Detecting Alpine Treeline Ecotones – an Automated Remote Sensing Approach

Thesis Paper for the Project Seminar: Treelines of the World in SS 2020

University Lecturer: Dr. Maaike Bader, Physical Geography, Philipps-Universität Marburg

Student: Agnes Schneider

Matriculation number: 2689766

E-Mail: Schnei7v@students.uni-marburg.de; euboia@gmail.com

Date of Submission: 01.01.2022

Abstract

1 Introduction

Alpine Treeline Ecotones (ATEs) are transitional zones between subalpine Forest and Alpine (tundra) Ecotones (Holtmeier & Broll (2005), Winings (2013)) also referred to as upper-treeline (Elliott (2017)) and occur globally (Singh et al. (2015), Bader et al. (2021)). They span between the actual Timberline/Economic Forest Line though the Upper/Physiognomic-Biologic Forest Line and the tree species line which is adjoining the actual Alpine zone (Chhetri & Thai (2019), 1543). The position of the treeline (which is not the same as the tree species line) is influenced by multiple factors at local and regional level, but temperature has been identified as the global driving factor (Körner (1998), Körner & Paulsen (2004), Holtmeier & Broll (2005), Bader et al. (2007), Barredo Cano et al. (2020)). The global pattern can be described by spatial patterns in the x-y plane (discrete, diffuse or their variation, Figure 1) and by changes in tree stature (abrupt, gradual, or their variation, Figure 2) in a multi-dimensional space (Harsch & Bader (2011), Bader et al. (2021)).

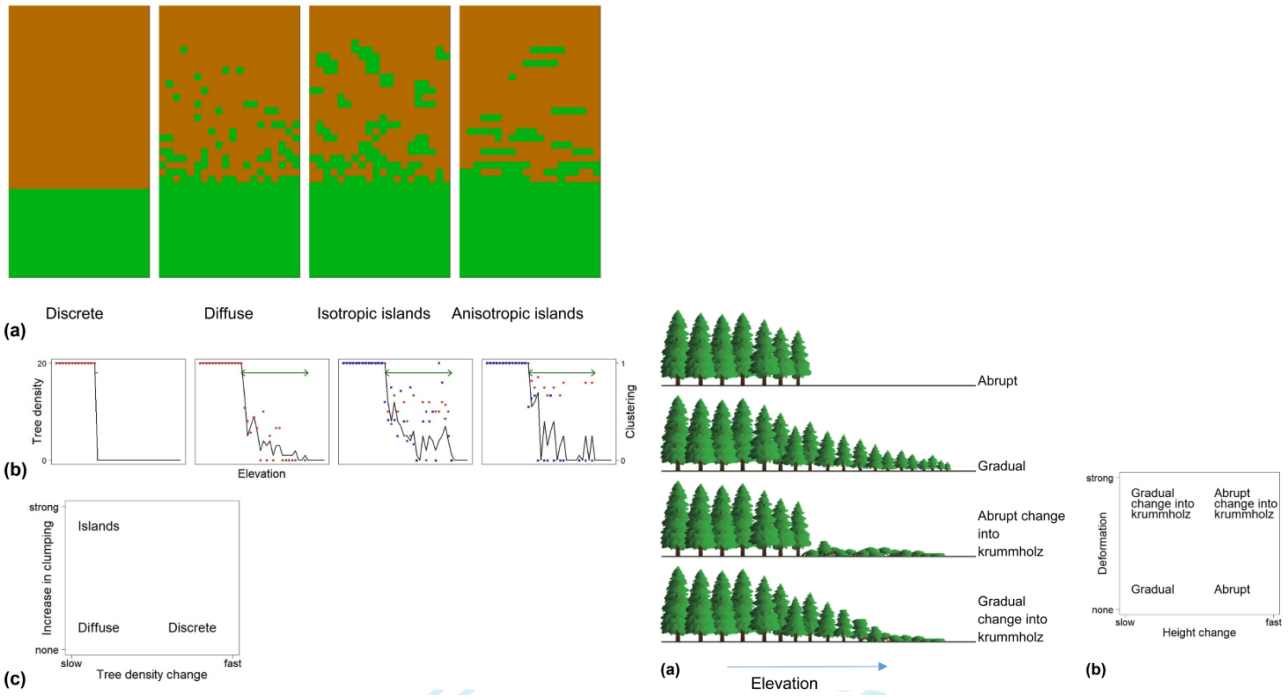

Figure 1: Left: Scheme of the spatial pattern of alpine treelines on the 2D x-y plane. a) Depicts the treeline as seen from above, while b) depicts the profile of the change of the treeline (clustering of islands). c) Represents an abstraction of the pattern of treelines based on tree density change and the clustering of individual trees. Source: Bader et al. (2021), Figure 1. Right: Scheme of (discrete) tree stature/height change responding to change in elevation. a) Vertical cross section. b) Abstraction of tree stature change based on height change and deformation of tree shape. Source: Bader et al. (2021), Figure 2.

Although recognized, the distribution of ATE patterns have neither been mapped, nor been described yet, let alone explained. Earlier studies have identified abrupt, diffuse, island and krummholz spatial patterns of ATEs (Harsch et al. (2009), Harsch & Bader (2011), Figure 2). As seen, treelines dispaly a high variability and differ in multiple dimensions. A comparison of multiple studies suggest, that the different spatial patterns of ATEs reflect fundamental ecological controlling processes and that different ATEs react differently to climate change (Harsch & Bader (2011), Figure 1) Figure 3.

	a) Rate of change	b) Type of change	Controlling processes (first-level)	Controlling processes (second- and third-level)
2-D pattern	**Discrete**		Mortality above but not in the ecotone	Any type of damage or stress. If gradual, growth limitation above the ecotone
	Ecotone	**Diffuse**	Mortality within the ecotone	Environmental heterogeneity or stochasticity. And/or ongoing seed-limited colonization
		Island	Mortality within the ecotone	Positive feedback due to radiation damage, freezing damage, wind damage, snow redistribution (snow moulds, growing-season length), clustered seed dispersal, clonal reproduction
Tree stature	**Abrupt**	Into alpine	Mortality	Overheating, desiccation, photoinhibition, freezing damage, soil limitations
		Into krummholz	Dieback	Winter desiccation, snow moulds, browsing, abrasion
	Gradual	No krummholz	Growth	Tissue formation (limited by a lack of heat)
		Into krummholz	Growth and dieback	Tissue formation + winter desiccation, snow moulds, browsing, abrasion

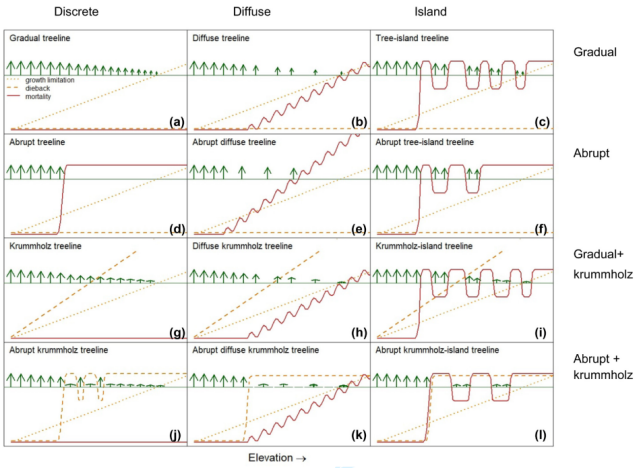

Figure 2: Left: Matrix of 2D spatial pattern, stature change and ecological processes which can contribute to the different types of ATEs. Source: Bader et al. (2021), Table 1. Right: Matrix of the multidimensional state space of treeline forms depicting extreme cases of the different dimensions. Columns represent the spatial patterns in the x-y plane and rows the change in tree stature (size and shape). The lines represents the hypothesized first-level ecological processes behind the patterns along an elevational gradient. The dotted line displays the growth limitation, the dashed line the dieback and the continuous line the mortality. Growth limitation always occurs while dieback only affects if krummholz is involved. Source: Bader et al. (2021), Figure 3.

To better understand and categorize ATEs a standardized description and terminology of spatial patterns has been proposed on hillslope and landscape scale by Bader et al. (2021), including hypotheses for the general mechanisms behind the patterns. The terminology and the multidimensional state-space can be most clearly understood from Figure 2, Right.

After the definition and the characterization of the spatial patterns of ATEs, a literature overview of ATE research shall serve as point of departure for the workflow proposed in this methodological paper to detect trees in general.

2 Overview of Alpine Treeline Ecotone Research with focus on methodology

As already implied, Alpine Treeline Ecotone analysis has been carried out so far at local, regional and global scales with different focus. Chhetri & Thai (2019) investigate the use of GIS and remote sensing methods in ATE research between 1980 and 2017 and demonstrate that with the advances in sensor technology the access to higher resolution data and different data types the analysis methods diversify. The idea of a research overview to underline the reasons for this project is based on and was inspired by Chhetri & Thai (2019), with the focus on the automated detection of ATEs. The following overview of remote sensing methods is crude and by no means comprehensive. In many cases it is hard to distinguish between the different methodological approaches because they all applied multiple methods and methods have been applied for diverse scopes.

2.1 Statistical – analytical approach

Until the dispersal of openly available remote sensing products the approach was mainly concentrated on physiological response based investigations with statistical analysis of local indicators based on *in situ* measurements. The main question was mainly: which environmental drivers control the location of ATE's and how complex are these (Körner 1998)? Drivers were differentiated on global, regional and local levels (Holtmeier – Broll 2005). Temperature was defined as the main global driver (Körner (1998), Koerner (2012), Holtmeier & Broll (2007), 2, Bonanomi et al. (2018)), but at regional and local levels it looks more diverse: topography (Brown (1994), Virtanen et al. (2004) and Bader & Ruijten (2008)), geomorphologic processes, herbivory or anthropogenic disturbance (Chhetri & Thai (2019)) were identified as drivers. Analysis of topographic actors with logistic regression was used frequently (e.g Brown (1994), Virtanen et al. (2004) and Bader & Ruijten (2008)).

2.2 Remote sensing approach

The detection or localization of ATEs is facilitated by the availability of remote sensing data (satellite, airborne lately hyperspectral and UAV derived ortho/imagery). Specialized sensors operating in the R, G, B, IR, short-wave, hyperspectral and thermal regions of the electromag-

netic spectrum give the possibility to approach and work with the specific spectral signatures (due to different absorption and reflection of radiation) of different vegetation entities. LiDAR data enables to include the factor height thus moving the analysis of treelines in a 3D space. The main focus of this methodological paper is on the remote sensing approach (automated detection) and exemplary papers are accentuated in a short summary. The use of GIS and remote sensing has been constantly increasing in treeline studies since 2000, with a few preceding pioneers. Earlier studies concentrated on mapping treeline positions and lately the interest shifted towards factors that control treeline variation (Chhetri & Thai (2019), 1543). It can also be seen that with the development of the respective sensors the interest and use moved to data with higher spatial resolution (LiDAR data, Hyperspectral data), which on the other hand attracts more cost and thus often thins out the studies due to the lack of monetary resources and also the use of proprietary software. This shows that there is a lot to do in means of open-source and reproducible best-practice applications and code-accessibility. Usually there is little information on the software used. The use of remotely sensed data is usually combined with classical approaches like statistical analysis in complex research questions, from which the following main directions emerge:

2.2.1 Mapping of Alpine Treelines

To map ATEs, studies use Aerial and Landsat Imagery to identify and quantify treelines (Brown (1994), W. L. Baker et al. (1995), Allen & Walsh (1996), Kimball & Weihrauch (2000), Virtanen et al. (2004), and Resler et al. (2004)). Frequently also vegetation indices are used (Myneni et al. (1997), Singh et al. (2015), and Mohapatra et al. (2019)) to analyse treeline elevation (Allen & Walsh (1996) and Kimball & Weihrauch (2000)) and topographic variables/geomorphological parameters (slope, angle, curvature, relief) to explain treeline structure (patch-metrics) (Kimball & Weihrauch (2000)). Tree population parameters are derived via PCA (e.g W. Baker & Weisberg (1997)) and also species distribution modelling is commonly applied (Chhetri et al. (2017)).

2.2.2 Montitoring Alpine Treelines and their Change detection

ATEs are space and time related phenomenons and they respond to changing environmental conditions, that is they can be sensitive to climate change (Singh et al. (2015), Holtmeier &

Broll (2005), Holtmeier & Broll (2007), Harsch & Bader (2011), and Bader et al. (2021)). The rise in global average temperatures seems to lead to the geographically varying shifting of ecotones: on regional level to upward shift ((Mohapatra et al., 2019)) but also stable or retracting ATEs can be determined (Winings (2013)). It still has to be understood if the results are due data quality. The identification and quantification of change in the ATEs were carried out with regional and global monitoring (Chhetri & Thai (2019)).

2.2.3 Automated detection and mapping of Alpine Treelines and ATEs

Recently several research projects, Master theses and PhDs have investigated (semi-)automated detection and mapping methods of Alpine Treelines (see a list until 2013 in Winings (2013) and also the recent literature cited here). As already stressed, the availability of high resolution data facilitate the use of more and more sophisticated methods. This methodological paper is concentrating specifically on automated analysis methods.

Automated methods imply the use of specific algorithms to extract information from remote sensing data, either Pixel- or Object Based or recently also using Deep Learning (mainly Convolutional Neural Networks - CNNs). Some studies compare Object- and Pixel-based or different Object-based Segmentation methods (Immitzer et al. (2012), Winings (2013), and Kupková et al. (2017)). Further it has to be emphasized, that the automated detection of ATEs relies heavily on tree detection. Parallel to elaborated workflows for the automated detection of Alpine Treelines and Ecotones continuous improvements are made on tree detection methods and tree cover estimation (Whiteside et al. (2020)) closely connected to the development of sensors and new data processing methods (Qiu et al. (2020), Weinstein et al. (2019) and Weinstein et al. (2020)) and form an important basis for automated analysis. In the following the most prominent automated methods are presented shortly including a few case studies.

Pixel-based Image Analysis is working with the information encoded in pixels – it assigns each pixel to a specific class on the basis of the respective values of the spectral bands, index or morphometric information (slope, aspect, etc.). One drawback is, that the context of the pixels and its neighborhood gets neglected and the pixel values can be affected by circumstantial effects, like reflectance differences (Stueve et al. (2011)), shadow or clouds (Allen & Walsh (1996)). Also the method doesn't deal with textures *per se*, and for this a textural analysis has to be done by using different filters (mean, sobel, focal, etc.).

Resler et al. (2004) use panchromatic aerial imagery where they incorporate spectral (brightness values) and spatial (textural) information to classify 4 classes (tundra/bare, alpine meadow, open forest/krummholz and closed canopy) representing the ATE using the maximum likelihood algorithm. A classification with and without textural information was done to assess the meaningfulness of textural information. The ERDAS modeler was used to extract textures.

Král (2009) uses CIR orthophotos to do a "classical" land cover classification using the maximum likelihood classifier which then is reclassified into 2 classes (spruce canopies and other). Subsequently a focal filter is applied to the spruce canopy closure class for texture analysis, which is then reclassified into 5 classes (no trees, emergent trees, groups of trees, open-canopy forest, closed canopy forest). Class 3, that is groups of trees (26-50% spruce) neighboring to alpine grassland and open-canopy forest was defined as ATE.

Immitzer et al. (2012) used WW-2 satellite data (8 + 4 bands) for the identification of 10 tree species by means of Random Forest classification using spectra of a) manually delineated tree crowns b) derived tree crown polygons and reference samples for tree species (object-based vs. pixel-based classification).

Winings (2013) used high resolution Aerial Imagery and LiDAR data in her Masters thesis to map the Alpine Treeline. She compared pixel- and object based classification. She used four different data input for both classification methods: NDVI, NDVI + Multispectral Aerial Imagery, NDVI + tree height or NDVI + Multispectral Aerial Imagery + tree height. In the case of the Pixel-based Classification the Maximum Likelihood and the Unsupervised ISODATA (Iterative Self-Organizing Data Analysis Technique) clustering algorithms were compared. For the Object-based Image Analysis Multiresolution Segmentation was conducted, using color and shape homogeneity. After the Segmentation, the classes (tree vs. non-tree) were assigned based on object feature threshold. The accuracy for the Pixel-based Classifications was between 85.3 and 88.4 and for the Object-based Image analysis between 81.5 and 92.9 %, resulting in the best classification on the data set with NDVI + Multispectral Aerial Imagery + tree height. For the Pixel-based Image Analysis ENVI and ERDAS and for the Object-based Image Analysis eCognition was used.

Kupková et al. (2017) used Airborne Hypersepctral (APEX and AISA DUAL) and Sentinel-A data for the classification of Tundra vegetation by comparing Pixel-based and Object-based

Image Analysis. Reference data was collected corresponding to 8 vegetation classes (anthropogenic areas, picea abies, pinus mugo dense, pinus mugo sparse, closed alpine grassland, grasses, alpine heathlands, wetlands and peat bogs; with a detailed and a simplified legend). Based on the difference in resolution the Hyperspectral data and the Satellite Imagery was classified separately. Latter was only classified pixel-based and with SVM (Support Vector Machines with radial basis function), NN (Neural Net) and MLC (Maximum Likelihood Classification) algorithms. The Hyperspectral data, having a higher spatial resolution was classified Pixel- and Object-based. For the Pixel-based Classification SVM, NN and MLC algorithms were used. For the Object-based Image analysis Edge-based Segmentation was used on the Hyperspectral data sets. The Hyperspectral data yielded better classification result than the Satellite data, with SVM pixel-based classification. ENVI was used for the study.

Object-based Image Analysis (GeOBIA) on the other hand is dealing with the grouping of pixels in homogeneous groups, that is segments which bear similar spectral, spatial and textural information. From each segment additional information can be extracted (statistical information, size, shape and context). Different Segmentation algorithms exist, which treat the image and the segments different.

Middleton et al. (2008) used the Feature Extraction Module (Fx) implemented in ENVI to extract tree crowns from two aerial photographs (one from 1947 and one from 2003) via Segmentation and Feature Classification with SVM with textural, spatial and spectral information. The results were compared to forest inventory information and an upward shift was recorded on Lommoltunturi fell.

Ranson et al. (2011) used MODIS VCF (Vegetation Continuous Fields) tree cover data and segmentation to delineate the circumpolar Taiga-Tundra ecotone (TTE). The multi-annual VCF was adjusted using linear regressions and a vector layer with previously delineated Taiga and Tundra Biomes. Water bodies were masked out. Subsequently multi-resolution Segmentation was carried out with eCongnition based on the homogeneity criterion. The resulting polygons were then classified on a specific range of adjusted VCF values which represent the TTE.

Mishra et al. (2018) used a UAV equipped with a Parrot Sequoia multispectral (Red, Green, Blue, Red Edge, Near Infra-Red) camera to acquire high resolution Imagery. Subsequently an SfM Orthoimage was calculated and then Multiresolution (based on the homogeneity criterion of

scale, shape/color and compactness/smoothness) and spectral difference Segmentation (merging neighboring objects based on a spectral threshold) was combined in eCognition to generate optimal feature space variables for the classes. Then the Random Forest Classifier was used for classification with 3 sets of features (spectral features; spectral features + geometric/shape features; spectral features + geometric/shape features + textural features) for species-level mapping of vegetation in the Himalayas.

Whiteside et al. (2020) used derivatives of Aerial Imagery and WW2 satellite data (TGI, NDVI) resampled to 1 m and filtered by a low-pass filter. Then a threshold-based Multiresolution segmentation was conducted with eCognition to assess the tree cover (in percentage) for each date (1964, 1976, 1981, 2010). The results were compared by date to assess the tree cover reduction (4%) during the 36 years.

Geping Luo & Li Dai (2013) used Aerial Imagery from 1962 and 1981 and a QuickBird Satellite image from 2006 as data input to map vegetation distribution after orthorectification and a DEM was generated. The land-cover types Schrenkiana, Sabina and other were delineated. Multiresolution Segmentation was conducted in eCognition subsequently combined with a k-Nearest Neighbour Classification. The result was compared to fieldwork data collection (2010, 2011) of the two species. Land-cover change was examined between 1962, 1981 and 2006.

Qiu et al. (2020) proposed a new Spectral Multiscale (SMS) Individual Tree Crown (ITC) delineation method using both brightness and spectra of high-resolution Multrispectral Imagery to be able to better delineate tree crowns in deciduous or mixed forests, where adjacent tree crowns are very close to each other. As the first step a morphological gradient map is calculated of multispectral images, then as a second step an Inverse Gradient Image. Then initial treetops were extracted by multiscale filtering and morphological operations with regard to tree crown shape which then were refined with the spectral reference of the neighboring treecrowns (creating a treetops map). Subsequently the morphological gradient map is segmented by marker-controlled Watershed Segmentation which is then refined by the treetops map, to receive an individual tree crown delineation map.

Deep Learning – contrary to Pixel-based Classification and Object-based Image Analsysis – works on scene level and enables thus to deal better with the complex semantic structure of the increasing resolution of remote sensing images. A multitude of different Deep Learning models

exist with different structures to fulfill different aims (e. g. Image Classification, Object Segmentation, e.g.). The most common Deep Learning model structure are CNNs – Convolutional Neural Networks: multi-layer networks with learning ability that consist of convolutional layers, pooling layers, and fully connected layers.

Fricker et al. (2019) used airborne Hyperspectral Imagery, LiDAR data with a CNN Framework to automate tree species classification. 7 dominant tree species and a dead tree class were identified to serve as reference data for the CNN. A LiDAR derived CHM was used to digitize the individual tree canopies to prepare their pixels for the species labeling for the CNN. The classification was executed separately on the RGB and the hyper-spectral data. The classification with the hyper-spectral data (0.73 – 0.90) yielded better classification results than the RGB classification (0.41 – 0.88). All code and data to ensure reproducibility can be found online.

Weinstein et al. (2019) proposed a Semi-Supervised CNN workflow based on the comparison of 3 unsupervised Tree-Crown Segmentation algorithms. The result of the chosen Tree Crown Segmentation (clustering of a CHM by tree height and crown width) of the LiDAR data was extracted as a bounding box from the RGB image, which' data set is then labeled self-supervised and pretrained by a retinanet CNN. The CNN was then retrained with a small hand-annotated data set (supervised classification), to correct errors from the initial Unsupervised Segmentation, which indeed improved the results of the prediction.

Weinstein et al. (2020) build on the results from **Weinstein et al. (2019)** and tested if training data sets can be generalized and be used on completely different forested areas. Generally the performance of the model performance decreased, but when they were applied to spatially and spectrally similar forested areas the performance increased. Best was again, when the CNN was retrained by a handful of hand-annotated data from the same area.

This overview of chosen literature demonstrated, that often it is not communicated which software was used and also no clear workflow, only a flowchart is published. The software used if mentioned is often proprietary (such as ENVI and eCognition), what means, that the results cannot be reproduced and even applied to and teste on different data sets. Only one of the investigated works are reproducible and replicable: Fricker et al. (2019).

Inspired by the previous literature a fully reproducible and replicable workflow for the detection of trees in general was developed. Based on the fact, that the automated detection of ATEs

relies heavily on tree detection, a workflow is proposed to detect trees using so-called shallow learning, that is Object-based Image Analysis in case of available LiDAR data (which is sadly mostly not the case) and Pixel-based Image Analysis in the case of Aerial Imagery or Satellite data (which is usually the case). If both data sets are available, then the results of the PBIA can be compared with the result of the Segmentation based on cell-statistics, bearing in mind the difference of both methods. The development of a Deep Learning framework would've exceeded the scope and possibilities of a seminar thesis. The detection and mapping of trees is a robust information base for future analysis of ATEs (which will be discussed in the Conclusion).

3 Methodology

3.1 The case study area, the Region of Interest and the data sets used in the case study

The workflow is presented on a case study from the North-Central Vandans region, situated in the South-West of the Vorarlberg Alps in the West of Austria. The Region of Interest (ROI) lies on the West mountainside of the *Mädli*, North of Voralpe Vilifrau. The ROI (250 x 190 m) was chosen at the upper treeline, which can be described as an *abrupt diffuse krummholz treeline* according to the classification of Bader et al. (2021), taking the young trees and occasional shrubs/krummholz below and around the treeline into account. **Figure 3** describes the situation best.

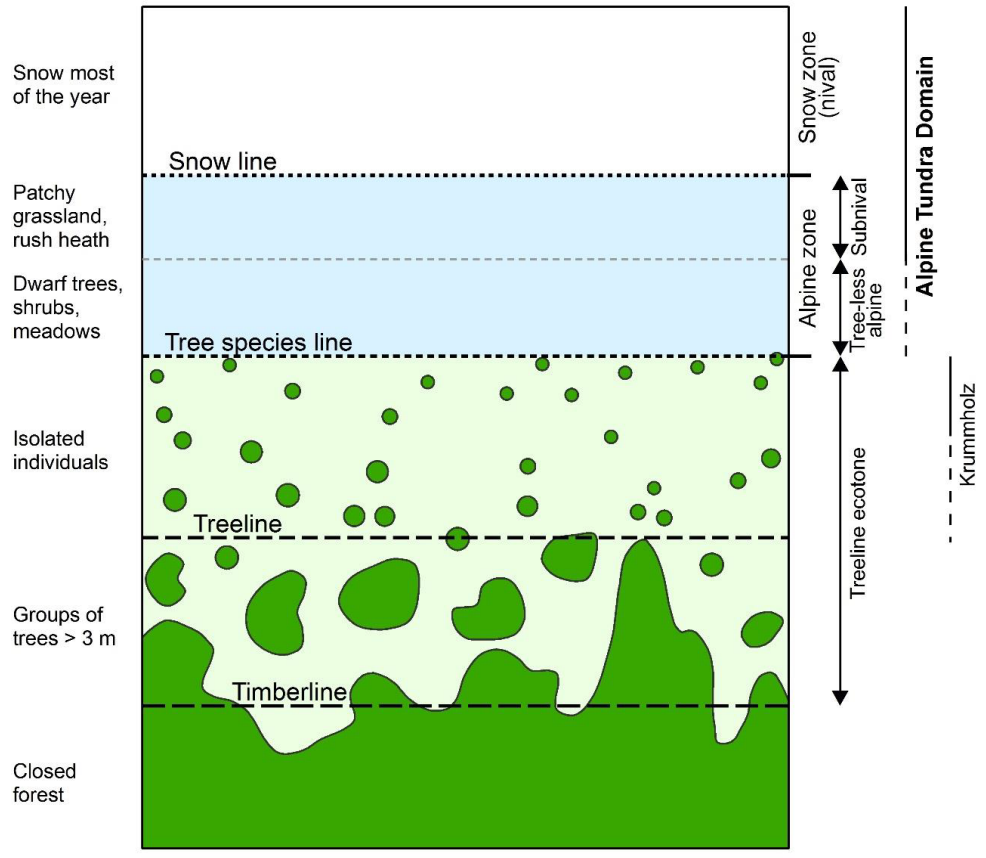

Figure 3: Schematic depiction of the ATE. It clearly sets the ROI of this case study above the timberline and still around the treeline. Source: Barredo Cano et al. (2020)

In this project LIDAR data (without any metadata and also the date of acquisition) as well as RGB and IR (Infra-Red) Aerial Imagery from 2012 was used, all downloaded from the VoGIS

website.

For the development of the workflow 4 test areas were defined inside the ROI to first test the values of the different algorithms on test area 1 and if needed also on the rest of the rest areas. Originally masks for each 4 test areas were created in QGIS and based on them, the Canopy Height Model (CHM) and the RGB and IR Aerial Imagery was cropped to the size of the test area masks. When it became clear, that the minimal computable raster size (on grounds of the restrictions of the `raster package` in `R`) was a lot smaller than the maximum size of the actual ROI originally decided, it was reduced. The sizes of the CHM test areas were defined using the **'Clip raster by Extent'** tool, with the ***'use map canvas'*** setting in QGIS, to define 4, approximately 32 x 32 m test areas (**Figure 4**). Then the respective CHM test areas were used as masks/extents to clip the respective RGB and IR test areas in QGIS.

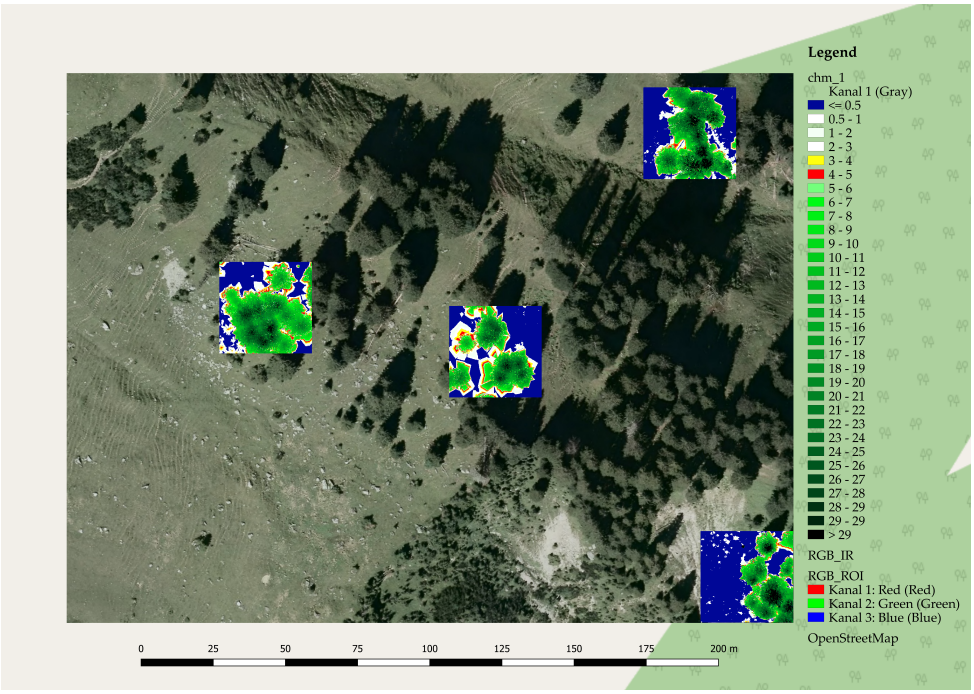

Figure 4: The RGB ROI with the 4 (in this case chms) test areas. The order of the test areas is the following: test area 1 is top right, test area 2 centre, test area 3 centre left and test area 4 bottom right. All chms are classified with the same value range, depicted in the legend. Scale 1:800

3.2 The reproducible workflow

The reproducible workflow is made up of the following files:

`00_library_n_prep.R`, `0_set_up_working_envrnmt.R`, `1_data_prep_general.R`,

`2_segmentation_test_area_1.R`, `3_CV_segmentation_test_areas.R`, `4_segmentation_ROI.R`, `5_data_prep_classification_test_area_1.R`, `6_classification_test_area_1.R`, `7_data_prep_classification_ROI.R`, `8_classification_ROI` and `9_validation_test_area_1_ROI`.

These commented `.R files` build up on each other step-by-step, starting with the pre-processing of the input data. This `.Rmd` only presents the backbone which links the `.R files` together.

To reproduce the results of the whole project you have to run the code in the `.R files`, numbered sequentially. `00_library_n_prep.R` is a library file, containing all the packages needed for this project. It installs packages from the R repository or from github if some would be missing. This file does not need running, unlike the next one, `0_set_up_working_envrnmnt.R`. It is going to set up the project folder structure on your computer/laptop, but you have to adapt the path of the project directory to the destined place on your computer/laptop and also attach or install the packages needed.
The data used in this project is stored on Google Drive, from where you have to download it to the respective folders (**dsm/**, **dem/**, **RGB_IR/**, **RGB_IR/**, **treepos/**, **train/**) set up on you device by the script you just ran (because a)**github** limits the size of data to be uploaded and b) the data used is not openly available, that is proprietary).

The main workflow consist of three sub-workflows, which are equally important: the Object Based Image Analysis (OBIA: `.R files 2-3-4`, working with the LiDAR data set) and the Pixel-Based Image Classification (PBIA: `.R files 5-6-7-8`, working with the Aerial imagery), which latter is then validated in `.R file 9` by the result of the Object-based Image Analysis (**Figure 5**).

The workflow is designed in such a way, that the individual steps were developed on test area 1 and tested (generalized) on the other test areas to deal with the unavoidable variability of data (at least in the case of the Object-based Image Analysis). The resulting settings and variables are then to applied to the ROI. This multistep procedure is used to make sure that the algorithms applied and values used are effective and robust.

The `.R files` are commented in a manner that they can be used as a manual or tutorial together with this study. This chapter contains the workflow and explanations connected to

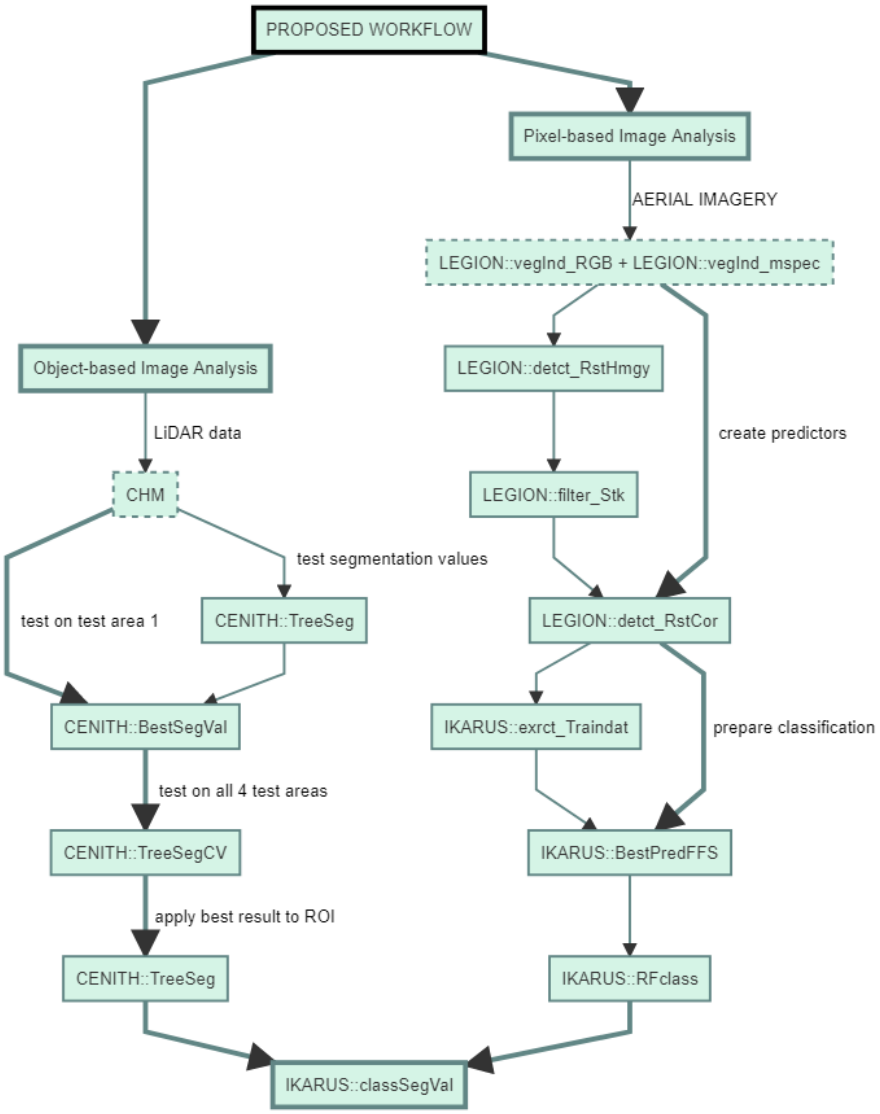

Figure 5: Flowchart of the workflow developed for the automated detection of trees in general using LiDAR data and Aerial Imagey. The workflow consists of three main parts: a workflow for Object-based Image Analysis, a workflow for Pixel-based image analysis and the validtion of the result of the Classification with the results of the Segmentation.

each step and the results will be discussed in **Chapter 4** to keep it short and condensed.

For this project three R packages were developed: `CENITH`, `LEGION` and `IKARUS`.
`CENITH` is a wrapper for the `ForestTools` package (Plowright & Roussel (2021)) to perform tree Segmentation on any specific area. It also provides cross validation to estimate and validate the performance of a Segmentation model for multiple test areas. See more details in the help pages and in the tutorial of the package.
`LEGION` computes indices from RGB and Multispectral Imagery and provides also their basic statistics (correlation and homogeneity). See more details in the help pages and in the tutorial of the package.
`IKARUS` is a wrapper for the `CAST` package (H. Meyer et al. (2022)) to perform Pixel-based Image Analysis and Classification.See more details in the help pages and in the tutorial of the package.

3.3 General data preparation

`1_data_prep_general.R`

In the data preparation files the data set(s) used in the workflow are prepared.
The CHM was created by subtracting the Digital Elevation Model (DEM) form the Digital Surface Model (DSM) (*1.1* - these numbers refer to the number of `.R files` and the respective section in the code, not the chapters in the text). This way only the LiDAR points connected to the vegetation above ground are preserved and can be used for calculations (**Figure 6**).

3.4 Segmentation of the test area 1 `2_segmentation_test_area_1.R`

With `.R script 2` the first sub-workflow, the Object-based Image Analysis of the test area 1 starts. The aim of this first step is to find an accurate segmentation for test area 1 which can be tested on the other test areas and then applied to the ROI.

The segmentation is performed using the `CENITH` package. As `CENITH` is a wrapper for `ForestTools`, it also uses *'Watershed Segmentation'* with markers (F. Meyer & Beucher (1990)) which is implemented in `ForestTools`. *'Watershed Segmentation'* is an *'Edge-based Region Growing Segmentation'* technique and simulates real-life flooding. It identifies clear segment boundaries which it then "fills up." Markers guide the algorithm how/ from which point to "fill

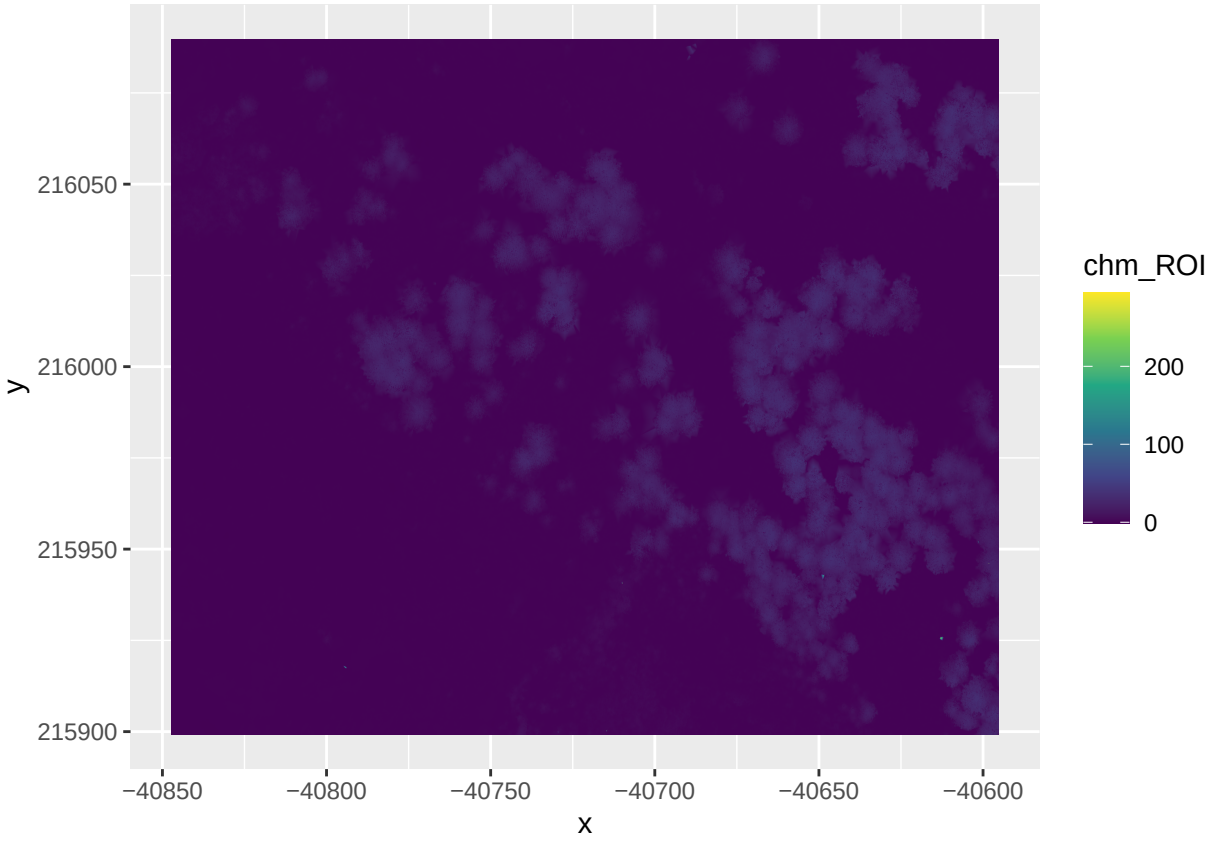

Figure 6: The created CHM of the Region of Interest (ROI). The maximum value of the CHM is 293.8199 m, most probably detection error.

up" the basins/segments (**Figure 7**, Roffler (2020), 33).

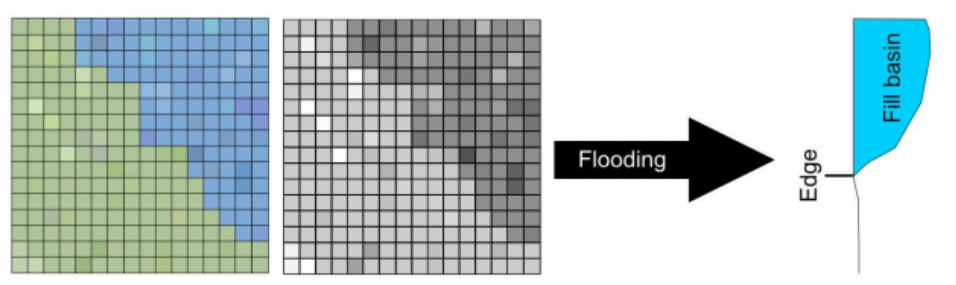

Figure 7: Operating principle of the 'Watershed Segmentation' Roffler (2020), 33.

The first task is to set verification points (this can be also done with the `ForestTools package`, but we preferred to do it by hand thus it was not implemented in the `CENITH package`) - that is locate the positions of the trees in the test areas using QGIS.

Before defining the tree positions in each test areas, first of all the minimum height of the trees has to be identified (which of course is region, area and species dependent). In the Alpine region the vegetation consists of trees but also shrubs/krummholz, which have to be distinguished from trees. If also young trees trees are present and should be considered, then it is important not to confuse them with shrubs, which is often rather complicated and even impossible. If Aerial or Satellite Imagery is also available (as in our case), it can help to consider the difference in the spectral and textural signature.

The identification of the minimum tree-height can be best determined using QGIS. Thus to distinguish the actual trees, for one the spectral information of the Satellite Imagery (which will be mainly used for the Pixel-based Image Classification) was exploited. Trees demonstrate - apart from their height in the CMHs - much more branches which leads to less homogeneous green color (mixed with black and creme colored pixels) as that of the shrubs which show in their entirety a single color. On the other side the CHMs were displayed by giving each height between < 0.5 m and > 29 ma different color (see the QGIS style file).

Combining these two sets of information, the minimum tree height was defined at 4-5 meters (**Figure 8**). This height included the lowest young trees but not the shrubs and Krummholz. To be able to include the young trees, it was opted for h=4 instead of 5 meters.

The exact position of the individual trees (including young trees) was based on on the decision of the minimum tree height based on the RGB and IR Aerial Imagery and was determined by

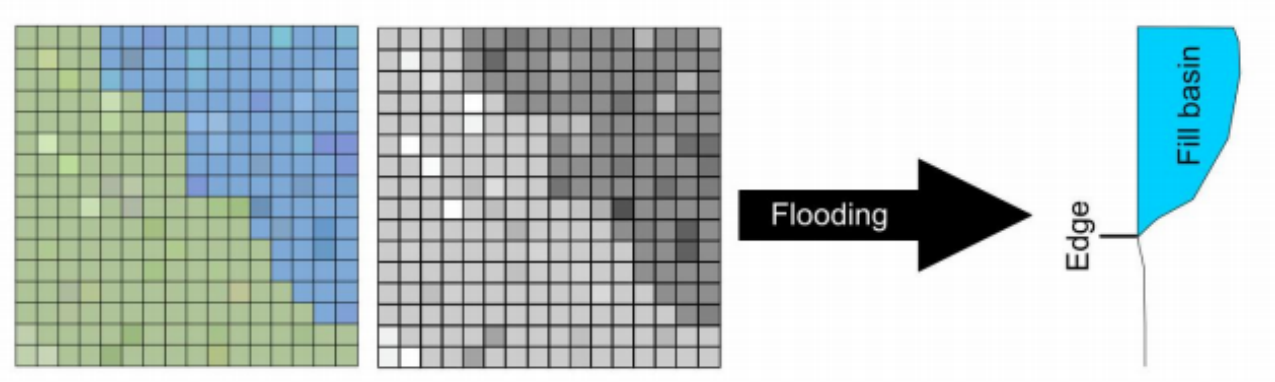

Figure 8: Comparision of RGB Satellite Imagery and LIDAR-derived CHMs classified by height using different colors (blue < o.5 m; white > 0.5 < 3; yellow > 3 < 4m, red > 4 < 5m; green to black < 5m) Detail of test area 4.

looking for the highest point of the CHMs (**Figure 9**).

```
## Reading layer `vp_1' from data source
##   `C:\Users\kelto\Documents\detectATE\analysis\data\treepos\vp_1.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 15 features and 1 field
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: -40638.55 ymin: 216053.3 xmax: -40615.55 ymax: 216083.1
## Projected CRS: MGI_Austria_GK_West_with_axis_order_normalized_for_visualization
```

The segmentation itself can be executed with the `CENITH::TreeSeg` function. This function works with a *MovingWindow* of size a * b (x,y) at height h. `a`, `b` are horizontal and vertical values (x, y) in meters and `h` is height in meters on the CHM.

This function takes single values, with which different and various values of the segmentation variable can be tested. The results can be checked in QGIS.

The variables `MIN` and `MAX` filter the segments to a certain size. Because with the variables used there is small chance to get too big segments, the `MAX` variable is not used, only the `MIN` variable, to filter out small segments. Another variable, which controls and filters out small artifacts and data errors - directly on the chm is the `chmFilter`.

The tests with `CENITH::TreeSeg` show (*2.3* - these numbers refer to the respective section in the code), that it is very useful to apply a `sum filtered` CHM (see the difference between the

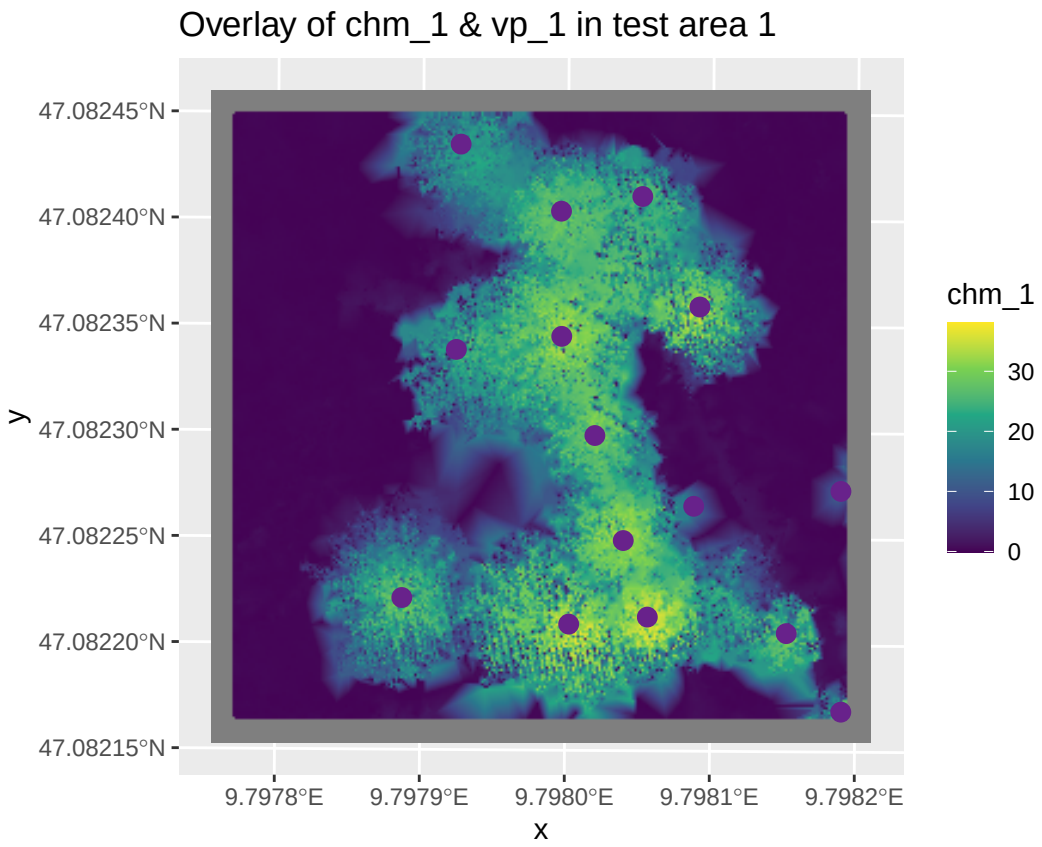

Figure 9: The defined tree positions (vp) in chm 1 of the test area 1. It is visible, that still with a CHM and also IR Aerial Imagery it is not an easy task to define trees.

results **test_0** and **test_1** in QGIS) and give an estimate of how big or small the variables `a`, `b` and `MIN` have to be set. Based on the fact that there are 15 trees in test area 1 we can estimate how accurate the segments might be. Taking a look at test 1, 2 & 3 we can see how the reduction of `a` and `b` elevated the number of trees. This is useful information for `CENITH::BestSegVal`.

The function `CENITH::BestSegVal` can be fed with long sequences of values for each variable (this is called parameter tuning). N.B. long sequences mean longer calculation time, thus it is best to test out different variable settings with `CENITH::TreeSeg` to narrow down in which setting range a, b and h might move. In the second part of the script `2_segmentation_test_area_1.R`, the use of `CENITH::BestSegVal` is demonstrated (*2.4*). In this function also the vps (tree positions) are taken into account as a validation basis. All calculation in the resulting table are based on the comparison of the vps/tree positions to the segmentation results. The variable `hit.vp` shows explicitly, how many vps/tree positions are 'hit' by the resulting segments.

In the first run the values `a=seq(0.05, 0.1, 0.01), b=seq(0.05, 0.1, 0.01), h=seq(3.5, 6, 0.5), MIN=seq(0.1, 0.5, 0.1)` were tested, to go a bit below and above the defined height of the trees and to determine the best `a`, `b` and `MIN` values. `CHMfilter` is set at 3, because we want to do only minimal filtering to minimize small artifacts and gaps in the chm.

In the first run (*4.1*) we got 1080 results and 45 maximal segments. Because there are 15 trees in test area 1, the tibble was filtered to 20 segments (to count in possible undersegmentation) and arranged according to the best `hitrate` (which was **0.733333**) and still got 664 results. These are still too much results to decide which combination of settings should be applied to the other test areas. Thus the results were filtered to a `hitrate >= 0.7` (the best results), `height <= 4.0`. (the height of trees) and the total segments were ordered in descending order. This resulted in 40 observations (*4.1a*), from which the first four were actually calculated with `CENITH::TreeSeg` and compared in QGIS. The lesson learned was, that it became clear that MIN (the size of the smallest segment) has to be set to 0.1 and that the range of a & b of the best results is > 0.1.

Thus a second run of `CENITH::BestSegVal` was set up (*4.2*) with the variables `MIN=0.1, a=seq(0.01, 0.1, 0.01), b=seq(0.01, 0.1, 0.01)`, to find better segment sizes and

`h=seq(3.5, 6, 0.5)` with the same settings as in the first run, to enable a systematic analysis. The 600 results were filtered the same way as the first run: filtering to 20 segments lead to 204 results, still keeping the best `hitrate` at **0.733333**. The filtering to `hitrate >= 0.7` and `height <= 4.0` lead to 8 observations (**Table 1**), of which six are the same results of the first round (*4.2a*). This broad correspondence of these 8 results confirmed, that all should be applied to the other test areas in the next step (`3_CV_segmentation_test_areas.R`). All steps can be computed in the respective .R files.

Table 1: Results of the second run of CENITH::BestSegVal

a	b	height	total_seg	hit.vp	under	over	area	hitrate	underrate	overrate	Seg_qualy	MIN	chm
0.07	0.07	3.5	20	11 / 15	2	7	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.08	3.5	20	11 / 15	2	7	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.09	3.5	20	11 / 15	2	7	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.07	4.0	20	11 / 15	2	7	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.08	4.0	20	11 / 15	2	7	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.09	4.0	20	11 / 15	2	7	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28	0.1	chm_1_3
0.07	0.10	3.5	19	11 / 15	2	6	543.1562	0.7333333	0.1052632	0.3157895	0.73 @ 0.26	0.1	chm_1_3
0.07	0.10	4.0	19	11 / 15	2	6	535.0156	0.7333333	0.1052632	0.3157895	0.73 @ 0.26	0.1	chm_1_3

3.5 Cross-validated Segmentation of the test areas

`3_CV_segmentation_test_areas.R`

The filtered results of the segmentation of test area 1 resulted in eight possible segmentations which are going to be tested if and how they fit the other test areas. Up to some degree it is an experiment to test which might be the possibly best segmentation which fits the other test areas. That is why all eight results from `best_seg_vp_1_filt_v2_filt` were processed. The eight results contain also h=3.5 m, so it can be checked which height really represents best the whole ROI. The cross-validates segmentation is done by the `CENITH::TreeSegCV` function.

The test areas and vps (tree positions) are given as a list to the function. The length of both lists/number of test areas and vps have to equal. The number of test areas and vps set the number of folds used, in this case 4. It gives a similar table as result as `CENITH::BestSegVal`, but calculates also the overall performance of the segmentation quality (which is based on how many and how much of the vps/tree positions were 'hit'), printed out when the cross-validated segmentation has finished.

All eight settings produce on all test areas an overall performance (mean performance in Table

Table 2: Results of the cross-validations cv1 - cv8

cv1	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.07	4	20.0	11.00	15.00	2.0	7.00	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.07	4	28.0	10.00	12.00	1.0	17.00	421.0312	0.8333333	0.0357143	0.6071429	0.83 @ 0.34
3	chm_3	0.07	0.07	4	40.0	16.00	18.00	1.0	23.00	605.9531	0.8888889	0.0250000	0.5750000	0.89 @ 0.31
4	chm_4	0.07	0.07	4	22.0	8.00	12.00	2.0	12.00	369.2031	0.6666667	0.0909091	0.5454545	0.67 @ 0.36
5	Mean	0.07	0.07	4	27.5	11.25	14.25	1.5	14.75	482.8008	0.7805556	0.0629058	0.5193994	0.78 @ 0.32

cv2	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.08	3.5	20.00	11.00	15.00	2.0	7.0	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.08	3.5	28.00	10.00	12.00	1.0	17.0	443.3438	0.8333333	0.0357143	0.6071429	0.83 @ 0.34
3	chm_3	0.07	0.08	3.5	43.00	16.00	18.00	1.0	26.0	623.5000	0.8888889	0.0232558	0.6046512	0.89 @ 0.33
4	chm_4	0.07	0.08	3.5	22.00	8.00	12.00	2.0	12.0	377.0000	0.6666667	0.0909091	0.5454545	0.67 @ 0.36
5	Mean	0.07	0.08	3.5	28.25	11.25	14.25	1.5	15.5	496.7461	0.7805556	0.0624698	0.5268121	0.78 @ 0.33

cv3	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.09	3.5	20.0	11.00	15.00	2.0	7.00	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.09	3.5	27.0	10.00	12.00	1.0	16.00	443.4219	0.8333333	0.0370370	0.5925926	0.83 @ 0.33
3	chm_3	0.07	0.09	3.5	42.0	16.00	18.00	1.0	25.00	623.5156	0.8888889	0.0238095	0.5952381	0.89 @ 0.32
4	chm_4	0.07	0.09	3.5	21.0	8.00	12.00	2.0	11.00	377.0156	0.6666667	0.0952381	0.5238095	0.67 @ 0.36
5	Mean	0.07	0.09	3.5	27.5	11.25	14.25	1.5	14.75	496.7734	0.7805556	0.0640212	0.5154101	0.78 @ 0.32

cv4	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.07	3.5	20.00	11.00	15.00	2.0	7.0	543.1406	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.07	3.5	28.00	10.00	12.00	1.0	17.0	443.3438	0.8333333	0.0357143	0.6071429	0.83 @ 0.34
3	chm_3	0.07	0.07	3.5	43.00	16.00	18.00	1.0	26.0	623.5000	0.8888889	0.0232558	0.6046512	0.89 @ 0.33
4	chm_4	0.07	0.07	3.5	22.00	8.00	12.00	2.0	12.0	377.0000	0.6666667	0.0909091	0.5454545	0.67 @ 0.36
5	Mean	0.07	0.07	3.5	28.25	11.25	14.25	1.5	15.5	496.7461	0.7805556	0.0624698	0.5268121	0.78 @ 0.33

cv5	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.08	4	20.0	11.00	15.00	2.0	7.00	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.08	4	28.0	10.00	12.00	1.0	17.00	421.0312	0.8333333	0.0357143	0.6071429	0.83 @ 0.34
3	chm_3	0.07	0.08	4	40.0	16.00	18.00	1.0	23.00	605.9531	0.8888889	0.0250000	0.5750000	0.89 @ 0.31
4	chm_4	0.07	0.08	4	22.0	8.00	12.00	2.0	12.00	369.2031	0.6666667	0.0909091	0.5454545	0.67 @ 0.36
5	Mean	0.07	0.08	4	27.5	11.25	14.25	1.5	14.75	482.8008	0.7805556	0.0629058	0.5193994	0.78 @ 0.32

cv6	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.09	4	20.00	11.00	15.00	2.0	7	535.0156	0.7333333	0.1000000	0.3500000	0.73 @ 0.28
2	chm_2	0.07	0.09	4	27.00	10.00	12.00	1.0	16	421.1094	0.8333333	0.0370370	0.5925926	0.83 @ 0.33
3	chm_3	0.07	0.09	4	39.00	16.00	18.00	1.0	22	605.9531	0.8888889	0.0256410	0.5641026	0.89 @ 0.31
4	chm_4	0.07	0.09	4	21.00	8.00	12.00	2.0	11	369.2344	0.6666667	0.0952381	0.5238095	0.67 @ 0.36
5	Mean	0.07	0.09	4	26.75	11.25	14.25	1.5	14	482.8281	0.7805556	0.0644790	0.5076262	0.78 @ 0.32

cv7	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.1	3.5	19.0	11.00	15.00	2.0	6.00	543.1562	0.7333333	0.1052632	0.3157895	0.73 @ 0.26
2	chm_2	0.07	0.1	3.5	26.0	10.00	12.00	1.0	15.00	443.4688	0.8333333	0.0384615	0.5769231	0.83 @ 0.33
3	chm_3	0.07	0.1	3.5	41.0	16.00	18.00	1.0	24.00	623.5781	0.8888889	0.0243902	0.5853659	0.89 @ 0.32
4	chm_4	0.07	0.1	3.5	20.0	8.00	12.00	2.0	10.00	377.0312	0.6666667	0.1000000	0.5000000	0.67 @ 0.35
5	Mean	0.07	0.1	3.5	26.5	11.25	14.25	1.5	13.75	496.8086	0.7805556	0.0670287	0.4945196	0.78 @ 0.31

cv8	sites	a	b	height	total_seg	hit	vp	under	over	area	hitrate	underrate	overrate	Seg_qualy
1	chm_1	0.07	0.1	4	19.00	11.00	15.00	2.0	6	535.0156	0.7333333	0.1052632	0.3157895	0.73 @ 0.26
2	chm_2	0.07	0.1	4	26.00	10.00	12.00	1.0	15	421.1406	0.8333333	0.0384615	0.5769231	0.83 @ 0.33
3	chm_3	0.07	0.1	4	38.00	16.00	18.00	1.0	21	605.9531	0.8888889	0.0263158	0.5526316	0.89 @ 0.3
4	chm_4	0.07	0.1	4	20.00	8.00	12.00	2.0	10	369.2500	0.6666667	0.1000000	0.5000000	0.67 @ 0.35
5	Mean	0.07	0.1	4	25.75	11.25	14.25	1.5	13	482.8398	0.7805556	0.0675101	0.4863360	0.78 @ 0.31

1) between **0.78 @ 0.31 and 0.78 @ 0.33**. This value is given right away by the algorithm as result. Looking at the data frame created by each cross-validated segmentation, the following can be observed (**Table 2**):

Inspecting the performance of the individual test areas, it is visible, that the behave similarly in each segmentation, but also that test area 4 has the lowest values all variables considered. This shows, that the performance of the different test areas depend on each other and balance each other out.

3.6 Segmentation of ROI

`4_segmentation_ROI.R`

Given the explanatory aim of this work, all eight results of `best_seg_vp_1_filt_v2_filt` (created in Chapter 3.4 and tested on all four test areas in Chapter 3.5) were applied to the ROI. The segmentation results, obtained with `CENITH::TreeSeg`, can be visually inspected and compared in QGIS. It became swiftly clear (as expected), that h=3.5 m is too low as tree height because it may result in segmenting some of the shrub as well (note the different colors of the chm heights in QGIS). Because of the similar height of the seedlings and young trees and shrubs it is practically impossible to distinguish shrubs and seedlings based purely on height (as discussed before) and thus the choice of 4 m as tree height was not revised.

All segmentations demonstrate almost the same overall performance between `0.78 @ 0.31 and 0.78 @ 0.33`. `segm_ROI_8` was favored for a most fitting segmentation (**Table 3**):

Table 3: Settings of **segm-ROI-8**

a	b	height	total_seg	hit.vp	under	over	area	hitrate	underrate	overrate	Seg_qualy	MIN	chm
0.07	0.1	4	19	11 / 15	2	6	535.0156	0.7333333	0.1052632	0.3157895	0.73 @ 0.26	0.1	chm_1_3

But what happens with heights above 4 m? 4.5, 5 and 5.1 m (`segm_ROI9` to `segm_ROI_12`) were tested to test if the results of `segm_ROI_8` can be refined. With a tree-height over 4 m, the number of charted young trees and the tree crown area is decreasing. Thus the optimal result is `segm_ROI_8`. The result of the segmentation is discussed in Chapter 4, Results.

3.7 Data preparation for the Classification of test area 1

`5_data_prep_classification_test_area_1.R`

With R script 5 the second sub-workflow, the Pixel-based Image Analysis of test area 1 is started. In this first step the RGB (`RGB_1`), IR Imagery (`IR_1`) and combined (`RGBNIR_1`) is prepared for the classification/prediction. The aim of this first step is to test settings for test area 1 which can be applied to the ROI.

This data preparation for PBIA is performed by using the `LEGION package` by computing RGB (`LEGION::vegInd_RGB`) and multi-spectral (MS, `LEGION::vegInd_mspec`) indices which can be filtered and then tested on correlation (*5.3*). Indices are used to enhance the spectral differences

between the different object classes (***1-tree, 2-shrubs, 3-grass, 4-tree in shadow, 5-shadow*** and ***6-stone***) which are to be detected. Currently 11 RGB and 4 MS indices are implemented. During the testing it was understood, that certain indices (`VARI`, `RI` and `HI`) contain ample homogeneous areas (**Figure 10**). Including these indices in further calculations would distort the results and deform the prediction.

```
##
## ### LEGION calculating (Visible Vegetation Index (VVI)) ###
##
## ### LEGION calculating (Visible Atmospherically Resistant Index (VARI)) ###
##
## ### LEGION calculating (Normalized difference turbidity index (NDTI)) ###
##
## ### LEGION calculating (Redness index (RI)) ###
##
## ### LEGION calculating (Soil Colour Index (CI)) ###
##
## ### LEGION calculating (Brightness Index (BI)) ###
##
## ### LEGION calculating (Spectra Slope Saturation Index (SI)) ###
##
## ### LEGION calculating (Primary colours Hue Index (HI)) ###
##
## ### LEGION calculating (Triangular greenness index (TGI)) ###
##
## ### LEGION calculating (Green leaf index (GLI)) ###
##
## ### LEGION calculating (Normalized green red difference index  (NGRDI)) ###
##
## ############################
## ### The LEGION is ready ###
```

Before dealing with the problem of homogeneous areas, first of all two index configurations

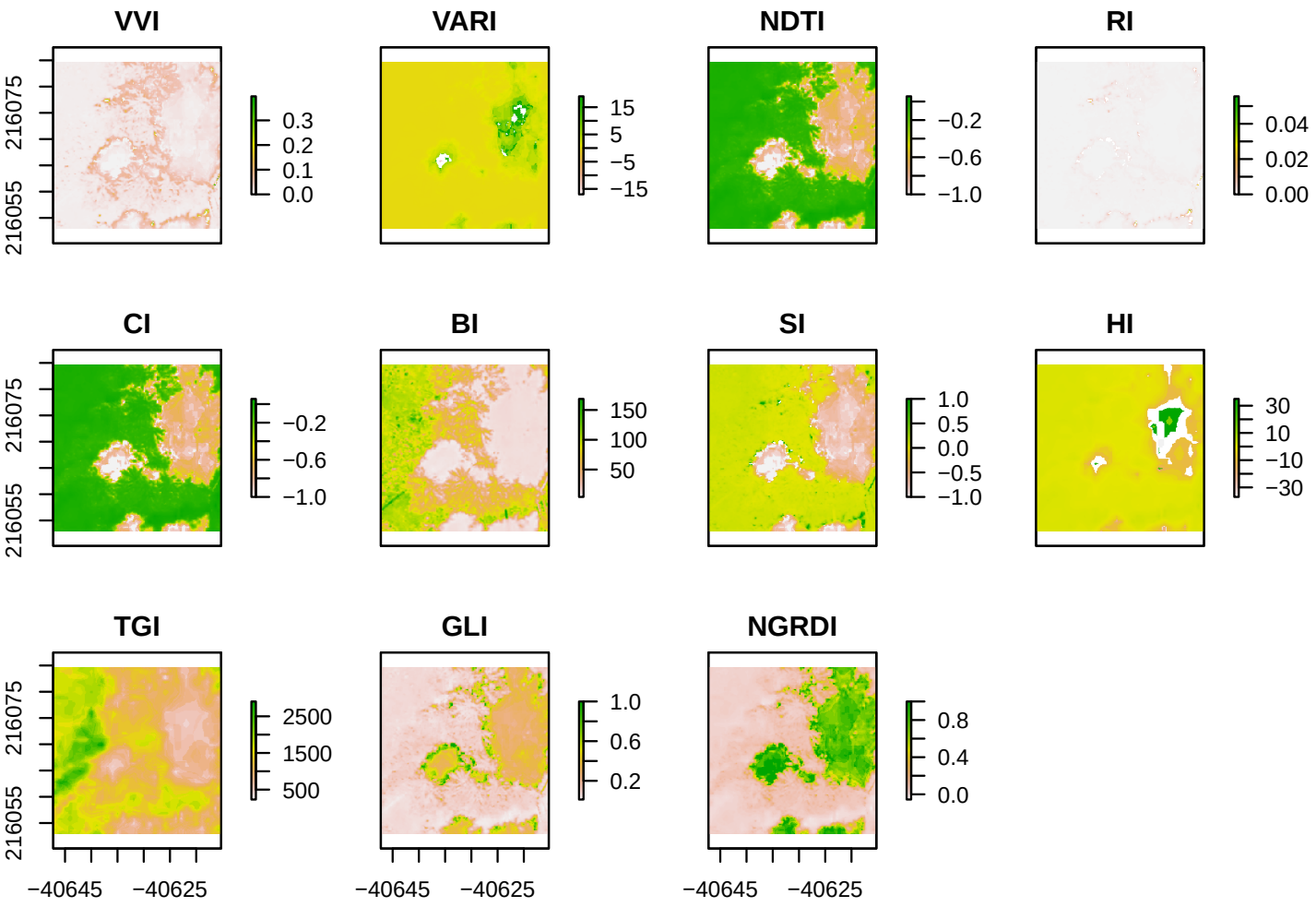

Figure 10: The computed 11 RGB Indices of test area 1.

were created (*5.4*):

`RGB_1_ind` - 11 layers: `VVI, VARI, NDTI, RI, CI, BI,SI,HI, TGI, GLI` and `NGRDI`

`RGBNIR_1_ind` - 4 layers: `NDVI, TDVI, SR and MSR`

These are then stacked together to form the following data stacks:

`ALL_ind_stack_1 (RGB_1_ind, RGBNIR_1_ind)` - 15 layers: `VVI, VARI, NDTI, RI, CI, BI, SI, HI, TGI, GLI, NGRDI, NDVI, TDVI, SR` and `MSR`

`ALL_RGBMS_stack_1 (RGB_1_ind, RGBNIR_1_ind, RGBNIR_1)` 18 layers:`VVI, VARI, NDTI, RI, CI, BI, SI, HI, TGI, GLI, NGRDI, NDVI, TDVI, SR, MSR, Red, Green, Blue` and `IR_1`

The idea behind these raster stacks is to test if and how the composition of these stacks influence, which spectral raster are in the end chosen as predictors (this is happening in step *5.7*, during correlation).

To solve the problem of homogeneity in broad areas in indices, a step for testing for homogeneity was implemented by the use `LEGION::detct_RstHmgy` function (*5.5*). The function works with `valueRange` which sets the value for in how much percentage the data is distributed in the raster. `THvalue` sets the threshold of homogeneity.

First `ALL_ind_stack_1` was tested with homogeneity thresholds between 0.4 and 0.9. The settings `valueRange = 0.1` and `THvalue = 0.9` led to drop exactly those three indices which demonstrated ample homogeneous areas: `VARI, RI` and `HI`. Thus they were removed statistically. These settings were applied also to `ALL_RGBMS_stack_1`, creating `ALL_RGBMS_stack_1_homog09` (and `ALL_ind_stack_1_homog09`).

To create different textural representations of the training data for the classification, different spatial filters are applied to the index stacks `ALL_ind_stack_1_homog09` and `ALL_RGBMS_stack_1_homog09`, using the `LEGION::filter_Stk` function (*5.6*). Filter enhance special properties of indices and the overall homogeneity of spectral areas and enhance the edges and between the different objects. The filters `sum, min, max, sd, mean, modal, sobel , sobel_hrzt` and `sobel_vert` have been implemented and used. How they work is described in the Help of the function. `ALL_ind_stack_1_homog09_filt` yielded **108** and `ALL_RGBMS_stack_1_homog09_filt` **144** filtered spectral indices.

The last step of the preparation of the spectral predictors for the classification is testing

the index stacks on correlation with the `LEGION::detct_RstCor` function (*5.7*). Testing on correlation between spectral indices is to support unbiased construction of predictors and to limit inter-dependency between spectral indices which are fed to the classification algorithm.
The fundamental difference between the two index stack is that `ALL_RGBMS_stack_1` compared to `ALL_ind_stack_1` also contains the RGB and IR spectral raster (as already mentioned above). Testing on correlation resulted in `clean_ALL_ind_stack_1_homog09_filt` with **18 layers** and `clean_ALL_RGBMS_stack_1_homog09_filt` with **19 layer**. Apart from three, respective four instances they contain the same indices. The differences are: In case of `clean_ALL_ind_stack_1_homog09_filt`: `GLI_modal3, GLI_sobel_v3, MSR_min3` and `MSR_sobel_v3`. In case of `clean_ALL_RGBMS_stack_1_homog09_filt`: `GLI_max3, NGRDI_sobel3, NGRDI_sobel_v3, Blue_min` and `Blue_sobel_v3`.
The correlation dropped the `Red, Green` and `IR` spectral raster. It was clear, that `Red` and `IR` would be correlate too much with each other. It is interesting to see, that with the addition of four spectral (non-index) raster, the composition of the remaining raster after a correlation test changed completely.

Another data preparation task, but independent from the preparation of the spectral data is the generation of training polygons. In other works shape files with examples of the spectral classes in training area 1 were prepared in QGIS. The different classes are: ***1-tree, 2-shrubs, 3-grass, 4-tree in shadow, 5-shadow*** and ***6-stone***.
Two training shapes were created, to compare a brute-force approach using minimal number of arbitrary sized and shaped coarse training polygons (1 polygon/class, `train_2`) to accurate, 3x3 pixel training polygons (`train_1`). `train_1` consists of 43 polygons, of which the classes ***1-tree***, ***4-tree_in shadow*** and ***5-shadow*** (which have a more complicated spectral signature and are especially interesting from the point of view of the project) are present 10 instances each, classes ***3-grass*** and ***2-shrubs*** present 5 instances each and class ***6-stone*** presents 3 instances (**Figure 11**).

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\kelto\Documents\detectATE\analysis\data\train\train_1.shp", layer:
## with 43 features
## It has 2 fields
## Integer64 fields read as strings:  id
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Users\kelto\Documents\detectATE\analysis\data\train\train_2.shp", layer:
## with 6 features
## It has 2 fields
## Integer64 fields read as strings:  id
```

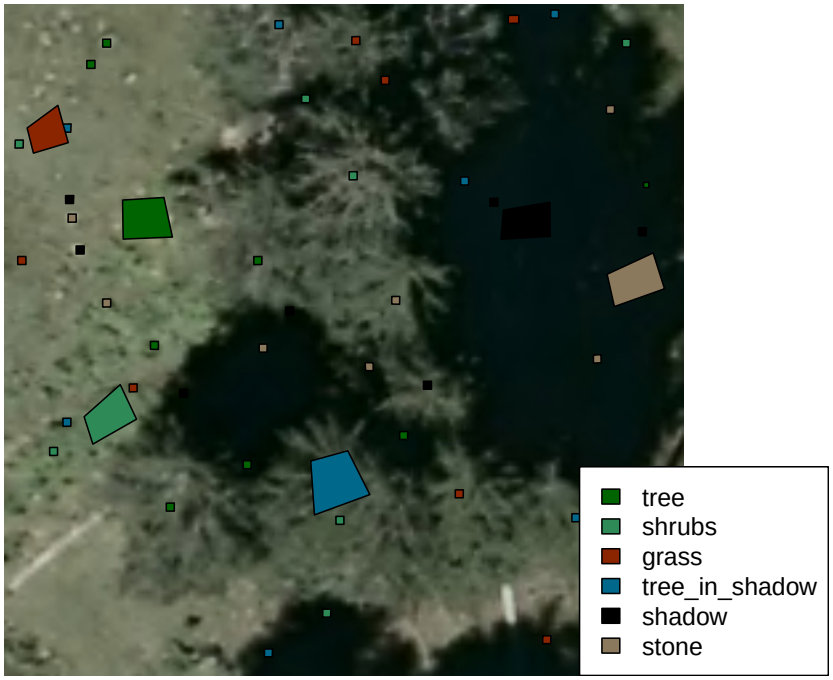

Figure 11: The training shapes train 1 and train 2 projected on RGB Aerial Imagery of test area 1.

3.8 Classification of test area 1

`6_classification_test_area_1.R`

The aim of this step is to find an accurate classification for test area 1 which can be then applied to the ROI.

The classification is performed using the `IKARUS package`. `IKARUS` is a wrapper for `CAST`, and it uses *'Random Forest'* as classification algorithm. *'Random Forest'* classifiers are supervised learning algorithms which consist of an ensemble of decision trees. Each (unrelated) decision tree is trained using a random subset of the training data. Each of these trees will give a prediction for a data point.Then,the prediction of all decision trees is averaged by a majority vote to a final prediction (**Figure 12**). The independence of the different decision trees increases

the accuracy of the prediction and also eliminates problems that can be caused by outliers in the data set and works also well with small data sets (Breiman 2001, Kuhn - Johnson 2013).

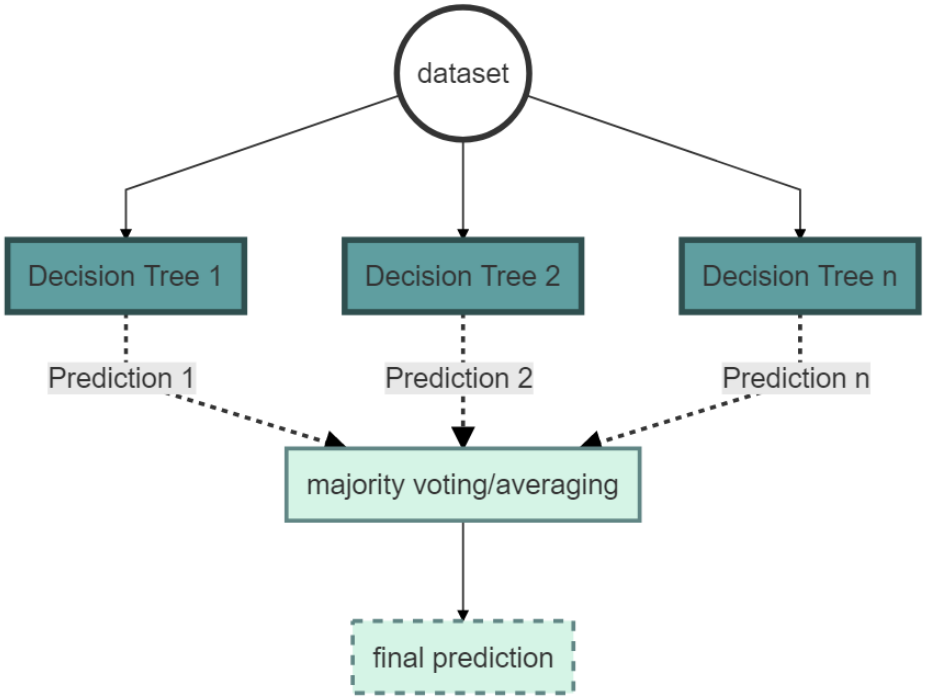

Figure 12: Work mechanism of the Random Forest Algorithm.

There are two important step before the actual machine learning can be started. One is the creation of meaningful **training data sets** and the other is the choice of most meaningful predictors and data dimension reduction.

The spectral predictors (`clean_ALL_ind_stack_1_homog09_filt/predictors 1` with **18 layers**, `clean_ALL_RGBMS_stack_1_homog09_filt/predictors 2` with **19 layers**) and the training polygons have been created in the previous steps. In the case of the spectral predictors all precautions were taken to create unbiased predictors (*5.5 - 5.7*) and in the case of the training polygons different amount, shape and size make them a good comparison of brute force and carefully considered polygons. These constitute the fundament of the two data sets which will serve as input for the machine learning algorithm.

First, values for all defined 6 classes have to be extracted from the predictors into a data frame. This is done using the `IKARUS::exrct_Traindat` function (*6.2.1*). By working with two training polygons and two spectral predictor stacks, altogether four **initial training data frames** (`train1_pred1`, `train1_pred2`, `train2_pred1` and `train2_pred2`) were created.

To accelerate and enhance the effectiveness of the machine learning algorithm, the most meaningful predictors are chosen by the `IKARUS::BestPredFFS` function from the just created four training data sets (*6.2.2*). This function is a wrapper for the `CAST::ffs` function, which uses **FFS**, that is *'Forward Feature Selection'* (Meyer et al. 2018) that is a data analysis method which identifies predictors which can lead to over-fitting in models. `IKARUS::BestPredFFS` calculates all possible models from the input training data set and looks for the model with the least number of predictors but the best performance . This model is the output of the `IKARUS::BestPredFFS`function (in our case `FFS_1 - FFS_4`). `FFS_1$selectedvars` accesses the list of best performing predictors. `IKARUS::BestPredFFS` also saves processing time by eliminating the less conclusive and possibly still biased predictors.

Once provided with the information of the best performing and most meaningful predictors selected by `IKARUS::BestPredFFS`, two **final training data sets** are prepared as input for the machine learning.
The first data set is the training data frame (`train_FFS_1 - train_FFS_4`) with the *response/dependent variable*, that is the `classes` we want to predict and the extracted values from the *independent variables* or `predictors` (by the shape files, in step 6.2.1), selected by `IKARUS::BestPredFFS`. It was created by extracting the chosen predictor columns and the class column from the initial data frame.
The second data set is the predictor stack (`predictors_FFS_1 - predictors_FFS_4`), which are selected predictors by `IKARUS::BestPredFFS`.

The machine learning model building and classification is executed by the function `IKARUS::RFclass`(*6.3*). It builds a model and performs a *'Random Forest'* classification in one step. These two procedures are often also carried out separately. `IKARUS::RFclass` is a wrapper for `caret::train` with 5-fold cross-validation and *'Random Forest'* as classification algorithm. The two **final training data sets** `train_FFS_1 - train_FFS_4` and `predictors_FFS_1 - predictors_FFS_4` are handed to this function. The results are the model used for the prediction and the prediction itself. THe results of the prediction are discussed in Chapter 4.

3.9 7. Data preparation for the Classification of ROI

7_data_prep_classification_ROI.R

Given the explanatory aim of this work, all steps which were applied to test area 1 were also applied to the ROI. For conciseness a workflow is outlined in the next two chapters and only differences to are going to be discussed in depth. The data preparation for the classification can be described with the following steps:

7.1 Load spectral data

7.2 Stack `RGB_ROI`, `IR_ROI` and `RGBNIR_ROI`

7.3 Compute Indices with `LEGION::vegInd_RGB` and `LEGION::vegInd_mspec`

7.4 Compute Index stacks `ALL_ind_stack_ROI`(15 layer) and `ALL_RGBMS_stack_ROI`(19 layer)

7.5 Test Index stack on homogeneity with `LEGION::detct_RstHmgy`

The indices `VARI`, `RI` and `HI` presented similarly ample homogeneous areas and the plan was to eliminate them out statistically, like in the case of test area 1. In the case of the ROI, the `THvalue` had to be raised to **0.93**, otherwise with **0.9** and **0.92** `VVI` too was dropped and with **0.95** only `VARI` and `RI` were dropped.

7.6 Filter both index stacks with `LEGION::filter_Stk`: `ALL_ind_stack_ROI_homog093_filt` (108 layer) and `ALL_RGBMS_stack_ROI_homog093_filt`(144 filter)

7.7 Test filtered index stacks on correlation with `LEGION::detct_RstCor`.

The two resulting index stacks are `clean_ALL_ind_stack_ROI_homog093_filt` (17 layer) and `clean_ALL_RGBMS_stack_ROI_homog093_filt` (17 layer).

3.10 8.Classification of ROI

8_Classification_ROI.R

Also in the case of the classification the same steps have been taken as in the case of test area 1. The steps are mentioned in the form of a workflow outline and only the differences are discussed in this chapter in depth.

8.1 Load spectral index stacks `clean_ALL_ind_stack_ROI_homog093_filt`, `clean_ALL_ind_stack_ROI_h` and `train_1` and `train_2`.

8.2 Prepare the spectral index stacks for the prediction.

8.2.1 Extract **initial training data frames** `train1_pred1`, `train1_pred2`, `train2_pred1`

and `train2_pred2` with `IKARUS::exrct_Traindat`.

8.2.2 *'Forward Feature Selection'* with `IKARUS::BestPredFFS`

8.2.3 Extract training data frame `train_FFS_1_ROI - train_FFS_4_ROI`

8.2.4 Stack final predictors `predictors_FFS1_ROI - predictors_FFS_4_ROI`

8.3 Create model and predict using `IKARUS::RFclass`

4 Results

In this chapter first the results of the Segmentation and the Classification of test area 1 and the ROI is presented. Then the validation of the Classification results by the Segmentation are explained.

4.1 Segmentation results of test area 1 and the ROI

4.1.1 Results of the segmentation of test area 1

During the development of the workflow the choice for the most successful Segmentation for test area 1 fell on number **4** of the segmentation results of both the first (yielding 40 results) and the second run (yielding 8 results) (*2.4.1, 2.4.2*). The choice was made 'based on numbers': `h=4`, `total segment = 20`(more than of segmentation result **8** in the second run) and it was simply assumed, that it is better to have equal `a` and `b` (`0.07, 0.07`). Only when the cross-validated Segmentation (*3.3*) gave almost the same results for all 8 Segmentation settings from `best_seg_vp_1_v2_filt` became clear, that it would be best to test all 8 Segmentation settings.

4.1.2 Results of the segmentation of the ROI

Comparing all eight Segmentation results in QGIS visually, `segm_ROI_8` delivers the best result. Although `segm_ROI_8` is not so accurate with the number and size/area of the individual trees, it is by far the most accurate segmentation which segments only trees and young trees and less shrubs/krummholz (*Figure 13*). The chosen Segmentation `segm_ROI_8`(**Figure 12**) yielded **78% accuracy** (*3.3*). It definitely could be improved by e.g. choosing smaller steps of `h`.

```
## Reading layer `segm_ROI_8' from data source
##   `C:\Users\kelto\Documents\detectATE\analysis\results\segm\segm_ROI\segm_ROI_8.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 786 features and 4 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -40847 ymin: 215899.4 xmax: -40595.5 ymax: 216089.5
## Projected CRS: MGI_Austria_GK_West_with_axis_order_normalized_for_visualization
```

Figure 13: The training shapes train 1 and train 2 projected on RGB Satellite Imagery of test area 1.

4.2 Classification results of test area 1 and ROI

4.2.1 Results of test area 1

When inspecting the prediction results, the first thing that catches the eye is that **predictions 3 and 4** (*Figure 14*) yield the same results. When we look at the result of `IKARUS::BestPredFFS`, the *'Forward Feature Selection'*, it becomes clear, that the stacks consist of the same predictors (`NDVI_min3` and `TGI_max3`). If we go back one step to the spectral predictors (`clean_ALL_ind_stack_1_homog09_filt/predictors 1` and `clean_ALL_RGBMS_stack_1_homog09_filt/predictors 2`) created in step *5.7*, then we can recall, that they differed by 4 and 5 predictors. The predictors remaining after the *'Forward Feature Selection'* were not of those which differed in the two spectral predictor set, but those which were present in both predictor sets. **Predictions 3 and 4** were carried out with training data sets which were created using the training shape file `train_2`, which consists of minimal number of arbitrary sized and shaped coarse training polygons. This means, that the big training shapes contained pixels which in fact belonged to more than one class. This made made the classes less disparate and thus the differing 4 and 5 predictors did not deliver meaningful information for the best model and were sorted out by the *'Forward Feature*

Selection'.

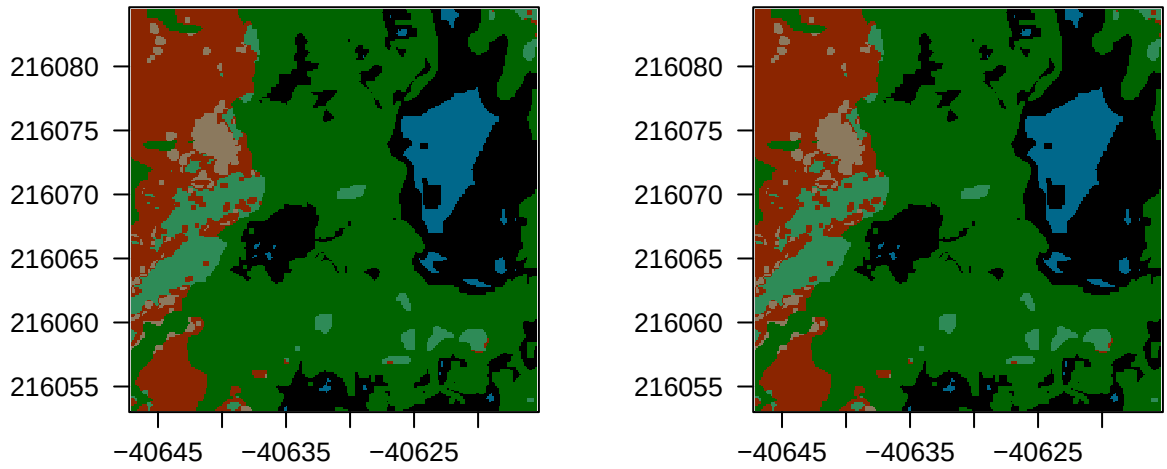

Figure 14: Prediction 3 and 4 of test area 1. Note the equivalence of the result. The traning shape 2 provided unclear information of the class of the pixels and the spectral predictor stack for both predictions was equal.

Predictions 1 and 2 (*Figure 15*) on the other hand were created using the accurate, 3x3 pixel training polygons of the shape file `train_1` and mainly differ in respect of the amount of polygons in the respective classes. **Prediction 1** is more accurate when compared to the distribution of the actual pixels. In **Prediction 2** more pixels were classified as tree instead of shrubs and grass. This is due to the fact that although the resolution of the pixels is 12.5 cm, the spectral representation of classes ***tree***, ***shrub*** and ***grass*** is quite similar and these classes are hard to distinguish. This applies even more the classes ***shadow*** and ***tree in shadow***. Considering all classes and their reflection in the classification, **Prediction 1** is delivers the best results.

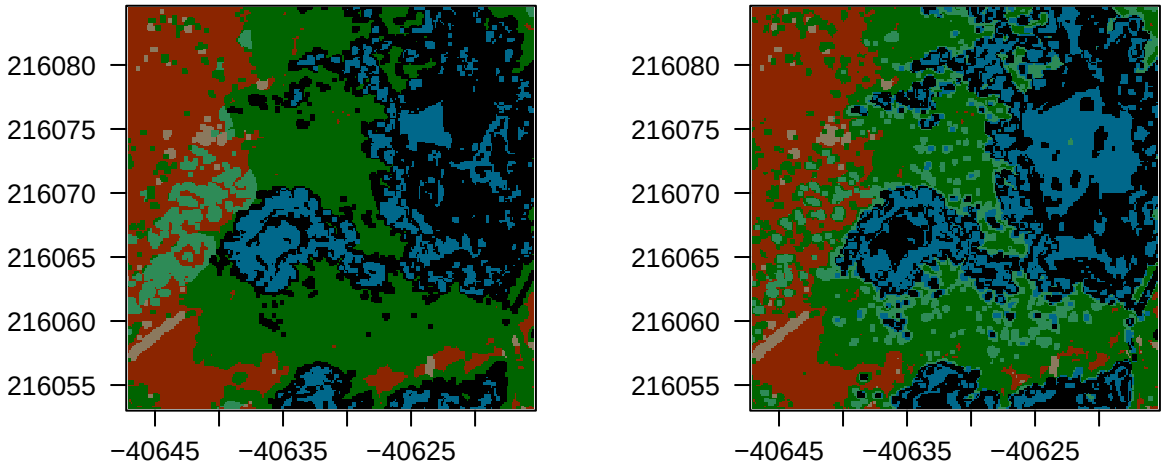

Figure 15: Classification results of ROI.

4.2.2 Results of ROI

The four Prediction results form a slow but sure degradation of results when reflected upon from Prediction 1 towards Prediction 4. For a better perception all Predictions are plotted separately. Similarly to test area 1, also in the case of the Prediction of the ROI the already noticed differences of the two training shape files occur. Unlike before (in the case of test are 1 the two FFS spectral stacks were the same), in this case the difference appear in the shift in the amount of classified pixels per class. All four Predictions contain misclassifications, on the grounds of what has already been mentioned regarding test area 1: although the resolution of the pixels is 12.5 cm (way higher than that of Satellite data), the spectral representation of the classes ***tree***, ***shrub*** and ***grass*** is quite similar and thus they are hard to distinguish. This applies even more to the classes ***shadow*** and ***tree in shadow***.

Prediction 4 (*Figure 16*) is the most 'extreme' of the Predictions, classifying too much pixels as class ***tree***. Most of class ***grass*** was classified as ***tree*** and ***shrubs***, although their spectral signature quite distinct. On the other hand class ***shrub*** is in its spectral signature similar to that of class ***tree***, but it is more classes ***grass***, ***tree*** and ***stone*** that are classified as ***shrub*** and not only class ***tree***. Class ***tree in shadow*** is clearly misrepresented apart from a few instances. This is due to the strong spectral similarity to class ***shadow***. Before drawing any conclusions, let's investigate **Prediction 3**!

Prediction 3 (*Figure 17*) still bears an over classification as class ***tree***, even more so than in the case of **Prediction 4**, in fact ***grass*** is again classified as ***tree***, ***shrubs*** and ***grass***, ***tree*** and ***stone*** is classified as ***shrubs***. The prediction of class ***shadow*** is again fairly accurate. On the other hand class ***tree in shadow*** is represented better, but still not good enough.

Based on Prediction 4 and 3 we can underline what was understood regarding test area 1. In the case of Prediction 3 and 4 the training shape 2 was used with the arbitrary shapes. Polygons in the training shape are meant to represent pixels of a certain class, but in the case of `train 2` the polygons contain pixels of multiple classes and it is a gamble how many classes and how many pixels of each classes are represented in the polygons. The majority voting of the 'Random Forest' enforces this 'randomness' effect and this is reflected in the Predictions.

Prediction 2 (*Figure 18*) captures class ***tree*** almost to the same extent as Prediction 1, apart from small incongruences. Class ***tree in shadow*** is on the other hand over classified: when

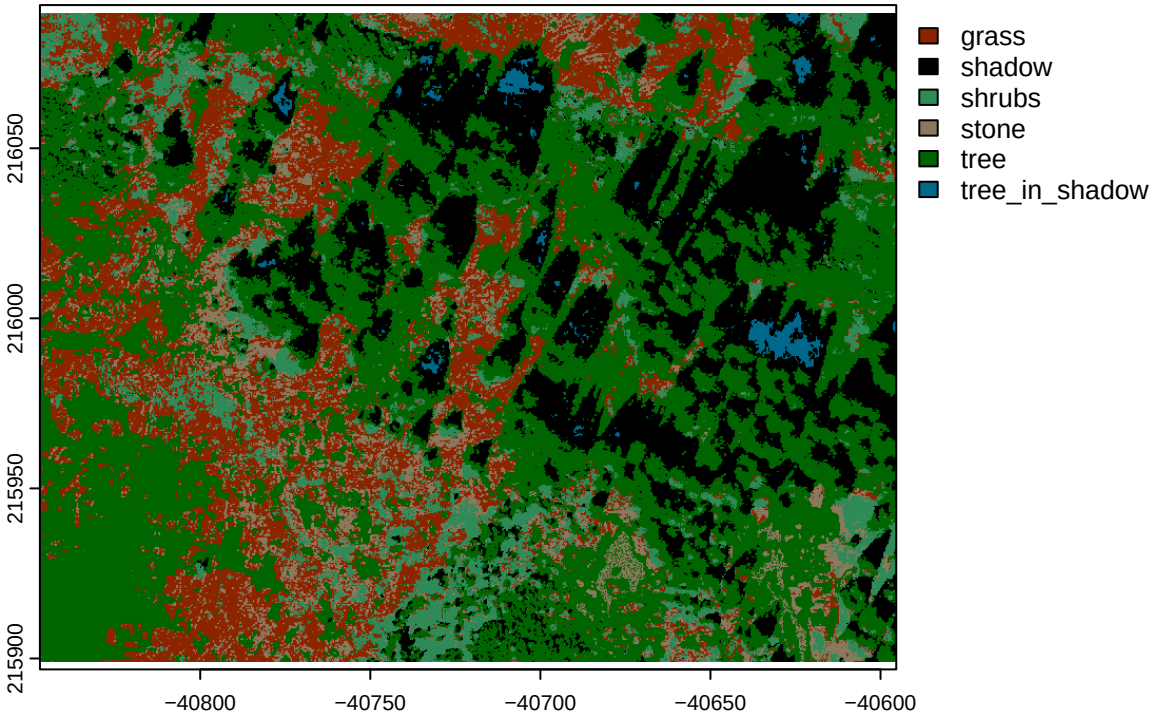

Figure 16: Classification results of ROI.

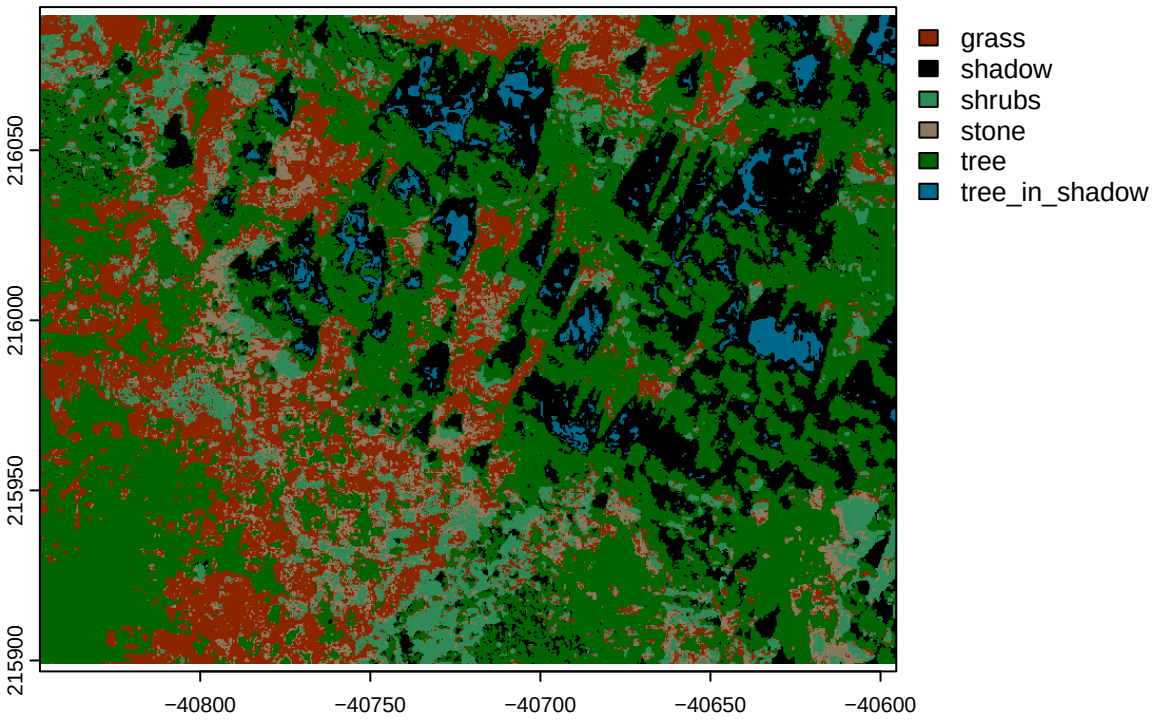

Figure 17: Classification results of ROI.

comparing to the RGB Aerial Image of the ROI, it is visible that much more pixels were classified as class ***tree in shadow*** than as class ***shadow***. The spectral predictors chosen by the `FFS` contain probably very similar values for class ***shadow*** and ***tree in shadow***. Further classes ***stone*** and ***shrubs*** are also over classified: **grass** is frequently classified as **shrubs** and the area around the actual stones is also classified as **stones**, thus a bit amplified.

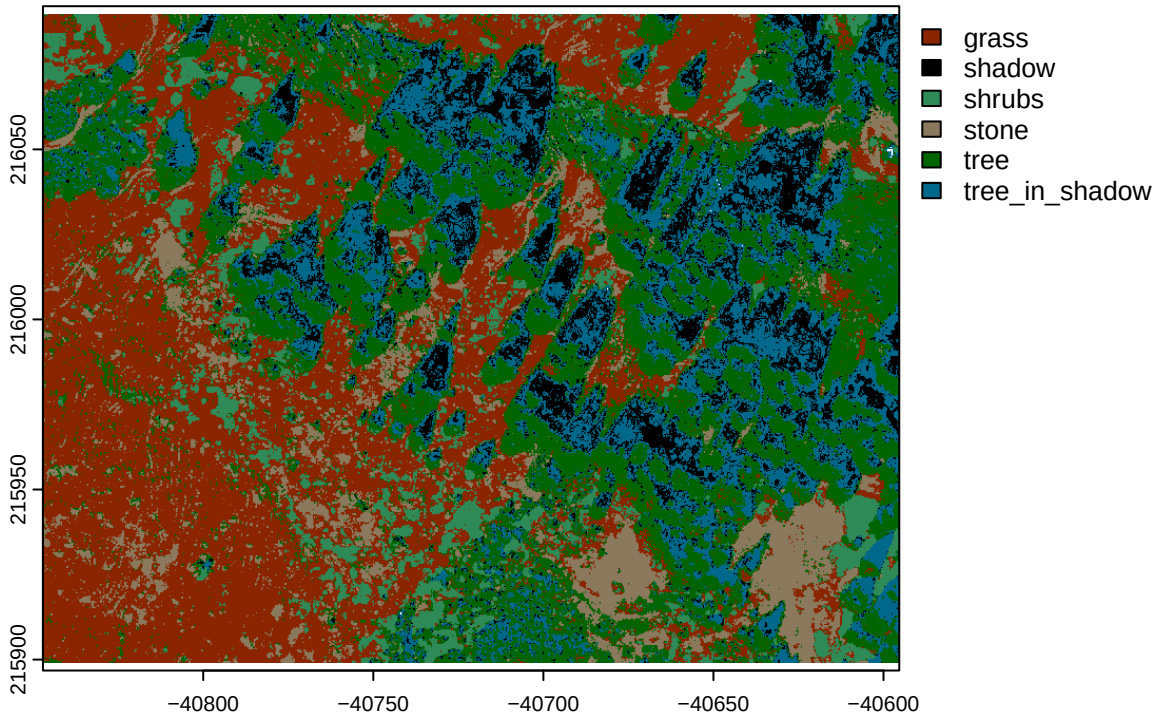

Figure 18: Classification results of ROI.

Prediction 1 (*Figure 19*) delivers the best result of all the Predictions of the ROI. There are clear misclassifications of class ***shrub*** as class ***tree*** and of class ***shadow*** as class ***tree in shadow***.

Based on the experiences of the classification of ***shadow*** and ***tree in shadow*** in the **Predictions 4, 3 and 2**, it can be said, that both classes together are constant in the Predictions (in the RGB Aerial Imagery it is visible that that is the area of trees in shadow and the shadow cast by the trees); it only differs how the two classers are distributed. When inspecting the classification of trees in all Predictions of the ROI it is noticeable, that their shadow (to the NE) has a certain halo of pixels of class ***tree***. The explanation of this phenomenon is that a

so-called '*optical edge-effect*' occurs: the spectral value of the pixels surrounding the shadow of the trees in about 1 pixel radius form a spectral transition from class ***shadow*** black pixels to the surrounding other, mainly class ***grass*** lighter pixels. This same effect occurs on the border from trees to grass: the spectral value of the 1 pixel radius pixels between the two class becomes spectrally mixed and thus falls into class ***shrub*** which corresponds to the spectral value of ***shrub*** pixels. This is an optical problem which occurs when registering the spectral band with the specific cameras and as visible, has a strong effect on the classification results.

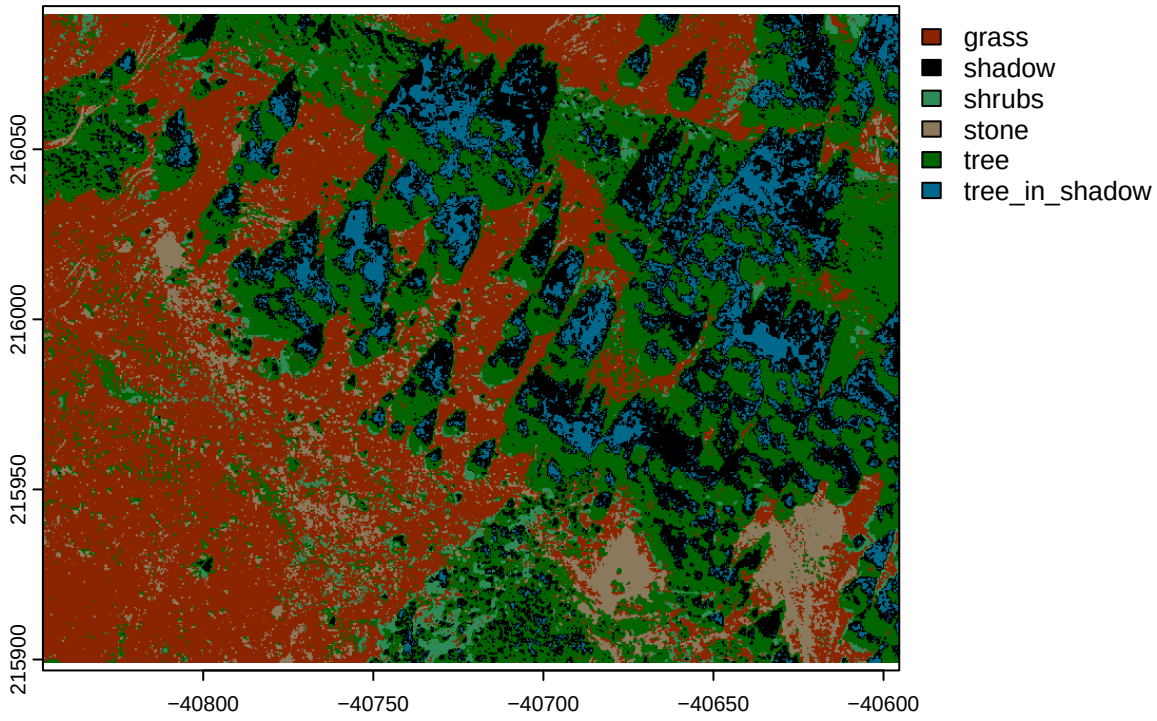

Figure 19: Classification results of ROI.

4.3 Validation of test area 1 and ROI

`9_validation_test_area_1_ROI.R`

Before looking at the results of the statistical validation of the Segmentation and the Classification of the ROI, we have to bear in mind, that these two methods work completely differently and both have their own (technical) limitations and thus their statistical comparison should be accepted with reservation. However the actual decision of the success of either method should

not depend on the result of the validation.

Let's compare the two results visually to be able to understand what the results of the validation mean:

```
## Reading layer `segm_ROI_8' from data source
##   `C:\Users\kelto\Documents\detectATE\analysis\results\segm\segm_ROI\segm_ROI_8.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 786 features and 4 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: -40847 ymin: 215899.4 xmax: -40595.5 ymax: 216089.5
## Projected CRS: MGI_Austria_GK_West_with_axis_order_normalized_for_visualization
```

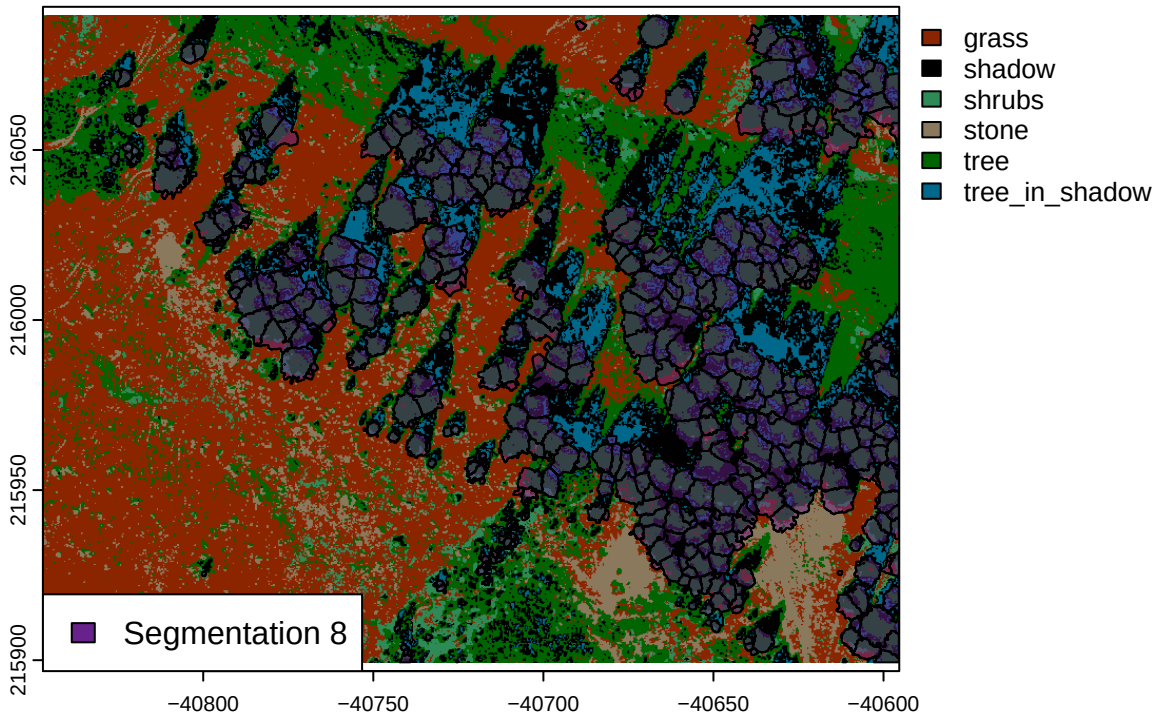

Figure 20: Prediction of the ROI overlayed by the Segmentation of the ROI.

The validation of the Classification with the Segmentation is carried out with the function **IKARUS::classSegVal**. The function first reclassifies the six classes of a Prediction into two - classes tree and not tree. In the next step the Segmentation is rasterized. Then the two

results are compared using cell statistics. In essence: the amount the pixels of class ***tree*** in the Segmentation is compared with the amount of pixels of class ***tree*** in the Prediction. Further the possibility exists to reclassify any given class as ***tree*** and to involve it in the cell statistics. This was implemented because pixels of class ***tree in shadow*** are technically part of trees even if they are spectrally different.

The result is a raster file containing the reclassified classification to tree and not tree (including the eventually reclassified pixels) and a short table is printed with the following results: `nclass`, `nseg`, `overclass`, `underclass`, `hit`, `hitrate`, `rate underclass` and `rate overclass`.

`nclass` - gives a value for amount of pixels of class ***tree*** in the Classification (including reclassification)

`nseg` - gives a value for the amount of pixels of class ***tree*** in the rasterized Segmentation

`overclass` - gives a value for pixels occurring in the Classification but not in the rasterized Segmentation

`underclass` - gives a value for pixels occurring in the rasterized Segmentation but not in the Classification

`hit` - gives a value for pixels occurring both in the Classification and in the rasterized Segmentation

`hitrate` - gives a % value for pixels occurring both in the Classification and in the rasterized Segmentation

`rate underclass` - gives a % value for pixels occurring in the rasterized Segmentation but not in the Classification

`rate overclass` - gives a % value for pixels occurring in the classification but not in the rasterized Segmentation pixels

`validation score` - prints together the `hitrate` (in %) @ `rate overclass` (in %) + `rate underclass` (in %)

4.3.1 Results for test area 1

4.4 Validation of pixels of class *tree*

valdiation score: 0.8093 @ 0.6656

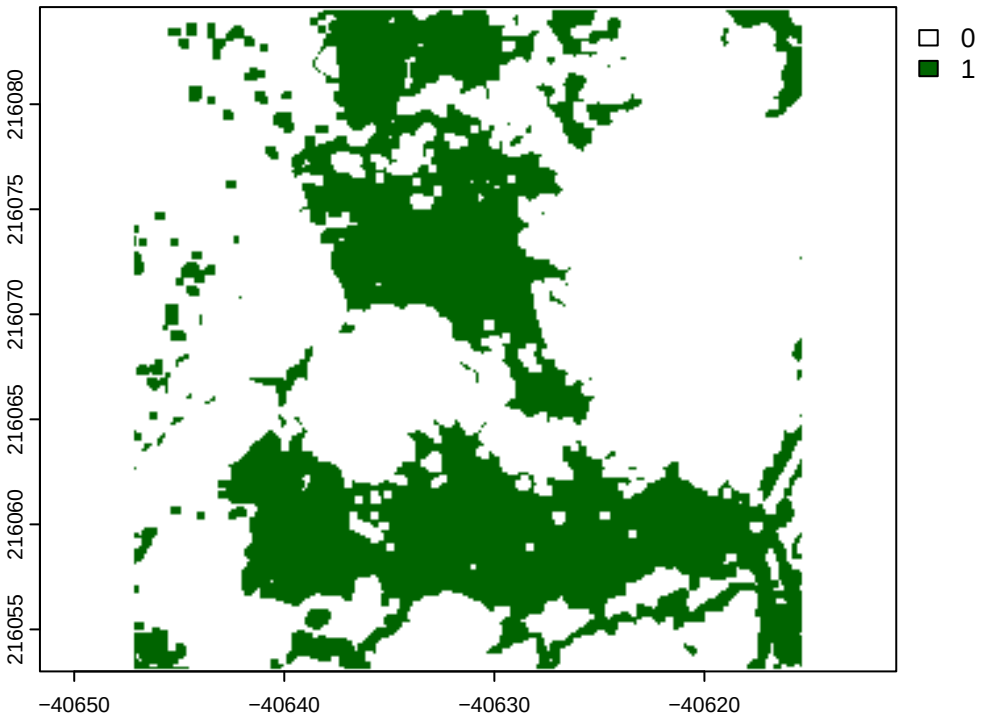

Figure 21: Prediction of the ROI overlayed by the Segmentation of the ROI.

nclass	nseg	overclass	underclass	hit	hitrate	rate underclass	rate overclass
22012	33930	4197	16115	17815	0.8093	0.4749	0.1907

The validation score means, that 0.8093 % of the pixels occur both in the Classification and in the rasterized Segmentation; 0.1907 % of pixels occur in the classification but not in the rasterized Segmentation pixels (over classification) and 0.4749 % value of pixels occur in the rasterized Segmentation but not in the Classification (under classification).

4.5 Validation of pixels of class *tree + tree in shadow*

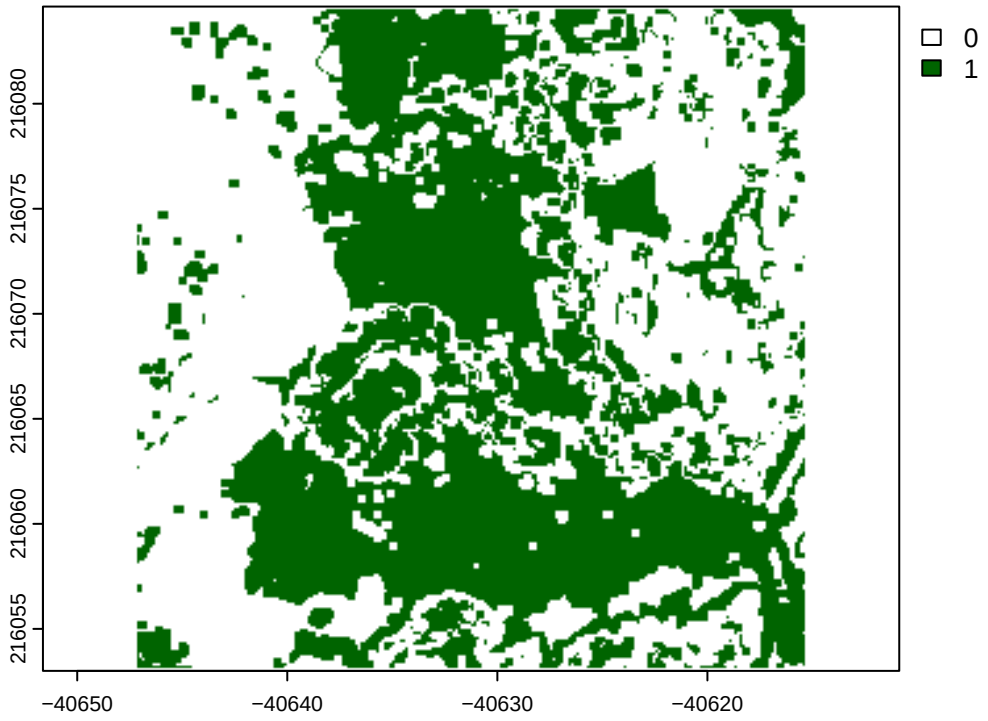

Figure 22: Prediction of the ROI overlayed by the Segmentation of the ROI.

valdiation score: 0.7593 @ 0.5776

nclass	nseg	overclass	underclass	hit	hitrate	rate underclass	rate overclass
29633	33930	7133	11430	22500	0.7593	0.3369	0.2407

The validation score means, that 0.7593 % of the pixels occur both in the Classification and in the rasterized Segmentation; 0.2407 % of pixels occur in the classification but not in the rasterized Segmentation pixels (over classification) and 0.3369 % value of pixels occur in the rasterized Segmentation but not in the Classification (under classification).

Comparing the two validations it is visible, that when class ***tree*** is reclassified with pixels of class ***tree in shadow***, the hitrate and under classification decreases but the over classification increases, thus impairing the Classification of only class ***tree*** when compared to the result of the Segmentation.

When class ***shadow*** is included in the reclassification, then the validation score is even lower: 0.6544 @ 0.4264

4.5.1 Results for ROI

4.6 Validation of pixels of class *tree*

valdiation score: 0.4411 @ 1.0089

nclass	nseg	overclass	underclass	hit	hitrate	rate underclass	rate overclass
886769	711102	495632	319965	391137	0.4411	0.45	0.5589

The validation score means, that 0.4411 % of the pixels occur both in the Classification and in the rasterized Segmentation; 0.5589 % of pixels occur in the classification but not in the rasterized Segmentation pixels (over classification) and 0.45 % value of pixels occur in the rasterized Segmentation but not in the Classification (under classification).

4.7 Validation of pixels of class *tree + tree in shadow*

valdiation score: 0.4222 @ 0.9187

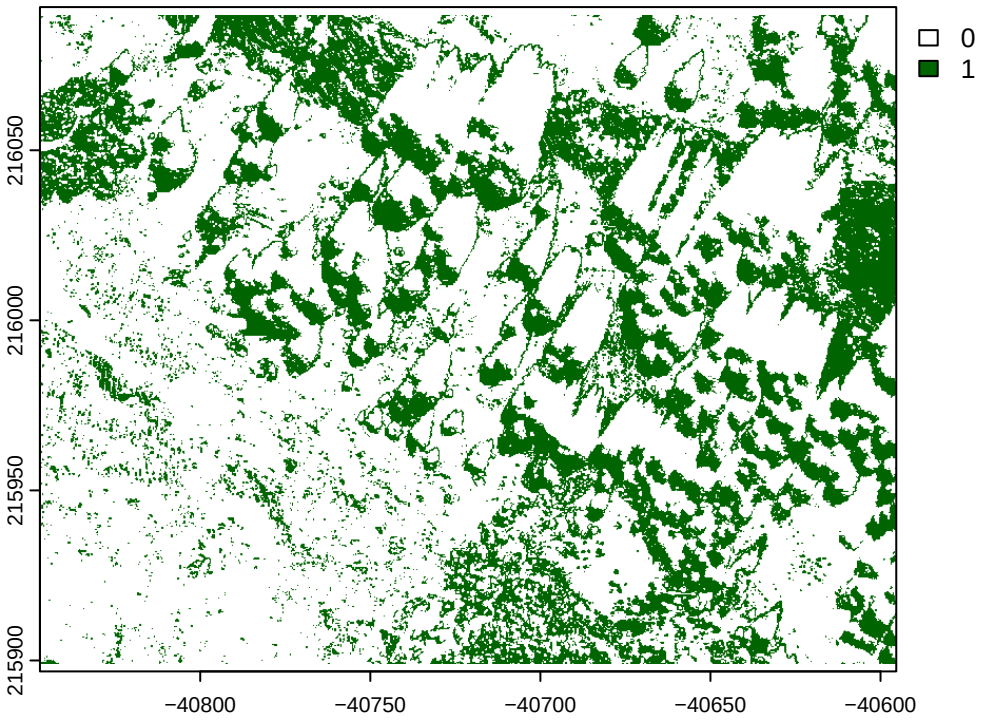

Figure 23: Prediction of the ROI overlayed by the Segmentation of the ROI.

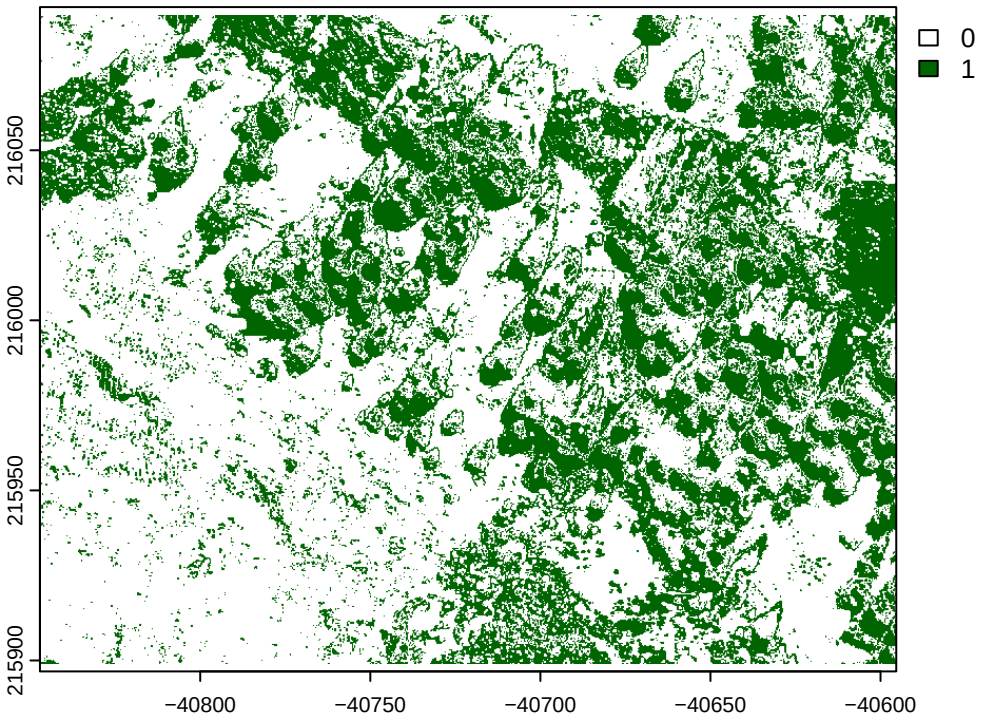

Figure 24: Prediction of the ROI overlayed by the Segmentation of the ROI.

nclass	nseg	overclass	underclass	hit	hitrate	rate underclass	rate overclass
1110145	711102	641439	242396	468706	0.4222	0.3409	0.5778

The validation score means, that 0.4222 % of the pixels occur both in the Classification and in the rasterized Segmentation; 0.5778 % of pixels occur in the classification but not in the rasterized Segmentation pixels (over classification) and 0.3409 % value of pixels occur in the rasterized Segmentation but not in the Classification (under classification).

Comparing the two validations it is visible, that when class ***tree*** is reclassified with pixels of class ***tree in shadow***, the hitrate and under classification decreases but the over classification increases, thus impairing the Classification of only class ***tree*** when compared to the result of the Segmentation.

When class ***shadow*** is included in the reclassification, then the validation score is even lower: 0.3917 @ 0.6899

It has to be said that it makes no real sense to thoroughly compare the segmentation results with the Aerial Imagery - rather the IR Imagery. The segmentation is based on the Canopy Height Model and LiDAR data and the RGB and IR Imagery on spectral data. E.g. in the IR Imagery you can see the young trees and seedling in a lighter red.

5 Discussion

- Do different spectral raster stacks influence the choice of *'Forward Feature Selection'* with `IKARUS::BestPredFFS`?
- Do different training shapes (number of polygons and their size) influence *'Forward Feature Selection'* with `IKARUS::BestPredFFS`

homogeneity - to exclude VARI, RI and HI correlation test FFS

CAST::plot_ffs(FFS1)

test area 1 It is interesting to see, that with the addition of four spectral (non-index) raster, the composition of the remaining raster after a correlation test changed completely. But based on the shape polygon size after the FFS the spectral predictor sets done with train 2 yielded the same meaningful predictors

Comparing `FFS_1_ROI - FFS_4_ROI` with `FFS_1 - FFS_4` we can see the following investigation the predictors (order based on frequency):

FFS	PREDICTOR1	PREDICTOR2	PREDICTOR3	PREDICTOR4	PREDICTOR5
FFS_1	TGI_max3	VVI_min3			
FFS_1_ROI	TGI_max3	VVI_min3			
FFS_2	TGI_max3	VVI_min3			
FFS_2_ROI	TGI_max3		GLI_min3	VVI_sobel3	VVI_modal3
FFS_3	TGI_max3				
FFS_3_ROI	TGI_max3		GLI_min3	VVI_sobel3	VVI_modal3
FFS_4	TGI_max3				
FFS_4_ROI	TGI_max3	VVI_min3	GLI_min3		

FFS	PREDICTOR6	PREDICTOR7	PREDICTOR8	PREDICTOR9	PREDICTOR10
FFS1		MSR_min3			
FFS_1_ROI		MSR_min3			
FFS_2			Blue_min3	GLI_max3	
FFS_2_ROI					SI_max3

FFS	PREDICTOR6	PREDICTOR7	PREDICTOR8	PREDICTOR9	PREDICTOR10
FFS_3	NDVI_min3				
FFS_3_ROI					
FFS_4	NDVI_min3				
FFS_4_ROI			Blue_min3		

10 different index predictors were chosen by *'Forward Feature Selection'* from the correlated indices in general for classification, out of the 18/19 of area 1 and the 17 of the ROI. In the case of test area 1 this means five different index predictors, compared to the ROI where eight different ones were founds meaningful. `FFS2_ROI` and `FSS3_ROI` consist of almost the same predictors save `SI_max3` in favor of the latter.

FFS_1 and FFS2 were created using the accurate, 3x3 pixel training polygons of the training shape file `train_1`, while FFS_3 and FFS_4 were created using the training shape file `train_2`, which consists of minimal number of arbitrary sized and shaped coarse training polygons.

PREDICTORS	train_1 (FFS1&2)	train_2 (FFS3&4)
TGI_max3	ta1, ta2 & ROI1, ROI2	ta3, ta4 & ROI3, ROI4
VVI_min3	ta1, ta2 & ROI1	ROI4
GLI_min3	ROI2	ROI3, ROI4
VVI_sobel3	ROI2	ROI3
VVI_modal3	ROI2	ROI3
NDVI_min3		ROI3, ROI4
MSR_min3	ta1, ROI1	
Blue_min3	ta2	ROI4
GLI_max3	ta2	
SI_max3	ROI2	

It is safe to say, that `VVI_min3` is more often in the combination of training shape `train_1` and `GLI_min3` is more often in combination of `train_2`. `VVI_sobel3`, `VVI_modal3` and `Blue_min3` occur equally in combination with training shapes `train_1` and `train_2`. Only two predictors

stick clearly to different training shapes: NDVI_min3to train_2 and MSR_min3 to train_1. GLI_max3 and SI_max3 are also associated with train_1.

Turning the investigation to the relation of the areas to the predictor indices: four are used in the prediction of both areas:

AREA	PREDICTOR1	PREDICTOR2	PREDICTOR7	PREDICTOR8
test area 1	TGI_max3(4x)	VVI_min3(FFS1&2)	Blue_min3(FFS2)	MSR_min3(FFS1)
ROI	TGI_max3(4x)	VVI_min3(FFS1&4)	Blue_min3(FFS4)	MSR_min3(FFS4)

TGI_max3 is the only index predictor, which was selected in association of both area nad both traning shapes. It seems to be robust enough to create meaningful models with shape train_2 to be used for prediction.

Otherwise only VVI_min3 was present both in the first predictor stack of both the training area and the ROI.

When inspecting the remaining predictors, the following can be observed:

AREA	PREDICTOR3	PREDICTOR4	PREDICTOR5
test area 1			
ROI	GLI_min3(FSS2,3&4)	VVI_sobel3(FSS2&3)	VVI_modal3(FFS2&3)

AREA	PREDICTOR6	PREDICTOR9	PREDICTOR10
test area 1	NDVI_min3(FFS3&4)	GLI_max3(FFS1)	
ROI			SI_max3(FFS2)

GLI_min3, VVI_sobel3, VVI_modal3 and SI_max3 are only in index predictor stacks for the ROI and NDVI_min3 and GLI_max3 only for those of test area 1. TGI_max3,VVI_min3, Blue_min3 and MSR_min3 are associated with both areas.

In connection with the FFS and the training shapes the question arises how do different training shapes (number of polygons and their size) influence prediction?

training area 1 **Predictions 3 and 4** were carried out with training data sets which were created using the training shape `train_2`, which consists of minimal number of arbitrary sized and shaped coarse training polygons. This means, that using the coarse big polygons the differing 4 and 5 predictors did not give crucial more information but made the classes less disparate.

comparing varImp of test area a ROI

6 Conclusion and Outlook

Both the results of the PBIA and the GeOBIA are valid, based on the type of data set. The detection and mapping of trees is a robust information base for future analysis of ATEs They deliver a robust information base for future analysis for patch-based metrics and even landscape metrics

For the spectra data: even with the 12,5 cm resolution the spectral signature of certain classes is very similar; thus a better spectral resolution of the classes is needed, such a Hyperspectral data

Hamendianfar et al. (Hamedianfar et al., 2022) DeepForest https://deepforest.readthedocs.io/en/latest/landing.html Egli article

7 References

Allen, T. R., & Walsh, S. J. (1996). Spatial and compositional pattern of alpine treeline, Glacier National Park, Montana. *Photogrammetric Engineering and Remote Sensing (USA)*. https://scholar.google.com/scholar_lookup?title=Spatial+and+compositional+pattern+of+alpine+treeline%2C+Glacier+National+Park%2C+Montana&author=Allen%2C+T.R.+%28University+of+Vermont%2C+Burlington%2C+VT.%29&publication_year=1996

Bader, M. Y., Llambí, L. D., Case, B. S., Buckley, H. L., Toivonen, J. M., Camarero, J. J., Cairns, D. M., Brown, C. D., Wiegand, T., & Resler, L. M. (2021). A global framework for linking alpine-treeline ecotone patterns to underlying processes. *Ecography*, *44*(2), 265–292. https://doi.org/10.1111/ecog.05285

Bader, M. Y., Rietkerk, M., & Bregt, A. K. (2007). Vegetation Structure and Temperature Regimes of Tropical Alpine Treelines. *Arctic, Antarctic, and Alpine Research*, *39*(3), 353–364. https://doi.org/10.1657/1523-0430(06-055)%5BBADER%5D2.0.CO;2

Bader, M. Y., & Ruijten, J. J. A. (2008). A topography-based model of forest cover at the alpine tree line in the tropical Andes. *Journal of Biogeography*, *35*(4), 711–723. https://doi.org/10.1111/j.1365-2699.2007.01818.x

Baker, W. L., Honaker, J. J., & Wiesberg, P. J. (1995). *Using aerial photography and GIS to map the forest-tundra ecotone in Rocky Mountain National Park, Colorado, for global change research.* https://scholar.google.com/citations?view_op=view_citation&hl=en&user=kgaHKiwAAAAJ&alert_preview_top_rm=2&citation_for_view=kgaHKiwAAAAJ:qjMakFHDy7sC

Baker, W., & Weisberg, P. (1997). Using GIS to model tree population parameters in the Rocky Mountain National Park forest–tundra ecotone. *Journal of Biogeography*, *24*(4), 513–526. https://doi.org/10.1111/j.1365-2699.1997.00130.x

Barredo Cano, J., Mauri, A., & Caudullo, G. (2020). *Impacts of climate change in European mountains: Alpine tundra habitat loss and treeline shifts under future global warming : JRC PESETA IV project : Task 8.* (No. JRC115186). Publications Office of the European Union. https://data.europa.eu/doi/10.2760/653658

Bonanomi, G., Rita, A., Allevato, E., Cesarano, G., Saulino, L., Di Pasquale, G., Allegrezza, M., Pesaresi, S., Borghetti, M., Rossi, S., & Saracino, A. (2018). Anthropogenic and environmental factors affect the tree line position of Fagus sylvatica along the Apennines (Italy). *Journal of Biogeography*, *45*(11), 2595–2608. https://doi.org/10.1111/jbi.13408

Brown, D. G. (1994). Predicting Vegetation Types at Treeline Using Topography and Biophysical Disturbance Variables. *Journal of Vegetation Science*, *5*(5), 641–656. https://doi.org/10.2307/3235880

Chhetri, P. K., Shrestha, K. B., & Cairns, D. M. (2017). Topography and human disturbances are major controlling factors in treeline pattern at Barun and Manang area in the Nepal Himalaya. *Journal of Mountain Science*, *14*(1), 119–127. https://doi.org/10.1007/s11629-016-4198-6

Chhetri, P. K., & Thai, E. (2019). Remote sensing and geographic information systems techniques in studies on treeline ecotone dynamics. *Journal of Forestry Research*, *30*(5), 1543–1553. https://doi.org/10.1007/s11676-019-00897-x

Elliott, G. P. (2017). Treeline Ecotones. In D. Richardson, A. Castree, M. F. Goodchild, A. Kobayashi, W. Liu, & R. A. Marston (Eds.), *The International Encyclopedia of Geography* (pp. 1–10.). Wiley & Sons. https://doi.org/10.1002/9781118786352.wbieg0539

Fricker, G. A., Ventura, J. D., Wolf, J. A., North, M. P., Davis, F. W., & Franklin, J. (2019). A Convolutional Neural Network Classifier Identifies Tree Species in Mixed-Conifer Forest from Hyperspectral Imagery. *Remote Sensing*, *11*(19, 19), 2326. https://doi.org/10.3390/rs11192326

Geping Luo, & Li Dai. (2013). Detection of alpine tree line change with high spatial resolution remotely sensed data. *Journal of Applied Remote Sensing*, *7*(1), 1–14. https://doi.org/10.1117/1.JRS.7.073520

Hamedianfar, A., Mohamedou, C., Kangas, A., & Vauhkonen, J. (2022). Deep learning for forest inventory and planning: A critical review on the remote sensing approaches so far and prospects for further applications. *Forestry: An International Journal of Forest Research*, cpac002. https://doi.org/10.1093/forestry/cpac002

Harsch, M. A., & Bader, M. Y. (2011). Treeline form – a potential key to understanding

treeline dynamics. *Global Ecology and Biogeography*, *20*(4), 582–596. https://doi.org/10.1111/j.1466-8238.2010.00622.x

Harsch, M. A., Hulme, P. E., McGlone, M. S., & Duncan, R. P. (2009). Are treelines advancing? A global meta-analysis of treeline response to climate warming. *Ecology Letters*, *12*(10), 1040–1049. https://doi.org/10.1111/j.1461-0248.2009.01355.x

Holtmeier, F.-K., & Broll, G. (2005). Sensitivity and response of northern hemisphere altitudinal and polar treelines to environmental change at landscape and local scales. *Global Ecology and Biogeography*, *14*(5), 395–410. https://doi.org/10.1111/j.1466-822X.2005.00168.x

Holtmeier, F. K., & Broll, G. E. (2007). Treeline advance - driving processes and adverse factors. *Landscape Online*, *1*, 1–33. https://doi.org/10.3097/LO.200701

Immitzer, M., Atzberger, C., & Koukal, T. (2012). Tree Species Classification with Random Forest Using Very High Spatial Resolution 8-Band WorldView-2 Satellite Data. *Remote Sensing*, *4*(9, 9), 2661–2693. https://doi.org/10.3390/rs4092661

Kimball, K. D., & Weihrauch, D. M. (2000). Alpine vegetation communities and the alpine-treeline ecotone boundary in New England as biomonitors for climate change. *In: McCool, Stephen F.; Cole, David N.; Borrie, William T.; O'Loughlin, Jennifer, Comps. 2000. Wilderness Science in a Time of Change Conference—Volume 3: Wilderness as a Place for Scientific Inquiry; 1999 May 23–27; Missoula, MT. Proceedings RMRS-P-15-VOL-3. Ogden, UT: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. P. 93-101*, *015*. http://www.fs.usda.gov/treesearch/pubs/21974

Koerner, C. (2012). Was steuert das Pflanzenwachstum? *Biologie in unserer Zeit*, *42*(4), 238–243. https://doi.org/10.1002/biuz.201210484

Körner, C. (1998). A re-assessment of high elevation treeline positions and their explanation. *Oecologia*, *115*(4), 445–459. https://doi.org/10.1007/s004420050540

Körner, C., & Paulsen, J. (2004). A world-wide study of high altitude treeline temperatures. *Journal of Biogeography*, *31*(5), 713–732. https://doi.org/10.1111/j.1365-2699.2003.01043.x

Král, K. (2009). Classification of Current Vegetation Cover and Alpine Treeline Ecotone in the Pradĕd Reserve (Czech Republic), Using Remote Sensing. *Mountain Research and*

Development, *29*(2), 177–183. https://doi.org/10.1659/mrd.1077

Kupková, L., Červená, L., Suchá, R., Jakešová, L., Zagajewski, B., Březina, S., & Albrechtová, J. (2017). Classification of Tundra Vegetation in the Krkonoše Mts. National Park Using APEX, AISA Dual and Sentinel-2A Data. *European Journal of Remote Sensing*, *50*(1), 29–46. https://doi.org/10.1080/22797254.2017.1274573

Meyer, F., & Beucher, S. (1990). Morphological segmentation. *Journal of Visual Communication and Image Representation*, *1*(1), 21–46. https://doi.org/10.1016/1047-3203(90)90014-M

Meyer, H., Ludwig, M., Reudenbach, C., Nauss, T., & Pebesma, E. (2022). *CAST: 'Caret' Applications for Spatial-Temporal Models* (Version 0.6.0) [Computer software]. https://CRAN.R-project.org/package=CAST

Middleton, M., Närhi, P., Sutinen, M., & Sutinen, R. (2008). *Object based change detection of historical aerial photographs reveals altitudinal forest expansion.* https://www.semanticscholar.org/paper/OBJECT-BASED-CHANGE-DETECTION-OF-HISTORICAL-AERIAL-Middleton-N%C3%A4rhi/373b86df6f5979767ef6adbcfcc0b987fa37693f

Mishra, N. B., Mainali, K. P., Shrestha, B. B., Radenz, J., & Karki, D. (2018). Species-Level Vegetation Mapping in a Himalayan Treeline Ecotone Using Unmanned Aerial System (UAS) Imagery. *ISPRS International Journal of Geo-Information*, *7*(11, 11), 445. https://doi.org/10.3390/ijgi7110445

Mohapatra, J., Singh, C. P., Tripathi, O. P., & Pandya, H. A. (2019). Remote sensing of alpine treeline ecotone dynamics and phenology in Arunachal Pradesh Himalaya. *International Journal of Remote Sensing*, *40*(20), 7986–8009. https://doi.org/10.1080/01431161.2019.1608383

Myneni, R. B., Keeling, C. D., Tucker, C. J., Asrar, G., & Nemani, R. R. (1997). Increased plant growth in the northern high latitudes from 1981 to 1991. *Nature*, *386*(6626, 6626), 698–702. https://doi.org/10.1038/386698a0

Plowright, A., & Roussel, J.-R. (2021). *ForestTools: Analyzing Remotely Sensed Forest Data* (Version 0.2.5) [Computer software]. https://CRAN.R-project.org/package=ForestTools

Qiu, L., Jing, L., Hu, B., Li, H., & Tang, Y. (2020). A New Individual Tree Crown Delineation

Method for High Resolution Multispectral Imagery. *Remote Sensing*, *12*(3, 3), 585. https://doi.org/10.3390/rs12030585

Ranson, K. J., Montesano, P. M., & Nelson, R. (2011). Object-based mapping of the circumpolar taiga–tundra ecotone with MODIS tree cover. *Remote Sensing of Environment*, *115*(12), 3670–3680. https://doi.org/10.1016/j.rse.2011.09.006

Resler, L. M., Fonstad, M. A., & Butler, D. R. (2004). Mapping the Alpine Treeline Ecotone with Digital Aerial Photography and Textural Analysis. *Geocarto International*, *19*(1), 37–44. https://doi.org/10.1080/10106040408542297

Roffler, P. (2020). *Step by Step: Developing a Geographic Object-Based Image Analysis Workflow for the Terraced Landscape of the Lower Engadine, Switzerland | Student Repository* [MSc, Leiden University]. https://studenttheses.universiteitleiden.nl/handle/1887/136416

Singh, C. P., Mohapatra, J., & Dharaiya, N. A. (2015). Remote Sensing of alpine Treeline Dynamics. *ISG Newsletter*, *21*(4), 3–8. https://www.researchgate.net/publication/283837476_Remote_Sensing_of_Alpine_Treeline_Dynamics

Stueve, K. M., Isaacs, R. E., Tyrrell, L. E., & Densmore, R. V. (2011). Spatial variability of biotic and abiotic tree establishment constraints across a treeline ecotone in the Alaska Range. *Ecology*, *92*(2), 496–506. https://doi.org/10.1890/09-1725.1

Virtanen, T., Mikkola, K., Nikula, A., Christensen, J. H., Mazhitova, G. G., Oberman, N. G., & Kuhry, P. (2004). Modeling the Location of the Forest Line in Northeast European Russia with Remotely Sensed Vegetation and GIS-Based Climate and Terrain Data. *Arctic, Antarctic, and Alpine Research*, *36*(3), 314–322. http://www.jstor.org/stable/1552640

Weinstein, B. G., Marconi, S., Bohlman, S. A., Zare, A., & White, E. P. (2020). Cross-site learning in deep learning RGB tree crown detection. *Ecological Informatics*, *56*, 101061. https://doi.org/10.1016/j.ecoinf.2020.101061

Weinstein, B. G., Marconi, S., Bohlman, S., Zare, A., & White, E. (2019). Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks. *Remote Sensing*, *11*(11, 11), 1309. https://doi.org/10.3390/rs11111309

Whiteside, T. G., Esparon, A. J., & Bartolo, R. E. (2020). A semi-automated approach for

quantitative mapping of woody cover from historical time series aerial photography and satellite imagery. *Ecological Informatics*, *55*, 101012. https://doi.org/10.1016/j.ecoinf.2019.101012

Winings, C. J. (2013). Mapping alpine treeline with high resolution imagery and LiDAR data in North Cascades National Park, Washington. *WWU Graduate School Collection*. https://doi.org/10.25710/9zkh-h696