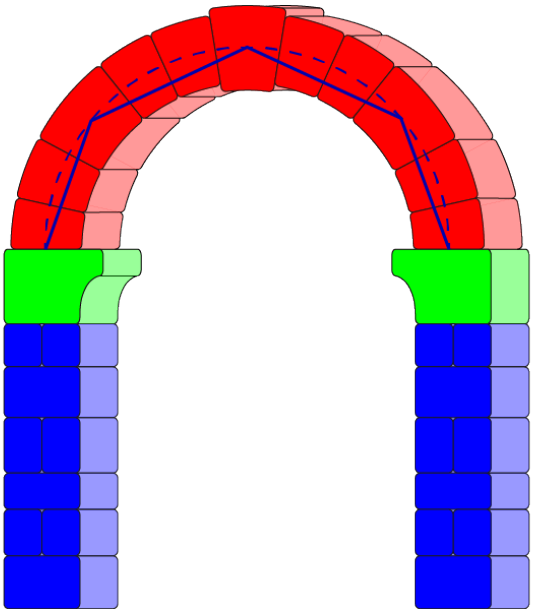


SEG4D

Segmentation for Cultural Heritage Diagnosis

This software has been funded by the Comunidad de Madrid through the call Research Grants for Young Investigators from Universidad Politécnica de Madrid

Comunidad de Madrid

Version: Alpha

USER GUIDE

Construction System Segmentation

Index

1 Construction system segmentation

In this tab you can perform an automatic segmentation of the construction systems of the point cloud of a historical building using artificial intelligence methods (machine learning and deep learning). You can also generate synthetic point clouds for the training phase and export your point cloud into BIM environment.

IMPORTANT: You must first prepare your point cloud to work correctly with this software. The following section «Pre-processing» explains how to do it.

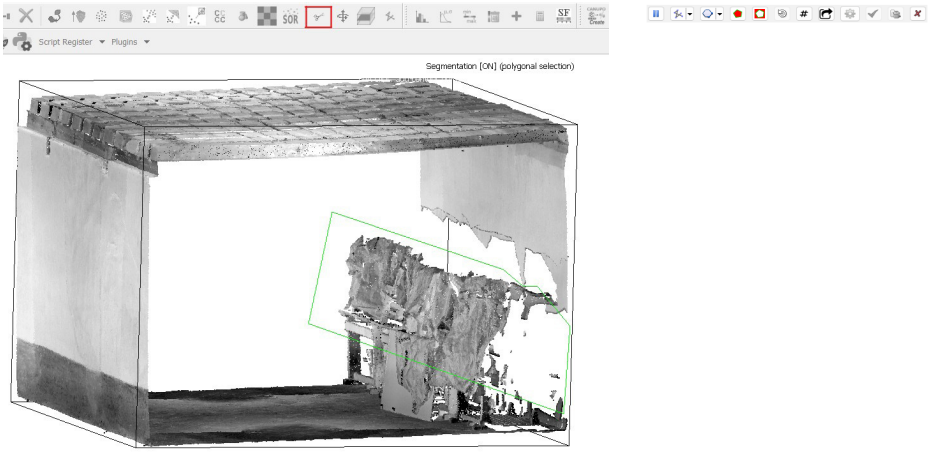
Pre-processing

Cleaning the point cloud

First of all, you will have to remove from your point cloud everything that is not «architectural», e.g. people, furniture, trees, etc. To do this, you can use the «scissors» tool to cut out everything you are not interested in.

Workflow:

1. *Edit*
2. *Segment*
3. *Inside*
4. *Offside*
5. *Cut*

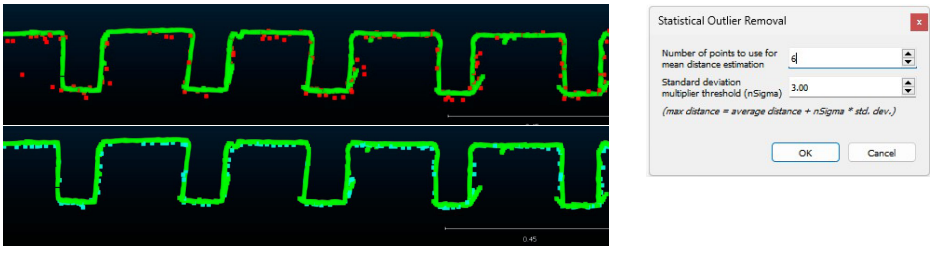


1.1. *Use of «scissors» tool.*

Then, to leave the point cloud cleaner, we recommend that you use the S.O.R. (Statiscal Outlier Removal) tool to remove all the scattered points that do not correspond to your point cloud.

Workflow:

1. *Tools*
2. *Clean*
3. *S.O.R. filter*

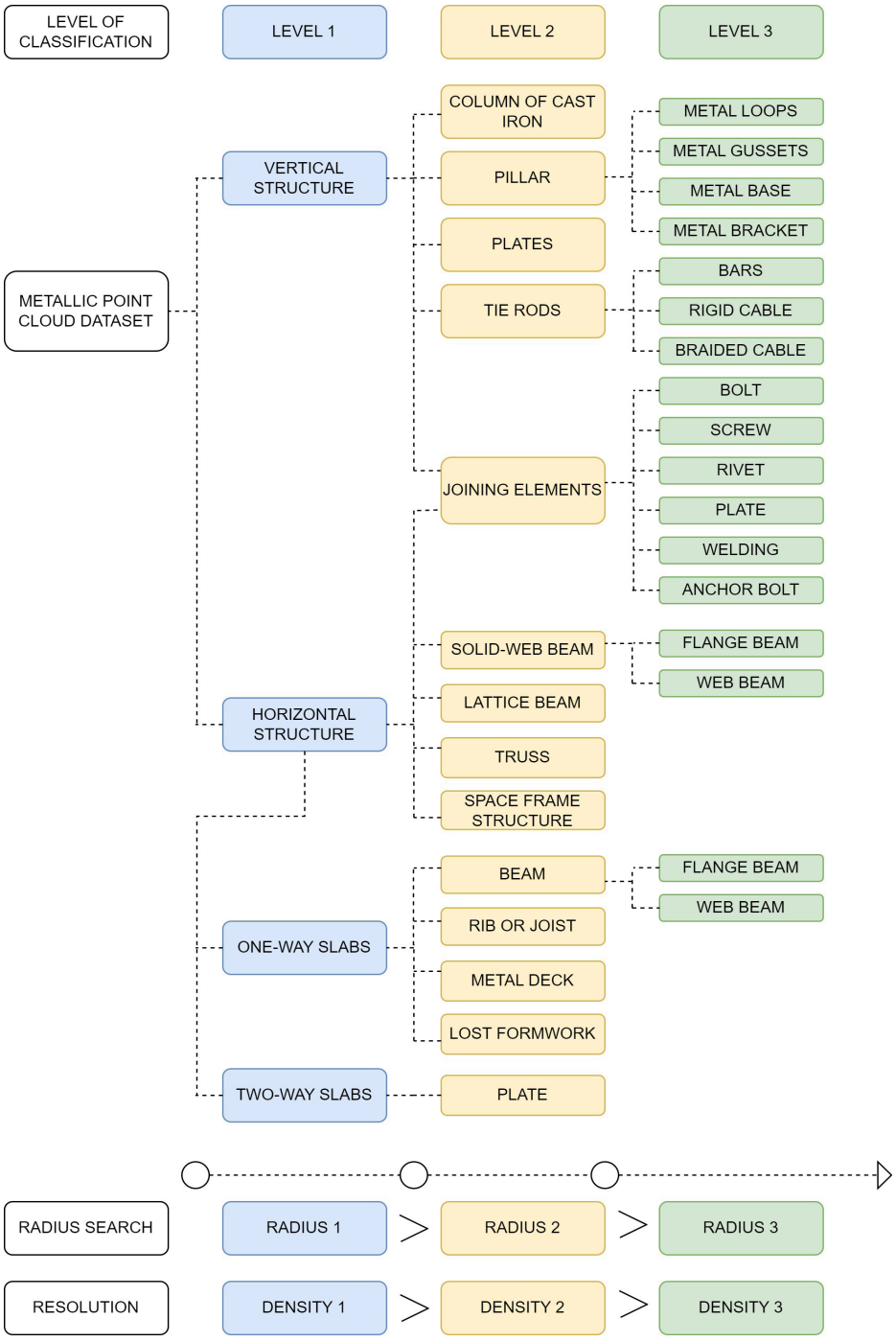


1.2. *Use of «S.O.R.» tool.*

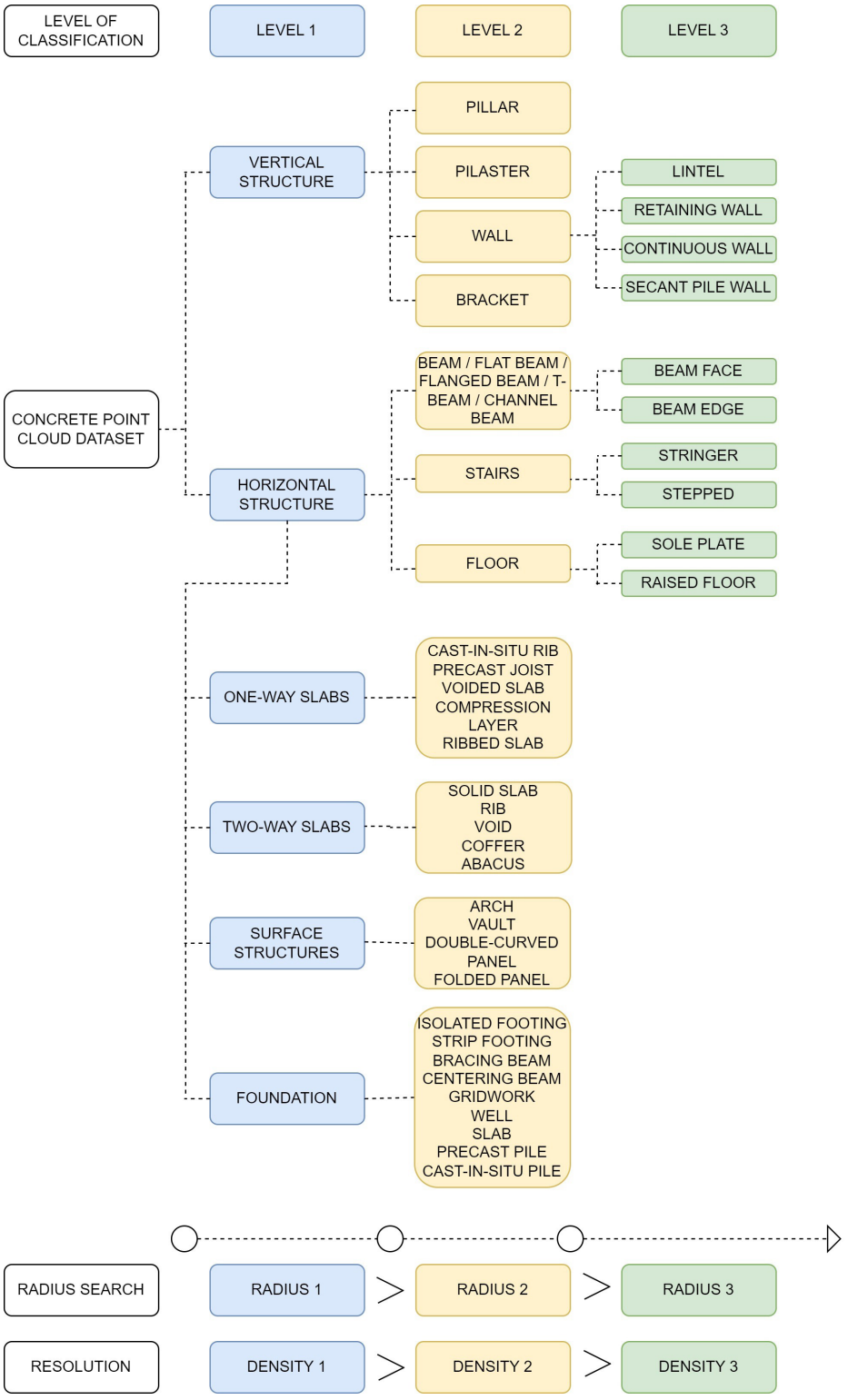
Recommended parameters:
- *Number of points:* 6
- *Standard deviation:* 3

Segmentation proposal

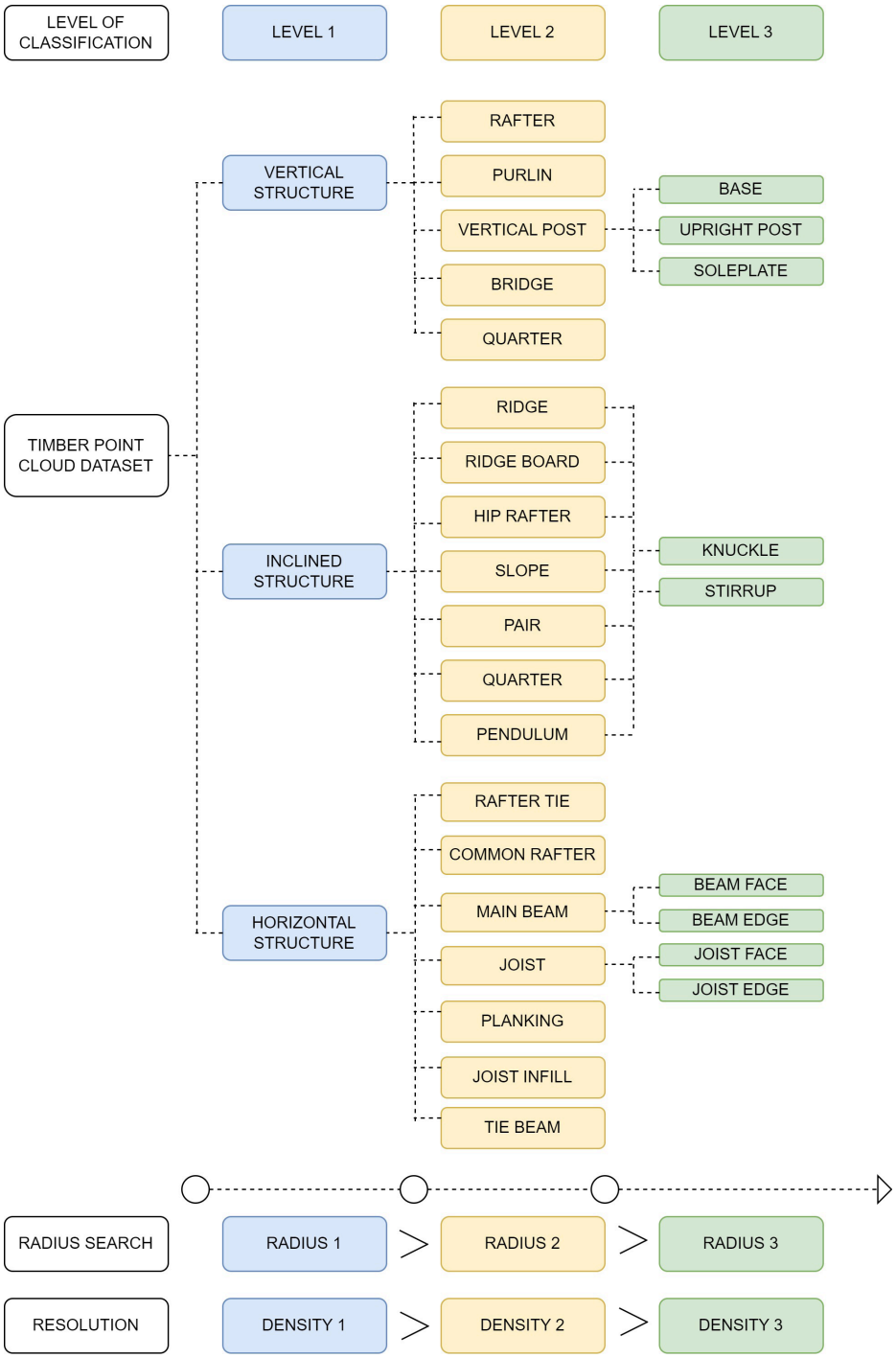
In the next step you will have to manually segment your point cloud by construction systems. In this step you can use the same tool «scissors» to segment each class. For an optimal segmentation you can follow the example of the following scheme, in which the classes are divided by levels and each level has a different point density and search radius (explained in the section «Feature extraction»). Also, the segmentation between exterior and interior is recommended.



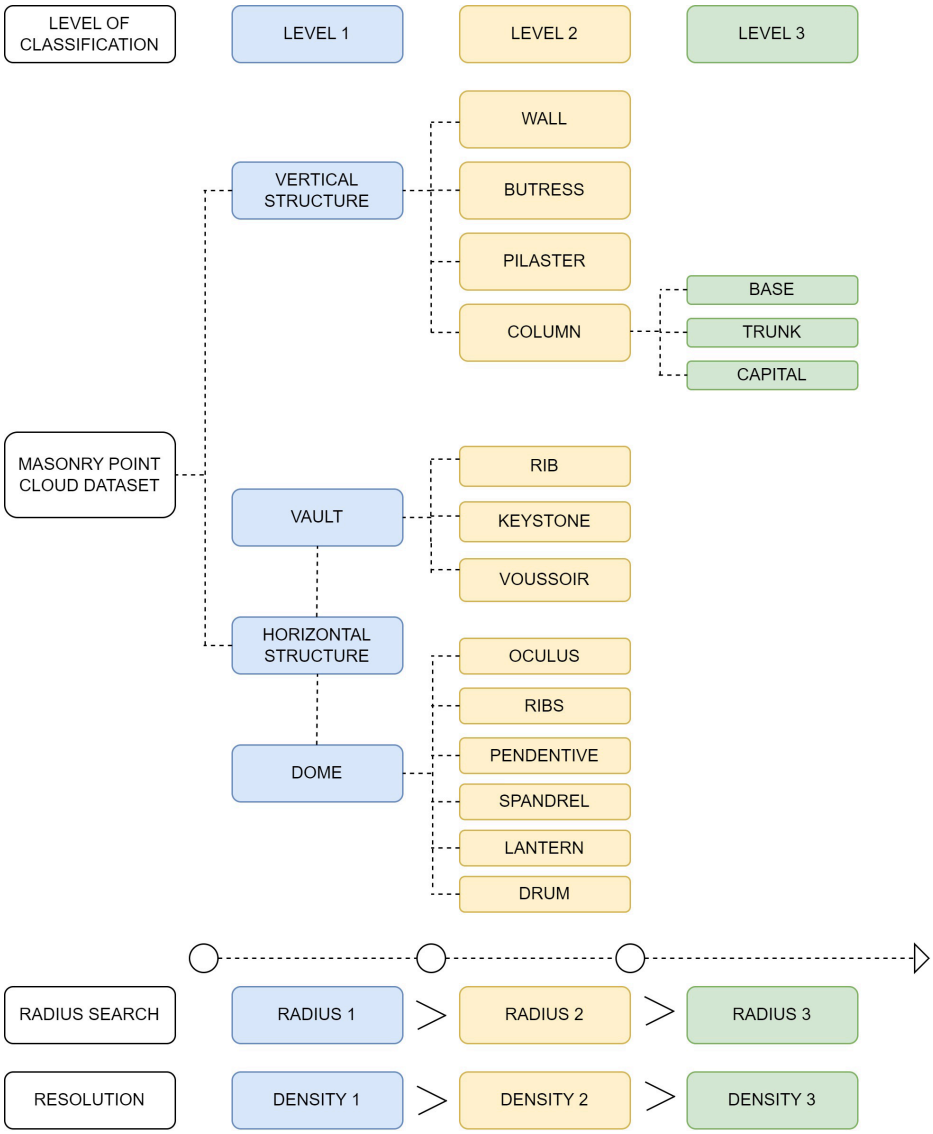
1.3. Metallic point cloud segmentation proposal



1.4. Concrete point cloud segmentation proposal



1.5. Timber point cloud segmentation proposal

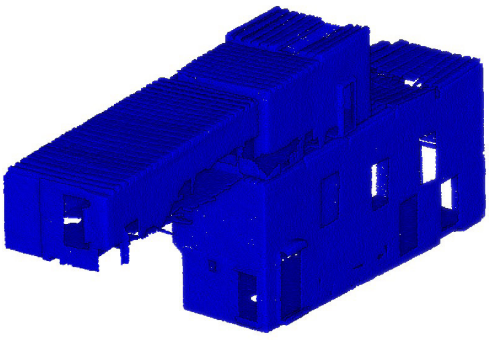


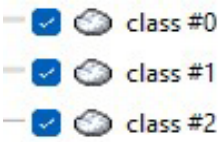
1.6. Mansory point cloud segmentation proposal

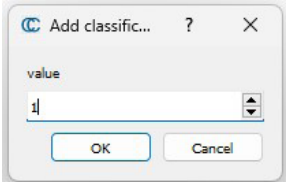
Once all classes have been segmented for each level, all construction systems will have to be grouped in a point cloud with a common Scalar Field called «Classification». To do this, add the classification number to each class and then join all point clouds of the same level. Then you can see to which class number each point belongs.

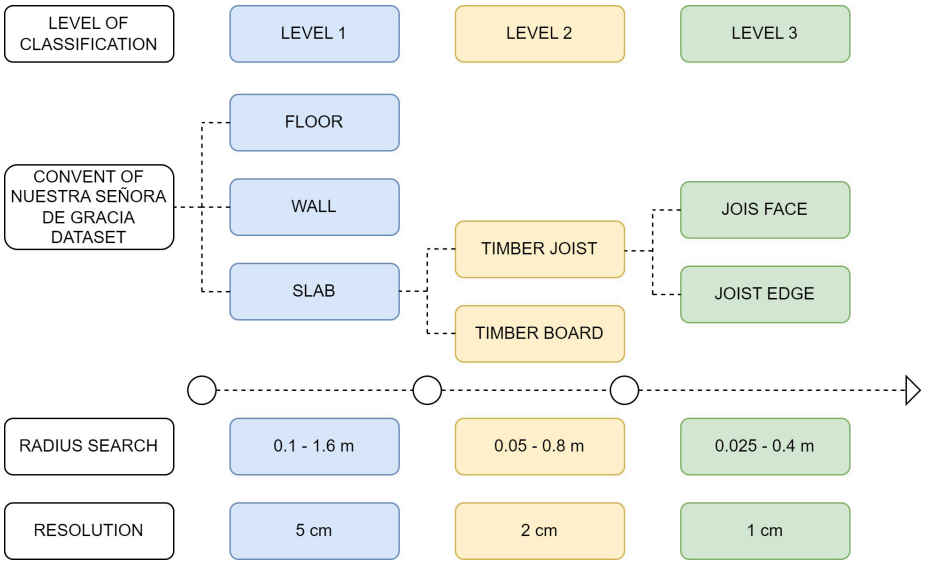
Workflow:

1. *Select each pc class*
   - *1.1 Edit*
   - *1.2. Scalar Fields*
   - *1.3. Add Classification*
2. *Select all pc classes*
3. *Merge point cloud*
   - *3.1. Don't generate S.F.*
4. *View class*

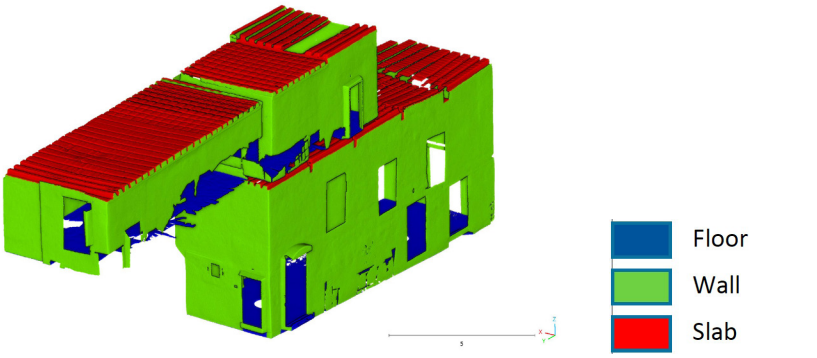




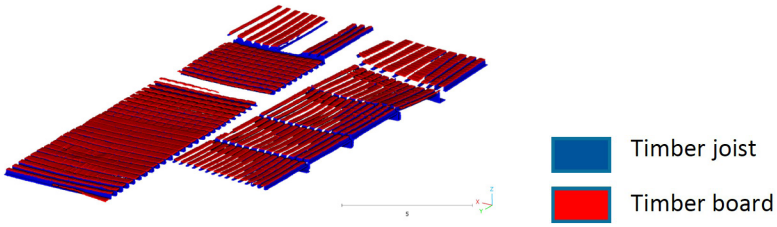




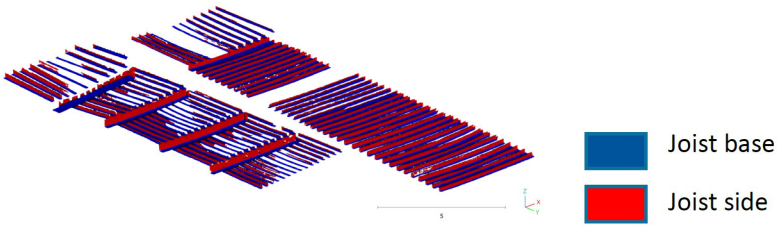
1.7. Segmentation proposal of the point cloud example



1.8. Level 1 segmentation proposal



1.9. Level 2 segmentation proposal



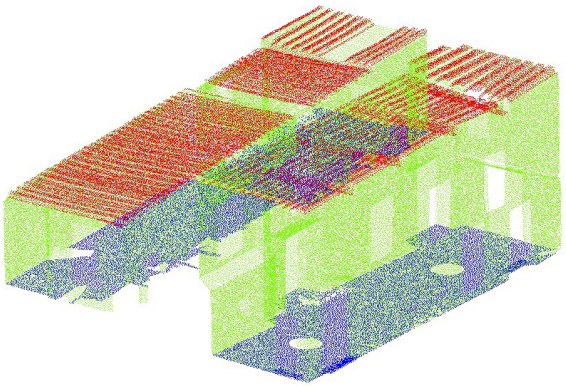
1.10. Level 3 segmentation proposal

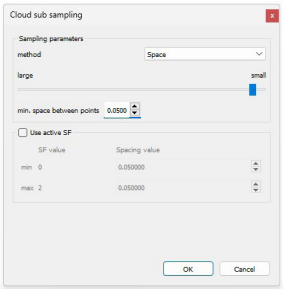
Subsampling

For subsampling you can use CloudCompare's «Subsample a point cloud» tool. You will have to do a subsample for each level you use. For the first level it is recommended to use a lower point density than the following levels, but not so low that you cannot distinguish each class. For example, you could use 5 cm for the first level, 2 cm for the second level and 1 cm for the third level, but this will work depending on the size of the building.

Workflow:

1. *Edit*
2. *Subsample*





1.11. Use of «Subsample point cloud» tool

Features Computation

Finally, you will need to compute the features of your point cloud, which will give it the geometric or statistical information necessary for the artificial intelligence to work properly. Features are computed on a local basis. To compute them you can already do it in the software, in the first section «Feature computation».

Geometrical features

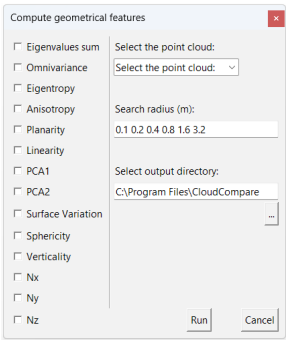
This tool allows you to compute several geometric features on one or several point clouds (in a batch). Geometric features that can be computed this way are: Sum of eigenvalues, Omnivariance, Eigenentropy, Anisotropy, Planarity, Linearity, PCA 1, PCA 2, Surface variation, Sphericity, Verticality, Nx, Ny and Nz.

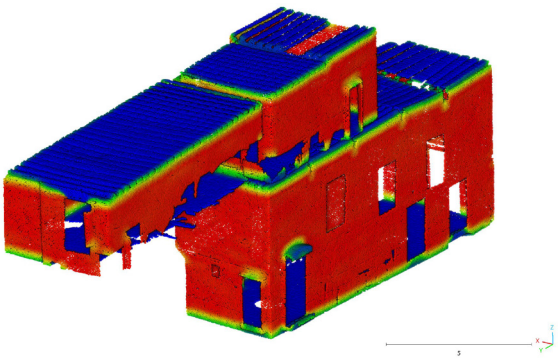
Firstly, choose the geometric feature(s) that should be computed (several features can be selected/checked at the same time).

Then set the 'Radius search' on which the selected features will be computed. In this step you can compute all the features selected with one or more search radius. If you want to compute with more radius search use "Spaces" to separate each radius search. Then choose an output directory and execute the algorithm with "Run".

Workflow:

1. *Select point cloud*
2. *Select geom. features*
3. *Select radius search*
4. *Select output*





1.12. Computing geometrical features to point cloud

Statistical features

This tool, as well as the geometrical features extraction, allows you to compute several textural features on one or several point clouds (in a batch). The behaviour of certain statistical measures provides information on the distribution and characteristics of the data, thus enabling the definition of

certain areas with insufficient content to be improved. Stadistical features that can be computed this way are:

- *Mean value:* Represents the trend by the sum of all neighbouring point values divided by the total number.

- *Standard deviation:* Measures the dispersion of the points around the mean from the average of the squared differences between each point and the mean of its neighbours.

- *Range:* Distance between the maximum value and the minimum value of the scalar.

- *Energy:* Statistical measure of randomness in the population of scalar fields.

- *Entropy:* It is also known as the angular second moment or uniformity.

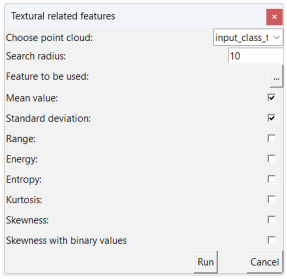
- *Kurtosis:* This variable Measures the stability of the distribution, which relates to the normal distribution of the scalar values.

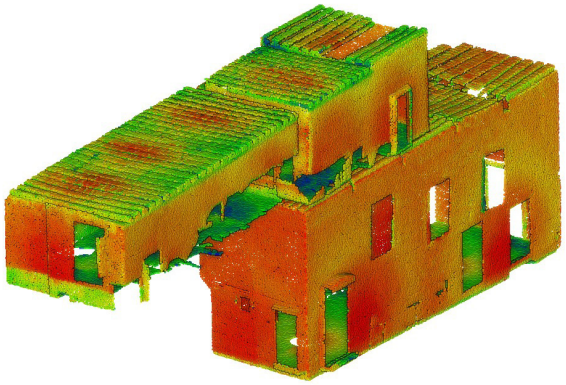
- *Skewness:* It is a measure of the degree of asymmetry in the distribution. Positive assimmetry indicates a distribution with a long right tail, while negative skewness indicates a long-left tail.

- *Skewness with binary values:* Is a feature which is «1» if the skewness is possitive and «0» if is negative.

Firstly, choose the textural feature(s) that should be computed (several features can be selected/checked at the same time).

Then set the 'Radius search' on which the selected features will be computed. In this step you can compute all the features selected with one or more search radius. If you want to compute with more radius search use "Spaces" to separate each radius search. Then choose an output directory and execute the algorithm with "Run".





Workflow:

1. Select point cloud

2. Select stad. features

3. Select radius search

4. Select output

1.13. Computing stadistical features from point cloud

Color Conversion

This function simply converts the channel R,G & B (firstly converted into Scalar Fields) into another cromatic combination. There are four options:

- *HSV*: Hue Saturation Value

- *YCbCr*: «Y» represents the luma component and the «Cb» and «Cr» signals are the blue difference and red difference chrominance components, respectively.

- *YIQ*: «Y» represents the luminance information, «I» and «Q» represent the chrominance information, orange-blue and purple-green range respectively.

- *YUV*: «Y» represents the luminance information, «U» and «V» represent the chrominance information, red and blue range respectively.

The new point cloud will be opened in the same folder.

Machine Learning

Supervised machine learning means that you will need a training model for it to work properly. In contrast, unsupervised does not require a training model. For supervised, it is advisable to divide the point cloud in such a way that 25% is for training and the remaining is for testing. Take both point clouds in the same folder.

Workflow:

1. Select point cloud

2. Viewing Tools

2.1. Set Top View

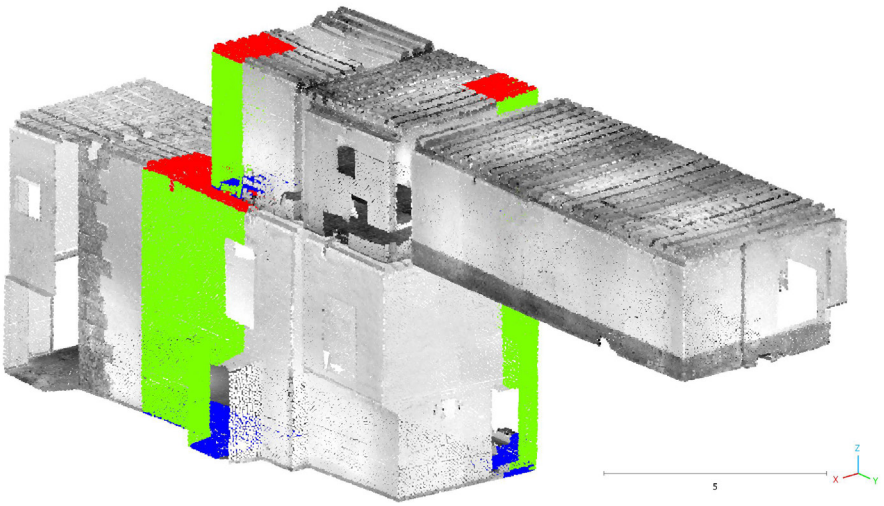
3. Edit

3.1. Segment

3.2. Inside

3.3.Offside

3.4. Cut



1.14. Point cloud division (training and testing)

Supervised Machine Learning

A) The main function of Feature selection tab is to select the best features for the next step, Classification. You must have the features computated in the Scalar Fields of the point cloud before applying the feature selection. You can compute the features in the “Features Computation” section.

In this tab, you can choose the point cloud, apply the selected selectors, choose the features that you want to work with, choose the number of features that you want to obtain, the number of folds for cross-validation and choose output directory.

- *Selectors*: Current version’s default available selectors are [‘kbest_f’,’rfe_lr’,’rfe_svm’,’rfe_tree’,’rfe_rf’,’rfecv_svm’,’rfecv_tree’,’rfecv_rf’]. (NOTE: SVM based selectors are highly sensitive to the number of features(high-dimension) and training records number, i.e.rfe_svm and rfecv_svm. When features number > 50 w/ records number over 50K, better exclude these 2 selectors, otherwise will result in long processing time.)

[1] https://optimal-flow.readthedocs.io/en/latest/autoFS.html#dynafs-clf.

- *Number of features to consider*: Set the number of features want to select out

- *Folds for cross-validation*: Number of folds for cross-validation.

The result is a Document File (.txt) with the names of the features selected.

B) In the classification tab, firstly you have to divide the point cloud into training (25%) and testing (75%). Make sure that all classes are present in both training and testing.

This tab works with different algorithms:

Random Forest

Parameters:

- *The number of trees in the forest*.

- *The function to measure the quality of a split:* Supported criteria are "gini" for the Gini impurity and "log_loss" and "entropy" both for the Shannon information gain, see Mathematical formulation. Note: This parameter is tree-specific.

-*The maximum depth of the tree:* If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

- *The minimum number of samples required to split an internal node:* If int, then consider min_samples_split as the minimum number. If float, then min_samples_split is a fraction and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

- *The minimum number of samples required to be at a leaf node:* A split point at any depth will only be considered if it leaves at least min_samples_leaf training samples in each of the left and right branches.

- *The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node:* Samples have equal weight when sample_weight is not provided.

-*The number of features to consider when looking for the best split:* Note: the search for a split does not stop until at least one valid partition of the node samples is found, even if it requires to effectively inspect more than max_features features.

- *Maximum number of leaf nodes:* Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.

- *Impurity to split the node:* A node will be split if this split induces a decrease of the impurity greater than or equal to this value.

- *Use of bootstrap*: Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

- *Complexity parameter used for Minimal Cost:* The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed.

[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Support Vector Machine

Parameters:

[3] https://scikit-learn.org/stable/modules/svm.html

- *Regularization parameter:* The strength of the regularization is inversely proportional to C. Must be strictly positive. The penalty is a squared l2 penalty.

- *Kernel type:* Specifies the kernel type to be used in the algorithm. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples).

- *Degree of the polynomial kernel function:* Must be non-negative. Ignored by all other kernels.

- *Kernel coefficient:* If gamma='scale' (default) is passed then it uses 1 / (n_features * X.var()) as value of gamma, if 'auto', uses 1 / n_features and if float, must be non-negative.

- *Independent term in kernel function:* It is only significant in 'poly' and 'sigmoid'.

- *Use shrinking heuristics:* If the number of iterations is large, then shrinking can shorten the training time.

- *Enable probability estimates:* This must be enabled prior to calling fit, will slow down that method as it internally uses 5-fold cross-validation, and predict_proba may be inconsistent with predict.

- *Tolerance for stopping criterion.*

- *Class weight:* Set the parameter C of class i to class_weight[i]*C for SVC. If not given, all classes are supposed to have weight one.

- *Max number of iterations:* Hard limit on iterations within solver, or -1 for no limit.

- *Returned function:* Whether to return a one-vs-rest ('ovr') decision function of shape (n_samples, n_classes) as all other classifiers, or the original one-vs-one ('ovo') decision function of libsvm which has shape (n_samples, n_classes * (n_classes - 1) / 2).

- *Break ties:* If true, decision_function_shape='ovr', and number of classes > 2, predict will break ties according to the confidence values of decision_function; otherwise the first class among the tied classes is returned.

Logistic Regression

[4] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Parameters:

- *Penalty function*: None: no penalty is added; 'l2': add a L2 penalty term and it is the default choice; 'l1': add a L1 penalty term; 'elasticnet': both L1 and L2 penalty terms are added.

- *Constrained problem (dual):* Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features.

- *Tolerance for stopping criterion.*

- *Inverse of regularization strength:* Must be a positive number. Like in support vector machines, smaller values specify stronger regularization.

- *Add a bias to the model:* Specifies if a constant should be added to the decision function.

- *Intercept scaling:* In this case, x becomes [x, self.intercept_scaling], i.e. a "synthetic" feature with constant value equal to intercept_scaling is appended to the instance vector.

- *Class weight:* If not given, all classes are supposed to have weight one.

- *Type of solver:* Algorithm to use in the optimization problem. Default is 'lbfgs'. To choose a solver, you might want to consider the following aspects: For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones; For multiclass problems, only 'newton-cg', 'sag', 'saga' and 'lbfgs' handle multinomial loss; 'liblinear' is limited to one-versus-rest schemes; 'newton-cholesky' is a good choice for n_samples >> n_features, especially with one-hot encoded categorical features with rare categories.

- *Max number of iterations:* Maximum number of iterations taken for the solvers to converge.

- *Type of multiclass fitting strategy:* If the option chosen is 'ovr', then a binary problem is fit for each label. For 'multinomial' the loss minimised is the multinomial loss fit across the entire probability distribution, even when the data is binary. 'multinomial' is unavailable when solver='liblinear'. 'auto' selects 'ovr' if the data is binary, or if solver='liblinear', and otherwise selects 'multinomial'.

- *Elastic-Net mixing parameter:* With 0 <= l1_ratio <= 1. Only used if penalty='elasticnet'. Setting l1_ratio=0 is equivalent to using penalty='l2', while setting l1_ratio=1 is equivalent to using penalty='l1'. For 0 < l1_ratio <1, the penalty is a combination of L1 and L2.

- *Number of cores to use:* Number of CPU cores used when parallelizing over classes if multi_class='ovr'". This parameter is ignored when the solver is set to 'liblinear' regardless of whether 'multi_class' is specified or not. -1 means using all processors.

Auto Machine Learning

It helps you out with the most tedious part of machine learning by intelligently exploring thousands of possible pipelines to find the best one for your data.

Parameters:

- *Number of generations:* Number of iterations to the run pipeline optimization process. It must be a positive number or None. If None, the parameter "Maximum total time" must be defined as the runtime limit. Generally, it will work better when you give it more generations (and therefore time) to optimize the pipeline.

-*Population size:* Number of individuals to retain in the genetic programming population every generation. Must be a positive number. Generally, it will work better when you give it more individuals with which to optimize the pipeline.

- *Mutation rate:* Mutation rate for the genetic programming algorithm in the range [0.0, 1.0]. This parameter tells the GP algorithm how many pipelines to apply random changes to every generation. Mutation rate + crossover rate cannot exceed 1.0. We recommend using the default parameter unless you understand how the mutation rate affects GP algorithms.

[5] https://epistasislab.github.io/tpot/

- *Crossover rate:* This parameter tells the genetic programming algorithm how many pipelines to «breed» every generation.

- *Function used to evaluate the quality:* See the section on scoring functions for more details.

- *Cross-validation*: Cross-validation strategy used when evaluating pipelines.

- *Maximum total time*: How many minutes TPOT has to optimize the pipeline. If not None, this setting will allow TPOT to run until "Maximum total time" minutes elapsed and then stop. It will stop earlier if generations is set and all generations are already evaluated.

- *Maximum time per evaluation*: How many minutes TPOT has to evaluate a single pipeline. Setting this parameter to higher values will allow TPOT to evaluate more complex pipelines, but will also allow TPOT to run longer. Use this parameter to help prevent TPOT from wasting time on evaluating time-consuming pipelines.

- *Number of generations without improvement*: Ends the optimization process if there is no improvement in the given number of generations.

C) Prediction tab consists of predicting the classification with the previous training file. In the previous step, it launched a PKL file with the training configuration, which is necessary for this step. This algorithm works with Scikit-Learn library.

Is mandatory to send in the buttons the required information: firstly, you have to choose the point cloud to predict, then select the feature and the PKL file launched in the classification step. Then choose an output directory and execute the algorithm with "Run".

Unsupervised Machine Learning

A) Training tab works with different algorithms:

K-Means

Parameters:

- *Number of clusters:* The number of clusters to form as well as the number of centroids to generate.

- *Number of iterations:* Maximum number of iterations of the k-means algorithm for a single run.

- *Optimization strategy*: Due to the fact that the optimal number of clusters is not known in advance, by selecting this option, it recommends the optimal number of clusters. The available options are: elbow method and silhouete method, Calinski Harabasz, Davies Bouldin.

Elbow method: The KElbowVisualizer implements the "elbow" method to help data scientists select the optimal number of clusters by fitting the model with a range of values for "K". https://www.scikit-yb.org/en/latest/api/cluster/elbow.html

Silhouete Coefficient: Is used when the ground-truth about the dataset is unknown and computes the density of clusters computed by the model. https://www.scikit-yb.org/en/latest/api/cluster/silhouette.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Calinski Harabasz index: It is also known as the Variance Ratio Criterion. The score is defined as ratio of the sum of between-cluster dispersion and of within-cluster dispersion. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html#sklearn.metrics.calinski_harabasz_score

Davies Bouldin index: The score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html

- *Maximum number of clusters:* Depending on the choice of the optimization strategy, set a maximum number of clusters.

- *Minimum number of clusters:* Depending on the choice of the optimization strategy, set a minimum number of clusters.

Fuzzy-K-Means

Parameters:

The parameters are the same as K-Means (previous).

[7] https://fda.readthedocs.io/en/latest/modules/ml/autosummary/skfda.ml.clustering.FuzzyCMeans.html

Hierarchical Clustering

Parameters:

- *Number of clusters:* The number of clusters to find. It must be None if distance threshold is not None.

- *Metric for calculating the distance between instances:* Can be "euclidean", "l1", "l2", "manhattan", "cosine", or "precomputed". If linkage is "ward", only "euclidean" is accepted. If "precomputed", a distance matrix is needed as input for the fit method.

- *Linkage criterion:* The linkage criterion determines which distance to use between sets of observation. The algorithm will merge the pairs of cluster that minimize this criterion.

'ward' minimizes the variance of the clusters being merged.

'average' uses the average of the distances of each observation of the two sets.

'complete' or 'maximum' linkage uses the maximum distances between all observations of the two sets.

'single' uses the minimum of the distances between all observations of the two sets.

- *Linkage distance threshold*: The linkage distance threshold at or above which clusters will not be merged. If not None, number of clusters must be None and compute full tree must be True.

- *Computes distances between clusters:* Even if distance threshold is not used. This can be used to make dendrogram visualization, but introduces a computational and memory overhead.

[8] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

DBSCAN

Parameters:

[9] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

- *Epsilon*: The maximum distance between two samples for one to be considered as in the neighborhood of the other. This is not a maximum bound on the distances of points within a cluster.

- *Minimum number of points to create a cluster*: The number of samples (or total weight) in a neighborhood for a point to be considered as a core point. This includes the point itself. If min samples is set to a higher value, DBSCAN will find denser clusters, whereas if it is set to a lower value, the found clusters will be sparser.

OPTICS

[10] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.OPTICS.html

Parameters:

- *The number of samples in a neighborhood to be considered as cluster:* Also, up and down steep regions can't have more than min_samples consecutive non-steep points.

- *Epsilon:* The maximum distance between two samples for one to be considered as in the neighborhood of the other.

- *Metric for distance computation*: Any metric from scikit-learn or scipy. spatial.distance can be used. If metric is a callable function, it is called on each pair of instances (rows) and the resulting value recorded. See the documentation for scipy.spatial.distance for details on these metrics.

- *Extraction method:* Used to extract clusters using the calculated reachability and ordering. Possible values are "xi" and "dbscan".

- *Minimum steepness:* Determines the minimum steepness on the reachability plot that constitutes a cluster boundary. For example, an upwards point in the reachability plot is defined by the ratio from one point to its successor being at most 1-xi. Used only when extraction method='xi'.

- *Minimum cluster size:* Minimum number of samples in an OPTICS cluster, expressed as an absolute number or a fraction of the number of samples (rounded to be at least 2). If None, the value of min_samples are used instead. Used only when cluster_method='xi'.

B) Prediction tab consists of predicting the classification with the previous training file. In the previous step, it launched a PKL file with the training configuration, which is necessary for this step. This algorithm works with Scikit-Learn library.

It is mandatory to send in the buttons the required information:

Firstly, you have to choose the point cloud to predict, then select the feature and the PKL file launched in the classification step. Then choose an output directory and execute the algorithm with "Run".

Results

For all the Machine Learning algorithms used, different graphs representing the values of the results obtained will be generated in the Output directory.

-*Importance* (only in Random Forest algorithm): This graph represents the importance of the features by scale, the higher a feature is, the more important it has been for the classification.

- *Classification matrix*: This matrix represents the level of accuracy and precision of the algorithm training. The above part represents it by levels of classification, and under it, the global values in percentage. Parameters:

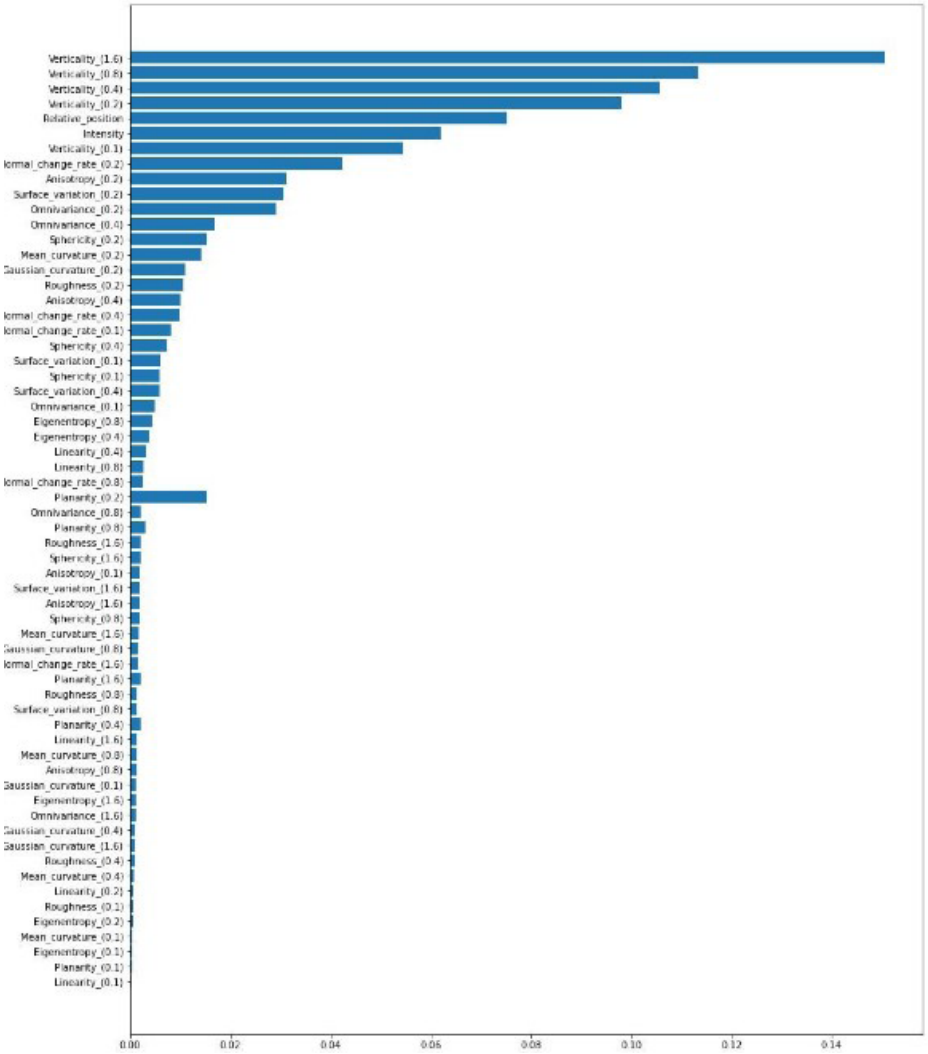
Accuracy: Accuracy is a measure of the overall correctness of a model. It calculates the ratio of correctly predicted instances to the total instances.

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. It measures the accuracy of positive predictions.

Recall: Recall is the ratio of correctly predicted positive observations to all observations in actual class. It measures the ability of the model to capture all the positive instances.

F1 Score: The F1 Score is the harmonic means of precision and recall. It provides a balance between precision and recall.

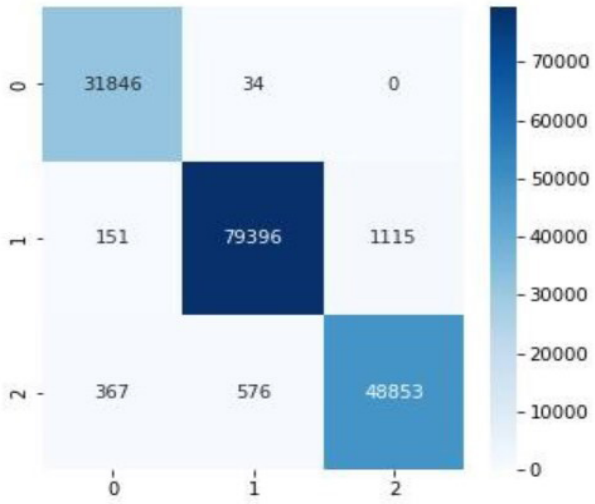
- *Confusion matrix:* This matrix represents the number of points of a level that has been confused with another level.



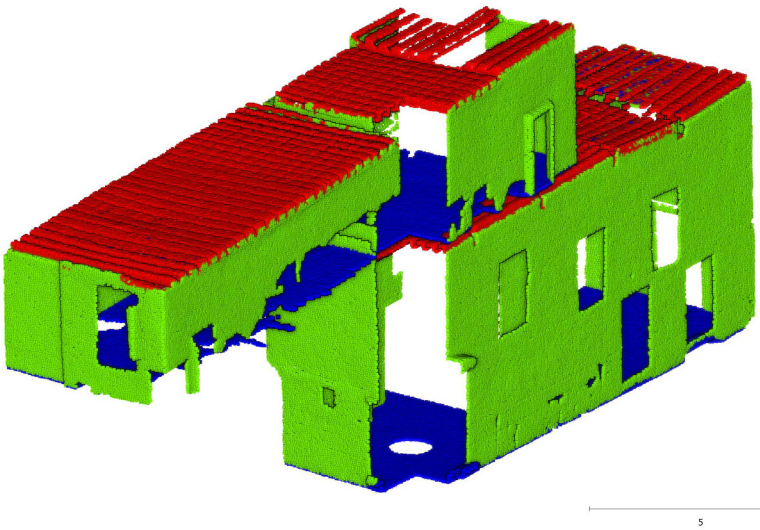
1.15. Importance

	precision	recall	f1-score	support
0.0	0.984	0.999	0.991	31880
1.0	0.992	0.984	0.988	80662
2.0	0.978	0.981	0.979	49796
accuracy			0.986	162338
macro avg	0.985	0.988	0.986	162338
weighted avg	0.986	0.986	0.986	162338

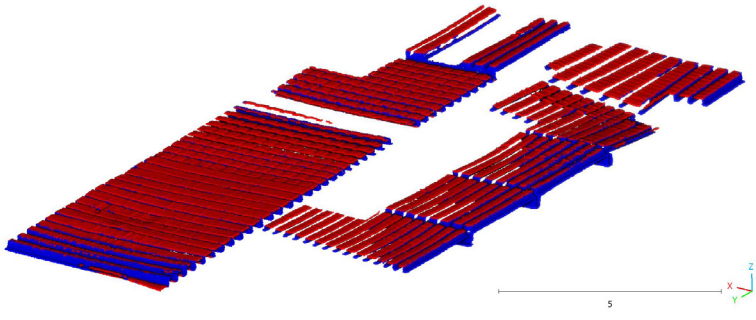
1.16. Classification matrix



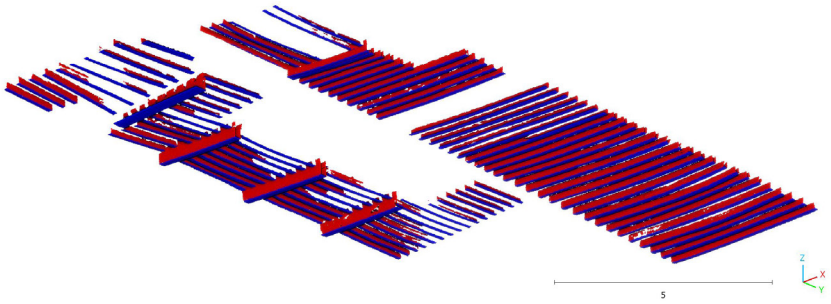
1.17. Confusion matrix



1.18. Classification result of the first level of classification



1.19. Classification result of the second level of classification



1.20. Classification result of the third level of classification

Deep Learning

Deep Learning works with Point Transformer algorithm.

IMPORTANT: **The first numbered class in the point cloud must start with 0, the next 1 and so on for the rest of the classes (see bottom of page 5).**

A) The Training tab consists of, as well as Machine Learning, with a training and testing point cloud. First you have to choose them correctly (it is recommended to take 25% of the point cloud for training and 75% of the rest for testing). Then, as well as Machine Learning, choose the features you want to add to the training. GPU consumption means how much power from your computer you want to use. Choose the number of iterations to the run pipeline optimization process (it must be a positive number or None). Then choose an output directory and execute the algorithm with "Run".

B) The Classification tab is the next step after the Training. In the previous step, it launched a PKL file with the training configuration, which is necessary for this step. Firstly, you have to choose the point cloud to classify, then select the feature and the PKL file launched in the classification step. Then choose a GPU consumption and an output directory and execute the algorithm with "Run".

The results will be in the output folder.

BIM Integration

This tool will be coming soon.