buildings

MDPI

Article

Automatic Classification and Coding of Prefabricated Components Using IFC and the Random Forest Algorithm

Zhao Xu [1,*], Zheng Xie [2], Xuerong Wang [3] and Mi Niu [1]

[1] Department of Civil Engineering, Southeast University, Nanjing 210000, China; niumi@seu.edu.cn
[2] Department of Software Engineering, Southeast University, Suzhou 215000, China; 220205635@seu.edu.cn
[3] School of Information Engineering, Nanjing Audit University, Nanjing 210000, China; wangxuerong4079@126.com
* Correspondence: xuzhao@seu.edu.cn

Abstract: The management of prefabricated component staging and turnover lacks the effective integration of informatization and complexity, as relevant information is stored in the heterogeneous systems of various stakeholders. BIM and its underlying data schema, IFC, provide for information collaboration and sharing. In this paper, an automatic classification and coding system for prefabricated building, based on BIM technology and Random Forest, is developed so as to enable the unique representation of components. The proposed approach starts with classifying and coding information regarding the overall design of the components. With the classification criteria, the required attributes of the components are extracted, and the process of attribute extraction is illustrated in detail using wall components as an example. The Random Forest model is then employed for IFC building component classification training and testing, which includes the selection of the datasets, the construction of CART, and the voting of the component classification results. The experiment results illustrate that the approach can automate the uniform and unique coding of each component on a Python basis, while also reducing the workload of designers. Finally, based on the IFC physical file, an extended implementation process for component encoding information is designed to achieve information integrity for prefabricated component descriptions. Additionally, in the subsequent research, it can be further combined with Internet-of-Things technology to achieve the real-time collection of construction process information and the real-time control of building components.

Keywords: prefabrication; IFC; Random Forest; automatic coding; information classification

Citation: Xu, Z.; Xie, Z.; Wang, X.; Niu, M. Automatic Classification and Coding of Prefabricated Components Using IFC and the Random Forest Algorithm. *Buildings* **2022**, *12*, 688. https://doi.org/10.3390/buildings12050688

Academic Editors: Yuan Chen, Xianfei Yin, Bo Xiao, Yinghua Shen and Hexu Liu

Received: 21 April 2022
Accepted: 18 May 2022
Published: 20 May 2022

1. Introduction

Construction is the leading industry in China's economy, and it is essential in promoting national economic development. As an environmentally friendly, energy-saving, and efficient building method, prefabricated buildings compensate for the low productivity, poor production environments, and high safety risks encountered in the traditional construction industry. They can also meet the development requirements of the modern construction industry [1]. Assembly uses off-site construction, where prefabricated components are produced in a factory and are then transported to the construction site for assembly [2]. Compared with traditional building construction methods, the prefabricated components and assembly structures of prefabricated buildings are complex and delicate, involving many related variables, large-scale production, and several uncertainties in the process of production and assembly. Therefore, it is necessary to consider the efficient integration of informatization and the delicate management of prefabricated components. The construction organization of prefabricated buildings requires planning for the production, transportation, stacking, and assembly of components, in which the production and transportation of the prefabricated components are not performed at

the construction site [3]. It is necessary to apply building information modeling (BIM) technology to the above processes in order to facilitate the collaboration and sharing of information among all the participants in the life cycle of the building and to perform comprehensive information management for the building. BIM has been used for the parametric design [4], automated scheduling and control [5], life cycle assessment (LCA), and life cycle costing (LCC) [6] of industrialized buildings. Therefore, the combination of BIM technology and prefabricated buildings can promote the unification of informatization and industrialization [7]. In the application of BIM technology, several BIM tools adopt different data standards, and the interoperability between different software is poor. Thus, standards and interactivity can present obstacles to the development of BIM [8]. A study also highlights that the interoperability of BIM is crucial for the integration of the industrialized construction process [9]. The effective way to solve this problem is to use a unified standard data storage format. Industry Foundation Classes (IFC) offer an open and neutral ISO standard, which play a crucial role in enabling interoperability [10].

A component is considered to be the basic unit of a prefabricated building. Due to the characteristics of large quantities, variety, heaviness, complex structure, and the large amount of information required for prefabricated components, problems such as the storage and utilization of component-parameterized information and the component entity model have become prominent. Thus, the efficient management of component information has become essential to ensure the smooth development of the entire assembly project. Modern information and communication technologies, such as radio frequency [11], two-dimensional code, BIM, coding, and component database technologies, provide technical support for component tracking and the optimization of supply chain operation. Jeong et al., mention that the management of prefabricated components can efficiently extract the information of prefabricated components from IFC physical files and correctly classify prefabricated components, thus improving the efficiency of production management, reducing costs, and improving the quality of the prefabricated components [12]. The coding of prefabricated components after classification is conducted by computer identification and processing [13]. Through this process, the characteristics and attributes of individual components are classified, expressed by code, and then combined according to the actual meaning in order to perform the unique expression of specific objects. Coding is the process of assigning a code to something. It is a technical means of unifying people's understanding and exchanging information, minimizing misunderstandings and losses caused by the inconsistent naming and description of information. Information coding is directly related to the efficiency and level of automation of information processing, transmission, and retrieval.

While some studies have attempted to code prefabricated components, traditional manual coding requires a lot of time and human resources, and it is error-prone. Some researchers have explored the use of machine learning (ML) and natural language recognition (NLP) technologies to perform the automatic coding of project information [14]. In contrast to the current manual coding method, intelligent coding technology can make use of the efficient computing power and logical analytic ability of the algorithm to initialize, organize, and program the building model information according to pre-customized coding rules and specifications, and perform automatic coding. This paper proposes a method for component automatic classification and coding by machine learning. This study aims at performing the intelligent coding of prefabricated components in prefabricated buildings. The core idea of the proposed approach consists in using the component attributes in BIM models in order to train machine learning models to recognize and distinguish BIM model elements. Given a new IFC model file, the trained models can automatically classify and code the components. The process can be divided into three main steps. The first step consists of initializing the model, extracting the component attribute characteristics from the IFC file, obtaining the relevant information of the prefabricated component, and performing intelligent identification of the component. The second step

consists of training the Random Forest model to perform automatic coding according to pre-established component coding principles. Finally, through the IFC extension, the coding results are output in an IFC file, and the coding is completed. The advent of automatic coding provides new opportunities to further exploit the efficiency of prefabricated components. In the BIM technical standard, information coding is a process of giving information element codes in information processing in order to facilitate the storage, retrieval, and use of information. Through the collection and acquisition of component information, the dynamic management of prefabricated components is performed to solve the problem of information asymmetry in the process of construction management. The proposed method is of great significance to the actual prefabricated building construction process and provides technical support for the accurate management and long-term development of prefabricated buildings.

2. Literature Review

2.1. Intelligent Management of Prefabricated Components

With the development of prefabricated buildings, problems caused by the extensive management mode in the use of prefabricated components have gradually emerged. Some researchers have begun to pay attention to the studies on the information and intelligent management of prefabricated components. The application of BIM technology in the prefabricated building industry is the main content of prefabricated component information research, which can integrate all of the information in the whole life cycle of prefabricated buildings [15]. BIM has been used widely in computer design [7], sustainable development [16], on-site logistics planning and control [17], etc. However, the application of BIM technology in prefabricated building engineering is not enough to further improve the collaborative work and decision-making efficiency of on-site assembly services. In other words, the information sharing efficiency of building components in Building Lifecycle Management needs to be improved. BIM technology gradually integrates various new information technologies to be applied in prefabricated buildings. Solihin et al., have effectively transformed BIM data into an open and queryable database for better visibility into BIM data [18]. More studies on BIM and RFID integration can be seen from Majrouhi [19], who identifies and tracks construction materials using radio frequency identification (RFID) technology. Li et al., have also developed an IoT-enabled platform by integrating the Internet of Things and BIM technologies to supervise the construction progress and approximate cost information in a real-time manner for providing various decision support tools and services to different stakeholders [20].

The previously mentioned study discusses the application prospect of information technology, such as BIM, in each stage of prefabricated construction engineering, but it lacks attention to the information flow in the supply chain of prefabricated components. In order to perform the information sharing of assembly components, it is necessary to code the component information into a unified and identifiable language.

In this paper, the prefabricated components of prefabricated buildings are reclassified to form a coding system suitable for prefabricated components of prefabricated buildings, with reference to the OmniClass information classification method to realize the unique identification and the integrity of information descriptions of prefabricated components based on BIM.

2.2. Information Classification and Coding System

The Building Information Classification System (BICS) was fully developed in the USA and Europe in the early 20th century. Due to differences in national laws, regulations, and building environments, significant differences exist in the classification methods, classification structures, and scope of application in the different regional classification systems. These differences seriously hinder the application of information technology within the construction context and have a negative impact on the international development of the construction industry. In the 1990s, in order to make IT more fully developed in the

construction industry, a number of developed countries and the International Standards Organization (ISO) issued classification systems, such as ISO 12006, based on studies on building information systems. The ISO 12006-2 standard for building classification is intended to be used by organizations in order to develop a framework for a building classification system on a national or regional basis. The United States gradually established relatively complete engineering project coding systems, such as Uniformat II and MasterFormat in the 1970s, which have had far-reaching influence in North America and achieved constancy and use in a wide range of applications. Among them, the positioning of UNIFORMAT II is for the full cycle of engineering projects, and the coding structure is used for description, cost analysis, and project management [21], while it does not include civil engineering components. UNIFORMAT II uses a mixed coding approach, combining letters and numbers, with the code levels reflecting the classification levels. MasterFormat is jointly published by the Canadian Building Code Association and the American Institute of Architects. Its classification is for trades/materials, and it is mainly designed for North American construction projects and serves as a platform for communication between architects, contractors, subcontractors, and suppliers. The MasterFormat classification system uses numeric coding, with the four levels identified by eight digits. OmniClass has been developed and maintained by the Construction Specifications Institute (CSI) and the Canadian Construction Specifications Council (CSC) in order to meet the need for information classification for all disciplines and construction projects throughout the whole-body life cycle. This is a complete building information classification exercise based on the existing building classification systems, such as MasterFormat, UNIFORMAT II, and EPIC, for example. OmniClass is intended to be the means for organizing, sorting, and retrieving information and deriving relational computer applications [22]. Currently, several developed countries, such as the United States, the United Kingdom, Europe, Canada, Singapore, and other countries, have developed a unified construction project coding system as required by the domestic construction industry. The application of coding in project management plays a crucial role in promoting the development of the construction industry. Several domestic and international standards for construction information classification methods and coding systems exist. Each one of them has typical characteristics, significant differences, and connections.

UNIFORMAT II and OmniClass play a crucial role in integrating the industry's new, dynamic BIM technologies. Currently, there are no good standards to standardize the information classification and coding systems for prefabricated building components in China. From this perspective, based on existing information classification and domestic and international coding systems, this study investigates coding methods for prefabricated buildings and proposes a classification method which is suitable for the characteristics of prefabricated buildings in order to better suit the trending construction industry.

2.3. Random Forest

The use of machine learning methods to classify BIM objects has important research and practical implications. The existing studies focus on exploring BIM information, text, or image recognition classification. A method is proposed to automatically categorize BIM case studies by measuring and comparing the similarities between the definitions of BIM uses and phrases, using unsupervised learning methods [23]. Support vector machines are used to classify building elements for checking the semantic integrity of building information models [14]. Yuan et al., develop a terrestrial, laser-scanner-data-based classification method for common building materials using machine learning techniques [24]. The Random Forest (RF) algorithm was proposed by Breiman in 2001 [25]. As an important part of machine learning methods, artificial neural networks (especially back-propagation (BP) neural networks) [26], Random Forest and support vector machine (SVM) have been widely used for prediction. In the 1980s, a classification tree algorithm was proposed. It greatly reduces the amount and time of computation by repeating the classification or regression

of binary data. In 2001, some researchers combined many classification trees into random forests [27], which enhances the prediction ability of Random Forest without increasing the calculation amount. Random Forest is an efficient prediction model in the machine learning algorithm. It is an effective combination of a combinatorial classifier algorithm and a decision-tree classification algorithm. Compared with other classification algorithms, Random Forest integrates single classifiers for voting, which can lead to higher classification accuracy, minimize the total classification error rate and process class-imbalanced data. Therefore, it is widely used in classification prediction. Its theory and method research have been relatively mature, and it has been applied to industrial automation [28], food science [29], medicine [30], image recognition, architecture [31], and many other application domains. Upon a review of machine learning approaches based on relevant literature and experiments with multiple machine learning, Random Forest was identified as the classifier which is most suited to meet the aforementioned requirements.

This study aims to solve the above-mentioned deficiencies in the management process of prefabricated components and provide an automatic classification and coding system for prefabricated components based on BIM technology and random forests. In this way, the automatic classification and coding of prefabricated components will be realized, and the purpose of intelligent query and management of construction process information will be achieved.

3. Methodology

Figure 1 shows the overall process developed for this study. The automatic classification and coding of prefabricated components is divided into three stages. The first stage aims to create a coding system suitable for prefabricated components. The prefabricated component information is classified, and the overall design of the component coding is detailed. The second stage describes the training and testing process of a Random Forest model for the classification of IFC building components, based on the comparison of different machine learning classification models. It involves selecting features that reflect the key characteristics of prefabricated components, as well as generating a feature vector set. A classification and regression tree (CART) is then constructed, and the results of the component classification are voted on. In addition, ten classified features are analyzed with a feature index, and each feature is assigned a reasonable weight. A prefabricated component classification model based on a weighted Random Forest classification is then developed. Section 3 describes the component code information extension process. Based on the latest release of IFC4, the IFC standard format and entity representation methods are systematically analyzed. Consequently, the coded information of prefabricated components is extended using an attribute-set-based approach to achieve the integrity of BIM-based prefabricated component information description. The automatic classification and coding process of prefabricated components is finally performed, which improves the management efficiency of prefabricated components.

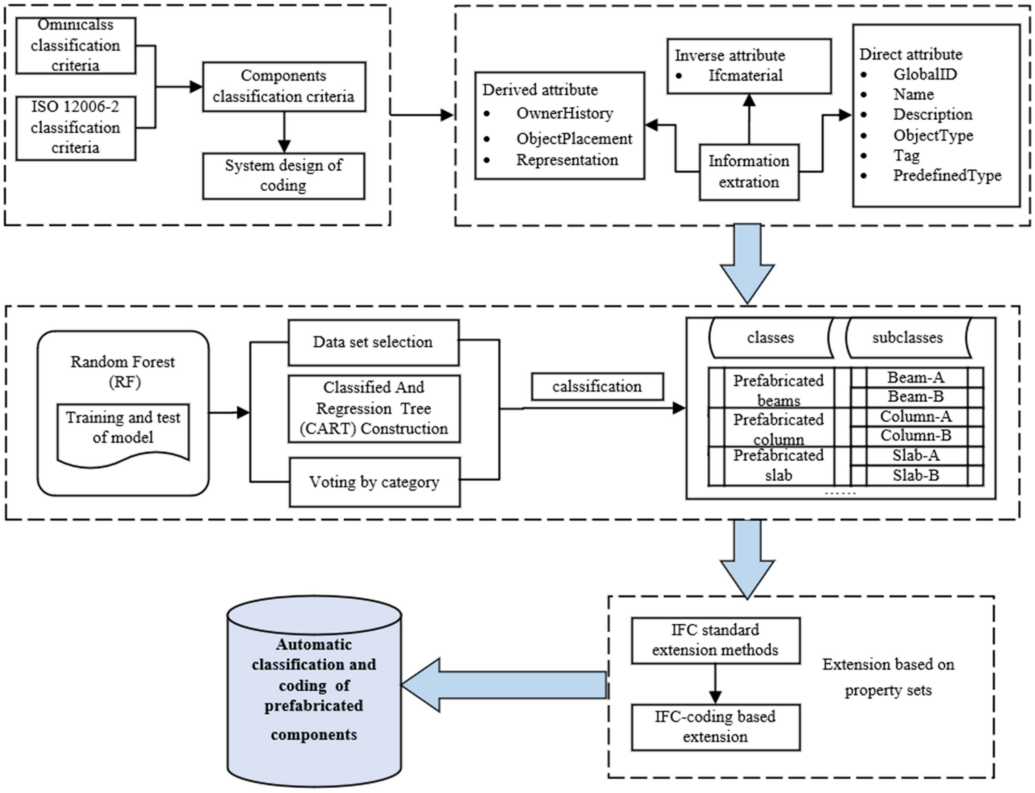

Figure 1. Overall Research Methodology.

3.1. Classification Criteria and System Design of Coding for Prefabricated Components

The classification of component information should have the characteristics of integration, systematization, and standardization. For a long time, neither a unified consensus nor unified standards have been shared among all parties involved in construction projects. This leads to poor information transfer efficiency, low sharing capacity, and the ineffective use of information, which results in information redundancy. Therefore, issues faced in the process of information management of construction projects can be improved based on the building components, thus improving the management and efficiency of each stage in order to strengthen the whole construction cycle.

3.1.1. Information Classification of Prefabricated Components

Building information should be classified and coded according to the needs of information management in the construction field. This includes relevant national standards, industry standards, and enterprise standards, as well as the characteristics of different types of information content. Certain principles and methods are used to distinguish and classify information and develop a coding system in order to ensure the uniqueness of information exchange, management, and use. The classification table that the industry has created is the OmniClass standard. Specifically, the OmniClass Construction Classification System (OCCS) is a classification system for the construction industry which incorporates many of the classification systems in use as the basis for its tables, such as the use of MasterFormat as the basis for a classification table for construction components. More precisely, the scope of OmniClass encompasses a multi-scale description of the whole built environment from erection to completion. In addition, it is applicable to the building life cycle, from conception, design, and construction to final demolition and recycling. Therefore, the OmniClass information classification system contributes to the development of assembled buildings in China, particularly in terms of information classification and integration. The classification method for prefabricated components designed in this paper is based on reference to, and borrowing from, the OmniClass classification system.

This study mainly refers to OmniClass information classification ideas for information classification. Prefabricated buildings can be divided into four main categories of structural systems: prefabricated frame, prefabricated shear wall, prefabricated frame–shear wall, and steel structure. The application scenarios of each structural system are different. The prefabricated elements used in prefabricated buildings are also different from one structural system to another. For instance, the prefabricated frame structures mainly use prefabricated beams, prefabricated columns, prefabricated floors, and prefabricated stairs. The prefabricated shear wall structures include prefabricated walls, floors, and stairs. This paper uses the prefabricated components as the object of classification and the whole life cycle of precast building structures and precast components as the main classification basis for the classification of information. An example of an assembled concrete frame structure is shown in Table 1.

Table 1. Information classification of precast integrated concrete frame structure.

Structure Type	Component Category	Component Name	Component Classification	Component Subdivision
Assemble integral concrete frame structure	Precast beam	Precast beam	Precast main beam Precast secondary beam Prestressed composite beam Ordinary composite beam	
		Precast composite beam		
		Precast lintel		
	Precast slab	Precast slab	Prestressed hollow slab Prestressed laminated slab Common laminated slab Steel bar truss laminated slab Prestressed grooved slab Common grooved slab	
		Precast balcony	Fully precast balcony Precast balcony slab	
		Precast stair	Precast slab stair Precast beam stair Precast single run stair Precast double run stair Precast scissor stair Precast clockwise stair Precast counterclockwise stair	
		Prefabricated canopy	Prefabricated aluminum alloy sunshade canopy Prefabricated plastic steel canopy Prefabricated French canopy Prefabricated mobile push–pull canopy Prefabricated shrinkage push–pull canopy	
		Prefabricated air conditioning panel		
	Precast column	Precast column		
	Precast wall	Precast exterior wall	Precast non-bearing exterior wall	Precast GRC insulation exterior wall panel Precast sandwich insulation exterior wall panel Precast single exterior wall panel Precast hanging exterior wall panel Precast side connected exterior wall panel
		Precast wall hanging panel	Precast fiber cement exterior wall hanging panel Precast metal wall hanging panel Precast PVC wall hanging panel Precast solid wood wall hanging panel Precast stone wall hanging panel	
		Precast interior wall panel	Precast non-load bearing interior panel	Precast solid interior wall panel Precast hollow interior wall panel
	Prefabricated window	Precast integrated door and window wall panel	Precast integrated door and window solid wall panel Precast integrated door and window laminated wall panel	
		Precast parapet		
		Prefabricated bay window	Prefabricated inner bay window Prefabricated exterior bay window	
		Ordinary prefabricated window		

3.1.2. Component Coding System

In this study, the OmniClass classification system is used to implement the coding of assembled prefabricated components. According to the characteristics of assembled buildings, the coding system is created based on the component as unit. The code combines the idea of tracking information throughout the life cycle of a project. The first level of numbering distinguishes the life cycle by production, construction, operation, and maintenance. The second level of coding distinguishes the structure by frame, shear wall, frame shear wall, steel structure, and connections. The third, fourth, and fifth level of coding then indicate the large, medium, and small categories of components, in that order. Finally, the sixth level of coding segments indicates the component size information. In view of the speed of identification in production construction, a mixture of abbreviations and numbers is used for the coding. The result is a six-segment, nine-digit code. The structure of the sixth code segment, representing dimensional information in the code, is different for the different component categories. The specific structure of the sixth code segment for the different component categories is presented as follows, while all the measurements are in millimeters:

(1) Beam: The sixth code segment is composed of three numbers. From front to back, the width, height, and length of the section are respectively denoted. The numbers for each segment are connected by "-".
(2) Slab: The sixth code segment is composed of a number, which is the value of the slab thickness.
(3) Column: The sixth code segment is composed of two numbers. From front to back are, respectively, the values for the cross-section side length of the column and the length of the column, with a "-" connecting each segment of numbers.
(4) Wall: The sixth code segment is composed of a number, which is the wall thickness.
(5) Stair: The sixth code segment is composed of four numbers. The width of the ladder section, the riser height, the tread depth, and the number of risers is sequentially denoted. Each number is connected by "-".

The content of the sixth code segment should be filled in with specific information about the component. Therefore, in the table of the classification and the coding of information on the construction process, only the contents of the first five code segments are provided for the sake of simplicity and clarity. Table 2 presents the coding structure. Figure 2 shows the meaning of the variables in each formula and the relationship between them.

Table 2. Table of significance of prefabricated component code segments.

Code Segment Names	Code Segment System	Code Segment Description
Life Cycle Stages	P	Produce
Life Cycle Stages	C	Construct
Life Cycle Stages	O	Operate
Structure Type	FS	Frame structure
Structure Type	SW	Shear wall structure
Structure Type	FW	Frame–shear wall structure
Structure Type	SS	Steel structure
Structure Type	AP	Adapting piece
Component Category	BE	Beam
Component Category	SL	Slab
Component Category	CO	Column
Component Category	WA	Wall
Component Category	WI	Window
Component Name	01	Precast beam
Component Name	...	...
Component Name	16	Ordinary prefabricated window
Component Classification	01	Precast main beam
Component Classification	...	...
Component Classification	40	Ordinary prefabricated window

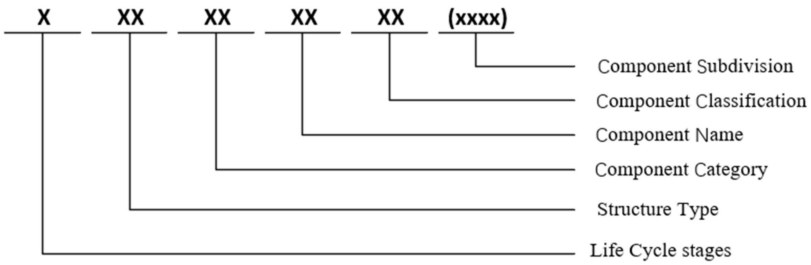

Figure 2. The coding structure.

3.2. Training and Testing of the Machine Learning

3.2.1. Information Extraction Based on IFC Schema

In component classification coding, it is necessary to obtain data information from the model that is relevant to the classification. It is necessary to first define the component information, which is detailed in the IFC standard, as well as the geometric and attribute information of the components contained in the data format. Therefore, this paper scrutinizes the basic information and system information, as well as the geometric and attribute information of the BIM model components based on the IFC standard, in order to achieve the fast and accurate extraction of information from the model. The features of each component are extracted using libraries from IfcOpenShell, an open source (LGPL) software library which is used to process the IFC file format. IfcOpenShell provides a powerful API through which IFC entities can be directly and easily retrieved. These features are then presented in tabular form for subsequent machine learning.

The physical files of the IFC standard expresses the model through entities which are the basic units of the IFC physical files. In an IFC physical file, any entity describes its information by attributes that are divided into three types: direct attributes, export attributes, and inverse attributes. The direct attributes can be directly expressed in a single entity statement in a physical file, usually as scalars or strings, such as Globalid, Name and ObjectType, for example. The export attributes are described by other entities, such as OwnerHistory, ObjectPlacement, and Representation. The inverse attributes are associated with a component through an associated entity so that the component has the attributes of the attribute entity. The material properties of a building component are inverse properties. The associated entity associates the component with the material so that the component has the material properties associated to it.

The properties of IFC entities are obtained by inheritance. For instance, the building component "IfcBeam" has 33 properties in IFC4, but only one property is a PredefinedType, while the other 32 properties are inherited from the properties contained in IfcRoot to IfcBuildingElement. In the IFC physical file, the statement IfcBeam shows only 9 attributes, including direct and exported attributes, while the remaining 24 attributes are inverse attributes (see Figure 3).

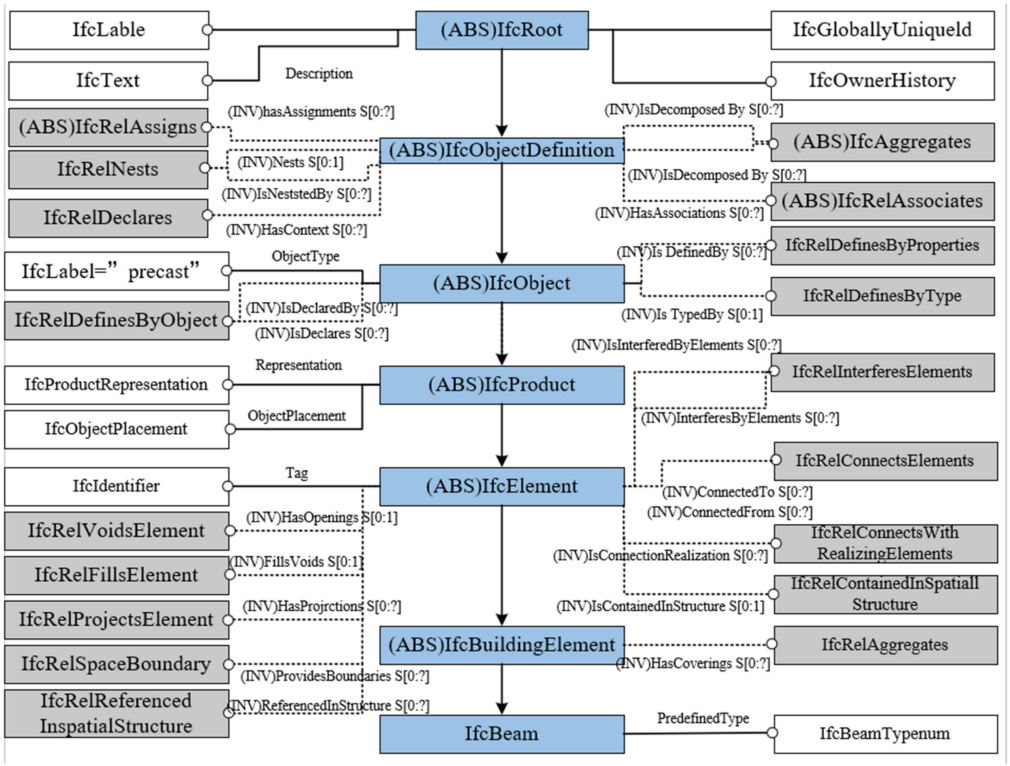

Figure 3. Definition of the IFC Architecture of a Beam.

3.2.2. Comparison of Machine Learning

Testing as to whether or not the proposed features can successfully classify prefabricated building elements is performed by creating a sample IFC dataset, generating the representative feature vectors, and then training machine learning models using the features. Considering the performance of different classifiers, the individual algorithms are tested and compared based on their prediction accuracy in order to select the classifier which most fits the building element classification. In order to train and test the machine learning for classifying building elements of IFC, the dataset is divided into training and test sets by a ratio of 8:2, which is a standard ratio used and recommended in the machine learning literature (Koo, 2019). A 10-fold cross-validation is performed on the training set. Cross-validation can be used to validate the performance of a classifier, as measured by evaluation metrics, such as accuracy, precision, recall, and *F*1-score. The accuracy represents the number of correctly classified test instances as a proportion of the total number of test instances, as shown in Equation (1). However, in the case of a positive and negative sample imbalance, evaluation by accuracy alone is not scientific. Recall is designed for the original sample and indicates how many positive classes in the sample are correctly predicted, as shown in Equation (2). The precision is the proportion of positive examples of the data predicted correctly to those predicted as positive for the predicted outcome, as shown in Equation (3). The *F*1-score is the harmonic average of the precision and recall, as shown in Equation (4). The effectiveness of machine learning in classifying IFC instances is measured using these metrics. Figure 4 shows the meaning of the variables in each formula and the relationship between them. *TP* (true positive) represents the number of predicted positives. *TN* (true negative) denotes the number of predicted negative. *FP* (false positive) is the number of predicted positives, and *FN* (false negative) represents the number of predicted negatives.

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

$$\text{Re}call = \frac{TP}{TP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$F1score = \frac{2 * precision * recall}{precision + recall} \tag{4}$$

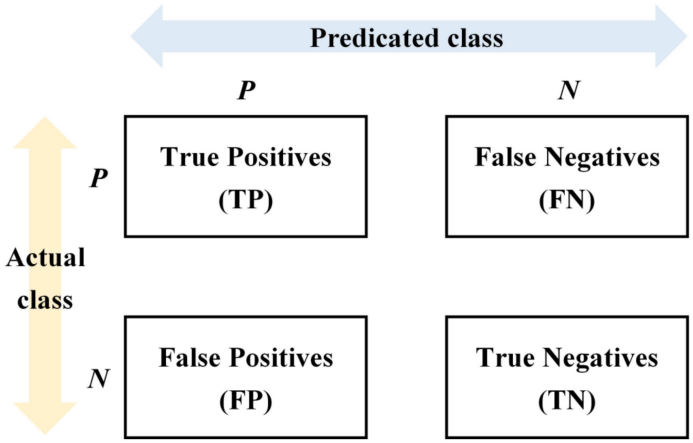

Figure 4. Truth Table Confusion Matrix.

3.2.3. Training and Testing of the Random Forest Models

Step 1: Randomly Select Datasets

The original sample set is composed of multiple types of prefabricated component data. The original sample set is divided into the training set and test set, according to a ratio of 8:2. The bootstrap sampling method is used to randomly select N subsample sets from the training set in order to construct N classification and regression trees (CARTs).

Step 2: CART Construction Based on Random Component Dataset

Since CART decision trees are used for the construction of the random forest, the Gini coefficient, which is the criterion of CART, is also used to evaluate the classification's effectiveness. For classification problems (assigning a sample to a category), i.e., discrete variable problems, CART uses the Gini value as an evaluation criterion, which is expressed as Equation (5):

$$Gini = 1 - \sum[p(i) * p(i)] \tag{5}$$

where p (i) is the proportion of class i samples in the dataset on the decision tree's node. For example, divided into 2 classes, there are 100 samples on the decision tree's node, 70 of which belong to the first category, while 30 of which belong to the second category. Thus, $Gini = 1 - 0.7 \times 0.7 - 0.3 \times 0.3 = 0.42$.This shows that the more evenly distributed the categories, the larger the *Gini* value, while the more unevenly distributed the categories, the smaller the *Gini* value.

The number of decision trees in the random forest and the maximum depth of each decision tree directly affect the final performance of Random Forest. A too-small or too-large decision tree number degrades the performance of the Random Forest prediction. The smaller the maximum depth of each decision tree, the more likely it is to be underfitted, while the larger the maximum depth, the more likely it is to be overfitted. Therefore, in this paper, the grid search method is used to determine the optimal number of decision trees and the maximum depth, while the number of decision trees is between 20 and 100 searches, and the maximum depth is between 2 and 20 searches. The grid search method performs training and testing on each combination in order to reach an optimal quantity and depth. The optimal combination in this study is a maximum depth of 16 with 69 decision trees, as shown in Figure 5:

```
In  [18]: grid_search.best_params_
Out[18]: {'max_depth': 16, 'n_estimators': 69}
```

Figure 5. Maximum Depth and Number of Decision Trees.

Step 3: Component Classification Result Voting

After the construction of n CART decision trees, the test set data is used for verification. The test sample a is input as the random forest, and the output of the K-th decision tree is calculated using Equation (5).

$$f_k(a) = i, i = \begin{cases} 0 & , \quad Not \quad such \quad a \quad component \\ 1 & , \quad Such \quad a \quad component \end{cases} \tag{6}$$

The output of the Random Forest classification model is given by:

$$f_{RF}(a) = \underset{i=0,1}{\operatorname{argmax}} \left\{ \sum_{k=1}^{n} [fk(a) = i] \right\} \tag{7}$$

where f_k is the single decision-tree classifier model, i is the classification result of a single decision tree ($i = 0$ indicates that the predicted result is a nonsuch component, and $i = 1$ indicates that the predicted result is a such component), f_{RF} is the Random Forest classification model output, and n is the Random Forest model containing the total number of decision trees.

3.2.4. Component Coding Information Extension Based on IFC

The IFC standard is open, and users can extend the IFC data model according to their actual needs by referring to the model architecture of the IFC standard. This provides three extension mechanisms: extension based on adding entity definitions, extension based on IfcProxy entities, and extension based on attribute sets.

In this paper, the code to be extended is the attribute information of the components. The components, such as columns, beams, and slabs, have already been defined in the IFC standard. Therefore, the attribute-set-based extension can be used. Figure 6 shows the flow of a property-set-based extension, where the property set is described by the IfcPropertySet entity and each property set contains at least one property, as described by the IfcProperty entity. The property sets are linked to the corresponding entities through the associated entity IfcRelDefinesByProperties, so that the entity has the property information described by the property set. Based on the latest release of IFC4, this study provides a systematic analysis of the IFC standard format and entity representation methods. On this basis, the prefabricated component coding information is extended in order to achieve the completeness of the BIM-based, prefabricated component information description.

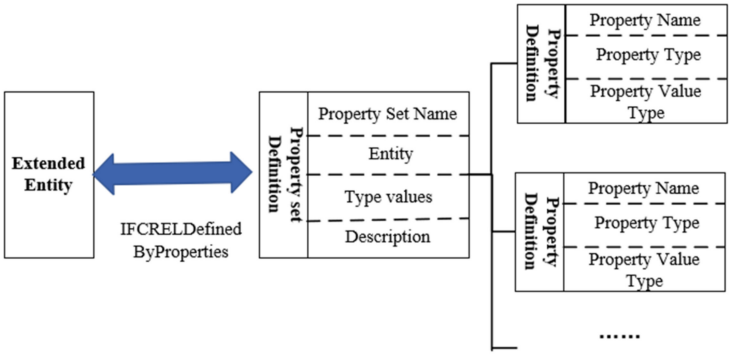

Figure 6. IFC Extension Process Based on Attribute Sets.

4. Experiment

The experiment performed in this paper is divided into four steps. The first step aims at feature extraction of wall instances from the original IFC file based on IfcOpenShell. The second step consists of of classifying and labeling the extracted data according to their characteristics. The third step consists of training the Random Forest model according to the dataset, and measuring the performance based on the accuracy of the prediction using the 10-fold cross-validation. Finally, the coding information of the component is written into the IFC physical file through the IFC extension, and the extension results are verified.

Due to the absence of an available dataset, this experiment collects some Revit models, extracts relevant attributes through IfcOpenShell, and annotates the component features through attribute Name, ObjectType, and the viewing of the location of components in BIM model. Therefore, there is the problem of a small amount of data in the experiment. On the other hand, we selected wall components for this experiment. The classification of building components is universal and can be easily extended from wall components to all building components.

4.1. Data Collection and Preparation

Nine building information models are considered for study based on their availability and scope. They are all architectural models, including villas, office buildings, flats, etc. The goal is to collect wall instances from the models for training and testing purposes. All of the wall instances are extracted using libraries from IfcOpenShell, which is an open source (LGPL) software library for working with the IFC file format. IfcOpenShell provides a powerful API through which IFC entities can be directly and easily retrieved. These features are then presented in tabular form for subsequent machine learning. The IfcWallStandardCase is considered since it has the largest number of elements and several different sizes, locations, and materials for easy classification. The attribute "Name" is a label added to the IFC entity when exporting the IFC physical file, the value of the attribute can be null, and the description of the actual component entity of this attribute usually corresponds to the modeled component "Name". As the names of different components are not usually identical, the entity name can distinguish between the components in the project. ObjectType is similar to the Name attribute. The latter is the IFC label added to the entity in the physical file. However, the difference is that the value of the ObjectType attribute is the same for components of the same type, such as columns and beams of the same section, doors and windows of the same size, etc. Therefore, the entities in IFC physical files can also be classified according to the attribute value of the ObjectType of the components. However, Name and ObjectType are not part of the input. Each instance is manually labeled by referring to the characteristic attributes, and each component is observed in the BIM software in order to construct a dataset. In fact, the location and material properties of the components are important bases for their classification. Using the different characteristics of the different prefabricated components based on walls—Name, CartesianPointX, CartesianPointY, Material, MaterialThickness, LoadBearing, External, LocationX, LocationY, and LocationZ—the position attribute, material attribute, and thickness of each material layer of the wall are related to its actual use. According to these characteristics, they can be input into the machine learning algorithm for classification. In addition, due to the greater workload of component management in prefabricated buildings, a component identification is required in the construction process, and the components can be templated. Therefore, the research object focuses on prefabricated components. In fact, different projects of cast-in-situ components have great differences. Thus, the specific classification attributes should be considered in combination with the reality. Note that this paper does not study cast-in-situ components.

4.2. Implementation of Automatic Classification and Coding for Prefabricated Building Components

4.2.1. Step 1: Data Extraction and Annotation

A total of 1718 wall instances (see Figure 7) are first collected in the model, and the following characteristic variables are selected: Name, CartesianPointX, CartesianPointY, Material, MaterialThickness, LoadBearing, External, LocationX, LocationY, and LocationZ.

writer_1

	Unnamed: 0	id	Name	CartesianPointX	CartesianPointY	Material1	Material1thickness	Material2	Material2thickness	LoadBearing	IsExternal	LocationX	LocationY	LocationZ	ObjectType	ID	Unnamed: 32	Label
0	0.0	279920	BasicWall:A-Q-200:529156	401424.435963	142498.417405	Concrete block\n	200	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	401424.435963	142498.417405	-80.0	BasicWall:A-Q-200:483071	NaN	NaN	W-10-01-01
1	1.0	279877	BasicWall:A-Q-200:529155	400524.721104	145257.179982	Concrete block\n	200	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	400524.721104	145257.179982	-80.0	BasicWall:A-Q-200:483071	NaN	NaN	W-10-01-01
2	2.0	279841	BasicWall:A-Q-200:529154	399692.080105	145059.947757	Concrete block\n	200	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	399692.080105	145059.947757	-80.0	BasicWall:A-Q-200:483071	NaN	NaN	W-10-01-01
3	3.0	279798	BasicWall:A-Q-200:529153	401424.435963	142498.417405	Concrete block\n	200	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	401424.435963	142498.417405	-80.0	BasicWall:A-Q-200:483071	NaN	NaN	W-10-01-01
4	4.0	279762	BasicWall:A-Q-200:529152	400524.721104	145257.179982	Concrete block\n	200	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	400524.721104	145257.179982	-80.0	BasicWall:A-Q-200:483071	NaN	NaN	W-10-01-01
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1713	365.0	379	BasicWall:A-Q-900mm 2:569050	110382.586852	282334.732966	Concrete - cast in situ concrete	900	NaN	0	IfcBoolean(.F.)	IfcBoolean(.F.)	110382.586852	282334.732966	0.0	BasicWall:A-Q-900mm 2:569078	NaN	NaN	W-10-01-02
1714	366.0	344	BasicWall:A-Q-300mm:568906	149889.750474	277394.427051	\nConcrete - cast in situ concrete	300	NaN	0	IfcBoolean(.F.)	IfcBoolean(.T.)	149889.750474	277394.427051	0.0	BasicWall:A-Q-300mm:476396	NaN	NaN	W-00-01-02
1715	367.0	309	BasicWall:A-Q-300mm:568829	153302.227024	272270.625768	\nConcrete - cast in situ concrete	300	NaN	0	IfcBoolean(.F.)	IfcBoolean(.T.)	153302.227024	272270.625768	0.0	BasicWall:A-Q-300mm:476396	NaN	NaN	W-00-01-02
1716	368.0	274	BasicWall:A-Q-300mm:568706	111630.599233	275510.323461	\nConcrete - cast in situ concrete	300	NaN	0	IfcBoolean(.F.)	IfcBoolean(.T.)	111630.599233	275510.323461	0.0	BasicWall:A-Q-300mm:476396	NaN	NaN	W-00-01-02
1717	369.0	204	BasicWall:A-Q-300mm:568593	111141.907737	276079.252825	\nConcrete - cast in situ concrete	300	NaN	0	IfcBoolean(.F.)	IfcBoolean(.T.)	111141.907737	276079.252825	0.0	BasicWall:A-Q-300mm:476396	NaN	NaN	W-00-01-02

1718 rows × 34 columns

Figure 7. Data Extraction.

These characteristics for each instance are extracted using libraries in the IfcOpenShell and stored in the matrix in the form of Python data boxes. The complete code is summarized as Algorithm 1 show:

Algorithm 1 Extract IFC

```
products = f.by_type('IfcProduct') #Instantiate object
for product in products:
    if(product.is_a() == 'IfcWallStandardCase'):
        dict = [['null','Name',0,0,'NULL',0,'NULL',0,'NULL',0,'NULL',0,'NULL',0,
                 'NULL',0,'NULL',0,'NULL',0,'NULL',0,'NULL',0,'T','T',0,0,0,'NULL']]
        dict[0][0] = product.id() #ID Property
        dict[0][1] = product.Name #Name Property
        dict[0][2] = product.ObjectPlacement[1][0][0][0] #CartesianPointX Property
        dict[0][3] = product.ObjectPlacement[1][0][0][1] #CartesianPointY Property
        i = 0
        #Take out the types and properties of materials 1 ~ 9 in turn
        for c in product.HasAssociations[0][5][0][0]:
            dict[0][4+i] = c[0][0] #Material type
            dict[0][5+i] = c[1] #Material thickness
            i += 2
        for definition in product.IsDefinedBy:
            if definition.is_a('IfcRelDefinesByProperties'):
                property_set = definition.RelatingPropertyDefinition
                if(property_set[2] == ('Pset_WallCommon')):
                    dict[0][24] = property_set[4][1][2]
                      #LoadBearing Property, whether or not to bear weight
                    dict[0][25] = property_set[4][3][2]
                      #IsExternal Property, Whether the representative is outside
```

Algorithm 1 Extract IFC

```
        #LocationX, LocationY, LocationZ
        dict[0][26] = product.ObjectPlacement.RelativePlacement.Location[0][0]
        dict[0][27] = product.ObjectPlacement.RelativePlacement.Location[0][1]
        dict[0][28] = product.ObjectPlacement.RelativePlacement.Location[0][2]
        #ObjectType
dict[0][29] = product.ObjectType
frame1 = pd.DataFrame(dict,columns=
('id','Name','CartesianPointX','CartesianPointY','Material1','Material1thickness','Material2',
'Material2thickness','Material3','Material3thickness','Material4','Material4thickness','Material5',
'Material5thickness','Material6','Material6thickness','Material7','Material7thickness','Material8',
'Material8thickness','Material9','Material9thickness','Material10','Material10thickness',
'LoadBearing','IsExternal','LocationX','LocationY','LocationZ','ObjectType'))
        frame = pd.concat([frame1,frame], axis = 0 ,ignore_index = True)
```

The matrix is exported to an Excel file, and each instance is manually annotated. A new column of "Class" is then created at the end of the table to annotate the data, as shown in Figure 8.

id	Name	CartesianPointX	CartesianPointY	Material1	rial1thick	LoadBearing	IsExternal	LocationX	LocationY	LocationZ	Label
279920	Basic Wall:A-Q-200:529156	401424.436	142498.4174	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	401424.4	142498.4	-80	W-10-01-01-200
279877	Basic Wall:A-Q-200:529155	400524.7211	145257.18	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	400524.7	145257.2	-80	W-10-01-01-200
279841	Basic Wall:A-Q-200:529154	399692.0801	145059.9478	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	399692.1	145059.9	-80	W-10-01-01-200
279798	Basic Wall:A-Q-200:529153	401424.436	142498.4174	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	401424.4	142498.4	-80	W-10-01-01-200
279762	Basic Wall:A-Q-200:529152	400524.7211	145257.18	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	400524.7	145257.2	-80	W-10-01-01-200
279726	Basic Wall:A-Q-200:529151	399785.8564	145094.1493	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	399785.9	145094.1	-80	W-10-01-01-200
279690	Basic Wall:A-Q-200:529150	401424.436	142498.4174	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	401424.4	142498.4	-80	W-10-01-01-200
279654	Basic Wall:A-Q-200:529149	400524.7211	145257.18	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	400524.7	145257.2	-80	W-10-01-01-200
279618	Basic Wall:A-Q-200:529148	399785.8564	145094.1493	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	399785.9	145094.1	-80	W-10-01-01-200
279583	Basic Wall:S-Q-200:529147	374290.3114	138678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374290.3	138678.8	0	W-11-01-02-200
279550	Basic Wall:S-Q-200:529146	374190.3114	141678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374190.3	141678.8	0	W-11-01-02-200
279517	Basic Wall:S-Q-200:529145	374894.0114	141778.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374894	141778.8	0	W-11-01-02-200
279475	Basic Wall:S-Q-200:529144	374290.3114	138678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374290.3	138678.8	0	W-11-01-02-200
279442	Basic Wall:S-Q-200:529143	374190.3114	141678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374190.3	141678.8	0	W-11-01-02-200
279409	Basic Wall:S-Q-200:529142	374894.0114	141778.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374894	141778.8	0	W-11-01-02-200
279367	Basic Wall:S-Q-200:529141	374290.3114	138678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374290.3	138678.8	0	W-11-01-02-200
279327	Basic Wall:S-Q-200:529140	374190.3114	141678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374190.3	141678.8	0	W-11-01-02-200
279294	Basic Wall:S-Q-200:529139	374944.0114	141778.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374944	141778.8	0	W-11-01-02-200
279210	Basic Wall:A-Q-100:528951	394235.2459	144932.5229	Block	100	IfcBoolean(.F.)	IfcBoolean(.T.)	394235.2	144932.5	0	W-10-01-01-100
277994	Basic Wall:A-Q-200:528611	399070.7415	146908.98	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	399070.7	146909	1380	W-10-01-01-200
277912	Basic Wall:A-Q-200:528596	401424.436	142498.4174	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	401424.4	142498.4	-80	W-10-01-01-200
277869	Basic Wall:A-Q-200:528595	400524.7211	145257.18	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	400524.7	145257.2	-80	W-10-01-01-200
277833	Basic Wall:A-Q-200:528594	399692.0801	145059.9478	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	399692.1	145059.9	-80	W-10-01-01-200
277788	Basic Wall:A-Q-200:528593	400660.1419	142113.2188	Block	200	IfcBoolean(.F.)	IfcBoolean(.T.)	400660.1	142113.2	-80	W-10-01-01-200
269162	Basic Wall:S-Q-200:528302	375050.3114	138778.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	375050.3	138778.8	18600	W-11-01-02-200
269129	Basic Wall:S-Q-200:528301	374290.3114	138678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374290.3	138678.8	18600	W-11-01-02-200
269096	Basic Wall:S-Q-200:528300	374190.3114	141678.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374190.3	141678.8	18600	W-11-01-02-200
269063	Basic Wall:S-Q-200:528299	374943.9458	141778.7799	Concrete	200	IfcBoolean(.T.)	IfcBoolean(.T.)	374943.9	141778.8	18600	W-11-01-02-200

Figure 8. Data Annotation.

The used format of the classification and annotation is W-AB-XX-YY-000. W represents the meaning of the wall instance, A indicates whether the component is load-bearing (0 for non-load-bearing, 1 for load-bearing), B indicates whether the component is inside the building (0 for internal walls, 1 for external walls), XX represents its category (the corresponding rules are 01 for basic walls, 02 for infill walls, 03 for parapet, 04 for decorative partition walls, and 05 for curtain walls), YY indicates the wall material (the corresponding rules are 01 for block, 02 for concrete, 03 for aluminum panels. and 04 for corrugated panels), and 000 represents the wall thickness (e.g., 150/200/300, in millimeters).

4.2.2. Step 2: Training and Testing of the Random Forest Model

Before starting the model training, a simple data cleaning and pre-processing is performed. Data normalization is performed for numerical attributes, while non-numeric attributes are converted into the form of one-hot code, and pre-processing is conducted by breaking up and rearranging the data. The models are trained and tested using the RandomForestClassifier package and the GridSearchCV package in the scikit-learn framework. Scikit-learn (sklearn) is a common, third-party module for machine learning which encapsulates common machine learning methods, and users can directly call its library functions to use powerful machine learning algorithms. The Gini coefficient is used as the

criterion for evaluation. In the RandomForestClassifier package, a random forest is formed by the default use of a CART decision tree, and, therefore, the Gini coefficient is used as a criterion. The sampling method of the random forest is considered in a put-back form, which ensures randomness and reduces the problem of overfitting to a certain extent. The Random Forest model is an integration of decision trees. The used number of decision trees and the maximum depth of each decision tree directly determine the performance of the Random Forest model. Therefore, the GridSearchCV package is used for the grid-search method, while the number of decision trees searched is between 20 and 100, and the maximum depth of decision trees searched is between 2 and 20. Ten-fold cross-validation is used to automatically divide the datasets into 8:2. The prediction accuracy on the test set is used as the evaluation index. A part of the code is provided as Algorithm 2 show:

Algorithm 2 Random Forest training

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
param_grid = [{'n_estimators':range(20,100),'max_depth':range(2,20)}]
forest_clf = RandomForestClassifier()
grid_search = GridSearchCV(forest_clf,param_grid,cv=5,scoring='accuracy')
grid_search.fit (X,Y)
```

4.2.3. Step 3: Component Coding and IFC Extension

The definition of an attribute set includes the attribute set name, the entity, the type of value, and the attribute set description. Table 3 presents the definition of the attribute set for the wall component code. The attribute set name is Pset_CodeOfWall. The definition of the attribute includes the attribute name, attribute type, and attribute value type, where the attribute is the coded value of the component. The attribute definitions are provided in Table 4.

Table 3. IFC wall component code attribute set definition.

Attribute Set Name	Pset_CodeOfWall
Entity	IFCWall
Type Value	Wall/UserDefined
Description	Description of the IFCWall entity instance coding information

Table 4. Code attribute definition.

Attribute Name	Attribute Type	Attribute Value Type
Code	IfcPropertySingleValue	IfcIdentifier

The information for the prediction component of the Random Forest model is written back into Excel, and a new Pred-Class column is created to save the prediction information. In order to extend the component-encoding information in the IFC file, an ExtensionIfcClass function, which first creates an identifier and then creates a single value for the IFC property associated with it, is defined. The IfcPropertySet property set contains GlobaliD and OwnerHistory. Therefore, the Python's UUID module is introduced to create a global ID. The UUID module can create a universally unique identifier, and then create a PropertySet. Finally, the IfcRelDefinesByProperties association "property" is created in order to associate the PropertySet with the product component passed into the function, completing the IFC extension.

After the definition of the function, each component is iteratively correlated. The Excel table, in which the prediction information has been stored, is first loaded. Each wall instance is then traversed to obtain the ID of the wall. Afterwards, the PredClass information corresponding to the ID in the table is searched. Finally, the ExtensionIfcClass function is executed for correlation.

The iterative code is shown in Algorithm 3:

Algorithm 3 Extension Ifc

```
writer_1=pd.read_excel('C:\\Users\\Administrator\\Desktop\\handson-ml-
master\\wow.xlsx')
products = f.by_type('IfcProduct')
for product in products:
  if(product.is_a()=='IfcWallStandardCase'):
    id = product.id()
    tag=writer_1[(writer_1['id'] == id)]['PredClass'].max()
#Use Max function to convert the returned information into tag
    extensionIfcClass(product,tag)
```

5. Result and Discussion

5.1. Results

5.1.1. Classification Results

Different machine learning algorithms are compared. The best algorithm shows an average classification accuracy of 98.9%. Table 5 shows the results of different machine learning methods. Validated on a test set containing 343 instances, the precision, recall, and F1-scores of the Random Forest model are all equal to 0.99. The metric scores of the different classifiers demonstrate that the Random Forest model has an advantage over SVM and KNN, and it can be used to better classify elements.

Table 5. Classification results of different machine learning algorithms.

Machine Learning	Precision	Recall	F1-Score	Accuracy
RF	0.99	0.99	0.99	98.9%
SVM	0.95	0.96	0.95	95.7%
KNN	0.90	0.90	0.88	89.5%

After model training, the results show that the Random Forest model composed of 69 decision trees with a maximum depth of 16 has the highest precision. Its classification accuracy reaches 98.9%. Table 6 shows the classification results of RF for each component type.

Table 6. RF Classification results.

Code	Precision	Recall	F1-Score	Support
W-00-01-01	0.98	1.00	0.99	41
W-00-01-02	0.98	1.00	0.99	53
W-01-01-02	1.00	1.00	1.00	2
W-10-01-01	1.00	0.99	0.99	76
W-10-01-02	1.00	0.99	0.99	82
W-10-01-04	1.00	1.00	1.00	7
W-10-02-01	1.00	1.00	1.00	55
W-10-03-02	1.00	1.00	1.00	12
W-10-04-01	1.00	1.00	1.00	8
W-10-05-01	1.00	1.00	1.00	2
W-10-05-03	1.00	1.00	1.00	4
W-11-01-02	1.00	1.00	1.00	1

5.1.2. IFC Extension Results

We use BIM vision to verify the classification results of machine learning algorithm. BIM Vision is a free-to-use IFC model viewer. It can view the virtual models coming from Revit systems. The results of the Random Forest output are written into the Classification of forecasts, and the results of the IFC extension are shown in Figure 9.

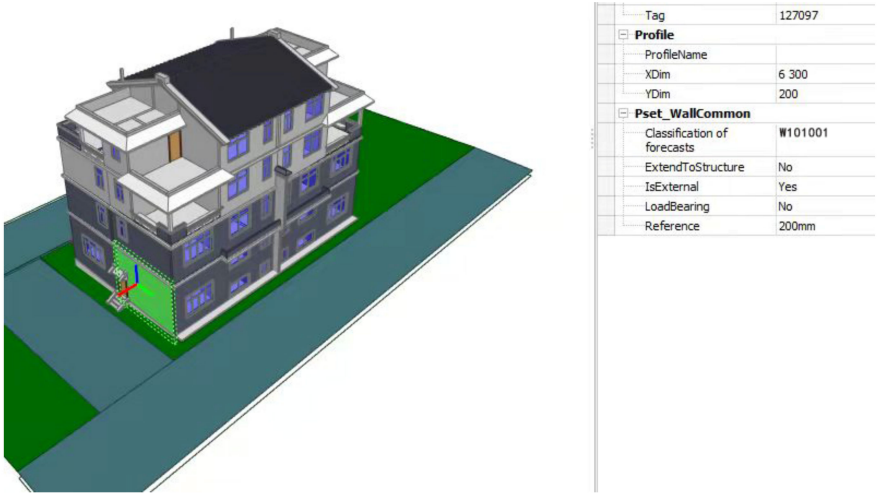

Figure 9. IFC Extension Results.

The line shows that the #289608 correlation property is created, associating #2318 with #289607. The #2318 and #289607 properties are shown in Figure 10, where #289607 represents the property set, #2318 is a wall instance, and the result shows that the extended property set has been associated with the wall instance #2318.

```
: f[289608]
: #289608=IfcRelDefinesByProperties('3HIz4iWbKHwxDWWAM9K5mO',#41,$,$,(#2318),#289607)

f[2318].IsDefinedBy[2]
#289608=IfcRelDefinesByProperties('3HIz4iWbKHwxDWWAM9K5mO',#41,$,$,(#2318),#289607)

f[2318].IsDefinedBy[2][5]
#289607=IfcPropertySet('3HIweTWbKHwwC7WAM9K5mO',#41,'Pset_WallCommon',$,(#289606))

f[2318].IsDefinedBy[2][5][4]
(#289606=IfcPropertySingleValue('Classification of forecasts',$,IfcIdentifier('W100101200'),$),)
```

Figure 10. IFC Extension Verification.

The extended property is then accessed from wall instance #2318 by another method of verification. #289608 is exactly the property just created. Accessing the #289607 attribute of #289608 yields an attribute set described as Pset_WallCommon, while accessing #289606 of #289607 yields an attribute value described as "Classification of forecasts" that contains the classification information W100101200. This shows that the described implementation based on the extension of the attribute set is valid, and the semantics of the resulting IFC physical file are correct.

Finally, one must write it back into IFC, as Algorithm 4 show:

Algorithm 4 Write back IFC file
f.write('./ifc library/Four story double row villa(Modified).ifc')

5.2. Discussion and Limitation

5.2.1. Discussion of Results

Random Forest is an integrated algorithm composed of decision trees. It exhibits a high level of performance in many classification tasks. The Random Forest algorithm introduces randomized samples and randomized features, making it noise-resistant and giving it an

advantage over other algorithms. As Random Forest is a combination of decision trees, it can handle non-linear data. It is also able to handle very high dimensional data and does not need to do feature selection, making it highly adaptable to the dataset. In addition, it can handle both discrete and continuous data, and the dataset requires normalization. Therefore, it performs much better in the experiment. The classical SVM model used in this experiment requires great care in the pre-processing all the data. Through the grid-search method of SVM parameters and kernel function to obtain the best parameters, it is deduced that the performance of SVM classification is close to that of the Random Forest algorithm. SVM shows good potential in this task. The performance of SVM may be further improved using more advanced SVM optimization techniques, which paves the way to further studies, while the KNN algorithm is unable to cope with sample imbalance, leading to the lowest performance in the classification for this task.

Table 7 provides the classification results in the form of a confusion matrix for the Random Forest model.

Table 7. Confusion Matrix.

	W000101	**W000102**	**W010102**	**W100101**	**W100102**	**W100104**	**W100201**	**W100302**	**W100401**	**W100501**	**W100503**	**W110102**
W000101	41	0	0	0	0	0	0	0	0	0	0	0
W000102	0	53	0	0	0	0	0	0	0	0	0	0
W010102	0	0	2	0	0	0	0	0	0	0	0	0
W100101	1	0	0	75	0	0	0	0	0	0	0	0
W100102	0	1	0	0	81	0	0	0	0	0	0	0
W100104	0	0	0	0	0	7	0	0	0	0	0	0
W100201	0	0	0	0	0	0	55	0	0	0	0	0
W100302	0	0	0	0	0	0	0	12	0	0	0	0
W100401	0	0	0	0	0	0	0	0	8	0	0	0
W100501	0	0	0	0	0	0	0	0	0	2	0	0
W100503	0	0	0	0	0	0	0	0	0	0	4	0
W110102	0	0	0	0	0	0	0	0	0	0	0	1

Through a series of indicators and the confusion matrix mentioned above, we found that the Random Forest algorithm has a good level of performance for the component classification task. The misclassification observed through the confusion matrix indicates that additional features may need to be added to distinguish components. For example, there is a w100101 element that is incorrectly classified as w000101. A possible reason for this is that there may be a lack of related load-bearing features in some components. Adding other related features may help to correct such errors. This requires further research to determine which features need to be added to improve the accuracy of prediction.

Second, when applying machine learning methods, we also found that making large datasets is very important. BIM models typically have a disproportionate number of components in various wall components. In this study, the number of some wall components is large. Too few other wall components create an "unbalanced" dataset, which in turn can lead to incorrect classification. Although the RF algorithm has good performance in the task of dealing with some unbalanced datasets, it is necessary to carefully create evenly balanced datasets and use indicators, such as accuracy and recall, ensuring the correct evaluation of performance.

5.2.2. Limitations

In this paper, the construction process information classification and coding system design based on BIM technology and its application, exhibit a high level of performance, using wall components as example. However, some deficiencies and limitations still exist. Although this study classifies the component information of prefabricated concrete buildings and improves the coding system for component types, some missing component types still exist in the coding system. Moreover, this paper uses a machine learning approach to perform the component classification, and there is still room for improvement in the selection of the dataset. The dataset considered in this paper also has some limitations.

In fact, it lacks a large amount of data. This study has verified the feasibility of the classification system, but it needs to establish a larger dataset in engineering practice. In addition, noise exists in some samples. In the manual annotation stage, it was deduced that the properties of some instances of walls are inconsistent with their real types, and that has an impact on the robustness of the model. Although some standards have been issued for BIM technology, the fine management of construction enterprises is far from enough. Therefore, it is necessary to further perfect the unified BIM format and design standards.

5.3. Practical Application

In subsequent research, it can be further combined with Internet-of-Things technology. For example, in the production process of prefabricated components, the classification code is written into the RFID chip to realize the real-time collection of construction process information and achieve the purpose of real-time control of construction process components. Specifically, at the design stage, BIM technology is used to create a BIM information model of the prefabricated components library, and the classification and coding standards of the proposed project are compiled and uploaded to the BIM collaborative management platform for sharing. In the component production stage, RFID chips are embedded, containing the code and basic information of the components to ensure that each component corresponds to a unique identification, ensuring the number of components produced, the transportation order, stacking location, and lifting order. Once an RFID tag is added to a part, and the computer system is notified, the part becomes trackable. During the construction phase of the components, there are many different types and large numbers of components on site, and through BIM and RFID technology, the specific location of the corresponding coded components is determined. RFID-based systems have been used in different applications in construction and maintenance, such as asset tracking and locating, inventory management, equipment monitoring, progress management, facilities management, tool tracking, material management, and quality control.

6. Conclusions

This study explores the use of the Random Forest algorithm to automatically classify and code prefabricated components, with the goal of improving the efficiency of prefabricated component management. The main results are summarized as follows:

(1) An information classification and coding system based on BIM technology is proposed. Based on the OmniClass classification system, this paper proposes an information classification method based on the structure and function of the prefabricated components. In addition, it designs a BIM-based coding system for prefabricated components based on this classification method.

(2) Comparing the performance of the Random Forest, SVM, and KNN algorithms in a prefabricated component classification task, we find that the Random Forest algorithm can better deal with the task of component classification of prefabricated components. The intelligent labelling of building component information based on BIM technology is performed. Based on the above information classification and coding rules, an automatic coding program for prefabricated components is also developed. Through the Python-based building information component property, information extraction and Random Forest model training, the unified coding of prefabricated components is automatically carried out, which performs the unique identification of prefabricated components and reduces the workload of designers.

(3) The extension of IFC-based component coding information is proposed. Based on the IFC4 version, the IFC standard format and entity expression methods are analyzed. Based on the IFC physical file, the process of implementing the extension of component coding information is designed to realize the integrity of the BIM-based prefabricated component information description. This provides a data basis for construction management combined with the Internet of Things. In addition, it is more suitable for

the concepts of standardizing the design of prefabricated buildings and industrializing the production of components in future developments.

Overall, the integration of information on prefabricated components is achieved by proposing a model for the classification and coding of components in IFC files. Future work includes the information tracking of prefabricated components included in the code by combining with the Internet of Things technology. We further propose a BIM-RFID technology application scheme based on the automatic coding of prefabricated components in all phases of assembled buildings, which would have a positive effect on the circulation and sharing of the coded information of prefabricated components. Thus, the proposed approach could save further time and effort by simplifying the process of component information exchange and improving the efficiency of prefabricated component information management.

Author Contributions: Conceptualization, Z.X. (Zhao Xu) and X.W.; methodology, Z.X. (Zhao Xu), M.N., and Z.X. (Zheng Xie); software, Z.X. (Zheng Xie); validation, Z.X. (Zheng Xie) and M.N.; investigation, X.W.; writing—Original draft preparation, Z.X., M.N., and Z.X.; writing—Review and editing, X.W.; visualization, Z.X. (Zheng Xie) and M.N.; funding acquisition, Z.X. (Zhao Xu). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by "National Natural Science Foundation of China" (NSFC-72071043), Natural Science Foundation of Jiangsu Province (BK20201280), MOE (Ministry of Education in China) Project of Humanities and Social Sciences (20YJAZH114).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article. Additional supporting data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Asadi, S.; Roshan, S.; Kattan, M.W. Random Forest Swarm Optimization-Based for Heart Diseases Diagnosis. *J. Biomed. Inform.* **2021**, *115*, 103690. [CrossRef] [PubMed]
2. Badenko, V.; Fedotov, A.; Zotov, D.; Lytkin, S. Scan-to-BIM methodology adapted for different application. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *42*, 1–7. [CrossRef]
3. Bi, W. Research on the Way of Prefabricated Building Information Sharing Based on Computer Software BIM. *J. Phys. Conf. Ser.* **2021**, *1744*, 22078. [CrossRef]
4. Bortolini, R.; Formoso, C.T.; Viana, D.D. Site Logistics Planning and Control for Engineer-To-Order Prefabricated Building Systems Using BIM 4D Modeling. *Autom. Constr.* **2019**, *98*, 248–264. [CrossRef]
5. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
6. Chang, Y.; Li, X.; Masanet, E.; Zhang, L.; Huang, Z.; Ries, R. Unlocking the Green Opportunity for Prefabricated Buildings and Construction in China. *Resour. Conserv. Recycl.* **2018**, *139*, 259–261. [CrossRef]
7. Čuš-Babič, N.; Rebolj, D.; Nekrep-Perc, M.; Podbreznik, P. Supply-Chain Transparency within Industrialized Construction Projects. *Comput. Ind.* **2014**, *65*, 345–353. [CrossRef]
8. De Santana, F.B.; Borges Neto, W.; Poppi, R.J. Random Forest as One-Class Classifier and Infrared Spectroscopy for Food Adulteration Detection. *Food Chem.* **2019**, *293*, 323–332. [CrossRef]
9. Girardet, A.; Boton, C. A Parametric BIM Approach to Foster Bridge Project Design and Analysis. *Autom. Constr.* **2021**, *126*, 103679. [CrossRef]
10. Hao, J.L.; Cheng, B.; Lu, W.; Xu, J.; Wang, J.; Bu, W.; Guo, Z. Carbon Emission Reduction in Prefabrication Construction During Materialization Stage: A BIM-based Life-Cycle Assessment Approach. *Sci. Total Environ.* **2020**, *723*, 137870. [CrossRef]
11. He, R.; Li, M.; Gan, V.J.L.; Ma, J. BIM-enabled Computerized Design and Digital Fabrication of Industrialized Buildings: A Case Study. *J. Clean. Prod.* **2021**, *278*, 123505. [CrossRef]
12. Isaac, S.; Shimanovich, M. Automated Scheduling and Control of Mechanical and Electrical Works with BIM. *Autom. Constr.* **2021**, *124*, 103600. [CrossRef]
13. Jeong, Y.S.; Eastman, C.M.; Sacks, R.; Kaner, I. Benchmark tests for BIM data exchanges of precast concrete. *Autom. Constr.* **2009**, *18*, 469–484. [CrossRef]
14. Jung, N.; Lee, G. Automated Classification of Building Information Modeling (BIM) Case Studies by BIM Use Based on Natural Language Processing (NLP) and Unsupervised Learning. *Adv. Eng. Inform.* **2019**, *41*, 100917. [CrossRef]

15. Koo, B.; La, S.; Cho, N.; Yu, Y. Using Support Vector Machines to Classify Building Elements for Checking the Semantic Integrity of Building Information Models. *Autom. Constr.* **2019**, *98*, 183–194. [CrossRef]
16. Langroodi, A.K.; Vahdatikhaki, F.; Doree, A. Activity Recognition of Construction Equipment Using Fractional Random Forest. *Autom. Constr.* **2021**, *122*, 103465. [CrossRef]
17. Li, C.Z.; Xue, F.; Li, X.; Hong, J.; Shen, G.Q. An Internet of Things-enabled BIM Platform for On-Site Assembly Services in Prefabricated Construction. *Autom. Constr.* **2018**, *89*, 146–161. [CrossRef]
18. Li, X.T.; Zhang, W.H.; Liu, Z.Z.; Qu, W.Y.; Wu, Z.H.; Wang, L. Application of BIM coding and example modelling for typical asphalt pavement diseases based on Omni Class. *J. Phys. Conf. Ser.* **2021**, *2044*, 012153. [CrossRef]
19. Majrouhi Sardroud, J. Influence of RFID Technology on Automated Management of Construction Materials and Components. *Sci. Iran.* **2012**, *19*, 381–392. [CrossRef]
20. Maritz, T.; Klopper, C.; Siglé, T. Developing a National Standard/Code of Practice for the Classification of Construction Information in South Africa. *Build. Environ.* **2005**, *40*, 1003–1009. [CrossRef]
21. Marmo, R.; Polverino, F.; Nicolella, M.; Tibaut, A. Building Performance and Maintenance Information Model Based on IFC Schema. *Autom. Constr.* **2020**, *118*, 103275. [CrossRef]
22. Mohana, R.M.; Reddy, C.K.K.; Anisha, P.R.; Murthy, B.V.R. Random Forest Algorithms for the Classification of Tree-Based Ensemble. *Mater. Today Proc.* **2021**. [CrossRef]
23. Montes, C.; Kapelan, Z.; Saldarriaga, J. Predicting Non-Deposition Sediment Transport in Sewer Pipes Using Random Forest. *Water Res.* **2021**, *189*, 116639. [CrossRef] [PubMed]
24. Naranje, V.; Swarnalatha, R. Design of Tracking System for Prefabricated Building Components using RFID Technology and CAD Model. *Procedia Manuf.* **2019**, *32*, 928–935. [CrossRef]
25. Santos, R.; Costa, A.A.; Silvestre, J.D.; Vandenbergh, T.; Pyl, L. BIM-based Life Cycle Assessment and Life Cycle Costing of an Office Building in Western Europe. *Build. Environ.* **2020**, *169*, 106568. [CrossRef]
26. Solihin, W.; Eastman, C.; Lee, Y.; Yang, D. A Simplified Relational Database Schema for Transformation of BIM Data Into a query-efficient and Spatially Enabled Database. *Autom. Constr.* **2017**, *84*, 367–383. [CrossRef]
27. Tan, T.; Chen, K.; Xue, F.; Lu, W. Barriers to Building Information Modeling (BIM) Implementation in China's Prefabricated Construction: An Interpretive Structural Modeling (ISM) Approach. *J. Clean. Prod.* **2019**, *219*, 949–959. [CrossRef]
28. Yao, F.; Ji, Y.; Tong, W.; Li, H.; Liu, G. Sensing Technology Based Quality Control and Warning Systems for Sleeve Grouting of Prefabricated Buildings. *Autom. Constr.* **2021**, *123*, 103537. [CrossRef]
29. Yuan, L.; Guo, J.J.; Wang, Q. Automatic Classification of Common Building Materials From 3D Terrestrial Laser Scan Data. *Autom. Constr.* **2020**, *110*, 103017. [CrossRef]
30. Zhang, D.; Lou, S. The Application Research of Neural Network and BP Algorithm in Stock Price Pattern Classification and Prediction. *Future Gener. Comput. Syst.* **2021**, *115*, 872–879. [CrossRef]
31. Zhang, H.; Yu, L. Site Layout Planning for Prefabricated Components Subject to Dynamic and Interactive Constraints. *Autom. Constr.* **2021**, *126*, 103693. [CrossRef]