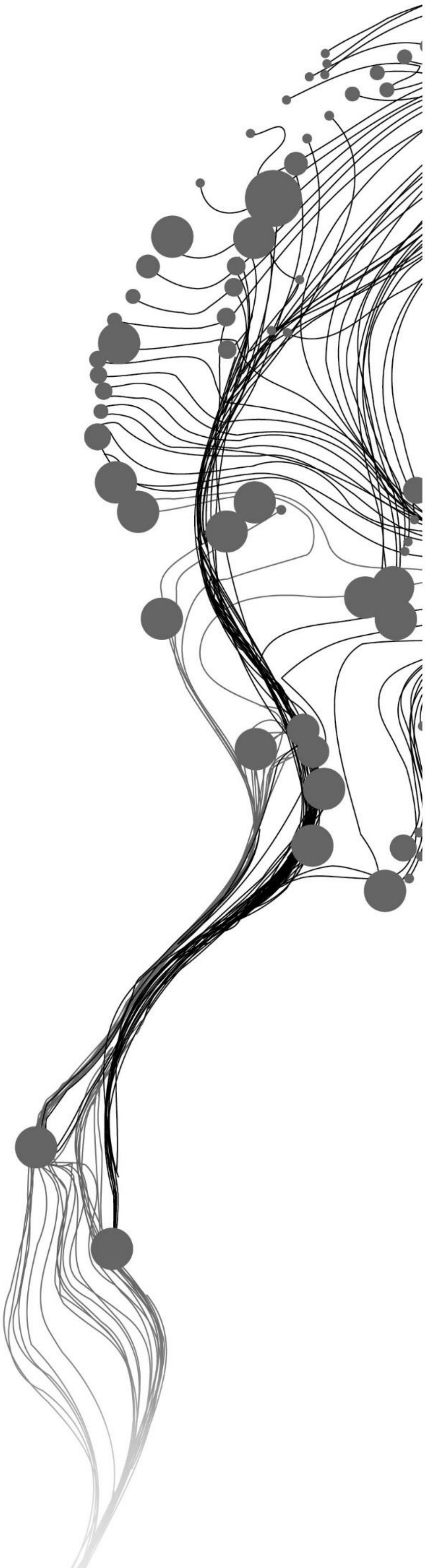
Digital Twin-Based Planning Support System for Urban Heat Island Mitigation

AMIR AFZALINEZHAD

July, 2024

SUPERVISORS:
Dr., M.K, Mila
Dr., P.N, Pirouz



Digital Twin-Based Planning Support System for Urban Heat Island Mitigation

AMIR AFZALINEZHAD
Enschede, The Netherlands, July 2024

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: Urban Planning and Management

SUPERVISORS:
Dr., M.K, Mila
Dr., P.N, Pirouz

THESIS ASSESSMENT BOARD:
Dr., J.M, Javier (Chair)
Dr., S.G, Srinidhi (External Examiner, ITC, University of Twente)

DISCLAIMER

This document describes work undertaken as part of a programme of study at the Faculty of Geo-Information Science and Earth Observation of the University of Twente. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the Faculty.

ABSTRACT

Urban growth is accompanied by increased impervious surfaces and urban building densities, contributing to the (UHI) phenomenon. This phenomenon refers to areas in cities with higher temperatures than rural areas. This phenomenon has a negative impact on human health and quality of life by increasing heat stress and the urban environment. Due to UHI's adverse effects, multiple studies have been conducted to address this challenge and mitigate UHI in urban areas. This study introduces an innovative Digital Twin-Based Planning Support System (DT-PSS) that can help urban planners mitigate the UHI formation and its effect in cities. This methodological research explored the development of a DT-PSS within the Unreal Engine (UE) platform for Wuppertal City, Germany. The research integrated real-time temperature data, a Machine Learning (ML) model, and remote sensing data to predict the temperature and assess the impact of urban planning scenarios on UHI ahead of implementation.

This research is divided into three parts: data analysis, ML model training, and DT-PSS creation. Nine variables are selected as predictor variables of LST from the literature: Normalized Difference Vegetation Index (NDVI), Normalized Difference Water Index (NDWI), Normalized Difference built-up Index (NDBI), Patch Density (PD), Edge Density (ED), Aggregation Index (AI), population density, Land use, and Digital Elevation Model (DEM). Among the selected variables, NDVI, NDBI, and NDWI showed the highest correlation with LST. The study used four different regression models to train them based on the predictor variables: Random Forest (RF), Support Vector Machine (SVM), Artificial Neural Networks (ANNs), and Polynomial regression mode to predict LST and UHI. The RF model demonstrated high accuracy in predicting LST, with an R-squared value of 0.86 and a Mean Absolute Error (MAE) of 1.09, outperforming other ML models.

To create DT-PSS, a 3D city model of Wuppertal is developed in City Engine and imported into the UE to reduce the computational workload for the UE platform. The trained RF model and real-time temperature data from sensors in Wuppertal City are imported into UE to include prediction capability to the DT-PSS. Integrating the RF model and sensor data into the UE allowed for real-time updates and scenario assessment. The DT-PSS prototype is developed for a small neighbourhood as a test area. The DT-PSS performance is evaluated by holding a workshop with Wuppertal municipality officials. Feedback is received regarding tool usability and the potential to enhance the performance of the tool for future research. The research also found that involving stakeholders from the beginning step is one of the primary and fundamental requirements in the process of creating and designing a DT-PSS. User requirements and needs should be met since they are the final users of the tool.

The DT-PSS suggested in this study provides urban planners with an innovative tool to evaluate the effects of different planning/mitigation scenarios on urban temperature and UHI. These research findings proved that satellite imagery provides valuable insights regarding the historical and current data and the state of land cover and temperature. This research also identified several limitations, such as data availability, GIS software and UE interoperability, and computation constraints. However, the findings of this research highlight the potential of the proposed DT-PSS tool for broader application. Further research should focus on enhancing model accuracy and data integration and including more predictor variables to expand the functionality of the tool for more complex and detailed urban scenarios such as green roofs or redesigning streets with vegetation.

ACKNOWLEDGMENTS

This thesis represents a journey in my academic career during which I have grown both personally and professionally. This journey would not have been possible without the support and guidance of many individuals to whom I am deeply grateful.

First and foremost, I would like to express my sincere gratitude to my supervisors, Dr. Mila Koeva and Dr. Pirouz Nourian, for their invaluable guidance, encouragement, and support throughout this research. Their expertise and insights were instrumental in shaping the direction and outcome of this thesis.

I also extend my heartfelt thanks to Dr. Christine Pohl for her help and support throughout my research. She provided data and helped organize a workshop where I had the opportunity to present my work.

A heartfelt thanks goes to my family. To my parents, Suraya and Hamid, for their endless support and encouragement throughout my academic journey. Your belief in me kept me motivated even in the most challenging times. To my sister, Parisa, for her understanding and support. Even though we are miles apart, with me in the Netherlands and her in Iran, her support has always been a source of strength for me. Your patience, encouragement, and constant belief in me have been vital throughout this journey. The distance never diminished your support, and for that, I am grateful. Your messages, calls, and words have always lifted my spirits and kept me focused on my goals.

A very special thank you to my wife, Majedeh. Your patience, love, and support have been my rock throughout this journey. Thank you for believing in me and standing by my side. I never forget your support.

I am also grateful to my best friends, Alireza and Saeid, for their constant support and for inspiring me to continue my studies abroad.

I extend my gratitude to my classmates and friends from UPM. Your kindness and friendship made this journey more enjoyable. Special thanks to Pritam and Wibi, my dear friends, for always being there for me. Your friendship and support were invaluable.

Lastly, I would like to acknowledge everyone who, in one way or another, contributed to the successful completion of this thesis. Your help and support are deeply appreciated.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Aggregation Index
ANNs	Artificial Neural Networks
DEM	Digital Elevation Model
DL	Deep Learning
DM	Digital Model DM
DS	Digital Shadow
DT	Digital Twin
DT-PSS	Digital Twin-Based Planning Support System
ED	Edge Density
GEE	Google Earth Engine
LULC	Land use Land Cover
LST	Land Surface Temperature
ML	Machine Learning
NDBI	Normalized Difference Built-up Index
NDVI	Normalized Difference Vegetation Index
NDWI	Normalized Difference Water Index
RF	Random Forest
RMSE	Root Mean Square Error
PD	Patch Density
PT	Partial Twin
PSS	Planning Support System
UE	Unreal Engine
UHI	Urban Heat Island
UI	User Interface
ST	Subject Twin
SVF	Sky View Factor
SVM	Support Vector Machine
DUT-W	Wuppertal has a Digital Urban Twin

1. INTRODUCTION

About 68% of the population in the world is expected to live in cities by 2050, resulting in increased urbanization in the following decades (United Nations, 2022). The effects of urbanization on the environment will be widespread, making sustainable development challenging (Ravanelli et al., 2018). Rapid urbanization in recent decades has resulted in several environmental challenges, such as the Urban Heat Island (UHI) (Mutani & Todeschi, 2020). In the two previous decades, UHI has become a more common phenomenon, impacting the quality of life in cities through increasing temperatures in urban areas (Helmholz et al., 2021). Based on the current trend of global urbanization, the intensification and prevalence of UHI are expected to rise (Deilami et al., 2018). This highlights the urgent need for effective mitigation measures in the upcoming years.

Extensive usage of impervious surfaces in the cities, such as asphalt and concrete, has a crucial role in UHI formation. The capacity of the mentioned surface materials to absorb and re-radiate the heat is higher than that of natural landscapes (Mohajerani et al., 2017). Additionally, UHI formation is intricately linked to land use, building configuration, and vegetation density variables. Thus, a comprehensive understanding of the relationship between these factors and UHI is essential to address and mitigate the adverse impacts of UHI.

It has been proven that UHI has consequences such as reducing air quality, posing a threat to the urban ecosystem (Mohajerani et al., 2017), adverse effects on human health (Debbage & Shepherd, 2015), and an increase in energy demand (Gago et al., 2013). For example, retaining heat and increasing temperature in urban areas leads to more energy consumption for cooling effect and water usage. Several studies measure the increase in energy to almost double for cooling buildings (Santamouris, 2013). Increasing energy consumption for cooling buildings also produces more greenhouse gas emissions and air pollution, especially ground ozone (Ahmed Memon et al., 2008; Kolokotroni et al., 2006). Moreover, an increase in temperature significantly affects human health and quality of life in urban areas. Increases in temperature have a negative impact on many people's lives, particularly vulnerable groups such as the elderly, and create environmental issues (Mohajerani et al., 2017). It has been shown that most heat-related deaths happen in urban areas (Stone et al., 2010). Therefore, UHI is the main focus of many studies in several domains, including urban planning and urban geography (Estoque et al., 2017).

To Mitigate UHI, researchers have employed different approaches to find the key factors influencing UHI. For instance, Huang & Wang (2019) explored the impact of 2D and 3D factors on LST and UHI using remote sensing images. They measure the effect of urban morphology and urban function zones on urban heat islands. Helmholz et al. (2021) quantified the impact of different infill urban planning scenarios on UHI by modelling. They suggested a simulation-based approach to compare different densities in buildings and greeneries for residential areas. They also showed that different building and vegetation densities significantly affect UHI and temperature (Helmholz et al., 2021). In addition, other scholars have introduced the use of nature-based solutions such as water bodies or green roofs to mitigate UHI. Mutani & Todeschi (2020) investigate the effect of green roofs in Italy to measure the impact of UHI, outdoor comfort zones, and energy saving.

1.1. Research Problem

Wuppertal City in Germany faces UHI effects due to its unique topography, high population density, and extensive urbanization in the city centre. During the night, there is a difference in the temperature in urban areas compared to rural areas in Wuppertal. Wuppertal is located in the Wupper Valley, and the valley structure restricts air movement. Dense built-up areas, especially along with Wupper Valley in the city centre, absorb heat during the day and release it at night. Limited airflow impedes the cooling air effect, accumulating heat in the lower part of the city. Lack of ventilation, surface materials, materials used in construction, degree of sealing, and high population density are known as the main causes of UHI formation in Wuppertal (Municipality of Wuppertal, 2020).

Future scenarios expect UHI intensity and heat stresses to increase in the city by 2060 (Municipality of Wuppertal, 2024). The future scenarios for UHI in Wuppertal expect a severe impact on UHI by 2060 based on the current global greenhouse gas emissions trend. The number of days above 30 degrees Celsius (hot days) has significantly increased from 100 years ago to almost double, ten days during the summertime. There is also expected to be a 100% increase in hot days in the next 50 years. This significantly impacts the liveability of urban areas, quality of life, biodiversity, and energy consumption for cooling in Wuppertal (Steinrücke et al., 2019).

The increasing number of hot days in Wuppertal might result in drought conditions and threaten urban vegetation cover. Impacting urban vegetation cover can reduce their cooling effect and even increase UHI formation in Wuppertal. This also impacts biodiversity in urban areas in Wuppertal (Scharte, 2020). Moreover, the increasing number of hot days has an adverse impact on vulnerability groups. The UHI in Wuppertal has health-related risks, especially for young and aged people. With the current population growth trend and demographic profile in Wuppertal, the number of aged and young people is expected to increase in future years (Steinrücke et al., 2019). This results in more heat-related health risks for the aged and young generation in Wuppertal City. Therefore, UHI is an urgent problem in Wuppertal City that needs to be addressed.

1.2. Main Objectives and Research Questions

This research aims to build a DT-PSS to support the decision-making process for UHI mitigation in Wuppertal.

Sub-Objectives (SO) and Research Questions (RQ):

SO1: To conduct a comprehensive literature review to find existing methods to build DT-PSS for UHI.

RQ1: Which variables are identified in the literature that (statistically) have the most effect on LST?
RQ2: What are the existing methods in the literature for building DT-PSS for UHI?
RQ3: What data is needed to build DT-PSS for UHI?

SQ2: To build a DT-PSS to assess the impact of different urban planning decision variables on the UHI.

RQ1: What is the most suitable method for building a DT-PSS for Wuppertal City for UHI?
RQ2: How can we integrate different data to create a DT-PSS tool?
RQ3: Which planning processes might affect the UHI phenomenon, and how can their effects be traced and assessed using the proposed DT-PSS?

SQ3: To evaluate the DT-PSS performance.

RQ1: In which parts of the planning process may the DT-PSS tool support decision-makers in UHI mitigation?
RQ2: What are the suggestions for further research to improve the performance of the DT-PSS tool?

2. LITERATURE REVIEW

2.1. Urban Heat Island (UHI)

Urban Heat Islands refer to the areas in the city with higher temperatures than rural areas. This phenomenon is the primary cause of heat-related death, global warming, and unpredictable weather patterns (Deilami et al., 2018). Due to the adverse effects of UHI on human health and cities, scholars investigated the main variables that might increase UHI or LST (as the proxy of UHI). Variables driving UHI and temperature can be categorized into two main categories. The first are variables not under our control, such as global warming and natural topography. The second are variables under our control, such as surface and building materials and green spaces (Ahmed Memon et al., 2008; Garzón et al., 2021; Mohajerani et al., 2017). By establishing this distinction, understanding the spatial-temporal pattern of under-control variables that might impact UHI is crucial in urban planning and policymaking.

In a comprehensive study of over a hundred cities, Santamouris (2015) explored how urban morphology, anthropogenic heat, and building materials affect UHI formations. Their analysis showed that denser urban fabrics and specific construction materials increase UHI impacts. Human activities, such as transportation and manufacturing operations, contribute significantly to UHI by emitting heat into metropolitan areas. Areas with higher population density have more concentrated heat sources from human activity, contributing to increased UHI effects. The form and arrangement of streets and buildings also may trap heat and prevent air from flowing to reduce it. This can result in UHI in metropolitan areas (Santamouris, 2015).

Another critical factor influencing LST is land cover type, as highlighted by Wu et al. (2014). Vegetated regions, such as forests and croplands, cool the environment, whereas built-up areas raise the temperature. They found that water bodies also help decrease UHI impacts due to their large covering and cooling qualities. Their study underlines that both landscape composition and the arrangement of land uses have a significant effect on the thermal environment (Wu et al., 2014). Changes in land use/land cover (LULC) are also identified as a factor that has a relationship with change in LST. Ravanelli et al. (2018) conducted a spatial-temporal analysis between 1992 and 2011 to see how LULC changes contributed to the UHI pattern. They used the Google Earth Engine (GEE) to analyse more than 6000 Landsat images. They found an increase in the pattern and intensity of UHI due to the land cover change in six metropolitan areas in the USA (Ravanelli et al., 2018).

Furthermore, researchers have used spectral indices and terrain factors to predict temperature. The common approach used is ML algorithms to model the relationship between predictors such as NDVI and NDBI (Wu & Li, 2019; Yang et al., 2017). Li et al. (2019) used terrain factors such as slope, the Digital Elevation Model (DEM), and spectral indices such as NDVI and NDBI to terrain the model. They included more predictors in their studies to increase the accuracy of the models for LST and UHI prediction. Their study also recommended including other variables, such as soil moisture and humidity, to improve the accuracy in future studies and highlighted the importance of terrain factors such as DEM and slope.

Moreover, 3D factors such as building height and Sky View Factor (SVF) are also used to predict LST. Hu et al. (2020) used the combination of 2/3D factors to find the relationship between variables and LST. They found that the relationship between variables and LST is not linear and changes during the seasons. They also found that 2D variables strongly correlate with LST during warmer seasons. In contrast, 3D variables strongly correlate with LST during the colder seasons (Hu et al., 2020).

In addition, landscape indices such as patch density, edge density, and aggregation index are found to have a relationship with LST. These factors contribute to landscape composition in urban areas. Estoque et al. (2017) classified land cover into different categories: water, impervious surface, green space, etc. Then, they calculated the spatial indices for each category and found the correlation between these indices and LST. They discovered that impervious areas have a strong positive relationship with LST, and green spaces have a strong negative relationship with LST (Estoque et al., 2017).

Population is another variable that may influence UHI and LST. H. Zhang et al. (2013) included the population in their analysis to see the change in both LULC and population between 1997 and 2008 and the impact on LST. They tried quantitively examining the relationship between population, LCLU, and LST during the period. They saw a significant change in population and urban growth as an urban sprawl. The main development happened in shrubs, cropland, and forests, reducing water bodies and vegetation areas. This change in LULC affected the spatiotemporal pattern of UHI and intensified it (H. Zhang et al., 2013).

Depending on the data availability and the context, scholars have selected different variables and factors to predict/quantify UHI and LST patterns. However, more and more predictors are included in the recent studies to increase the accuracy of the results. Especially for the studies that used ML algorithms to predict the LST and UHI patterns. Table 1 shows the selected variables from the literature for this study.

Table 1: Selected variables from the literature.

Factor	Definition	Reference
NDVI	normalized difference vegetation index range between +1 to -1	(Garzón et al., 2021)
NDBI	normalized difference built-up index ranges from 1+ to -1	(Garzón et al., 2021)
NDWI	normalized difference water index ranges from 1+ to -1	(H. Zhang et al., 2013)
Population Density	Refers to the number of populations per unit area	(H. Zhang et al., 2013)
Land-use	land use refers to specific human activities for which land is utilized, and land cover refers to different natural or artificial features that cover the land.	(Wu et al., 2014)
DEM	It is a 3D digital representation of the surface topography or terrain of the Earth.	(Li et al., 2019)
Patch Density	Patch density is a metric used in landscape ecology to calculate the number of patches in an area. This metric can help to comprehend the level of segmentation.	(Wu et al., 2014)
Edge Density	Edge density is a measure used in landscape ecology to quantify the number of edges or boundaries per unit area of a landscape. This measure is beneficial for analysing landscape patterns.	(Wu et al., 2014)
Landscape shape index	The landscape shape index (LSI) is a statistic used in landscape ecology to assess the shape complexity of patches or features in a landscape.	(Estoque et al., 2017)
Aggregation index	The aggregate index is a statistical measure that quantifies the degree of concentration or dispersion of a built-up area within a city.	(Estoque et al., 2017)

Source: Author, 2024.

2.2. Digital Twin and 3D City Modelling

Over the last two decades, 3D city modelling has been the main topic of much research in several domains of urban planning, including maintenance, disaster management, and infrastructure management (Biljecki et al., 2015). Many 3D city models are designed to help local and national governments, institutes, or companies in decision-making in various domains such as solar potential (Romero Rodríguez et al., 2017), flooding, wind safety (Blocken et al., 2012), and indoor and outdoor emergency response (Tashakkori et al., 2015). For example, Schrotter and Hürzeler (2020) demonstrated how 3D city modelling helps understand urban planning decisions that affect the built environment by visualising the outcome of possible planning scenarios and their consequences. To illustrate how planning scenarios impact the built environment, they demonstrated a number of 3D prototypes for air pollution, noise, radiation from mobile phones, and solar potential made in Zurich (Schrotter & Hürzeler, 2020).

Additionally, other researchers employed 3D city modelling to support decision-making by comparing the outcomes of possible solutions at the building and city levels. Chen et al. (2015) created a novel 3D tool to assess the impact of new development on existing buildings and in the neighbourhood. The dynamic 3D tool allows users to replace the existing building with a new one. Their study demonstrated how 3D visualisation is efficient in showing specific variables, such as sky exposure or landmark change, due to new city developments and constructions (Chen et al., 2015.)

The development of static 3D models resulted in a new concept of real-time simulation, the "Digital Twin." Digital Twin (DT) is being used in the industry for the first time. Despite the concept being over 20 years old and having continuous evolution and application in different domains, it has a diverse definition (VanDerHorn & Mahadevan, 2021). The Digital Twin GeoHub group under the Digital Society Institute of the University of Twente states that "DT is not merely a geometric (2D and 3D) representation of static assets, but a dynamic/live model that represents their past, current, and future states" (Digital Society Institute, 2022). Three components constitute the general framework for DT: the virtual world, the physical world, and the relationship between these two (Botín-Sanabria et al., 2022). Each element has a few components depending on the situation and the user's needs. Figure 1: Digital Twin elements. Source: Caprari et al., 2022Figure 1 shows the elements and the concept.

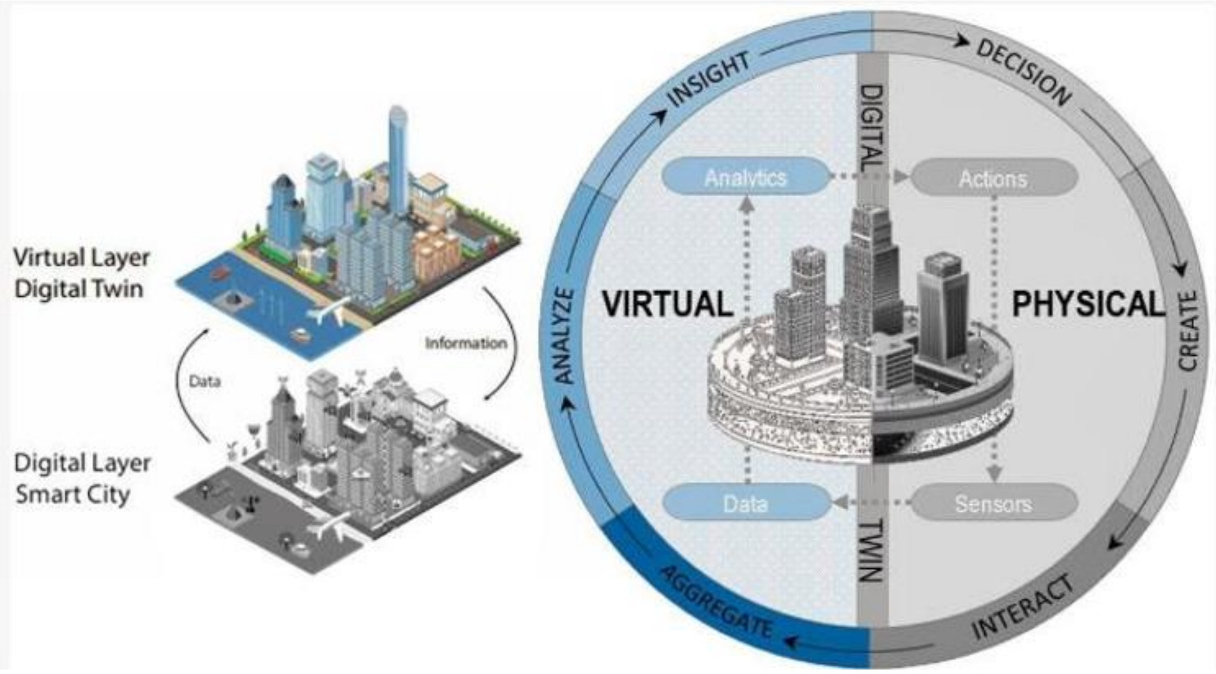


Figure 1: Digital Twin elements. Source: Caprari et al., 2022.

Fuller et al. (2020) reviewed more than 88 articles in DT in different domains. They found that many scholars used the DT concept in the article while it was a Digital Shadow or Digital Model. The literature has a big misconception about DT (Fuller et al., 2020; Kritzinger et al., 2018). There are a few similar words with different meanings: Digital Model (DM), Digital Shadow (DS), and Digital Twin (DT).

DM represents a physical asset, but there is no real-time data flow. Digital data of the asset might be used for creating the model, but the data flow is not automated integration, and all the digital information is added to the model manually (Fuller et al., 2020; Kritzinger et al., 2018). In DS, the data integration is automated, but the direction of the data is only from the physical asset to the virtual asset. In DT, data integration is automated, and it is done in both directions: physical assets and virtual assets (Khajavi et al., 2019; Kritzinger et al., 2018). Figure 2 shows the concept of the idea.

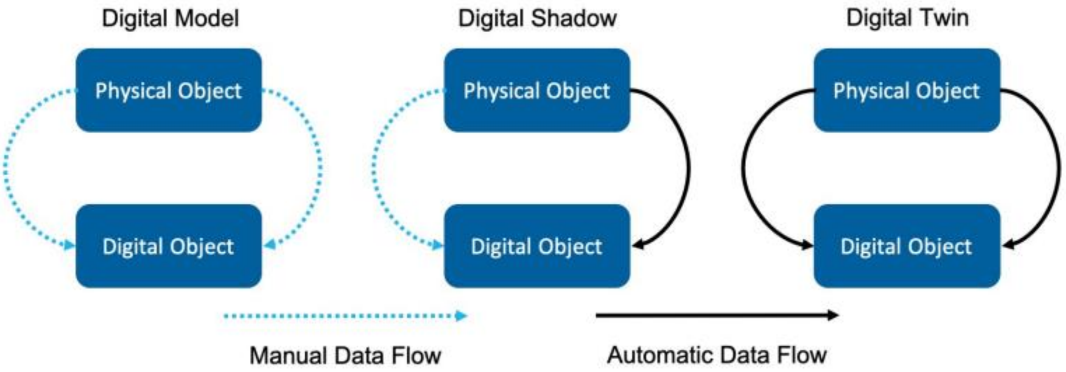


Figure 2: Concepts of DM, DS, and DT. Source: Fuller et al. (2020)

Building on the mentioned foundational concept, recent architectural and urban planning research projects have used DTs as an innovative idea. For example, DTs have been applied during many construction phases, including design and maintenance. Jiang et al. (2022) used DT to build new roadways and see the impact of different roadway scenarios in advance. DT technology has also expanded to include building life cycle management (Nica et al., 2023). Khajavi et al. (2019) used sensors to create a small DT of the building facade for life cycle management.

Furthermore, DTs are also used as an essential tool in decision and planning support systems. Scholars have shown the potential of DTs in helping the process of decision-making (Macchi et al., 2018). According to Nica et al. (2023), DTs might help urban planning and management by providing a virtual urban model that sees dynamic and complex relations between urban elements. In addition, DTs might improve decision-making by using real-time data, predictive analytics, and visualisation technologies. DT helps decision-makers in scenario-based planning, enabling them to examine the consequences of possible scenarios before implementing them. Caprari et al. (2022) studied the potential applications of DTs in urban planning in the context of smart cities. This study focuses on how the DT tool may help create a platform where decision-makers can assess urban scenario simulations and get public feedback before making decisions (Caprari et al., 2022).

In addition, research emphasis the benefits of DT as a planning support system in urban planning by suggesting a digital twin-based tool. K. Zhang et al. (2022) explored a tool named SpoVis for site locating. They created a DT tool to select optimal locations for sports facilities in the city. The research highlighted the advantages of DTs in city planning and development. K. Zhang et al. (2022) found that using DT, real-time data, and advanced modelling increases the accuracy and efficiency of the process of site locating. This technology facilitated decision-making processes and enhanced sports facility planning(K. Zhang et al., 2022). Similarly, White et al. (2021) demonstrated that using DT helps participatory planning. Their research focuses on how residents interact and provides feedback on urban planning scenarios that may change skylines and green spaces (White et al., 2021). According to them, releasing an open and public DT model allows planners to get feedback and promotes transparency to the public prior to action. As a result, DT supports decision-making and design by forecasting the influence of possible planning scenarios on built and non-built environments (Shahat et al., 2021).

Despite the advances, uses, and research mentioned in DTs, several technical challenges, such as data collection and model accuracy, remain (VanDerHorn & Mahadevan, 2021). There are also a few challenges with the implementation of DTs, such as sensor placement, sensor integration, IoT system limitations for connectivity, current infrastructure for building DT, and cost associated with setting up and maintenance of DTs (Attaran & Celik, 2023; Khajavi et al., 2019). Also, high-quality data and data management are the challenges related to the data in DTs (Kikuchi et al., 2022; Onile et al., 2021; Rasheed et al., 2020). Austin et al. (2020) mentioned that integrating various data sources while maintaining privacy and safety is another difficulty in creating DTs. Furthermore, DTs rely heavily on technologies, and this reliance makes them very vulnerable, especially to cyberattacks. (Nica et al., 2023)

Moreover, the current literature lacks practical applications in DT. The existing literature on DT is focused mainly on methodology and technical approaches rather than the implementation of DT. VanDerHorn & Mahadevan (2021) suggest demonstrating tangible benefits and practical applications of DTs to substantiate their value.

2.3. Machine Learning for Temperature Prediction

Researchers have used Machine Learning (ML) to interpolate temperature spatially. Scholars also have used ML to downscale or interpolate LST using satellite imagery data. They usually do this to increase the coarse resolution of satellite images and prepare them for further research. There are a few methods to do this process. The linear and multi-linear regression models are commonly used to find the relationship between LST and high-resolution remote sensing indices such as NDVI (Kustas et al., 2003). The most common statistical regression model is linear regression (Agam et al., 2007, 2008; Kustas et al., 2003). However, linear regression is incapable of representing nonlinear relationships. That is why other complex algorithms and models, such as artificial neural networks (ANNs), random forests, and support vector machines (SVM), have been used to capture the relationships between LST and predictors. Since the relationship between predictors and LST is complex, researchers used algorithms capable of capturing this relationship. Hu et al. (2020a) stated that the relationship between variables and LST is not linear and changes during the seasons. That is why they used a Boosted Regression Tree in their research. A boosted regression model might find linear and non-linear relationships between dependent and independent variables. Li et al. (2019) also used ANNs, SVM, and Random Forest (RF) algorithms. The results of this research showed that RF and ANNs provided more accuracy than SVMs. Random Forest is one of the regression algorithms that can efficiently find this complex relationship. Among ML models used in the studies, RF has shown high accuracy in predicting the LST (Yang et al., 2017).

2.4. Machine Learning in DT

ML models allow the prediction to be made possible inside the DTs. DTs get their prediction power either from ML models or a simulation model. A number of scholars recommend using the ML model in DT. New technologies such as ML and Artificial Intelligence (AI) have made DTs more efficient than before (Rasheed et al., 2020). Also, Austin et al. (2020) showed that AI and ML should be included to enhance the functionality of DTs.

Furthermore, researchers also used real-time data and the ML model to exploit the potential of DTs as temporary self-evolving models. For instance, Edington et al. (2023) described a time-evolving DT tool for engineering dynamics applications and demonstrated how DTs may be more flexible and responsive than static 3D models. They created a DT by integrating physics-based and data-driven models. Based on the collected data, the concept allows the DT to upgrade over time (Edington et al., 2023). Strauss & Bulatov (2022) also explored a unique way of creating a digital thermal twin using an ML methodology. This work investigated how DT and ML technologies may be used for the complex dynamics of heat in urban environments, which is an essential factor in UHI assessment.

As mentioned earlier in the DT literature part, DTs have three main elements: the virtual and physical models and the connection between these two. However, scholars add other elements, such as ML or AI, to the DT based on their research objective. Research shows that adding an ML model to DTs allows DTs for analysing the data and predicting the future state of the system (Austin et al., 2020; Edington et al., 2023; Khajavi et al., 2019; Rasheed et al., 2020).

2.5. Literature Summary

UHI refers to the area with higher temperatures than rural areas. This phenomenon has significant challenges, such as increasing heat-related health risks and decreasing quality of life in urban areas. Effective UHI mitigation requires understanding the factors driving UHI. Factors driving UHI are generally divided into two groups: those not under human control, like natural topography and global warming, and those under human control, such as surface material and green spaces. Understanding under-control variables in urban planning and policymaking is essential to mitigate UHI formation. Previous research incorporated ML algorithms and variables such as spectral indices to model LST and predict UHI pattern and formation. Studies included 3D factors such as SVF, highlighting their importance alongside 2D factors such as land cover. Recent studies included more factors to increase the accuracy of ML models and in the prediction of UHI formation.

The evolution of 3D city models to dynamic DT models marks a development in the decision-making in urban planning. By integrating real-time data and predictive analysis, DT represents the past, current, and future state of the urban assets. The DT model consists of three main components: the virtual model, the physical world, and the interactions between them. Despite misunderstandings that exist for DTs like DM and DS, DTs have bidirectional data flow between the physical world and the virtual model. Recent studies demonstrated the potential of using DTs to enhance decision-making in urban planning. DTs have been used in different areas, such as design, maintenance, and site locating in urban planning. However, a number of challenges, such as sensor integration, data collection, and model accuracy, remain. The literature indicates that integrating ML into DTs provides DTs with analysis and prediction capability. This is an essential feature in decision-making for assessing the impact of decisions in advance.

ML techniques have proven to be effective in predicting temperature. Although linear regression models are commonly used, they often fail to capture the nonlinear relationship between LST and the factors driving it. As a result, more sophisticated algorithms like ANNs, RF, and SVM are needed to capture the complex and nonlinear relationships between factors.

2.6. Research Gap

The literature categorizes the main driving variables of UHI and LST into under-control and not under-control variables. Since urban planning/design has a significant role in determining the characteristics of the under-control variables, considering their effect on UHI and LST in the urban planning/design process is essential (Deilami et al., 2018; Gago et al., 2013). However, a few issues about including the effect of the under-control variables on UHI and LST in the decision-making process remain unanswered. First, where should planners consider the variables that affect UHI and LST during the planning process? Second, how can the impact of the variables be evaluated before bringing the decisions into action?

To answer the first question, planners do not work with urban fabric variables like building height, land cover, and SVF. They deal with decision variables, including density, building codes and regulations, land use, etc. These decision variables indirectly or directly influence the urban fabric variables. For Instance, if urban planners want to increase density in a given area, they might increase the number of floors for buildings in the neighbourhood (this is related to building requirements in detailed plans). This directly impacts the SVF, which is one of the variables driving UHI. As a result, it is essential to determine which decision variables influence the urban fabric, resulting in an increase or decrease in LST and UHI (Ahmed Memon et al., 2008; Deilami et al., 2018; Gago et al., 2013).

As previously stated, the relationship between urban fabrics and LST or UHI is complex. A non-linear regression task is needed to find the relationship between variables and LST and UHI (Hou et al., 2023; Hu et al., 2020; Huang & Wang, 2019; Jing et al., 2016). Previous research studies have been conducted to predict UHI and LST use of ML (Jing et al., 2016). However, the research found a static relationship, while the relationship between the variables (that drive UHI and LST) changes dynamically during seasons and nocturnal and diurnal cycles (Huang & Wang, 2019). This relationship brings another issue for finding the relationship between variables. That is why a real-time model is needed to continuously update and show this relationship. Hence, to address the second question, using new technologies such as DT might help in the real-time evaluation of the impact of variables on UHI and LST (Caprari et al., 2022). This is due to the ability of DT to illustrate the complex relation between variables.

Overall, a number of gaps are found in the literature on UHI and DT. First, several studies have used remote sensing data to analyse the relationship between spectral indices and LST, such as Estoque et al. (2017) and Qi et al. (2022). However, limited research has tried to train an ML model for predicting LST in DT. Moreover, scholars who attempted to create DT (for UHI) created a DS or DM model instead of DT (Strauss & Bulatov, 2022). Second, while several research studies have employed DT as a decision-making tool (Strauss & Bulatov, 2022; White et al., 2021), limited research has been done on real-time what-if scenario examination on UHI. Finally, the literature indicates a gap in the practical application of DTs, highlighting the need for more specific examples of implementations(VanDerHorn & Mahadevan, 2021).

This research intended to fill the mentioned gaps by creating a DT-PSS that supports decision-making for UHI mitigation. This research also explores the possibility of using remote sensing data (as historical data) for DT. The DT-PSS tool in this study is suggested as an innovative idea for predicting UHI and temperature after simulating the impact of different urban planning scenarios in the DT.

3. STUDY AREA

The case study for this research is Wuppertal City in Germany. Wuppertal has a population of approximately 350,000 people and is the largest economy in Bergisches Land (Figure 3). The city is in the western German state of Noordrijn-Westfalen and is located between Cologne (southwest), Essen (north), and Düsseldorf (west). The city has grown in a valley along the Wupper River (Wuppertal.de, n.d.).

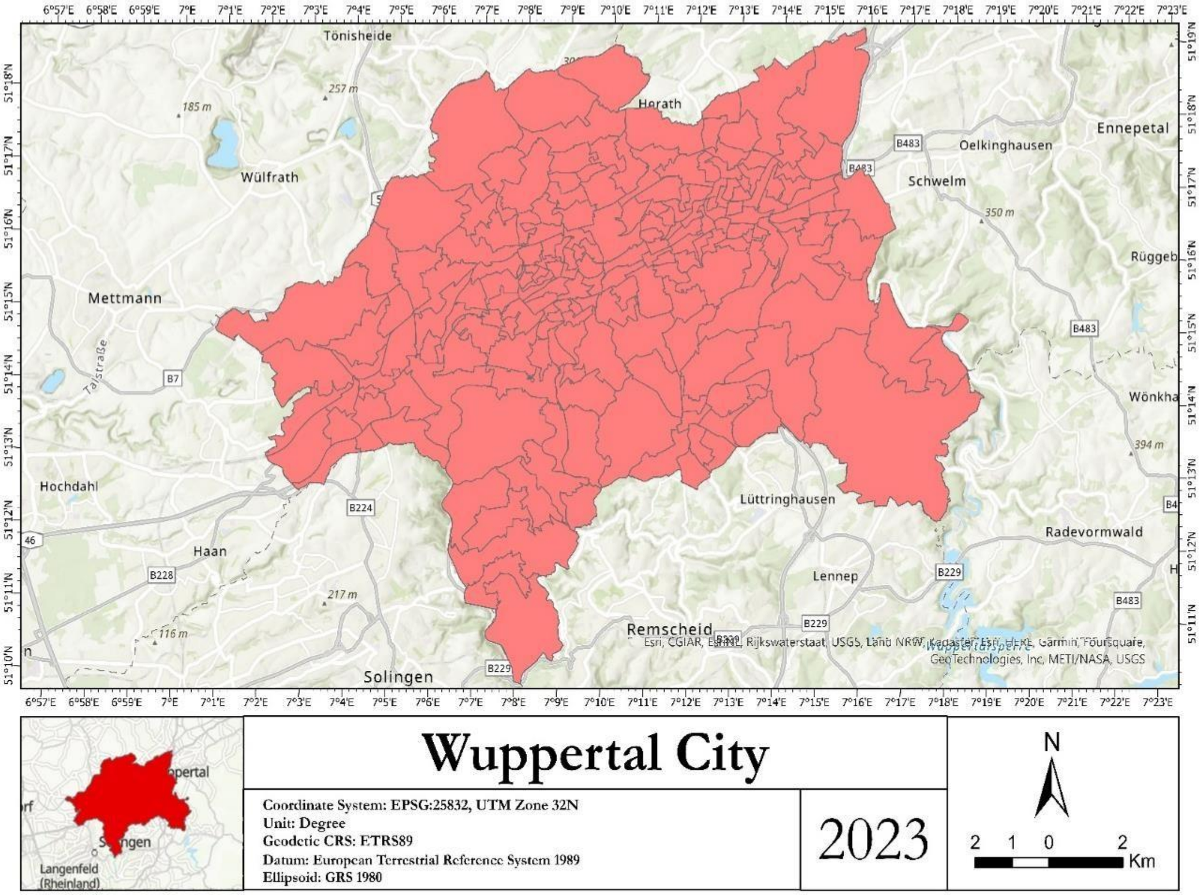


Figure 3: Wuppertal City. Source: Author, 2024.

In the framework of smart cities in Germany, the municipality of Wuppertal is interested in developing a DT tool. In the context of smart cities, Wuppertal municipality wants to use information and communication technologies to link the city infrastructure, such as energy, buildings, and water, to sustainable development. Wuppertal has a Digital Urban Twin (DUT-W), a project funded by the Ministry of Housing, Urban Development, and Building in Germany. Wuppertal is one of 73 German cities that intend to use digitization to support sustainable urban development. Creating a DT tool that maintains and processes real-time data and visualises the environmental effects of various decision variables in urban planning might help municipality authorities in sustainable urban planning (Municipality of Wuppertal, 2024.).

The municipality of Wuppertal has identified five key areas in DUT-W that need to evolve as Subject Twin (ST), small DTs in the DUT-W project. The five ST areas are climate change, sustainable urban development, urban resilience, green infrastructure, and mobility(Figure 4). In the ST climate change, the municipality has two Partial Twins (PT), urban heat, and a heavy rain hazard map. The primary application of this study is PT urban heat in DUT-W. This research may be utilized to develop PT urban heat, in which the impact of urban planning decision variables can be simulated in a game engine environment, and decision-makers in the municipality of Wuppertal can examine the impact of decision variables on UHI formation.

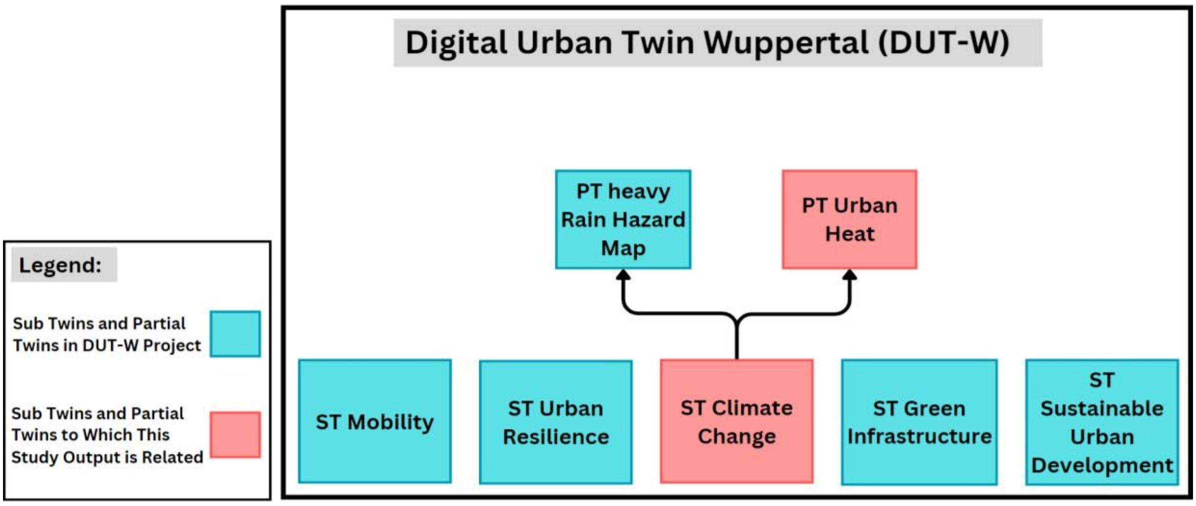

Digital Urban Twin Wuppertal (DUT-W)
PT heavy Rain Hazard Map
PT Urban Heat
ST Mobility
ST Urban Resilience
ST Climate Change
ST Green Infrastructure
ST Sustainable Urban Development
Legend:
Sub Twins and Partial Twins in DUT-W Project
Sub Twins and Partial Twins to Which This Study Output is Related

Figure 4: Digital Urban Twin Wuppertal Project. Source: Municipality of Wuppertal, 2024

3.1. Current Situation of UHI in Wuppertal

Using Google Earth Engine (GEE), primary research is done to determine the current UHI situation in Wuppertal. For this purpose, the mean LST for rural and urban areas is calculated for years between 2013 and 2024. Modis and Landsat images are used for this research. Both rural and Urban areas are defined manually in GEE based on the locations. Areas outside of Wuppertal city with high vegetation are selected as rural areas. In contrast, areas within the Wuppertal city with high building density are selected as urban areas. For both areas, mean LST between 2013 and 2024 is calculated. The result showed that between 2013 and 2024, the difference in LST in the two areas is between 4 and 8 degrees Celsius during the day, depending on the seasons. This difference is lower at night compared to day, but the result still showed a difference from 1 to 4 degrees Celsius in LST between the rural and urban areas in Wuppertal. The difference in LST between the two areas is higher in warmer than colder seasons. Figure 5 shows the selected rural and urban areas for this preliminary research. Rural areas are orange, and the urban regions are magenta. The related GEE code is provided in the Annex A.

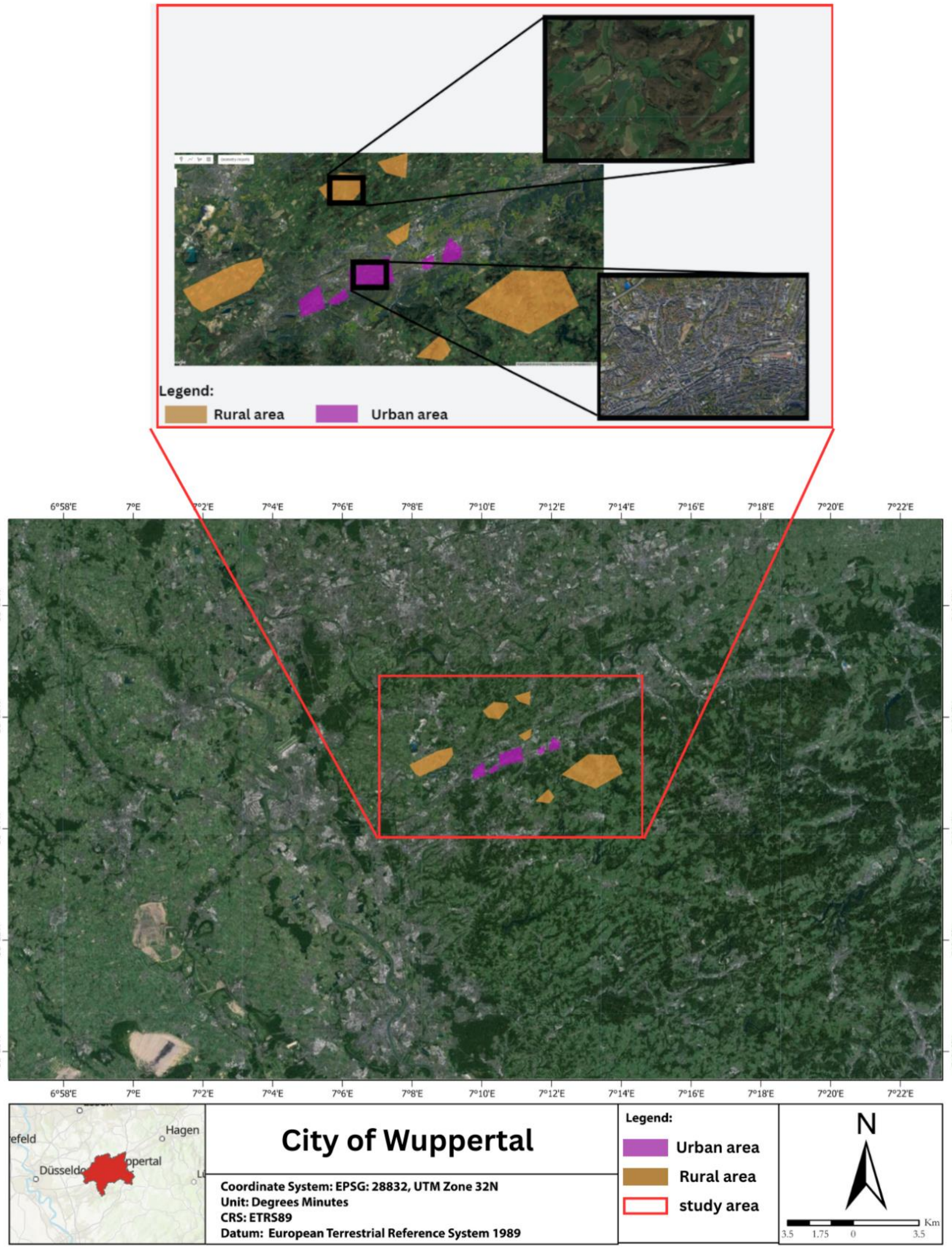


Figure 5: Selected rural and urban areas for preliminary research. Source: Author, 2024.

4. CONCEPTUAL FRAMEWORK

By combining real-time data and ML models with modelling capabilities, DTs can considerably help decision-making for UHI mitigation. The proposed DT-PSS tool in this study provides dynamic and detailed information on temperature so that decision-makers can do ex-ante assessments and make informed decisions to mitigate UHI.

The conceptual framework of this research is not only based on the literature but also on the requirements of the context to create a DT tool for UHI. The existing literature for DT lacks implemented examples of DT for temperature and UHI in urban planning. Therefore, the author proposed the general framework based on the existing literature and his innovation to create the DT-PSS for UHI. The conceptual framework applied to develop the DT-PSS for UHI has an extra element: a trained ML model. The ML model in the DT is used to predict future temperatures. Thus, the author's innovation lies in creating the framework for making the DT tool and adding the ML model that provides the predictive capability in the DT for future prediction of temperature and UHI formation. Figure 6 shows the DT framework for this study.

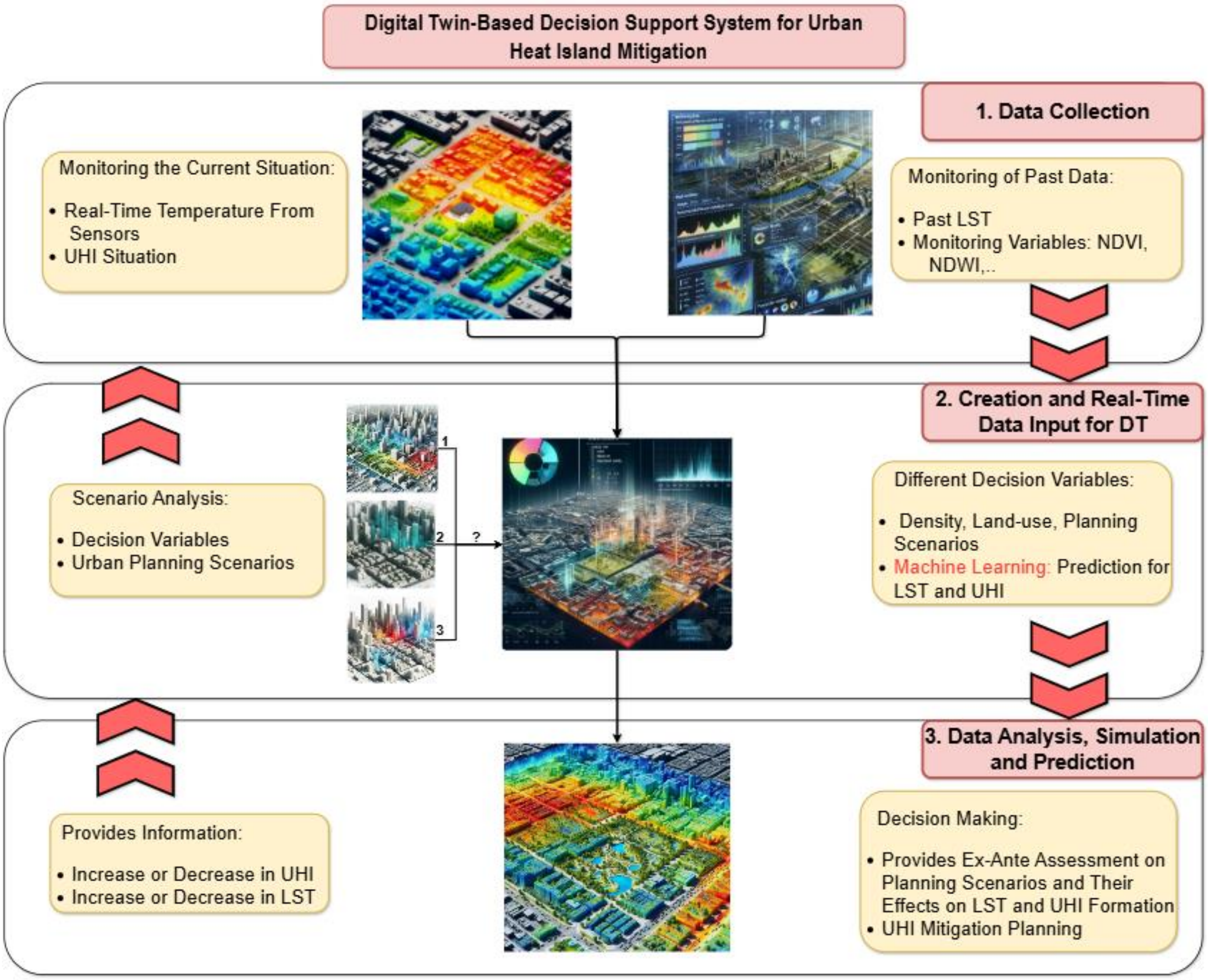

Digital Twin-Based Decision Support System for Urban Heat Island Mitigation
1. Data Collection
Monitoring the Current Situation:
• Real-Time Temperature From Sensors
• UHI Situation
Monitoring of Past Data:
• Past LST
• Monitoring Variables: NDVI, NDWI,..
2. Creation and Real-Time Data Input for DT
Scenario Analysis:
• Decision Variables
• Urban Planning Scenarios
1
2
?
3
Different Decision Variables:
• Density, Land-use, Planning Scenarios
• Machine Learning: Prediction for LST and UHI
3. Data Analysis, Simulation and Prediction
Provides Information:
• Increase or Decrease in UHI
• Increase or Decrease in LST
Decision Making:
• Provides Ex-Ante Assessment on Planning Scenarios and Their Effects on LST and UHI Formation
• UHI Mitigation Planning

Figure 6: Applied farmwork for DT for Wuppertal City. Source: Author, 2024

In the proposed framework, the real-time data is collected using the existing sensors for temperature in Wuppertal. This data is provided to the trained model in DT to make the real-time prediction. Four temperature sensors are used in this study. Figure 7 shows the location of the sensors.

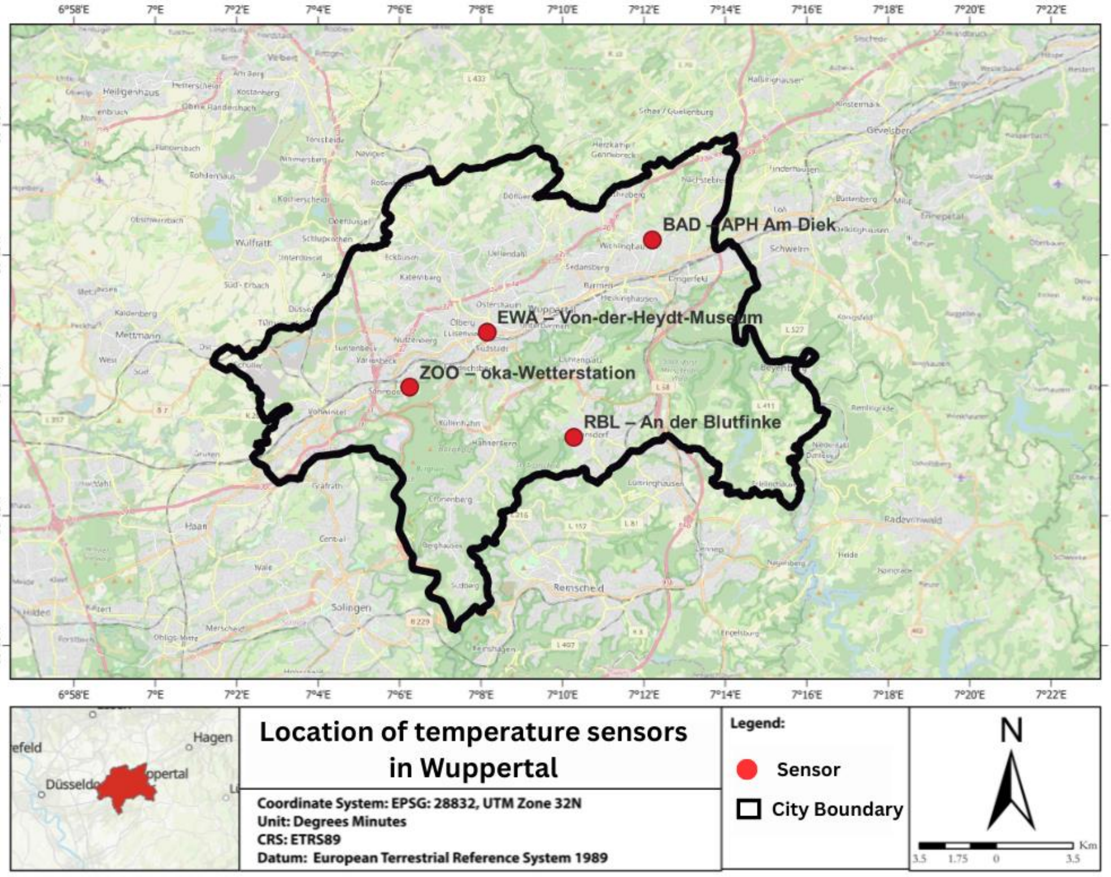

BAD – APH Am Diek
EWA – Von-der-Heydt-Museum
ZOO – oka-Wetterstation
RBL – An der Blutfinke
Location of temperature sensors in Wuppertal
Coordinate System: EPSG: 28832, UTM Zone 32N
Unit: Degrees Minutes
CRS: ETRS89
Datum: European Terrestrial Reference System 1989
Legend:
● Sensor
□ City Boundary
N
3.5 1.75 0 3.5 Km

Figure 7: The location of temperature sensors in Wuppertal City. Source: Author, 2024

5. METHODOLOGY

The figure below shows the proposed flowchart for this study (Figure 8). In the following, each phase is explained in detail.

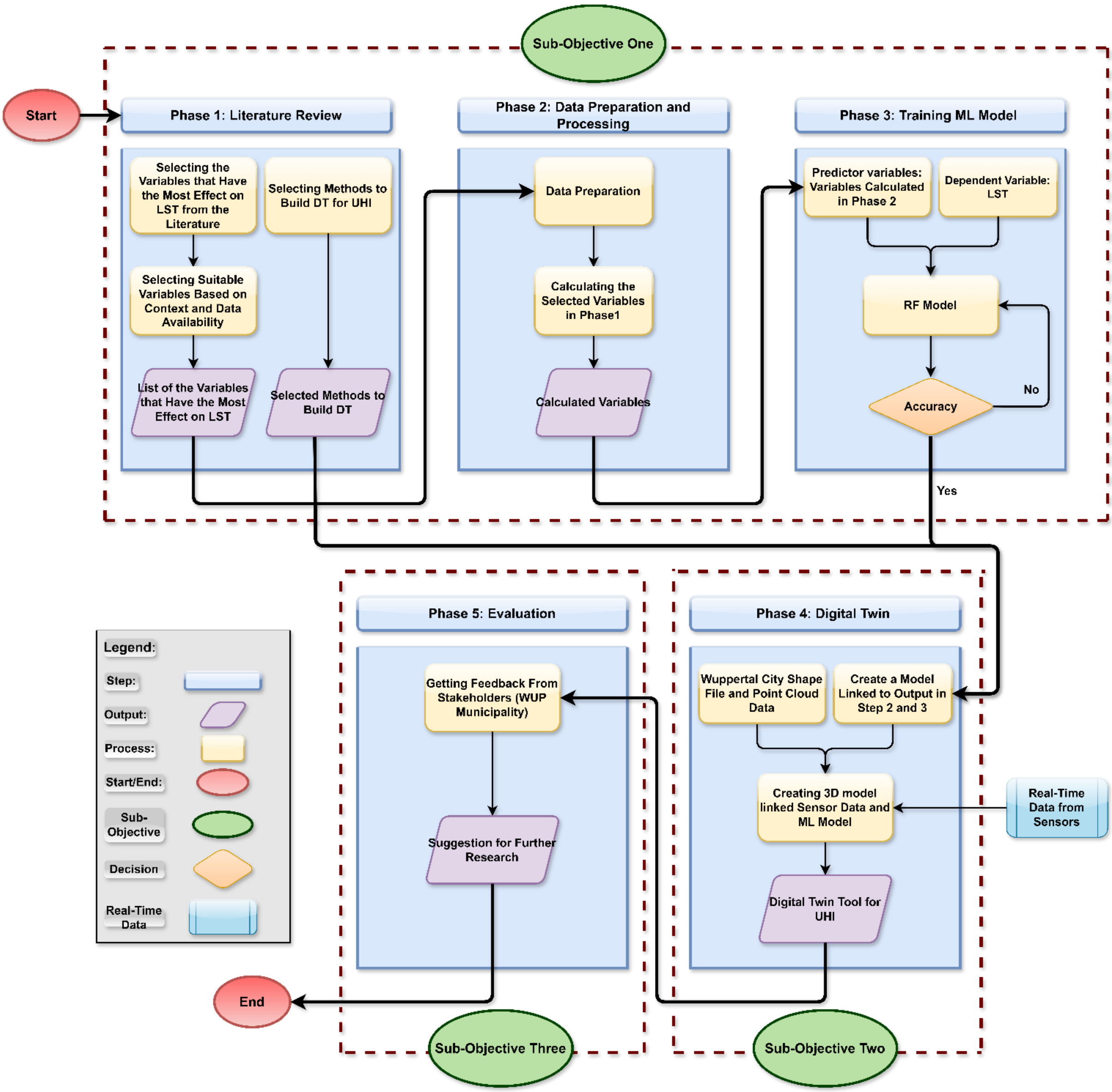

Sub-Objective One
Start
Phase 1: Literature Review
Phase 2: Data Preparation and Processing
Phase 3: Training ML Model
Selecting the Variables that Have the Most Effect on LST from the Literature
Selecting Methods to Build DT for UHI
Selecting Suitable Variables Based on Context and Data Availability
List of the Variables that Have the Most Effect on LST
Selected Methods to Build DT
Data Preparation
Calculating the Selected Variables in Phase1
Calculated Variables
Predictor variables: Variables Calculated in Phase 2
Dependent Variable: LST
RF Model
Accuracy
No
Yes
Phase 5: Evaluation
Phase 4: Digital Twin
Getting Feedback From Stakeholders (WUP Municipality)
Suggestion for Further Research
Wuppertal City Shape File and Point Cloud Data
Create a Model Linked to Output in Step 2 and 3
Creating 3D model linked Sensor Data and ML Model
Real-Time Data from Sensors
Digital Twin Tool for UHI
Legend:
Step:
Output:
Process:
Start/End:
Sub-Objective
Decision
Real-Time Data
End
Sub-Objective Three
Sub-Objective Two

Figure 8: Flowchart of the study. Source: Author, 2024

5.1. Phase 1 and Phase 2: Literature Review and Data Preparation

A comprehensive literature review is done in the first phase to select variables that statistically have a significant relationship with LST and UHI. Then, the author selected variables for further analysis based on data availability. Table 1 shows the variables identified from the literature for this study. Wind and SVF are excluded from this study while they are identified in the literature have significant relation with LST. The reason behind excluding SVF is a limitation in data availability. Moreover, including wind in the analysis required simulation of wind movement. This is out of the scope of this research due to the nature of wind movement and the difficulty of its simulation. A physics-based simulation is needed to capture the movement. It is worth mentioning that the past data for wind was also unavailable.

After selecting variables from the literature, the data required to calculate each variable is downloaded from available open sources. The other data necessary for the 3D city model, such as Wuppertal building shapefiles and Lidar data, are provided on the Wuppertal municipality website[1]. Table 2 shows the data used in this research. In addition, initial research is done to find the existing method and data needed to build the DT tool for UHI.

[1] https://www.offenedaten-wuppertal.de/search/tags/Geo-17

Table 2: Data used for this study.

Name of Data File	**Source**	**owner**	**Restrictions and License**	**Data Format**	**Contains Personal Information**
Sentinel 2	Secondary	European Space Agency. https://scihub.copernicus.eu/	open data	Raster (10m)	No
MOD11A1 (MODIS Terra)	Secondary	NASA. https://modis.gsfc.nasa.gov/	open data	Raster (1000m)	No
MYD11A1 (MODIS Aqua)	Secondary	NASA. https://modis.gsfc.nasa.gov/	open data	Raster (1000m)	No
Landsat 8	Secondary	NASA. https://www.usgs.gov/landsat-missions/landsat-collection-2	open data	Raster (30m)	No
Building Shape File: Gebaeude_EPSG25832_SHAPE.shp	Secondary	Wuppertal Municipality	open data	SHP file	No
Population File: Bevoelkerungsdichte_EPSG3857_S HAPE.shp	Secondary	Wuppertal Municipality	open data	SHP file	No
Wuppertal Thermal Sensors	Secondary	APIs for Wuppertal Municipality	open data	JSON	No
Contour line File: Hoehenlinien-10M_EPSG3857_SHAPE	Secondary	https://www.geoportal.nrw/?activetab=map&openDownloadclient=true	open data	SHP file	No
Lidar Data File: 3dm_32_371_5679_1_nw	Secondary	https://www.geoportal.nrw/?activetab=map&openDownloadclient=true	open data	.Laz	No
land use File: ALKIS Tatsächliche Nutzung	Secondary	https://www.offenedaten-wuppertal.de/search/tags/Geo-17	open data	.Laz	No

Source: Author, 2024.

In this part, 15219 sample points within the boundary of Wuppertal City are selected to collect and calculate the variables that are further used as the training data set for training the ML models. Stratified random sampling is used to select sample points within the Wuppertal boundary due to spatial Autocorrelation (refer to section 6.1). This sampling method divides samples into subgroups based on specific characteristics and then randomly selects samples to ensure they represent a variety of data. This sample selection method reduces bias and increases the accuracy and precision of the data (Howell et al., 2020). The variables that are calculated in this study are explained below.

LST: The study uses the Landsat data to calculate LST. Through several steps, LST is calculated using the USGS formula (Avdan & Jovanovska, 2016). The steps and the formula are explained below.

Step 1: Calculation of TOA spectral radiance

$$TOA\ (L) = ML * Qcal + AL \tag{1}$$

Where:

ML is a band-specific multiplicative rescaling factor.

$Qcal$ corresponds to band 10.

AL is a band-specific additive rescaling factor.

Step 2: TOA to Brightness Temperature conversion

$$BT = (K2\ /\ (ln\ (K1\ /\ L) + 1)) - 273.15 \tag{2}$$

Where:

$K1$ is band-specific thermal conversion constant.

$K2$ is band-specific thermal conversion constant.

Step 3: Calculate the NDVI

Step 4: Calculate the proportion of vegetation Pv

$$Pv = Square\ ((NDVI - NDVImin)\ /\ (NDVImax - NDVImin)) \tag{3}$$

Step 5: Calculate Emissivity ε
Step 6: Calculate LST

$$LST = (BT\ /\ (1 + (0.00115 * BT\ /\ 1.4388) * Ln(\varepsilon))) \tag{4}$$

NDVI: normalized difference vegetation index range between +1 to -1 calculated using red and near-infrared bands (NIR) of remote sensing imagery such as sentinel 2. The sentinel 2 data is downloaded from the Copernicus Open Access Hub[2] Below is the formula to calculate NDVI (equation 5).

$$NDVI = (NIR - RED)/(NIR + RED) \tag{5}$$

Since the Sentinel-2 NIR band is band 8 and the red band is band 4, the formula looks like this:

$$NDVI = (B8 - B4)/(B8 + B4) \tag{6}$$

[2] https://scihub.copernicus.eu/

NDBI: normalized difference built-up index ranges from 1+ to -1, calculated using NIR and shortwave infrared (SWIR) bands. Similar to NDVI, Sentinel-2 data is used to calculate the NDBI). Below is the formula for NDBI (equation 7).

$$NDBI = (SWIR - NIR)/(SWIR + NIR) \tag{7}$$

Since the Sentinel-2 NIR band is band four and the SWIR band is band 11, the formula looks like this:

$$NDBI = (B11 - B4)/(B11 + B4) \tag{8}$$

NDWI: normalized difference water index ranges from 1+ to -1, calculated using NIR and green bands. Sentinel-2 data is used to calculate NDWI (equation 9). Below is the formula to calculate it:

$$NDWI = (G - NIR)/(G + NIR) \tag{9}$$

Since the Sentinel-2 NIR band is band 8 and the green band is band 3, the formula looks like this:

$$NDWI = (B3 - B8)/(B3 + B8) \tag{10}$$

Edge Density (ED): Edge density is a measure used in landscape ecology to quantify the number of edges or boundaries per unit area of a landscape. This measure is beneficial for analysing landscape patterns (*Fragstats*, 2024). Equation 11 is the formula to calculate ED.

$$ED = A/E * C \tag{11}$$

Where:

E is the total length of the edges

A is the total area of the landscape,

C is a constant to convert the units to a standard form (e.g., meters of edge per hectare).

Patch Density (PD): Patch density is a metric used in landscape ecology to calculate the number of patches in an area. This metric can help to comprehend the level of segmentation (*Fragstats*, n.d.). Equation 12 is the formula to calculate PD.

$$PD = (ni * 10000/A) * 100 \tag{12}$$

Where:

ni is the total number of patches in the landscape.

A is the total landscape area (m2)

Aggregation Index (AG): The aggregate index is a statistical measure that quantifies the degree of concentration or dispersion of a built-up area within a city (*Fragstats*, n.d.). Equation 13 is the formula to calculate AG.

$$AG = (\sum ni = 1(gi * pi)/ Gmax) * 100 \tag{13}$$

Where:

gi is the number of like adjacencies (i.e., bordering cells of the same habitat type) involving patch type **i** based on a user-defined adjacency rule (usually four-cell or eight-cell rule),

pi is the perimeter of patch type i.

$Gmax$ is the maximum possible value of $\sum(gi * pi)$,

n is the number of patches of the focal type.

Land Scape Shape Index (LI): The Land Scape Shape Index is a metric that measures the irregularity of the landscape patch in comparison to the standard shape and shows how the patches deviate from being compact (circle shape) (*Fragstats*, n.d.). Equation 14 is the formula to calculate LI.

$$LSI = P \ / \ 2\sqrt{\pi A} \tag{14}$$

Where:

E is the length of edges in the landscape (m),

A is total landscape area (m^2),

Population Density: Population density measures the number of people living in an area, and the unit usually is people per square kilometer (km^2). Equation 15 shows the calculation for population density.

$$Population\ Density = Total\ Population \ / \ Total\ Land\ Area \tag{15}$$

Where:

Total Population is the total number of people living in the area,

Total Land Area is the size of the area (km^2)

Land-use: Land-use refers to how people utilize the land. In this study, the land use map is downloaded from the municipality of Wuppertal website[3]. Then, the land use is reclassified into the primary land use classes, such as vegetation, water, residential, etc. Since few land use classes, such as shrubs and trees, could generally be classified as vegetation, reclassification reduces the number of land use classes.

DEM: DEM refers to the 3D representation of the surface of a terrain created from terrain elevation data. For calculating DEM in this study, contour lines representing lines of equal elevation are used to create a Triangular Irregular Network (TIN) (Bonin & Rousseaux, 2005). Then, the TIN is converted into a raster format. The result is a grid-based DEM.

Moreover, this study also calculated UHI intensity. UHI intensity refers to the difference in temperature between rural and urban areas (Deilami et al., 2018). UHI intensity is calculated and used in the DT model to predict the change in UHI formation for scenario simulation. The change in UHI intensity is simulated in percent. Thus, 100 % is equivalent to 8 degrees Celsius (maximum temperature difference between rural and urban areas), as calculated for Wuppertal City (refer to section 3.1).

[3] https://www.offenedaten-wuppertal.de/search/tags/Geo-17

5.2. Phase 3: Training ML model

In this phase, four ML algorithms are used to predict UHI and LST. This study used ML instead of Deep Learning (DL) because of two main reasons. The first reason is that the data type in this study is vector data, and DL is usually used for another type of data, such as raster data (images). The second reason is that the driving variables of UHI are identified from the literature, and DL models are usually used when the driving values are unknown. Several variables from the literature have been determined to have a statistically significant relationship with LST and UHI. Therefore, it is known which variables drive UHI and LST. However, the relationship between the variables is complex, and regression tasks are needed to find this relationship. Therefore, ML models are used to capture this relationship.

This phase is separated into two steps: 1-preparing the data for training the model and 2- training the model.

1. The data preparation involved the relevance predictor variables (Table 1) and LST. In this step, all the variables calculated for each point are exported and combined into a .SHP file to make further analysis easier. The data (calculated LST and predictors) is divided into two parts randomly using Python. The first part consists of 80% of the data for training the model, and the second part consists of 20% for checking the accuracy of models. This process is done randomly to reduce the bias for both the test and the train data set. The logic behind separating the data into two separate parts is that, after training the model, the model accuracy can be measured using the test data.

2. Four regression algorithms are used to evaluate their performance in predicting LST: Support Vector Machine (SVM) and Random Forest (RF), Artificial Neural Networks (ANNs), and Polynomial Regression. The algorithms are selected because of their proven performance in predicting LST and UHI in the literature. For training the model, the Scikit-Learn Library in Python is used. The four algorithms of SV, RF, ANNs, and Polynomial regression model) are used to train with the sample data to make the most accurate model. Usually, SVM and RF algorithms are used for classification tasks, but in this study, they are used for regression tasks. In the sample data, LST is the dependent variable. NDVI, NDWI, NDBI, population density, PD, ED, AI, and land use are independent variables (predictors). The Python code for this part is provided in 0. The figure below shows the ML model training process (Figure 9).

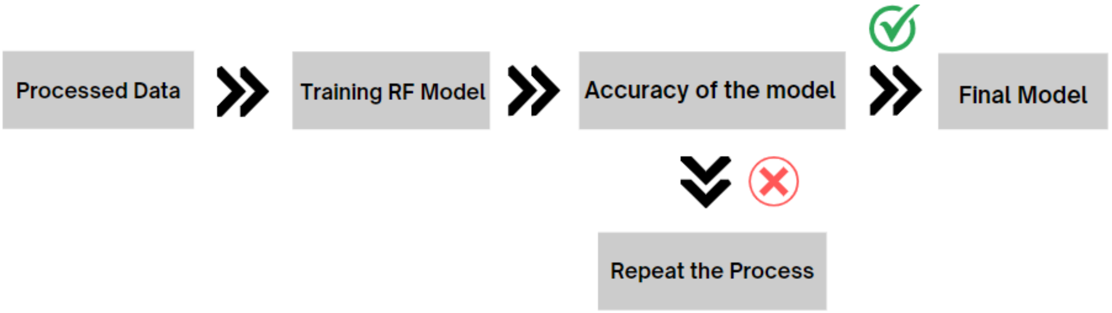


Figure 9: Process of training the ML model. Source: Author, 2024.

5.3. Phase 4: Creating the DT-PSS Tool

This part explains the process of creating the DT-PSS tool in detail. This part also includes methodology and rational software selection for creating the DT-PSS tool. It consists of several sections, each explained in the following. Figure 11 shows the process of creating the DT-PSS tool. It shows how the different pieces, such as the RF model, real-time data, and 3D models, are imported and included in the UE to create the DT-PSS.

As mentioned in the literature part, DTs have three main elements: the physical model, the virtual model, and the connection between them. Other elements like ML or DL are added to DT based on the objective of the DT tool in different research. From the literature, several examples of DT in different domains are explored. However, the literature lacks examples of implementation for building DT for UHI. Several requirements are identified for building DT for UHI from the literature and author innovation. In the following, identified requirements used in this study to create a suggested DT-PSS toll for UHI are explained.

- 3D city Model: 3D representation of urban areas with buildings, roads, vegetation, water bodies, terrain, and surface. Overall, a 3D city model contains all the elements within the urban area. The 3D city model serves as the virtual twin in the DT.

- Machine Learning (ML): Integrating ML models into DT allows the DT to predict the future state of the LST and UHI. The ML algorithms are trained based on historical and real-time data to predict temperature changes.

- Real-Time Data: Update the virtual twin based on the real-time data from the sensors.

- Platform to Implement DT: A platform is needed to integrate all the DT elements. The platform must have 3D model visualisation, real-time data integration, and analysis capability. Thus, choosing a suitable platform to create DT is one of the major prerequisites for building DT for UHI.

From the literature and findings of this research, the below data is also crucial to creating DT for UHI:

- Satellite Imagery: Satellite images from different sources, such as MODIS, Landsat, and Sentinel, are crucial for creating a DT tool for UHI. Satellite images provide valuable information about the historical and current state of urban and vegetation cover. Satellite images are essential in calculating spectral indices such as NDVI, NDWI, and NDBI.

- Sensor Data: Importing real-time temperature data into the DT tool is crucial since it updates the model and enhances the simulation accuracy.

- Geospatial Data: Land use or land cover map, building configurations, population density map, DEM, etc., helps to understand the spatial distribution of urban elements and their impact on UHI formation.

- Meteorological Data: Weather data such as wind is essential in creating DT tools for UHI since it can help increase simulation assessment accuracy. Additionally, including variables that are not under control, such as climate scenarios, will improve the accuracy of the models. Although these variables are not under planners' control, including them in the DT tool enhances the model performance.

This research used UE 4 as the main platform to create the DT tool. Several critical factors of UE 4 are the driving factors for the tool selection. This includes the UE 4 advanced visualisation capabilities, real-time data integration features, analysis, support for complex 3D city models, and the ability to create an interactive interface. These abilities are essential for creating an interactive DT tool for UHI. The UE has the possibility to modify and extend the engine functionality for different purposes. Therefore, developing C ++ or Python code-based functions in the UE projects is possible.

Unreal Engine primarily uses C++ programming language. However, it also supports Python scripting through various plugins. There are open-source and paid plugins for using Python in UE. The plugins allow developers to call UE functions from Python or perform automation tasks. This study used two plugins, "Sequencer Scripting" and "Python Editor Script Plugin," to extend the functionality of the UE engine. (refer to Annex C). Both mentioned plugins are open source.

The process of creating the DT tool is divided into several parts. The first part relates to incorporating the stakeholders' requirements for the DT tool. A survey is provided to include users' and stakeholders' opinions in making the DT tool. The link for the survey with the received responses is provided in Annex D

Since people in the Wuppertal municipality are the final users of the tool, the survey is distributed among people responsible for the DUT-W project, urban and environmental planning, and urban management. In the survey, a video is provided to show stakeholders an example of a planning support system tool in urban planning. This helped to bring stakeholders to a common ground and convey the purpose of this study as a DT-PSS. Then, the questions are provided to know their needs and expectations. The survey helped to engage the stakeholders in the first steps of designing the DT-PSS tool. Since the primary users of the DT tool are people in the municipality of Wuppertal, it is crucial to know their expectations and requirements for creating the DT tool. The survey results are analysed and used to make the DT tool and design interface and functions in UE for the users. Here, a summary of several questions from the survey is provided. Due to privacy, the survey results are anonymously provided.

- The respondents stated that creating a model for UHI elements such as buildings, land cover, green spaces, wind, and water bodies is essential. They expect these elements to be provided in the model.

- The respondents mentioned they expect the planning support system for UHI to have several functionalities, such as data visualisation, scenario model capability, and interactive interface.

- Based on the stakeholder's opinion, the DT model should have a middle to a high level of interactivity. This means the DT-PSS tool should enable users to remove or add elements such as buildings, water bodies, or greeneries to see the future change in temperature and UHI (refer to).

- Most stakeholders suggested that inner city areas with high population density and built-up areas be used as test areas to create a DT-PSS prototype. Between the suggestions, the Elberfeld and Barmen areas are mainly repeated. Based on the stakeholder's suggestions and requirements for temperature interpolation (discussed in the section0 6.5.1), the Südstadt neighbourhood in the Elberfeld area is selected to create the DT prototype. Here are suggestions for the area for making the DT tool from stakeholders:

  "Two city centres: Elberfeld and Barmen," "Fußgängerzonen Elberfeld und Barmen, Alles rund um die Kreuzungen Robert-Daum-Platz und Döppersberg," "Valley axis, City Elberfeld and Barmen," "All densely populated areas within the valley axis and in particular the inner cities of Elberfeld and Barmen," "Very densely built-up areas exposed to the sun in the inner cities of Elberfeld and Barmen.", "city centres in Elberfeld and Barmen, B7 and surrounding", "There are some poorer neighbourhoods like Barmen, Uellendahl, Langerfeld which need more support."

The second part is creating the 3D city model of Wuppertal. For developing the 3D model, 3D buildings are made using Lidar data. Building points are filtered from the other Lidar data points (e.g., vegetation, terrain, and noise points). Then, a level of detail 2 (LOD 2) model for buildings is created using the building footprints and filter building points. Figure 10 shows the process of creating a LOD 2 model for buildings.

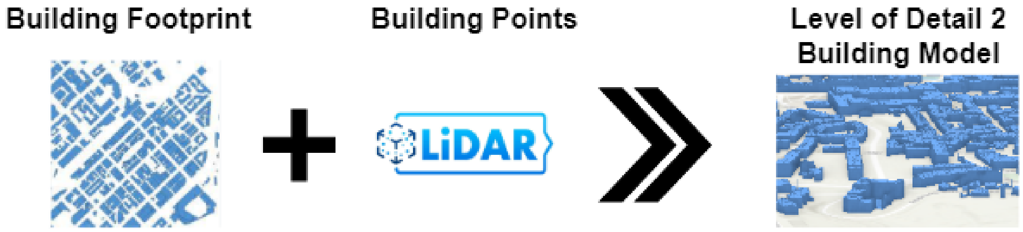


Figure 10: the process of creating the LOD 2 model for building. Source: Author, 2024.

Afterwards, the model is imported to the City Engine software to integrate terrain and street layout into the model. This is an essential step in enhancing the accuracy of the representation of urban elements in the model and creating the 3D city model of Wuppertal. Then, the model is exported in .FBX format to facilitate compatibility with the UE software. This format also ensures that all the objects in the UE software are modifiable.

The third part integrates sensor data and the trained ML model into the 3D model to create the DT-PSS. The real-time temperature from sensors is connected to the 3D city model in this part to include temperature sensor data to the DT-PSS. This is done using the VaRest plugin in UE. The Plugin enables communication and data transfer between the model and sensor data. Moreover, the temperature is interpolated using Inverse Distance Weighted (IDW) interpolation (refer to section 6.5.1). The IDW interpolation is chosen because it effectively estimates unknown values based on the location of known values (Ozelkan et al., 2015).

Finally, the trained ML model is imported to the Unreal Engine using the Python script function in Blueprint. This is crucial for creating the DT-PSS since integrating the ML model gives the ability of prediction to DT-pSS to predict LST and UHI formation. This also allows the DT-PSS to predict LST based on user interactions dynamically.

To make the platform interactive, several widgets are created to allow users to change the variables and see the changes in LST and UHI formation. This interactivity is essential for exploring different scenarios and understanding the impact of various factors in UHI. Figure 11 shows the process of creating the DT-PSS tool in this research.

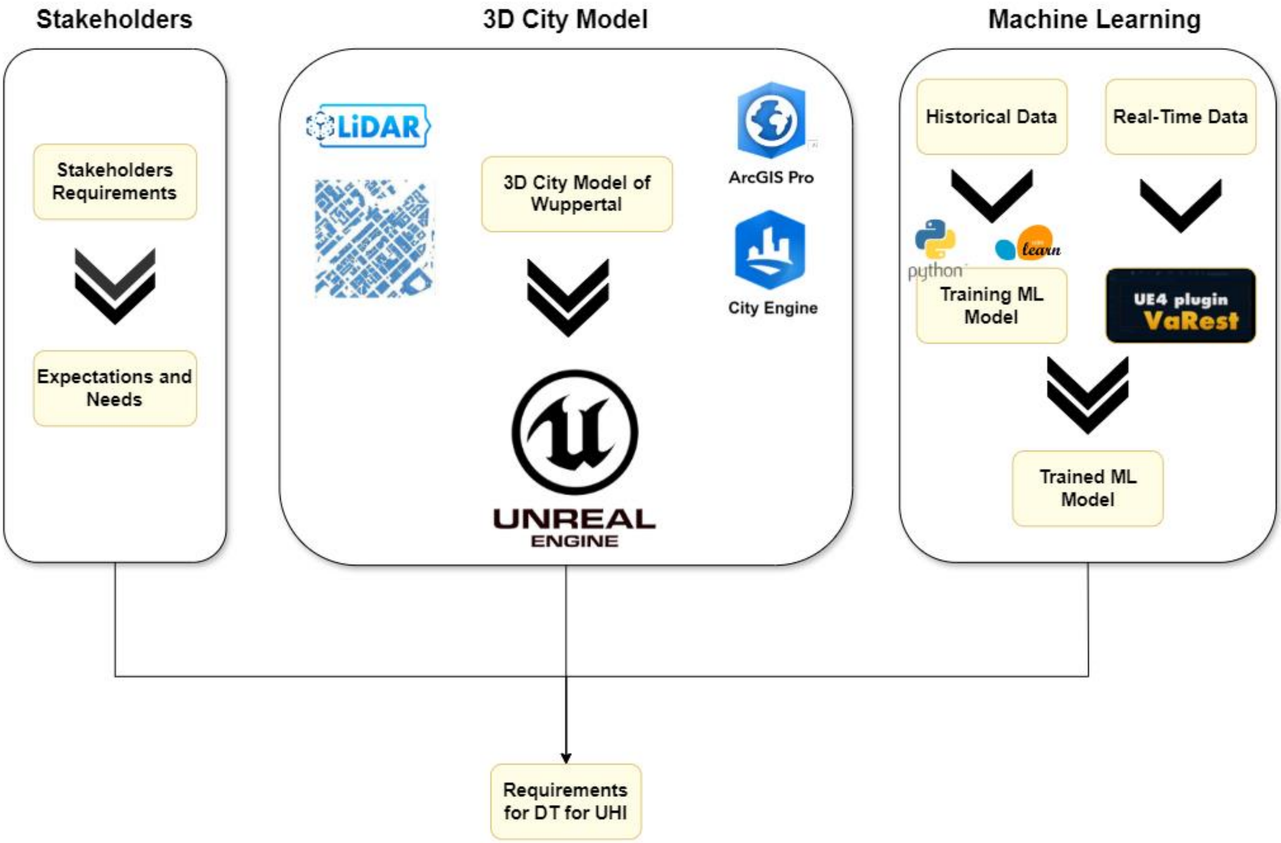


Figure 11: Process of creating the DT tool for this research. Source: Author, 2024.

5.4. Phase 5: Evaluation

This research topic is aligned with the ongoing projects in the municipality of Wuppertal for developing DUT-W to address current urban challenges in Wuppertal, such as UHI. During the initial meeting with representatives from the municipality of Wuppertal, several possible topics for the MSc thesis were discussed. The municipality representatives shared a document that briefly explained the DUT-W project in Wuppertal City and several of the current urban challenges in the city. Then, the meeting shaped the topic of this thesis. More information about the DUT-W project is explained in the case study section3.

After creating the DT-PSS prototype, the performance is shown to the users for feedback. The author did this by holding a workshop with Wuppertal Municipality officials. They saw the performance of the DT tool during the workshop and shared their experience. Then, the workshop output is used to provide suggestions for further research. The workshop output is provided in the evaluation section 6.6.

6. RESULTS AND DISCUSSION

This section presents and discusses the results of the study. This section consists of several sections. In the following, each section is explained.

6.1. Selecting Sample Points for Collecting Data

This section outlines the selection of sample points for data collection within the Wuppertal boundaries. These sample points served as training data for training the ML models. The dependent and independent (predictors) variables are calculated and stored in the attribute table of sample points for training the ML models. For example, LST (dependent variable) and variables selected for this study (Table 1) are calculated for each point. Then, this data in the attribute table of each point is added as sample data for training the ML model. Since the dependent variable is LST, the LST raster image calculated from the Landsat 8 image is converted to points. The sample points are the centre of each pixel in the LST raster images. Then, 185776 points are selected within the Wuppertal boundaries (Figure 12).

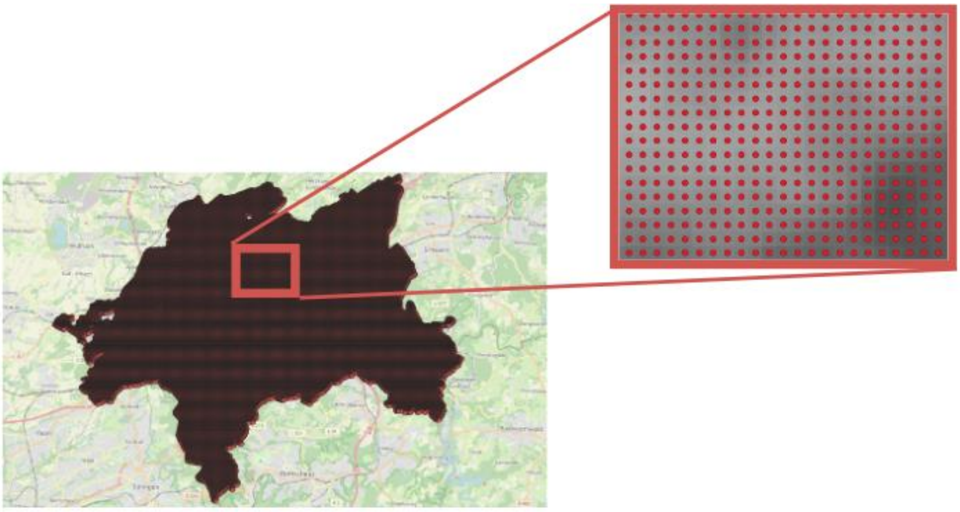


Figure 12: Sample points for collecting the data in Wuppertal. Source: Author, 2024.

Afterwards, Spatial Autocorrelation is checked before calculating the variables for the sample points to ensure that sample points are not clustered. Spatial Autocorrelation refers to the degree to which spatial objects and their values are clustered in the space. This is crucial in the analysis because if the sample points are clustered, this might impact the trained model performance and cause bias. When the data is clustered, neighbouring data points are more similar than distance data points. This means the neighbouring data points have similar values to those of the distance data points. Consequently, clustered data reduces the generalizability of the model since the model is trained based on the cluster data. This is because the model learns patterns specific to the clusters instead of learning general relationships. Clustered data also overestimate the model performance and underestimate the uncertainties. This happens when the train and test data contain similar points, making them seem more accurate than when trained by independent data.

Spatial Autocorrelation measures the data based on attribute value and object location using Global Morons' I. The result showed that there is a high Spatial Autocorrelation between the sample points, and the data points are clustered. Figure 13 shows Global Moron's I results for the reference data points. The results showed that the z-score is higher than 2.58 and the p-value is less than 0.01, meaning there is less than a 1% likelihood that the clustered pattern could result from random chance. This means the Spatial Autocorrelation for sample points is not from a random chance, and Spatial Autocorrelation is between the sample points.

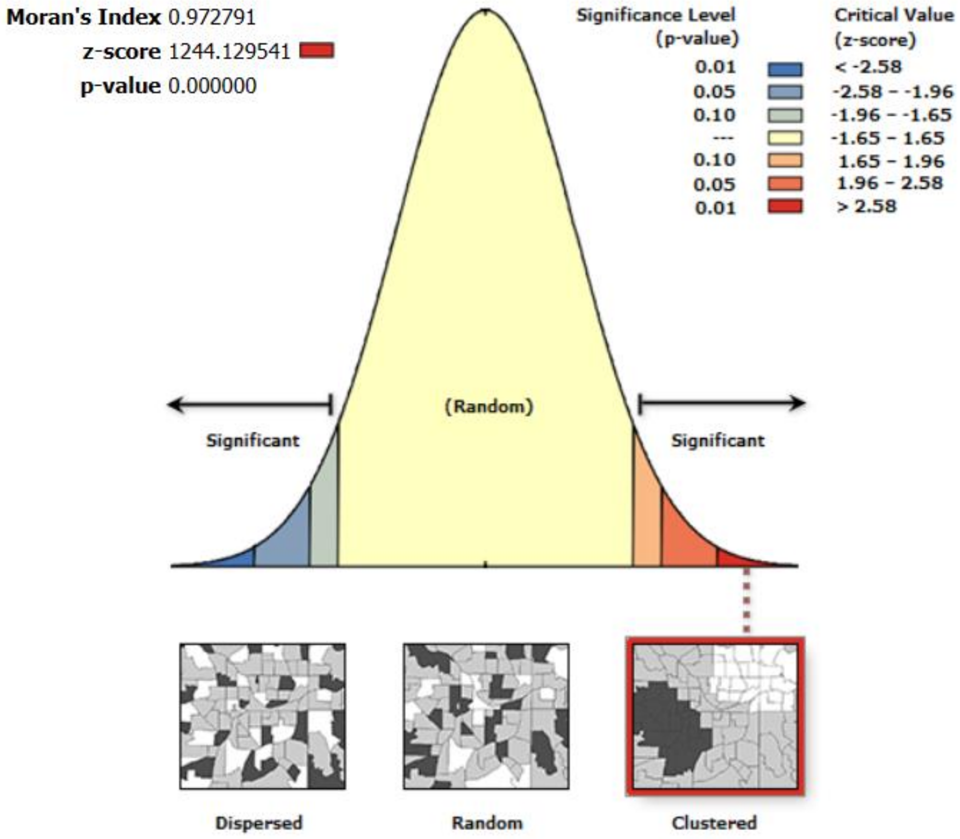


Figure 13: Global Morons' I result for reference data points. Source: Author, 2024.

Therefore, to avoid spatial autocorrelation, from 185776 points, 15219 points are randomly selected using stratified random sampling as final sample points for calculating the variables. Stratified random sampling selects points based on their cluster (refer to methodology section 5.1). This random selection helped to have an unbiased and more balanced data set for training ML models. Figure 14 shows the selected sample points for this research.

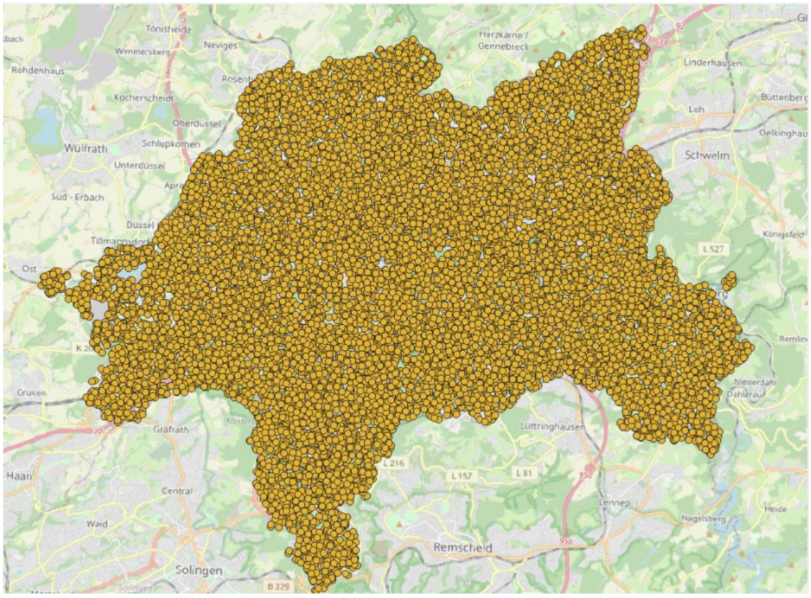


Figure 14: Selected points for this research. Source: Author, 2024.

6.2. Calculating and Collecting Data for Each Sample Point

This section elaborates on the process of calculation and preparation of predictor variables (Table 1) and the dependent variable (LST). These variables are essential for the training and validation of ML models in this study. For LST, Landsat 8 images with 30m spatial resolution are used to calculate LST. Also, for spectral indices such as NDVI, NDBI, and NDWI, Sentinel-2 images with 10m spatial resolution are used. Due to the different temporal resolutions of Landsat8 and Sentinel-2, images are downloaded for the dates that both satellites overpassed Wuppertal. This ensured the consistency of temporal data coverage. Moreover, this study selected and used images with a cloud cover of less than 20% to minimize the impact of clouds on the data quality.

6.2.1. Calculating Land Surface Temperature

For calculating LST, Landsat 8 images between 2017 and 202 are downloaded from the USGS website[4] and processed using the steps mentioned in the section 5.1, methodology. After calculating the LST, raster images are converted to points. The figure below shows the calculated LST for the 1st of June 2023 (Figure 15). The areas with higher temperatures are red, and those with lower temperatures are blue. The figure below provides a spatial understanding of temperature variation in Wuppertal City. The temperature in urban areas is higher than in the areas covered with vegetation and trees. Moreover, the temperature in peripheries with fewer buildings compared to urban areas is still higher than in areas fully covered with vegetation.

[4] https://www.usgs.gov/landsat-missions/landsat-collection-2

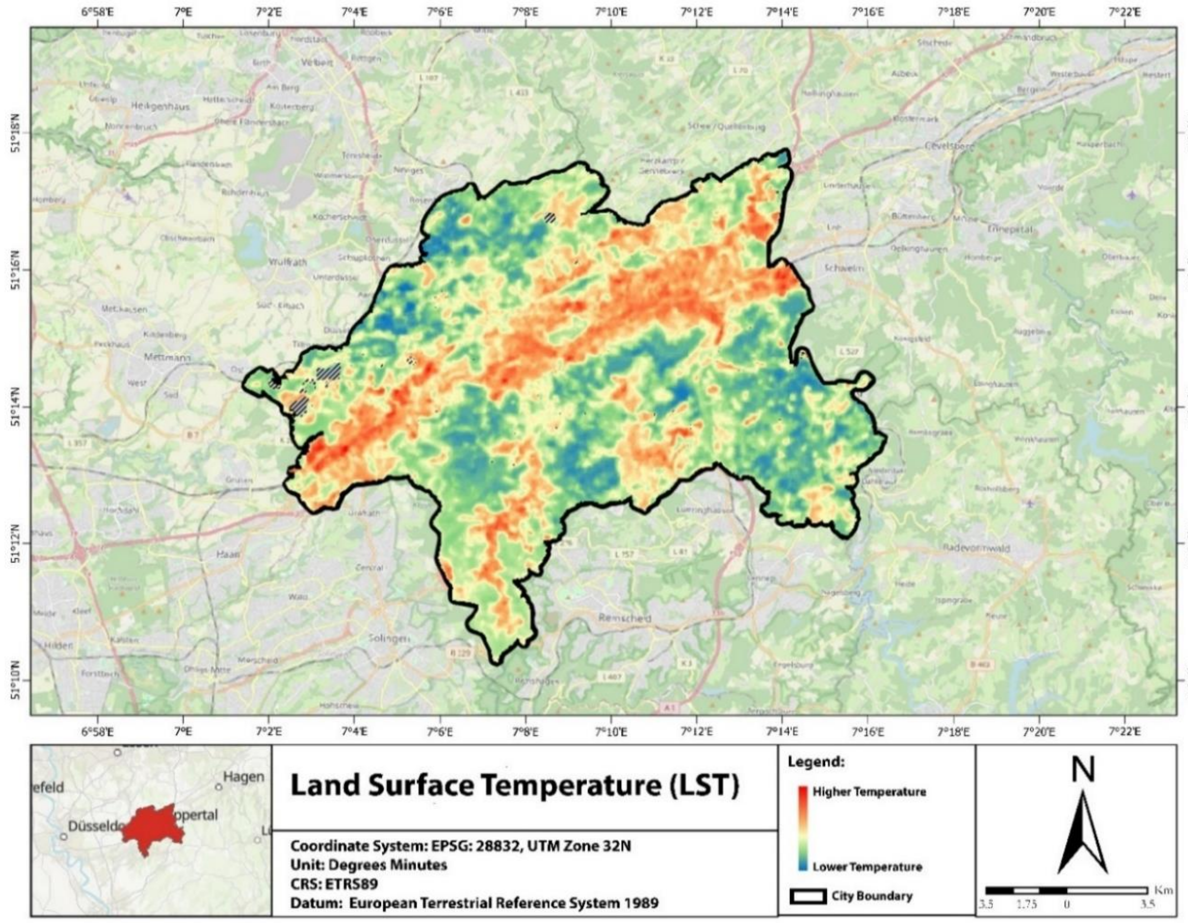


Figure 15: Calculated LST using Landsat 8 Image for 1st of June 2023. Source: Author, 2024.

This pattern is also found in the Wuppertal municipality report for temperature. Along the Wupper River, where the density of the built-up area is high, the temperature is high. Furthermore, the municipality of Wuppertal identified heat stress and strong heat stress locations in Wuppertal City. Figure 16 shows the heat stress and strong heat stress areas identified by the municipality. Red areas are strong heat stress areas, and the yellow areas are heat stress areas. Comparing the LST map and heat stress map (Figure 17) shows a similar pattern between the created LST map in this study and the heat stress map.

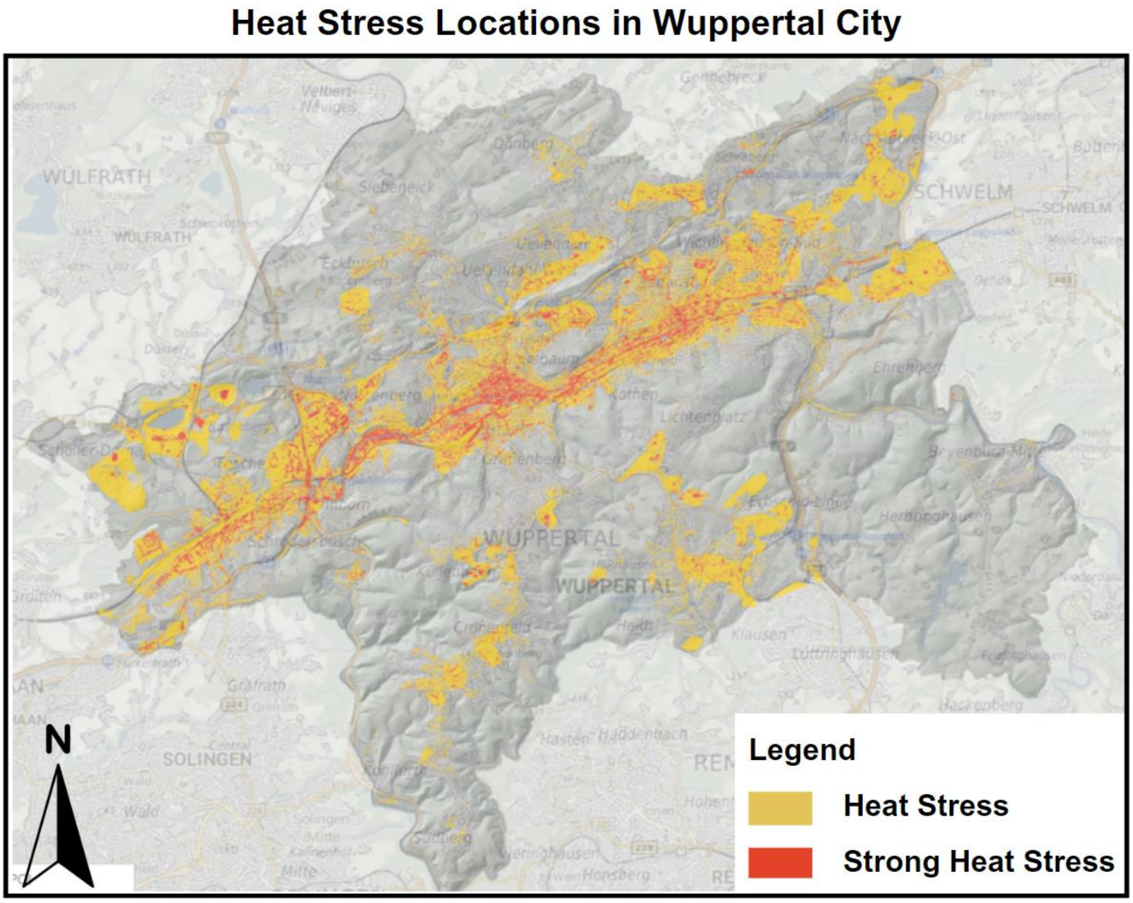


Figure 16: Heat stress and strong heat stress areas in Wuppertal City. Source: Municipality of Wuppertal, 2024.

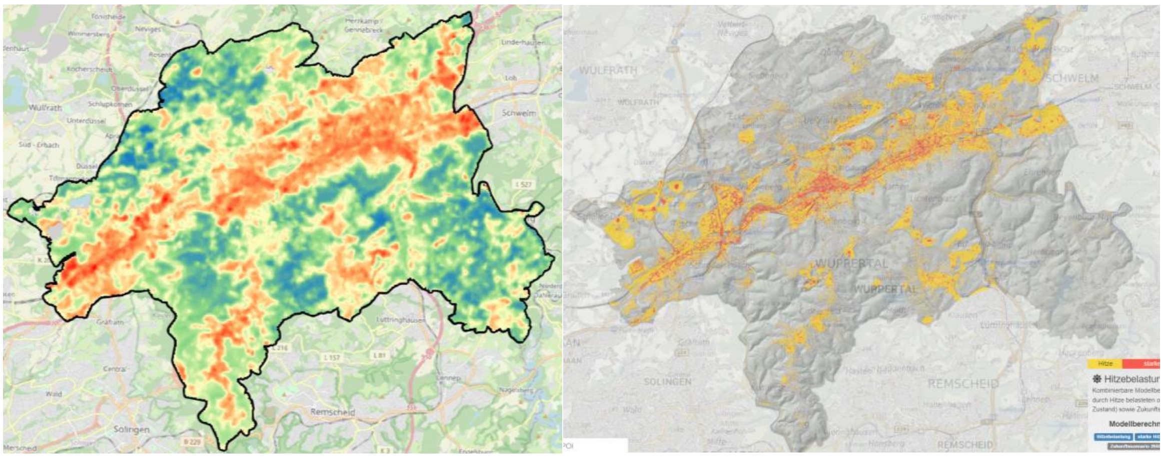


Figure 17: Comparing the LST map and the heat stress map. The right image is the LST map. The left image is the heat stress map. Source: Author, 2024.

6.2.2. Calculating Spectral Indices

Sentinel-2 images are downloaded from the Copernicus open-access hub[5]. to calculate NDVI, NDWI, and NDBI. All three indices are calculated using the formula mentioned in the methodology part. Figures 18, 20, and 22 show the results of the calculated NDVI, NDWI, and NDBI for the 1st of June 2023, respectively. The higher values in the map are in red, and the lower values are in blue. For instance, in Figure

[5] https://scihub.copernicus.eu/

18, red areas have higher NDVI values, meaning that these areas have higher vegetation. On the other hand, blue and yellow areas have lower NDVI values, meaning that these areas have lower vegetation.

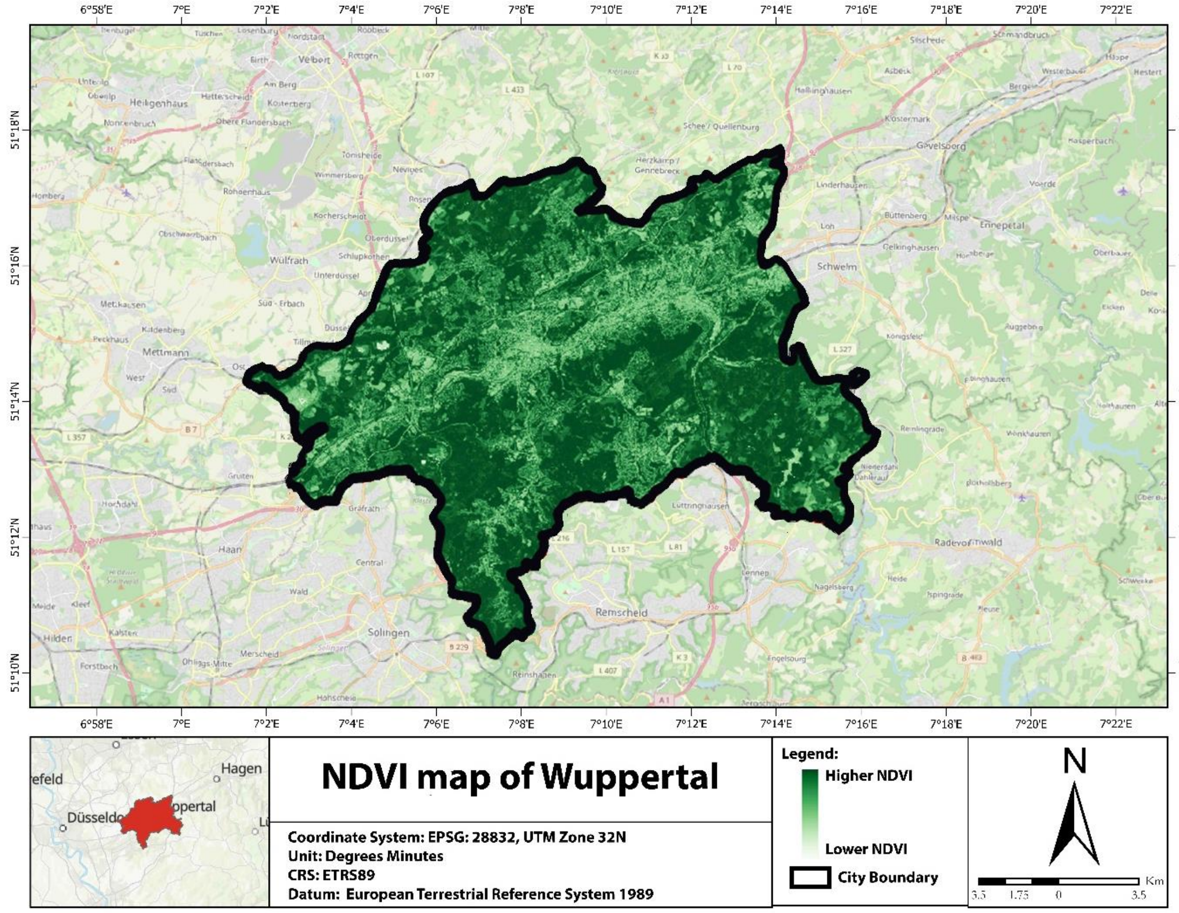


Figure 18: Calculated NDVI using Sentinel-2 Image for 1st of June 2023. Source: Author, 2024.

Comparing the NDVI map with the LST map (Figure 19) shows that the areas with lower NDVI have higher temperatures than those with higher NDVI. To be more specific, areas with more vegetation cover tend to have lower temperatures than those with less vegetation coverage. From the literature, NDVI and vegetation cover have also been identified to have a significant correlation with LST (Li et al., 2019; Ravanelli et al., 2018). The created NDVI map is used to calculate NDVI values for sample points to train ML models.

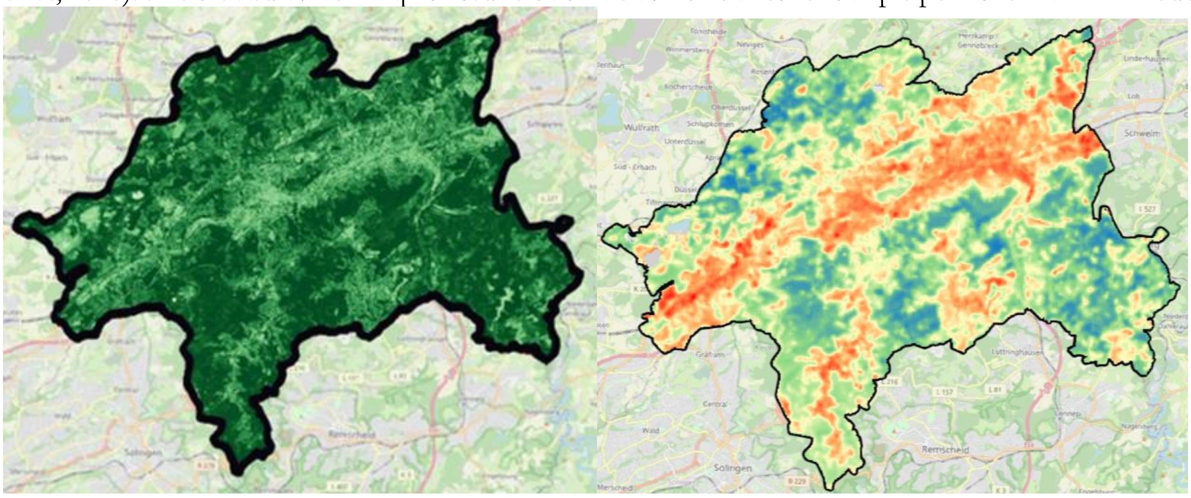


Figure 19: Comparing the NDVI map and the LST map. The right image is the NDVI map. The left image is the LST map. Source: Author, 2024.

Figure 20 shows the NDBI map. Areas with higher NDBI values are in red, and areas with lower NDBI values are in blue. The map shows that NDBI values are high in the Wuppertal centre where the building density is high. Comparing this map to the LST map (Figure 21) shows that where the NDBI value and building density are high, the temperature is higher than in areas with less building density. This correlation is identified in the previous research. Scholars stated that building density has a significant positive correlation with temperature (Garzón et al., 2021).

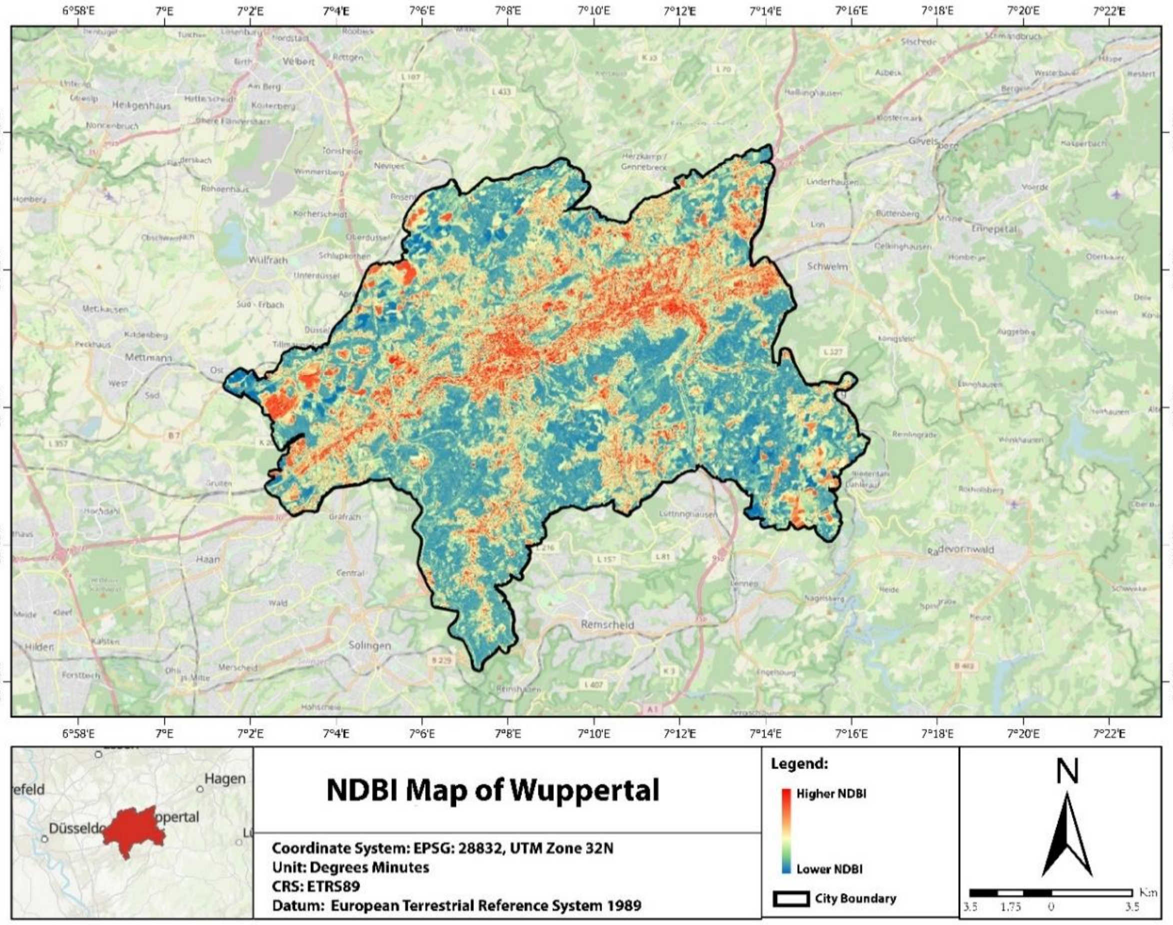


Figure 20: Calculated NDBI using Sentinel-2 Image for 1st of June 2024. Source: Author, 2024.

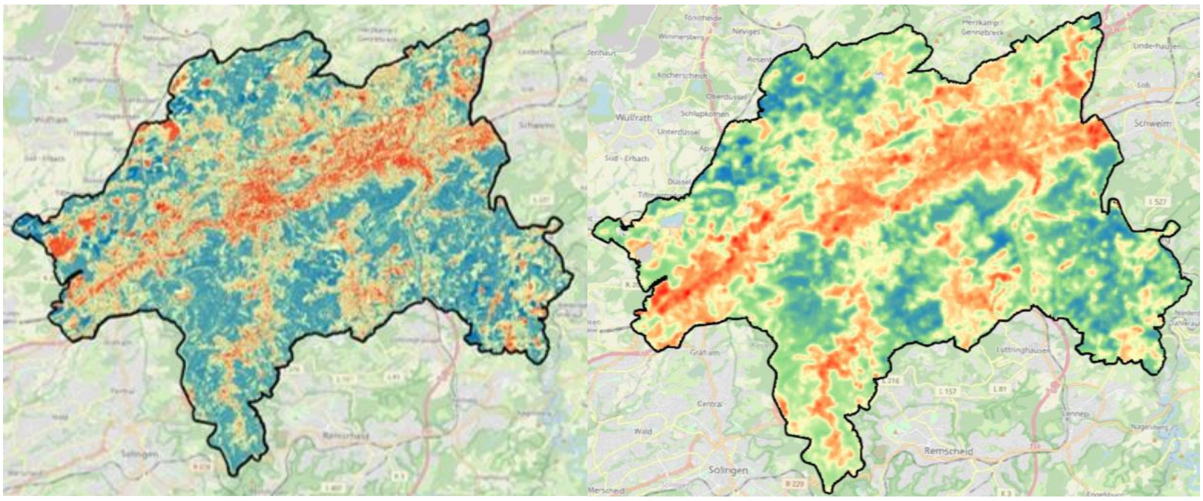


Figure 21: Comparing the NDBI map and the LST map. The right image is the NDBI map. The left image is the LST map. Source: Author, 2024.

Figure 22 shows the NDWI map for Wuppertal. The higher values are in red, and the lower values are in blue. Comparing this map to the LST map shows that where NDWI values are moderate to high (except with the very high value, which is coloured in red), the temperature is high. Moderate to high values represent the build-up areas in Wuppertal city. This is because NIR bands are used to calculate NDWI, and the reflectance of material used in construction or pavements is usually moderate in NIR bands (and results in over-estimating water bodies). To prevent errors in the training data, the values between -0.1 and 0.2 are clustered as built-up areas and removed from the analysis. The strong red areas on the map are water bodies in Wuppertal City. Figure 23 shows the comparison between the LST map and the NDWI map. It shows that the temperature is low where the water bodies are (strong red area).

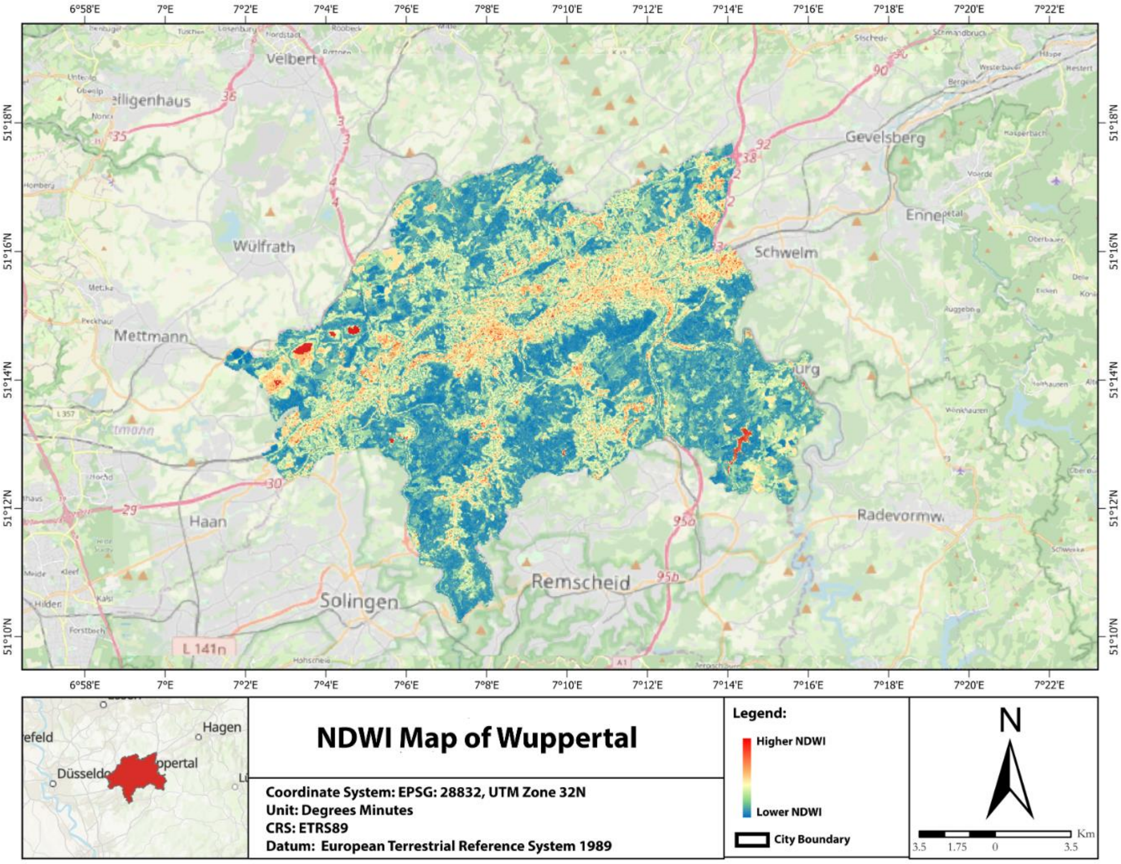


Figure 22: Calculated NDWI using Sentinel-2 Image for 1st of June 2023. Source: Author, 2024.

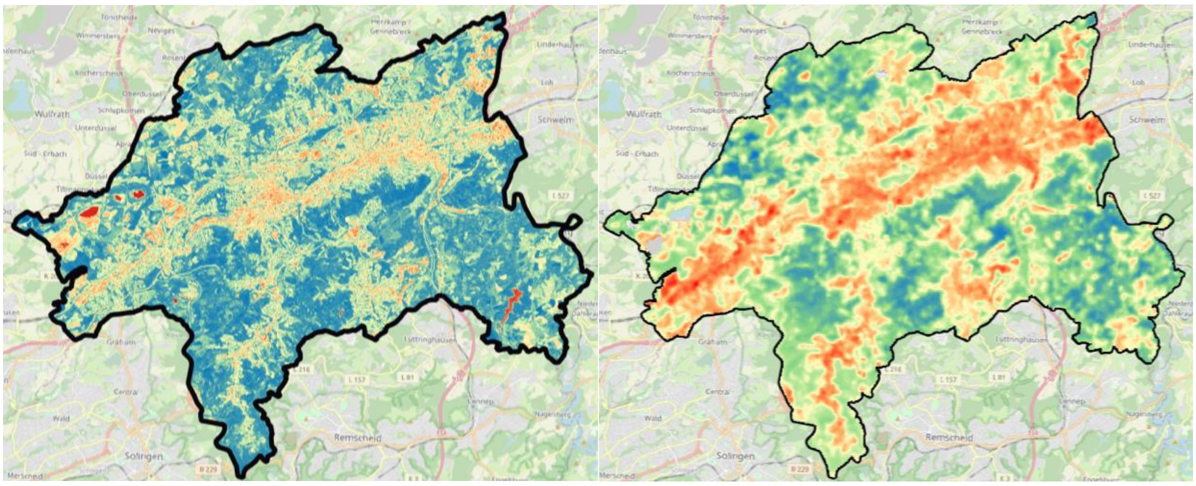


Figure 23: Comparing the NDWI map and the LST map. The right image is the NDWI map. The left image is the LST map. Source: Author, 2024.

6.2.3. Land Use Classification

The land use map is downloaded from the Wuppertal municipality website[6]. The land use map contained many classes, few of which were similar. As a result, the land use map is reclassified and coded. Figure 24 shows that the final land use map consists of eight land use classes. For example, green areas are the vegetation areas. Yellow areas represent residential, and red areas represent commercial and industrial areas.

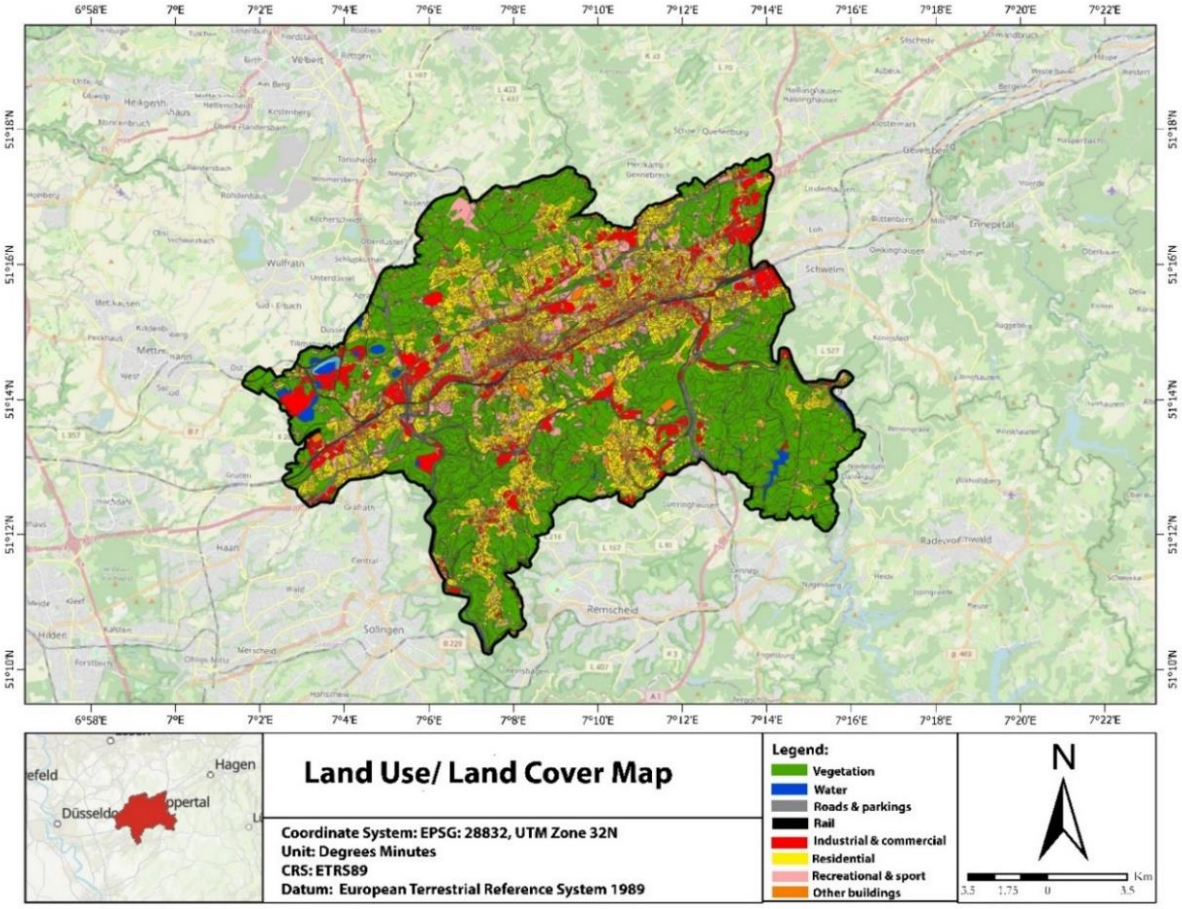


Figure 24: Land-use map for the Wuppertal City. Source: Author, 2024.

Comparing the land use map to the LST map (Figure 25) shows where the land use is vegetation, water, and recreational, and the temperature is low. On the other hand, where the land use class is residential, industrial, and road, the temperature is high. This highlights the impact of temperature on surface material and construction materials. From the literature, the type of land use is also identified to impact temperature and UHI (H. Zhang et al., 2013). This map is used to calculate land use type for sample data to train the ML model in this study.

[6] https://www.offenedaten-wuppertal.de/search/tags/Geo-17

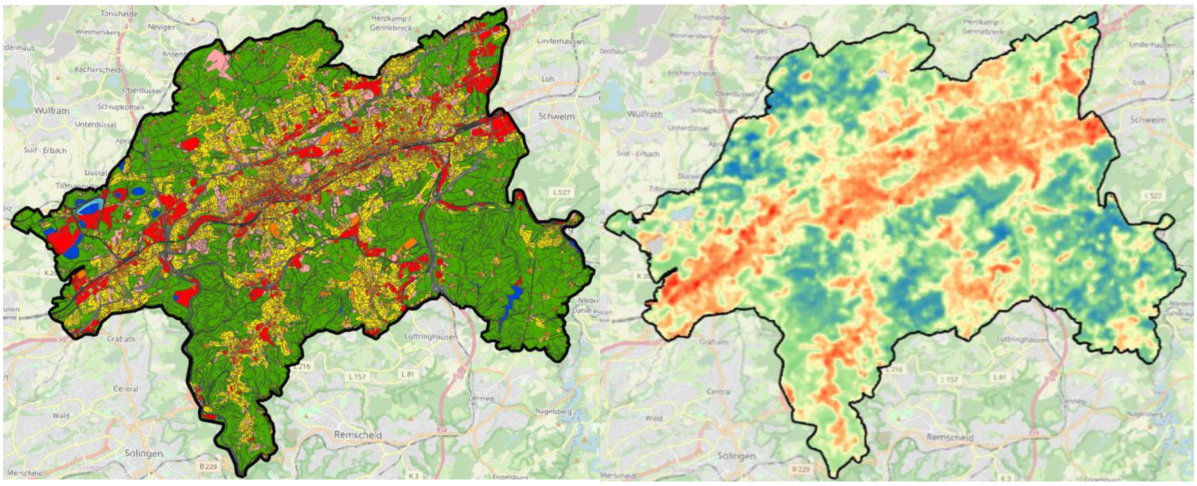


Figure 25: Comparing the land use map and the LST map. The right image is the land use map. The left image is the LST map. Source: Author, 2024.

6.2.4. Population Density

Population density is downloaded from the Wuppertal municipality website[7]. The resolution for population density is not very high due to privacy. Figure 26 shows the population density map for Wuppertal city. The map resolution is 1000m per pixel. The areas with a higher population density are in dark red, and the areas with lower population density are in white. Comparing the population density map to the LST map (Figure 27) shows that the area with higher population density has higher temperatures than areas with lower population density. The literature also identified that anthropogenic heat released from human activities impacts temperature. Thus, where the population density is high, the temperature is higher than in areas with less population density (Santamouris, 2015). This map is used to calculate population density for sample data points to train the ML models.

[7] https://www.offenedaten-wuppertal.de/search/tags/Geo-17

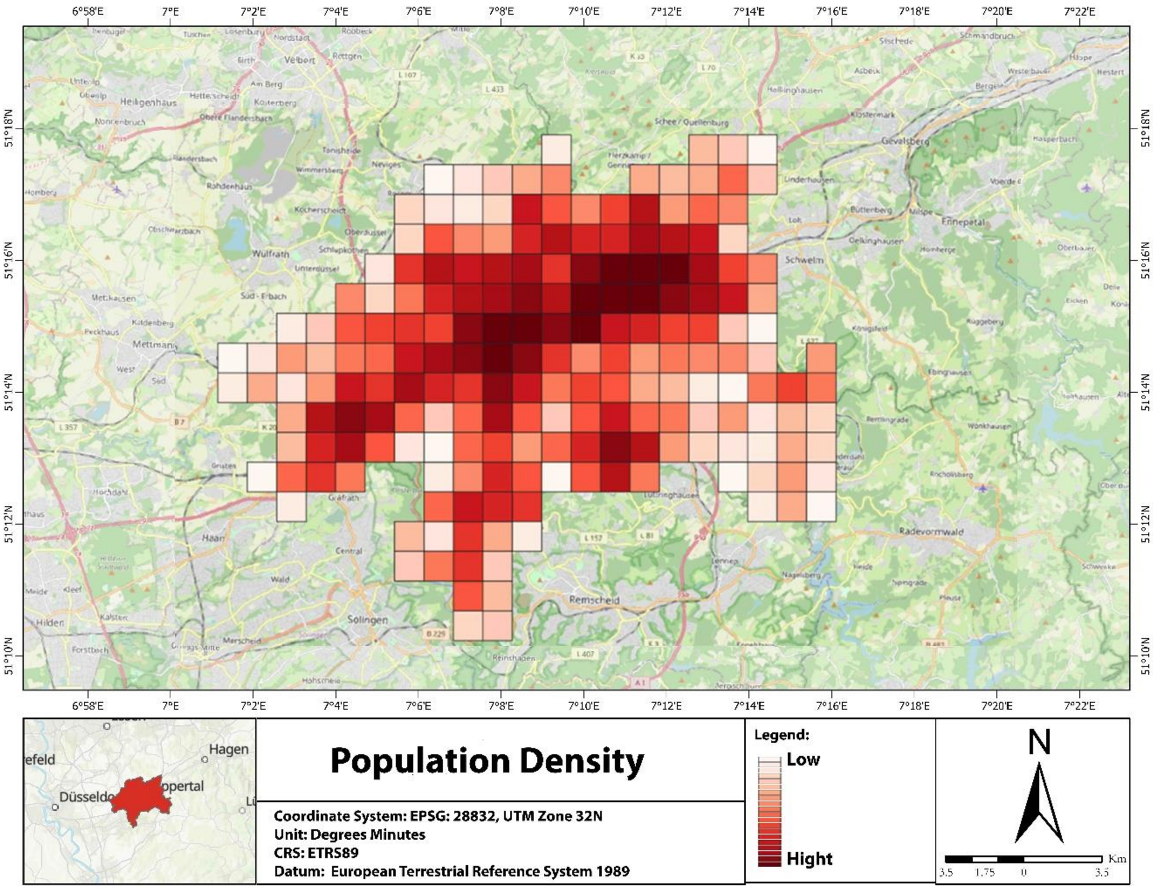


Figure 26: Population density map. Source: Author, 2024.

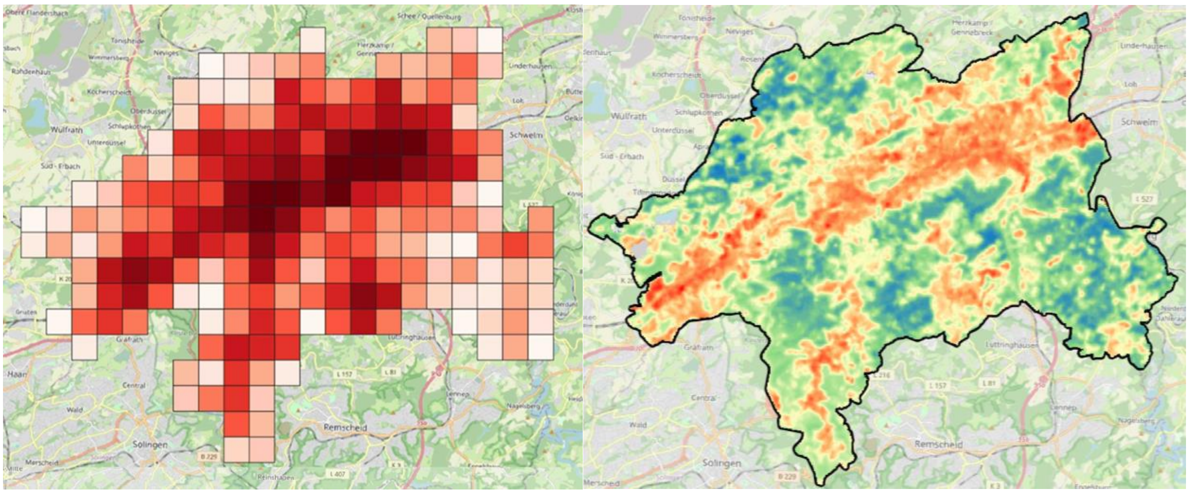


Figure 27: Comparing the population density map and the LST map. The right image is the population density map. The left image is the LST map. Source: Author, 2024.

6.2.5. Digital Elevation Model

This study generated DEM using the contour map downloaded from the Wuppertal municipality website[8]. The process of creating a DEM map is explained in section 5.1 of the methodology. The resolution for the DEM is 10m. Figure 28 shows the DEM for Wuppertal city. Areas with higher elevation are displayed in red, and areas with lower elevation are displayed in blue.

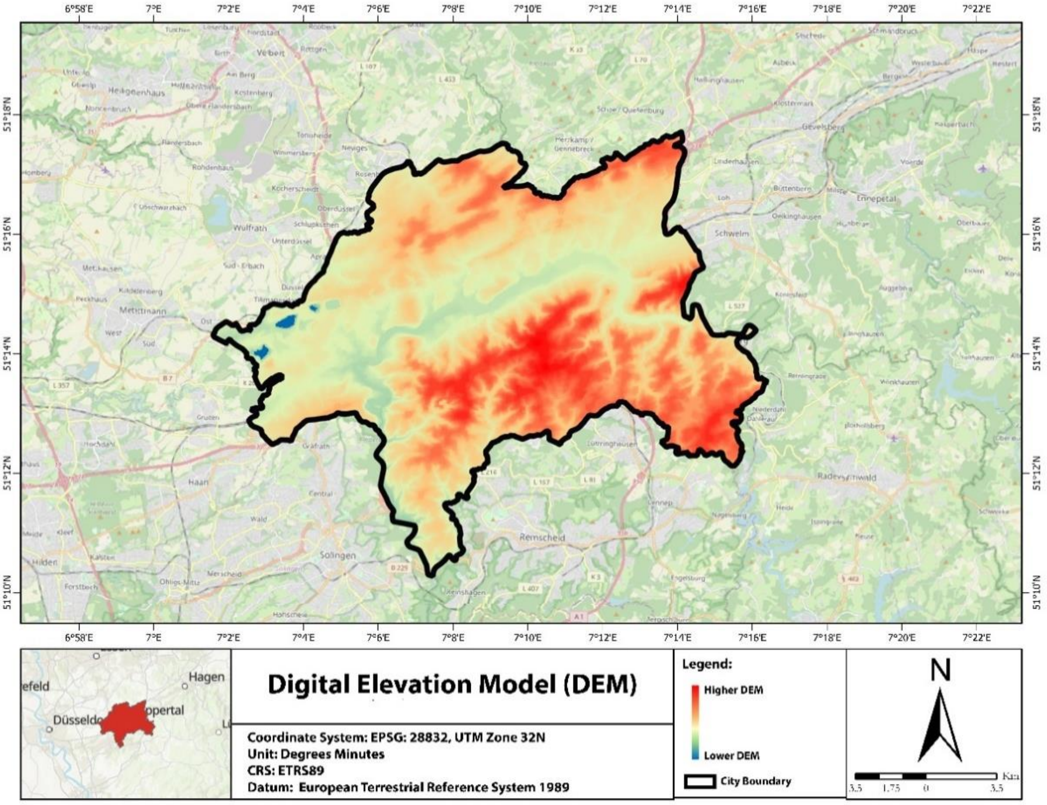


Figure 28: Digital Elevation Mode. Source: Author, 2024.

Figure 29 shows the comparison between the DEM map and the LST map. This comparison shows that areas with higher elevations have a higher temperature compared to areas with lower temperatures. The blue line in the middle of the map is the Wupper Vally (Wupper River), where Wuppertal city is located around. As stated in the research problem, one of the reasons for UHI formation in the centre part of Wuppertal is the Wupper Vally. Wind cannot reduce temperature since the valley prevents wind flow in the city centre. This might be the result of the temperature difference between the low and high-elevated areas in Wuppertal City. The DEM map created in this part is used to calculate the elevation of sample points in this research.

[8] https://www.offenedaten-wuppertal.de/search/tags/Geo-17

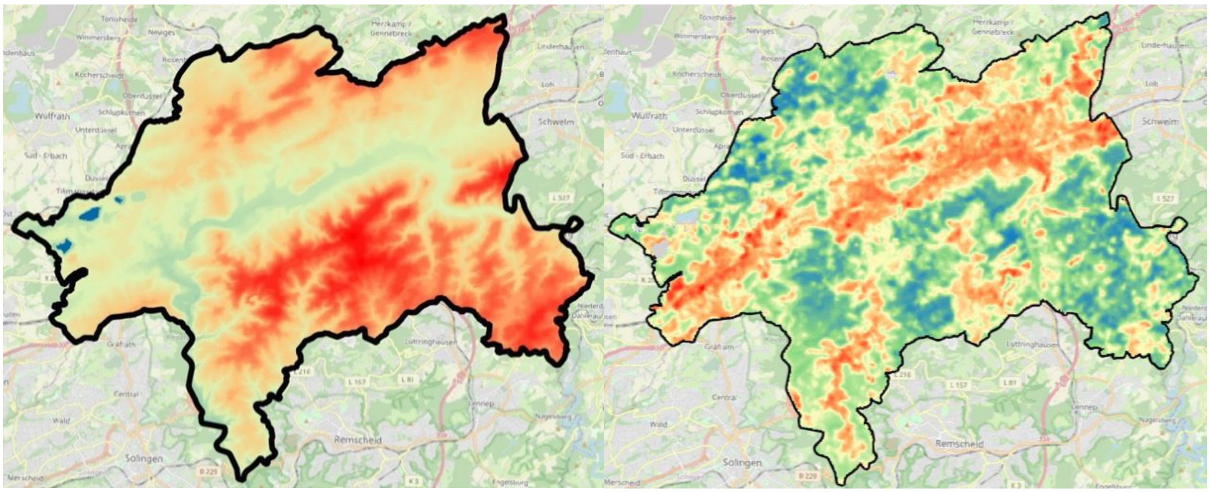


Figure 29: Comparing the DEM map and the LST map. The right image is the DEM map. The left image is the LST map.

6.2.6. Calculating Landscape Indices for Sample Points

For calculating landscape indices, the built-up area for Wuppertal city is generated. The built-up area is generated using the building footprints shape file (Table 2). The shape file is converted to a raster file to generate a build-up map, and the values of 1 and 0 are assigned to the built-up and non-built-up areas, respectively. The resolution for the raster image is 2m. Figure 30 shows the built-up area map for Wuppertal City. The non-built-up areas are black, and the built-up areas are white. Comparing the built-up map with the LST map (Figure 31) show that higher built-up areas have higher temperatures compared to non-built-up areas. The literature also identifies this relation (Wu et al., 2014). Due to the construction materials and pavements in the built-up areas, the temperature is higher than in non-built-up areas. The Build-up map is used to calculate landscape indices for sample points in this study.

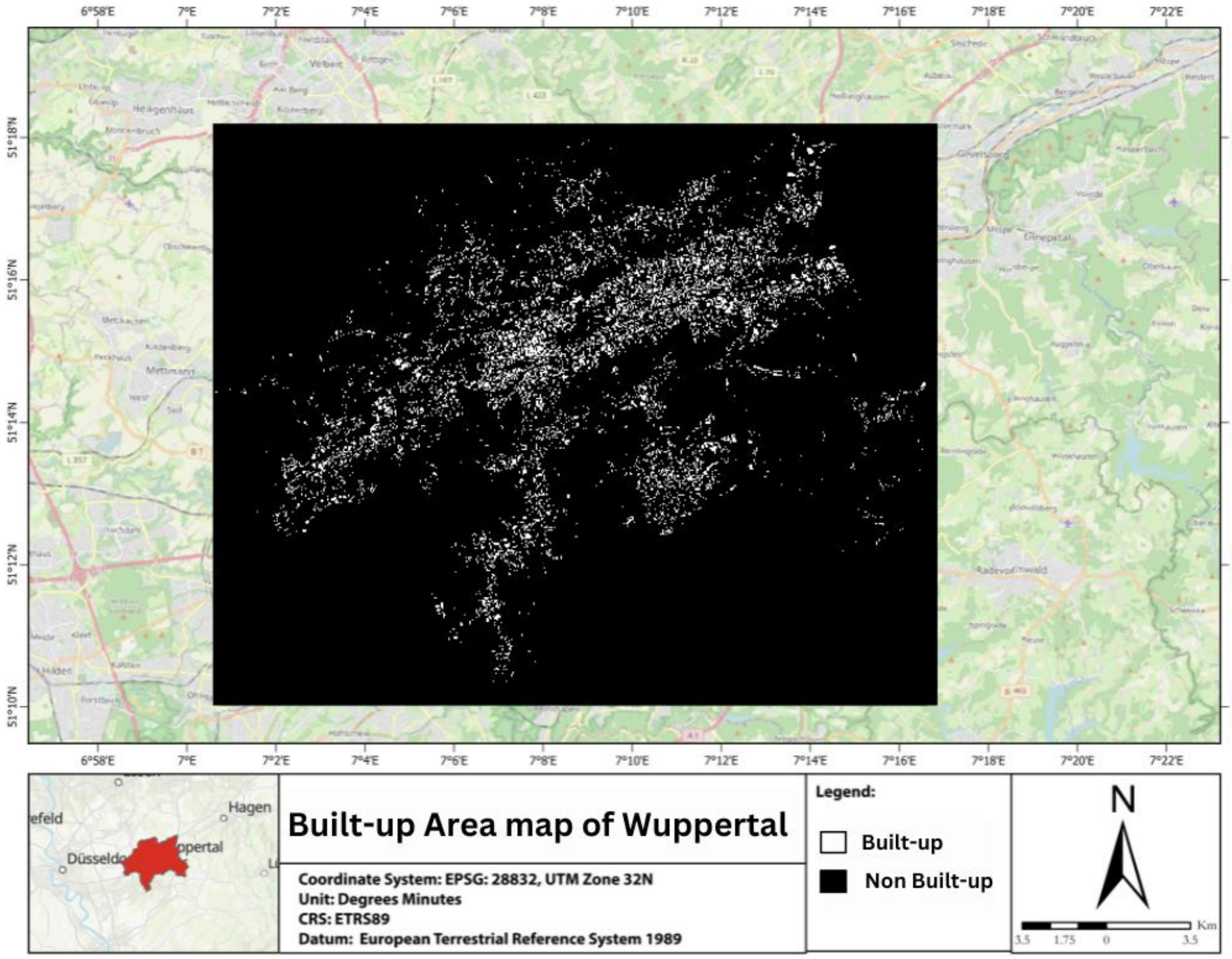


Figure 30: Buil-up map for Wuppertal City. Source: Author, 2024.

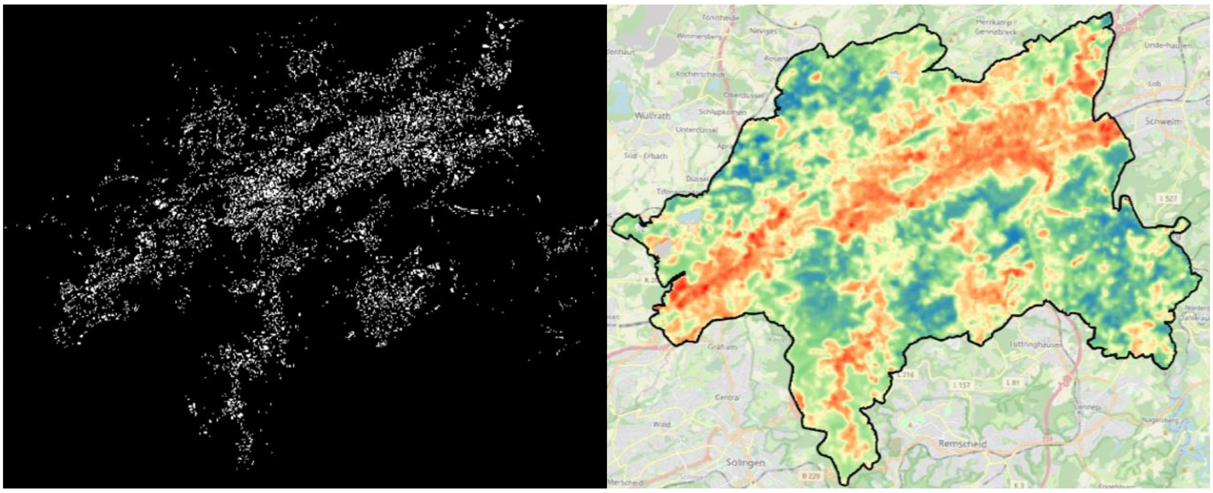
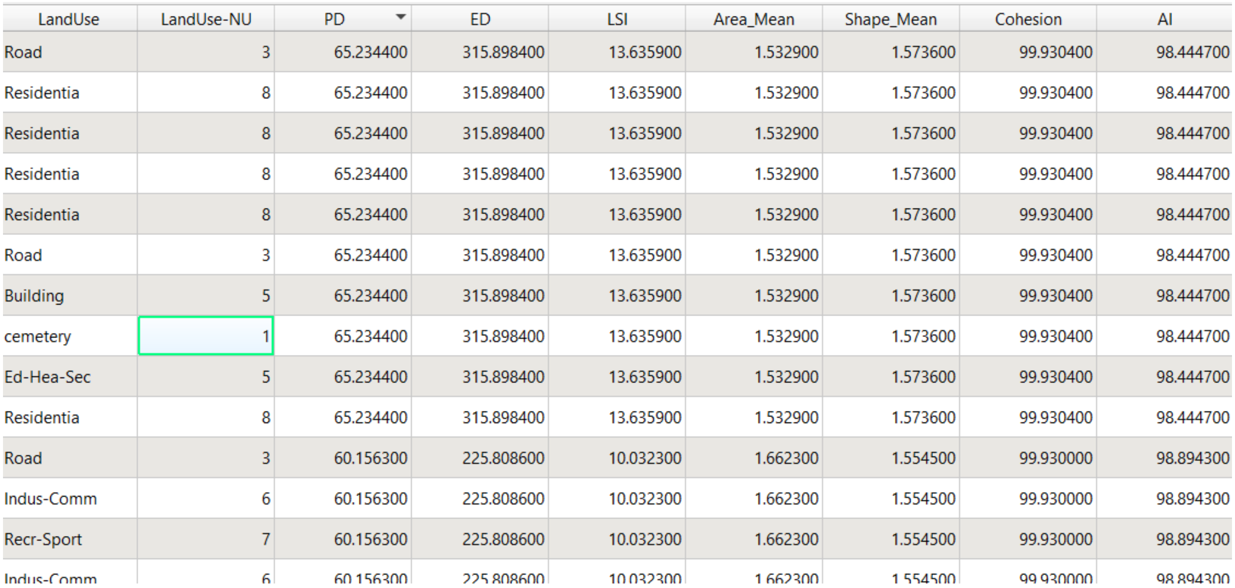


Figure 31: Comparing the built-up map and the LST map. The right image is the built-up map. The left image is the LST map. Source: Author, 2024.

Previous research typically calculated landscape indices at the city scale to facilitate comparison between different cities. However, in this study, the scale of analysis is much lower than the scale of the city. Therefore, the built-up map is divided into smaller tiles to include this lower scale (1000m pixel size). Because of neighbourhood boundary irregularity, the author did not select the neighbourhood boundaries for dividing the built-up map. The irregularity of neighbourhood boundaries might bias the landscape indices analysis. Hence, the scale of landscape analysis is based on the population density layer. The population layer resolution is neither very high nor coarse, with a 1000 m pixel size (high-resolution photos do not yield valuable results for landscape indices).

To calculate landscape indices, the built-up raster image is divided into smaller tiles, the same as the population density tile size. Then, each tile is added to the Fragstats software to calculate the landscape indices. Fragstats is a landscape ecology tool that can calculate various landscape indices. Fragstast performance has been proven in several scientific research. The software has a different input format and analytical setting to calculate patch, class, and landscape scale metrics.

After calculating the landscape indices for each tile, the results are added to the points inside each tile. This is done by selecting points inside each tile and adding the result of related landscape indices to the attribute table of the points. Figure 32 shows the attribute table of each point after adding the landscape indices.



LandUse	LandUse-NU	PD	ED	LSI	Area_Mean	Shape_Mean	Cohesion	AI
Road	3	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Residentia	8	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Residentia	8	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Residentia	8	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Residentia	8	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Road	3	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Building	5	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
cemetery	1	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Ed-Hea-Sec	5	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Residentia	8	65.234400	315.898400	13.635900	1.532900	1.573600	99.930400	98.444700
Road	3	60.156300	225.808600	10.032300	1.662300	1.554500	99.930000	98.894300
Indus-Comm	6	60.156300	225.808600	10.032300	1.662300	1.554500	99.930000	98.894300
Recr-Sport	7	60.156300	225.808600	10.032300	1.662300	1.554500	99.930000	98.894300
Indus-Comm	6	60.156300	225.808600	10.032300	1.662300	1.554500	99.930000	98.894300

Figure 32: Importing calculated landscape indices into the attribute table of selected points. Source: Author, 2024.

6.2.7. Calculating Population Density and Land Use for Sample Points

For calculating population density and land use for sample points, the points that are located in each tile (in the population density layer) and polygon (in the land use layer) are selected, and the corresponding values for population density and land use layer are added to the attribute table of sample points. For example, if the selected points are in a polygon with a land-use class of 1, then this land-use class is added to their attribute table.

6.3. Calculating Spectral Indices and DEM for Sample Points

After calculating variables, sample points are located on the calculated variables data layer using Python. For example, Figure 33 shows how sample points are located on the NDVI raster layer. Then, the mean pixel values around each sample point are calculated using Python code to calculate spectral indices and DEM for each sample point.

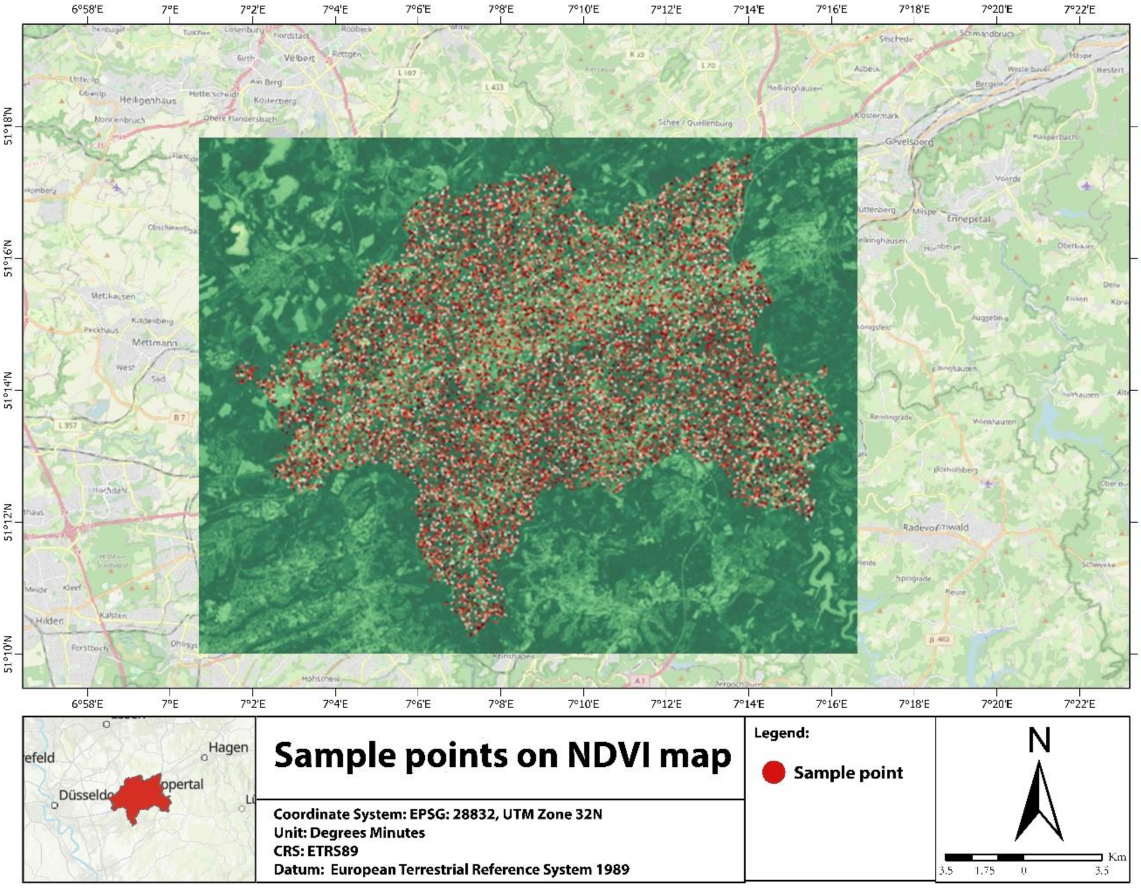


Figure 33: Locating sample points on the NDVI raster layer using (X, Y) coordinates. Source: Author, 2024.

The window size (number of pixels around each point) is set to 1 pixel. This means that 1 pixel is chosen from each side of the points, as illustrated in Figure 34. With a pixel size of 10 meters for NDVI, NDWI, NDBI, and DEM, this configuration gives a mean calculation across a 400-square-meter area surrounding each point.

Figure 34: Number of selected pixels around each sample point. Source: Author, 2024.

A window size of more than 1 pixel means a bigger area around each point, which is meaningful for the analysis. For example, a window size of 2 pixels selects 2 pixels from each side. This results in a total area of 1600m around each pixel. This area is high for correlation analysis. Figure 35 shows the number of pixels on each side for window size 2.

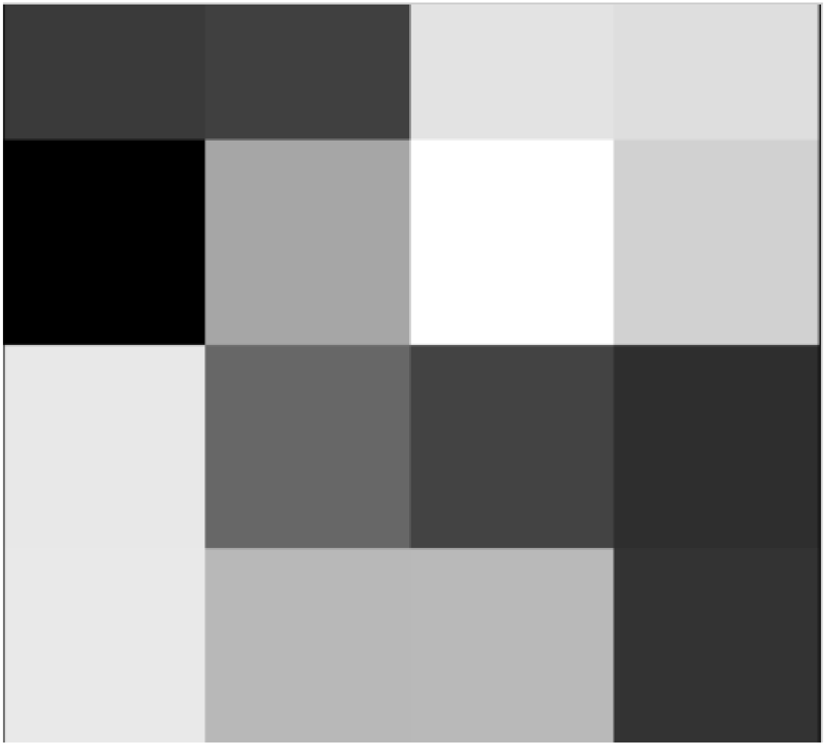


Figure 35: Number of pixels around each sample point for window size 2. Source: Author, 2024.

After calculating the variables, all the variables are added to the attribute table of each point using Python code. The code is available on the 0Annex E. Then, a correlation matrix is created between variables to see the relationship between variables. The correlation matrix is created to ensure that all the variables have a significant relationship with LST before using them as training data for ML models. All the variables show a moderate to significant relationship with LST. Figure 36 shows the correlation matrix between calculated variables for each sample point and LST. Spectral indices, NDVI, NDWI, and NDBI have the highest correlation with LST compared to other variables. The correlation coefficient for these indices is around 0.75, a very strong correlation. However, the built-up area value in the NDWI map affected the correlation result between NDWI and LST. After removing the values between -0.1 and 0.2 (refer to the explanation for Figure 23) the correlation is -0.58. Moreover, population density has the lowest correlation with LST, among other variables. The correlation is 0.46, a moderate correlation with LST. The correlation for landscape indices, PD, AI, ED, and LSI is also strong and higher than 0.5. Correlation is significant at the 0.01 level. This means that there is a 1% probability that the correlation happened by chance. Therefore, the correlation values are reliable. Moreover, the

Figure 36

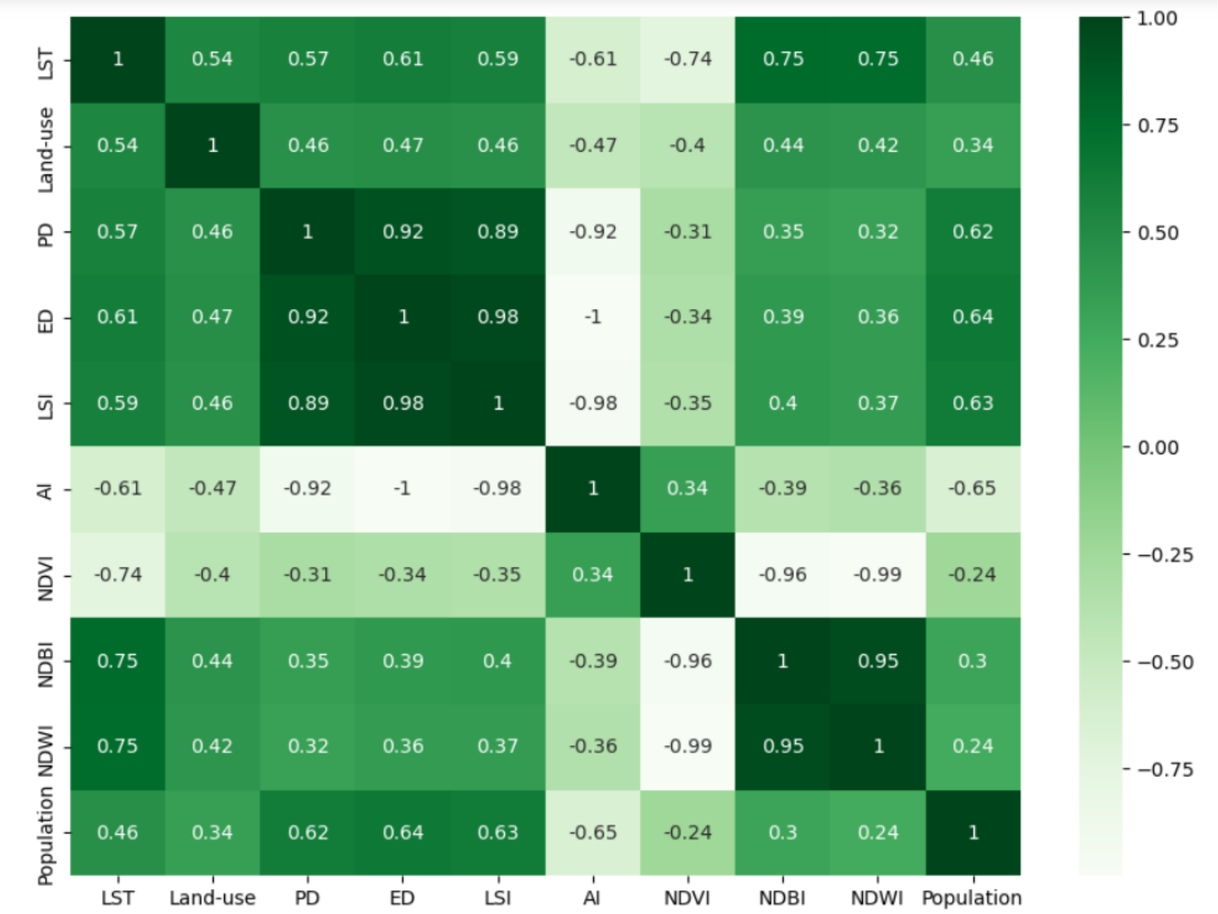


Figure 36: Correlation matrix between variables. Source: Author, 2024.

Correlation is significant at the 0.01 level.

6.4. Machine Learning Models

Two data sets are used to train and test the ML models (refer to methodology, section 5.2). Using the training data set, the ML models are trained. The test data set is used to check the accuracy of the trained models. Four regression algorithms, SVM, RF, ANNs, and the polynomial regression model, are also used in this research. The process of preparing train and test data sets and the logic behind selecting the four regression algorithms are provided in the section 5.2, in methodology.

The accuracy of the trained model is evaluated using two metrics, R-squared and Mean Absolute Error (MAE), which are evaluated using Python code. The Python code for checking the accuracy is available in the Annex B. The R-squared value, also known as a coefficient determination, shows to what extent the variance of the dependent variable is predicted using the independent variables. The more the R-squared, the better the model. Moreover, MAE measures the differences between predicted and actual values in the model. The less the MAE values, the better the model.

For the trained RF model, the result for R-squared is 0.863, meaning the RF model could predict 86 percent of the variance of the LST by the predictors. The result for MAE is 0.98 degrees Celsius, which means, on average, the LST values made by the RF model are around 0.02 degrees Celsius away from the actual LST. R-squared and MAE are 0.75 and 1.44 degrees Celsius for the polynomial regression model, respectively. This means the trained polynomial regression model predicted 75% of the temperature variance, and the mean predicted temperature is 0.44 degrees Celsius away from the actual temperature. The accuracy for the ANNs and SVM models is low. For ANNs, the R-squared is 0.59, and the MAE is 1.8 degrees Celsius.

This means the ANNs model only predicted 59 % of the variance of the LST. For the SVM model, the R-squared is 0.33, and the MAE is 2.3 degrees Celsius. This means the SVM model only predicted 33 % of the variance of the LST. The low accuracy of the SVM and ANNs model might be due to the land use data. Land-use data is categorical, and SVM and ANNs are not developed to be trained by categorical data. They are generally effective with numerical data. Because of its accuracy, the RF model is chosen as the final model for creating the DT-PSS. Table 3 shows the accuracy of the trained models in summary.

Table 3: Accuracy of trained ML models.

Model	R-squared	MAE
Random Forest	0.86	0.98
Support Vector machine	0.33	2.3
Artificial Neural Networks	0.59	1.8
Polynomial Regression Model	0.75	1.44

Source: Author, 2024.

6.5. Creating the DT-PSS Tool for UHI

This part explains the process of creating the DT-PSS tool in detail. It consists of several sections. In the following section, each section is explained. Figure 11 (methodology part) shows the process of creating the DT tool. It shows how the different elements, such as the RF model, real-time data, and 3D models, are imported and included in the UE to create the DT-PSS tool.

A neighbourhood in Wuppertal City is selected as a test case for creating the DT-PSS tool. Due to computational resource constraints, Graphics Processing Unit (GPU), and memory capacity limitations, creating the DT-PSS tool for the whole of Wuppertal City was infeasible. As a result, a smaller prototype is developed and tested on a neighbourhood scale in Wuppertal City to test the feasibility of creating and performing the DT-PSS tool. This also helped show the potential of the DT-PSS tool before implementing it for the whole city. The neighbourhood that is selected for creating DT is Südstadt (Figure 37). This neighbourhood is mixed with residential, commercial, and other buildings. It also is a dense neighbourhood in Wuppertal City that is suggested by stakeholders to be a test case for the DT tool (refer to the methodology section 5.3).

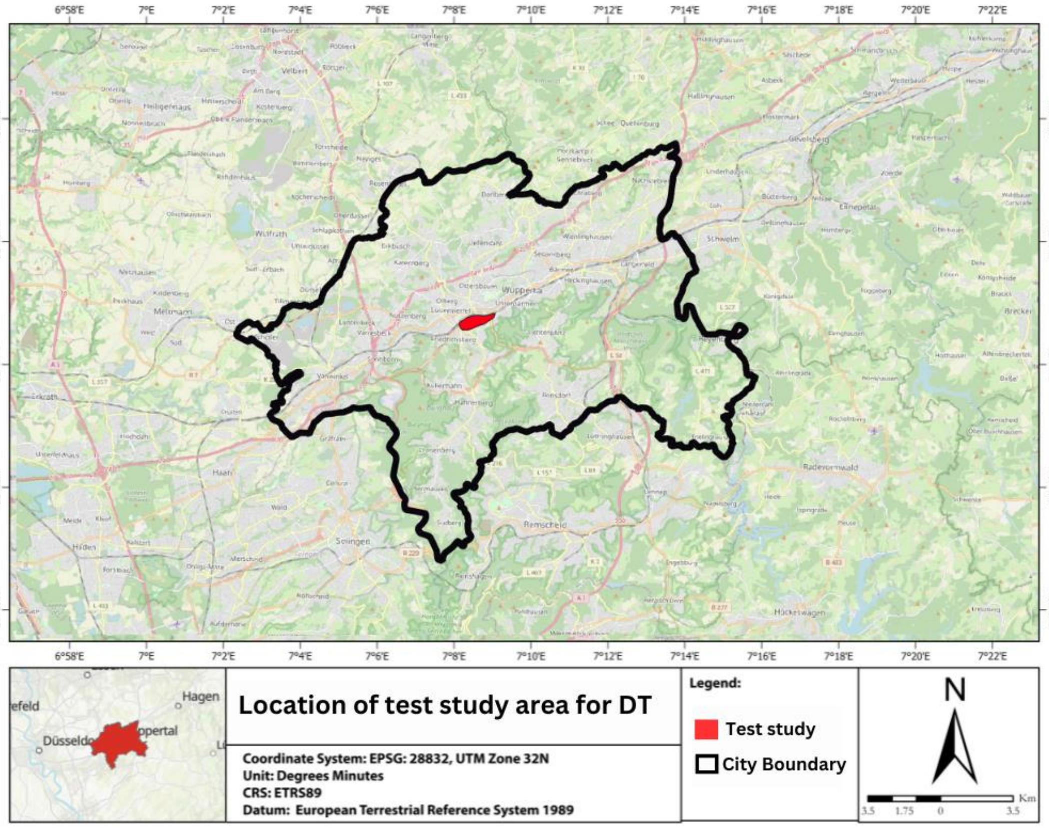


Figure 37: Südstadt neighbourhood location in Wuppertal city. Source: Author, 2024.

One of the reasons for selecting the neighbourhood is that the Südstadt is located between the location of the temperature sensors. The location of sensors creates two triangles, and the selected neighbourhood is located between the two triangles. This makes the analysis for interpolating temperature more precise. Figure 38 shows the two triangles between the sensors. Sensors located at each angle of the right triangle are used to interpolate the temperature for the right side of the neighbourhood. Also, sensors located at each angle of the left triangle are used to interpolate the temperature for the left side of the neighbourhood.

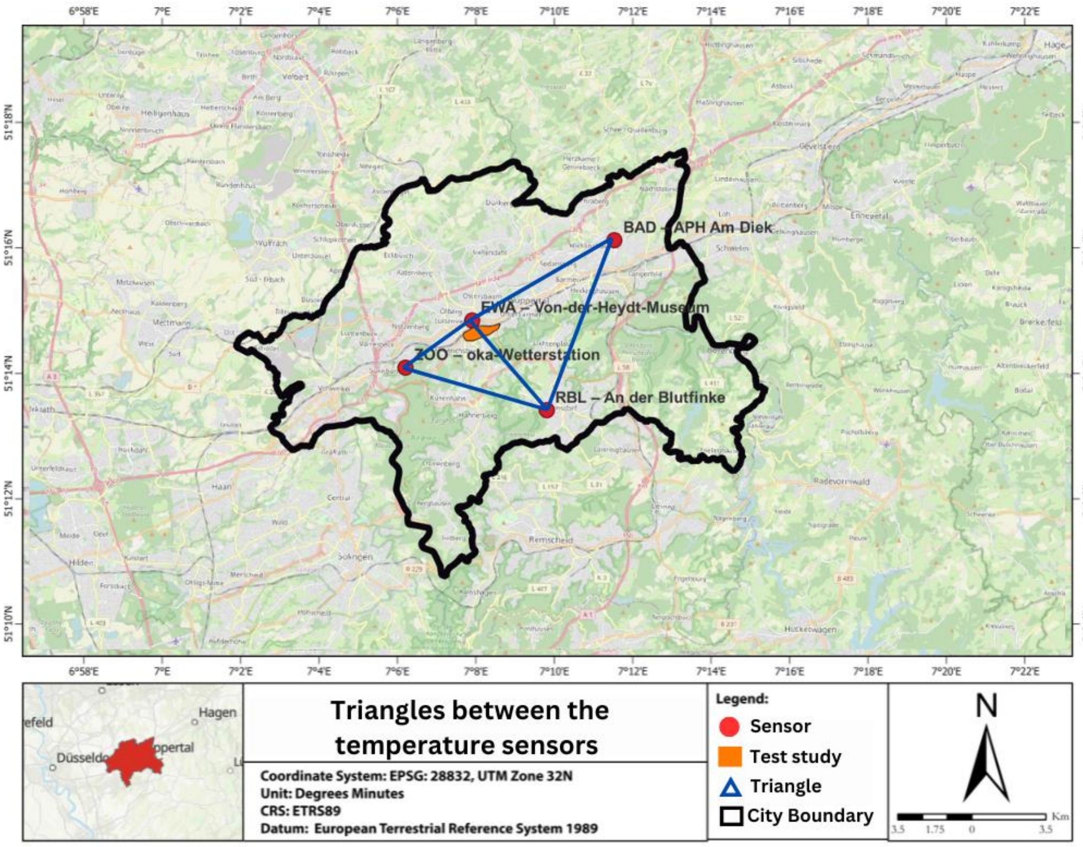


Figure 38: The two triangles between sensors. Source: Author, 2024.

6.5.1. Importing Real-Time Temperature Data

Currently, the municipality of Wuppertal is digitalizing the Wuppertal city due to the DUT-W project. They are installing temperature sensors and plan to install more in the future. So far, they have installed four sensors in different locations. Figure 38 shows the location of different sensors in the city. The real-time data for the sensors is accessible using APIs provided by the municipality. The API addresses and information of sensors are provided in the Annex F.

The VaRest plugin is used to import real-time data into UE. VaRest Plugin is an extension for UE that allows interaction and communication with web services using APIs. VaRest allows UE projects to send and receive the data. Therefore, real-time data can be imported into the UE project, and the project can use this real-time data for analysis. Since the APIs also provide other unnecessary data, the data has been filtered, and only temperature is imported into UE. Annex G shows the visual code (Blueprint) for importing the sensor data into UE. The sensors update the temperature every 10 minutes. After importing the real-time data for temperature, a button on the screen (in the user widget interface) is designed for users to update the temperature (Figure 39). The user can import the real-time data for temperature into the project by pressing this button.

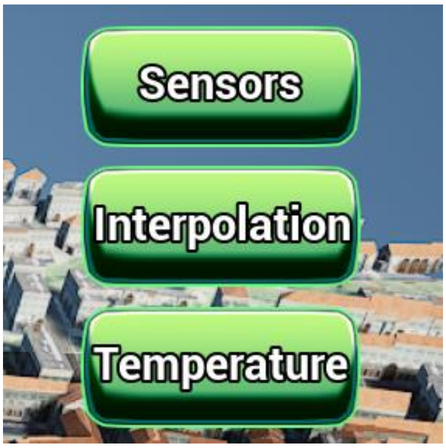


Figure 39: Design a button on the user widget interface to update the temperature. Source: Author, 2024.

After importing the temperature data, the temperature is interpolated for the selected locations within the Südstadt neighbourhood boundary using IDW interpolation. Figure 40 shows the location of selected points inside the Südstadt boundary. The black points represent the selected locations within the boundary of the Südstadt neighbourhood to interpolate real-time temperature.

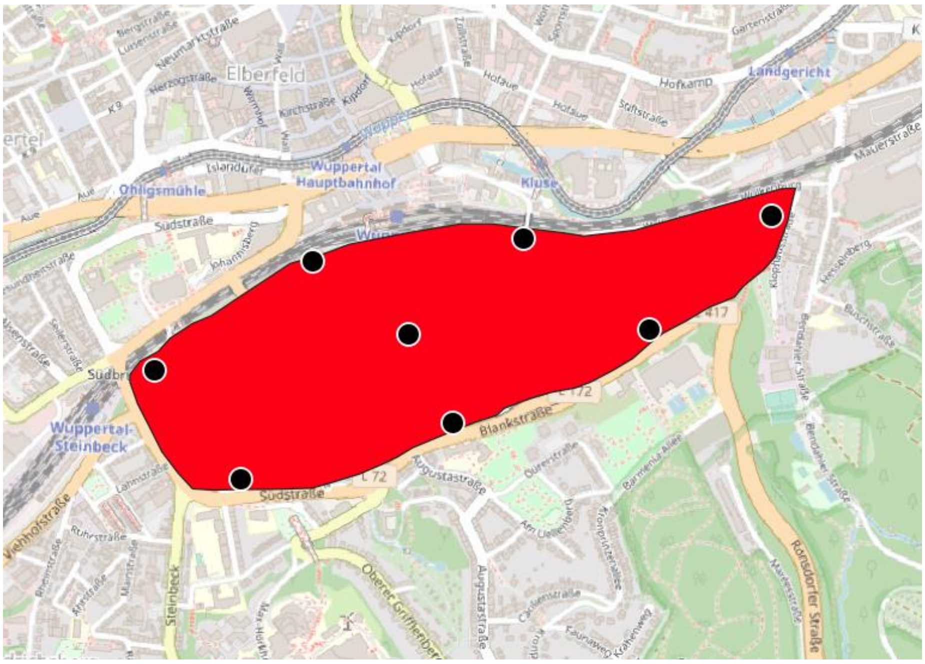


Figure 40: The location of selected points for IDW interpolation. Source: Author, 2024.

The longitude and latitude of the sensors and the selected points inside the Südstadt boundary are used to interpolate the real-time temperature from the sensors to the selected point within the neighbourhood boundary. At first, the temperature data from sensor APIs is fetched. Then, it is interpolated to each selected point using IDW interpolation. The interpolation is done using two coordinate systems: geographical and cartesian coordinates. To make the interpolation more precise, geographical coordinates are converted to cartesian coordinates (due to the curvature of Earth). The difference between the output of these two coordinate systems is slight, around 0.2 degrees Celsius. This study used cartesian coordinates. The Python code for this step is provided in Annex H. The output is in string format for use in UE.

6.5.2. Creating 3D Model of Wuppertal City

To create a 3D city model of Wuppertal, the method in the section 5 (methodology) is followed. The method part explains how the LOD 2 building model is created using Lidar data (Figure 41). All the buildings in the model are joined together as one object. Due to this issue, applying city engine rules to the buildings is impossible. This is because separated objects are needed to apply rules, such as building a façade. Otherwise, the buildings are not separated, and it is impossible to use rules separately for each building. As a result, to apply City Engine rules to the model, the model is divided into separated faces using the separated faces function in City Engine. Figure 42 shows the model after separating all buildings and faces.

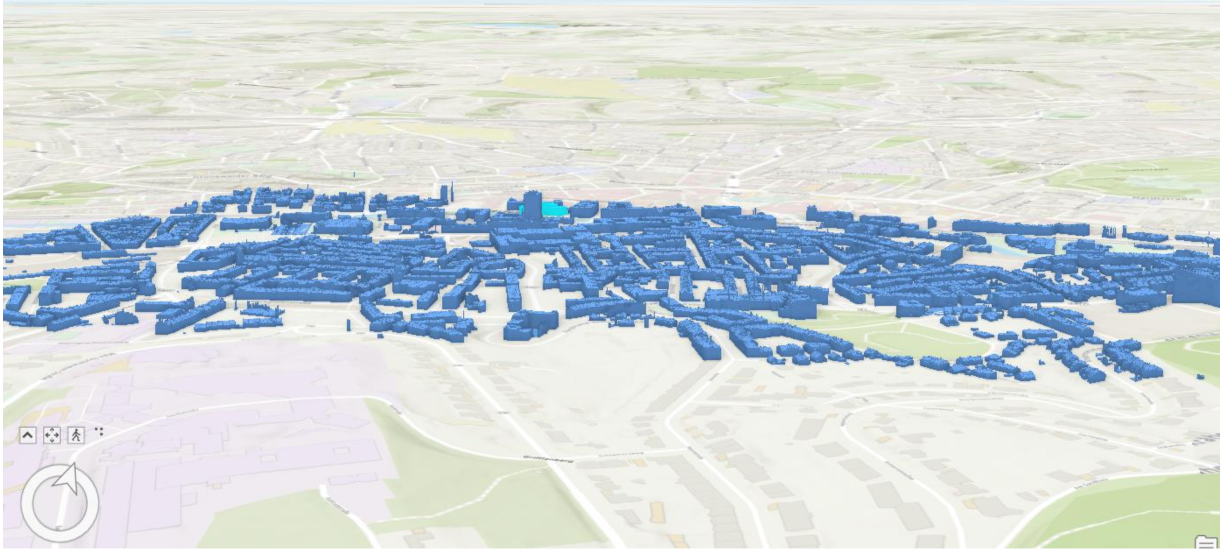


Figure 41: The LOD 2 building model. Source: Author, 2024.

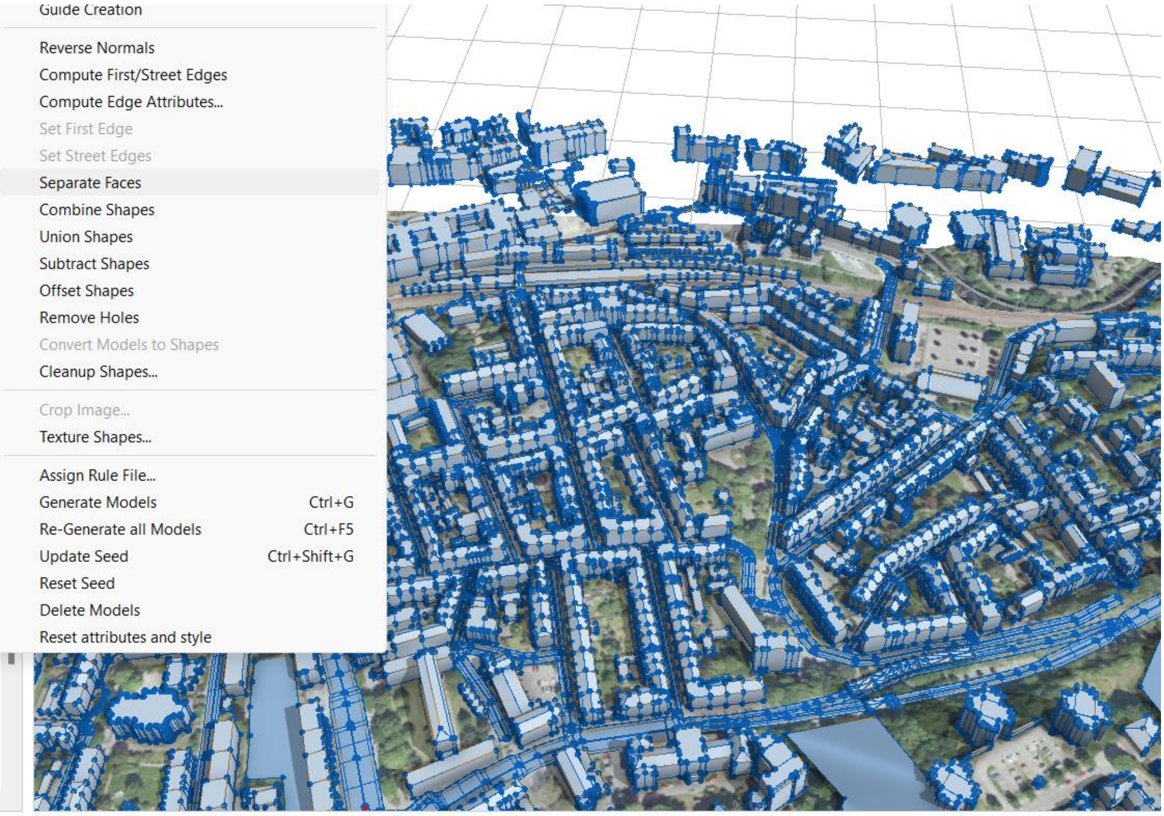


Figure 42: Separating 3D models into building faces. Source: Author, 2024.

In the next step, City Engine rules enhanced the model realism. City Engine rules are procedural scripting written in Computer Graphic Architecture (CGA) language. The rules are written for the procedural generation of 3D urban environments. It is also possible to write new rules for different purposes, which is out of the scope of this research. After separating the model into faces, textures are added to the building facade to make the model more realistic. Predefined building textures and facades existed in City Engine for 3D city modelling, such as New York or Zurich. The building texture of Zurich is used on the model to make a 3D model like Wuppertal City (Figure 43).

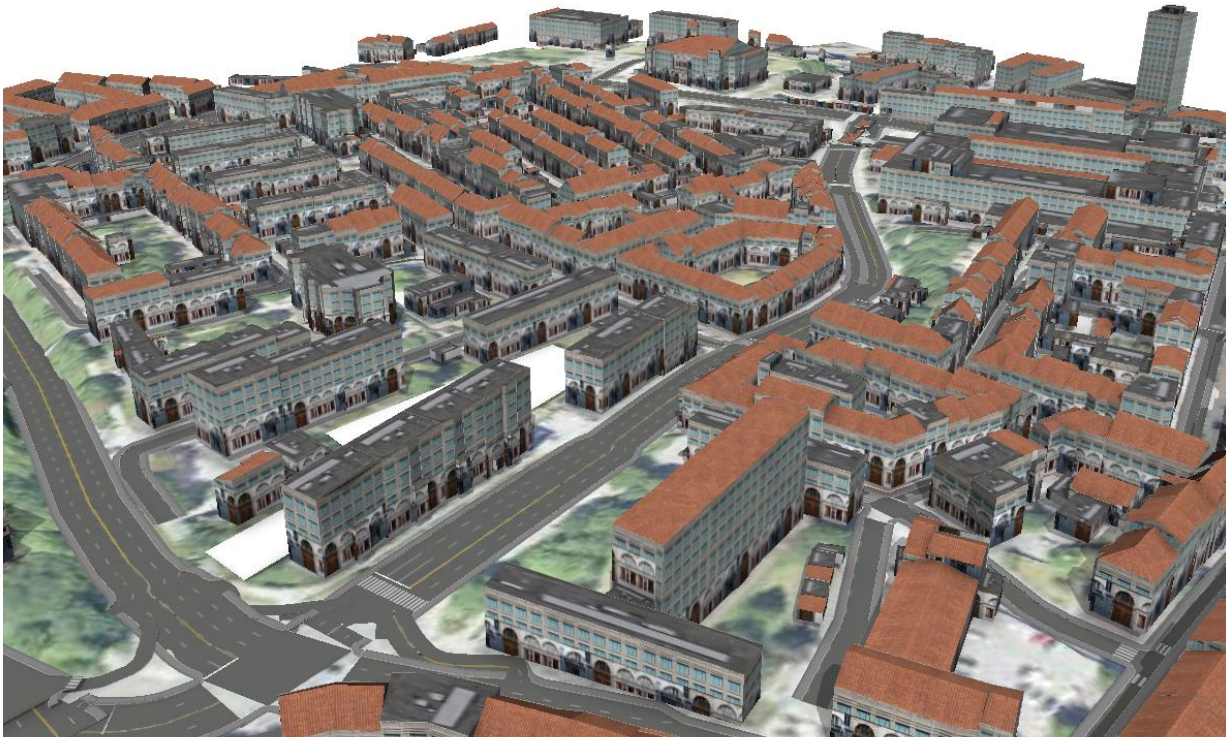


Figure 43: Adding texture to the building facade using the CGA rule for Zurich City. Source: Author, 2024.

Following this, roads and terrain are added to the model by getting map data in City Engine. The base map had the selected neighbourhood elevation and an orthophoto of the neighbourhood (Figure 44). The CGA rule for the complete street is also used to add texture to the streets in the model. The rule is downloaded from the ESRI website[9].

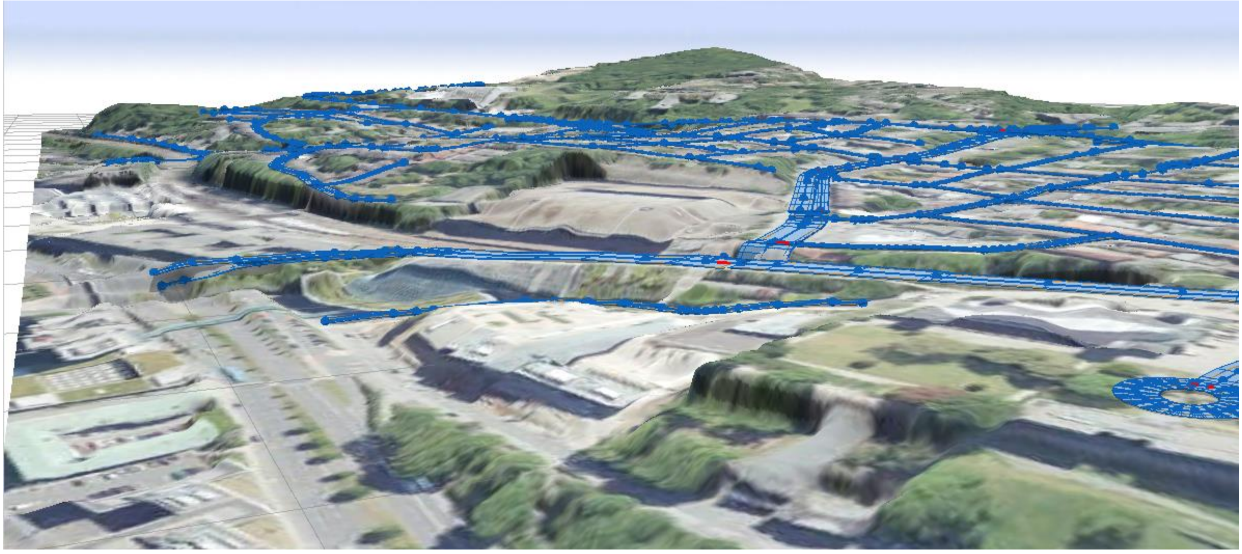


Figure 44: Importing base map and roads into the model. Source: Author, 2024.

[9] https://www.arcgis.com/home/item.html?id=863f4e7139314101a5cee1d7cde079d9

Subsequently, the model is exported from City Engine in the format of .FBX and is imported into Blender to join the building together. This is because after importing the model directly from City Engine into UE, the laptop used for this study could not handle importing and reading the data due to the limitations and constraints of GPU and memory. The created model had over 20,000 separate mesh (mesh pieces for building facades, roads, etc.), making it massive information for the laptop to read and visualise. To address this issue, separated parts in the model, such as building facades and buildings, are joined together to make an urban block as an object in the model instead of separate buildings. Figure 45 shows an object (an urban block) in the model made by joining buildings.

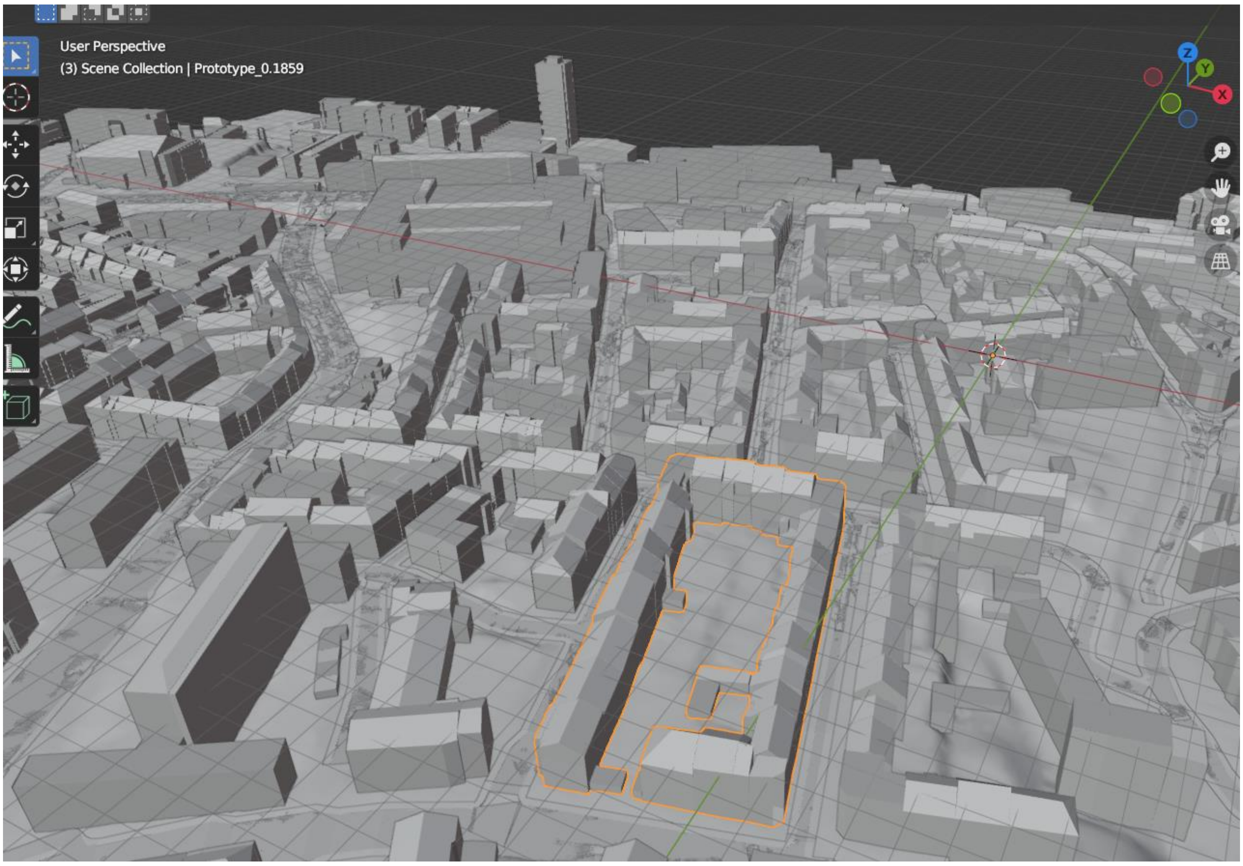


Figure 45: Joining faces and buildings together to make the urban block as an object in the 3D model. Source: Author, 2024.

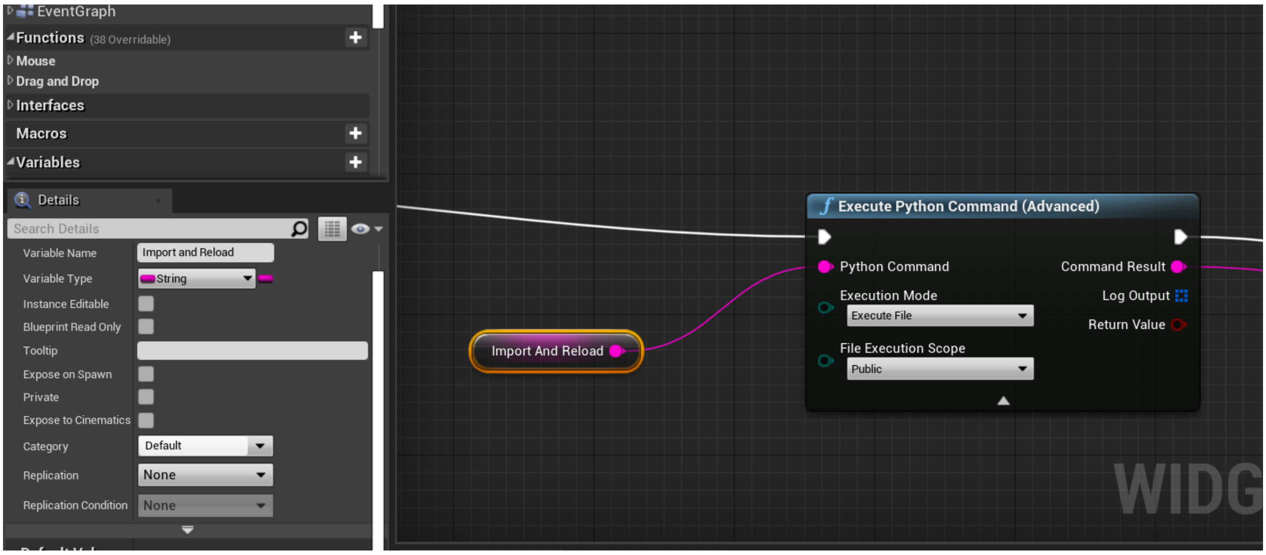
Afterward, the model is exported in .FBX format and imported in UE. The number of pieces is reduced to around 200. Then, UE could read and visualise the model faster and without lagging. Reduction in the number of separated meshed results in improving the performance of UE. This shows the importance of reducing complexity and unnecessary information from the model to match the hardware capability, especially when the model is at the urban scale.

6.5.3. Using Python in Unreal Engine

After using Python scrips and importing the Python files in UE, the output of the functions is not usable in other functions. This issue affected running functions in sequence. To solve this issue, the output of Python codes is changed to string format. Additionally, the output must be a string that prints the result on the screen. As a result, all the code output, such as interpolation or prediction, is formatted as strings (refer to Annex I).

6.5.4. Importing Trained RF Model into Unreal Engine

To Import the trained RF model into UE, the model is saved in .joblib format. This file format ensures that the model is preserved with all its trained parameters intact. Then, using "Execute Python Command" in the UE blueprint, the RF model is imported to the UE. For this purpose, the execution mode is set to execute file (Figure 46). This configuration allowed Python codes to run directly from saved files, facilitating the integration of complex models like the RF.



Variable Name: Import and Reload
Variable Type: String
Instance Editable: □
Blueprint Read Only: □
Tooltip:
Expose on Spawn: □
Private: □
Expose to Cinematics: □
Category: Default
Replication: None
Replication Condition: None

Execution Mode: Execute File
File Execution Scope: Public

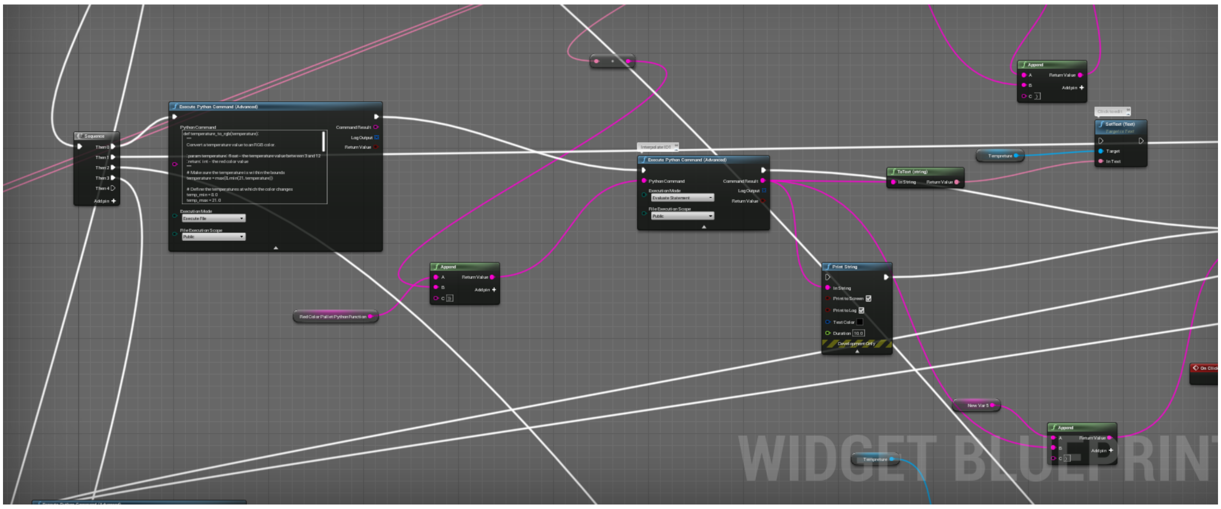
Figure 46: Importing and loading the RF model in UE. Source: Author, 2024.

After importing the model, input variables for the RF model are linked to the Python function. The input variables for the RF model are defined as new float and integer variables in Blueprint that matched the predictor variables used during the model training process. As a result, nine new variables are defined in UE with the same type of input variables (integer type for land use and float type for the other variables). Defining variables in UE is enabled using the RF model in the UE platform. If the variables change in runtime in UE, the RF model can predict the new temperature.

6.5.5. Development of User Interface of DT-PSS

Several key components are used to create an interactive and user-friendly user interface for the DT-PSS. User interface (UI) is a graphical interface that allows users to interact, edit, or create content in UE. It consists of several elements such as "Main Editor Window," "Viewport," "Content Browser," "Detail Panel," and "Viewport Navigation." For example, the main editor window allows users to change game content or various tools. Viewport Navigation also allows users to navigate the viewport using a mouse, keyboard shortcuts, or navigation control.

The Blueprint Widget is used for this study to create the user interface. The Blueprint widget is a UI created using the visual scripting code Blueprint. First, several elements, such as sliders, buttons, and text fields, are added to the widget window. These features are designed to interact with the underlying playing logic, allowing users to alter variables and see changes in real-time. For example, sliders are utilized to change environmental settings, and buttons are added to the model to perform specified activities. Next, the Blueprint system in UE connected UI components to their related functions and data. This is done using visual scripting code in UE. Figure 47 shows an example of visual scripting code in UE.



```
def temperature_to_rgb(temperature):
    """
    Convert a temperature value to an RGB color.

    :param temperature: float - the temperature value between 3 and 12
    :return: int - the red color value
    """
    # Make sure the temperature is within the bounds
    temperature = max(8, min(21, temperature))

    # Define the temperatures at which the color changes
    temp_min = 8.0
    temp_max = 21.0
```

Figure 47: Visual scripting code, Blueprint in UE. Source: Author, 2024.

Blueprint in UE utilizes Event-driven Programming. This capability defines custom events or functions in the game environment. Additionally, Blueprint Widgets support dynamic changes and updates. This allows users to modify elements in UI in runtime and see the real-time changes in the game environment. This flexibility enabled dynamic responses based on runtime changes in DT-PSS, which is an essential capability for creating DT in UE. Figure 48 is the Blueprint widget for the user interface of this study. Each button is an Event linked to a function or sequence of functions in the Event Graph (where visual scripting is created) in UE.

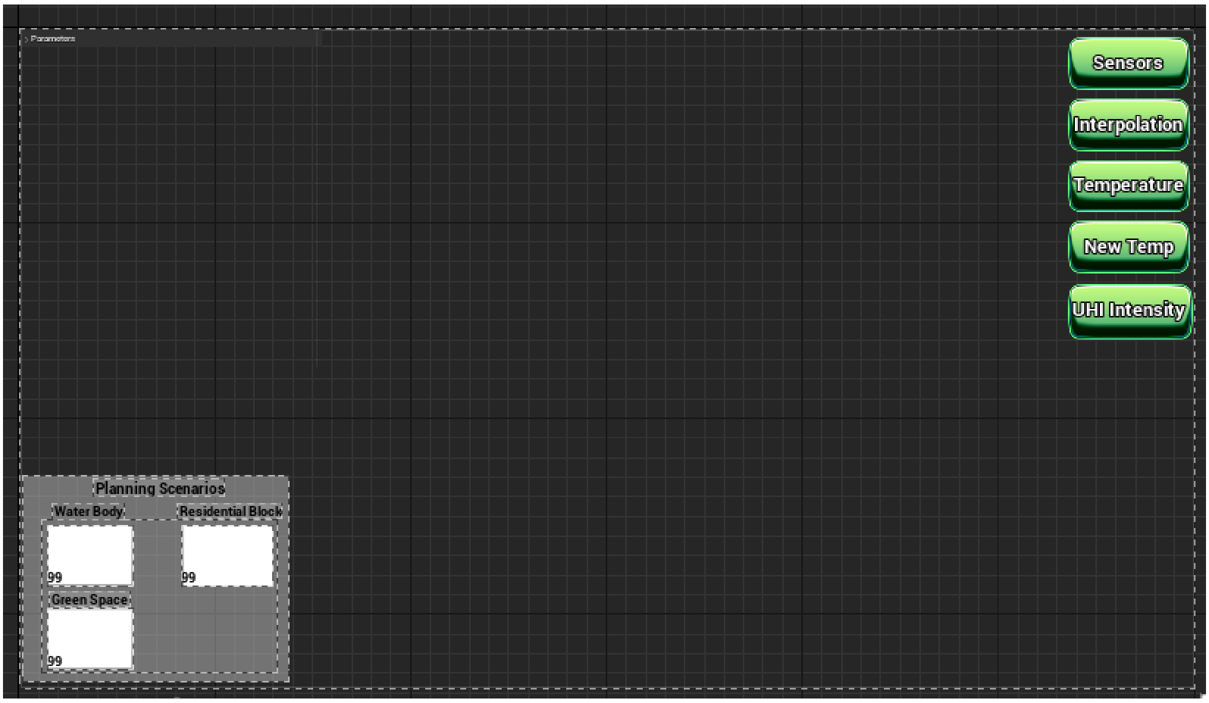


Figure 48: Blueprint widget for UI is used for this study. Source: Author, 2024.

Afterwards, the buttons are linked to Event Graphs by creating Events. Different types of Events for each button run the visual code when users utilize them in the simulation mode. For example, Figure 49 shows different types of events for each button. For this study, the “On Clicked” event (the first one) is used. This means that if the users click on the button, it runs the visual codes or sequence of functions in

Blueprint. The "On Clicked" event in this study is linked to the visual code in Blueprint. Figure 50 shows a link between the "On Clicked" event and the visual code in Blueprint.

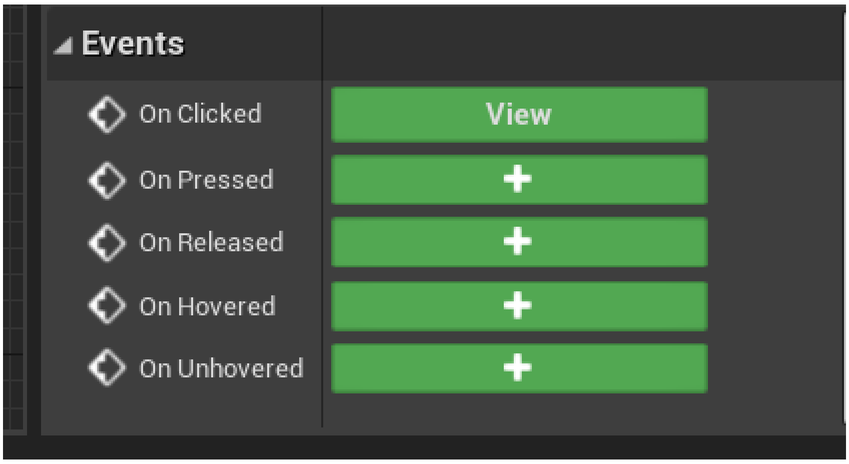


Figure 49: Different Events for buttons in UE. Source: Author, 2024.

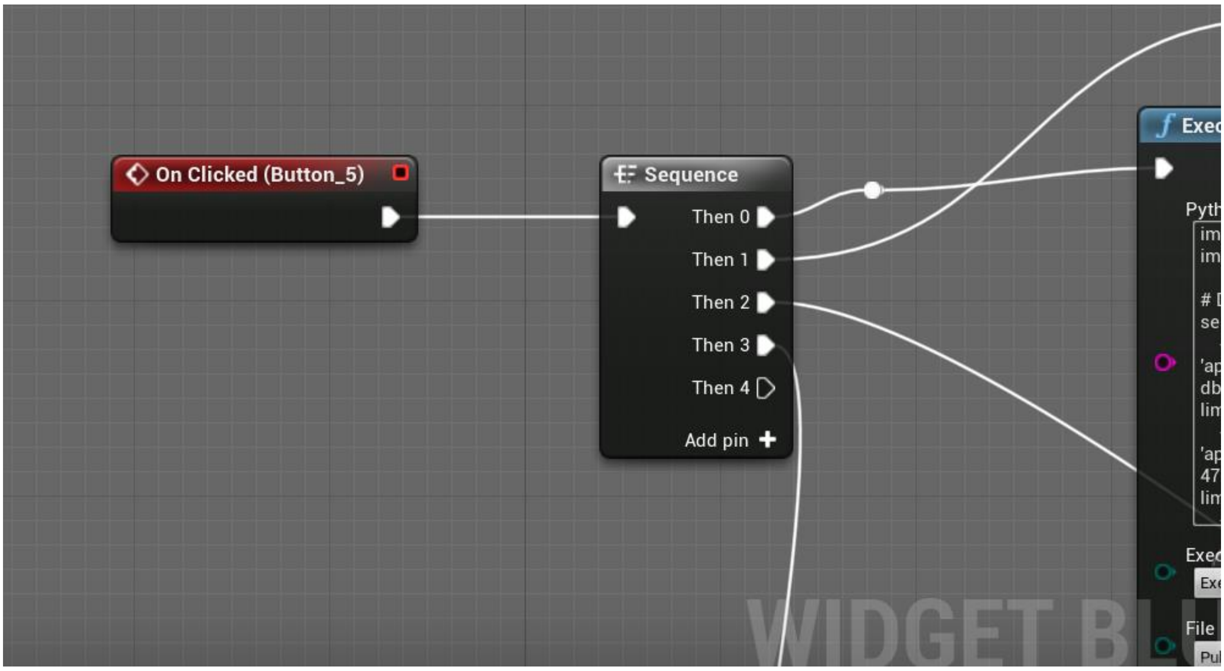


Figure 50: Linking event to visual code in Blueprint. Source: Author, 2024.

6.5.6. Coordinate Reference System in Unreal Engine

This research utilized the UE Coordinate Reference System (CRS) to locate the object within the UE scene. UE employs a standard 3D CRS to define and locate the objects in the scene. UE operated two spaces: world and local space. World spaces refers to the global coordinate system of the entire game environment. In the world space, locations and positions of all objects are related to the origin of the game world, which is (0,0,0). Object space refers to a local coordinate of objects. This means all the positions and locations are related to the origins of an object. For this study, the world space coordinate system is used to determine the position and location of objects in UE. It is also possible to bring a GIS coordinate system into UE. First, the coordinate system must be compatible (e.g., UTM) with UE, and coordinate system plugins must be installed.

6.5.7. Scenarios in the DT-PSS Tool

To assess the impact of planning scenarios and processes by the proposed DT-PSS tool in this study, translating decision variables to input variables for the ML model is required. This is because the input variables on the DT tool to predict temperature differs from the decision variables. For example, spectral indices such as NDVI and NDBI are not decision variables in the planning process (they are input variables for the ML model in DT-PSS). Instead, increasing greeneries or building density are examples of decision variables in the planning process. After translating the planning processes to input variables for the DT-PSS, the tool can assess the effects of the planning process on the UHI. Thus, this study translated planning scenarios into input variables to define scenarios in the DT-PSS. The following describes the process of designing and defining planning scenarios in the DT tool.

A feature is defined in UI to develop multiple urban planning scenarios in the UE platform. This feature includes planning/mitigation scenarios, such as constructing new residential blocks, green spaces, or water bodies (Figure 51). Users can drag and drop these scenarios into the scene to see the change in temperature resulting from this intervention.

The planning scenarios in the UE widget interface are defined by translating urban planning scenarios to input variables for the RF ML model in DT. For example, one scenario is constructing new residential buildings by removing greenery. This scenario translated urban planning variables like population density and land use into ML model inputs like NDVI and landscape indices. For instance, constructing new residential buildings will decrease NDVI and increase population density.

Two steps are taken after introducing scenarios to the UE scene to analyse the impact of scenarios. First, new input variables, such as NDVI, NDWI, and landscape indices, are calculated. Second, these calculated variables are imported into the RF model as input to calculate the new temperature and UHI intensity.

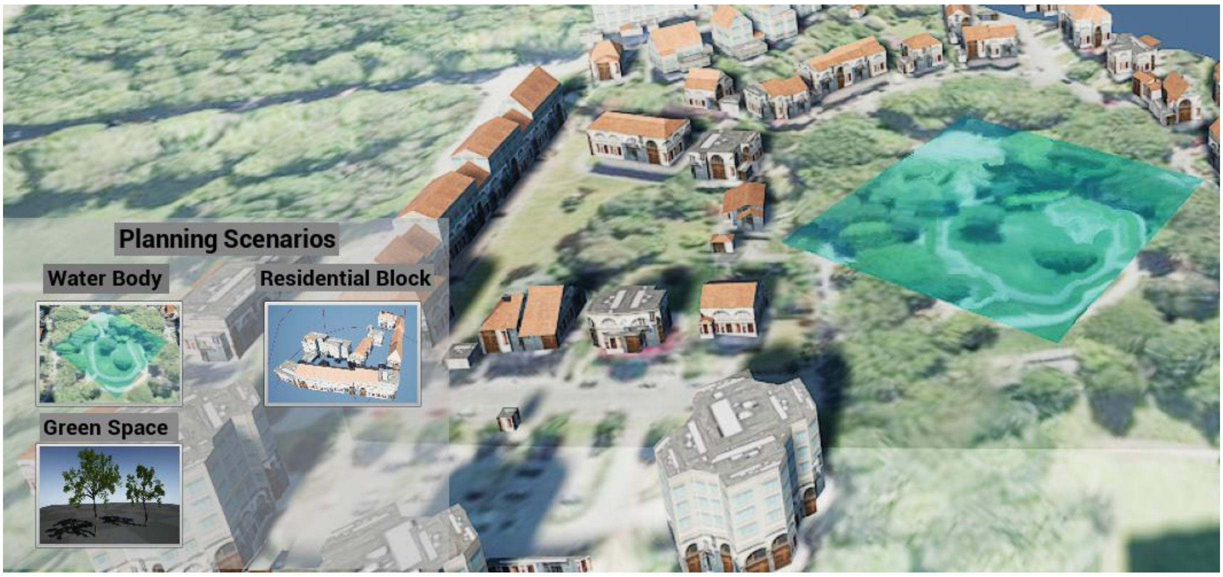
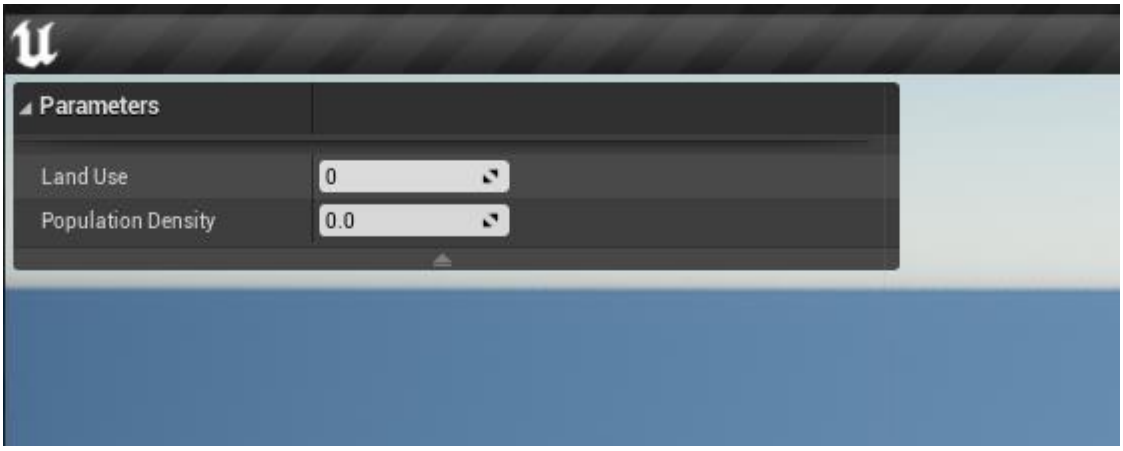


Figure 51: Planning scenarios in the UE widget interface. Source: Author, 2024.

The second step involves using new variables as an integer input for the RF model and creating a function that links them to the RF model (refer to section 6.5.4, importing ML model into UE). No new calculations are needed for land use and population density variables. This is because a new value can be defined by the user for these two variables. In addition, these two mentioned variables are decision variables in urban planning. For instance, if the new scenario is about increasing the population density by 10%, the user can write this number as a new variable for population density. Similarly, users can change land use by changing the land use class in the widget. Figure 52 displays the land use and population density variables in the user widget interface so that users can change them.



Parameters

Land Use: 0

Population Density: 0.0

Figure 52: Land use and population density in the user widget interface. Source: Author, 2024.

New variables for spectral indices are calculated by corresponding values for each surface. For example, values approaching -1 are water, or values between 0.2 and 0.4 are shrub. So, for constructing a water body, the NDVI value is -1. Although it is possible to calculate spectral indices instead of default values, due to the constraints in computational resources, corresponding values for each scenario are used.

Calculating landscape indices for new buildings is more challenging since changes in the built-up area influence all landscape indices. The new landscape indices should be calculated using software such as Fragstat, and it is not possible to calculate them in UE. There is a possibility of writing Python code that also calculates the landscape indices, but it is out of the scope of this research due to time constraints. Thus, for proposed scenarios in UE (the scenarios developed as a feature in the UI), the landscape indices are calculated using Fragstat software. New variables are imported for the scenarios in the UI. Then, when users drag new scenarios to the scene, the trained RF model calculates the new temperature based on the predefined landscape indices.

6.6. Evaluation

This part explains the results of workshops and discusses the feedback from stakeholders. The workshop is organized to facilitate dialogue between researchers and stakeholders. Representatives from different departments of Wuppertal municipality participated. The main objective of the workshop was to showcase the capability of the DT tool and gather feedback for further improvement of the model. During the workshop session, the research results and the model performance were presented to the stakeholders in Wuppertal municipality (Figure 53). Along with the model performance, the potential of the DT tool for further development is discussed with municipality officials. Below quotes and feedback from stakeholders are provided:

- Stakeholders stated that the scenarios designed in the DT-PSS tool efficiently demonstrate the ability of the tool. However, the scale of ongoing plans in Wuppertal is much smaller. For example, the suggested scenarios include the implementation of green roofs or redesigning streets with greenery. They are also currently considering designing the bike lanes and streets in Wuppertal.

  "More detail and realistic scenarios could be designed."

- Stakeholders mentioned that future climate scenarios could be integrated into the model simulation to increase the robustness of the model. This would involve including data regarding different scenarios for extreme weather or precipitation changes.

"Climate scenarios could be integrated."

- stakeholders stated that wind is one of the main variables that has impacted UHI in Wuppertal and including it in the DT-PSS is crucial.

  "Wind could be included as a variable in the model."

- Stakeholders stated that the DT-PSS has financial and environmental benefits. Environmentally, managing UHI reduces the demand for cooling energy; thereby, it will reduce CO2 emissions that result from the production of electricity. Financially, the tool helps reduce the UHI effect so that households can save on the energy cost for cooling effects.

  "The municipality will benefit from urban heat island management in terms of CO2 reduction. Such tools will contribute to sustainable urban planning and also to environmental aspects." Urban heat island management reduced reliance on air conditioning. Reduced reliance on air conditioning also leads to financial savings for households."

- During the workshop, stakeholders provided valuable feedback on the proposed DT-PSS. They highlighted the role of the tool in decision-making and implementing sustainable planning. They stated that the tool could help them to assess the consequences of their decision in advance. The tool also can help them improve sustainable development.

  "*Urban development and planning can only reach sustainability if different scenarios can be evaluated based on the current situation and future planning. Therefore, considering the effects of urban planning on urban heat islands is essential. The only framework that enables the city of Wuppertal to implement sustainable planning is the concept of urban digital twins.*"

  "Climate change adaptation is important. Urban planners must consider the consequences of their planning with respect to living quality and heat-related burdens on urban citizens. The presented tool is a valuable contribution to knowledge transfer and visualisation".

- Based on the stakeholder's point of view, the DT-PSS tool is designed to be user-friendly but needs further development. More functions and features regarding data visualisation and scenario assessment should be included in the tool.

  "*The tool successfully integrates diverse data, such as remote sensing and in situ almost real-time climate data. The processing and visualisation using a gamification approach is user-friendly and will help the municipality deal with climate change issues more efficiently and actively. The municipality is planning to further evolve the tool and use the protype in the context of the urban digital twin of Wuppertal.*"

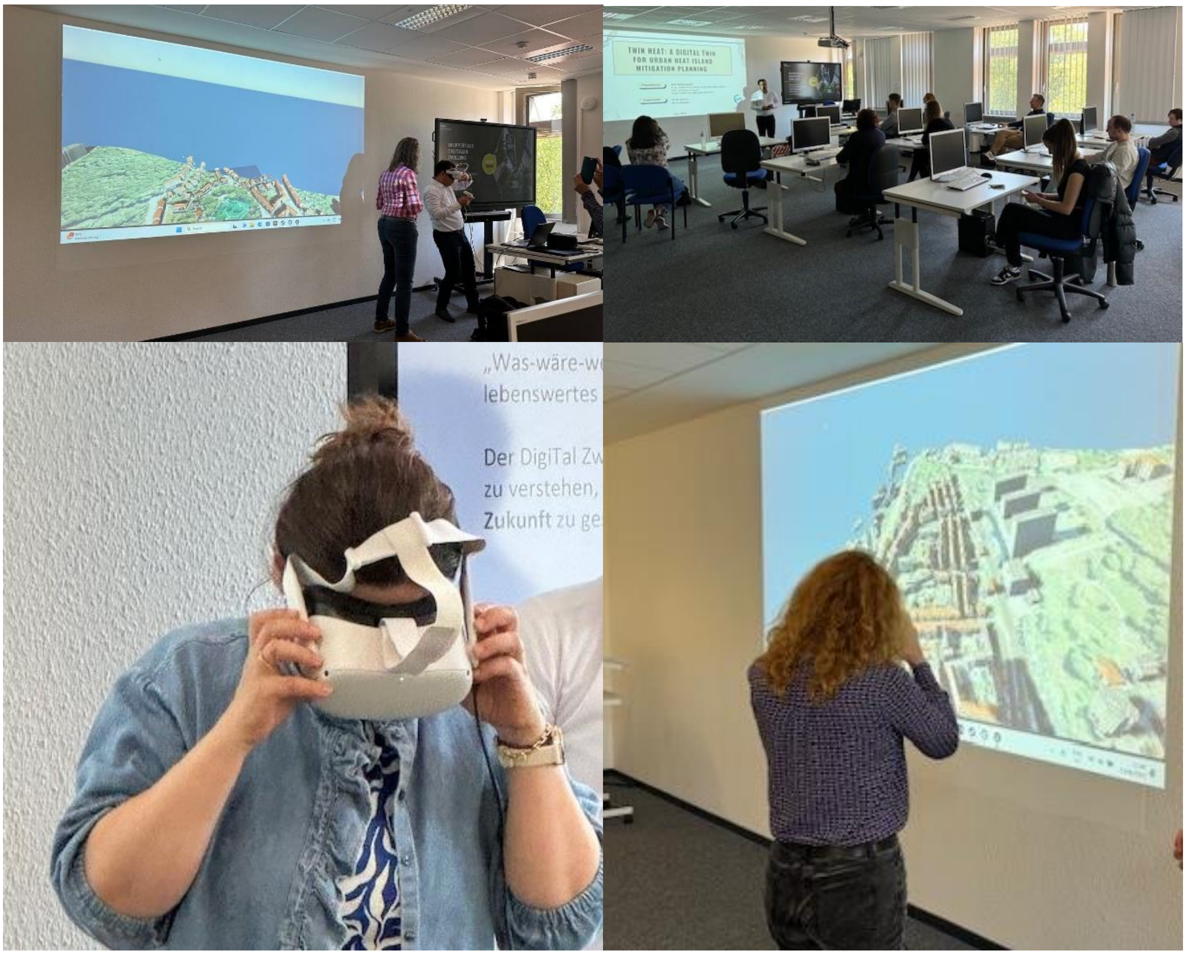

TWIN HEAT: A DIGITAL TWIN FOR URBAN HEAT ISLAND MITIGATION PLANNING
„Was-wäre-w
lebenswertes
Der DigiTal Zw
zu verstehen,
Zukunft zu ge

Figure 53: Workshop with Stakeholders in Wuppertal municipality. Source: Author, 2024.

7. CONCLUSION AND RECOMMENDATIONS

This research developed a DT-PSS in the UE platform to help urban planners mitigate UHI formation. The research shows that by integrating real-time temperature data from sensors and an RF ML model into DT, it is possible to assess the impact of urban planning scenarios on UHI formation. The created DT-PSS in this study provided urban planners with a beneficial tool for assessing the impacts of planning decisions on temperatures and UHI formation before implementing them. The DT-PSS tool in this study provided an ex-ante assessment for planners. Moreover, the tool allowed users to modify variables in DT in runtime and see the real-time changes in the virtual twin. This flexibility enabled dynamic responses based on runtime changes in UE, which is necessary for creating DT.

Furthermore, several key areas are identified in the urban planning and design in which the DT-PSS can be used. The tool can be used to define population density and land use policies. Additionally, the tool provides insights into the impact of different population densities on UHI formation. The tool helps revise design policies, select surface material, and locate green infrastructure to maximize UHI mitigation efforts in urban planning. The tool also can continuously monitor and analyse the data and provide feedback on UHI mitigation measures for adjusting and improving decisions.

Involving stakeholders in creating a PSS is a significant requirement for creating the DT-PSS. The final users should be involved in the design process from the beginning steps since they are the primary users of the tool. By involving stakeholders in the process of creating the DT-PSS, this study ensures that the tool meets the practical needs and user requirements. Engaging stakeholders in the process of creating the tool can enhance the usability of the DT tool.

Furthermore, geospatial data integration provides significant insights into the historical data in Wuppertal City. Since the temperature sensors are installed recently in Wuppertal, there was no historical information on temperature. However, satellite imagery archives make it possible to download and process satellite images of the past years in Wuppertal City. As a result, using satellite images for LST and UHI studies is essential.

Moreover, the DT-PSS tool showed broader application potential. The tool can potentially expand the application to address other environmental challenges. For instance, the tool can be adopted for air quality monitoring, traffic management, energy consumption, and urban green space management. Stakeholders also mentioned this during the workshop in the municipality of Wuppertal.

Additionally, the integration of sensor data for temperature provided the model with up-to-date information, making the simulation more accurate. Accessing the real-time data helps the model reflect the current situation and then predict the future state. This might help planners with more reliable data and information to evaluate the impact of different planning scenarios.

Integrating the RF ML model in DT-PSS allowed the tool to analyse and forecast LST and UHI. Since the RF model is trained based on past data from environmental and urban factors, the trained RF model in this study helped with more accurate temperature forecasting. Moreover, the relationship between variables that drive temperature and UHI is complex, and the trained RF model helped to find the relationship between variables and amplify the predictive capabilities of the model. Thus, integrating the ML in DTs is essential in creating a DT for UHI.

Including uncontrollable factors related to climate changes, such as future temperature trends or extreme weather scenarios in the model, might enhance the prediction and accuracy of the model. This is also mentioned by stakeholders during the municipality workshop (refer to section 6.6, stakeholders). Although planners have no control over this variable, including them in the model increases the prediction performance. This is because planners evaluate the urban planning scenarios and select the best one that might tackle climate change scenarios.

Standardizing data formats is also essential for creating the DT tool, as it requires integrating various data types from different sources. Converting data format to a standard and compatible format is a fundamental step for integrating data in DT. Additionally, data should be cleaned and filtered to ensure that there is no error. Also, temporal and spatial data should be synchronized to ensure consistency between the data.

Last but not least, the created DT tool in this study is a good example of DT practical application. While the study made a small prototype of the DT tool for a neighbourhood in Wuppertal City due to the computational constraints, the methodology showed that the DT-PSS model is applicable to the whole city.

7.1. Limitations

Limited computational resources are one of the main challenges of this study. Due to the limited computational resources, the DT-PSS tool is made only for a small neighbourhood in Wuppertal City. This limitation prevents the demonstration of the full potential of the tool for the whole of Wuppertal City.

In addition to computational constraints, data availability might impact the results of this study. The real-time data in this study is imported from four sensors in the study area. That is why IDW interpolation is used to calculate the real-time temperature of the Südstadt neighbourhood. The more sensors for real-time data would provide more accurate simulation. Additionally, the past data for wind and the data to calculate SVF for the whole city were unavailable. Furthermore, privacy concerns resulted in low-resolution data for population density. The resolution of population density used in this study is low due to privacy, which might affect the result.

Another significant issue is interoperability between different software tools. Interoperability between GIS software and UE is another challenge that limited the functionality of the DT tool. For defining new scenarios in UE, several variables, such as PD and ED, are calculated in another software (Fragstat) before being brought into UE. Thus, interoperability between different software for calculating variables in real time was a major issue in this study.

Choosing suitable ML models also presented a problem. The RF model showed the highest accuracy in this study compared to the other three used models: SVM, ANNs, and the Polynomial regression model. However, this might be due to one of the input variables, land use. This is because the land use data was categorical, and SVM, ANNs, and Polynomial regression models are more proficient with numerical data. Therefore, selecting the models that can handle categorical data is another limitation of this study that could not be solved in the time scope of this research.

Finally, there was a limitation in defining scenarios in UE for the DT assessment. This limitation is due to the limited time and knowledge of working with UE. The time limitation was a significant constraint in exploring the abilities of the UE platform. Hence, more scenarios, such as ongoing urban planning plans in Wuppertal city, could be considered and included in the DT tool.

7.2. Further Research

- **Inclusion of Additional Variables**: From the literature review, more variables are identified that statistically have a significant correlation with temperature and UHI. However, due to the data limitation, only the selected variables (Table 1) are included in this research. Other variables, such as wind and SVF, are essential for predicting temperature and should be considered in future research.
- **Accuracy of ML Models**: In this study, four ML models are trained, and the RF model showed the highest accuracy compared to the other three models. However, further research can increase the accuracy of ML models. For example, decay rate and iteration in ANNs, tuning hyper-parameter, and choosing kernels in SVM are important to achieve more accuracy for ML models. More advanced algorithms can also be used for future research. Moreover, the data for this study was collected between 2017 and 2023. For future studies, the data can be collected for a longer period to include more past data in training the models. Furthermore, this study also used categorical data to train ML models, which might result in the lowest accuracy of ML models. Therefore, future studies can explore which ML algorithms are more compatible with the categorical data to increase the accuracy of the ML model for UHI/LST prediction.
- **Visualisation**: The user interface created for the tool was simple and had limitations. Moreover, the UI was created using blueprint scripting due to the limited time and skills in C++ programming. In future research, improving the UI and providing more user capabilities, functions, and scenarios might be possible. More functions can be developed to increase the interactivity of the DT tool.
- **Developing a Mobile Version**: The UE platform is a powerful game engine that allows mobile platform development for iOS and Android. Therefore, using UE, it is possible to create a mobile version of the DT tool. This might enhance collaboration and public engagement. Additionally, it

provides residents with smooth access to the platform, and they can give their feedback regarding urban planning scenarios using the platform.

- **Integrating Private Data**: Due to concerns about privacy, data such as population density cannot be integrated into DTs, or at least they can be included after reducing the spatial resolution. This might directly affect the DT performance. Consequently, this raises a problem in using private data as real-time data in DT, which needs to be addressed by the data governance framework and further research.

8. ETHICAL CONSIDERATIONS

Privacy concerns are carefully considered during the research process to protect private information. No phase of the research process involves the sharing or release of any personal data. The primary data of this study is obtained from reliable, publicly accessible databases to ensure that no personal data existed in this research. A survey was also conducted to obtain information needed to create and design the suggested DT-PSS tool for this study. The result of the survey is anonymously reflected in this research to prevent publishing any private data related to participants.

LIST OF REFERENCES

Agam, N., Kustas, W. P., Anderson, M. C., Li, F., & Colaizzi, P. D. (2007). Utility of thermal sharpening over Texas high plains irrigated agricultural fields. *Journal of Geophysical Research Atmospheres*, *112*(19). https://doi.org/10.1029/2007JD008407

Agam, N., Kustas, W. P., Anderson, M. C., Li, F., & Colaizzi, P. D. (2008). Utility of thermal image sharpening for monitoring field-scale evapotranspiration over rainfed and irrigated agricultural regions. *Geophysical Research Letters*, *35*(2). https://doi.org/10.1029/2007GL032195

Ahmed Memon, R., Leung, D. Y., & Chunho, L. (2008). A review on the generation, determination and mitigation of Urban Heat Island. In *Journal of Environmental Sciences* (Vol. 20).

Attaran, M., & Celik, B. G. (2023). Digital Twin: Benefits, use cases, challenges, and opportunities. *Decision Analytics Journal*, *6*. https://doi.org/10.1016/j.dajour.2023.100165

Austin, M., Delgoshaei, P., Coelho, M., & Heidarinejad, M. (2020). Architecting Smart City Digital Twins: Combined Semantic Model and Machine Learning Approach. *Journal of Management in Engineering*, *36*(4). https://doi.org/10.1061/(asce)me.1943-5479.0000774

Avdan, U., & Jovanovska, G. (2016). Algorithm for automated mapping of land surface temperature using LANDSAT 8 satellite data. *Journal of Sensors*, *2016*. https://doi.org/10.1155/2016/1480307

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D city models: State of the art review. In *ISPRS International Journal of Geo-Information* (Vol. 4, Issue 4, pp. 2842–2889). MDPI AG. https://doi.org/10.3390/ijgi4042842

Blocken, B., Janssen, W. D., & van Hooff, T. (2012). CFD simulation for pedestrian wind comfort and wind safety in urban areas: General decision framework and case study for the Eindhoven University campus. *Environmental Modelling and Software*, *30*, 15–34. https://doi.org/10.1016/j.envsoft.2011.11.009

Bonin, O., & Rousseaux, F. (2005). *Digital Terrain Model Computation from Contour Lines: How to Derive Quality Information from Artifact Analysis*.

Botín-Sanabria, D. M., Mihaita, S., Peimbert-García, R. E., Ramírez-Moreno, M. A., Ramírez-Mendoza, R. A., & Lozoya-Santos, J. de J. (2022). Digital Twin Technology Challenges and Applications: A Comprehensive Review. In *Remote Sensing* (Vol. 14, Issue 6). MDPI. https://doi.org/10.3390/rs14061335

Caprari, G., Castelli, G., Montuori, M., Camardelli, M., & Malvezzi, R. (2022). Digital Twin for Urban Planning in the Green Deal Era: A State of the Art and Future Perspectives. *Sustainability (Switzerland)*, *14*(10). https://doi.org/10.3390/su14106263

Chen, M., Andrienko, G., IEEE Computer Society. Technical Committee on Visualization and Graphics, & Institute of Electrical and Electronics Engineers. (n.d.). *2015 IEEE Conference on Visual Analytics Science and Technology : proceedings : Chicago, Illinois, USA, 25-30 October 2015*.

Debbage, N., & Shepherd, J. M. (2015). The urban heat island effect and city contiguity. *Computers, Environment and Urban Systems*, *54*, 181–194. https://doi.org/10.1016/j.compenvurbsys.2015.08.002

Deilami, K., Kamruzzaman, M., & Liu, Y. (2018). Urban heat island effect: A systematic review of spatio-temporal factors, data, methods, and mitigation measures. In *International Journal of Applied Earth Observation and Geoinformation* (Vol. 67, pp. 30–42). Elsevier B.V. https://doi.org/10.1016/j.jag.2017.12.009

Edington, L., Dervilis, N., Ben Abdessalem, A., & Wagg, D. (2023). A time-evolving digital twin tool for engineering dynamics applications. *Mechanical Systems and Signal Processing*, *188*. https://doi.org/10.1016/j.ymssp.2022.109971

Estoque, R. C., Murayama, Y., & Myint, S. W. (2017). Effects of landscape composition and pattern on land surface temperature: An urban heat island study in the megacities of Southeast Asia. *Science of the Total Environment*, *577*, 349–359. https://doi.org/10.1016/j.scitotenv.2016.10.195

Fragstats: Spatial Pattern Analysis Program for Categorical Maps. (n.d.). https://www.fragstats.org/

Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital Twin: Enabling Technologies, Challenges and Open Research. *IEEE Access*, *8*, 108952–108971. https://doi.org/10.1109/ACCESS.2020.2998358

Gago, E. J., Roldan, J., Pacheco-Torres, R., & Ordóñez, J. (2013). The city and urban heat islands: A review of strategies to mitigate adverse effects. In *Renewable and Sustainable Energy Reviews* (Vol. 25, pp. 749–758). https://doi.org/10.1016/j.rser.2013.05.057

Garzón, J., Molina, I., Velasco, J., & Calabia, A. (2021). A remote sensing approach for surface urban heat island modeling in a tropical colombian city using regression analysis and machine learning algorithms. *Remote Sensing*, *13*(21). https://doi.org/10.3390/rs13214256

Helmholz, P., Bulatov, D., Kottler, B., Burton, P., Mancini, F., May, M., Strauß, E., & Hecht, M. (2021). QUANTIFYING the IMPACT of URBAN INFILL on the URBAN HEAT ISLAND EFFECT - A CASE STUDY for AN ALTERNATIVE MEDIUM DENSITY MODEL. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *46*(4/W1-2021), 43–50. https://doi.org/10.5194/isprs-archives-XLVI-4-W1-2021-43-2021

Hou, H., Longyang, Q., Su, H., Zeng, R., Xu, T., & Wang, Z. H. (2023). Prioritizing environmental determinants of urban heat islands: A machine learning study for major cities in China. In *International Journal of Applied Earth Observation and Geoinformation* (Vol. 122). Elsevier B.V. https://doi.org/10.1016/j.jag.2023.103411

Howell, C. R., Su, W., Nassel, A. F., Agne, A. A., & Cherrington, A. L. (2020). Area based stratified random sampling using geospatial technology in a community-based survey. *BMC Public Health*, *20*(1). https://doi.org/10.1186/s12889-020-09793-0

Hu, Y., Dai, Z., & Guldmann, J. M. (2020). Modeling the impact of 2D/3D urban indicators on the urban heat island over different seasons: A boosted regression tree approach. *Journal of Environmental Management*, *266*. https://doi.org/10.1016/j.jenvman.2020.110424

Huang, X., & Wang, Y. (2019). Investigating the effects of 3D urban morphology on the surface urban heat island effect in urban functional zones by using high-resolution remote sensing data: A case study of Wuhan, Central China. *ISPRS Journal of Photogrammetry and Remote Sensing*, *152*, 119–131. https://doi.org/10.1016/j.isprsjprs.2019.04.010

Jiang, F., Ma, L., Broyd, T., Chen, W., & Luo, H. (2022). Digital twin enabled sustainable urban road planning. *Sustainable Cities and Society*, *78*. https://doi.org/10.1016/j.scs.2021.103645

Jing, W., Yang, Y., Yue, X., & Zhao, X. (2016). A spatial downscaling algorithm for satellite-based precipitation over the Tibetan plateau based on NDVI, DEM, and land surface temperature. *Remote Sensing*, *8*(8). https://doi.org/10.3390/rs8080655

Khajavi, S. H., Motlagh, N. H., Jaribion, A., Werner, L. C., & Holmstrom, J. (2019). Digital Twin: Vision, benefits, boundaries, and creation for buildings. *IEEE Access*, *7*, 147406–147419. https://doi.org/10.1109/ACCESS.2019.2946515

Kikuchi, N., Fukuda, T., & Yabuki, N. (2022). Future landscape visualization using a city digital twin: integration of augmented reality and drones with implementation of 3D model-based occlusion handling. *Journal of Computational Design and Engineering*, *9*(2), 837–856. https://doi.org/10.1093/jcde/qwac032

Kolokotroni, M., Giannitsaris, I., & Watkins, R. (2006). The effect of the London urban heat island on building summer cooling demand and night ventilation strategies. *Solar Energy*, *80*(4), 383–392. https://doi.org/10.1016/j.solener.2005.03.010

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, *51*(11), 1016–1022. https://doi.org/10.1016/j.ifacol.2018.08.474

Kustas, W. P., Norman, J. M., Anderson, M. C., & French, A. N. (2003). Estimating subpixel surface temperatures and energy fluxes from the vegetation index-radiometric temperature relationship. *Remote Sensing of Environment*, *85*(4), 429–440. https://doi.org/10.1016/S0034-4257(03)00036-1

Li, W., Ni, L., Li, Z. L., Duan, S. B., & Wu, H. (2019). Evaluation of machine learning algorithms in spatial downscaling of modis land surface temperature. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *12*(7), 2299–2307. https://doi.org/10.1109/JSTARS.2019.2896923

Macchi, M., Roda, I., Negri, E., & Fumagalli, L. (2018). Exploring the role of Digital Twin for Asset Lifecycle Management. *IFAC-PapersOnLine*, *51*(11), 790–795. https://doi.org/10.1016/j.ifacol.2018.08.415

McCartney, S.; Mehta, A.; Shandas, V.; Gallo, K.; Paris, G.; Nix, S.; Quintero, T.; Tomlinson, A.; Xian, G. (2020). Satellite Remote Sensing for Urban Heat Islands. NASA Applied Remote Sensing Training Program (ARSET). https://appliedsciences.nasa.gov/join-mission/training/english/arset-satellite-remote-sensing-urban-heat-islands

Municipality of Wuppertal. (2024). *Hitzebelastungskarte*. Retrieved 2024, from https://www.wuppertal.de/microsite/geoportal/topicmaps/contentseiten/hitzebelastungskarte.php

Mohajerani, A., Bakaric, J., & Jeffrey-Bailey, T. (2017). The urban heat island effect, its causes, and mitigation, with reference to the thermal properties of asphalt concrete. In *Journal of Environmental Management* (Vol. 197, pp. 522–538). Academic Press. https://doi.org/10.1016/j.jenvman.2017.03.095

Mutani, G., & Todeschi, V. (2020). The effects of green roofs on outdoor thermal comfort, urban heat island mitigation and energy savings. *Atmosphere*, *11*(2). https://doi.org/10.3390/atmos11020123

Nica, E., Popescu, G. H., Poliak, M., Kliestik, T., & Sabie, O. M. (2023). Digital Twin Simulation Tools, Spatial Cognition Algorithms, and Multi-Sensor Fusion Technology in Sustainable Urban Governance Networks. *Mathematics*, *11*(9). https://doi.org/10.3390/math11091981

Onile, A. E., Machlev, R., Petlenkov, E., Levron, Y., & Belikov, J. (2021). Uses of the digital twins concept for energy services, intelligent recommendation systems, and demand side management: A review. In *Energy Reports* (Vol. 7, pp. 997–1015). Elsevier Ltd. https://doi.org/10.1016/j.egyr.2021.01.090

Ozelkan, E., Bagis, S., Ozelkan, E. C., Ustundag, B. B., Yucel, M., & Ormeci, C. (2015). Spatial interpolation of climatic variables using land surface temperature and modified inverse distance weighting. *International Journal of Remote Sensing*, *36*(4), 1000–1025. https://doi.org/10.1080/01431161.2015.1007248

Rasheed, A., San, O., & Kvamsdal, T. (2020). Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access*, *8*, 21980–22012. https://doi.org/10.1109/ACCESS.2020.2970143

Ravanelli, R., Nascetti, A., Cirigliano, R. V., Di Rico, C., Leuzzi, G., Monti, P., & Crespi, M. (2018). Monitoring the impact of land cover change on surface urban heat island through Google Earth Engine: Proposal of a global methodology, first applications and problems. *Remote Sensing*, *10*(9). https://doi.org/10.3390/rs10091488

Romero Rodríguez, L., Duminil, E., Sánchez Ramos, J., & Eicker, U. (2017). Assessment of the photovoltaic potential at urban level based on 3D city models: A case study and new methodological approach. *Solar Energy*, *146*, 264–275. https://doi.org/10.1016/j.solener.2017.02.043

Santamouris, M. (2013). Using cool pavements as a mitigation strategy to fight urban heat island - A review of the actual developments. In *Renewable and Sustainable Energy Reviews* (Vol. 26, pp. 224–240). https://doi.org/10.1016/j.rser.2013.05.047

Santamouris, M. (2015). Analyzing the heat island magnitude and characteristics in one hundred Asian and Australian cities and regions. *Science of the Total Environment*, *512–513*, 582–598. https://doi.org/10.1016/j.scitotenv.2015.01.060

Municipality of Wuppertal. (2020). *Klimaschutzkonzept mit integriertem Handlungsfeld Klimafolgenanpassung.* Gertec GmbH Ingenieurgesellschaft, EPC – Projektgesellschaft für Klima. Nachhaltigkeit. Kommunikation. mbH, K. PLAN - Klima.Umwelt & Planung GmbH. Auftraggeber: Stadt Wuppertal. Retrieved from https://static.werdenktwas.de/domain/145/fs/VO_0549_20_Langfassung_Klimaschutzkonzept1.pdf.

Schrotter, G., & Hürzeler, C. (2020). The Digital Twin of the City of Zurich for Urban Planning. *PFG - Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, *88*(1), 99–112. https://doi.org/10.1007/s41064-020-00092-2

Shahat, E., Hyun, C. T., & Yeom, C. (2021). City digital twin potentials: A review and research agenda. In *Sustainability (Switzerland)* (Vol. 13, Issue 6). MDPI AG. https://doi.org/10.3390/su13063386

Steinrücke, M., Ahlemann, D., & Schrödter, S. (2019). *Gutachten Hitze in der Stadt.* K. PLAN Klima.Umwelt&Planung GmbH. Auftraggeber: Stadt Wuppertal. Retrieved from: https://www.wuppertal.de/microsite/klimaschutz/dokumente_downloads/Gutachten_Hitze-in-der-Stadt.pdf.

Stone, B., Hess, J. J., & Frumkin, H. (2010). Urban form and extreme heat events: Are sprawling cities more vulnerable to climate change than compact cities? *Environmental Health Perspectives*, *118*(10), 1425–1428. https://doi.org/10.1289/ehp.0901879

Strauss, E., & Bulatov, D. (2022). A Region-Based Machine Learning Approach for Self-Diagnosis of A 4D Digital Thermal Twin. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *10*(4/W2-2022), 265–272. https://doi.org/10.5194/isprs-annals-X-4-W2-2022-265-2022

Tashakkori, H., Rajabifard, A., & Kalantari, M. (2015). A new 3D indoor/outdoor spatial model for indoor emergency response facilitation. *Building and Environment*, *89*, 170–182. https://doi.org/10.1016/j.buildenv.2015.02.036

United Nations. (2022). Envisaging the Future of Cities. https://unhabitat.org/sites/default/files/2022/06/wcr_2022.pdfEuropean Space Agency. (n.d.). *Copernicus Open Access Hub.* https://scihub.copernicus.eu/

VanDerHorn, E., & Mahadevan, S. (2021). Digital Twin: Generalization, characterization and implementation. *Decision Support Systems*, *145*. https://doi.org/10.1016/j.dss.2021.113524

White, G., Zink, A., Codecá, L., & Clarke, S. (2021). A digital twin smart city for citizen feedback. *Cities*, *110*. https://doi.org/10.1016/j.cities.2020.103064

Wu, H., & Li, W. (2019). Downscaling Land Surface Temperatures Using a Random Forest Regression Model with Multitype Predictor Variables. *IEEE Access*, *7*, 21904–21916. https://doi.org/10.1109/ACCESS.2019.2896241

Wu, H., Ye, L. P., Shi, W. Z., & Clarke, K. C. (2014). Assessing the effects of land use spatial structure on urban heatislands using HJ-1B remote sensing imagery in Wuhan, China. *International Journal of Applied Earth Observation and Geoinformation*, *32*(1), 67–78. https://doi.org/10.1016/j.jag.2014.03.019

Yang, Y., Cao, C., Pan, X., Li, X., & Zhu, X. (2017). Downscaling land surface temperature in an arid area by using multiple remote sensingindices with random forest regression. *Remote Sensing*, *9*(8). https://doi.org/10.3390/rs9080789

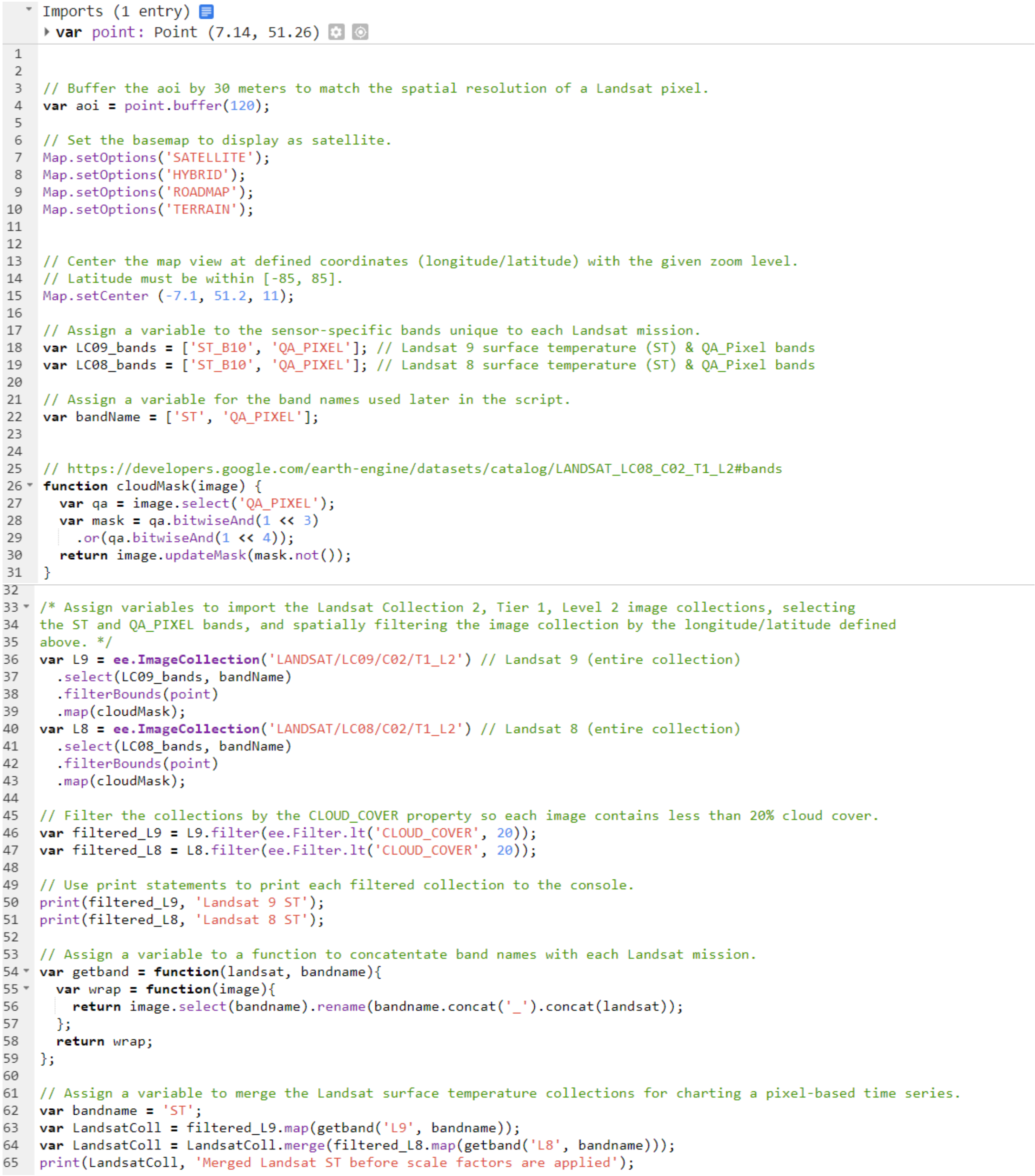
Zhang, H., Qi, Z. fang, Ye, X. yue, Cai, Y. bin, Ma, W. chun, & Chen, M. nan. (2013). Analysis of land use/land cover change, population shift, and their effects on spatiotemporal patterns of urban heat islands in metropolitan Shanghai, China. *Applied Geography*, *44*, 121–133. https://doi.org/10.1016/j.apgeog.2013.07.021

Zhang, K., Chen, H., Dai, H. N., Liu, H., & Lin, Z. (2022). SpoVis: Decision Support System for Site Selection of Sports Facilities in Digital Twinning Cities. *IEEE Transactions on Industrial Informatics*, *18*(2), 1424–1434. https://doi.org/10.1109/TII.2021.3089330

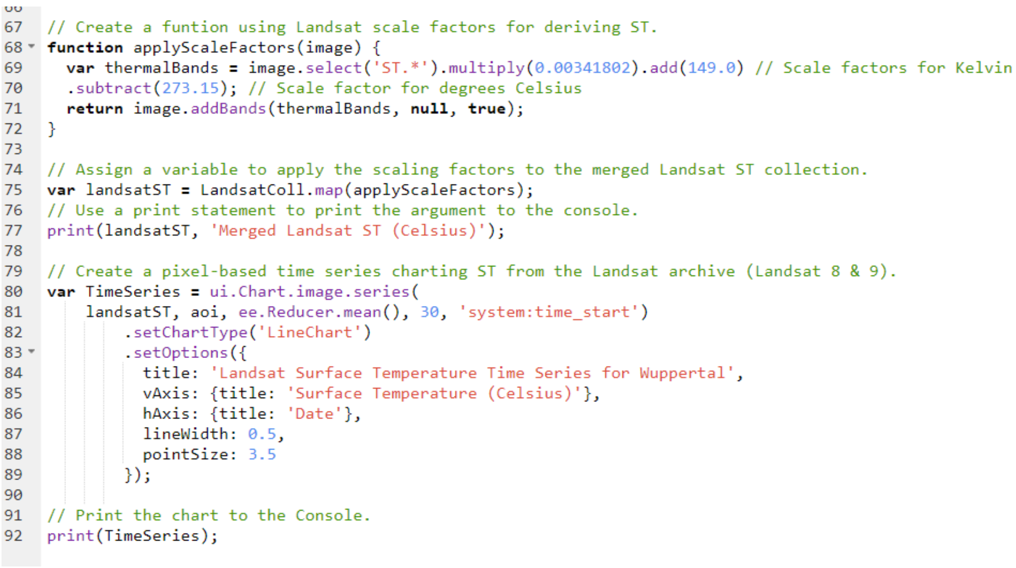
9. ANNEXES

Annex A

In this Annex, the GEE code is used to calculate the mean LST value for urban and rural areas in Wuppertal City from 2013 to 2024 is provided (Figure 54). This code calculates the mean LST within the defined polygons in GEE. Before calculating mean LST, several polygons are defined as rural and urban areas within the boundary of Wuppertal (Figure 54).



```
Imports (1 entry)
 var point: Point (7.14, 51.26)


// Buffer the aoi by 30 meters to match the spatial resolution of a Landsat pixel.
var aoi = point.buffer(120);

// Set the basemap to display as satellite.
Map.setOptions('SATELLITE');
Map.setOptions('HYBRID');
Map.setOptions('ROADMAP');
Map.setOptions('TERRAIN');


// Center the map view at defined coordinates (longitude/latitude) with the given zoom level.
// Latitude must be within [-85, 85].
Map.setCenter (-7.1, 51.2, 11);

// Assign a variable to the sensor-specific bands unique to each Landsat mission.
var LC09_bands = ['ST_B10', 'QA_PIXEL']; // Landsat 9 surface temperature (ST) & QA_Pixel bands
var LC08_bands = ['ST_B10', 'QA_PIXEL']; // Landsat 8 surface temperature (ST) & QA_Pixel bands

// Assign a variable for the band names used later in the script.
var bandName = ['ST', 'QA_PIXEL'];


// https://developers.google.com/earth-engine/datasets/catalog/LANDSAT_LC08_C02_T1_L2#bands
function cloudMask(image) {
  var qa = image.select('QA_PIXEL');
  var mask = qa.bitwiseAnd(1 << 3)
    .or(qa.bitwiseAnd(1 << 4));
  return image.updateMask(mask.not());
}

/* Assign variables to import the Landsat Collection 2, Tier 1, Level 2 image collections, selecting
the ST and QA_PIXEL bands, and spatially filtering the image collection by the longitude/latitude defined
above. */
var L9 = ee.ImageCollection('LANDSAT/LC09/C02/T1_L2') // Landsat 9 (entire collection)
  .select(LC09_bands, bandName)
  .filterBounds(point)
  .map(cloudMask);
var L8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2') // Landsat 8 (entire collection)
  .select(LC08_bands, bandName)
  .filterBounds(point)
  .map(cloudMask);

// Filter the collections by the CLOUD_COVER property so each image contains less than 20% cloud cover.
var filtered_L9 = L9.filter(ee.Filter.lt('CLOUD_COVER', 20));
var filtered_L8 = L8.filter(ee.Filter.lt('CLOUD_COVER', 20));

// Use print statements to print each filtered collection to the console.
print(filtered_L9, 'Landsat 9 ST');
print(filtered_L8, 'Landsat 8 ST');

// Assign a variable to a function to concatentate band names with each Landsat mission.
var getband = function(landsat, bandname){
  var wrap = function(image){
    return image.select(bandname).rename(bandname.concat('_').concat(landsat));
  };
  return wrap;
};

// Assign a variable to merge the Landsat surface temperature collections for charting a pixel-based time series.
var bandname = 'ST';
var LandsatColl = filtered_L9.map(getband('L9', bandname));
var LandsatColl = LandsatColl.merge(filtered_L8.map(getband('L8', bandname)));
print(LandsatColl, 'Merged Landsat ST before scale factors are applied');
```

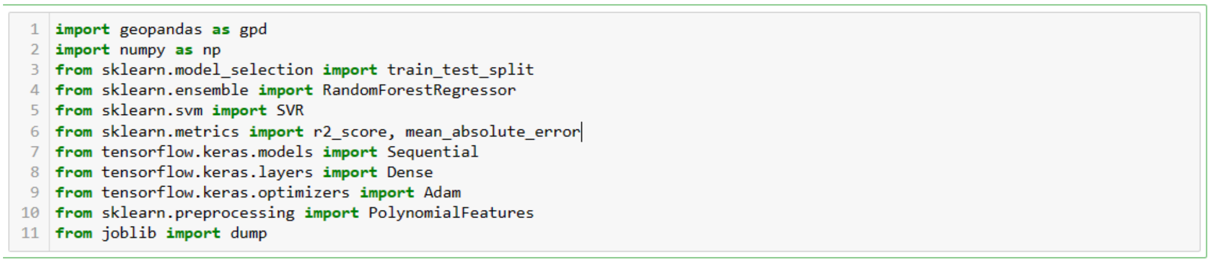


```
// Create a funtion using Landsat scale factors for deriving ST.
function applyScaleFactors(image) {
  var thermalBands = image.select('ST.*').multiply(0.00341802).add(149.0) // Scale factors for Kelvin
  .subtract(273.15); // Scale factor for degrees Celsius
  return image.addBands(thermalBands, null, true);
}

// Assign a variable to apply the scaling factors to the merged Landsat ST collection.
var landsatST = LandsatColl.map(applyScaleFactors);
// Use a print statement to print the argument to the console.
print(landsatST, 'Merged Landsat ST (Celsius)');

// Create a pixel-based time series charting ST from the Landsat archive (Landsat 8 & 9).
var TimeSeries = ui.Chart.image.series(
    landsatST, aoi, ee.Reducer.mean(), 30, 'system:time_start')
        .setChartType('LineChart')
        .setOptions({
          title: 'Landsat Surface Temperature Time Series for Wuppertal',
          vAxis: {title: 'Surface Temperature (Celsius)'},
          hAxis: {title: 'Date'},
          lineWidth: 0.5,
          pointSize: 3.5
        });

// Print the chart to the Console.
print(TimeSeries);
```

Figure 54: GEE code for calculating mean LST. Source: McCartney et al. (2020).

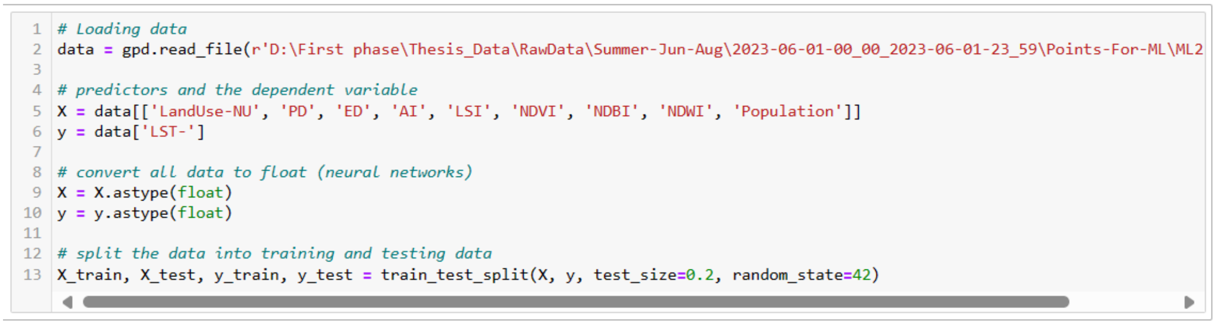
Annex B

In this Annex, python code for training and checking the accuracy of the ML models is provided. Figure 55 shows the necessary modules for training ML algorithms in this study. Figure 56 shows importing data and splitting it into two sets for training and testing data.



```
import geopandas as gpd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_absolute_error
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import PolynomialFeatures
from joblib import dump
```

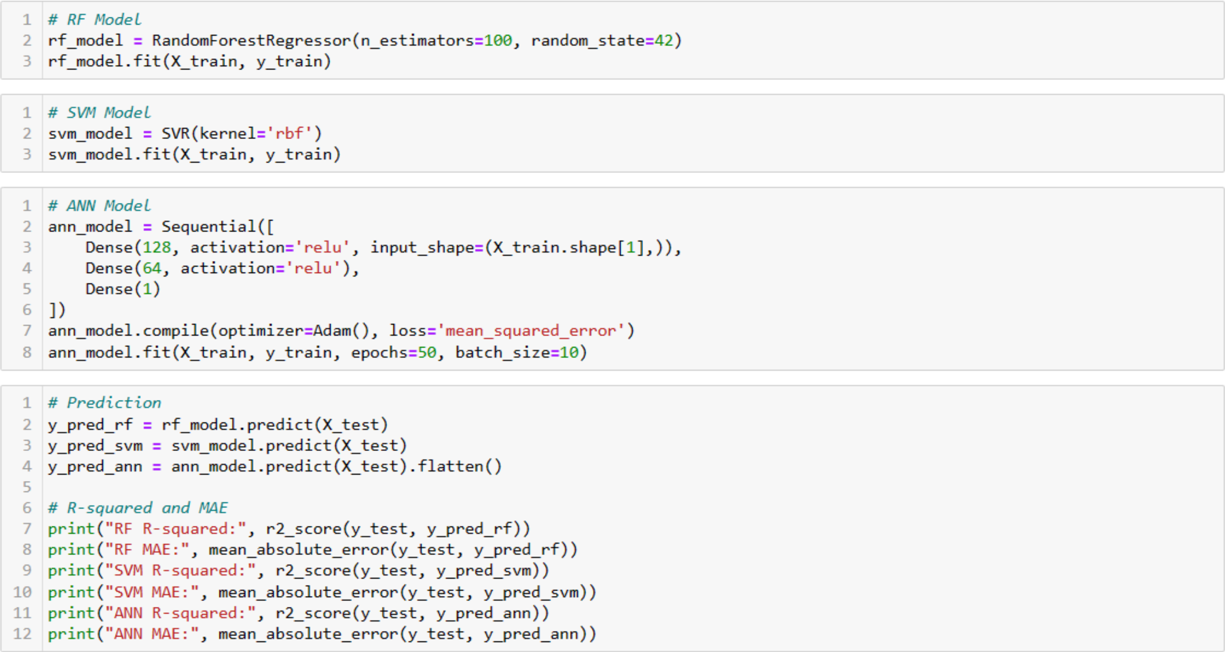
Figure 55: Importing necessary modules for the whole training ML algorithm process. Source: Author, 2024.



```
# Loading data
data = gpd.read_file(r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\Points-For-ML\ML2

# predictors and the dependent variable
X = data[['LandUse-NU', 'PD', 'ED', 'AI', 'LSI', 'NDVI', 'NDBI', 'NDWI', 'Population']]
y = data['LST-']

# convert all data to float (neural networks)
X = X.astype(float)
y = y.astype(float)

# split the data into training and testing data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 56: Loading data and splitting the data into training and test sets. Source: Author, 2024.

The code below is for training three regression algorithms of RF, SVM, ANNs, and the Polynomial regression model (Figure 57). After checking the accuracy, RF and polynomial models are saved for use in UE for temperature prediction. Figure 58 shows the Python code for saving the models as .Joblib format (Figure 58).



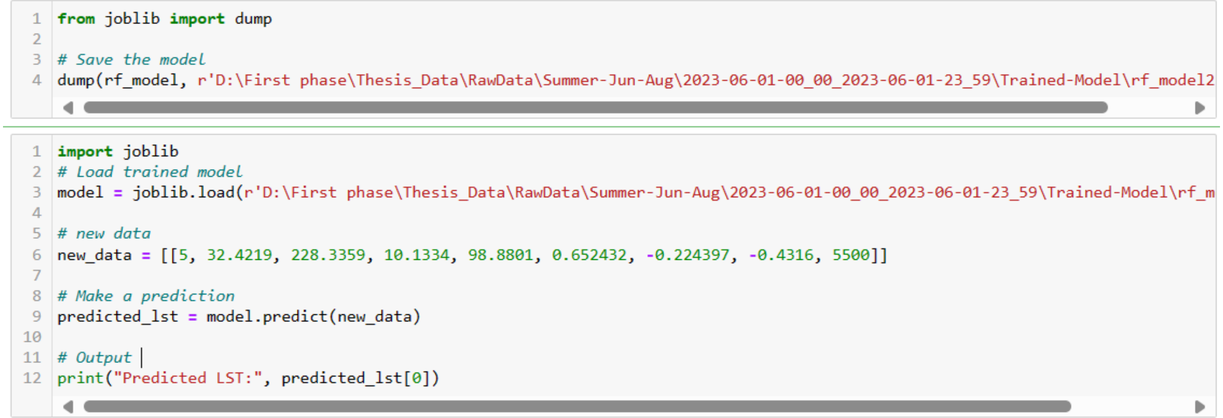
```python
# RF Model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

```python
# SVM Model
svm_model = SVR(kernel='rbf')
svm_model.fit(X_train, y_train)
```

```python
# ANN Model
ann_model = Sequential([
    Dense(128, activation='relu', input_shape=(X_train.shape[1],)),
    Dense(64, activation='relu'),
    Dense(1)
])
ann_model.compile(optimizer=Adam(), loss='mean_squared_error')
ann_model.fit(X_train, y_train, epochs=50, batch_size=10)
```

```python
# Prediction
y_pred_rf = rf_model.predict(X_test)
y_pred_svm = svm_model.predict(X_test)
y_pred_ann = ann_model.predict(X_test).flatten()

# R-squared and MAE
print("RF R-squared:", r2_score(y_test, y_pred_rf))
print("RF MAE:", mean_absolute_error(y_test, y_pred_rf))
print("SVM R-squared:", r2_score(y_test, y_pred_svm))
print("SVM MAE:", mean_absolute_error(y_test, y_pred_svm))
print("ANN R-squared:", r2_score(y_test, y_pred_ann))
print("ANN MAE:", mean_absolute_error(y_test, y_pred_ann))
```

Figure 57: Python code for training RF, SVM, ANNs regression algorithm, and the code for checking the accuracy. Source: Author, 2024.



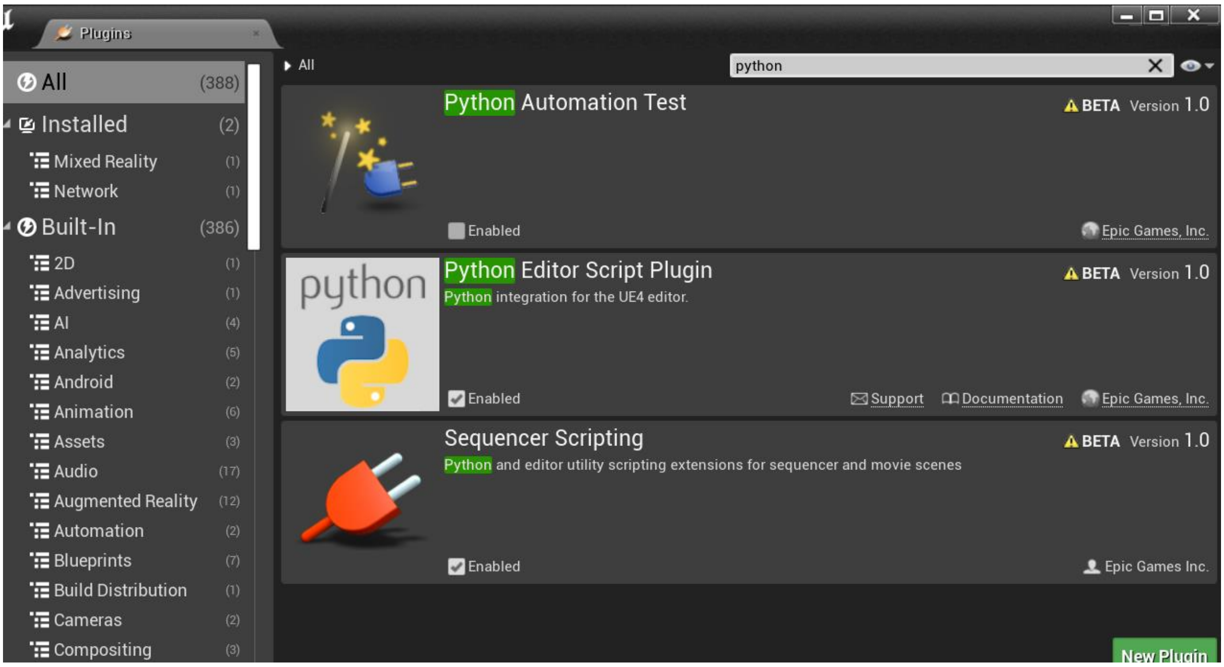
```python
from joblib import dump

# Save the model
dump(rf_model, r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\Trained-Model\rf_model2
```

```python
import joblib
# Load trained model
model = joblib.load(r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\Trained-Model\rf_m

# new data
new_data = [[5, 32.4219, 228.3359, 10.1334, 98.8801, 0.652432, -0.224397, -0.4316, 5500]]

# Make a prediction
predicted_lst = model.predict(new_data)

# Output
print("Predicted LST:", predicted_lst[0])
```

Figure 58: Saving the model and checking the performance. Source: Author, 2024.

Annex C

There are several plugins for using Python in UE, such as “Python Editor” and “Python Runtime”. A few of them are commercial but have more functionality in UE. This study used only free and non-commercial plugins for Python in UE. Figure 59 shows the free plugins that are used in this study, “Sequencer Scripting” and “Python Editor Script Plugin.”



Python Automation Test — BETA Version 1.0 — ☐ Enabled — Epic Games, Inc.

Python Editor Script Plugin — BETA Version 1.0 — Python integration for the UE4 editor. — ☑ Enabled — Support — Documentation — Epic Games, Inc.

Sequencer Scripting — BETA Version 1.0 — Python and editor utility scripting extensions for sequencer and movie scenes — ☑ Enabled — Epic Games Inc.

Figure 59: Python plugins used in the study. Source: Author, 2024.

Annex D

In this annex, a summary of the survey is provided. Also, a link to the survey is provided. In total, 13 people from the municipality of Wuppertal participated in the survey. Due to the privacy and prevention of publishing private information of respondents, the results related to questions such as respondents' role in the Wuppertal municipality are not provided. the survey results that directly is linked to the design DT toll are provided in the methodology section 5.4. In the following survey, questions and answers are provided.

1. In your opinion, what information is essential to create a 2\3D model of an urban heat island (multiple answers)?

The respondents stated that for creating a 3D model for UHI buildings, land cover/land use, green spaces, wind, temperature, and water bodies are important information. They believe that this information should be included in the UHI 3D city model. Figure 60 show the result of this question.

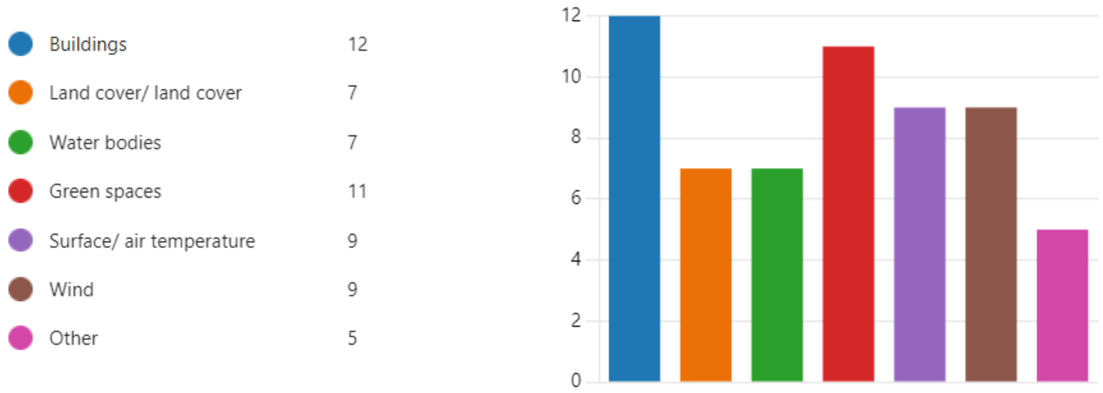


Figure 60: Stakeholders’ opinions regarding the essential information for creating a 3D model for UHI. Source: Author, 2024.

2. What features or functionalities would you like to see incorporated into the Decision Support System tool to make it more useful and user-friendly for your purposes (multiple answers)?
Stakeholders mentioned that data visualisation, scenario modelling, and interactive map are the most important functionalities and features that should be included in the PSS tool for UHI. Figure 61 show the result of this question.

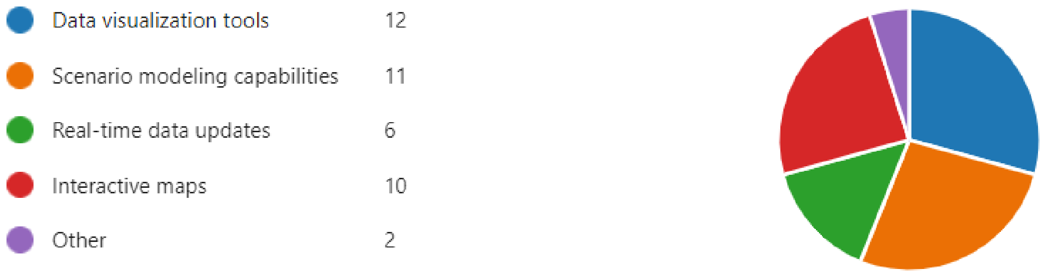


Figure 61: Stakeholders' opinions regarding the important features and functionalities that should be included in the PSS tool. Source: Author, 2024.

1. What level of interactivity (based on your opinion) is helpful for the Decision Support System tool? The level of interactivity allows the users to make changes in the model. The video provided in section one has the highest level of interactivity.

Stakeholders believed that the PSS tool for UHI should have a basic to a high level of interactivity. Figure 62 show the result of this question.

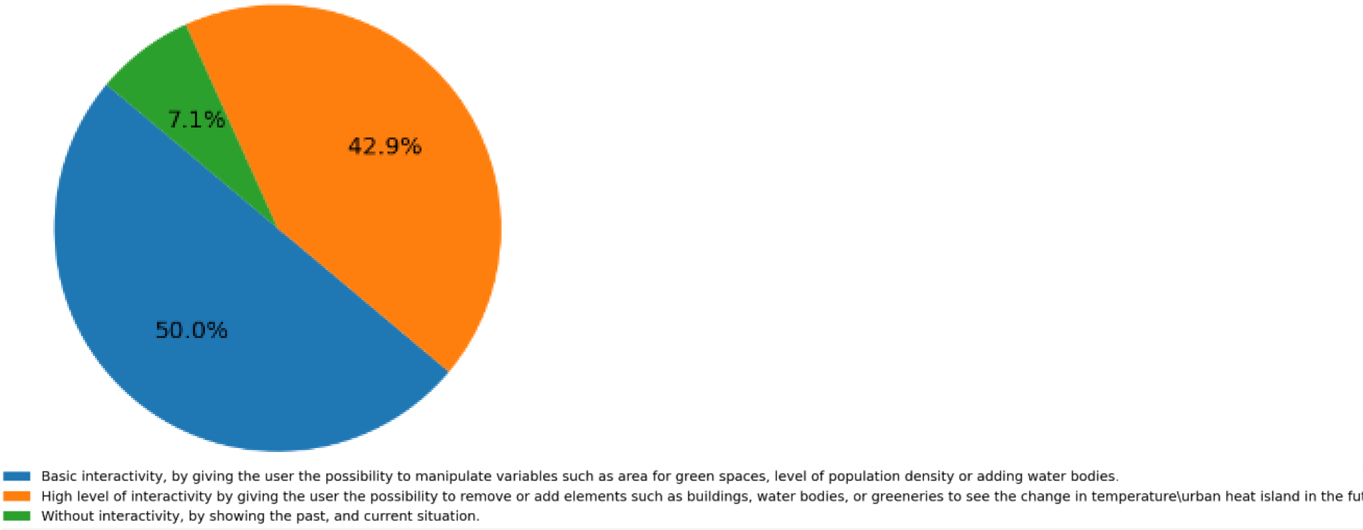


Figure 62: Stakeholders' opinions regarding the level of the interactivity of the PSS tool. Source: Author, 2024.

4. What is the role of a Decision Support System tool in your work or decision-making processes related to urban heat island mitigation?
Stakeholders stated the tool is helpful in their workflow for UHI mitigation planning. Figure 63 show the result of this question.

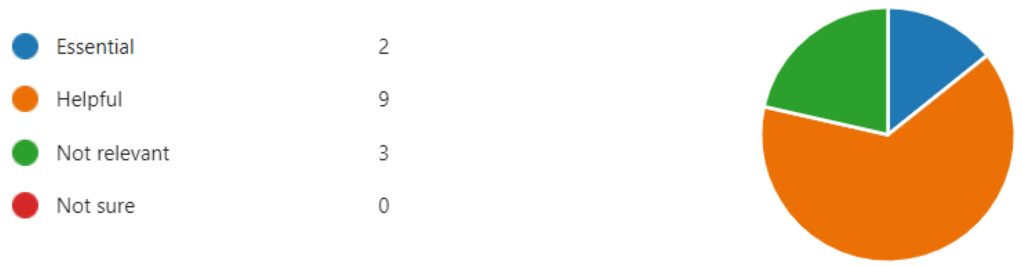


Figure 63: Stakeholders' opinions regarding how the tool can help them in UHI mitigation. Source: Author, 2024.

5. After creating the Decision Support System tool in the future, what will the level of integration of the Decision Support System tool be in the decision-making procedures (in urban planning) in Wuppertal municipality?

Based on stakeholders' opinions, the level of integration of PSS is still unclear in their planning process. More research is needed to integrate PSS into the planning process. Figure 64 shows the result for this question.

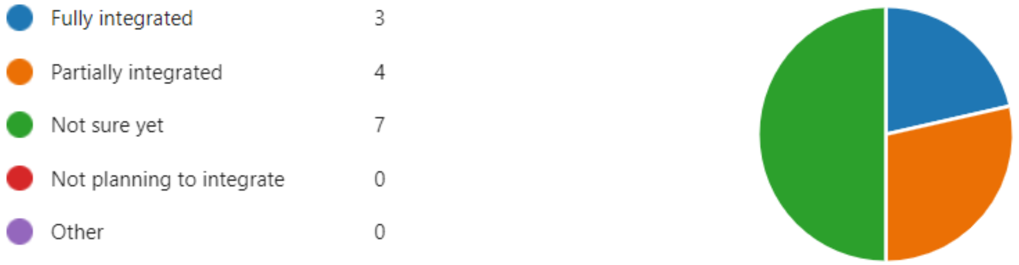


Figure 64: Stakeholders' opinions regarding the level of the interactivity of the PSS tool. Source: Author, 2024.

6. How can the Decision Support System tool be designed to encourage active participation and collaboration among stakeholders and community members (multiple answers)?

From the stakeholders' points of view, designing user-friendly interfaces in the PSS tool and educational resources can increase the participation of stakeholders and community members. Figure 65 show the result of this question.

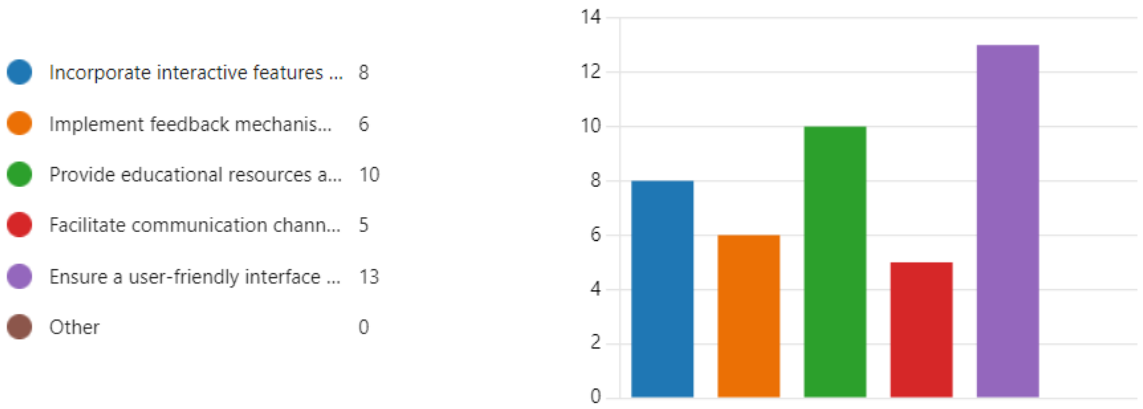
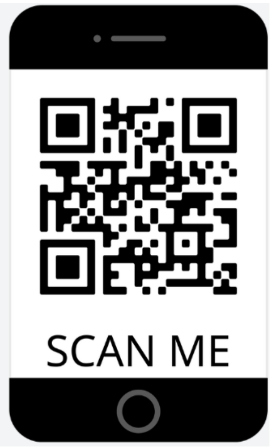


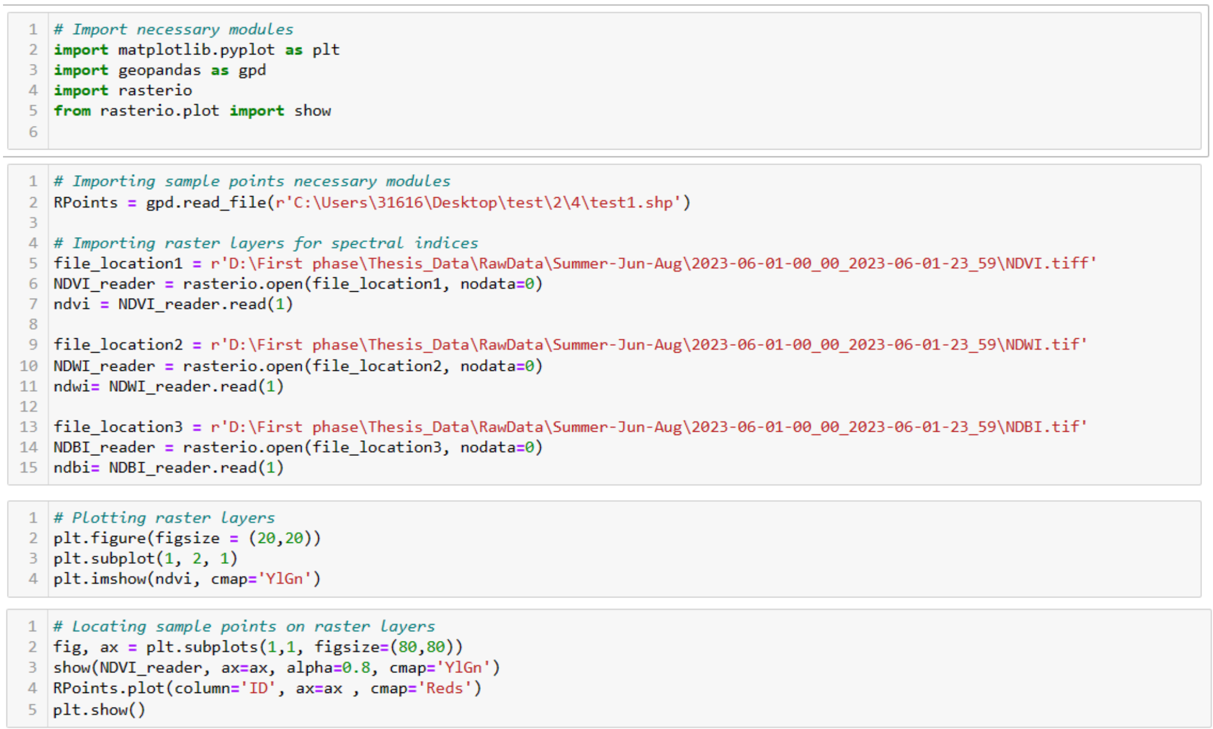
Figure 65: Stakeholders' opinions regarding the level of the interactivity of the PSS tool. Source: Author, 2024.

Here the link for the survey is provided:
https://forms.office.com/Pages/ResponsePage.aspx?id=oUYycvXDxUOs3EOttASsTSqNb0ZO0RlAjs76cUYcwGNUMlE4Szg0RVZUSUxIWkcyWlFLSzdBNFRJSi4u



Annex E

This part provides Python code calculating spectral indices of the ML model in this research. The code below (Figure 66) shows the necessary modules for importing data and calculating spectral indices for each sample point.

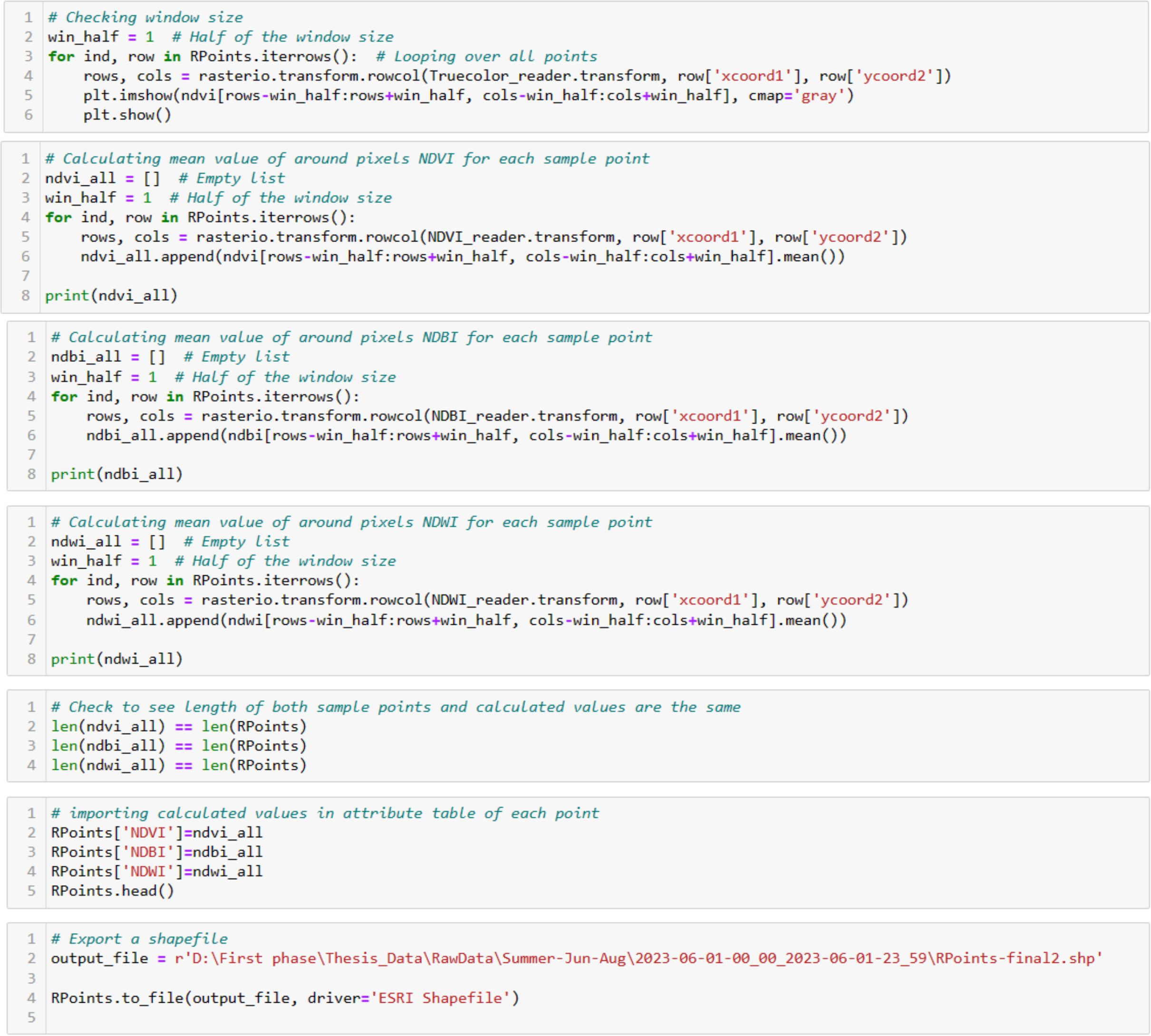


```
# Import necessary modules
import matplotlib.pyplot as plt
import geopandas as gpd
import rasterio
from rasterio.plot import show

```

```
# Importing sample points necessary modules
RPoints = gpd.read_file(r'C:\Users\31616\Desktop\test\2\4\test1.shp')

# Importing raster layers for spectral indices
file_location1 = r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\NDVI.tiff'
NDVI_reader = rasterio.open(file_location1, nodata=0)
ndvi = NDVI_reader.read(1)

file_location2 = r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\NDWI.tif'
NDWI_reader = rasterio.open(file_location2, nodata=0)
ndwi= NDWI_reader.read(1)

file_location3 = r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\NDBI.tif'
NDBI_reader = rasterio.open(file_location3, nodata=0)
ndbi= NDBI_reader.read(1)
```

```
# Plotting raster layers
plt.figure(figsize = (20,20))
plt.subplot(1, 2, 1)
plt.imshow(ndvi, cmap='YlGn')
```

```
# Locating sample points on raster layers
fig, ax = plt.subplots(1,1, figsize=(80,80))
show(NDVI_reader, ax=ax, alpha=0.8, cmap='YlGn')
RPoints.plot(column='ID', ax=ax , cmap='Reds')
plt.show()
```



```python
# Checking window size
win_half = 1  # Half of the window size
for ind, row in RPoints.iterrows():  # Looping over all points
    rows, cols = rasterio.transform.rowcol(Truecolor_reader.transform, row['xcoord1'], row['ycoord2'])
    plt.imshow(ndvi[rows-win_half:rows+win_half, cols-win_half:cols+win_half], cmap='gray')
    plt.show()
```

```python
# Calculating mean value of around pixels NDVI for each sample point
ndvi_all = []  # Empty list
win_half = 1  # Half of the window size
for ind, row in RPoints.iterrows():
    rows, cols = rasterio.transform.rowcol(NDVI_reader.transform, row['xcoord1'], row['ycoord2'])
    ndvi_all.append(ndvi[rows-win_half:rows+win_half, cols-win_half:cols+win_half].mean())

print(ndvi_all)
```

```python
# Calculating mean value of around pixels NDBI for each sample point
ndbi_all = []  # Empty list
win_half = 1  # Half of the window size
for ind, row in RPoints.iterrows():
    rows, cols = rasterio.transform.rowcol(NDBI_reader.transform, row['xcoord1'], row['ycoord2'])
    ndbi_all.append(ndbi[rows-win_half:rows+win_half, cols-win_half:cols+win_half].mean())

print(ndbi_all)
```

```python
# Calculating mean value of around pixels NDWI for each sample point
ndwi_all = []  # Empty list
win_half = 1  # Half of the window size
for ind, row in RPoints.iterrows():
    rows, cols = rasterio.transform.rowcol(NDWI_reader.transform, row['xcoord1'], row['ycoord2'])
    ndwi_all.append(ndwi[rows-win_half:rows+win_half, cols-win_half:cols+win_half].mean())

print(ndwi_all)
```

```python
# Check to see length of both sample points and calculated values are the same
len(ndvi_all) == len(RPoints)
len(ndbi_all) == len(RPoints)
len(ndwi_all) == len(RPoints)
```

```python
# importing calculated values in attribute table of each point
RPoints['NDVI']=ndvi_all
RPoints['NDBI']=ndbi_all
RPoints['NDWI']=ndwi_all
RPoints.head()
```

```python
# Export a shapefile
output_file = r'D:\First phase\Thesis_Data\RawData\Summer-Jun-Aug\2023-06-01-00_00_2023-06-01-23_59\RPoints-final2.shp'

RPoints.to_file(output_file, driver='ESRI Shapefile')

```

Figure 66: Python code for calculating spectral indices for sample points. Source: Author, 2024.

Annex F

Information for each sensor is provided below. The information consists of related API, and Longitude and latitude are provided:

1. BAD – APH Am Diek

Latitude: 51.28579071558207
Longitude: 7.226010517432712
API:https://element-iot.com/api/v1/devices/e10dcd31-db06-484b-9e85-149b9e3ca412/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854

2. EWA – Von-der-Heydt-Museum

Latitude: 51.25632580035739
Longitude: 7.146776594655364
API:https://element-iot.com/api/v1/devices/471beb73-8f31-44b4-8e00070089894ece/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854

3. RBL – An der Blutfinke

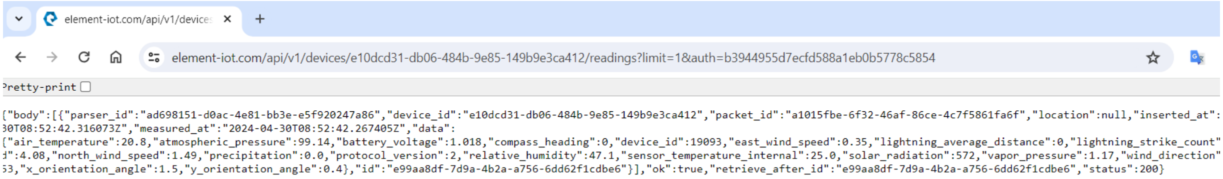
Latitude: 51.225220707271404
Longitude: 7.1899988651351805
API:https://element-iot.com/api/v1/devices/0b2c081e-e0d3-4ea5-a342-c62b815ec4bb/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854

4. ZOO – oka-Wetterstation

Latitude: 51.239055407584544
Longitude: 7.1099243537421035
API:https://element-iot.com/api/v1/devices/4d2d12a3-f8d2-4bfe-aa50-4c31aeaa86f4/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854

Annex G

This Annex provides the visual code for importing real-time temperature data. VaRest plugin is used to import real-time data from sensors. Figure 67 shows the API information for one of the sensors. As shown, the information that is provided is not in a structured way. This issue made it difficult to find which array and object should be extracted by visual code to get temperature data. So, Postman is used to find the structure of code in APIs. Figure 68 shows the JSON code after importing the API in Postman. As it isclear after importing the API in Postman, the temperature is inside the body array and inside the data.

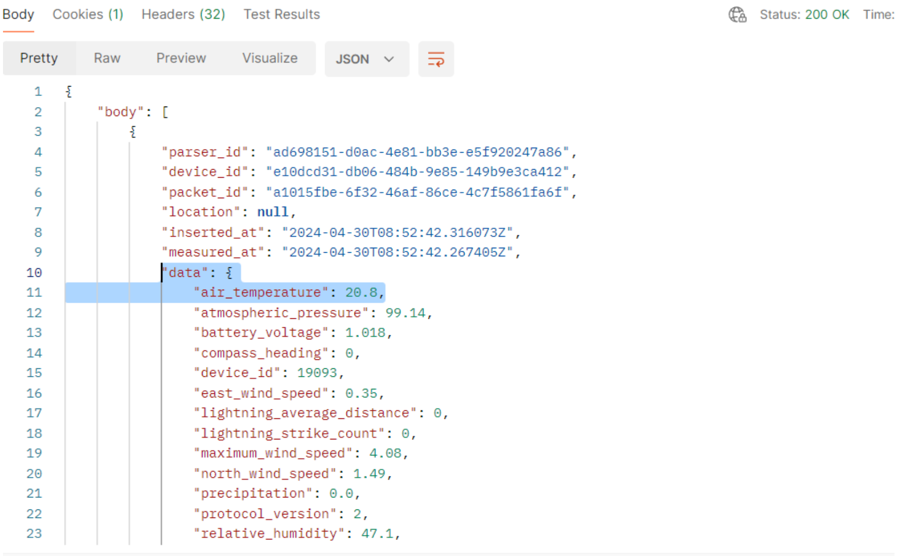


element-iot.com/api/v1/devices
element-iot.com/api/v1/devices/e10dcd31-db06-484b-9e85-149b9e3ca412/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854
Pretty-print □

```
{"body":[{"parser_id":"ad698151-d0ac-4e81-bb3e-e5f920247a86","device_id":"e10dcd31-db06-484b-9e85-149b9e3ca412","packet_id":"a1015fbe-6f32-46af-86ce-4c7f5861fa6f","location":null,"inserted_at":
30T08:52:42.316073Z","measured_at":"2024-04-30T08:52:42.267405Z","data":
{"air_temperature":20.8,"atmospheric_pressure":99.14,"battery_voltage":1.018,"compass_heading":0,"device_id":19093,"east_wind_speed":0.35,"lightning_average_distance":0,"lightning_strike_count"
d":4.08,"north_wind_speed":1.49,"precipitation":0.0,"protocol_version":2,"relative_humidity":47.1,"sensor_temperature_internal":25.0,"solar_radiation":572,"vapor_pressure":1.17,"wind_direction"
53,"x_orientation_angle":1.5,"y_orientation_angle":0.4},"id":"e99aa8df-7d9a-4b2a-a756-6dd62f1cdbe6"}],"ok":true,"retrieve_after_id":"e99aa8df-7d9a-4b2a-a756-6dd62f1cdbe6","status":200}
```

Figure 67: API information for sensors. Source: Author, 2024.

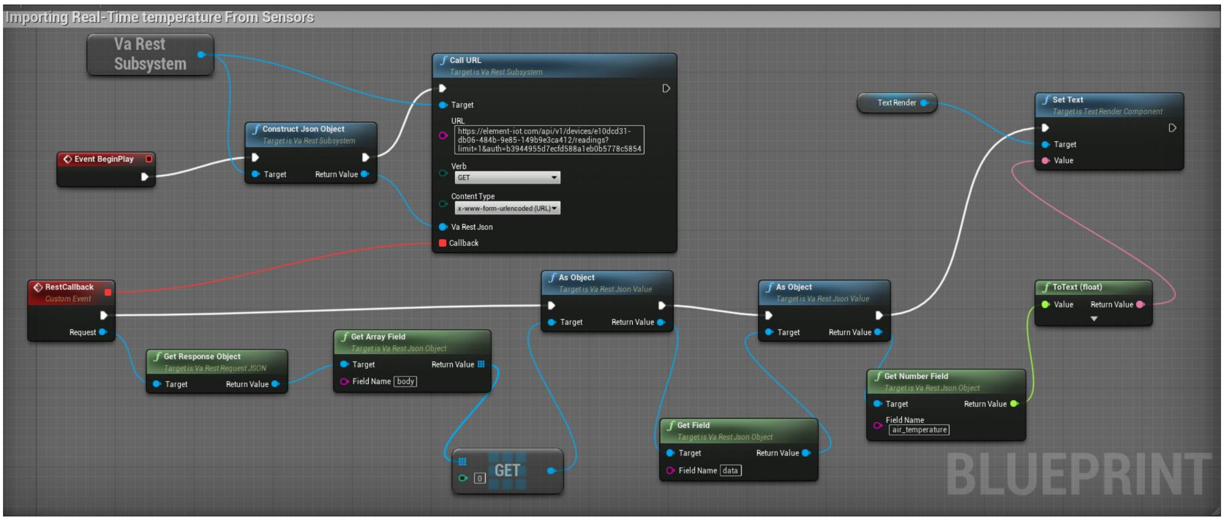
.



```
{
    "body": [
        {
            "parser_id": "ad698151-d0ac-4e81-bb3e-e5f920247a86",
            "device_id": "e10dcd31-db06-484b-9e85-149b9e3ca412",
            "packet_id": "a1015fbe-6f32-46af-86ce-4c7f5861fa6f",
            "location": null,
            "inserted_at": "2024-04-30T08:52:42.316073Z",
            "measured_at": "2024-04-30T08:52:42.267405Z",
            "data": {
                "air_temperature": 20.8,
                "atmospheric_pressure": 99.14,
                "battery_voltage": 1.018,
                "compass_heading": 0,
                "device_id": 19093,
                "east_wind_speed": 0.35,
                "lightning_average_distance": 0,
                "lightning_strike_count": 0,
                "maximum_wind_speed": 4.08,
                "north_wind_speed": 1.49,
                "precipitation": 0.0,
                "protocol_version": 2,
                "relative_humidity": 47.1,
```

Figure 68: API information after importing it in Postman. Source: Author, 2024.

After finding the structure of JSON code for APIs, through Calling URL and Contact JSON Object Functions, API information is imported in UE. After that, the temperature data is extracted and printed on the screen using the Get Array Field and Get Field functions. The output of this step is imported as an input for IDW interpolation. The figure below shows the whole visual code for this part (Figure 69). Event Begin Play is used to run the code. So, as the visualisation starts, the code will get the information from sensors.


URL https://element-iot.com/api/v1/devices/e10dcd31-db06-484b-9e85-149b9e3ca412/readings?limit=1&auth=b3944955d7ecfd588a1eb0b5778c5854

Verb GET

Content Type x-www-form-urlencoded (URL)

Field Name body

Field Name data

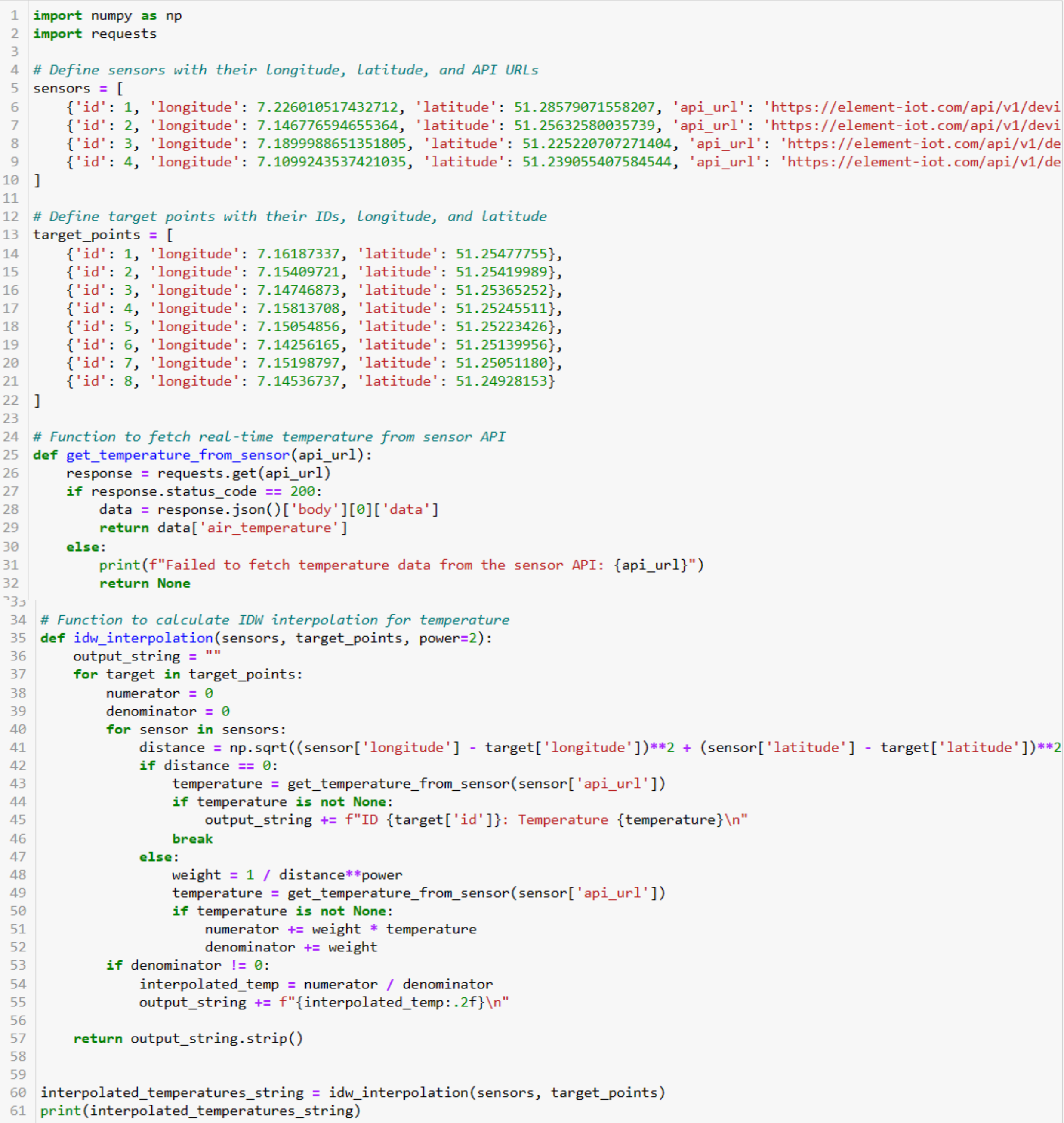
Field Name air_temperature

Importing Real-Time temperature From Sensors
Va Rest Subsystem
Event BeginPlay
Construct Json Object
Target is Va Rest Subsystem
Target
Return Value
Call URL
Target is Va Rest Subsystem
Target
Va Rest Json
Callback
RestCallback
Custom Event
Request
Get Response Object
Target is Va Rest Request JSON
Get Array Field
Target is Va Rest Json Object
GET
0
As Object
Target is Va Rest Json Value
Get Field
Target is Va Rest Json Object
Get Number Field
Target is Va Rest Json Object
Text Render
Set Text
Target is Text Render Component
Value
ToText (float)
BLUEPRINT

Figure 69: Visual code for importing real-time data for temperature form sensors. Source: Author, 2024.

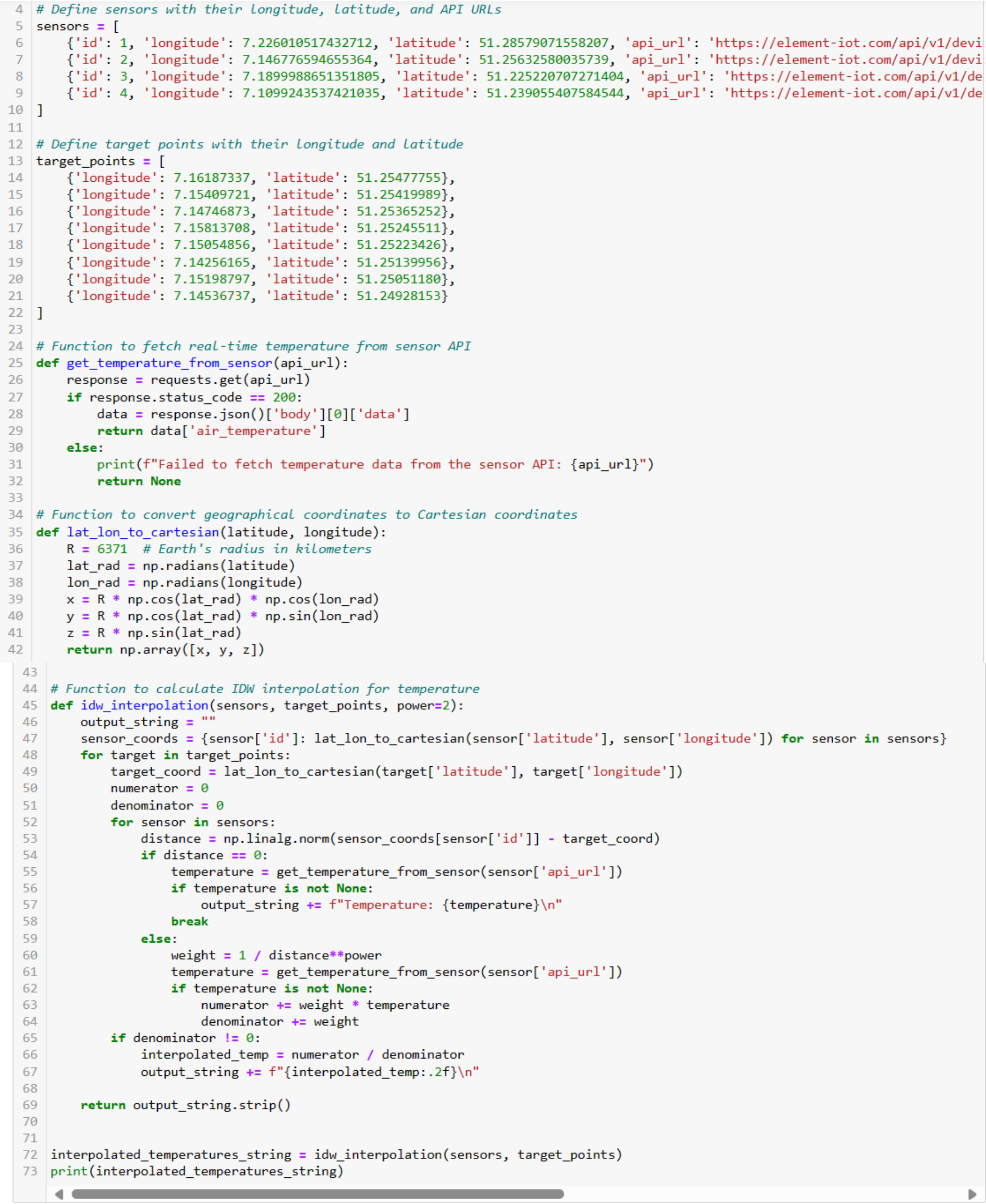
Annex H

In this annex, the Python code for IDW interpolation is provided. The first code is for IDW interpolation using geographical coordinates, and the second code is for IDW interpolation using Cartesian coordinates (Figure 70 and 71). At first, information on each sensor, such as longitude, latitude, API URL, and ID, is imported. Then, the temperature interpolated to each point. The output is in string format and can be used in UE.



```python
import numpy as np
import requests

# Define sensors with their longitude, latitude, and API URLs
sensors = [
    {'id': 1, 'longitude': 7.226010517432712, 'latitude': 51.28579071558207, 'api_url': 'https://element-iot.com/api/v1/devi
    {'id': 2, 'longitude': 7.146776594655364, 'latitude': 51.25632580035739, 'api_url': 'https://element-iot.com/api/v1/devi
    {'id': 3, 'longitude': 7.1899988651351805, 'latitude': 51.225220707271404, 'api_url': 'https://element-iot.com/api/v1/de
    {'id': 4, 'longitude': 7.1099243537421035, 'latitude': 51.239055407584544, 'api_url': 'https://element-iot.com/api/v1/de
]

# Define target points with their IDs, longitude, and latitude
target_points = [
    {'id': 1, 'longitude': 7.16187337, 'latitude': 51.25477755},
    {'id': 2, 'longitude': 7.15409721, 'latitude': 51.25419989},
    {'id': 3, 'longitude': 7.14746873, 'latitude': 51.25365252},
    {'id': 4, 'longitude': 7.15813708, 'latitude': 51.25245511},
    {'id': 5, 'longitude': 7.15054856, 'latitude': 51.25223426},
    {'id': 6, 'longitude': 7.14256165, 'latitude': 51.25139956},
    {'id': 7, 'longitude': 7.15198797, 'latitude': 51.25051180},
    {'id': 8, 'longitude': 7.14536737, 'latitude': 51.24928153}
]

# Function to fetch real-time temperature from sensor API
def get_temperature_from_sensor(api_url):
    response = requests.get(api_url)
    if response.status_code == 200:
        data = response.json()['body'][0]['data']
        return data['air_temperature']
    else:
        print(f"Failed to fetch temperature data from the sensor API: {api_url}")
        return None

# Function to calculate IDW interpolation for temperature
def idw_interpolation(sensors, target_points, power=2):
    output_string = ""
    for target in target_points:
        numerator = 0
        denominator = 0
        for sensor in sensors:
            distance = np.sqrt((sensor['longitude'] - target['longitude'])**2 + (sensor['latitude'] - target['latitude'])**2
            if distance == 0:
                temperature = get_temperature_from_sensor(sensor['api_url'])
                if temperature is not None:
                    output_string += f"ID {target['id']}: Temperature {temperature}\n"
                break
            else:
                weight = 1 / distance**power
                temperature = get_temperature_from_sensor(sensor['api_url'])
                if temperature is not None:
                    numerator += weight * temperature
                    denominator += weight
        if denominator != 0:
            interpolated_temp = numerator / denominator
            output_string += f"{interpolated_temp:.2f}\n"

    return output_string.strip()


interpolated_temperatures_string = idw_interpolation(sensors, target_points)
print(interpolated_temperatures_string)
```

Figure 70: IDW interpolation using geographical coordinates. Source: Author, 2024.

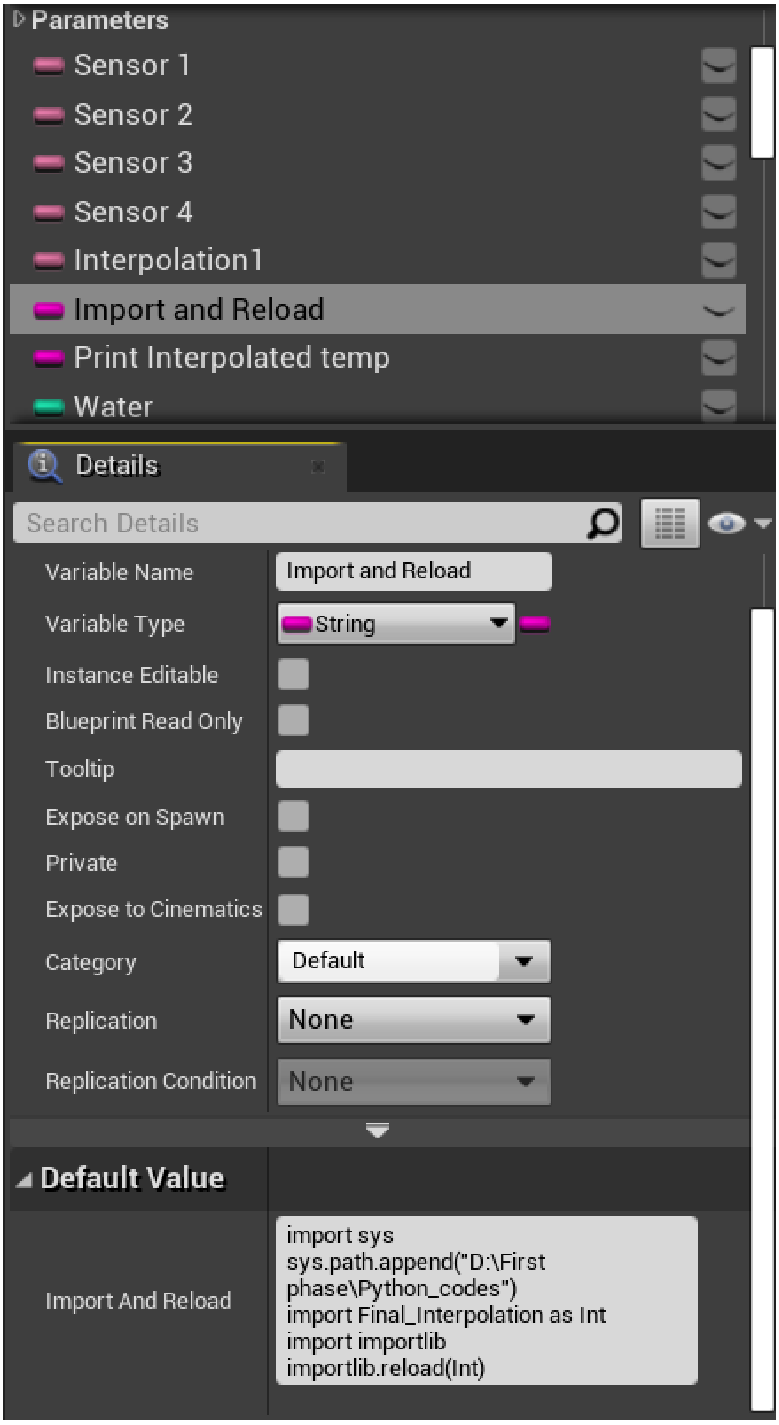


```
# Define sensors with their longitude, latitude, and API URLs
sensors = [
    {'id': 1, 'longitude': 7.226010517432712, 'latitude': 51.28579071558207, 'api_url': 'https://element-iot.com/api/v1/devi
    {'id': 2, 'longitude': 7.146776594655364, 'latitude': 51.25632580035739, 'api_url': 'https://element-iot.com/api/v1/devi
    {'id': 3, 'longitude': 7.1899988651351805, 'latitude': 51.225220707271404, 'api_url': 'https://element-iot.com/api/v1/de
    {'id': 4, 'longitude': 7.1099243537421035, 'latitude': 51.239055407584544, 'api_url': 'https://element-iot.com/api/v1/de
]

# Define target points with their longitude and latitude
target_points = [
    {'longitude': 7.16187337, 'latitude': 51.25477755},
    {'longitude': 7.15409721, 'latitude': 51.25419989},
    {'longitude': 7.14746873, 'latitude': 51.25365252},
    {'longitude': 7.15813708, 'latitude': 51.25245511},
    {'longitude': 7.15054856, 'latitude': 51.25223426},
    {'longitude': 7.14256165, 'latitude': 51.25139956},
    {'longitude': 7.15198797, 'latitude': 51.25051180},
    {'longitude': 7.14536737, 'latitude': 51.24928153}
]

# Function to fetch real-time temperature from sensor API
def get_temperature_from_sensor(api_url):
    response = requests.get(api_url)
    if response.status_code == 200:
        data = response.json()['body'][0]['data']
        return data['air_temperature']
    else:
        print(f"Failed to fetch temperature data from the sensor API: {api_url}")
        return None

# Function to convert geographical coordinates to Cartesian coordinates
def lat_lon_to_cartesian(latitude, longitude):
    R = 6371  # Earth's radius in kilometers
    lat_rad = np.radians(latitude)
    lon_rad = np.radians(longitude)
    x = R * np.cos(lat_rad) * np.cos(lon_rad)
    y = R * np.cos(lat_rad) * np.sin(lon_rad)
    z = R * np.sin(lat_rad)
    return np.array([x, y, z])

# Function to calculate IDW interpolation for temperature
def idw_interpolation(sensors, target_points, power=2):
    output_string = ""
    sensor_coords = {sensor['id']: lat_lon_to_cartesian(sensor['latitude'], sensor['longitude']) for sensor in sensors}
    for target in target_points:
        target_coord = lat_lon_to_cartesian(target['latitude'], target['longitude'])
        numerator = 0
        denominator = 0
        for sensor in sensors:
            distance = np.linalg.norm(sensor_coords[sensor['id']] - target_coord)
            if distance == 0:
                temperature = get_temperature_from_sensor(sensor['api_url'])
                if temperature is not None:
                    output_string += f"Temperature: {temperature}\n"
                break
            else:
                weight = 1 / distance**power
                temperature = get_temperature_from_sensor(sensor['api_url'])
                if temperature is not None:
                    numerator += weight * temperature
                    denominator += weight
        if denominator != 0:
            interpolated_temp = numerator / denominator
            output_string += f"{interpolated_temp:.2f}\n"

    return output_string.strip()


interpolated_temperatures_string = idw_interpolation(sensors, target_points)
print(interpolated_temperatures_string)
```

Figure 71: IDW interpolation using Cartesian coordinates. Source: Author, 2024.

Annex I

This part explains how Python code is executed in UE. The "Execute Python Command" function is used to execute Python files in UE. The Python codes that are written in Jupyter Notebook are saved in .Py format. After that, the files are imported into the UE. For importing the .Py files, the mode of the "Execute Python Command" function is set to execute a file, and a string variable is defined and linked to the function to get access to the file by calling the file and importing it using the file address. Figure 72 shows the string variable that is defined for importing and running the Python code. In the Details panel of the string variable (named Import and Reload), Python code for importing the file is written. The file is for IDW interpolation. Figure 73 also shows the link between the string variable and the "Execute Python Command" function in Blueprint.



Variable Name: Import and Reload
Variable Type: String
Instance Editable □
Blueprint Read Only □
Tooltip
Expose on Spawn □
Private □
Expose to Cinematics □
Category: Default
Replication: None
Replication Condition: None

Default Value

Import And Reload

```
import sys
sys.path.append("D:\First
phase\Python_codes")
import Final_Interpolation as Int
import importlib
importlib.reload(Int)
```

Figure 72: String variable for importing and running Python files. Source: Author, 2024.

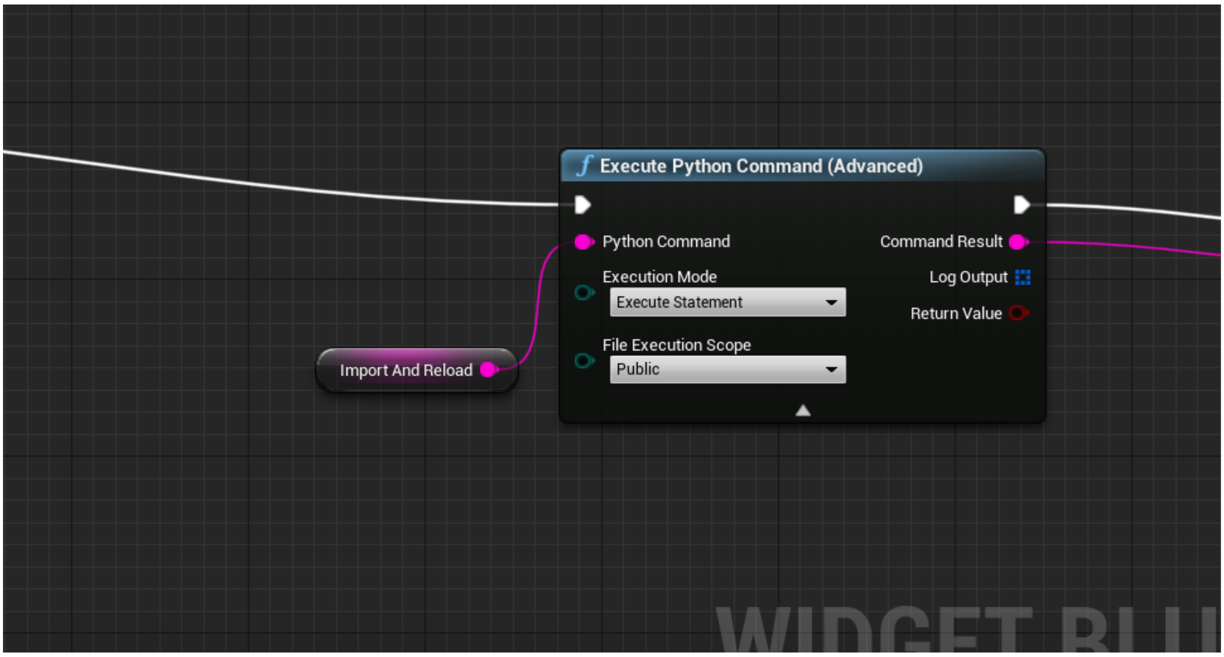


Figure 73: linking the string variable to the Execute Python Command function. Source: Author, 2024.