Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

Integration einer grafischen Benutzeroberfläche zum zeitreihenbasierten Monitoring von (Privat-) Wald in Rheinland-Pfalz mittels Google Earth Engine

vorgelegt von

Anna Göbel
106672, UNIGIS MSc Jahrgang 2020

Betreuer/in:
Prof. Dr. Thomas Udelhoven

Zur Erlangung des Grades
„Master of Science – MSc”

Kalenborn, 12.07.2022

Danksagung

Ich bedanke mich bei Prof. Dr. Thomas Udelhoven und im Besonderen bei Sascha Nink für die Betreuung dieser Arbeit. Vielen Dank für die vielen konstruktiven Gespräche, die fachliche Unterstützung und das offene Ohr. Ich bedanke mich außerdem bei meinen Kolleg*innen für das Verständnis und die Rücksichtnahme in stressvollen Zeiten. Vor allem bedanke ich mich bei meinem Freund, meiner Familie und meinen Freund*innen für die aufbauenden Worte und die seelische Unterstützung.

Abstract

Rheinland-Pfalz ist eines der waldreichsten Bundesländer, dessen Wald zu mehr als einem Viertel der Fläche in Privatbesitz liegt. Da der Privatwald nicht Teil der regelmäßigen Inventur ist, existieren zu dessen Zustand keine belastbaren Daten. In den Jahren 2018 bis 2020 kam es vermehrt zu Walddynamiken in Rheinland-Pfalz. Neben klimawandelinduzierten Schäden wie Borkenkäferkalamitäten traten auch Kahlschläge, Neupflanzungen und weitere Dynamiken auf. Im Rahmen dieser Arbeit wurde untersucht, inwiefern sich die cloudbasierte Plattform Google Earth Engine (GEE) eignet, diese Dynamiken zu detektieren. Es wurde eine GUI erstellt, mit der für ausgewählte Flächen Zeitreihen zu den Flächenanteilen der Waldklassen in den Frühjahren der Jahre 2018 bis 2022 ausgegeben werden können. Die pixelbasierte, überwachte Klassifizierung erfolgte mithilfe der Machine Learning-Algorithmen CART, RF und GTB anhand der Vegetationsindizes NDVI und TC auf Basis von Sentinel-2-Daten. Dabei wurde zwischen Laubwald, Nadelwald und Nicht-Wald unterschieden. Insgesamt lag die overall accuracy (OA) zwischen 69 und 86 Prozent, wobei die Kombination TC und GTB die validesten Ergebnisse lieferte. Es konnte festgestellt werden, dass sich GEE mit einigen Einschränkungen für diese Klassifizierung eignet, da die begrenzte Rechenkapazität pro User dazu führt, dass Variablenwerte wie der Größe des Trainingsdatensatzes reduziert werden mussten.

Rhineland-Palatinate is one of the most densely forested German ‘Länder’, with more than a quarter of its forest area being in private ownership. Since the privately owned forest is not part of the regular forest inventory, reliable data on its state does not exist. Between 2018 to 2020, increased forest dynamics were registered in Rhineland-Palatinate. Besides climate change-induced damages (e.g., bark beetle calamities), clearcuts, tree planting and other dynamics occurred. In the context of this work, the extent to which the cloud-based platform Google Earth Engine (GEE) is suitable to detect these dynamics was analyzed. A GUI aimed at providing time series of forest class area distributions for selected areas during spring seasons between 2018 to 2022 was created. A pixel-based supervised classification using the machine learning algorithms CART, RF, and GTB was performed based on the vegetation indices NDVI, and TC derived from Sentinel-2 data. A distinction was made between broadleaf forest, coniferous forest, and non-forest. Finally, the overall accuracy (OA) ranged from 69 to 86 percent, with the combination of TC and GTB providing the most valid results. It was found that GEE is suitable for this form of classification. However, some limitations were detected since the limited computational capacity per user results in the need to reduce variable values such as the size of the training dataset.

Inhaltsverzeichnis

1	**Einleitung**	**1**
2	**Datengrundlage und Methode**	**4**
2.1	Datengrundlage	5
2.2	Verwendete Software	9
2.3	Vegetationsindizes	11
2.3.1	NDVI	11
2.3.2	Tasseled Cap-Transformation	11
2.4	Klassifikationsalgorithmen	15
2.4.1	Classification and Regression Trees	18
2.4.2	Random Forest	20
2.4.3	Gradient Tree Boost	21
2.5	Methode	22
2.5.1	Klassifizierung	25
2.5.2	Validierung	29
2.5.3	Grafische Benutzeroberfläche	33
3	**Ergebnisse und Diskussion**	**35**
3.1	Ergebnisse der Validierung	35
3.2	Zeitreihendarstellung	39
3.3	Diskussion	47
4	**Fazit und Ausblick**	**49**

Abbildungsverzeichnis

Abbildung 1: Darstellung der Methodik 5
Abbildung 2: Als Nicht-Wald gelabelte Flächen aus Forsteinrichtungsdaten (WöFIS) 7
Abbildung 3: Stichprobennetz der BWI (BMEL 2022) 8
Abbildung 4: GEE-API (code.earthengine.google.com) 10
Abbildung 5: Dimensionen TC-*greenness* und -*brightness* (CRIST und CICONE 1984: 259) 12
Abbildung 6: Ergebnisse TC (Laubwald) 14
Abbildung 7: Ergebnisse TC (Nadelwald) 14
Abbildung 8: Ergebnisse TC (Wiese) 14
Abbildung 9: Verteilung von Laub- (grün) und Nadelwald (blau, NINK et al. 2019: 8) 18
Abbildung 10: Classification Tree nach BREIMAN et al. 1984: 2 19
Abbildung 11: Modell des Random Forest-Algorithmus (eigene Darstellung nach HO 1995) 21
Abbildung 12: *Bagging* (unabhängige Vorhersagen, RF) vs. *boosting* (sequenzielle Vorhersagen, GTB) (PAL 2020) 22
Abbildung 13: GUI der Klassifizierung 26
Abbildung 14: GUI der Validierung 30
Abbildung 15: Berechnung des IoU-Verhältnisses (ESRI o. J.) 32
Abbildung 16: Initialdarstellung der GUI 33
Abbildung 17: Zeitreihendarstellung (hier: Gemeindeebene) 34
Abbildung 18: Ergebnistabelle (OA je VI und MLA) 36
Abbildung 19: Gemeinde Achtelsbach – Klassifikationsergebnis 2022 und Zeitreihe 41
Abbildung 20: Gemeinde Hillscheid – Klassifikationsergebnis 2022 und Zeitreihe 42
Abbildung 21: Ausschnitt des Klassifikationsergebnisses der Gemeinde Körperich (Lärchenwald gelb markiert) 43
Abbildung 22: Ausschnitt S2-Szene nach Anwendung der Wolkenmaske 45
Abbildung 23: Klassifikationsergebnis der Stadt Cochem (links: 2021, rechts: 2022) 46
Abbildung 24: Zeitreihe der Stadt Cochem 46

Tabellenverzeichnis

Tabelle 1: Bänder der S2-Satelliten (ESA o. J.) 6
Tabelle 2: Koeffizienten nach CRIST und CICONE (1984) 13
Tabelle 3: Asset-Datensätze 24
Tabelle 4: Zeitspannen der S2-Daten der jeweiligen Jahre 27
Tabelle 5: Werte des S2A-SCL-Bands (GOOGLE DEVELOPERS o. J.) 27
Tabelle 6: Ergebnis IoU (Flächenangaben in ha) 39

Abkürzungsverzeichnis

API	Application Programming Interface (Programmierschnittstelle)
ATKIS	Amtliches Topografisch-Kartografisches Informationssystem
BKG	Bundesamt für Kartografie und Geodäsie
BMEL	Bundesministerium für Ernährung und Landwirtschaft
BWI	Bundeswaldinventur
CART	Classification and Regression Trees
GEE	Google Earth Engine
GTB	Gradient Tree Boost
GUI	Graphical User Interface (Grafische Benutzeroberfläche)
IDE	Integrated Development Environment (Integrierte Entwicklungsumgebung)
IoU	Intersect over Union (Jaccard-Index)
LVermGeo	Landesamt für Vermessung und Geobasisinformation
LWI	Landeswaldinventur
MLA	Machine Learning-Algorithmen
MSI	Multispectral Imagers
MTP	Minimum Turning Point
NDVI	Normalized Difference Vegetation Index
o. J.	ohne Jahresangabe
OA	overall accuracy
OBIA	object-based image analysis
PA	producer’s accuracy
PBIA	pixel-based image analysis
RF	Random Forest
RLP	Rheinland-Pfalz
S2	Sentinel-2-Satellitensystem
SCL	Scene Classification
SWIR	short wavelength infrared (Kurzwellen-Infrarot)
TC	Tasseled Cap Transformation
UA	user’s accuracy
VI	Vegetationsindex
VNIR	visible and near infrared
WöFIS	Waldökologisches Forstinformationssystem

1 Einleitung

Rheinland-Pfalz (RLP) ist mit 42,3 % der Landesfläche eines der waldreichsten Bundesländer in Deutschland (SCHAEFER, VANVOLXEM 2016: 2). Wald besitzt die Fähigkeit, große Massen an Kohlenstoffdioxid zu binden, ist Lebensraum für unzählige Tier- und Pflanzenarten und erfüllt wichtige Funktionen wie Lärmschutz, Temperaturausgleich (Erhöhung der Luftfeuchtigkeit), Bodenschutz (Erosionsschutz), Immissionsschutz (Luftfilterung und -reinigung) und Erholung (BÜRGER-ARNDT 2013: 26ff.). Der Wald gilt aber nicht nur im Sinne des Klimaschutzes als eines der bedeutendsten Ökosysteme. Er stellt auch eine wichtige wirtschaftliche Ressource für die Waldbesitzenden dar. Um alle Interessen zu bedienen, muss eine nachhaltige und naturnahe Bewirtschaftung erfolgen. Dies ist vor allem in Privatwäldern, die in Rheinland-Pfalz 26,7% der Waldflächen ausmachen (SCHAEFER, VANVOLXEM 2016: 3), oft nicht der Fall. Zudem unterliegen Privatwälder keinen regelmäßigen Inventuren, sodass keine belastbaren Daten zu diesen Flächen vorliegen. Um den Waldzustand in Privatwäldern überhaupt einschätzen zu können, wurde von NINK et al. (2019) eine Methode zum satellitenbildbasierten Monitoring von Privatwald in Rheinland-Pfalz, dem Saarland und Luxemburg entwickelt, an der sich diese Arbeit orientiert. Die Methode wird in dieser Arbeit auf die Daten der Jahre 2018-2022 angewendet, wodurch eine Zeitreihe abgebildet werden kann. Dabei wird der Wald pro Jahr in Laub-, Nadel- und Nicht-Wald klassifiziert, wobei reale Daten aus Wäldern außerhalb des Privatwaldes verwendet und auf die gesamte Waldfläche in Rheinland-Pfalz übertragen werden (NINK et al. 2019). Neben dem Privatwald besteht der Wald in Rheinland-Pfalz zu 27,2% aus Staatswald, also Wald im Eigentum der Länder und des Bundes, und zu den restlichen 46,1% aus Körperschaftswald, wobei zu den Waldbesitzenden Gemeinden, Städte und sonstige Körperschaften gehören.

Es soll ein Tool entwickelt werden, mit dem die gewonnenen Daten zentral und individuell abgerufen und ausgewertet werden können. Für die Erstellung eines solchen Tools eignet sich Google Earth Engine (GEE), die in der Vergangenheit bereits mehrfach für solche Methoden herangezogen wurde (vgl. XULU et al. 2020, HUANG et al. 2021). Dieses Tool in Form einer grafischen Benutzeroberfläche/Graphical User Interface (GUI) soll es insbesondere für Privatwaldbesitzende und Behörden möglich machen, den Zustand und die Dynamik ihrer Wälder einzusehen und zu analysieren und Entscheidungen über anstehende Maßnahmen zu treffen. Es kann eine zeitreihenbasierte Auswertung der Veränderungen einzelner Waldflächen in Rheinland-Pfalz erfolgen, z.B. eine Analyse, wo sich Flächen ohne Änderungen und kahl

geschlagene Flächen befinden und welche Jahre betroffen sind. Die GUI soll außerdem die Möglichkeiten bieten, zum einen weitere Funktionen zu integrieren, z.B. die Anzeige von Flächen mit Schäden durch Borkenkäferbefall oder die Einbindung von Baumartenauswertungen, und zum anderen den Untersuchungsraum auf andere naturräumlich ähnliche Gebiete innerhalb der Großregion zu erweitern.

Grund dafür, dass für immer mehr Fragestellungen in der passiv-optischen Fernerkundung und auch für diese Arbeit auf GEE als Tool zurückgegriffen wird, ist, dass bei der Datenauswahl, -prozessierung und -darstellung normalerweise eine Vielzahl von Hürden überwunden werden muss. Eine der größten Hürden ist das grundlegende IT-Management: Datenerfassung und -speicherung, Parsing unbekannter und unterschiedlicher Dateiformate, Verwaltung von Datenbanken, Auswahl lokaler Maschinen für Prozessierung, Arbeitsspeicherkapazitäten sowie die Verwendung einer Vielzahl von Frameworks für die Geodatenverarbeitung. Mit der cloudbasierten Lösung des Google-Konzerns fallen solche Hürden weg, da alle Berechnungen innerhalb dieses einen Systems stattfinden können und keine eigenen Ressourcen benötigt werden. Außerdem bietet es die Möglichkeit, die Ergebnisse leichter mit anderen Forschenden, politischen Entscheidungsträgern oder der Öffentlichkeit in Form der Veröffentlichung des Programmcodes oder als interaktive Webanwendungen zu teilen, ohne eine Expertise für Anwendungsentwicklung, Webprogrammierung oder HTML haben zu müssen (GORELICK et al. 2017: 2f.).

Das Ziel der vorliegenden Arbeit ist es herauszufinden, inwiefern sich GEE eignet, die Methode nach NINK et al. (2019) umzusetzen und diese auf die Jahre 2018-2022 zu übertragen, um daraus eine jährliche Karte zu erstellen. Im Gegensatz zum Threshold-basierten Ansatz bei NINK et al. (2019) wird in dieser Arbeit auf Machine Learning-Algorithmen (MLA) als Klassifizierungsmethode zurückgegriffen, da diese in GEE bereits implementiert sind. Dabei müssen verschiedene Parameter im Prozess geändert und die Ergebnisse verglichen werden. Folgende Forschungsfragen werden im Rahmen der Arbeit beantwortet:

1. Welcher Vegetationsindex (VI) liefert die besseren Ergebnisse auf Grundlage von Sentinel-2 (S2) -Daten für Wälder in Rheinland-Pfalz?

2. Mithilfe welches MLA können die Daten am besten in Laub-, Nadel- und Nicht-Wald klassifiziert werden?

3. Inwiefern können die Klassifikationsergebnisse mithilfe GEE in Echtzeit ausgewertet und abgerufen werden?

Zur Beantwortung der Forschungsfragen wird zunächst der Klassifizierungsprozess in GEE implementiert. Dabei werden für die Jahre 2018-2022 die optimalen S2-Bilder gesucht, daraus die beiden VI Normalized Difference Vegetation Index (NDVI) und Tasseled Cap (TC) -*brightness*, -*wetness* und -*greenness* berechnet und mithilfe der drei Klassifikationsalgorithmen Classification and Regression Trees (CART), Random Forest (RF) und Gradient Tree Boost (GTB) in die Klassen Laubwald, Nadelwald und Nicht-Wald klassifiziert. Die Ergebnisse aller Kombinationen werden anschließend statistisch ausgewertet. Die besten Ergebnisse der jeweiligen Jahre werden schließlich über die grafische Benutzeroberfläche abrufbar gemacht. In der Benutzeroberfläche können sich die User entweder für ganze Gemeinde- oder Forstamtsgebiete oder für individuell gezeichnete Flächen Auswertungen über die Waldtypen in der jeweiligen Fläche ausgeben lassen.

PONGANAN et al. (2021), die sich beim Untersuchungsgebiet auf Reisanbauflächen in Thailand beschränkten und andere Vegetationsindizes verwendeten, kamen in ihrer Arbeit zu dem Ergebnis, dass GTB von allen drei MLA die validesten Ergebnisse und CART die geringsten Genauigkeitswerte lieferte. Für deren Anwendungsfall konnten bei der PBIA-Methode der MLA GTB eine OA von 82%, für RF eine OA von 77% und für CART eine Genauigkeit von 73% erreicht werden (PONGANAN et al. 2021: 36). Auch SUJUD et al. (2021) konnten in ihrer Arbeit zur Klassifikation der Landnutzung und Cannabis-Detektion in GEE feststellen, dass GTB gegenüber CART und RF validere Ergebnisse erzielte. Für die Klassifikationen auf Grundlage der MLA erzielte GTB eine Genauigkeit von 93%, wobei sie bei RF und CART bei 92%, bzw. 87% lag (SUJUD et al. 2021: 9).

Diese Arbeit erfolgte in Zusammenarbeit mit dem Fachbereich VI – Umweltfernerkundung und Geoinformatik der Universität Trier (Prof. Dr. Thomas Udelhoven und Sascha Nink), deren Arbeit Teil des Projekts RegioWood II ist. Regiowood II ist ein von der EU finanziertes Umweltprojekt, das sich mit dem Monitoring der Wälder in der Großregion SaarLorLux (Saarland, Rheinland-Pfalz, Wallonien, Lothringen und Luxemburg) beschäftigt (RESSOURCES NATURELLES DÉVELOPPEMENT ASBL 2018, vgl. STOFFELS et al. 2015, NINK et al. 2019).

2 Datengrundlage und Methode

Bei der Erstellung der vorliegenden Arbeit lag das Hauptaugenmerk auf der Implementierung in GEE und dem Vergleich der Ergebnisse verschiedener Parameteränderungen im Prozess, um den Nutzenden der GUI ein möglichst valides Endprodukt zu liefern. Es wurden verschiedene Datenquellen verwendet, die zum einen Teil zur freien Verwendung und kostenlos verfügbar sind (S2-Daten, Luftbilder, Verwaltungsgrenzen) und zum anderen Teil von der Forstbehörde des Landes Rheinland-Pfalz zum Zwecke dieser Arbeit zur Verfügung gestellt wurden (Trainingsdaten, Validierungsdaten, Waldmaske).

Im ersten Schritt wurden die benötigten Daten gesammelt, vorverarbeitet (QGIS) und in der GEE-Entwicklungsumgebung (Application Programming Interface (API)) entweder direkt verwendet oder als sog. Asset eingebunden. Danach wurden anhand der Satellitendaten zwei verschiedene Vegetationsindizes berechnet (NDVI und TC). Anschließend wurden die Resultate in die Klassen Nadelwald, Laubwald und Nicht-Wald klassifiziert, wobei die vorverarbeiteten Daten aus dem waldökologischen Forstinformationssystem (WöFIS) als Trainingsdatensatz dienten. Für die Klassifizierung wurden drei verschiedene MLA verwendet. Die Validierung erfolgte durch den Vergleich mit punktuellen Daten der Landeswaldinventur (LWI). Die Ergebnisse wurden dann statistisch ausgewertet, um herauszufinden, welche der verschiedenen Klassifizierungsmethoden das Ergebnis mit der höchsten Genauigkeit lieferte. Dieses Ergebnis fand innerhalb der GEE-API zur Implementierung der GUI Verwendung, um damit eine Zeitreihe zu erstellen. Für die Zeitreihe wurden S2-Daten der Jahre 2018 bis 2022 herangezogen. Abbildung 1 stellt den Prozess von der Auswahl der Daten bis zur Zeitreihenanalyse grafisch dar. In den nachfolgenden Kapiteln werden die verwendeten Datengrundlagen und die Implementierung in GEE eingehend beschrieben.

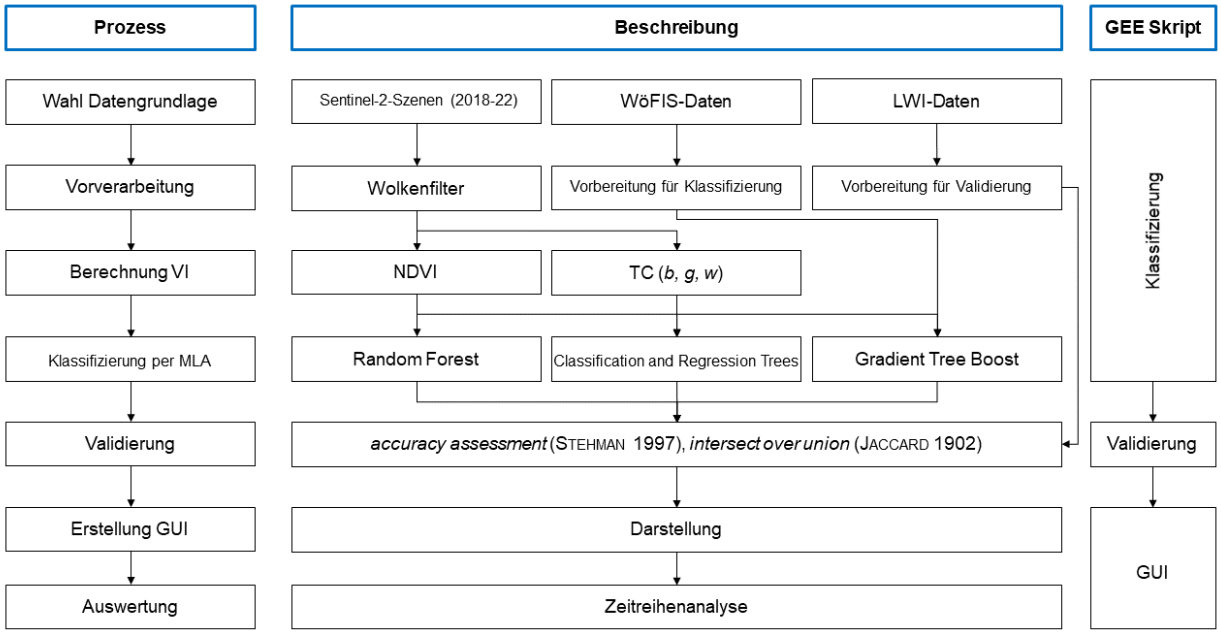


Abbildung 1: Darstellung der Methodik

2.1 Datengrundlage

Als Hauptdatengrundlage wurden multispektrale Daten der S2-Satelliten verwendet. Die Sentinel-Satelliten sind Kernelement des Copernicus-Programms, eine gemeinsame Initiative der Europäischen Union, der Europäischen Raumfahrtagentur (ESA), der Europäischen Organisation für meteorologische Satelliten (EUMETSAT) und deren jeweiligen Mitgliedsstaaten (DLR o. J.), deren Bilder zur freien Verfügung verwendbar sind. S2 besteht aus zwei polarumlaufenden Satelliten (S2A und S2B) mit einer kombinierten Wiederholungsrate von 5 Tagen. Die Multispectral Imagers (MSI) der beiden Satelliten liefern 13 Bänder in einem Wellenlängenbereich von ca. 432 (ultra blue) bis ca. 2290 nm (short wavelength infrared (SWIR)). Die räumliche Auflösung variiert dabei je nach Band zwischen 10 und 60 Metern (ESA o. J., s. Tabelle 1).

Sentinel-2A wurde 2015 in Betrieb genommen, zwei Jahre später folgte die Inbetriebnahme des Sentinel-2B-Satelliten. In GEE können Sentinel1- und Sentinel2-Daten verschiedener Level direkt in den Programmcode eingebunden werden. Für diese Arbeit wurden die Level-2A-Daten verwendet (`ee.ImageCollection("COPERNICUS/S2_SR")`). Diese liefern „Bottom of Atmosphere“-Bilder, die seit März 2018 systematisch (zunächst nur für Europa) vorprozessiert wurden. Die Produktion wurde dann im Dezember 2018 auf die globale Ebene

ausgeweitet (ESA o. J.). Deswegen wurden für die Zeitreihe lediglich die Daten zwischen 2018 und 2022 erhoben, wodurch eine gewisse Datenkonsistenz gewährleistet wird. Dabei können u.a. Borkenkäferkalamitäten abgebildet werden, die v.a. 2018 und 2019 vermehrt in RLP auftraten.

Tabelle 1: Bänder der S2-Satelliten (ESA o. J.)

Band	mittlere Wellenlänge (nm)	Bandbreite (nm)	mittlere Wellenlänge (nm)	Bandbreite (nm)	räumliche Auflösung (m)	Beschreibung
	S2A		S2B			
1	442,7	21	442,2	21	60	ultra blue
2	492,4	66	492,1	66	10	blue
3	559,8	36	559,0	36	10	green
4	664,6	31	664,9	31	10	red
5	704,1	15	703,8	16	20	VNIR
6	740,5	15	739,1	15	20	VNIR
7	782,8	20	779,7	20	20	VNIR
8	832,8	106	832,9	106	10	VNIR
8a	864,7	21	864,0	22	20	SWIR
9	945,1	20	943,2	21	60	SWIR
10	1373,5	31	1376,9	30	60	SWIR
11	1613,7	91	1610,4	94	20	SWIR
12	2202,4	175	2185,7	185	20	SWIR

Als Trainingsdatensatz für die Klassifizierung wurden Daten der Forsteinrichtung in RLP (WöFIS-Daten) zur Verfügung gestellt. Dabei handelt es sich um Polygon-Datensätze aus dem Jahr 2016, die alle forstlich genutzten Flächen in Rheinland-Pfalz abdecken, wobei Privatwälder nicht abgebildet werden, sondern nur Staats- und Körperschaftswälder. Bei einem Großteil der Polygone werden über die Attribute Angaben zum Alter des Waldes, zu den vorhandenen Baumarten, deren Zusammensetzung usw. gemacht. Zu den forstlich genutzten Flächen gehören allerdings nicht nur Laub-, Nadel- und Mischwaldflächen, sondern auch unbewaldete oder nur teilweise bewaldete Freiflächen, z.B. Rückegassen, Flächen, die zur Jagdausübung als Wildwiesen, Kirrstellen zur Lockfütterung, Schussschneisen usw. genutzt werden, Kahlschlagflächen und Neupflanzungen, Wälder mit größeren Lichtungen oder Flächen unter Hochspannungsleitungen (s. Abbildung 2). Diese Flächen sind im WöFIS-Datensatz nicht attribuiert, d.h. zu den Flächen finden sich keine Angaben zu Baumartenzusammensetzung,

Alter etc. Bei der Auswahl der Trainingsdatensätze sind mehrere Aspekte zu beachten. Zum einen müssen die Trainings- und Testdaten unabhängig voneinander sein (z.B. aus verschiedenen Quellen oder verschiedene Stichproben aus einer Quelle). Zum anderen muss aufgrund der Sensibilität der Algorithmen hinsichtlich der Trainingsdaten darauf geachtet werden, dass der Trainingsdatensatz einerseits groß genug ist, um die Realität abzubilden, andererseits müssen die Mengenverhältnisse der Flächen verschiedener Klassen realistisch sein. Dies wird über die zufällige Selektion aus dem Gesamtdatensatz gewährleistet.

Abbildung 2: Als Nicht-Wald gelabelte Flächen aus Forsteinrichtungsdaten (WöFIS)

Für die Validierung der Ergebnisse standen punktuelle Daten zu den Waldflächen von Landesforsten RLP zur Verfügung. Diese wurden im Rahmen der LWI in einem regelmäßigen Punktgrid erhoben und flossen in die Daten der Bundeswaldinventur (BWI) ein. Die Erhebung der Daten für die BWI ist alle 10 Jahre gesetzlich vorgeschrieben und ist unabhängig von der Waldzustandserhebung, die seit 1984 bundesweit jedes Jahr erfolgt, und auch unabhängig von den Forsteinrichtungsdaten, die als Trainingsdaten für die Klassifikation genutzt werden.

Für die dritte Waldinventur haben 14 Inventurteams von Landesforsten in ganz RLP an über 8.000 dauerhaften Stichprobenpunkten mehr als 80.000 Bäume intensiv vermessen und dokumentiert. Die Erhebungen starteten mit einer Schulung der Aufnahmeteams im Frühjahr 2011 und wurden im Dezember 2012 abgeschlossen (LANDESFORSTEN RLP 2012). Zum Zeitpunkt der Bearbeitung der vorliegenden Arbeit standen die Ergebnisse der Landes- bzw. Bundeswaldinventur 2021/22 noch nicht zur Verfügung. Die Erhebungen der BWI erfolgen in RLP in einem Stichprobennetz von 2 x 2 km. Jede Stichprobe (Trakt) umfasst vier Stichprobenpunkte (Traktecken) mit einem Abstand von jeweils 150 m, an denen die Kartierung erfolgt (BMEL 2022, s. Abbildung 3).

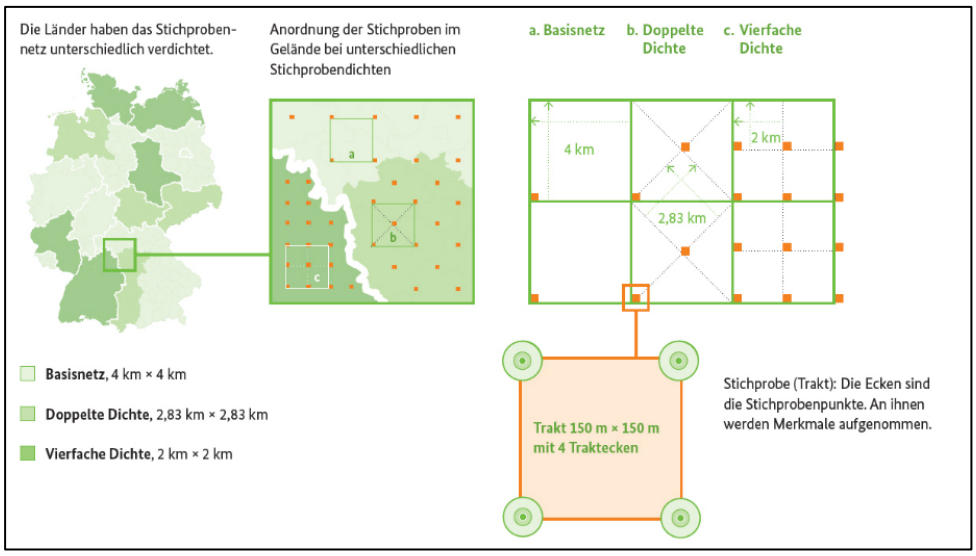


Abbildung 3: Stichprobennetz der BWI (BMEL 2022)

Um den Aufwand des Prozesses hinsichtlich der Rechenleistung möglichst gering zu halten, mussten die rasterförmigen Ergebnisbilder der Klassifikation innerhalb GEE auf eine Waldmaske zugeschnitten werden. Dafür wurden die Daten des Amtlichen Topografisch-Kartografischen Informationssystems (ATKIS) des Landesamts für Vermessung und Geobasisinformation Rheinland-Pfalz (LVermGeo) herangezogen. Diese standen der Universität Trier für Berechnungen zur Verfügung und konnten für diese Arbeit ebenfalls genutzt werden. In den ATKIS-Daten „werden topografische Informationen, wie Verkehrswege, Gewässer, bebaute Flächen, Vegetation, das Relief und vieles mehr digital geführt. Jedes Landschaftsobjekt wird nach Lage und Form durch Koordinaten und in seinen wesentlichen Eigenschaften durch Attribute und Namen beschrieben“ (LVERMGEO 2017). Von den ATKIS-Daten wurden nur die mit

„Wald“ gelabelten Flächen verwendet. Die Vorverarbeitung der Vektordaten wird in Kapitel 2.5 eingehend beschrieben.

2.2 Verwendete Software

Vorverarbeitet wurden die von Landesforsten Rheinland-Pfalz zur Verfügung gestellten Daten innerhalb der QGIS-Software. Es wurde die QGIS-Version 3.16 verwendet. QGIS (ehemals Quantum-GIS) ist ein Open-Source-Projekt, das von einem Team aus engagierten Freiwilligen und Organisationen entwickelt wird (QGIS 2022). Es bildet eine kostenlose Alternative zum Marktführer ArcGIS der Firma ESRI und enthält alle Werkzeuge, die zur Bearbeitung der Grundlagendaten für die vorliegende Arbeit benötigt wurden. Zu den verwendeten Werkzeugen gehören ausschließlich die Vektordatenbearbeitungstools, da die gesamte Rasterdatenbearbeitung innerhalb GEE stattfand.

GEE ist ein Dienst zur Prozessierung von Geodaten des Google-Unternehmens. Dessen Nutzung hat den Vorteil, dass die Geodatenverarbeitung in großem Maßstab durchgeführt werden kann, ohne dafür lokale Rechenleistung in Anspruch zu nehmen, da sie durch die Google Cloud-Plattform unterstützt wird. Laut GOOGLE DEVELOPERS (2022) ist der Zweck der Earth Engine die Bereitstellung einer interaktiven Plattform für die Entwicklung von Algorithmen für die Geodatenverarbeitung und damit die Ermöglichung hochwirksamer, datengesteuerter Wissenschaft, um substanzielle Fortschritte bei globalen Herausforderungen zu erzielen, die große Geodatensätze erfordern. Die Erstellung des Skripts kann in den Programmiersprachen JavaScript oder Python erfolgen. Die Programmierung in JavaScript erfolgt direkt in der browserbasierten GEE-API. Die Programmierung in Python kann nicht in der API vorgenommen werden, sondern erfolgt über andere Entwicklungsumgebungen (Integrated Development Environment (IDE)) wie Jupyter Notebook, RStudio oder auch über Plugins für z.B. QGIS. Für diese Arbeit wurde die Programmierung in der API in JavaScript durchgeführt. Das GEE-Interface besteht aus dem Codeblock mit der Konsole und der Kartenansicht (s. Abbildung 4). GEE ist auch für Programmier-Einsteiger geeignet, da für die JavaScript-Variante keinerlei Software installiert werden muss und der geschriebene Code direkt ausgeführt und über die Ausgabe in der Kartenansicht nachvollzogen werden kann. Sowohl die Skripte als auch die Datensätze werden über den Google-Account direkt in der Cloud gespeichert, sodass alle Schritte im Prozess innerhalb GEE stattfinden und nicht auf lokale Ressourcen zurückgegriffen werden kann und muss.

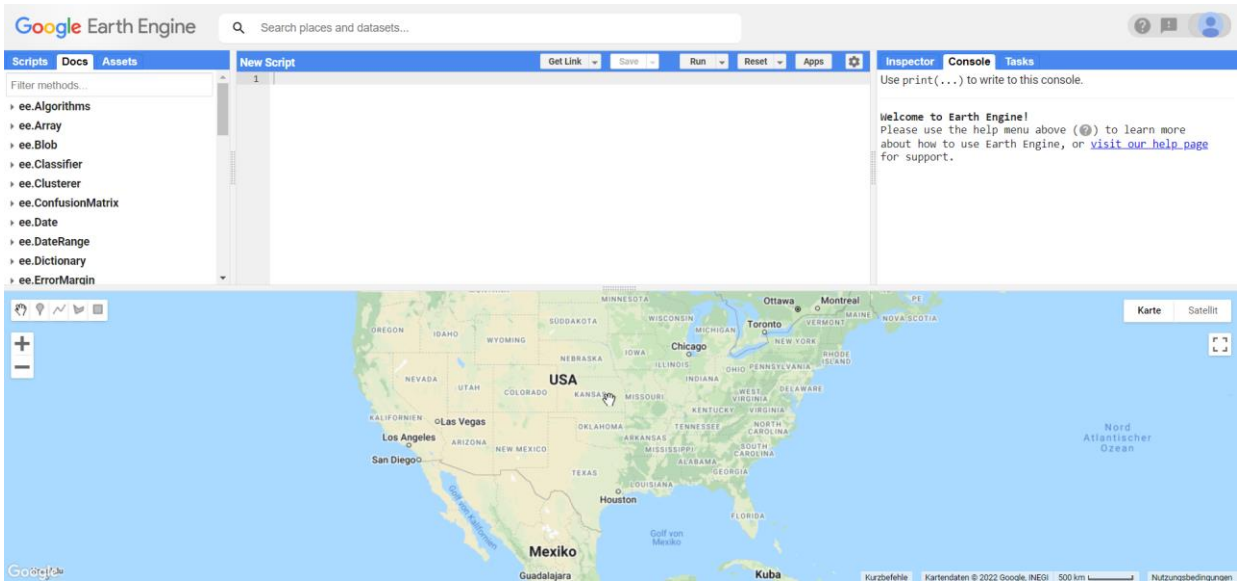


Abbildung 4: GEE-API (code.earthengine.google.com)

Da sich bei GEE die Nutzung der Rechenleistung von allen Usern geteilt wird, sind Einschränkungen und andere Schutzmaßnahmen erforderlich, um sicherzustellen, dass die User das System nicht monopolisieren. Für interaktive Anwendungen setzt GEE Grenzen für die maximale Dauer von Anfragen (derzeit 270 Sekunden), die Gesamtzahl der gleichzeitigen Anfragen pro Nutzer (40) und die Anzahl der gleichzeitigen Ausführung bestimmter komplexer Operationen wie z. B. räumlicher Aggregationen (25). Batch-Prozesse (Prozesse, die „gestapelt“, also nacheinander abgearbeitet werden) können ohne diese Einschränkungen durchgeführt werden, sodass man bei größeren Berechnungen darauf zurückgreifen kann. Dies funktioniert beispielsweise durch die Verlagerung der Datenbearbeitung aus dem Arbeitsspeicher in die Cloud. Sowohl für interaktive Berechnungen als auch für Batch-Prozesse gibt es allerdings noch eine zusätzliche Begrenzung: Objekte, die zwischengespeichert werden sollen, dürfen nicht größer als 100 MB sein. Diese Grenze kann schnell durch große Datensätze und/oder hohe Auflösungen erreicht werden. Diese Begrenzung kann z.B. bei Klassifikationsprozessen mit MLA dazu führen, dass der Trainingsdatensatz verringern werden muss (GORELICK et al. 2017: 25).

Die Validierung der Klassifikationsergebnisse erfolgt teils in GEE selbst (*accuracy assessment*) und teils in ArcGIS Pro (Jaccard-Index). ArcGIS Pro ist ein Programm der Firma ESRI, welches im Gegensatz zu QGIS nicht kostenlos verfügbar ist. Es bietet eine Vielzahl von Geodatenverarbeitungstools, wozu auch die Validierung über den Jaccard-Index gehört. Die

Ergebnisse der vorliegenden Arbeit werden unter Zuhilfenahme des *Intersect*-Tools in ArcGIS Pro mit den Ergebnissen von NINK et al. (2019) verglichen. Der Jaccard-Index wird zusätzlich zur Validierung in GEE berechnet, um die Genauigkeit der Ergebnisse umfassender zu beschreiben.

2.3 Vegetationsindizes

Um die Unterschiede der Reflexionsgrade von Laub- und Nadelwald hervorzuheben, genügt es nicht, die Ausgangsbilder in Echtfarbe (red, green, blue) zu klassifizieren. Dafür eignen sich Vegetationsindizes, die in einer Vielzahl für unterschiedliche Anwendungsfälle existieren. Bei S2-Daten können in der Regel die gleichen Vegetationsindizes und Koeffizienten wie für Landsat-Daten verwendet werden, da sich die Wellenlängenbereiche, die von den Sensoren aufgezeichnet werden, stark ähneln. In dieser Arbeit wurde entschieden, den ‚klassischen' NDVI mit der TC-Transformation zu vergleichen, auf die im Folgenden näher eingegangen wird.

2.3.1 NDVI

Der NDVI ist einer der am häufigsten verwendeten Vegetationsindizes in der passiv-optischen Fernerkundung. Er basiert auf dem Unterschied der Reflexionsgrade des VNIR- und des roten Bands:

$$\text{NDVI} = \frac{\text{rNIR} - \text{rRED}}{\text{rNIR} + \text{rRED}}$$

Dabei stellen ρNIR und ρRED jeweils die spektrale Reflexion des NIR- und roten Bands des Sensors dar. Hohe NDVI-Werte resultieren also aus einer Kombination aus einer hohen Reflexion im NIR-Bereich und einer geringen Reflexion im roten Bereich. Diese Kombination ist typisch für photosynthetisch aktive Vegetationsformen. Nicht bewachsene Flächen (Boden, Wasser, Schnee/Eis) hingegen bilden viel geringere NDVI-Werte (LILLESAND, KIEFER und CHIPMAN 2015: 520f.)

2.3.2 Tasseled Cap-Transformation

Neben dem NDVI wurde die TC-Methode verwendet, um Laub- und Nadelwald zu klassifizieren. Diese Methode wurde ursprünglich von KAUTH und THOMAS (1976) entwickelt, um Landsat-MSS-Daten zu klassifizieren. TC ist eine lineare Transformation von Landsat MSS-Daten, die Boden- und Vegetationsinformationen auf eine einzelne Ebene im multispektralen

Datenraum projiziert, wobei die wichtigsten Spektralkomponenten einer landwirtschaftlichen Szene in zwei Dimensionen dargestellt werden. Die Transformation nach KAUTH und THOMAS (1976) beruht auf gewichteten Kombinationen der ursprünglichen Spektralkanäle, um mithilfe von Koeffizienten für jeden Spektralkanal einen Satz von vier neuen Variablen zu erzeugen, die jeweils eine bestimmte Dimension der landwirtschaftlichen Szene beschreiben. Zweck der Gewichtung der Spektralkanäle ist die Kalibrierung der spezifischen Sensoren, um Rauschen zu eliminieren (CAMPBELL und WYNNE 2011: 495ff.). Das bedeutet, dass sich die Koeffizienten nach KAUTH und THOMAS (1976) nicht auf andere Sensoren übertragen lassen. KAUTH und THOMAS (1976: 42ff.) beschrieben die vier neu erzeugten Variablen als *wetness, greenness, yellowness* und *nonesuch.* Die TC-Transformation nach CRIST und CICONE (1984) erzeugt drei verschiedene Komponenten, die die Summe aller Landsat-TM-Bänder (gewichtet) darstellen und die spezifische Reflexion des Bodens abbilden. Die erste Komponente ist die Helligkeit (*brightness*). Die zweite und für diese Arbeit wichtigste Komponente ist der *greenness*-Faktor, der den Kontrast zwischen dem nahen Infrarot und den sichtbaren Bändern bildet. Die dritte Komponente (*wetness*) beschreibt die Feuchtigkeit der Baumkronen und des Bodens (LILLESAND, KIEFER und CHIPMAN 2015: 529f., CRIST und CICONE 1984: 256). In der grafischen Darstellung der Ebenen (z.B. *greenness* und *yellowness*, vgl. Abbildung 5) ähnelt die Kurve, die die gewichtete Reflexion einer landwirtschaftlichen Szene im zeitlichen Verlauf darstellt, der namensgebenden Bommelmütze („tasseled cap“). Da sich die Sensoren von Landsat-TM und S2 stark ähneln, können die Koeffizienten nach CRIST und CICONE (1984, s. Tabelle 2) auch auf S2-Bilder angewendet werden.

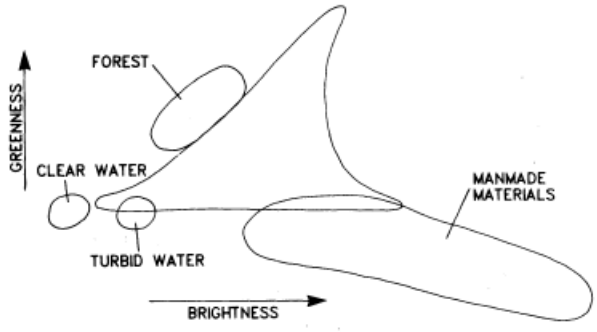


Abbildung 5: Dimensionen TC-*greenness* und -*brightness* (CRIST und CICONE 1984: 259)

Tabelle 2: Koeffizienten nach CRIST und CICONE (1984)

	Band 2	Band 3	Band 4	Band 8A	Band 11	Band 12
brightness	0,3037	0,2793	0,4743	0,5585	0,5082	0,1863
greenness	-0,2848	-0,2435	-0,5436	0,7243	0,0840	-0,1800
wetness	0,1509	0,1973	0,3279	0,3406	-0,7112	-0,4572

Bei der Unterscheidung von Laub- und Nadelwald werden üblicherweise vor allem der *greenness*- und *wetness*-Faktor betrachtet. Dass auch der *brightness*-Faktor eine Rolle spielt, zeigen die folgenden Abbildungen. Es wurden die Ergebnisse des TC-Layer aus dem Frühjahr 2019 (24.02.19 – 20.04.19) für jeweils einen zufälligen Punkt im Nadelwald, im Laubwald und für den Nicht-Wald beispielhaft für eine Wiese abgefragt. Die X-Achse zeigt dabei den Zeitverlauf und die Y-Achse den berechneten Faktor.

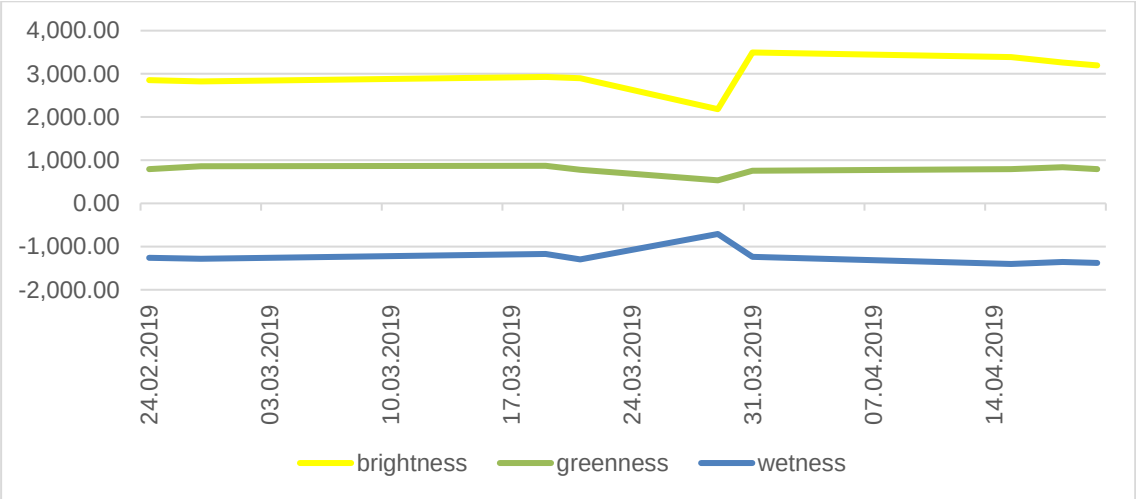


Abbildung 6: Ergebnisse TC (Laubwald)

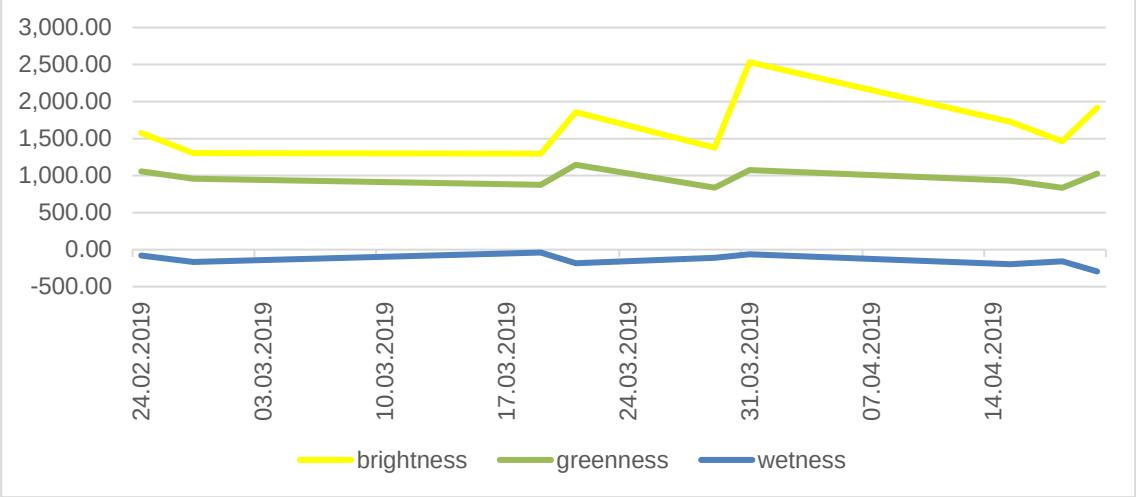


Abbildung 7: Ergebnisse TC (Nadelwald)

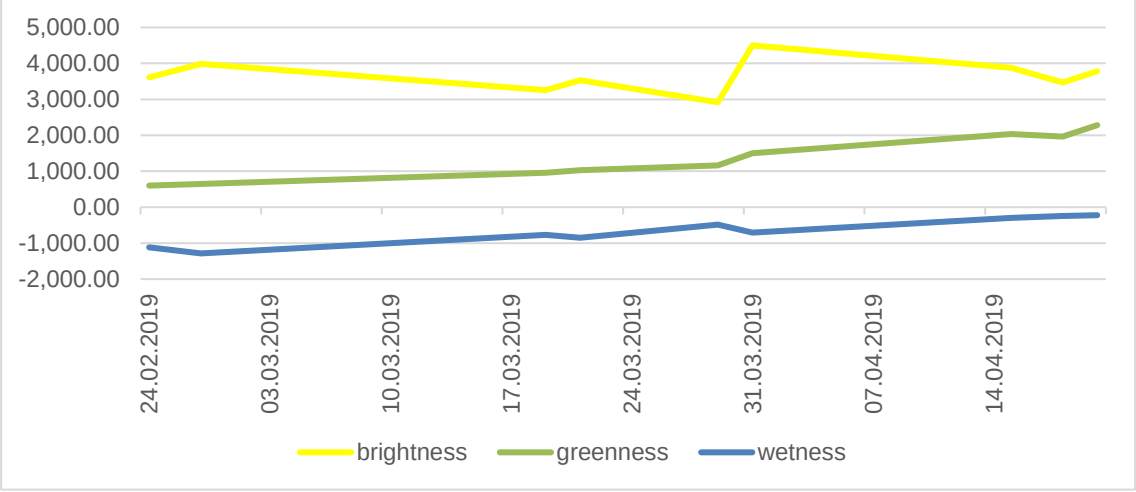


Abbildung 8: Ergebnisse TC (Wiese)

Wie in den vorhergehenden Abbildungen erkennbar wird, unterscheiden sich die Abfragewerte der Bänder *brightness*, *greenness* und *wetness* deutlich bei Laub-, Nadel- und Nicht-Wäldern. Der berechnete *brightness*-Faktor ist bei Laubwäldern höher als bei Nadelwäldern, da Laubwälder vor der Belaubung eine höhere Helligkeit als Nadelwälder aufweisen. Die Wiese lieferte in dem Fall die höchsten *brightness*-Werte. Die *greenness* liegt beim Nadelwald ungefähr beim Faktor 1000, während sie sich beim Laubwald leicht darunter um den Faktor 750 bewegt. Bei der Wiese hingegen ist ein leichter Aufwärtstrend zu beobachten, da die Wiesen und Weiden im Frühjahr allmählich grüner werden. Dieser Aufwärtstrend ist beim Laubwald in diesem Zeitraum (Frühjahr 2019) noch nicht zu beobachten, was bedeutet, dass die Belaubung noch nicht eingesetzt hat. Beim *wetness*-Faktor ist der Unterschied zwischen Laub- und Nadelwald deutlich zu beobachten. Er liegt zwar bei allen drei Klassen unter 0, aber beim Nadelwald zwischen 0 und -250 und beim Laubwald deutlich niedriger bei bis zu ca. -1300. Bei der Wiese ist auch hier wieder ein leichter Aufwärtstrend zu beobachten, da die Wiese über den o.g. Zeitraum hinweg einen immer höheren Wassergehalt aufweist. Diese Unterschiede hinsichtlich der berechneten Faktoren können genutzt werden, um die drei Klassen Laub-, Nadel- und Nicht-Wald zu klassifizieren. Dafür werden MLA verwendet, auf die im folgenden Kapitel eingegangen wird.

2.4 Klassifikationsalgorithmen

Die Bildklassifizierung ist ein Verfahren, das spektrale Muster verwendet, um jedes Pixel einzeln zu klassifizieren, und zwar auf der Grundlage eines statistischen oder deterministischen Modells, ohne die Nachbarn oder die Umgebung des zu klassifizierenden Pixels zu berücksichtigen. Diese Methode wird auch als *pixel-based image analysis* (PBIA) bezeichnet. Bei der *spatial pattern recognition* hingegen werden die Pixel auf Grundlage ihrer räumlichen Beziehung zu den umgebenden Pixeln klassifiziert. Räumliche Klassifikatoren verwenden Aspekte wie Textur, Flächengröße, Form, Kontext usw. Dieser Prozess erfordert Eingaben von einem menschlichen Analytiker und ist in der Regel komplexer als Verfahren zur spektralen Mustererkennung. Die *object-based image analysis* (OBIA) kann als eine Kombination dieser beiden Arten von Bildklassifikatoren verwendet werden (LILLESAND, KIEFER und CHIPMAN 2015: 537). Es gibt zwei Arten von Klassifizierungsverfahren: *supervised* und *unsupervised*. Bei der *supervised classification* werden die Eingaben des Image Analyst verwendet, der die Pixel-Klassifizierung "überwacht", indem er dem Computeralgorithmus numerische Beschreibungen (Labels) der Klassen vorgibt. Dabei werden repräsentative Stichproben von Bereichen, in denen die Klasse bekannt ist (Trainingsdatensätze), verwendet, um einen Interpretationsschlüssel zu

erstellen, der die spektralen Attribute für jeden Merkmalstyp von Interesse beschreibt. Jedes Pixel im Datensatz wird dann numerisch mit jeder Klasse im Interpretationsschlüssel verglichen und mit dem Klassenschlüssel gekennzeichnet, dem er „am ähnlichsten sieht“ (LILLESAND, KIEFER und CHIPMAN 2015: 538ff.). In der vorliegenden Arbeit erfolgt die Klassifizierung von Waldtypen anhand von Trainingsdatensätzen von Gebieten, in denen der Waldtyp bekannt ist (zum Beispiel Laubwald). Anhand dieser Trainingsdatensätze wird der Computeralgorithmus so trainiert, dass er jeden Laubwald in einem bestimmten Gebiet von Interesse findet, indem er die Merkmale der Gebiete des Trainingsdatensatzes (spektrales Muster) mit jedem Pixel in anderen Gebieten vergleicht. Bei der OBIA spielen außerdem, wie bereits erwähnt, neben dem spektralen Muster Form und Größe der Flächen eine Rolle. Zusammenfassend lässt sich sagen, dass die überwachte Klassifizierungsmethode aus einem Trainingsschritt gefolgt von einem Klassifizierungsschritt besteht. Die zweite Klassifizierungsmethode, die unüberwachte Klassifizierung, funktioniert anders. Die Bilddaten werden zunächst durch Clustering in natürliche Spektralgruppen klassifiziert, dann bestimmt der Image Analyst die Klassen dieser Spektralgruppen, indem er sie mit Bodenreferenzdaten vergleicht und ggf. zusammenfasst. Die Anzahl der Klassen wird vorher festgelegt, sodass das Ergebnis beeinflusst werden kann (LILLESAND, KIEFER und CHIPMAN 2015: 538, 556ff.).

Sowohl die PBIA-, als auch die OBIA-Methode bieten Vor- und Nachteile. Ein wichtiger Aspekt ist dabei die Klassifizierung von Flächen, die nicht nur einer Klasse zuzuordnen sind, z.B. nicht vollständig bewaldete Flächen oder Mischwälder (vgl. BRAUCHLER und STOFFELS 2020: 3ff.). Da bei der PBIA jedes Pixel einzeln betrachtet wird, ist die Zuordnung zu einer Klasse unabhängig von der Umgebung. Dadurch können z.B. Lichtungen im Wald im Optimalfall auch als solche erkannt werden. Bei der OBIA ist die Wahrscheinlichkeit hoch, dass diese Lichtungen nicht herausgefiltert werden, sondern zusammen mit den umgebenden Pixeln als eine zusammenhängende Waldfläche erkannt werden. Umgekehrt kann die PBIA-Methode zu einer unnötigen Komplexität des Ergebnisses führen, beispielsweise werden kleine unbewaldete Flächen (z.B. Lichtungen) als Nicht-Wald klassifiziert, die bei der OBIA-Methode ignoriert werden würden. Zum Zwecke der Unterscheidung zwischen Laub- und Nadelwald kann auch die OBIA-Methode gewählt werden. BRAUCHLER und STOFFELS (2020) konnten beispielsweise bei der Klassifikation von Waldtypen in Luxemburg über hochauflösende Luftbilder eine overall accuracy (OA) von 85% erreichen, wobei falsch klassifizierte Flächen vor allem durch over-/undersegmentation (zu geringe/zu hohe Komplexität der Flächen), Nadelwald-Neupflanzungen, Grenzbereiche zu Mischwäldern und Standorte am Waldrand begründet werden konnten (BRAUCHLER und STOFFELS 2020: 13, 16). Auch BRAUCHLER, STOFFELS und NINK (2022)

wandten die OBIA-Methode auf S2-Daten an und erreichten so eine OA von max. 89,3% (BRAUCHLER, STOFFELS und NINK 2022: 14).

Um das bestmögliche Klassifikationsergebnis zu erhalten, wurden in dieser Arbeit drei verschiedene PBIA-Klassifikationsalgorithmen verglichen, die alle den MLA zuzuordnen sind. Algorithmen, die maschinelle Lernmethoden verwenden, erfreuen sich in der passiv-optischen Fernerkundung einer immer größer werdenden Beliebtheit, da sich klassische Methoden aufgrund der hohen räumlichen und spektralen Vielfalt als schwierig erweisen. Das maschinelle Lernen wurde in der Fernerkundung bereits häufig erfolgreich für Klassifizierungen, Regressionsanalysen, Clustering etc. genutzt (CAMPS-VALLS 2009: 1).

Künstliche Intelligenz bezeichnet jede Technik, die es Computern ermöglicht, menschliche Intelligenz zu imitieren. MLA sind ein Teilbereich der künstlichen Intelligenz, die sich mit dem Prozess befassen, bei dem der Algorithmus eine Reihe von Trainingsdaten vorgelegt bekommt und dann herausfindet, wie sie unterschieden werden können. Alle überwachten Klassifizierungsmethoden sind Teil des maschinellen Lernens. Die MLA, die bei dieser Arbeit im Fokus stehen, stellen jeweils eine Form der *supervised pixel-based classification* dar, bei der die Algorithmen anhand von klassifizierten Polygonen (WöFIS-Daten) trainiert werden und die Klassifikation schließlich auf ein großflächigeres Gebiet übertragen wird. Der Klassifizierungsschritt wird in den nachfolgenden Kapiteln ausführlich beschrieben.

Für den Einsatz von maschinellen Lernmethoden werden neben den Trainings- auch Testdaten benötigt; die Trainingsdaten werden verwendet, um den Algorithmus für die Objekterkennung zu trainieren, während die Testdaten dazu dienen, die Ergebnisse der Algorithmen zu validieren. Als MLA wurden ausschließlich Methoden ausgewählt, die bereits in GEE implementiert sind und deshalb nicht erst manuell programmiert werden müssen. Dabei handelt es sich um die Algorithmen RF, CART und GTB, die in der Programmierumgebung direkt über den Aufruf einer Funktion mit bestimmten Übergabeparametern verwendet werden können (GOOGLE DEVELOPERS 2021).

Im Gegensatz zur Methodik in dieser Arbeit wurden die Klassen Laub- und Nadelwald in NINK et al. (2019) über einen Threshold-Ansatz getrennt. Dabei wird die Anzahl der Vorkommen jeder Klasse in den Referenzdaten berechnet, mit der dann der Minimum Turning Point (MTP) zwischen den beiden Klassen Laub- und Nadelwald ermittelt wird. Anhand dessen wird der Suchbereich für den optimalen Schwellenwert berechnet. In der folgenden Abbildung ist der MTP als vertikale schwarze Linie abgebildet.

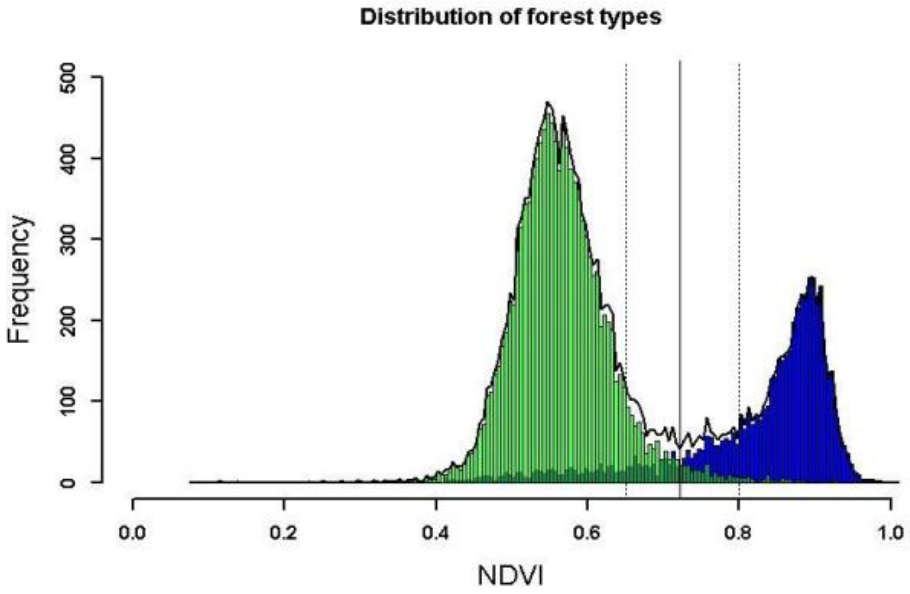


Abbildung 9: Verteilung von Laub- (grün) und Nadelwald (blau, NINK et al. 2019: 8)

2.4.1 Classification and Regression Trees

CART ist ein Klassifizierungsalgorithmus auf Grundlage des Konzepts von Entscheidungsbäumen, bei dem die Klassifizierung von Objekten auf den Merkmalen basiert, die in einer Klasse (Trainingsdaten) vorhanden sind. Er kann auch als eine „wenn-dann“-Methode bei der Objektklassifizierung bezeichnet werden. Er wurde von BREIMAN et al. (1984) entwickelt und gehört zu den überwachten Klassifizierungsmethoden. Es werden Stichproben trainiert, um zum Zwecke der Klassifikation oder Regression einen binären Entscheidungsbaum zu konstruieren. Binäre Entscheidungsbäume können für vielseitige Zwecke verwendet werden. In der folgenden Abbildung nach BREIMAN et al. (1984: 2) ist beispielhaft ein Entscheidungsbaum dargestellt, der herausfinden soll, ob ein Patient ein hohes (G) oder niedriges (F) Risiko besitzt, einen Herzinfarkt zu bekommen.

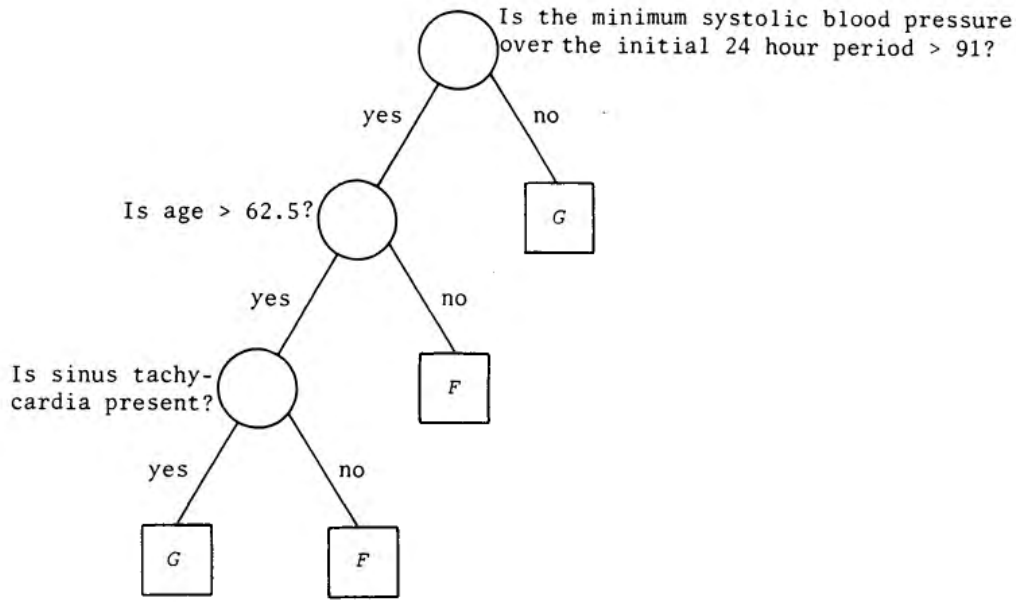


Abbildung 10: Classification Tree nach BREIMAN et al. 1984: 2

Die Besonderheit dieser Methode besteht darin, dass die binäre Baumstruktur vollständig ausgenutzt wird, d.h. der Wurzelknoten umfasst alle Stichproben. Der Wurzelknoten wird nach bestimmten Teilungsregeln in zwei Unterknoten aufgeteilt. Dieser Prozess wiederholt sich regressiv in den Unterknoten, bis der letzte Unterknoten nicht mehr in zwei Unterknoten aufgeteilt werden kann. Der Grundgedanke bei der Konstruktion eines CART besteht darin, auf der Grundlage der gesamten Stichprobendaten einen mehrstufigen und mehrblättrigen Knotenbaum zu erstellen, um die Beziehungen zwischen den Knoten abzubilden und dann den Baum zu zerteilen. Dadurch wird eine Reihe von Unterbäumen erstellt, woraus der geeignetste Baum für die Klassifikation ausgewählt wird. Im Einzelnen umfasst der Prozess die Erstellung eines Baums (*building*) und das Teilen (*pruning*) eines Baums (JIANG et al. 2010: 354f.). Im Gegensatz zu den anderen beiden Algorithmen können die Klassifikationsergebnisse nicht durch eine Erhöhung der Entscheidungsbäume verbessert werden, da lediglich ein Entscheidungsbaum gebildet wird. Dies führt zum Schwachpunkt der Methode: Sie ist sehr empfindlich gegenüber Trainingsdaten, d.h. die Ergebnisse sind abhängig von der randomisiert gewählten Stichprobe und Änderungen im Trainingsdatensatz führen zu unterschiedlichen Klassifizierungsergebnissen (KAMAL et al. 2019: 3).

2.4.2 Random Forest

Die Random Forest-Methode geht auf HO (1995), bzw. in der Weiterentwicklung auf BREIMAN (2001) zurück und basiert auf dem Prinzip der binären Entscheidungsbäume bzw. CART (s. Kapitel 2.4.1). Dabei handelt es sich um eine Methode, auf die bei der Beantwortung vieler statistischer Fragestellungen zurückgegriffen wird und die häufig auch in der Fernerkundung zur Klassifizierung von z.B. Landnutzung angewendet wird. Als Beispiele seien hierfür die Arbeiten von BRAUCHLER und STOFFELS (2020), KAMAL et al. (2019) und XULU et al. (2020) genannt, die u.a. diese Methode zur Klassifizierung von Waldtypen, Abgrenzung von Mangrovenwäldern, bzw. Erkennung von Erntezeitpunkten in Plantagen benutzten. Innerhalb eines Entscheidungsbaums wird das *training set* auf der Grundlage seiner Beziehung zur Vorhersagevariablen in immer homogenere Gruppen geteilt (s. Abbildung 11). Da einzelne Entscheidungsbäume (CART) sehr sensibel auf die Trainingsdaten reagieren und keine beliebige Komplexität abgebildet werden kann, entwickelte HO (1995) das Prinzip der Random Decision Forests. Dabei wird aus den Trainingsdaten eine beliebige Anzahl an zufällig ausgewählten *subsets* gewonnen, für die jeweils ein eigener, unabhängiger Entscheidungsbaum gebildet wird (HO 1995: 279). Diese Methode wird als *bootstrapping* bezeichnet und gehört zu den *resampling*-Methoden. Die Entscheidungen werden dabei nicht anhand aller Merkmale der Daten (*feature space*), sondern nur anhand ausgewählter *features* getroffen. Die konstruierten Entscheidungsbäume werden anschließend dafür genutzt, „neue“ Daten zu klassifizieren. Sie durchlaufen nacheinander alle Entscheidungsbäume und werden der Klasse zugwiesen, die in den Bäumen am häufigsten vorhergesagt wurde (*aggregation,* BREIMAN 2001: 5f.). Die Kombination der beiden randomisierten Prozesse *bootstrapping* und *aggregation* wird als *bagging* bezeichnet (BREIMAN 1996: 123).

Die RF-Methode hat den Vorteil, dass sie weniger sensibel gegenüber den Trainingsdaten ist, weil aufgrund der Erstellung von *subsets* unterschiedliche Entscheidungsbäume gebildet werden, die Ausreißer in den Daten ausgleichen können. Das bedeutet auch, dass sich die Genauigkeit der Ergebnisse erhöht, je mehr Entscheidungsbäume erstellt werden (HO 1995: 281f.). Gleichzeitig zieht eine Erhöhung der Entscheidungsbäume eine höhere Rechenleistung und damit auch eine Verschlechterung der Performanz des Programms mit sich. Deswegen wurde die Klassifikation durch die RF-Methode mehrfach mit wechselnden Anzahlen an Entscheidungsbäumen durchgeführt, um zu prüfen, ab welcher Anzahl keine signifikante Verbesserung der Klassifikation mehr stattfindet und es sich deswegen nicht mehr ‚lohnt‘, die Anzahl der Bäume zu erhöhen.

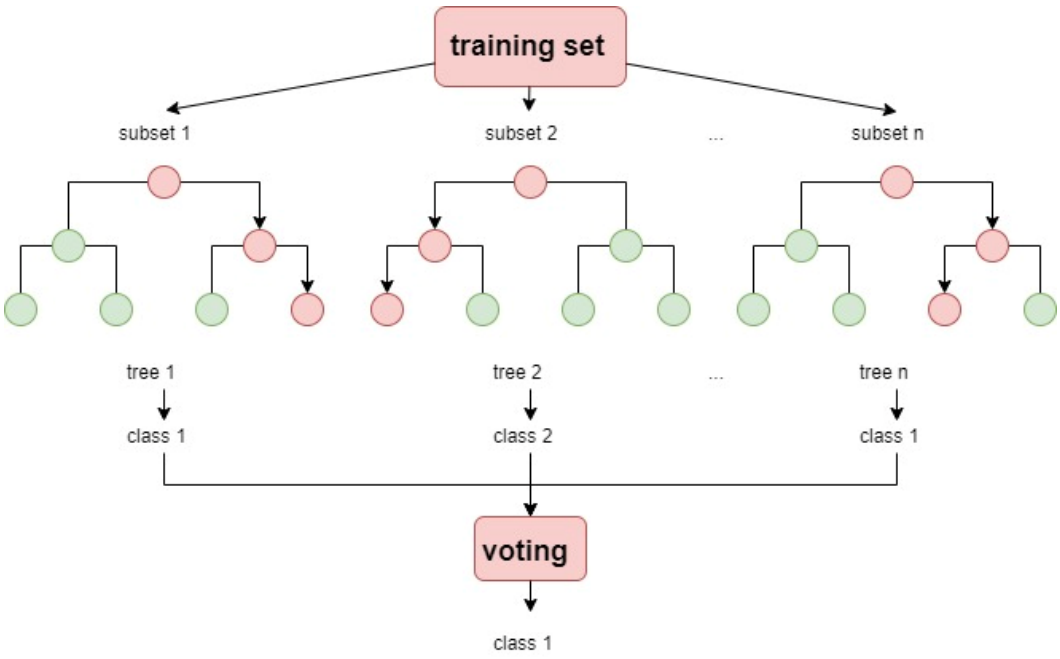


Abbildung 11: Modell des Random Forest-Algorithmus (eigene Darstellung nach HO 1995)

2.4.3 Gradient Tree Boost

Die *boosting*-Methode wurde von BREIMAN (1997) beschrieben und geht auf den „Adaboost"-Algorithmus von FREUND und SHAPIRE (1995) zurück (BREIMAN 1997: 1). *Boosting* bedeutet, dass ein Algorithmus in Serie kombiniert wird, um aus vielen aufeinander folgenden *weak learners* einen *strong learner* zu bilden. Im Falle des GTB-Algorithmus handelt es sich bei den *weak learners* um Entscheidungsbäume. Im Gegensatz zur RF-Methode, bei der mehrere Entscheidungsbäume parallel gebildet und anschließend durch Aggregation zusammengefasst werden (*bagging*), werden also beim GTB-Ansatz eine Serie von Entscheidungsbäumen gebildet, die nacheinander abgearbeitet werden, wobei jeder Baum die Fehler des vorherigen Baums zu korrigieren versucht. Das Hintereinanderschalten vieler Bäume und die Konzentration auf die Fehler des vorherigen Baums machen das *boosting* zu einem hocheffizienten und genauen Modell. Im Gegensatz zum *bagging* wird beim *boosting* kein Bootstrap-Sampling durchgeführt. Jedes Mal, wenn ein neuer Baum hinzugefügt wird, entspricht er einer modifizierten Version des Ausgangsdatensatzes. Das endgültige Modell fasst die Ergebnisse der einzelnen Schritte zusammen, wodurch ein *strong learner* gebildet wird (FREUND und SHAPIRE 1995: 120). Anhand der folgenden Abbildung wird deutlich, inwiefern sich die Methoden *bagging* (RF) und *boosting* (GTB) unterscheiden.

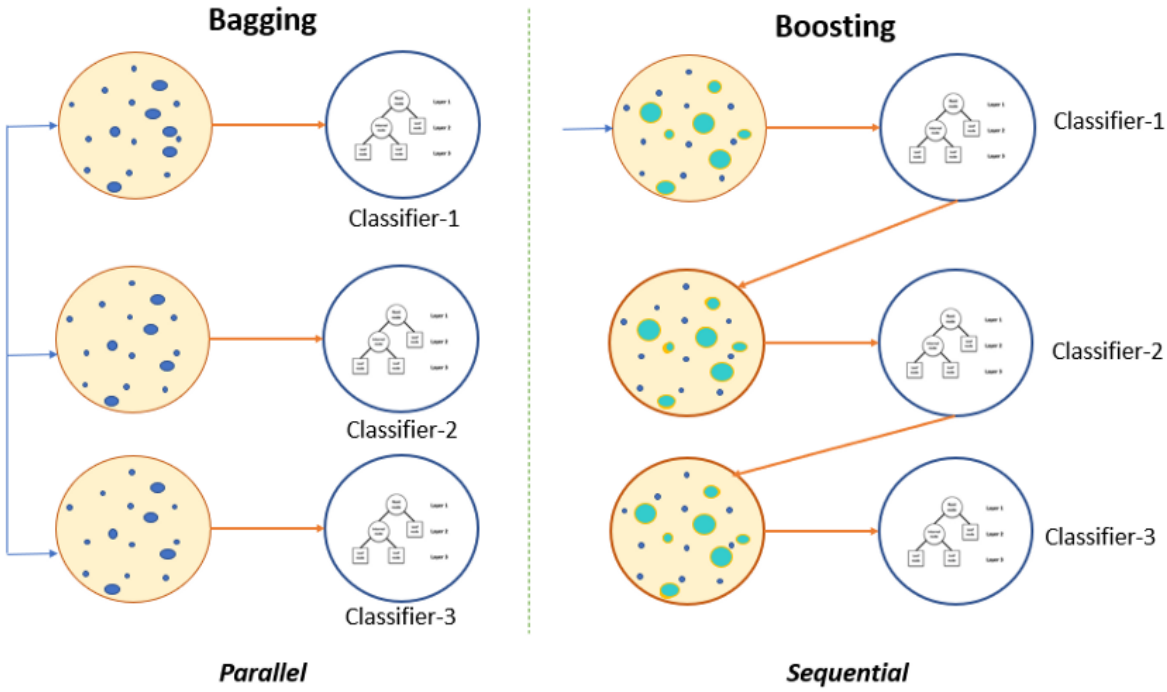


Abbildung 12: *Bagging* (unabhängige Vorhersagen, RF) vs. *boosting* (sequenzielle Vorhersagen, GTB) (PAL 2020)

2.5 Methode

Bei der Auswahl der Trainingsdaten für Laub- und Nadelwälder wurden ausschließlich Reinbestände gewählt, also nur Flächen mit 100% Laub- und 0% Nadelwald und umgekehrt. Daraus wurden über die *random selection*-Funktion in QGIS 1500 Flächen selektiert. Anschließend wurden diese mit dem jeweiligen Label versehen (0 = Laubwald, 1 = Nadelwald, 2 = Nicht-Wald). Leider befanden sich unter den nicht attribuierten Polygonen auch Laub- und Nadelwaldflächen sowie unbewaldete Flächen mit natürlicher Sukzession (vgl. Kapitel 2.1). Diese mussten in QGIS händisch über einen Luftbildabgleich gefiltert werden, damit bewaldete Flächen später nicht fälschlicherweise als Nicht-Wald klassifiziert werden. Nur die Polygone, mit denen ein Luftbildabgleich erfolgt ist und die als Nicht-Wald identifiziert werden konnten, wurden auch als Nicht-Wald gelabelt. Danach wurde die Zuordnung zu den Klassen Laub- und Nadelwald stichprobenartig anhand des Luftbilds überprüft, und ggf. Labels geändert. Insgesamt stand schließlich ein Trainingsdatensatz mit 1458 gelabelten Polygonen zur Verfügung (898 Laubwaldflächen, 523 Nadelwaldflächen und 37 Nicht-Waldflächen). NINK et al. (2019) verwendeten für die Klassifikation einen Trainingsdatensatz mit 17878 Flächen, wobei sich die

Trainingsflächen über RLP, Saarland und Luxemburg, also über ein deutlich größeres Untersuchungsgebiet erstreckten (NINK et al. 2019: 6). Die Klassifikation fand in dem Fall nicht über eine cloudbasierte Lösung statt, sondern es wurden lokale Rechenleistungen in Anspruch genommen. Durch die hohe Anzahl an gelabelten Trainingsdaten konnten die Berechnungen in GEE allerdings aufgrund der o.g. Beschränkungen gar nicht erst ausgeführt werden. Ausgehend von dieser Anzahl wurde der Trainingsdatensatz schrittweise verkleinert, bis die Klassifizierung erfolgreich ausgeführt wurde. Die Anzahl der Entscheidungsbäume wurde ebenfalls schrittweise herabgesetzt. Diese Tests ergaben, dass die Berechnung bei einer Anzahl von ca. 1500 Flächen im Trainingsdatensatz und einer Anzahl von 5 Entscheidungsbäumen bei der RF- und GTB-Methode noch funktioniert, die dem User zur Verfügung stehende Rechenleistung der GEE für eine erhöhte Komplexität allerdings nicht reichte.

Da nicht für jedes Jahr ein eigener Trainingsdatensatz erstellt werden kann, weil für die Jahre 2018-2022 keine Realdaten verfügbar sind, muss davon ausgegangen werden, dass sich die aus den WöFIS-Daten von 2016 gewonnenen Datensätze und deren Labels über den gesamten Zeitraum nicht verändert haben. Durch den stichprobenartigen Abgleich mit den rheinland-pfälzischen Luftbildern aus 2020 (LVERMGEO 2020) werden einzelne Flächen ausgesiebt, deren Labels nicht mehr mit den tatsächlichen Gegebenheiten übereinstimmen. Aus dem gebildeten Trainingsdatensatz wurde zusätzlich ein Layer erstellt, der mit einem Minuspuffer versehen wurde, um zu überprüfen, ob sich dieser durch eine Verbesserung in den Ergebnissen bemerkbar macht. NINK et al. (2019: 6) implementierten ebenfalls einen Minuspuffer, da sich geringfügig unterschiedliche Projektionen in den jeweiligen Layern auf die Klassifizierung der Grenzen der Flächen auswirken können. Um solche Grenzeffekte zu minimieren bzw. auszuschließen, wird bei den Flächen ein inverser Puffer von 10 m berechnet und die Ergebnisse mit denen des originalen Layers verglichen.

Bei der Vorverarbeitung der LWI-Daten erfolge ein ähnliches Vorgehen wie bei den Trainingsdaten. Zunächst wurden aus den Daten 3000 Punkte randomisiert selektiert und über die Attributtabelle mit Labels versehen. In den Originaldaten befinden sich zu den meisten Punkten Angaben, ob es sich um Laub- oder Nadelwald handelt. Wenn keine Angabe gemacht wurde, wurde zunächst davon ausgegangen, dass es sich bei diesen Punkten um Nicht-Wald-Flächen handelt und der Punkt wurde entsprechend gelabelt. Genau wie bei den Trainingsdaten wurden die Labels 0 für Laubwald, 1 für Nadelwald und 2 für Nicht-Wald vergeben. Bei den Laub- und Nadelwald-Punkten erfolgte wieder ein stichprobenartiger Abgleich mit dem Luftbild (LVERMGEO 2020), die Nicht-Wald-Punkte wurden individuell geprüft und ggf. entfernt, wenn das Label nicht übereinstimmte und an deren Stelle wurden neue, zufällige Punkte eingefügt.

Wie auch bei den Trainingsdaten wird davon ausgegangen, dass sich die Labels der nicht überprüften Punkte bis zum Jahr 2022 gegenüber den Realdaten nicht geändert haben. Die Wahrscheinlichkeit, dass sich die Klasse in dem Zeitraum geändert hat (z.B. Nadelwald wurde zur Kahlschlagfläche oder Freiflächen sind nun bewaldet) ist relativ gering, da Wälder (bis auf Naturkatastrophen oder Schädlingsbefall) in der Regel keinen großen Schwankungen unterliegen. Trotzdem kann nicht ausgeschlossen werden, dass sich unter den Punkten einzelne (jetzt) fehlerhafte Labels befinden. Abschließend standen genau 3000 gelabelte Validierungspunkte zur Verfügung, wobei 1752 Punkte zum Laubwald, 1214 Punkte zum Nadelwald und 34 Punkte zum Nicht-Wald zugeordnet werden konnten. Die Wahl der Anzahl von 3000 Validierungspunkten ergab sich im Prozess der Implementierung in GEE. Es stellte sich auch hier wieder heraus, dass die Auswahl eines größeren Punktdatensatzes zu einem Abbruch der Ausführung des Programms führte, da eine Erhöhung der Validierungsdaten zu einer erhöhten Rechenleistung führte. Auf dieses Problem wird in der Diskussion näher eingegangen.

Für die Verschneidung der Rasterdaten (klassifizierte S2-Daten) wurden die ATKIS-Daten herangezogen und vor dem Upload in GEE in QGIS bearbeitet. Alle Flächen, die mit „Wald“ in den ATKIS-Daten gelabelt sind, wurden in einen neuen Datensatz exportiert und die Grenzen aufgelöst (*dissolve*). Die Auflösung ist nötig, da in GEE bei Vektordaten für z.B. Verschneidungsprozesse lediglich eine Anzahl von 2 Mio. Knotenpunkten erlaubt ist, um die Ausführung des Prozesses zu gewährleisten.

Zu Darstellungs- und Abfragezwecken wurden außerdem die Forstamtsgrenzen von Landesforsten RLP und die Bundesland- und Gemeindegrenzen des Bundesamtes für Kartographie und Geodäsie (BKG) herangezogen (BKG 2022). In GEE gibt es die Möglichkeit, Raster- (z.B. GeoTIFF) und Vektordaten (z.B. Shapefile) als Asset hochzuladen. Diese Funktion wurde einschließlich der oben beschriebenen vorverarbeiteten Daten für folgende Vektordatensätze genutzt:

Tabelle 3: Asset-Datensätze

Datensatz	Quelle	Zweck
Grenze RLP	BKG	Darstellung in Karte, Auswahl des Landes für Auswertung
Gemeindegrenzen	BKG	Darstellung in Karte, Auswahl von Gemeinden für Auswertung
Forstamtsgrenzen	Landesforsten RLP	Darstellung in Karte, Auswahl von Forstämtern für Auswertung
Forsteinrichtungsdaten (flächig)	WöFIS	Trainingsdaten für Klassifizierungen
Waldflächen	ATKIS	Verschneidung mit Rasterdaten
Inventurdaten (punktuell)	LWI	Validierung der Klassifizierungen

Die Assets können nach dem Upload direkt im Skript im Format `FeatureCollection` eingebunden werden. Damit die Assets im Skript für die Benutzeroberfläche abgerufen und auch in der GUI dargestellt werden können, muss der Zugriff für jedes Asset geändert werden. In diesem Fall wurde die Möglichkeit genutzt, ausschließlich für die ‚App' des Hauptskripts zwar einen Lese-, aber keinen Schreibzugriff zu erlauben. Dies gewährleistet, dass außer über die App kein GEE-User die Möglichkeit hat, die Daten einzusehen, die nicht öffentlich zugänglich sind (LWI, ATKIS, WöFIS).

2.5.1 Klassifizierung

Die gesamte Klassifizierung und Vorbereitung der Layer wurde in ein zusätzliches Skript neben dem Hauptskript für die Implementierung der Benutzeroberfläche ausgelagert, damit nicht bei jedem Aufruf der Website die Klassifizierung für das ganze Bundesland ausgeführt werden muss. Nach der Klassifizierung wurden die Layer mit dem besten Ergebnis der jeweiligen Jahre ausgewählt, als Asset gespeichert und im Hauptskript direkt aufgerufen, sobald der User der GUI eine Berechnung durchführen möchte. Die nachfolgenden Ausführungen beziehen sich somit lediglich auf das Klassifizierungsskript.

Da für die Klassifizierung eine Vielzahl von Parameteränderungen getestet werden mussten, wurde für die Klassifizierung der Daten eine „Arbeits-GUI" erstellt, mit der die Klassifizierungsergebnisse einfacher erstellt werden konnten. Mithilfe dieser GUI können jeweils der zu berechnende VI und MLA ausgewählt werden, dessen Ergebnis anschließend als Asset oder im Google Drive-Konto hochgeladen werden kann.

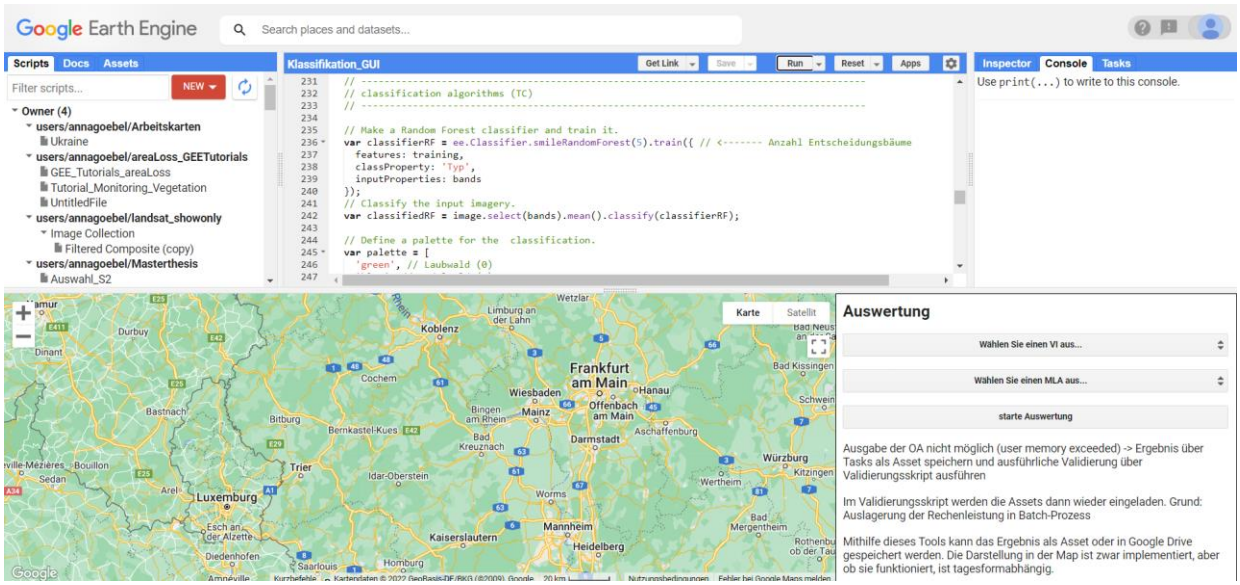


Abbildung 13: GUI der Klassifizierung

Grundlage der Klassifizierung bildeten die zwei verschiedenen Vegetationsindizes, die anhand von S2-Satellitendaten berechnet wurden. Die Satellitenbilder wurden direkt im Skript eingebunden. Dabei wurden lediglich die Bänder abgerufen, die zur Berechnung der Vegetationsindizes benötigt werden. Für den NDVI sind die S2-Bänder 4 (red) und 8 (VNIR) relevant, zur Berechnung der TC-Ergebnisse *wetness, greenness* und *brightness* werden die S2-Bänder 2 (blue), 3 (green), 4 (red), 8A (VNIR), 11 (SWIR) und 12 (SWIR) benötigt (vgl. Tabelle 1). Es wurden nur die Szenen ausgewählt, die sich ganz oder teilweise innerhalb der Grenzen von Rheinland-Pfalz befinden. Bei der Verwendung von Satellitendaten für Vegetationsanalysen wie der Klassifikation von Laub- und Nadelwald ist der Zeitpunkt der Aufnahme besonders wichtig. Der optimale Zeitpunkt liegt zwischen Mitte Februar und Ende April, also bevor die Laubbildung beginnt, da dann der Unterschied in den Reflexionsgraden von Laub- und Nadelwald am größten ist. Bilder aus dem Winter wären theoretisch auch geeignet, aber zu dem Zeitpunkt ist der Sonnenstand zu niedrig und dadurch keine radiometrische Korrektur möglich (NINK et al. 2019: 5). Zusätzlich mussten über die Datumsangabe Aufnahmezeitpunkte mit hohen Wolken- oder Schneebedeckungen herausgefiltert werden, d.h. die Zeitspanne musste für jedes Jahr individuell verkleinert werden. Die Auswahl der Zeitspannen erfolgte über die Angabe von *startDate* und *endDate* in den Metadaten. Die optimalen Ergebnisse lieferten Satellitendaten aus den folgenden Zeitspannen:

Tabelle 4: Zeitspannen der S2-Daten der jeweiligen Jahre

Jahr	startDate	endDate
2018	2018-02-15	2018-04-30
2019	2019-02-15	2019-04-30
2020	2020-02-15	2020-03-31
2021	2021-03-15	2021-03-31
2022	2022-02-15	2022-03-10

Gespeichert wurden die Szenen der jeweiligen Jahre in `ImageCollections`, die auf die Grenzen von Rheinland-Pfalz zugeschnitten wurden. Da die Bilder teilweise noch immer hohe Wolkenbedeckungen aufwiesen, wurden für die Szenen der jeweiligen Jahre Wolkenfilter implementiert. In der Vorbereitung der Wolkenfilterung wurden nur die S2-Bilder eingeladen, die eine Wolkenbedeckung von weniger als 40% aufwiesen (`ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',40)`). Innerhalb dieser Bilder wurden anschließend über das zusätzliche Band SCL (Scene Classification) des S2-Layers Pixel mit den im Folgenden angegebenen Werten ausgeschnitten. Wie der Tabelle zu entnehmen ist, wurden auch Wolkenschatten und Schnee/Eis herausgefiltert.

Tabelle 5: Werte des S2A-SCL-Bands (GOOGLE DEVELOPERS o. J.)

Wert	Beschreibung
2	Dark Area Pixels
3	Cloud Shadows
7	Clouds Low Probability / Unclassified
8	Clouds Medium Probability
9	Clouds High Probability
10	Cirrus
11	Snow / Ice

Im Anschluss wurden aus den vorbereiteten S2-Layern jeweils ein NDVI- und ein TC-Layer erstellt. Für die Implementierung des NDVI existiert in GEE eine Funktion, die über folgenden Befehl aufgerufen werden kann: `ee.Image.normalizedDifference()`. Der NDVI wird anhand der S2-Bänder 8 und 4 berechnet. Das Ergebnis wird als Band ‚NDVI' einem neu erstellten Objekt vom Typ `Image` angehängt. Wie in Kapitel 2.3.2 beschrieben, kann mit der TC-Transformation ein *greenness*-, ein *brightness*- und ein *wetness*-Faktor berechnet werden. Dafür werden die Koeffizienten nach CRIST und CICONE (1984) verwendet. Zur Berechnung der Faktoren werden von den wolkenfreien S2-Layern die Bänder 2, 3, 4, 8A, 11 und 12 selektiert und die Rückgabewerte der Bänder mit den jeweiligen Koeffizienten multipliziert und

diese wiederum miteinander addiert. Die Ergebnisse wurden anschließend einem neuen `Image` als Bänder angehängt. Produkt der Berechnung der Vegetationsindizes sind zehn `Images` (zwei VIs * fünf Jahre).

Die Ergebnisse wurden im Folgenden verwendet, um mithilfe dreier verschiedener MLA Klassifizierungen durchzuführen. Alle verwendeten Algorithmen sind der *pixel-based supervised classification* zuzuordnen (vgl. Kapitel 2.4). Als Trainingsdaten für die Klassifizierung aller zehn Kombinationen wurden, wie eingangs beschrieben, die Polygone aus den Forsteinrichtungsdaten (WöFIS) aus dem Jahr 2016 verwendet, die in QGIS bearbeitet, gefiltert und mit Labels versehen wurden (0 = Laubwald, 1 = Nadelwald, 2 = kein Wald). Der Schlüssel wurde in GEE über die Funktion `sampleRegions()` pixelweise (mit 10 m Auflösung) jeweils dem NDVI- und TC-*Image* als Bänder zugewiesen. Von den Pixeln, die übereinander lagen (weil sich die ausgewählten Szenen teilweise überschneiden), wurde über `mean()` jeweils der mittlere Pixelwert verwendet. Anhand der Trainingsdaten wurden anschließend die Bilder des gesamten Bundeslandes klassifiziert. Dies geschieht in GEE über den Aufruf der folgenden Funktionen, die jeweils ein leeres Objekt des Typs `ee.Classifier` kreieren (kursiv geschriebene Übergabeparameter sind optional):

1. `smileRandomForest(numberOfTrees, *variablesPerSplit*, *minLeafPopulation*, *bagFraction*, *maxNodes*, *seed*)`
2. `smileCart(*maxNodes*, *minLeafPopulation*)`
3. `smileGradientTreeBoost(numberOfTrees, *shrinkage*, *samplingRate*, *maxNodes*, *loss*, *seed*)`

Die Anzahl der Entscheidungsbäume muss über den Übergabeparameter `numberOfTrees` der Klassifikationsfunktionen `smileRandomForest` und `smileGradientBoost` angegeben werden. Das Objekt vom Typ `ee.Classifier` wird dann der Funktion `train()` übergeben, zusammen mit den Angaben für die `features`, die für die Klassifizierung verwendet werden (der vorher festgelegte Trainingsdatensatz als `Image`), der `classProperty` (Labels, die in der Eigenschaft ‚Typ' des `Images` gespeichert sind) und den `inputProperties`, die betrachtet werden sollen (Bänder ‚brightness', ‚greenness' und ‚wetness', die dem *Image* angehängt wurden). Mithilfe dieses Trainingsdatensatzes wurde schließlich die Klassifizierung über die Funktion `classify()` durchgeführt:

```
…
var training = image.select(bands).mean().sampleRegions({
    collection: laubnadelwald,
```

```
    properties: ['Typ'],
    scale: 10
  });
var classifier = ee.Classifier.smileRandomForest(5).train({
    features: training,
    classProperty: 'Typ',
    inputProperties: bands
});
var classifiedRF = image.select(bands).mean().classify(classifier);
…
```

Produkt der Klassifikationsdurchgänge sind insgesamt 30 `Images` (zwei VIs * fünf Jahre * drei MLA). Diese *Images* wurden auf die Waldmaske zugeschnitten und mit einer eindeutigen Bezeichnung als Asset in der Cloud gespeichert. Die Pixelanzahl musste dabei auf 1 Mrd. Pixel hochgesetzt werden, da der Standardwert von 100 Mio. unter der benötigten Pixelanzahl liegt.

2.5.2 Validierung

Die Validierungsschritte fanden in einem weiteren Skript statt. Zunächst wurde die jeweilige `ImageCollection` mit der entsprechenden Kombination (z.B. NDVI + GTB + 2020), die aus dem auf die Waldmaske zugeschnittenen Klassifizierungsergebnis besteht, eingebunden. Die Labels aus dem Testdatensatz (LWI-Daten) wurden anschließend wieder über die `sampleRegions()`-Funktion den Pixeln der `ImageCollection` angehängt. Aus diesem Datensatz wurde im nächsten Schritt eine Konfusionsmatrix der Inhalte aller Pixel erstellt, die sich mit den Punkten im Testdatensatz überschnitten (in dem Fall waren das die Eigenschaften ‚Typ' mit den Klassen im Testdatensatz und ‚Classification' mit den Ergebnissen der Klassifikation). Anhand dieser Matrix konnten schlussendlich die Ergebnisse in Form der *overall accuracy* (OA), *producer's accuracy* (PA), *user's accuracy* (UA) und des Kappa-Index berechnet werden. Die Ergebnisse wurden nach und nach in einem Array gespeichert, sodass sie für jede Berechnungskombination ausgegeben werden können. Da für diese Arbeit eine Vielzahl von Parameteränderungen getestet werden mussten, wurde für die Validierung ebenfalls eine „Arbeits-GUI" erstellt, mit der die Validierungsergebnisse einfacher abgerufen werden konnten (s. Abbildung 14).

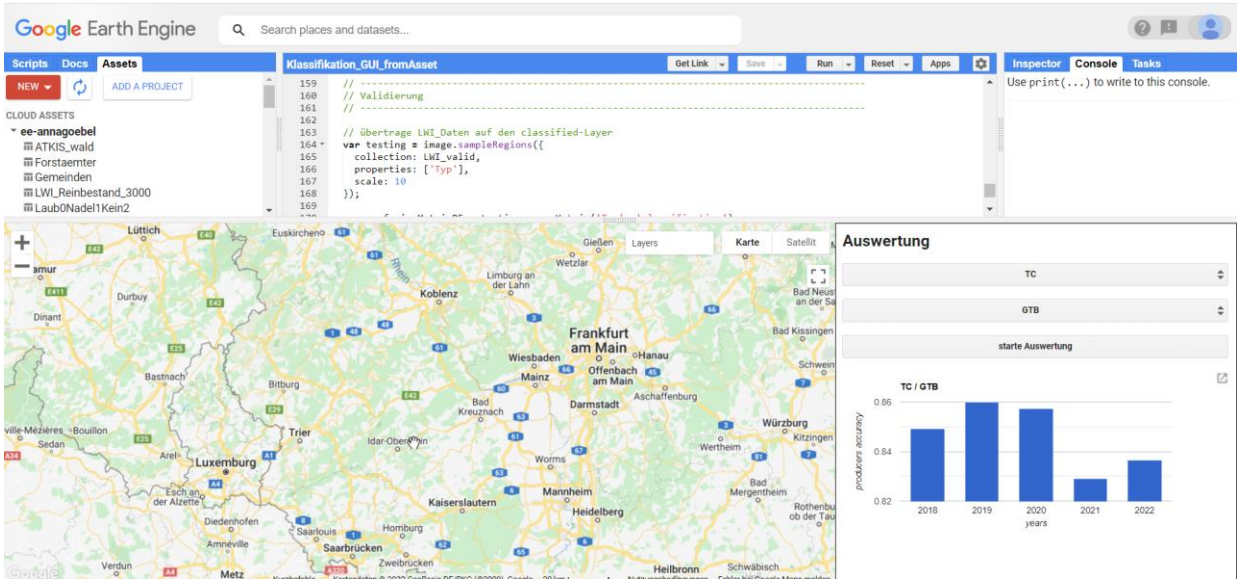


Abbildung 14: GUI der Validierung

Die Validierung der Ergebnisse erfolgte zum einen anhand des ‚klassischen' *accuracy assessment* nach STEHMAN (1997), zum anderen wurden die Ergebnisse anhand des Jaccard-Index überprüft. Das *accuracy assessment* ist ein Verfahren zur quantitativen Beschreibung der Klassifikation, das nicht nur eine statistische Aussage über den Anteil richtig klassifizierter Pixel gibt, sondern auch „Angaben über die gegenseitigen Abhängigkeiten von Fehlklassifikationen zwischen den verschiedenen Klassen" (LANGE 2020: 492) macht. Dabei wird das Klassifikationsergebnis mit Testdaten aus korrekt angenommenen Quellen (z.B. eigene oder externe Kartierungen) verglichen, wobei nur Testdatensätze verwendet werden dürfen, die vorher nicht bereits als Trainingsdaten eingesetzt wurden. Dafür werden entweder verschiedene Stichproben aus derselben Quelle verwendet, oder (wie in diesem Fall) zwei Datensätze aus unterschiedlichen Quellen genutzt. Die Mindestanzahl der genutzten Pixel sollte je Klasse nicht unter 50 liegen (LANGE 2020: 492).

Die Ergebnisse der Klassifikation werden den Referenzinformationen gegenübergestellt und in einer Konfusionsmatrix (`ee.confusionMatrix` in GEE) aufbereitet. Diese Matrix ermöglicht die Betrachtung der Klassifikationsergebnisse aus verschiedenen Perspektiven. Die OA (`ee.ConfusionMatrix.accuracy()`) wird durch den Quotienten aus allen korrekt klassifizierten Pixeln und der Gesamtzahl der Testpixel abgeschätzt. Die PA (`ee.ConfusionMatrix.producersAccuracy()`) errechnet sich durch die Division der Anzahl der richtig klassifizierten Pixel durch die Summe aller Pixel in derselben Klasse der Testdaten. Sie gibt

die Genauigkeit aus der Herstellersicht wieder. Durch Division der richtig klassifizierten Pixel durch die Summe der Pixel, die bei der Klassifikation derselben Klasse zugeordnet wurden, errechnet sich die UA; die Genauigkeit aus Benutzersicht (`ee.ConfusionMatrix.consumersAccuracy()`). Die Quantifizierung der Übereinstimmung der klassifizierten Daten mit den Referenzdaten erfolgt über die Errechnung des Kappa-Koeffizienten (`ee.ConfusionMatrix.kappa()`), der sowohl die falsch-positiven, als auch die falsch-negativen Klassifizierungen in die Beurteilung der Klassifikationsgüte einbezieht (LANGE 2020: 492f., GOOGLE DEVELOPERS 2021). Die Kappa-Werte schwanken zwischen 0 und 1, wobei ein Wert von 0 keine und ein Wert von 1 eine komplette Übereinstimmung bedeutet. Je höher der Kappa-Wert, desto höher ist also die Übereinstimmung des Klassifikationsergebnisses mit den Testdaten. Werte von Kappa größer als 0,75 deuten eine sehr gute, Werte kleiner als 0,4 eine schlechte Klassifizierungsgenauigkeit an (LANGE 2020: 493).

Die Kombination aus VI und MLA mit den Parametern, die das Ergebnis mit der höchsten Genauigkeit erzielt, wird für die Zeitreihendarstellung in der GUI eingebunden und verwendet. Dazu wird nach der Validierung die Dateienbezeichnung des Ergebnisses im Skript der GUI mit der jeweiligen Kombination händisch geändert, sodass nur diese Ergebnisse aus den jeweiligen Jahren im Skript eine Verwendung finden. Die lt. *accuracy assessment* besten Ergebnisse werden nicht nur zur weiteren Verarbeitung im anschließenden Skript genutzt, sondern zusätzlich über den Jaccard-Index validiert.

Die Methode der Quantifizierung des Klassifikationsergebnisses über den Jaccard-Index ist auch bekannt als Intersect over Union (IoU) und wurde von JACCARD (1902) entwickelt. In ArcGIS Pro wird dafür das *Intersect*-Tool verwendet. Es führt eine Verschneidung der eigenen Klassifikationsergebnisse mit den Ergebnissen von NINK et al. (2019) durch. Anschließend werden die jeweiligen Flächengrößen ermittelt und daraus die IoU-Faktoren (insgesamt und je Klasse) berechnet. Dabei wird angenommen, dass diese Ergebnisse 100% valide sind und sich auf die Jahre 2018-2022 übertragen lassen. In der Arbeit von NINK et al. (2019) wurde lediglich zwischen Laub- und Nadelwald unterschieden und es wurden nicht, wie in dieser Arbeit, zusätzlich die Nicht-Wald-Flächen betrachtet. Deswegen werden bei der Berechnung des Jaccard-Index die Nicht-Wald-Flächen außer Acht gelassen und nur die beiden Klassen Laub- und Nadelwald verglichen.

Das IoU-Verhältnis wird als Schwellenwert verwendet, um zu ermitteln, ob ein vorhergesagtes Ergebnis ein True Positive oder ein False Positive ist. Der IoU-Wert bildet das Verhältnis des

Überlappungsbereichs des vorhergesagten Objekts und der Referenzdaten zum Gesamtgebiet beider Flächen ab (ESRI o. J., s. Abbildung 15).

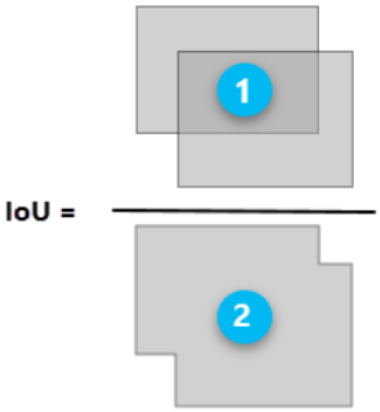


Abbildung 15: Berechnung des IoU-Verhältnisses (ESRI o. J.)

Um das beste Ergebnis in ArcGIS Pro zur Berechnung des Jaccard-Index verwenden zu können, muss es aus GEE heruntergeladen werden. Um die Bilder downloadbar zur machen, müssen sie zusätzlich zur Speicherung als Asset auch im Google Drive-Konto gespeichert werden. Standardmäßig wird als null-Wert der Pixel (also alle Pixel außerhalb der Wälder, die nicht klassifiziert sind) 0 eingetragen. Dieser Wert ist allerdings bereits als Schlüssel für die Klasse Laubwald besetzt. Deshalb müssen alle Pixelwerte vor der Speicherung geändert werden. Dazu wird die Funktion `ee.Image.remap()` ausgeführt, wobei alle Pixel mit dem Wert 2 (Nicht-Wald) zu 3 geändert werden, der Wert 1 (Nadelwald) zu 2 geändert und der Wert 0 (Laubwald) zu 1 geändert. Dadurch entsteht ein Bild mit vier unterschiedlichen Pixelwerten, wobei 0 nun den Null-Wert darstellt. Die Speicherung im Google Drive-Konto erfolgt über den Aufruf der Funktion `Export.image.toDrive()`. Anschließend kann die Datei heruntergeladen werden. In ArcGIS Pro erfolgt dann die weitere Verarbeitung: Konversion der Rasterdaten in Vektordaten (Shapefiles), Löschen der Null-Werte, Export der Laub- und Nadelwaldflächen in neue Shapefiles und schließlich die Ausführung des *Intersect*-Tools. Die Ermittlung der Flächengrößen erfolgt über *calculate geometry* (*area*) und der Export der Attributtabelle über *Table to Excel*. Die Berechnung der IoU-Verhältnisse wird dann in Excel vorgenommen.

2.5.3 Grafische Benutzeroberfläche

Im Skript der GUI wurden alle Assets eingebunden, die zur Darstellung und/oder Analyse benötigt werden, und zwar die Bundesland-, Gemeinde- und Forstamtsgrenzen. Diese drei Layer werden zur Orientierung für die User in der Karte dargestellt. Die Einbindung der Klassifikationsergebnisse aus dem Klassifikationsskript erfolgt innerhalb einer Schleife, nachdem das Polygon ausgewählt wurde. Auf Basis dessen kann die Auswertung erfolgen. Die Konfiguration, also die Auswahl der Kategorie (Auswahl des Polygons auf Basis der Gemeinde- oder Forstamtsgrenzen oder Zeichnen eines eigenen Polygons) wird vom User innerhalb eines `ui.Panel` mit einem `ui.select`-Feld und einem ‚berechnen'-Button vorgenommen (s. Abbildung 16). Im Panel findet man außerdem eine kurze Anleitung zur Verwendung des Tools und später das Ergebnis der Zeitreihenanalyse. Die Anwendung mit dem Namen „Waldscanner 2022" ist über folgenden Link aufrufbar:

https://annagoebel.users.earthengine.app/view/waldscanner-2022

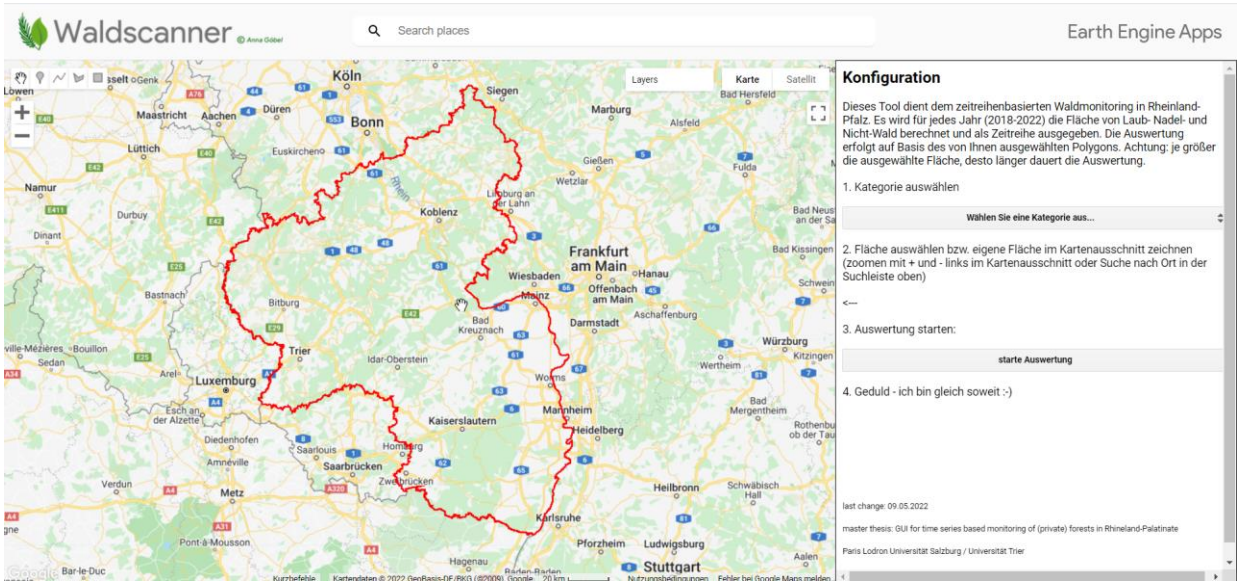


Abbildung 16: Initialdarstellung der GUI

Wenn als Kategorie entweder die Gemeinde- oder Forstamtsgrenzen ausgewählt werden, wird der jeweilige Layer sichtbar geschaltet und es kann über die Zeichenfunktion der Oberfläche (`drawingTools`) in der Karte ein Punkt gesetzt werden. Dieser Punkt durchläuft anschließend einen *spatial join* mit dem jeweiligen Layer (Abfrage, welches Polygon sich mit dem

gesetzten Punkt überschneidet). Die Bezeichnung des ausgewählten Polygons (z.B. „Gemeinde Hillscheid“ oder „Forstamt Cochem-Zell“) wird nach der Auswahl in einem Panel im oberen Bereich des Kartenausschnitts angezeigt. Wenn im select-Feld ‚eigenes Polygon zeichnen‘ ausgewählt wird, öffnen sich die `drawingTools` zum Zeichnen eines Polygons. Dieses wird direkt für die weiteren Berechnungen verwendet. Wenn das Polygon fertig konfiguriert ist, kann der User auf ‚berechnen‘ klicken und die Zeitreihenanalyse startet. Das Panel wird dann geleert und mit der Zeitreihe und ergänzenden Ausführungen zu den Ergebnissen gefüllt (s. Abbildung 17).

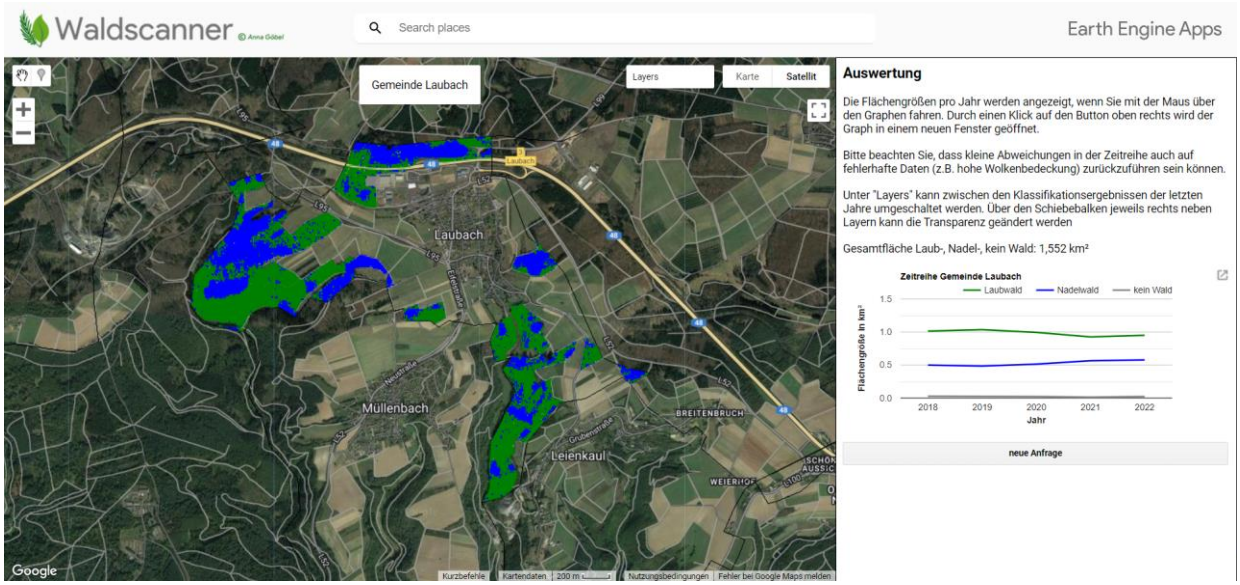


Abbildung 17: Zeitreihendarstellung (hier: Gemeindeebene)

Die drei Skripte (Klassifizierungsskript, Validierungsskript, Skript für GUI) wurden in einem GitHub-Repository gespeichert. Zur Verwendung der Skripte in GEE müssen sie lediglich in den GEE-Code-Bereich kopiert werden und die Pfade zu den Grundlagendaten (Assets) abgeändert werden. Das öffentliche Repository ist über folgende URL aufrufbar:

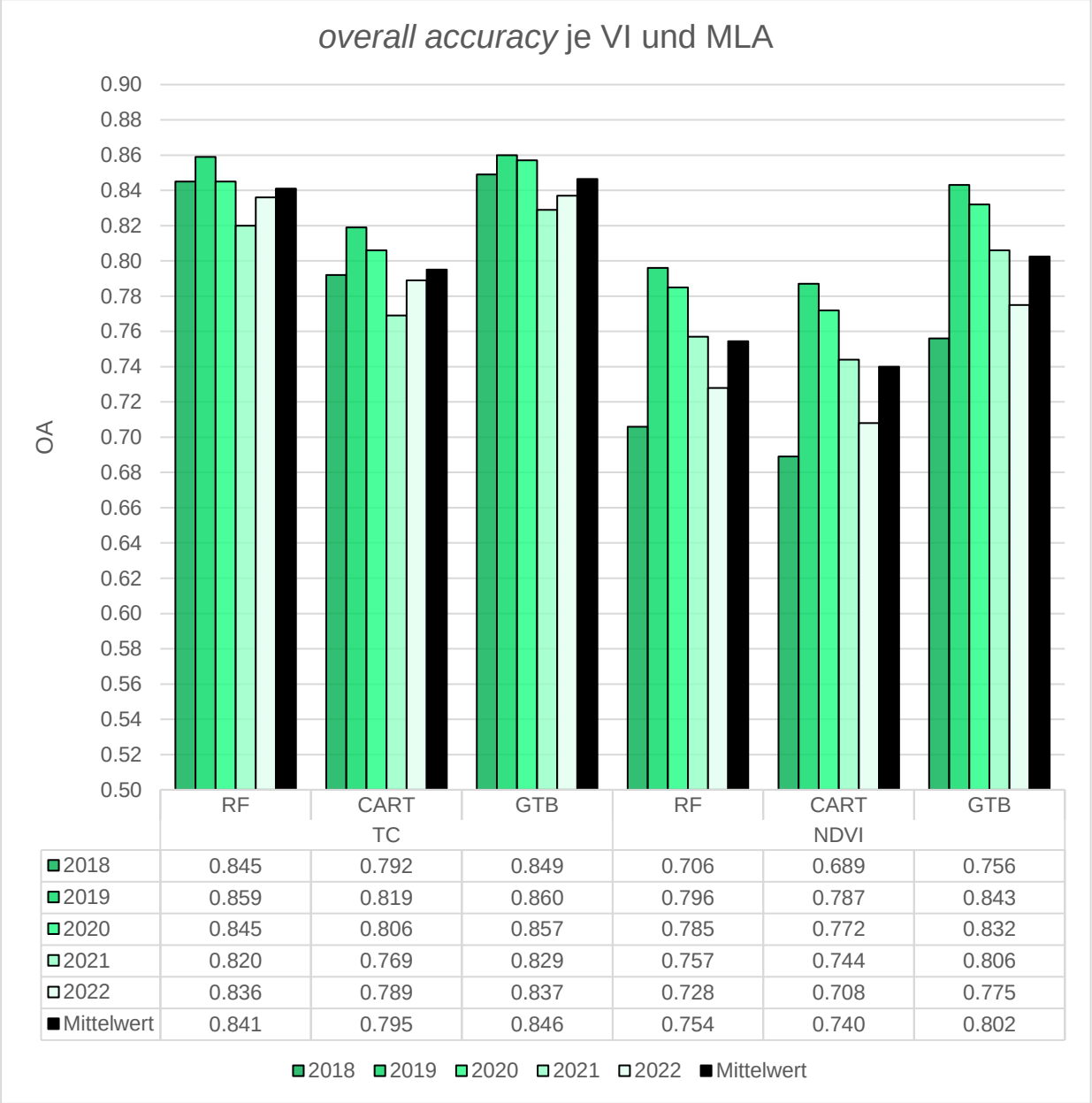
https://github.com/annagoebel/waldscanner

3 Ergebnisse und Diskussion

Im folgenden Kapitel werden zunächst die Ergebnisse der Validierungsmethode nach STEHMAN (1997) und die Resultate der IoU-Validierungsmethode beschrieben und es wird dargestellt, inwiefern sich Parameteränderungen auf die Genauigkeiten auswirken. Zudem wird darauf eingegangen, welche Hürden im Implementierungsprozess überwunden werden mussten. Anschließend werden die Ergebnisse der Implementierung der Zeitreihe innerhalb der GUI vorgestellt. Dabei wird nicht nur auf die Unterschiede in den Klassifikationsergebnissen von Flächen mit und ohne Änderungen eingegangen, sondern auch auf Flächen, für die eine Falschklassifikation erwartet wurde (Flächen mit starker Wolkenbedeckung und Lärchenwälder). Schließlich erfolgen die Analyse und Interpretation der Ergebnisse.

3.1 Ergebnisse der Validierung

Die Konfusionsmatrizen als Ergebnis des *accuracy assessment* können im Anhang eingesehen werden. Eine grafische Darstellung der OA aller Kombinationen erfolgt in Abbildung 18. Insgesamt wurden durch Anwendung des MLA CART die Ergebnisse mit der geringsten Genauigkeit und mit GTB die Ergebnisse mit der höchsten Genauigkeit erzielt. Zu einem ähnlichen Ergebnis kamen, wie bereits erwähnt, auch PONGANAN et al. (2021) und SUJUD et al. 2021: 9 in ihren Arbeiten. Die Kombination NDVI und CART erzeugte mit einer mittleren OA von 74,0% über alle fünf Jahre hinweg das Ergebnis mit der geringsten Genauigkeit. Das genauste Ergebnis lieferte TC und GTB mit einer mittleren OA von 84,6%, gefolgt von der Kombination TC und RF (84,1%). Wenn die Ergebnisse pro Jahr betrachtet werden, lieferten NDVI und CART des Jahres 2018 die geringste OA (68,9%) und die Kombination TC und GTB des Jahres 2019 die höchste Gesamtgenauigkeit (86,0%). Der Mittelwert der OA aller VI, MLA und Jahre liegt bei 79,7%. Im Durchschnitt lieferten die Ergebnisse des Jahres 2018 die geringste (77,3%) und des Jahres 2019 die höchste Gesamtgenauigkeit (82,7%).



	RF	CART TC	GTB	RF	CART NDVI	GTB
2018	0.845	0.792	0.849	0.706	0.689	0.756
2019	0.859	0.819	0.860	0.796	0.787	0.843
2020	0.845	0.806	0.857	0.785	0.772	0.832
2021	0.820	0.769	0.829	0.757	0.744	0.806
2022	0.836	0.789	0.837	0.728	0.708	0.775
Mittelwert	0.841	0.795	0.846	0.754	0.740	0.802

Abbildung 18: Ergebnistabelle (OA je VI und MLA)

Wie in Kapitel 2.5 beschrieben, bestand der Validierungsdatensatz (LWI-Daten) aus insgesamt 3000 Punkten (Laubwald: 1752 Punkte Laubwald, Nadelwald: 1214 Punkte, Nicht-Wald: 34 Punkte). Wenn man das beste Ergebnis betrachtet (TC + GTB, Jahr 2019), flossen insgesamt 2933 Punkte in die Validierung ein. Das liegt daran, dass sich in dem Fall die fehlenden 67 Punkte räumlich außerhalb des Klassifizierungsergebnisses befinden. Dies geschieht, wenn sich beispielsweise einer der Punkte unter Wolken oder im Schatten befinden oder weil sie

sich nicht innerhalb der in den ATKIS-Daten als Wald definierten Flächen befinden und sie deswegen aus der Szene ausgeschnitten wurden.

Aus Produzentensicht wurde die Klasse Laubwald annähernd perfekt abgebildet (93,6%), was bedeutet, dass sich die reale Klasse Laubwald zu 93,6% in den klassifizierten Daten wiederfindet. Die PA der Klasse Nadelwald liegt bei 77,6% und die Nicht-Wald-Klasse bei 0%. Auch aus Benutzersicht wurde bei der Nicht-Wald-Klasse eine UA von 0% errechnet. Keine der Nicht-Wald-Flächen aus dem Klassifikationsergebnis stimmt also mit den Referenzdaten überein und umgekehrt. Die UA der Klassen Laub- und Nadelwald liegen hingegen bei 85,3% bzw. 89,1%. Die UA besagt, zu welchem Prozentanteil bzw. mit welcher Wahrscheinlichkeit ein Benutzer des Klassifikationsergebnisses im Gelände tatsächlich auf die betreffende Klasse trifft. Trotz der falschen Klassifikation der Nicht-Wald-Flächen konnte diese Kombination des Jahres 2019 das beste Ergebnis liefern. Auch der Kappa-Wert ist der höchste aller Kombinationen. Mit einem Kappa-Wert von 71,2% ist das Klassifikationsergebnis als gut bis sehr gut einzuschätzen (vgl. Konfusionsmatrix im Anhang, S. 57).

Um dieses Ergebnis in Relation setzen zu können, seien hier noch die Genauigkeitswerte des lt. OA schlechtesten Ergebnisses (NDVI + CART, Jahr 2018) genannt. Die OA liegt bei 68,9% und der Kappa-Wert bei 37,6%. Damit ist das Klassifikationsergebnis als schlecht einzustufen. Die UA für Laub-, Nadel- und Nicht-Wald liegen bei 72,2%, 70,7% und 0,9%. Die PA der drei Klassen liegen bei 80,1%, 55% und 2,9%. Die höheren Werte der UA und PA bei der Nicht-Wald-Klasse begründen sich dadurch, dass einer der 34 Nicht-Wald-Punkte in den Referenzdaten auch als Nicht-Wald klassifiziert wurde (vgl. Konfusionsmatrix im Anhang, S. 59).

Wie in der Methodenbeschreibung erwähnt, wurde im Laufe des Implementierungsprozesses die Anzahl der Flächen im Trainingsdatensatz um das Dreifache erhöht, während die Anzahl der Entscheidungsbäume aufgrund der beschränkten Rechenkapazität bei den Algorithmen RF und GTB von 25 auf 5 herabgesetzt werden musste. Bei einer Anzahl von 537 Flächen im Trainingsdatensatz und 25 Entscheidungsbäumen lag die OA zwischen 67,3% (TC und CART) und 78,4% (TC und RF). Nach der Anpassung der Parameter (5 Entscheidungsbäume, 1458 Trainingsflächen) konnte eine Erhöhung der OA bei allen Kombinationen festgestellt werden.

Durch die Implementierung eines Minuspuffers von 10m auf die Trainingsflächen konnte entgegen der Erwartung keine signifikante Verbesserung des Ergebnisses beobachtet werden. Stattdessen fand eine leichte Verschlechterung der Ergebnisse statt. Beispielsweise

verschlechterte sich die OA bei den Kombinationen TC und CART um -0,05%, die Kombination TC und GTB verschlechterte sich durch den Minuspuffer um -0,53%.

Da in dieser Arbeit ein Augenmerk auf der Nicht-Wald-Klasse lag, da Veränderungen im Wald abgebildet werden sollten, soll hier noch auf das Ergebnis eingegangen werden, das die Nicht-Wald-Klasse aus Benutzer- und Produzentensicht am besten abbildet. Dabei handelt es sich um das Ergebnis der Kombination TC + CART aus dem Jahr 2022. Die OA liegt bei 78,9% und der Kappa-Wert bei 58,0%. Damit kann das Ergebnis als akzeptabel eingestuft werden. Die UA der Nicht-Wald-Klasse liegt bei 4,3% und die PA derselben Klasse bei 11,8%. Das bedeutet, dass sich die reale Klasse Laubwald zu 11,8% in den klassifizierten Daten wiederfindet. Die Zuordnung zum Nicht-Wald stimmt bei vier der 34 Punkte überein (vgl. Konfusionsmatrix im Anhang, S. 57). Weil allerdings die Ergebnisse der anderen Jahre dieser Kombination vergleichsweise schlechte Ergebnisse lieferten, wird diese Kombination nicht für die weitergehenden Analysen genutzt.

Die Kombination, die das beste Ergebnis lieferte, wenn man lediglich die OA betrachtet, wurde im Hauptskript für die Erstellung der Zeitreihe verwendet. Dabei handelt es sich, wie bereits erwähnt, um die Kombination TC + GTB mit einer OA von durchschnittlich 84,6%. Das beste Klassifikationsergebnis des Jahres 2019 dieser Kombination wurde einer zusätzlichen Validierungsmethode unterzogen (IoU). Dabei wird abgebildet, inwiefern die räumliche Ausdehnung der Flächen der jeweiligen Klassen mit validierten und veröffentlichten Vergleichsdaten übereinstimmt. Als Vergleichsdatensatz wurden die Ergebnisse aus der Arbeit von NINK et al. (2019) verwendet. Dabei wurde davon ausgegangen, dass der Vergleichsdatensatz zu 100% mit der Realität übereinstimmt. Tatsächlich lag die OA dort bei 86% (NINK et al. 2019: 13). Insgesamt konnte ein IoU-Wert von 87,5% festgestellt werden. Das bedeutet, dass 87,5% der Flächen richtig klassifiziert wurden, wenn angenommen wird, dass es sich bei dem Referenzdatensatz von NINK et al. (2019) um reale Gegebenheiten handelt. Die Klasse Laubwald wurde mit ca. 91% etwas besser abgebildet als die Klasse Nadelwald (82,8%). Die Nicht-Wald-Klasse wurde nicht betrachtet, da sie nicht Bestandteil der Klassifizierung in der Arbeit von NINK et al. (2019) war.

Tabelle 6: Ergebnis IoU (Flächenangaben in ha)

		Ergebnisse NINK et al (2019)	
		Laubwald	**Nadelwald**
Klassifikationsergebnis dieser Arbeit	**Laubwald**	141576,57	20145,58
	Nadelwald	14011,53	96691,28
Summe aller Flächen		272424,965	
IoU gesamt (True Positives/Summe)		87,46%	
IoU Laubwald (True Positives/Summe)		90,99%	
IoU Nadelwald (True Positives/Summe)		82,76%	

3.2 Zeitreihendarstellung

Um die Ergebnisse der Zeitreihenberechnung, die in der GUI nach Auswahl des Polygons dargestellt werden, in Relation setzen zu können, werden im Folgenden zwei Zeitreihen abgebildet; eine aus einer Gemeinde, in der in der Vergangenheit keine nennenswerten Veränderungen im Wald erfolgten und die andere aus einer Gemeinde, in der im Zeitraum 2018-2022 massive Veränderungen festzustellen waren.

Bei der ersten Gemeinde wurde Achtelsbach an der saarländischen Grenze gewählt, deren Wald sich größtenteils in den Zonen 1a und 1b des Nationalparks Hunsrück-Hochwald befindet. In Zone 1a (Wildnisbereich) wird sich der Wald selbst überlassen, in Zone 1b (Entwicklungsbereich) dürfen nur minimal-invasive Maßnahmen stattfinden (NATIONALPARKAMT HUNSRÜCK-HOCHWALD o. J.). In der Zeitreihe sind keine großen Veränderungen in der Verteilung der Klassen zu erkennen. Die Flächengröße des Laub- und Nadelwaldes bleibt zwischen 2018 und 2020 annähernd gleich. Die Laubwaldfläche verkleinert sich zwischen 2020 und 2022 leicht, während die Flächengröße des Nadelwaldes leicht zunimmt. Bei den Nicht-Wald-Flächen ist fast keine Veränderung zu erkennen (s. Abbildung 19).

Bei der zweiten Gemeinde fiel die Wahl auf die Gemeinde Hillscheid im Westerwald auf der Montabaurer Höhe (nordöstlich von Koblenz). Vor allem die Fichtenwälder in diesem Gebiet waren in den Jahren 2018, 2019 und 2020 stark von Walddynamiken in Folge von z.B. klimawandelbedingten Borkenkäferkalamitäten betroffen. Es wurden Maßnahmen gegen die starke Umbruchsdynamik ergriffen, mit dem Ziel der waldbaulichen Weiterführung eines Dauerwaldes durch Naturverjüngung. Dazu gehört die Pflanzung von klimawandelangepassten Baumarten wie Esskastanie, Eiche, Linde etc. (NR-KURIER 2019). Bei der Zeitreihe dieser Gemeinde kann im Gegensatz zur Zeitreihe von Achtelsbach eine größere Veränderung der

Klassengrößen festgestellt werden. Laut Klassifikationsergebnis stieg die Fläche des Laubwaldes zwischen 2018 und 2021 kontinuierlich von ca. 5,2 km² auf ca. 7,8 km² an, während die Größe des Nadelwaldes von ca. 5,3 km² auf ca. 2,6 km² abnahm. Bei der Nicht-Wald-Flächengröße ist nur ein kleiner Anstieg von 0,1 auf 0,2 km² zu beobachten (s. Abbildung 20). Die Veränderungen des Waldes werden also auch im Klassifikationsergebnis und der Zeitreihe deutlich. Allerdings geben die Ergebnisse Grund zur Annahme, dass sich die Pixel, die 2018 noch als Nadelwald klassifiziert wurden, 2021 zur Klasse Laubwald zugeordnet wurden. In der Realität erfolgten allerdings Kahlschläge, Borkenkäferschäden etc., wodurch sich diese Pixel eigentlich über die Zeitspanne hinweg von Nadelwald zu Nicht-Wald hätten entwickeln müssen. Dem liegt entweder zugrunde, dass für den Trainingsdatensatz nur eine geringe Anzahl an Nicht-Wald-Flächen gewählt wurde und die Pixel deshalb eher der Klasse Laubwald zugeordnet werden oder dass die eingeschlagenen bzw. geschädigten Nadelwälder durch die Sukzession, also die sich inzwischen gebildete Pioniervegetation im Hinblick auf die reflektierten Werte eher dem Laub- als dem Nicht-Wald ähneln. Eine weitere Hypothese ist, dass auf den genannten Flächen nach der Schädigung bzw. dem Kahlschlag noch totes stehendes Holz vorhanden ist und die reflektierten Signale deshalb und in Kombination mit bereits vorhandener Strauchbedeckung eher denen des Laubwaldes entsprechen und die Flächen dadurch eher als Laubwald statt als Nicht-Wald klassifiziert werden.

Insbesondere in den Jahren 2019 und 2020 kam es vermehrt zu Borkenkäferschäden in Rheinland-Pfalz. Das zu erwartende Ergebnis für Gebiete, in denen solche Schäden auftraten, war, dass die Nadelwald-Flächengrößen in den beiden Jahren abnahmen und sich die Nicht-Wald-Flächen gleichzeitig vergrößerten, während die Laubwald-Flächengrößen mehr oder weniger gleichblieben. Stattdessen kann in vielen Fällen beobachtet werden, dass die Nadelwaldflächen mit Borkenkäferschäden nun statt der Nicht-Wald-Klasse der Laubwald-Klasse zugeordnet werden.

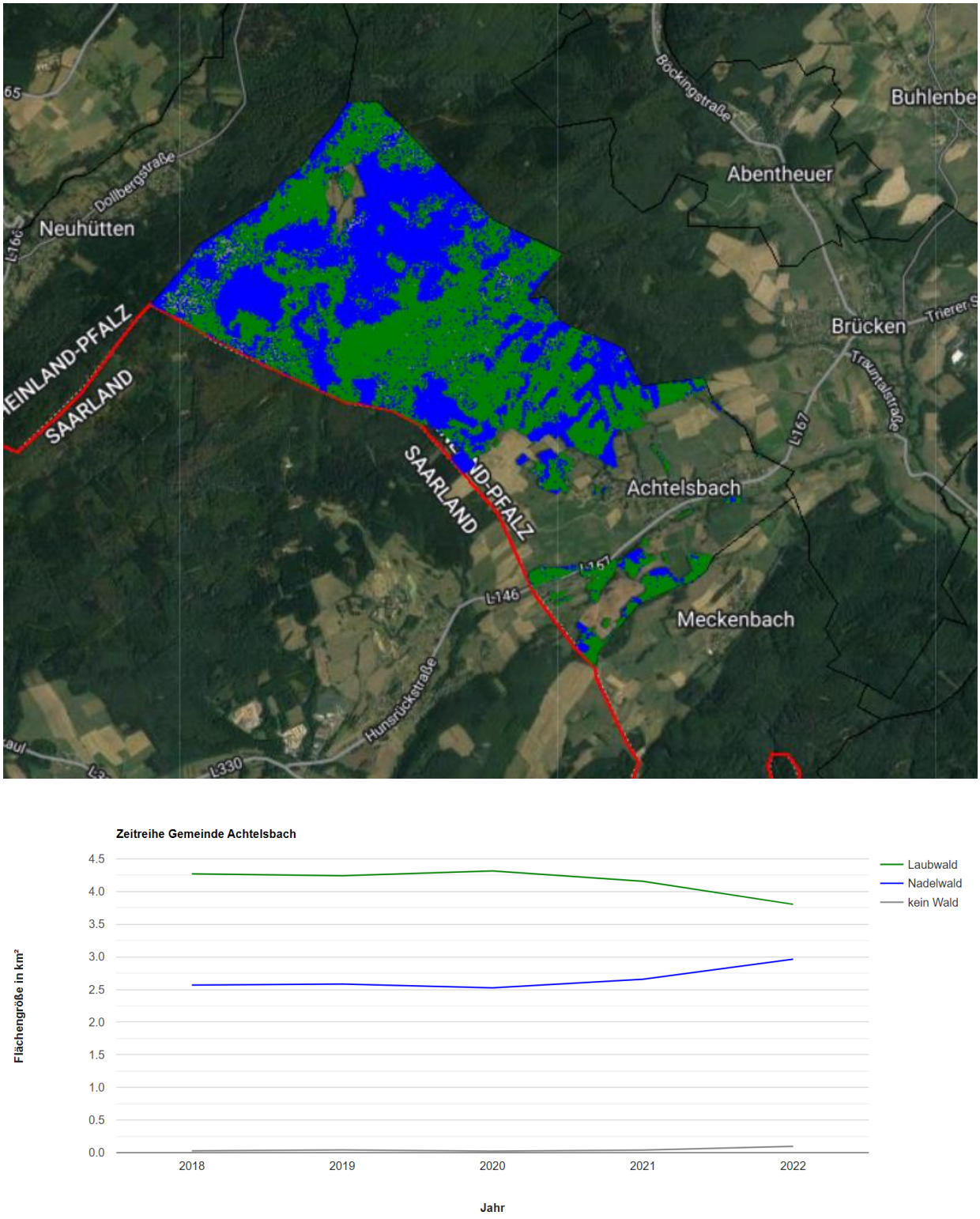
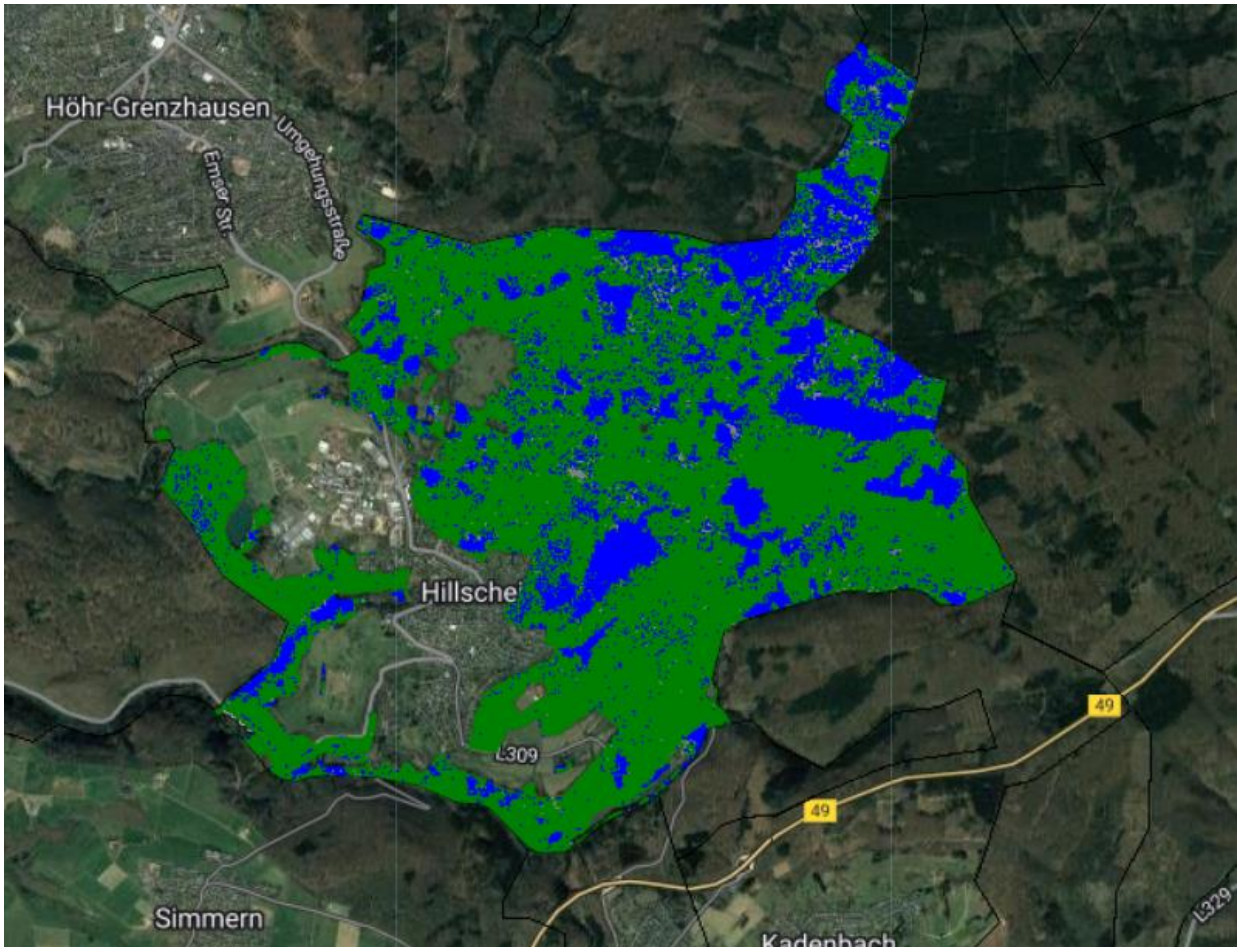


Abbildung 19: Gemeinde Achtelsbach – Klassifikationsergebnis 2022 und Zeitreihe



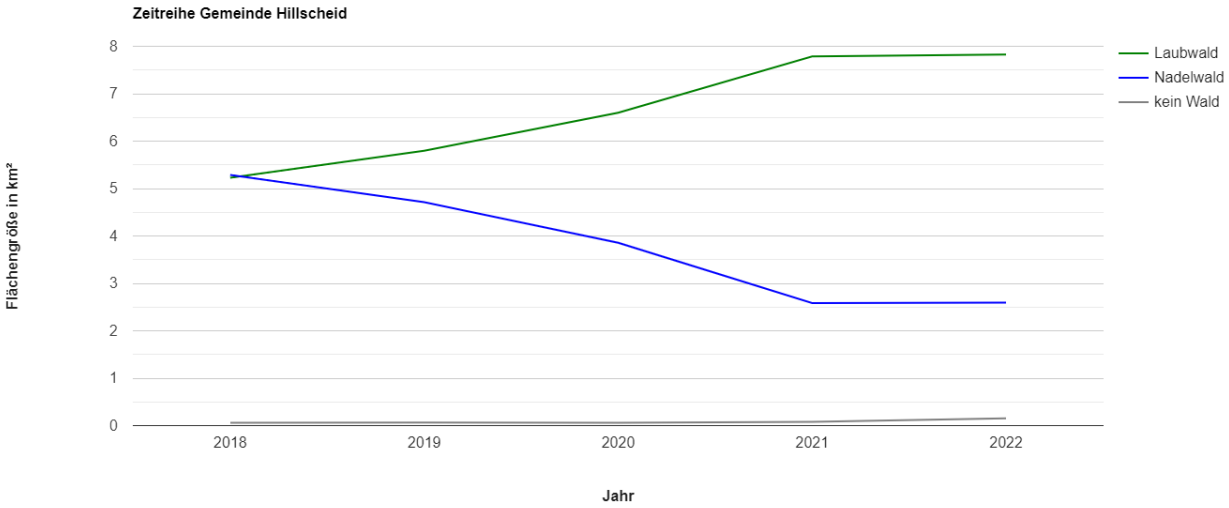


Abbildung 20: Gemeinde Hillscheid – Klassifikationsergebnis 2022 und Zeitreihe

Es soll hier außerdem beispielhaft auf Flächen eingegangen werden, für die eine hohe Fehleranfälligkeit erwartet wird. Dabei handelt es sich z.B. um reine Lärchenwälder, da die Lärche die einzige Nadelbaumart darstellt, die im Winter ihre Nadeln verliert. Dadurch unterscheidet sie sich hinsichtlich ihrer saisonalen Reflexionseigenschaften deutlich von anderen Nadelwald-Reinbeständen. Ein reiner Lärchenwald befindet sich beispielsweise an der luxemburgischen Grenze in der Gemeinde Körperich. In der nachfolgenden Abbildung ist das Klassifikationsergebnis der Gemeinde aus dem Jahr 2022 dargestellt. Wie der Abbildung zu entnehmen ist, wurde der Lärchenwald (gelb eingegrenzt) vollflächig als Laubwald (grün) klassifiziert, obwohl es sich faktisch um einen Nadelwald (blau) handelt. Da der Wald zum Zeitpunkt der Aufnahme nicht benadelt war, ähnelt er hinsichtlich der Reflexionswerte eher dem Laub- als dem Nadelwald.

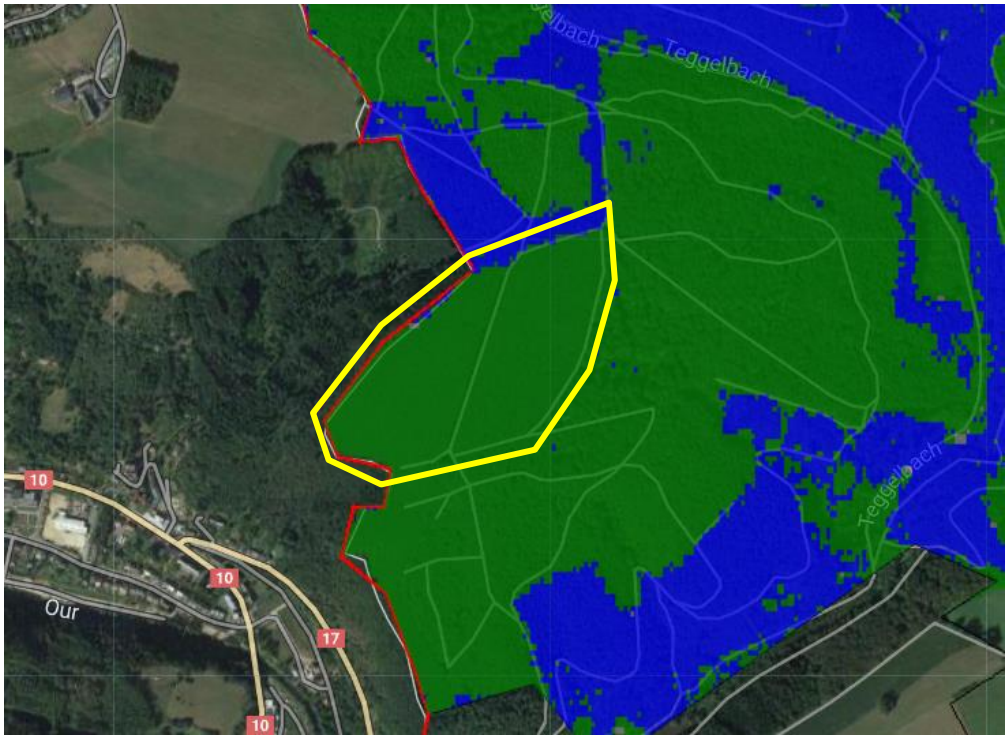


Abbildung 21: Ausschnitt des Klassifikationsergebnisses der Gemeinde Körperich (Lärchenwald gelb markiert)

Um fehlerhaft klassifizierte Flächen aufgrund von Wolkenbedeckung oder Schatten zu vermeiden, wurde im Vorfeld ein Wolkenfilter auf die S2-Rohdaten angewandt. Dabei wurden die betroffenen Pixel aus der Szene ausgeschnitten und nicht weiter betrachtet. Dadurch wurden allerdings auch Pixel ausgeschnitten, die in der Realität keine Wolken- oder Schattenbedeckung aufwiesen. Dazu gehören vor allem Straßen und vegetationslose Flächen wie Steilhänge im Bereich von Flüssen. Das Löschen dieser Flächen ist aber unproblematisch, da sie später bei der Verschneidung mit den ATKIS-Wäldern sowieso wegfallen. Problematisch ist aber das Löschen von Flächen, die fälschlicherweise als wolkenbedeckt definiert wurden und sich innerhalb von Wäldern befinden. Dadurch kann es beispielsweise sein, dass in der Zeitreihe abgebildet wird, dass in einem einzelnen Jahr ein drastischer Rückgang der Gesamtwaldfläche stattfand, obwohl in dem Jahr lediglich Pixel ausgeschnitten wurden. Im Gegenzug wurden auch teilweise Flächen fälschlicherweise nicht herausgefiltert, die in der Realität wolkenbedeckt waren. Die meisten fehlerhaft herausgefilterten (bzw. nicht herausgefilterten) Pixel traten im Jahr 2021 auf, da in dem Frühjahr dieses Jahrs eine besonders hohe Wolken- und somit auch Schattenbedeckung beobachtet werden konnte. In der folgenden Abbildung wird ein Ausschnitt aus einem S2-Datensatz aus dem Jahr 2021 dargestellt. Darin ist der Verlauf der Mosel und mittig die Stadt Cochem abgebildet. In diesem Ausschnitt befinden sich sowohl fälschlicherweise gelöschte Pixel (Straßen, vegetationsarme Gebiete und Waldflächen am Ufer der Mosel) als auch fälschlicherweise nicht gelöschte Pixel mit Wolkenbedeckung (im südöstlichen Bereich des Kartenausschnitts).

Abbildung 22: Ausschnitt S2-Szene nach Anwendung der Wolkenmaske

Wenn man sich in der GUI für die Stadt Cochem eine Auswertung über die letzten Jahre ausgeben lässt, fällt auf, dass Teile des Waldes im Jahr 2021 nicht klassifiziert wurden, im darauffolgenden Jahr allerdings schon (s. Abbildung 23). Dies ist die Folge von fälschlicherweise ausgeschnittenen Pixeln im Zuge der Wolkenfilterung. Da diese Veränderung auch in dem ausgegebenen Liniendiagramm abgebildet wird (s. Abbildung 24), muss für Laien, die nicht mit der Methode des Programms vertraut sind, angenommen werden, dass im Jahr 2021 eine signifikante Verkleinerung der Laub- und Nadelwaldflächen erfolgte. Durch diese fehlerhafte Darstellung der Zeitreihe in Bezug auf die Flächengrößen der drei Klassen besteht also die Gefahr, dass falsche Schlüsse gezogen werden.

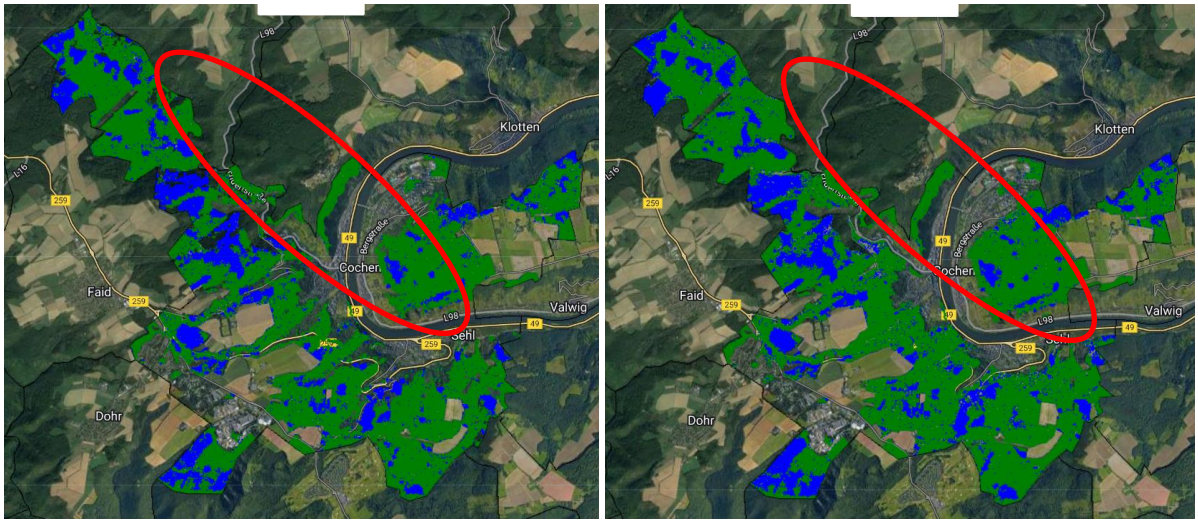


Abbildung 23: Klassifikationsergebnis der Stadt Cochem (links: 2021, rechts: 2022)

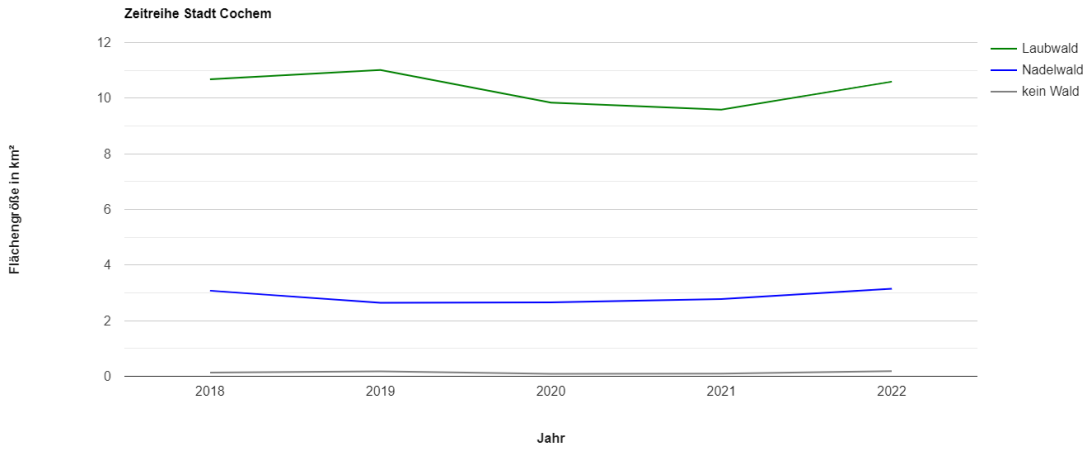


Abbildung 24: Zeitreihe der Stadt Cochem

3.3 Diskussion

Wie in Kapitel 2.2 beschrieben, wird die Rechenleistung pro User von GEE stark begrenzt. Wenn die Grenzen der Kapazität erreicht werden, wird ein Skript nicht mehr ausgeführt und es erscheint die Fehlermeldung „user memory limit exceeded“. Dies ist recht schnell der Fall, wenn zu viele Rechenprozesse gleichzeitig ausgeführt werden bzw. zu viele Anfragen auf einmal geschickt werden. Dies ist auch der Grund, warum die Parameter bei der Klassifizierung und Validierung stark begrenzt werden mussten. Dazu gehört das Herabsetzen der Referenzdatenmenge auf ca. 1500 und der Anzahl der Entscheidungsbäume auf 5. Außerdem musste das Programm auf drei getrennte Skripte aufgeteilt werden, um die Prozesse in Zwischenschritten durchzuführen und die GEE-Einschränkungen zu umgehen. Ursprünglich war die Implementierung in einem Skript und die Speicherung aller Zwischenschritte im Arbeitsspeicher geplant. Dies war aus den genannten Gründen nicht möglich, sodass die klassifizierten Rasterdaten über die Speicherung als Asset permanent gemacht werden mussten. Dies hatte einen erhöhten Zeitaufwand zur Folge, da der Upload eines Assets ca. 30 Minuten dauerte. Dadurch wurde es erschwert, eine hohe Anzahl an Parameteränderungen zu testen, weil für jede Änderung der erhöhte Zeitaufwand einkalkuliert werden musste. Die Höhe des Zeitaufwands für Uploads und weitere Ausführungen in der API waren tagesformabhängig. So konnte es sein, dass ein Skript morgens ohne Probleme und schnell ausgeführt werden konnte, wohingegen die Ausführung des identischen Prozesses am Abend zu einer Fehlermeldung und dem Abbruch der Kompilierung führte. Grund dafür ist vermutlich die erhöhte Nutzung der GEE zu bestimmten Zeiten (z.B. in den USA vormittags).

Der Grund dafür, dass die Klassifikation anhand des TC durchschnittlich bessere Ergebnisse lieferte als die Klassifikation anhand des NDVI ist vermutlich, dass dabei alle drei Parameter *brightness*, *greenness* und *wetness* in die Berechnung einfließen. Im Gegensatz dazu wird beim NDVI lediglich ein Faktor betrachtet. Zusätzlich werden beim NDVI nur zwei Wellenlängenbereiche einbezogen (NIR und red), während bei den TC-Faktoren sechs S2-Bänder mit unterschiedlichen Wellenlängenbereichen betrachtet werden. In diesem Fall lässt sich also sagen, dass je mehr Parameter verwendet werden, die die Klassen beschreiben, desto valider das Ergebnis.

Die Unterschiede in den Ergebnissen der drei Klassifikationsalgorithmen kommen dadurch zustande, dass sich von CART über RF zu GTB die Komplexität der Methode erhöht. Bei der CART-Methode wird lediglich ein Entscheidungsbaum erstellt, der fehleranfälliger in Relation

zu den anderen Methoden und sensibler gegenüber den Trainingsdaten ist, d.h. dass Ausreißer in den Referenzdaten einen hohen Einfluss auf die Definition der Grenzwerte zwischen den drei Klassen haben und sich in einem schlechteren Ergebnis widerspiegeln können. RF bildet zwar mehrere Entscheidungsbäume, aggregiert durch ein Voting die Ergebnisse aller Bäume und ist deshalb weniger fehleranfällig, allerdings reagiert dieser Algorithmus ebenfalls sensibel, sowohl im Hinblick auf die Anzahl der Entscheidungsbäume als auch im Hinblick auf die Referenzdaten, da auch hier Ausreißer in den Daten einen Einfluss auf die Entscheidung haben. In GEE ist die Erstellung einer hohen Anzahl an Entscheidungsbäumen nicht möglich, da sonst die für die User zur Verfügung stehende Rechenleistung überschritten wird. Da die Genauigkeit der Ergebnisse aber bei einer so geringen Anzahl an Entscheidungsbäumen bei Erhöhung der Anzahl stark ansteigt, macht es sich in der Genauigkeit bemerkbar, ob z.B. zehn statt fünf Entscheidungsbäume erstellt werden. GTB gehört zu den komplexesten und genauesten MLA, weil fehlerhafte Entscheidungen (durch z.B. Ausreißer) im *boosting*-Prozess eliminiert werden, da das finale Modell in Form eines Entscheidungsbaums aus den Fehlern der vorhergehenden Bäume (*weak learners*) lernt.

Neben der limitierten Rechenkapazität pro User, die zu einer Begrenzung der Anzahl an Entscheidungsbäumen und Größe des Trainingsdatensatzes führt, wird die Genauigkeit noch durch weitere Parameter im Prozess beeinflusst. Dazu gehört die Tatsache, dass der Trainingsdatensatz die quantitativen Anteile der drei Klassen nicht zu 100% repräsentiert. Der Anteil der Nicht-Wald-Flächen im Trainingsdatensatz entspricht lediglich 2,5%, wobei davon ausgegangen wird, dass dieser Anteil in der Realität höher liegt. Grund dafür ist die fehlende Attribuierung der Nicht-Wald-Flächen in den WöFIS-Daten und der geringen Anzahl von flächendeckend unbewaldeten Polygonen. Grundsätzlich kann ebenfalls nicht davon ausgegangen werden, dass sowohl Trainings- als auch Testdatensatz zu 100% der Realität entsprechen, da sie lediglich stichprobenartig über Luftbildabgleiche geprüft wurden. Eine weitere Annahme, die getroffen wurde, und die das Ergebnis beeinflussen könnte, ist, dass sich sowohl WöFIS- als auch LWI-Daten seit dem Zeitpunkt der Aufnahme (2016 bzw. 2012) nicht verändert haben. Methodisch korrekt wäre es, Trainings- und Testdaten individuell für jedes Jahr zu verwenden (z.B. Klassifikation von S2-Aufnahmen aus dem Jahr 2022 mit Trainingsdaten aus demselben Jahr und Validierung der Ergebnisse mit Testdaten aus demselben Jahr). Da die Kartierungen aber nicht jedes Jahr erfolgen, liegen nicht für alle Jahre Daten vor.

4 Fazit und Ausblick

Auf Grundlage der vorgehend vorgestellten Ergebnisse ist festzustellen, dass GEE sich mit einigen Einschränkungen für die Fragestellungen dieser Arbeit eignet. Die Programmierung in JavaScript gestaltet sich niederschwellig und bietet den Vorteil, dass über die cloudbasierte Lösung keine lokalen Rechenleistungen in Anspruch genommen werden müssen und die Daten nicht erst heruntergeladen werden müssen. Trotzdem ist GEE vorrangig für die Rasterdatenverarbeitung geeignet und gibt dabei einige Einschränkungen vor, weshalb die verwendeten Vektordaten extern vorbereitet werden müssen. Dazu gehört beispielsweise die Verringerung des Trainingsdatensatzes aufgrund der Beschränkung der Dateigröße und Rechenleistung oder die Verschmelzung der Flächen für die Waldmaske aufgrund der Obergrenze bei der Knotenanzahl von (Multi-)Polygondatensätzen.

GEE eignet sich nur bedingt für die Klassifizierung eines solch großen Untersuchungsgebietes wie RLP, da die dem User zur Verfügung stehende Rechenleistung stark begrenzt wird. Dadurch ist eine Vergrößerung von Parametern wie der Anzahl der Trainingsflächen und Entscheidungsbäume, die zu einer Verbesserung des Ergebnisses führen würde, nicht möglich. Auch die Parameter zur Validierung der Ergebnisse (Anzahl der Testpunkte) wird dadurch begrenzt. Eine Verringerung der in Anspruch zu nehmenden Rechenleistung könnte nur dadurch erreicht werden, dass grundlegende Parameter wie die Größe des Untersuchungsgebietes oder die räumliche Auflösung der Pixel angepasst werden. Insgesamt konnte das Hauptziel dieser Arbeit, die Erstellung einer grafischen Benutzeroberfläche, über die sich die User, also Privatwaldbesitzende oder Behörden, individuelle Auswertungen über die Waldtypenanteile in den gewünschten Flächen anzeigen lassen können, erreicht werden. Auch die Ausgabe der Verteilung der Waldtypen im gewünschten Gebiet ist über die Kartenansicht möglich. Die Zeitreihen können vom User pro angefragte Fläche heruntergeladen und für individuelle Zwecke gespeichert werden. Die Rasterbilder als Ergebnis der genauesten Klassifikation können vonseiten des Image Analyst (nicht für die User der GUI) für ganz RLP heruntergeladen und für weitergehende Berechnungen genutzt werden.

Um das Ziel der vorliegenden Arbeit zu erreichen, musste eine Vielzahl an Parameteränderungen im Prozess getestet werden. Damit die Berechnungen ausgeführt wurden, war die wichtigste Änderung die Verlagerung bestimmter komplexer Prozesse wie der Speicherung der Klassifikationsergebnisse im Rasterformat oder dem Export der Konfusionsmatrizen aus

dem Arbeitsspeicher über die Anwendung von Batch-Prozessen in die Cloud. Dadurch wurde ermöglicht, dass insbesondere große Dateien wie die Ergebnis-Rasterbilder in der Cloud als Asset gespeichert und im Programmcode der GUI nicht bei jeder Anfrage für ganz RLP neu erstellt, sondern nur noch über einen Aufruf aus den Assets abgerufen werden können. Einerseits können die Klassifikationen durch die Verlagerung der Prozesse aus dem Arbeitsspeicher in Batch-Prozesse nicht mehr in Echtzeit ausgeführt werden. Andererseits wird dadurch aber der Prozess beschleunigt und die Auswertung über die GUI dauert nicht mehr mehrere Minuten lang bzw. wird überhaupt erst ausgeführt. Schlussendlich führt die Aufteilung des Programms in mehrere Skripte, die den Prozess in Zwischenschritten bearbeiten, zu einer erhöhten Effizienz bezüglich der Nutzung von Kapazitäten und vermeidet redundante Berechnungen. Auf diese Lösung kann allerdings nur zurückgegriffen werden, wenn die Wiederholungsrate begrenzt ist. Das bedeutet, dass mit einer Erhöhung der zeitlichen Frequenz (z.B. Änderung der Frequenz von jährlich zu wöchentlich) mithilfe dieses Vorgehens ein erhöhter Aufwand verbunden wäre, da in dem Fall eine wöchentliche händische Aktualisierung erforderlich wäre.

Generell kann festgestellt werden, dass für diesen Anwendungsfall (Klassifizierung von Waldtypen im Naturraum von RLP) von den getesteten Kombinationen die Kombination GTB und TC-*brightness*, -*wetness* und -*greenness* das beste Ergebnis lieferte. Der MLA RF ist für diesen Anwendungsfall ebenfalls geeignet, lieferte aber als Resultat der Validierung etwas geringere Genauigkeiten als GTB, während der MLA CART die schlechtesten Ergebnisse lieferte. Publikationen mit ähnlichen Fragestellungen kamen zum gleichen Ergebnis (vgl. PONGANAN et al. 2021: 36, SUJUD et al. 2021: 9). Die Klassifizierung anhand des TC ist gegenüber des NDVI zu bevorzugen, da dieser VI durchweg validere Ergebnisse lieferte. Das Klassifikationsergebnis der Kombination GTB und TC des Jahres 2019 wurde über die Berechnung des IoU-Verhältnisses mit den Ergebnissen nach NINK et al. (2019) verglichen. Dabei wurden nur die Klassen Laub- und Nadelwald betrachtet. Es konnte festgestellt werden, dass 87,5% der Flächen richtig klassifiziert wurden, wenn angenommen wird, dass es sich bei dem Referenzdatensatz von NINK et al. (2019) um reale Gegebenheiten handelt. Die Klasse Laubwald wurde mit ca. 91% etwas besser abgebildet als die Klasse Nadelwald mit 82,8%.

Im validesten Ergebnis laut OA wurden die Klassen Laub- und Nadelwald sowohl aus Benutzer- als auch aus Produzentensicht sehr gut bis gut abgebildet, während die Klassifizierung der Nicht-Wald-Klasse als unzureichend einzustufen ist. Grund dafür ist zum einen die fehlende Attribuierung in den Trainingsdaten und zum anderen die schwierige Unterscheidung der Klassen Laub- und Nicht-Wald, da sich diese beiden Klassen durch bestehendes Totholz

und Strauchbedeckung bzw. neu gebildete Pioniervegetation hinsichtlich der (saisonal) reflektierten Signale stark ähneln. Für eine bessere Unterscheidung der beiden Klassen sind geeignetere Trainingsdatensätze und tiefergehende Klassifizierungen nötig. Beispielsweise könnte die Klassifizierung in zwei Schritte aufgeteilt werden: Zunächst wird das gesamte Untersuchungsgebiet in die Klassen Nadel- und Laubwald aufgeteilt, anschließend kann innerhalb der Laubwald-Klasse eine weitere Klassifizierung in Laub- und Nicht-Wald erfolgen. So werden die Unterschiede hinsichtlich der reflektierten Signale deutlicher und das Ergebnis genauer. Eine andere Möglichkeit wäre, Flächen, die sich innerhalb eines Jahres z.B. von Nadel- zu Laubwald geändert haben, herauszufiltern und als Störung bzw. Dynamik zu kennzeichnen.

Die fehlerhaften Klassifizierungen, nicht nur im Nicht-Wald-Bereich, sondern auch in Mischwäldern, Lärchenwäldern und Gebieten mit hoher Wolkenbedeckung, verfälschen die Ergebnisse, die in der Zeitreihe abgebildet werden, und lassen für den Endnutzer falsche Schlüsse zu. So werden für einzelne Jahre Bereiche innerhalb der S2-Szenen, die eine zu hohe Wolkenbedeckung auswiesen, ausgeschnitten und als Rückgang der Flächengrößen bestimmter Klassen abgebildet. Dieses Problem könnte dadurch umgangen werden, dass diese ausgeschnittenen Gebiete, die auch nicht mehr Teil des Klassifizierungsergebnisses im Rasterformat sind, als Null-Werte in der Zeitreihe abgebildet werden. Dafür ist lediglich ein Vergleich der addierten Flächengrößen aller Klassen im Klassifizierungsergebnis mit den Flächengrößen der ATKIS-Wälder nötig.

Prinzipiell ist das erstellte Programm beliebig erweiterbar, z.B. durch die Einbindung von weiteren Datengrundlagen zur Detektion von Schäden durch Borkenkäferbefall oder zur Auswertung von Baumartenverteilungen. Von der Erweiterung des Untersuchungsraums auf benachbarte Bundesländer oder die Großregion ist abzuraten, da GEE bei der Klassifikation von ganz RLP bereits an seine Grenzen stößt. Außerdem könnten die Berechnungen nicht einfach in den erstellten Code integriert werden, weil dafür individuelle Klassifikationen nötig wären, für die jeweils eigene Datengrundlagen aufbereitet werden müssten, denn die in dieser Arbeit verwendeten Trainings- und Testdaten sowie weitere Datengrundlagen bildet ausschließlich das Bundesland RLP ab und für die Großregion existieren bisher keine grenzüberschreitenden Forsteinrichtungsdaten.

Literatur

BI WALDZUKUNFT HILLSCHEID: Unser Wald in Not. In: waldzukunft.info. Abruf: 24.5.2022.

BRAUCHLER, M. und J. STOFFELS (2020): Leveraging OSM and GEOBIA to Create and Update Forest Type Maps. In: ISPRS International Journal of Geo-Information 9 (9).

BRAUCHLER, M., J. STOFFELS und S. NINK (2022): Extension of an Open GEOBIA Framework for Spatially Explicit Forest Stratification with Sentinel-2. In: Remote Sensing 14 (3).

BREIMAN, L., J. H. FRIEDMAN, R. OLSHEN und C. J. STONE (1984): Classification and Regression Trees.

BREIMAN, L. (1996): Bagging predictors. In: Machine learning 24 (2): 123–140.

BREIMAN, L. (1997): Arcing the edge. Technical Report 486. Statistics Department, University of California, Berkeley.

BREIMAN, L. (2001): Random forests. In: Machine learning 45 (1): 5–32.

BUNDESAMT FÜR KARTOGRAPHIE UND GEODÄSIE (2022): Verwaltungsgebiete 1:250 000 (Ebenen). In: gdz.bkg.bund.de/index.php/default/digitale-geodaten/verwaltungsgebiete/verwaltungsgebiete-1-250-000-ebenen-stand-01-01-vg250-ebenen-01-01.html. Abruf: 1.7.2022.

BUNDESMINISTERIUM FÜR ERNÄHRUNG UND LANDWIRTSCHAFT (2022): Dritte Bundeswaldinventur 2012. Vermessung des Waldes. In: www.bundeswaldinventur.de/dritte-bundeswaldinventur-2012/vermessung-des-waldes. Abruf: 18.5.2022.

BÜRGER-ARNDT, R. (2013): Waldfunktionen und Ökosystemleistungen im wissenschaftlichen Diskurs. In: Ring, I. (Hrsg.): Der Nutzen von Ökonomie und Ökosystemleistungen für die Naturschutzpraxis. Workshop III: Wälder. BfN-Skripten, Heft 334: 24–30.

CAMPBELL, J. B. UND WYNNE, R. H. (2011): Introduction to remote sensing. Guilford, New York.

CAMPS-VALLS, G.: Machine learning in remote sensing data processing. In: IEEE (Hrsg.): IEEE International Workshop 2009: 1–6.

CRIST, E. P. und R. C. CICONE (1984): Application of the tasseled cap concept to simulated thematic mapper data. In: Photogrammetric engineering and Remote sensing 50 (3): 343–352.

DEUTSCHES ZENTRUM FÜR LUFT- UND RAUMFAHRT E.V. (o. J.): Copernicus: Das Europäische Erdbeobachtungsprogramm. In: www.dlr.de/eoc/desktopdefault.aspx/tabid-5367/9013_read-16792. Abruf: 18.5.2022.

ESRI (o. J.): How the Compute Accuracy For Object Detection tool works. In: pro.arcgis.com/en/pro-app/latest/tool-reference/image-analyst/how-compute-accuracy-for-object-detection-works.htm. Abruf: 24.5.2022.

EUROPEAN SPACE AGENCY (o. J.): Level-2A. In: sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/product-types/level-2a. Abruf: 1.4.2022.

EUROPEAN SPACE AGENCY (o. J.): Sentinel-2. In: sentinel.esa.int/web/sentinel/missions/sentinel-2. Abruf: 25.3.2022.

FREUND, Y. UND R.E SCHAPIRE (1997): A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In: Journal of Computer and System Sciences 55 (1): 119–139.

GOOGLE DEVELOPERS (o. J.): Sentinel-2 MSI: MultiSpectral Instrument, Level-2A. In: developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S2_SR#bands. Abruf: 1.4.2022.

GOOGLE DEVELOPERS (2021): API Reference. In: https://developers.google.com/earth-engine/apidocs. Abruf: 23.5.2022.

GOOGLE DEVELOPERS (2022): Welcome to Google Earth Engine. In: developers.google.cn/earth-engine. Abruf: 8.4.2022.

GORELICK, N., M. HANCHER, M. DIXON, S. ILYUSHCHENKO, D. THAU und R. MOORE (2017): Google Earth Engine: Planetary-scale geospatial analysis for everyone. In: Remote Sensing of Environment 202: 18–27.

HO, T. K. (1995): Random decision forests. In: IEEE (Hrsg.): Proceedings of 3rd international conference on document analysis and recognition 1: 278–282.

HUANG, H., W. CHEN, Y. ZHANG, L. QIAO und Y. DU (2021): Analysis of ecological quality in Lhasa Metropolitan Area during 1990–2017 based on remote sensing and Google Earth Engine platform. In: Journal of Geographical Sciences 31 (2): 265–280.

JACCARD, P. (1902): Lois de distribution florale dans la zone alpine. In: Bull Soc Vaudoise Sci Nat 38: 69–130.

JIANG, L., W. WANG, X. YANG, N. XIE und Y. CHENG (2010): Classification methods of remote sensing image based on decision tree technologies. In: IFIP (Hrsg.): International Conference on Computer and Computing Technologies in Agriculture: 353–358.

KAMAL, M., I. JAMALUDDIN, A. PARELA und N. M. FARDA (2019): Comparison of Google Earth Engine (GEE)-based machine learning classifiers for mangrove mapping. In: ACRS (Hrsg.): Proceedings of the 40th Asian Conference Remote Sensing: 1–8.

LANDESAMT FÜR VERMESSUNG UND GEOBASISINFORMATION (2017): ATKIS - Amtliches Topografisch-Kartografisches Informationssystem. In: lvermgeo.rlp.de/de/ueber-uns/aufgaben/geotopografie/atkisr. Abruf: 18.5.2022.

LANDESAMT FÜR VERMESSUNG UND GEOBASISINFORMATION (2020): Luftbilder Rheinland-Pfalz DOP. In: daten.rlp.de/dataset/2b009ae4-aa3e-ff21-870b-49846d9561b2. Abruf: 18.5.2022.

LANDESFORSTEN RHEINLAND-PFALZ (2012): Bundeswaldinventur. In: www.wald.rlp.de/de/wald/beeindruckende-zahlen/bundeswaldinventur. Abruf: 18.5.2022.

LANGE, N. DE (2020): Geoinformatik in Theorie und Praxis: Grundlagen von Geoinformationssystemen, Fernerkundung und digitaler Bildverarbeitung. Springer-Verlag.

LILLESAND, T., R. W. KIEFER und J. CHIPMAN (2015): Remote sensing and image interpretation. John Wiley & Sons.

NATIONALPARKAMT HUNSRÜCK-HOCHWALD (o. J.): Gebiet & Zonen des Nationalparks. In: www.nationalpark-hunsrueck-hochwald.de/der-nationalpark-hunsrueck-hochwald/schutzgebiete-in-deutschland/gebiet-zonen-des-nationalparks.html. Abruf: 24.5.2022.

NINK, S., J. HILL, J. STOFFELS, H. BUDDENBAUM, D. FRANTZ und J. LANGSHAUSEN (2019): Using landsat and Sentinel-2 data for the generation of continuously updated forest type information layers in a cross-border region. In: Remote Sensing 11 (20): 2337.

NR-KURIER (2019): Unser Wald im Klimawandel (12.09.2019). In: https://www.nr-kurier.de/artikel/82556-unser-wald-im-klimawandel. Abruf: 8.7.2022.

PAL, A.: Gradient Boosting Trees for Classification: A Beginner's Guide. In: medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea. Abruf: 25.6.2022.

PONGANAN, N., T. HORANONT, K. ARTLERT und P. NUALLAONG (2021): Land Cover Classification using Google Earth Engine's Object-oriented and Machine Learning Classifier. In: IEEE (Hrsg.): 2nd International Conference on Big Data Analytics and Practices (IBDAP): 33–37.

QGIS (2022): QGIS Benutzerhandbuch 3.16. In: docs.qgis.org/3.16/de/docs/user_manual. Abruf: 8.4.2022.

RESSOURCES NATURELLES DÉVELOPPEMENT ASBL (2018): RegioWood II – Ein europäisches INTERREG VA-Projekt für die Wälder der Großregion. In: www.regiowood2.info/de. Abruf: 18.5.2022.

SCHÄFER, S. und P. VANVOLXEM (Hrsg.) (2016): Landeswaldgesetz (LWaldG) Rheinland-Pfalz - Kommentar.

STEHMAN, S. V. (1997): Selecting and interpreting measures of thematic classification accuracy. In: Remote Sensing of Environment 62 (1): 77–89.

STOFFELS, J., J. HILL, T. SACHTLEBER, S. MADER, H. BUDDENBAUM, O. STERN, J. LANGSHAUSEN, J. DIETZ und G. ONTRUP (2015): Satellite-Based Derivation of High-Resolution Forest Information Layers for Operational Forest Management. In: Forests 6 (12): 1982–2013.

SUJUD, L., H. JAAFAR, M. HASSAN und R. ZURAYK (2021): Cannabis detection from optical and RADAR data fusion: A comparative analysis of the SMILE machine learning algorithms in Google Earth Engine. In: Remote Sensing Applications: Society and Environment 24: 1–14.

XULU, S., N. MBATHA, K. PEERBHAY und M. GEBRESLASIE (2020): Detecting harvest events in plantation forest using Sentinel-1 and-2 data via Google Earth Engine. In: Forests 11 (12): 1283.

Anhang

Konfusionsmatrizen

TC/RF/2018						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1561	281	14	1856	0,841
	Nadelwald	118	917	19	1054	0,870
	kein Wald	18	3	1	22	0,045
	Summe	1697	1201	34	2932	
	PA	0,920	0,764	0,029		
	OA	0,845				
	Kappa	0,682				

TC/RF/2019						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1578	251	15	1844	0,856
	Nadelwald	108	941	18	1067	0,882
	kein Wald	11	10	1	22	0,045
	Summe	1697	1202	34	2933	
	PA	0,930	0,783	0,029		
	OA	0,859				
	Kappa	0,711				

TC/RF/2020						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1517	273	24	1814	0,836
	Nadelwald	125	911	10	1046	0,871
	kein Wald	7	8	0	15	0,000
	Summe	1649	1192	34	2875	
	PA	0,920	0,764	0,000		
	OA	0,845				
	Kappa	0,681				

TC/RF/2021						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1438	294	27	1759	0,818
	Nadelwald	146	811	5	962	0,843
	kein Wald	12	10	1	23	0,043
	Summe	1596	1115	33	2744	
	PA	0,901	0,727	0,030		
	OA	0,820				
	Kappa	0,628				

TC/RF/2022						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1551	289	29	1869	0,830
	Nadelwald	141	903	4	1048	0,862
	kein Wald	9	11	1	21	0,048
	Summe	1701	1203	34	2938	
	PA	0,912	0,751	0,029		
	OA	0,836				
	Kappa	0,661				

TC/CART/2018						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1426	290	15	1731	0,824
	Nadelwald	205	894	18	1117	0,800
	kein Wald	66	17	1	84	0,012
	Summe	1697	1201	34	2932	
	PA	0,840	0,744	0,029		
	OA	0,792				
	Kappa	0,585				
TC/CART/2019						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1474	254	18	1746	0,844
	Nadelwald	169	927	15	1111	0,834
	kein Wald	54	21	1	76	0,013
	Summe	1697	1202	34	2933	
	PA	0,869	0,771	0,029		
	OA	0,819				
	Kappa	0,638				
TC/CART/2020						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1429	277	25	1731	0,826
	Nadelwald	171	888	8	1067	0,832
	kein Wald	49	27	1	77	0,013
	Summe	1649	1192	34	2875	
	PA	0,867	0,745	0,029		
	OA	0,806				
	Kappa	0,613				
TC/CART/2021						
		LWI-Daten				

		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1314	299	24	1637	0,803
	Nadelwald	229	792	6	1027	0,771
	kein Wald	53	24	3	80	0,038
	Summe	1596	1115	33	2744	
	PA	0,823	0,710	0,091		
	OA	0,769				
	Kappa	0,538				

TC/CART/2022

		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1432	288	22	1742	0,822
	Nadelwald	212	882	8	1102	0,800
	kein Wald	57	33	4	94	0,043
	Summe	1701	1203	34	2938	
	PA	0,842	0,733	0,118		
	OA	0,789				
	Kappa	0,580				

TC/GTB/2018

		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1556	262	16	1834	0,848
	Nadelwald	124	933	17	1074	0,869
	kein Wald	17	6	1	24	0,042
	Summe	1697	1201	34	2932	
	PA	0,917	0,777	0,029		
	OA	0,849				
	Kappa	0,691				

TC/GTB/2019

		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1589	257	17	1863	0,853
	Nadelwald	97	933	17	1047	0,891
	kein Wald	11	12	0	23	0,000
	Summe	1697	1202	34	2933	
	PA	0,936	0,776	0,000		
	OA	0,860				
	Kappa	0,712				

TC/GTB/2020

		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte	Laubwald	1535	257	25	1817	0,845

	Nadelwald	110	929	8	1047	0,887
	kein Wald	4	6	1	11	0,091
	Summe	1649	1192	34	2875	
	PA	0,931	0,779	0,029		
	OA	0,857				
	Kappa	0,707				

TC/GTB/2021						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1457	293	26	1776	0,820
	Nadelwald	132	817	6	955	0,855
	kein Wald	7	5	1	13	0,077
	Summe	1596	1115	33	2744	
	PA	0,913	0,733	0,030		
	OA	0,829				
	Kappa	0,645				

TC/GTB/2022						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1569	302	26	1897	0,827
	Nadelwald	120	888	7	1015	0,875
	kein Wald	12	13	1	26	0,038
	Summe	1701	1203	34	2938	
	PA	0,922	0,738	0,029		
	OA	0,837				
	Kappa	0,663				

NDVI/RF/2018						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1414	511	26	1951	0,725
	Nadelwald	240	656	8	904	0,726
	kein Wald	43	34	0	77	0,000
	Summe	1697	1201	34	2932	
	PA	0,833	0,546	0,000		
	OA	0,706				
	Kappa	0,398				

NDVI/RF/2019						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1492	328	22	1842	0,810
	Nadelwald	152	842	11	1005	0,838
	kein Wald	53	32	1	86	0,012
	Summe	1697	1202	34	2933	

	PA	0,879	0,700	0,029		
	OA	0,796				
	Kappa	0,398				

NDVI/RF/2020						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1448	338	26	1812	0,799
	Nadelwald	161	808	6	975	0,829
	kein Wald	40	46	2	88	0,023
	Summe	1649	1192	34	2875	
	PA	0,878	0,678	0,059		
	OA	0,785				
	Kappa	0,398				

NDVI/RF/2021						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1342	340	26	1708	0,786
	Nadelwald	203	734	5	942	0,779
	kein Wald	51	41	2	94	0,021
	Summe	1596	1115	33	2744	
	PA	0,841	0,658	0,061		
	OA	0,757				
	Kappa	0,398				

NDVI/RF/2022						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1414	439	27	1880	0,752
	Nadelwald	234	726	7	967	0,751
	kein Wald	53	38	0	91	0,000
	Summe	1701	1203	34	2938	
	PA	0,831	0,603	0,000		
	OA	0,728				
	Kappa	0,398				

NDVI/CART/2018						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1359	496	26	1881	0,722
	Nadelwald	267	661	7	935	0,707
	kein Wald	71	44	1	116	0,009
	Summe	1697	1201	34	2932	
	PA	0,801	0,550	0,029		
	OA	0,689				
	Kappa	0,376				

NDVI/CART/2019						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1449	310	22	1781	0,814
	Nadelwald	180	857	11	1048	0,818
	kein Wald	68	35	1	104	0,010
	Summe	1697	1202	34	2933	
	PA	0,854	0,713	0,029		
	OA	0,787				
	Kappa	0,575				
NDVI/CART/2020						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1422	342	25	1789	0,795
	Nadelwald	162	795	7	964	0,825
	kein Wald	65	55	2	122	0,016
	Summe	1649	1192	34	2875	
	PA	0,862	0,667	0,059		
	OA	0,772				
	Kappa	0,547				
NDVI/CART/2021						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1312	332	27	1671	0,785
	Nadelwald	229	728	5	962	0,757
	kein Wald	55	55	1	111	0,009
	Summe	1596	1115	33	2744	
	PA	0,822	0,653	0,030		
	OA	0,744				
	Kappa	0,491				
NDVI/CART/2022						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifi-zierte Daten	Laubwald	1369	446	29	1844	0,742
	Nadelwald	274	710	5	989	0,718
	kein Wald	58	47	0	105	0,000
	Summe	1701	1203	34	2938	
	PA	0,805	0,590	0,000		
	OA	0,708				
	Kappa	0,413				

NDVI/GTB/2018						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1507	482	26	2015	0,748
	Nadelwald	182	711	8	901	0,789
	kein Wald	8	8	0	16	0,000
	Summe	1697	1201	34	2932	
	PA	0,888	0,592	0,000		
	OA	0,756				
	Kappa	0,489				

NDVI/GTB/2019						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1598	320	22	1940	0,824
	Nadelwald	91	875	12	978	0,895
	kein Wald	8	7	0	15	0,000
	Summe	1697	1202	34	2933	
	PA	0,942	0,728	0,000		
	OA	0,843				
	Kappa	0,674				

NDVI/GTB/2020						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1534	322	25	1881	0,816
	Nadelwald	107	856	8	971	0,882
	kein Wald	8	14	1	23	0,043
	Summe	1649	1192	34	2875	
	PA	0,930	0,718	0,029		
	OA	0,832				
	Kappa	0,653				

NDVI/GTB/2021						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1455	346	29	1830	0,795
	Nadelwald	133	758	4	895	0,847
	kein Wald	8	11	0	19	0,000
	Summe	1596	1115	33	2744	
	PA	0,912	0,680	0,000		
	OA	0,806				
	Kappa	0,596				

NDVI/GTB/2022						
		LWI-Daten				
		Laubwald	Nadelwald	kein Wald	Summe	**UA**
klassifizierte Daten	Laubwald	1527	446	27	2000	0,764

	Nadelwald	168	751	7	926	0,811
	kein Wald	6	6	0	12	0,000
	Summe	1701	1203	34	2938	
	PA	0,898	0,624	0,000		
	OA	0,775				
	Kappa	0,529				

Skript 1/3 (Klassifizierung)

```
// ------------------------------------------------------------------------------------------
//
// Masterthesis Anna Göbel
// anna.goebel@posteo.de
//
// Skript 1/3
// Klassifizierungsskript
//
// Dieses Skript klassifiziert die S2-Daten je nach Eingabe des VI und MLA
// und bietet die Möglichkeit, das Ergebnis als Asset oder in Google Drive zu speichern
// Speicherung als Asset: Einbindung der Ergebnisse im Validierungsskript
// Speicherung in Drive: Download der Rasterdaten
//
// ------------------------------------------------------------------------------------------

// ------------------------------------------------------------------------------------------
// Import FeatureCollections
// ------------------------------------------------------------------------------------------

// hier Links austauschen ------------------->
// Grenze RLP
var rlp = ee.FeatureCollection("projects/ee-annagoebel/assets/RLP");
// Trainingsdatensatz
var laubnadelwald = ee.FeatureCollection("projects/ee-annagoebel/assets/Reinbestand220509_1_5K");
// ATKIS-Wälder zum Clippen
var wald_gesamt = ee.FeatureCollection("projects/ee-annagoebel/assets/ATKIS_wald");


// ------------------------------------------------------------------------------------------
// GUI
// ------------------------------------------------------------------------------------------

// clear root, füge Map hinzu
ui.root.clear();
var karte = ui.Map();
ui.root.add(karte);

// Zoom auf RLP
karte.centerObject(rlp, 8);

// definiere Variablen für NDVI und TC
var ndvi;
var tc;
var VI = {
  'NDVI': ndvi,
  'TC': tc
};

// selector für VI
var selectVI = ui.Select({
  style: {width: '480px'},
  items: Object.keys(VI),
  placeholder: 'Wählen Sie einen VI aus...',
});

// definiere Variablen für Machine Learning Algorithmen (MLA)
var rf;
var cart;
var gtb;
var MLA = {
  'RF': rf,
  'CART': cart,
  'GTB': gtb
};

// selector für MLA
var selectMLA = ui.Select({
  style: {width: '480px'},
  items: Object.keys(MLA),
  placeholder: 'Wählen Sie einen MLA aus...',
});

// Button starte Auswertung
var berechnen = ui.Button({
  style: {width: '480px'},
  label: "starte Auswertung"
});
```

```
// Hauptpanel
var hauptpanel = ui.Panel({
  widgets:[
    ui.Label({value: 'Auswertung', style: {fontWeight: 'bold', fontSize: '20px'}}),
    selectVI,
    selectMLA,
    berechnen,
    ui.Label({value:"Ausgabe der OA nicht möglich (user memory exceeded) -> Ergebnis über Tasks als Asset
speichern und ausführliche Validierung über Validierungsskript ausführen"}),
    ui.Label({value:"Im Validierungsskript werden die Assets dann wieder eingeladen. Grund: Auslagerung der
Rechenleistung in Batch-Prozess"}),
    ui.Label({value:"Mithilfe dieses Tools kann das Ergebnis als Asset oder in Google Drive gespeichert
werden. Die Darstellung in der Map ist zwar implementiert, aber ob sie funktioniert, ist tagesformabhän-
gig."})
  ],
  style: {width: '500px',  border: '1px solid black'}
});

// Definition der Zeitpunkte zum Filtern der S2 Szene
var yearFile = [
    {year:"2018", startDate:"2018-02-15", endDate:"2018-04-30"},
    {year:"2019", startDate:"2019-02-15", endDate:"2019-04-30"},
    {year:"2020", startDate:"2020-02-15", endDate:"2020-03-31"},
    {year:"2021", startDate:"2021-03-15", endDate:"2021-03-31"},
    {year:"2022", startDate:"2022-02-15", endDate:"2022-03-10"},
];

// definiere Variablen
var img;
var img2;
var nameVI;
var nameMLA;
var bands;

// Auswahl des Features
berechnen.onClick(function() {
  // Klassifikation über alle Jahre hinweg
  for (var i = 0; i < yearFile.length; i++){
    // Abfrage der selctors
    if (selectVI.getValue() == "NDVI"){
      bands = ['NDVI'];
      nameVI = "NDVI";
      img = calculateS2(yearFile[i].startDate, yearFile[i].endDate, yearFile[i].year);
      img2 = img.map(calculateNDVI);
    } else {
      bands = ['brightness','greenness', 'wetness'];
      nameVI = "Tasseled Cap";
      img = calculateS2(yearFile[i].startDate, yearFile[i].endDate, yearFile[i].year);
      img2 = img.map(calculateTC);
    }

    if (selectMLA.getValue() == "RF"){
      nameMLA = "Random Forest";
      calculateRF(img2, yearFile[i].year);

    } else if (selectMLA.getValue() == "CART"){
      nameMLA = "Classification and Regression Trees";
      calculateCART(img2, yearFile[i].year);
    } else {
      nameMLA = "Gradient Tree Boost";
      calculateGTB(img2, yearFile[i].year);
    }


  }

});
ui.root.add(hauptpanel);


// ---------------------------------------------------------------------------------------------
// Sentinel-2 ImageCollections
// ---------------------------------------------------------------------------------------------

function calculateS2 (startDate, endDate, year) {

  // Definiere Sentinel ImageCollection
  var bands = ['B2', 'B3', 'B4','B5', 'B6', 'B7', 'B8', 'B8A', 'B11', 'B12', 'SCL'];

  var s2 = ee.ImageCollection('COPERNICUS/S2_SR')
            .filterDate(startDate, endDate)
            .filterBounds(rlp)
            .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',40))
```

```
            .select(bands)
            .map(function(image){return image.clipToCollection(rlp)});

  // ------------------------------------------------------------------------------------------
  // Wolkenmaske
  // ------------------------------------------------------------------------------------------

  function maskCloudAndShadowsS2(image) {
    var scl = image.select('SCL');
      var darkArea = scl.eq(2);
      var shadow = scl.eq(3); // 3 = Wolkenschatten
      var cloudsLowProbability =  scl.eq(7); //
      var cloudsMediumProbability =  scl.eq(8); //
      var cloudsHighProbability =  scl.eq(9); //
      var cirrus = scl.eq(10); // 10 = Cirrus
      var SnowIce = scl.eq(11);
    var mask = (darkArea.neq(1)).and(shadow.neq(1)).and(cloudsLowProbability.neq(1))
      .and(cloudsMediumProbability.neq(1)).and(cloudsHighProbability.neq(1)).and(cirrus.neq(1))
      .and(SnowIce.neq(1));
    return image.updateMask(mask);
  }

  var s2masked = s2.map(maskCloudAndShadowsS2);
  return s2masked;
}


// Berechnung tasseled cap
var calculateTC = function (image){
  var b = image.select("B2", "B3", "B4", "B8A", "B11", "B12");
  var brightness_coefficents= ee.Image([0.3037, 0.2793, 0.4743, 0.5585, 0.5082, 0.1863]);
  var greenness_coefficents= ee.Image([-0.2848, -0.2435, -0.5436, 0.7243, 0.0840, -0.1800]);
  var wetness_coefficents= ee.Image([0.1509, 0.1973, 0.3279, 0.3406, -0.7112, -0.4572]);

  var brightness = image.expression(
    '(B * BRIGHTNESS)',
    {'B':b, 'BRIGHTNESS': brightness_coefficents}
  );
  var greenness = image.expression(
    '(B * GREENNESS)',
    {'B':b, 'GREENNESS': greenness_coefficents}
  );
  var wetness = image.expression(
    '(B * WETNESS)',
    {'B':b, 'WETNESS': wetness_coefficents}
  );

  brightness = brightness.reduce(ee.call("Reducer.sum"));
  greenness = greenness.reduce(ee.call("Reducer.sum"));
  wetness = wetness.reduce(ee.call("Reducer.sum"));

  var tasseled_cap = ee.Image(brightness).addBands(greenness).addBands(wetness).rename('brightness','green-
ness','wetness');

  return tasseled_cap;
};


// Berechnung NDVI
var calculateNDVI = function(image) {
  var ndvi = image.normalizedDifference(['B8', 'B4']).rename('NDVI');
  return image.addBands(ndvi);
};


var calculateRF = function(image, year) {

  var training = image.select(bands).mean().sampleRegions({
    collection: laubnadelwald,
    properties: ['Typ'],
    scale: 10
  });


  // ------------------------------------------------------------------------------------------
  // classification algorithms (TC)
  // ------------------------------------------------------------------------------------------

  // Make a Random Forest classifier and train it.
  var classifierRF = ee.Classifier.smileRandomForest(5).train({ // <------- Anzahl Entscheidungsbäume
    features: training,
    classProperty: 'Typ',
    inputProperties: bands
```

```
  });
  // Classify the input imagery.
  var classifiedRF = image.select(bands).mean().classify(classifierRF);

  // Define a palette for the  classification.
  var palette = [
    'green', // Laubwald (0)
    'blue', // Nadelwald (1)
    'grey' // kein Wald (2)
  ];
  karte.addLayer(classifiedRF.clipToCollection(wald_gesamt), {palette: palette, min: 0, max: 2}, 'Classifi-
cation '+ selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year, true);

  Export.image.toAsset({
    image: classifiedRF.clip(wald_gesamt),
    description: ('asset_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });

  // aendere Pixelwerte, damit 0 als nullValue verwendet werden kann
  var new_image = classifiedRF.clip(wald_gesamt).remap([0,1,2],[1,2,3]);

  Export.image.toDrive({
    image: new_image,
    description: ('drive_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    //pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });

};

var calculateCART = function(image, year) {


  var training = image.select(bands).mean().sampleRegions({
    collection: laubnadelwald,
    properties: ['Typ'],
    scale: 10
  });

  // Make a CART classifier and train it.
  var classifierCart = ee.Classifier.smileCart().train({
    features: training,
    classProperty: 'Typ',
    inputProperties: bands
  });
  // Classify the input imagery.
  var classifiedCart = image.select(bands).mean().classify(classifierCart);

  // Define a palette for the  classification.
  var palette = [
    'green', // Laubwald (0)
    'blue', // Nadelwald (1)
    'grey' // kein Wald (2)
  ];
  karte.addLayer(classifiedCart, {palette: palette, min: 0, max: 2}, 'Classification '+ selectVI.getValue()
+ '_' + selectMLA.getValue() + '_' + year, true);

  Export.image.toAsset({
    image: classifiedCart.clip(wald_gesamt),
    description: ('asset_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });

  // aendere Pixelwerte, damit 0 als nullValue verwendet werden kann
  var new_image = classifiedCart.clip(wald_gesamt).remap([0,1,2],[1,2,3]);

  Export.image.toDrive({
    image: new_image,
    description: ('drive_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    //pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });
```

```
};

var calculateGTB = function(image, year) {
  var training = image.select(bands).mean().sampleRegions({
    collection: laubnadelwald,
    properties: ['Typ'],
    scale: 10
  });

  // Make a GTB classifier and train it.
  var classifierGTB = ee.Classifier.smileGradientTreeBoost(5).train({ // <------- Anzahl Entscheidungsbäume
    features: training,
    classProperty: 'Typ',
    inputProperties: bands
  });

  // Classify the input imagery.
  var classifiedGTB = image.select(bands).mean().classify(classifierGTB);

  // Define a palette for the  classification.
  var palette = [
    'green', // Laubwald (0)
    'blue', // Nadelwald (1)
    'grey' // kein Wald (2)
  ];
  karte.addLayer(classifiedGTB, {palette: palette, min: 0, max: 2}, 'Classification '+ selectVI.getValue()
+ '_' + selectMLA.getValue() + '_' + year, true);

  Export.image.toAsset({
    image: classifiedGTB.clip(wald_gesamt),
    description: ('asset_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });


  // aendere Pixelwerte, damit 0 als nullValue verwendet werden kann
  var new_image = classifiedGTB.clip(wald_gesamt).remap([0,1,2],[1,2,3]);

  Export.image.toDrive({
    image: new_image,
    description: ('drive_class_' + selectVI.getValue() + '_' + selectMLA.getValue() + '_' + year),
    region: rlp,
    //pyramidingPolicy: {".default": "min"},
    scale: 10,
    maxPixels: 1000000000
  });
};
```

Skript 2/3 (Validierung)

```
// ---------------------------------------------------------------------------------------
//
// Masterthesis Anna Göbel
// anna.goebel@posteo.de
//
// Skript 2/3
// Validierungsskript
//
// Dieses Skript validiert die Ergebnisse des Klassifizierungsskripts je nach Eingabe des VI und MLA
// und bietet die Möglichkeit, die Konfusionsmatrizen in Google Drive zu speichern
//
//
//
// ---------------------------------------------------------------------------------------

// ---------------------------------------------------------------------------------------
// Import FeatureCollections
// ---------------------------------------------------------------------------------------

// hier Links austauschen ------------------->
// Grenze RLP
var rlp = ee.FeatureCollection("projects/ee-annagoebel/assets/RLP");
// Testdatensatz für Validierung
var LWI_valid = ee.FeatureCollection("projects/ee-annagoebel/assets/LWI_Reinbestand_3000");


// ---------------------------------------------------------------------------------------
// GUI
// ---------------------------------------------------------------------------------------

// clear root, füge Map hinzu
ui.root.clear();
var karte = ui.Map();
ui.root.add(karte);

// Zoom auf RLP
karte.centerObject(rlp, 8);

// definiere Variablen für NDVI und TC
var ndvi;
var tc;
var VI = {
  'NDVI': ndvi,
  'TC': tc
};

// selector für VI
var selectVI = ui.Select({
  style: {width: '480px'},
  items: Object.keys(VI),
  placeholder: 'Wählen Sie einen VI aus...',
});

// definiere Variablen für Machine Learning Algorithmen (MLA)
var rf;
var cart;
var gtb;
var MLA = {
  'RF': rf,
  'Cart': cart,
  'GTB': gtb
};

// selector für MLA
var selectMLA = ui.Select({
  style: {width: '480px'},
  items: Object.keys(MLA),
  placeholder: 'Wählen Sie einen MLA aus...',
});

// definiere Variablen für Ausgabewert im chart
var oa;
var ua;
var pa;
var kappa;
var values = {
  'OA': oa,
  'UA': ua,
```

```
  'PA': pa,
  'Kappa': kappa
};

// selector für Value
var selectValue = ui.Select({
  style: {width: '480px'},
  items: Object.keys(values),
  placeholder: 'Wählen Sie einen Parameter aus...',
});

// Button starte Auswertung
var berechnen = ui.Button({
  style: {width: '480px'},
  label: "starte Auswertung"
});

var hauptpanel = ui.Panel({
  widgets:[
    ui.Label({value: 'Auswertung', style: {fontWeight: 'bold', fontSize: '20px'}}),
    selectVI,
    selectMLA,
    selectValue,
    berechnen
  ],
  style: {width: '500px',  border: '1px solid black'}
});

// Definition der Zeitpunkte zum Filtern der S2 Szene
var yearFile = [
    {year:"2018", startDate:"2018-02-15", endDate:"2018-04-30"},
    {year:"2019", startDate:"2019-02-15", endDate:"2019-04-30"},
    {year:"2020", startDate:"2020-02-15", endDate:"2020-03-31"},
    {year:"2021", startDate:"2021-03-15", endDate:"2021-03-31"},
    {year:"2022", startDate:"2022-02-15", endDate:"2022-03-10"},
];

// Auswahl des Features
berechnen.onClick(function() {

  // Classification über alle Jahre hinweg, Füllen des dataTable für chart
  for (var i = 0; i < yearFile.length; i++){
    var auswahlText = "projects/ee-annagoebel/assets/asset_class_" + selectVI.getValue() + "_" + select-
MLA.getValue() + "_" + yearFile[i].year;
    var auswahl = ee.Image(auswahlText);
    // Define a palette for the  classification.
    var palette = [
      'green', // Laubwald (0)
      'blue', // Nadelwald (1)
      'grey' // kein Wald (2)
    ];


    // Display the classification result and the input image.
    karte.addLayer(auswahl, {min: 0, max: 2, palette: palette}, selectMLA.getValue() + ' ' + selectVI.get-
Value() + ' ' + yearFile[i].year, false);

    var result = validate(auswahl, yearFile[i].year, selectMLA.getValue(), selectVI.getValue());

    // print table
    dataTable[i+1] = [
      yearFile[i].year,
      result
    ];
  }

  var dataTableNew = [
    [dataTable[1][0], dataTable[2][0], dataTable[3][0], dataTable[4][0], dataTable[5][0]],
    [dataTable[1][1], dataTable[2][1], dataTable[3][1], dataTable[4][1], dataTable[5][1]],
  ];

  var x = ee.List(ee.List(dataTableNew).get(0));
  var y = ee.List(ee.List(dataTableNew).get(1));

  var chart = ui.Chart.array.values({array: y, axis: 0, xLabels: x}).setChartType('ColumnChart').setOp-
tions({
    title: selectVI.getValue() + " / " + selectMLA.getValue(),
    hAxis: {title: 'Jahre'},
    vAxis: {title: selectValue.getValue()},
    legend: {position: 'none'}
  });
  hauptpanel.add(chart);
```

```
});

ui.root.add(hauptpanel);


// definition dataTable
var dataTable = [
  [{label: 'Jahr', role: 'domain', type: 'string'},
  {label: 'OA', role: 'data', type: 'number'}]

];


//implementation array for rlp featureCollection properties
var fc = ee.FeatureCollection([]);
var yearsArray = [];
var OA = [];

var validate = function(image, year, vi, mla){

  // ------------------------------------------------------------------------------------------
  // Validierung
  // ------------------------------------------------------------------------------------------

  // übertrage LWI_Daten auf den classified-Layer
  var testing = image.sampleRegions({
    collection: LWI_valid,
    properties: ['Typ'],
    scale: 10
  });

  var confusionMatrix = testing.errorMatrix('Typ', 'classification');

  // gebe confusionMatrix aus
  var exportAccuracy = ee.Feature(null, {matrix: confusionMatrix.getInfo()});
  Export.table.toDrive({
    collection: ee.FeatureCollection(exportAccuracy),
    description: 'validation_' + vi + '_' + mla + '_' + year,
    fileFormat: 'CSV'
  });
  print(year);
  print(confusionMatrix);

  // Abfrage + Definition Ausgabeparameter
  if (selectValue.getValue() == 'OA'){
    return(confusionMatrix.accuracy());
  } else if (selectValue.getValue() == 'UA'){
    return(confusionMatrix.consumersAccuracy());
  } else if (selectValue.getValue() == 'PA'){
    return(confusionMatrix.producersAccuracy());
  } else if (selectValue.getValue() == 'Kappa'){
    return(confusionMatrix.kappa());
  }

};
```

Skript 3/3 (GUI)

```
// ---------------------------------------------------------------------------------------
//
// Masterthesis Anna Göbel
// anna.goebel@posteo.de
//
// Skript 3/3
// Hauptskript GUI
//
// Dieses Skript implementiert die GUI.
// Die Ergebnisse des Klassifizierungsskripts werden aus den Assets eingeladen.
// Änderung der Auswahl der Kombination in Zeile 251 (momentan GTB + TC)
//
//
// ---------------------------------------------------------------------------------------

// ---------------------------------------------------------------------------------------
// Import FeatureCollections
// ---------------------------------------------------------------------------------------

// hier Links austauschen ------------------->
// Grenze RLP
var rlp = ee.FeatureCollection("projects/ee-annagoebel/assets/RLP");
// Gemeindegrenzen
var gemeinden = ee.FeatureCollection("projects/ee-annagoebel/assets/Gemeinden");
// Forstamtsgrenzen
var fa = ee.FeatureCollection("projects/ee-annagoebel/assets/Forstaemter");
// ATKIS-Wälder zum Clippen
var wald_gesamt = ee.FeatureCollection("projects/ee-annagoebel/assets/ATKIS_wald");

// ---------------------------------------------------------------------------------------
// GUI
// ---------------------------------------------------------------------------------------

// clear root, add Map
ui.root.clear();
var karte = ui.Map();
ui.root.add(karte);


// Zoom auf RLP
karte.centerObject(rlp, 8);

// definiere drawingTools zum Zeichnen eines eigenen Polygons
var drawingTools = karte.drawingTools();


//Hinzufügen der FeatureCollections zur Karte
//Gemeinden
var empty_gem = ee.Image().byte();
var outline_gem = empty_gem.paint({
  featureCollection: gemeinden,
  color: 1,
  width: 1
});
karte.addLayer(outline_gem, {palette: '000000'}, "Gemeindegrenzen", false);

//Forstämter
var empty_fa = ee.Image().byte();
var outline_fa = empty_fa.paint({
  featureCollection: fa,
  color: 1,
  width: 1.5
});
karte.addLayer(outline_fa, {palette: '006600'}, "Forstamtsgrenzen", false);

//RLP
var empty_rlp = ee.Image().byte();
var outline_rlp = empty_rlp.paint({
  featureCollection: rlp,
  color: 1,
  width: 2
});
karte.addLayer(outline_rlp, {palette: 'FF0000'}, "Bundeslandgrenze (RLP)");

var polygon;

// definiere Ebenen für select-Button
var ebenen = {
```

```
  'eigenes Polygon zeichnen': polygon,
  'Gemeinde': gemeinden,
  'Forstamt (nicht empfohlen)': fa,
};

var ebene;
// inspector-Panel definieren
var selectText;
var inspector = ui.Panel([ui.Label(selectText)]);

// select-Button zur Auswahl der Ebene/"Kategorie"
var select = ui.Select({
  style: {width: '480px'},
  items: Object.keys(ebenen),
  placeholder: 'Wählen Sie eine Kategorie aus...',
  onChange: function(key) {

    //Layer ausschalten
    karte.layers().get(1).setShown(false);
    karte.layers().get(0).setShown(false);

    // bei Auswahl: ändere drawing Tools + Layerdarstellung + wähle Ebene aus
    if (select.getValue() == 'Gemeinde'){
      karte.layers().get(0).setShown(true);
      selectText = "Wählen Sie eine Gemeinde aus";
      waehlePunkt();
      ebene = gemeinden;
    }
    else if (select.getValue() == 'eigenes Polygon zeichnen'){
      drawingTools.clear();
      drawingTools.setDrawModes(['polygon']);
      drawingTools.addLayer([]);
      drawingTools.setShape('polygon');
      drawingTools.draw();
      ebene = polygon;
    }
    else if (select.getValue() == 'Forstamt (nicht empfohlen)'){ //nicht empfohlen weil zu groß
      karte.layers().get(1).setShown(true);
      selectText = "Wählen Sie ein Forstamt aus";
      waehlePunkt();
      ebene = fa;
    }

    // Gemeinden und Forstämter werden über Punkte angewählt
    function waehlePunkt(){
      drawingTools.clear();
      drawingTools.setDrawModes(['point']);
      drawingTools.addLayer([]);
      drawingTools.setShape('point');
      drawingTools.draw();
    }

    // zeige Bezeichnung Gemeinde/Forstamt/eigenes Polygon im oberen Panel an
    karte.remove(inspector);
    inspector = ui.Panel([ui.Label(selectText)]);
    karte.add(inspector);
  }
});

// starte Auswertung - Button
var berechnen = ui.Button({
  style: {width: '480px'},
  label: "starte Auswertung"
});

// Hauptpanel zur Konfiguration
var hauptpanel = ui.Panel({
  widgets:[
    ui.Label({value: 'Konfiguration', style: {fontWeight: 'bold', fontSize: '20px'}}),
    ui.Label('Dieses Tool dient dem zeitreihenbasierten Waldmonitoring in Rheinland-Pfalz. Es wird für je-
des Jahr (2018-2022) die Fläche von Laub- Nadel- und Nicht-Wald berechnet und als Zeitreihe ausgegeben. Die
Auswertung erfolgt auf Basis des von Ihnen ausgewählten Polygons. Achtung: je größer die ausgewählte Flä-
che, desto länger dauert die Auswertung.'),
    ui.Label('1. Kategorie auswählen '),
    select,
    ui.Label('2. Fläche auswählen bzw. eigene Fläche im Kartenausschnitt zeichnen (zoomen mit + und - links
im Kartenausschnitt oder Suche nach Ort in der Suchleiste oben)'),
    ui.Label('   <---'),
    ui.Label('3. Auswertung starten:'),
    berechnen,
    ui.Label('4. Geduld - ich bin gleich soweit :-)'),
    ui.Label(''),
    ui.Label(''),
```

```
    ui.Label(''),
    ui.Label(''),
    ui.Label(''),
    ui.Label(''),
    ui.Label(''),
    ui.Label({value: "last change: 10.06.2022", style: { fontSize: '10px'}}),
    ui.Label({value: "master thesis: GUI for time series based monitoring of (private) forests in Rhine-
land-Palatinate", style: { fontSize: '10px'}}),
    ui.Label({value: "Paris Lodron Universität Salzburg / Universität Trier", style: { fontSize: '10px'}}),
    ui.Label({value: "author: Anna Göbel (anna.goebel@posteo.de)", style: { fontSize: '10px'}}),
  ],

  style: {width: '500px',  border: '1px solid black'}
});
ui.root.add(hauptpanel);

// Auswertungspanel
var auswertungspanel = ui.Panel({
  widgets:[ ],
  style: {width: '500px',  border: '1px solid black'}
});

// Start der Auswertung
berechnen.onClick(function() {
  karte.setOptions("HYBRID");
  // Definiere FeatureCollection mit den Koordinaten des Punkts
  var auswahl = drawingTools.layers().get(0).getEeObject();
  var spatialJoined;

  if (select.getValue() == "eigenes Polygon zeichnen"){
    drawingTools.setShown(true);
    spatialJoined = auswahl;
  }
  else {
    // Spatial Join mit Punkt als Input und Ebene (Gemeinden/Forstamt)
    var distFilter = ee.Filter.withinDistance({
      distance: 0,
      leftField: '.geo',
      rightField: '.geo',
      maxError: 10
    });
    var distSaveAll = ee.Join.saveAll({
      matchesKey: 'points',
      measureKey: 'distance'
    });

    // spatial join des geklickten Punktes mit dem Objekt
    spatialJoined = distSaveAll.apply(ebene, auswahl, distFilter);
  }

  drawingTools.clear();
  // auf ausgewähltes Objekt zoomen
  karte.centerObject(spatialJoined);
  // Panel entfernen nachdem Gemeinde ausgewählt wurde
  karte.remove(inspector);

  // Bezeichnung Gemeinde/Forstamt im Panel einfügen
  var gemText;
  if (select.getValue() == 'Forstamt (nicht empfohlen)'){
    gemText = spatialJoined.first().getString('FA_REF_NAM').getInfo();
    // Text aus properties in Panel einfügen
    inspector = ui.Panel([ui.Label(gemText)]);
    karte.add(inspector);
  } else if (select.getValue() == 'Gemeinde'){
    gemText = spatialJoined.first().getString('bez').getInfo() + " " + spa-
tialJoined.first().getString('gen').getInfo();
    // Text aus properties in Panel einfügen
    inspector = ui.Panel([ui.Label(gemText)]);
    karte.add(inspector);
  } else if (select.getValue() == 'eigenes Polygon zeichnen'){
    gemText = "eigenes Polygon";
  }

  // Definition der Zeitpunkte zum Filtern der S2 Szene
  var yearFile = [
    {year:"2018", startDate:"2018-02-15", endDate:"2018-04-30"},
    {year:"2019", startDate:"2019-02-15", endDate:"2019-04-30"},
    {year:"2020", startDate:"2020-02-15", endDate:"2020-03-31"},
    {year:"2021", startDate:"2021-03-15", endDate:"2021-03-31"},
    {year:"2022", startDate:"2022-02-15", endDate:"2022-03-10"},
  ];

  // Implementierung des dataTables für chart
```

```
  var dataTable = [
    [{label: 'Jahr', role: 'domain', type: 'string'},
      {label: 'Laubwald', role: 'data', type: 'number'},
      {label: 'Nadelwald', role: 'data', type: 'number'},
      {label: 'kein Wald', role: 'data', type: 'number'}]
  ];

  // Funktion zum Einladen der klassifizierten Rasterdaten aus den Assets
  function loadClassification (year) {
    var file = "projects/ee-annagoebel/assets/asset_class_TC_GTB_" + year; // hier die Bezeichnung des ge-
wünschten Assets eintragen (momentan wird bestes Ergebnis eingeladen - GTB/TC)
    return ee.Image(file);
  }

  // Laden der Classification über alle Jahre hinweg, Füllen des dataTable für chart
  for (var i = 0; i < yearFile.length; i++){
    // load classification, clip auf ausgewähltes Polygon
    var classified = loadClassification(yearFile[i].year).clipToCollection(ee.FeatureCollection(spa-
tialJoined));

    // Aufruf Funktion berechne Flächengröße
    var result = calculateArea(classified, yearFile[i].year);

    // Runden und in km2 umrechnen
    var resultLaub = result.getNumber('0').divide(ee.Number(1000)).round().divide(ee.Number(1000)).get-
Info();
    var resultNadel = result.getNumber('1').divide(ee.Number(1000)).round().divide(ee.Number(1000)).get-
Info();
    var resultKein = result.getNumber('2').divide(ee.Number(1000)).round().divide(ee.Number(1000)).get-
Info();

    // Summe
    var resultSum = (resultLaub + resultNadel + resultKein);

    //Füllen des dataTable
    dataTable[(i+1)] = [
      yearFile[i].year,
      resultLaub,
      resultNadel,
      resultKein
    ];

    // definiere Palette zur Darstellung
    var palette = [
      'green', // Laubwald (0)
      'blue', // Nadelwald (1)
      'grey' // kein Wald (2)
    ];

    // Definiere Sentinel ImageCollection
    var bands = ['B2', 'B3', 'B4'];

    var s2 = ee.ImageCollection('COPERNICUS/S2_SR')
              .filterDate(yearFile[i].startDate, yearFile[i].endDate)
              .filterBounds(rlp)
              .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',40))
              .select(bands)
              .map(function(image){return image.clip(rlp)});


    // Funktion, um Layer zu löschen, wenn bereits vorhanden
    var removeLayer = function(name) {
      var layers = karte.layers();
      // list of layers names
      var names = [];
      layers.forEach(function(lay) {
        var lay_name = lay.getName();
        names.push(lay_name);
      });
      // get index
      var index = names.indexOf(name);
      if (index > -1) {
        // if name in names
        var layer = layers.get(index);
        karte.remove(layer);
      }
    };

    var layername = 'Classification ' + yearFile[i].year;
    removeLayer(layername);
    if (i < yearFile.length - 1){
      karte.addLayer(classified, {min: 0, max: 2, palette: palette}, (layername), false);
    } else {
```

```
      karte.addLayer(classified, {min: 0, max: 2, palette: palette}, (layername), true);
    }
  }

  // erstelle Diagramm
  var chart = ui.Chart(dataTable).setChartType('LineChart').setOptions({
    title: ('Zeitreihe ' + gemText),
    hAxis: {title: 'Jahr', titleTextStyle: {italic: false, bold: true}},
    vAxis: {title: 'Flächengröße in km²', titleTextStyle: {italic: false, bold: true}},
    colors: ['green', 'blue', 'grey']
  });

  // entferne Hauptpanel, füge Auswertungspanel hinzu
  ui.root.remove(hauptpanel);
  ui.root.add(auswertungspanel);
  auswertungspanel.clear();
  auswertungspanel.add(ui.Label({value: 'Auswertung', style: {fontWeight: 'bold', fontSize: '18px'}}));
  auswertungspanel.add(ui.Label({value: 'Die Flächengrößen pro Jahr werden angezeigt, wenn Sie mit der Maus
über den Graphen fahren. Durch einen Klick auf den Button oben rechts wird der Graph in einem neuen Fenster
geöffnet.'}));
  auswertungspanel.add(ui.Label({value: 'Bitte beachten Sie, dass kleine Abweichungen in der Zeitreihe auch
auf fehlerhafte Daten (z.B. hohe Wolkenbedeckung) zurückzuführen sein können.'}));
  auswertungspanel.add(ui.Label({value: 'Unter "Layers" kann zwischen den Klassifikationsergebnissen der
letzten Jahre umgeschaltet werden. Über den Schiebebalken jeweils rechts neben Layern kann die Transparenz
geändert werden'}));
  auswertungspanel.add(ui.Label({value: 'Gesamtfläche Laub-, Nadel-, kein Wald: ' + resul-
tSum.toString().replace('.', ',').slice(0,6) + " km²"}));
  auswertungspanel.add(chart);

  // reset button
  var neu = ui.Button({
    style: {width: '480px'},
    label: "neue Anfrage"
  });
  auswertungspanel.add(neu);

  // reset zu ursprüngliche Ansicht
  neu.onClick(function() {
    karte.remove(inspector);
    karte.setOptions("ROADMAP");
    ui.root.remove(auswertungspanel);
    ui.root.add(hauptpanel);
    drawingTools.clear();
    drawingTools.draw();
  });

  //berechne Flächengrößen
  function calculateArea (img, year){

    var waldClip = img.reduceToVectors({
      geometry: spatialJoined,
      scale: 10
    });

    // Berechne Flächengröße jeder Klasse in m2
    var areaImage = ee.Image.pixelArea().addBands(img);
    var areas = areaImage.reduceRegion({ //areas: ee.Dictionary
      reducer: ee.Reducer.sum().group({
        groupField: 1,
        groupName: 'class',
      }),
      geometry: waldClip.geometry(),
      scale: 10,
      maxPixels: 1e13,
      tileScale: 8
    });

    var classAreas = ee.List(areas.get('groups'));
    var classAreaLists = classAreas.map(function(item) {
      var areaDict = ee.Dictionary(item);
      var classNumber = ee.Number(areaDict.get('class')).format();
      var area = ee.Number(areaDict.get('sum'));
      return ee.List([classNumber, area]);
    });
    var result = ee.Dictionary(classAreaLists.flatten());
    return(result);
  }
});
```