IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 19 May 2024, accepted 28 June 2024, date of publication 4 July 2024, date of current version 17 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3423425

RESEARCH ARTICLE

Extracting Structural Elements From 3D Point Clouds in Indoor Environments via Machine Learning Techniques

KORAY AKSU, (Graduate Student Member, IEEE), AND HANDE DEMIREL, (Member, IEEE)
Department of Geomatics Engineering, Faculty of Civil Engineering, Istanbul Technical University, 34469 Istanbul, Türkiye

Corresponding author: Koray Aksu (koray.aksu@itu.edu.tr)

This work was supported by the Scientific and Technological Research Council of Türkiye (Building Information Modeling (BIM) Based Fire Evacuation Simulation) under Project 121Y099. The thesis of Koray Aksu was supported under Project 121Y099.

ABSTRACT The utilization of three-dimensional point clouds is an advanced approach for detecting the geometry of objects within a building environment. Nonetheless, a vast amount of data still needs to be manually processed. Intelligent automation frameworks could be deployed to overcome such issues. Hence, this study proposes a machine learning-based framework for successfully classifying structural components in indoor environments. The proposed framework consists of four stages: pre-processing, feature extraction, feature selection, and interpretability of classification results using an explainable machine learning method. According to the proposed framework, the chi-squared test stands out for optimum local neighborhood radius determination and feature selection. The CatBoost model has the highest accuracy of 82.96%, whereas the Random Forest model's accuracy is 82.09%. However, the training time for the Random Forest is 27 times shorter than the CatBoost. Hence, both models could be preferred to other machine learning models for practical applications due to the good balance between accuracy and calculation efficiency. Additionally, the model with the highest accuracy, CatBoost, is evaluated using the Shapley Additive exPlanations to understand the impacts of features on predictions, and according to the results, Z coordinate and verticality had a relatively high impact on the model, while others had low impacts. The proposed framework uses machine learning to classify indoor point clouds, balancing processing time and accuracy for computational efficiency in practical applications. Hence, the framework could be utilized to automate the digitalization efforts of indoor environments effectively.

INDEX TERMS 3D point cloud, classification, explainable machine learning, indoor environment, local neighborhood, machine learning, structural element, terrestrial laser scanning.

I. INTRODUCTION

Three-dimensional (3D) point clouds emerge as data structures that can capture objects' geometric properties and precise positions with Industry 4.0 and the digital transformation process, where point clouds play an important role in digital twins. These high-accuracy data types can be obtained using terrestrial laser scanning and photogrammetric techniques [1] and have various uses in different fields [2], [3], [4], [5]. Several application domains exist, such as preserving cultural heritage, the robotics industry, digital twins, autonomous vehicles, indoor navigation, and others. For example, cultural heritage experts and researchers use point cloud data to conduct detailed analyses, allowing for a deeper understanding of historical sites without causing physical damage. In addition, digital replicas enable virtual exploration and educational experiences, making cultural heritage accessible to people worldwide [3], [6], [7]. Another essential use of the point cloud is for medical applications. Point clouds in medical imaging are pivotal in creating detailed, patient-specific 3D models. These models are essential for planning complex surgical procedures and

The associate editor coordinating the review of this manuscript and approving it for publication was Gerardo Di Martino.

designing custom prosthetics, leading to enhanced surgical precision and improved patient outcomes [8], [9]. With the development of the robotics industry in recent years, point clouds have become widespread in this field. Integrating 3D point clouds has significantly enhanced the capabilities and efficiency of robotic systems in the industry. Robots equipped with sensors that generate point clouds can accurately perceive their environment. This is crucial for object recognition, spatial mapping, and indoor navigation [10], [11]. Point cloud allows robots to interact with their surroundings more safely and efficiently [12]. It is applicable in various settings, such as manufacturing, warehouse logistics, and autonomous vehicles. Due to its high-quality realistic visualization and real-world 3D modeling, point cloud has become widely used in the gaming industry [13], [14]. Point clouds are significant for developing advanced Virtual Reality (VR) and Augmented Reality (AR) games, where the precision and depth of the environments are critical factors in creating an immersive experience [15], [16]. The 3D point cloud significantly advances the creation of digital twins, virtual replicas of physical environments [17]. Point clouds generate high-quality 3D models of buildings or even entire cities in this context, creating precise digital counterparts [18], [19]. These digital twins are used for various purposes, such as performance monitoring, predictive maintenance, and simulation of different scenarios, which allows for informed decision-making and operations optimization. The utilization of 3D point clouds in creating indoor information is another crucial application area. Point clouds enable highly accurate construction of all geometric aspects of a 3D indoor space [20], [21], [22]. This is particularly important for interior design and indoor navigation systems, especially in complex environments like shopping malls, public buildings, hospitals, and airports [23]. Hence, point cloud is widely used in various fields due to its valuable information and its high accuracy in geometrical representation. However, due to its complex data structure and large data volume, extracting information from the point cloud requires extensive effort, experience, and hardware infrastructure. Furthermore, most object classification efforts are conducted semi-automatically, where automatization efforts are still in progress. Moreover, managing such data can be challenging since a vast amount of complex 3D spatial data is produced.

In order to overcome such challenges, machine learning and deep learning approaches are highly preferred to automatize the digitalization effort of 3D point cloud classification. Several researchers have suggested 3D point classification using machine learning techniques such as Random Forest, XGBoost, LightGBM, CatBoost, Support Vector Machine, and Naïve Bayesian algorithm. The Random Forest algorithm is known for its fast processing, high accuracy, and efficiency, making it a reliable model for 3D point cloud classification [24], [25], [26], [27]. XGBoost, a gradient-boosting algorithm, has shown outstanding performance in classification of 3D point cloud of trees [25]. LightGBM, another gradient-boosting-based algorithm, has been effectively used for leaf and wood classification from LiDAR point cloud [28]. CatBoost is another machine learning model based on gradient boosting that can classify 3D point clouds with high precision [29]. The Support Vector Machine algorithm is also frequently employed for point cloud classification [27], [30], [31], [32]. The Naïve Bayesian algorithm is preferred for 3D point classification due to its fast-processing time [27]. Such studies are performed in an out environment, whereas indoor environments utilize deep learning algorithms. The PointNet combines local features to extract global features, making it suitable for generating final predictions [33]. PointNet++ is another deep learning model that uses PointNet as a base and is used for similar purposes. It is known for its ability to perform more effective feature extraction than PointNet [34]. PointSIFT is a model developed for object extraction in indoor and 3D models. This model captures information of different orientations and enables the processing of objects with various scales [35]. The VoteNet is a deep learning model that allows models to vote to object centroids directly from point clouds and learn to aggregate votes through their features and local geometry for indoor object detection [36]. PointGroup is another model used only indoors and developed for instance segmentation. This model assigns labels and offset vectors to points, clusters them based on original and offset-shifted coordinates, and optimizes precision by combining the two coordinate sets [37]. RandLA-Net was developed for information extraction from the point cloud. It stands out from other models due to its high training speed [38]. ResPointNet++ is a deep learning model designed to process industrial point clouds. It extracts significant features from 3D point clouds using residual connections and hierarchical feature learning [39]. KPConv is a deep learning model that processes 3D point clouds. It introduces kernel point convolutions that allow adaptive convolution operations on the point cloud, thus enabling effective feature learning [40]. According to the literature, deep learning models are mainly preferred when large datasets and structures are more complex and require high hardware capacity [41], [42]. This study will focus on machine learning models since the structural component is simple, machine learning models are more flexible, and studies deploying machine learning algorithms are rare. Hence, six machine learning methods will be deployed and evaluated in indoor environments.

In order to classify 3D point clouds with machine learning models, geometric and non-geometric features of objects are utilized. These features are employed as independent variables to provide input to machine learning methods. Geometric features include the basic geometric properties of individual points and their relationships within the point cloud [43], [44]. Furthermore, 3D point coordinates, surface normal and eigenvalue-based features are geometric features that provide valuable geometric information about the structures represented by the points [45], [46], [47],

[48], [49]. Non-geometric features such as colors and intensity values could also be used [43]. For geometric features, using eigenvalue-based geometric features such as anisotropy, eigenentropy, linearity, omnivariance, etc., is a well-established and highly applied approach for extracting significant spatial information from 3D point clouds [50], [51]. These techniques rely on the covariance matrix's eigenvalues derived from the data points' spatial coordinates. The data is transformed into a new space using the eigenvectors, enabling the eigenvalue-based geometric features to capture the dataset's inherent geometric structures and variances. Choosing appropriate parameters when extracting eigenvalue-based features from point clouds is crucial to achieving high classification performance. For this purpose, it is essential to determine the appropriate local neighborhood radius for each eigenvalue-based feature since each feature represents a different object that determines the classification performance. Different approaches can define neighborhood relations, such as k-nearest-neighbor search, fixed radius (local neighborhood radius), and histogram interpretation. Within the literature, several studies have determined the optimum radius for producing feature sets. Reference [25] extract feature sets with different radius values for each feature. They examined the feature value-radius histogram for each class to determine the optimum radius. Reference [52] used different search radiuses to produce features. Reference [53] used a fixed radius, a fixed number of points, and eigenentropy-based scale selection to extract features. Reference [45] extracts the eigenvalue-based features using a fixed radius to classify outdoor point clouds. Within this study, the determination of local neighborhood radius via statistical methods, which is widely preferred in feature selection in machine learning due to processing time and performance, is performed.

In order to classify the objects via machine learning methods, a feature selection step is required to reduce the data volume and improve the models' accuracy [54], [55]. Various techniques are used to select the most relevant features, such as filter-based, wrapper, and embedded [56]. When dealing with big datasets, selecting the most efficient processing technique and prioritizing processing time is essential. Filter-based feature selection approach evaluates the significance of each feature in the dataset using statistical tests and information-theoretic measures. The features with the highest scores on these metrics are then selected and included in the modeling process. This method is fast, scalable, and improves generalizability, making it an ideal choice for practitioners looking for a reliable and efficient feature selection approach [57], [58]. Wrapper-based feature selection integrates feature selection with the chosen machine learning algorithm to evaluate the impact of the feature set on performance metrics like accuracy [56], [59]. Embedded feature selection differs from filter-based or wrapping methods by integrating feature selection within the model training process [60], [61]. The field of point cloud classification has some significant research gaps that need to be addressed: a) Most of the studies in the literature use CNN-based deep learning algorithms to classify indoor 3D point clouds. However, machine learning algorithms, frequently preferred in different application domains, are rarely used in indoor point cloud classification, b) Different methods are used to determine local neighborhood radius while extracting eigenvalue-based features. However, it is uncertain whether the most statistically significant radius was utilized, c) It is necessary to incorporate objects' geometric and non-geometric features to facilitate the classification of 3D point clouds through machine learning models. However, the impacts of features on model predictions are rarely assessed.

Hence, the study's novelty is evaluating machine learning-based methods for 3D point cloud classification, deploying statistical methods for determining the optimum local neighborhood radius, and interpreting the machine learning model via an explainable machine learning within an indoor environment. Hence, via deploying this framework, the expected results are: a) to successfully implement machine learning frameworks in an indoor environment, b) to increase the classification performance of available machine learning models, and c) to explain the impacts of features on model predictions.

This research aims to propose a machine learning-based framework and interpret model results via an explainable machine learning within an indoor environment. Furthermore, an optimum local neighborhood radius for eigenvalue-based features is proposed via deploying statistical methods for available machine learning models to increase classification accuracy. The system architecture of the study includes four stages: pre-processing, feature extraction, feature selection, and evaluation of the model. The first stage covers creating the dataset and preparing it for training. In this context, splitting for the train/test dataset, sampling, and normalization were applied. The second stage includes feature extraction. The extracted features represent the geometric structure of the objects. In this stage, statistical methods were used to increase the representation of the geometric structures of the objects on the features. In the third stage, machine learning models are chosen using feature combinations and statistical feature selection techniques. The fourth stage includes interpreting the model using the SHAP method, an explainable machine-learning approach.

II. DATA AND METHODOLOGY

This study consists of four steps: pre-processing, feature extraction, feature selection, and explaining the trained model via an explainable machine learning technique. The study area, 3D point cloud, and system architecture are described in the following sections.

A. DATA AND STUDY AREA

The data acquisition and data analyses took place in a public building, Istanbul Technical University, Faculty of Civil Engineering, Türkiye. The 3D point cloud data was

obtained with a Leica C10 terrestrial laser scanner 4.5 mm scan resolution from 0-50 m, where only the 3rd floor and classrooms were scanned. The total area is 350 m^2. This dataset contains colors, surface normal, and intensity values. It includes ten classes of objects, such as structural elements, lighting, doors, and windows. The 3D point cloud data of the study area is illustrated in Fig. 1.

B. SYSTEM ARCHITECTURE

The system architecture comprises four essential steps: pre-processing, feature extraction, feature selection, and machine learning model explanation. Firstly, pre-processing steps were applied, including splitting the dataset, balancing the training dataset, and normalizing. Then, the feature extraction step was applied. This stage includes extracting geometric and non-geometric features and selecting the optimum local neighborhood radius through statistical methods for eigenvalue-based geometric features. Then, different feature combinations and feature selection methods were applied to evaluate model performance by F1 score and processing time. Then, the impacts of the features on the trained model and the relationships between the feature and the class were evaluated with an explainable machine learning model. The proposed framework is presented in Fig. 2.

C. PRE-PROCESSING

Completing a series of critical pre-processing steps is essential to effectively train the point cloud classification models. These steps include splitting datasets into the training and test datasets, resolving imbalances in the training data, and normalizing the data to enhance the model's accuracy. The first step involves dividing the dataset into training and test datasets. This division helps the model learn from diverse data, acquiring robust and generalized learning capabilities that can perform reliably on unseen samples. The second step is to address potential imbalances in the training dataset. Imbalanced datasets can cause biased model performance, leading to skewed results that favor majority classes and underrepresent minority classes [62]. Imbalanced datasets occur when the dataset's classes have significantly different frequencies, leading to biased model predictions. Machine learning commonly uses undersampling and oversampling to address this issue [63]. Undersampling aims to balance the dataset by selecting a subset of the majority class samples, thereby reducing the frequency of the majority class. This method maintains the classes' overall distribution while reducing the majority class's size. On the other hand, oversampling aims to balance the dataset by increasing the frequency of the minority class. This can be achieved by replicating or synthesizing new instances of the minority class. Both undersampling and oversampling techniques can be used to create a balanced dataset and improve the predictive accuracy of a model [64], [65]. In our study, we utilized random undersampling and random oversampling techniques. The last step in the pre-processing pipeline is normalization. It scales data to a consistent range, preventing specific attributes from dominating the training process solely based on their magnitude [66]. This creates a fair and stable learning environment, ultimately improving the accuracy and dependability of the model during training.

TABLE 1. Eigenvalue-based geometric features and equations.

Feature	Equation
Anisotropy	$(\lambda_1 - \lambda_3)/\lambda_1$
Eigenentropy	$-(\lambda_1 \ln \lambda_1 + \lambda_2 \ln \lambda_2 + \lambda_3 \ln \lambda_3)$
Linearity	$(\lambda_1 - \lambda_2)/\lambda_1$
Omnivariance	$\sqrt[3]{\lambda_1 \lambda_2 \lambda_3}$
Planarity	$(\lambda_2 - \lambda_3)/\lambda_1$
Sphericity	λ_3/λ_1
Sum of eigenvalues	$\lambda_1 + \lambda_2 + \lambda_3$
Surface variation	$\lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$
Verticality	$1 - \lvert <[0,0,1], e_3> \rvert$

D. FEATURE EXTRACTION

Machine learning techniques use various features of point cloud data for efficient classification. Some features are generated during the data collection step, while others are produced later using different feature extraction techniques. A total of 19 geometric and non-geometric features were used in the study. The geometric features consist of eigenvalue-based features (9), the 3D coordinates of the point (3), and surface normals along three axes (3). The non-geometric features consist of four features: RGB color information (3) and the intensity value of the point (1). The numbers in parentheses indicate the number of relevant features. Eigenvalue-based features that offer valuable insights into the location and structure of objects. These features are computed from the eigenvalues of the covariance matrix, which is derived from the 3D coordinates of the points. The equations of eigenvalue-based features are illustrated in Table 1.

CloudCompare environment was utilized to create features based on eigenvalues. Batch processing was applied through the software "command line mode" using Python programming to generate features with different radius for each eigenvalue-based feature [67]. Then, statistical tests were applied for each eigenvalue-based feature using the scikit-learn machine learning library, and the optimum local neighborhood radius was determined for each eigenvalue-based feature [68].

Eigenvalues are calculated using the covariance matrix equation in (1). Here, C denotes the covariance matrix, N value, number of neighboring points, p_i the vector with the coordinates of the point, and $\bar{p}$ the vector with the mean coordinates of the neighboring points. Equation (2) is used to calculate eigenvalues using covariance matrices. C is the covariance matrix, I is the unit matrix, and λ is the eigenvalue. Eigenvectors are calculated using (3) where e_i refers to the

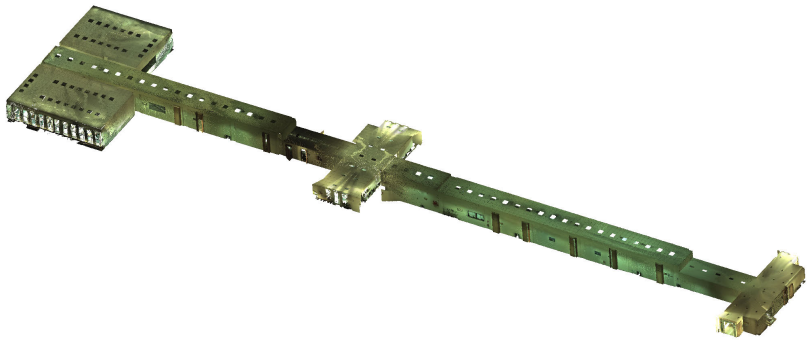

FIGURE 1. 3D point cloud of the study area.

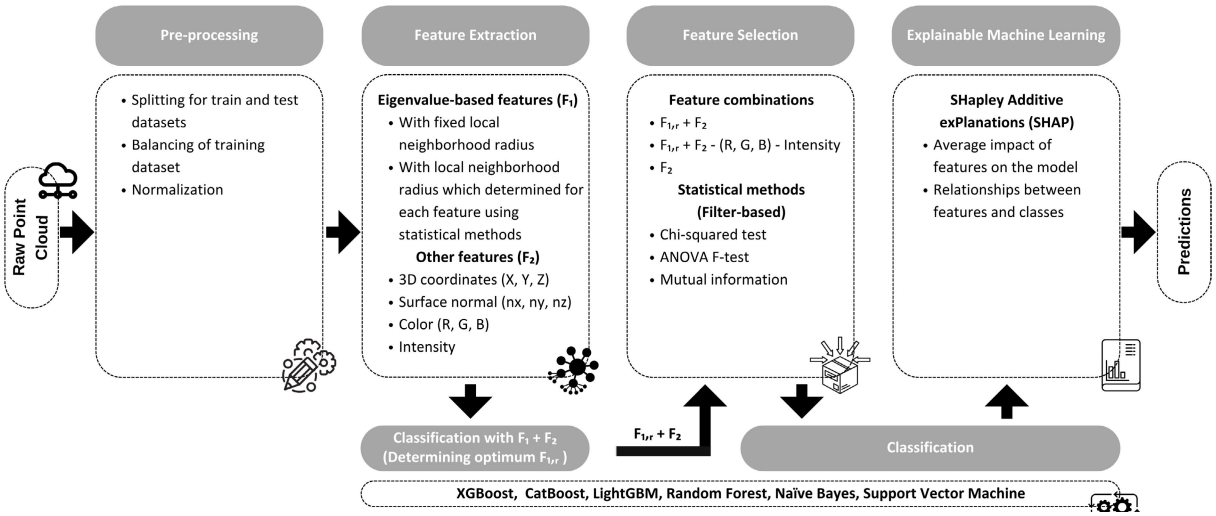

FIGURE 2. The proposed framework.

eigenvector.

$$C = \frac{1}{N}\sum_{i=1}^{N}(p_i - \bar{p})(p_i - \bar{p})^T \tag{1}$$

$$\det(C - \lambda I) = 0 \quad \rightarrow \quad \lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0 \tag{2}$$

$$(C - \lambda_i I)\, e_i = 0 \tag{3}$$

The λ_1, λ_2, and λ_3 values in the equations below represent the eigenvalues produced from the covariance matrix. Each eigenvalue-based geometric feature embodies objects characterized by diverse geometric dimensions and shapes. Therefore, selecting each feature's optimum local neighborhood radius is essential when extracting the geometric features. As the number of local neighborhood radius increases, the processing time required for analysis also increases. Conducting repeated model training to determine the most suitable radius is not feasible due to the processing time involved. Therefore, selecting the radius based on statistical methods can make the process more manageable. This way, the impact of the features determined for each radius value on the model's accuracy can be evaluated objectively. When considering the applicability of statistical methods to large datasets and processing time, the Chi-squared test, ANOVA F-test, and Mutual Information statistical methods are commonly used [69]. Machine learning often applies these methods to reduce data size for model training and feature importance calculation. These methods are called filter-based feature selection in machine learning. Relevant statistical methods have been examined in detail in the following sections.

E. FEATURE SELECTION

Feature selection is a crucial step in machine learning to identify the most relevant features of the data. The study employed filter-based feature selection methods: the chi-squared test, ANOVA F-test, and Mutual Information. The following headings explain these methods in detail.

1) CHI-SQUARED TEST

The chi-squared test (χ^2) is often used to determine whether two categorical variables in a dataset are independent or associated [70]. The difference between the observed and expected frequencies of each category for the variables is calculated by this test, and the significance of this

difference is quantified. The features likely to contribute significantly to the model's performance can be determined by analyzing the association between each feature and the target. The chi-squared value is computed using (4), where O_i represents the observed value, and E_i shows the expected value. The chi-squared value can be improved by refining feature selection techniques and optimizing E_i. These help to uncover and understand the underlying relationship between variables that leads to an improved machine learning model performance.

$$\chi^2 = \frac{\sum_i (O_i - E_i)^2}{E_i} \tag{4}$$

Suppose the target variable and feature value in the dataset are independent. In that case, the relevant feature does not contribute to the model training, and the chi-squared value of these features is small. While selecting the feature, the relationship of all the features in the dataset with the target variable is examined, and feature selection is performed according to the decreasing chi-squared value.

2) ANOVA F-TEST

ANOVA F-test was used to identify significant differences between the groups. This approach evaluates the relevance of different features for distinguishing between different classes within a dataset [61]. The F-statistic is calculated by comparing the between group and within group variance. The mean and variance of each feature are calculated for each group, and the overall mean and variance of each feature across all groups are computed. These values calculate the variances of between group and within group for each feature. The features are ranked based on their F-statistic values. When testing hypotheses and analyzing data, the Sum of Squares Between Groups (SSB) and the Sum of Squares Within Groups (SSW) are necessary to evaluate the variance and dispersion of data points across different groups or classes in a dataset. The sum of squared differences between the groups is demonstrated by (5), which indicates the variation among the means of individual classes. The sum of squared differences between each data point is demonstrated by (6). The level of dispersion present among the data points within each group is indicated by this calculation. The k parameter represents the number of classes, while the n parameter represents the total number of samples in the dataset. The relevant F-score is calculated with (7), which enables the determination of significant differences between the means of various groups.

$$SSB = \sum_{j=1}^{k} \left(\bar{X}_j - \bar{X}\right)^2 \tag{5}$$

$$SSW = \sum_{j=1}^{k} \sum_{j=1}^{l} \left(X - \bar{X}_j\right)^2 \tag{6}$$

$$F - score = \frac{SSB/(k-1)}{SSW/(n-k)} \tag{7}$$

3) MUTUAL INFORMATION

The Mutual Information (MI) technique helps to identify the most informative features in a dataset that can distinguish between different classes [71]. It also measures the statistical dependence between each feature and the target class variable. The MI value is calculated with (8). $P(x, y)$ represents the joint probability mass of x and y together, while $P(x)$ and $P(y)$ represent their individual probability mass. A higher MI score indicates more informative and relevant features and stronger dependency. Conversely, lower values suggest weaker or no dependence between the two variables.

$$MI = \sum_{x,\, y} P(x, y)\, log \frac{P(x, y)}{P(x)\, P(y)} \tag{8}$$

F. CLASSIFICATION

This research employed six distinct models: XGBoost, LightGBM, CatBoost, Random Forest, Gaussian Naïve Bayes, and Support Vector Machine. Their accuracy and computational efficiency were evaluated. XGBoost is a popular gradient-boosting framework used for supervised machine-learning tasks due to its high effectiveness and efficiency [72]. It works by building an ensemble of weak learners sequentially, with each new tree added aiming to correct the errors made by the previous ones. This produces a powerful ensemble of decision trees contributing to the final prediction. XGBoost uses a depth-first approach to construct decision trees, optimizing the objective function at each step. It also employs approximate learning to efficiently identify the best-split points for each feature, granting XGBoost a significant speed advantage over traditional gradient boosting methods. To avoid overfitting, XGBoost incorporates L1 and L2 regularization terms in its objective function, penalizing large weights assigned to features in decision trees and maintaining a balance between model complexity and performance. LightGBM is another robust gradient-boosting framework developed by Microsoft that focuses on efficiency, scalability, and the ability to handle large datasets [73]. One of its notable features is the Gradient-based One-Side Sampling technique, which reduces the number of data instances used during training by selecting and keeping the data instances with large gradients contributing more to the model's updates. This approach significantly reduces training time and memory usage without affecting the model's performance. Also, this model employs Histogram-based Gradient Boosting, which allows for faster computations of split points during tree-building, contributing to the algorithm's speed and scalability. CatBoost is an advanced algorithm developed by Yandex for gradient boosting that handles both categorical and numerical features effectively [74]. This algorithm uses a variant of ordered boosting that incorporates statistical methods. During training, the algorithm treats missing values as a separate category, simplifying data preparation and preventing the exclusion of valuable information due to missing values. Furthermore, CatBoost automatically tunes the learning rate during training to ensure a balanced

trade-off between fast convergence and accurate predictions. CatBoost uses optimized algorithms for calculating feature statistics and gradient updates to speed up computation and enhance training efficiency, resulting in faster model training and better scalability for large datasets. Random Forest is an ensemble learning technique widely preferred for classification [75]. When building a model, avoiding overfitting by introducing randomness through two methods is essential. Firstly, each decision tree is trained on a random subset of the data. Secondly, only a random subset of features is considered at each split in the tree. By doing this, the trees can capture different data patterns, enhancing the model's generalization ability. The Gaussian Naive Bayes algorithm assumes features that follow a Gaussian distribution and applies Bayes' theorem to make predictions [76]. By estimating the likelihood and prior probability, it calculates the posterior probability and selects the predicted class with the highest probability based on a given feature vector. Support Vector Machine is a machine learning algorithm that locates the optimal hyperplane in a high-dimensional feature space that can effectively separate data points from different classes [77]. The support vectors, the data points closest to the hyperplane, are crucial in defining the hyperplane and influencing its position. The margin around the hyperplane is vital for the algorithm's ability to generalize well to unseen data.

1) PERFORMANCE EVALUATION METRICS

Accuracy, Precision, Recall, and F1 score are frequently used for classification performance evaluation. These metrics are calculated with true positive (TP), false positive (FP), true negative (TN), and false negative (FN) values from the confusion matrix. The accuracy value is calculated using (9). This metric provides a general assessment of a model's performance. The precision value calculated with (10) expresses the ratio of positively predicted samples to true positives. High precision gives information about the reliability of optimistic predictions. The recall value assesses how accurately a classification model identifies positive instances among the true positive instances, expressed by (11). The F1 score, the harmonic means of precision and recall values, is utilized through (12). This score is used where there is an imbalance in the classes within the dataset.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{12}$$

G. EXPLAINABLE MACHINE LEARNING

The results from the models trained with different machine-learning methods must be interpretable. Here, by determining the relationships between input variables and outputs, the results obtained are improved, and the decision-making ability of the model is strengthened. By using Explainable Machine Learning methods, the trained models are made interpretable, and performance improvements can be made by determining the impacts and contributions of the model variables to the model [78], [79]. Game theory and perturbation-based approaches have been developed to explain the models. The Shapley value is used in the game theory-based approach, which expresses the contribution of each input used in model training to the result [80], [81]. In calculating the Shapley value, the interaction of each player's earnings with other players is considered. Using the Shapley value in machine learning, the importance and impact rates of the features in the prediction results are calculated [82]. Shapley Additive exPlanations is a frequently preferred library where Shapley value and game theory-based approaches and machine learning models are interpreted [83]. To calculate the Shapley value, the first step is to define a set of all features. Then, subsets are created, and the impacts of each subset's features are calculated. This process is applied to all subsets to determine the Shapley value of each feature. Shapley value is calculated using (13). Within this context, $\emptyset_i$ denotes the Shapley value for feature i. F represents the set of all features, while S refers to any subset of players that excludes feature i. $|F|$ and $|S|$ signify the size of the set of all features and subsets, respectively. $f_S\ (x_S)$ represents the value calculated for the subset S. $f_{S\cup\{i\}}\ \left(x_{S\cup\{i\}}\right)$ indicates the new value obtained by adding the feature i.

$$\emptyset_i = \sum_{S \subseteq Fi} \frac{|S|!\ (|F| - |S| - 1)!}{|F|!} \left[f_{S\cup\{i\}}\left(x_{S\cup\{i\}}\right) - f_S\ (x_S)\right] \tag{13}$$

This study utilized SHAP to interpret the model and explain the effect of features on the model and the relationships between features and classes.

III. RESULT AND DISCUSSION

In the study area, a comparison was made between six machine-learning models to assess their training time, testing time, accuracy metrics, and explanation of the trained model. The first step was pre-processing the dataset before training and evaluating the models. Statistical feature selection methods were used to determine the optimum local neighborhood radius for eigenvalue-based geometric features and comparison with a fixed radius. The best models were then examined in detail regarding their training time and accuracy. Six different feature combinations were created and trained to evaluate the effect of features on classification accuracy, training, and testing time. Finally, the best model regarding accuracy was selected, and evaluation metrics were examined. Explainable machine learning was used to observe the impact of features on class predictions and their average impacts on the model. All models are trained on the grid computing system, with detailed information in Table 2.

TABLE 2. Hardware specifications of a grid computing system.

Processor	Dell R640, Intel Xeon Scalable Gold 6148
Cores/process	12
Nodes/process	1
defMemPerCore/process (MB)	8000

TABLE 3. Number of points for each class of the training and testing dataset.

Class	Train dataset	Test dataset
Beam	300,000	303,705
Ceiling	300,000	1,580,565
Column	300,000	200,133
Desk	300,000	241,909
Door	300,000	289,803
Floor	300,000	1,132,767
Lighting	300,000	152,541
Other	300,000	183,539
Wall	300,000	1,399,022
Window	300,000	56,611

A. PRE-PROCESSING

The initial step of the pre-processing varies for the selected six distinct models. The dataset is divided into a 70% training area and a 30% testing area to ensure the accuracy and reliability of the machine learning model. Within the training area, 80% is used for training, whereas 20% is used for validation for XGBoost, LightGBM, and CatBoost. An out-of-back (OOB) technique is applied for the Random Forest model, where there is no need for a validation dataset. Furthermore, Gaussian Naïve Bayes and Support Vector Machine do not support direct validation datasets. It supports only calculation expensive methods that use k-fold-cross validation. The partitioning of the dataset is presented in Fig. 3, with blue and orange regions representing the training and test datasets, respectively.

The number of points for each class in the training dataset is unbalanced. This could lead to an imbalanced model training. To address this issue, random undersampling and random oversampling techniques were applied while considering the dataset size and hardware capacity. During the sampling process, the class frequencies were thoroughly examined, and the average frequency of classes was established at 300,000. Subsequently, the dataset was balanced by applying sampling techniques to the relevant classes, and the number of points for all classes was resampled to 300,000 for each class, ensuring equitable and unbiased model training. Once the dataset was balanced, the features were normalized to a range of 0-1 to reduce data size and standardize inputs. The number of points for each class of the training and testing dataset are listed in Table 3.

B. FEATURE EXTRACTION

The classification features are categorized into eigenvalue-based geometric features ($F_{1,r}$) and other features (F_2).

TABLE 4. Eigenvalue-based geometric features and other features.

Eigenvalue-based features ($F_{1,r}$)	Other features (F_2)
Anisotropy	3D coordinates (X, Y, Z)
Eigenentropy	Colors (R, G, B)
Linearity	Surface normal (N_x, N_y, N_z)
Omnivariance	Intensity
Planarity	
Sphericity	
Sum of eigenvalues	
Surface variation	
Verticality	

TABLE 5. Determination of the local neighborhood radius (cm) with fixed values and statistically.

Eigenvalue-based features ($F_{1,r}$)	Fixed radius r_n	Statistical methods		
		r_{χ^2}	$r_{F-value}$	r_{MI}
Anisotropy	20, 40, 60, 80, 100	6	8	82
Eigenentropy		100	100	4
Linearity		22	22	4
Omnivariance		94	92	86
Planarity		22	24	4
Sphericity		84	8	82
Sum of eigenvalues		4	30	4
Surface variation		82	8	82
Verticality		20	72	36

There are 19 features, with nine being eigenvalue-based and 10 being other features are given in Table 4.

$F_{1,r}$ feature set were extracted using a fixed local neighborhood radius of 20 cm, 40 cm, 60 cm, 80 cm, and 100 cm, resulting in five feature sets ($F_{1,r_{20}}$, $F_{1,r_{40}}$, $F_{1,r_{60}}$, $F_{1,r_{80}}$, $F_{1,r_{100}}$), each set contains nine distinct features. After that, the relevant radius for each feature was determined by applying statistical methods, namely the chi-squared test, ANOVA F-test, and mutual information. To perform this, features were also extracted using 50 different local neighborhood radiuses, ranging from 2 cm to 100 cm with 2 cm intervals, and the relevant local neighborhood radius was determined for each feature. For example, the chi-squared values of verticality, linearity, and sphericity features are shown in Fig. 4, where the vertical axis represents the chi-squared value. The horizontal axis is the local neighborhood radius, and the red-colored columns imply the highest local neighborhood radius, chi-squared value, or significance. This analysis applied to all eigenvalue-based features.

The local neighborhood radius of the eigenvalue-based features with fixed values and determined based on the statistics, is illustrated in Table 5.

Performance evaluations were completed using six machine-learning models: XGBoost, CatBoost, LightGBM, Random Forest, Gaussian Naïve Bayes, and Support Vector

FIGURE 3. Train and test area.

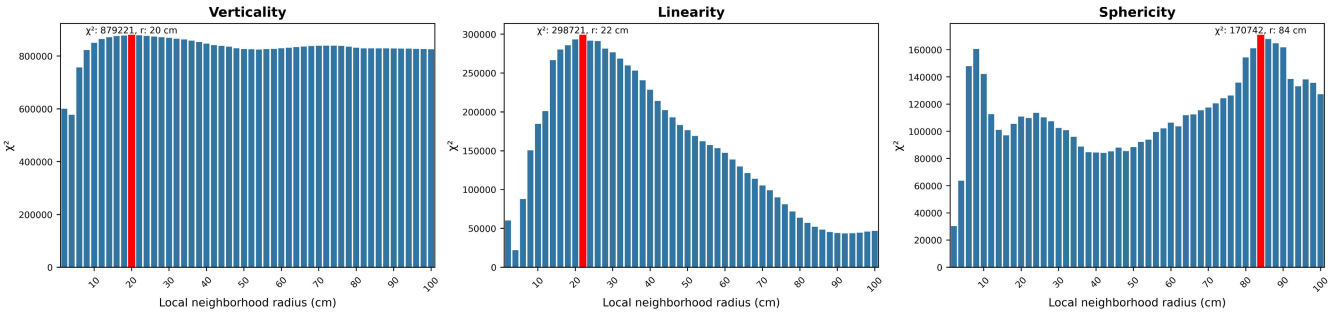

FIGURE 4. Local neighborhood radius graphs for verticality, linearity and sphericity features by applying the chi-squared test.

Machine. From the models mentioned above, Gaussian Naïve Bayes and Support Vector Machine are simpler models used as reference models within this study. The accuracy values for the analyzes are illustrated in Table 6. The lowest accuracy value was obtained in the Gaussian Naïve Bayes ($F_{1,r_{60}} + F_2$) model, 52.40%, and the highest in the Random Forest ($F_{1,r_{\chi^2}} + F_2$) model, 79.66%. It is observed that most of the results that have the highest accuracy are obtained with $F_{1,r_{\chi^2}} + F_2$ features.

F1 scores of the related neighborhood radius and class values of the models are presented in Table 7. While the F1 score of beam, column, other, and window classes is high in the LightGBM ($F_{1,r_{\chi^2}} + F_2$) model, The F1 score of the ceiling, desk, door, floor, lighting, and wall class is higher in the Random Forest ($F_{1,r_{\chi^2}} + F_2$) model model. The F1 scores of the column and window classes are approximately the same in both models.

The confusion matrix of the Random Forest ($F_{1,r_{\chi^2}} + F_2$) model is shown in Fig. 5. It was observed that the classification accuracy of ceiling, floor, lighting, and wall is high. However, some points were classified as walls, even though these are points of the beam, column, door, other, and window class. Moreover, some parts of the floor class were classified as desks. Furthermore, the classification accuracy of the objects represented by lower density is lower than expected, although the balancing procedure was applied.

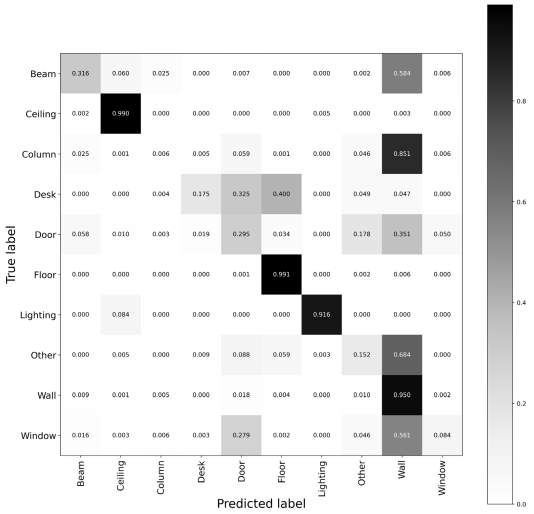

FIGURE 5. Normalized confusion matrix of the Random Forest ($F_{1,r_{\chi^2}} + F_2$) model.

C. EVALUATION OF FEATURE COMBINATIONS

To investigate the impact of features on model accuracy, training, and testing time, analyses were conducted by creating various feature combinations. During the evaluation phase, six distinct feature combinations were experimented. These combinations are illustrated in Table 8. The initial combination (C_1) included all 19 features. The second combination (C_2) excluded RGB color and intensity values. This combination was chosen to investigate the influence of RGB color information obtained through camera and laser scanner fusion and the intensity value of points. These features include features that relate to objects' optical properties, roughness, material type, and light conditions. The use of

TABLE 6. The accuracy values (%) according to the $F_{1,r_n} + F_2$ features.

Features	XGBoost	CatBoost	LightGBM	Random Forest	Gaussian Naïve Bayes	Support Vector Machine
$F_{1,r_{20}}+F_2$	76.33	76.56	76.42	77.07	77.20	78.84
$F_{1,r_{40}}+F_2$	75.16	77.73	76.71	74.47	69.36	75.03
$F_{1,r_{60}}+F_2$	74.24	78.49	75.92	77.05	52.40	62.78
$F_{1,r_{80}}+F_2$	77.01	77.88	76.27	78.99	52.86	66.61
$F_{1,r_{100}}+F_2$	77.06	77.98	78.06	78.88	58.44	67.76
$F_{1,r_{\chi 2}}+F_2$	78.69	78.55	79.26	79.66	69.09	78.07
$F_{1,r_{MI}}+F_2$	77.53	78.25	78.16	79.27	76.93	75.55
$F_{1,r_{F-value}}+F_2$	77.26	77.89	78.71	78.58	58.78	74.66

TABLE 7. F1 scores (%) of classes obtained using $F_{1,r_{opt}} + F_2$ features.

Classes	XGBoost	CatBoost	LightGBM	Random Forest	Gaussian Naïve Bayes	Support Vector Machine
Beam	53.84	45.39	60.10	43.78	51.97	61.92
Ceiling	98.25	98.39	98.01	98.35	96.27	96.47
Column	0.41	1.49	1.24	1.18	27.28	18.27
Desk	0.03	0.17	1.32	28.91	39.62	75.83
Door	32.87	31.90	32.45	32.54	47.39	35.38
Floor	90.20	92.50	92.20	94.34	94.96	94.99
Lighting	92.60	92.36	90.70	93.10	82.41	82.91
Other	22.92	13.58	31.68	18.32	12.27	13.41
Wall	77.46	75.57	79.98	79.17	77.58	75.63
Window	2.39	6.40	9.21	11.65	38.77	32.23

eigenvalue-based features for point cloud classification is a common practice. However, their impact on classification accuracy has yet to be fully understood. To address this issue, the third feature combination (C_3) was generated to exclude these features intentionally. This exclusion provides a more complete understanding of the effects of eigenvalue-based features on classification accuracy. The feature combinations of C_4, C_5, and C_6 were generated to reduce the total number of features. This was achieved by employing statistical tests and observing their effects on accuracy and processing time. Following this, the top 10 important features were selected from the 19 features, reducing approximately 50% of the total number of features. C_4 involved feature selection using the chi-squared, C_5 utilized ANOVA F-test, and C_6 generated with an MI-based feature selection method.

- C_1: $F_{1,r_{opt}} + F_2 \rightarrow$ (19 features)
- C_2: $F_{1,r_{opt}} + F_2$ - Colors (R,G,B) - Intensity $\rightarrow$ (15 features)
- C_3: $F_2 \rightarrow$ (10 features)
- C_4: Feature selection according to chi-squared test for $F_{1,r_{opt}} + F_2 \rightarrow$ (10 features)
- C_5: Feature selection according to ANOVA F-test for $F_{1,r_{opt}} + F_2 \rightarrow$ (10 features)
- C_6: Feature selection according to mutual information for $F_{1,r_{opt}} + F_2 \rightarrow$ (10 features)

The accuracy for different feature combinations is presented in Table 8. For C_2, the XGBoost model decreased by 2.62%, the CatBoost model by 1.14%, the LightGBM model by 3.05%, the Random Forest model by 0.71%, the Gaussian Naïve Bayes model by 1.10%, and the Support Vector Machine model by 3.80%. These results suggest that the model's accuracy is not adversely affected even in errors due to different lighting conditions. In the case of C_3, the XGBoost model decreased by 12.60%, the CatBoost model by 7.68%, the LightGBM model by 12.87%, the Random Forest model by 9.48%, the Gaussian Naïve Bayes model by 13.96%, and the Support Vector Machine model by 18.03%. These findings indicate that eigenvalue-based geometric features significantly impact 3D point cloud classification. After examining the accuracy values of the related combinations, it was observed that in the case of C_4, the XGBoost model increased by 2.67%, the CatBoost model by 4.41%, and the LightGBM model increased by 1.39%, the Random Forest model by 2.43%, the Gaussian Naïve Bayes model by 0.35%, and the Support Vector Machine model decreased by 4.00%. After applying feature selection (C_5), the accuracy of the XGBoost, CatBoost, and Random Forest models increased by 0.90%, 1.01%, and 1.47%, respectively. However, the accuracy of LightGBM, Gaussian Naïve Bayes, and Support Vector Machine models decreased by 0.03%, 0.32%, and 1.29%. Despite reducing the features in the dataset by 47%, the accuracy results remained approximately the same. On the other hand, when feature selection was made using mutual information (C_6), there was a decrease of 5.25%

in the XGBoost model, 3.21% in the CatBoost model, 6.83% in the LightGBM model, 5.63% in the Random Forest model, 0.82% in the Gaussian Naïve Bayes model, and 9.80% in the Support Vector Machine model.

In order to compare processing times consistently, hardware capacity was kept constant in all processes. The hardware specifications of which are explained in detail in Table 2. The effects of different feature combinations on training and testing time are shown in Table 9 and Table 10, respectively. When the results were examined, it was observed that reducing the features used in model training by applying feature selection did not affect the training or testing time. The training times for different feature combinations have been detailed in Table 9. Upon evaluating the model performance regarding average training time, the CatBoost model exhibits the highest training time of 3053.159 seconds, while the Gaussian Naïve Bayes model demonstrates the lowest training time of merely 1.056 seconds.

The testing times for feature combinations are listed in Table 10. When the model performance is evaluated in terms of average testing time the LightGBM model have the longest average testing time, with 50.783 seconds. On the other hand, the Support Vector Machine model has the shortest average testing time, taking only 0.792 seconds.

The balance between accuracy and exact processing time is essential for computational efficiency when dealing with practical applications. When comparing the Random Forest and CatBoost, it becomes apparent that there is a notable difference in accuracy, training, and testing time. Although both models have similar accuracy across all feature combinations, they differ sharply in processing time. For example, the Random Forest model achieved 82.09% accuracy for the C_4 combination, with a training time of 106.142 seconds and a testing time of 9.272 seconds. In comparison, the CatBoost achieved the highest accuracy, with 82.96% for the same feature combination, but with a longer training time of 2422.738 seconds and a testing time of 14.237 seconds. Despite the CatBoost having the highest accuracy, the Random Forest only lost 0.87% accuracy for the C_4 feature combination. On the other hand, the Random Forest saved 2316.596 seconds of training time and 4.965 seconds of testing time. The same results have been observed in other feature combinations for relevant models. Since feature selection techniques are applied when creating C_4, C_5, and C_6 feature combinations, including these processing times in the total processing time is essential. The ten most important features among 19 features were selected statistically, using the Chi-squared test, ANOVA F-test, and Mutual Information for relevant feature combinations. The Chi-squared test was applied for the C_4 combination. It took 1.132 seconds, the ANOVA F-test for the C_5 combination took 1.147 seconds, and the Mutual Information calculation for the C_6 feature combination with 449.119 seconds.

The results show that the Chi-squared test and ANOVA F-test have similar processing times. However, the highest

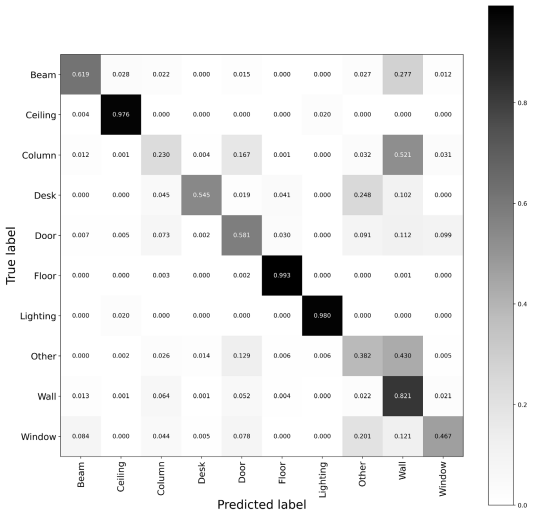

FIGURE 6. Normalized confusion matrix of CatBoost ($F_{1,r_{\chi 2}} + F_2$, C_4) model.

accuracy was obtained in four out of six machine learning models when feature selection was applied with the Chi-squared test. In contrast, using Mutual Information resulted in a decrease in accuracy and longer processing time when compared to other methods. Therefore, the Chi-squared test is the most suitable statistical feature selection method among the compared methods regarding processing time and accuracy. When the highest accuracy among all feature combinations is examined, it has been determined that the highest accuracy can be achieved by training CatBoost ($F_{1,r_{\chi 2}} + F_2$, C_4) model. The model's class value performance metrics have been analyzed in Table 11. It has been noticed that the column has the lowest F1 score of only 23.85%, while the floor class has the highest F1 score, impressively coming in at 98.49%. According to the results in Table 7, the F1 scores for columns, desks, and windows are very low when using CatBoost. However, the F1 scores for column, desk, and window presented in Table 11 for the CatBoost are significantly better. The reason for such an increase could be due to the fact that Table 7 deploys 19 features, whereas Table 11 uses the best ten features selected statistically. Hence, removing these nine features, such as intensity, anisotropy, and the sum of eigenvalues, significantly improved the classification results within the CatBoost model.

The confusion matrix of the model is shown in Fig. 6. There were misclassifications during the training, where particular beams, columns, and other classes were predicted as walls, and some desk and window classes were mixed with other classes.

A visual comparison of the results obtained by training the CatBoost model with the reference data is presented in Fig. 7. Fig. 7.a includes the general view of the predictions, Fig. 7.b the general view of the ground truth, Fig. 7.c the predictions without ceilings and lightings so that the inside of the building

TABLE 8. The accuracy values (%) of feature combinations for $F_{1,r_{opt}} + F_2$.

C_n	XGBoost	CatBoost	LightGBM	Random Forest	Gaussian Naïve Bayes	Support Vector Machine
C_1	78.69	78.55	79.26	79.66	77.20	78.84
C_2	76.07	77.41	76.21	78.95	76.10	75.04
C_3	66.09	70.87	66.39	70.18	63.24	60.81
C_4	81.36	82.96	80.65	82.09	77.55	74.84
C_5	79.59	79.56	79.23	81.13	76.88	77.55
C_6	73.44	75.34	72.43	74.03	78.02	69.04

TABLE 9. Training times (second) of $F_{1,r_{opt}} + F_2$ features.

C_n	XGBoost	CatBoost	LightGBM	Random Forest	Gaussian Naïve Bayes	Support Vector Machine
C_1	735.976	4994.650	40.882	165.593	1.241	917.203
C_2	713.281	3795.859	30.938	98.554	1.031	1612.149
C_3	1189.645	2735.194	107.862	70.070	0.780	1119.812
C_4	506.164	2422.738	36.961	106.142	1.044	440.021
C_5	1638.909	1822.277	207.380	144.845	1.103	524.409
C_6	1404.862	2548.234	34.661	81.942	1.134	503.595
t_{avg}	1031.473	3053.159	76.447	111.191	1.056	852.865

TABLE 10. Testing times (second) of $F_{1,r_{opt}} + F_2$ features.

C_n	XGBoost	CatBoost	LightGBM	Random Forest	Gaussian Naïve Bayes	Support Vector Machine
C_1	8.255	6.901	27.392	13.833	8.698	0.541
C_2	8.088	15.014	14.200	9.283	6.618	1.479
C_3	27.193	8.197	95.511	8.942	4.785	1.289
C_4	7.903	14.237	35.419	9.272	6.814	0.495
C_5	16.954	7.289	104.227	23.306	7.317	0.474
C_6	24.464	11.640	27.947	8.949	5.966	0.475
t_{avg}	15.476	10.546	50.783	12.264	6.700	0.792

TABLE 11. Precision (%), Recall (%), and F1 scores (%) of classes obtained CatBoost ($F_{1,r_{\chi 2}} + F_2$, C_4) model.

Class	Precision	Recall	F1 score
Beam	85.04	61.91	71.65
Ceiling	99.03	97.60	98.31
Column	24.77	23.00	23.85
Desk	95.83	54.48	69.47
Door	53.64	58.11	55.78
Floor	97.74	99.25	98.49
Lighting	81.98	98.03	89.29
Other	32.69	38.15	35.21
Wall	77.52	82.15	79.76
Window	27.82	46.69	34.87

can be seen, and Fig. 7.d the general view of the ground truth data without ceilings and lightings.

D. MODEL INTERPRETATION WITH EXPLAINABLE MACHINE LEARNING

The Shapley values were used to examine the impacts of the feature for the trained model in each class. The average impacts of features on the predictions are illustrated in Fig. 8. The average impact of the Z coordinate and verticality feature on the model is relatively high. In contrast, the contribution of the linearity feature to the model prediction values is relatively low.

Examining the confusion matrix in Fig. 6, it is clear that the wall and column classes have been mixed up. Hence, examining the impacts of the features used in model training on the wall and column classes is necessary. In this context, after examining Fig. 9, it is seen that low and high values of verticality and Z coordinate, which have high impacts on column and wall classes, show similar behavior. Therefore, the accuracy of the wall and column classes decreases.

After analyzing the confusion matrix in Fig. 6, it becomes apparent that the desk and other classes are mixing. In order to comprehend this situation, it is essential to examine how the characteristics utilized in the training of models can affect these categories. Upon examining Fig. 10, it is seen that the Z coordinate, which is the most impactful feature representing the column and wall classes, shows similar behavior for both low and high values. As a result, the accuracy of the desk and other classes decreases.

According to the achieved results, although the floor and floor classes are geometrically similar, they are classified

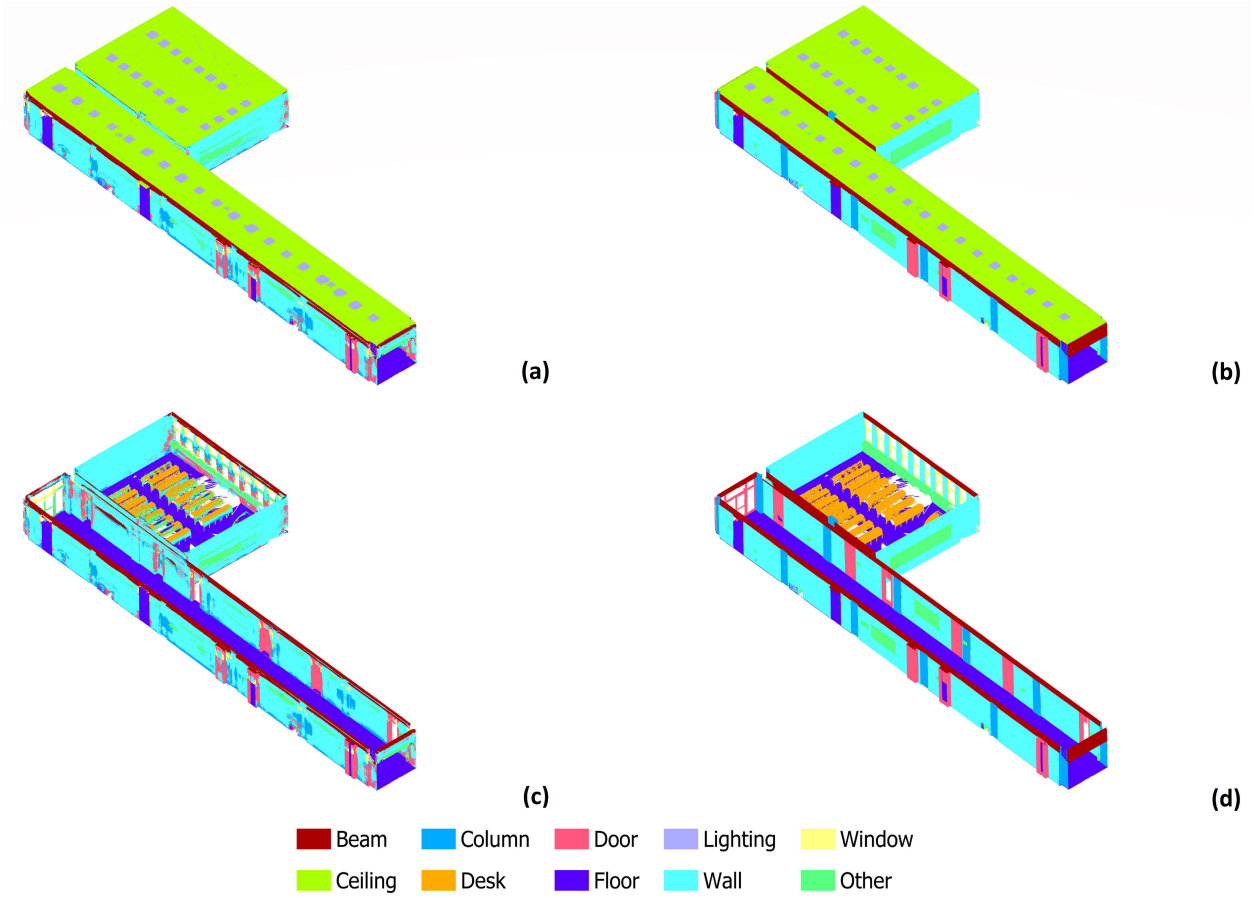

FIGURE 7. Inference of CatBoost ($F_{1,r_{\chi^2}} + F_2, C_4$) model: (a) General view according to predicted values. (b) General view of ground truth. (c) General view without lighting and ceiling according to predicted values. (d) General view without lighting and ceiling regarding ground truth.

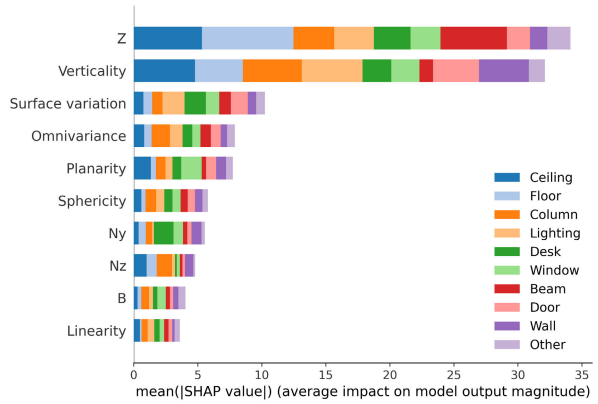

FIGURE 8. Average impact of features on predictions for the CatBoost ($F_{1,r_{\chi^2}} + F_2, C_4$) model.

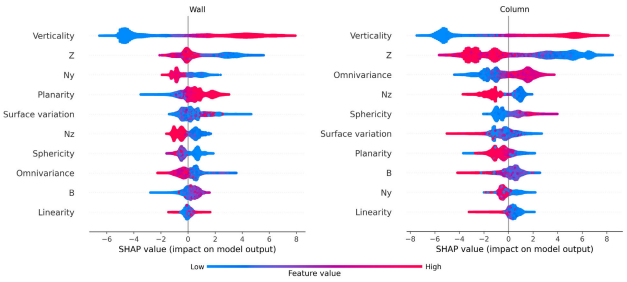

FIGURE 9. Impact of the features on wall class and column class for CatBoost ($F_{1,r_{\chi^2}} + F_2, C_4$) model.

with high success rates. When Fig. 11 is examined, the two most important features are identical for both classes. However, the high and low values of Z coordinate, which is the most important feature, are the opposite of each other. Thus, this has ensured that the two classes do not mix with each other.

When using the Random Forest model and the CatBoost model for different applications, it is crucial to consider their accuracy and time consumption and the number of hyperparameters and supported processing units. The Random Forest model has fewer hyperparameters than the CatBoost model. The CatBoost model may take longer to process in studies where hyperparameter optimization is essential. However, it is more advantageous than the Random Forest model when customizing the model according to

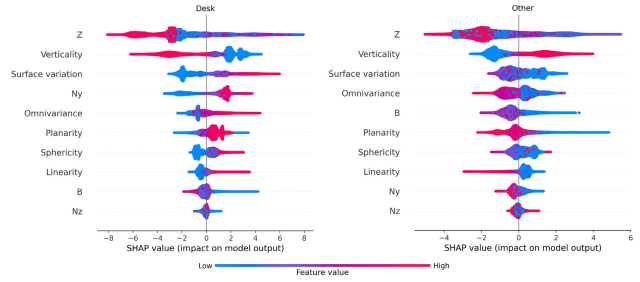

FIGURE 10. Impact of the features on desk class and other class for CatBoost ($F_{1,r_{\chi^2}} + F_2$, C_4) model.

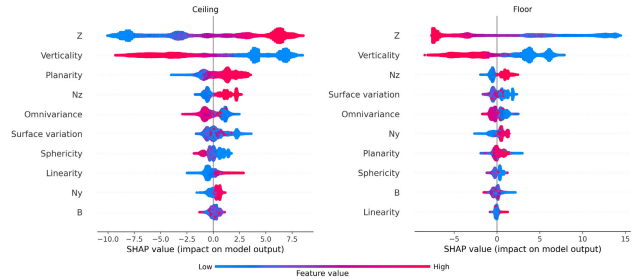

FIGURE 11. Impact of the features on ceiling and floor class for CatBoost ($F_{1,r_{\chi^2}} + F_2$, C_4) model.

the dataset. Both models support a central processing unit (CPU) and graphics processing unit (GPU). In this study, calculations were made on the CPU, but using the GPU calculation unit can significantly reduce processing time. In addition, when evaluating the results, it is also essential to consider factors such as the dataset used, the dimensions of the classified objects, their spatial relationships, the environment's lighting conditions, the type of objects present, and surface roughness. These factors are specific to the dataset used. The parameters used in the pre-processing and feature extraction phases are determined based on the hardware capacity and not just the dataset. To achieve similar accuracies in different studies, all these conditions must be considered, especially for datasets containing objects of different sizes. Object size and spatial relationships are critical factors in radius selection, especially for eigenvalue-based features. Model training and pre-processing steps were performed in a high-capacity grid computing system. Hardware capacity can be a potential limitation when dealing with larger data sets. To overcome this limitation, the method and parameter selection in subsampling applied to the data are very important to maintain the model accuracy. Here, it is necessary to determine the appropriate subsampling method and parameters in a way that will least affect the geometric structure of the data.

IV. CONCLUSION

This study successfully applied machine learning methods to classify structural components from 3D point clouds in indoor environments. A proposed novel framework consists of four stages: pre-processing, geometric feature extraction, feature selection, and interpretation of the classification results. Six machine learning models, namely XGBoost, CatBoost, LightGBM, Random Forest, Gaussian Naïve Bayes, and Support Vector Machine, were used to evaluate classification performance regarding their accuracy and processing time. The Chi-squared test is the most appropriate statistical method for selecting the optimal local neighborhood radius. The same statistical method is proposed for feature selection as well. According to the classification results, the CatBoost model has the highest accuracy, and the Random Forest is the most suitable model for a good balance between accuracy and calculation efficiency. Hence, the Random Forest model is well-suited for practical applications. The Shapley values are used to interpret model classification results, where the CatBoost model has the highest accuracy. According to the model interpretation, among 19 features, Z coordinate, and verticality features have a relatively high impact on classification. The blue color and linearity feature has the lowest impact on classification results. Overall, the findings offer valuable insights for future studies in indoor building environments and could ultimately contribute to advancing indoor 3D point cloud analysis and applications. The most appropriate processing time, geometric features, and models are obtained without data loss, and structural components from 3D point clouds are automatically classified. Hence, following the framework described, this study could be replicated successfully for further indoor environments.

ACKNOWLEDGMENT

The numerical calculations reported in this paper were partially performed at Turkish National e-Science e-Infrastructure (TRUBA).

REFERENCES

[1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[2] Q. Wang and M.-K. Kim, "Applications of 3D point cloud data in the construction industry: A fifteen-year review from 2004 to 2018," *Adv. Eng. Informat.*, vol. 39, pp. 306–319, Jan. 2019.

[3] R. Pierdicca, M. Paolanti, F. Matrone, M. Martini, C. Morbidoni, E. S. Malinverni, E. Frontoni, and A. M. Lingua, "Point cloud semantic segmentation using a deep learning framework for cultural heritage," *Remote Sens.*, vol. 12, no. 6, p. 1005, Mar. 2020.

[4] K. Mirzaei, M. Arashpour, E. Asadi, H. Masoumi, Y. Bai, and A. Behnood, "3D point cloud data processing with machine learning for construction and infrastructure applications: A comprehensive review," *Adv. Eng. Informat.*, vol. 51, Jan. 2022, Art. no. 101501.

[5] V. A. Cotella, "From 3D point clouds to HBIM: Application of artificial intelligence in cultural heritage," *Autom. Construct.*, vol. 152, Aug. 2023, Art. no. 104936.

[6] M. Andriasyan, J. Moyano, J. E. Nieto-Julián, and D. Antón, "From point cloud data to building information modelling: An automatic parametric workflow for heritage," *Remote Sens.*, vol. 12, no. 7, p. 1094, Mar. 2020.

[7] S. Yang, M. Hou, and S. Li, "Three-dimensional point cloud semantic segmentation for cultural heritage: A comprehensive review," *Remote Sens.*, vol. 15, no. 3, p. 548, Jan. 2023.

[8] Q. Cheng, P. Sun, C. Yang, Y. Yang, and P. X. Liu, "A morphing-based 3D point cloud reconstruction framework for medical image processing," *Comput. Methods Programs Biomed.*, vol. 193, Sep. 2020, Art. no. 105495.

[9] D. Xiao, C. Lian, H. Deng, T. Kuang, Q. Liu, L. Ma, D. Kim, Y. Lang, X. Chen, J. Gateno, S. G. Shen, J. J. Xia, and P.-T. Yap, "Estimating reference bony shape models for orthognathic surgical planning using 3D point-cloud deep learning," *IEEE J. Biomed. Health Informat.*, vol. 25, no. 8, pp. 2958–2966, Aug. 2021.

[10] X. Wang, X. Zhang, X. Ren, L. Li, H. Feng, Y. He, H. Chen, and X. Chen, "Point cloud 3D parent surface reconstruction and weld seam feature extraction for robotic grinding path planning," *Int. J. Adv. Manuf. Technol.*, vol. 107, nos. 1–2, pp. 827–841, Mar. 2020.

[11] L. Yang, Y. Liu, J. Peng, and Z. Liang, "A novel system for off-line 3D seam extraction and path planning based on point cloud segmentation for arc welding robot," *Robot. Comput.-Integr. Manuf.*, vol. 64, Aug. 2020, Art. no. 101929.

[12] Q. Jin, Q. Hu, P. Zhao, S. Wang, and M. Ai, "An improved probabilistic roadmap planning method for safe indoor flights of unmanned aerial vehicles," *Drones*, vol. 7, no. 2, p. 92, Jan. 2023.

[13] Z. Liu, R. Fu, L. Wang, Y. Jin, T. Papakostas, X. U. Mainelli, R. Voute, and E. Verbree, "Game engine-based point cloud visualization and perception for situation awareness of crisis indoor environments," in *Proc. 16th Int. Conf. Location Based Services*, 2021, pp. 183–194.

[14] J.-P. Virtanen, S. Daniel, T. Turppa, L. Zhu, A. Julin, H. Hyyppä, and J. Hyyppä, "Interactive dense point clouds in a game engine," *ISPRS J. Photogramm. Remote Sens.*, vol. 163, pp. 375–389, May 2020.

[15] E. Alexiou, E. Upenik, and T. Ebrahimi, "Towards subjective quality assessment of point cloud imaging in augmented reality," in *Proc. IEEE 19th Int. Workshop Multimedia Signal Process. (MMSP)*, Oct. 2017, pp. 1–6.

[16] A. Kharroubi, R. Hajji, R. Billen, and F. Poux, "Classification and integration of massive 3D points clouds in a virtual reality (VR) environment," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 42, pp. 165–171, Nov. 2019.

[17] J. Döllner, "Geospatial artificial intelligence: Potentials of machine learning for 3D point clouds and geospatial digital twins," *J. Photogramm., Remote Sens. Geoinformation Sci.*, vol. 88, no. 1, pp. 15–24, Feb. 2020, doi: 10.1007/s41064-020-00102-3.

[18] F. Xue, W. Lu, Z. Chen, and C. J. Webster, "From LiDAR point cloud towards digital twin city: Clustering city objects based on gestalt principles," *ISPRS J. Photogramm. Remote Sens.*, vol. 167, pp. 418–431, Sep. 2020.

[19] K. Mirzaei, M. Arashpour, E. Asadi, H. Masoumi, and H. Li, "Automatic generation of structural geometric digital twins from point clouds," *Sci. Rep.*, vol. 12, no. 1, p. 22321, Dec. 2022.

[20] F. Bosché, M. Ahmed, Y. Turkan, C. T. Haas, and R. Haas, "The value of integrating scan-to-BIM and scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components," *Autom. Construct.*, vol. 49, pp. 201–213, Jan. 2015.

[21] R. Romero-Jarén and J. J. Arranz, "Automatic segmentation and classification of BIM elements from point clouds," *Autom. Construct.*, vol. 124, Apr. 2021, Art. no. 103576.

[22] Y. Perez-Perez, M. Golparvar-Fard, and K. El-Rayes, "Scan2BIM-NET: Deep learning method for segmentation of point clouds for scan-to-BIM," *J. Construct. Eng. Manage.*, vol. 147, no. 9, Sep. 2021, Art. no. 04021107.

[23] J. Chen, S. Li, and W. Lu, "Align to locate: Registering photogrammetric point clouds to BIM for robust indoor localization," *Building Environ.*, vol. 209, Feb. 2022, Art. no. 108675.

[24] M. Mohamed, S. Morsy, and A. El-Shazly, "Evaluation of data subsampling and neighbourhood selection for mobile LiDAR data classification," *Egyptian J. Remote Sens. Space Sci.*, vol. 24, no. 3, pp. 799–804, Dec. 2021.

[25] T. Han and G. A. Sánchez-Azofeifa, "Extraction of liana stems using geometric features from terrestrial laser scanning point clouds," *Remote Sens.*, vol. 14, no. 16, p. 4039, Aug. 2022.

[26] H. Aljumaily, D. F. Laefer, D. Cuadra, and M. Velasco, "Point cloud voxel classification of aerial urban LiDAR using voxel attributes and random forest approach," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 118, Apr. 2023, Art. no. 103208.

[27] B. Xiang, J. Yao, X. Lu, L. Li, R. Xie, and J. Li, "Segmentation-based classification for 3D point clouds in the road environment," *Int. J. Remote Sens.*, vol. 39, no. 19, pp. 6182–6212, Oct. 2018, doi: 10.1080/01431161.2018.1455235.

[28] S. M. Krishna Moorthy, K. Calders, M. B. Vicari, and H. Verbeeck, "Improved supervised learning-based approach for leaf and wood classification from LiDAR point clouds of forests," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 5, pp. 3057–3070, May 2020.

[29] F. J. Aguilar, A. Nemmaoui, M. A. Aguilar, and A. Peñalver, "Building tree allometry relationships based on TLS point clouds and machine learning regression," *Appl. Sci.*, vol. 11, no. 21, p. 10139, Oct. 2021.

[30] Y. Perez-Perez, M. Golparvar-Fard, and K. El-Rayes, "Segmentation of point clouds via joint semantic and geometric features for 3D modeling of the built environment," *Autom. Construct.*, vol. 125, May 2021, Art. no. 103584.

[31] B. Koo, R. Jung, and Y. Yu, "Automatic classification of wall and door BIM element subtypes using 3D geometric deep neural networks," *Adv. Eng. Informat.*, vol. 47, Jan. 2021, Art. no. 101200.

[32] M. E. Atik, Z. Duran, and D. Z. Seker, "Machine learning-based supervised classification of point clouds using multiscale geometric features," *ISPRS Int. J. Geo-Inf.*, vol. 10, no. 3, p. 187, Mar. 2021.

[33] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, May 2017, pp. 652–660.

[34] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–32.

[35] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation," 2018, *arXiv:1807.00652*.

[36] C. R. Qi, O. Litany, K. He, and L. Guibas, "Deep Hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9276–9285.

[37] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4866–4875.

[38] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11105–11114.

[39] C. Yin, B. Wang, V. J. L. Gan, M. Wang, and J. C. P. Cheng, "Automated semantic segmentation of industrial point clouds using ResPointNet++," *Autom. Construct.*, vol. 130, Oct. 2021, Art. no. 103874.

[40] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6410–6419.

[41] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[42] M. M. Taye, "Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions," *Computers*, vol. 12, no. 5, p. 91, Apr. 2023.

[43] M. Weinmann, B. Jutzi, S. Hinz, and C. Mallet, "Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers," *ISPRS J. Photogramm. Remote Sens.*, vol. 105, pp. 286–304, Jul. 2015.

[44] V. Kasireddy and B. Akinci, "Assessing the impact of 3D point neighborhood size selection on unsupervised spall classification with 3D bridge point clouds," *Adv. Eng. Informat.*, vol. 52, Apr. 2022, Art. no. 101624.

[45] A. Capolupo, "Accuracy assessment of cultural heritage models extracting 3D point cloud geometric features with RPAS SfM-MVS and TLS techniques," *Drones*, vol. 5, no. 4, p. 145, Dec. 2021.

[46] M. E. Atik and Z. Duran, "Selection of relevant geometric features using filter-based algorithms for point cloud semantic segmentation," *Electronics*, vol. 11, no. 20, p. 3310, Oct. 2022.

[47] S. Morsy and A. Shaker, "Evaluation of LiDAR-derived features relevance and training data minimization for 3D point cloud classification," *Remote Sens.*, vol. 14, no. 23, p. 5934, Nov. 2022.

[48] M. A. Gunen, "Adaptive neighborhood size and effective geometric features selection for 3D scattered point cloud classification," *Appl. Soft Comput.*, vol. 115, Jan. 2022, Art. no. 108196.

[49] O. Ozturk, M. S. Isik, M. Kada, and D. Z. Seker, "Improving road segmentation by combining satellite images and LiDAR data with a feature-wise fusion strategy," *Appl. Sci.*, vol. 13, no. 10, p. 6161, May 2023.
[50] J. Demantké, C. Mallet, N. David, and B. Vallet, "Dimensionality based scale selection in 3D LiDAR point clouds," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 38, pp. 97–102, Sep. 2012.
[51] Y. Xu, X. Tong, and U. Stilla, "Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry," *Autom. Construct.*, vol. 126, Jun. 2021, Art. no. 103675.
[52] A. Kumar, K. Anders, L. Winiwarter, and B. Höfle, "Feature relevance analysis for 3D point cloud classification using deep learning," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 4, pp. 373–380, May 2019.
[53] M. Weinmann, B. Jutzi, C. Mallet, and M. Weinmann, "Geometric features and their relevance for 3D point cloud classification," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 4, pp. 157–164, May 2017.
[54] I. K. Fodor, *A Survey of Dimension Reduction Techniques*. Livermore, CA, USA: Lawrence Livermore National Lab, 2002.
[55] B. Venkatesh and J. Anuradha, "A review of feature selection and its methods," *Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, Mar. 2019.
[56] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data," *Knowl. Inf. Syst.*, vol. 34, no. 3, pp. 483–519, Mar. 2013, doi: 10.1007/s10115-012-0487-8.
[57] K. Hopf and S. Reifenrath, "Filter methods for feature selection in supervised machine learning applications—Review and benchmark," 2021, *arXiv:2111.12140*.
[58] A. Bommert, T. Welchowski, M. Schmid, and J. Rahnenfuhrer, "Benchmark of filter methods for feature selection in high-dimensional gene expression survival data," *Briefings Bioinf.*, vol. 23, no. 1, Jan. 2022, Art. no. bbab354.
[59] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surveys*, vol. 50, no. 6, pp. 1–45, Nov. 2018, doi: 10.1145/3136625.
[60] D. Jain and V. Singh, "Feature selection and classification systems for chronic disease prediction: A review," *Egyptian Informat. J.*, vol. 19, no. 3, pp. 179–189, Nov. 2018.
[61] P. Dhal and C. Azad, "A comprehensive survey on feature selection in the various fields of machine learning," *Int. J. Speech Technol.*, vol. 52, no. 4, pp. 4543–4581, Mar. 2022, doi: 10.1007/s10489-021-02550-9.
[62] X. Li and L. Zhang, "Unbalanced data processing using deep sparse learning technique," *Future Gener. Comput. Syst.*, vol. 125, pp. 480–484, Dec. 2021.
[63] S. Tyagi and S. Mittal, "Sampling approaches for imbalanced data classification problem in machine learning," in *Proc. ICRIC*, vol. 597, 2019, pp. 209–221.
[64] R. Barandela, R. M. Valdovinos, J. S. Snchez, and F. J. Ferri, "The imbalanced training sample problem: Under or over sampling?" *Struct., Syntactic, Stat. Pattern Recognit.*, vol. 3138, pp. 806–814, Nov. 2004.
[65] N. Junsomboon and T. Phienthrakul, "Combining over-sampling and under-sampling techniques for imbalance dataset," in *Proc. 9th Int. Conf. Mach. Learn. Comput.*, Feb. 2017, pp. 243–247, doi: 10.1145/3055635.3056643.
[66] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Appl. Soft Comput.*, vol. 97, Dec. 2020, Art. no. 105524.
[67] D. Girardeau-Montaut, "Cloud compare," *France, EDF RD Telecom ParisTech*, vol. 11, no. 5, pp. 1–38, 2016.
[68] F. Pedregosa, S. Varoquaux, A. Gramfort, V. Michel, and B. Thirion, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Dec. 2011.
[69] T. M. T. A. Hamid, R. Sallehuddin, Z. M. Yunos, and A. Ali, "Ensemble based filter feature selection with harmonize particle swarm optimization and support vector machine for optimal cancer classification," *Mach. Learn. Appl.*, vol. 5, Sep. 2021, Art. no. 100054.
[70] A. F. Alkarkhi, *Applications of Hypothesis Testing for Environmental Science*. Amsterdam, The Netherlands: Elsevier, 2020.
[71] J. R. Vergara and P. A. Estévez, "A review of feature selection methods based on mutual information," *Neural Comput. Appl.*, vol. 24, no. 1, pp. 175–186, Jan. 2014.
[72] T. Chen and T. He, "XGBoost: Extreme gradient boosting," *R Package Version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015.
[73] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–23.
[74] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–13.
[75] G. Biau and E. Scornet, "A random forest guided tour," *TEST*, vol. 25, no. 2, pp. 197–227, Jun. 2016, doi: 10.1007/s11749-016-0481-7.
[76] A. H. Jahromi and M. Taheri, "A non-parametric mixture of Gaussian naive Bayes classifiers based on local independent features," in *Proc. Artif. Intell. Signal Process. Conf. (AISP)*, Oct. 2017, pp. 209–212.
[77] M. Awad and R. Khanna, *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Cham, Switzerland: Springer, 2015.
[78] A. B. Arrieta, N. Díaz-Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82–115, Jun. 2020.
[79] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.
[80] A. E. Roth, *The Shapley Value: Essays in Honor of Lloyd S. Shapley*. Cambridge, U.K.: Cambridge Univ. Press, 1988.
[81] Y. Nohara, K. Matsumoto, H. Soejima, and N. Nakashima, "Explanation of machine learning models using Shapley additive explanation and application for real data in hospital," *Comput. Methods Programs Biomed.*, vol. 214, Feb. 2022, Art. no. 106584.
[82] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, and W. Samek, "Explainable AI methods—A brief overview," in *Proc. Int. Workshop Extending Explainable AI Beyond Deep Models*, 2022, pp. 13–38.
[83] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–17.

KORAY AKSU (Graduate Student Member, IEEE) received the bachelor's degree in geomatics engineering and civil engineering and the M.Sc. degree in geomatics engineering from Istanbul Technical University, Türkiye. He is currently pursuing the Ph.D. degree in geomatics engineering, with a focus on geographic information systems, agent-based modeling, and deep/reinforcement learning. His current research interests include spatial data analysis, BIM and game engine integration, rule-based spatial simulations, 3D reconstruction, scan to BIM, and point cloud segmentation.

HANDE DEMIREL (Member, IEEE) received the Ph.D. degree in geoinformatics from Technical University Berlin (TUB), Germany. She was acted as a Scientific Researcher with the Institute for Prospective Technological Studies (IPTS), European Commission's Joint Research Centre (EC-JRC), from 2011 to 2014. She is currently a Professor with the Department of Geomatics Engineering, Istanbul Technical University. She is giving lectures on geographic information systems, systems engineering, and photogrammetry.