Soil + Rainfall + Temp,
                   data = Output.sp,
                   bw = bw,
                   kernel = "gaussian",
                   adaptive = TRUE,
                   F123.test = TRUE)



#Evaluating the results

# To access the data table in the SDF slot @data is added to this.

tab.gwr <- rbind(apply(m.gwr$SDF@data[, 1:5], 2, summary), coef(model))
rownames(tab.gwr)[7] <- "Global"
tab.gwr <- round(tab.gwr, 4)

# And you can inspect this table
# note that the t() function transposes the table:
t(tab.gwr)

# # GWR outputs
names(m.gwr)
results <-as.data.frame(m.gwr$SDF)
```

```
summary(results)

names(results)

gwr.map <- cbind(Output.scaled, as.matrix(results))
names(gwr.map)

p = tm_shape(gwr.map) +
  tm_fill("Local_R2", n = 5, style = "quantile",
  title = "Local_R2 Values")  +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = F,
            main.title = "GWR Local R-squared",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

tmap_save(p, filename = "R2_GWR1.png", width = 12, height = 6)


# create tmap objects
map1 <- tm_shape(gwr.map) +
```

```
  tm_fill("DEM.1", n = 5, style = "quantile",
  title = "Coefficient Values")  +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(a) GWR:Elevation Coefficients",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map2 <- tm_shape(gwr.map) +
  tm_fill("Rainfall.1", n = 5, style = "quantile",
  title = "Coefficient Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(b) GWR:Rainfall Coefficients",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
```

```
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


map3 <- tm_shape(gwr.map) +
  tm_fill("Soil.1", n = 5, style = "quantile",
  title = "Coefficient Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(c) GWR:Soil Moisture Coefficients",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",        # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


map4 <- tm_shape(gwr.map) +
  tm_fill("Temp.1", n = 5, style = "quantile",
  title = "Coefficient Values") +
```

```
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = " (d) GWR:Temperature Coefficients",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

combined_map <- tmap_arrange(map1, map2, map3, map4, ncol = 2)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "coef_GWR.png", width = 12, height = 6)


# Extract t-values
tval = m.gwr$SDF$DEM_TV


# Identify significant t-values
signif = tval < -1.96 | tval > 1.96

# Categorize t-values
results$t_DEM <- cut(tval,
```

```
                        breaks=c(min(tval), -1.96, 1.96, max(tval)),
                        labels=c("sig", "nonsig", "sig"))


# Create the map
map_sig = tm_shape(m.gwr$SDF) +
  tm_fill(col = "t_DEM3", title = "DEM: significant", legend.hist = TRUE,
          midpoint = NA, textNA = "", colorNA = "white") +  # add fill
  tm_borders(col = "white", lwd = .1)  +  # add scale bar
  tm_layout(frame = F, bg.color = "white", legend.outside = TRUE)

# Add significant t-values layer
map_sig + tm_shape(m.gwr$SDF[signif,]) +
  tm_borders(col = "white", lwd = 0.5) # add borders
```

GWRF

```
Output.scaled <- st_read("C:/Users/Public/Sahel","Scaled_data")
df = st_drop_geometry(Output.scaled)

# Calculate the centroid of each polygon
centroids <- st_centroid(Output.scaled)

# Extract the coordinates
coords <- st_coordinates(centroids)

GRFbandwidth <- grf.bw(NPP ~ DEM + Temp + Rainfall + Soil, dataset = df,
                       kernel = "adaptive", coords = coords, bw.min = 20,
                       bw.max = 30, step = 1,forests = FALSE, geo.weigthed = T)
set.seed(1999)
grf.model <- grf(NPP ~ DEM + Temp + Rainfall + Soil,
                 dframe = df,
```

```
                  bw = GRFbandwidth$best ,
                  ntree = 1000,
                  mtry = 3,
                  kernel ="adaptive",
                  forests = TRUE,
                  coords = coords)


## Global Variable importance

# Assuming grf.model$Global.Model$variable.importance is a named vector
variable_importance <- grf.model$Global.Model$variable.importance

# Convert to a data frame for ggplot2
importance_df <- data.frame(
  Variable = names(variable_importance),
  Importance = variable_importance
)

# Create the horizontal bar plot
p1 =ggplot(importance_df, aes(x = reorder(Variable, Importance),
y = Importance)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +  # Flip coordinates to make it horizontal
  labs(title = "Global Variable Importance",
       x = "Variable",
       y = "Importance") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 9),
    strip.text = element_text(size = 16, face = "bold"),
```

```
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

ggsave("Global_Importance.jpg", plot = p1, width = 12, height = 6)

## Mapping local feature importance

results <-as.data.frame(grf.model$Local.Variable.Importance)
grf.map <- cbind(Output.scaled, as.matrix(results))

tm_shape(grf.map) +
  tm_polygons(c("DEM.1", "Temp.1", "Rainfall.1", "Soil.1"),
              title = "Local Feature Importance",
              palette = "-RdYlBu",
              n = 4) +
  tm_facets(free.scales = FALSE, nrow = 2, ncol = 2) +
  tm_layout(panel.labels = c("DEM", "Temperature", "Rainfall", "Soil"),
            title = "Local Feature Importance by Variable",
            legend.position = c("right", "bottom"))

# create tmap objects
map1 <- tm_shape(grf.map) +
  tm_fill("DEM.1", n = 5, style = "quantile",
  title = "IncMSE Values:DEM")  +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(a) GWRF:Elevation Variable Importance",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
```

```
            legend.title.size = 1,
            legend.text.fontface = "bold",          # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


map2 <- tm_shape(grf.map) +
  tm_fill("Rainfall.1", n = 5, style = "quantile",
  title = "IncMSE Values:Rainfall") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(b) GWRF:Rainfall Variable Importance",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",          # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map3 <- tm_shape(grf.map) +
  tm_fill("Soil.1", n = 5, style = "quantile",
```

```
  title = "IncMSE Values:Soil Moisture") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(c) GWRF:Soil Moisture Variable Importance",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map4 <- tm_shape(grf.map) +
  tm_fill("Temp.1", n = 5, style = "quantile",
  title = "IncMSE Values:Temperature") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(d) GWRF:Temperature Variable Importance",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
```

```
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

combined_map <- tmap_arrange(map1,map2,map3,map4, nrow = 2)

# Increase dimensions for visibility
tmap_save(combined_map,
        filename = "Local_GWRF_Importance.png",
        width = 12, height = 6)


## Partial Dependency Plots

library(pdp)

# Assuming grf.model$Global.Model is a ranger object
global_model <- grf.model$Global.Model

# Create a partial dependency plot for the 'Temp' feature
pdp_temp <- partial(global_model, pred.var = "Temp", train = df)


# Partial dependence for DEM
pdp_dem <- partial(global_model, pred.var = "DEM", train = df)


# Partial dependence for Soil
pdp_soil <- partial(global_model, pred.var = "Soil", train = df)
```

```r
# Partial dependence for Rainfall
pdp_rainfall <- partial(global_model, pred.var = "Rainfall", train = df)

library(ggplot2)

# Plot PDP for DEM
p1 <- ggplot(pdp_dem, aes(x = DEM, y = yhat)) +
  geom_line() +
  labs(title = "(a)Partial Dependence on DEM",
  y = "Average Prediction", x = "DEM") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 10, face = "bold"),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

# Plot PDP for Temperature
p2 <- ggplot(pdp_temp, aes(x = Temp, y = yhat)) +
  geom_line() +
  labs(title = "(d)Partial Dependence on Temperature",
  y = "Average Prediction", x = "Temperature") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 10, face = "bold"),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

# Plot PDP for Soil
p3 <- ggplot(pdp_soil, aes(x = Soil, y = yhat)) +
```

```
  geom_line() +
  labs(title = "(c)Partial Dependence on Soil Moisture",
       y = "Average Prediction", x = "Soil Moisture") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 10, face = "bold"),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

# Plot PDP for Rainfall
p4 <- ggplot(pdp_rainfall, aes(x = Rainfall, y = yhat)) +
  geom_line() +
  labs(title = "(b)Partial Dependence on Rainfall",
  y = "Average Prediction", x = "Rainfall") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 10, face = "bold"),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

# Arrange all the plots in a 2x2 grid
library(gridExtra)
pdp_all = grid.arrange(p1, p4, p3, p2, nrow = 2)

ggsave("Partial_Dependence.jpg", plot = pdp_all, width = 12, height = 6)

## The average local variable importance in the GW-RF model
# Extract variable names and mean importance values
variable_names <- names(grf.model$Local.Variable.Importance)
```

```
mean_importance <- grf.model$LocalModelSummary$l.VariableImportance$Mean

# Create a data frame for the variable importance values
importance_data <- data.frame(
  Variable = variable_names,
  MeanImportance = mean_importance
)

# Plot the bar chart
p1 = ggplot(importance_data, aes(x = reorder(Variable, MeanImportance),
y = MeanImportance)) +
  geom_bar(stat = "identity", fill = "steelblue", width = 0.7) +
  coord_flip() +  # Flip the axes for horizontal bars
  labs(x = "Variable",y = "Mean Local Importance") +
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 9),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))

ggsave("Mean_Local_Importance.jpg", plot = p1, width = 12, height = 6)

## Map the GWRF R2
names(grf.model)
results.1 <-as.data.frame(grf.model$LGofFit)
summary(results.1)

names(results.1)

grf.map.1 <- cbind(Output.scaled, as.matrix(results.1))
```

```r
names(grf.map.1)

# Define the breaks and labels for the local R2 ranges
breaks <- c(-Inf, 0.2, 0.4, 0.6, 0.8, Inf)
labels <- c(" 0.2", "(0.2, 0.4]", "(0.4, 0.6]", "(0.6, 0.8]", "> 0.8")

# Create the map with the specified ranges and label the legend
p <- tm_shape(grf.map.1) +
  tm_polygons("LM_Rsq100",
              breaks = breaks,
              labels = labels,
              title = "Local_R2 Intervals") +
  tm_layout(frame = FALSE,
            main.title = "(b) GWRF Local R-squared",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

# Increase dimensions for visibility
tmap_save(p, filename = "R2_GWRF.png", width = 12, height = 6)
```

```
# calculate the % of counties in the specified local R2 ranges using the LM_Rsq100

# Define the ranges for local R2
results.1 <- results.1 %>%
  mutate(R2_range = case_when(
    LM_Rsq100 <= 0.2 ~ " 0.2",
    LM_Rsq100 > 0.2 & LM_Rsq100 <= 0.4 ~ "(0.2, 0.4]",
    LM_Rsq100 > 0.4 & LM_Rsq100 <= 0.6 ~ "(0.4, 0.6]",
    LM_Rsq100 > 0.6 & LM_Rsq100 <= 0.8 ~ "(0.6, 0.8]",
    LM_Rsq100 > 0.8 ~ "> 0.8"
  ))

# Calculate the percentage of counties in each range
percentages <- results.1 %>%
  group_by(R2_range) %>%
  summarise(Count = n()) %>%
  mutate(Percentage = (Count / sum(Count)) * 100)
```

GWNN

```
################################################################################
################################################################################
################################GWNN############################################
################################################################################
# Load required libraries
library(gwann)
library(dplyr)

# Prepare data matrices
x_train <- as.matrix(df[, c("DEM", "Temp", "Rainfall", "Soil")]
y_train <- as.numeric(df$NPP)
w_train <- as.matrix(dist(coords))
```

```r
# Step 1: Train GWANN Model
# Running the GWANN model
set.seed(123)  # For reproducibility
model_gwann <- gwann(
  x_train = x_train,
  y_train = y_train,
  w_train = w_train,
  x_pred = x_train,  # Predict on the training set for now
  w_pred = w_train,  # Distance matrix for predictions
  norm = F,
  nrHidden = 6,  # Number of hidden neurons
  batchSize = 50,  # Batch size
  optimizer = "adam",  # Optimizer
  lr = 0.001,  # Learning rate
  kernel = "gaussian",  # Kernel type
  cv_patience = 999,
  iterations = 10424,
  bandwidth = 4,  # Let the model determine the bandwidth
  adaptive = TRUE,  # Use adaptive bandwidth
  bwSearch = "goldenSection",  # Bandwidth search method
  bwMin = min(w_train) / 4,  # Bandwidth lower bound
  bwMax = max(w_train) / 4,  # Bandwidth upper bound
  threads = 1, # Number of threads for computation
)
# Predictions
predicted_values <- diag(model_gwann$predictions)

# Evaluate model performance
print(paste("RMSE: ", sqrt(mean((predicted_values - y_train)^2))))
print(paste("Iterations: ", model_gwann$iterations))
```

```
print(paste("Bandwidth: ", model_gwann$bandwidth))
print(paste("Seconds: ", model_gwann$seconds))
print(paste("R2: ", cor(predicted_values, y_train)^2))



#residuals
residuals <- y_train - predicted_values

#plot predictions
s<-cbind(predicted_values, Output.scaled)

tm_shape(s) +
  tm_fill("predicted_values", n = 5, style = "quantile", title = "GWNN_pred") +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("left", "top"), legend.outside = T)

p = cbind(s, residuals)


# Heatmap for weights

library(ggplot2)
library(reshape2)

# Access the first list
first_list <- model_gwann$weights[[1]]



#Remove the NaN columns for visualization
```

```
first_list_clean <- first_list[, !apply(is.na(first_list), 2, any)]

# Convert the matrix to a data frame for ggplot2
first_df <- melt(first_list_clean)
colnames(first_df) <- c("Input", "Hidden", "Weight")

# Plot the heatmap
ggplot(first_df, aes(x = Hidden, y = Input, fill = Weight)) +
  geom_tile() +
  geom_text(aes(label = round(Weight, 3)), color = "black") +
  scale_fill_gradient2(low = "blue", high = "red",
  mid = "white", midpoint = 0,
                       limits = c(-0.5, 1)) +
  labs(title = "Input to Hidden Layer Weights", x = "Hidden Neurons",
  y = "Input Features") +
  scale_y_continuous(breaks = 1:5,
  labels = c("DEM", "Temperature", "Rainfall", "Soil", "Bias")) +
  scale_x_continuous(breaks = 1:6, labels = paste("Hidden", 1:6))

# Access the second list
second_list <- model_gwann$weights[[2]]

# Aggregate the weights by summing across all output neurons
aggregated_weights <- rowSums(second_list)

# Create a data frame for ggplot2
aggregated_weights_df <- data.frame(Hidden = 1:7, Weight = aggregated_weights)

# Plot the bar chart with custom x-axis labels
ggplot(aggregated_weights_df, aes(x = factor(Hidden), y = Weight,
fill = Weight)) +
```

```
  geom_bar(stat = "identity") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0) +
  labs(title = "Aggregated Hidden to Output Layer Weights", x = "Hidden Neurons",
       y = "Aggregated Weight") +
  scale_x_discrete(labels = c("Hidden 1", "Hidden 2", "Hidden 3", "Hidden 4",
                              "Hidden 5", "Hidden 6", "Bias"))+
  theme_minimal(base_size = 14) +  # Set base font size
  theme(
    plot.title = element_text(size = 12, face = "bold"),
    strip.text = element_text(size = 16, face = "bold"),
    axis.title = element_text(size = 14, face = "bold"),
    axis.text = element_text(size = 12,face = "bold"))
ggsave("aggregated.jpg", plot = p_all, width = 12, height = 6)



## map visualization of hidden layers
transposed_matrix <- t(second_list)

# Converting to a dataframe
df1 <- as.data.frame(transposed_matrix)

# Renaming the columns
colnames(df1) <- c("Hidden 1", "Hidden 2", "Hidden 3", "Hidden 4",
                   "Hidden 5", "Hidden 6", "Bias")

Hidden_df <-cbind(Output.scaled, as.matrix(df1))


# create tmap objects
map1 <- tm_shape(Hidden_df) +
```

```
  tm_fill("Hidden.1", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(a) Hidden 1 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


map2 <- tm_shape(Hidden_df) +
  tm_fill("Hidden.2", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(b) Hidden 2 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))
```

```
map3 <- tm_shape(Hidden_df) +
  tm_fill("Hidden.3", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(c) Hidden 3 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map4 <- tm_shape(Hidden_df) +
  tm_fill("Hidden.4", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(d) Hidden 4 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
```

```
  tm_scale_bar(position = c("left", "bottom"))

map5 <- tm_shape(Hidden_df) +
  tm_fill("Hidden.5", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(e) Hidden 5 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map6 <- tm_shape(Hidden_df) +
  tm_fill("Hidden.6", n = 5, style = "quantile", title = "Weights Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(f) Hidden 6 to Output Neuron Connection Weights",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.5,
            legend.title.size = 0.6,
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
```

```
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

combined_map <- tmap_arrange(map1,map2,map3,map4,map5,map6, ncol = 2)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "comb_map12.png", width = 12, height = 6)
```

Spatial Autocorrelation

```
##############################################################################
##############################################################################
########################Spatial Residuals Autocorrelation#####################
##############################################################################

# 1. GWR
library(spdep)

## SpatialPolygonsDataFrame
SPDF_gwr = m.gwr$SDF

# Calculate neighbours
neighbours <- poly2nb(SPDF_gwr)
neighbours

# We can plot the links between neighbours to visualise
#their distribution across space.

plot(SPDF_gwr, border = 'lightgrey')
plot(neighbours, coordinates(SPDF_gwr), add=TRUE, col='red')
```

```
# Convert the neighbour data to a listw object
listw <- nb2listw(neighbours)
listw

# global spatial autocorrelation
moran.test(SPDF_gwr$residual, listw)

# Running a local spatial autocorrelation

# creates a moran plot
moran <- moran.plot(SPDF_gwr$residual, listw = nb2listw(neighbours, style = "W"))

# creates a local moran output
local_gwr <- localmoran(x = SPDF_gwr$residual,
                        listw = nb2listw(neighbours,
                        style = "W"))
local_gwr
summary(local_gwr)

## 2.GWRF

# Extract numeric data frame with residuals
LGOfFit <- grf.model$LGofFit

# #Combine geometry with extracted residuals
SF_grf <- cbind(Output.scaled, as.matrix(LGOfFit))

# Convert the sf format to sp format
SPDF_grf <- as(SF_grf, "Spatial")
```

```
# Calculate neighbours
neighbours_grf <- poly2nb(SPDF_grf)
neighbours_grf

# Convert the neighbour data to a listw object
listw_grf <- nb2listw(neighbours_grf)
listw_grf

# global spatial autocorrelation
moran.test(SPDF_grf$LM_ResOOB, listw_grf)

# creates a local moran output
local_grf <- localmoran(x = SPDF_grf$LM_ResOOB,
                        listw = nb2listw(neighbours_grf, style = "W"))
local
summary(local_grf)


## 3. GWNN

# Ouploat the gwann sf file results
SF_gwann <- st_read("C:/Users/Public/Sahel","gwann")

# Convert the sf format to sp format
SPDF_gwann <- as(SF_gwann, "Spatial")

# Calculate neighbours
neighbours_gwann <- poly2nb(SPDF_gwann)
neighbours_gwann

# Convert the neighbour data to a listw object
```

```
listw_gwann <- nb2listw(neighbours_gwann)
listw_gwann

# global spatial autocorrelation
moran.test(SPDF_gwann$residls, listw_gwann)

# creates a local moran output
local_gwann <- localmoran(x = SPDF_gwann$residls,
                          listw = nb2listw(neighbours_gwann, style = "W"))
local
summary(local_gwann)

#######

# binds results to our polygon shapefile
moran_gwr <- cbind(Output, local_gwr)
moran_grf <- cbind(Output, local_grf)
moran_gwann <- cbind(Output, local_gwann)

# maps the results
tm_shape(moran_gwr) + tm_fill(col = "Ii", style = "quantile",
                              title = "local moran statistic")
tm_shape(moran_grf) + tm_fill(col = "Ii", style = "quantile",
                               title = "local moran statistic")
tm_shape(moran_gwann) + tm_fill(col = "Ii", style = "quantile",
                                title = "local moran statistic")

# Plot LISA clusters for GWR
quadrant_gwr <- vector(mode="numeric",length=nrow(local_gwr))

# centers the variable of interest around its mean
```

```
m.npp <- df$NPP - mean(df$NPP)

# centers the local Moran's around the mean
m.local_gwr <- local_gwr[,1] - mean(local_gwr[,1])

# significance threshold
signif <- 0.1

# builds a data quadrant
quadrant_gwr[m.npp >0 & m.local_gwr>0] <- 4
quadrant_gwr[m.npp <0 & m.local_gwr<0] <- 1
quadrant_gwr[m.npp <0 & m.local_gwr>0] <- 2
quadrant_gwr[m.npp >0 & m.local_gwr<0] <- 3
quadrant_gwr[local_gwr[,5]>signif] <- 0

# plot in r
# Add quadrant_gwr as a new column in the SPDF_gwr data frame
SPDF_gwr$quadrant_gwr <- factor(quadrant_gwr,
                                levels = c(0, 1, 2, 3, 4),
                                labels = c("Insignificant", "Low-Low",
                                           "Low-High", "High-Low",
                                           "High-High"))

# Define colors
colors <- c("white", "blue", rgb(0, 0, 1, alpha = 0.4),
            rgb(1, 0, 0, alpha = 0.4), "red")

# Plot using tmap
tm_shape(SPDF_gwr) +
  tm_polygons("quadrant_gwr",
              palette = colors,
```

```
              title = "LISA Clusters") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("right", "top"), legend.outside = T)


# Plot LISA clusters for GWRF
quadrant_grf <- vector(mode="numeric",length=nrow(local_grf))

# centers the variable of interest around its mean
m.npp <- df$NPP - mean(df$NPP)

# centers the local Moran's around the mean
m.local_grf <- local_grf[,1] - mean(local_grf[,1])

# significance threshold
signif <- 0.1

# builds a data quadrant
quadrant_grf[m.npp >0 & m.local_grf>0] <- 4
quadrant_grf[m.npp <0 & m.local_grf<0] <- 1
quadrant_grf[m.npp <0 & m.local_grf>0] <- 2
quadrant_grf[m.npp >0 & m.local_grf<0] <- 3
quadrant_grf[local_grf[,5]>signif] <- 0

# plot in r
# Add quadrant_gwr as a new column in the SPDF_gwr data frame
SPDF_grf$quadrant_grf <- factor(quadrant_grf,
                                levels = c(0, 1, 2, 3, 4),
                                labels = c("Insignificant", "Low-Low",
                                           "Low-High", "High-Low", "High-High"))
```

```r
# Define colors
colors <- c("white", "blue", rgb(0, 0, 1, alpha = 0.4),
            rgb(1, 0, 0, alpha = 0.4), "red")

# Plot using tmap
tm_shape(SPDF_grf) +
  tm_polygons("quadrant_grf",
              palette = colors,
              title = "LISA Clusters") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("right", "bottom"), legend.outside = F)


# Plot LISA clusters for GWANN
quadrant_gwann <- vector(mode="numeric",length=nrow(local_gwann))

# centers the variable of interest around its mean
m.npp <- df$NPP - mean(df$NPP)

# centers the local Moran's around the mean
m.local_gwann <- local_gwann[,1] - mean(local_gwann[,1])

# significance threshold
signif <- 0.1

# builds a data quadrant
quadrant_gwann[m.npp >0 & m.local_gwann>0] <- 4
quadrant_gwann[m.npp <0 & m.local_gwann<0] <- 1
quadrant_gwann[m.npp <0 & m.local_gwann>0] <- 2
```

```
quadrant_gwann[m.npp >0 & m.local_gwann<0] <- 3
quadrant_gwann[local_gwann[,5]>signif] <- 0

# plot in r
# Add quadrant_gwr as a new column in the SPDF_gwr data frame
SPDF_gwann$quadrant_gwann <- factor(quadrant_gwann,
                                    levels = c(0, 1, 2, 3, 4),
                                    labels = c("Insignificant", "Low-Low",
                                               "Low-High", "High-Low", "High-High"))

# Define colors
colors <- c("white", "blue", rgb(0, 0, 1, alpha = 0.4),
            rgb(1, 0, 0, alpha = 0.4), "red")

# Plot using tmap
tm_shape(SPDF_gwann) +
  tm_polygons("quadrant_gwann",
              palette = colors,
              title = "LISA Clusters") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("left", "top"), legend.outside = T)


##########Mapping residuals and local statistics

# create tmap objects
map1 <- tm_shape(SPDF_gwr) +
  tm_fill("residual", n = 5, style = "quantile",
  title = "Local Residuals Values") +
  tm_borders(col = "black", lwd = 0.2) +
```

```
  tm_layout(frame = FALSE,
            main.title = "(a) GWR Local Residuals",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map2 <- tm_shape(SPDF_gwr) +
  tm_polygons("quadrant_gwr", palette = colors, title = "LISA Clusters") +
  tm_layout(frame = FALSE,
            main.title = "(b) GWR Local Moran's I",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))
```

```
map3 <- tm_shape(SPDF_grf) +
  tm_fill("LM_ResOOB", n = 5, style = "quantile",
  title = "Local Residuals Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(c) GWRF Local Residuals",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",        # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map4 <- tm_shape(SPDF_grf) +
  tm_polygons("quadrant_grf", palette = colors, title = "LISA Clusters")+
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(d) GWRF Local Moran's I",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",        # Bold legend values
```

```
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))

map5 <- tm_shape(SPDF_gwann) +
  tm_fill("residls", n = 5, style = "quantile",
  title = "Local Residuals Values") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
            main.title = "(e) GWNN Local Residuals",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",        # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


map6 <- tm_shape(SPDF_gwann) +
  tm_polygons("quadrant_gwann", palette = colors, title = "LISA Clusters") +
  tm_borders(col = "black", lwd = 0.2) +
  tm_layout(frame = FALSE,
```

```
            main.title = "(f) GWNN Local Moran's I",
            main.title.position = "left",
            main.title.color = "black",
            main.title.size = 1,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.text.fontface = "bold",         # Bold legend values
            legend.title.fontface = "bold",
            legend.position = c("left", "center"),
            legend.outside = TRUE,
            legend.height = 10) +
  tm_compass(position = c(0.005, 0.69))+
  tm_scale_bar(position = c("left", "bottom"))


combined_map <- tmap_arrange(map1, map2, map3, map4, map5, map6, ncol = 2)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "model_residuals.png", width = 12, height = 6)


##############################################################################
##############################################################################

map1 <- tm_shape(SF_gwann) +
  tm_fill("prdctd_", n = 5, style = "quantile", title = "GWNN-Prediction") +
  tm_layout(frame = FALSE,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.outside=T,
            legend.position = c("right", "bottom"),
```

```
            title.size = 1)

map2 <- tm_shape(SPDF_gwr) +
  tm_fill("yhat", n = 5, style = "quantile", title = "GWR-Prediction") +
  tm_layout(frame = FALSE,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.outside=T,
            legend.position = c("right", "bottom"),
            title.size = 1)

map3 <- tm_shape(SF_grf) +
  tm_fill("LM_yfitOOB", n = 5, style = "quantile", title = "GWRF-Prediction") +
  tm_layout(frame = FALSE,
            legend.text.size = 0.8,
            legend.title.size = 1,
            legend.outside=T,
            legend.position = c("right", "bottom"),
            title.size = 1)

map4 <- tm_shape(SPDF_gwr) +
  tm_polygons("quadrant_gwr",
              palette = colors,
              title = "Local Moran's I (GWR)") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("right", "bottom"), legend.outside = T)

map5 <- tm_shape(SPDF_grf) +
  tm_polygons("quadrant_grf",
              palette = colors,
```

```
              title = "Local Moran's I (GWRF)") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("right", "bottom"), legend.outside = T)

map6 <- tm_shape(SPDF_gwann) +
  tm_polygons("quadrant_gwann",
              palette = colors,
              title = "Local Moran's I (GWNN)") +
  tm_legend(title.size = 1) +
  tm_layout(frame = FALSE, legend.text.size = 0.5, legend.title.size = 0.6,
            legend.position = c("right", "bottom"), legend.outside = T)

combined_map <- tmap_arrange(map2, map3, map1, map4, map5, map6, ncol = 3)

# Increase dimensions for visibility
tmap_save(combined_map, filename = "comb_map.png", width = 15, height = 10)
```

Cross Validation

```
##############################################################################
###################################CV#########################################
##############################################################################
library(GWmodel)

# Generate 10-fold indices
set.seed(123)  # For reproducibility
folds <- createFolds(1:nrow(Output.scaled), k = 10, list = TRUE)

# Initialize vectors to store performance metrics
mse_gwr <- c()
```

```
rmse_gwr <- c()
r2_gwr <- c()
mae_gwr <- c()  # New vector for MAE

# Loop through each fold
for (i in 1:10) {
  # Separate training and testing data
  train_index <- unlist(folds[-i])  # Combine all folds except the current one
  test_index <- unlist(folds[i])    # Current fold as the test set

  train_gwr <- Output.scaled[train_index, ]
  test_gwr <- Output.scaled[test_index, ]

  # Train the GWR model
  gwr_model <- gwr.predict(
    NPP ~ DEM + Soil + Rainfall + Temp,
    data = train_gwr,
    predictdata = test_gwr,
    bw = 20,
    kernel = "gaussian",
    adaptive = TRUE
  )

  # Get predictions
  pred_gwr <- gwr_model$SDF$prediction

  # Calculate metrics for this fold
  mse <- mean((pred_gwr - test_gwr$NPP)^2)
  rmse <- sqrt(mse)
  mae <- mean(abs(pred_gwr - test_gwr$NPP))  # MAE calculation
  r2 <- cor(pred_gwr, test_gwr$NPP)^2
```

```
  # Store results
  mse_gwr <- c(mse_gwr, mse)
  rmse_gwr <- c(rmse_gwr, rmse)
  mae_gwr <- c(mae_gwr, mae)
  r2_gwr <- c(r2_gwr, r2)
}

# Calculate average metrics over folds
avg_mse <- mean(mse_gwr)
avg_rmse <- mean(rmse_gwr)
avg_mae <- mean(mae_gwr)  # Average MAE
avg_r2 <- mean(r2_gwr)

# Output the metrics
cat("Average MSE:", avg_mse, "\n")
cat("Average RMSE:", avg_rmse, "\n")
cat("Average MAE:", avg_mae, "\n")  # Print MAE
cat("Average R2:", avg_r2, "\n")



################################################################################
################################################################################
################################################################################
library(caret)

# Set up to store performance metrics
mse_grf <- c()
rmse_grf <- c()
r2_grf <- c()
mae_grf <- c()
```

```r
# Create 10 folds
set.seed(123) # Ensure reproducibility
folds_grf <- createFolds(df$NPP, k = 10, list = TRUE)

# Perform 10-fold cross-validation
for (i in 1:10) {
  # Extract train and test indices
  test_index.grf <- folds_grf[[i]]
  train_index.grf <- setdiff(1:nrow(df), test_index.grf)

  # Split the data
  train_grf <- df[train_index.grf, ]
  test_grf <- df[test_index.grf, ]

  # Extract coordinates for test data
  test_coords <- coords[test_index.grf, ]
  test_grf <- cbind(test_grf, test_coords)
  colnames(test_grf)[(ncol(test_grf)-1):ncol(test_grf)] <- c("X", "Y")

  # Train the GRF model with the training data
  grf_model <- grf(NPP ~ DEM + Temp + Rainfall + Soil,
                   dframe = train_grf,
                   bw = 24,
                   ntree = 1000,
                   mtry = 3,
                   kernel = "adaptive",
                   forests = TRUE,
                   coords = coords[train_index.grf, ])

  # Make predictions, ensuring we drop geometry
```

```
  pred_grf <- predict.grf(grf_model,
                          new.data = test_grf,
                          x.var.name = "X",
                          y.var.name = "Y",
                          local.w = 1,
                            global.w = 0)

  # Calculate performance metrics
  mse <- mean((pred_grf - test_grf$NPP) ^ 2)
  rmse <- sqrt(mse)
  mae <- mean(abs(pred_grf - test_grf$NPP))
  r2 <- cor(pred_grf, test_grf$NPP) ^ 2

  # Store metrics
  mse_grf <- c(mse_grf, mse)
  rmse_grf <- c(rmse_grf, rmse)
  mae_grf <- c(mae_grf, mae)
  r2_grf <- c(r2_grf, r2)
}

# Calculate average metrics over folds
avg_mse_grf <- mean(mse_grf)
avg_rmse_grf <- mean(rmse_grf)
avg_mae_grf <- mean(mae_grf)
avg_r2_grf <- mean(r2_grf)

# Output the metrics
cat("Average MSE:", avg_mse_grf, "\n")
cat("Average RMSE:", avg_rmse_grf, "\n")
cat("Average MAE:", avg_mae_grf, "\n")
cat("Average R²:", avg_r2_grf, "\n")
```

```
################################################################################
################################################################################
################################################################################
library(caret)

# Prepare the data
x <- as.matrix(df[, c("DEM", "Rainfall", "Temp", "Soil")]) # Independent variables
y <- as.numeric(df[, "NPP"]) # Dependent variable
dm <- as.matrix(dist(coords)) # Distance matrix based on spatial coordinates

# Set up to store performance metrics
mse_gwnn <- c()
rmse_gwnn <- c()
mae_gwnn <- c()
r2_gwnn <- c()

# Create 10 folds
set.seed(123) # Ensure reproducibility
folds_gwnn <- createFolds(y, k = 10, list = TRUE)

# Perform 10-fold cross-validation
for (i in 1:10) {
  # Extract train and test indices
  test_index.gwnn <- folds_gwnn[[i]]
  train_index.gwnn <- setdiff(1:nrow(x), test_index.gwnn)

  # Split the data
  x_train <- x[train_index.gwnn, ]
  y_train <- y[train_index.gwnn]
```

```r
x_test <- x[test_index.gwnn, ]
y_test <- y[test_index.gwnn]
w_train <- dm[train_index.gwnn, train_index.gwnn]
w_pred <- dm[train_index.gwnn, test_index.gwnn]

# Fit the GWANN model

r <- gwann(
  x_train = x_train,            # Training data for predictors
  y_train = y_train,            # Training data for response
  w_train = w_train,            # Distance weights for training data
  x_pred = x_test,              # Prediction data for predictors
  w_pred = w_pred,              # Distance weights for prediction data
  norm = F,
  nrHidden = 6,                 # Number of hidden neurons
  batchSize = 50,               # Batch size
  optimizer = "adam",
  kernel = "gaussian",
  lr = 0.01,                     # Learning rate
  cv_patience = 999,            # Patience for cross-validation
  adaptive = T,                 # Use fixed bandwidth  # Bandwidth search method
  bandwidth = 4,
  threads = 1                   # Number of threads
)

# Predictions
predicted_values <- diag(r$predictions)

# Calculate performance metrics
mse <- mean((predicted_values - y_test) ^ 2)
rmse <- sqrt(mse)
```

```
  mae <- mean(abs(predicted_values - y_test))
  r2 <- cor(predicted_values, y_test) ^ 2

  # Store metrics
  mse_gwnn <- c(mse_gwnn, mse)
  rmse_gwnn <- c(rmse_gwnn, rmse)
  mae_gwnn <- c(mae_gwnn, mae)
  r2_gwnn <- c(r2_gwnn, r2)
}

# Calculate average metrics over folds
avg_mse_gwnn <- mean(mse_gwnn)
avg_rmse_gwnn <- mean(rmse_gwnn)
avg_mae_gwnn <- mean(mae_gwnn)
avg_r2_gwnn <- mean(r2_gwnn)

# Output the metrics
cat("Average MSE:", avg_mse_gwnn, "\n")
cat("Average RMSE:", avg_rmse_gwnn, "\n")
cat("Average MAE:", avg_mae_gwnn, "\n")
cat("Average R²:", avg_r2_gwnn, "\n")


##############################################################################
##############################################################################
##############################################################################
# Prepare the data
x_nnet <- as.matrix(df[, c("DEM", "Rainfall", "Temp", "Soil")])
y_nnet <- as.numeric(df[, "NPP"]) # Dependent variable

# Set up to store performance metrics
```

```
mse_nnet <- c()
rmse_nnet <- c()
mae_nnet <- c()
r2_nnet <- c()

# Create 10 folds for cross-validation
library(caret)
set.seed(123) # Ensure reproducibility
folds_nnet <- createFolds(y_nnet, k = 10, list = TRUE)

# Perform 10-fold cross-validation
for (i in 1:10) {
  # Extract train and test indices
  test_index.nnet <- folds_nnet[[i]]
  train_index.nnet <- setdiff(1:nrow(x_nnet), test_index.nnet)

  # Split the data into training and testing
  x_train.nnet <- x_nnet[train_index.nnet, ]
  y_train.nnet <- y_nnet[train_index.nnet]
  x_test.nnet <- x_nnet[test_index.nnet, ]
  y_test.nnet <- y_nnet[test_index.nnet]

  # Train the Neural Network model
  nn <- nnet(
    x_train = x_train.nnet,
    y_train = y_train.nnet,
    x_pred = x_test.nnet,
    norm = TRUE,
    nrHidden = 6,             # Number of hidden neurons
    batchSize = 50,           # Batch size
    optimizer = "adam",       # Optimizer
```

```
    lr = 0.01,                # Learning rate
    linOut = TRUE,            # Linear output layer
    iterations = NA,
    cv_max_iterations = Inf,
    cv_patience = 999,
    cv_folds = 10,
    cv_repeats = 1,
    permutations = 0,
    threads = 1
  )

  # Get predictions
  pred_nnet <- nn$predictions

  # Calculate metrics
  mse <- mean((pred_nnet - y_test.nnet)^2)
  rmse <- sqrt(mse)
  mae <- mean(abs(pred_nnet - y_test.nnet))
  r2 <- cor(pred_nnet, y_test.nnet)^2

  # Store metrics
  mse_nnet <- c(mse_nnet, mse)
  rmse_nnet <- c(rmse_nnet, rmse)
  mae_nnet <- c(mae_nnet, mae)
  r2_nnet <- c(r2_nnet, r2)
}

# Calculate average metrics over folds
avg_mse.nnet <- mean(mse_nnet)
avg_rmse.nnet <- mean(rmse_nnet)
avg_mae.nnet <- mean(mae_nnet)
```

```r
avg_r2.nnet <- mean(r2_nnet)

# Output the metrics
cat("Average MSE:", avg_mse.nnet, "\n")
cat("Average RMSE:", avg_rmse.nnet, "\n")
cat("Average MAE:", avg_mae.nnet, "\n")
cat("Average R2:", avg_r2.nnet, "\n")



################################################################################
################################################################################

# Required library for creating folds
library(caret)

# Prepare the data
set.seed(123) # For reproducibility
folds_ols <- createFolds(df$NPP, k = 10, list = TRUE)

# Initialize vectors to store metrics
mse_ols <- c()
rmse_ols <- c()
mae_ols <- c()
r2_ols <- c()

# Perform 10-fold cross-validation
for (i in 1:10) {
  # Get train and test indices
  test_indices <- folds_ols[[i]]
  train_ols <- df[-test_indices, ]
  test_ols <- df[test_indices, ]
```

```r
  # Fit the OLS model on training data
  model <- lm(NPP ~ DEM + Soil + Rainfall + Temp, data = train_ols)

  # Predict on test data
  pred_ols <- predict(model, newdata = test_ols)

  # Extract actual values
  actual <- test_ols$NPP

  # Calculate performance metrics
  mse <- mean((pred_ols - actual) ^ 2)
  rmse <- sqrt(mse)
  mae <- mean(abs(pred_ols - actual))
  r2 <- cor(pred_ols, actual) ^ 2

  # Store metrics
  mse_ols <- c(mse_ols, mse)
  rmse_ols <- c(rmse_ols, rmse)
  mae_ols <- c(mae_ols, mae)
  r2_ols <- c(r2_ols, r2)
}

# Calculate average metrics over folds
avg_mse_ols <- mean(mse_ols)
avg_rmse_ols <- mean(rmse_ols)
avg_mae_ols <- mean(mae_ols)
avg_r2_ols <- mean(r2_ols)

# Output the metrics
cat("Average MSE:", avg_mse_ols, "\n")
```

```r
cat("Average RMSE:", avg_rmse_ols, "\n")
cat("Average MAE:", avg_mae_ols, "\n")
cat("Average R2:", avg_r2_ols, "\n")


################################################################################
################################################################################
################################################################################
# Required libraries
library(caret)
library(randomForest)

# Prepare the data
set.seed(123) # For reproducibility
folds_rf <- createFolds(df$NPP, k = 10, list = TRUE)

# Initialize vectors to store metrics
mse_rf <- c()
rmse_rf <- c()
mae_rf <- c()
r2_rf <- c()

# Perform 10-fold cross-validation
for (i in 1:10) {
  # Get train and test indices
  test_indices.rf <- folds_rf[[i]]
  train_rf <- df[-test_indices.rf, ]
  test_rf <- df[test_indices.rf, ]

  # Fit the Random Forest model on training data
  rf_model <- randomForest(NPP ~ DEM + Soil + Rainfall + Temp,
```

```
                          data = train_rf,
                          ntree = 1000,
                          mtry = 3,
                          importance = TRUE)

  # Predict on test data
  pred_rf <- predict(rf_model, newdata = test_rf)

  # Extract actual values
  actual <- test_rf$NPP

  # Calculate performance metrics
  mse <- mean((pred_rf - actual) ^ 2)
  rmse <- sqrt(mse)
  mae <- mean(abs(pred_rf - actual))
  r2 <- cor(pred_rf, actual) ^ 2

  # Store metrics
  mse_rf <- c(mse_rf, mse)
  rmse_rf <- c(rmse_rf, rmse)
  mae_rf <- c(mae_rf, mae)
  r2_rf <- c(r2_rf, r2)
}

# Calculate average metrics over folds
avg_mse_rf <- mean(mse_rf)
avg_rmse_rf <- mean(rmse_rf)
avg_mae_rf <- mean(mae_rf)
avg_r2_rf <- mean(r2_rf)

# Output the metrics
```

```
cat("Average MSE:", avg_mse_rf, "\n")
cat("Average RMSE:", avg_rmse_rf, "\n")
cat("Average MAE:", avg_mae_rf, "\n")
cat("Average R2:", avg_r2_rf, "\n")

################################################################################
################################################################################
#create data frame of actual and predicted values
values_rf <- data.frame(actual=test_rf$NPP, predicted=pred_rf)
values_ols <- data.frame(actual=test_ols$NPP, predicted=pred_ols)
values_nnet <- data.frame(actual=y_test.nnet, predicted=pred_nnet)
values_gwann <- data.frame(actual=y_test, predicted=predicted_values)
values_grf <- data.frame(actual=test_grf$NPP, predicted=pred_grf)
values_gwr <- data.frame(actual=test_gwr$NPP, predicted=pred_gwr)


################################################################################
# Load necessary library
library(ggplot2)


# Function to plot and calculate metrics
plot_model <- function(data, title) {
  r_sq <- cor(data$predicted, data$actual) ^ 2
  rmse <- sqrt(mean((data$predicted - data$actual)^2))
  x_limits <- range(data$actual)
  y_limits <- range(data$predicted)

  ggplot(data, aes(x = actual, y = predicted)) +
    geom_point(color = "grey") +
    geom_smooth(method = "lm", color = "blue", se = FALSE) +
```

```
    labs(title = title, x = "Actual", y = "Predicted") +
    annotate("text", x = 0.1, y = 0.45,
             label = paste("R² =", round(r_sq, 2), "\nRMSE =", round(rmse, 2)),
                           hjust = 0) +
    theme_minimal() +
    xlim(x_limits) +
    ylim(y_limits) +
    theme_minimal(base_size = 14) +  # Set base font size
    theme(
      plot.title = element_text(size = 12, face = "bold"),
      strip.text = element_text(size = 16, face = "bold"),
      axis.title = element_text(size = 14, face = "bold"),
      axis.text = element_text(size = 12,face = "bold"))
}


# Plot each model
p1 <- plot_model(values_ols, "(a) OLS")
p2 <- plot_model(values_rf, "(b) RF")
p3 <- plot_model(values_nnet, "(c) NN")
p4 <- plot_model(values_gwr, "(d) GWR")
p5 <- plot_model(values_grf, "(e) GWRF")
p6 <- plot_model(values_gwann, "(f) GWNN")

# Arrange plots in a 2x2 grid
dev.new()
library(gridExtra)
grid.arrange(p1, p2, p3, p4, p5, p6, ncol = 2)

ggsave("predictive_perfomance.jpg", plot = p_all, width = 12, height = 6)
```

SCALABILITY AND COMPUTATIONAL COST

```
################################################################################
################################################################################
##################SCALABILITY AND COMPUTATIONAL COST#############################
################################################################################
################################################################################

## This analysis involved using subsets of the dataset,
## representing 25%, 50%, and 75% of the original data,
## to train and assess the models.

## To achive the percentage split, Stratified Sampling was utilised
## In order to ensure that the sampled data(subsets)
## look similar to the original data

### 1. Stratified Sampling Code for 25% Subset

## Firstly, in the dataset we have a character variable called COUNTRY
## Convert "COUNTRY" to factor
Output.scaled$COUNTRY <- as.factor(Output.scaled$COUNTRY)

Output.scaled$COUNTRY%>%levels() # Chad, Niger, Sudan 3 levels
                                 # The split is going to revolve around this Variable


## What is the proportion of "Chad", "Niger", and "Sudan" in the "COUNTRY" Variable
Output.scaled %>%
  group_by(COUNTRY) %>%
  summarise(fred = n()) %>%
  mutate(prep = fred/nrow(Output.scaled)) ## Remember there are 939 record
                                          ## in the dataset
```

```
# COUNTRY     fred    prep
# 1. Chad     427     0.455
# 2. Niger    179     0.187
# 3. Sudan    336     0.358


## We conduct "Stratified Sampling" when we want to preserve the proportion of
## "Chad", "Niger", and "Sudan" in the sampled data same as original data

## In other words, we want 45.5% "Chad", 18.7% "Niger", and 35.8% "Sudan" of
## the sampled data after splitting the original data and
## that sampled data must represent the original data


### 25% Stratified Sampling

## Package needed to do Stratified Sampling:rsample
library(rsample)

## Conduct Stratified Sampling using initial_split()
## For 25 % split
## Notice: prop = 0.25 --> assign 25% of the dataset to
                           # use for scalability analysis
## Notice: strata = COUNTRY --> variable whose proportion of
                                 # "Chad", "Niger", and "Sudan" we are interested in

stratify_25 <- initial_split(Output.scaled,
                             prop = 0.25,
                             strata = COUNTRY)

stratify_25
```

```
## <Training/Testing/Total>
## <234/705/939>
## This means among 939 observations, 234 (25% of the dataset) will be used to
## assess scalabilty and computational cost in Local Models


## Create the dataset of the samole 25% for the analysis using "training()"
stratify_25_data <- training(stratify_25)

stratify_25_data %>% nrow()  #234
## We see 234 observations, 25% of the original data


## Now lets check the proportion of "Chad", "Niger", and "Sudan" for
## our new sampled data (stratify_25_data)
## We want to see if the distribution is preseved in ou new 25% sampled data
stratify_25_data %>%
  group_by(COUNTRY) %>%
  summarise(fred = n()) %>%
  mutate(prep = fred/nrow(stratify_25_data))
## We see 45.3% of Chad, 18.9% of Niger, and 35.9% of Sudan !!
## Proportion is preserved


#### SIMALAR PROCEDURE IS MAINTAINED OF OTHER 50% AND 75% SPLIT

### 2. Stratified Sampling Code for 50% Subset

stratify_50 <- initial_split(Output.scaled,
                             prop = 0.50,
                             strata = COUNTRY)
```

```
stratify_50

stratify_50_data <- training(stratify_50)

stratify_50_data %>% nrow()

stratify_50_data %>%
  group_by(COUNTRY) %>%
  summarise(fred = n()) %>%
  mutate(prep = fred/nrow(stratify_50_data))

### 3. Stratified Sampling Code for 75% Subset

stratify_75 <- initial_split(Output.scaled,
                             prop = 0.75,
                             strata = COUNTRY)

stratify_75

stratify_75_data <- training(stratify_75)

stratify_75_data %>% nrow()

stratify_75_data %>%
  group_by(COUNTRY) %>%
  summarise(fred = n()) %>%
  mutate(prep = fred/nrow(stratify_75_data))
```

```r
############################## SCALABILITY ASSESSMENT ##############################


#################################### 1.GWR ####################################
###  (a) 25% with GWR
library(GWmodel)

set.seed(123)
train_index_gwr.25 <- sample(1:nrow(stratify_25_data),
                             size = 0.9 * nrow(stratify_25_data))
train_gwr.25 <- stratify_25_data[train_index_gwr.25, ]
test_gwr.25 <- stratify_25_data[-train_index_gwr.25, ]

pre_gwr.25 <- gwr.predict(NPP ~ DEM + Soil + Rainfall + Temp,
                        data = train_gwr.25 ,
                        predictdata = test_gwr.25,
                        bw = 20 ,
                        kernel="gaussian",
                        adaptive=T)

# Compute training duration
training_time <- pre_gwr.25$timings$stop - pre_gwr.25$timings$start

# Output training time
cat("Training Time:", training_time, "seconds\n")

# Get predictions
pred_gwr.25 <- pre_gwr.25$SDF$prediction

# Calculate metrics
mse <- mean((pred_gwr.25 - test_gwr.25$NPP)^2)
```

```
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwr.25 - test_gwr.25$NPP))
r2 <- cor(pred_gwr.25, test_gwr.25$NPP)^2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R2:", r2, "\n")


###  (b) 50% with GWR
library(GWmodel)

set.seed(123)
train_index_gwr.50 <- sample(1:nrow(stratify_50_data),
                             size = 0.9 * nrow(stratify_50_data))
train_gwr.50 <- stratify_50_data[train_index_gwr.50, ]
test_gwr.50 <- stratify_50_data[-train_index_gwr.50, ]

pre_gwr.50 <- gwr.predict(NPP ~ DEM + Soil + Rainfall + Temp,
                          data = train_gwr.50 ,
                          predictdata = test_gwr.50,
                          bw = 20 ,
                          kernel="gaussian",
                          adaptive=T)

# Compute training duration
training_time <- pre_gwr.50$timings$stop - pre_gwr.50$timings$start

# Output training time
```

```
cat("Training Time:", training_time, "seconds\n")

# Get predictions
pred_gwr.50 <- pre_gwr.50$SDF$prediction

# Calculate metrics
mse <- mean((pred_gwr.50 - test_gwr.50$NPP)^2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwr.50 - test_gwr.50$NPP))
r2 <- cor(pred_gwr.50, test_gwr.50$NPP)^2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R2:", r2, "\n")


### (c) 75% with GWR
library(GWmodel)

set.seed(123)
train_index_gwr.75 <- sample(1:nrow(stratify_75_data),
                             size = 0.9 * nrow(stratify_75_data))
train_gwr.75 <- stratify_75_data[train_index_gwr.75, ]
test_gwr.75 <- stratify_75_data[-train_index_gwr.75, ]

pre_gwr.75 <- gwr.predict(NPP ~ DEM + Soil + Rainfall + Temp,
                          data = train_gwr.75 ,
                          predictdata = test_gwr.75,
                          bw = 20 ,
```

```
                          kernel="gaussian",
                          adaptive=T)

# Compute training duration
training_time <- pre_gwr.75$timings$stop - pre_gwr.75$timings$start

# Output training time
cat("Training Time:", training_time, "seconds\n")

# Get predictions
pred_gwr.75 <- pre_gwr.75$SDF$prediction

# Calculate metrics
mse <- mean((pred_gwr.75 - test_gwr.75$NPP)^2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwr.75 - test_gwr.75$NPP))
r2 <- cor(pred_gwr.75, test_gwr.75$NPP)^2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R2:", r2, "\n")


##################################### 2.GWRF ###################################

### (a) 25% with GWRF

library(SpatialML)
```

```r
df_25 = st_drop_geometry(stratify_25_data)

# Calculate the centroid of each polygon
centroids_25 <- st_centroid(stratify_25_data)

# Extract the coordinates
coords_25 <- st_coordinates(centroids_25)

# Set seed for reproducibility
set.seed(123)

# Split data into 90% training and 10% testing
train_index_grf.25 <- sample(1:nrow(df_25), size = 0.9 * nrow(df_25))
train_grf.25 <- df_25[train_index_grf.25, ]
test_grf.25 <- df_25[-train_index_grf.25, ]

# Extract coordinates for test data
test_coords.25 <- coords_25[-train_index_grf.25, ]
test_grf.25 <- cbind(test_grf.25, test_coords.25)
colnames(test_grf.25)[(ncol(test_grf.25)-1):ncol(test_grf.25)] <- c("X", "Y")

# Train the GRF model on the training data
set.seed(1999)
grf_model.25 <- grf(NPP ~ DEM + Temp + Rainfall + Soil,
                    dframe = train_grf.25,
                    bw = 24,
                    ntree = 1000,
                    mtry = 3,
                    kernel = "adaptive",
                    forests = TRUE,
                    coords = coords_25[train_index_grf.25, ])
```

```r
# Make predictions on the test set
pred_grf.25 <- predict.grf(grf_model.25,
                           new.data = test_grf.25,
                           x.var.name = "X",
                           y.var.name = "Y",
                           local.w = 1,
                           global.w = 0)

# Calculate performance metrics
mse <- mean((pred_grf.25 - test_grf.25$NPP) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_grf.25 - test_grf.25$NPP))
r2 <- cor(pred_grf.25, test_grf.25$NPP) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")


### (b) 50% with GWRF

library(SpatialML)

df_50 = st_drop_geometry(stratify_50_data)

# Calculate the centroid of each polygon
centroids_50 <- st_centroid(stratify_50_data)
```

```
# Extract the coordinates
coords_50 <- st_coordinates(centroids_50)

# Set seed for reproducibility
set.seed(123)

# Split data into 90% training and 10% testing
train_index_grf.50 <- sample(1:nrow(df_50), size = 0.9 * nrow(df_50))
train_grf.50 <- df_50[train_index_grf.50, ]
test_grf.50 <- df_50[-train_index_grf.50, ]

# Extract coordinates for test data
test_coords.50 <- coords_50[-train_index_grf.50, ]
test_grf.50 <- cbind(test_grf.50, test_coords.50)
colnames(test_grf.50)[(ncol(test_grf.50)-1):ncol(test_grf.50)] <- c("X", "Y")

# Train the GRF model on the training data
set.seed(1999)
grf_model.50 <- grf(NPP ~ DEM + Temp + Rainfall + Soil,
                    dframe = train_grf.50,
                    bw = 24,
                    ntree = 1000,
                    mtry = 3,
                    kernel = "adaptive",
                    forests = TRUE,
                    coords = coords_50[train_index_grf.50, ])

# Make predictions on the test set
pred_grf.50 <- predict.grf(grf_model.50,
                           new.data = test_grf.50,
                           x.var.name = "X",
```

```
                        y.var.name = "Y",
                        local.w = 1,
                        global.w = 0)

# Calculate performance metrics
mse <- mean((pred_grf.50 - test_grf.50$NPP) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_grf.50 - test_grf.50$NPP))
r2 <- cor(pred_grf.50, test_grf.50$NPP) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")


### (c) 75% with GWRF

library(SpatialML)

df_75 = st_drop_geometry(stratify_75_data)

# Calculate the centroid of each polygon
centroids_75 <- st_centroid(stratify_75_data)

# Extract the coordinates
coords_75 <- st_coordinates(centroids_75)

# Set seed for reproducibility
set.seed(123)
```

```r
# Split data into 90% training and 10% testing
train_index_grf.75 <- sample(1:nrow(df_75), size = 0.9 * nrow(df_75))
train_grf.75 <- df_75[train_index_grf.75, ]
test_grf.75 <- df_75[-train_index_grf.75, ]

# Extract coordinates for test data
test_coords.75 <- coords_75[-train_index_grf.75, ]
test_grf.75 <- cbind(test_grf.75, test_coords.75)
colnames(test_grf.75)[(ncol(test_grf.75)-1):ncol(test_grf.75)] <- c("X", "Y")

# Train the GRF model on the training data
set.seed(1999)
grf_model.75 <- grf(NPP ~ DEM + Temp + Rainfall + Soil,
                    dframe = train_grf.75,
                    bw = 24,
                    ntree = 1000,
                    mtry = 3,
                    kernel = "adaptive",
                    forests = TRUE,
                    coords = coords_75[train_index_grf.75, ])

# Make predictions on the test set
pred_grf.75 <- predict.grf(grf_model.75,
                           new.data = test_grf.75,
                           x.var.name = "X",
                           y.var.name = "Y",
                           local.w = 1,
                           global.w = 0)

# Calculate performance metrics
```

```r
mse <- mean((pred_grf.75 - test_grf.75$NPP) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_grf.75 - test_grf.75$NPP))
r2 <- cor(pred_grf.75, test_grf.75$NPP) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")


################################### 2.GWNN ###################################

### (a) 25% with GWNN

library(caret)


# Prepare the data
x_25 <- as.matrix(df_25[, c("DEM", "Rainfall", "Temp", "Soil")]) # Independent
          # variables
y_25 <- as.numeric(df_25[, "NPP"]) # Dependent variable
dm_25 <- as.matrix(dist(coords_25)) # Distance matrix based on spatial coordinates

# Set seed for reproducibility
set.seed(123)

# Split data into 90% training and 10% testing
train_index_gwnn.25 <- sample(1:nrow(df_25), size = 0.9 * nrow(df_25))
x_train.25 <- x_25[train_index_gwnn.25, ]
```

```
y_train.25 <- y_25[train_index_gwnn.25]
x_test.25 <- x_25[-train_index_gwnn.25, ]
y_test.25 <- y_25[-train_index_gwnn.25]
w_train.25 <- dm_25[train_index_gwnn.25, train_index_gwnn.25]
w_pred.25 <- dm_25[train_index_gwnn.25, -train_index_gwnn.25]

# Fit the GWNN model
set.seed(123)
start_time <- Sys.time()
gwnn_model.25 <- gwann(
  x_train = x_train.25,           # Training data for predictors
  y_train = y_train.25,           # Training data for response
  w_train = w_train.25,           # Distance weights for training data
  x_pred = x_test.25,             # Prediction data for predictors
  w_pred = w_pred.25,             # Distance weights for prediction data
  norm = FALSE,
  nrHidden = 6,                 # Number of hidden neurons
  batchSize = 50,               # Batch size
  optimizer = "adam",
  kernel = "gaussian",
  lr = 0.01,                     # Learning rate
  cv_patience = 999,            # Patience for cross-validation
  adaptive = TRUE,              # Use fixed bandwidth
  bandwidth = 4,
  threads = 1                   # Number of threads
)

# Capture end time after training
end_time <- Sys.time()

# Compute the total training time
```

```
training_time <- end_time - start_time
cat("Training Time:", training_time, "seconds\n")
# Predictions
pred_gwnn.25 <- diag(gwnn_model.25$predictions)

# Calculate performance metrics
mse <- mean((pred_gwnn.25 - y_test.25) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwnn.25 - y_test.25))
r2 <- cor(pred_gwnn.25, y_test.25) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")


### (b) 50% with GWNN

library(caret)


# Prepare the data
x_50 <- as.matrix(df_50[, c("DEM", "Rainfall", "Temp", "Soil")]) # Independent
          # variables
y_50 <- as.numeric(df_50[, "NPP"]) # Dependent variable
dm_50 <- as.matrix(dist(coords_50)) # Distance matrix based on spatial coordinates

# Set seed for reproducibility
set.seed(123)
```

```r
# Split data into 90% training and 10% testing
train_index_gwnn.50 <- sample(1:nrow(df_50), size = 0.9 * nrow(df_50))
x_train.50 <- x_50[train_index_gwnn.50, ]
y_train.50 <- y_50[train_index_gwnn.50]
x_test.50 <- x_50[-train_index_gwnn.50, ]
y_test.50 <- y_50[-train_index_gwnn.50]
w_train.50 <- dm_50[train_index_gwnn.50, train_index_gwnn.50]
w_pred.50 <- dm_50[train_index_gwnn.50, -train_index_gwnn.50]

# Fit the GWNN model
set.seed(123)
start_time <- Sys.time()
gwnn_model.50 <- gwann(
  x_train = x_train.50,           # Training data for predictors
  y_train = y_train.50,           # Training data for response
  w_train = w_train.50,           # Distance weights for training data
  x_pred = x_test.50,             # Prediction data for predictors
  w_pred = w_pred.50,             # Distance weights for prediction data
  norm = FALSE,
  nrHidden = 6,                 # Number of hidden neurons
  batchSize = 50,               # Batch size
  optimizer = "adam",
  kernel = "gaussian",
  lr = 0.01,                     # Learning rate
  cv_patience = 999,            # Patience for cross-validation
  adaptive = TRUE,              # Use fixed bandwidth
  bandwidth = 4,
  threads = 1                   # Number of threads
)
```

```r
# Capture end time after training
end_time <- Sys.time()

# Compute the total training time
training_time <- end_time - start_time
cat("Training Time:", training_time, "seconds\n")
# Predictions
pred_gwnn.50 <- diag(gwnn_model.50$predictions)

# Calculate performance metrics
mse <- mean((pred_gwnn.50 - y_test.50) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwnn.50 - y_test.50))
r2 <- cor(pred_gwnn.50, y_test.50) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")


### (c) 75% with GWNN

library(caret)


# Prepare the data
x_75 <- as.matrix(df_75[, c("DEM", "Rainfall", "Temp", "Soil")]) # Independent
          # variables
y_75 <- as.numeric(df_75[, "NPP"]) # Dependent variable
```

```r
dm_75 <- as.matrix(dist(coords_75)) # Distance matrix based on spatial coordinates

# Set seed for reproducibility
set.seed(123)

# Split data into 90% training and 10% testing
train_index_gwnn.75 <- sample(1:nrow(df_75), size = 0.9 * nrow(df_75))
x_train.75 <- x_75[train_index_gwnn.75, ]
y_train.75 <- y_75[train_index_gwnn.75]
x_test.75 <- x_75[-train_index_gwnn.75, ]
y_test.75 <- y_75[-train_index_gwnn.75]
w_train.75 <- dm_75[train_index_gwnn.75, train_index_gwnn.75]
w_pred.75 <- dm_75[train_index_gwnn.75, -train_index_gwnn.75]

# Fit the GWNN model
set.seed(123)
start_time <- Sys.time()
gwnn_model.75 <- gwann(
  x_train = x_train.75,           # Training data for predictors
  y_train = y_train.75,           # Training data for response
  w_train = w_train.75,           # Distance weights for training data
  x_pred = x_test.75,             # Prediction data for predictors
  w_pred = w_pred.75,             # Distance weights for prediction data
  norm = FALSE,
  nrHidden = 6,                 # Number of hidden neurons
  batchSize = 50,               # Batch size
  optimizer = "adam",
  kernel = "gaussian",
  lr = 0.01,                     # Learning rate
  cv_patience = 999,            # Patience for cross-validation
  adaptive = TRUE,              # Use fixed bandwidth
```

```
  bandwidth = 4,
  threads = 1                    # Number of threads
)

# Capture end time after training
end_time <- Sys.time()

# Compute the total training time
training_time <- end_time - start_time
cat("Training Time:", training_time, "seconds\n")
# Predictions
pred_gwnn.75 <- diag(gwnn_model.75$predictions)

# Calculate performance metrics
mse <- mean((pred_gwnn.75 - y_test.75) ^ 2)
rmse <- sqrt(mse)
mae <- mean(abs(pred_gwnn.75 - y_test.75))
r2 <- cor(pred_gwnn.75, y_test.75) ^ 2

# Output the metrics
cat("MSE:", mse, "\n")
cat("RMSE:", rmse, "\n")
cat("MAE:", mae, "\n")
cat("R²:", r2, "\n")
```

The R code is also available at the following link: https://colab.research.google.com/drive/1MbGx9nakBH3fBSSwkB_wRUS7s6gLQz6v?usp=sharing