DM

TerraSenseTK

A toolkit for remote soil nutrient estimation

MASTER DISSERTATION

Manuel Afonso Soares Pereira
MASTER IN INFORMATICS ENGINEERING

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

September | 2022

TerraSenseTK

A toolkit for remote soil nutrient estimation

MASTER DISSERTATION

Manuel Afonso Soares Pereira
MASTER IN INFORMATICS ENGINEERING

ORIENTATION
Filipe Magno Gouveia Quintal

CO-ORIENTATION
Amâncio Lucas de Sousa Pereira

Acknowledgments

I would like to thank my friends and family, especially my mother, for supporting me and encouraging me to follow my objectives.

I am deeply indebted to Professor Lucas Pereira and Professor Filipe Quintal for its mentoring and overall help, notably Professor Lucas Pereira for his guidance and his ability to motivate and deliver an higher quality work.

I am also grateful to the Further Energy and Environment research Laboratory (FEELab) at Interactive Technologies Institute (ITI) for providing the necessary equipment to run the thesis' experiments. I would also like to recognize ISOPlexis, in the person of Eng. Humberto Nóbrega, for clearing some technical questions and providing data over fields of Madeira Island. This work wouldn't be the same without the help of everyone involved.

Abstract

Intensive farming endangers soil quality in various ways. Researchers show that if these practices continue, humanity will be faced with food production issues. For this matter, Earth Observation, more concretely Soil Sensing, along with Machine Learning, can be employed to monitor several indicators of soil degradation, such as soil salinity, soil heavy metal contamination and soil nutrients estimation. More concretely, Soil Nutrients are of great importance. For instance, to understand which crop better suits the land, the soil nutrients must be identified. However, sampling soil is a laborous and expensive task, which can be leveraged by Remote Sensing and Machine Learning.

Several studies have already been developed in this matter, although many gaps still exist. Among them, the lack of cross-dataset evaluations of existing algorithms, and also the steep learning curve to the Earth Observation domain that prevents many researchers from embracing this field. In this sense, we propose TerraSense ToolKit (TSTK), a python toolkit that addresses these challenges.

In this work, the possibility to use Remote sensing along with Machine Learning algorithms to perform Soil Nutrient Estimation is explored, additionally, a nutrient estimation toolkit is proposed, and the effectivity of it is tested in a soil nutrient estimation case study.

This toolkit is capable of simplifying Remote Sensing experiments and aims at reducing the barrier to entry to the field of Earth Observation. It comes with a preconfigured case study which implements a soil sensing pipeline. To evaluate the usability of the toolkit, experiments with five different crops were executed, namely with Wheat, Barley, Maize, Sunflower and Vineyards. This case study gave visibility to an underlying unbalanced data problem, which is not well addressed in the current State of the Art.

Keywords: remote sensing, soil nutrient estimation, toolkit, python, satellite imagery, soil sensing

Resumo

A agricultura intensiva põe em perigo a qualidade do solo de várias formas. Os investigadores mostram que, se continuarmos com estas práticas, a humanidade será confrontada com questões de produção alimentar. Para este efeito, a Observação da Terra, mais concretamente o Sensoriamento do Solo, juntamente com a aprendizagem automática, podem ser utilizadas para monitorizar vários indicadores da degradação do solo, tais como a salinidade do solo, a contaminação do solo por metais pesados e a quantificação dos nutrientes do solo. Mais concretamente, os Nutrientes do Solo são de grande importância. Por exemplo para compreender qual a cultura que melhor se adapta ao solo, os nutrientes do solo devem ser identificados. No entanto, a amostragem do solo é uma tarefa trabalhosa e dispendiosa, que pode ser impulsionada pela percepção remota e pela aprendizagem automática.

Já foram desenvolvidos vários estudos sobre este assunto, embora ainda existam muitas lacunas. Entre eles, a falta de avaliações cruzadas dos algoritmos existentes, e também a curva de aprendizagem acentuada para o campo de Observação da Terra que impede muitos investigadores de enveredar por este campo. Neste sentido, propomos TSTK, um toolkit em python que aborda estes desafios.

Neste trabalho, é explorada a possibilidade de usar a Percepção Remota juntamente com os algoritmos de Aprendizagem Automática para realizar a Estimativa de Nutrientes do Solo. Além disso, é proposto um toolkit de estimativa de nutrientes e também um pipeline para o devido efeito, a efetividade do toolkit é testada num caso de estudo de Estimação de Nutrientes no Solo.

Este toolkit é capaz de simplificar as experiências de Percepção Remota e visa reduzir a barreira de entrada no campo da Observação da Terra. Para avaliar a usabilidade do toolkit, foram executadas experiências com cinco culturas diferentes, nomeadamente Trigo, Cevada, Milho, Girassol e Vinha. Este caso de estudo deu visibilidade a um problema subjacente de dados desiquilibrados, o qual não é bem identificado no Estado da Arte atual.

Keywords: deteção remota, estimação de nutrientes no solo, toolkit, python, imagens por satélite, deteção de solo, sensoriamento do solo

Contents

List of Tables vi

List of Figures vii

1 Introduction 2
- 1.1 Motivation . . . 2
- 1.2 Research Objectives . . . 4
- 1.3 Document Organization . . . 4

2 Technological Background 5
- 2.1 Earth Observation Using Satellites . . . 5
  - 2.1.1 Remote sensing . . . 5
  - 2.1.2 Spacial Programs . . . 5
  - 2.1.3 Orbits . . . 6
- 2.2 Satellite Data . . . 6
  - 2.2.1 Electromagnetic Spectrum - **EMS** . . . 7
  - 2.2.2 Hyperspectral Imagery (HI) . . . 8
  - 2.2.3 Data Acquisition . . . 9
    - The Copernicus Sentinel Program . . . 9
  - 2.2.4 Data Management and Analysis . . . 11

3 State-of-the-Art 12
- 3.1 Brief Machine Learning Introduction . . . 12
  - 3.1.1 Supervised Learning . . . 12
  - 3.1.2 Unsupervised Learning . . . 13
  - 3.1.3 Reinforcement Learning . . . 13
- 3.2 Machine Learning and Satellite Data in Soil Sensing . . . 13
- 3.3 Publicly Available Datasets . . . 16
- 3.4 Development Options . . . 17
  - 3.4.1 Libraries . . . 17
  - 3.4.2 Frameworks . . . 18
  - 3.4.3 Toolkits . . . 19
- 3.5 Summary . . . 19

4 Proposed Solutions 21
- 4.1 TerraSenseTK . . . 21
  - 4.1.1 Datasets . . . 21
  - 4.1.2 Algorithms . . . 22

Feature Selection . . . 22
Hyper-parameter Optimization . . . 22
4.1.3 Performance Evaluation . . . 23
4.1.4 Experiment . . . 23
4.2 Soil Nutrient Estimation Pipeline . . . 23

5 TerraSenseTK Implementation **26**
5.1 TerraSenseTK Overview . . . 26
5.2 Dataset Module . . . 27
5.2.1 TSPatch . . . 27
5.2.2 Downloader . . . 28
5.2.3 Dataset . . . 28
5.2.4 Parser . . . 28
5.3 Machine Learning Algorithms Module . . . 28
5.3.1 Regression Algorithms . . . 29
Multi-Layer Perceptron . . . 29
Support Vector Machines . . . 30
Partial Least Squares . . . 30
Gradient Boosting . . . 31
Random Forest . . . 32
5.3.2 Feature Selection . . . 32
Sequential Feature Selection . . . 32
5.4 Performance Evaluation Module . . . 33
5.4.1 Metrics . . . 33
5.4.2 Cross Validation . . . 33
5.5 Experiment Module . . . 33
5.6 Utils Module . . . 35
5.6.1 Results . . . 35
5.6.2 EOTasks . . . 35
5.6.3 LUCAS dataset pickle . . . 35

6 Case Study **36**
6.1 Data Engineering . . . 36
6.1.1 Dataset Creation . . . 36
Ground-Truth Mapping . . . 37
6.1.2 Outlier Detection and Removal . . . 39
6.2 Crop Selection . . . 40
6.3 Feature Engineering . . . 40
6.3.1 Feature Overview . . . 40
6.3.2 Manual Feature Selection . . . 42
6.3.3 Automatic Feature Selection . . . 43
6.4 Machine Learning Algorithms and Hyper-parameter Optimization . . . 43
6.4.1 Algorithm Selection . . . 44
6.4.2 Target Logarithmic Transformation . . . 45
6.4.3 Hyper-Parameter Optimization . . . 45
6.4.4 Training and Validation . . . 47
6.5 Results and Discussion . . . 47

6.5.1 Results Overview - Using ML Flow . . . 47
6.5.2 Performance Per Crop . . . 47
Original Data . . . 48
Logarithmic Transformation . . . 49
Discussion . . . 50

7 Conclusion **52**
7.1 Summary . . . 52
7.2 Research Implications and Potential Applications . . . 52
7.3 Limitations and Future Work Directions . . . 53

Bibliography **54**

A Appendix chapter **61**

Glossary **64**

Acronyms **65**

List of Tables

2.1 Bands that the MSI on the Sentinel 2 observes[1] 10

3.1 Brief conclusion of the studies presented in the State of the Art 16

5.1 Implemented metrics 33

6.1 LUCAS Copernicus attributes, based on [63] 38
6.2 Spectral indices calculated from spectral measurements for Sentinel2 [85] 39
6.3 Top 40 crops with most occurences 41
6.4 Selected input features for each crop 44
6.5 Quantity of images for each crop 47
6.6 Metrics obtained by original data for all crops 49
6.7 Metrics obtained by performing log transformation to remaining crops 49

A.1 Original algorithms' Hyper-parameter Optimization 62
A.2 Algorithms' Hyper-parameter Optimization For Log-Scaled data 63

List of Figures

2.1 The Electromagnetic Spectrum [2] . . . 7
2.2 Comparison between True color and False Color image over Madeira island in December 2020 [3] . . . 8

3.1 Output of the work in Efremova et al, 2019. [43], dry soil is marked with red and wet soil with blue. . . . 14

4.1 TerraSenseTK core diagram . . . 22
4.2 Earth Observation Pipeline . . . 24
4.3 Pipeline for nutrient estimation. . . . 24
4.4 Process of data acquisition . . . 25

5.1 TerraSenseTK folder structure. . . . 26
5.2 Dataset module entity relationship diagram . . . 27
5.3 Data model of an EOPatch . . . 27
5.4 The Data Conversion Process in the Parser Class . . . 29
5.5 Entity Relationship diagram of the algorithm module . . . 30
5.6 Class diagram of the Experiment and Results classes . . . 34

6.1 Dataset consolidation process . . . 36
6.2 Comparisson of groundtruth data for Common wheat values with outliers(left) and after running outlier cleaning(right) . . . 39
6.3 Distribution of the bands of the dataset . . . 41
6.4 Distribution of the indices of the dataset . . . 42
6.5 Spearman correlation performed over the Common wheat crop . . . 43
6.6 Data Distribution for original data, top three plots, Barley, Wheat, and Maize, bottom two plots, Sunflower, and Vineyards, respectively . . . 44
6.7 Data Distribution with logarithmic scaling, top three plots, Barley, Wheat and Maize, bottom two plots, Sunflower, and Vineyards, respectively . . . 46
6.8 Feature importance of Random Forest for Common wheat . . . 46
6.9 MLFLow user interface . . . 48
6.10 MLFLow user interface auto-generated graphs . . . 48
6.11 Prediction of the original data with a) Random Forest and b) Gradient Boosting Machines 50
6.12 Prediction of the log-scaled data with a) Random Forest and b) Gradient Boosting Machines 51

Chapter 1

Introduction

1.1 Motivation

Satellite imagery and Earth Observation (EO) have been two growing fields since the start of LANDSAT in 1972, with the launch of the ERTS-1 satellite. Consequently, there are numerous research programs aiming for EO, such as LANDSAT[1] and Copernicus[2]. Since the decade of 1980 researchers have been equipping multispectral imagery systems in satellites to try to improve the understanding of Earth. These allow for visualization of the earth as a whole, being even capable of seeing far beyond what the human eye can see[1].

Many areas have had breakthroughs through the usage of satellite imagery, such as weather forecasting and land usage monitoring. For example, in Toth et al, 2013. [2], the authors have used satellite imagery to survey the size of agricultural fields. Climate change, such is the case in [3], authors predicted the amount of trapped Carbon in soils from different regions using satellite imagery and Machine Learning Techniques. In [4], researchers gathered information on the nutrients of the soil in Spruce plantations around Norway with the aid of remote sensing. There have been also advancements in agriculture, authors in Karthikeyan et al, 2010. [5], found correlations in vegetation growth and soil properties by using near-infrared imaging stemming from satellite imagery.

Agriculture, in general, has not advanced much. In fact, after 50 years without significant changes, agriculture and food production are now facing unprecedented challenges [6, 7], for example, increased demand. It is estimated that by 2050 the population grows by a third [8]. According to the same source, land and water availability are increasingly stressed and becoming diminished due to resource degradation. In addition, intensive farming techniques, for instance, excessive application of insecticides and fertilizers are playing a large role in soil erosion, leaving a never-ending trace of fauna and flora extinction.

According to the Food and Agriculture Organization (FAO), if the practice of intensive farming continues, by 2070 there could be no more arable and productive topsoil[7]. Furthermore, considering that food production takes almost half of the habitable land, and agriculture alone makes up for about 14% of global greenhouse gas emissions [9, 10], it becomes evident that agriculture must mitigate its carbon footprint.

Traditional agriculture is affected in several ways, but one of the most important is the crop yields [11]. Most of the time these are unknown, given that crop yield is affected by many factors, such as pests, temperature, moisture, and most importantly, soil quality. For instance, when a crop gets infected by a pest, by the time the farmers spot it, the pest has already spread to a large portion of the field,

[1] https://landsat.gsfc.nasa.gov/
[2] https://www.copernicus.eu/pt-pt

leading to the loss of that crop.

What intensive farming upholds is to apply insecticide beforehand, not knowing if the crops are going to get infected. Which leads to soil and groundwater massive contamination. Given that soil quality is of great importance, one of the areas that one should focus is on soil sensing. Monitoring soil the traditional way is extremely expensive and time-consuming. This is done by probing the ground with special sensors or collecting samples and further analyzing them in a laboratory, which only covers small areas and does not have soil heterogeneity in mind. Given the dimension of farming sites, this method is, without a doubt, impractical.

In the last decade, Machine Learning (ML) techniques have seen tremendous improvements, mostly driven by the abundance of data and the unprecedented advancements in high-performance computing. As such, in recent years, ML has been recognized as one of the most powerful tools for tackling climate change [10]. Unsurprisingly, precision agriculture (or more broadly speaking farms and forecasts) is among the focus areas. Furthermore, with Earth Observation Data (EOD) becoming widely available, it was just a matter of time before the research community started to explore how these two fields can be leveraged toward more sustainable agricultural practices.

For example, in order to obtain more reliable estimates of crop yields, scientists have developed prediction models [12] based on weather forecasts, and water availability. Along with these models, satellite data can help by gathering extra data and adding new variables that can better predict the crop yield [13]. For pest detection, hyperspectral imagery sensors equipped in satellites, along with machine learning are being used in plant phenotyping. Recent studies such as Wahabzada et al. 2016 [14], have shown the possibility of perceiving the health status of the crop and alerting the farmer beforehand. This allows for the minimization of insecticide usage and the health hazards associated with it, by treating only the affected crops.

Soil monitoring can be aided by the inclusion of satellite imagery. Given that vast areas can be covered, there is no need for groundtruth data, drastically decreasing the cost and time of soil sensing [15]. On a further note according to [16], the processing of each sample taken from the field costs around 24$ not accounting for the values of transportation and labor, meaning that remote sensing is capable of reducing the cost of soil sensing in a value of 424 million dollars for an area of 50x50 km.

Yet, despite the growing interest in combining machine learning with earth observation data in the most diverse application domains (e.g., [17, 18]), there are several fundamental challenges to be addressed before ML techniques for EOD can be widely adopted by researchers, practitioners and application developers [19, 20]. In a report from the 2020 edition of the NASA workshop on ML4EO[3], these challenges are grouped into three major categories: 1) **training data**, 2) **algorithms and models**, and 3) **tools and analytic frameworks** [20]. Among the challenges discussed in this report, one of the most crucial is the limited number of open-source software and ML frameworks to develop, evaluate and share ML learning models. Instead, the current practice is to keep algorithms private which leads to reproducibility issues and consequently difficult the comparison of results across studies. Likewise, another crucial challenge is the lack of baseline implementations of ML models that allow for benchmarks with other ML models and even across datasets.

Interestingly, these challenges are transversal to many domains in ML research. For example, in the field of Non-Intrusive Load Monitoring (NILM) [21] where ML is used to infer the consumption of individual appliances from the total energy consumption of a building, it was only after almost three decades since the first published work (in 1985 [22]) that NILM researchers released the Non-Intrusive Load Monitoring Toolkit (NILMTK) [23, 24].

[3] see https://www.radiant.earth/events/nasa-ml-2020/

1.2 Research Objectives

Against this background, and inspired by the success of NILMTK, in this thesis we set to lay the foundations for the development of a new toolkit for ML applications in EO data with a focus on agricultural applications. To state more precisely, this thesis has two main research contributions.

1. TerraSenseTK, a toolkit that enables the estimation of nutrients in the soil with information acquired from remote sensing, and additionally to any existing variable in the dataset at hand.
2. A novel Soil Nutrient Estimation Pipeline to streamline the usage of the toolkit

Ultimately, pushing towards the development of such a platform will not only represent a very good engineering achievement, but it will also provide a strong contribution to the research community ML4EO, providing a tool for newcomers to EO that will reduce the barrier to entry in the field. This is especially important due to the steep learning curve of these technologies and the broad range of knowledge necessary to understand and analyze EO data. Our goal is to achieve a useful framework to improve the current process of performing Remote Sensing by simplifying the process to the end-user, by defining a structure of dataset/information regarding the soil and baseline implementations to further develop ML models. Especially, by using the toolkit for a specific problem of soil nutrient estimation, as a Proof of Concept (PoC).

1.3 Document Organization

The remaining of this thesis is organized as follows. Firstly in Chapter 2, additional background will be given in order to further understand the contents of this work as well as information regarding the history of EO. In Chapter 3, the most recent approaches to EO will be presented. Then, the rationale of the proposed solution in Chapter 4 will be explained, giving a more theoretical explanation. Additionally, Chapter 5 explains each module of the toolkit. Moreover, the proposed work will be explained and a number of practical examples will be given. In addition, the Chapter 6 contains the evaluation process, namely how the dataset was built and the performance obtained by the toolkit in a set of experiments. The document ends with Chapter 7 with an overall conclusion and prospects for the future.

Chapter 2

Technological Background

Considering the broad nature of the fields of EO and Satellite data, this chapter provides technical background in this topic such that the reader can more easily understand the remaining of this thesis. This chapter is divided into two sections. The first part includes an explanation of terms related to satellites, a number of existing spacial programs, and the different types of orbits. The second part describes the electromagnetic spectrum and the different sensors that satellites are equipped with.

2.1 Earth Observation Using Satellites

2.1.1 Remote sensing

Remote sensing is the gathering of data, through sensors without actually touching the object. An example of remote sensing in the medical field is the utilization ofMagnetic Resonance Imaging (MRI) and X-RAY, since the sensor is not physically touching the subject.

According to [25], there are two types of remote sensing, active and passive. Active remote sensing is when the sensor emits a certain signal and captures it, while in passive remote sensing the sensor captures a signal from another source. The example that was given before, X-RAY is active remote sensing because it outputs X-radiation and the sensor captures it. Another example of active remote sensing is sonar or even Radio Detection And Ranging (RADAR), since they emit a frequency that will be captured by the sensors. An example of passive remote sensing is a digital camera, given that it captures the light of the sun that is reflected by the objects.

2.1.2 Spacial Programs

Several efforts have been made on satellites for EO. The European Space Agency (ESA) and National Aeronautics and Space Administration (NASA) are the most well-known agencies in the western world, having programs focused on Earth Observation.

NASA launched the LANDSAT program in 1972, its EO program and since then, it has completed nine missions (LANDSAT 1 - 9). Additionally, NASA also has the Moderate Resolution Imaging Spectroradiometer (MODIS)[26] program with a constellation of two satellites, but, this program, as the name suggests, has a much coarser spatial resolution. ESA started Copernicus in 2014, launching five missions satellites (SENTINEL 1, 2, 3, 5P, and 6), but, it is important to mention that there are still two more missions in development (SENTINEL 4 and 5).

Both these projects have several missions, this is given to the fact that each mission focuses on a specific task. For instance, Copernicus has six main topics: Atmospheric, Land, Marine, Climate

Change, Security, and Emergency.

There are also few other spacial programs, namely from Japan Aerospatial Exploration Agency (JAXA)[1] and Indian Space Research Organisation (ISRO)[2].

However, at this point, it is important to stress that the main focus of this thesis will be on Copernicus, since it is being developed in Europe, and it is more active in the remote sensing scene, constituting the State-of-the-Art regarding this area.

2.1.3 Orbits

An orbit is the course in which an object travels around another large object in space. In this case, the object is a satellite and the larger object is the earth. There are different types of orbits: Geostationary, Low Earth, Medium Earth, Polar, and Sun Synchronous orbits. A brief description for each of the orbit types is described below. (see [3] for additional details).

Geostationary Orbit - GEO In a geostationary orbit, a satellite appears to be over a fixed position on earth. Particularly useful for telecommunications or weather observation over a particular location, along the equator, over a precise altitude of 35786 km (much farther than most orbits).

Low Earth Orbit - LEO This orbit is the most common for imagery satellites allowing for better resolution images. While orbiting on low earth, the traveling speed is nearly 8 km per second meaning it has a revisiting period (the time it takes to be over the same position) of 90 minutes and the altitude is nearly 1000 km above the earth.

Medium Earth Orbit - MEO This one is similar to LEO, but at farther distances, ranging from 2000 km to 35 000 km.

Polar Orbit and Sun-Synchronous Orbit - SSO This orbit rather than going from west to east, it travels from north to south, but with a slight deviation of about 30 degrees. These are also called sun-synchronous because they can pass through a certain place at an exact time, for example passing over Paris at 7 pm every day. These are the most useful orbits for remote sensing because they have complete earth coverage. Sometimes satellites in SSO are synchronized so that, in a way, they are constantly in a determined time, for example, dusk, in order to avoid earth shadowing the sun. In a Sun Synchronous Orbit (SSO), the altitude is about 200-800km, with a travel speed of about 7.5 km/s

2.2 Satellite Data

Data gathered from satellites is considered satellite data. There are several different types of data and different potential usages. Scientists have mounted different instrument payloads to satellites such as Synthetic Aperture Radar (SAR) [27], which is a Radar that allows the mapping of 2D or 3D objects. There are also multispectral instruments(MSI), which are clusters of sensors that capture different bands of the electromagnetic spectrum. Then there are several iterations of these instruments, which allow for capturing different objects.

[1] https://www.eorc.jaxa.jp/en/distribution/standard_dataset/
[2] https://www.isro.gov.in/spacecraft/list-of-earth-observation-satellites
[3] https://www.esa.int/Enabling_Support/Space_Transportation/Types_of_orbits

2.2.1 Electromagnetic Spectrum - EMS

To further discuss Remote Sensing, several important aspects regarding the field of Electromagnetic Spectrum (EMS) need to be addressed. There are different wavelengths, a number of them are visible, while others are not. Similarly, some are harmful, while others are harmless. Regular cameras only capture the visible range, capturing the red, green, and blue hues(from 400 to 800 nm). Hyperspectral/Multispectral cameras, on the other hand, capture different parts of the EMS, in order to detect other features invisible to the human eye/visible range, such as gaseous trades, and diverse chemical compounds.

The EMS has been split into seven categories, even though there is not a well-established division between them. Figure 2.1 shows a graphical representation of the Electromagnetic Spectrum. Radio Waves, Microwaves, Infrared, Visible light, Ultraviolet and X-ray and Gamma bands will briefly be explained as well as a few potential uses [28].

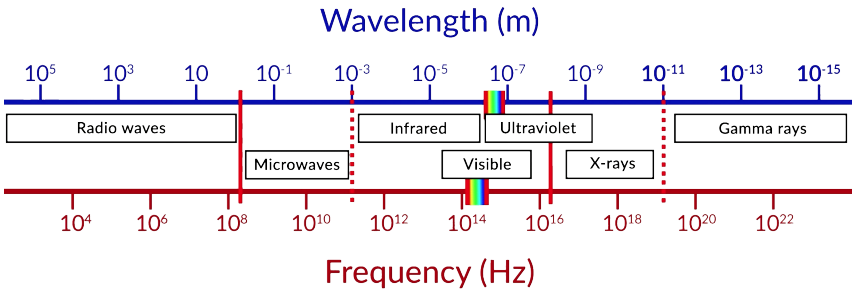


Figure 2.1: The Electromagnetic Spectrum [4]

Radio waves refer to the energy that has the greater wavelength. Are commonly used for communication, and have low surpassing capabilities, meaning it has to have a direct line of sight to the receptor.

Microwaves: are radio waves with shorter wavelength, about 0.1m (see Figure 2.1). Has better surpassing capabilities, and it can pass through clouds and hazy weather.

Infrared: are the most interesting in the scope of this work since most hyperspectral imagery systems are tuned to capture this category. Infrared radiation covers the range of about 1mm to 750 nm. At these lengths, the objects have different reflectance values [28] and are divided into three parts: Far Infrared (FIR), Middle Infrared (MIR), Near Infrared (NIR).

- FIR - From 1mm to 10 μm. The water in Earth's atmosphere absorbs strongly in this range.
- MIR - From 10 - 2,5 μm. Hot objects radiate strongly in this region. It is also called *fingerprint region* since the absorption spectrum for each chemical compound or element is very specific.[5]
- NIR - From 2500 to 750 nm. This region has several usages, such as pulse oximetry, and neuroimaging in the medical fields[6], but also in atmospheric chemistry and agrochemical industry[29].

[4] https://www.jove.com/science-education/11295/the-electromagnetic-spectrum
[5] https://www.chemguide.co.uk/analysis/ir/fingerprint.html
[6] https://www.medscape.com/viewarticle/778219

Visible light: is the region the human eye is the most sensitive to. When a person views an object it is simply viewing the object's reflection of a light source, such as the sun or a lamp, in the range of 400nm (red) to 780nm (violet).

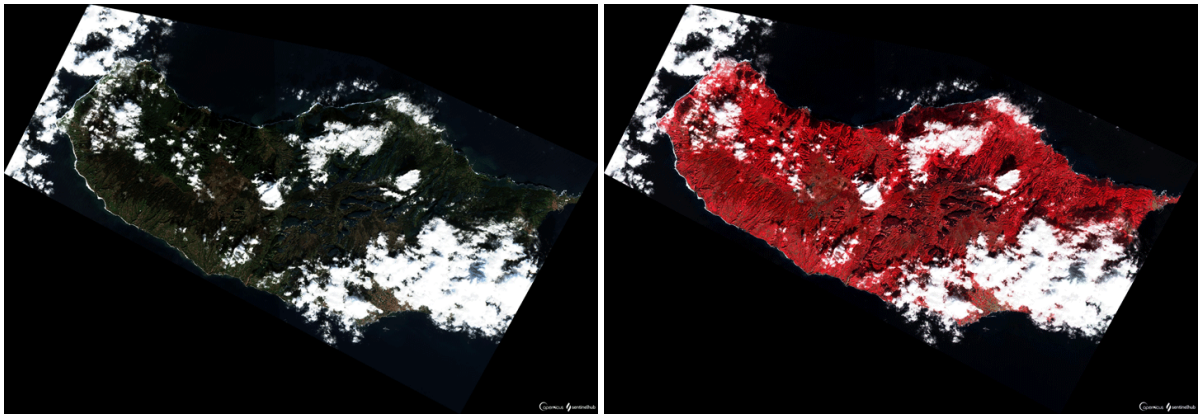
Ultraviolet (UV): about 10% of the total sun energy is UV. This energy is extremely disruptive to biological organisms, luckily the atmosphere captures most of the dangerous UV rays, and various compounds react to UV, through the process known as fluorescence.

X-rays: are extremely important to the medical field. It can pass through soft substances, although they are very energetic and can cause long-term damage. Our atmosphere completely blocks these types of energy, so they do not have much use for EO [30].

Gamma rays: are the most disruptive type of energy. There is still no limit to how small its wavelength can be. They are used normally for killing cancerogenic cells, i.e. cancer treatment.

2.2.2 Hyperspectral Imagery (HI)

One might think that an hyperspectral image is similar to a regular color image, but it is quite different. An RGB image is composed of three filters that capture energy in the visible range, namely, Red, Green, and Blue. For each pixel, there is a value for each of the filters, and that is how an image is composed. In an hyperspectral image, instead of three filters, there can be dozens or even hundreds of filters, outside of the visible range. Thus, these images cannot be represented as easily as regular images.



(a) True Color Image (b) False Color Image

Figure 2.2: Comparison between True color and False Color image over Madeira island in December 2020 [7]

In order to get meaningful visual insights out of HI, one must transform them to RGB, through a process called false color [31]. There are several formulas for it, depending on the filters that the imaging instrument has. For example, the Normalized Difference Vegetation Index (NDVI)[32], where the authors used the *Sentinel 1* Near-Infrared channels to monitor the phenological cycle of crops, and in tasseled cap [33], authors found that the maturity state and the crop itself can be guessed with a good degree of certainty, based on its reflectance values, using LANDSAT imagery, again in the Near-Infrared spectral band.

[7] Images taken from EO Browser: https://cutt.ly/ajTseiu

The images in Figure 2.2 were captured with the *Sentinel 2* in 14th December 2020 over Madeira Island. These images are a practical example of a True Color image (Figure 2.2a) and a False Color one (Figure 2.2b). In this case, the False Color composite used is a composition of the near-infrared band, green and blue, and is commonly used to assess plant density and health, since vegetation reflects near-infrared and green light. Cities and exposed ground are grey or brown, and water appears blue or black[8]. It is possible to verify that in Figure 2.2b, Funchal appears as brown given that it does not have as much vegetation as the remaining land.

2.2.3 Data Acquisition

In order to capture Earth Observation data, these satellites need a device that captures information, such is the case of sensors. There are several types of sensors, such as imaging, measuring, and aerosol sensors. Firstly, the terminology of the field will be explained. Additionally, each of the Copernicus' Sentinels' payloads will be presented, given that these present the State-of-the-Art in the remote sensing environment and were deployed in Europe [9].

Spacial Resolution Spatial resolution is a common term in EO. It measures the distance each pixel covers. For regular pan-sharpened imaging, it is about 30 centimeters per pixel[10]. On multispectral equipment, it is around 10 meters as is the case of the Sentinel 2.[11]

Scenes are aggregations of images, given that most satellites do not capture a single image, rather it is captured continuously. In this sense, the correct term should be scene instead of an image.

The Copernicus Sentinel Program

Sentinel 1 Was the premiere of the Copernicus program. It is composed of a constellation of two satellites (Sentinel 1A and 1B), launched in 2014 and 2016 respectively, and travels in an SSO. Its main sensor is a SAR in the C-band, meaning it can capture images independently of the time and weather. Its revisit time is 6 days. Regarding the spatial resolution, it changes with its capture mode, but at best it is 5 meters. This mission is approaching its expected end of life (EOL) as of the writing of this thesis.

Sentinel 2 Is composed of two satellites (Sentinel 2A and 2B), and has a revisit time of five days in an SSO. The payload consists of an Multispectral Instrument (MSI) with thirteen spectral bands, with different spacial resolutions at *ten*, *twenty*, and *sixty* meters. Six of the bands are in the Near-Infrared and Infrared regions, while the remaining seven are in the visible region. In Table 2.1 it is possible to see that the filters differ very slightly in both satellites. Unlike the C-band Synthetic Aperture Radar (SAR) in Sentinel 1, the MSI is a light sensor, meaning it can only capture with appropriate weather conditions. The main applications for Sentinel 2 are climate change, land monitoring, and emergency management.

Sentinel 3 Sentinel 3 is also constituted by a constellation of two satellites (S-3A and S-3B), although it has a more complex payload than the other missions. It has an Ocean and Land Colour Instrument (OLCI), Sea and Land Surface Temperature Instrument (SLSTR), SAR Radar Altimeter, and Microwave Radiometer (MWR). This mission is focused mainly on the sea, such as ocean floor mapping, sea

[8] see https://earthobservatory.nasa.gov/features/FalseColor/page6.php
[9] see https://sentinel.esa.int/web/sentinel/missions/
[10] https://www.euspaceimaging.com/products/image-specs/
[11] https://sentinel.esa.int/web/sentinel/missions/sentinel-2/overview
[11] see https://sentinel.esa.int/web/sentinel/missions/sentinel-2/instrument-payload/resolution-and-swath

Table 2.1: Bands that the MSI on the Sentinel 2 observes[12]

Spacial Resolution (m)	Band No	S2A		S2B	
		Wavelength (nm)	Bandwidth (nm)	Wavelength (nm)	Bandwidth (nm)
10	2	492.4	66	492.1	66
	3	559.8	36	559.0	36
	4	664.6	31	664.9	31
	8	832.8	106	832.9	106
20	5	704.1	15	703.8	16
	6	740.5	15	739.1	15
	7	782.8	20	779.7	20
	8A	864.7	21	865.0	22
	11	1613.7	91	1610.4	94
	12	2202.4	175	2185.7	185
60	1	442.7	21	442.2	21
	9	945.1	20	943.2	21
	10	1373.5	31	1376.9	30

temperature, humidity measurement, and sea surface speed measurements, but most importantly for this thesis, it also compliments the vegetation products started with Sentinel 2.

ESA makes available two types of data from this mission. In a near real-time operational and timely manner, Level-1B visible, shortwave, and thermal infra-red radiances and Level-2 topography products.

Sentinel 4 As of the writing of this thesis, this satellite is due to be launched later this year along with the Sentinel 5. It will have a geostationary orbit over Europe and has the objective of measuring air quality trace gases such as O_3 (Ozone), NO_2 (Nitrogen dioxide), SO_2 (Sulfur dioxide), CH_2O (Formaldehyde), $C_2H_2O_2$ (glyoxal), as well as aerosol and cloud properties. It will be equipped with three spectrometers in the Ultraviolet, Visible, and Near Infrared bands, and will have a much coarser spatial resolution of about 8 km.

Sentinel 5P The P in 5P stands for Precursor. Unlike Sentinel 4 and 5, the 5P has already been launched. It is on a LEO/SSO and its main objective is to perform atmospheric measurements to be used for air quality, ozone, UV radiation, and climate monitoring & forecasting. It is equipped with TROPOMI: an instrument constituted by four spectrometers observing the Ultra Violet (UV), Visible, NIR and Short Wave Near Infrared (SWIR) spectrum, with a spatial resolution of 7 km.

Sentinel 5 Is equipped with five dedicated spectrometers (one in UV, one ranging from ultraviolet to visible band(UV2VIS), one in NIR, and two in SWIR). The UV spectrometer has a spatial resolution of 52 km while the remaining ones will have a spatial resolution of 7 km.

Sentinel 6 Was launched back in November 2020, and has the objective of measuring sea surface topography. Its payload is a SAR tuned in the L band based on the *Sentinel 3* payload and an Advanced Microwave Radiometer (AMR).

2.2.4 Data Management and Analysis

As one can imagine, these satellites produce large amounts of data. Project Copernicus alone rakes up to fifteen *terabits* worth of data every single day. One can download these datasets, but given the dimension it, it is impractical to download a whole day's worth of data. Project Copernicus makes available its data to users in several processing levels. The first processing levels are not available for public use, i.e. raw format. Only after performing the orthorectification process, the data is available to the user. The access levels of Copernicus are the following[13]:

> Level-0 is compressed raw data. The Level-0 product contains all the information required to generate the Level-1 (and upper) product levels.
>
> Level-1A is uncompressed raw data with spectral bands coarsely coregistered and ancillary data appended.
>
> Level-1B data is radiometrically corrected radiance data. The physical geometric model is refined using available ground control points and appended to the product, but not applied.
>
> Note: Level-0, Level-1A and Level-1B products are not disseminated to users.
>
> Level-1C product provides orthorectified Top-Of-Atmosphere (TOA) reflectance, with sub-pixel multispectral registration. Cloud and land/water masks are included in the product.
>
> Level-2A product provides orthorectified Bottom-Of-Atmosphere (BOA) reflectance, with sub-pixel multispectral registration. A Scene Classification map (cloud, cloud shadows, vegetation, soils/deserts, water and snow) is included in the product.

Given the size of data, the current practice is to use an EO tool, such as EOBrowser[14], or even Google Earth Engine [15]. These software products allow for easy manipulation of the data, and select locations and dates of interest. In the case of Google Earth Engine, also offers an Integrated Development Environment (IDE).

Additionally, a number of web-based sources to visualize EO data exists, such as the case of EOBrowser and Soobloo. The most uncomplicated visualizer is the Sentinel Hub EOBrowser, which has a well-thought graphic interface, which allows users to download scenes from the Copernicus program satellites and perform experiments with small sets of data directly from the *internet browser*. Another visualizer is Sobloo[16], which is a marketplace and visualizer for EO, where they make available False Color filters for specific use cases. It is important to note that it has data regarding the Copernicus and Landsat-9 missions.

For programmers, Copernicus allows for scene download through SciHub either from their website or by their Application Programming Interface (API), APIHUB[17].

Google Earth Engine is the most flexible of the tools, according to Gomes et al.[34]. Furthermore, it also enables the application of machine learning models, offering support to implementations of keras[18]. Finally, it is the platform that gathers the most part of publicly available datasets.

[13] For more information regarding the access levels of Copernicus, visit: https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-2-msi/products-algorithms

[14] EOBrowser, Earth observation tool, available at https://apps.sentinel-hub.com/eo-browser

[15] Google Earth Engine, available at https://code.earthengine.google.com/

[16] https://sobloo.eu/

[17] APIHUB documentation, available at https://scihub.copernicus.eu/twiki/do/view/SciHubWebPortal/APIHubDescription

[18] Keras is the most used deep learning framework, see: https://keras.io/

Chapter 3

State-of-the-Art

In this chapter, we briefly describe the State-of-the-Art (SOA) in the application of ML and EOD for agricultural applications with a focus on soil sensing. This chapter is divided into four sections: First, in section Section 3.1 we introduce the field of machine learning and talk about its underlying features. Second, Section 3.2 we provide an overview of how ML has been applied to solve this problem. Third, in Section 3.3 we survey publicly available related datasets. Finally, in Section 3.4 we provide an overview of the different options for application developers and researchers.

3.1 Brief Machine Learning Introduction

Machine Learning is a branch of Artificial Intelligence. With Machine Learning, a mathematical algorithm learns the data, with minimal human intervention. In simple terms, machine learning uses data to train a mathematical model, and when similar data is fed to that model, it knows how to categorize it. Classical Machine learning can be subdivided into 3 main categories [35]: supervised learning, unsupervised learning, and Reinforcement Learning (RL). Recent works have surfaced mentioning new types of Machine Learning, in particular self-supervised learning, such is the case of Kolesnikov et al.[36], but those were considered out of scope. For the scope of this thesis, only classical machine learning, and particularly, supervised learning, is going to be addressed. Nonetheless, a brief overview of other categories will be given.

3.1.1 Supervised Learning

On Supervised Learning, the data used must contain a corresponding label indicating the desired output of the model. A common use case is, the input can be an image and the label what that image contains, like fish, cat, turtle, or fruit, for example. The most two common approaches for Supervised Learning are Classification and Regression.

For Classification, the model divides the data into a discrete number of classes(labels). Given a dataset of animals, the model will try and place each image in one of the classes known to the model.

On the other hand, Regression is used when the output is a numeric value. A practical example of regression in the scope of this thesis is to try to predict the number of nutrients (g/kg) based on aerial images of crops.

Artificial Neural Network (ANN) are mathematical models that resemble brain functioning. Each ANN has several layers of neurons, each of those layers has several neurons, and each neuron has a weight. The output of each neuron is composed by a determined function in which the parameters are the output of the previous layer. The model then arranges the weights of the neurons until it manages to

reach a determined output. For the most part, ANNs are included in Supervised Learning. According to the literature, there is a significant amount of different implementations of ANN, such are the case of Long Short Term Memory (LSTM)[37], Recurrent Neural Network (RNN)[38] and autoencoders [39]. It is important to note that these implementations are based on Deep Learning Networks, which in turn are specifications of ANN. A Deep Learning network can be simply characterized as an ANN with a large number of hidden layers.

3.1.2 Unsupervised Learning

In Unsupervised Learning, the best-known technique is clustering, no labeling is required. The algorithm tries to aggregate the similarities or patterns found in the records. The researcher/user has the responsibility to label the automatic aggregation provided by the algorithm.

3.1.3 Reinforcement Learning

RL is a paradigm of Machine Learning that studies the outcome of the exploration of the universe based on weights, or rather rewards. In this sense, an intelligent agent explores the universe. The actions it takes will receive a reward or not and the agent tries to maximize the reward by exploring arrays of possibilities, in a number of iterations/epochs. It should be noted that RL is out of scope for this thesis.

3.2 Machine Learning and Satellite Data in Soil Sensing

In the past years, various works in the field of Agriculture using machine learning have surfaced. Plant phenology [14], disease forecast [14], crop yield forecast [13], crop type mapping [40, 41], biomass estimation [42], soil moisture monitoring,[43, 44], and soil nutrient prediction [15, 45] are fields which are increasingly getting more relevant.

Since the focus of this thesis is on soil sensing applications, the related works will be shown in more detail.

A number of the related works are soil moisture. It is a crucial aspect in crop growing, given that ideal moisture conditions lead to fewer pests and optimal growth. In Efremova et al.[43] authors proposed a system to be integrated into a decision support tool for farmers, which they could visually identify the moisture of the soil. The input of the system is based on Sentinel 1 and 2 data along with groundtruth data to verify the model performance. Then, RNN and LSTM networks combined with encoder-decoder networks were used, resulting in a trained model, which was able to identify the dampness of the soil. The output of the model was false-color images that indicate whether the location is dry (pointed with red) or wet (pointed with blue) as shown in Figure 3.1. The system proved useful, given that the system works quasi-real-time (limited by the satellites' revisit time) and drastically decreased previous costs of in-soil sensors.

Another contribution in soil moisture estimation is present in Gao et al.[46], where authors were able to predict the soil dampness variations with errors of around 0.070 m^3/m^3, which were considered usable for most use cases in agriculture. The authors used over one year of *Sentinel 1* C-SAR instrument data in two zones of Spain with two different change detection methods. The first method was using the radar signal change detection technique, which has previously shown good results in Zribi et al, 2008. [47]. The second method uses the difference between the back-scattered[1] signals observed on two consecutive days. Considering that the temporal gap is small, the roughness of the land and vegetation

[1] The SAR is an active remote sensing instrument, meaning that it emits microwave signals and captures the response.

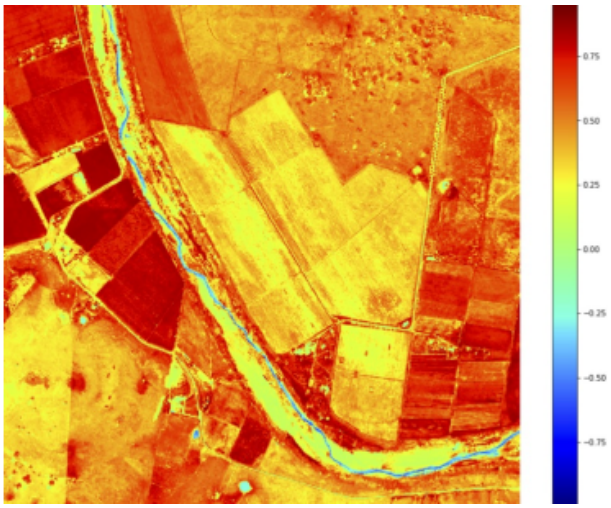


Figure 3.1: Output of the work in Efremova et al, 2019. [43], dry soil is marked with red and wet soil with blue.

are constant meaning that the only variation is the soil moisture, i.e. the authors assumed that the differences in images are related to soil moisture, and did not take into consideration other variables, given that two days are not enough time to have significant vegetation modifications. To validate the results, soil moisture measurements were collected from the fields. The results show that the values obtained by the two methods were very similar. The greater differences were noted in high vegetation density, authors justify that vegetation can significantly attenuate the signal, thus leading to higher errors in soil moisture estimation. Although the method results were similar, the second method is more robust, given that it uses a low temporal gap in order to measure the soil humidity, while the first method uses the lowest value in each pixel, which is potentially prone to errors under extreme conditions.

Soil organic carbon (SOC) is one of the main causes of soil degradation in croplands. Additionally, SOC stock is one of the three sub-indicators available on the Sustainable Development Goal (SDG) 15.3.1 defining the proportion of land that is degraded over total land area[2]. The Voluntary Guidelines for Sustainable Soil Management published by Intergovernmental Technical Panel on Soils [3] also mentions the loss of SOC as one of the main causes of soil degradation. Therefore monitoring and estimating SOC contents through satellite imagery poses a great opportunity.

The authors in Castaldi et al, 2019. [48] proposed the use of the *Sentinel 2* data along with the LUCAS topsoil database, for SOC estimation, by building SOC prediction models and comparing the results with LUCAS samples. One of the challenges found was that there were many disturbing factors, such as vegetation, residues, roughness, and soil moisture. An area in Germany was chosen for the study, where 35 samples from the database were available. The results obtained in the laboratory setting showed an acceptable correlation with bands similar to *Sentinel 2*. However, when applied to actual data, the accuracy was unsatisfactory. Authors justify this due to the most important bands (B11 and B12) having a lower spatial resolution of 20 m, and disturbance of other spectral signals (mineralogy, soil moisture, or vegetation cover).

Soil nutrients also do have a great role in agricultural productivity. Remote sensing nutrients proves a challenge since there is no evidence of any linear correlations between multispectral bands and soil nutrients, meaning that Machine Learning methods can be used to overcome/mitigate that challenge.

Researchers in Song et al, 2018. [45] estimated the levels of Nitrogen (N), Phosphorus (P), Potas-

[2]see more about the Sustainable Development Goal in https://knowledge.unccd.int/knowledge-products-and-pillars/ldn-monitoring/sdg-indicator-1531

[3]see more about the Voluntary Guidelines for Sustainable Soil Management in http://www.fao.org/3/a-bl813e.pdf

sium (K) which correspond to the macronutrients in the soil because they are tied with the nutrient cycling for crop growth. Scenes from the Chinese Environmental 1A hyperspectral [49] satellite used, which has 115 spectral bands. To serve as groundtruth,1297 soil samples were collected. 324 of them were used for testing the remnant was used for algorithm training. The machine learning methods used were Stepwise Linear Regression (SLR), Support Vector Machines (SVM)[50], Random Forest (RF), Neural Networks (NN) and Hybrid Kriging Model (BPNNOK).

The results of the paper indicate that there is a weak linear relationship between soil nutrients and hyperspectral variables, but also has significant nonlinear and multi-directional characteristics. The SLR model hardly fits the complex relationship between soil nutrients and Hyperspectral variables. The SVM had low predictive accuracy, although authors justify this by the fact that were only used the visible spectrum bands as input for the SVM. The RF model struggled, given that with the growth of the trees and learning increments, the computational efficiency drastically decreased. The NN used complex weight calculation along with hidden layers and proved a strong ability to fit the non-linearity between hyperspectral variables and soil nutrients. For NPK values, it provided an accuracy of 44.24%, 42.91%, and 48.53% respectively. The BPNNOK significantly improved the predictive accuracy between hyperspectral variables and soil nutrients, since the results obtained for NPK values provided an accuracy of 68.51%, 69.30%, and 70.55%, respectively.

Authors conclude that the adsorption mode of soil nutrients in farmlands is divided into two types: external adsorption, which occurs on the surface of soil organic matter, and relatively stable internal adsorption, which is usually adsorbed on the soil minerals. Meaning that the spectral absorption of soil nutrients varies greatly at different wavelengths. This presents a challenge for multispectral bands. With a hyperspectral instrument, such as the one authors used, it is possible to sweep several continuous bands and, in this way, estimate soil nutrient count. In this sense, this approach proves that soil nutrient estimation is possible using hyperspectral scenes as input.

Another aspect of agriculture that has a significant impact is soil contamination. According to Naveedullah et al.[51], the contamination of agricultural soil by toxic elements(such as heavy metals) is of great importance considering that metals will build up in the soil and be consequently absorbed by crops, potentially posing a significant threat to human health. Researchers in [52] used 300 ground samples along with Landsat 8 satellite imagery to estimate the distribution of Arsenic (As), Chromium (Cr), Nickel (Ni), Copper (Cu), Lead (Pb) and Zinc (Zn) in Qatari soils. The data was processed using Cubist[4]. The results obtained were that Cu had the highest accuracy with 74%, followed by As with 60%, Pb with 53%, Cr with 53%, Zn with 51%, and Ni with 47%. Similar results can be achieved with *Sentinel 2*, given that the bands used in Landsat 8 are also similar to those available with *Sentinel 2*. Should be noted that, since the results given were done on Qatar, an arid zone, results may vary for different soil types as discussed by [53].

Soil salinity is a widespread environmental hazard, and one of the main causes of land degradation and desertification. According to the United Nations' Food and Agriculture Organization, 397 million hectares of land around the world are affected by salinity and spreads with a rate of up to 2 million hectares each year [5]. In agriculture, it has a substantial negative impact on soil fertility and agricultural production. The first step in mitigating this problem is the monitoring and estimation of the salt contents on fields. Naturally, satellite imagery can help in that regard. Many efforts have been done to the monitorization of soil salinity. For example, in Khan et al.[54], studies were conducted in the development of Normalized Difference Saline Index (NDSI), and Saline Index (SI). Others have found that the NDVI is an indirect indicator of soil salinity, given that most crops do not grow with a certain level of soil

[4]Cubist is a powerful data-mining tool that has been widely used in spatial modeling and mapping of a range of soil attributes in recent years: https://www.rulequest.com/cubist-info.html

[5]see http://www.fao.org/3/i0372e/i0372e.pdf

salinity[55].

Taghadosi et al. [56] from Iran, tried to use Land Surface Temperature (LST) and Digital Elevation Mapping (DEM), but found no correlation to soil salinity, so they used Soil Electrical Conductivity[6] to estimate soil salinity contents, over the providence of Qom. This study showed the potential of the Sentinel 2 and Regression techniques in estimating the potential soil salinity over that area. To train and test the algorithms, ground samples were used. Results show that Support Vector Regression (SVR) showed potential in salinity detection, with an accuracy value of 83% using the test set. An overview of the study areas that were discussed is provided in Table 3.1.

Table 3.1: Brief conclusion of the studies presented in the State of the Art

Area of Study	Findings
Soil Moisture Estimation	High correlation with ground samples [43]
Soil Nutrient Estimation	Achieved significant accuracy with hyperspectral imagery [52]
Soil Organic Carbon Estimation	Weak correlation, many factors can disturb the results [48]
Soil Contamination Estimation	Achieved significant accuracy [52]
Soil Salinity Estimation	Found indirect correlations with other indicators [54, 56]

3.3 Publicly Available Datasets

There is yet to be provided an official definition of a dataset in the field of earth observation. However, in a broad sense, a dataset is a collection of related sets of data that are composed of separate elements but can be manipulated as a unit by a computer [57]. In this thesis, a dataset is defined as *an aggregation of satellite data, either raster or vector, with the option of having ground truth annotations, such as in-situ measurements, spectral information, or simply ground truth data*. In other words, a collection of data that can be used to train and evaluate the performance of ML algorithms.

Given the increasing interest in the EO community, various datasets exist, each focused on a specific area. A comprehensive list of datasets can be found in Awesome Sattelite Imagery Datasets[7]. As of the date of this thesis, the list is comprised of around 80 datasets, split into use cases, namely: instance segmentation, object detection, semantic segmentation, scene classification, and Multiple Tasks. Although these are important, very few were related to the scope of this thesis.

Even though a large number of datasets was compiled, only four of these datasets are related to agriculture: 1) PASTIS [8], 2) Agriculture vision [58], 3) LPIS agricultural field boudaries[9] and 4) Agricultural Crop Cover Classification [10]. These contain agricultural parcels from around the world with labels for which kind of crop is growing, these datasets count with over 1 million images, from different data sources such as Unmanned Aerial Vehicle i.e. Drone (UAV), helicopter and satellite data (provinient of Sentinel 1,2 and Landsat 8).

iSAID [59] is an image dataset with several annotated objects. It has potential for image classification algorithms since it counts more than 500 000 annotations for different objects such as planes, ships, storage tanks, baseball diamonds, tennis courts, basketball courts, ground track fields, harbors, bridges,

[6] Higher salinity levels are directly related to higher electrical conductivity
[7] List of datasets: https://github.com/chrieke/awesome-satellite-imagery-datasets
[8] PASTIS dataset, available at: https://github.com/VSainteuf/pastis-benchmark
[9] LPIS dataset, available at: https://collections.eurodatacube.com/denmark-lpis/readme.html
[10] Agricultural Crop Cover Classification Challenge dataset, available at: https://www.crowdanalytix.com/contests/agricultural-crop-cover-classification-challenge

vehicles, helicopters, roundabouts, soccer fields, and swimming pools. The dataset is composed by satellite images along with vector coordinates of the annotated objects. More work similar to iSAID exists, for instance in Yang et al, 2010. [60, 61].

Although it was not possible to find any dedicated soil sensing datasets with associated images, the LUCAS [62], an effort of European Soil Data Centre (ESDAC), is a dataset of multiple soil surveys done throughout the years. It currently holds around 250 000 collected samples, where each sample holds records of soil properties, such as spectral/chemical/physical properties of the soil and pointers of soil degradation. Table 6.1 contains a comprehensive list of the fields recorded by LUCAS. It is important to note that the LUCAS project is not a dataset *per se*, nonetheless, it can be used as such when coupled with aerial imagery for the same region and date, as seen in [48]. On a further note, the authors of Lucas Copernicus[63] measured a polygon of up to 51 meters in all four directions, against the LUCAS dataset points. That polygon indicates the soil inside it can be used with the soil analysis in the dataset. Although the samples only account for a single point, when aggregated with LUCAS Copernicus, it was assumed the whole polygon contains the same nutrient levels as measured in the sample. This assumption should be taken into consideration.

3.4 Development Options

With the increased traction of Earth Observation, several libraries are being developed as a community effort, mostly for the Python programming language, given it is a fast prototyping language, with little overhead. In addition, several libraries for ML exist for this ecosystem, posing as one of the best ecosystems for that kind of work.

As it was previously mentioned in Section 1.1, despite having a vast number of works, most of the existing works are not reproducible given that no source code is available or the necessary steps are not properly documented, and the data used comes from several different sources. In the interest of mitigating such problems, several tools have been developed by the research community. For that reason, this chapter shows few pieces of software, namely Libraries, Frameworks, and Toolkits, that are developed specifically for Earth Observation or have the potential to simplify the workflow of Earth Observation.

3.4.1 Libraries

This section presents libraries that are used in the Geospatial field. Additionally, other libraries aid the process of performing machine learning. It is noteworthy to know that a library is a distributable piece of software that simplifies a determined process.

Rasterio [11] is a raster image visualizer, that acts as a wrapper for GDAL, the Geospatial Data Abstraction Library written in C, and its main purpose is to visualize, annotate and save datasets in the GeoTIFF format.

GeoPandas [64] is a map projection visualizer, that allows for making map projections and earth data representation in an easy manner. GeoPandas is an extension of the Pandas data analysis and manipulation tool built on top of the Python programming language[12].

Geospatial-learn[13] is a library that contains a handful of functions commonly used in EO, such as image processing, vector processing for feature extraction, machine learning model creation, and plotting.

[11] Rasterio library, available at https://rasterio.readthedocs.io/en/latest/
[12] The pandas project: https://pandas.pydata.org/
[13] Geospatial-learn, see https://github.com/Ciaran1981/geospatial-learn

EarthPy is another library, based on Rasterio and Geopandas. It provides a small set of features that researchers found the most useful, namely rendering false-color images and other composites based on the available bands, plot drawings, and masking irrelevant data.

For the actual process of performing machine learning, additional libraries are necessary to ensure optimal results, reproducibility, and storage of models, as are the cases of is the case of Optuna and MLFlow.

Optuna [65] is a library developed to perform automatic hyperparameter optimization in order to find the best hyperparameters for a determined model and dataset. The functioning of Optuna is platform-agnostic, meaning that it can optimize any model regardless of the ML framework. To optimize a model, the user has to come up with an objective function, which contains the parameters of the models and which values the hyper-parameters can vary between. Given that the number of combinations of parameters grows exponentially, Optuna minimizes that risk by using Bayesian Optimization, instead of performing an exhaustive search/brute force, drastically reducing the search space by doing early stops when the metric gets worse or similar results.

MLFlow [66] is a python toolkit that keeps track of the machine learning experiments. Additionally, it provides automatic record tracking for most ML libraries, but any model can be logged manually. Meaning it is also platform agnostic and provides a good Graphical User Interface (GUI) where users can explore past experiments and compare the results obtained. According to [67], it is a must-have library because 1) it focuses mainly on the reproducibility aspect of machine learning, which was identified as a challenge, and 2) allows for comparisons between several iterations of an experiment.

3.4.2 Frameworks

In computer science, a Framework is a piece of software that defines a structure for working on a solution for a problem, that simplifies the process of development. According to CocoSolution[14], a framework is a kind of template, scheme, or technology-based conceptual structure that allows working at a greater abstraction level.

Researchers in Stoneback et al, 2018. [68] proposed the *pySat* framework, a python satellite data analysis toolkit that implements a generalized process for satellite data analysis. More concretely, it provides all the steps necessary to produce a data product (such as downloading, loading, and pre-processing data), along with analysis tools that are data source and instrument-independent. In other words, it is a very useful framework, which is given emphasis once future missions such as COSMIC [15] and ICON [16] are already incorporating this framework in the pre-processing of the collected data.

Rastervision[17] is an open-source framework for computer vision models intended for satellite, drone, and other imagery observation, meaning that it works with EO particularities, as opposed to regular computer vision libraries, for instance, usage of more than three channels(RGB), extremely large scenes, and GeoTIFF/GeoJSON ready. Additionally, it allows for quick deployment of pipelines for the most frequent workflow in machine learning, which are, analyzing training data, creating training groups, training models, creating predictions, evaluating models, bundling the model files, and deployment.

Open Data Cube (ODC) is an open-source framework for accessing, managing, and analyzing large quantities of Geographic Information System (GIS), in particular EO. It uses PostgreSQL and Redis. According to its web-page[18], the Open Data Cube system is designed to:

[14] CocoSolution https://cocosolution.com/en/what-is-a-framework/

[15] see more about the COSMIC mission: https://directory.eoportal.org/web/eoportal/satellite-missions/content/-/article/formosat-7

[16] see more about ICON mission https://www.nasa.gov/icon

[17] Rastervision project, accessible at https://docs.rastervision.io/

[18] see https://www.opendatacube.org/overview

- Catalogue large amounts of Earth Observation data
- Provide a Python-based API for high-performance querying and data access
- Give scientists and other users easy ability to perform Exploratory Data Analysis
- Allow scalable continent-scale processing of the stored data
- Track the provenance of all the contained data to allow for quality control and updates

eo-learn[19] is an effort of the Sentinel Hub company aiming at providing a set of libraries to make prototyping of complex EO workflows as easy, fast, and accessible as possible [69]. Its main features pass by input and output of images, download of sentinel-hub services, masking of cloud and irrelevant data, the transformation of vector and raster data, and Image Co-registration[20], feature extraction and manipulation. It is also ML-ready, meaning it contains tools for classification, training, testing, and validation. It is worth noting that eo-learn is an open-source initiative, and one can extend its functionality freely.

3.4.3 Toolkits

A toolkit is a bundle of tools for performing a determined task but does not define a mandatory structure, unlike a library.

Solaris[21] is a python toolkit aiming to bridge the gap between machine learning and EO, and the reproducibility of results. It proposes a unified data format, prepares the data in a standardized manner, trains computer vision models, generates a prediction on EO data using common Deep Learning (DL) frameworks, and performs calculations using relevant metrics. The main focus of Solaris, however, is geospatial computer vision, more concretely, identification of foundational objects, i.e. roads and buildings. The general output of this toolkit is a mask with predictions of roads and buildings. Solaris contains implementations of award-winning algorithms. As a matter of fact, the company that composed Solaris, CosmiqWorks [22], has developed over 20 projects applied to Geospatial Research. A few of them will be presented:

- SPACENET: There are several iterations of SPACENET from 1 to 7. In the first iterations, they developed algorithms capable of detecting buildings, roads, and building analysis. Finally, in the last iteration, it tracks the development of cities across the years. The dataset that was used was imagery provided by the Planet mosaic dataset[23].
- CRESI: Road network identification and extraction of route traveling time.
- RAREPLANES: Detects models of aircraft that are parked in the runways.

3.5 Summary

Overall, the surveyed research studies show the limitations of remote sensing technology, particularly the spectral resolution. Other studies discuss the entropy introduced by foreign variables such as mineralogy and other reflectance variables that were not covered. On a further note, most of the models used did not

[19] see https://eo-learn.readthedocs.io/
[20] Image co-registration is the process of geometrically aligning two or more images so that corresponding pixels representing the same objects may be integrated or fused.[70]
[21] see https://solaris.readthedocs.io/
[22] CosmiqWorks projects: https://www.cosmiqworks.org/category/archived-projects/
[23] Planet mosaic dataset, see: https://www.planet.com/products/basemap/

achieve desirable results, meanwhile, hybrid approaches presented good results(error rounding 30%). The major problems that were identified in these studies, are that 1) the authors do not make their code available and 2) most of the studies gather their own datasets and do not release them to the public. These 2 challenges lead to reproducibility issues, resulting in a weak contribution to the field in the sense that no further comparisons can be made with existing works.

Various open-source earth observation datasets were described, but none of them fit the purposes of this thesis, hence it was opted to create datasets in the scope of this thesis, comprised of Sentinel-2 imagery over Europe and soil information gathered by the LUCAS dataset [2, 62, 63].

Several software tools have emerged to aid the processing and usage of remote sensing(GeoPandas, Rasterio, and eo-learn). After a broad analysis, it was found that these libraries complete each other, and the toolkit in question will rely on these three libraries. The Solaris toolkit was also described but was deemed out of scope for the thesis.

Chapter 4

Proposed Solutions

Against this background, TerraSenseTK is proposed, an EO Toolkit aimed at reducing the learning curve of the field entry, by providing a ”turnkey” experiment and API. The toolkit is also focused on reducing the two reproducibility challenges that were raised by providing a reproducible nutrient estimation pipeline ready for different experiments, and datasets disregarding the data source(i.e, any form of imagery can be used, either drone or satellite).

This chapter presents the two contributions of the thesis: the first one is the creation of TSTK and the second contribution is the Soil Nutrient estimation pipeline.

4.1 TerraSenseTK

Given the challenges aforementioned, the aim of the toolkit is to allow newcome users to perform geospatial soil sensing experiments. Tackling in this way the underlying problem of difficult barrier to entry. More concretely, the toolkit must be capable of reading a dataset, indicating the features the user wants to predict from the available information, and measuring the estimation capacity of the desired output features. Resulting in a trained model allowing the user to estimate similar data in the future.

The toolkit is split into independent modules: the TSTK Core(which contains the Dataset, Algorithms, and Performance) and the Experiment. Though the actual process is made on the Experiment module, the heavy lifting is done by the TSTK Core. The thought behind this separation of concerns arose from the need for this toolkit to be as versatile and extensible as possible, meaning that if a proficient user wants to implement new functionalities, such as new datasources, datasets, algorithms, metrics, or even experiments, they can simply implement it by using the provided abstract classes, without needing to change the codebase.

In order to establish the functioning of the toolkit, the diagram in Figure 4.1 was conceptualized. The TSTK Core is based on 3 modules, namely Dataset, Algorithms, and Performance and its organization is as follows:

4.1.1 Datasets

The Dataset module is responsible for all the data manipulation tasks, such as downloading/creating images for a determined region, adding spectral indices in order to obtain better correlations with the underlying data, masking data that is outside of the region of interest, the associating of ground truth to the images, and processing the images into data ready for the machine learning algorithms. The importance of this module is given to the fact that by keeping all the data aspects to one module, the researchers can focus only on the development of the algorithm. It also provides a common ground

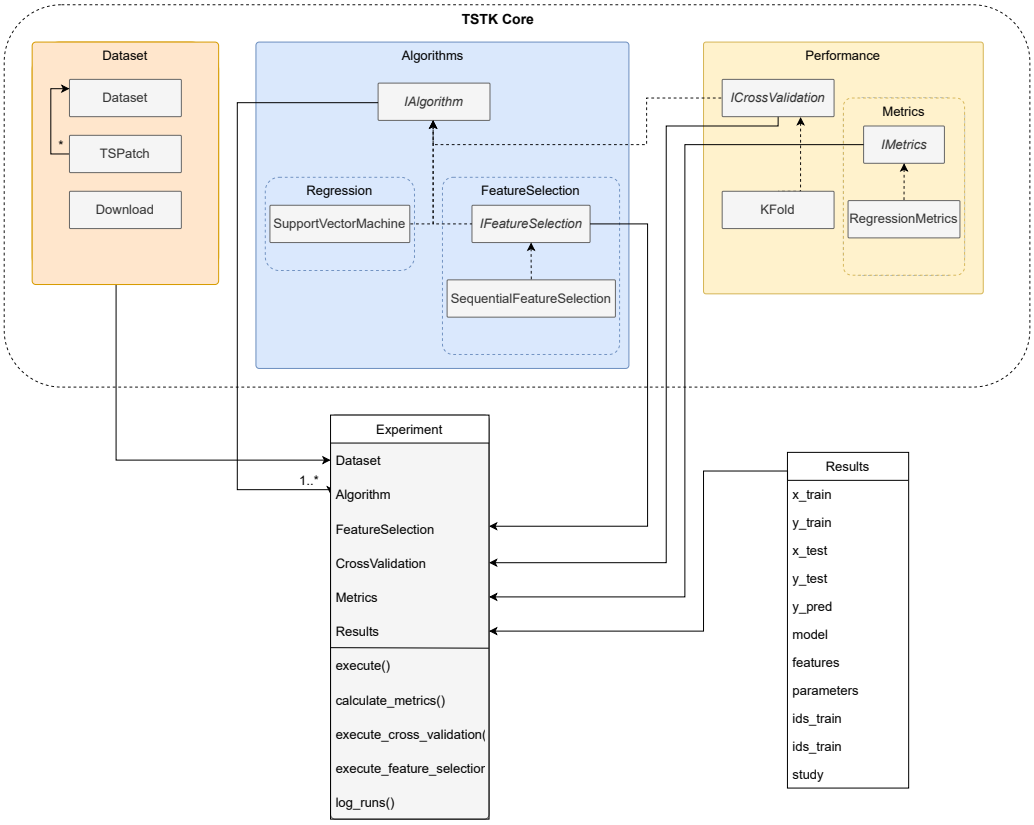


Figure 4.1: TerraSenseTK core diagram

for other researchers to share their datasets in a common format, which contributes by providing the distribution of datasets. Being this module independent from the rest of the toolkit it is even possible to use the toolkit only for producing a standardized dataset.

4.1.2 Algorithms

In the Algorithm module, it was opted to separate algorithms by their concept. In this sense, this module contains two submodules for now, which are Regression and Feature Selection, as well as an interface to support the development of new algorithms.

Feature Selection

The Feature Selection module contains interfaces for the Feature Selection algorithms. Feature Selection is a crucial aspect to take into consideration when performing machine learning, considering that using features that do not add any value to the problem in question, can potentially increase the error of estimation and the time needed to train the model as mentioned in Agrawal et al, 2021. [71]. For that reason, Feature Selection is performed where redundant features are removed automatically.

Hyper-parameter Optimization

It is noteworthy to mention that the process for implementing Hyper Parameter Optimization (HPO) is contained inside the algorithm module. For instance, if the user wants to implement HPO in its algorithm, it must implement a simple objective function that receives the parameters of the model and

the universe of said parameters, eventually returning a metric value for which the optimization library will try to minimize.

4.1.3 Performance Evaluation

The Performance module contains the structures needed to assess the model's performance. It is comprised of the performance metrics and cross-validation module.

Within the performance module, there is a submodule dedicated to metrics, the main focus is to allow, in a simple manner, the implementation of new metrics by a more experienced user. The addition of metrics is implemented in an easily extendable manner: a new method is created in the Metrics class, assuming the arguments are the predicted values and the ground-truth values. After running the experience, users specify which of the metrics they want to use from the list including their own added metric, and when finalizing an experience, the results for each metric get returned. It should be added that a number of metrics are included in the toolkit which is ready to use for the end-users.

Cross Validation (CV) is another very important aspect of ML, with CV, the data is split into several groups in various iterations. The model is then trained several times, ensuring the model is properly fitted (i.e it is not overfitted or biased). Performing CV is optional, given the fact that it is particularly relevant when the datasets do not contain enough data to train a model, as it allows to train many models using the same data. When a dataset contains enough data and CV is not necessary, the experiment can be performed using a train-test split. It should be noted that the user is responsible for selecting either cross-validation or train-test split.

4.1.4 Experiment

The Experiment Module is responsible for the execution of a ML experiment. To execute it, the user provides the dataset to be used, which models are used for fitting, whether it wants to perform feature selection, cross-validation, hyperparameter optimization, and which metrics the user wants to be reported back. The experiment implements the nutrient estimation pipeline presented in Figure 4.3. Although the main focus is the estimation of nutrients, it is possible to perform any kind of regression with the data provided in the dataset.

The more detailed pipeline is present in Figure 4.2, as well as its relation to the modules present in the diagram in Figure 4.1. A brief overview of the experimental procedure is provided below. All the downloading and processing of images/metadata tasks are done in the Dataset module. The feature selection is done automatically by an algorithm of the end-users choice, and the feature selection module is included in the algorithms package. When no Feature Selection is supplied, all of the bands/indices that exist in the dataset are used. The Feature Selection is included in the Algorithms module, along with the Machine Learning algorithms used in the train/test workflow. The CV is also optional. After performing the cross-validation, the several folds get trained and tested with an algorithm chosen by the user, the toolkit then returns for each fold the results for each metric chosen. In a similar manner, if the CV is not supplied, then a train/test split of 70/30 will be performed. With the training concluded, the users are also free to choose which metrics they deem fit for the problem at hand. Both the cross-validation and the metrics are submodules of the performance module.

4.2 Soil Nutrient Estimation Pipeline

The second research contribution of this thesis is the development and evaluation of an ML pipeline to perform soil nutrient estimation. Remote Soil nutrient estimation is based on the bands available on

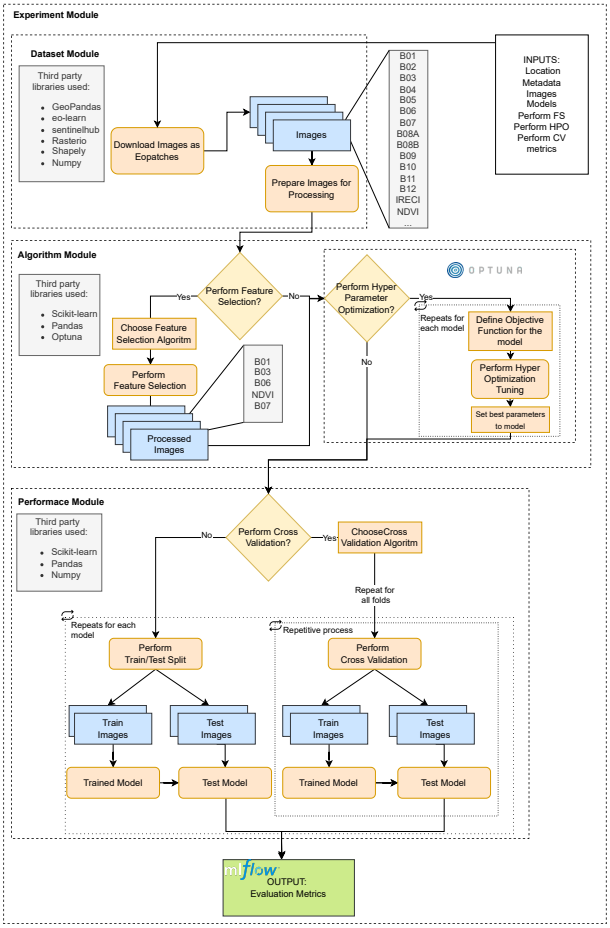


Figure 4.2: Earth Observation Pipeline

aerial/satellite imagery. In light of this, the goal is to estimate the nutrients in the soil. Given that the field of remote soil sensing counts a small number of works, the goal is to assess the possibility to use, in this case, Sentinel-2 to perform Nutrient Estimation.

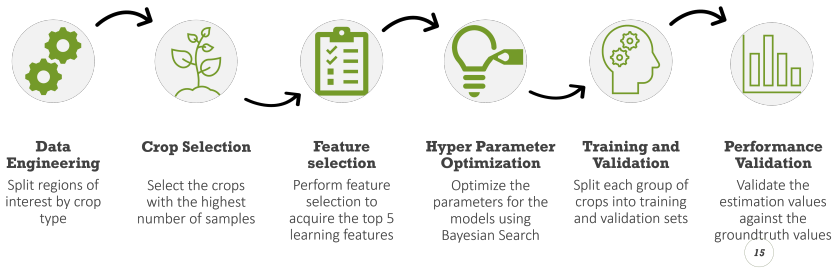


Figure 4.3: Pipeline for nutrient estimation.

To achieve nutrient estimation, a pipeline was developed to streamline the workflow. In a broader perspective, this pipeline is represented in Figure 4.3.

1. In the first part, i.e, Data Engineering, revolves around the acquisition and processing of images(such as downloading images, an association of ground-truth, masking the Regions of Interest (ROI)) in order to exclude irrelevant information. The process of Data Engineering is summarized in the Figure 4.4. In this example, the dataset portion of Portugal was taken, then the polygon that contains data from the dataset gets selected. Only the area in which there is groundtruth is

selected, so most of the polygon area is discarded.

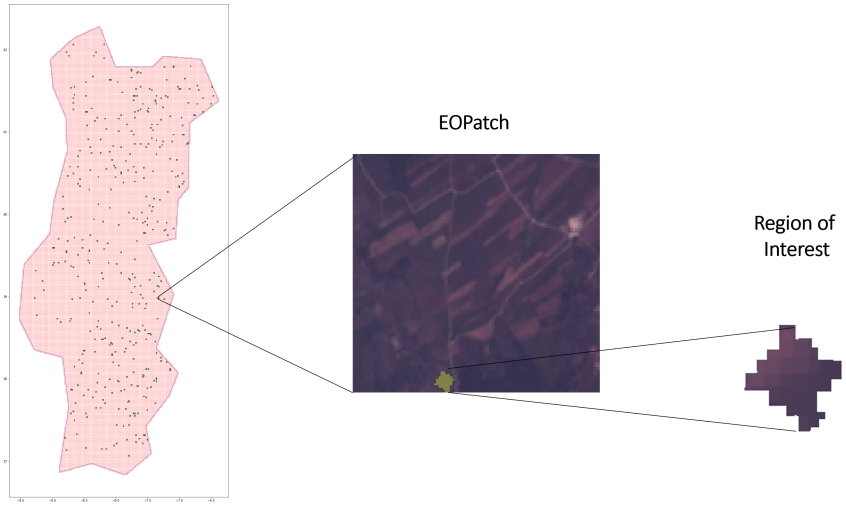


Figure 4.4: Process of data acquisition

2. The second stage is crop selection. In this step, the existing crops in the dataset are analyzed and the crops with the most occurrences are selected. Given the physiognomy and phenology of crops, it was understood that crops do not all reflect the energy in the same spectral bands, so in order to improve the results for the models, it was opted to segment the data per crop.

3. The third step is the feature selection process, which was tested manually with the correlation between the nutrient and the bands and automatically by discarding the bands that delivered roughly the same information. Given that the results of both methods were similar, the automatic feature selection method was opted for. The automatic feature selection is performed by using a Feature Selection algorithm. The type of operation depends on the algorithm selected. For example, the Sequential Feature Selection runs several stages. At each stage, it selects the best feature to remove based on a cross-validation score of a chosen estimator. It should be noted that this Feature Selection Algorithm chooses the best features for a specific estimator, for this reason, great care should be taken when using Sequential Feature Selection.

4. The fourth step is Hyper Parameter Optimization, which is an approach to test and improve the results of the chosen models. This is done by providing a universe of parameters in which the optimizer should look for the best set of parameters based on a user-defined metric. After performing the optimization, the set of parameters that provided the best results will be returned. Granting in this way, the best parameters for the provided data and model.

5. The fifth step is to perform cross-validation or train test split to validate the performance of the trained model. When training a model, one must perform train and test sets to ensure the model's capability of estimation. The rationale behind this method is to reduce the overfitting of the model, i.e. learning really well a set of values. When a model overfits, the future estimations will provide very poor results. In this way, it is ensured that a part of the dataset is saved for testing, meaning new inputs for the model, that it has never seen. Cross-validation is an even more robust measure to reduce overfitting, where the data is split into an arbitrary number of sets, then it is performed training and testing over all the sets created.

6. Lastly, the performance step is the comparison of the results obtained against the groundtruth to obtain a score that can be measurable, i.e., an evaluation metric.

Chapter 5

TerraSenseTK Implementation

This chapter explains in detail the implementation of the toolkit and is divided as follows: Firstly, a brief overview of the structure of the toolkit is given. Then each module is presented in detail, namely, how the dataset module is built. Then, the intent for the algorithms module is explained as well as the algorithms that were implemented with the toolkit. Additionally, the performance module is explained (which contains the implemented metrics) as well as the Experiment module, which implements the nutrient estimation pipeline is described. Lastly, the Utils module is shown, which is where the reusable functions of the toolkit are defined.

5.1 TerraSenseTK Overview

For the reasons stated before in Chapter 3, python was selected as the implementation language. The TerraSenseTK follows a structure of a typical python library (see Figure 5.1). Each folder contains a __init__.py file that allows the python interpreter to recognize that folder as a module. The project contains five modules: Dataset, Algorithm, Performance, Experiment, and Utils. The latter is auxiliary.

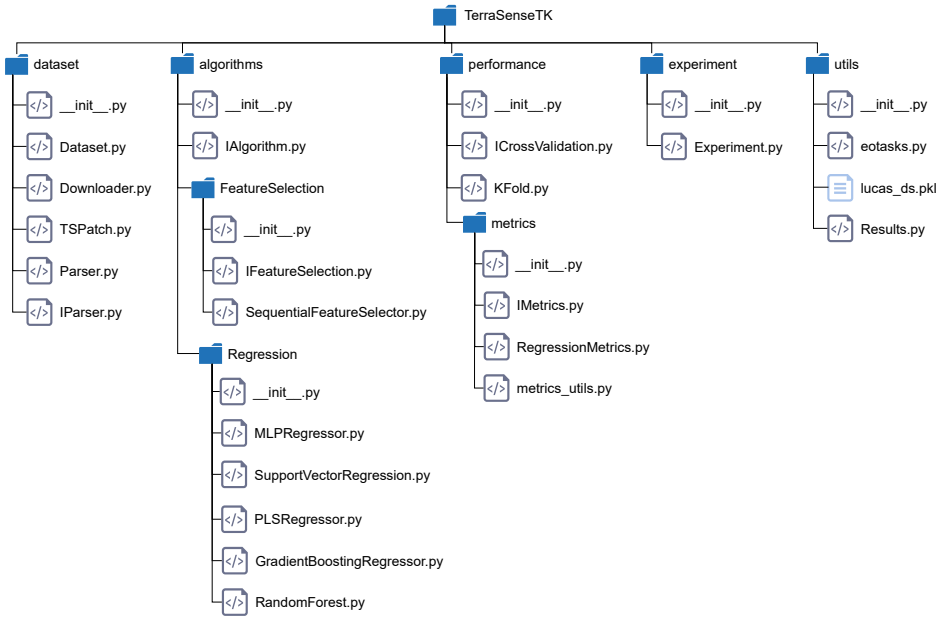


Figure 5.1: TerraSenseTK folder structure.

5.2 Dataset Module

In the dataset module, there are 3 classes: the Dataset, Downloader and TSPatch. The structure and relations of this module are visible in the Figure 5.2. The dataset is the class responsible for downloading images, through the Downloader, adding indices to the dataset imagery, and, most importantly, the representation of the dataset. The dataset class is comprised of several TSPatches.

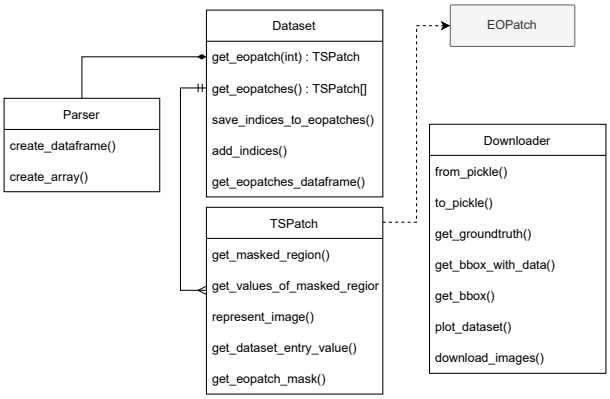


Figure 5.2: Dataset module entity relationship diagram

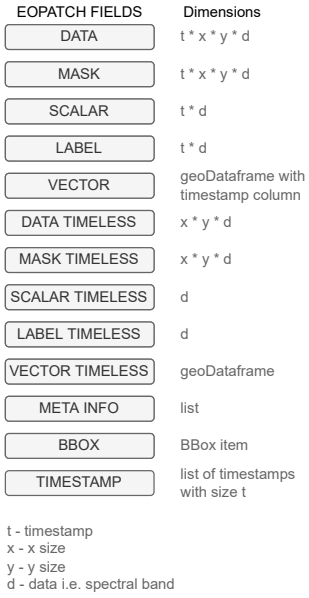


Figure 5.3: Data model of an EOPatch

5.2.1 TSPatch

The TSPatch is a specialization of the EOPatch object from eo-learn (see Figure 5.3). This was done in order to simplify the retrieval of the information added by the TerraSenseTK. The TSPatch adds the following methods to the EOPatch: *get_masked_region*, *get_values_of_masked_region*, *represent_image*, *get_dataset_entry_value*, *get_eopatch_mask*, as seen in Figure 5.2. All these methods are used in order to translate the raster image into useful data that algorithms can comprehend. This object simplifies the acquisition of information at each point of the dataset, abstracting the difficulties to obtain information regarding the EOPatch. For instance, instead of gathering images from the EOPatch using

the EO-learn notation *eop.data["BANDS"][-1][...,[3,2,1]]*, with the TSPatch, it is done by simply calling *eop.represent_image()*. The TSPatch includes all the previous methods, to ensure backward compatibility with the eo-learn implementation

5.2.2 Downloader

The Downloader class downloads the imagery to the user's filesystem, which can then be instantiated by using the Dataset class, in the following manner: `dataset = Dataset("path\to\dataset")`. This module hides a great part of the complexity of the task of creating a dataset. The Downloader uses the SentinelHub API to download the locations that are specified in the dataset. Each collected point is referenced by GPS coordinates which are later translated to polygons in a determined Coordinate Reference System (CRS), which is the way locations are represented in SentinelHub. In addition, this module allows previewing of the location which the dataset is being collected from, although, unfortunately, this visualization is limited by country, for memory constraints reasons.

The API allows for downloading imagery from all the Copernicus program satellites and Landsat 5 and 7.

5.2.3 Dataset

The dataset class has 5 principal functions, which are *get_eopatch()*, *get_eopatches()*, *add_index()*, *save_indices_to_eopatches()*, *get_eopatches_dataframe()*. The *get_eopatches* function returns all the TSPatches in the dataset, while the *get_eopatch()* returns a single EOPatch from the dataset. The *add_indices* method is how the user can add new indices to the dataset, by indicating which is the name of the index and the formula for it in a simple notation. For example `add_index("NDVI","(B04-B05)/(B04+B05)")`, then the user can run the *save_indices_to_eopatches()* and the indices added previously will be written in the dataset, for all TSPatches.

5.2.4 Parser

The Parser class is responsible for transforming the imagery into information that the algorithms can actually understand. The current implementation of Parser transforms the imagery data which is saved in a 2D array into several 1D arrays(one for each spectral band/index). For that, it is mandatory to extract the ROI and flatten the data for each sensor.

The process to parse the data is illustrated in Figure 5.4. First, the selected EOPatches images get converted to an N-feature dimensional array (i.e. the X). Additionally, for each image a 1D array with the same length is created, but, its values, instead of being the values gotten from the images, is the value of the data that we demand to fit, i.e. the Y, which for instance can be Nitrogen. Moreover, the values from X and Y for each image get all concatenated.

5.3 Machine Learning Algorithms Module

The diagram in Figure 5.5 represents the Algorithm module. This module contains two sub-modules: the regression algorithms module, which inherits directly from IAlgorithm, and the feature selection algorithms module. Each of these modules has an interface that defines the methods needed to function such as the case of *fit, predict* and *clone*. The IFeatureSelection contains two extra methods that are the *set_number_of_features* and *get_number_of_features*. These two abstract classes were added to indicate how many features we want to maintain. Gradient Boosting, Multi-Layer Perceptron, Partial

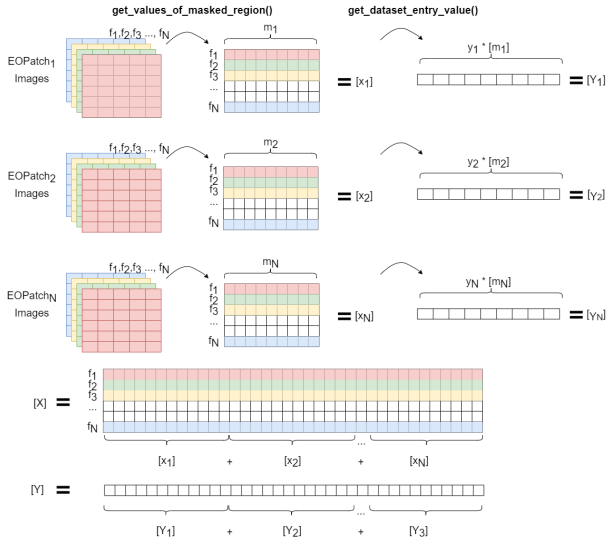


Figure 5.4: The Data Conversion Process in the Parser Class

Least Squares, Random Forest, Support Vector Regression, and Sequential Feature Selection are the baseline implementations. It is of great importance to stress that although classic machine learning algorithms were selected for implementation, more robust and advanced techniques can be used.

5.3.1 Regression Algorithms

For the Regression algorithms, six different models were implemented in order to perform comparisons between models. The implemented models are Multi-Layer Perceptron (MLP), SVR , Partial Least Squares (PLS), Gradient Boosting (GB) and RF. In addition, for Feature Selection, only an algorithm was implemented: the Sequential Feature Selector. As a demonstration, these algorithms of the sci-kit learn library was implemented by using the available TerraSenseTK algorithm interface. Since the algorithms were already implemented in the Sci-Kit Learn, the interface is used similarly as an Adapter pattern.

Multi-Layer Perceptron

The Multi-layer Perceptron[72], originally investigated by Hinton, in 1989 [73], is a basic artificial neural network. An MLP contains several layers: an input layer, an arbitrary number of hidden layers, and an output layer. Each layer gets as input the neurons of the previous layer. The neurons work by using an activation function, which is an arbitrary function that takes the inputs and provides an output. Each neuron has weights that adjust as new epochs are executed until it converges into an acceptable estimation or reaches the maximum number of epochs/iterations. The hyperparameters that were chosen to optimize the performance of this model are the solver, and the activation function.

- The solver is the function that optimizes the weights of the neurons. It can optimize between two options: Adam, which is an implementation of a stochastic gradient-based optimizer, or SGD which is a stochastic gradient descent optimizer.

- The activation is the function that controls the hidden layers. The possible actions are logistic and relu functions.

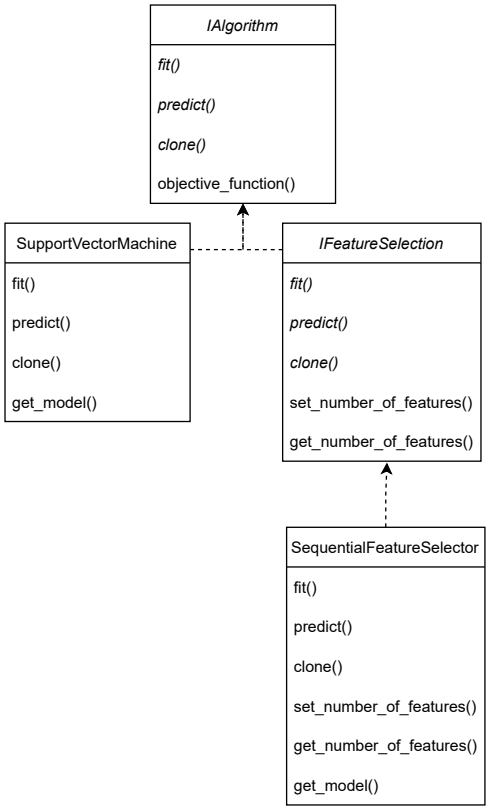


Figure 5.5: Entity Relationship diagram of the algorithm module

Support Vector Machines

The Support Vector Machine Regression [74] is considered one of the most robust prediction models. Its functioning is based on the discovery of an hyperplane. An hyperplane is an N-dimensional plane that splits the features of the classes into separate groups.

In Listing 5.1, it is possible to see the implementation of this algorithm. It is of great importance to stress that this implementation is virtually the same for all algorithms given that these are from the same library. For instance, only two functions are changed: the objective_function and the __init__ methods. For this reason and for the sake of brevity, only the implementation of the SVR algorithm is shown.

The hyperparameters chosen to optimize the performance of SVM are the kernel, c constant, and gamma.

- The kernel function is used to transform the inputs into an higher dimensional feature space to help draw the hyperplane. The possible values for the kernel are Radial Basis Function (RBF) and the Sigmoid function.
- C constant is the regularization parameter. The hyperplane must have at least the distance of c from its inputs. The value ranges between 0.1 and 3 with steps of 0.5.
- The gamma is the coefficient of the kernel which varies with the number of inputs. It has two possible values: scale and auto.

Partial Least Squares

The Partial Least Squares[75] is considered one of the simplest models [76]. It is based on Ordinary Least Squares, the mathematical approach to resolve linear regression. It is based on the Principal Com-

```python
from sklearn.svm import SVR
from ..IAlgorithm import IAlgorithm
from sklearn.base import clone as skclone
from ...performance.metrics import RegressionMetrics
class SupportVectorRegression(IAlgorithm):

    def __init__(self,args={},**kwargs):
        model = SVR
        if args:
            self.model = model(**args)
        elif kwargs:
            self.model = model(**kwargs)
        else:
            self.model = model()

    def fit(self,x_values,y_values,*args):
        return self.model.fit(x_values,y_values,*args)

    def predict(self,x_values):
        if(self.model.fit_status_ != 0):
            raise TypeError("The model is not fitted yet")
        return self.model.predict(x_values)

    def get_model(self):
        return self.model

    def clone(self):
        return SupportVectorRegression(self.get_params())

    def get_params(self):
        return self.model.get_params()

    def set_params(self,params):
        return self.model.set_params(**params)

    def objective_function(self,trial,x_train,y_train,x_test,y_test):
        metric = RegressionMetrics()
        kernel=trial.suggest_categorical('kernel',['rbf','sigmoid'])
        c=trial.suggest_float("C",0.1,3.0,step=0.5)
        gamma=trial.suggest_categorical('gamma',['auto','scale'])
        regr = SupportVectorRegression({'kernel': kernel, 'C': c, 'gamma': gamma})
        regr.fit(x_train, y_train)
        y_pred = regr.predict(x_test)
        return metric.cmd_rmse(y_test, y_pred)
```

Listing 5.1: Python implementation of the SVR algorithm

ponent Analysis method, which combines features to try to reduce the colinearity between the features. The hyperparameters chosen to optimize the Partial Least Squares was the number of components.

- The value for the number of components should be between 1 and the number of input features, so this value varies between 1 and the number of features.

Gradient Boosting

The Gradient Boosting model[77, 78] is an ensemble algorithm, meaning it recurs to other simpler algorithms to achieve better results, in this case, decision trees. This algorithm works by chaining several decision trees and minimizing the difference between the groundtruth and the predicted value. In the first iteration of Gradient Boosting, it calculates the average of the groundtruth values, then calculates the difference between each point and the average (this value is called the pseudo residual). The objective of Gradient Boosting is to reduce the pseudo residual until the estimated value is similar enough to the groundtruth value. It does this by building several trees, adding the residual values for each tree, and multiplying it by the learning rate. The learning rate varies between 0 and 1, and it is a parameter of

great importance given that it improves the accuracy of the fitting of the algorithm, normally the lower the better, until the point that the residual value does not improve.

The hyperparameters chosen to optimize the performance of Gradient Boosting are the loss function, learning rate, and the number of estimators.

- The loss function is the function that the model uses to optimize the value and the possible values are squared error, absolute error, huber, and quantile.
- The learning rate shrinks the contribution of each tree by its value. This practice helps to reduce the models' data bias. This value should be a small value, so the learning rate varies between 0.05 and 0.5 with steps of 0.05.
- The number of estimators indicates how many boosting stages to perform. The values range between 200 and 1000 estimators, with increments of 100.

Random Forest

The Random Forest [79], originally mentioned by Breiman, 2001 [80], is an ensemble model for regression and classification tasks, that calculates a large group of decision trees. It fixes the underlying data sensitivity problem of the decision trees, by creating random multiple datasets with the original dataset. For regression tasks, the estimation is calculated by the average prediction of the decision trees.

The hyperparameters chosen to optimize the performance of Gradient Boosting are the criterion, the maximum features, and the number of estimators.

- The Criterion is used to measure the quality of a split by fitting for that specific tree. The existing values are squared error and absolute error.
- The maximum number of features represents the number of features to consider when splitting the data. The possible values are the square root and the Logarithm base 2 of the number of features.
- The number of estimators indicates how trees are created. The values range between 30 and 300 estimators.

5.3.2 Feature Selection

According to HeavyAI[1], a commercial Artificial Intelligence solution, Feature Selection is the process of isolating the most consistent, non-redundant, and relevant features to use in model construction. With the usage of feature selection, it is possible to reduce the size of training data, by removing features that would most likely penalize the training of the model, given its redundancy[71].

Sequential Feature Selection

The Sequential Feature Selector is an iterative process. This algorithm has several special implementations. The case that is used in the toolkit is the Sequential Backward Selection which removes a feature from the group of features at a time, then uses a regression algorithm to calculate a score with the features used. Additionally, the group of features with the best score is selected, discarding, in this way, a feature, meaning that the feature discarded impacts less the scoring metric. This process is then repeated until it reaches the desired number of features.

It is important to stress that although the implemented algorithms are from the sci-kit learn library, this code structure allows support for virtually any other python ML library, with minimal effort.

[1] HeavyAI, see more information at: https://www.heavy.ai/technical-glossary/feature-selection

5.4 Performance Evaluation Module

The Performance Evaluation module contains the tools capable of assessing the performance of the algorithms, namely, the performance metrics used to evaluate the models and the testing performance.

5.4.1 Metrics

It was chosen to create an Interface both in the metrics and in the performance modules that allows users to create their own implementations. Along with the interfaces for creating new metrics, it also

Table 5.1: Implemented metrics

Metric	Formula	Description
Root Mean Squared Error (RMSE)	$\sqrt{\frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{n}}$	Measures the average difference between the estimated values and the groundtruth values. In general this metric penalizes the most the larger errors [81]
Mean Squared Error (MSE)	$\frac{\sum(\hat{y}_i-y_i)^2}{n}$	The Mean Squared Error aids to explain how the average absolute error, by itself isn't of much use but it is a pre-requisite of the RMSE. In general this metric greatly penalizes the errors[82]
Mean Absolute Error (MAE)	$\frac{\sum_{i=1}^{n}\lvert y_i-\hat{y}_i\rvert}{n}$	The mean absolute error of the estimation against the groundtruth, by itself it isn't a very useful metric, but along with the RMSE can help improve the understanding of the estimation. It doesn't penalize the values as much as RMSE and MSE. [81]
Normalized Root Mean Squared Error (NRMSE)	$\frac{\sqrt{\frac{\sum_{i=1}^{n}y_i-\hat{y}_i)^2}{n}}}{y_{imax}-y_{imin}}$	Measures the average difference between the estimated values and the groundtruth values in the same magnitude of the values. [83]
Standard Deviation Error (SDE)	$\sum(\hat{y}_i-y_i)^2-\frac{\sum_{i=1}^{n}\lvert y_i-\hat{y}_i\rvert}{n}$	The Standard Deviation Error or σ explains the dispersion of the error on the mean.[83]
R^2	$\frac{\sum\hat{y}_i-\bar{y}^2}{\sum y_i-\bar{y}^2}$	The R squared metric value is useful for linear regression. Given that it measures the correlation between the input and the output [84]

contains baseline implementations, such as the RMSE, NRMSE, SDE, MSE, and MAE metrics.

In Table 5.1 the metrics are clarified along with their use cases in machine learning along with its formula. To better clarify, the y_i is the ground truth value for the pixel i, the $\hat{y}_i$ is the estimation for the pixel i, the $\bar{y}$ is the average of the ground-truth values, and n is the number of samples.

5.4.2 Cross Validation

The Cross-validation is executed once per experiment. It is very important to be able to provide the same cross-validation to all algorithms, in order to be able to compare the performance between models with the same data. Along with the implementation of the KFold cross-validation algorithm, the importance of Cross-Validation was already discussed in Chapter 4.

Although Cross Validation is most of the time an added value, it might not be needed. In that regard, Cross-Validation is an optional feature, for instance, if a Cross-Validation algorithm is not provided, a train-test split will be performed with a ratio of 70-30, meaning that the data is divided into two sets: a training set containing 70% of the data, and a test set which contains the remaining 30% of data. Naturally, this ratio can be customized by the user.

5.5 Experiment Module

The diagram in Figure 5.6, represents the experiment, the results classes and the relation between them as well. There is only one class in the Experiment Module: the Experiment class. The experiment module is where the nutrient estimation pipeline is implemented, as seen in Figure 4.2.

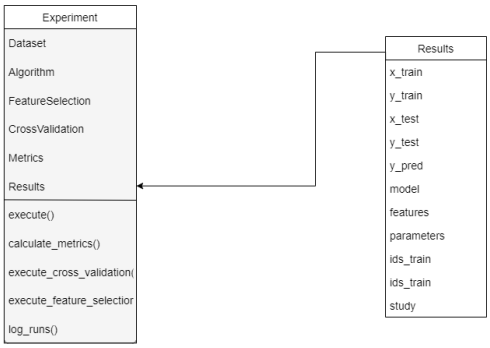


Figure 5.6: Class diagram of the Experiment and Results classes

It was opted to give the user the option to perform feature selection and cross-validation offline, meaning it can perform these two tasks without the need to execute the experiment. This was done in order to perform experiments with the same configurations, tackling the reproducibility challenges that were identified. On that account, there are three execute methods, one for feature selection, one for cross-validation, and the last one for the experiment. The latter can also execute the three tasks altogether if the need arises.

To perform an experiment, the user must prepare the following input: a dataset, an algorithm or set of algorithms for training, and optionally a feature selection algorithm and a cross-validation algorithm.

```
import terrasensetk as terrasensetk

dataset = terrasensetk.Dataset("C:/eopatches_maize")

parser = terrasensetk.Parser(dataset)

svr_model = terrasensetk.algorithms.Regression.SupportVectorRegression()
mlp_model = terrasensetk.algorithms.Regression.GradientBoostingRegressor()

feature_selection = terrasensetk.algorithms.FeatureSelection.SequentialFeatureSelector()

cross_validation_model = terrasensetk.performance.KFold(n_folds=5)

experiment = terrasensetk.Experiment("first_experiment",parser,[svr_model,mlp_model],
    feature_selection=feature_selection,cross_validation=cross_validation_model,
    fit_for_variable="N")

features = experiment.execute_feature_selection()
folds = experiment.execute_cross_validation()
results = experiment.execute(perform_optimization=False,folds=folds)

metrics = experiment.calculate_metrics(metric,['rmse','ae','sde'])
experiment.log_runs()
```

Listing 5.2: Code for performing an Experiment

In Listing 5.2, the code listing illustrated allows the execution of an experiment with two regression algorithms, namely the Support Vector Regression and Multi-Layer Perceptron. Additionally, it performs cross-validation with the KFold algorithm where $K = 5$, along with the Sequential Feature Selector feature selection algorithm. It should also be noted that no Hyper Parameter Optimization is executed.

After training the algorithm, by calling the *execute* method, the results get recorded in an object of the Results class and optionally, in the MLFlow framework, by using the method *log_runs()*. The user can then call the Experiment function *calculate_metrics()* and indicate which metrics the user wants to be reported back in a tabular array. The MLFlow was integrated into the TK given its standardized way to view and compare algorithms.

5.6 Utils Module

The Utils is a module that contains diverse code snippets that are used throughout the package. Inside this module, there are the Results class, the EOTasks class and the LUCAS dataset pickle.

5.6.1 Results

The Results object must contain all the information generated by TSTK, in the sense that any further data insight the end-user might want to perform must be possible through this object. In that regard, Figure 5.6 depicts the fields of the Results Object. This object keeps track of the values for the input features (spectral indices and bands) and targets(Nutrient values) (in other words, x and y, respectively) for both the train and test, as well as the labels of the selected input features and the trained model, along with the model parameters, the ids of the EOPatches used for train and testing.

5.6.2 EOTasks

The EOTask is an abstraction from eo-learn. It is the main component to process data from EOPatches, when performing any writing task with an EOPatch. This task is executed through an EOTask. For instance, to add an index to the dataset, internally it uses the custom EOTask (AddIndicesTask) which runs each EOPatch and calculates the indices based on the bands of that EOPatch. Then uses an out-of-the-box task, the SaveTask, that saves the new data to the EOPatch.

5.6.3 LUCAS dataset pickle

In Python, the term pickle refers to the serialization process of data. When python developers want to make their data persist in the filesystem without recurring to a Database Management System (DBMS), the pickle object is often used. In that regard, after producing the TSTK version of LUCAS Copernicus(which contains a merged version of LUCAS, the Soil Analysis, Copernicus, and the original LUCAS), it is saved in the toolkit in the form of a pickle. It should be noted that this pickle is only used when downloading new data.

Chapter 6

Case Study

This chapter showcases and further explains the functioning of TerraSenseTK and its usage. To evaluate the toolkit, this chapter presents a case study performed on several crops with data across Europe gathered from the LUCAS dataset along with imagery from Sentinel-2, where we try to predict the quantity of *Nitrogen*, according to the pipeline proposed in Chapter 4. The chapter is organized in the following themes 1) how the data was gathered, 2) which crops were selected, 3) how the spectral bands/indices or features were selected, 4) which machine learning algorithms were used, 5) how the generation of training and validation sets were performed and 6) the final results.

6.1 Data Engineering

6.1.1 Dataset Creation

Given that no Nutrient Estimation datasets containing imagery and groundtruth data were found, it was decided that a dataset would be built, comprised of images gathered by Copernicus' Sentinel-2 and groundtruth data from the LUCAS Dataset [2]. In addition, an effort from ESDAC established LUCAS Copernicus [63]. The original LUCAS counts with 97 information fields. A special edition of LUCAS,

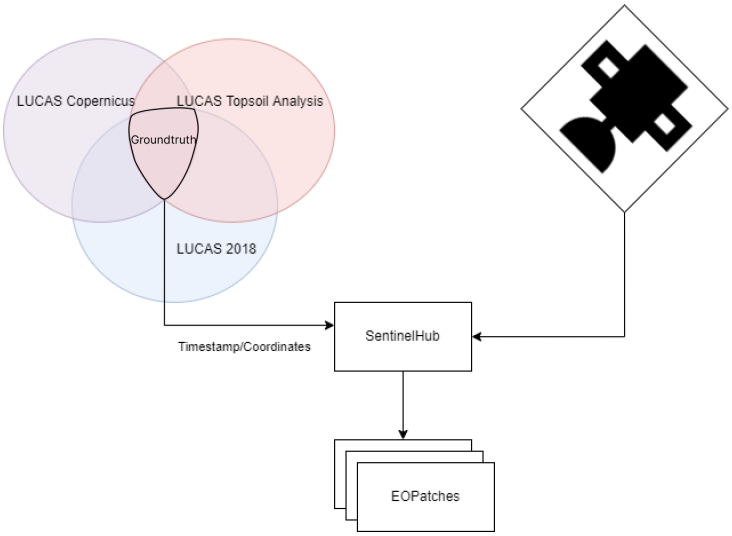


Figure 6.1: Dataset consolidation process

the Lucas Topsoil Analysis [62], adds soil information, such as the percentage of Clay, Sand, Silt, and Coarse Dirt, the pH of the soil in an H_2O solution and another in a $CaCl_2$ solution, the quantity of

$CaCO_3$, values for Nitrogen, Phosphorus and Potassium (NPK), and the elevation regarding the sea level.

Given the importance of these features for the experiment in hand, these were added to the dataset, along with more descriptive information regarding the Land Use Case, which was acquired by LUCAS Copernicus. In Table 6.1, the fields that were used in TSTK are depicted. In short, three versions of the LUCAS dataset were gathered: the original LUCAS, the LUCAS Copernicus, and the LUCAS Top Soil Analysis. These three were merged into a single source, ensuring that only data that contained geometric polygons or Area of Interest (AOI) (i.e LUCAS Copernicus $\bigcap$ (Original LUCAS $\bigcup$ LUCAS Top Soil Analysis)), as shown in Figure 6.1.

The LUCAS Copernicus dataset already contains the polygons/AOI regarding the area that was analyzed. Then a collection was made with all the information that comes from LUCAS Topsoil Analysis. Afterward, an eo-learn abstraction for storing geospatial data was used, known as an EOPatch, which has been improved by TSTK into a TSPatch. An EOPatch is an object that gathers metadata with actual images. In Figure 5.3 the EOPatch concept is illustrated. It has multiple fields with specific use cases, for example, image masks with and without temporal variables, and much more information. It also contains a field that accepts any structured information, the META INFO. This field stores the groundtruth data comprised by the intersection of the three versions of LUCAS. The TSPatch abstracts all this information from the end user, instead of performing cumbersome tasks to acquire information. The reasons for this implementation are described in Section 5.2.1.

Each TSPatch corresponds to a location in which data was gathered, the datasets created are comprised of about 18000 TSPatches with each one of those containing information about the terrain, soil constituents such as coarse soil, silt, and sand as well as the pH values, and levels of Nitrogen, Phosphorus and Potassium (NPK) in the soil. To reduce the dataset size, the size of the TSPatch was limited to an area around the polygon, i.e, a small buffer was added to the polygon defined by LUCAS Copernicus and a square around that buffer was drawn. In this way, it is possible to ensure that each TSPatch has one row of the LUCAS dataset.

Ground-Truth Mapping

After designing the storage of our dataset, it is necessary to segment the image information regarding the area in which the dataset contains groundtruth. For that matter, the polygons were rasterized and saved in the mask timeless chunk of the TSPatch. This way, it is possible to extract only the information regarding the AOI. This is done by multiplying the mask with the data values and getting the corresponding image of the AOI.

TSPatches were designed to contain several pieces of information. In this specific case, it was chosen to save the following data. The DATA chunk contains information regarding the raw bands from the Sentinel-2 from B01 to B12. It also contains the processed indices calculated with the bands, which were inspired in the work of Martinez, 2017 [85], where several indices were used (Inverted Red-edge Chlorophyll index (IRECI), NDVI, Normalized Difference Water Index (NDWI), NDI45, Red Edge Position: linear interpolation method (REPLI), Red Edge Chlorophyll index (RECHI) and Green Normalized Difference Vegetation Index (GNDVI)).

The Table 6.2 illustrates the indices that were implemented in the Experiment as well as their origin. Each of these indices was calculated to a determined use case and allows the exploration of more possibilities on the images.

According to Martinez [85], the reflectance in the red-edge region is important because of its relationship with the biophysical and biochemical parameters of plants, hence the inclusion of several Red Edge Position (REP) and relationships between the red-edge peak and the chlorophyll content indices,

Table 6.1: LUCAS Copernicus attributes, based on [63]

Origin of Attributes	#	Attribute Names
Original Fields	97	POINT_ID, NUTS0, NUTS1, NUTS2, NUTS3, TH_LAT, TH_LONG, OFFICE_PI, EX_ANTE, SURVEY_DATE, CAR_LATITUDE, CAR_EW, CAR_LONGITUDE, GPS_PROJ, GPS_PREC, GPS_ALTITUDE, GPS_LAT, GPS_EW, GPS_LONG, OBS_DIST, OBS_DIRECT, OBS_TYPE, LC1, LC1_SPEC, LC1_PERC, LC2, LC2_SPEC, LC2_PERC, LU1, LU1_TYPE, LU1_PERC, LU2, LU2_TYPE, LU2_PERC, PARCEL_AREA_HA, TREE_HEIGHT_SURVEY, TREE_HEIGHT_MATURITY, FEATURE_WIDTH, LNDMNG_PLOUGH, LM_PLOUGH_SLOPE, LM_PLOUGH_DIRECT, LM_STONE_WALLS, CROP_RESIDUES, LM_GRASS_MARGINS, GRAZING, SPECIAL_STATUS, LC_LU_SPECIAL_REMARK, CPRN_CANDO, CPRN_LC, CPRN_LC1N, CPRNC_LC1E, CPRNC_LC1S, CPRNC_LC1W, CPRN_LC1N_BRDTH, CPRN_LC1E_BRDTH, CPRN_LC1S_BRDTH, CPRN_LC1W_BRDTH, CPRN_LC1N_NEXT, CPRN_LC1E_NEXT, CPRN_LC1S_NEXT, CPRN_LC1W_NEXT, CPRN_URBAN, CPRN_IMPERVIOUS_PERC, INSPIRE_PLCC1, INSPIRE_PLCC2, INSPIRE_PLCC3, INSPIRE_PLCC4, INSPIRE_PLCC5, INSPIRE_PLCC6, INSPIRE_PLCC7, INSPIRE_PLCC8, EUNIS_COMPLEX, GRASSLAND_SAMPLE, GRASS_CANDO, WM, WM_SOURCE, WM_TYPE, WM_DELIVERY, EROSION_CANDO, SOIL_STONES_PERC, BIO_SAMPLE, SOIL_BIO_TAKEN, BULK0_10_SAMPLE, SOIL_BLK_0_10_TAKEN, BULK10_20_SAMPLE, SOIL_BLK_10_20_TAKEN, BULK20_30_SAMPLE, SOIL_BLK_20_30_TAKEN, STANDARD_SAMPLE, SOIL_STD_TAKEN, ORGANIC_SAMPLE, SOIL_ORG_DEPTH_CANDO, PHOTO_POINT, PHOTO_NORTH, PHOTO_EAST, PHOTO_SOUTH, PHOTO_WEST
Label Explicit Fields	9	LC1_LABEL, LC2_LABEL, LC1_SPEC_LABEL, LC2_SPEC_LABEL, LU1_LABEL, LU2_LABEL, LU1_TYPE_LABEL, LU2_TYPE_LABEL, CPRN_LC_LABEL
LUCAS Copernicus Fields	3	CPRN_LC_SAME_LC1, LUCAS_CORE_INTERSECT, COPERNICUS_CLEANED, POLYGON
LUCAS TOP Soil Analysis	15	COARSE, CLAY, SAND, SILT, pH(CaCl2), pH(H2O), EC, OC, CaCO3, N, P, K, ELEVATION, LC1_DESC, LU1_DESC

which could be of great relevance for the nutrient estimation prediction.

The dataset was created using the TSTK interfaces, and, for that, the code in Listing 6.1 was used. It should be stressed that the complexities that were addressed in this section are abstracted from the user. By default, when calling `get_groundtruth()`, it includes all the crops in the dataset and generates the dataset automatically. Additionally, the user can also acquire a filtered portion of the information available by calling `country.get_groundtruth(crop = "Common wheat")`.

```
path = "C:/eopatches"
country = terrasensetk.Downloader(continent="Europe",eopatch_size=40)
country.get_bbox()
country.get_groundtruth()
country.get_bbox_with_data()
country.download_images(path,country.get_bbox_with_data())

dataset = terrasensetk.Dataset(path)
dataset.add_index("NDVI74","(B07-B04)/(B07+B04)")
dataset.add_index("IRECI","(B07-B04)/(B05/B06)")
dataset.add_index("REM","(B07/B05)-1")
dataset.add_index("GM","(B07/B03)-1")
dataset.add_index("REPLI","700+40*(((B04+B07)/2)-B05)/(B06-B05)")
dataset.add_index("RECHI","(B06-B05)/B05")
dataset.add_index("S2REP","705+35*((((B07+B04)/2)-B05)/(B06-B05))")
dataset.add_index("NDI45","(B05-B04)/(B05+B04)")
dataset.add_index("GNDVI","(B08-B03)/(B08+B03)")
dataset.add_index("NDVINB","(B8A-B04)/(B8A+B04)")
dataset.add_index("RVI","(B08/B04)")
dataset.save_indices_to_eopatches()
```

Listing 6.1: Code for creating the dataset

Table 6.2: Spectral indices calculated from spectral measurements for Sentinel2 [85]

Formula	Name	Calculation with Sentinel 2	Source
$\frac{NIR-Red}{NIR+Red}$	NDVI74	$\frac{B07-B04}{B07+B04}$	*Rouse et al. (1974)*
$\frac{NIR-RED}{\frac{RE2}{RE1}}$	Inverted Red-Edge Chlorophyll index - IRECI	$\frac{B07-B04}{\frac{B05}{B06}}$	*Frampton et al. (2013)*
$\frac{R783}{R705}-1$	Red Edge Model (REM)	$\frac{B07}{B05}-1$	*Gitelson et al. (2005)*
$\frac{R783}{R560}-1$	Green Model (GM)	$\frac{B07}{B03}-1$	*Gitelson et al. (2005)*
$700+40\left(\frac{\frac{R670+R780}{2}-R700}{R740-R700}\right)$	Red Edge Position: linear interpolation method REP-LI	$700+40\left(\frac{\frac{B04+B07}{2}-B05}{B06-B05}\right)$	*Guyot and Baret (1988)*
$\frac{R750-R700}{R700}$	Red Edge Chlorophyll index (ReChi)	$\frac{B06-B05}{B05}$	*Gitelson et al. (1996)*
$705+35\left(\frac{\frac{NIR+R}{2}-RE1}{RE2-RE1}\right)$	S2-REP	$705+35\left(\frac{\frac{B07+B04}{2}-B05}{B06-B05}\right)$	*Frampton et al. (2013)*
$\frac{NIR-Red}{NIR+Red}$	NDI 45	$\frac{B05-B04}{B05+B04}$	*Delegido et al. (2011)*
$\frac{R790-R550}{R790+R550}$	Green Normalized Difference Vegetation Index (GNDVI)	$\frac{B08-B03}{B08+B03}$	*Gitelson et al. (1996)*
$\frac{R860-R660}{R860+R660}$	NDVI Narrow Band (NDVI-NB)	$\frac{B8A-B04}{B8A+B04}$	*Qi et al. (2011)*
$\frac{R810}{R660}$	Ratio Vegetation Index (RVI)	$\frac{B08}{B04}$	*Jordan (1969)*

6.1.2 Outlier Detection and Removal

After analyzing the data for the case study, it was noticed that further processing was needed, given that there were a number of *outliers*, as evidenced on Figure 6.2(left). When training a model, the outliers should be removed when not enough data exists for those extreme points to be properly generalized. For that reason, we calculated the average (μ) and standard deviation (σ) and removed the values that were outside the value of ($\mu \pm 3 \cdot \sigma$). In Figure 6.2(right), it is possible to see that the values that were above five g/kg were discarded. Although this practice reduces the number of training samples, according to Suresh [86], it helps model performance, by reducing the statistical distribution of the dataset.

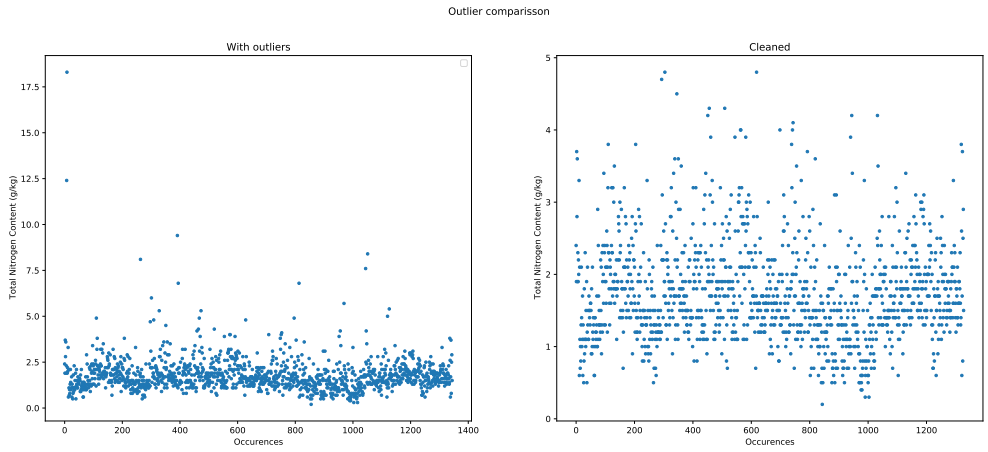


Figure 6.2: Comparisson of groundtruth data for Common wheat values with outliers(left) and after running outlier cleaning(right)

6.2 Crop Selection

For the preliminary results, a model was trained with all the available data (18983 different points), but the results were not too favorable. This occurrence was mainly because each crop reflects in different regions of the electromagnetic spectrum. In simple terms, the wheat is yellow, so it reflects more in the yellow region (around 600nm). While vineyards, for instance, which are mainly green from an aerial perspective, reflects more in the green region (around 550nm). Taking these spectral differences into consideration, it was opted to separate the data into different crops, in order to get a more concise and specific trained model that fits well for a crop.

Although this segregation by crop, theoretically offers better results, it raises new challenges. For instance, crops that have a low number of patches may not have enough data to properly fit a model. In that regard, the crops that have more occurrences were chosen. The Table 6.3 was created to identify the forty crops with the most occurrences. It should be noted that the column that contains the crop type, the LC1_LABEL, describes the land use case. For that reason, it includes other labels that are out of the scope of this case study.

In addition, the table contains the different Nomenclature of Territorial Units for Statistics (NUTS) code where those crops appear. The NUTS code is a standard for land division. In NUTS0, it is split by country; NUTS 1 describes major socio-economic regions; NUTS 2 describes basic regions for the application of regional policies and NUTS 3 describes small regions for specific diagnoses[1]. With this information, it is possible to know, for instance, that wheat fields were reported in 24 different countries.

Among the crops, there are other label values for grassland, forests, and other Artificial areas which should not be considered, once they are not crops. Additionally, labels that mixes crops should not also be included, such are the cases of Mixed cereals for fodder, other leguminous and mixtures for fodder.

Ultimately, the Case Study will consider the top five crops, highlighted in bold in Table 6.3, namely Common wheat, Barley, Maize, Sunflower, and Vineyards. These were chosen because 1) were the crops with the most occurrences, 2) ongoing studies in Madeira for Vineyards with Isoplexis, and 3) are actually crops.

6.3 Feature Engineering

The Feature Engineering process is of great importance, as previously mentioned in Section 5.3.2. In this case study, two different approaches for feature selection were tested, namely manual and automatic Feature Selection.

6.3.1 Feature Overview

The input features have been analyzed in Figures 6.3 and 6.4, to see their distribution. These plots were generated using the toolkit's parser module, and the code is available in Listing 6.2. The toolkit is only used to parse the data from the dataset into a dataframe, which then iterates over all features to generate the plots for each feature.

Further analyzing the input features in Figure 6.4, it was noted that the IRECI and S2REP indices do not have variability. This is mostly explained given that in their formulas, which are present in Table 6.2, a constant value of 700 is added, which escalates compared to other features. In general, all features are somewhat skewed, which corresponds to the target values which are also skewed(will be shown later), that can be considered as an advantage given that it can probably describe the target values.

[1] The NUTS standard: https://ec.europa.eu/eurostat/web/nuts/background

Table 6.3: Top 40 crops with most occurences

Crop Type	Occurrences	NUTS0	NUTS1	NUTS3
Grassland without tree/shrub cover	2543	26	91	703
Broadleaved woodland	1659	26	87	545
Common wheat	1345	24	77	381
Pine dominated coniferous woodland	1113	22	56	226
Barley	796	23	65	247
Spruce dominated coniferous woodland	722	20	45	183
Spruce dominated mixed woodland	642	18	41	145
Maize	591	23	69	308
Other bare soil	505	23	65	171
Olive groves	401	5	18	91
Shrubland without tree cover	349	18	48	132
Other mixed woodland	346	24	50	145
Grassland with sparse tree/shrub cover	336	23	62	199
Rape and turnip rape	327	18	56	168
Sunflower	274	11	25	101
Shrubland with sparse tree cover	250	20	40	125
Other coniferous woodland	244	19	35	88
Vineyards	236	11	28	91
Rye	185	15	35	100
Oats	179	20	44	115
Durum wheat	177	15	34	86
Lucerne	168	13	36	102
Dry pulses	141	16	37	84
Nuts trees	111	9	18	43
Sugar beet	98	12	29	61
Triticale	90	7	27	63
Other leguminous and mixtures for fodder	81	16	29	66
Potatoes	76	18	36	57
Mixed cereals for fodder	40	6	12	33
Clovers	39	12	21	35
Apple fruit	37	14	26	30
Soya	37	8	14	30
Other artificial areas	37	14	27	35
Other fresh vegetables	34	13	20	28
Cotton	26	2	3	9

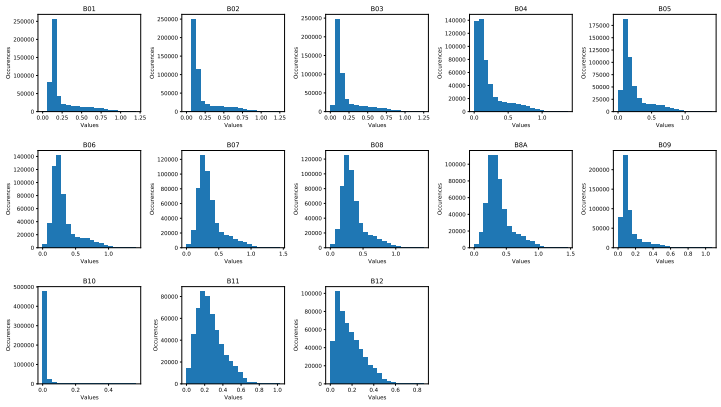


Figure 6.3: Distribution of the bands of the dataset

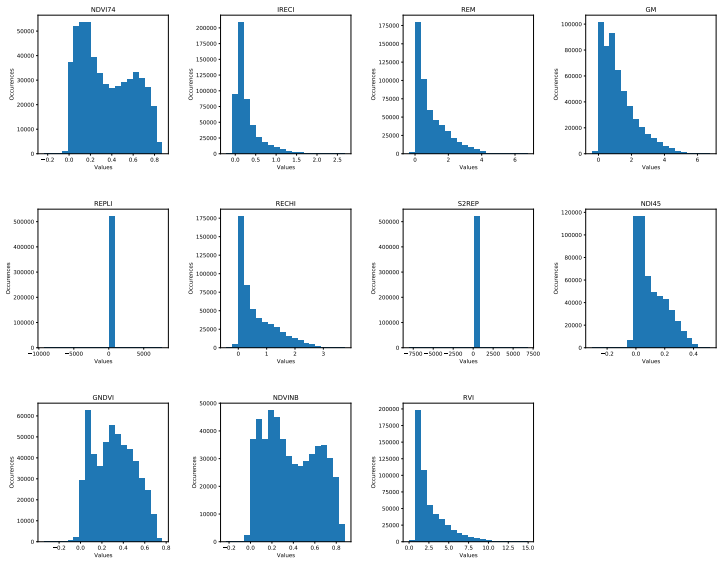


Figure 6.4: Distribution of the indices of the dataset

```
parser = terrasensetk.Parser(dataset)
df = parser.create_dataframe()
fig,axs = plt.subplots(3,4,figsize=(18,14))
flat_ax = axs.flatten()
plt.tight_layout()
plt.subplots_adjust(wspace=0.4,hspace=0.4)
for i,col in enumerate(df.columns[:-3]):
    cur_ax = flat_ax[i]
    cur_ax.hist(df[col],20)
    cur_ax.set_title(col)
    cur_ax.set_ylabel("Occurences")
    cur_ax.set_xlabel("Values")
```

Listing 6.2: Code for performing feature distribution plots

6.3.2 Manual Feature Selection

First, manual feature selection was tested. This was done by calculating Spearman correlation[87] between the input features (i.e. the bands and the spectral indices) and the outputs (i.e, the value of the nutrients, in this case). Input features that showed less correlation results would be discarded and the top N features were selected. The Spearman correlation is a rank correlation algorithm, which measures the strength of a linear/monotonic correlation between variables. It ranks the pair of features with a value from -1 until 1. The greater the absolute correlation value, the stronger the correlation between the features is. Naturally, the closer the value is to zero, the weaker the correlation.

In the case of common wheat, Figure 6.5 depicts the *absolute* correlation between each feature. This process is done by choosing the features that are closer to the extreme values (-1 and 1) for the values of NPK. It is visible that, there are very weak correlations to the NPK values(< 0.2).

Although useful, with manual feature selection, it is not immediately evident which features should be selected. Besides, it is a cumbersome process and not systematic. For instance, one would find the features with the highest correlation to the output, but it does not ensure to be the best features for the dataset.

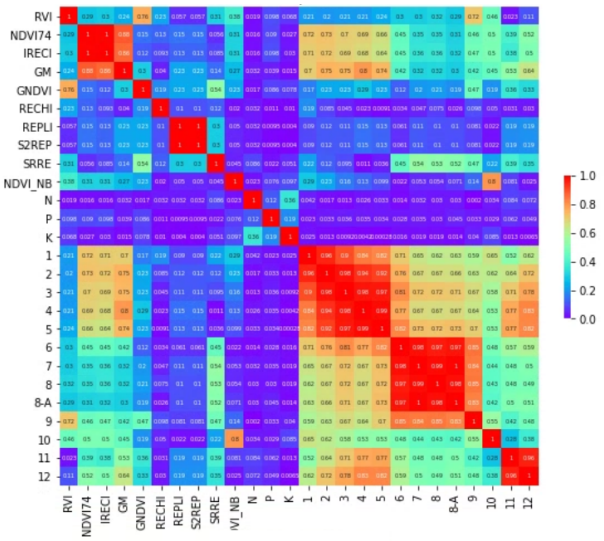


Figure 6.5: Spearman correlation performed over the Common wheat crop

6.3.3 Automatic Feature Selection

For the purpose of this case study, a Sequential Feature Selector was tested which uses the Gradient Boosting Regressor model to automatically identify the most useful features. The inputs for this automatic feature selector are the same as the regressor. It receives the input features and the groundtruth values. Ultimately, it was found that automatic feature selection does not require effort, from the user perspective, i.e. simpler to use, and it ensures that the features selected by the model are those that best describe the dataset. On that premise, the Automatic Feature Selection was selected as the method for performing feature selection throughout the remaining case study.

Taking this into consideration, in the case study, Automatic Feature Selection was conducted, and the code sample for it is present in Listing 5.2, where, on lines 10 the feature selection algorithm is instantiated, then, in line 14, it gets performed internally, when running the experiment. It is also possible to run feature selection *offline* by calling `features = experiment.execute_feature_selection()`. Given the volume of patches, it was opted to use *one-third* of the data to perform feature selection, in order to decrease the time to execute the feature selection. Additionally, to ensure that the data is geographically diversified, the order of images was randomized.

The results of the feature selection for each crop are presented in Table 6.4. It was expected that the NDI45 and NDVI74 would be chosen given that it is correlated with the levels of chlorophyll of the crop which could potentially be connected to the nutrients in the soil. However, B10 which corresponds to the reflectance on the Infrared (IR) band was not foreseen as a selected band. This can be justified by the fact that B10 has a lower spacial resolution, meaning it has a greater chance to correlate to the outputs. Interestingly, IRECI and S2REP were selected, even though it was previously explained that these bands were the worst ones. The reason why these bands contribute the most to the problem is unknown but may be due to the large number which can cause variance errors in the algorithms.

6.4 Machine Learning Algorithms and Hyper-parameter Optimization

The fulcral point of this case study is to assess the possibility to perform nutrient estimation with a group of crops. For this matter, it is extremely important to choose an adequate algorithm with the right hyper-

Table 6.4: Selected input features for each crop

Crop Type	Features selected by Sequential Feature Selector for original data				
Common wheat	NDVI74	REPLI	NDI45	S2REP	B10
Barley	S2REP	REPLI	NDI45	B09	B12
Maize	NDVI74	REPLI	NDI45	B01	B11
Sunflower	IRECI	GM	NDI45	B08	B10
Vineyards	IRECI	REPLI	NDI45	S2REP	B10

parameters. To this end, the first step was to look at the target data for each crop to select the algorithms that would make more sense to the problem at hand. Then, for each of the selected algorithms, the HPO feature of TerraSenseTK was applied.

6.4.1 Algorithm Selection

The first step was to display the distribution of the target values. To this end, the Nutrient values for each crop were analyzed using Histograms. The resulting histograms are presented in Figure 6.6.

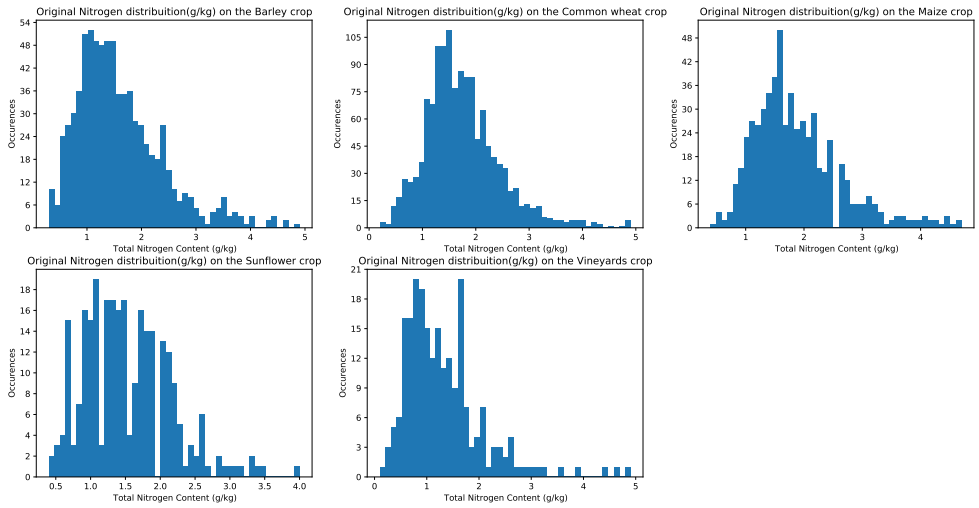


Figure 6.6: Data Distribution for original data, top three plots, Barley, Wheat, and Maize, bottom two plots, Sunflower, and Vineyards, respectively

It is visible in Figure 6.7 that the data shows common *data imbalance* signs, which happens when the dataset does not have a balanced data distribution, meaning that several ”classes” or regions in regression are *over-represented*, i.e, have significantly more occurrences that other regions. This issue leads to always predicting the values with the most occurrences hence the overfitting of the simpler algorithms[88]. Another sign of Data Imbalance is the existence of skewed distributions with a long tail [89] which, in this case, ranges from three to five g/kg in the selected crops. According to literature, there are two approaches to tackle this problem[89]: the *algorithm centered approach*, where suitable algorithms are chosen, and the *data centered approach*, where the data is manipulated to try and mitigate the Data Imbalance.

Considering that ensemble algorithms are able to prevent overfitting and potentially reduce the data imbalance effect, the simpler algorithms will be discarded and only Random Forest and Gradient Boosting will be used.

The field of imbalanced domains applied to regression is still a vastly unexplored area that counts with very few works. The most well-known work is Branco, 2017 [90], which proposed Synthetic Minority Oversampling With Gaussian Noise (SMOGN), a technique based on the Synthetic Minority Oversampling Technique (SMOTE) which was coined in Chawla et al, 2002. [91]. SMOTE is a technique used in classification to upsample (by interpolating the nearest data points) the classes with the least data and downsample the classes with the most data. In a more pragmatic setting, [4] used SMOGN to improve the imbalanced dataset used to estimate areas with potential contamination of arsenic and obtained a 70% increase of samples and an improvement of accuracy (R^2 = 0.492 with no processing to R^2= 0.840 with SMOGN application). However, researchers in Yang et al,2021. [89], mentioned that SMOGN interpolation does not lead to meaningful new synthetic samples, given the nonlinear relationship between the nature of the data and the output.

Another simpler and overall effective technique is the logarithmic scaling of target features, which helps follow a normal/Gaussian distribution. With these techniques in consideration, logarithmic normalization of the target features was tested to mitigate the imbalance of data in our dataset, as well as the usage of robust ensemble algorithms such as the case of Random Forest and Gradient Boosting Machines.

6.4.2 Target Logarithmic Transformation

The logarithmic technique allows for two things. The first is reducing the range of the input and the second is to scale the data to follow a normal distribution. The data fed to the algorithms were normalized with the logarithm of the target feature(T_y), i.e. Equation (6.1), but in order to compare the results obtained, predicted data must be rescaled back y' into $\hat{y}$. For that matter, the reverse operation was applied, i.e., Equation (6.2). With this operation, the metrics can be compared with other approaches, given that our groundtruth will be the same scale as the other experiments. The Figure 6.7 was generated to show the effect of log-scaling produced in the original data. This transformation removed the long tail the dataset had and now it resembles more a gaussian curve.

$$T_y = \ln y \tag{6.1}$$

$$\hat{y} = e^{y'} \tag{6.2}$$

When applying log-scaling, the target data changes, meaning that the data engineering process that was performed on the original data must be done again for the log-scaled data. For that reason, the hyperparameter and feature selection process was repeated.

To perform the log-scaling numpy and the TSTK were used, and the data was rewritten with the code in Listing 6.3, which depicts how the data was log-scaled. It should be noted that the remaining experiment code stays the same.

6.4.3 Hyper-Parameter Optimization

Hyperparameter is a very important aspect that theoretically improves the results by using adequate hyperparameters. Another important aspect is the choice of good machine learning algorithms. For that reason, two *classical* Machine Learning algorithms were trained for each of the crops: Gradient Boosting Machine and Random Forest.

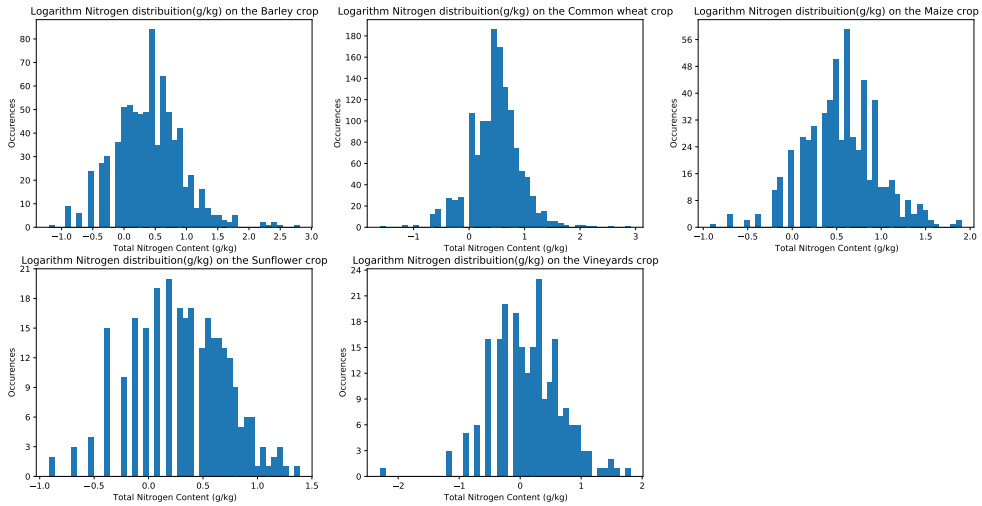


Figure 6.7: Data Distribution with logarithmic scaling, top three plots, Barley, Wheat and Maize, bottom two plots, Sunflower, and Vineyards, respectively

```
parser = terrasensetk.Parser(dataset,crop_eopatches)
parser.create_dataframe()
new_df = parser.save_dataframe.copy()
new_df["N"] = np.log(new_df["N"])
parser.save_dataframe = new_df
```

Listing 6.3: Code for log-scaling the data

To determine the effectiveness of the Hyperparameter Optimization, the experiments will be run twice: one with hyperparameters and the other with the default parameters of the algorithms. Once again, *one-third* of the log-scaled and original data was used to accelerate the time it takes to perform the Hyper-parameter optimization. Tables A.1 and A.2 depicts the hyperparameters obtained for all the crops both original and log-scaled data, respectively. These hyper-parameters were chosen given that their RMSE were the lowest of the feature space for 100 trials.

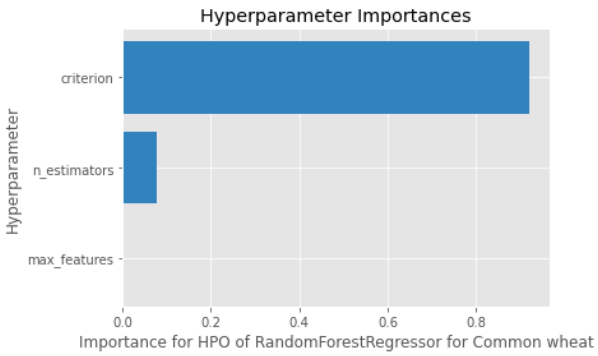


Figure 6.8: Feature importance of Random Forest for Common wheat

On a further note, Figure 6.8 depicts the Feature Importance. With this graph, it is possible to denote that the greater importance is within the choice of the criterion method.

6.4.4 Training and Validation

For the training and validation of the models, Cross Validation was performed. Despite the fact that Common wheat contains enough samples to execute a train-test split, it was opted to perform 5-Fold Cross Validation on all crops, for comparability reasons, given that other crops have significantly fewer samples. In this sense, for each crop, each model was executed five times with different folds. It should be noted that the same set of folds are used throughout the models and experiments, for the sake of reproducibility. The number of samples existing in each train-test set are shown in Table 6.5.

Table 6.5: Quantity of images for each crop

Crop Type	Train Samples	Test Samples
Common Wheat	1061	266
Barley	626	157
Maize	463	116
Sunflower	216	55
Vineyards	180	45

For validation, a few of the metrics present in Table 5.1 were selected, namely the MAE, RMSE, and the normalized version of these two. The results obtained are present in Tables 6.6 and 6.7.

6.5 Results and Discussion

This section presents and briefly discusses the results obtained in the different crops against the two approaches: the original data and the log-scaled data.

6.5.1 Results Overview - Using ML Flow

MLFlow can be used to keep track of Machine Learning experiments. To simplify the usage of MLFlow, a method to record the experiments is included in TSTK. MLFlow includes both an API to retrieve and analyze the experiments, as well as a GUI that can be used to visualize and further organize the experiments.

In Figures 6.9 and 6.10 the user interface is shown, where graphs can be plotted with information regarding the various experiments. It should be stressed that MLFlow can plot any information saved in the model, such as hyperparameters and metrics, for instance.

6.5.2 Performance Per Crop

To calculate the performance of the algorithms 4 metrics are calculated, the RMSE (Equation (6.5)), MAE (Equation (6.4)) and their normalized versions, NRMSE (Equation (6.3)) and NMAE (Equation (6.6)).

$$NRMSE = \frac{\sqrt{\frac{\sum_{i=1}^{n} y_i - \hat{y}_i)^2}{n}}}{y_{imax} - y_{imin}} \tag{6.3}$$

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{6.4}$$

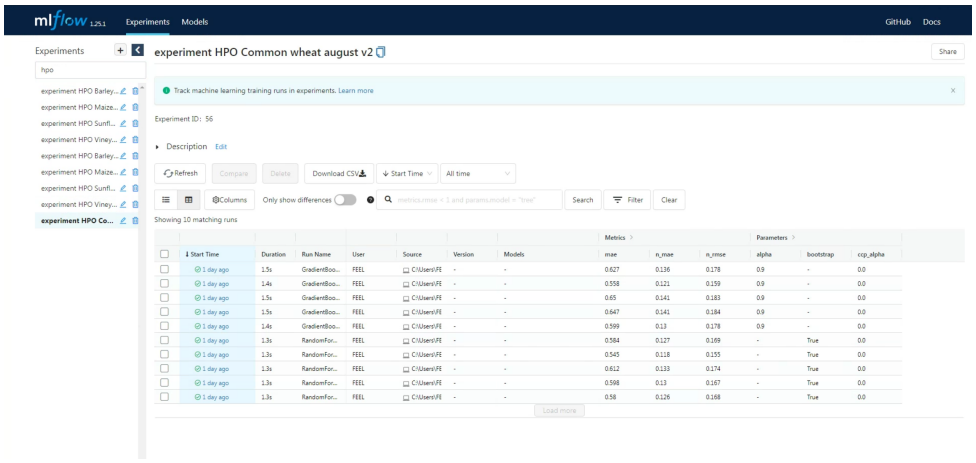


Figure 6.9: MLFLow user interface

Figure 6.10: MLFLow user interface auto-generated graphs

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{6.5}$$

$$NMAE = \frac{\frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}}{y_{imax} - y_{imin}} \tag{6.6}$$

These two base metrics were chosen because is possible to determine the accuracy between different datasets with the NRMSE. Given that it is normalized, the value of the metrics gives the error percentage, i.e, 0.14 means 14% error, while NMAE shows the average variance of the error. For comparison reasons, the data distribution will be shown for each of the crops, then the summarized results table will be presented and finally, the individual performance for each crop and algorithm selected will be demonstrated.

Original Data

The original data performance was calculated by averaging the results of the five folds, according to the metrics in Table 6.6. The NRMSE, which corresponds to the errors, goes from 15% to 30%. Common Wheat shows better results than other crops, which can be justified by the increased number of samples.

Table 6.6: Metrics obtained by original data for all crops

Crop Type	Algorithm	RMSE	MAE	NRMSE	NMAE
Common Wheat	GBM	0.691	0.512	0.151	0.116
	RF	0.762	0.561	0.160	0.124
Barley	GBM	0.885	0.663	0.306	0.229
	RF	0.894	0.671	0.309	0.232
Maize	GBM	0.869	0.662	0.301	0.229
	RF	0.825	0.636	0.286	0.220
Sunflower	GBM	0.634	0.492	0.220	0.170
	RF	0.608	0.471	0.210	0.163
Vineyards	GBM	0.586	0.436	0.203	0.151
	RF	0.589	0.445	0.204	0.154

Barley and Maize show virtually the same error, 30%, while Sunflowers have a better error of 20%. Once again, the hyperparameters were selected for the original data in Table A.1 in order to compare the results with the log-scaled hyperparameters.

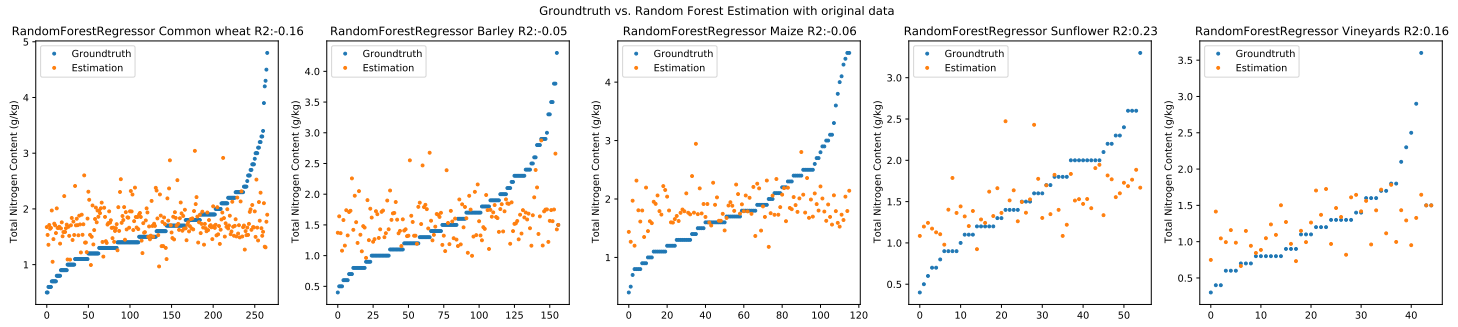
After further analyzing the prediction against the groundtruth, in Figure 6.11 a trendline is clearly noticeable in the Common Wheat, Barley, and Maize, given that they have more data points (160 and 120 occurrences, respectively), while Sunflower and Vineyards are not quite as noticeable taking in consideration that they have fewer datapoints (60 and 40, respectively).

Logarithmic Transformation

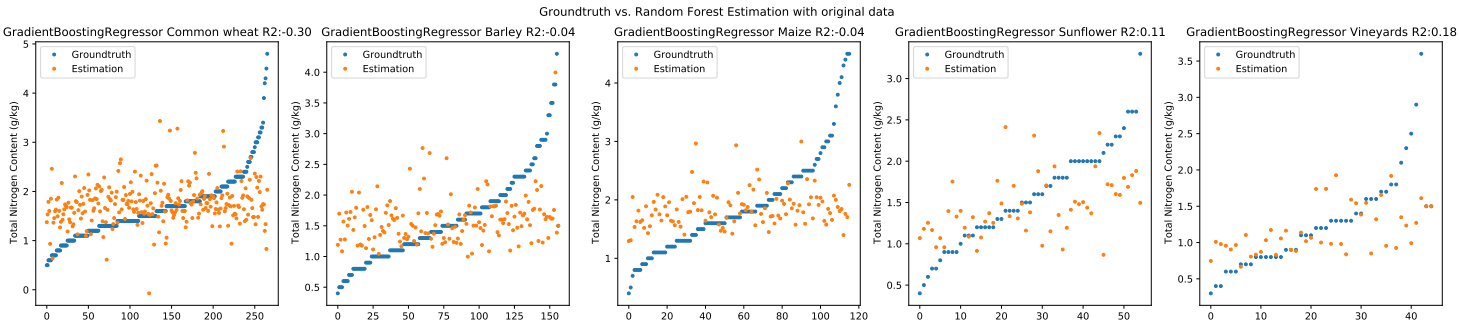
Table 6.7: Metrics obtained by performing log transformation to remaining crops

Crop Type	Algorithm	RMSE	MAE	NRMSE	NMAE
Common Wheat	GBM	0.692	0.522	0.150	0.113
	RF	0.785	0.599	0.171	0.130
Barley	GBR	0.916	0.652	0.177	0.126
	RF	0.958	0.690	0.185	0.133
Maize	GBR	0.863	0.651	0.200	0.151
	RF	0.820	0.619	0.190	0.144
Sunflower	GBR	0.563	0.424	0.202	0.152
	RF	0.531	0.403	0.190	0.144
Vineyards	GBR	0.642	0.457	0.222	0.158
	RF	0.632	0.455	0.219	0.157

After applying the logarithm, the Figure 6.7 was generated. It is noticeable that the new distribution follows a normalized pattern, with no tails, although many regions have missing data and a number of regions have many more occurrences.



(a)



(b)

Figure 6.11: Prediction of the original data with a) Random Forest and b) Gradient Boosting Machines

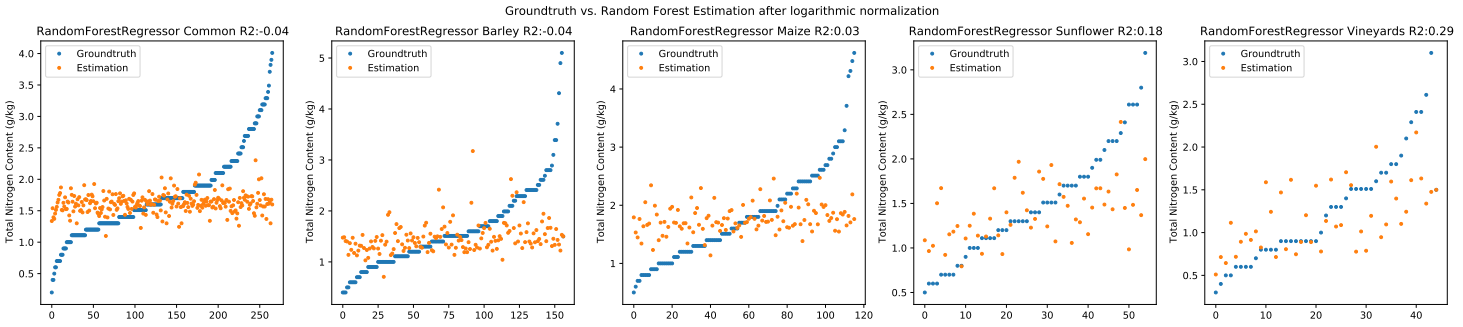
Interestingly the logarithm distribution obtained disparate results: few experiments improved while others worsened. For instance, Barley and Maize(Figure 6.6), describes an NRMSE of 0.17 and 0.18 for GBM and RF, respectively for the log-scaled data, versus an NRMSE of 0.30 for the original data with both algorithms, (an improvement of roughly 10%). On the other hand, the Sunflower crop obtained just about the same results, where original data shows a 2% better prediction. On the same note, Vineyards show a 1% improvement prediction with the original data. Interestingly, both algorithms show very similar results, which can be justified by the fact that both algorithms are bootstrapping algorithms based on decision trees.

Discussion

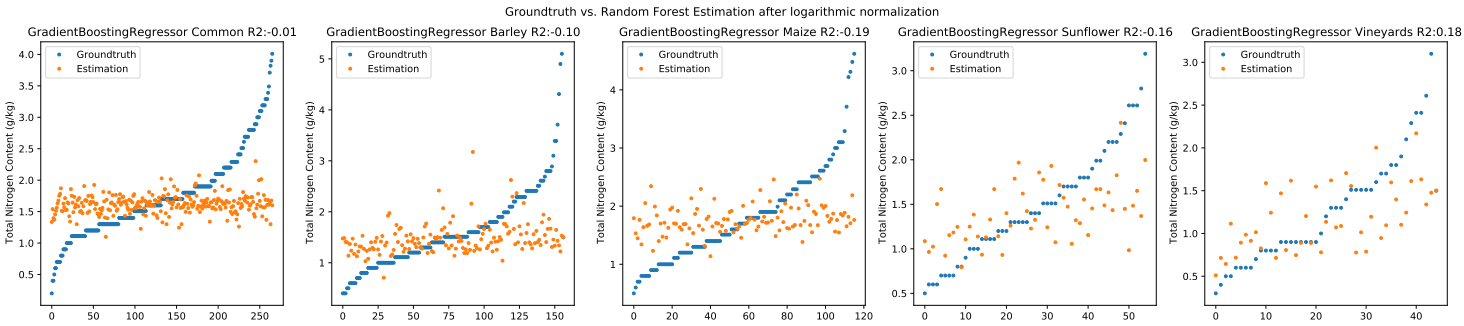
With the reported results, it cannot be concluded that these approaches actually mitigated the imbalanced data problem, given that Figure 6.12 still shows that ensemble algorithms do not accurately predict the values.

For a number of crops, log scaling managed to improve the metrics against the original scaling. It was reported, for instance, that for Barley and Maize the results were 10% better when performing log-scaling, which did not translate to a better ”accuracy” when visually comparing the groundtruth against the prediction. While Vineyards and Sunflowers obtained an improvement of 2% of the NRMSE. In this sense, the reported results are not consistent, and more experiments should be performed with other datasets or crops to come to a better conclusion.

Unfortunately, most research papers only report metrics. For instance, while we have an NRMSE of 0.13(13% error), which is better than the approach used by Song et al, 2018 [45], which presented an error of 0.30(30%)), which indicates our results as significantly better. However, when the results are visualized, it is noticeable that the predictions are not as precise as expected. In this sense, additional



(a)



(b)

Figure 6.12: Prediction of the log-scaled data with a) Random Forest and b) Gradient Boosting Machines

domain-specific metrics should be proposed to help pinpoint the accuracy of the model, or even perform the extra steps (such as the ones that were executed in this thesis and check for each datapoint the correctness of the prediction), as well as further studies, must be conducted to find new techniques to mitigate the effects of imbalanced data in regression. Given that there is yet to be defined a standard to compare results, most of the State-of-the-Art approaches cannot be compared with our results. In this sense, it is believed this work may have the potential to set the base of comparability in this field of research.

Chapter 7

Conclusion

This chapter presents the conclusion and is divided into three parts, 1) the summary of the thesis, 2) the research implications and potential applications, and 3) limitations and future work directions.

7.1 Summary

The main objectives of this thesis were to tackle the issues that serve as a barrier to the entry of Earth Observation. This is done by creating a toolkit that simplifies the process of performing soil estimation experiments and proposing a pipeline that was used in the Case Study.

The overall objectives of this thesis were accomplished, namely the creation of a toolkit aimed at reducing the learning curve of the field, by allowing to perform experiments with few lines of code, providing support to reproducibility issues, such as making the code available, and simple dataset sharing/creation with different data sources. Another objective was proposing a nutrient estimation pipeline, which is included in the toolkit. In this note, TSTK can be used, not only to perform experiments of soil sensing, but also for creating datasets which can then be exported and used to perform other machine learning experiments outside of the toolkit. Meaning it can be used as a dataset-generating tool, which is one of the current issues.

7.2 Research Implications and Potential Applications

With TerraSenseTK, several challenges that are present in the community are tackled, namely 1) barrier to entry, and 2) difficulty to define an experience, i.e., only ML practitioners can perform experiments. 3) streamlines the development of an experiment, making it easy to share the steps taken. The main target groups for TerraSenseTK were Machine Learning Researchers and Agriculture Researchers/Farmers, although the current release of TSTK is a programming tool rather than a user interface, so for now its usage is more focused on users with fair programming knowledge. Therefore, further efforts are being developed to create a User Interface. On that note, two internship proposals for developing a data representation module to be included in the toolkit were proposed. One major implication that this thesis brings is that metrics used in literature to report the results are most of the time an oversimplification of the problem. For instance, the results obtained in the Wheat experiment represent an NRMSE of 0.10 which means an error of 10%, which according to the literature, is a considerable achievement, but on further inspection, the algorithms are not capable of predicting the nutrient levels as expected.

In this sense, further research must be conducted to find *domain specific* metrics that better describe the assessment of the prediction against groundtruth, since most literature only reports metrics and not

in-depth analysis of the results.

7.3 Limitations and Future Work Directions

After analyzing the results, we stumbled upon a challenge that is not yet mature enough to be solved: the *imbalanced domains* for regression. Although various approaches were tested, such as logarithmic scaling, the results obtained did not improve the outcome of the experiment. The possible solutions for the data imbalance problem, are 1) drastically downsampling the dataset and having all ”classes” with the same number of occurrences, 2) implementing the Label Distribution Smoothing (LDS) and Feature Distribution Smoothing (FDS) methods proposed by Yang et al., 2021 [89], which will potentially solve the imbalanced data problem, 3) The REBAGG algorithm[92] is an algorithm aimed at imbalanced datasets, so, implementing it, should also be an option to consider.

Another future work direction is, for instance, implementing classification into the toolkit, which is a simple task given that to implement this it is only needed a new *specialization* of the parser class. Furthermore, taking into account that most crops in the dataset contain few samples, another opportunity is the implementation of semi-supervised algorithms, which will most likely yield better results than supervised learning. As it was previously mentioned, ongoing future work is being developed, which is the creation of a novel data representation module.

Another important step is the creation of a user-friendly interface for users from other areas, such as agriculture, allowing the toolkit to become a program (similarly to WEKA[1], which is a program aimed for field specialists to apply machine learning to their research) and to perform data cleaning because sometimes the images captured by the satellites contain clouds, and for the bands in the visible spectrum it is most likely that no correlation can be defined. Finally, new algorithms and datasets can be created without changing the codebase.

To conclude, with these prospects in mind, the TSTK has the potential to lay the foundations to mitigate the existing comparability challenge in the field of EO.

[1] WEKA - Machine Learning at Waikato University, available at https://www.cs.waikato.ac.nz/ml/index.html

Bibliography

[1] P. Geladi and H. F. Grahn, "Multivariate Image Analysis," in *Encyclopedia of Analytical Chemistry*, American Cancer Society, 2006. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470027318.a8106.

[2] G. Tóth, A. Jones, and L. Montanarella, "The LUCAS topsoil database and derived information on the regional variability of cropland topsoil properties in the European Union," *Environmental Monitoring and Assessment*, vol. 185, pp. 7409–7425, Sept. 2013.

[3] M. Emadi, R. Taghizadeh-Mehrjardi, A. Cherati, M. Danesh, A. Mosavi, and T. Scholten, "Predicting and Mapping of Soil Organic Carbon Using Machine Learning Algorithms in Northern Iran," *Remote Sensing*, vol. 12, p. 2234, Jan. 2020. Number: 14 Publisher: Multidisciplinary Digital Publishing Institute.

[4] D. Walshe, D. McInerney, R. V. De Kerchove, C. Goyens, P. Balaji, and K. A. Byrne, "Detecting nutrient deficiency in spruce forests using multispectral satellite imagery," *International Journal of Applied Earth Observation and Geoinformation*, vol. 86, p. 101975, Apr. 2020.

[5] N. Karthikeyan, M. Shashikkumar, and J. Ramanamurthy, "A study on vegetation vigour as affected by soil properties using remote sensing approach," *IEEE*, Nov. 2010.

[6] D. Rolnick, P. Donti, L. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, A. Luccioni, T. Maharaj, E. Sherwin, s. K. Mukkavilli, K. Kording, C. Gomes, A. Ng, D. Hassabis, J. Platt, and Y. Bengio, "Tackling climate change with machine learning," 06 2019.

[7] C. Arsenault, ""Only 60 Years of Farming Left If Soil Degradation Continues"," Dec. 2017.

[8] N. Alexandratos and J. Bruinsma, "World agriculture towards 2030/2050: the 2012 revision," June 2012.

[9] I. P. on Climate Change and O. Edenhofer, eds., *Climate change 2014: mitigation of climate change: Working Group III contribution to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. New York, NY: Cambridge University Press, 2014. OCLC: ocn892580682.

[10] D. Rolnick, P. L. Donti, L. H. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, A. Luccioni, T. Maharaj, E. D. Sherwin, S. K. Mukkavilli, K. P. Kording, C. Gomes, A. Y. Ng, D. Hassabis, J. C. Platt, F. Creutzig, J. Chayes, and Y. Bengio, "Tackling Climate Change with Machine Learning," *arXiv:1906.05433 [cs, stat]*, June 2019.

[11] F. Rembold, C. Atzberger, I. Savin, and O. Rojas, "Using Low Resolution Satellite Imagery for Yield Prediction and Yield Anomaly Detection," *Remote Sensing*, vol. 5, pp. 1704–1733, Apr. 2013. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[12] E. Manjula, “A Model for Prediction of Crop Yield,” 2017.

[13] A. Chlingaryan, S. Sukkarieh, and B. Whelan, “Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review,” *Computers and Electronics in Agriculture*, vol. 151, pp. 61–69, Aug. 2018.

[14] M. Wahabzada, A.-K. Mahlein, C. Bauckhage, U. Steiner, E.-C. Oerke, and K. Kersting, “Plant Phenotyping using Probabilistic Topic Models: Uncovering the Hyperspectral Language of Plants,” *Scientific Reports*, vol. 6, pp. 1–11, Mar. 2016.

[15] Y. Ge, J. A. Thomasson, and R. Sui, “Remote sensing of soil properties in precision agriculture: A review,” *Frontiers of Earth Science*, vol. 5, pp. 229–238, Sept. 2011.

[16] Y. Peng, L. Zhao, Y. Hu, G. Wang, L. Wang, and Z. Liu, “Prediction of Soil Nutrient Contents Using Visible and Near-Infrared Reflectance Spectroscopy,” *ISPRS International Journal of Geo-Information*, vol. 8, p. 437, Oct. 2019. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

[17] D. J. Lary, G. K. Zewdie, X. Liu, D. Wu, E. Levetin, R. J. Allee, N. Malakar, A. Walker, H. Mussa, A. Mannino, and D. Aurin, “Machine Learning Applications for Earth Observation,” in *Earth Observation Open Science and Innovation* (P.-P. Mathieu and C. Aubrecht, eds.), ISSI Scientific Report Series, pp. 165–218, Cham: Springer International Publishing, 2018.

[18] L. Wang, J. Yan, L. Mu, and L. Huang, “Knowledge discovery from remote sensing images: A review,” *WIREs Data Mining and Knowledge Discovery*, vol. 10, no. 5, p. e1371, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1371.

[19] A. Karpatne, I. Ebert-Uphoff, S. Ravela, H. A. Babaie, and V. Kumar, “Machine Learning for the Geosciences: Challenges and Opportunities,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, pp. 1544–1554, Aug. 2019.

[20] Radiant Earth Foundation and N. E. S. D. Systems, “Advancing Machine Learning Tools for Earth Science: Workshop Report,” tech. rep., NASA, Washington DC, DC, USA, Jan. 2020.

[21] L. Pereira and N. Nunes, “Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools—A review,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, p. e1265, Nov. 2018.

[22] G. Hart, “Prototype Nonintrusive Appliance Load Monitor,” tech. rep., MIT Energy Laboratory Technical Report, and Electric Power Research Institute Technical Report, Sept. 1985.

[23] N. Batra, J. Kelly, O. Parson, H. Dutta, W. Knottenbelt, A. Rogers, A. Singh, and M. Srivastava, “NILMTK: An Open Source Toolkit for Non-intrusive Load Monitoring,” *Proceedings of the 5th international conference on Future energy systems*, pp. 265–276, June 2014. arXiv: 1404.3878.

[24] N. Batra, R. Kukunuri, A. Pandey, R. Malakar, R. Kumar, O. Krystalakos, M. Zhong, P. Meira, and O. Parson, “Towards Reproducible State-of-the-art Energy Disaggregation,” in *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys ’19, (New York, NY, USA), pp. 193–202, ACM, 2019. event-place: New York, NY, USA.

[25] M. Sharma, “Different types of Remote Sensing - Active and Passive,” Oct. 2019.

[26] NASA, “MODIS Web,” 2013.

[27] NASA, "What is Synthetic Aperture Radar? | Earthdata," 2020.

[28] R. World, "Uses of Electromagnetic Waves | Revision World," 2019.

[29] V. Vaida, J. S. Daniel, H. G. Kjaergaard, L. M. Goss, and A. F. Tuck, "Atmospheric absorption of near infrared and visible solar radiation by the hydrogen bonded water dimer," *Quarterly Journal of the Royal Meteorological Society*, vol. 127, no. 575, pp. 1627–1643, 2001. _eprint: https://rmets.onlinelibrary.wiley.com/doi/pdf/10.1002/qj.49712757509.

[30] ESA, "Observations: Seeing in X-ray wavelengths," 2019.

[31] M. Teke, E. Baseski, A. O. Ok, B. Yuksel, and C. Senaras, "Multi-spectral false color shadow detection," in *Proceedings of the 2011 ISPRS conference on Photogrammetric image analysis*, PIA'11, (Berlin, Heidelberg), pp. 109–119, Springer-Verlag, Oct. 2011.

[32] R. Filgueiras, E. C. Mantovani, D. Althoff, E. I. Fernandes Filho, and F. F. d. Cunha, "Crop NDVI Monitoring Based on Sentinel 1," *Remote Sensing*, vol. 11, p. 1441, Jan. 2019. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[33] P. E-pubs, R. J. Kauth, G. S. Thomas, R. J. Kauth, and G. S. Thomas, *The Tasselled Cap– a Graphic Description of the Spectral-Temporal Development of Agricultural Crops as Seen by Landsat**. Proceedings, 1976.

[34] V. C. F. Gomes, G. R. Queiroz, and K. R. Ferreira, "An Overview of Platforms for Big Earth Observation Data Management and Analysis," *Remote Sensing*, vol. 12, p. 1253, Jan. 2020. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.

[35] R. Allen, "A Gentle Introduction to Machine Learning Concepts," Feb. 2020.

[36] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting Self-Supervised Visual Representation Learning," Jan. 2019. Number: arXiv:1901.09005 arXiv:1901.09005 [cs].

[37] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'96, (Cambridge, MA, USA), pp. 473–479, MIT Press, Dec. 1996.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986. Number: 6088 Publisher: Nature Publishing Group.

[39] G. E. Hinton and R. Zemel, "Autoencoders, minimum description length and helmholtz free energy," in *Advances in Neural Information Processing Systems* (J. Cowan, G. Tesauro, and J. Alspector, eds.), vol. 6, Morgan-Kaufmann, 1993.

[40] S. Wang, G. Azzari, and D. B. Lobell, "Crop type mapping without field-level labels: Random forest transfer and unsupervised clustering techniques," *Remote Sensing of Environment*, vol. 222, pp. 303–317, Mar. 2019.

[41] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep Learning Classification of Land Cover and Crop Types Using Remote Sensing Data," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, pp. 778–782, May 2017. Conference Name: IEEE Geoscience and Remote Sensing Letters.

[42] P. Rodríguez-Veiga, S. Quegan, J. Carreiras, H. J. Persson, J. E. S. Fransson, A. Hoscilo, D. Ziółkowski, K. Stereńczak, S. Lohberger, M. Stängel, A. Berninger, F. Siegert, V. Avitabile, M. Herold, S. Mermoz, A. Bouvet, T. Le Toan, N. Carvalhais, M. Santoro, O. Cartus, Y. Rauste, R. Mathieu, G. P. Asner, C. Thiel, C. Pathe, C. Schmullius, F. M. Seifert, K. Tansey, and H. Balzter, "Forest biomass retrieval approaches from earth observation in different biomes," *International Journal of Applied Earth Observation and Geoinformation*, vol. 77, pp. 53–68, May 2019.

[43] N. Efremova, D. Zausaev, and G. Antipov, "Prediction of Soil Moisture Content Based On Satellite Data and Sequence-to-Sequence Networks," *arXiv:1907.03697 [cs, eess]*, June 2019. arXiv: 1907.03697.

[44] Z. Akhavan, M. Hasanlou, M. Hosseini, and H. McNairn, "Decomposition-Based Soil Moisture Estimation Using UAVSAR Fully Polarimetric Images," *Agronomy*, vol. 11, p. 145, Jan. 2021. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[45] Y.-Q. Song, X. Zhao, H.-Y. Su, B. Li, Y.-M. Hu, and X.-S. Cui, "Predicting Spatial Variations in Soil Nutrients with Hyperspectral Remote Sensing at Regional Scale," *Sensors (Basel, Switzerland)*, vol. 18, Sept. 2018.

[46] Q. Gao, M. Zribi, M. J. Escorihuela, and N. Baghdadi, "Synergetic Use of Sentinel-1 and Sentinel-2 Data for Soil Moisture Mapping at 100 m Resolution," *Sensors*, vol. 17, p. 1966, Sept. 2017. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute.

[47] M. Zribi, C. Andre, and B. Decharme, "A Method for Soil Moisture Estimation in Western Africa Based on the ERS Scatterometer," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, pp. 438–448, Feb. 2008. Conference Name: IEEE Transactions on Geoscience and Remote Sensing.

[48] F. Castaldi, S. Chabrillat, A. Don, and B. van Wesemael, "Soil Organic Carbon Mapping Using LUCAS Topsoil Database and Sentinel-2 Data: An Approach to Reduce Soil Moisture and Crop Residue Effects," *Remote Sensing*, vol. 11, p. 2121, Jan. 2019. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute.

[49] Q. Wang, C. Wu, Q. Li, and J. Li, "Chinese HJ-1A/B satellites and data characteristics," *Science China Earth Sciences*, vol. 53, pp. 51–57, Dec. 2010.

[50] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, Sept. 1995.

[51] Naveedullah, M. Z. Hashmi, C. Yu, H. Shen, D. Duan, C. Shen, L. Lou, and Y. Chen, "Risk Assessment of Heavy Metals Pollution in Agricultural Soils of Siling Reservoir Watershed in Zhejiang Province, China," Sept. 2013. ISSN: 2314-6133 Pages: e590306 Publisher: Hindawi Volume: 2013.

[52] Y. Peng, R. B. Kheir, K. Adhikari, R. Malinowski, M. B. Greve, M. Knadel, and M. H. Greve, "Digital Mapping of Toxic Metals in Qatari Soils Using Remote Sensing and Ancillary Data," *Remote Sensing*, vol. 8, p. 1003, Dec. 2016. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[53] F. Wang, J. Gao, and Y. Zha, "Hyperspectral sensing of heavy metals in soil and vegetation: Feasibility and challenges," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 136, pp. 73–84, Feb. 2018.

[54] N. M. Khan, V. V. Rastoskuev, Y. Sato, and S. Shiozawa, "Assessment of hydrosaline land degradation by using a simple approach of remote sensing indicators," *Agricultural Water Management*, vol. 77, pp. 96–109, Aug. 2005.

[55] A. Bannari, A. M. Guedon, A. El-Harti, F. Z. Cherkaoui, and A. El-Ghmari, "Characterization of Slightly and Moderately Saline and Sodic Soils in Irrigated Agricultural Land using Simulated Data of Advanced Land Imaging (EO-1) Sensor," *Communications in Soil Science and Plant Analysis*, vol. 39, pp. 2795–2811, Nov. 2008. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00103620802432717.

[56] M. M. Taghadosi, M. Hasanlou, and K. Eftekhari, "Retrieval of soil salinity from Sentinel-2 multispectral imagery," *European Journal of Remote Sensing*, vol. 52, pp. 138–154, Jan. 2019. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/22797254.2019.1571870.

[57] C. Snijders, U. Matzat, and U.-D. Reips, ""Big Data" : Big Gaps of Knowledge in the Field of Internet Science," *International Journal of Internet Science*, vol. 7, pp. 1–5, Jan. 2012.

[58] M. T. Chiu, X. Xu, Y. Wei, Z. Huang, A. Schwing, R. Brunner, H. Khachatrian, H. Karapetyan, I. Dozier, G. Rose, D. Wilson, A. Tudor, N. Hovakimyan, T. S. Huang, and H. Shi, "Agriculture-Vision: A Large Aerial Image Database for Agricultural Pattern Analysis," *IEEE*, Jan. 2020.

[59] S. Waqas Zamir, A. Arora, A. Gupta, S. Khan, G. Sun, F. Shahbaz Khan, F. Zhu, L. Shao, G.-S. Xia, and X. Bai, "isaid: A large-scale dataset for instance segmentation in aerial images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 28–37, 2019.

[60] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS)*, 2010.

[61] G. Sumbul, M. Charfuelan, B. Demir, and V. Markl, "BigEarthNet: A Large-Scale Benchmark Archive For Remote Sensing Image Understanding," *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 5901–5904, July 2019. arXiv: 1902.06148.

[62] A. Orgiazzi, C. Ballabio, P. Panagos, A. Jones, and O. Fernández-Ugalde, "LUCAS Soil, the largest expandable soil dataset for Europe: a review: LUCAS Soil, pan-European open-access soil dataset," *European Journal of Soil Science*, vol. 69, pp. 140–153, Jan. 2018.

[63] R. d'Andrimont, A. Verhegghen, M. Meroni, G. Lemoine, P. Strobl, B. Eiselt, M. Yordanov, L. Martinez-Sanchez, and M. van der Velde, "LUCAS Copernicus 2018: Earth Observation relevant in-situ data on land cover throughout the European Union," *Earth System Science Data Discussions*, pp. 1–19, Aug. 2020. Publisher: Copernicus GmbH.

[64] K. Jordahl, J. V. den Bossche, J. Wasserman, J. McBride, J. Gerard, M. Fleischmann, J. Tratner, M. Perry, C. Farmer, G. A. Hjelle, S. Gillies, M. Cochran, M. Bartos, L. Culbertson, N. Eubank, maxalbert, S. Rey, A. Bilogur, D. Arribas-Bel, C. Ren, J. Wilson, M. Journois, L. J. Wolf, L. Wasser, Ömer Özak, YuichiNotoya, F. Leblanc, Filipe, C. Holdgraf, and A. Greenhall, "geopandas/geopandas: v0.6.1," Oct. 2019.

[65] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework," July 2019. arXiv:1907.10902 [cs, stat].

[66] A. Chen, A. Chow, A. Davidson, A. DCunha, A. Ghodsi, S. A. Hong, A. Konwinski, C. Mewald, S. Murching, T. Nykodym, P. Ogilvie, M. Parkhe, A. Singh, F. Xie, M. Zaharia, R. Zang, J. Zheng, and C. Zumar, "Developments in mlflow: A system to accelerate the machine learning lifecycle," in *Proceedings of the Fourth International Workshop on Data Management for End-to-End Machine Learning*, DEEM'20, (New York, NY, USA), Association for Computing Machinery, 2020.

[67] M. Zaharia, A. Chen, A. Davidson, A. Ghodsi, S. A. Hong, A. Konwinski, S. Murching, T. Nykodym, P. Ogilvie, M. Parkhe, *et al.*, "Accelerating the machine learning lifecycle with mlflow.," *IEEE Data Eng. Bull.*, vol. 41, no. 4, pp. 39–45, 2018.

[68] R. A. Stoneback, A. G. Burrell, J. Klenzing, and M. D. Depew, "PYSAT: Python Satellite Data Analysis Toolkit," *Journal of Geophysical Research: Space Physics*, vol. 123, no. 6, pp. 5271–5283, 2018. _eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2018JA025297.

[69] D. Peressutti, "Introducing eo-learn," May 2019.

[70] H. Huhdanpaa, D. H. Hwang, G. G. Gasparian, M. T. Booker, Y. Cen, A. Lerner, O. B. Boyko, J. L. Go, P. E. Kim, A. Rajamohan, M. Law, and M. S. Shiroishi, "Image Coregistration: Quantitative Processing Framework for the Assessment of Brain Lesions," *Journal of Digital Imaging*, vol. 27, pp. 369–379, June 2014.

[71] A. Agrawal and M. R. Petersen, "Detecting arsenic contamination using satellite imagery and machine learning," *Toxics*, vol. 9, no. 12, 2021.

[72] "Multilayer perceptron regressor - scikit learn." https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html. Accessed: 2022-06-16.

[73] G. E. Hinton, "Connectionist Learning Procedures," 1989.

[74] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pp. 61–74, MIT Press, 1999.

[75] "Partial least squares regression - scikit learn." https://scikit-learn.org/stable/modules/generated/sklearn.cross_decomposition.PLSRegression.html. Accessed: 2022-06-16.

[76] I. S. Helland, "Partial least squares regression and statistical models," *Scandinavian Journal of Statistics*, vol. 17, no. 2, pp. 97–114, 1990.

[77] "Gradient boosting regressor - scikit learn." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html. Accessed: 2022-06-16.

[78] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002. Nonlinear Methods and Data Mining.

[79] "Random forest regressor - scikit learn." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html. Accessed: 2022-06-16.

[80] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, pp. 5–32, Oct. 2001.

[81] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature," *Geoscientific Model Development*, vol. 7, pp. 1247–1250, June 2014. Publisher: Copernicus GmbH.

[82] K. Das, J. Jiang, and J. N. K. Rao, "Mean squared error of empirical predictor," *The Annals of Statistics*, vol. 32, pp. 818–840, Apr. 2004. Publisher: Institute of Mathematical Statistics.

[83] M. Shcherbakov, A. Brebels, N. Shcherbakova, A. Tyukov, T. Janovsky, and V. Kamaev, "A survey of forecast error measures," *World Applied Sciences Journal*, vol. 24, pp. 171–176, Jan. 2013.

[84] J. Miles, "R-Squared, Adjusted R-Squared," in *Encyclopedia of Statistics in Behavioral Science*, John Wiley & Sons, Ltd, 2005. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013192.bsa526.

[85] J. Martinez, "Relationship between crop nutritional status, spectral measurements and Sentinel 2 images," *Agronomía Colombiana*, vol. 35, pp. 205–215, May 2017.

[86] A. Suresh, "How to Remove Outliers for Machine Learning?," Oct. 2020.

[87] I. Wier, "Spearman's Rank Correlation- Introduction."

[88] P. Branco, L. Torgo, and R. P. Ribeiro, "A Survey of Predictive Modeling on Imbalanced Domains," *ACM Computing Surveys*, vol. 49, pp. 31:1–31:50, Aug. 2016.

[89] Y. Yang, K. Zha, Y.-C. Chen, H. Wang, and D. Katabi, "Delving into Deep Imbalanced Regression," May 2021. arXiv:2102.09554 [cs].

[90] P. Branco, L. Torgo, and R. Ribeiro, *SMOGN: a Pre-processing Approach for Imbalanced Regression*. Proceedings of Machine Learning Research, Sept. 2017.

[91] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Oversampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, June 2002.

[92] P. Branco, L. Torgo, and R. P. Ribeiro, "Rebagg: Resampled bagging for imbalanced regression," in *Second International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pp. 67–81, PMLR, 2018.

Appendix A

Appendix chapter

Table A.1: Original algorithms' Hyper-parameter Optimization

Algorithm	**Optimal Hyper Parameters**	**Default Values**
	Common Wheat	
Gradient Boosting Regression	Loss: Huber Learning Rate: 0.4 Number of Estimators: 700	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Squared Root Number of Estimators: 176	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Barley	
Gradient Boosting Regression	Loss: Squared Error Learning Rate: 0.25 Number of Estimators: 500	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Log Squared Number of Estimators: 181	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Maize	
Gradient Boosting Regression	Loss: Squared Error Learning Rate: 0.15 Number of Estimators: 1000	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Log Squared Number of Estimators: 179	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Sunflower	
Gradient Boosting Regression	Loss: Huber Learning Rate: 0.2 Number of Estimators: 800	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Squared Error Maximum Features: Squared Root Number of Estimators: 242	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Vineyards	
Gradient Boosting Regression	Loss: MSE Learning Rate: 0.1 Number of Estimators: 600	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Squared Root Number of Estimators: 74	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100

Table A.2: Algorithms' Hyper-parameter Optimization For Log-Scaled data

Algorithm	**Optimal Hyper Parameters**	**Default Values**
	Common Wheat	
Gradient Boosting Regression	Loss: Squared Error Learning Rate: 0.5 Number of Estimators: 100	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Log Squared Number of Estimators: 255	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Barley	
Gradient Boosting Regression	Loss: Squared Error Learning Rate: 0.3 Number of Estimators: 900	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: All Number of Estimators: 211	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Maize	
Gradient Boosting Regression	Loss: Squared Error Learning Rate: 0.5 Number of Estimators: 900	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Log Squared Number of Estimators: 198	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Sunflower	
Gradient Boosting Regression	Loss: Huber Learning Rate: 0.2 Number of Estimators: 600	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Squared Error Maximum Features: Log Squared Number of Estimators: 237	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100
	Vineyards	
Gradient Boosting Regression	Loss: MSE Learning Rate: 0.1 Number of Estimators: 800	Loss: Squared Error Learning Rate: 0.1 Number of Estimators: 100
Random Forest	Criterion: Absolute Error Maximum Features: Squared Root Number of Estimators: 179	Criterion: Squared Error Maximum Features: 1.0 Number of Estimators: 100

Glossary

ensemble An ensemble model is a machine learning model that uses multiple algorithms to achieve a higher predictive performance, generally better than the lone algorithm itself.. 31

LUCAS Land Use and Coverage Area frame Survey is a dataset with more than 20000 soil samples from 25 countries, it consists in spectral measurements and 12 chemical and physical variables for each sample.. 14, 17, 20

orthorectification Is the process of correcting the imagery having in mind the movement of the earth, distortion and equipment calibration.. 11

phenotyping Plant phenotyping is the science of the characterization of the crops which is particularly important for decision support in agriculture.. 3

rasterized The process of transforming a vector image into a pixelized/raster image.. 37

Acronyms

AMR Advanced Microwave Radiometer. 10

ANN Artificial Neural Network. 12

AOI Area of Interest. 37

API Application Programming Interface. 11, 18, 21, 28, 47

As Arsenic. 15

BPNNOK Hybrid Kriging Model. 15

Cr Chromium. 15

CRS Coordinate Reference System. 28

Cu Copper. 15

CV Cross Validation. 23

DBMS Database Management System. 35

DEM Digital Elevation Mapping. 16

DL Deep Learning. 19

EMS Electromagnetic Spectrum. 7

EO Earth Observation. 2, 4, 5, 8, 9, 11, 18, 21, 53

EOD Earth Observation Data. 3, 12

ESA European Space Agency. 5

ESDAC European Soil Data Centre. 17, 36

FAO Food and Agriculture Organization. 2

FDS Feature Distribution Smoothing. 53

FIR Far Infrared. 7

GB Gradient Boosting. 29

GIS Geographic Information System. 18

GNDVI Green Normalized Difference Vegetation Index. 37

GUI Graphical User Interface. 18, 47

HPO Hyper Parameter Optimization. 22

IR Infrared. 43

IRECI Inverted Red-edge Chlorophyll index. 37

ISRO Indian Space Research Organisation. 6

JAXA Japan Aerospatial Exploration Agency. 6

K Potassium. 14

LDS Label Distribution Smoothing. 53

LST Land Surface Temperature. 16

LSTM Long Short Term Memory. 13

MAE Mean Absolute Error. 33, 47

MIR Middle Infrared. 7

ML Machine Learning. 3, 4, 12, 16, 17, 23, 32

MLP Multi-Layer Perceptron. 29

MODIS Moderate Resolution Imaging Spectroradiometer. 5

MRI Magnetic Resonance Imaging. 5

MSE Mean Squared Error. 33

MSI Multispectral Instrument. 9

MWR Microwave Radiometer. 9

N Nitrogen. 14

NASA National Aeronautics and Space Administration. 5

NDSI Normalized Difference Saline Index. 15

NDVI Normalized Difference Vegetation Index. 8, 15, 37

NDWI Normalized Difference Water Index. 37

Ni Nickel. 15

NILM Non-Intrusive Load Monitoring. 3

NILMTK Non-Intrusive Load Monitoring Toolkit. 3

NIR Near Infrared. 7, 10

NN Neural Networks. 15

NPK Nitrogen, Phosphorus and Potassium. 37

NRMSE Normalized Root Mean Squared Error. 33

NUTS Nomenclature of Territorial Units for Statistics. 40

ODC Open Data Cube. 18

OLCI Ocean and Land Colour Instrument. 9

P Phosphorus. 14

Pb Lead. 15

PLS Partial Least Squares. 29

PoC Proof of Concept. 4

RADAR Radio Detection And Ranging. 5

RBF Radial Basis Function. 30

RECHI Red Edge Chlorophyll index. 37

REP Red Edge Position. 37

REPLI Red Edge Position: linear interpolation method. 37

RF Random Forest. 15, 29

RL Reinforcement Learning. 12, 13

RMSE Root Mean Squared Error. 33, 46, 47

RNN Recurrent Neural Network. 13

ROI Regions of Interest. 24, 28

SAR Synthetic Aperture Radar. 9, 10

SDE Standard Deviation Error. 33

SI Saline Index. 15

SLR Stepwise Linear Regression. 15

SLSTR Sea and Land Surface Temperature Instrument. 9

SMOGN Synthetic Minority Oversampling With Gaussian Noise. 45

SMOTE Synthetic Minority Oversampling Technique. 45

SOA State-of-the-Art. 12

SOC Soil organic carbon. 14

SSO Sun Synchronous Orbit. 6, 9

SVM Support Vector Machines. 15, 30

SVR Support Vector Regression. 16, 29, 30

SWIR Short Wave Near Infrared. 10

TSTK TerraSense ToolKit. iii, v, 21, 52, 53

UAV Unmanned Aerial Vehicle i.e. Drone. 16

UV Ultra Violet. 10

Zn Zinc. 15