A introduction to Data Mining and its applications with Point Clouds

Matthias Weil
School of Computation, Information and Technology, Informatics
Techincal University Munich
Munich, Germany
matthias.weil@tum.de

***Abstract*— In this paper, we focus on the necessary steps in Data Mining and how this process is applied with Point Clouds in Geospatial Data. We will shine light on several steps of a Data Mining Pipeline, including Data Cleaning, Feature Enginnering followed by model selection and important aspects that have to be considered when training the selected model.**

***Index terms*—Data Mining, Geospatial Data, PointClouds, 3D, Feature Extraction, Classification, Co-Registration**

I. Introduction

Data Mining and extracting data from Point Clouds are necessary skills when working with Geospatial Data sets. Many methods and how they can be practically applied are explained.

II. Data Mining

A. Data Cleaning

It is not uncommon for the data obtained to be of poor quality and difficult to work with. This may include the presence of missing values, outliers, or spelling mistakes, which are particularly prevalent in user-generated data.

B. Feature Extraction

Machine learning algorithms require the given data to be in certain numerical formats. In this step, it is necessary to transform the data into the needed format to allow further processing. Additionally, this allows the machine learning algorithm to receive a more effective set of inputs, which increases accuracy and lowers the computational need for machine learning algorithms.

1) *Over- and Underfitting:*

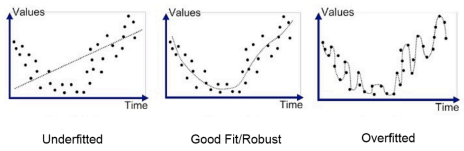

Figure 1: A common problem when working with a machine learning algorithm is to create a model that discovers patterns in unknown data. On the left is Underfitting, when the model is too simple to capture the complexity of the data. In the middle is the desired outcome of a data model. On the right is the example of Overfitting which occurs when a machine learning algorithm detects noise as data. Figure from [1].

C. Feature Selection

Feature selection is a critical part of data mining due to numerous reasons. As only relevant features are being considered, the dimensionality of the data set is reduced and is more efficient, accurate, and less prone to overfitting. Furthermore less features lead to a better understanding of the learning result. The following figure explains the Hughes phenomenon, where a higher amount of features deteriorates the model's performance. Additional features may introduce noise, which leads to a decrease in classification accuracy [2].

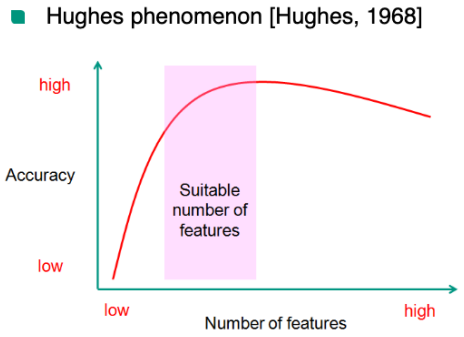

Figure 2: Hughes phenomenon, a increase of number of features decreases the accuracy of the classification model. Figure from [3].

There are three notable categories of feature selection. Filter-based methods include statistical tests to assess the correlation between various features. These filters for example remove features with low variance, since they are deemed to have lit-

tle information. Because they are classifier-independent, simple, and efficient they are often used. Alternatively, Wrapper-based methods or Embedded methods can be used [4].

D. Model Selection

Two overarching themes emerge when selecting a model: supervised and unsupervised learning models. Supervised machine learning includes classification which is heavily used in image recognition. Examples of classification models that categorize data into labels, are Decision Trees, Support Vector Machines, and k-nearest Neighbors. Supervised learning also includes regression models which describe functions that model the correlation of an independent variable and a target variable. The correlation between systolic blood pressure and the amount of coffee a person drinks is an example of a linear regression.

In unsupervised classification, the algorithm tries to assign each data point to a cluster. The number of clusters to be extracted depends on the algorithm. In the case of kMeans, the user must input the desired number of clusters. Alternatively, in the case of DBSCAN, the user must define the distance between two points to form a new cluster. These algorithms work without training labels, so the user has to understand the output clusters.

Deep learning as the machine learning algorithm called representation learning harvests the benefit, of the algorithm which learns the features from the data itself. The algorithm is then capable of representing the data in a way that eases classification or regression [5].

E. Model Training

Models are prone to be overfitted when too many features are selected, sufficient data not being available to train the model, or if the data is noisy. This noise may easily be detected as a pattern by the model. To detect overfitting the data set is split into a training and a validation set. After training, the model is presented with the validation set, and the performance is tracked. If they diverge, then it can be concluded that the model is overfitting. To tackle underfitting additional features can be selected, and the complexity of the model can be raised or an increase of the duration of training the model [6].

F. Model Evaluation

To evaluate the model various methodologies may be applied. For regression, Mean Absolute Error or Mean Squared Error is often applied. To evaluate a classification model one can calculate the Accuracy which is the proportion of correctly classified instances. Furthermore, the Precision, which measures the quality of positive predictions, Recall, the ability of the classifier to find all the positive samples, and the F1-score, which is the Harmonic mean of precision and recall, illustrate various metrics to evaluate a classifier [6], [7].

III. Geospatial Data Mining

In the following, we showcase a typical workflow for classification that involves first having a point cloud that can be expanded by co-registration. Then neighborhood selection, followed by feature extraction and feature selection. Lastly, we choose our classification model.

A. Co-Registration

When working with point cloud data there may be two acquisitions from the same structure, therefore it is beneficial to apply Co-registration to combine the multiple point clouds of interest. Typical algorithms include iterative closest point algorithm or feature based matching algorithms.

Figure 3: Co-registration of two point clouds, showing the Arc de Triomphe from different angles. Here a feature based matching algorithm, along the corners of the building, might be more computationally efficient. Figure from [8].

As can be seen in Figure 3, with two point clouds of the Arc de Triomphe in Paris, the co-registered point cloud has more information about the subject of interest. Often co-registration is conducted to detect changes in the area, for example after natural disasters or after longer periods to show change in the scanned area.

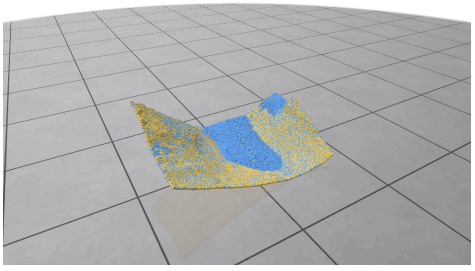

Figure 4: Two point cloud scans of the hellstugubrean glacier visualized using python.The point cloud from 2009 showing the intensity is colored in blue and the same glacier from 2017 colored in yellow. Figure by author.

Figure 4 illustrates how Co-registration is a vital procedure when working with point-cloud data. [5] notes that the difference in height between the two point clouds, which can be seen most clearly in the center of the image, is due to a "significan[t] loss of mass from the glacier over these 8 years".

B. Neighborhood Selection

Calculations based on neighborhoods are among the most important ones, to gain additional information. They are necessary for any filtering, smoothing, or interpolation step and information extraction.

Spatial neighborhoods are defined either by a certain distance to other points or to a certain amount of fixed neighbors [3]. If a fixed distance is given, and the height can be neglected, the neighborhood is commonly referred to as a cylindrical neighborhood. Otherwise, if it's 3D then they are called spherical neighborhoods because their height is also relevant. A point cloud neighborhood can also be defined by a fixed amount of neighbors. Fixed neighbors are also commonly called k-nearest neighbors and the size can be highly variable depending on the point density at that area [5].

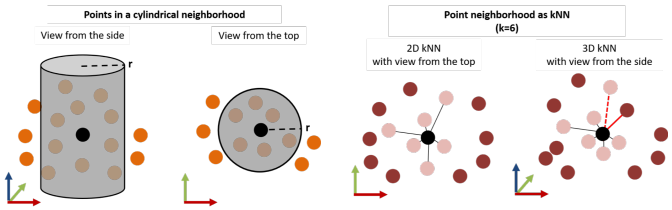

Figure 5: Spatial and k-nearest neighbor neighborhoods shown side by side. Figures by [5].

C. *Feature Extraction*

After neighborhoods are defined, various features can be extracted. 3D-features can be calculated by using eigenvalues or looking at the geometric properties. Spatial neighborhoods allow the derivation of local surface roughness.

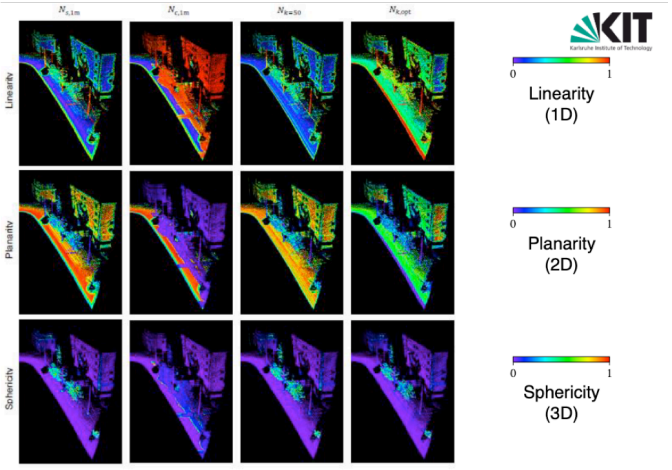

Figure 6: Features that can be easily derived from eigenwerte. Figure and more details in [9].

The following code provided by [5] demonstrates how a feature vector can be created, focusing on extracting features that can be calculated by using eigenwerte. They include planarity, linearity, omnivariance, roughness, and slope. These features are calculated using eigenvalues derived from the covariance matrix of neighboring points. There are a multitude of other features that can be extracted [4].

```python
def create_feature_vector(neighbour_points):
    # structure tensor
    struct_tensor = np.cov(neighbour_points.T)
    # eigenvalue decomposition
    eigvals, eigvec = np.linalg.eigh(struct_tensor)
    l3, l2, l1 = eigvals

    # find eigenvector to smallest eigenvalue = normal vector to best fitting plane
    normalvector = eigvec[:, 0]
    # flip so that it always points "upwards"
    if normalvector[2] < 0:
        normalvector *= -1

    # feature calculation
    planarity = (l2-l3)/l1
    linearity = (l1-l2)/l1
    omnivariance = (l1*l2*l3)**(1./3)
    roughness = l3
    slope = np.arctan2(np.linalg.norm(normalvector[:2]), normalvector[2])

    return np.array([planarity,
                     linearity,
                     omnivariance,
                     roughness,
                     slope])
```

1) *Ground Removal:* Ground removal is an important procedure when working with point cloud data. Depending on the specific situation, it might be beneficial or even necessary to remove ground points. Sometimes it is cost-effective to remove them to reduce computing power and time needed to process the data. Notable algorithms for this purpose are RANSAC (Random Sample Consensus), Ground Plane Fitting, and Patchwork++.

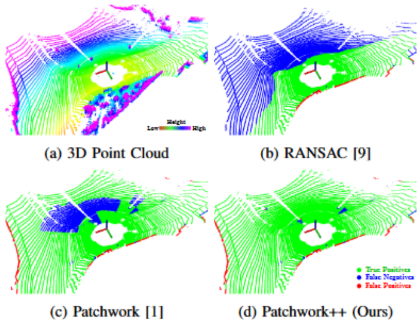

Figure 7: Showcasing different ground removal algorithms and Patchwork++ being shown as least prone to under-segmentation (blue area). *Green*: True Positive *Blue*: False Negatives *Red*: False Positives. Figure from [10].

D. Feature Selection

For a detailed explanation of feature selection, please refer back to Section II.C. Next, we show an example of a filter based method to select valuable features. The following code snippet, taken from [6] is designed to remove features "that are either one or zero in more than 80% of the samples.", according to scikit-learn documentation. As they are boolean features the threshold is given by

$$\mathrm{Var}[X] = p(1-p) \tag{1}$$

so the correct threshold for the function is $.8 * (1 - .8)$

```python
from sklearn.feature_selection import VarianceThreshold

X = [[0, 0, 1], [0, 1, 0], [1, 0, 0], [0, 1, 1], [0, 1, 0], [0, 1, 1]]

sel = VarianceThreshold(threshold=(.8 * (1 - .8)))

sel.fit_transform(X)
```

E. Classification

The extracted features can then be used as input for classifiers that have been trained with representable data. Noteworthy classifiers include Random Forest, Support Vector Machines, Nearest Neighbor classifiers, Decision Trees, Naïve bayesian classifiers and Linear Discriminant Analysis.

References

[1] "over and Underfitting." [Online]. Available: https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76

[2] "On the mean accuracy of statistical pattern recog- nizers.," 1968.

[3] M. Weinmann, B. Jutzi, S. Hinz, and C. Mallet, "Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 105, pp. 286–304, 2015, doi: https://doi.org/10.1016/j.isprsjprs.2015.01.016.

[4] J. B. Blomley R. and M. Weinmann, "Classification of air- borne laser scanning data using geometric multi-scale features and dif- ferent neighbourhood types.," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2016, [Online]. Available: https://isprs-annals.copernicus.org/articles/III-3/169/2016/isprs-annals-III-3-169-2016.pdf

[5] "E-learning course on Time Series Analysis in Remote Sensing for Understanding Human-Environment Interactions (E-TRAINEE)." [Online]. Available: https://3dgeo-heidelberg.github.io/etrainee/index.html

[6] "Scikit-learn." [Online]. Available: https://scikit-learn.org/

[7] C. D. Manning, P. Raghavan, and H. Schütze, *Information retrieval*. 2009. [Online]. Available: https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf

[8] "point-cloud-registration." [Online]. Available: https://www.thinkautonomous.ai/blog/point-cloud-registration/

[9] M. Weinmann, "Semantic Segmentation of Dense Point Clouds [PowerPoint slides]." [Online]. Available: http://www.eurosdr.net/sites/default/files/images/inline/05-2019-12-04_presentation_pcp_workshop_weinmann.pdf

[10] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and Robust Ground Segmentation Solving Partial Under-Segmentation Using 3D Point Cloud." 2022.