MSc thesis in Geomatics

Inferring the residential building type from 3DBAG

Hoi-Kang Poon (Chris)
2024

Inferring the residential building type from 3DBAG

Master thesis P5

Hoi-Kang Poon (Chris)
student #4355938

1st supervisor: Camilo León-Sánchez
2nd supervisor: Giorgio Agugiaro
Co-reader: Nail Ibrahimli

January, 2024

Abstract

Urban Energy Modeling (UEM) provides a comprehensive approach to urban planning, helping to create sustainable, resilient, and energy-efficient cities that meet the needs of current and future generations. The key inputs for UEM methodologies and tools are the geometry of the building stock and its thermophysical properties. In the Netherlands, the 3DBAG provides the building stock geometry, while the thermophysical properties can be approximated using energy consumption estimates specific to each residential building type from the IEE project TABULA. However, a challenge arises as open data on residential building types at a national level is currently not readily available, necessitating the development of a method to infer this information from other accessible data sources.

In response to similar successful studies, this thesis also focuses on utilising machine learning to address this challenge. Support Vector Machine (SVM) and Random Forest (RF) algorithms were tested and compared. These algorithms were trained using data on the residential building types obtained from the Rijssen-Holten energy testbed and EP-online, with the latter requiring preprocessing to obtain the relevant information. Additionally, 25 features derived from cadastral data and building geometry underwent a selection process to identify the essential features for accurate classification of building types.

Eight models were trained and applied across eight case studies, containing subsets of the Netherlands representative for the whole country. The combined results were analyzed to determine necessary features, required data, and the most suitable machine learning approach for this research.

The results revealed that features such as adjacency to other buildings, width, and volume in LoD2.2 correlated the most with Dutch residential building types. However, critical features like the number of storeys, the presence of an open porch, or galleries were not available as open data, even though they directly relate to the definition of certain residential building types.

This thesis presented successful models, demonstrating accuracies between 61.1% and 98.5%, and balanced accuracies ranging from 51.6% to 94.2%. Importantly, performance differences were observed in various case studies, particularly in distinguishing accuracy between multi-dwelling and single-dwelling houses. Despite the longer tuning and training time, the suitability and accuracy of the RF models generally outperformed the SVM models.

These findings highlights the capacity of machine learning to attain robust classification outcomes for the Dutch building stock when trained on representative datasets. Nevertheless, it is emphasized that the accuracy of results is contingent on data quality, and difficulties may arise in scenarios involving intricate buildings with multiple components and ambiguous classification rules.

Acknowledgements

I extend my heartfelt gratitude to my main supervisor, Camilo León-Sánchez, whose continuous guidance and patience played a pivotal role in shaping this master thesis. His mentorship provided a steady foundation and a source of inspiration throughout the research process. Special appreciation is also owed to my second supervisor, Giorgio Agugiaro, for his invaluable suggestions and insightful suggestions, which significantly enriched the quality of this work.

I express sincere thanks to Nail Ibrahimli for his meticulous review and constructive feedback on the thesis document. His insights added depth and clarity to the presentation of ideas.

A profound acknowledgment goes to my family and friends for their unwavering support during this academic journey. Their encouragement and understanding provided the essential backdrop to surmount the challenges encountered.

In conclusion, I appreciate the collaborative efforts and contributions of everyone mentioned above, as their collective support played a crucial role in the successful completion of this thesis.

Contents

1 | Introduction

The 17 Sustainable Development Goals are a universal call to action to end poverty, protect the planet and improve the lives and prospects of everyone and everywhere. These goals are adopted by all 193 Member States of the United Nations as part of the 2030 Agenda for Sustainable Development (United Nations, 2015b). From these 17 Sustainable Development Goals, number 11 concerns making cities and human settlements inclusive, safe, resilient, and sustainable. And, stresses the importance of cities and settlements to improve living standards and decrease energy consumption (United Nations, 2015a; Ferrando et al., 2020). One of the most pressing challenges that cities face today is the levels of urban energy consumption. Cities occupy only 3 percent of the Earth's land but account for 60 to 80 percent of all energy consumption. To decrease urban energy consumption, and to achieve Sustainable Development Goal 11, energy in cities must be better managed and the cities' design must be optimised. Many Urban Energy Modeling (UEM) methodologies and tools have been developed to do so. However, if UEM is not applied, cities may not have an accurate understanding of their energy use and may miss opportunities for energy savings. For example, a city may not realise that a significant portion of its energy use is coming from inefficient buildings or outdated infrastructure. This could lead to the city continuing to invest in expensive, unsustainable energy sources rather than investing in energy-efficient upgrades or renewable energy. Two main inputs required for these UEM methodologies and tools are first, the building stock geometry and secondly, the thermophysical properties associated with the entities in the geometry (Ferrando et al., 2020).

The 3DBAG is an open data set containing 3D models of the building stock of the Netherlands at multiple levels of detail. This dataset is generated by combining two open data sets: the building data from the Register of Buildings and Addresses (BAG) and the height data from the National Height Model of the Netherlands (AHN) (Peters et al., 2022). However, the 3DBAG lacks building data on thermophysical properties, which are required for UEM, for example, the construction characteristics of buildings such as materials, size, and order of construction layers.

On the other hand, the IEE project TABULA defined residential building typologies for 13 European countries, where each national typology consists of a classification scheme to group buildings according to their size, age, and other parameters. The TABULA WebTool then provides an online calculation of the exemplary buildings representing the building types and displays their energy-related features (Episcope, 2012). These exemplary buildings can be used to give an estimation of the energy consumption of a building stock by classifying its buildings into the residential building typologies, substituting the need for thermophysical properties of each building of a building stock.

Recent studies such as "3D Building Metrics for building morphology" and "Global Building Morphology Indicators" (Labetski et al., 2022; Biljecki & Chow, 2022) introduce metrics calculated from building models that could potentially be used as features for a machine learning algorithm to classify certain building types.

The master thesis aims to infer the types of residential buildings from the building stock of the Netherlands in the 3DBAG using feature engineering and machine learning, as a way to add thermophysical properties to the residential buildings, which can then be used in UEM. This master thesis is divided into two main parts, firstly to calculate and evaluate the accuracy assessment of the introduced features (feature engineering) extracted from the 3D building models from the 3DBAG; secondly to implement machine learning methods to calculate the building type of residential buildings on the IEE project TABULA residential building typologies for the Netherlands.

1.1 Research questions

The goal of this master thesis is to develop a method to infer the residential building type from the 3DBAG. To achieve this goal machine learning will be used. Multiple combinations of attributes will be considered to be used as features to get accurate building type predictions. These features will primarily be extracted from the 3DBAG, for geometrical features. But, other datasets will also be considered, for example, the BAG for cadastral features. Furthermore, additional data will be needed containing the building types to be used as ground truth for machine learning.

Based on this the main research question can be formulated as follows:

To what extent can machine learning correctly classify the building stock of The Netherlands?

To answer the main research question the following sub-questions are defined:

- What features are needed to infer the building types of the buildings of the 3DBAG?

- What data is required?

- Which (combination of) machine learning algorithm is the most suitable to be used for the classification of the building stock of the Netherlands, with regards to the size and nature of the data used, the availability of computational resources, the interpretability of the results and the desired level of accuracy?

1.2 Scope

The scope of this thesis in inferring the residential building type from 3DBAG is outlined as follows:

- The geographic extent will be the Netherlands since the 3DBAG only contains the building stock of the Netherlands. However, only a subset representative of the whole of the Netherlands will be used to test the method in this thesis. But, the method introduced is expected to be replicated for the whole country or different parts of the country (different subsets).
- Only the residential building types will be considered since the classification performed is to be used with the energy-related features from the IEE project TABULA for UEM. The classification will be based on the method employed by the Dutch cadastre (The Kadaster), but expanded in this thesis by also including the subtypes of apartment blocks.
- Furthermore, the 3D Building Metrics will be employed to obtain features, however, only a selection of metrics are considered due to the complexity of a number of the introduced metrics. Also, the data used to obtain the features and the ground truth will solely be open data.
- Lastly, the machine learning algorithms considered in this thesis are selected with regards to the criteria set in the third sub-question. As such, Support Vector Machine and Random Forest algorithms are selected and will be compared in this thesis.

1.3 Outline

This thesis comprises six main chapters. In Chapter 2, the background and relevant prior research pertaining to this thesis are presented. The Dutch residential building types and different classification systems of these types are introduced, as well as the machine learning algorithms used. Additionally, indicators and metrics introduced in GBMI and 3DBM are presented. Lastly, the chapter explores a previous application of machine learning similar to the research topic of this thesis.

Chapter 3 offers an overview of the method employed to be able to answer the research questions. Subsequently, Chapter 4 delves into the details of the implementation of the method, while also describing the case studies defined in this thesis. Chapter 5 then presents the results derived from the implementation of the method, comparing the performance of the obtained prediction models on the case studies. In Chapter 6 these results are then further discussed and analysed.

Lastly, Chapter 7 concludes this thesis by giving a research overview, where the research questions are presented once more and are answered. Followed by a discussion of the limitations of this thesis and recommendations for future research.

2 | Theoretical background and related work

2.1 Theoretical background

2.1.1 Dutch residential building types

To classify a building from the 3DBAG dataset into one of the residential building types specified in the IEE project TABULA, the criteria used to differentiate each residential building type needs to be examined first. These criteria, defining how one residential building type differs from another, can then be used as features for machine learning algorithms to classify the buildings into their residential building type.

The IEE project TABULA classifies the residential buildings of all participating countries into four generic building types: **single-family houses**, **terraced houses**, **multi-family houses**, and **apartment blocks**. These generic building types are then further divided into building types specific to the Netherlands: **detached single-family houses** and **semi-detached single-family houses** (twee-onder-een-kap), **middle-row terraced houses** (tussenwoning) and **end-house terraced houses** (hoekwoning), **common staircase with galleries apartment block** and **common staircase without galleries apartment block**, and lastly **maisonettes**. Each type is also classified based on their year of construction: before 1964, between 1965 and 1974, between 1975 and 1991, between 1992 and 2005, and after 2006. Each generic building type and subdivision building type has its own exemplary building for each construction year class with its energy-related features (Episcope, 2014). See Figure 2.1 below for the diagram of this classification made by the IEE project TABULA.

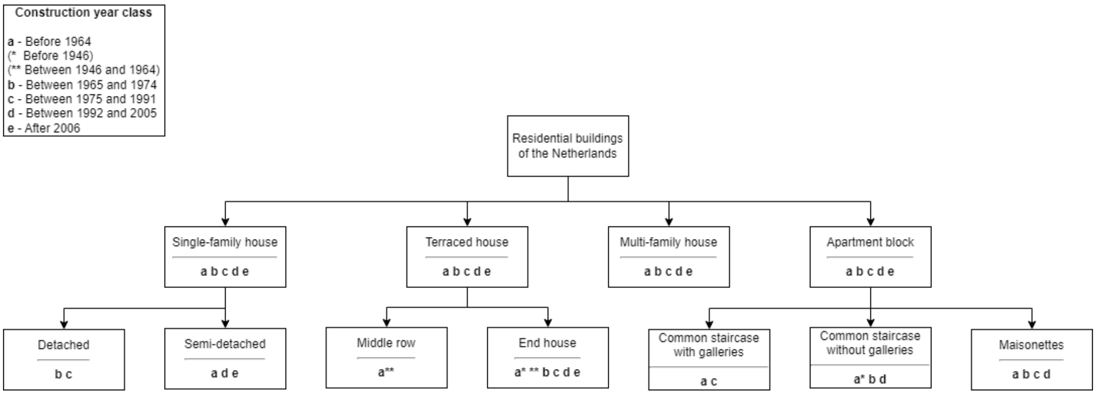


Figure 2.1: TABULA classification diagram.

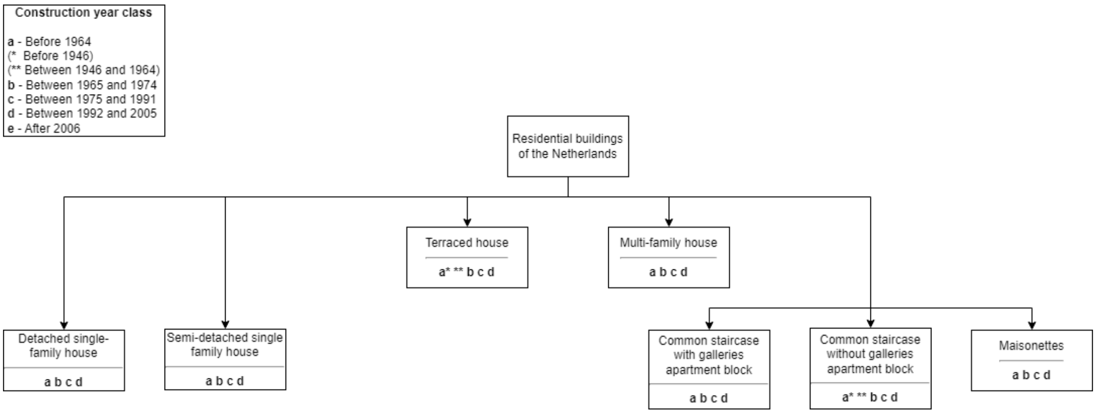
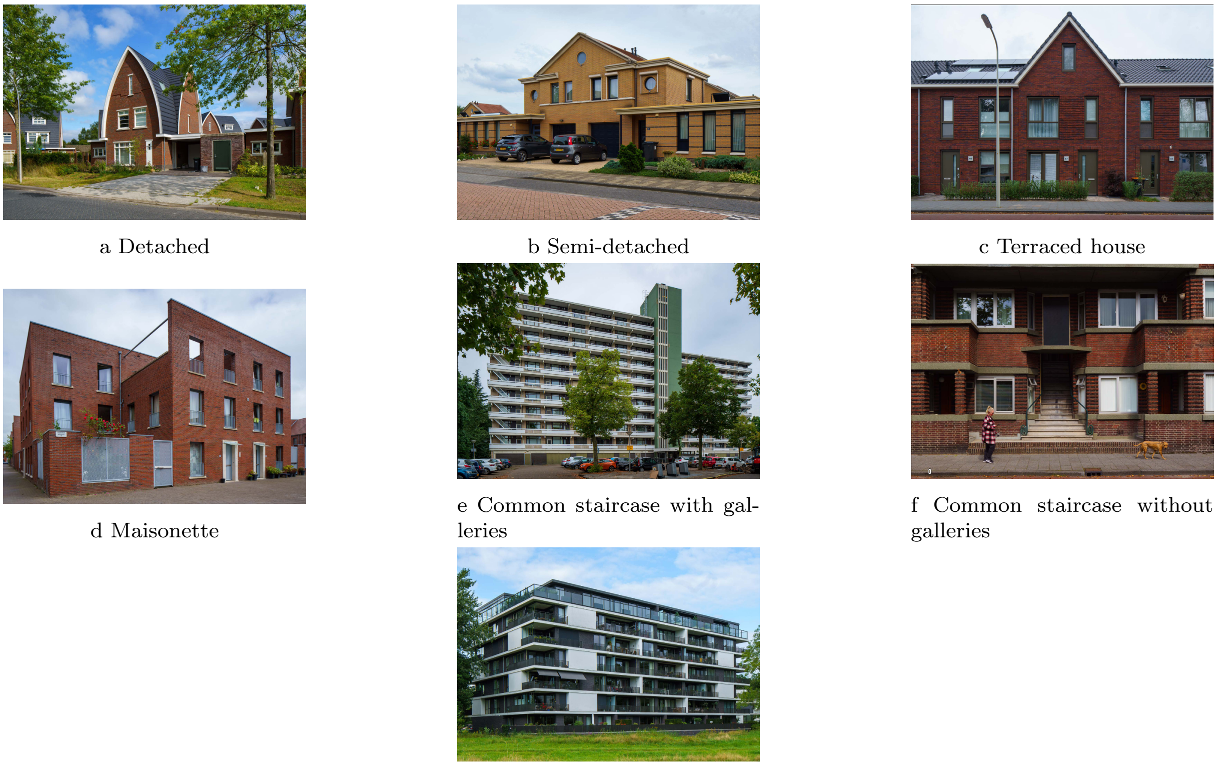

Construction year class
a - Before 1964
(* Before 1946)
(** Between 1946 and 1964)
b - Between 1965 and 1974
c - Between 1975 and 1991
d - Between 1992 and 2005
e - After 2006
Residential buildings of the Netherlands
Terraced house
a* ** b c d
Multi-family house
a b c d
Detached single-family house
a b c d
Semi-detached single family house
a b c d
Common staircase with galleries apartment block
a b c d
Common staircase without galleries apartment block
a* ** b c d
Maisonettes
a b c d

Figure 2.2: Voorbeeldwoningen 2011 classification diagram.

Additionally, the brochure Voorbeeldwoningen 2011 published by Agentschap NL (2011) describes reference dwellings with their energy-related features and the impact of refurbishment measures with the same building types as the IEE project TABULA (see Table 2.1), but with a less hierarchical distinction (see Figure 2.2). For example, there are no generic building types, and the distinction between a **middle row** and **end house terraced house** is not made, but specific building properties are given for front-back facades and side facades. Also, for some of the building types, the number of floors and the floor area for each building type are specified. While in the TABULA the number of floors is not specified and some of the referenced floor areas are estimated (Episcope, 2013). See figure 2.3 for examples showcasing the Dutch residential building types mentioned in IEE project TABULA and Voorbeeldwoning 2011.

Voorbeeldwoningen 2011	**IEE project TABULA**
Vrijstaande woning	Detached
2 onder 1 kap woning	Semi-detached
Rijwoning	Terraced house
Maisonettewoning	Maisonette
Galerijwoning	Common staircase with galleries
Portiekwoning	Common staircase without galleries
(Overig) flatwoning	Multi-family house

Table 2.1: Same building types between Voorbeeldwoningen and IEE project TABULA.



a Detached

b Semi-detached

c Terraced house

d Maisonette

e Common staircase with galleries

f Common staircase without galleries

g Multi-family house

Figure 2.3: Examples of the Dutch residential building types (Rijksdienst voor Ondernemend Nederland, 2023).

Moreover, the Kadaster (2015) utilises a classification system somewhat similar to that of the Voorbeeldwoningen 2011 and the IEE project TABULA. Like the IEE project TABULA, the Kadaster makes a distinction between middle-row and end-row terraced houses. However, it does not differentiate between various types such as **multi-family houses**, **maisonettes**, **apartment blocks with common staircases and galleries**, and **apartment blocks with common staircases but without galleries**. Instead, all these are collectively classified as **apartments** in the Kadaster's classification system.

The Kadaster (2015) elaborates on the classification process of the building types using a flowchart. This classification process classifies a building by linking its address to an exemplary building type or reference dwelling type. This flowchart can be found in Figure 2.4.

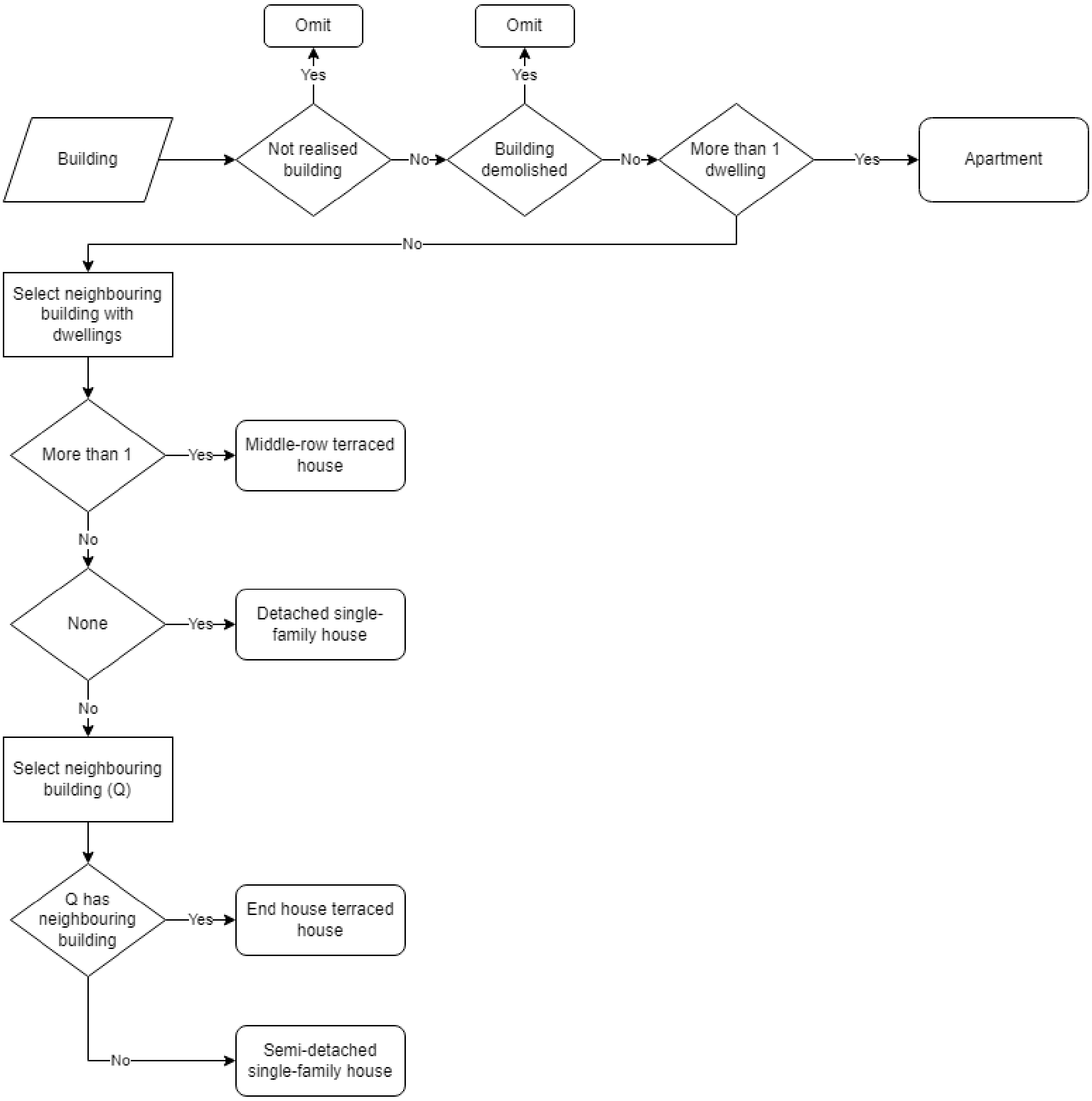

Omit
Omit
Yes
Yes
Building
Not realised building
No
Building demolished
No
More than 1 dwelling
Yes
Apartment
No
Select neighbouring building with dwellings
More than 1
Yes
Middle-row terraced house
No
None
Yes
Detached single-family house
No
Select neighbouring building (Q)
Q has neighbouring building
Yes
End house terraced house
No
Semi-detached single-family house

Figure 2.4: Flowchart of the Kadaster's classification process of buildings.

From this flowchart, the residential building types of the Netherlands can be defined:

- **Single-family houses** (See Figure 2.5):
  - **Detached single-family houses** (DH) are buildings with 1 dwelling and no neighbouring buildings with dwellings.
  - **Semi-detached single-family houses** (SDH) are buildings with 1 dwelling. And 1 neighbouring building with dwellings, and that neighbouring building has no other neighbouring buildings.
  - **End house terraced houses** (EH) are buildings with 1 dwelling. And 1 neighbouring building with dwellings, and that neighbouring building has more than 1 neighbouring building.
  - **Middle-row terraced houses** (TH) are buildings with 1 dwelling and more than 1 neighbouring building with dwellings.
- While **apartments** are buildings with more than 1 dwelling.

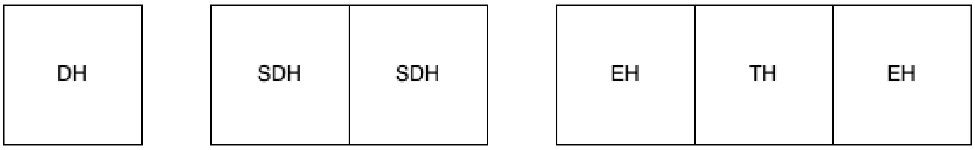


Figure 2.5: Top view 2D illustration of the single-family houses.

According to the Kadaster (2015), using their classification process in 2014 there were less than 1 million classified as non-residential from the 8.5 million addresses in the Netherlands. And less than 3000 addresses which were not classified. The reason for these unclassified addresses is, overlapping buildings in the BAG dataset, (neighbouring) buildings with the same addresses and buildings which were not in use (for example, buildings for sale). There were also errors propagated from errors in the BAG. For example, a building missing data on its dwelling will lead to incorrectly classifying its type. A small space between buildings can lead to classifying a building as an **end house** instead of a **middle-row terraced house** or **semi-detached single-family house**.

Furthermore, the Kadaster's classification process groups the **multi-family houses**, **maisonettes**, **common staircase with galleries apartment blocks** and **common staircase without galleries apartment blocks** into a single type: **apartments**.

The elimination of the subdivision of the **apartments** was also made in the Referentiewoningen nieuwbouw 2013 published by Agentschap NL (2013), a brochure giving exemplary buildings to be used as a reference for new residential buildings. But still they divide the **apartments** into gallery buildings and apartment buildings, while acknowledging the significant diversity within these types and therefore the energy-related values given are an average of the diverse subtypes.

However, Rijksdienst voor Ondernemend Nederland (2023) recently released Voorbeeldwoning 2022, reintroducing the subdivision of the **apartments**. Hence, it remains important to incorporate these apartment subtypes in this thesis, as the current building stock includes these subtypes. Moreover, these subtypes are featured again in the new Voorbeeldwoning 2022, indicating that upcoming constructions will reference these apartment subtype buildings.

2.1.2 Machine learning

Machine learning (ML) is a powerful subfield of artificial intelligence that enables the computer to learn and improve itself through experience and data. Instead of following explicit instructions, machine learning uses algorithms to discover patterns and insights in data, and make predictions or perform tasks based on them (Géron, 2019).

Machine learning can be classified into two main categories: supervised learning and unsupervised learning. The difference between the two is the input data used that a computer learns from. With supervised learning the training data is labelled. This means that the observations in the data set each have their descriptive variables (or features) and their desired outcome of the prediction. A model is then trained to map the features to those predictions to accurately predict the outcome of input descriptive variables for future observations. On the other hand, unsupervised learning does not require labelled training data. Unsupervised models are trained with descriptive variables only, it learns patterns and determines structures in the data without knowing beforehand (Tangirala, 2020).

Supervised learning can be further divided into two major types of supervised learning problems, which are classification and regression. Classification aims to predict class labels, which is a categorical variable from a predefined list. While regression aims to predict a continuous number (Müller, 2017).

It is clear that this master thesis, inferring the residential building type, is a classification problem. The Dutch residential building types described above are the categorical variables from a predefined list which needs to be predicted.

The classification algorithms used in this master thesis are Support Vector Machine and Random Forest. This is mainly due to the interpretability of these machine learning algorithms. Furthermore, Support Vector Machines are effective in high dimensional spaces (input data having many features) while also being memory efficient (Scikit-learn, 2023c). Random Forests are one of the most used machine learning methods. While they are powerful, they often do not require heavy hyperparameter tuning and the scaling of data (Müller, 2017).

Support vector machine (SVM)

Support Vector Machines are powerful supervised machine learning algorithms, which are particularly popular for classification problems but they can also be used for regression tasks. In the case of classification problems, SVM is also called Support Vector Classification (SVC). The goal of SVMs is to find a hyperplane that best separates data points belonging to different classes while at the same time maximising the margin between them.

A hyperplane is the separation of data in higher dimensions, in a two-dimensional space the hyperplane is a straight line. The margin is the distance between the hyperplane and the nearest data points from each class, maximising this margin allows SVMs to perform better on unseen data. The nearest data points to a hyperplane are called Support Vectors, these are crucial in determining the position and orientation of a hyperplane. A kernel function can be used to map the data into a higher-dimensional space allowing SVMs to handle non-linearly separable data (Cristianini & Shawe-Taylor, 2000). However, for this master thesis, only the linear kernel is considered, since the fit time for other kernels scales quadratically with the number of samples (or observations) which will be impractical with more than tens of thousands of samples. Therefore linear SVC is suggested for large datasets, like the building stock of the Netherlands (Scikit-learn, 2023p).

Random Forest (RF)

Random Forests (Breiman, 2001) are ensemble machine learning algorithms that combine multiple decision trees. Decision trees are predictive models that recursively split the input dataset into subsets based on the provided features, making decisions at each node to predict the target variable. Each tree is trained on a different subset of the input data and using different subsets of features. This improves the robustness and therefore the accuracy of predictions of the Random Forest. In both classification and regression tasks, Random Forests are widely used.

Random Forests are effective due to two sources of randomness. The first source is bootstrap aggregating (or bagging), each decision tree in a Random Forest is trained on a randomly sampled (with replacement) subset of the training data. This introduces diversity among the decision trees. The second source is feature randomness, at each node where the input dataset is split into smaller subsets, a random subset of features is considered for splitting. This reduces the chance of overfitting and makes Random Forests more robust. In Random Forest the average of the predictions of each decision tree is taken to cancel out some of the errors. The combination of diverse decision trees reduces the variance, sometimes at the cost of an increased bias. However, in practice, the reduction in variance is often significant enough yielding an overall better model (Scikit-learn, 2023a).

2.2 Related work

2.2.1 Building indicators and metrics

The Global Building Morphology Indicators (GBMI) (Biljecki & Chow, 2022) is a comprehensive compilation of building form multi-scale measures resulting from a systematic literature review. It also encompasses a methodology and a tool designed for the computation of these metrics in a database suitable for extensive data and comparative research. These building-level indicators, both independent and contextual, are detailed in Table 2.2.

Concurrently, the 3D building metrics for urban morphology (3DBM) (Labetski et al., 2022) offer an extensive array of 3D metrics that extend beyond basic building metrics, considering full 3D aspects and intricate building shapes. These metrics (see Table 2.3) are individually computed for each building and can be accessed through a software package that processes 3D city models, storing the metrics systematically for data analysis. These indicators and metrics are integral to the study of urban form and play a pivotal role in numerous large-scale research endeavours (Biljecki & Chow, 2022; Labetski et al., 2022).

These indicators and metrics serve as essential components for classifying the building stock of the Netherlands into residential building types. They also aid in categorising generalised apartment types defined by the Kadaster (2015) into subdivisions from the IEE project TABULA (Episcope, 2014) and the Voorbeeldwoningen brochure from Agentschap NL (2013). However, it is worth noting that adapting the GBMI is needed to accommodate 3DBAG as input data, and as such, falls outside the scope of this thesis.

Indicator	**Data type**	**Unit**
Footprint area	Decimal	m^2
Perimeter	Decimal	m
Height	Decimal	m
Height to footprint area ratio	Decimal	m^{-1}
Volume	Decimal	m^3
Wall area	Decimal	m^2
Evelope area	Decimal	m^2
Number of vertices	Integer	
Complexity	Decimal	
Compactness	Decimal	
Equivalent rectangular index	Decimal	
Mimimum Bounding Rectangle Length	Decimal	m
Mimimum Bounding Rectangle Width	Decimal	m
Mimimum Bounding Rectangle Area	Decimal	m^2
Orientation (azimuth)	Decimal	degree
Number of storeys	Integer	
Floor area	Decimal	m^2
Number of neighbours	Integer	
Site coverage in the buffer	Integer	
Distance to neighbours		
-Minimum	Decimal	
-Median	Decimal	
-Mean	Decimal	
-Maximum	Decimal	
-Sum	Decimal	
-Standard deviation (SD)	Decimal	
-Index of dispersion (D)	Decimal	
-Coefficient of variation (CV)	Decimal	
Neighbour footprint area (varying buffer)		
-Same 8 descriptive statics as above		
Ratio neighbour height to distance		
-Same 8 descriptive statics as above		

Table 2.2: List of indicators at the building level from the GBMI (Biljecki & Chow, 2022).

Geometric properties	Number of vertices, Number of surfaces, Number of vertices by semantic type (i.e. ground, roof, wall), Number of surfaces by semantic type (i.e. ground, roof, wall), Min/Max/Range/Mean/Median /Std/Mode height
Derived properties	Footprint perimeter, Volume, Volume of convex hull, Volume of Object-Oriented Bounding Box, Volume of Axis-Oriented Bounding Box, Volume of voxelised building, Length and width of the Object-Oriented Bounding Box, Surface area, Surface area by semantic surface, Horizontal elongation, Min/Max vertical elongation, Form factor
Spatial distribution	Shared walls, Nearest Neighbour
Space indices	Circularity/Hemisphericality, Convexity 2D/3D, Fractality 2D/3D, Rectangularity/Cuboidness, Squareness/Cubeness, Cohesion 2D/ 3D, Proximity 2D/3D, Exchange 2D/3D, Spin 2D/3D, Perimeter/ Circumference, Depth 2D/3D, Girth 2D/3D, Dispersion 2D/3D, Range 2D/3D, Equivalent Rectangular/Cuboid, Roughness

Table 2.3: Metrics, from the 3D building metrics for urban morphology (Labetski et al., 2022), computed per building based on category.

2.2.2 Inferring the number of floors

Finally, the journal paper "Inferring the number of floors for residential buildings" by Roy et al. (2023) has been used as inspiration for its clear structure and the similarities to this master thesis. In this paper Roy et al focuses on inferring the number of floors of the buildings from the 3DBAG by using supervised machine learning techniques. The labels, in this case, are the building floor count and the features are the building properties. Three machine learning algorithms were used in this paper: Random Forest Regression, Gradient Boosting Regression and Support Vector Regression. And, it was identified that inferring the number of floors is a regression problem since classification would require the training data to include all possible floor counts that exist in reality, which would be difficult to find in practice.

However, as mentioned above in this master thesis the problem is classification, since the building needs to be inferred to discrete residential building types and the training data can include all the possible residential building types of the Netherlands. Nonetheless, the features considered in Roy et al's (2023) paper should be relevant for this master thesis as well. Since the number of floors can be considered a feature in inferring the residential building type. The features can be subdivided into cadastral, geometric and census features. The cadastral features were obtained from the BAG. The geometric features are split into 2D and 3D features, where the 2D features are extracted from the BAG and the 3D features from the 3DBAG. Lastly, the census features were obtained from the Centraal Bureau voor de Statiek (CBS), a government agency responsible for collecting statistical information about the Netherlands. All these features can be found in Tables 2.4, with their details and relevance. The relevances are described for the number of floors but are applicable for the residential building type as well.

	Feature	Details and relevance
1	Construction year	Construction period is often related to storey height.
2	Building function	A distinction is made between residential and mixed-residential, as the latter have been found to exhibit different proteries than purely residential buildings.
3	Net internal area	Taller buildings (with more storeys) generally have a higher net internal area.
4	Number of units	Similar to the net internal area, buildings with more storeys generally contain more building units (for example, apartment blocks).
5	Area	Dividing the net internal area by the footprint area can provide an indication of the number of floors.
6	Perimeter	In combination with area, perimeter can provide information about the footprint shape, such as its compactness and complexity
7	No. vertices	A higher number of footprint vertices could indicate a more complex shape. Computed after simplification by Douglas-Peucker
8	No. neighbours	The number of neighbouring building centroids within 100m radius of the footprint centroid. Buildings with many storeys are generally surrounded by more open space. Buildings in rural areas also generally have fewer neighbours.
9	No. adjacent buildings	The number of buildings within a 0.1m buffer of each footprint. Lower buildings in urban areas generally have more immedidate neighbours
10	Building height	Computed for the minimum, maximum, 50th and 70th roof height percentiles. Building height is strongly related to number of floors, especially for residential buildings.
11	Roof shape	In combination with building height, roof shape could provide information about the likelihood that storeys are present beneath slanted roofs.
12	Ridge vs. eave height	The difference between the height of the ridge and eaves of the roof. Similar to roof shape, this could provide some indication of whether storeys might be present beneath slanted roofs.
13	Roof surface area	Computed for both LoD1.2 and LoD2.2 to describe building geometry.
14	Wall surface area	Computed for both LoD1.2 and LoD2.2 to describe building geometry.
15	Building volume	Computed for both LoD1.2 and LoD2.2. A larger volume is somewhat linked to a larger number of floors.
16	Population per km	Areas with higher population density generally have more high storey buildings to accommodate all residents.
17	Percent multi-household	Multi-household buildings, such as apartment blocks, generally have more storeys than single family homes.
18	Average no. of cafes in 1km	The average number of cafes shows a strong link to area morphology and could be used to distinguish central business districts from rural and suburban areas. Other amenities were also considered but the average number of cafes showed the clearest relationship to area morphology.

Table 2.4: Features in Inferring the number of floors for residential buildings (Roy et al., 2023).

3 | Method

In this chapter the method used to address the research questions will be described in a general manner while giving examples specific to this thesis. The method consists of four main stages:

1. Data collection and preparation
2. Feature extraction and selection
3. Modelling and prediction
4. Accuracy assessment

The flowchart in Figure 3.1 shows the iterative process of the four main stages in this method.

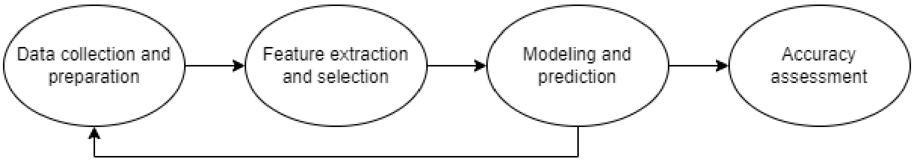

Data collection and preparation
Feature extraction and selection
Modeling and prediction
Accuracy assessment

Figure 3.1: Flowchart of method.

3.1 Data collection and preparation

First, the data required for classification needs to be collected. A semantic 3D city model is necessary to extract geometrical features, for example, from the 3DBAG, which is the basis of this thesis. Additionally, a cadastre database is needed to obtain data such as each building's use, its address, and the number of dwellings in each building. The building's use will also allow the filtering of residential buildings. In this thesis, the BAG is used. Labelled data is also required to provide a ground truth for the machine learning algorithms. It needs to consist of unique building identifiers and building types.

For this thesis, the open testbed for energy applications located in the municipality of Rijssen-Holten and the official national database containing energy labels and energy performance indicators of Dutch buildings (EP-online) are used to create labelled data.

The Rijssen-Holten energy testbed (León-Sánchez et al., 2022) is a dataset created from the 3DBAG, covering only the municipality of Rijssen-Holten. This dataset includes different attributes, some of which are relevant, namely, the building type, the number of adjacent buildings, and the number of storeys of each building. It is used to obtain the building type labels for the municipality of Rijssen-Holten.

On the other hand, EP-online (Rijksoverheid, 2023) contains the energy labels and performance of each dwelling in the buildings of the Netherlands, and also the dwelling's building type. This dataset is used to get the building type labels for the rest of the Netherlands. However, the building type is given for each dwelling in this dataset, and therefore one building can contain many dwellings with different building types. As such, preprocessing is required to obtain the specific building type of each building from EP-online.

Lastly, further data preparation needs to be performed, including removing demolished buildings and standardizing the building type names originating from the different datasets. See Table 3.1 for the open datasets used in this thesis.

Dataset name	Description	Version	Source
3DBAG	3D building models of the building stock of the Netherlands	21.09.8	(Peters et al., 2022)
BAG	National cadastral dataset	01-08-2023	(Kadaster, 2023a)
Rijssen-Holten	Open testbed for energy applications, study area is located in the municipality of Rijssen-Holten	11-07-2022	(León-Sánchez et al., 2022)
EP-online	Official national database containing energy labels and energy performance indicators of buildings	01-01-2023	(Rijksoverheid, 2023)

Table 3.1: Relevant datasets.

3.2 Feature extraction and selection

Second, features which describe the properties of each building need to be extracted from the collected data. The methodology and tools of the 3DBM will be used and expanded with features which are needed for the classification of the residential buildings for this thesis. The required features will be determined by definitions of each Dutch residential building type.

In this thesis, several required features can be derived from the classification process of the Kadaster, namely the *number of dwellings*, the *number of neighbouring buildings with dwellings* and the *number of neighbouring buildings of the neighbouring building.* These cadastral features can be obtained from the BAG. The geometrical features extracted with the 3DBM from the 3DBAG will be used to further classify the apartment residential building type into its subdivisions.

Several computed features, including the features extracted with the 3DBM, are then validated to make sure the extracted features are correctly computed and represent the feature required for the classification. These features will then need to be analysed and assessed to eliminate any redundant or irrelevant features, a process known as feature selection or feature elimination. Feature selection helps with selecting a subset of features that can provide a concise description of the training dataset, while still generating accurate predictions (Chandrashekar & Sahin, 2014).

3.3 Modelling and prediction

Third, the classification of Dutch residential buildings in this thesis will be carried out using SVM and RF algorithms. Before training the algorithms, data preprocessing is necessary, involving the conversion and appropriate scaling of extracted features for machine learning. Additionally, 80% of the data will be extracted for training, while the remaining 20% will serve for model evaluation to provide an unbiased performance assessment.

The evaluation of model performance will involve the use of various error metrics. To determine the most suitable machine learning approach for classifying Dutch residential buildings, the results of these algorithms must be compared to the ground truth. This comparison is typically facilitated through the utilisation of a confusion matrix, which includes True Positive, True Negative, False Positive, and False Negative values, and is usually presented in a tabular format as shown in Figure 3.2. The confusion matrix is a special kind of contingency table, which is introduced by Pearson (1904). Fawcett (2006), among many others, adapted this to be used for organising classifiers and visualising their performance.

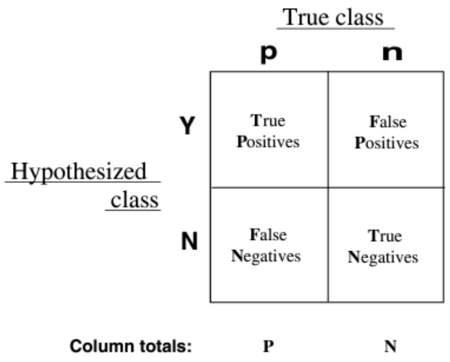


Figure 3.2: Confusion matrix (Fawcett, 2006), true class = label and hypothesized class = prediction

Each prediction from a model can be one of these four types:

- True Positive (TP), is when a sample is predicted to be positive (for example, a building is predicted to belong to a certain residential type) and its label is actually positive (for example, the building actually belongs to that residential type).
- True Negative (TN), is when a sample is predicted to be negative and its label is actually negative.
- False Positive (FP), is when a sample is predicted to be positive, but its label is actually negative.
- False Negative (FN), is when a sample is predicted to be negative, but its label is actually positive.

With these values, several performance metrics can be computed, namely Precision, Recall and the F1 score. These metrics can be used to evaluate the performance of the models for this thesis while being able to handle imbalanced class distributions.

Precision is the ratio of correctly predicted positive outcomes to all positive predictions made by the model. It also represents the model's capability to avoid misclassifying a positive sample as negative (Scikit-learn, 2023l):

$$\text{Precision} = \frac{TP}{(TP + FP)} \tag{3.1}$$

While, recall is the fraction of actual positives out of all positive predictions and can also be described as the ability of the model to find all positive samples (Scikit-learn, 2023m):

$$\text{Recall} = \frac{TP}{(TP + FN)} \tag{3.2}$$

Finally, the F1-score is the harmonic mean between the precision and recall, an F1-score of value 1 is considered the best and a value of 0 the worst. It can be described as the balanced ability of the model to both capture positive cases (recall) and be accurate with the cases it does capture (precision) (Scikit-learn, 2023k):

$$\text{F1-score} = \frac{2(Precision \cdot Recall)}{(Precision + Recall)} \tag{3.3}$$

Furthermore, in scikit-learn (Pedregosa et al., 2011) the accuracy is computed as the weighted average of recall. While the balanced accuracy is computed as the average recall obtained on each class, this allows the balanced accuracy to deal with imbalanced datasets and give a better performance metric (Scikit-learn, 2023j). Both will be used as a summation of the introduced metrics to compare the performance of each trained model.

To optimise the performance of each model, hyperparameter tuning is necessary. Hyperparameter tuning involves discovering the most effective combination of hyperparameters for a given model on a specific task. These hyperparameters are configuration settings predetermined before training a machine learning model. They govern the model's behaviour during training and impact its performance and generalisation on unseen data.

The process of hyperparameter tuning begins by defining a search space encompassing all potential values or ranges for each hyperparameter. This search space is explored using stratified K-fold cross-validation (Scikit-learn, 2023o) in combination with a randomised search (Scikit-learn, 2023n). In stratified K-fold cross-validation, the training data is divided into smaller sets, preserving class distribution, with one fold used for evaluation while the others are used for training.

A randomised search over parameters involves randomly sampling settings from a subset of possible parameter values or ranges. This approach offers advantages over exhaustive grid searches, such as flexibility in setting a budget independent of the number of parameters and their possible values. Additionally, it avoids including parameters that do not affect model performance, thereby optimising efficiency. While there is a minimal risk of performing worse than a grid search, a randomised search enables swift parameter tuning while delivering the best overall model performance (Scikit-learn, 2023i).

3.4 Accuracy assessment

In the accuracy assessment, the trained models will be used on different smaller areas of the Netherlands to assess the performance of models on different urban fabric, which are the physical characteristics of urban areas (for example, villages, towns and cities). These smaller areas are selected to represent the whole of the Netherlands. To get insight on how well the models will perform if applied to the whole of the Netherlands. These model applications will also be evaluated with the Precision, Recall, F1-score, accuracy and balanced accuracy (Scikit-learn, 2023f, 2023d, 2023e).

Lastly, a hit-and-miss analysis will be performed, where a random sample of predictions is taken and visualised through Google Street View to confirm whether the label was correct or not and whether the prediction was correct or not. This will give insight into the true performance of the final models on the classification of the residential buildings of the Netherlands. At the same time the impact of the data available, in particular how accurate the labels of building type classification are in the Rijssen-Holten dataset (León-Sánchez et al., 2022) and the Ep-online dataset (Rijksoverheid, 2023).

4 | Implementation

Before implementation, the flowchart introduced by the Kadaster (2015) needs to be first adapted to include the specific classification of apartments by IEE project TABULA (Episcope, 2012) using additional information gathered from Voorbeeldwoningen 2011 (Agentschap NL, 2011). This flowchart can be found in Figure 4.1.

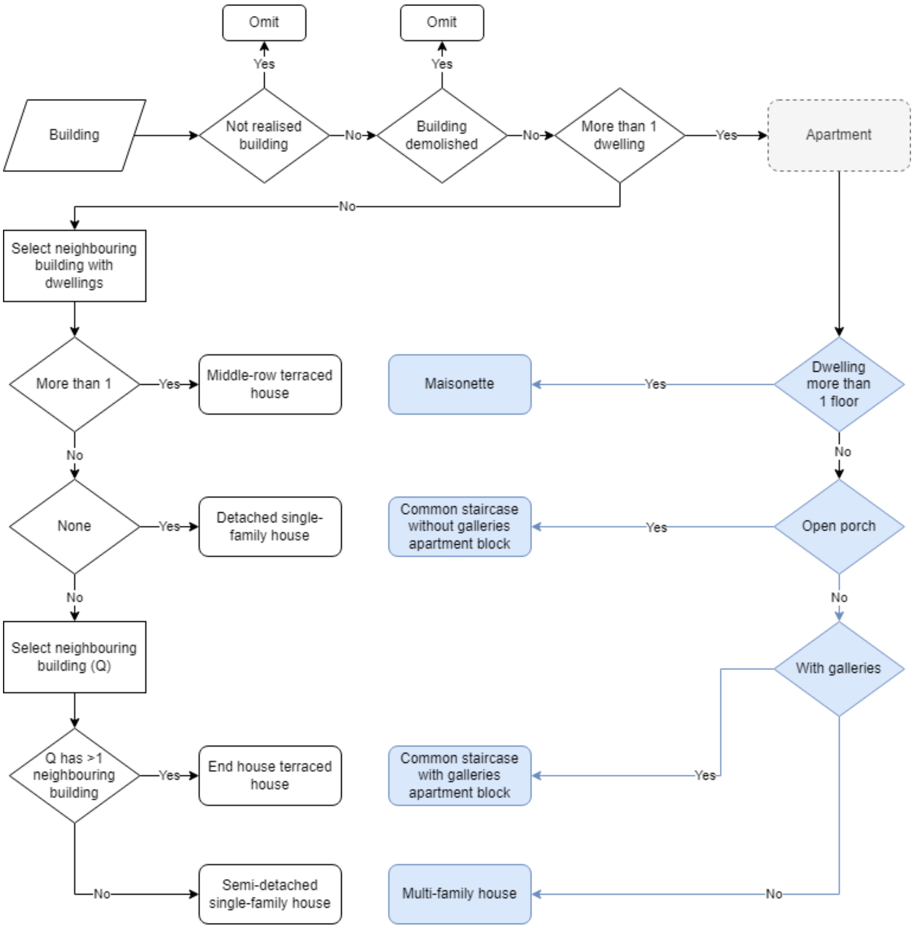

Omit
Omit
Yes
Yes
Building
Not realised building
No
Building demolished
No
More than 1 dwelling
Yes
Apartment
No
Select neighbouring building with dwellings
More than 1
Yes
Middle-row terraced house
Maisonette
Yes
Dwelling more than 1 floor
No
No
None
Yes
Detached single-family house
Common staircase without galleries apartment block
Yes
Open porch
No
No
Select neighbouring building (Q)
With galleries
Q has >1 neighbouring building
Yes
End house terraced house
Common staircase with galleries apartment block
Yes
No
Semi-detached single-family house
Multi-family house
No

Figure 4.1: Flowchart of the Kadaster's classification process of buildings, expanded to include the specific classifications of apartments (in blue).

From this flowchart, the residential building types of the Netherlands, for single-family houses and apartments can be defined, see Table 4.1 below. And thereby, also define the features needed to classify the residential building types.

Building type	Definition	
Single-family houses	dwellings	features
Detached	1	no neighbouring buildings with dwellings
Semi-detached	1	1 neighbouring building with dwellings, which has no other neighbouring building
End house terraced house	1	1 neighbouring building with dwellings, which has 1 or more other neighbouring buildings
Middle-row terraced house	1	more than 1 neighbouring building with dwellings
Apartments	dwellings	features
Maisonettes (See Figure 4.2	1	Its dwellings have more than 1 floor
Common staircase without galleries	1	Building has an open porch (see Figure 4.3 - left)
Common staircase with galleries	1	Building has galleries (see Figure 4.3 - right
Multi-family house	1	None of the above

Table 4.1: Features defining the Dutch residential building types.

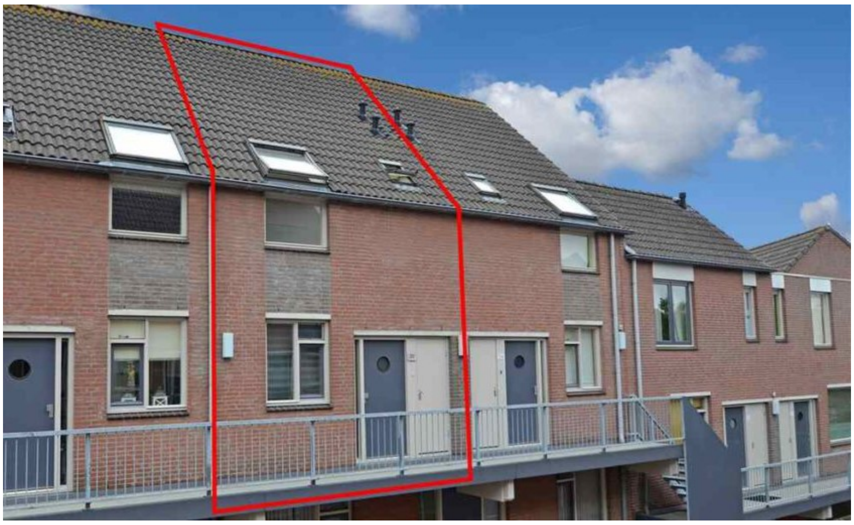
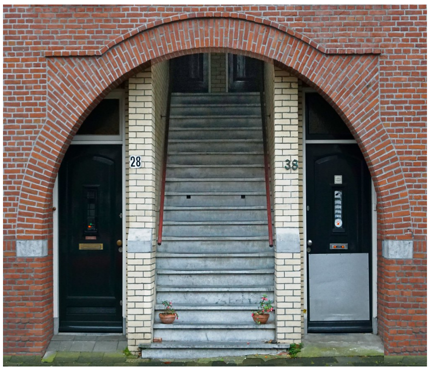


Figure 4.2: Maisonettes are dwellings with more than 1 floor in an apartment block.



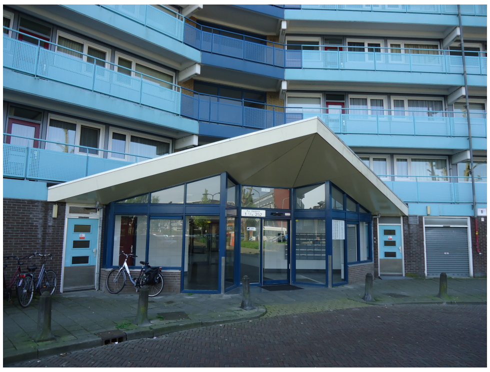


Figure 4.3: An open porch, typical for a common staircase without galleries apartment block (left) and galleries often have the appearance of long, threaded balconies, which lead to the entrances of the dwellings (right).

However, the features that differentiate the apartment buildings: the *number of floors per dwelling*, the building *having an open porch* and the building *having galleries* are not available or are not available as open data. Hence, the classification of the apartment buildings will have to be done by substituting these features with the features introduced in 3DBM (Labetski et al., 2022). Assuming that the building form relates to each specific residential apartment building type. In total 25 features will be extracted, see Table 4.2.

	Features	**Source**	**Details and relevance**
1	Number of adjacent buildings	BAG	See classification process of the Kadaster
2	Number of adjacent buildings of adjacent buildings.	BAG	See classification process of the Kadaster
3	Year of construction	BAG	Construction year might not be relevant to inferring the residential building type, but it is relevant to determining the construction year class of a residential building type.
4	Number of dwellings in the building	BAG	See classification process of the Kadaster
5	Footprint area	BAG	Describes the building geometry.
6	Footprint perimeter	BAG	Describes the building geometry and provides additional information about the footprint shape, like the compactness and complexity.
7	Number of the vertices in footprint	BAG	The number of the vertices gives another indication of the complexity of the footprint shape.
8	The number of neighbouring buildings (radius: 25m)	BAG	The number of neighbouring building centroids within a certain radius of the footprint centroid. For example, taller buildings, like apartment blocks generally have more open space in the surroundings.
9	The number of neighbouring buildings (radius: 50m)	BAG	The same as above, but larger radius
10	The number of neighbouring buildings (radius: 75m)	BAG	The same as above, but larger radius
11	The number of neighbouring buildings (radius: 100m)	BAG	The same as above, but larger radius
12	Actual volume in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
13	Convex hull volume in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
14	Oriented bounding box width in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
15	Oriented bounding box length in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
16	Total wall surface area in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
17	Total roof surface area in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
18	Maximum height in LoD1.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
19	Actual volume in LoD2.2	3DBAG	Describes building geometry, utilising LoD1.2 of the 3DBAG
20	Convex hull volume in LoD2.2	3DBAG	Describes building geometry, utilising LoD2.2 of the 3DBAG
21	Total wall surface area in LoD2.2	3DBAG	Describes building geometry, utilising LoD2.2 of the 3DBAG
22	Total roof surface area in LoD2.2	3DBAG	Describes building geometry, utilising LoD2.2 of the 3DBAG
23	Maximum height in LoD2.2	3DBAG	Describes building geometry, utilising LoD2.2 of the 3DBAG
24	Height (without roof) in LoD2.2	3DBAG	Describes building geometry, utilising LoD2.2 of the 3DBAG
25	*Number of storeys of the building*	*Rijssen-Holten*	*Number of storeys of the building*

Table 4.2: List of extracted features.

Note that several features have a LoD1.2 and LoD2.2 version. This is because the 3DBAG contains 3D geometries of the building stock of the Netherlands in LoD1.2, LoD1.3 and LoD2.2. For this thesis, LoD1.2 and LoD2.2 geometries are considered in the feature extraction process. This is to compare the relevance of features with different level of details.

Furthermore, the *number of storeys* of each building was extracted as a feature from the Rijssen-Holten energy testbed dataset (León-Sánchez et al., 2022) as well. This feature also went through the feature selection process. However, it did not get selected as a feature to be used in the training process. Since the feature was among the five lowest scoring features across all the selection methods used in this thesis (see Section 4.3.3). Also, this feature is not available in the Dutch national energy label dataset (Rijksoverheid, 2023) and could not be extracted as a feature for areas in the Netherlands outside of Rijssen-Holten.

4.1 Case studies

During the implementation, several case studies were explored. The **first case study** was created by using the Rijssen-Holten energy testbed dataset (León-Sánchez et al., 2022) as ground truth, since this dataset already includes the building type of each building in Rijssen-Holten and no further preprocessing is required to obtain the building type of each building. However, initial analysis shows that the class distribution of the building types of the Rijssen-Holten energy testbed dataset presents a severe class imbalance, with the smallest minority class being only 0.03% of the entire dataset, see Table 4.3. The table shows that the majority of the classes are single-family houses: **end**, **detached**, **semi-detached** and **terraced houses**. And that the minority of the classes (about 1%) are **common staircase with galleries**, **common staircase without galleries**, **maisonettes** and **multi-family houses**. This is also reflected in the housing characteristics of Rijssen-Holten, where buildings containing only 1 dwelling make up 79% of all the buildings and buildings containing more than 1 dwelling only make up 21% (Allecijfers.nl, 2023b).

Building type	**Name**	**Count**	**Percentage**
Galerijwoning	Common staircase with galleries	4	0.06%
Maisonnettewoning	Maisonettes	4	0.06%
Portiekwoning	Common staircase without galleries	8	0.11%
Flatwoning	Multi-family house	58	0.80%
Hoekwoning	End house	1452	20.06%
Vrijstaande Woning	Detached	1783	24.63%
Twee-onder-een-kapwoning	Semi-detached	1785	24.66%
Tussenwoning	Terraced house	2145	29.63%
TOTAL		**7239**	**100.00%**

Table 4.3: Class distribution of Case Study 1 (after feature extraction).

Class distribution statistics are important in machine learning classification problems because the distribution can affect the performance of the model (Branco, Torgo, & Ribeiro, 2015). If the classes are imbalanced, meaning one class has significantly more or fewer data records than the other classes, it can lead to biassed model predictions towards the majority class. This is because the model can achieve high accuracy by simply predicting the majority class for all instances while ignoring the minority class (Weiss, 2013). However, this can be problematic if the minority class is the one of interest, like one of the residential building types of the Netherlands.

Therefore, a **second case study** was created, but this case study will have to be created by using the Dutch energy label dataset (Rijksoverheid, 2023) and preprocessing will be required to determine the building type for each building. Delft was picked as the study area for this case study, because of the familiarity with Delft and its buildings, which will help with determining the building types. Also, housing characteristics for Delft show the inverse of the housing characteristics for Rijssen-Holten. 33% of all buildings in Delft contain 1 dwelling and 67% contain more than 1 (Allecijfers.nl, 2023a).

Building type	**Name**	**Count**	**Percentage**
Galerijwoning	Common staircase with galleries	63	0.60%
Portiekwoning	Common staircase without galleries	118	1.12%
Vrijstaande Woning	Detached	253	2.39%
Twee-onder-een-kapwoning	Semi-detached	552	5.22%
Maisonnettewoning	Maisonettes	681	6.44%
Hoekwoning	End house	1437	13.58%
Flatwoning	Multi-family house	1476	13.95%
Tussenwoning	Terraced house	5999	56.71%
TOTAL		**10579**	**100.00%**

Table 4.4: Class distribution of Case Study 2 (after feature extraction).

However, the class distribution for Delft is also imbalanced, but the minority classes are better represented, see Table 4.4. In many real-world scenarios, class distribution imbalances naturally exist due to the nature of the data being collected. For the cases of this master thesis specifically, the class distribution can be explained by the example of a multi-family house. A multi-family house consists of many more dwellings than 1, while a terraced house can only consist of 1 dwelling. Due to the amount of dwellings inside one multi-family house building the size of such a building is automatically bigger than compared to a terraced house. Therefore, such a building will occur less in a specific demarcated area compared to a small building like a terrace house.

Hence, models from both **Case Study 1** (Rijssen-Holten) and **Case Study 2** (Delft) are created and tested on each other's dataset. But also tested on 6 more other case studies, focussing on one of the building types, to evaluate the effect of a severe class imbalance and less severe class imbalance. And to also evaluate the performance of the models on different subsets of the Netherlands. Note that prediction models are only created for **Case Study 1** and **2**, while **Case Studies 3 through 8** are used to evaluate these models.

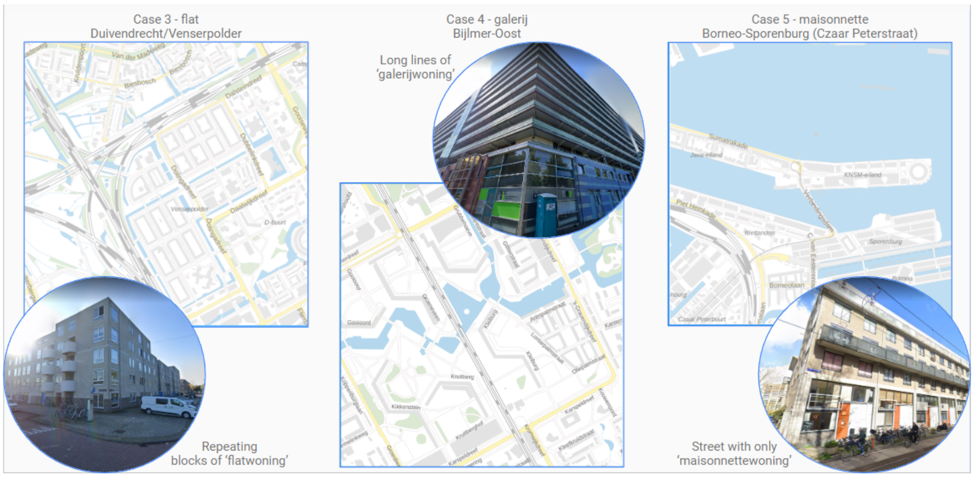


Figure 4.4: Visual representation of the study area and focused building type of Case Studies 3, 4 and 5.

Case Study 3 Duivendrecht/Venserpolder is situated on the outskirts of Amsterdam and centres on **multi-family houses**. It contains repeating blocks of **multi-family houses** evident in the map of case study 3 in Figure 4.4.
Case Study 4 Bijlmer-Oost, focuses on **common staircase apartment blocks with galleries**. Figure 4.4 illustrates one such block, easily showcasing the galleries and the map shows these blocks are repeated in the form of diagonal lines. Bijlmer-Oost is a medium-sized district in Amsterdam.
Case Study 5 Borneo-Sporenburg features two peninsulas surrounded by docklands in the Amsterdam neighbourhood. Notably, the Czaar Peterstraat in this area primarily consists of **maisonettes**, see Figure 4.4.

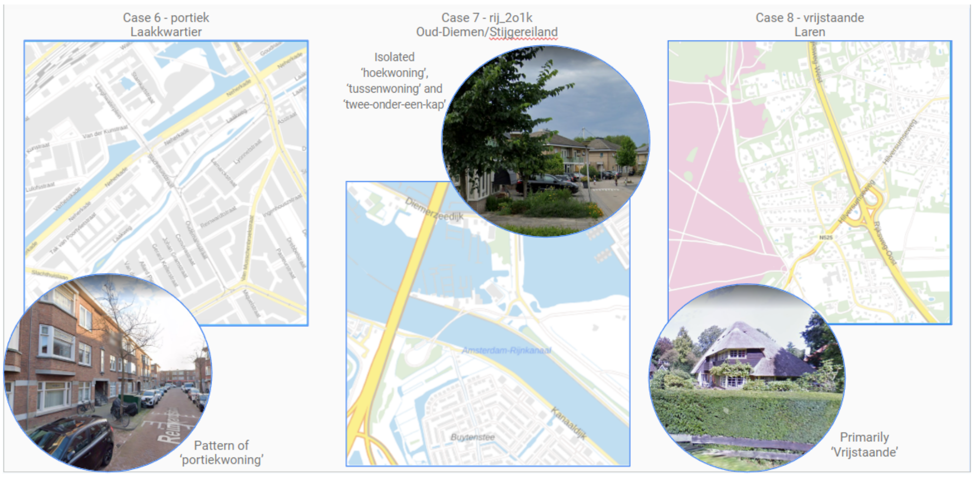


Figure 4.5: Visual representation of the study area and focused building type of Case studies 6, 7 and 8.

Case Study 6 Laakkwartier is a neighbourhood in The Hague, where **common staircase apartment blocks without galleries** occur frequently in the urban fabric. In Figure 4.5, these blocks can be observed between other building types. This pattern of **common staircase apartment blocks without galleries** exists throughout all the housing blocks observed on the map.
Case Study 7, Oud-Diemen/Steigereiland, focuses on **semi-detached houses**, **terraced houses**, and **end houses**. Figure 4.5 depicts several **semi-detached houses**, easily observable on the map. Oud-Diemen is a neighbourhood in Diemen, a town in the Netherlands.
Finally, in **Case Study 8** Laren, which is a town in the Netherlands, focuses on **detached houses**. These can be observed in the figure and on the corresponding map it can be seen that most of the buildings are **detached houses**, see Figure 4.5.

The distribution of classes for all these cases is detailed in Table 4.5 below. It is important to note that in **Case Study 6**, which is intended to focus on **common staircase apartment blocks without galleries**, only 6 buildings have been classified as such, based on labels extracted from the Dutch energy label dataset (Rijksoverheid, 2023). This count remains unchanged and is indicated by an asterisk in the table. The reason for this is to assess whether the models can correctly identify **common staircase apartment blocks without galleries**, even when there might be false positives from other building types.

Case Study	MFH	Galleries	Maisonette	Without Galleries	EH	TH	SDH	DH
3	420	17	24	3	196	621	5	9
4	38	12	9	-	64	331	1	8
5	236	-	69	4	28	376	19	4
6	649	5	96	6*	5	86	-	4
7	49	-	6	-	103	287	69	88
8	7	-	-	-	2	2	25	163

Table 4.5: Class distribution of Case Studies 3 through 8.

Meanwhile, the datasets for **Case Studies 2, 3, 4, 5, 7 and 8** have undergone preprocessing steps. Specifically, SQL scripts were employed to first determine what building type a specific building should have with different building types for each dwelling. Since the building type is given per dwelling in the Dutch energy label dataset (Rijksoverheid, 2023) and a building can consist of one or more dwellings. The building type extracted from EP-online might be a list of building types possibly consisting of different building types. The building type of a building with a list of building types is determined by counting the number of occurrences of each building type in this list, the highest number of occurrences is then taken as the building type for that building.

Furthermore, there are also buildings with only one dwelling but the building type given for this one dwelling is 'twee-onder-een-kap of hoekwoning' (**semi-detached house** or **end house**) or this one dwelling is given the building type 'appartement' (apartment block) or even given as 'NULL'. The building type is then determined by its adjacency features or manually by visual inspection using Google Street View or by information gathered from housing websites when available. And there were also buildings where the extracted list of building types only consisted of one entry, the building type of this one entry is then taken as the building type. Table 4.6 shows the amount of these cases for each building for each case study is given.

Case Study	list of building types	SD or EH	Apartment	NULL	one building type in list	TOTAL
2	1841	1719	88	33	6873	**10554**
3	421	200	12	1	660	**1294**
4	51	72	0	0	340	**463**
5	360	58	26	18	392	**854**
6	618	7	26	1	198	**850**
7	42	204	7	0	349	**602**
8	6	23	0	0	170	**199**

Table 4.6: Amount of each different case after extracting the building types from EP-online.

4.2 Programming details

The method was implemented using Python and SQL, the code is available at `https://github.com/tudelft3d/ML_bldg_type`. A PostgreSQL (2023) database extended with PostGIS (2023) was used in combination with 3DCityDB (Yao et al., 2018) to store the data required for this thesis, while several features were extracted using PostGIS functions. The features were validated by also using the PostGIS functions and by comparing them with similar features extracted from the 3DBM.

See Figure 4.6 for an overview of the database structure used for the implementation. The Rijssen-Holten dataset (León-Sánchez et al., 2022) and the subsets of the Netherlands (**Case Studies 2 through 8**) from the 3DBAG (Peters et al., 2022) are imported to 3DCityDB in separate citydb schemas using the 3DCityDB Importer/Exporter.

The Dutch energy label dataset (EP-online) (Rijksoverheid, 2023) is imported as a .csv file into the database as well as the features obtained from 3DBM. While the BAG dataset (Kadaster, 2023a) was imported with the ogr2ogr command line tool (Rouault et al., 2023). These datasets are stored in the input data schema in their specific tables.

The extracted features are then stored as training data for each case in their tables. Lastly, the training data schema also contains the temporary tables made to validate the features for **Case Study 1**, which are discussed further in this chapter.

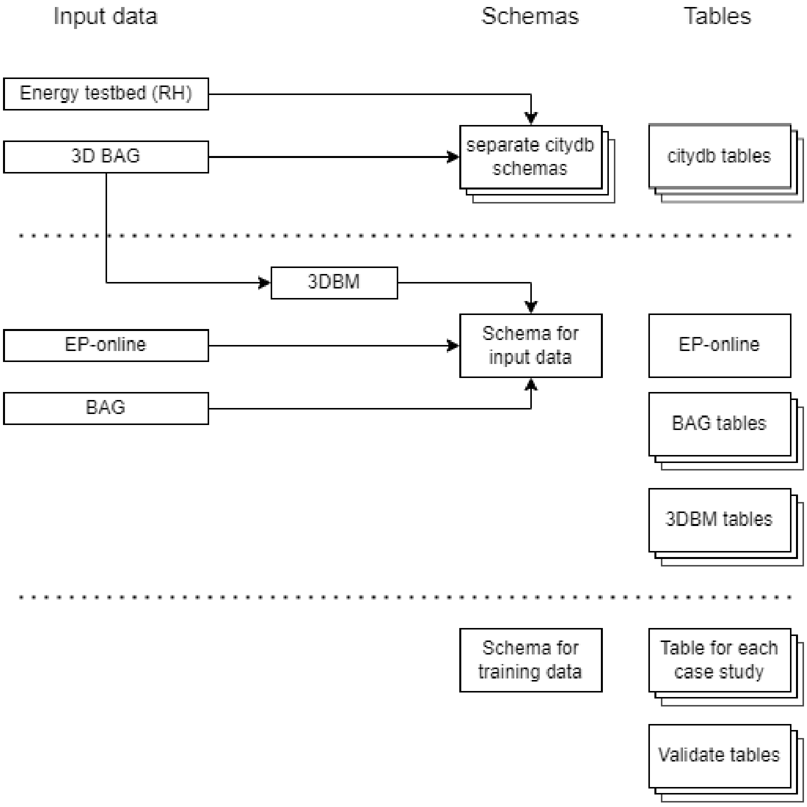


Figure 4.6: Overview of the structure of the PostGIS database.

A connection to the database with the psycopg2 library (2023) was used to be able to perform data preparation in the database using Python and SQL queries utilising PostGIS as well. However, the correction of the labels of the Dutch energy label dataset (Rijksoverheid, 2023) was applied directly in the database using SQL queries alone. These queries can also be found on the GitHub repository.

To perform the machine learning steps the open-source scikit-learn library in Python (Pedregosa et al., 2011) is used. The library has a consistent API for different algorithms, which allows switching between models without a major change in the code. This consistency simplifies the development and experimentation process. Furthermore, scikit-learn integrates other popular data science libraries in Python, such as NumPy (Harris et al., 2020), Pandas (pandas development team, 2020), and Matplotlib (Hunter, 2007). This facilitates data preprocessing, exploration, and visualisation tasks. The library also provides tools for data preprocessing including feature scaling and more. Lastly, scikit-learn library includes utilities for model evaluation, hyperparameter tuning, and model selection. This allows the comparison of different models and the selection of the model that performs best in the specific problem of this thesis.

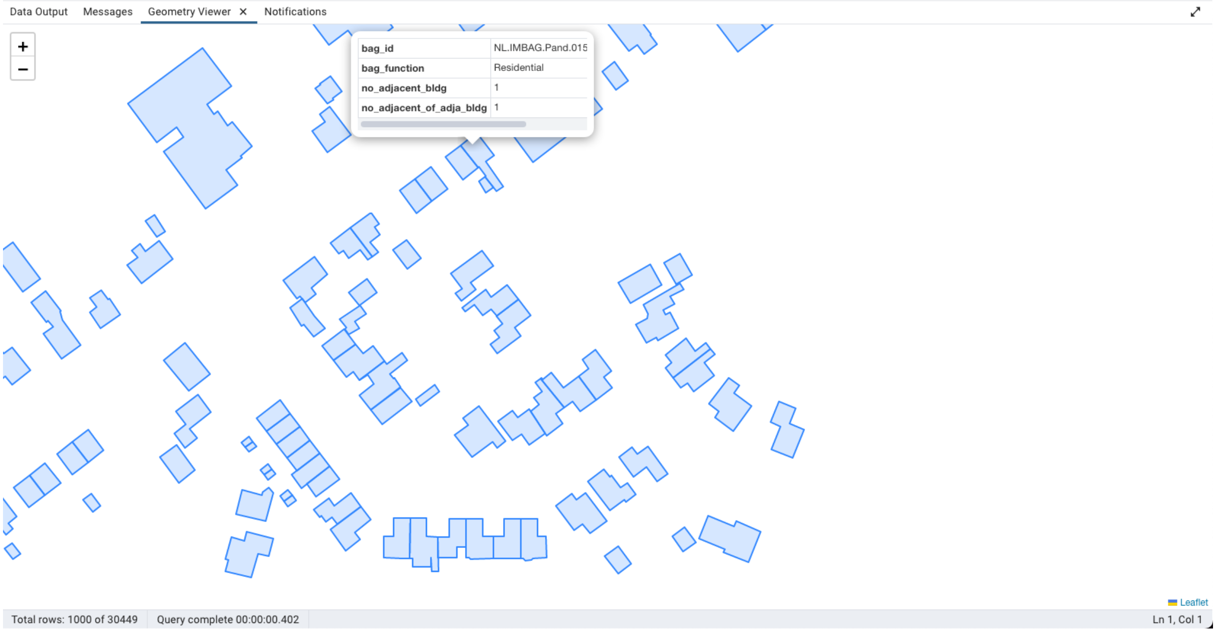
4.3 Feature engineering

4.3.1 Validation

To be able to answer one of the research questions: What features are needed to infer the building types of the buildings of the 3DBAG? The features need to be validated first, to make sure the extracted features are correctly computed and represent the feature required for the classification. The feature validation was also performed to test the 3DBM results. SQL queries are used for each feature validated, these queries can be found on the GitHub repository also.

Adjacency

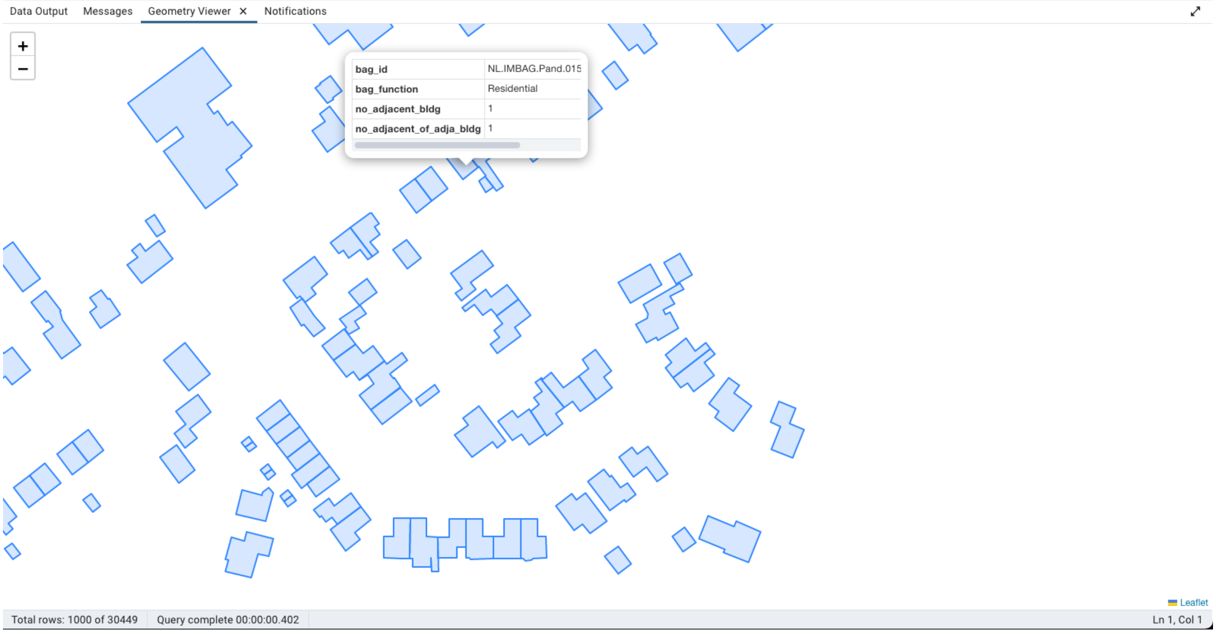
The adjacency features were first validated by visualising the footprints while showing these adjacency features and the building functions. The adjacency features are then validated by visual inspection using a Geometry Viewer. This inspection is performed by checking the adjacency features in the pop-up balloons of randomly sampled buildings. See Figures 4.7, 4.8 and 4.9 for one of these buildings. It can be observed that the two residential building footprints show an adjacency of one (Figure 4.7 and 4.8), while not counting the footprint of the building with an unknown function (Figure 4.9). This is working as intended since the code does not take buildings with unknown (or 'Other') functions. Buildings with the 'Other' function consist only of dwellings in which no people reside, like garage boxes (Kadaster, 2023b). Therefore, the adjacency feature ignores buildings with these building functions.



bag_id	NL.IMBAG.Pand.015
bag_function	Residential
no_adjacent_bldg	1
no_adjacent_of_adja_bldg	1

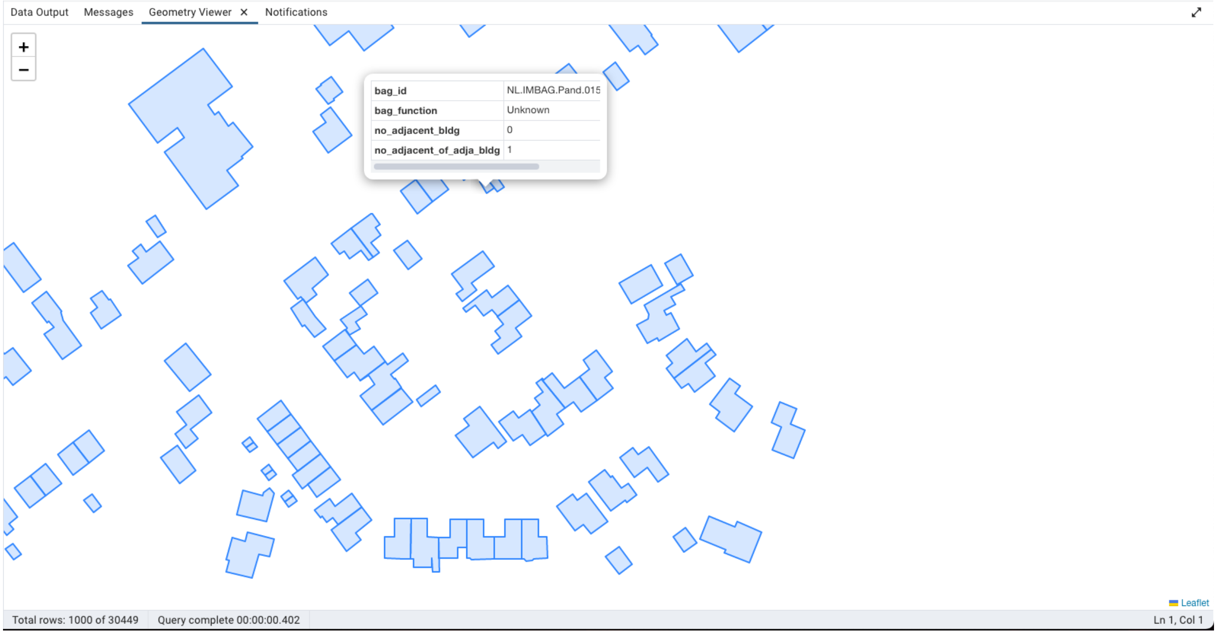
Total rows: 1000 of 30449 Query complete 00:00:00.402 Ln 1, Col 1

Figure 4.7: Validation of adjacency feature by visualising building footprints (1/3).



bag_id	NL.IMBAG.Pand.015
bag_function	Residential
no_adjacent_bldg	1
no_adjacent_of_adja_bldg	1

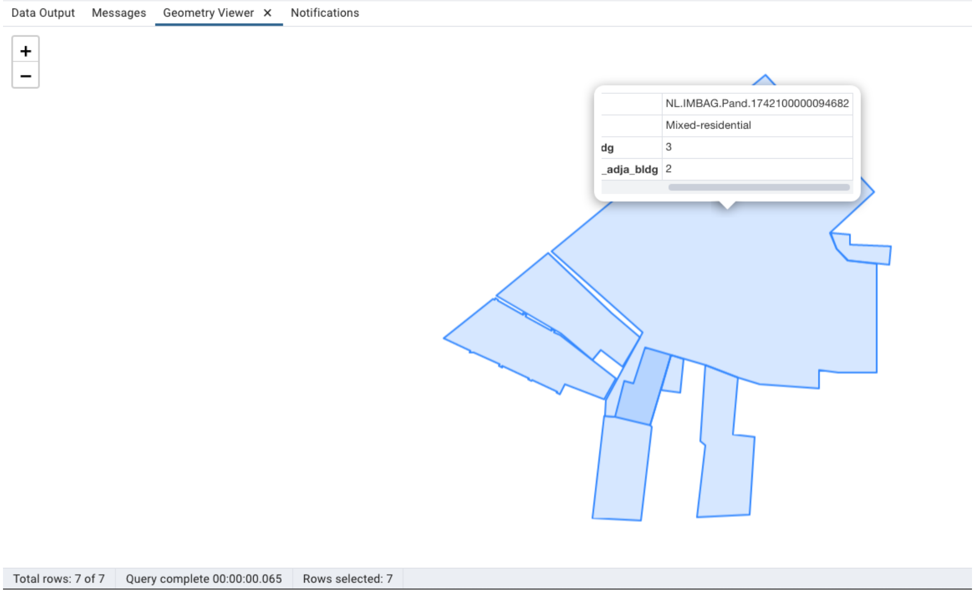
Figure 4.8: Validation of adjacency feature by visualising building footprints (2/3).



bag_id	NL.IMBAG.Pand.015
bag_function	Unknown
no_adjacent_bldg	0
no_adjacent_of_adja_bldg	1

Figure 4.9: Validation of adjacency feature by visualising building footprints (3/3).

In another example, the building 'NL.IMBAG.Pand.1742100000094682' is inspected by again visualising this building's footprint together with the footprints of its adjacent buildings. The visualisation for this inspection can be found in Figures 4.10, 4.11 and 4.12.



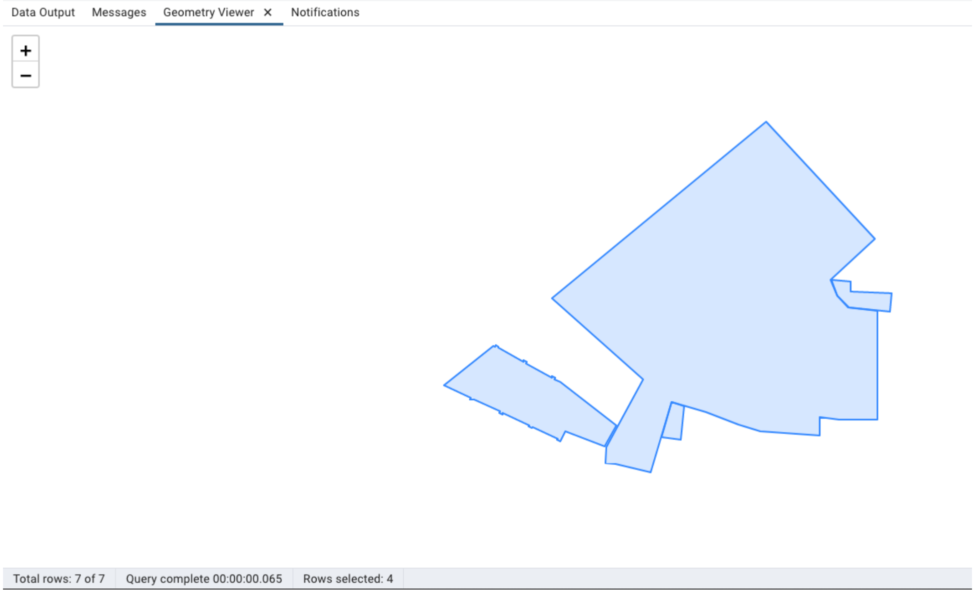
	NL.IMBAG.Pand.1742100000094682
	Mixed-residential
dg	3
_adja_bldg	2

Total rows: 7 of 7 Query complete 00:00:00.065 Rows selected: 7

Figure 4.10: Visualisation of the *big* building together with its adjacent buildings.

Figure 4.10 shows that the *big* building 'NL.IMBAG.Pand.1742100000094682' has an *adjacency* value of 3, while there are more than 3 footprints adjacent to it. This is because the leftmost building is not adjacent to the *big* building and the two rightmost buildings have an unknown function, see Figure 4.11 for the visualisation.

Observing Figures 4.10 and 4.12 reveals that one of the building footprints overlaps with the *big* building, and explains why the *big* building has a number of adjacent buildings of 3.



Total rows: 7 of 7 Query complete 00:00:00.065 Rows selected: 4

Figure 4.11: Visualisation of the *big* building together with its adjacent buildings, which are not taken as adjacent.

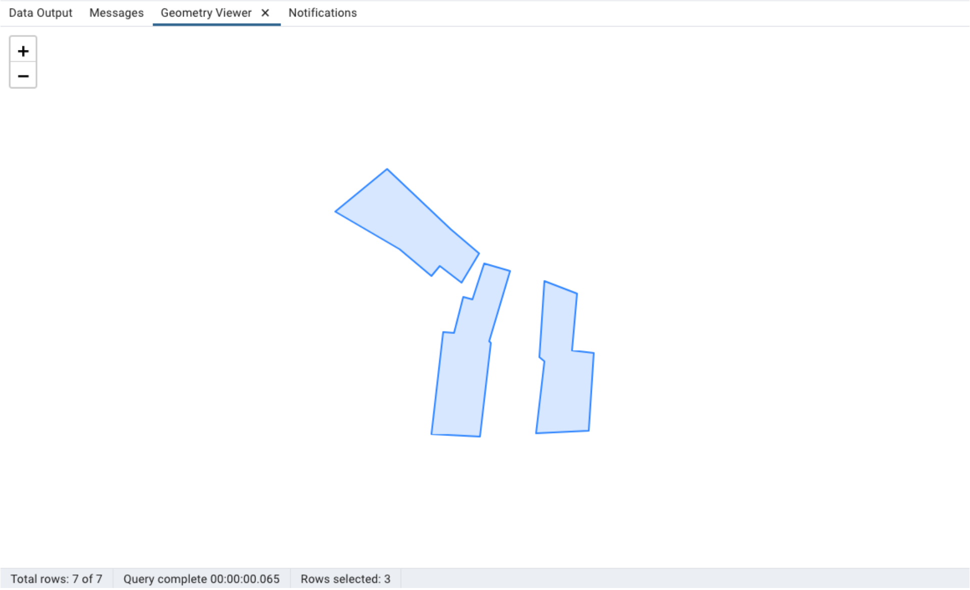


Figure 4.12: Visualisation of the buildings, which are taken as being adjacent to the *big* building.

Furthermore, in Figure 4.13 it can be observed that buildings in the same street may have a tiny space between each other. This is why it is possible for the leftmost building in Figure 4.10 to not be adjacent to the *big* building and why the buffer size of 0.10 was the right choice. However, it is still necessary to be cognizant of the cases where footprints should be taken as adjacent, but are not taken as adjacent due to inaccuracy of the drawing of the footprint where there is a gap in the drawn footprint but not in the real world.

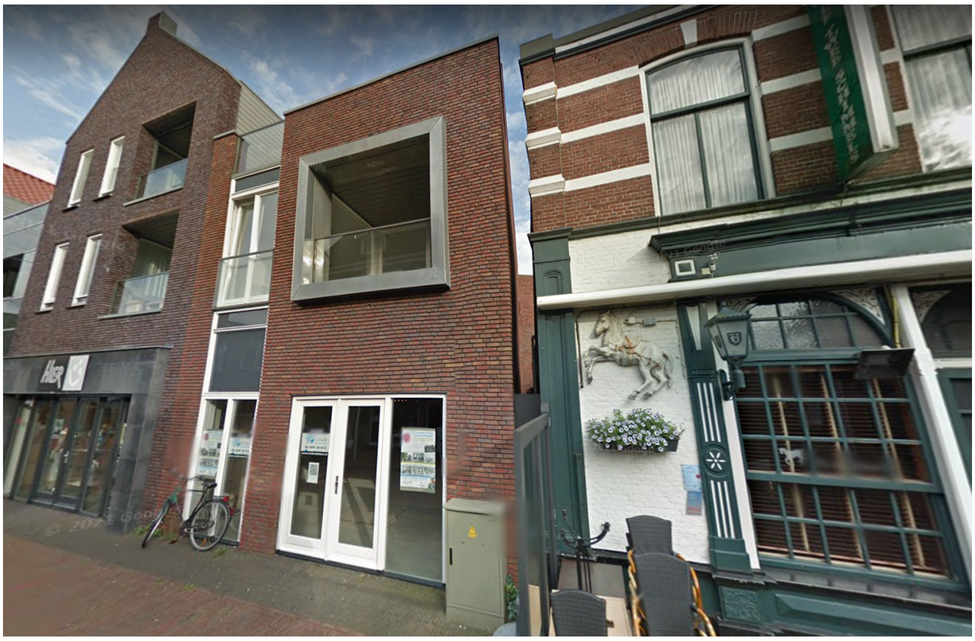


Figure 4.13: Google Street View of neighbouring buildings of the *big* building.

Neighbours

The neighbours features can be validated by querying the different distance neighbour features and by finding an example with increasingly more neighbours with each distance (see Table 4.7 for a part of the resulting table). Then, visualising that example for each of the neighbours feature distances and count the buildings in the visualisations, see Figure 4.14. The amount of neighbouring buildings in the visualisations corresponds to the amount of neighbouring buildings in the table for the example.

BAG ID	number of neighbouring buildings (radius: 25m)	number of neighbouring buildings (radius: 50m)	number of neighbouring buildings (radius: 75m)	number of neighbouring buildings (radius: 100m)
NL.IMBAG.Pand.1742100000005458	2	4	9	15
NL.IMBAG.Pand.1742100000007079	0	5	10	15
NL.IMBAG.Pand.1742100000004050	**2**	**7**	**8**	**15**
NL.IMBAG.Pand.1742100000007121	2	7	9	15

Table 4.7: Table showing a part of the results of the different distance neighbour features with the selected example highlighted in bold.

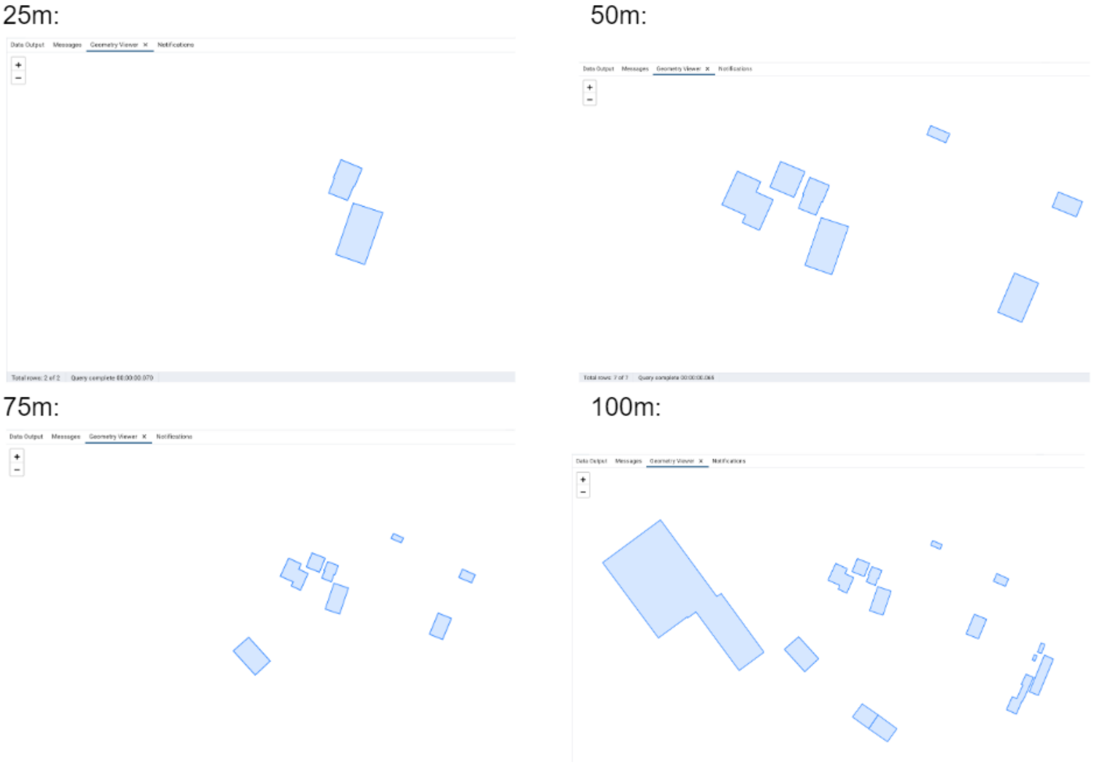


Figure 4.14: Visualisation of the example building with its neighbouring buildings for each distance.

Volume

To validate the volume features obtained from 3DBM, similar volume features are computed using PostGIS functions to compare with the 3DBM volume features. However, computing the volumes from the LoD1.2 geometries obtained from the Rijssen-Holten energy testbed dataset returned errors of invalid polygons where points are not in the same plane. The models obtained from the 3DBAG representing the same buildings were therefore validated using val3dity (2018). Some of these returned error codes after validating, however filtering these 3DBAG models with error codes out still returned errors of invalid polygons while computing the volume. This is because the PostGIS volume function relies on SFCGAL backend (Borne et al., 2023) and the SFCGAL functions related to 3D volume exhibit high sensitivity towards tolerance for coplanarity of polygons. As a result the PostGIS volume function does not take the models as input and returns an error.

Also, an unsuccessful attempt was made using PostGIS functions to make valid models. As well as triangulating the 3DBAG models, this is because PostGIS triangulation functions do not work well with vertical geometries. Hence, only a few buildings, which did not give an error of invalid polygons using the PostGIS function, were used to compare and validate the features.

Using LoD2.2 geometries resulted in all buildings giving the same error of invalid polygons. The 'Lod2_volume' attribute in the Rijssen-Holten energy testbed dataset was used instead for the comparison of LoD2.2 volumes. A part of the table for the comparison of the volumes can be found in Table 4.8 below.

BAG ID	**Actual volume LoD1 PostGIS**	**Actual volume LoD1 3DBM**	**C. hull volume LoD1 3DBM**	**Actual volume LoD2 RH**	**Actual volume LoD2 3DBM**	**C. hull volume LoD2 3DBM**
NL.IMBAG.Pand.1742100000000001	323.630	348.671	398.461	302.955	302.948	347.261
NL.IMBAG.Pand.1742100000000002	567.737	647.255	850.661	558.298	558.295	756.897
NL.IMBAG.Pand.1742100000000004	571.787	675.942	743.828	542.455	542.447	641.584
NL.IMBAG.Pand.1742100000000005	604.046	684.259	856.243	562.826	562.843	723.139
NL.IMBAG.Pand.1742100000000006	1128.359	1287.406	1707.087	1071.107	1071.077	1455.888
NL.IMBAG.Pand.1742100000000007	685.462	913.435	1285.164	643.845	643.857	1122.691
NL.IMBAG.Pand.1742100000000009	434.129	438.137	438.206	403.750	403.746	403.823
NL.IMBAG.Pand.1742100000000010	431.355	435.413	435.477	401.291	401.299	401.363
NL.IMBAG.Pand.1742100000000011	NULL	444.303	444.388	410.421	410.426	410.507
NL.IMBAG.Pand.1742100000000012	NULL	685.860	840.225	512.431	512.424	700.177

Table 4.8: Table for the comparison and validation of the volume features obtained from 3DBM.

Summary statistics are calculated for each column to compare to see if there are any significant differences, see Table 4.9 below. It can be observed that from the small sample size comparison between the *actual volume LoD1.2* computed from PostGIS and 3DBM is overall larger. While the *actual volume* from the Rijssen-Holten dataset and computed from 3DBM are almost the same.

	Actual volume PostGIS	**Actual volume LoD1 3DBM**	**C. hull volume LoD1 3DBM**	**Actual volume LoD2 RH**	**Actual volume LoD2 3DBM**	**C. hull volume LoD2 3DBM**
Mean	593.31	678.81	844.42	617.39	617.38	782.50
Median	569.76	661.60	623.51	464.46	464.45	554.60
Range	804.73	938.73	88606.09	36937.82	36937.75	100199.034
Standard Dev.	245.04	305.34	2009.20	1152.07	1152.08	2136.45
Sample size	8	8	7166	7166	7166	7166

Table 4.9: Summary statistics of each volume feature.

In Figure 4.15 the building with ID 'NL.IMBAG.Pand.1742100000000007' is depicted in Google Street View. It can be observed that this building has a garagebox and it seems that the garagebox is taken as well in the computation for this building's volume. The large volume difference between the computation with the PostGIS function and 3DBM can be explained by the different geometric algorithms to calculate volumes. Nevertheless, not enough building volumes could be calculated with the PostGIS function to compare and prove this.

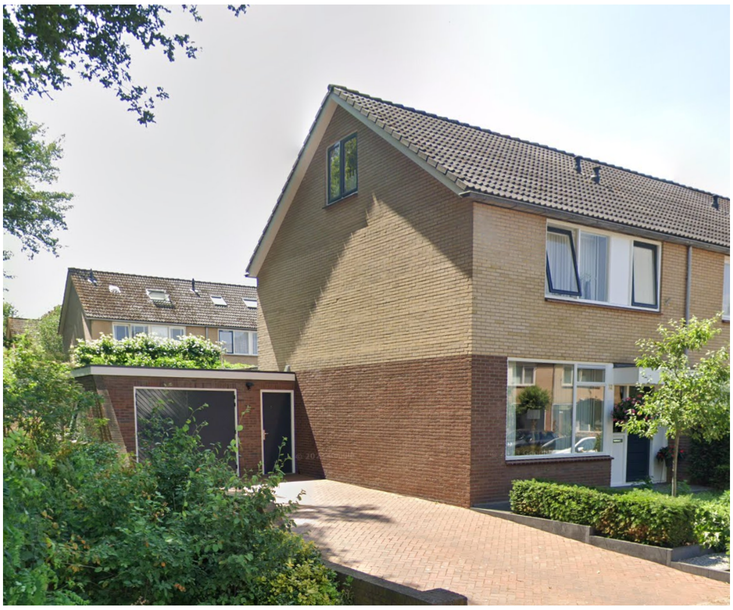


Figure 4.15: Building with ID 'NL.IMBAG.Pand.1742100000000007' with garage box.

Lastly, the way the *convex hull volume* is computed in 3DBM is shown in Figure 4.16. Several attempts were made to compute this with PostGIS functions, but these gave an error where the given geometry was unknown to the convex hull function. Even when the given geometries were aggregated into a geometry collection, which is supposed to be a correct input for the convex hull function. The computation of the convex hull of the buildings is, therefore, left outside the scope of this thesis.

It has been stated that the *actual volumes* should be smaller than the *convex hull volumes*, and never significantly larger than the *convex hull volumes*. Also, the convex hull calculation is more reliable than the actual volume calculation, since the actual volume calculation is more sensitive to errors (Labetski et al., 2022). On these notes, the volume features obtained from 3DBM are still used in the following steps.

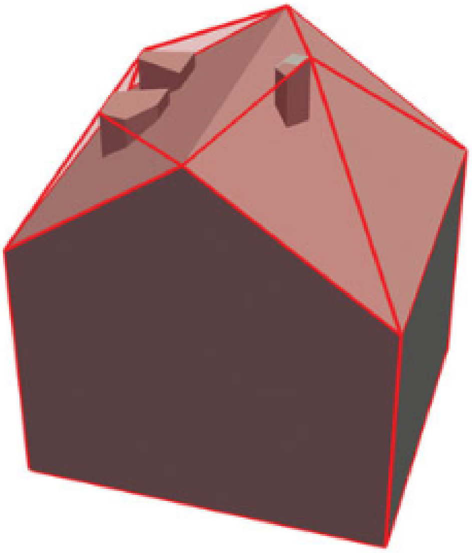


Figure 4.16: Visualisation of the convex hull volume from 3DBM.

Length and width

The *length* and *width of the oriented bounding box (obb)* for each building obtained from 3DBM are validated by multiplying the length with the width to get the area (column 'area_check'). This can then be compared with the bounding box area computed by PostGIS functions. A part of the table for this comparison can be found below in Table 4.10.

BAG ID	**obb width** **LoD1 3DBM**	**obb length** **LoD1 3DBM**	**area_check**	**area from** **bounding box**
NL.IMBAG.Pand.1742100000000001	6.486	9.876	64.049	64.080
NL.IMBAG.Pand.1742100000000002	9.780	15.354	150.164	102.282
NL.IMBAG.Pand.1742100000000004	10.156	10.566	107.317	111.455
NL.IMBAG.Pand.1742100000000005	11.140	13.135	146.316	155.425
NL.IMBAG.Pand.1742100000000006	13.140	16.397	215.448	216.373
NL.IMBAG.Pand.1742100000000007	11.313	20.907	236.516	246.673
NL.IMBAG.Pand.1742100000000009	5.981	9.062	54.199	54.213
NL.IMBAG.Pand.1742100000000010	5.975	9.063	54.150	54.166
NL.IMBAG.Pand.1742100000000011	6.005	9.063	54.425	54.726
NL.IMBAG.Pand.1742100000000012	6.264	18.461	115.632	199.668

Table 4.10: Table for validation of the oriented bounding box length and width.

Summary statistics are calculated for the area calculated from the oriented bounding box and multiplying length with width to compare to see if there are any significant differences (see Table 4.11). It can be observed that the mean difference is only 2.26 m^2. While the difference in the median is only 0.25 m^2. The difference in standard deviation is 6.79 m^2. The difference in the computed areas can be explained by the presence of possible complex footprint shapes, which results in the length multiplied by the width not being the exact area of the footprint. However, the difference is so small according to the summary statistics, hence the oriented bounding box length and width features are still used in further steps.

	area from bbox	**area check**
Mean	133.29	131.03
Median	97.80	98.05
Range	18491.43	18396.86
Standard Dev.	306.57	299.78
Sample size	7166	7166

Table 4.11: Summary statistics for area comparison to validate oriented bounding box width and length.

Surface areas

The surface areas are again validated by comparing the surface area features obtained from 3DBM with self-computed surface areas by using PostGIS functions. However, the PostGIS function which computes the area of an angled surface gave errors of invalid polygons as well. Again this is because of the SFCGAL backend (Borne et al., 2023) exhibiting sensitivity towards tolerance for coplanarity of polygons.

Therefore, the LoD2.2 surface area attributes in the Rijssen-Holten dataset were used instead. Also, the LoD1.2 surfaces could not be extracted from the Rijssen-Holten dataset since the testbed does not have a model in that level of detail, so the *LoD1.2 wall areas* could not be validated. Lastly, the *LoD1.2 roof area* was computed from LoD2.2 roof surfaces using ST_Area(), which is a PostGIS function that computes the area of a surface while ignoring the z-coordinates. A part of the table for this comparison can be found in Table 4.12 below.

BAG ID	**Wall area** **LoD1 3DBM**	**Roof area** **LoD1 3DBM**	**Roof area** **LoD1 PostGIS**	**Wall area** **LoD2 3DBM**	**Wall area** **LoD2 RH**	**Roof area** **LoD2 3DBM**	**Roof area** **LoD2 RH**
NL.IMBAG.Pand.1742100000000001	229.380	49.690	49.690	181.441	127.500	57.585	57.585
NL.IMBAG.Pand.1742100000000002	362.092	90.919	90.920	283.065	214.883	108.261	102.552
NL.IMBAG.Pand.1742100000000004	317.608	87.990	87.990	250.217	250.217	101.520	102.071
NL.IMBAG.Pand.1742100000000005	363.043	93.670	93.670	261.748	261.748	111.722	111.722
NL.IMBAG.Pand.1742100000000006	597.947	143.141	143.141	447.241	433.273	165.107	164.886
NL.IMBAG.Pand.1742100000000007	452.623	129.676	129.676	313.733	245.746	142.241	142.240
NL.IMBAG.Pand.1742100000000009	243.296	54.165	54.165	205.281	71.094	66.203	66.203
NL.IMBAG.Pand.1742100000000010	241.936	54.109	54.109	204.152	70.087	66.056	66.056
NL.IMBAG.Pand.1742100000000011	247.116	54.025	54.025	209.489	75.618	65.942	65.941
NL.IMBAG.Pand.1742100000000012	416.351	87.304	87.304	277.007	210.303	106.594	98.797

Table 4.12: Table for the comparison of the surface areas.

The summary statistics for this comparison can be found in Table 4.13. The *LoD1.2 roof area* from 3DBM is nearly identical to the LoD1.2 roof area computed with PostGIS functions. While there are noticeable differences in the *LoD2.2 wall areas* obtained from 3DBM and LoD2.2 wall areas obtained from the Rijssen-Holten dataset, the same for *LoD2.2 roof areas*. The reason for the differences in the surface areas might be because a different method is used. Investigating the methods used from both sources is however left outside the scope of this thesis. Therefore, the surface areas obtained from 3DBM are used, since one of the aims of this thesis is to investigate how well these features perform in the classification of the building stock of the Netherlands.

	Roof area LoD1 3DBM	**Roof area LoD1 PostGIS**	**Wall area LoD2 3DBM**	**Wall area LoD2 RH**	**Roof area LoD2 3DBM**	**Roof area LoD2 RH**
Mean	99.10	99.05	250.63	187.25	121.56	118.88
Median	78.85	78.85	227.30	162.83	96.94	95.06
Range	6166.43	6166.43	6197.56	6217.87	6207.50	6166.70
Standard Dev.	145.08	144.67	194.89	204.18	153.27	149.05
Sample size	7167	7167	7167	7167	7167	7167

Table 4.13: Summary statistics for the comparison of the surface areas.

Height

To validate the height features from 3DBM the height values are reverted to the raw height values of 'max_Z', 'min_Z', and 'ground_Z' from 'max_height' and 'min_roof_height'. These raw height values were extracted directly from 3DBM. And were processed using the computations below:

$$Maximum\,height\,in\,LoD1.2 = max\_z\_lod1 - ground\_z\_lod1 \tag{4.1}$$

$$Maximum\,height\,in\,LoD2.2 = max\_z\_lod2 - ground\_z\_lod2 \tag{4.2}$$

$$Height\,(without\,roof)\,in\,LoD2.2 = min\_z\_lod2 - ground\_z\_lod2 \tag{4.3}$$

The 'max_Z', 'min_Z', and 'ground_Z' correspond to the height values from the 3DBAG. 'Max_Z' corresponds to the 3DBAG's 'h_dak_max' which is the maximum elevation of the roof. 'Min_Z' corresponds to 'h_dak_min' which is the minimum elevation of the roof. While 'ground_Z' corresponds to 'h_maaiveld' which is the ground elevation. Table 4.14 shows part of the comparison of the height values from 3DBM and 3DBAG.

BAG ID	**h_dak _max**	**max_z _lod2**	**h_dak _min**	**min_z _lod2**	**h_ maaiveld**	**ground_z _lod1**	**ground_z _lod2**	**h_dak _70p**	**max_z _lod1**	**min_z _lod1**
NL.IMBAG.Pand.1742100000000001	20.486	20.430	15.669	15.462	12.712	12.712	12.712	19.705	19.729	19.729
NL.IMBAG.Pand.1742100000000002	22.123	22.052	16.757	16.760	13.876	13.876	13.876	21.030	20.995	20.995
NL.IMBAG.Pand.1742100000000003	17.969	17.922	16.702	16.710	14.084	14.084	14.084	17.536	17.521	17.521
NL.IMBAG.Pand.1742100000000004	24.077	24.048	18.585	18.656	14.983	14.983	14.983	22.652	22.665	22.665
NL.IMBAG.Pand.1742100000000005	23.458	23.433	17.761	17.813	14.956	14.956	14.956	22.311	22.261	22.261
NL.IMBAG.Pand.1742100000000006	27.233	27.167	21.209	21.383	16.895	16.895	16.895	25.903	25.889	25.889
NL.IMBAG.Pand.1742100000000007	24.715	24.682	18.243	18.353	15.445	15.445	15.445	22.514	22.489	22.489

Table 4.14: Table for the height validation.

Summary statistics are computed for this comparison as well and can be found in Table 4.15. From this table, it can be observed that 'h_dak_max' is similar to 'max_z_lod2', since the summary metrics only differ from each other by a decimal value. The same for 'h_dak_min' and 'min_z_lod2'. Also, the same for 'h_maaiveld', 'ground_z_lod1' and 'ground_z_lod2'. And, as well as for 'h_dak_70p', 'max_z_lod1' and 'min_z_lod1'. Therefore, the height values of 3DBM have been validated and are ready to be used in the following steps.

	h_dak _max	**max_z _lod2**	**h_dak _min**	**min_z _lod2**	**h_ maaiveld**	**ground_z _lod1**	**ground_z _lod2**	**h_dak _70p**	**max_z _lod1**	**min_z _lod1**
Mean	21.34	21.20	17.62	17.75	14.57	14.60	14.70	20.28	20.26	20.26
Median	20.16	20.04	15.56	15.64	12.53	12.55	12.56	18.81	18.73	18.73
Mode	19.61	19.72	13.23	13.52	10.72	10.72	9.64	14.93	17.94	17.94
Range	44.05	43.92	45.54	45.69	27.79	27.79	27.79	44.40	44.40	44.40
Standard Dev.	5.88	5.98	5.75	5.84	5.54	5.54	5.63	5.79	5.80	5.80
Sample size	18330	18330	18330	18330	18330	18330	18330	18330	18330	18330

Table 4.15: Summary statistics for the comparison of the height values.

4.3.2 Analysis

Below is the data analysis for **Case Study 1** in which the ground truth was acquired from the Rijssen-Holten energy testbed dataset (León-Sánchez et al., 2022). The same data analysis was performed for **Case Study 2** in which the ground truth was acquired from EP-online (2023). The figures from this data analysis for **Case Study 2** can be found in the Appendix (see Figure A.1, A.2 and A.3).

Correlation between attributes

The relationship between two variables is called correlation. The Pearson coefficient captures the linear relationships between two variables (Chandrashekar & Sahin, 2014). In Figure 4.17 the Pearson's Correlation Coefficient between the features in the acquired dataset of Rijssen-Holten is shown. A coefficient value of 1 represents a full positive correlation between the variables (for example, the yellow diagonal line formed by the correlation of a variable with itself), a 0 represents no correlation at all between the variables and -1 represents a full negative correlation between variables.

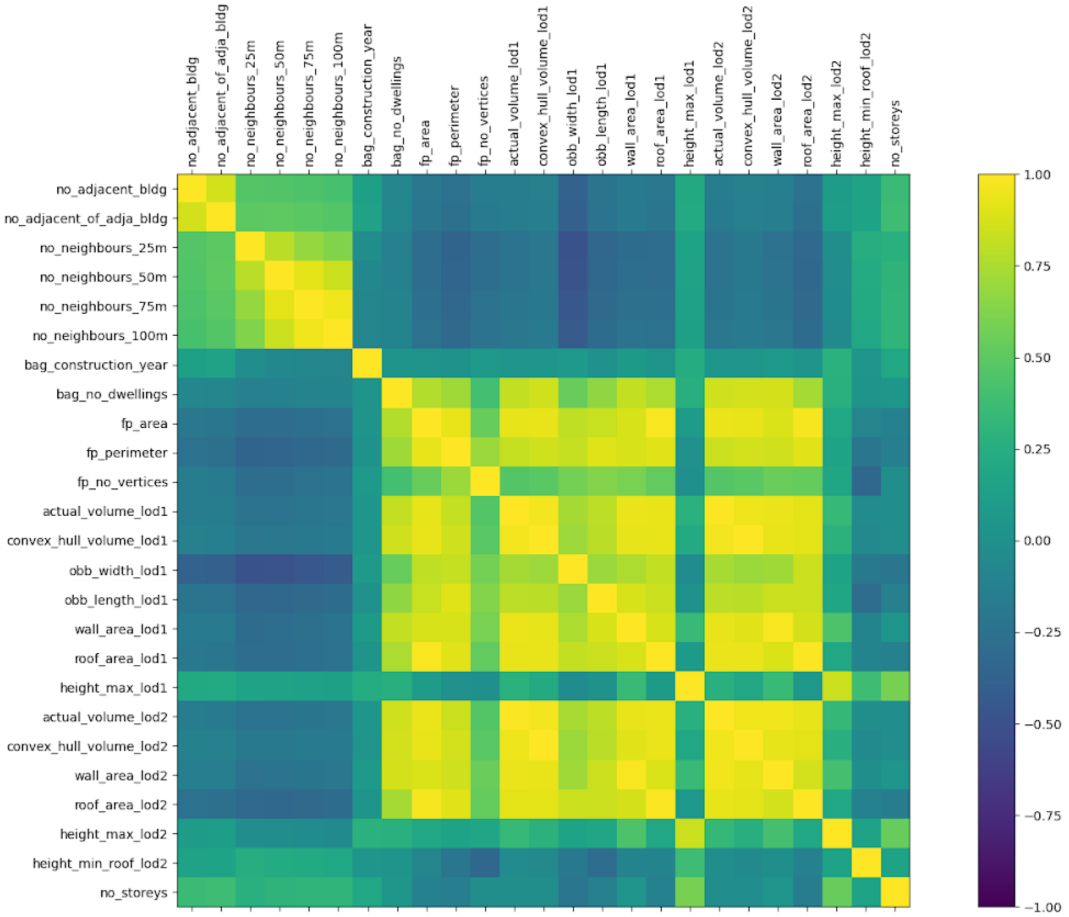
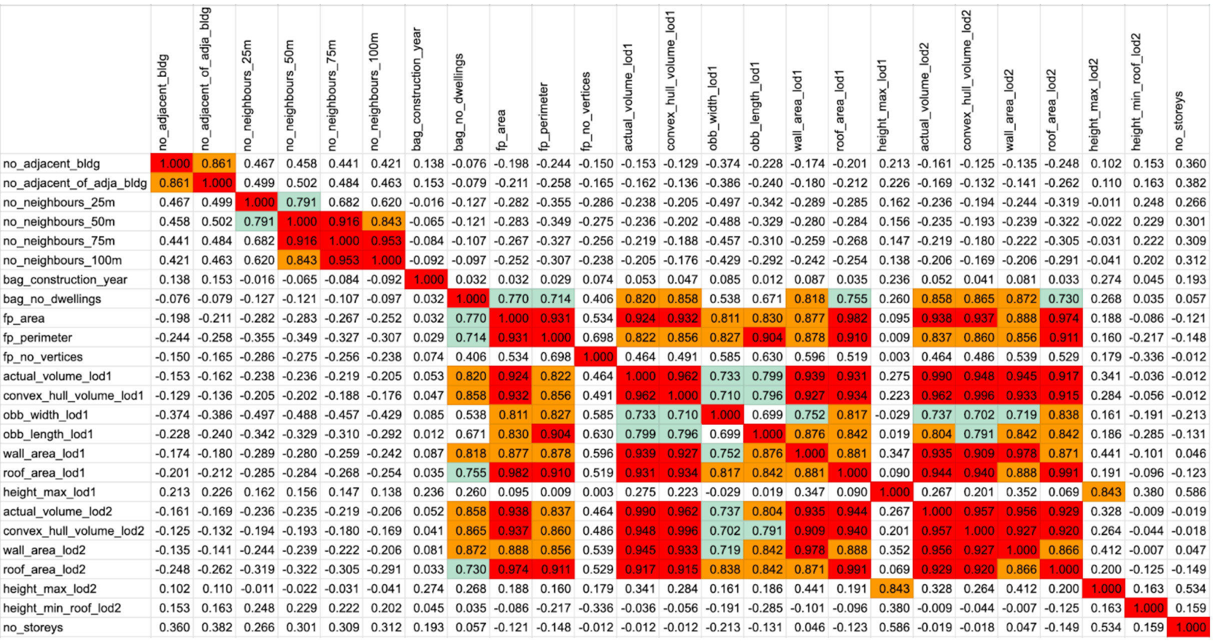


Figure 4.17: Correlation matrix.

Table 4.18 shows the same correlation, but expressed in values instead of colours. From these correlation matrices, it can be observed that several features are highly correlated, namely, the *number of neighbouring buildings* in the radii of 50 metres, 75 metres and 100 metres. Also, features that describe the form of the building, such as the *footprint area*, *footprint perimeter* and the features with different levels of detail (LoD1 or LoD2) specifically the *actual volume*, the *convex hull volume*, the *wall area*, the *roof area* and the *maximum height*. Based on these findings alone it can be concluded that usage of both LoD1 and LoD2 for certain features and the two types of volumes are not necessary. As a result, the following features are removed: *actual volume in LoD1*, *convex hull volume in LoD1*, *convex hull volume in LoD2*, *wall area in LoD1*, *roof area in LoD1* and *maximum height in LoD1*.



	no_adjacent_bldg	no_adjacent_of_adja_bldg	no_neighbours_25m	no_neighbours_50m	no_neighbours_75m	no_neighbours_100m	bag_construction_year	bag_no_dwellings	fp_area	fp_perimeter	fp_no_vertices	actual_volume_lod1	convex_hull_volume_lod1	obb_width_lod1	obb_length_lod1	wall_area_lod1	roof_area_lod1	height_max_lod1	actual_volume_lod2	convex_hull_volume_lod2	wall_area_lod2	roof_area_lod2	height_max_lod2	height_min_roof_lod2	no_storeys
no_adjacent_bldg	1.000	0.861	0.467	0.458	0.441	0.421	0.138	-0.076	-0.198	-0.244	-0.150	-0.153	-0.129	-0.374	-0.228	-0.174	-0.201	0.213	-0.161	-0.125	-0.135	-0.248	0.102	0.153	0.360
no_adjacent_of_adja_bldg	0.861	1.000	0.499	0.502	0.484	0.463	0.153	-0.079	-0.211	-0.258	-0.165	-0.162	-0.136	-0.386	-0.240	-0.180	-0.212	0.226	-0.169	-0.132	-0.141	-0.262	0.110	0.163	0.382
no_neighbours_25m	0.467	0.499	1.000	0.791	0.682	0.620	-0.016	-0.127	-0.282	-0.355	-0.286	-0.238	-0.205	-0.497	-0.342	-0.289	-0.285	0.162	-0.236	-0.194	-0.244	-0.319	-0.011	0.248	0.266
no_neighbours_50m	0.458	0.502	0.791	1.000	0.916	0.843	-0.065	-0.121	-0.283	-0.349	-0.275	-0.236	-0.202	-0.488	-0.329	-0.280	-0.284	0.156	-0.235	-0.193	-0.239	-0.322	-0.022	0.229	0.301
no_neighbours_75m	0.441	0.484	0.682	0.916	1.000	0.953	-0.084	-0.107	-0.267	-0.327	-0.256	-0.219	-0.188	-0.457	-0.310	-0.259	-0.268	0.147	-0.219	-0.180	-0.222	-0.305	-0.031	0.222	0.309
no_neighbours_100m	0.421	0.463	0.620	0.843	0.953	1.000	-0.092	-0.097	-0.252	-0.307	-0.238	-0.205	-0.176	-0.429	-0.292	-0.242	-0.254	0.138	-0.206	-0.169	-0.206	-0.291	-0.041	0.202	0.312
bag_construction_year	0.138	0.153	-0.016	-0.065	-0.084	-0.092	1.000	0.032	0.032	0.029	0.074	0.053	0.047	0.085	0.012	0.087	0.035	0.236	0.052	0.041	0.081	0.033	0.274	0.045	0.193
bag_no_dwellings	-0.076	-0.079	-0.127	-0.121	-0.107	-0.097	0.032	1.000	0.770	0.714	0.406	0.820	0.858	0.538	0.671	0.818	0.755	0.260	0.858	0.865	0.872	0.730	0.268	0.035	0.057
fp_area	-0.198	-0.211	-0.282	-0.283	-0.267	-0.252	0.032	0.770	1.000	0.931	0.534	0.924	0.932	0.811	0.830	0.877	0.982	0.095	0.938	0.937	0.888	0.974	0.188	-0.086	-0.121
fp_perimeter	-0.244	-0.258	-0.355	-0.349	-0.327	-0.307	0.029	0.714	0.931	1.000	0.698	0.822	0.856	0.827	0.904	0.878	0.910	0.009	0.837	0.860	0.856	0.911	0.160	-0.217	-0.148
fp_no_vertices	-0.150	-0.165	-0.286	-0.275	-0.256	-0.238	0.074	0.406	0.534	0.698	1.000	0.464	0.491	0.585	0.630	0.596	0.519	0.003	0.464	0.486	0.539	0.529	0.179	-0.336	-0.012
actual_volume_lod1	-0.153	-0.162	-0.238	-0.236	-0.219	-0.205	0.053	0.820	0.924	0.822	0.464	1.000	0.962	0.733	0.799	0.939	0.931	0.275	0.990	0.948	0.945	0.917	0.341	-0.036	-0.012
convex_hull_volume_lod1	-0.129	-0.136	-0.205	-0.202	-0.188	-0.176	0.047	0.858	0.932	0.856	0.491	0.962	1.000	0.710	0.796	0.927	0.934	0.223	0.962	0.996	0.933	0.915	0.284	-0.056	-0.012
obb_width_lod1	-0.374	-0.386	-0.497	-0.488	-0.457	-0.429	0.085	0.538	0.811	0.827	0.585	0.733	0.710	1.000	0.699	0.752	0.817	-0.029	0.737	0.702	0.719	0.838	0.161	-0.191	-0.213
obb_length_lod1	-0.228	-0.240	-0.342	-0.329	-0.310	-0.292	0.012	0.671	0.830	0.904	0.630	0.799	0.796	0.699	1.000	0.876	0.842	0.019	0.804	0.791	0.842	0.842	0.186	-0.285	-0.131
wall_area_lod1	-0.174	-0.180	-0.289	-0.280	-0.259	-0.242	0.087	0.818	0.877	0.878	0.596	0.939	0.927	0.752	0.876	1.000	0.881	0.347	0.935	0.909	0.978	0.871	0.441	-0.101	0.046
roof_area_lod1	-0.201	-0.212	-0.285	-0.284	-0.268	-0.254	0.035	0.755	0.982	0.910	0.519	0.931	0.934	0.817	0.842	0.881	1.000	0.090	0.944	0.940	0.888	0.991	0.191	-0.096	-0.123
height_max_lod1	0.213	0.226	0.162	0.156	0.147	0.138	0.236	0.260	0.095	0.009	0.003	0.275	0.223	-0.029	0.019	0.347	0.090	1.000	0.267	0.201	0.352	0.069	0.843	0.380	0.586
actual_volume_lod2	-0.161	-0.169	-0.236	-0.235	-0.219	-0.206	0.052	0.858	0.938	0.837	0.464	0.990	0.962	0.737	0.804	0.935	0.944	0.267	1.000	0.957	0.956	0.929	0.328	-0.009	-0.019
convex_hull_volume_lod2	-0.125	-0.132	-0.194	-0.193	-0.180	-0.169	0.041	0.865	0.937	0.860	0.486	0.948	0.996	0.702	0.791	0.909	0.940	0.201	0.957	1.000	0.927	0.920	0.264	-0.044	-0.018
wall_area_lod2	-0.135	-0.141	-0.244	-0.239	-0.222	-0.206	0.081	0.872	0.888	0.856	0.539	0.945	0.933	0.719	0.842	0.978	0.888	0.352	0.956	0.927	1.000	0.866	0.412	-0.007	0.047
roof_area_lod2	-0.248	-0.262	-0.319	-0.322	-0.305	-0.291	0.033	0.730	0.974	0.911	0.529	0.917	0.915	0.838	0.842	0.871	0.991	0.069	0.929	0.920	0.866	1.000	0.200	-0.125	-0.149
height_max_lod2	0.102	0.110	-0.011	-0.022	-0.031	-0.041	0.274	0.268	0.188	0.160	0.179	0.341	0.284	0.161	0.186	0.441	0.191	0.843	0.328	0.264	0.412	0.200	1.000	0.163	0.534
height_min_roof_lod2	0.153	0.163	0.248	0.229	0.222	0.202	0.045	0.035	-0.086	-0.217	-0.336	-0.036	-0.056	-0.191	-0.285	-0.101	-0.096	0.380	-0.009	-0.044	-0.007	-0.125	0.163	1.000	0.159
no_storeys	0.360	0.382	0.266	0.301	0.309	0.312	0.193	0.057	-0.121	-0.148	-0.012	-0.012	-0.012	-0.213	-0.131	0.046	-0.123	0.586	-0.019	-0.018	0.047	-0.149	0.534	0.159	1.000

Figure 4.18: Correlation matrix with exact coefficient values.

Distribution of each feature

A method to visualise the distribution of each feature in the acquired Rijssen-Holten training dataset is the usage of histograms. It provides the count of the number of observations in each bin created for the visualisation, which allows the interpretation of the type of distribution.

See Figure 4.19, it can be observed that the *number of neighbouring buildings* (25m, 50m, 75m and 100m) may have a normal distribution, while the *footprint perimeter*, *footprint number of vertices*, *bounding box length and width*, and *roof and wall area* may have a skewed distribution.

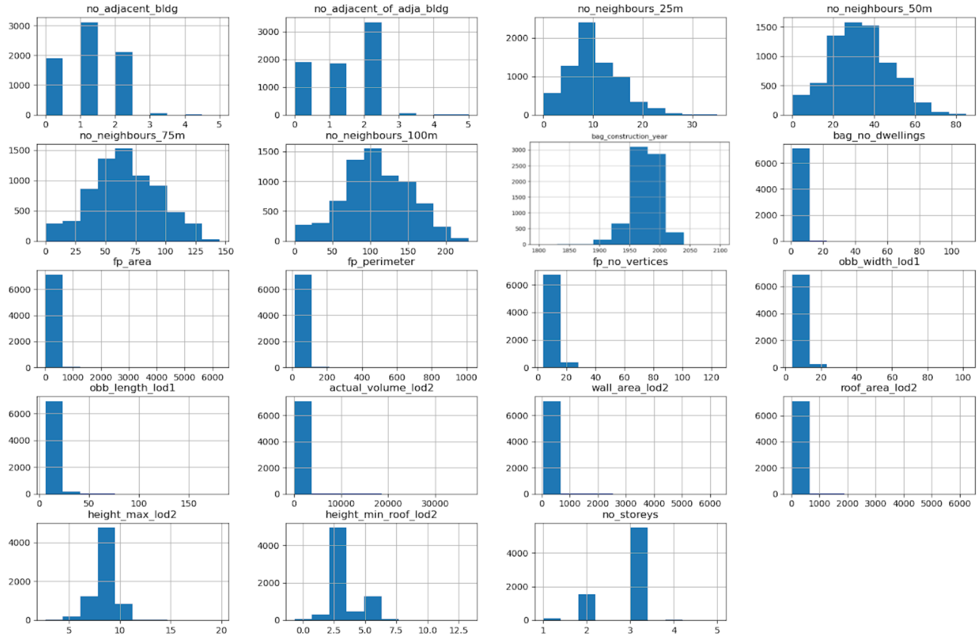


Figure 4.19: Histograms of the features of the acquired training dataset.

In many modelling scenarios, normal distributed features in the dataset are desirable. To map data from any distribution to as close to a normal distribution as possible power transforms can be used (Scikit-learn, 2023h), which are a family of parametric, monotonic transformations. This will in turn stabilise the variance and minimise the skewness. However, this is not necessary for support vector machines and random forests. Nonetheless, it is important to scale the data for support vector machines as the algorithm is sensitive to the scale of the features. If the features have different scales, then the SVM may be biassed towards features with larger values (Géron, 2019; Goodfellow et al., 2013).

4.3.3 Selection

Feature selection methods are techniques used to identify the most useful input variables for a machine learning model to predict the target variable. In some predictive modelling problems, there may be a large number of input variables that can slow down model development and require excessive system memory. Moreover, including irrelevant input variables may adversely affect the performance of some models. Feature selection methods can be divided into three categories: filter, embedded and wrapper methods (Kuhn & Johnson, 2013; Chandrashekar & Sahin, 2014).

Filter methods use statistical techniques to evaluate the relationship between each input variable and the target variable. The scores obtained are then used to select (filter) the input variables to be used in the model. On the other hand, some machine learning algorithms have built-in feature selection methods that automatically select the most relevant input variables as part of the learning process (such as random forests). These methods are known as embedded feature selection methods. Lastly, wrapper methods create multiple models with different subsets of input features and choose the best-performing model based on a performance metric. These methods are not concerned with the variable types but can be computationally expensive.

For these selection methods, the remaining highly correlated features are removed based on their rankings and the correlation coefficient values (see **Correlation between attributes** in Section 4.3.2) and the highest ranked features will be kept.

Filter

For the filter methods, the **ANOVA-F** score and the **Mutual Information** score are used, since these are recommended for classification problems (Scikit-learn, 2023b). Analysis of Variance (ANOVA) is a statistical method used to check the means of two or more groups that are significantly different from each other. While the Mutual Information value measures the dependency between variables. See Table 4.16, the best scoring features are then selected while also considering their correlations with each other. See Table 4.17 for the selected features with the ANOVA-f method and the MI (Mutual Information) method for **Case Study 1**. The same selection process was performed for **Case Study 2** and those tables can be found in the Appendix (see Table A.1 and A.2).

feature	name	score
0	no_adjacent_bldg	4709.01
1	no_adjacent_of_adja_bldg	3902.59
7	bag_no_dwellings	985.03
14	wall_area_lod2	930.18
13	actual_volume_lod2	766.27
8	fp_area	474.81
11	obb_width_lod1	474.05
15	roof_area_lod2	442.78
9	fp_perimeter	417.50
12	obb_length_lod1	402.23
3	no_neighbours_50m	307.44
2	no_neighbours_25m	300.68
4	no_neighbours_75m	283.17
5	no_neighbours_100m	252.84
18	no_storeys	184.79
10	fp_no_vertices	147.01
16	height_max_lod2	128.25
17	height_min_roof_lod2	56.88
6	bag_construction_year	48.55

feature	name	score
0	no_adjacent_bldg	0.816
1	no_adjacent_of_adja_bldg	0.759
11	obb_width_lod1	0.286
15	roof_area_lod2	0.252
8	fp_area	0.235
2	no_neighbours_25m	0.201
13	actual_volume_lod2	0.193
3	no_neighbours_50m	0.174
9	fp_perimeter	0.173
6	bag_construction_year	0.167
4	no_neighbours_75m	0.163
5	no_neighbours_100m	0.148
12	obb_length_lod1	0.120
16	height_max_lod2	0.112
14	wall_area_lod2	0.098
18	no_storeys	0.097
17	height_min_roof_lod2	0.085
7	bag_no_dwellings	0.077
10	fp_no_vertices	0.067

Table 4.16: ANOVA-F score (left) and MI score (right) of the features.

feature	name
0	no_adjacent_bldg
1	no_adjacent_of_adja_bldg
7	bag_no_dwellings
14	wall_area_lod2
13	actual_volume_lod2
11	obb_width_lod1
3	no_neighbours_50m

feature	name
0	no_adjacent_bldg
1	no_adjacent_of_adja_bldg
11	obb_width_lod1
8	fp_area
2	no_neighbours_50m
13	actual_volume_lod2
6	bag_construction_year

Table 4.17: ANOVA-F features (left) & MI features (right)

Embedded

For the embedded method the **Random Forest impurity-based feature importances** are computed. However, these importances can be misleading for high cardinality features (many unique values) (Scikit-learn, 2023g). This leads to impurity-based feature importances for trees, which are strongly biassed and favour high cardinality features. However, the top features for this embedded method, while considering their correlations, are still used in Random Forest to compare with the features from the wrapper method described in the next paragraph. See Table 4.18 for the best-scoring features based on impurity and the selected features. Their **Case Study 2** counterparts can be found in the Appendix (see Table A.3).

feature	name	score
0	no_adjacent_bldg	0.314
1	no_adjacent_of_adja_bldg	0.296
11	obb_width_lod1	0.050
15	roof_area_lod2	0.041
8	fp_area	0.033
6	bag_construction_year	0.031
13	actual_volume_lod2	0.028
16	height_max_lod2	0.024
5	no_neighbours_100m	0.021
2	no_neighbours_25m	0.021
9	fp_perimeter	0.020
14	wall_area_lod2	0.020
4	no_neighbours_75m	0.019
3	no_neighbours_50m	0.019
17	height_min_roof_lod2	0.019
12	obb_length_lod1	0.018
10	fp_no_vertices	0.011
7	bag_no_dwellings	0.009
18	no_storeys	0.008

feature	name
0	no_adjacent_bldg
1	no_adjacent_of_adja_bldg
11	obb_width_lod1
6	bag_construction_year
13	actual_volume_lod2
16	height_max_lod2
2	no_neighbours_25m

Table 4.18: Random Forest impurity-based feature importances (left) & impurity features (right).

Wrapper

Random Forest permutation-based feature importances are used since it does not exhibit a bias towards high cardinality features. Permutation feature importance is a technique that measures the impact of individual features on a model's performance. The method involves randomly shuffling the values of a single feature, thereby breaking the relationship between that feature and the target variable. The resulting drop in model score indicates how much the model relies on that feature. This technique is model-agnostic and can be calculated multiple times with various permutations of the feature, making it a useful tool for evaluating feature importance in machine learning models.

In Table 4.19 below the ranking of the extracted features based on feature importance and the selected features are shown. The features for the wrapper are manually selected, since apart from the adjacency features, all other features have a minimal decrease in accuracy score when removed. The selection of the features for this method is highly based on the correlation between the features. See Table A.4 in the Appendix for **Case Study 2** rankings and selected features.

feature	name	ranking
0	no_adjacent_bldg	1
1	no_adjacent_of_adja_bldg	2
7	bag_no_dwellings	3
6	bag_construction_year	4
14	wall_area_lod2	5
11	obb_width_lod1	6
5	no_neighbours_100m	7
12	obb_length_lod1	8
8	fp_area	9
13	actual_volume_lod2	10
15	roof_area_lod2	11
17	height_min_roof_lod2	12
10	fp_no_vertices	13
16	height_max_lod2	14
2	no_neighbours_25m	15
9	fp_perimeter	16
3	no_neighbours_50m	17
18	no_storeys	18
4	no_neighbours_75m	19

feature	name
0	no_adjacent_bldg
1	no_adjacent_of_adja_bldg
7	bag_no_dwellings
6	bag_construction_year
11	obb_width_lod1
14	wall_area_lod2
2	no_neighbours_25m

Table 4.19: Random Forest permutation-based feature importances (left) & permutation features (right)

4.3.4 Hyperparameter tuning

Hyperparameter tuning is performed using a randomised grid search over 75 different candidate parameter combinations. To gain an understanding of the range of the search for each hyperparameter a validation curve has been plotted. These curves are obtained by isolating one hyperparameter and giving it a range to search on. The performance of each isolated hyperparameter is then evaluated by a 3-fold cross-validation using the accuracy metric. However, the results of these curves are not representative of the final model performance, since the hyperparameters are tuned separately in the validation curves. But it gives an indication of the search range for hyperparameter tuning for the best performances. Balanced accuracy has also been considered since all classes are important for this classification. But, the class imbalance for this case study might be too severe.

SVM

Linear SVM has 7 hyperparameters (default values are in **bold**):

- **Penalty**: specifies the norm in penalization. (‘l1’ or **‘l2’**)
- **Loss**: specifies the loss function. (‘hinge’ or **‘squared_hinge’**)
- **Dual**: selects the algorithm to either solve the dual or primal optimization problem, dual=False is preferred when n_samples > n_features. (**True** or False)
- **Tol**: tolerance for stopping criteria (float, **1e-4**)
- **C**: Regularisation parameter. The strength of the regularisation is inversely proportional to C. Must be strictly positive. (float, **1.0**)
- **Class_weight**: Gives a weight to each class. The ‘balanced’ mode uses the values of y (target) to automatically adjust weights inversely proportional to class frequencies in the input data. (dict or ‘balanced’ or **None**)
- **Max_iter**: The maximum number of iterations to be run. (int, **1000**)

A search range of 1 to 10 with steps of 1 was defined for the tolerance (tol). While the regularisation parameter (C) had a search range of 2.0 to 11.0 with steps of 1.0. And, the search range for the maximum number of iterations was set to 1000 to 5000 with steps of 500. See Figure A.4 for the validation curves using features selected with the ANOVA-F method and Figure A.5 using MI features both can be found in the Appendix.

Random Forest

Random Forest Classifier has 8 hyperparameters (default values are in **bold**):

- **N_estimators**: The number of trees in the forest (int, **100**)
- **Criterion**: The function to measure the quality of a split. (**'gini'**, 'entropy', 'log_loss')
- **Max_depth**: The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. (int, **None**)
- **Min_samples_split**: The minimum number of samples required to split an internal node. (int or float, **2**)
- **Min_samples_leaf**: The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches. (int or float, **1**)
- **Max_features**: The number of features to consider when looking for the best split (**'sqrt'**, 'log2', None or int or float)
- **Bootstrap**: Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree. (**True** or False)
- **Class_weight**: Give a weight to each class. The 'balanced' mode uses the values of y (target) to automatically adjust weights inversely proportional to class frequencies in the input data. The 'balanced_subsample' mode is the same as 'balanced' except that weights are computed based on the bootstrap sample for every tree grown (dict or 'balanced' or 'balanced_subsample' or **None**)

A search range of 100 to 1000 with steps of 100 was defined for the validation curve of the number of trees in the forest (n_estimators). The maximum depth of the tree (max_depth) had a search range of 5 to 55 with steps of 5. And, the minimum number of samples required to split an internal node (min_samples_split) had a search range of 2 to 62 with steps of 4. Furthermore, the search range for the minimum number of samples required to be at a leaf node (min_samples_leaf) was defined at 0 to 50 with steps of 5. Lastly, the search range for the number of features to consider for the best split (max_features) was 1 to 10 with steps of 1. See Figure A.6 for the validation curves using the impurity-based features and Figure A.7 using the permutation-based features, both also in the Appendix.

5 | Results

For each of **Case Study 1** and **Case Study 2**, four models were trained, resulting in a total of eight models obtained. See Table 5.1 below for the 8 different models and their given model name.

	Machine Learning algorithm	Features selection method	Model name
Case Study 1	Support Vector Machine	ANOVA-F	c1 SVM ANOVA-F
	Support Vector Machine	Mutual Information	c1 SVM MI
	Random Forest	Impurity-based	c1 RF impurity
	Random Forest	Permutation-based	c1 RF permutation
Case Study 2	Support Vector Machine	ANOVA-F	c2 SVM ANOVA-F
	Support Vector Machine	Mutual Information	c2 SVM MI
	Random Forest	Impurity-based	c2 RF impurity
	Random Forest	Permutation-based	c2 RF permutation

Table 5.1: The prediction models obtained for this thesis with their given name.

To evaluate the performance of each model on their test splits a confusion matrix was used and in turn the Precision, Recall and F1-score were computed. The weighted average Recall is equal to the accuracy, while the macro average Recall is equal to the balanced accuracy. The full results with tuning time, training time, best parameters, features used, confusion matrix and the performance metrics can be found in the Appendix (See **Case Study 1** and **Case Study 2** in Section A.1).

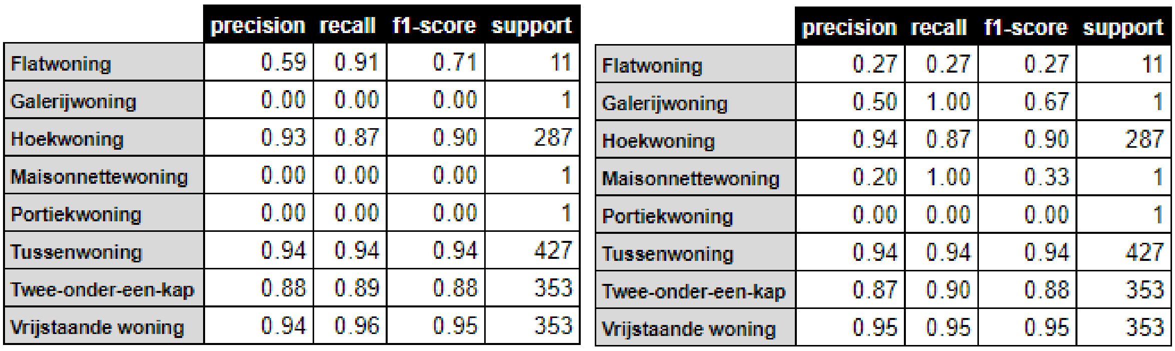
In this chapter the summary of these results are discussed. First **Case Study 1 and 2** will be addressed in Section 5.1 and 5.2. Then a comparison will be made between **Case Study 1 and 2** results in Section 5.3, including the model performance of each case study on the other. Lastly, both c1 and c2 model performances on **Case Studies 3 through 8** are compared in Section 5.4.

5.1 Case Study 1

In Table 5.2 a summary of the performance of the training models for **Case Study 1** on their test splits is given. It can be observed that the accuracy for these training models is high, over 91%. However, the balanced accuracies are significantly lower than the accuracies. This can be explained by Figure 5.1, the classification report for the **c1 SVM MI model** and the **c1 RF impurity model** shows the effects of an imbalance class distribution described in Section 4.1.

Model	Tuning time (s)	Training time (s)	Accuracy	Balanced accuracy
SVM ANOVA-F	25.2	1.42	0.918	0.583
SVM MI	20.43	0.35	0.917	0.571
RF impurity	426.31	2.87	0.911	0.741
RF permutation	344.27	2.48	0.918	0.652

Table 5.2: Summary of **Case Study 1** model performances.



	precision	recall	f1-score	support
Flatwoning	0.59	0.91	0.71	11
Galerijwoning	0.00	0.00	0.00	1
Hoekwoning	0.93	0.87	0.90	287
Maisonnettewoning	0.00	0.00	0.00	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.94	0.94	0.94	427
Twee-onder-een-kap	0.88	0.89	0.88	353
Vrijstaande woning	0.94	0.96	0.95	353

	precision	recall	f1-score	support
Flatwoning	0.27	0.27	0.27	11
Galerijwoning	0.50	1.00	0.67	1
Hoekwoning	0.94	0.87	0.90	287
Maisonnettewoning	0.20	1.00	0.33	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.94	0.94	0.94	427
Twee-onder-een-kap	0.87	0.90	0.88	353
Vrijstaande woning	0.95	0.95	0.95	353

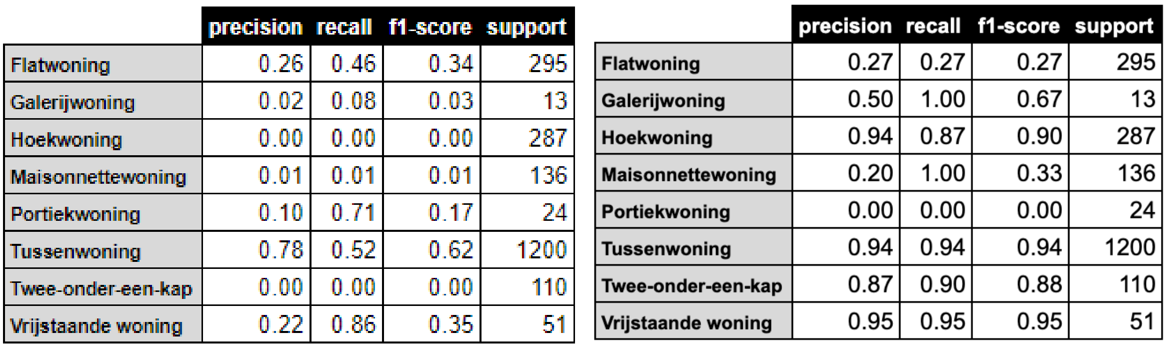
Figure 5.1: Classification report for the **c1 SVM MI model** (left) and for the **c1 RF impurity model** (right).

5.2 Case Study 2

Therefore **Case Study 2** was created, Table 5.3 shows the summary of the performance of the training models for **Case Study 2** on their test splits. And, in Figure 5.2 the classification report for the **c2 SVM MI model** and the **c2 RF impurity model** for this case can be found. The accuracy for these models is noticeably lower than the accuracy of **Case Study 1** models. However, the balanced accuracies for the **c2 RF models** are higher. In the classification report, it can be observed that the sample sizes are at least higher than 1, which means the evaluation of these models for these classes is at least more reliable than the **Case Study 1** models.

Model	Tuning time (s)	Training time (s)	Accuracy	Balanced accuracy
SVM ANOVA-F	112.9	7.04	0.672	0.395
SVM MI	240.39	8.75	0.389	0.330
RF impurity	996.91	14.16	0.711	0.737
RF permutation	1105.94	17.31	0.733	0.737

Table 5.3: Summary of **Case Study 2** model performances.



	precision	recall	f1-score	support
Flatwoning	0.26	0.46	0.34	295
Galerijwoning	0.02	0.08	0.03	13
Hoekwoning	0.00	0.00	0.00	287
Maisonnettewoning	0.01	0.01	0.01	136
Portiekwoning	0.10	0.71	0.17	24
Tussenwoning	0.78	0.52	0.62	1200
Twee-onder-een-kap	0.00	0.00	0.00	110
Vrijstaande woning	0.22	0.86	0.35	51

	precision	recall	f1-score	support
Flatwoning	0.27	0.27	0.27	295
Galerijwoning	0.50	1.00	0.67	13
Hoekwoning	0.94	0.87	0.90	287
Maisonnettewoning	0.20	1.00	0.33	136
Portiekwoning	0.00	0.00	0.00	24
Tussenwoning	0.94	0.94	0.94	1200
Twee-onder-een-kap	0.87	0.90	0.88	110
Vrijstaande woning	0.95	0.95	0.95	51

Figure 5.2: Classification report for the **c2 SVM MI model** (left) and for the **c2 RF impurity model** (right).

5.3 Comparison: Case Studies 1 and 2

Furthermore, Table 5.4 gives an overview of the accuracies of the models from both **Case Studies 1 and 2**. The accuracies are given for the classification on their test splits and on each other's whole dataset (See **Case Study 1 predictions using c2 models** and **Case Study 2 predictions using c1 models** in Section A.1 in the Appendix for the full results). However, the accuracy and the balanced accuracies for these models were lower than expected. This might be because the class distribution is still imbalanced.

		Case Study 1				**Case Study 2**			
		SVM ANOVA-F	**SVM MI**	**RF impurity**	**RF permu.**	**SVM ANOVA-F**	**SVM MI**	**RF impurity**	**RF permu.**
Test split (20%)	Accuracy	91.8%	91.7%	91.1%	91.8%	67.2%	38.9%	71.1%	73.3%
	Balanced accuracy	58.3%	57.1%	74.1%	65.2%	39.5%	33.0%	73.7%	73.7%
On other case (whole dataset)	Accuracy	55.3%	55.6%	56.6%	56.1%	60.3%	38.3%	81.2%	84.0%
	Balanced accuracy	42.0%	40.4%	40.4%	44.7%	47.6%	23.8%	58.0%	62.5%

Table 5.4: Overview of **Case Study 1 and 2** models on their test splits and each other's whole dataset.

5.4 c1 and c2 models applied on Case Studies 3 through 8

Hence, **Case Studies 3 through 8** were created to test the performance of each model on each specific building type class. In Table 5.5 an overview of the model performances on these case studies can be found with the highest accuracy and balanced accuracy for each case highlighted in **bold**.

		Case Study 1				**Case Study 2**			
		SVM ANOVA-F	**SVM MI**	**RF impurity**	**RF permu.**	**SVM ANOVA-F**	**SVM MI**	**RF impurity**	**RF permu.**
Case Study 3 flat	Accuracy	75.8%	79.5%	87.9%	82.1%	85.4%	43.7%	80.2%	**90.4%**
	Balanced accuracy	49.2%	50.6%	53.9%	51.6%	36.7%	24.9%	**74.9%**	66.7%
Case Study 4 galerij	Accuracy	85.5%	86.0%	87.7%	87.3%	80.3%	30.0%	**90.3%**	89.4%
	Balanced accuracy	59.6%	60.4%	64.5%	60.5%	32.3%	11.3%	78.2%	**81.5%**
Case Study 5 maisonnette	Accuracy	60.1%	59.9%	63.5%	61.1%	**75.0%**	53.4%	67.5%	69.8%
	Balanced accuracy	54.7%	49.6%	62.3%	**65.5%**	39.6%	26.0%	60.5%	60.6%
Case Study 6 portiek	Accuracy	17.0%	18.1%	26.7%	63.1%	**82.1%**	34.0%	68.5%	71.2%
	Balanced accuracy	42.8%	43.0%	44.8%	51.6%	40.2%	33.0%	**68.6%**	62.6%
Case Study 7 rij_2o1k	Accuracy	**89.4%**	83.2%	89.2%	89.0%	60.8%	27.9%	80.2%	86.0%
	Balanced accuracy	71.8%	62.8%	67.7%	67.4%	35.4%	23.9%	74.9%	**78.9%**
Case Study 8 vrijstaande	Accuracy	98.0%	**98.5%**	81.9%	**98.5%**	86.9%	83.4%	93.0%	93.0%
	Balanced accuracy	90.6%	**94.2%**	66.6%	91.4%	43.4%	28.6%	72.0%	64.2%

Table 5.5: Overview of case study 1 and 2 models on their test splits and each other's whole dataset.

Lastly, in Table 5.6 the highest F1-score for each of these specific case studies are shown together with the model used. The F1-score is the balanced ability of the model to both classify positive cases and be accurate with the cases it classifies for a specific class, which in this case means how good a model is at recognizing a specific building type class. Highlighted in bold are the models which did not have the best accuracies, but are still best in recognizing the specific building type.

Note that for **Case Study 6** the F1-score for recognizing **common staircase without galleries** (portiekwoning) was irrelevant due to the fact manual preprocessing of the building types was not performed on this dataset. This means some of the 'portiekwoning' buildings are still labelled as a different building type prior to the prediction, which then means that the F1-score does not accurately capture the balanced ability of the model to classify the 'portiekwoning' buildings. Also, F1-score for **Case Study 7** is the average of the F1-scores of **end house** (hoekwoning), **terraced house** (tussenwoning) and **semi-detached house** (twee-onder-een-kap).

	F1-score	Model
Case study 3 flat	0.95	c2 SVM ANOVA-F
Case study 4 galerij	0.69	**c2 RF impurity, c2 RF permutation**
Case study 5 maisonnette	0.30	**c2 RF permutation**
Case study 6 portiek	-	-
Case study 7 rij_2o1k	0.94*	c1 RF impurity, c1 RF permutation, c2 RF permutation
Case study 8 vrijstaande	1.00	**c1 SVM MI, c1 RF permutation**

Table 5.6: F1-score of highest performing model for each of **Case Studies 3 through 8**.

6 | Discussion

In this chapter the results in Chapter 5 are further discussed. First the results of **Case Studies 1 and 2** will be analysed in Section 6.1, then the c1 and c2 model performances applied to **Case Studies 3 through 8** in Section 6.2. Finally, a summary of a hit-and-miss analysis on the c1 and c2 model performances on their respective datasets is given in Section 6.3.

6.1 Case Studies 1 and 2

From Table 5.2 it can be observed that the **c1 SVM MI model** has the highest accuracy and lowest tuning and training time. While the **c1 RF impurity model** has the highest balanced accuracy, the tuning time is about 20 times longer compared to the SVM models. Although the computation time needs to be taken into account while deciding the most suitable machine learning algorithm, according to the third research subquestion. The balanced accuracy metric should be considered more important, since the class imbalance in case study 1 is the most severe. Therefore, the most suitable algorithm in case study 1 is the **c1 RF impurity model**.

Table 5.3 shows that the **c2 RF permutation model** is the most suitable model for **Case Study 2**, since the model has the highest accuracy and the highest balanced accuracy, which is even higher than its accuracy. However, the **c2 SVM models** performed notably worse, both their accuracy and balanced accuracy are lower than the **c2 RF models**. Which is not the case in **Case study 1**, where the accuracies of **c1 SVM models** and **c1 RF models** are similar.

Table 5.4 shows the model performance of each trained model on its test splits and the other whole dataset, c1 models on **Case Study 2** dataset and vice versa. The c1 models performed noticeably worse on the **Case Study 2** dataset, with a drop in accuracy of 35% for all four models. This can be explained by the class imbalance of the dataset the models were trained on. The models are, therefore, unable to classify the building types that are underrepresented. This can also be observed in the classification reports in the Appendix under **Case Study 2 predictions using c1 models** in Section A.1, where the precision, recall and F1 score are zero for ‘Galerijwoning’, ‘Maisonnettewoning’ and ‘Portiekwoning’. Similar to the results on its test split, the **c2 RF permutation model** performed the best on **Case Study 1** dataset, with even higher accuracy than the results on the test split but with lower balanced accuracy.

6.2 Model application

Table 5.5 shows the performances of the trained models on the unseen data of **Case Study 3 through 8**. In **Case Study 3**, the **c2 RF impurity model** has the highest balanced accuracy out of the eight trained models. While the **c1 RF impurity model** from **Case Study 1** performed the best out of the c1 models.

In **Case Study 4** the best performing model is the **c2 RF permutation model** instead of the **c2 RF impurity model**. On the other hand, the **c1 RF impurity model** is still the best performer out of the four c1 models applied to **Case Study 4**.

In **Case Study 5** the **c1 RF permutation model** has the highest balanced accuracy, but the balanced accuracy is lower compared to the best models in the previous cases. This can be explained with the definition of **maisonnette** introduced in the expanded flowchart. **Maisonettes** are defined as buildings with dwellings which have more than 1 floor, however, the data on the number of floors for each dwelling is not available for the whole of the Netherlands. So it is not a feature in the datasets, which makes the classification of maisonettes difficult.

The balanced accuracy of the best performer in **Case Study 6** is even lower than the balanced accuracy of the best performer in **Case Study 5**. This was expected, since the manual correction of the labels was skipped for **Case Study 6** to see if the models were still able to recognize **common staircase without galleries blocks** ('portiekwoning'). This was also observed in the confusion matrix and classification report, see Figure 6.1. 370 buildings with the label 'flatwoning' are classified as **common staircase without galleries blocks** ('portiekwoning') instead. However, this needs to be investigated further with a hit-and-miss analysis to confirm these findings.

228	49	0	0	**370**	0	2
3	0	0	0	**2**	0	0
0	0	0	0	**0**	4	1
50	**6**	**1**	**0**	**36**	**0**	**3**
1	0	0	0	**5**	0	0
4	1	0	12	**3**	54	12
0	0	0	1	**0**	1	2

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.80	0.35	0.49	649
Galerijwoning	0.00	0.00	0.00	5
Hoekwoning	0.00	0.00	0.00	5
Maisonettewoning	0.00	0.00	0.00	96
Portiekwoning	0.01	0.83	0.02	6
Tussenwoning	0.92	0.63	0.74	86
Vrijstaande Woning	0.10	0.50	0.17	4
macro avg	0.26	0.33	0.20	851
weighted avg	0.7	0.34	0.45	851
Accuracy	0.340			
Balanced Accuracy	0.330			

Table 6.1: Confusion matrix and classification report of c2 SVM MI model on Case Study 6.

For **Case Study 7**, the **c2 RF permutation model** has the highest balanced accuracy. While the **c1 SVM ANOVA-F model** is surprisingly the best performer out of the four c1 models applied to **Case Study 7**. This can be explained by the imbalanced class distribution in the dataset of **Case Study 1**. Since the single-dwelling houses are better represented for **Case Study 1**, the model trained on **Case Study 1** will perform better on these single-dwelling houses compared to models trained on **Case Study 2**.

This is also true for **Case Study 8**, where the **c1 SVM MI model** performed the best. For the c2 models, the **c2 RF impurity model** has the highest accuracy and highest balanced accuracy.

From Tables 5.4, 5.5 and 5.6, it can be established that the RF models using permutation features performed overall better than the SVM models. With **c1 RF models** perform better on the single-dwelling building types: **end house**, **terraced house**, **semi-detached house** and **detached house**. While the **c2 RF models** perform better on the multi-dwelling building types: **multi-family house**, **common staircase apartment blocks with** and **without galleries**.

6.3 Hit-and-miss analysis

A hit-and-miss analysis is performed to further analyse the model performances and to validate the ground truth. In this section the summary of the findings for **Case Study 1** and **Case Study 2** are discussed.

In **Case Study 1**, a lot of misses are because the labels given are different from what label the building would have been given with the proposed classification rules in this thesis. This can be because different classification rules are applied, see Figure 6.1 for an example. The label given for the building in the example was 'tussenwoning' (**terraced house**). But, with the proposed classification rules the building with the footprint marked with a black dot should not be considered as an adjacent building. Hence, the prediction given by the models was 'twee-onder-een-kap' (**semi-detached house**).

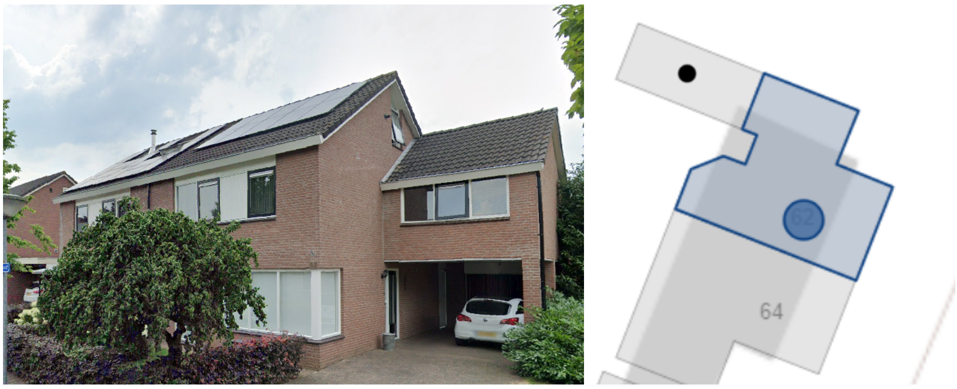


Figure 6.1: Google street view of building 'NL.IMBAG.Pand.1742100000007039' (left) and its footprint (right).

Other misses are because of misleading definitions for the different types of multi-family houses. Also, in certain cases, the classification is even hard for people to do. For example, in multi-part buildings, one part of the building might have galleries while a different part of the building might have dwellings with 2 floors (**maisonnette**). See Figure 6.2 for another similar example of misleading definitions.

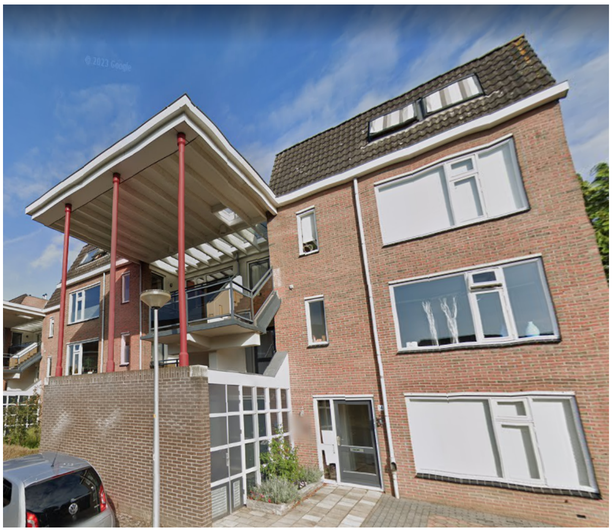


Figure 6.2: Google street view of building 'NL.IMBAG.Pand.1742100000005434'.

The label given for this example building is 'portiekwoning' (**common staircase without galleries**). However, with the proposed classification rules this building should be considered a **multi-family house** ('flatwoning'). In Figure 6.2 the cause of this misclassification can be seen, it lies in the misconception of an open porch. The building has an open stairwell above ground level, which can be interpreted as an open porch, but on the ground level this stairwell is closed.

Also, the manner of drawing the footprints of the buildings and their garage boxes influences the adjacency feature, see Figure 6.3. The label here given is ‘twee-onder-een-kap” (**semi-detached**), which is correct. However, the prediction made by the models is ‘tussenwoning’ (**terraced house**). Since the garage boxes attached to the buildings are taken with the buildings as one, see the footprints in Figure 6.3. For other cases or areas these footprints are drawn separately, see footprints in Figure 6.1 for example.

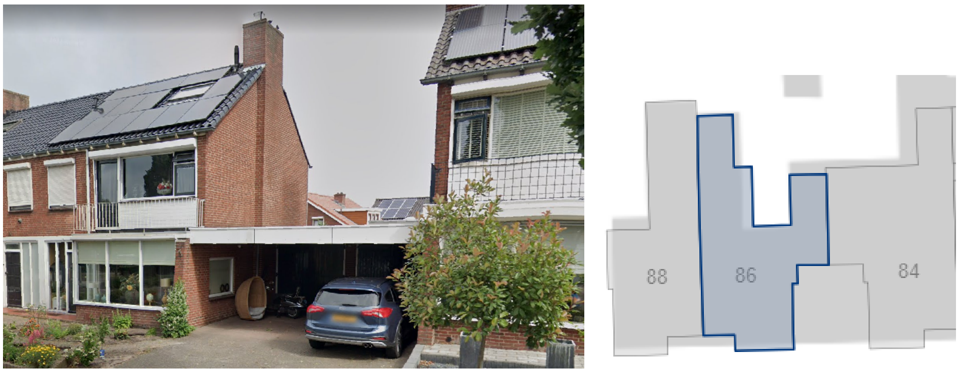


Figure 6.3: Google street view of building ‘NL.IMBAG.Pand.1742100000008886’ (left) and its footprint (right).

On top of that, complex shapes of the footprints can lead to unexpected adjacencies, see Figure 6.4. The label given was ‘vrijstaande woning’ (**detached house**), but the prediction given was a **semi-detached house**. The complex shapes of the footprints, also because the building attachments are taken with the building as one, led to the example building (highlighted and with number 2) being adjacent to the building with number 23.

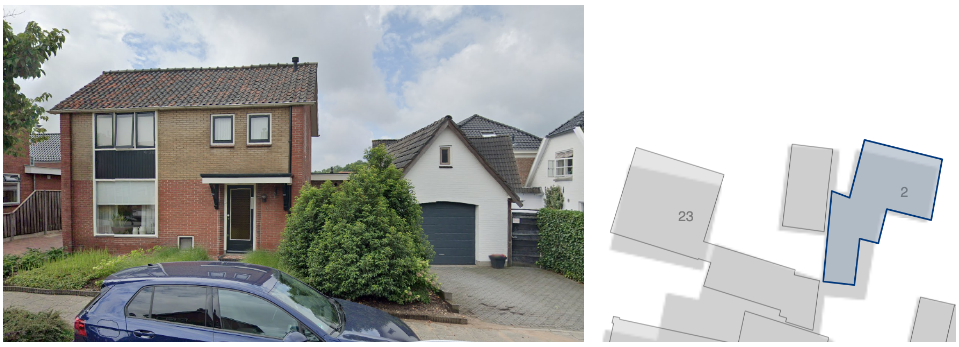


Figure 6.4: Google street view of building ‘NL.IMBAG.Pand.1742100000013965’ (left) and its footprint (right).

Furthermore, for some models, the *number of dwellings* feature did not score well enough to be selected in the feature selection process. This however led to misclassification of **multi-family houses** and **detached houses**. This might be because of the imbalance in the class distribution; fewer buildings of the multi-family house types means that the *number of dwellings* feature is less important.

In **Case Study 2** it becomes more apparent that some buildings were given the wrong label, some due to the fact of human errors and some also because the definitions of some of the classes are vague or interpreted differently.

For example, in building 'NL.IMBAG.Pand.0503100000000019' (see Figure 6.5) the label given was 'flatwoning' (**multi-family house**), however, the correct label should have been 'galerijwoning' (**common staircase with galleries**). Since the building has galleries. The SVM models predicted this building to be 'portiekwoning' (**common staircase without galleries**), however, the RF models were able to predict this building correctly as 'galerijwoning'.

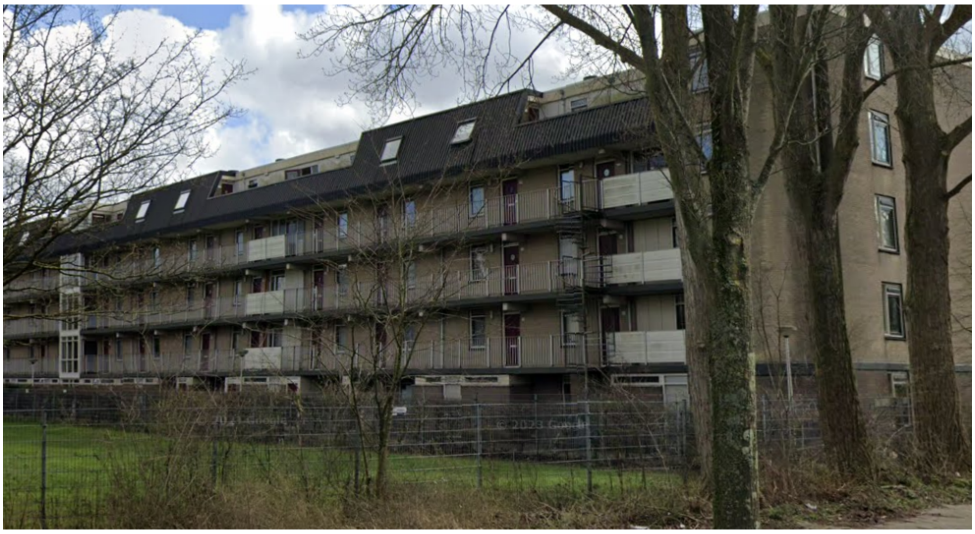


Figure 6.5: Google street view of building 'NL.IMBAG.Pand.0503100000000019'.

In a similar example, building 'NL.IMBAG.Pand.0503100000024958' (see Figure 6.6) was given the label 'galerijwoning' (**common staircase with galleries**), but the correct label should have been 'flatwoning' (**multi-family house**). Since the building does not have galleries. However, all the models except for the SVM MI model were able to classify it correctly as 'flatwoning'.

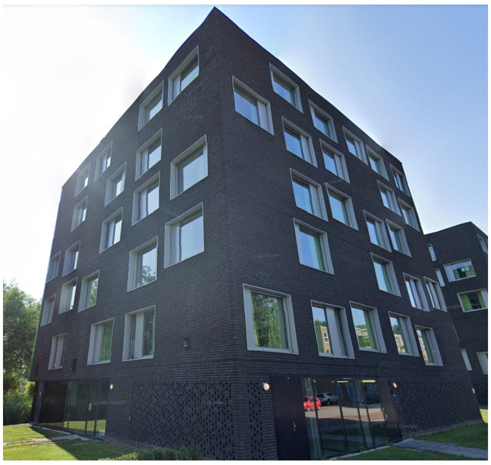


Figure 6.6: Google street view of building 'NL.IMBAG.Pand.0503100000024958'.

The following example showcases a multi-part building, where each part of the building can be classified differently. See Figure 6.7 for building 'NL.IMBAG.Pand.0503100000002054'. The two ends of this building have galleries, while the tallest centre part does not. The building type of this building is up for discussion and can be either classified as 'galerijwoning' (**common staircase with galleries**) or 'flatwoning' (**multi-family house**). The label given to this building was 'galerijwoning', however, this building has been classified as 'flatwoning' (**multi-family house**) by the models. This can be explained by the class distribution of **Case Study 2**, where there are more **multi-family house** than **common staircase apartment blocks with galleries**. The models are therefore biased towards **multi-family houses**.

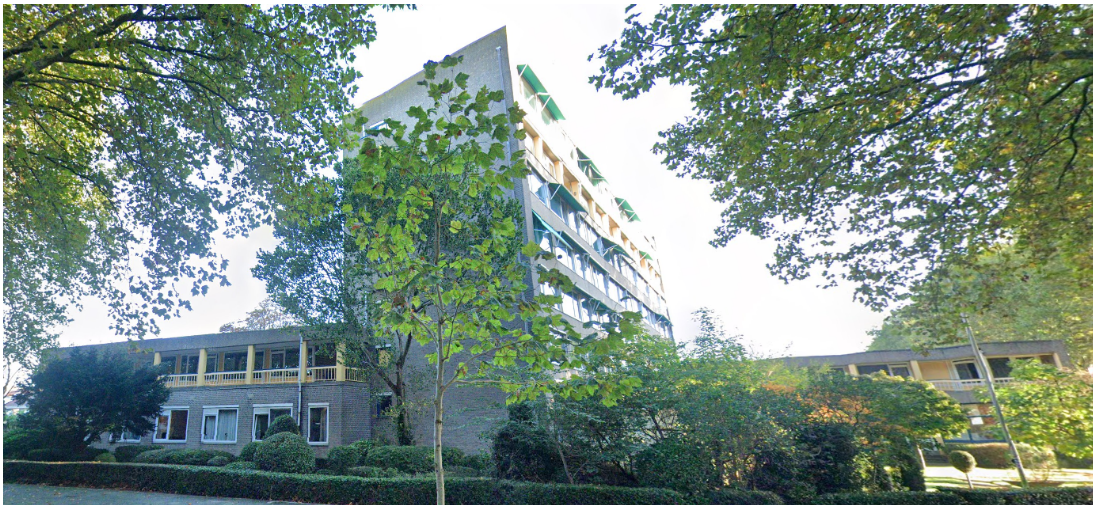


Figure 6.7: Google street view of building 'NL.IMBAG.Pand.0503100000002054'.

The next example presents a building with a building type which is a combined subtype of 'portiekwoning' (**common staircase without galleries**) and 'flatwoning' (**multi-family house**). See figure 6.8. The building type of this building is also up for discussion and depends on how the building type definitions are interpreted. And also, what the proposed classification rules are. The building was given the label 'portiekwoning' (**common staircase without galleries**). Following the proposed classification rules of this thesis, this building should be classified as 'flatwoning' (**multi-family house**). The SVM models classified this building as 'flatwoning' (**multi-family house**) accordingly, while the RF models classified this building as 'portiekwoning' (**common staircase without galleries**).

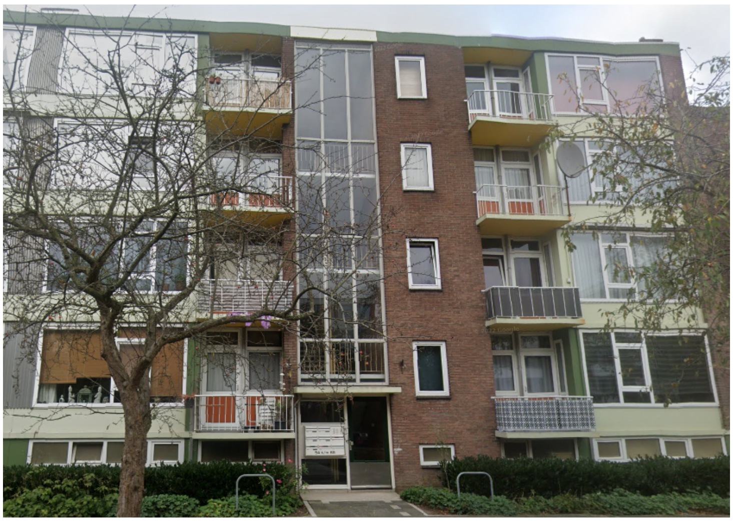


Figure 6.8: Google street view of building 'NL.IMBAG.Pand.0503100000001549'.

EP-online (2023) was used as a basis for the ground truth of the buildings outside of Rijssen-Holten (**Case Studies 2 through 8**). Manual preprocessing steps have been taken to correct some of these errors, but some errors persist as shown in the previous examples.

7 | Conclusion

7.1 Research overview

This thesis aimed to develop a method to infer the residential building type from the 3DBAG, the residential building type can then be used with the IEE project TABULA to approximate the energy consumption of a specific building. The main research question for this thesis was defined in the introduction with three additional subquestions. To answer the main question the sub-questions need to be answered first.

Sub question 1: What features are needed to infer the building types of the buildings of the 3DBAG?

From the expanded flowchart and the definition of the building types it can be concluded that the following features are needed to infer the building types of the 3DBAG:

- the number of neighbouring buildings,
- the number of neighbouring buildings of neighbouring building,
- the number of dwellings in a building,
- building having an open porch,
- building having galleries.

However, only the following features could be extracted from open data:

- the number of neighbouring buildings,
- the number of neighbouring buildings of neighbouring building,
- the number of dwellings in a building.

The absence of an open porch feature and galleries feature had to be covered by features describing the form of a building. Assuming that there should be a relation between the form of a building and the building being one of the multi-dwelling types.

Furthermore, the feature selection has shown that the extracted features needed to infer the building types of the buildings of the 3DBAG are different for each machine learning algorithm. Also, different for each filtering method and different areas of study yield different subsets of features as well. Nonetheless some of the features are included in most of the subsets of features, see Table 7.1. It can be observed that the *adjacency features*, the *oriented bounding box width* and *the actual volume in LoD2.2* were the most relevant. Since it was used the most in the trained models. Lastly, the feature engineering process showed that there is no difference in using LoD1.2 and LoD2.2 features, since the features of different level of details are highly correlated.

		Case Study 1				**Case Study 2**				
		SVM ANOVA-F	**SVM MI**	**RF impurity**	**RF permu.**	**SVM ANOVA-F**	**SVM MI**	**RF impurity**	**RF permu.**	**Count**
1	Number of adjacent buildings	x	x	x	x	x		x	x	7
2	Number of adjacent buildings of adjacent buildings.	x	x	x	x			x	x	6
3	Year of construction		x	x	x					3
4	Number of dwellings in the building	x			x	x	x	x	x	6
5	Footprint area		x				x	x	x	4
6	Footprint perimeter					x	x			2
7	Number of the vertices in footprint									0
8	The number of neighbouring building (radius: 25m)			x	x					2
9	The number of neighbouring building (radius: 50m)	x	x							2
10	The number of neighbouring building (radius: 75m)									0
11	The number of neighbouring building (radius: 100m)									0
12	Oriented bounding box width in LoD1.2	x	x	x	x	x	x	x	x	8
13	Oriented bounding box length in LoD1.2					x	x		x	3
14	Actual volume in LoD2.2	x	x	x			x	x		5
15	Total wall surface area in LoD2.2	x			x	x			x	4
16	Total roof surface area in LoD2.2						x			1
17	Maximum height in LoD2.2			x		x		x		3
18	Height (without roof) in LoD2.2									0

Table 7.1: Overview of features selected for each model.

Sub question 2: What data is required?

A cadastre database (for example, the BAG for the Netherlands) is necessary to extract features and building information, along with a semantic 3D city model (for example, 3DBAG) for the geometrical features. Additionally, labelled datasets (for example, Rijssen-Holten and EP-online datasets) are needed to provide a ground truth. However, challenges arose in feature validation and the hit-and-miss analysis, underscoring the importance of a comprehensive dataset following proposed classification rules.

Furthermore, data on *number of storeys for each dwelling* can be invaluable in classifying **maisonnettes**. The same can be said for data on buildings *having an open porch* or *having galleries*, for classifying **common staircase apartment blocks without galleries** and **common staircase apartment blocks with galleries** respectively. However, only data on the *number of storeys for each dwelling* for Dutch buildings exists in the form of the BAG+ (Gemeente Amsterdam, 2023). But, the BAG+ is only available to a few cities in the Netherlands and is not open data.

Sub question 3: Which (combination of) machine learning algorithm is the most suitable to be used for the classification of the building stock of the Netherlands, with regards to the size and nature of the data used, the availability of computational resources, the interpretability of the results and the desired level of accuracy?

Selecting the most suitable machine learning algorithm for classifying the Dutch building stock proved challenging due to varying algorithm performance across case studies. While Random Forest models generally outperformed SVM models, trade-offs between performance, tuning time, and training time were observed. Despite the longer tuning and training time for Random Forest models, their overall suitability, especially in terms of accuracy, was highlighted.

Main research question: To what extent can machine learning correctly classify the building stock of the Netherlands?

The thesis showcased well-performing models with accuracies ranging from 61.1% to 98.5% and balanced accuracies from 51.6% to 94.2%. Notably, performance varied across case studies, with distinctions in accuracy when focusing on multi-dwelling versus single-dwelling houses. These outcomes underscored the potential of machine learning to achieve high classification results for the Dutch building stock when trained on representative datasets. However, the quality of results depends on data quality, and challenges arise in scenarios involving complex buildings with multiple parts and unclear classification rules.

In summary, this research provides valuable insights into feature selection, data requirements, and algorithm performance, contributing to the broader understanding of machine learning applications in building classification.

7.2 Limitations

Despite the high accuracy achieved by the trained models, several limitations hinder the robustness of the approach:

- **Ground truth**:
    - **Classification Rule Differences**: Discrepancies between the classification rules used in the Rijssen-Holten dataset's ground truth and the proposed rules impacted model performance. Additionally, inconsistencies in the Ep-online dataset's classification, likely due to varied contributors, further complicated reliable ground truth acquisition.
    - **EP-online Dataset Challenges**: The EP-online dataset, classified at the dwelling level, introduced complexities, potentially resulting in the same building having multiple classifications. This inconsistency led to models being trained on inaccurately labelled buildings, reducing overall performance and making result interpretation challenging.
    - **Correction Efforts**: Attempts to rectify labels from the EP-online dataset were time-consuming, requiring manual verification of thousands of buildings either in person or via Google Street View. Despite correction efforts, the training data retained errors, affecting the model's reliability.

- **Features**: Certain crucial features needed to adhere to the proposed classification rules were unavailable. Notably, the absence of features such as the number of floors per dwelling, open porch presence, and galleries hindered the complete application of the proposed classification rules.

- **Feature selection**: The feature selection process resulted in the creation of eight models (four for **Case Study 1** and four for **Case Study 2**). While deemed necessary due to severe class imbalances and differing ground truths, assessing the performances of all eight models provided insights into the impact of imbalanced class distributions but added analytical intricacies.

- **Model assessment and results**: The unreliable nature of the ground truth extended to the results and assessments, introducing uncertainty. A misclassification in the prediction could be compounded by inaccurate ground truth labels, prompting the need for a labour-intensive hit-and-miss analysis.

In conclusion, the limitations outlined above underscore the challenges in achieving a robust and accurate machine learning classification of the Dutch building stock. Addressing these limitations is essential for enhancing the reliability and interpretability of the models and results.

7.3 Recommendations and future work

Based on the limitations of this thesis, several recommendations are provided for further research. First, creating an improved ground truth should be explored. This could be done by establishing standardized classification rules and ensuring consistency across the dataset. This will mitigate discrepancies in the classification results, for example, the wrongly classified buildings because of different interpretations of building type definitions or different classification rules. The classification rules should also include handling cases where the building type is up for discussion, for example, multi-part buildings or buildings where the building type is a combination of building types. This would improve the classification results, interpretability of these results and the performance of the trained models.

Second, additional subsets of the Netherlands could be explored in the model creation and application. To compare results, but to also address the severe class imbalance observed in **Case Study 1** and **Case Study 2**. Severe imbalance in class distribution could also be prevented by creating a synthetic dataset while keeping the balance of the class distribution. However, a balanced class distribution usually does not occur in real-world scenarios. And, could lead to unexpected class bias issues.

Third, investigate alternative sources or methods to supplement missing features, such as the number of floors per dwelling, open porch presence, and galleries. The BAG+, for example, could be used to obtain the number of floors per dwelling. Furthermore, the space indices introduced in 3DBM could also be used to better describe the form of a building and supplement the missing features. Or even, collaborating with domain experts to explore additional relevant features. All this could contribute to a more comprehensive classification.

Fourth, in terms of feature selection, the process described in this thesis resulted in the creation of eight models. Unnecessarily increasing the complexity of model evaluation should be avoided. A more incremental approach should be used, starting with one machine learning algorithm using a small number of feature selection methods. Or several machine learning algorithms using one feature selection method for each algorithm.

Fifth, the model assessment could be improved by developing robust validation strategies, considering the uncertainties in ground truth, to ensure more reliable model assessments. For example, implementing cross-validation techniques that account for potential misclassifications in the ground truth provides a more accurate estimation of model performance.

In addition to these recommendations, possible future work can be defined as follows:

- **Automated Ground Truth Correction**: Investigate the development of automated algorithms or tools to assist in the correction of ground truth labels, potentially leveraging machine learning techniques for label verification.

- **Integration of Additional Data Sources**: Explore the integration of complementary data sources to enhance the availability of critical features, improving the overall completeness of the classification rules.

- **Advanced machine learning algorithms**: Investigate the applicability of more advanced machine learning algorithms, such as neural networks, to potentially capture complex relationships and patterns within the data.

- **Longitudinal Analysis**: Conduct a longitudinal analysis to observe changes in the Dutch building stock over time, potentially uncovering trends and patterns that can further refine classification models.

- **Collaboration with Stakeholders**: Engage in collaborative efforts with relevant stakeholders, including local authorities and data contributors, to establish a more standardized and consistent classification framework.

- **User-Friendly Model Interpretation**: Develop user-friendly tools or interfaces that facilitate the interpretation of model results, considering the complexities introduced by uncertain ground truth labels.

7.4 Reflection

- **The relationship between the methodical line of approach of the Master Geomatics and the method chosen by the student in this framework**: The methodical line of approach in the Master Geomatics program emphasizes a comprehensive understanding of geospatial data processing and analysis. In accordance with this, the research employed machine learning techniques for the classification of Dutch building stock. The utilisation of these methods allowed for a nuanced exploration of spatial patterns and relationships within the dataset, aligning with the program's emphasis on cutting-edge methodologies.

- **The relationship between the conducted research and application of the field geomatics**: The research outcomes hold practical implications for the field of geomatics by offering a novel approach to classifying building types. The application of machine learning algorithms not only enhances the efficiency of classification but also opens avenues for automating building stock analyses. This not only aligns with the technological advancements promoted in the geomatics field but also underscores the potential for real-world applications in urban planning, resource management, and infrastructure development.

- **The relationship between the project and the wider social context**: The research's findings transcend the academic setting and contribute to addressing broader societal challenges. Accurate classification of building types has direct implications for sustainable urban development, resource optimization, and disaster preparedness. By providing a robust method for understanding the spatial distribution of building types, the research adds value to urban planning initiatives, supporting informed decision-making processes that ultimately benefit communities and the environment.

- **Overall reflection**: The research undertaken seamlessly integrates with the Master Geomatics program's commitment to advancing geospatial methodologies and their practical applications. By navigating the complexities of building stock classification, the project not only contributes to the academic discourse but also presents a tangible solution with far-reaching implications for urban planning and societal well-being. This journey has reinforced the significance of bridging theoretical knowledge with real-world challenges, solidifying my commitment to making meaningful contributions in the field of geomatics.

A | Appendix

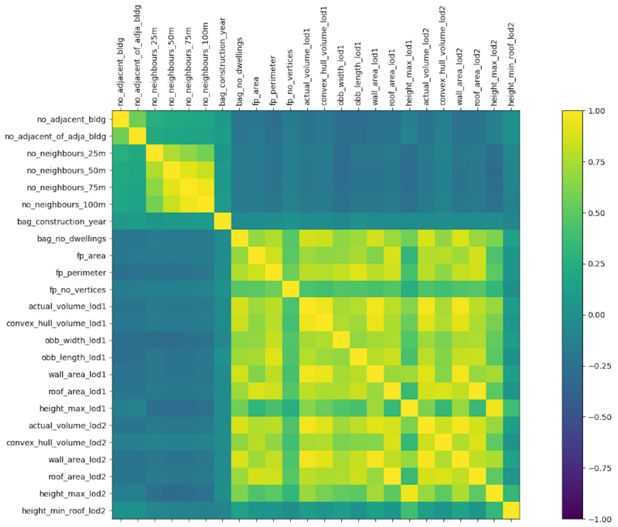
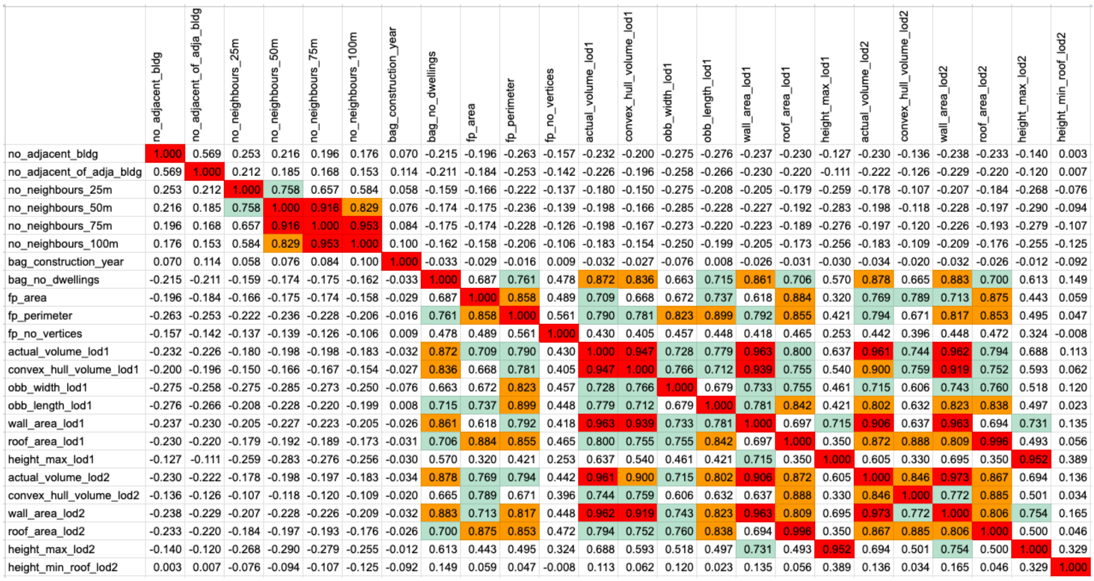


Figure A.1: Correlation matrix for **Case Study 2**.



	no_adjacent_bldg	no_adjacent_of_adja_bldg	no_neighbours_25m	no_neighbours_50m	no_neighbours_75m	no_neighbours_100m	bag_construction_year	bag_no_dwellings	fp_area	fp_perimeter	fp_no_vertices	actual_volume_lod1	convex_hull_volume_lod1	obb_width_lod1	obb_length_lod1	wall_area_lod1	roof_area_lod1	height_max_lod1	actual_volume_lod2	convex_hull_volume_lod2	wall_area_lod2	roof_area_lod2	height_max_lod2	height_min_roof_lod2
no_adjacent_bldg	1.000	0.569	0.253	0.216	0.196	0.176	0.070	-0.215	-0.196	-0.263	-0.157	-0.232	-0.200	-0.275	-0.276	-0.237	-0.230	-0.127	-0.230	-0.136	-0.238	-0.233	-0.140	0.003
no_adjacent_of_adja_bldg	0.569	1.000	0.212	0.185	0.168	0.153	0.114	-0.211	-0.184	-0.253	-0.142	-0.226	-0.196	-0.258	-0.266	-0.230	-0.220	-0.111	-0.222	-0.126	-0.229	-0.220	-0.120	0.007
no_neighbours_25m	0.253	0.212	1.000	0.758	0.657	0.584	0.058	-0.159	-0.166	-0.222	-0.137	-0.180	-0.150	-0.275	-0.208	-0.205	-0.179	-0.259	-0.178	-0.107	-0.207	-0.184	-0.268	-0.076
no_neighbours_50m	0.216	0.185	0.758	1.000	0.916	0.829	0.076	-0.174	-0.175	-0.236	-0.139	-0.198	-0.166	-0.285	-0.228	-0.227	-0.192	-0.283	-0.198	-0.118	-0.228	-0.197	-0.290	-0.094
no_neighbours_75m	0.196	0.168	0.657	0.916	1.000	0.953	0.084	-0.175	-0.174	-0.228	-0.126	-0.198	-0.167	-0.273	-0.220	-0.223	-0.189	-0.276	-0.197	-0.120	-0.226	-0.193	-0.279	-0.107
no_neighbours_100m	0.176	0.153	0.584	0.829	0.953	1.000	0.100	-0.162	-0.158	-0.206	-0.106	-0.183	-0.154	-0.250	-0.199	-0.205	-0.173	-0.256	-0.183	-0.109	-0.209	-0.176	-0.255	-0.125
bag_construction_year	0.070	0.114	0.058	0.076	0.084	0.100	1.000	-0.033	-0.029	-0.016	0.009	-0.032	-0.027	-0.076	0.008	-0.026	-0.031	-0.030	-0.034	-0.020	-0.032	-0.026	-0.012	-0.092
bag_no_dwellings	-0.215	-0.211	-0.159	-0.174	-0.175	-0.162	-0.033	1.000	0.687	0.761	0.478	0.872	0.836	0.663	0.715	0.861	0.706	0.570	0.878	0.665	0.883	0.700	0.613	0.149
fp_area	-0.196	-0.184	-0.166	-0.175	-0.174	-0.158	-0.029	0.687	1.000	0.858	0.489	0.709	0.668	0.672	0.737	0.618	0.884	0.320	0.769	0.789	0.713	0.875	0.443	0.059
fp_perimeter	-0.263	-0.253	-0.222	-0.236	-0.228	-0.206	-0.016	0.761	0.858	1.000	0.561	0.790	0.781	0.823	0.899	0.792	0.855	0.421	0.794	0.671	0.817	0.853	0.495	0.047
fp_no_vertices	-0.157	-0.142	-0.137	-0.139	-0.126	-0.106	0.009	0.478	0.489	0.561	1.000	0.430	0.405	0.457	0.448	0.418	0.465	0.253	0.442	0.396	0.448	0.472	0.324	-0.008
actual_volume_lod1	-0.232	-0.226	-0.180	-0.198	-0.198	-0.183	-0.032	0.872	0.709	0.790	0.430	1.000	0.947	0.728	0.779	0.963	0.800	0.637	0.961	0.744	0.962	0.794	0.688	0.113
convex_hull_volume_lod1	-0.200	-0.196	-0.150	-0.166	-0.167	-0.154	-0.027	0.836	0.668	0.781	0.405	0.947	1.000	0.766	0.712	0.939	0.755	0.540	0.900	0.759	0.919	0.752	0.593	0.062
obb_width_lod1	-0.275	-0.258	-0.275	-0.285	-0.273	-0.250	-0.076	0.663	0.672	0.823	0.457	0.728	0.766	1.000	0.679	0.733	0.755	0.461	0.715	0.606	0.743	0.760	0.518	0.120
obb_length_lod1	-0.276	-0.266	-0.208	-0.228	-0.220	-0.199	0.008	0.715	0.737	0.899	0.448	0.779	0.712	0.679	1.000	0.781	0.842	0.421	0.802	0.632	0.823	0.838	0.497	0.023
wall_area_lod1	-0.237	-0.230	-0.205	-0.227	-0.223	-0.205	-0.026	0.861	0.618	0.792	0.418	0.963	0.939	0.733	0.781	1.000	0.697	0.715	0.906	0.637	0.963	0.694	0.731	0.135
roof_area_lod1	-0.230	-0.220	-0.179	-0.192	-0.189	-0.173	-0.031	0.706	0.884	0.855	0.465	0.800	0.755	0.755	0.842	0.697	1.000	0.350	0.872	0.888	0.809	0.996	0.493	0.056
height_max_lod1	-0.127	-0.111	-0.259	-0.283	-0.276	-0.256	-0.030	0.570	0.320	0.421	0.253	0.637	0.540	0.461	0.421	0.715	0.350	1.000	0.605	0.330	0.695	0.350	0.952	0.389
actual_volume_lod2	-0.230	-0.222	-0.178	-0.198	-0.197	-0.183	-0.034	0.878	0.769	0.794	0.442	0.961	0.900	0.715	0.802	0.906	0.872	0.605	1.000	0.846	0.973	0.867	0.694	0.136
convex_hull_volume_lod2	-0.136	-0.126	-0.107	-0.118	-0.120	-0.109	-0.020	0.665	0.789	0.671	0.396	0.744	0.759	0.606	0.632	0.637	0.888	0.330	0.846	1.000	0.772	0.885	0.501	0.034
wall_area_lod2	-0.238	-0.229	-0.207	-0.228	-0.226	-0.209	-0.032	0.883	0.713	0.817	0.448	0.962	0.919	0.743	0.823	0.963	0.809	0.695	0.973	0.772	1.000	0.806	0.754	0.165
roof_area_lod2	-0.233	-0.220	-0.184	-0.197	-0.193	-0.176	-0.026	0.700	0.875	0.853	0.472	0.794	0.752	0.760	0.838	0.694	0.996	0.350	0.867	0.885	0.806	1.000	0.500	0.046
height_max_lod2	-0.140	-0.120	-0.268	-0.290	-0.279	-0.255	-0.012	0.613	0.443	0.495	0.324	0.688	0.593	0.518	0.497	0.731	0.493	0.952	0.694	0.501	0.754	0.500	1.000	0.329
height_min_roof_lod2	0.003	0.007	-0.076	-0.094	-0.107	-0.125	-0.092	0.149	0.059	0.047	-0.008	0.113	0.062	0.120	0.023	0.135	0.056	0.389	0.136	0.034	0.165	0.046	0.329	1.000

Figure A.2: Correlation matrix with exact coefficient values for **Case Study 2**.

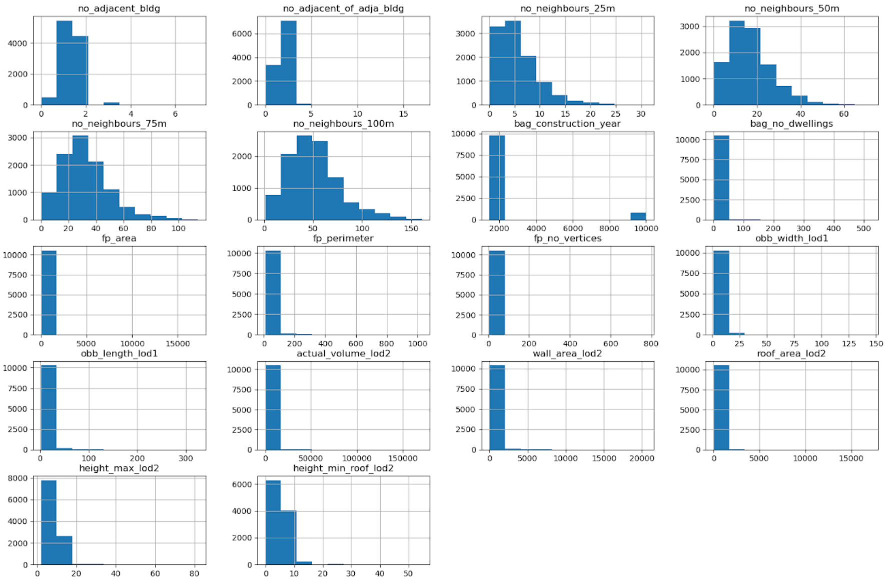


Figure A.3: Histograms of the features of the acquired training dataset for **Case Study 2**.

feature	name	score
11	obb_width_lod1	417.04
16	height_max_lod2	400.64
12	obb_length_lod1	370.06
9	fp_perimeter	351.90
14	wall_area_lod2	332.24
0	no_adjacent_bldg	318.57
7	bag_no_dwellings	275.42
13	actual_volume_lod2	228.20
1	no_adjacent_of_adja_bldg	191.52
15	roof_area_lod2	175.76
17	height_min_roof_lod2	165.79
8	fp_area	136.97
3	no_neighbours_50m	89.38
10	fp_no_vertices	86.91
2	no_neighbours_25m	84.31
4	no_neighbours_75m	82.78
5	no_neighbours_100m	72.91
6	bag_construction_year	5.45

feature	name	score
7	bag_no_dwellings	0.518
11	obb_width_lod1	0.397
13	actual_volume_lod2	0.369
8	fp_area	0.363
14	wall_area_lod2	0.340
9	fp_perimeter	0.325
15	roof_area_lod2	0.295
12	obb_length_lod1	0.253
16	height_max_lod2	0.235
0	no_adjacent_bldg	0.205
6	bag_construction_year	0.177
1	no_adjacent_of_adja_bldg	0.164
17	height_min_roof_lod2	0.154
10	fp_no_vertices	0.116
2	no_neighbours_25m	0.080
3	no_neighbours_50m	0.069
5	no_neighbours_100m	0.058
4	no_neighbours_75m	0.056

Table A.1: ANOVA-F score (left) and MI score (right) of the features for **Case Study 2**.

feature	name
11	obb_width_lod1
16	height_max_lod2
12	obb_length_lod1
9	fp_perimeter
14	wall_area_lod2
0	no_adjacent_bldg
7	bag_no_dwellings

feature	name
7	bag_no_dwellings
11	obb_width_lod1
13	actual_volume_lod2
8	fp_area
9	fp_perimeter
15	roof_area_lod2
12	obb_length_lod1

Table A.2: ANOVA-F features (left) & MI features (right) for **Case Study 2**.

feature	name	score
7	bag_no_dwellings	0.152
0	no_adjacent_bldg	0.093
11	obb_width_lod1	0.088
13	actual_volume_lod2	0.078
8	fp_area	0.067
14	wall_area_lod2	0.066
1	no_adjacent_of_adja_bldg	0.060
15	roof_area_lod2	0.051
16	height_max_lod2	0.047
9	fp_perimeter	0.044
12	obb_length_lod1	0.044
17	height_min_roof_lod2	0.041
6	bag_construction_year	0.036
5	no_neighbours_100m	0.030
4	no_neighbours_75m	0.028
3	no_neighbours_50m	0.027
10	fp_no_vertices	0.026
2	no_neighbours_25m	0.023

feature	name
7	bag_no_dwellings
0	no_adjacent_bldg
11	obb_width_lod1
13	actual_volume_lod2
8	fp_area
1	no_adjacent_of_adja_bldg
16	height_max_lod2

Table A.3: Random Forest impurity-based feature importances (left) & impurity features (right) for **Case Study 2**.

feature	name	ranking
7	bag_no_dwellings	1
0	no_adjacent_bldg	2
1	no_adjacent_of_adja_bldg	3
11	obb_width_lod1	4
12	obb_length_lod1	5
8	fp_area	6
14	wall_area_lod2	7
6	bag_construction_year	8
13	actual_volume_lod2	9
17	height_min_roof_lod2	10
16	height_max_lod2	11
15	roof_area_lod2	12
2	no_neighbours_25m	13
5	no_neighbours_100m	14
3	no_neighbours_50m	15
10	fp_no_vertices	16
9	fp_perimeter	17
4	no_neighbours_75m	18

feature	name
7	bag_no_dwellings
0	no_adjacent_bldg
1	no_adjacent_of_adja_bldg
11	obb_width_lod1
12	obb_length_lod1
8	fp_area
14	wall_area_lod2

Table A.4: Random Forest permutation-based feature importances (left) & permutation features (right) for **Case Study 2**

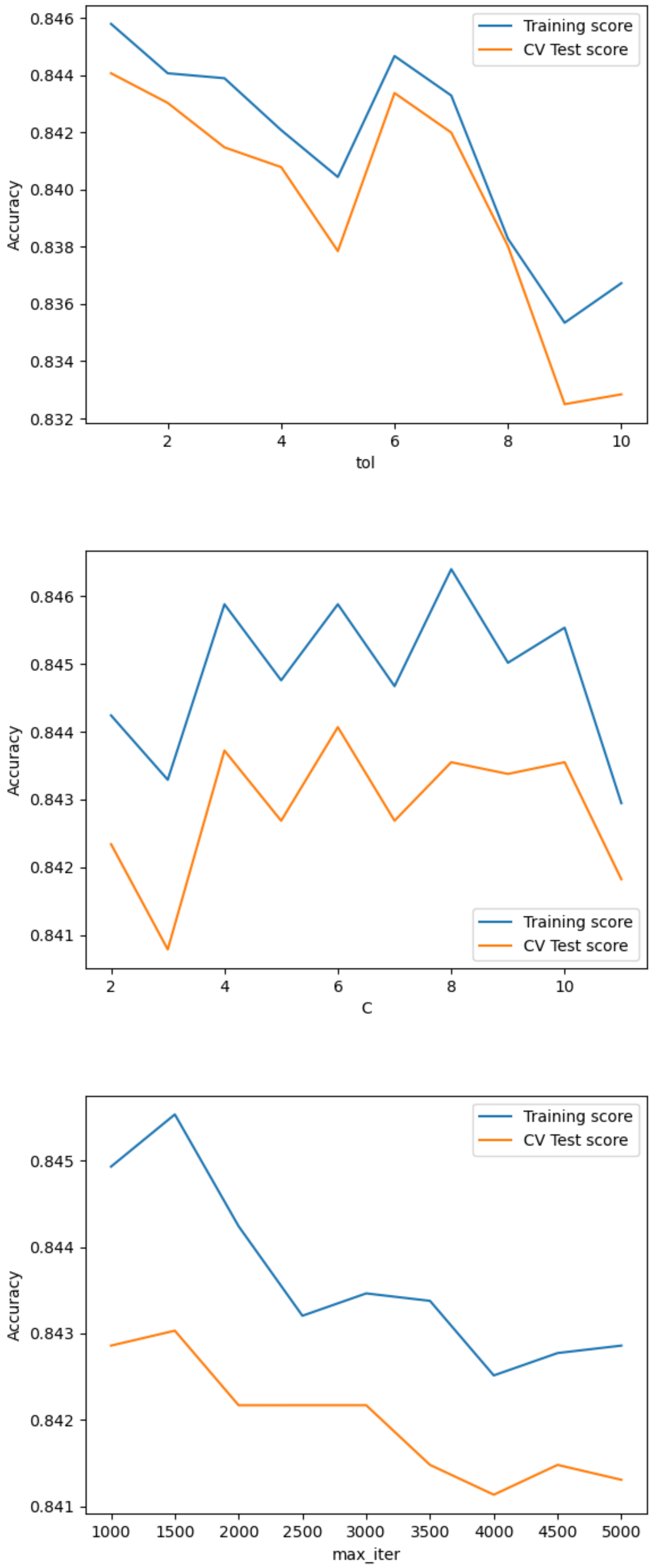


Figure A.4: Validation curves for SVM using ANOVA-F features.

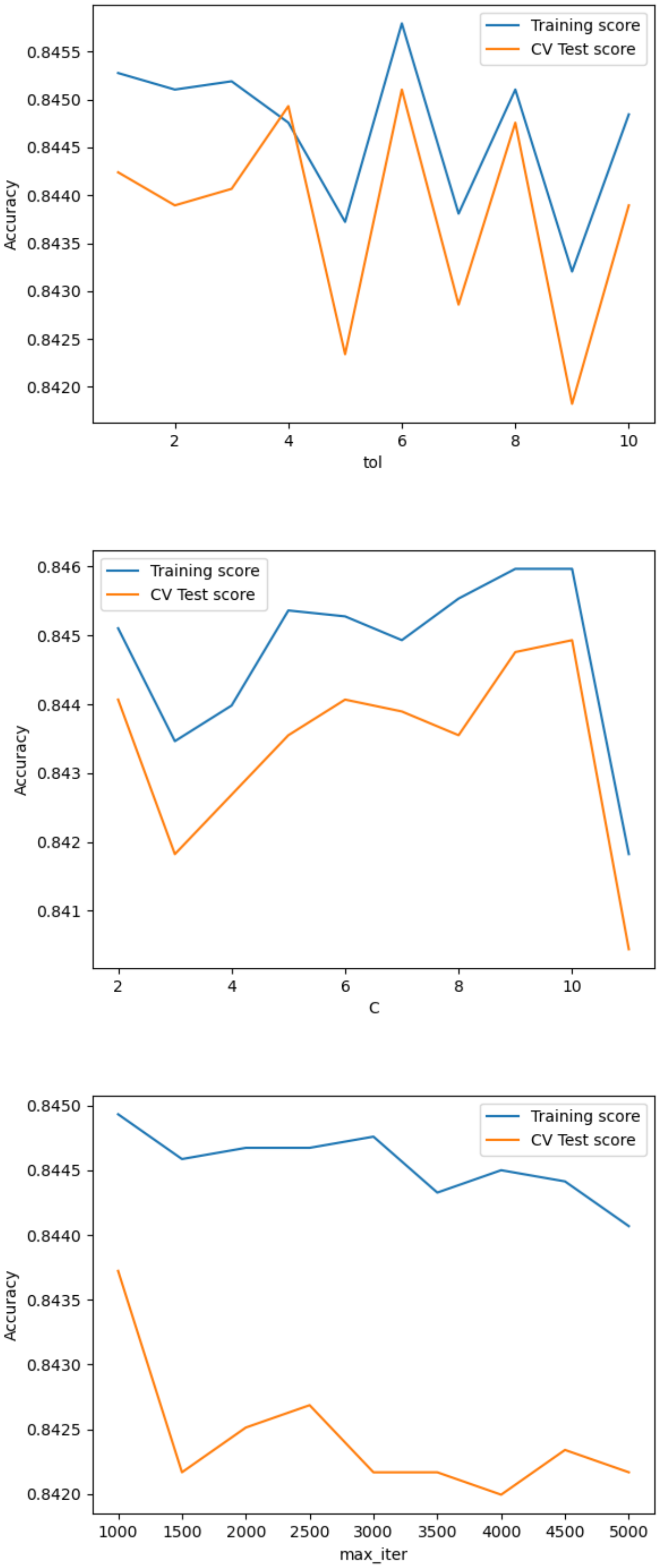


Figure A.5: Validation curves for SVM using MI features.

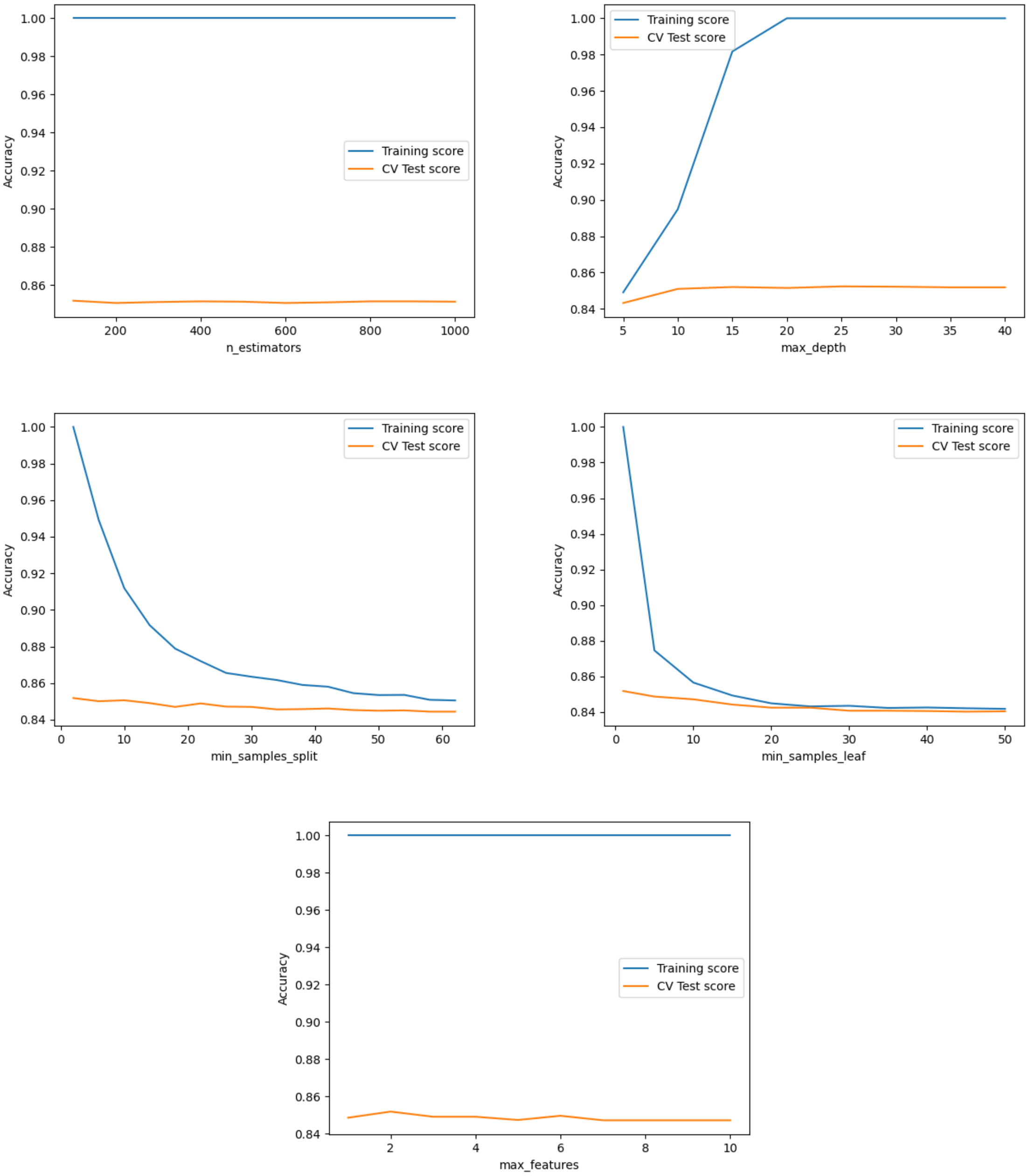


Figure A.6: Validation curves for RF using impurity features.

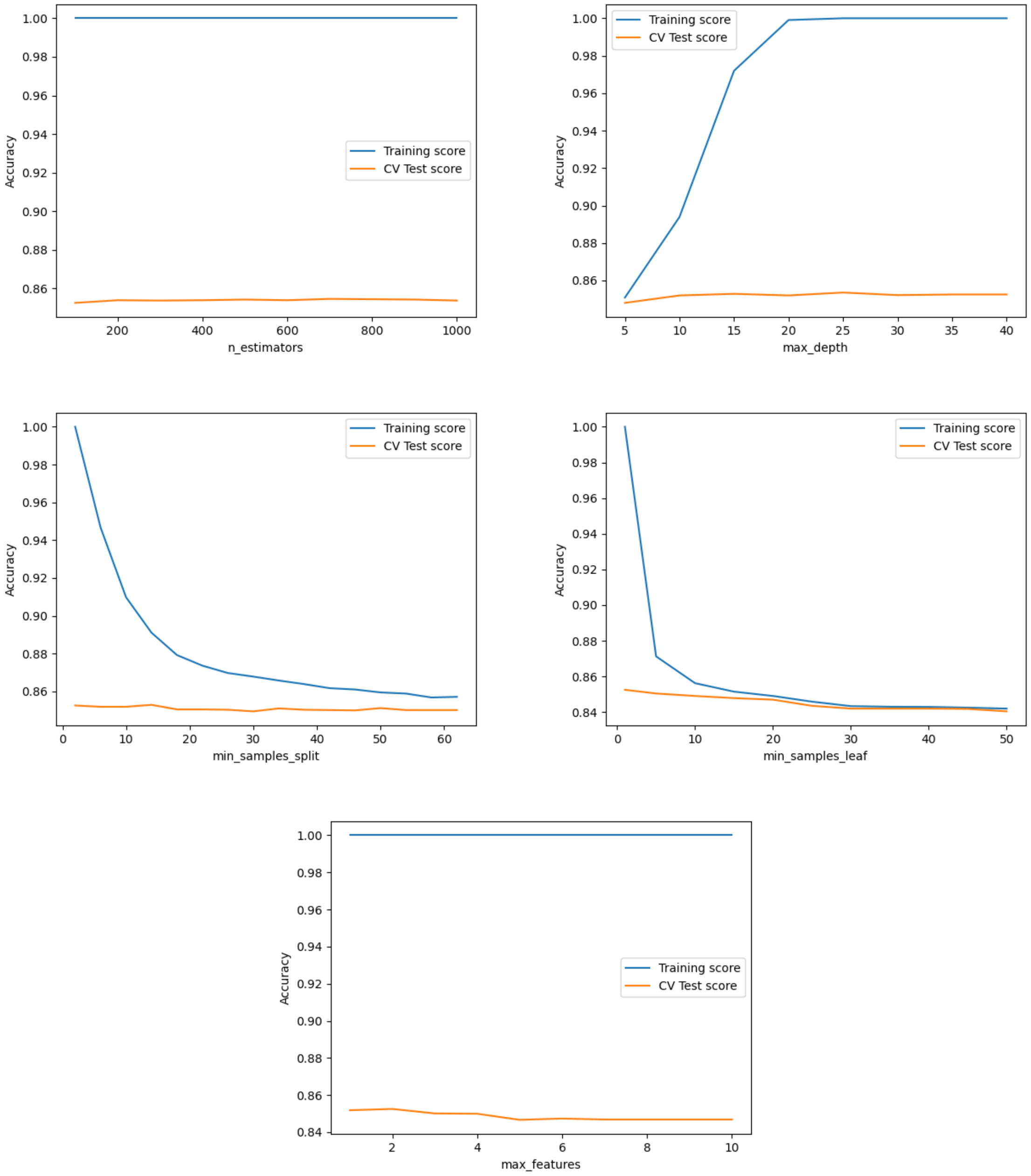
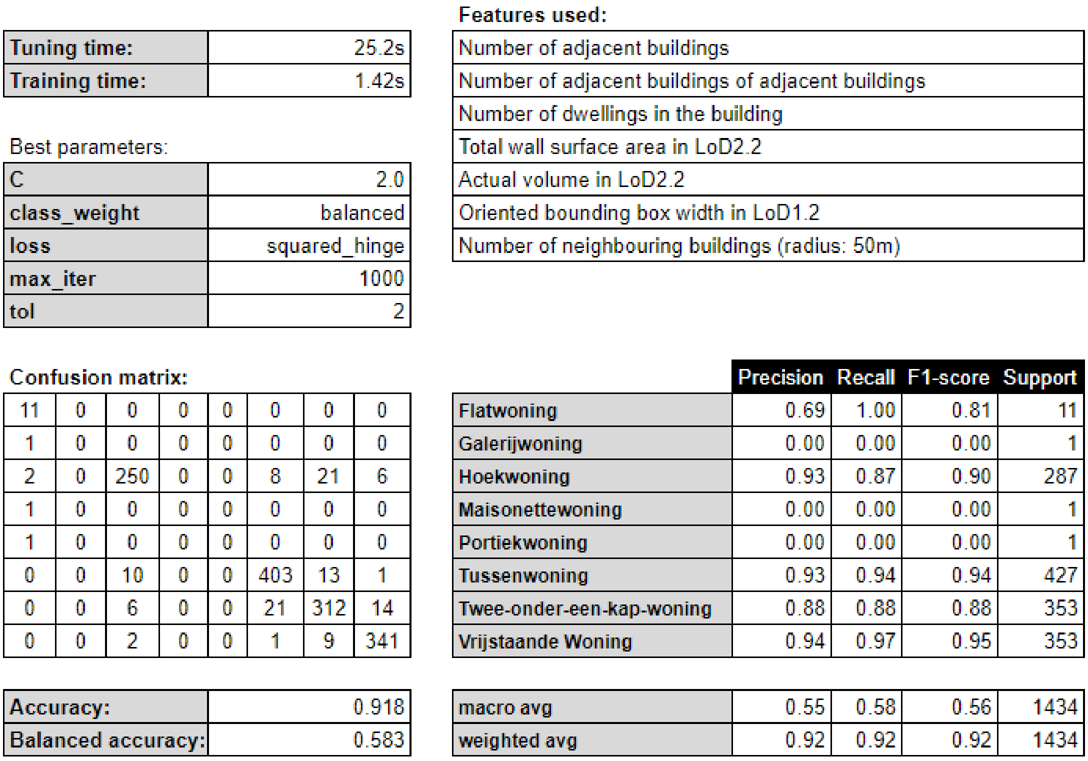


Figure A.7: Validation curves for RF using permutation features.

A.1 Results

A.1.1 Case Study 1



Tuning time:	25.2s
Training time:	1.42s

Best parameters:

C	2.0
class_weight	balanced
loss	squared_hinge
max_iter	1000
tol	2

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Number of dwellings in the building
Total wall surface area in LoD2.2
Actual volume in LoD2.2
Oriented bounding box width in LoD1.2
Number of neighbouring buildings (radius: 50m)

Confusion matrix:

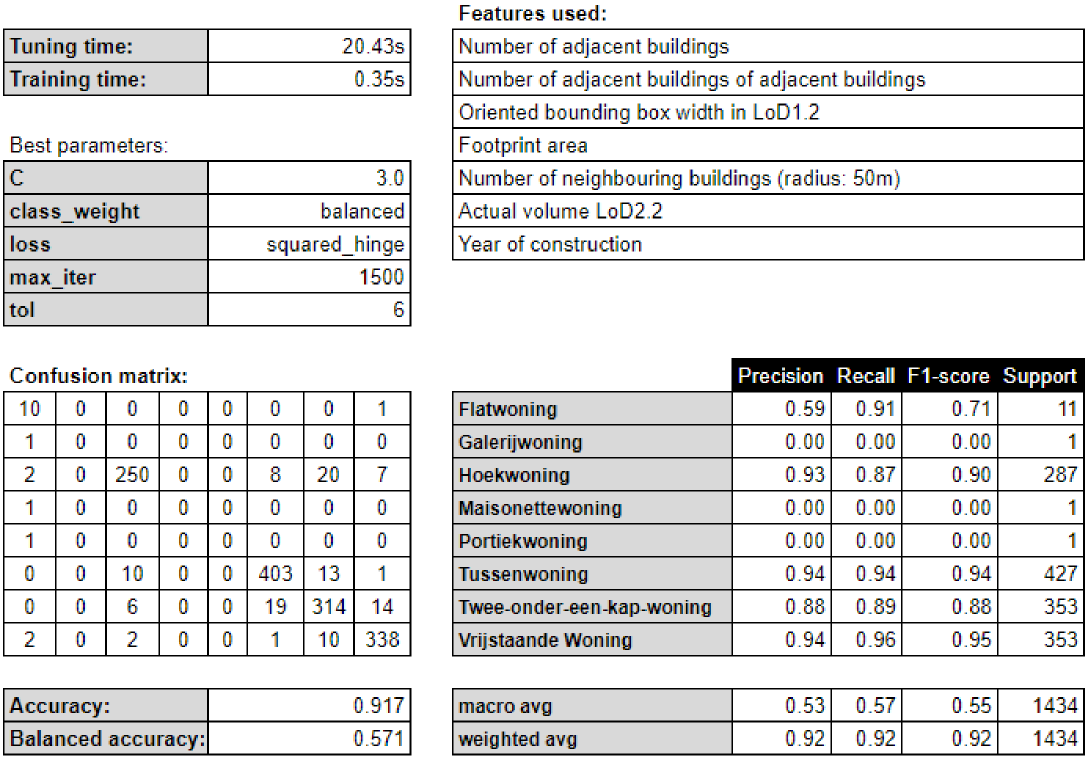
11	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
2	0	250	0	0	8	21	6
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	10	0	0	403	13	1
0	0	6	0	0	21	312	14
0	0	2	0	0	1	9	341

	Precision	Recall	F1-score	Support
Flatwoning	0.69	1.00	0.81	11
Galerijwoning	0.00	0.00	0.00	1
Hoekwoning	0.93	0.87	0.90	287
Maisonettewoning	0.00	0.00	0.00	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.93	0.94	0.94	427
Twee-onder-een-kap-woning	0.88	0.88	0.88	353
Vrijstaande Woning	0.94	0.97	0.95	353

Accuracy:	0.918
Balanced accuracy:	0.583

macro avg	0.55	0.58	0.56	1434
weighted avg	0.92	0.92	0.92	1434

Figure A.8: SVM using ANOVA-F features on **Case Study 1** test split (0.2).



Tuning time:	20.43s
Training time:	0.35s

Best parameters:

C	3.0
class_weight	balanced
loss	squared_hinge
max_iter	1500
tol	6

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Oriented bounding box width in LoD1.2
Footprint area
Number of neighbouring buildings (radius: 50m)
Actual volume LoD2.2
Year of construction

Confusion matrix:

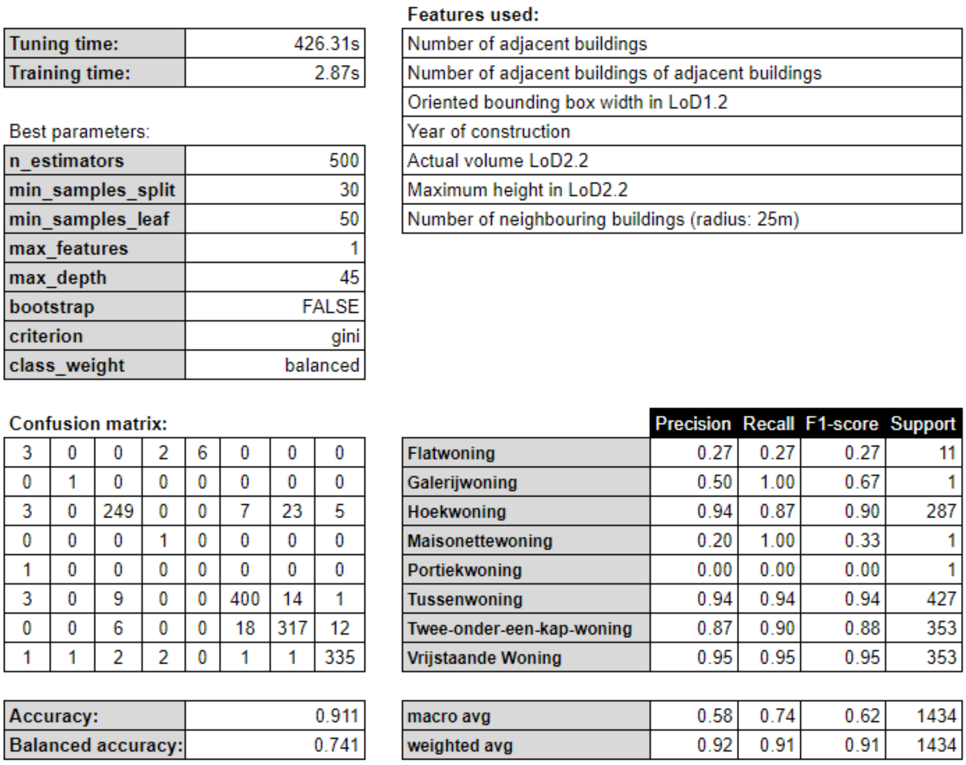
10	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0
2	0	250	0	0	8	20	7
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	10	0	0	403	13	1
0	0	6	0	0	19	314	14
2	0	2	0	0	1	10	338

	Precision	Recall	F1-score	Support
Flatwoning	0.59	0.91	0.71	11
Galerijwoning	0.00	0.00	0.00	1
Hoekwoning	0.93	0.87	0.90	287
Maisonettewoning	0.00	0.00	0.00	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.94	0.94	0.94	427
Twee-onder-een-kap-woning	0.88	0.89	0.88	353
Vrijstaande Woning	0.94	0.96	0.95	353

Accuracy:	0.917
Balanced accuracy:	0.571

macro avg	0.53	0.57	0.55	1434
weighted avg	0.92	0.92	0.92	1434

Figure A.9: SVM using MI features on **Case Study 1** test split (0.2).



Tuning time:	426.31s
Training time:	2.87s

Best parameters:

n_estimators	500
min_samples_split	30
min_samples_leaf	50
max_features	1
max_depth	45
bootstrap	FALSE
criterion	gini
class_weight	balanced

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Oriented bounding box width in LoD1.2
Year of construction
Actual volume LoD2.2
Maximum height in LoD2.2
Number of neighbouring buildings (radius: 25m)

Confusion matrix:

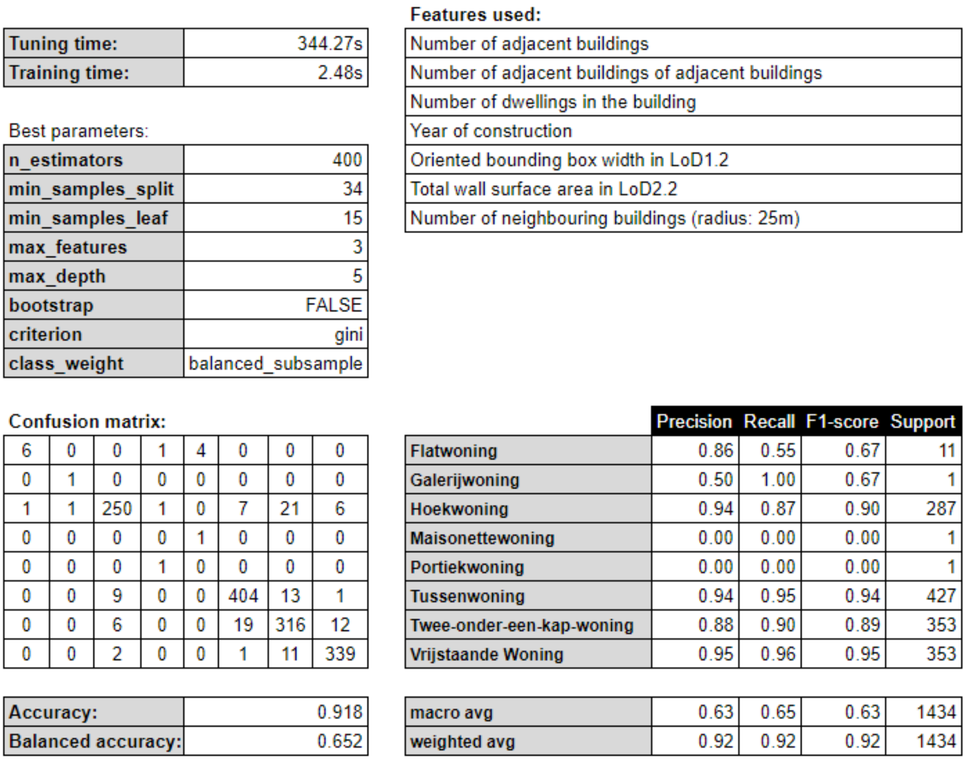
3	0	0	2	6	0	0	0
0	1	0	0	0	0	0	0
3	0	249	0	0	7	23	5
0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0
3	0	9	0	0	400	14	1
0	0	6	0	0	18	317	12
1	1	2	2	0	1	1	335

	Precision	Recall	F1-score	Support
Flatwoning	0.27	0.27	0.27	11
Galerijwoning	0.50	1.00	0.67	1
Hoekwoning	0.94	0.87	0.90	287
Maisonettewoning	0.20	1.00	0.33	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.94	0.94	0.94	427
Twee-onder-een-kap-woning	0.87	0.90	0.88	353
Vrijstaande Woning	0.95	0.95	0.95	353

Accuracy:	0.911
Balanced accuracy:	0.741

macro avg	0.58	0.74	0.62	1434
weighted avg	0.92	0.91	0.91	1434

Figure A.10: RF using impurity features on **Case Study 1** test split (0.2).



Tuning time:	344.27s
Training time:	2.48s

Best parameters:

n_estimators	400
min_samples_split	34
min_samples_leaf	15
max_features	3
max_depth	5
bootstrap	FALSE
criterion	gini
class_weight	balanced_subsample

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Number of dwellings in the building
Year of construction
Oriented bounding box width in LoD1.2
Total wall surface area in LoD2.2
Number of neighbouring buildings (radius: 25m)

Confusion matrix:

6	0	0	1	4	0	0	0
0	1	0	0	0	0	0	0
1	1	250	1	0	7	21	6
0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0
0	0	9	0	0	404	13	1
0	0	6	0	0	19	316	12
0	0	2	0	0	1	11	339

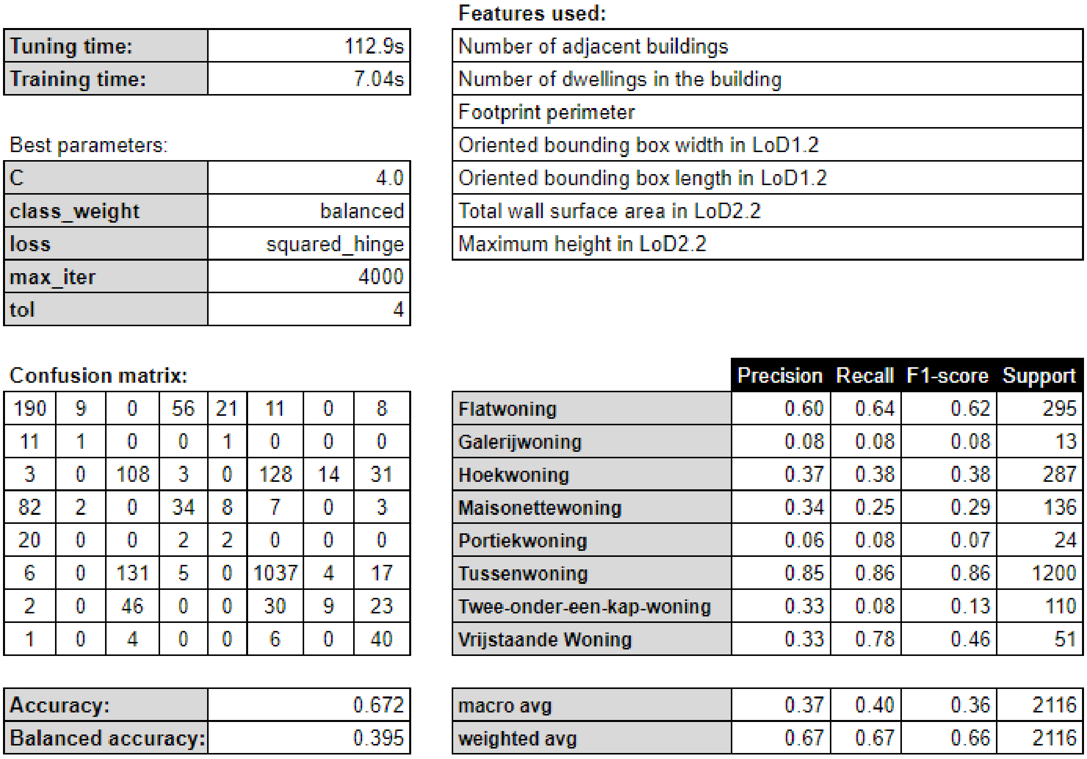
	Precision	Recall	F1-score	Support
Flatwoning	0.86	0.55	0.67	11
Galerijwoning	0.50	1.00	0.67	1
Hoekwoning	0.94	0.87	0.90	287
Maisonettewoning	0.00	0.00	0.00	1
Portiekwoning	0.00	0.00	0.00	1
Tussenwoning	0.94	0.95	0.94	427
Twee-onder-een-kap-woning	0.88	0.90	0.89	353
Vrijstaande Woning	0.95	0.96	0.95	353

Accuracy:	0.918
Balanced accuracy:	0.652

macro avg	0.63	0.65	0.63	1434
weighted avg	0.92	0.92	0.92	1434

Figure A.11: RF using permutation features on **Case Study 1** test split (0.2).

A.1.2 Case Study 2



Tuning time:	112.9s
Training time:	7.04s

Best parameters:

C	4.0
class_weight	balanced
loss	squared_hinge
max_iter	4000
tol	4

Features used:

Number of adjacent buildings
Number of dwellings in the building
Footprint perimeter
Oriented bounding box width in LoD1.2
Oriented bounding box length in LoD1.2
Total wall surface area in LoD2.2
Maximum height in LoD2.2

Confusion matrix:

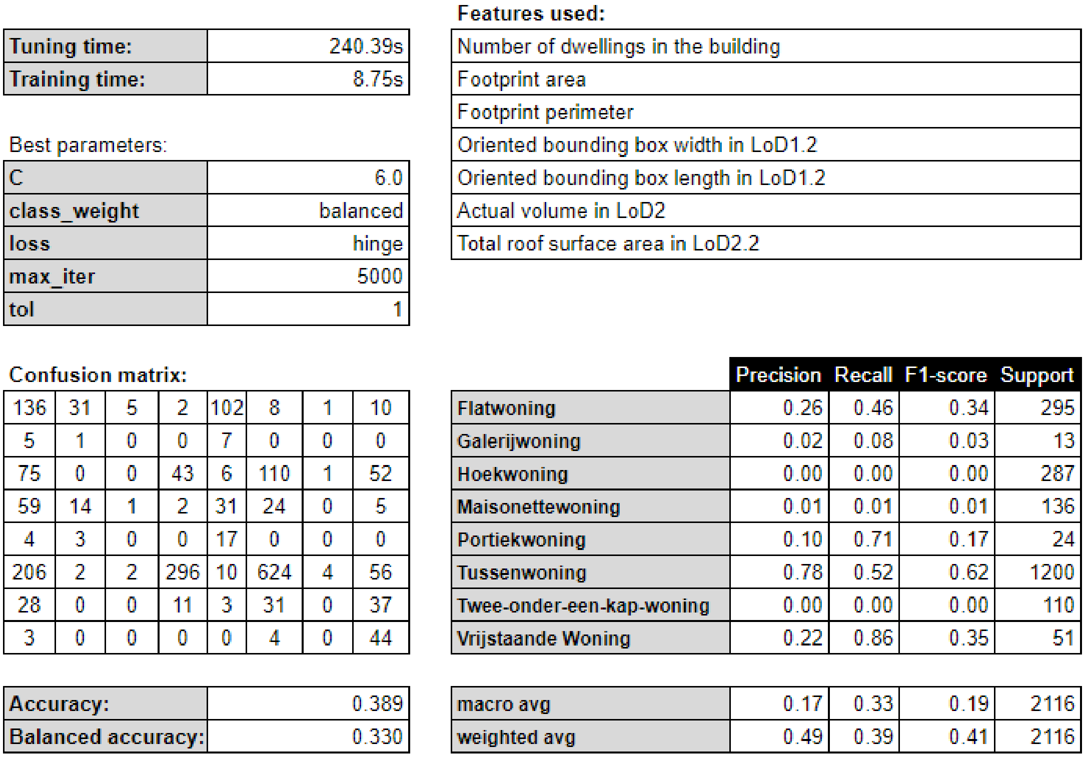
190	9	0	56	21	11	0	8
11	1	0	0	1	0	0	0
3	0	108	3	0	128	14	31
82	2	0	34	8	7	0	3
20	0	0	2	2	0	0	0
6	0	131	5	0	1037	4	17
2	0	46	0	0	30	9	23
1	0	4	0	0	6	0	40

	Precision	Recall	F1-score	Support
Flatwoning	0.60	0.64	0.62	295
Galerijwoning	0.08	0.08	0.08	13
Hoekwoning	0.37	0.38	0.38	287
Maisonettewoning	0.34	0.25	0.29	136
Portiekwoning	0.06	0.08	0.07	24
Tussenwoning	0.85	0.86	0.86	1200
Twee-onder-een-kap-woning	0.33	0.08	0.13	110
Vrijstaande Woning	0.33	0.78	0.46	51

Accuracy:	0.672
Balanced accuracy:	0.395

macro avg	0.37	0.40	0.36	2116
weighted avg	0.67	0.67	0.66	2116

Figure A.12: SVM using ANOVA-F features on **Case Study 2** test split (0.2).



Tuning time:	240.39s
Training time:	8.75s

Best parameters:

C	6.0
class_weight	balanced
loss	hinge
max_iter	5000
tol	1

Features used:

Number of dwellings in the building
Footprint area
Footprint perimeter
Oriented bounding box width in LoD1.2
Oriented bounding box length in LoD1.2
Actual volume in LoD2
Total roof surface area in LoD2.2

Confusion matrix:

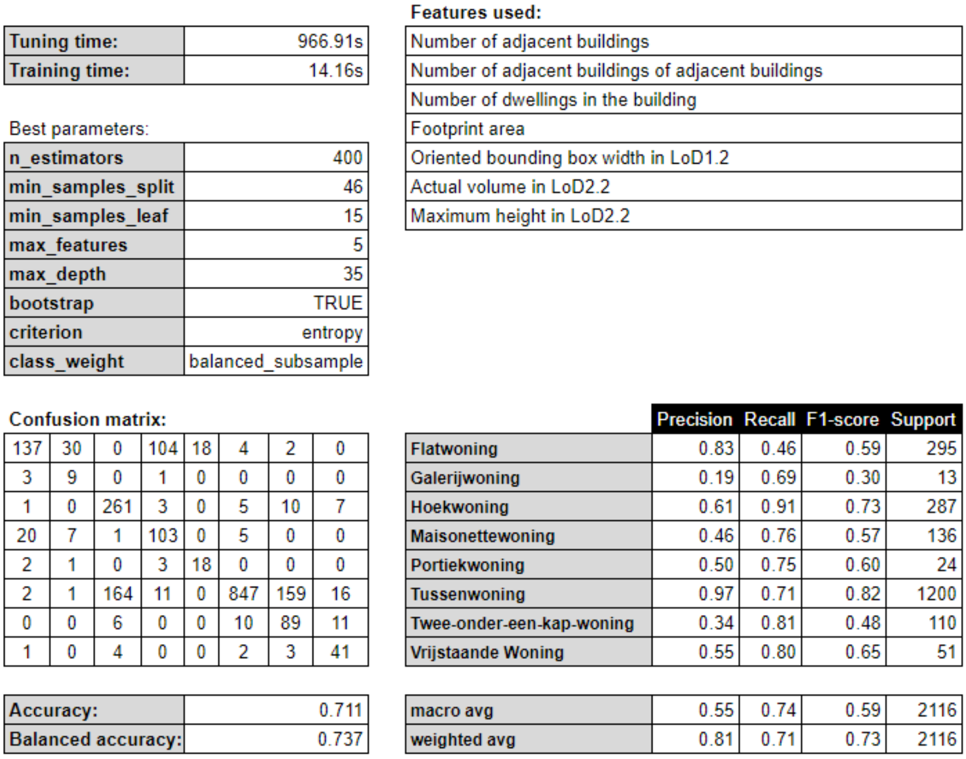
136	31	5	2	102	8	1	10
5	1	0	0	7	0	0	0
75	0	0	43	6	110	1	52
59	14	1	2	31	24	0	5
4	3	0	0	17	0	0	0
206	2	2	296	10	624	4	56
28	0	0	11	3	31	0	37
3	0	0	0	0	4	0	44

	Precision	Recall	F1-score	Support
Flatwoning	0.26	0.46	0.34	295
Galerijwoning	0.02	0.08	0.03	13
Hoekwoning	0.00	0.00	0.00	287
Maisonettewoning	0.01	0.01	0.01	136
Portiekwoning	0.10	0.71	0.17	24
Tussenwoning	0.78	0.52	0.62	1200
Twee-onder-een-kap-woning	0.00	0.00	0.00	110
Vrijstaande Woning	0.22	0.86	0.35	51

Accuracy:	0.389
Balanced accuracy:	0.330

macro avg	0.17	0.33	0.19	2116
weighted avg	0.49	0.39	0.41	2116

Figure A.13: SVM using MI features on **Case Study 2** test split (0.2).



Tuning time:	966.91s
Training time:	14.16s

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Number of dwellings in the building
Footprint area
Oriented bounding box width in LoD1.2
Actual volume in LoD2.2
Maximum height in LoD2.2

Best parameters:

n_estimators	400
min_samples_split	46
min_samples_leaf	15
max_features	5
max_depth	35
bootstrap	TRUE
criterion	entropy
class_weight	balanced_subsample

Confusion matrix:

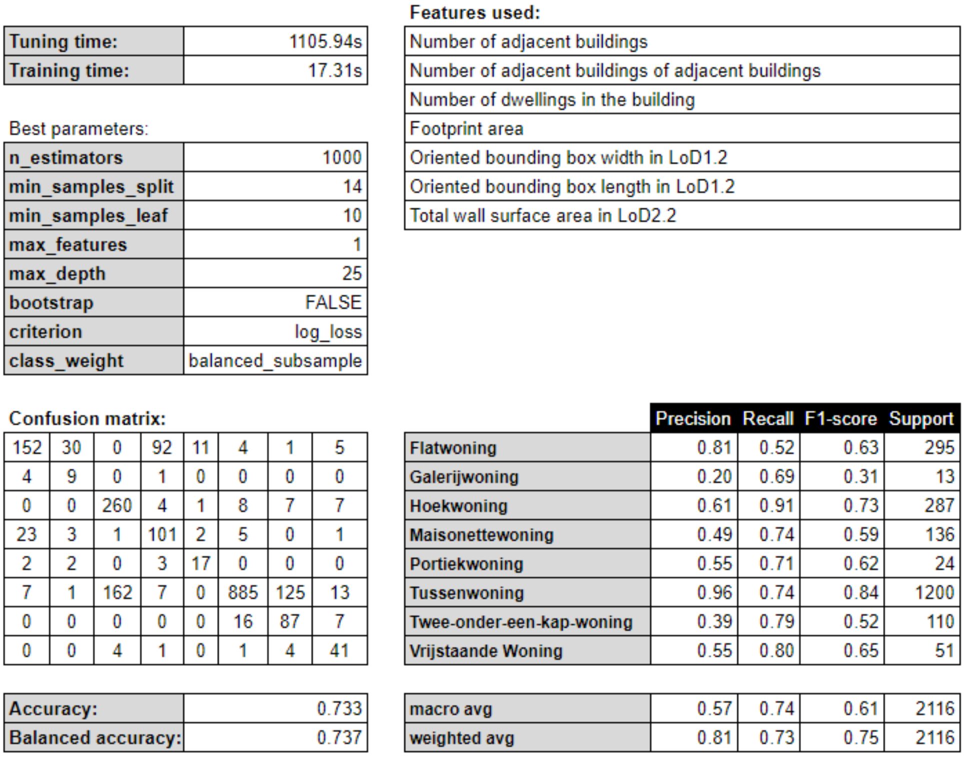
137	30	0	104	18	4	2	0
3	9	0	1	0	0	0	0
1	0	261	3	0	5	10	7
20	7	1	103	0	5	0	0
2	1	0	3	18	0	0	0
2	1	164	11	0	847	159	16
0	0	6	0	0	10	89	11
1	0	4	0	0	2	3	41

	Precision	Recall	F1-score	Support
Flatwoning	0.83	0.46	0.59	295
Galerijwoning	0.19	0.69	0.30	13
Hoekwoning	0.61	0.91	0.73	287
Maisonettewoning	0.46	0.76	0.57	136
Portiekwoning	0.50	0.75	0.60	24
Tussenwoning	0.97	0.71	0.82	1200
Twee-onder-een-kap-woning	0.34	0.81	0.48	110
Vrijstaande Woning	0.55	0.80	0.65	51

Accuracy:	0.711
Balanced accuracy:	0.737

macro avg	0.55	0.74	0.59	2116
weighted avg	0.81	0.71	0.73	2116

Figure A.14: RF using impurity features on **Case Study 2** test split (0.2).



Tuning time:	1105.94s
Training time:	17.31s

Features used:

Number of adjacent buildings
Number of adjacent buildings of adjacent buildings
Number of dwellings in the building
Footprint area
Oriented bounding box width in LoD1.2
Oriented bounding box length in LoD1.2
Total wall surface area in LoD2.2

Best parameters:

n_estimators	1000
min_samples_split	14
min_samples_leaf	10
max_features	1
max_depth	25
bootstrap	FALSE
criterion	log_loss
class_weight	balanced_subsample

Confusion matrix:

152	30	0	92	11	4	1	5
4	9	0	1	0	0	0	0
0	0	260	4	1	8	7	7
23	3	1	101	2	5	0	1
2	2	0	3	17	0	0	0
7	1	162	7	0	885	125	13
0	0	0	0	0	16	87	7
0	0	4	1	0	1	4	41

	Precision	Recall	F1-score	Support
Flatwoning	0.81	0.52	0.63	295
Galerijwoning	0.20	0.69	0.31	13
Hoekwoning	0.61	0.91	0.73	287
Maisonettewoning	0.49	0.74	0.59	136
Portiekwoning	0.55	0.71	0.62	24
Tussenwoning	0.96	0.74	0.84	1200
Twee-onder-een-kap-woning	0.39	0.79	0.52	110
Vrijstaande Woning	0.55	0.80	0.65	51

Accuracy:	0.733
Balanced accuracy:	0.737

macro avg	0.57	0.74	0.61	2116
weighted avg	0.81	0.73	0.75	2116

Figure A.15: RF using permutation features on **Case Study 2** test split (0.2).

A.1.3 Case Study 1 predictions using c2 models

44	0	0	3	5	0	0	6
4	0	0	0	0	0	0	0
6	0	548	7	3	247	110	512
2	0	0	2	0	0	0	0
6	0	0	0	2	0	0	0
4	0	24	14	7	1987	9	87
2	0	537	1	1	214	103	905
1	0	116	0	1	7	2	1637

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.64	0.76	0.69	58
Galerijwoning	0.00	0.00	0.00	4
Hoekwoning	0.45	0.38	0.41	1433
Maisonettewoning	0.07	0.50	0.13	4
Portiekwoning	0.11	0.25	0.15	8
Tussenwoning	0.81	0.93	0.87	2132
Twee-onder-een-kapwoning	0.46	0.06	0.10	1763
Vrijstaande Woning	0.52	0.93	0.67	1764
macro avg	0.38	0.48	0.38	7166
weighted avg	0.58	0.6	0.54	7166
Accuracy	0.603			
Balanced Accuracy	0.476			

Table A.5: Classification report of **c2 SVM ANOVA-F model** on **Case Study 1**

27	22	0	0	3	0	0	6
4	0	0	0	0	0	0	0
246	6	0	21	24	439	3	694
0	4	0	0	0	0	0	0
3	5	0	0	0	0	0	0
356	10	2	42	20	1017	2	683
224	3	0	15	9	293	4	1215
49	2	0	1	4	13	0	1695

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.03	0.47	0.06	58
Galerijwoning	0.00	0.00	0.00	4
Hoekwoning	0.00	0.00	0.00	1433
Maisonettewoning	0.00	0.00	0.00	4
Portiekwoning	0.00	0.00	0.00	8
Tussenwoning	0.58	0.48	0.52	2132
Twee-onder-een-kapwoning	0.44	0.00	0.00	1763
Vrijstaande Woning	0.39	0.96	0.56	1764
macro avg	0.18	0.24	0.14	7166
weighted avg	0.38	0.38	0.29	7166
Accuracy	0.383			
Balanced Accuracy	0.238			

Table A.6: **Classification report of c2 SVM MI model** on **Case Study 1**

21	32	0	1	4	0	0	0
0	4	0	0	0	0	0	0
7	1	1170	17	1	36	63	138
0	4	0	0	0	0	0	0
0	8	0	0	0	0	0	0
8	1	53	18	1	1651	61	339
8	5	35	11	0	83	1356	265
58	8	12	43	0	3	25	1615

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.21	0.36	0.26	58
Galerijwoning	0.06	1.00	0.12	4
Hoekwoning	0.92	0.82	0.87	1433
Maisonettewoning	0.00	0.00	0.00	4
Portiekwoning	0.00	0.00	0.00	8
Tussenwoning	0.93	0.77	0.85	2132
Twee-onder-een-kapwoning	0.90	0.77	0.83	1763
Vrijstaande Woning	0.69	0.92	0.78	1764
macro avg	0.46	0.58	0.46	7166
weighted avg	0.85	0.81	0.82	7166
Accuracy	0.812			
Balanced Accuracy	0.58			

Table A.7: Classification report of **c2 RF impurity model** on **Case Study 1**

36	17	0	0	5	0	0	0
0	4	0	0	0	0	0	0
11	3	1188	8	3	38	66	116
3	1	0	0	0	0	0	0
4	4	0	0	0	0	0	0
10	1	53	17	1	1689	58	303
3	1	36	12	0	92	1392	227
12	1	11	0	1	4	28	1707

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.46	0.62	0.53	58
Galerijwoning	0.12	1.00	0.22	4
Hoekwoning	0.92	0.83	0.87	1433
Maisonettewoning	0.00	0.00	0.00	4
Portiekwoning	0.00	0.00	0.00	8
Tussenwoning	0.93	0.79	0.84	2132
Twee-onder-een-kapwoning	0.90	0.79	0.84	1763
Vrijstaande Woning	0.73	0.97	0.83	1764
macro avg	0.51	0.62	0.52	7166
weighted avg	0.86	0.84	0.84	7166
Accuracy	0.840			
Balanced Accuracy	0.625			

Table A.8: Classification report of **c2 RF permutation model** on **Case Study 1**

A.1.4 Case Study 2 predictions using c1 models

379	0	401	0	0	537	139	20
57	0	3	0	0	2	1	0
0	0	1351	0	0	16	67	3
66	0	196	0	0	315	99	5
48	0	22	0	0	39	8	1
1	4	1210	0	0	3428	1353	3
0	2	1	0	0	11	534	4
0	0	27	0	0	25	44	157

	precision	recall	f1-score	support
Flatwoning	0.69	0.26	0.37	1476
Galerijwoning	0.00	0.00	0.00	63
Hoekwoning	0.42	0.94	0.58	1437
Maisonettewoning	0.00	0.00	0.00	681
Portiekwoning	0.00	0.00	0.00	118
Tussenwoning	0.78	0.57	0.66	5999
Twee-onder-een-kapwoning	0.24	0.97	0.38	552
Vrijstaande Woning	0.81	0.62	0.70	253
macro avg	0.37	0.42	0.34	10579
weighted avg	0.63	0.55	0.54	10579
Accuracy	0.553			
Balanced Accuracy	0.42			

Table A.9: Classification report of **c1 SVM ANOVA-F model** on **Case Study 2**

344	0	369	0	3	600	118	42
50	0	7	0	0	2	1	3
0	0	1304	0	0	69	62	2
59	0	174	0	1	354	80	13
31	0	40	0	0	36	8	3
2	0	1076	0	0	3694	1226	1
0	0	1	0	0	37	511	3
17	0	26	0	0	31	39	140

	precision	recall	f1-score	support
Flatwoning	0.68	0.23	0.35	1476
Galerijwoning	0.00	0.00	0.00	63
Hoekwoning	0.44	0.91	0.59	1437
Maisonettewoning	0.00	0.00	0.00	681
Portiekwoning	0.00	0.00	0.00	118
Tussenwoning	0.77	0.62	0.68	5999
Twee-onder-een-kapwoning	0.25	0.93	0.39	552
Vrijstaande Woning	0.68	0.55	0.61	253
macro avg	0.35	0.40	0.33	10579
weighted avg	0.62	0.57	0.55	10579
Accuracy	0.566			
Balanced Accuracy	0.404			

Table A.10: Classification report of **c1 SVM MI model** on **Case Study 2**

222	175	247	10	151	429	214	28
7	28	3	6	16	2	1	0
0	0	1324	0	0	18	92	3
27	26	168	6	14	307	125	8
31	22	12	4	14	22	12	1
4	0	1071	1	0	3486	1435	2
0	0	1	0	0	3	546	2
0	0	25	5	0	24	48	151

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.68	0.23	0.35	1476
Galerijwoning	0.00	0.00	0.00	63
Hoekwoning	0.44	0.91	0.59	1437
Maisonettewoning	0.00	0.00	0.00	681
Portiekwoning	0.00	0.00	0.00	118
Tussenwoning	0.77	0.62	0.68	5999
Twee-onder-een-kapwoning	0.25	0.93	0.39	552
Vrijstaande Woning	0.68	0.55	0.61	253
macro avg	0.35	0.40	0.33	10579
weighted avg	0.62	0.57	0.55	10579
Accuracy	0.566			
Balanced Accuracy	0.404			

Table A.11: Classification report of **c1 RF impurity model** on **Case Study 2**

306	195	237	9	160	406	144	19
10	12	3	1	34	2	1	0
0	0	1348	0	0	18	68	3
45	24	165	3	19	324	97	4
62	27	5	0	6	10	8	0
0	0	1075	0	1	3560	1361	2
0	0	1	0	0	8	541	2
0	0	25	0	0	27	46	155

	precision	**recall**	**f1-score**	**support**
Flatwoning	0.72	0.21	0.32	1476
Galerijwoning	0.05	0.19	0.07	63
Hoekwoning	0.47	0.94	0.63	1437
Maisonettewoning	0.23	0.00	0.01	681
Portiekwoning	0.03	0.05	0.04	118
Tussenwoning	0.82	0.59	0.69	5999
Twee-onder-een-kapwoning	0.24	0.98	0.38	552
Vrijstaande Woning	0.84	0.61	0.71	253
macro avg	0.42	0.45	0.36	10579
weighted avg	0.68	0.56	0.56	10579
Accuracy	0.561			
Balanced Accuracy	0.447			

Table A.12: Classification report of **c1 RF permutation model** on **Case Study 2**

References

Agentschap NL. (2011). Voorbeeldwoningen 2011: Bestaande bouw..

Agentschap NL. (2013). Referentiewoningen nieuwbouw 2013..

Allecijfers.nl. (2023a). *Gemeente Delft in cijfers en grafieken (bijgewerkt 2023!).* Retrieved from `https://allecijfers.nl/gemeente/delft/`

Allecijfers.nl. (2023b). *Gemeente Rijssen-Holten in cijfers en grafieken (bijgewerkt 2023!).* Retrieved from `AlleCijfers.nl.https://allecijfers.nl/gemeente/rijssen-holten/`

Biljecki, F., & Chow, Y. S. (2022, July). Global Building Morphology Indicators. *Computers, Environment and Urban Systems*, *95*, 101809. doi: 10.1016/j.compenvurbsys.2022.101809

Borne, M., Mercier, H., Mora, V., & Courtin, O. (2023). *SFCGAL.* Retrieved from `https://sfcgal.gitlab.io/SFCGAL/index.html`

Branco, P., Torgo, L., & Ribeiro, R. (2015, May 14). *A Survey of Predictive Modelling under Imbalanced Distributions.* arXiv. doi: 10.48550/ARXIV.1505.01658

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Chandrashekar, G., & Sahin, F. (2014, January 7). A survey on feature selection methods. *Computers and Electrical Engineering*, *40*(1), 16–28. doi: 10.1016/j.compeleceng.2013.11.024

Cristianini, N., & Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press. doi: 10.1017/cbo9780511801389

Episcope. (2012). *IEE Project TABULA.* Retrieved from `https://episcope.eu/iee-project/tabula/`

Episcope. (2013). *IEE Project TABULA webtool.* Retrieved from `https://episcope.eu/building-typology/webtool/`

Episcope. (2014). *IEE Project TABULA: The Netherlands, Residential Building Typology.* Retrieved from `https://episcope.eu/building-typology/country/nl.html`

Fawcett, T. (2006, June). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. doi: 10.1016/j.patrec.2005.10.010

Ferrando, M., Causone, F., Hong, T., & Chen, Y. (2020). Urban building energy modeling (UBEM) tools: A state-of-the-art review of bottom-up physics-based approaches. *Sustainable Cities and Society*, *62*, 102408. doi: 10.1016/j.scs.2020.102408

Gemeente Amsterdam. (2023). *BAG en BAG+ gegevens.* Retrieved from `https://www.amsterdam.nl/stelselpedia/bag-index/handboek-inwinnen/introductie-bag/registratie/`

Goodfellow, I., Bengio, Y., & Courville Contents, A. (2013). Deep Learning. In (Vol. 35, pp. 1902–1914). doi: 10.1109/tpami.2012.273

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.* O'Reilly.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020, September). Array programming with NumPy. *Nature*, *585*(7825), 357–362. Retrieved from `https://doi.org/10.1038/s41586-020-2649-2` doi: 10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. doi: 10.1109/MCSE.2007.55

Kadaster. (2015). Woningtypering..

Kadaster. (2023a, January). *BAG 2.0 Extract.* Retrieved from `https://www.kadaster.nl/zakelijk/producten/adressen-en-gebouwen/bag-2.0-extract`

Kadaster. (2023b). *overige gebruiksfunctie - Kadaster: bag.* Retrieved from `https://catalogus.kadaster.nl/bag/nl/page/OverigeGebruiksfunctie`

Kuhn, M., & Johnson, K. (2013). *Applied Predictive Modeling.* Springer New York. doi: 10.1007/978-1-4614-6849-3

Labetski, A., Vitalis, S., Biljecki, F., Ohori, K. A., & Stoter, J. (2022, August). 3D building metrics for urban morphology. *International Journal of Geographical Information Science*, *37*(1), 36–67. doi: 10.1080/13658816.2022.2103818

Ledoux, H. (2018). val3dity: validation of 3D GIS primitives according to the international standards. *Open Geospatial Data, Software and Standards*, *3*(1), 1. doi: http://dx.doi.org/10.1186/s40965-018-0043-x

León-Sánchez, C., Agugiaro, G., & Stoter, J. (2022, October). Creation of a citygml-based 3d city model testbed for energy-related applications. *The International Archives of the Photogrammetry,* *Remote Sensing and Spatial Information Sciences*, *XLVIII-4/W5-2022*, 97–103. doi: 10.5194/isprs-archives-xlviii-4-w5-2022-97-2022

Müller, A. C. (2017). *Introduction to machine learning with Python* (First edition ed.; S. Guido, Ed.). Sebastopol, CA: O'Reilly Media.

pandas development team, T. (2020, February). *pandas-dev/pandas: Pandas.* Zenodo. Retrieved from https://doi.org/10.5281/zenodo.3509134 doi: 10.5281/zenodo.3509134

Pearson, K. (1904). *Mathematical Contributions to the Theory of Evolution.* Dulau and co., 37 Soho Square, W. Retrieved from http://ia600408.us.archive.org/18/items/cu31924003064833/cu31924003064833.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peters, R., Dukai, B., Vitalis, S., van Liempt, J., & Stoter, J. (2022). *Automated 3D reconstruction of LoD2 and LoD1 models for all 10 million buildings of the Netherlands.* arXiv. doi: 10.48550/ARXIV.2201.01191

PostGIS. (2023). *PostGIS.* Retrieved from https://postgis.net/

PostgreSQL. (2023). *PostgreSQL.* Retrieved from https://www.postgresql.org/

Psycopg2. (2023). *Psycopg - PostgreSQL database adapter for Python.* Retrieved from https://psycopg.com

Rijksdienst voor Ondernemend Nederland. (2023). Voorbeeldwoningen 2022: Bestaande bouw.. Retrieved from https://www.rvo.nl/sites/default/files/2023-01/brochure-voorbeeldwoningen-bestaande-bouw-2022.pdf

Rijksoverheid. (2023, January). *EP-Online.* Retrieved from https://www.ep-online.nl

Rouault, E., Warmerdam, F., Schwehr, K., Kiselev, A., Butler, H., Łoskot, M., … Miura, H. (2023). *GDAL.* Zenodo. doi: 10.5281/ZENODO.5884351

Roy, E., Pronk, M., Agugiaro, G., & Ledoux, H. (2023, December). Inferring the number of floors for residential buildings. *International Journal of Geographical Information Science*, *37*(4), 938–962. doi: 10.1080/13658816.2022.2160454

Scikit-learn. (2023a). *1.11. Ensemble methods — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/ensemble.html#random-forests

Scikit-learn. (2023b). *1.13. Feature selection — scikit-learn 1.3.1 documentation.* Retrieved from https://scikit-learn.org/stable/modules/feature_selection.html#univariate-feature-selection

Scikit-learn. (2023c). *1.4. Support Vector Machines — scikit-learn 1.3.1 documentation.* Retrieved from https://scikit-learn.org/stable/modules/svm.html#classification

Scikit-learn. (2023d). *3.3.2.2. Accuracy score.* Retrieved from https://scikit-learn.org/stable/modules/model_evaluation.html#accuracy-score

Scikit-learn. (2023e). *3.3.2.4. Balanced accuracy score.* Retrieved from https://scikit-learn.org/stable/modules/model_evaluation.html#balanced-accuracy-score

Scikit-learn. (2023f). *3.3.2.9. Precision, recall and F-measures.* Retrieved from https://scikit-learn.org/stable/modules/model_evaluation.html#precision-recall-f-measure-metrics

Scikit-learn. (2023g). *4.2. Permutation feature importance — scikit-learn 1.3.1 documentation.* Retrieved from https://scikit-learn.org/stable/modules/permutation_importance.html#relation-to-impurity-based-importance-in-trees

Scikit-learn. (2023h). *6.3. Preprocessing data — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/preprocessing.html#mapping-to-a-gaussian-distribution

Scikit-learn. (2023i). *Comparing randomized search and grid search for hyperparameter estimation.* Retrieved from https://scikit-learn.org/stable/auto_examples/model_selection/plot_randomized_search.html#

Scikit-learn. (2023j). *sklearn.metrics.balanced_accuracy_score — scikit-learn 1.3.1 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html

Scikit-learn. (2023k). *sklearn.metrics.f1_score — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score

Scikit-learn. (2023l). *sklearn.metrics.precision_score — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision _score.html

Scikit-learn. (2023m). *sklearn.metrics.recall_score — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html

Scikit-learn. (2023n). *sklearn.model_selection.RandomizedSearchCV — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html#sklearn.model_selection.RandomizedSearchCV

Scikit-learn. (2023o). *sklearn.model_selection.StratifiedKFold — scikit-learn 1.3.0 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html

Scikit-learn. (2023p). *sklearn.svm.SVC — scikit-learn 1.3.1 documentation.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

Tangirala, S. (2020). Evaluating the Impact of GINI Index and Information Gain on Classification using Decision Tree Classifier Algorithm. *International Journal of Advanced Computer Science and Applications*, *11*(2). doi: 10.14569/ijacsa.2020.0110277

United Nations. (2015a). *Goal 11: Make cities inclusive, safe, resilient and sustainable.* Retrieved from https://www.un.org/sustainabledevelopment/cities/

United Nations. (2015b). *The Sustainable Development Agenda.* Retrieved from https://www.un.org/sustainabledevelopment/development-agenda/

Weiss, G. M. (2013, June 21). Foundations of Imbalanced Learning. In (pp. 13–41). Wiley. doi: 10.1002/9781118646106.ch2

Yao, Z., Nagel, C., Kunde, F., Hudra, G., Willkomm, P., Donaubauer, A., . . . Kolbe, T. H. (2018, May). 3DCityDB - a 3D geodatabase solution for the management, analysis, and visualization of semantic 3D city models based on CityGML. *Open Geospatial Data, Software and Standards*, *3*(1). doi: 10.1186/s40965-018-0046-7