Didier Narciso Dias

Bachelor in Computer Science

Soil Classification Resorting to Machine Learning Techniques

Dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Computer Science and Engineering

Adviser: João Carlos Gomes Moura Pires, Professor, NOVA University of Lisbon

Co-adviser: Bruno Emanuel Da Graça Martins, Professor, University of Lisbon

Examination Committee

Chairperson:	Name of the male committee chairperson
Raporteurs:	Name of a female raporteur
	Name of another (male) raporteur
Members:	Another member of the committee
	Yet another member of the committee

September, 2019

Soil Classification Resorting to Machine Learning Techniques

This document was created using the (pdf)LaTeX processor, based in the "novathesis" template[1], developed at the Dep. Informática of FCT-NOVA [2].
[1] https://github.com/joaomlourenco/novathesis [2] http://www.di.fct.unl.pt

Acknowledgements

In the first place I would like to thank my advisers, Professors João Pires and Bruno Martins, for all the help, guidance and learning experience was provided to me. I am also grateful to the rest of the MORENA research group, both the professors and the students, and Luís de Sousa by helping me tremendously during this thesis, by providing insight on areas where I had limited knowledge.

I would also like to thank my girlfriend Helena Nobre, for all the help she gave me during this project, for always being interested in my work, proofreading this document many times with me, and always motivating me to follow my dreams and do my best. My family also helped me throughout my whole life, providing everything I needed, be it to study or in general, raising me to become the person I am today, and for always helping me achieve my goals, therefore I thank my father, mother, sister, and my dog.

My friends also helped me throughout my academic journey, namely André Neves, Daniel Henriques, Dinis Cabanas, Ivo Rocha and Pedro Almeida, always ready to help. I also extend these acknowledgements to my best friends outside of college, André Lobo, Fabio Santos and Miguel Santos, for all the help having fun, even in harder times.

This research was supported through Fundação para a Ciência e Tecnologia (FCT), through the project grant with reference PTDC/CCI-CIF/32607/2017 (MIMU), as well as through the INESC-ID (UID/CEC/50021/2019) and NOVA LINCS (UID/CEC/04516/2019) multi-annual funding.

Abstract

Soil classification is the act of resuming the most relevant information about a soil profile into a single class, from which we can infer a large amount of properties without extensive knowledge of the subject. These classes then make the communication of soils, and how they can best be used in areas such as agriculture and forestry, simpler and easier to understand. Unfortunately soil classification is expensive and requires that specialists perform varied experiments, to be able to precisely attribute a class to a soil profile.

This master's thesis focuses on machine learning algorithms for soil classification mainly based on its intrinsic attributes, in the Mexico region. The data set used contains 6 760 soil profiles, the 19 464 horizons that constitute them, as well as physical and chemical properties, such as pH or organic content, belonging to those horizons.

Four data modelling methods were tested (i.e., standard depths, n first layers, thickness, and area weighted thickness), as well as different values for a k-Nearest Neighbours imputation. A comparison between state of the art machine learning algorithms was also made, namely Random Forests, Gradient Tree Boosting, Deep Neural Networks and Recurrent Neural Networks.

All of our modelling methods provided very similar results, when properly parametrised, reaching Kappa values of 0.504 and an accuracy of 0.554, with the standard depths method providing the most consistent results. The k parameter for the imputation showed very little impact on the variation on the results. Gradient Tree Boosting was the algorithm with the best overall results, closely followed by the Random Forests model. The neuron based methods never achieved a Kappa score over 0.4, therefore providing substantially worse results.

Keywords: Soil Classification, Soil Properties, Ensemble Learning, Neural Networks, Gradient Tree Boosting, Random Forests, World Reference Base, Machine Learning

Resumo

A classificação de solos é o ato de resumir a informação sobre um perfil do solo em uma única classe, da qual é possivel inferir várias propriedades, mesmo com a ausência de conhecimento sobre a área de estudo. Estas classes fazem a comunicação dos solos e de como estes podem ser usados, em áreas como a agricultura e silvicultura, mais simples de perceber. Infelizmente a classificação de solos é dispendiosa, demorada, e requer especialistas para realizar as experiências necessárias para classificar corretamente o solo em causa.

A presente tese de mestrado focou-se na avaliação de algoritmos de aprendizagem automática para o problema de classificação de solos, baseada maioritariamente nos atributos intrínsecos destes, na região do México. Foi utilizada uma base de dados contendo 6 760 perfis de solos, os 19 464 horizontes que os constituem, e as propriedades químicas e físicas, como o pH e a percentagem de barro, pertencentes a esses horizontes.

Quatro métodos de modelação de dados foram testados (standard depths, n first layers, thickness, e area weighted thickness), tal como diferentes valores para uma imputação baseada em k-Nearest Neighbours. Também foi realizada uma comparação entre algoritmos de aprendizagem automática, nomeadamente Random Forests, Gradient Tree Boosting, Deep Neural Networks e Recurrent Neural Networks.

Todas as modelações de dados providenciaram resultados similares, quando propriamente parametrisados, atingindo valores de Kappa de 0.504 e *accuracy* de 0.554, sendo que o métdodo *standard depths* obteve uma performance mais consistente. O parâmetro *k*, referente ao método de imputação, revelou ter pouco impacto na variação dos resultados. O algoritmo Gradient Tree Boosting foi o que obteve melhores resultados, seguido de perto pelo modelo de Random Forests. Os métodos baseados em neurónios tiveram resultados substancialmente piores, nunca superando um valor de Kappa de 0.4.

Palavras-chave: Classificação de Solos, World Reference Base, Aprendizagem Automática, Aprendizagem Conjunto, Redes Neuronais, Gradient Tree Boosting, Random Forests

Contents

List of Figures xiii

List of Tables xvii

Acronyms xix

1 Introduction 1
- 1.1 Motivation and Context . . . 1
- 1.2 The Problem . . . 2
- 1.3 Contributions . . . 3
- 1.4 Document Structure . . . 3

2 Fundamental Concepts 5
- 2.1 Soil Classification . . . 5
  - 2.1.1 Profiles and Horizons . . . 5
  - 2.1.2 Soil Classification Systems . . . 6
- 2.2 Automatic Classification . . . 7
  - 2.2.1 Classification and Regression Trees . . . 7
  - 2.2.2 Random Forests . . . 8
  - 2.2.3 Gradient Tree Boosting . . . 9
  - 2.2.4 Artificial Neural Networks . . . 10
  - 2.2.5 Recurrent Neural Networks . . . 11
- 2.3 Validation . . . 12
  - 2.3.1 Confusion Matrix . . . 12
  - 2.3.2 Accuracy and Kappa . . . 14

3 State of the Art 17
- 3.1 SoilGrids250m . . . 17
- 3.2 Other Digital Soil Mapping Studies . . . 18
- 3.3 Punctual Experimentations . . . 20
- 3.4 Conclusion . . . 21

4 Data 23
- 4.1 Description and Analysis . . . 23

4.2 Data Preparation 31
4.2.1 Cleaning 31
4.2.2 Imputation 31

5 Approach **33**
5.1 Data Modelling 33
5.1.1 *N* First Layers 33
5.1.2 Standard Depth 34
5.1.3 Thickness 35
5.1.4 Area Weighted Thickness 35
5.2 Clustering 36
5.3 Model Training 37
5.3.1 Random Forests 37
5.3.2 Gradient Tree Boosting 38
5.3.3 Deep Neural Networks 39
5.3.4 Recurrent Neural Networks 40

6 Experimentation and Analysis **43**
6.1 Impact of Data Modelling on the Classifiers 43
6.2 Clustering 46
6.3 Classification Algorithms Comparison 50
6.4 Results Examination 53
6.5 Other Experimentations 57
6.5.1 One vs All and Ensemble Models 57
6.5.2 Adding Remote Sensing Data 58

7 Conclusion and Future Work **61**
7.1 Conclusions 61
7.2 Future Work 62

Bibliography **63**

List of Figures

2.1 Examples of soil profiles, taken from the ISRIC website and classified following WRB. On the left a soil identified as *Phaeozem* containing two horizons, a deep black mollic layer abruptly overlying a calcareous substrate, from a location in China. On the right, we have a dark red *Durisol* from South Africa. 6

2.2 Example of a simple classification decision tree. 8

2.3 Visualisation of a simple Gradient Tree Boosting algorithm training [46]. . . 10

2.4 Example of a simple perceptron. 11

2.5 Visualisation of the true positives (TP), false positives (FP),true negatives (TN) and false negatives (FN), in the context of a confusion matrix. 14

4.1 Venn diagram of the number of profiles containing each classification. ST - USDA Soil Taxonomy, WRB - World Reference Base, FAO - FAO Legend. . . 24

4.2 Density map of the complete set of profiles. 25

4.3 Density map of the profiles classified using World Reference Base (WRB). . . 25

4.4 Number of profiles per reference soil group, together with the number of horizons per profile, for the data subset covering Mexico. 26

4.5 Distribution of data points in Mexico, coloured by class, of only the 10 most predominant soil classes. 27

4.6 Distribution of data points in Mexico, by several classes that show a clear geographical preference. 28

4.7 Average maximum depth of the profiles and thickness of the horizons, distributed by class. The numbers below the bars represent the average number of horizons, per profile. 29

4.8 Distribution of six of the soil properties present in our data, throughout the 5 most predominant classes. 30

4.9 Percentage of missing values for each variable, for the layers. 32

5.1 Data representations. On the left the initial data disposition is shown. On the right is an arbitrary method that merges the layers into a single data point, representing the whole profile. 34

5.2 Illustration of 3 of the different data modelling techniques that were used. The letters represent the value, for an arbitrary feature, in each layer. 35

5.3 Example of a Silhouette analysis plot, where the dashed line represents the average value. 37
5.4 Representation of the final architecture that we used for the Recurrent Neural Networks (RNN) algorithm. 40

6.1 Results from the data modelling tests for the different variations of k-Nearest Neighbours k value, as well as the representation parameters, for Random Forests and Gradient Tree Boosting. 44
6.2 Analysis of the different parameters, applied to the representations, averaged over both Random Forests and Gradient Tree Boosting. 44
6.3 Analysis of the k-Means Imputation parameter k, applied to the representations, averaged over both Random Forests and Gradient Tree Boosting. . . . 45
6.4 Visualisation of number of clusters against the within cluster sum of errors (wcss), to help choose the best value for k, for the k-Means algorithm. 46
6.5 Average Silhouette scores for various values of k. 47
6.6 On the left we have the sizes of the different clusters, using k-Means with $k = 6$. On the right, a visualisation of the Silhouette scores also for $k = 6$, value where the average score was higher, i.e. 0.168. In this last graph, the dashed line represents the average silhouette score. 47
6.7 Distribution of each class by the different clusters, using k-Means with $k = 6$. 48
6.8 Geographical distribution of the different clusters, generated using k-Means with $k = 6$. 49
6.9 Accuracy and Kappa values for the best performing prediction models, obtained by the training of each algorithm. 50
6.10 Comparison of the number of correct and incorrect predictions for the trained models on the validation set. 51
6.11 Comparison of the per class accuracy on the validation set. 51
6.12 Time to train for each of our tested algorithms, for their best model. 52
6.13 Feature importance for the standard depth data representation, with an imputation based on $k = 2$, using the Random Forest classifier. lower depth = lower depth, clay value avg = average clay percentage, phaq value avg = average pH measured in H2O, elcosp value avg = electrical conductivity in saturated paste. The numbers next to the variable names represent the depth at which the measurement was taken. 54
6.14 Confusion matrix for the standard depth data representation, with an imputation based on $k = 2$, using the Random Forest classifier, ordered by the most representative classes. 55
6.15 Comparison of the 6 most important featured, arranged by their importance. The chemical properties are averaged over all the layers for each class. 56

6.16 On the left, the accuracy values for each class, as given by the general model. On the right we can see the accuracy values for each specific model created for every class. 58

List of Tables

2.1 Example of a Confusion Matrix. 12

Acronyms

ANN	Artificial Neural Networks.
CART	Classification and Regression Trees.
CNN	Convolutional Neural Networks.
CV	Cross Validation.
DEM	Digital Elevation Model.
DNN	Deep Neural Networks.
DSM	Digital Soil Mapping.
EVI	Enhanced Vegetation Index.
FAO	Food and Agriculture Organisation.
FCT-UNL	Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa.
GPU	Graphical Processing Unit.
GTB	Gradient Tree Boosting.
ISRIC	International Soil Reference and Information Centre.
IST-UL	Instituto Superior Técnico da Universidade de Lisboa.
IUSS	International Union of Soil Sciences.
LSTM	Long Short-Term Memory.
ML	Machine Learning.
MLP	Multi-Layer Perceptron.
MRVBF	Multi-resolution index of valley bottom flatness.
NDVI	Normalized Difference Vegetation Index.

NDWI	Normalized Difference Water Index.
NIR	Near Infrared.
PCA	Principal Component Analysis.
ReLU	Rectified Linear Unit.
RF	Random Forests.
RFE	Recursive Feature Elimination.
RNN	Recurrent Neural Networks.
ROS	Random Oversampling.
RSG	Reference Soil Group.
SVM	Support Vector Machines.
TWI	Topographic Wetness Index.
USA	United States of America.
WoSIS	World Soil Information System.
WRB	World Reference Base.

Chapter 1

Introduction

The soil is very important for humanity and the planet, as it provides materials, filters the water that infiltrates through it, and is home to plants, trees and animals. Not all soils can be used for the same purposes, due to the different possible characteristics that they can have, making the knowledge of the possible combinations of attributes very important.

Nowadays multiple sources provide us with detailed maps of soil properties for the entire world[16, 23], facilitating the access to a location's attributes and to the information on how these may impact the utilisation of the field. However, most of the efforts made in the soil area have focused on using data from satellites to classify the profiles mostly ignoring their properties, only using them as ground truth for validation.

In this chapter, we will describe the motivation for the study, the context, and the contributions to the soil science community that we intend to provide with the dissertation.

1.1 Motivation and Context

Soil attributes are very important for agriculture and many other areas since some of the characteristics of the soil can severely impact the various activities conducted on the land [2, 37, 39]. The changes that these actions have on the soil including its degradation, and how to reduce or prevent it, has also been the target of significant research [10, 34].

Soil properties are usually complex and it may be hard to understand what effects they may have on the different uses of the soil unless advised by an expert. As such, it is common to utilise a classification system that can synthesise and simplify the sharing and understanding of this information. There are many classification methods, two of the most commonly used worldwide being USDA Soil Taxonomy [43] and the WRB [45], this last being the focus of this study.

A big problem in the area is that, to create knowledge about the various properties of

a specific location's soil, experts are needed to dig a vertical section and perform various experiments. This method is very expensive and does not scale easily, leaving much of the land still to be tested [3]. As such, we are seeing an increase in the research of methods for predicting these attributes using automatic data analysis, mainly through the use of Machine Learning (ML) [18, 19], which algorithms perform better and what are the most important features that can lead us to an increase in performance.

Most of the research in the area has been performed using remote sensing data, generally to create maps of a specific region. These studies use soil data as ground truth and, with the help of ML and satellite data, try to predict soil classes for regions where no measures have been made [18, 19]. In the context of this M.Sc. thesis, we want to tackle a different problem than that of other studies, using the properties of the various layers of a soil profile to infer its class. This is done to uncover relations between these properties and the classification, following the WRB system.

This M.Sc. thesis is inserted in a collaboration between Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa (FCT-UNL), Instituto Superior Técnico da Universidade de Lisboa (IST-UL) and the International Soil Reference and Information Centre (ISRIC) which has provided us with a database containing around 48 000 distinct profiles, with about half of them having a classification following the WRB classification system. This database was provided through the WoSIS database and is freely accessible to anyone [1]. A previous publication entitled ***SoilGrids250m: Global gridded soil information based on machine learning*** [18] resorted to an earlier version of this data, focusing on remote sensing covariates and in the prediction of the soil classes and attributes of the whole world. We will be using some of the results and procedures from the aforementioned study for comparison, as well as improving some other areas.

1.2 The Problem

There are many soil classification systems such as WRB and USDA Soil Taxonomy, which are used internationally, but there are also some other methods of classification that are used only at the national level such as the Brazilian Soil Classification System [41], from Brazil. These systems, for the most part, do not have a direct translation to others, hindering the international communication of this type of data, since to understand a classified soil profile we are required to understand the classification system and the possible classes. This is an area that has yet to be studied in-depth, as current the state of the art focuses on creating maps of the classes, using remote sensing data.

Another problem identified during this study is the large number of possible classes on the WRB classification system, further explored in Section 2.1.2. This makes it hard for the models that we are trying to train to give a correct prediction. Such a problem

[1] https://www.isric.org/explore/wosis

is usually attenuated by using smaller regions that contain fewer classes, in the studied state of the art methods [7, 31].

1.3 Contributions

The main contribution from this M.Sc. thesis is to understand how accurately we can predict soil classes, following the WRB classification system, using only the properties of the soil. With this, we intend to improve on the international communication of soil profiles which use different classification systems, by providing a means to generate the WRB class from the measurements taken on the soil, even if such measurements resulted in a classification following another system.

We also plan to research if this method can be improved by resorting to some remote sensing information, as used in the current state of the art methodologies. At the same time, we intend to provide insight into which ML algorithms will have the best results in this problem, giving a detailed comparison between them.

As the last point, we want to see if the chemical and physical information available can be used to obtain groups that reflect the classes or even groups of classes, resorting to clustering methods. This is an attempt to solve the problem of the high number of classes, while also maintaining the separability of the classes based on their properties to the maximum.

1.4 Document Structure

This document is structured in different chapters:

- **Introduction**

  The first chapter of this document, where an introduction to the problem, it's context, and intended results, are briefly and concisely presented.

- **Fundamental Concepts**

  The second chapter gives an introduction of fundamental concepts related to both soil classification and machine learning methods, as well as the validation methodology to compare the results.

- **State of the Art**

  The state of the art methods related to the area of soil classification are presented in the third chapter. Some ideas, methodologies, and results from these studies were used during the development of the solution.

- **Data**

  The fourth chapter report presents an analysis of the data set that was utilised, raising issues found in it. The methods used to treat the data, namely its cleaning and imputation, are also described in this section.

- **Approach**

  In the fifth chapter of this report, we discuss the approach that was taken and how we solved the problem presented. The data modelling methods created, how the algorithms were trained and the clustering methodology are thoroughly described.

- **Experimentation and Analysis**

  The results, as well as an analysis and discussion of them, are presented in the sixth chapter of the dissertation.

- **Conclusions and Future Work**

  The seventh and final chapter of the document describes the conclusions taken from this project and the future work that could possibly improve on the results, as well as provide more insight into the soil classification problem.

Chapter 2

Fundamental Concepts

In this chapter, we will provide a simple explanation of fundamental concepts within the soil research area, followed by a description of the ML algorithms we will be using during the development of this research.

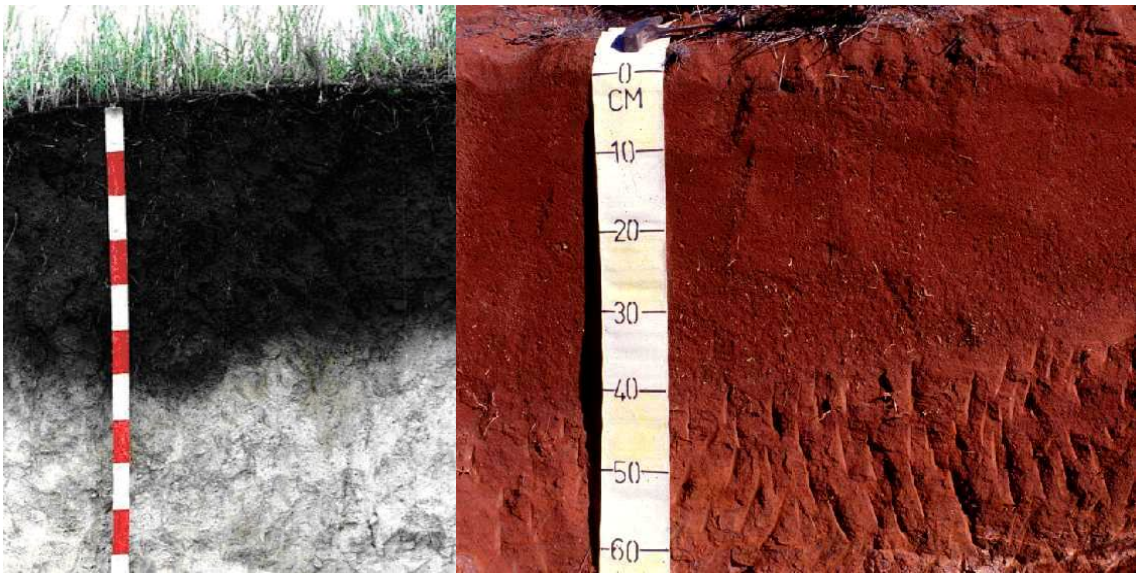
2.1 Soil Classification

Soil is defined as the most superficial layer of the planet, starting at the surface and extending to the depth that plants roots can achieve [43]. For classification purposes, the depth is usually limited to 2 meters unless explicitly stated otherwise [43, 45].

2.1.1 Profiles and Horizons

The soil class is usually attributed to a **profile**, which is defined as being a vertical section of the soil, in a specific location. If we observe a cross-section of the profile in its entirety it is usually possible to discern different layers, as seen in the example of Figure 2.1 *(a)*. These layers can also be called **horizons**, and we will be using both of these names throughout the document. To attribute a classification to the soil we are required to examine the attributes of all the layers belonging to it, and it is common to dig a vertical section of the soil. This section, the profile, will contain all the horizons belonging to the soil in that location, where the number layers it contains is varied.

The horizons are examined using properties observable in the area (e.g., colour or width of the layer), as well as some others assessed in laboratories through scientific tests such as pH and electrical conductivity. After all the layers are thoroughly explored, the correct classification can then be chosen for the profile to which they belong to.



(a) *Phaeozem* (b) *Durisol*

Figure 2.1: Examples of soil profiles, taken from the ISRIC website and classified following WRB. On the left a soil identified as *Phaeozem* containing two horizons, a deep black mollic layer abruptly overlying a calcareous substrate, from a location in China. On the right, we have a dark red *Durisol* from South Africa.

2.1.2 Soil Classification Systems

There are many possible methods to classify the soil and many countries have their own classification [22, 43]. This makes the communication of soil attributes between different countries complicated since most of these systems cannot be directly translated into other classifications, without extensively knowing how both of the systems work.

Some classification systems have emerged with the intent of solving this problem, the most used being the **USDA Soil Taxonomy** [43] and the World Reference Base. USDA Soil Taxonomy is a classification method that was created in the United States of America, with many other countries having adopted it worldwide.

In this thesis, we will be focusing on the WRB, an international soil classification methodology created by the International Union of Soil Sciences (IUSS) which has its origins on the Food and Agriculture Organisation (FAO) Legend [45], while also borrowing some of the concepts presented in the Soil Taxonomy. It tries to provide a simple classification method that uses data observable in the field and to be compatible with other national classifications, not with the intent of replacing them but as a means to simplify international communication. The authors of this system go as far as mentioning that, while it is currently being used for soil mapping, it should be accompanied by a local soil classification system, that can better accommodate the local variance of the soil.

This system classifies profiles in two levels of detail, being the Reference Soil Group (RSG) the first and broader level. An example of a RSG can be seen in Figure 2.1 *(a)* - a Phaeozem which is characterised by having a dark horizon with an elevated concentration

of organic material, between other specifications [45]. The second level is a union between the RSG and several prefix and suffix qualifiers that give a more specific description of the soil and its layers. For example, the profile in Figure 2.1 *(a)* could also be classified as a *Mollic Phaeozem*, where the *Mollic* is an additional qualifier that further characterises the surface horizon, with a dark colour and a high amount of organic matter [45].

In this dissertation we will only work at the RSG level, since it would be very hard with the amount of data available to try to predict the classification into the most specific level, due to the enormous amount of different combinations of the RSG and their qualifiers [23]. Therefore we will be focusing on the 32 currently existing RSG, each of them representing different soil classes whose attributes, specifications, and descriptions are detailed in the **World Reference Base for Soil Resources** [45].

Due to the possible range of values that each attribute in each horizon can take, the number of variables that can be measured, and even the subjectivity of some properties, requiring significant experience in the area, there is not a simple methodology that can be used by non-experts to classify a profile.

2.2 Automatic Classification

In this section, we will give an introduction to the methods that we chose to evaluate during this investigation. This choice was based on the recently reported performances for these classifiers, both in general ML tasks and soil classification research.

2.2.1 Classification and Regression Trees

Classification and Regression Trees (CART) [6] are based on a tree that tries to split the training data as much as possible into different leaf nodes, in which the point being analysed best fits. For classification problems, a leaf corresponds to a class, while in the case of regression, it represents the best value associated with the data point.

Figure 2.2 shows an example of a single classification tree that will classify people on their gender, according to their height and weight. The circles represent the nodes that will filter the features, to lead the data being assessed to the leaf, represented as a rectangle, that best corresponds to it. This method is very flexible, performing both classification and regression, accepting most types of inputs without normalisation nor scaling, and without previous assumptions being inserted into the model [19].

To create these decision trees from the training data, a recursive method is used, in which the data set is split based on a feature and the value that minimises a cost function. For classification purposes this function is usually the Gini impurity:

$$G = \sum_{k} P(k) \times (1 - P(k)) \tag{2.1}$$

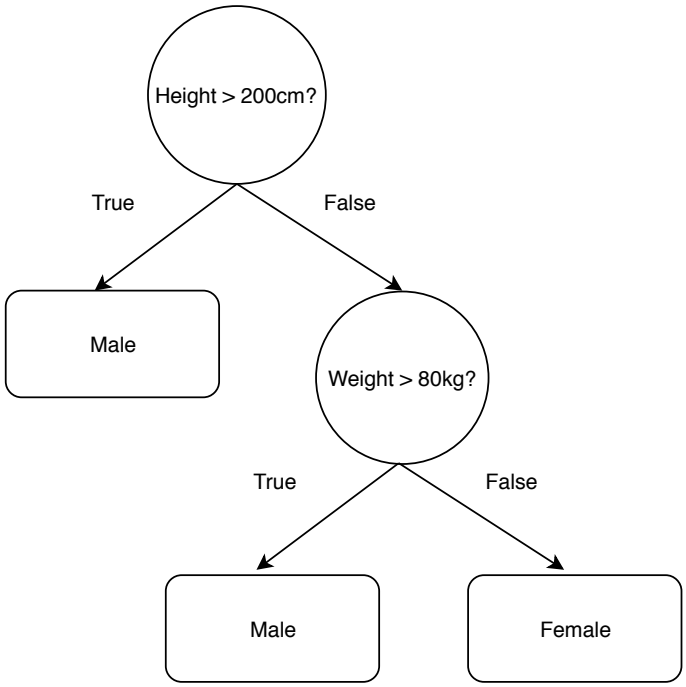


Figure 2.2: Example of a simple classification decision tree.

In the previous Equation 2.1, $P(k)$ is the probability of class k in the data set. As such, the best split possible would be one where the result of the Gini function would be 0, meaning that both classes are perfectly separated.

This method is repeated, splitting the original training set into smaller groups, based on the features and their values, until a stopping condition is encountered, usually the tree reaching a predetermined maximum depth, to prevent overfitting.

Overfitting is a recurrent problem in decision trees since there is no method to know exactly when to stop the training. Overfitting can make it so that the tree adapts in exaggeration to our training data and has problems generalising. To combat this issue we can *prune* the trees after training, by removing the leaves that, if absent, reduce the error on the validation set. Another method widely used to improve this algorithm is the use of a multitude of simple trees, in an *ensemble*. In this thesis, we will use two of those ensemble algorithms, i.e. Random Forests and Gradient Tree Boosting.

2.2.2 Random Forests

Random Forests (RF) is an algorithm which uses an ensemble of decision trees that are trained on a different and random subset of all the features and training data [5].

The method of generating the different subsets of data used in RF is called *bootstrap aggregating*, also known as *bagging*, in which we randomly select data points from the original set without removing them, leaving the possibility of them being selected again. With this method we will obtain different and random groups of the training data, introducing a higher variation on the resulting trees. The data points from the original set that

are not used in the training of the tree are called the *out-of-bag* samples.

Another method of introducing randomness used is the limiting of the features that each tree will have available during its training. This will introduce a higher *bias* since some trees will not have the features that best split the data, providing worse results in the singular trees, but will also give a better opportunity for other features to provide insight on the data. An ensemble using this procedure tends to have better predictions than using the full set of variables since it also results in less overfitting [5].

During training, the model also identifies the most important covariates existent, using the *Gini Importance* [6] which is calculated for each feature and is the sum of the Gini decrease, averaged across all the trees of the ensemble and weighted by the number of samples it splits. With this information, it is possible to better understand the most important variables in the problem, and even remove the least required features, decreasing processing time while maintaining the performance of the model.

2.2.3 Gradient Tree Boosting

Gradient Tree Boosting (GTB) is also an ensemble method based on decision trees, but it takes a different approach than RF. Instead of focusing on adding randomness to make each tree in the ensemble different and independent of each other, this algorithm tries to sequentially improve the results of the ensemble [8].

The algorithm first trains a simple decision tree on the data set and creates a residual error from the mistakes made. After that, a new tree is trained, trying to fit the residual error of the preceding iteration, by providing a higher weight to the data points that were wrongly classified in the previous model. This method is called gradient boosting since the residuals are based on the gradient of the loss function currently being used, which can be specified by the user. We then add this new model to the ensemble, with a weight assigned to it, and repeat this process until we have a classifier that can fit our data to the best extent [8], while trying to not overfit the training data set.

As we can see in Figure 2.3, we start with a very simple tree and then create further trees, also very simple, that all in conjunction can solve the problem. This means that our singular decision trees introduce a lot of bias, making many mistakes since they are more focused on specific data points, while the overall model ends up having a consistently better performance.

To prevent overfitting we can again use pruning after the training, or use another method called *early stopping* in which the training ends when the out-of-bag error consistently increases, while the training error is still decreasing. This means the model is overfitting the training data, and the algorithm stops the training.

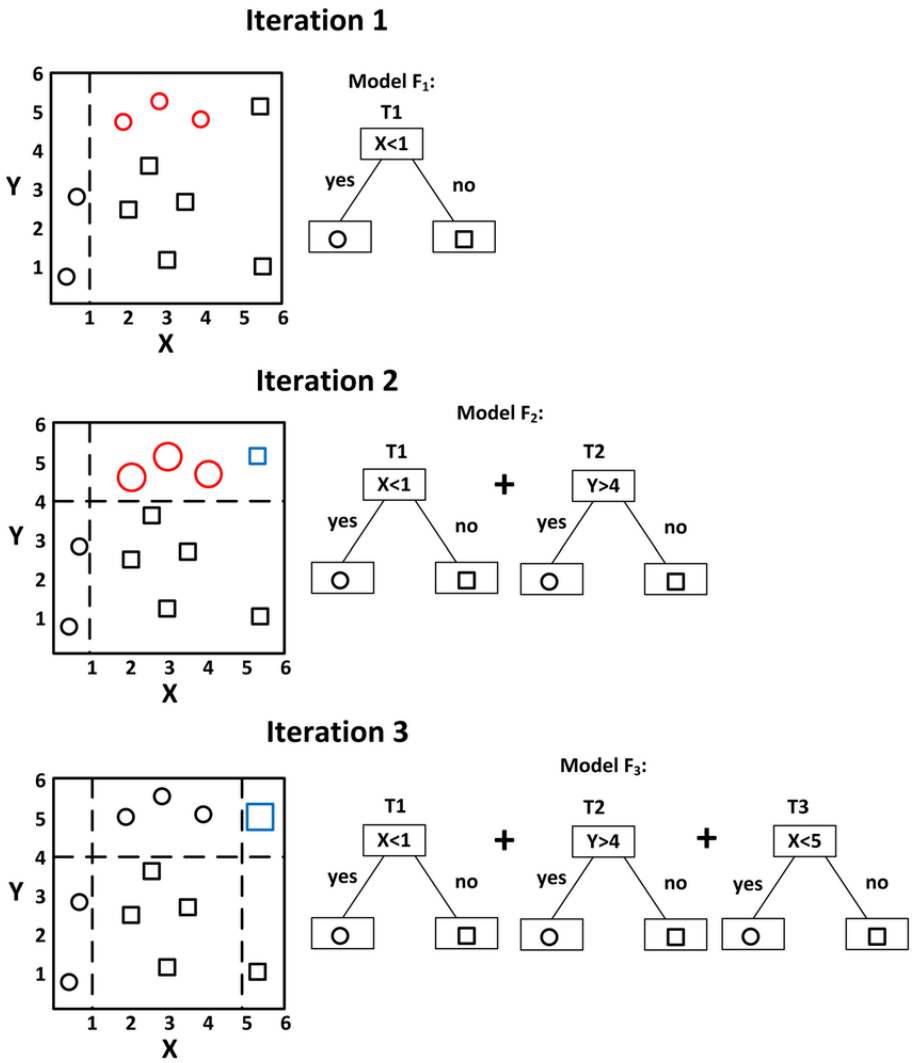


Figure 2.3: Visualisation of a simple Gradient Tree Boosting algorithm training [46].

2.2.4 Artificial Neural Networks

Artificial Neural Networks (ANN) are based on the structure that has been seen in animal brains, where the neurons are connected to each other and receive inputs, process them, and return an output to other cells. Comparatively, the building block of ANN is the *Perceptron* [35], a node that takes a finite number of inputs, each of them having a weight, and sends out a single output based on an activation function. The weights are altered during training so that the perceptron can predict the training set to its best extent.

A single perceptron has some limitations, like not being able to correctly classify non-linearly separable functions [32]. To fix this problem we can use multiple perceptrons organised in different layers, where each perceptron receives all the inputs from the previous layer - a fully connected Multi-Layer Perceptron (MLP) [32]. We can have as many layers between our inputs and our outputs as needed, called the hidden layers. Models with a high count of these are instead called Deep Neural Networks (DNN).

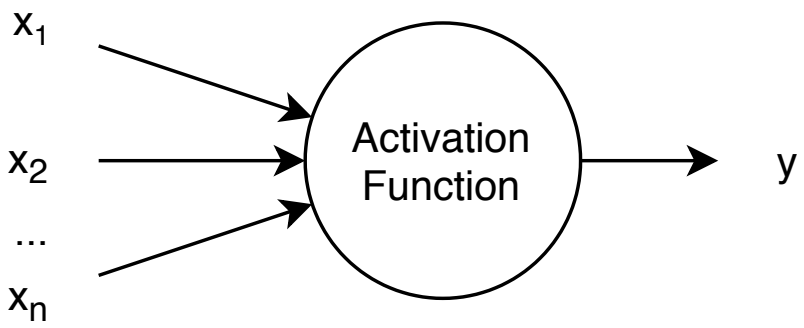


Figure 2.4: Example of a simple perceptron.

To train one single perceptron we would simply adjust the weights of the inputs based on the gradient of the error function. This can not be directly applied to the MLP since we have many nodes in different layers and not all perceptrons contribute equally to the prediction. As such *back-propagation* [44] is used, an algorithm that distributes the error, calculated from the output of the MLP and the actual expected output, through all the perceptrons based on their contribution to the final result.

More recent and advanced neural networks have been developed with different purposes. Convolutional Neural Networks (CNN) is a method widely explored, using convolutions to detect basic patterns mostly in images, which are then used to find more complex patterns. This is mainly used in image analysis due to the high amount of similar patterns in nature that can more easily be learned using this method [26]. Another algorithm being developed is the RNN which is meant to better predict time series problems such as weather and stock values, but can also be used in data that has a sequential structure such as speech and text problems, where the previous words can be used to improve the prediction of the following sentence [17].

2.2.5 Recurrent Neural Networks

Recurrent Neural Networks(RNN) also utilise perceptrons and are related to the previously explained approach. These are made to improve on a problem existent in many of the algorithms presented in this thesis, the lack of memory between inputs [12].

To learn data that is sequenced such as time-series or even our profiles, where the properties of a layer are related to those of the layers near it, we would have to present the entire sequence as a single data point. This will be further explained in Section 5.1. This means that the information about the actual sequence is somewhat lost, resulting in less data for the algorithms to use, which could have otherwise improved the results. RNN, on the other hand, is an algorithm specifically made for this type of sequential data, maintaining a state of the previous elements in the sequence [12].

There are multiple layers that can be used to implement a RNN, in this thesis we will be using the Long Short-Term Memory (LSTM) [20] which has seen a wide usage in the

recent times [17], due to its ability to decrease the impact of the vanishing gradient, where the information is lost on deeper layers, by directly feeding the output from shallow layers onto deeper ones.

2.3 Validation

Validation methods provide us with means to evaluate the algorithms and compare the results between them, and to other studies. As such, the appropriate error measures for our specific problem must be used. A problem that is recurrent in the soil classification area, and that is also present in our data, is the highly imbalanced classes [7, 19, 21]. In a simple example of only 2 classes, in that one appears 90% of the times, a classifier that always chooses the majority class will be right on 90% of his predictions. While this seems like high performance, our model is not trying to understand and predict the data that we have and is instead always answering with the majority class.

To accurately retrieve the metrics to compare our methods and algorithms, we used the Cross Validation (CV) method in which the training data is split into k sets, the model is trained with $k-1$ sets at a time and validated with the data left out. This is repeated until all the k sets have been used for validation. The accuracy of the model will be the average of the k models trained, to account for the randomness of the partition [19]. We used different validation methods throughout our dissertation, namely the Confusion Matrix, Accuracy and the Kappa Statistic.

2.3.1 Confusion Matrix

The confusion matrix is a table layout that compares the class that the model predicted and the actual class, of each data point being assessed. This method is widely utilised in the ML literature and also in soil classification [2, 21, 24, 31].

With this table, we can visualise how our model is performing in each different class, and see if it is confusing one class with another. An example, similar to the one previously given, is presented in Table 2.1 where we can easily assess that our predictor is choosing class A almost always, and we could then act accordingly to fix this problem.

		Predicted Class	
		A	B
Actual Class	A	90	1
	B	10	2

Table 2.1: Example of a Confusion Matrix.

We can use this method to identify issues in our classifiers, especially cases where the models may be confusing classes, by analysing the true positives (TP), false positives (FP)

as well as the true negatives (TN) and false negatives (FN). In Figure 2.5 [1] we can see how these values can be examined in a confusion matrix. When analysing the results of a specific class, the best scenario would be that where we have a large number of true positives, in which we are correctly predicting the class, as well as true negatives, where we are accurately determining that a specific instance is not from the class being analysed.

On the other hand, if we have a large amount of false positives, our classifier is then wrongly classifying other classes as the one being analysed. There is also the case of false negatives, where an instance that should be classified with the current label, is incorrectly classified as another class. We expect both of these problems in our experiments since they are very recurrent in unbalanced data sets, where the minority classes tend to be labelled as the majority classes, thus increasing the FP and the FN amount. The confusion matrix is then a very good method to examine our results, since, from the information we extract, we may implement some methods to reduce the amount of FP and FN, such as data balancing and sampling, as necessary.

One problem with this matrix is that when we have numerous classes to predict, as is our case with the 32 possible RSG, it becomes very big and confusing. A procedure used to combat this is to create a simpler matrix in which we will only have two possibilities, the data point either belongs to the class or it does not, simplifying the examination of a single class, but making us have one table for each possible classification.

While a confusion matrix is a good solution to compare the performance of different algorithms in each class, it is usually hard to perform a simple and fast comparison between them. As such it is usual to use some other metrics to perform such evaluation.

[1] Image from https://stackoverflow.com/questions/50666091/true-positive-rate-and-false-positive-rate-tpr-fpr-for-multi-class-data-in-py

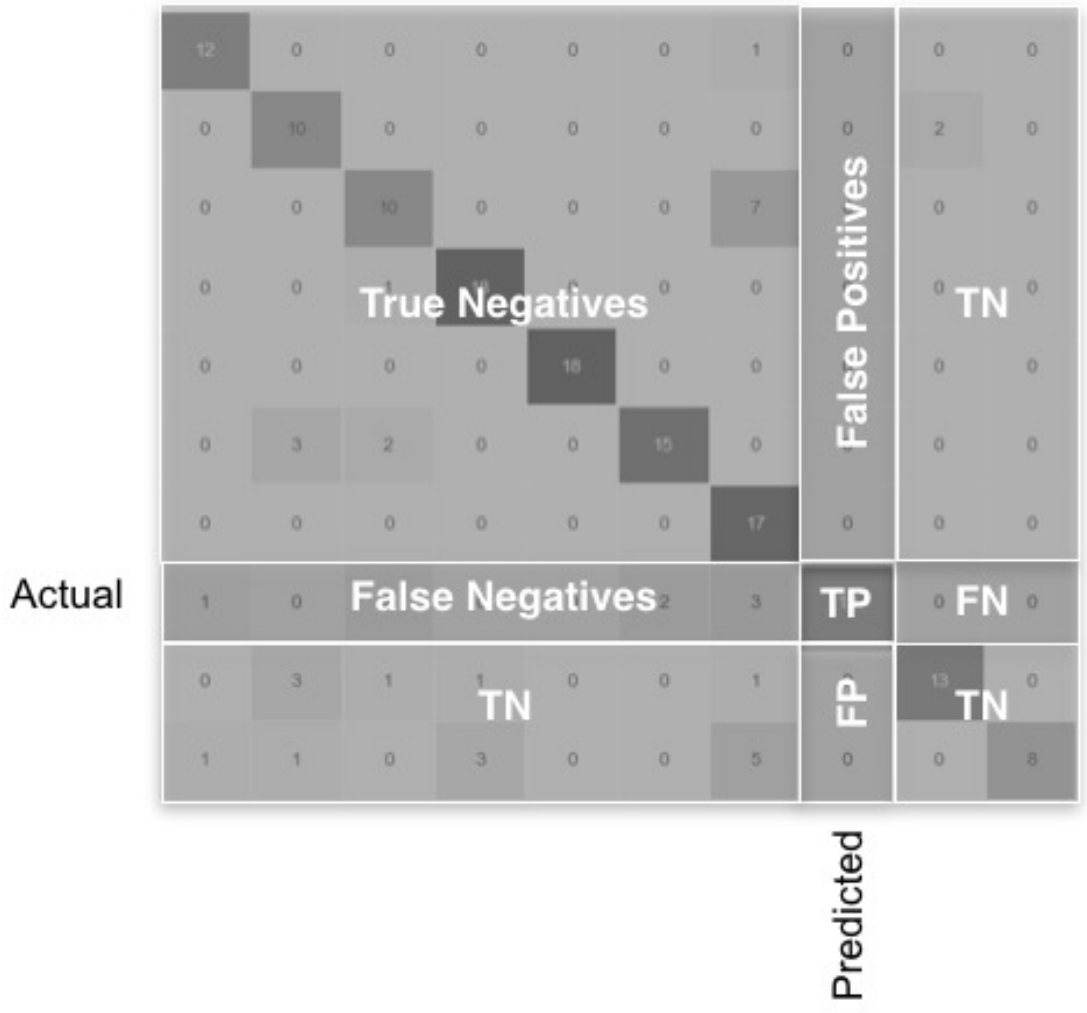


Figure 2.5: Visualisation of the true positives (TP), false positives (FP),true negatives (TN) and false negatives (FN), in the context of a confusion matrix.

2.3.2 Accuracy and Kappa

The accuracy is a metric that is widely used throughout the general machine learning projects and also in soil science [2, 19]. It is described as the fraction of predictions that were correctly classified by our model and is calculated as shown in Equation 2.2.

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \tag{2.2}$$

This value then informs us of how many of the predictions were correct. This has many useful applications but has some glaring flaws if the results are not carefully analysed. The accuracy value for the example in 2.1 is 0.89, a seemingly large value but as we have seen it might be equivalent to very bad results, depending on the problem that we are treating.

The Kappa value k, on the other hand, provides insight into the performance of a classifier while accounting for chance agreement [13]. This is very important to take into account when the classes of the data being analysed are imbalanced since a high accuracy could be the result of classifying all the observations as being from the majority class, as seen in the example given in the previous Section 2.3.1 [7].

This provides us with a fast method to compare different algorithms since it is a single value that can be easily attained and observed. The defining function of k is shown in Equation 2.3, where p_o is the observed agreement and p_e represents chance agreement.

$$k = \frac{p_o - p_e}{1 - p_e} \tag{2.3}$$

The observed agreement is the total number of instances where the model correctly predicted the class, divided by the total number of observations. By using the example presented in Table 2.1 we would have:

$$p_o = \frac{90 + 2}{90 + 10 + 1 + 2} \approx 0.89 \tag{2.4}$$

To calculate the expected accuracy we need to sum all the instances where the classifier predicted one class, do the same for the actual class of the point, multiply these values and divide the result by the total number of instances. In our example, for class A this would result in $\frac{(90+1)*(90+10)}{90+10+1+2} \approx 88.35$. This is done for all the classes (0.35 for class B) present in our problem and then the results are summed and again divided by the total number of instances. Again, on the example, we would have $\frac{88.35+0.34}{103} \approx 0.86$.

As such, our kappa value would be:

$$k = \frac{0.89 - 0.86}{1 - 0.86} \approx 0.21 \tag{2.5}$$

We will be using this metric to compare our results, considering a k higher than 0.80 to be a very strong classifier, between 0.80 to 0.40 as a moderate agreement, and lower than 0.40 as poor agreement, resulting in a near-random model. A negative number for this value would mean that our model is worse than a random classifier. These classification thresholds have been used extensively in other studies [7, 21].

Chapter 3

State of the Art

Automated soil classification is an area that has seen a huge increase with the availability of free and easily accessible remote sensing data, as well as with the recent advances in many ML algorithms. In this chapter, we will show the results of the research, made during the preparation of the dissertation, of the state of the art. First, we introduce the *SoilGrids250m* project, since it resorted to the same data set that is we are using for this dissertation, and due to its results being different from the remaining state of the art, creating predictions for the whole world. We then introduce other studies in the Digital Soil Mapping (DSM) area, which resort to machine learning methods. Finally, we go over some more specific studies, that might be interesting in the context of this work.

3.1 SoilGrids250m

ISRIC is an international institution which has as main objective to provide reliable and easily accessible information on the soil to help address some problems such as its misuse and degradation. In its website[1] it is possible to access digitised maps, download information on soil (similar to our database), and access the results from the SoilGrids project.

SoilGrids[2] is a resource that shows us the predictions of various soil attributes (e.g. organic content or pH) for the whole world, using a resolution of 250 meters. It also provides its classification following WRB and the Soil Taxonomy. The work that went into the development of this tool is described in a previous publication [18].

Around 150,000 soil profiles, their horizons, and properties were used as labels in the training of the models. This is an earlier version of the data set that we will be using. This

[1] https://www.isric.org/
[2] https://soilgrids.org

data is obtained through a collaboration between ISRIC and several other institutions that provided the data, which was then treated and standardised. Some regions where there was little if any information about the soil, and where the properties are homogeneous and predictable, such as deserts and glaciers, had some pseudo-information added so that the classifiers can more correctly predict their attributes.

The covariates used were all based on remote sensing data, totalling 158 different features. These variables were selected with the intent of representing soil formation factors such as the temperature, moistness, and vegetation cover, between others.

The ML algorithms used were Random Forests, Gradient Boosting and Multinomial Logistic Regression. To find the best parameters for each classifier, smaller subsets of the full data were used, to reduce the training time in these experiments, where 5-10% of the total amount of points were chosen randomly. The model was then trained multiple times in this data set and, using cross-validation, the best parameters were chosen and used in the training of the complete model, using all the data available. For each value, more than one model was trained, being the final prediction a weighted average based on model accuracy.

The best covariates depended greatly on what was trying to be predicted. For the WRB classification, the features that most contributed to the prediction were precipitation, elevation from a Digital Elevation Model (DEM), Enhanced Vegetation Index (EVI), mean monthly temperatures, and mean monthly MODIS Near Infrared (NIR) band reflectance. The results, for the WRB classification, achieved a weighted kappa value of 0.42 and an average accuracy on the out of bag test set between 0.20 to 0.28.

The authors conclude that to improve the accuracy of the predictions, as, in most ML projects, the quantity and quality of the data should be improved, especially in under-represented areas of the world. They also admit that there will be a higher bias in the areas with fewer points since the sampling locations are located largely on agricultural areas, where the investment of expert evaluation is considered worthwhile.

3.2 Other Digital Soil Mapping Studies

Several studies have focused on the comparison of various ML algorithms in the DSM problem. In this section, we will provide insight into three of papers speciallized in this area and their results.

Brungard et al.[7] compared 11 different models for the classification of soil profiles, following the Soil Taxonomy system. This study was made in three different semi-arid regions of the United States of America (USA), i.e. New Mexico with 10 different classes, Utah with 15, and 5 classes Wyoming. For validation purposes, a total of around 450 soil profiles were dug out and classified by experts. To compare the different algorithms, the Kappa statistic was the metric used, due to it taking chance agreement into account.

Random Forests were consistently the better classifier in all the three regions that were studied, closely followed by Support Vector Machines (SVM) using a Radial-basis

kernel. The researchers saw a direct correlation between the increase of the number of classes and the imbalance of their distribution and the decrease of accuracy between the regions. This means adding more classes to the problem, as well as using imbalanced classes, resulted in poorer results. The authors, therefore, recommend balanced data sets with few soil classes for higher performance.

Another focus of this study was in the method of choosing the covariates to use during training. Three different methods were used: (i) covariates chosen by soil experts, (ii) the aforementioned set including 113 additional covariates, and a final set, (iii) based on all the variables used, refined with the help of automated feature selection. The features present in these sets mostly originated from remote sensing data such as elevation and slope, vegetation indexes, and wetness indexes.

The best results were achieved when using the (iii) set, which resorted to Recursive Feature Elimination (RFE) to reduce the number of features. A Kappa value of 0.53 was the best result achieved in the region with the least classes i.e. Wyoming, while New Mexico had 0.32 and, in the Utah region, with the largest amount of classes, the best Kappa value stayed at 0.19.

Heung et al. [19] performed DSM for classification using the Soil Taxonomy, resorting to 10 distinct algorithms in a region of Canada, to classify 20 possible classes. A total of 20 covariates were used in the training of these models. The covariates were chosen based on a larger set of variables which was then filtered using Principal Component Analysis (PCA), they represent the climate (e.g. surface temperature in summer and winter), the vegetation (e.g. Normalized Difference Vegetation Index (NDVI), Normalized Difference Water Index (NDWI)) and the topography of the area studied (e.g., elevation, curvature).

For validation purposes, 262 legacy soil data points were used, meaning profiles previously measured by experts. The authors point out a great imbalance in the data where, of the 20 total great groups observed in the region, two of them accounted for more than 60% of all the points, and some classes being responsible for less than 1% of the area. As in the previous study, the authors found RF to be the most accurate classifier, closely followed by Radial-basis SVM and CART with bagging. While SVM had very good accuracy, the study points the challenge that occurred when trying to find the best parameters to train this algorithm, due to the large processing time required. It is to be noted that the validation and comparison methods used in this study do not take into account chance agreement, while the class imbalance is very noticeable, meaning that the accuracy might be biased towards the majority classes. The authors do notice that this might be a problem in the k-Nearest Neighbours algorithm, which had a high accuracy while providing seemingly random predictions.

More recently, *Meier et al.* [31] also performed such a comparison, in this case using 8 different algorithms. A single region, a tropical mountainous zone in Brazil, was the focus of this paper, containing 6 different classes. Similar to *Brungard et al.*, this study also only used actual soil samples for validation purposes, specifically around 140 profiles following the Brazilian National classification system.

Of the initial 73 covariates obtained for training purposes, including orbital images, DSM, spectral images, between others, only ten of them were used. These variables were filtered by removing those who had a large correlation with others and then by resorting to RFE. The most important features are related to the soil genesis and the climate, including a Multi-resolution index of valley bottom flatness (MRVBF), rainfall precipitation, annual temperature range, and the Topographic Wetness Index (TWI).

The Kappa index was used for comparison, in which the better performing classifiers where RF, Ada Boost and Gradient Tree Boosting. The worst performing model was the SVM using a polynomial kernel, although all the algorithms used had very similar results.

3.3 Punctual Experimentations

Since there are several recurring problems in the use of machine learning as a method of classifying soil profiles, related to the imbalance of the different classes, several studies have focused on this area.

Heung et al. [19], described in the previous section, also compared 4 different methods of data sampling:

- **Equal class**, where an equal number of samples is chosen for each class.
- **By-polygon**, where the area is divided in polygons, each of them containing the same amount of points.
- **Area weighted**, in which the number of samples is a proportion of the classes' extent in the area studied.
- **Random oversampling applied to the area-weighted sampling**, where the minority classes are sampled randomly with repetition until all the groups have the same size.

The authors concluded that the best method for their case study was the area-weighted approach, which saw a significant increase in the performance of the models that were studied. Random Oversampling (ROS) used with conjunction with area-weighted resulted in a minor increase in accuracy over the previous method, but the authors realised it may not be worthwhile to use it on larger data sets due to the amount of additional processing time required, especially given the small improvement.

Another more recent study by *Hounkpatin et al.* [21] explored a pruning method in which the majority class had a percentage of its observations removed. This was made by first assessing the major variable, i.e. the covariate with the most importance for the classification, using RF. Consequently, a range of the values from the selected variable is chosen, so that the instances that fall outside it, in a cumulative percentage of the major variable(s), are excluded. In the study, the best results were attained by removing all points lower than 5% and higher than 95% of the cumulative percentage of the major

variable. This method yielded a very good increase in accuracy, presumably by reducing noisy data and outliers, as well as providing a better balance between the classes.

The authors also analysed the use of ROS in conjunction with pruning but, just like *Heung et al.*, found very little improvement. The authors explain that since there is a huge disparity between the number of instances in the majority classes compared to the minority, by using the same data points multiple times, there is no new information being generated, and may instead introduce overfitting to those repeated instances.

In a more recent publication, *A. Sharififar, et al.* [38] conducted a study on using over- and under-sampling in soil classification, as a means to decrease the issues of the imbalanced data sets that usually occur in this area. This study was performed in a region of about 12 000 ha, in the northwest of Iran, and used 452 soil profiles that were measured for this study and classified following the USDA Soil Taxonomy. This soil analysis resulted in 8 possible classes, where their distribution is highly unbalanced, being that the majority class contains 160 profiles, over 35% of the data, while there are two classes which account for less than 10 instances.

To classify these profiles three different algorithms were used, i.e. Random Forests, Decision Trees and Multinomial Logistic Regression, using remote sensing data from digital elevation models, at a 32m resolution. These algorithms were trained on two data sets, the first containing the original information and, the second, obtained as a result of the re-sampling of the data. This was made by under-sampling the majority classes and over-sampling the classes for which the least data was available.

The balancing of the data set proved to overcome some problems in the predictions, in the sense of maintaining the minority classes in the final models. It, however, decreased the overall accuracy in the validation data set. The authors also state that, for their specific problem, the Decision Tree algorithm provided the best results, and was also the one to better perform regarding the data re-sampling.

3.4 Conclusion

All of the studies that we researched resort to remote sensing data to create the predictions, be it classes or values for specific attributes in the soil. Actual measurements of the soil are mainly used for both validation and as ground truth for the training of the models. This is used with the intent of being able to generalise the patterns found in the data to the whole world since remote sensing data is readily available for the entirety of our planet. As such, there is not much research in the area that we will be focusing on, i.e. the prediction of soil classes resorting mainly to the attributes of said soil.

Tree-based algorithms are the approaches that have seen the most use in the soil classification area, as well as having a consistently high performance. Another constant in these projects was the imbalance of the classes since the regions covered by the different types of soil are wildly different.

Chapter 4

Data

The data we used was provided by ISRIC, through the World Soil Information System (WoSIS) database, and is the result of a collaboration between various national and international agencies. For more information on how the data was treated and compiled see the **SoilGrids250m** [18] paper, where the process is thoroughly explained.

In order to find patterns regarding the features and the classifications in our data, an analysis was performed on the information available, as well as the quality of the dataset, to find the problems that must be treated to improve the performance of the models.

4.1 Description and Analysis

The dataset can be separated in two levels:

- **Profiles** - containing information about its location and class following various classification systems.
- **Horizons** - with a reference to the profile they belong to, and all of the physical and chemical measurements made on that layer.

We have 48 342 total soil profiles spread all over the world, consisting of 237 007 horizons. However, not all of this data is classified using WRB, the classification we will be using for this thesis. As seen in Figure 4.1 we have three different soil classifications, namely WRB [45], Soil Taxonomy [43] and FAO Legend [16]. We can see in Figure 4.1, by summing the portions that refer to data using WRB, that only 24 284 of the profiles have a WRB class attributed, while most of the remaining profiles follow the **Soil Taxonomy**.

Many of the data points were examined before the creation of WRB and as such they were classified following FAO's classification, which is the root of WRB [45]. As such, it is possible to directly translate the class from one system to the other. This process was

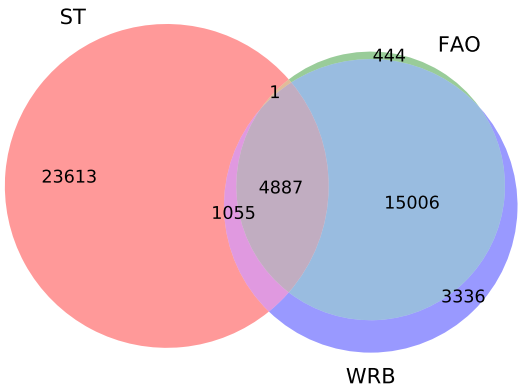


Figure 4.1: Venn diagram of the number of profiles containing each classification. ST - USDA Soil Taxonomy, WRB - World Reference Base, FAO - FAO Legend.

used during the creation of the data set [18], which is why we see such a large number of points classified using both WRB and FAO Legend classifications. Unfortunately, there is no such conversion from USDA Soil Taxonomy to WRB, and therefore we will not be able to use the soil profiles classified only by that system during the training of our algorithms.

In Figures 4.2 and 4.3 two density maps are presented, one with all the profiles, and the second containing only of those classified using WRB. Some areas, such as the Sahara Desert and the polar regions, are also not well represented due to their inaccessibility and lack of interesting data, since their soil is mostly consistent, predictable, and does not have many possible uses [18].

By filtering our data to find only the profiles classified following the WRB we end up excluding a large portion of the data points that were located in the United States of America, which mostly follow the Soil Taxonomy.

Since this dataset is a collection of data from different countries, the information is not perfectly consistent, due to the difference between the measuring methods used, and even the variables that are analysed. As such, we decided to focus our study on the Mexico data subset, due to its high density of consistent measurements, as well as keeping a large number of different classes, 24 out of the total 32 RSG.

Analysing Figure 4.4 we can see that we are dealing with a highly imbalanced dataset where, while *Regosols* accounts for more than 15% of the data points, there are 13 classes that contain less than 2% of the total profiles. The data imbalance may be a problem, as

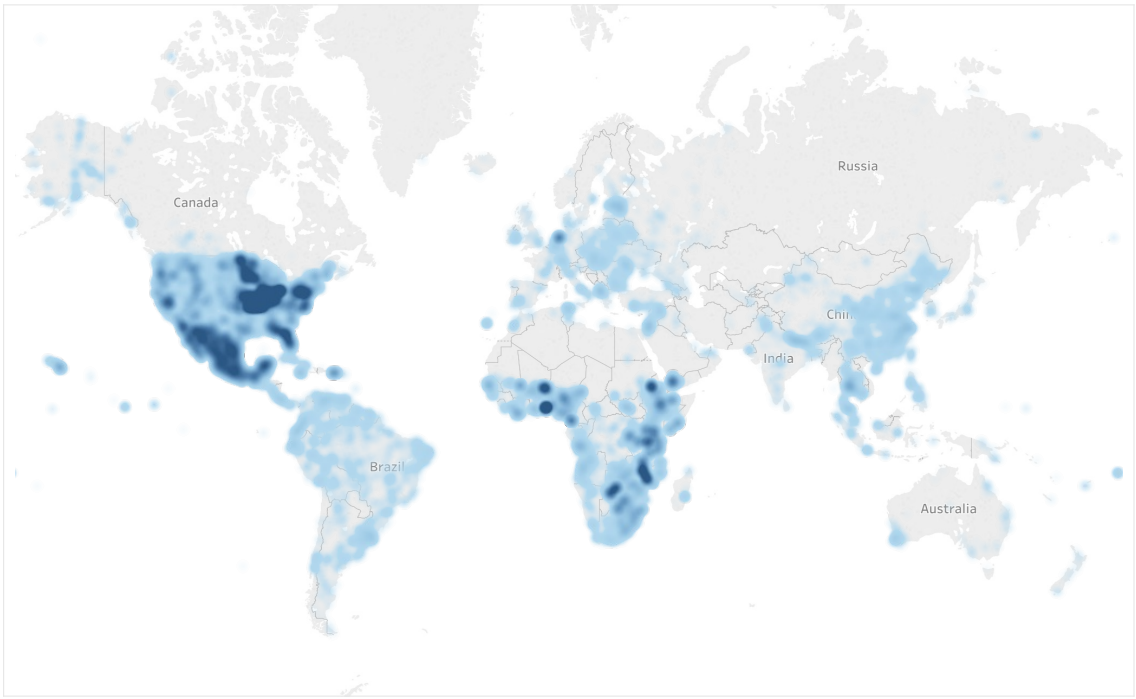


Figure 4.2: Density map of the complete set of profiles.

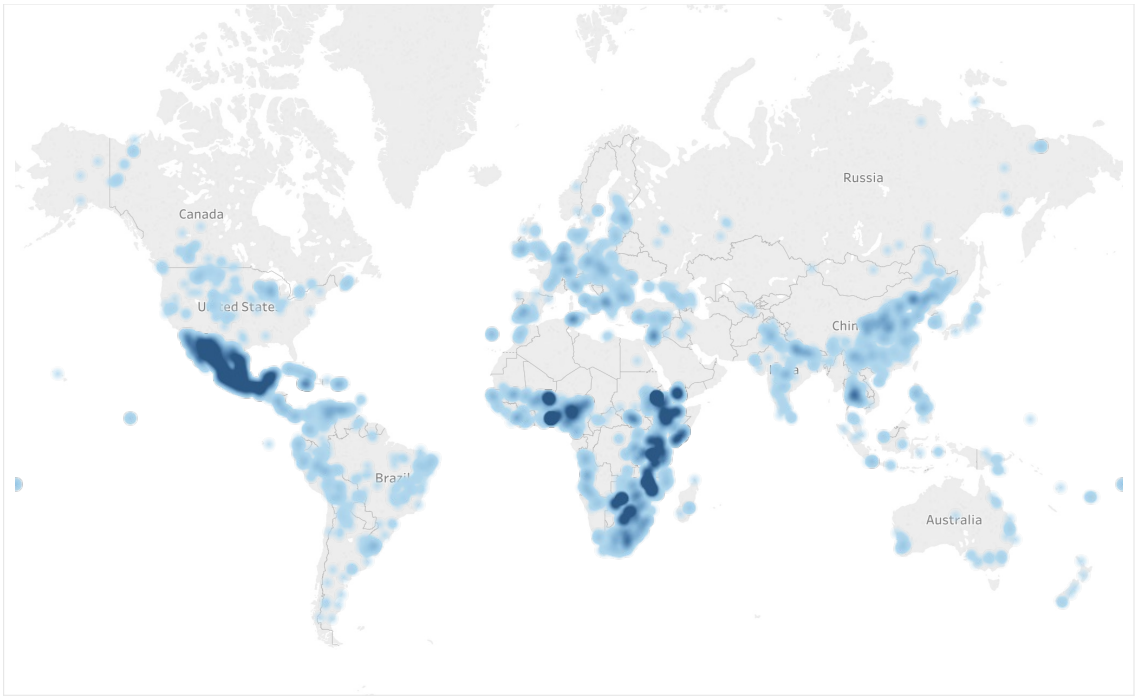


Figure 4.3: Density map of the profiles classified using WRB.

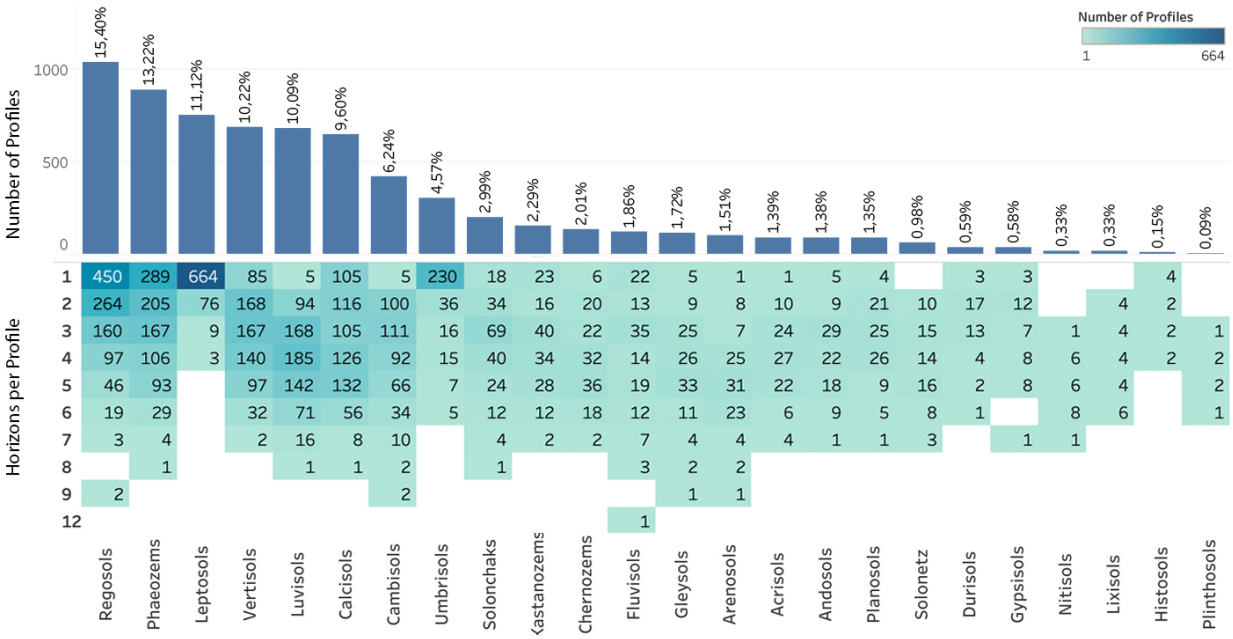


Figure 4.4: Number of profiles per reference soil group, together with the number of horizons per profile, for the data subset covering Mexico.

seen in other studies [19, 29], leading the models to wrongful conclusions in which they prefer to choose the classes that represent the majority of the data points. This problem, and methods to improve it will be discussed in Section 4.2.

The profile is constituted by a variable number of horizons and, as we can observe in Figure 4.4, most classes tend to have 3 to 5 layers. However, some classes, such as the *Leptosols*, tend to have a single layer, meaning that the number of horizons seems to impact the class in some manner.

In Figure 4.5 we can again see the density and spatial homogeneity of our data, in Mexico. We can also discern that there is a relation between some classes and their geographical location. This can be especially seen in Figure 4.6 where the data points of several classes are presented by their location. It is easy to see that these specific soil variations have preferred geographical positions. For example, the *Luvisols* tend to appear near the western shore of the country, while the *Gleysols* are mostly concentrated on the south-east.

Analysing the average depth of the profiles for each class, presented in Figure 4.7, we can immediately see a variation between classes. The number of horizons also rises together with the depth, meaning that a deeper profile tends to have a larger number of layers. This relation is also true with the average thickness of the horizons, getting larger with deeper profiles. It is also possible to understand that most of our profiles are, on average, less than 100cm deep and contain, again on average, 3 to 4 horizons per profile.

Regarding the actual properties of the soil layers, in Mexico, we have information about the upper and lower bounds of each layer, its average pH and electrical conductivity, organic carbon and calcium carbonate amount, as well as the percentages for sand, silt

Figure 4.5: Distribution of data points in Mexico, coloured by class, of only the 10 most predominant soil classes.

and clay.

In Figure 4.8 we can see how some of the properties in our data are distributed, throughout the RSG. We can already discern that some classes have different ranges of possible values, on the chemical and physical properties, but also that there is a large overlap in these ranges. However, not all the horizons have information on each of these properties, as such it was required that we first prepare our data for our algorithms.

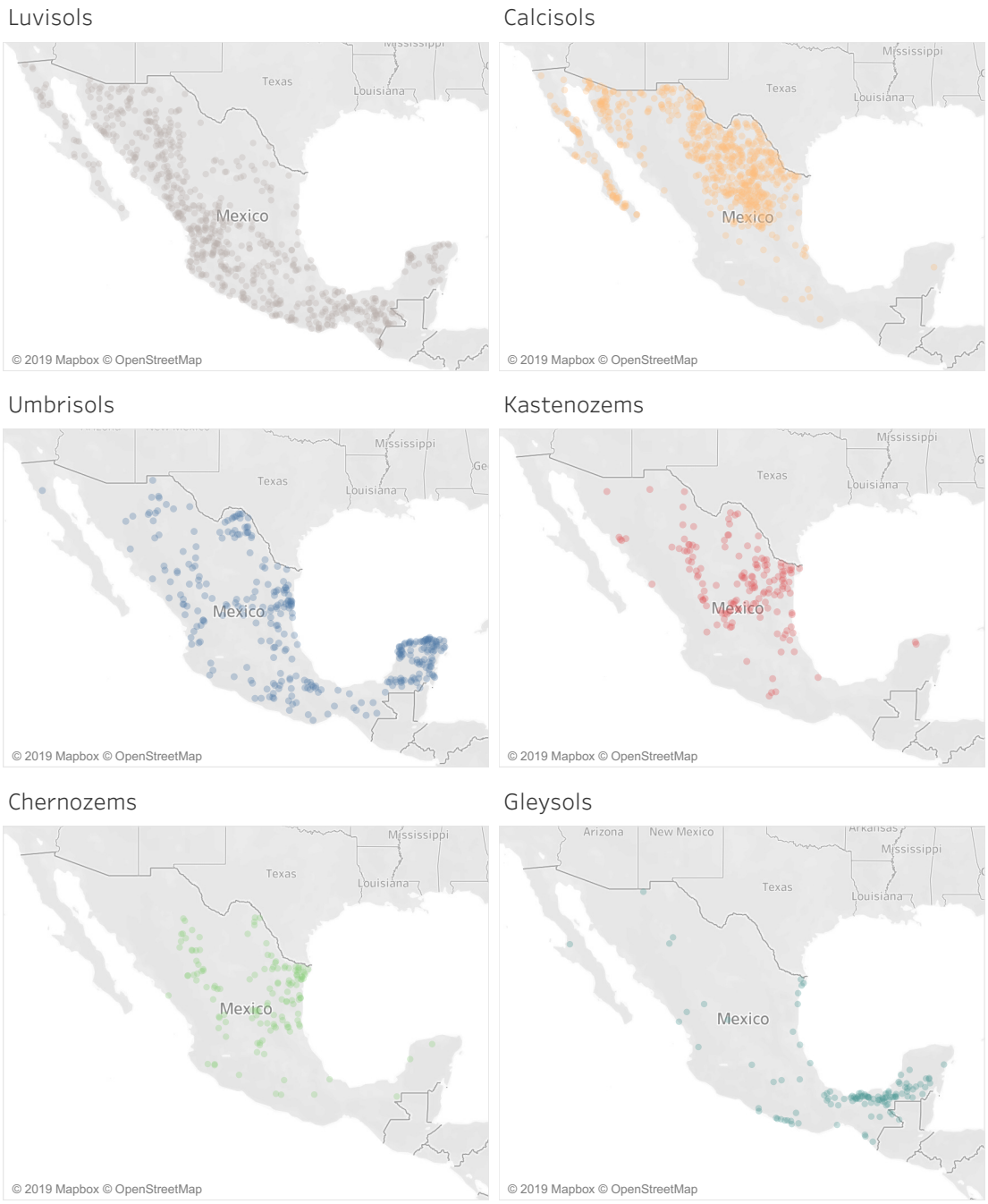


Figure 4.6: Distribution of data points in Mexico, by several classes that show a clear geographical preference.

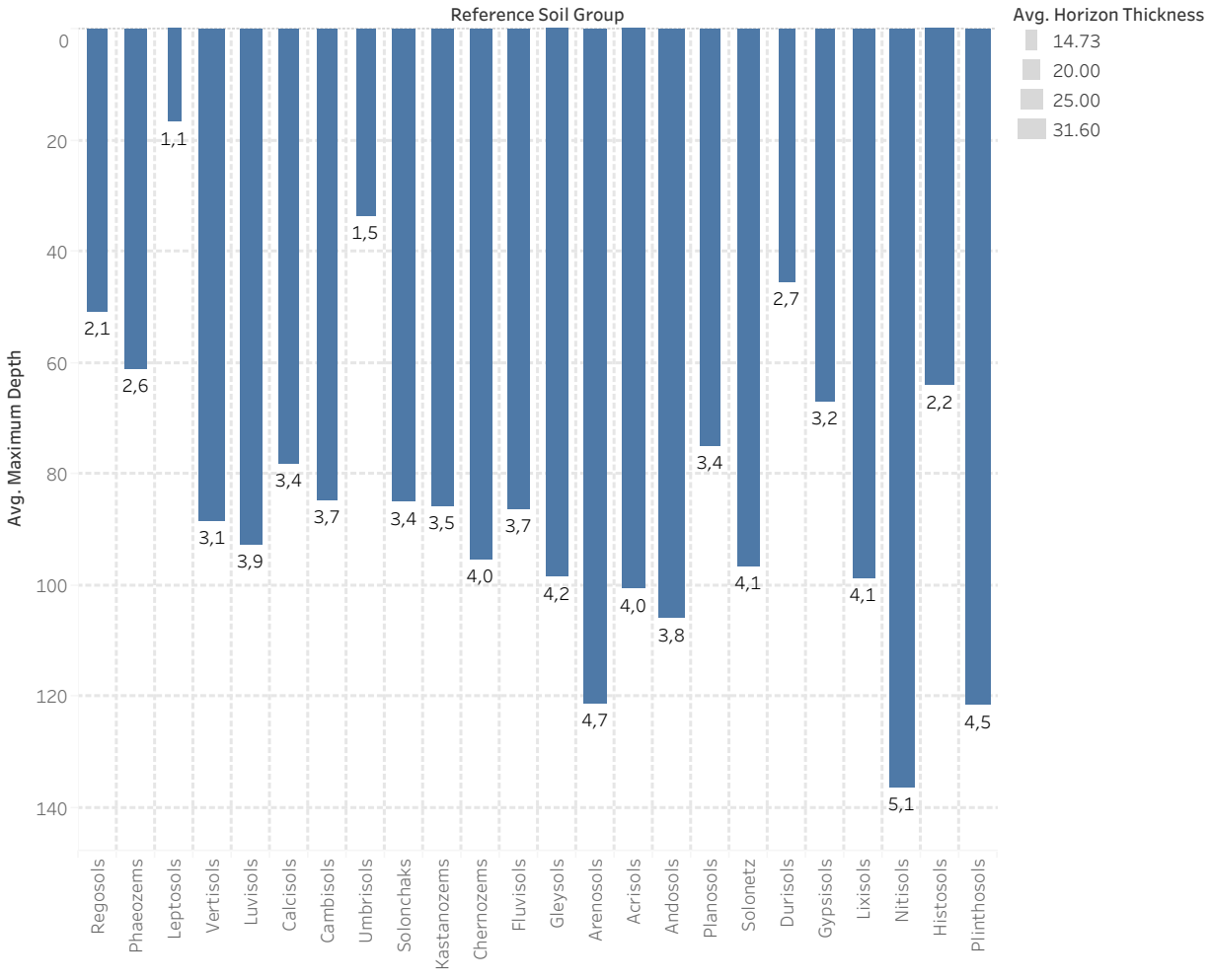


Figure 4.7: Average maximum depth of the profiles and thickness of the horizons, distributed by class. The numbers below the bars represent the average number of horizons, per profile.

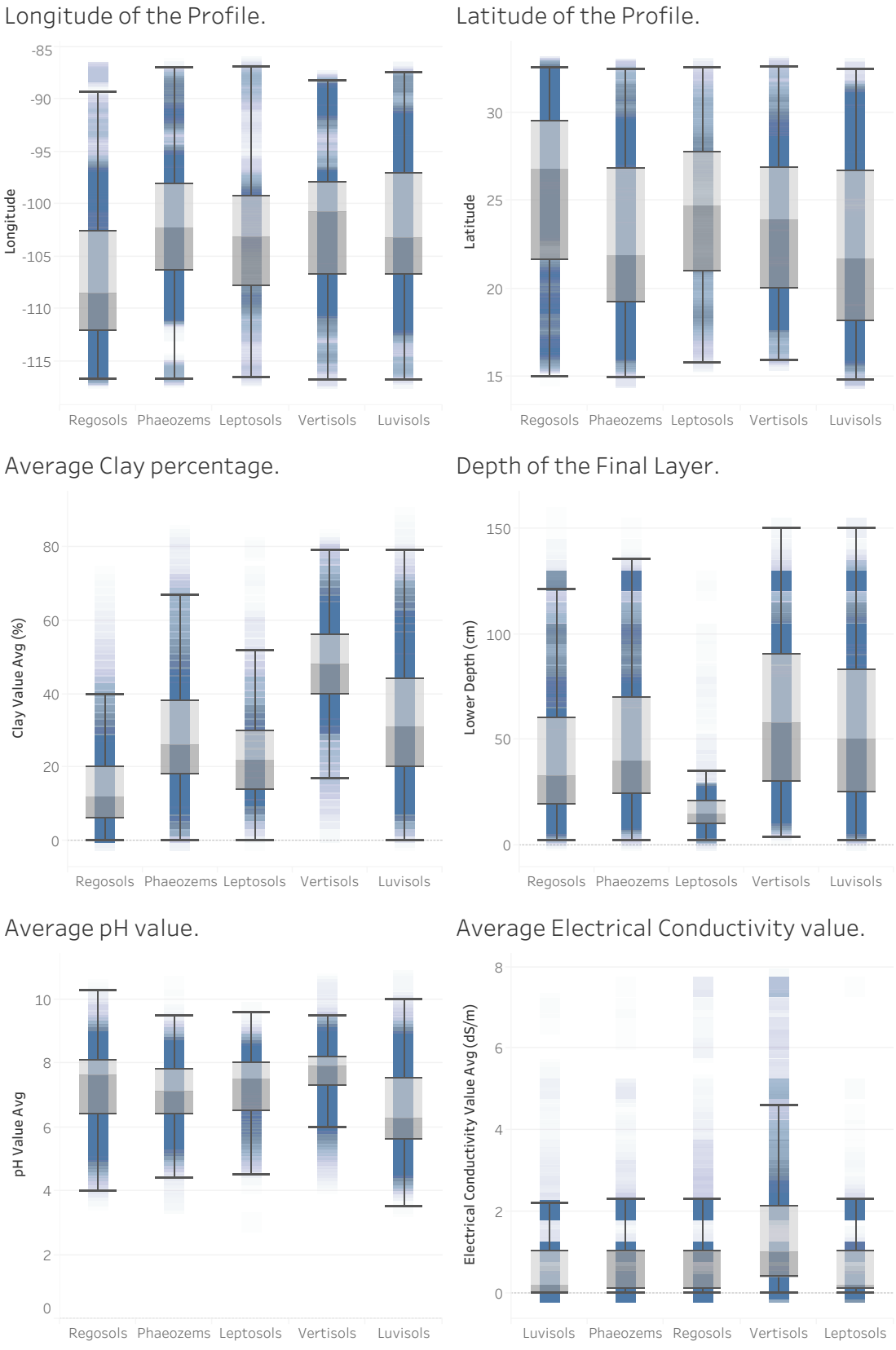


Figure 4.8: Distribution of six of the soil properties present in our data, throughout the 5 most predominant classes.

4.2 Data Preparation

In this section, we describe some of the problems encountered during the analysis of the data, and the approach we took to try to improve on those areas, namely data cleaning and imputation of missing values, with the intent of improving the results of our algorithms.

4.2.1 Cleaning

Like previously mentioned, we chose the Mexico region for our study due to its high density of data and relatively consistent measurements. Even then, there are some variables for which we do not have much information. As such, we decided to remove the features that are present in less than 1% of our total measurements, since they will most likely hinder our algorithms, instead of providing any useful information [7, 31]. We also removed some properties that will not provide insight into the classes such as licences, and the versions of the classifications.

Lack of information and data imbalance was also a problem seen in the data, where some classes have an insignificant number of points in our data set, not enough for our algorithms to be able to learn them. As such we also removed the classes that amount to less than 0.25% of our measures, namely *Plinthosols* and *Histosols*, neither of them having more than 15 profiles recorded.

Some clear outliers were detected, namely some layers that had a lower depth of 999 cm, a value set as an indicator that such data was missing or not correct. All those horizons were removed from our data set, i.e. 24 layers, most of them belonging to the *Calcisols* class.

We also added some computed variables based on information missing in our data that may be helpful. The number of layers was added to our data, as an extra property of the profile, since it seems to provide some insight into the classes We also inserted the thickness of each layer, as a means to help our algorithms extract more information from the data available.

After all this treatment we ended up with 22 RSG, and the properties that we have for training are displayed in Figure 4.9, as well as the percentage of the total horizons for which we do not have that data.

4.2.2 Imputation

As we can see in Figure 4.9, the Calcium carbonate equivalent total average (tceq_value_avg) has over 60% of missing values. This means that most of our horizons have not been measured for this variable. This happens, although to a lesser extent, on other features present in our data set. A more in-depth analysis of the occurrences of these missing values showed that they are evenly distributed throughout all the classes. This means that they are not properties measured for a single profile type, for example, in which case using imputation of the values could lead to undesirable results, since the simple

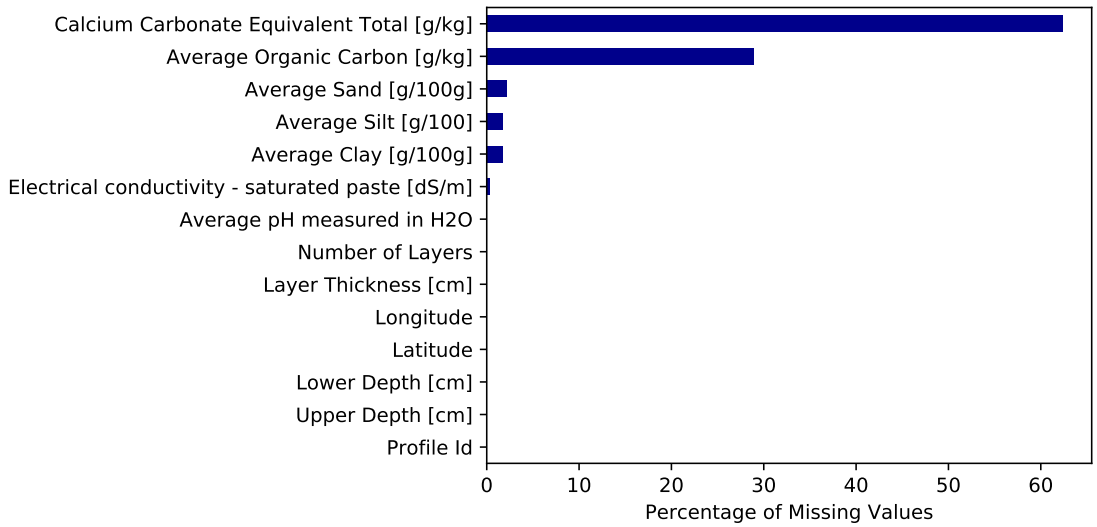


Figure 4.9: Percentage of missing values for each variable, for the layers.

availability of the measure would provide information to our algorithms, regarding the classification.

Our algorithms, for the most part, do not accept null values. This means we can either fill these missing properties with a default value informing the lack of data (e.g. -999), or we can try to impute these features based on the remaining data. In our research, we focused on this last approach, namely using the k-Nearest Neighbours algorithm [14]. This was due to some earlier tests, where using the imputation method provided us with better results when compared to the placeholder variant of treating the data.

This procedure searches for the k most similar data points, using a mean squared difference, based on the other properties of the samples. The resulting feature is the average of the values that were observed in those neighbours, for that same variable.

The value of k is a parameter that has great importance, since a k of 1 will mean that the property being imputed will have the same value than that of the nearest neighbour. This may have problems in cases where that sample is an outlier, resulting in a wrong value. On the other hand, with $k = 3$ it will be the average of the 3 most similar samples which, while resulting in smoother value, it reduces the influence of possible outliers.

We tested this imputation with k varying between 1,2 and 3, choosing the one with the best results for the final model. We also decided the method should only take into account the physical and chemical properties to find the nearest neighbour, therefore the latitude and longitude were removed during this procedure. This decision lies in the fact that our dataset is not geographically dense enough, meaning that the nearest point using the actual distance may be many kilometres away, in a very different soil type.

Chapter 5

Approach

In this section, we describe the methodology that was used in this dissertation. We first present the data representations that were created, followed by a clustering analysis performed on the data set. Finally, we explain how the machine learning algorithms were trained, parametrised, and optimised for the best results.

5.1 Data Modelling

The data provided to us is organised in layers, which contain the properties of the soil and a reference to the profile they belong to, as seen in Figure 5.1-(A). On the other hand, our algorithms must be given the whole profile as a single instance, including all the variables in every layer, to correctly provide a classification for the profile, as shown in Figure 5.1-(B). The number of layers that each profile contains is variable, as described in Chapter 4, which is a problem since our algorithms require a constant input, with the exception of RNN, which will be further explained.

Therefore we are required to create a data representation, where all the profiles have the same number of variables. In this project we tested four different data modelling methods, to assess the one that would provide the most information to our algorithms.

The work done in this section of the thesis is further described in the paper **Soil Classification Based on Physical and Chemical Properties Using Random Forests** [15]

5.1.1 *N* First Layers

The more superficial a layer is, the more soil information we can extract from it [23]. With this in mind, one simple method of modelling our data is to use only the N first layers of our original profile. With this procedure, we always extract the layers that provide us

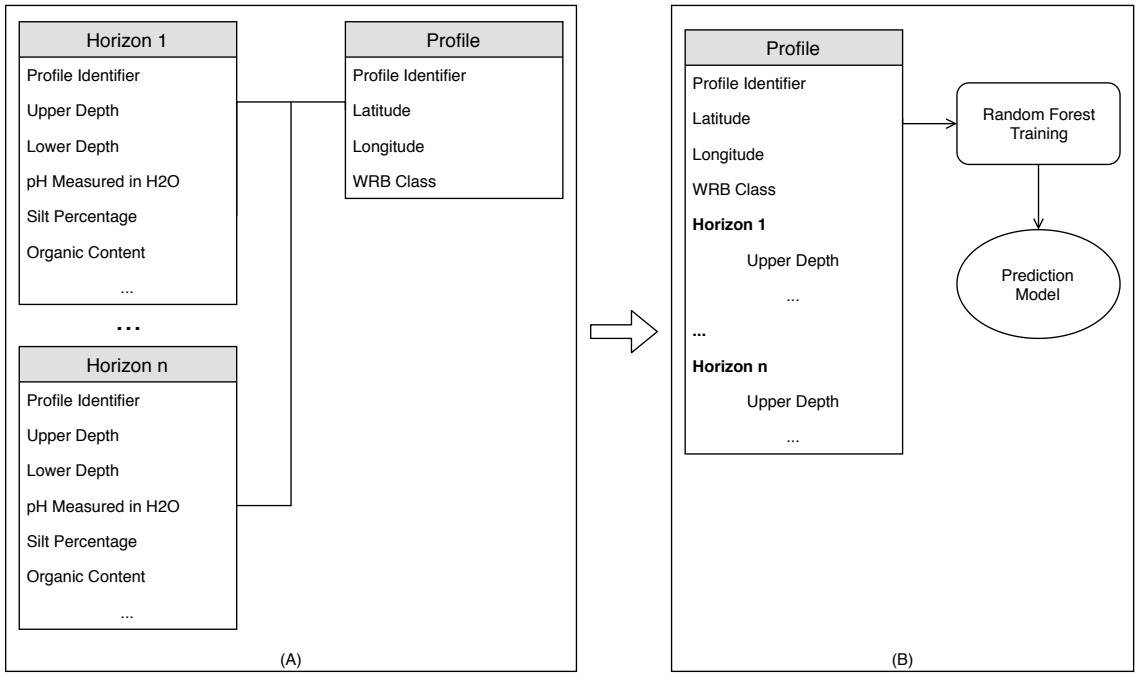


Figure 5.1: Data representations. On the left the initial data disposition is shown. On the right is an arbitrary method that merges the layers into a single data point, representing the whole profile.

with the most information while, if the original profile has a number of horizons larger than N, the deeper layers are discarded.

One problem found in this method occurs when our original data does not have at least N layers. As a solution to this problem, the deepest horizon is repeated, alongside with its properties, resulting in data repetition. This choice was made due to some initial tests, that compared this method with the possibility of inserting predetermined values which represent missing data, where the first provided better results. The parameter N was tested with values of 1, 3, 5, 7, in order to find the one which provides the best balance.

5.1.2 Standard Depth

This method is based on standard values that have been created for the depth of the layers in a profile, i.e. 0 - 5, 5 - 15, 15 - 30, 30 - 60, 60 - 100, and 100 - 200 centimetres, totalling 6 horizons. These values have been predetermined, and are used by scientists when measuring soil information [1]. To find the original layer that coincides with the new representation, we use the middle point for the standard layer, e.g. 80 cm for the 60 - 100 cm layer, and retrieve the values for the original horizon that is present at that depth.

This representation again focuses primarily on using information closer to the surface, with a higher density of measures in that region. It, however, introduces some problems that were not present in the previous method such as some shallow layers not being used, as is the case for layer e, shown in Figure 5.1-2. We again are required to repeat the last

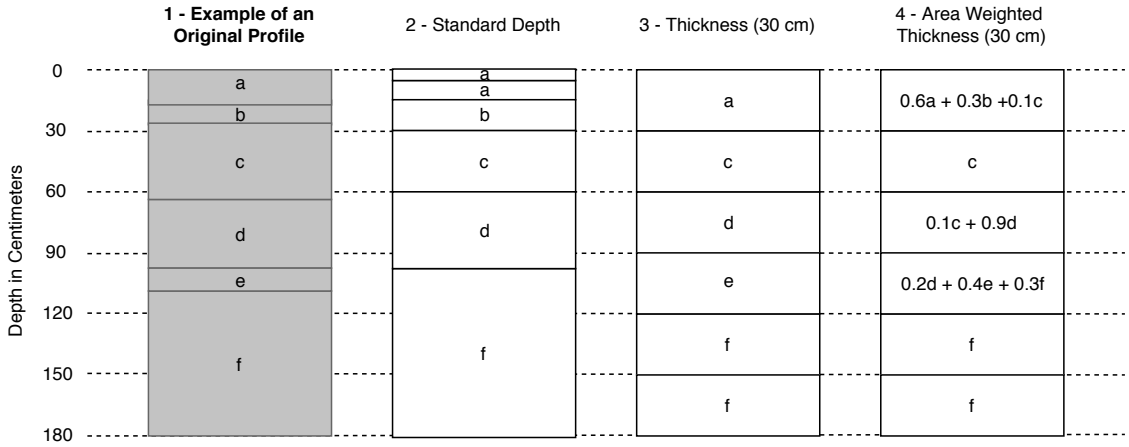


Figure 5.2: Illustration of 3 of the different data modelling techniques that were used. The letters represent the value, for an arbitrary feature, in each layer.

layer of the original profile, if it does not contain enough information to be completely represented in this data modelling method.

5.1.3 Thickness

This approach is similar to the previous method, where there are predetermined depths for each horizon. In this case, we set a specific value for the thickness of each layer, which is then used for all of them. Again, to find the values that should be represented in the final profile, we take a measure at the middle depth of each of the representation's layers.

The thickness value t can be varied and was tested with 10, 30, and 60 cm. The final profile of this representation will then have $\lfloor 200/t \rfloor$ horizons since, for classification purposes, the maximum depth is limited at 200cm, value which we respected in this regard [43, 45]. We again introduce issues, such as when there are not enough layers in the original profile, and of some layers not being used independently of how much of the final horizon is actually occupied by it.

5.1.4 Area Weighted Thickness

This representation was created as an attempt to eliminate some of the errors identified in the preceding methods. It is based on the previous model, presented in Subsection 5.1.3, using a weighted approach, where each original layer that is contained in the final representation contributes to the final values, based on a weight that is attributed to it.

The weight value is directly related to how much of the layer being analysed is contained in the resulting horizon, where the sum of the weights for the final layer must be equal to 1. In Figure 5.2, more specifically by comparing the first layer of the thickness and the area-weighted thickness approaches, we can see that the weighted method utilises the information from three original horizons, unlike the thickness representation which only uses data from a single layer.

This may, however, introduce a smoothing problem, where the abrupt changes in the values between layers may not be well represented, due to the weighted sum of multiple values from different horizons. Also note that, while all the features used in our research are numeric, if there was any categorical variable we would be required to modify this method since it only works directly on numeric values.

5.2 Clustering

In our data, after the treatment, our profiles can be assigned to one out of 22 possible classes, a substantial number when it comes to automatic classification. This may result in poor predictions by our models, as many previous studies have shown [19, 21], especially if the data is not homogeneously balanced between the classes, which is the case.

One method to improve the performance, at the cost of a less specific classification, would be to aggregate multiple classes into groups. This would mean that our algorithms would be required to learn a smaller number of possible labels, improving their predictions. On the other hand, we would not be able to specify the exact class that the soil profile being analysed belongs to. This may not be an issue depending on the problem being solved. If we were, for example, looking for a general type of soil, which can be described by any option from a group of classes with similar properties, this such method would provide sufficient specification, while improving the accuracy of the predictions.

To create these groups we used clustering, an unsupervised learning technique, resorting to the k-Means algorithm [30]. This method focuses on finding a predetermined number of clusters k, based on the data it is given. We only used the physical and chemical properties found on our data for this research, since we also intended to understand if these features can lead us to the geographical preference of some soil types, as seen in some classes in Chapter 4.

To find the best number of clusters for our problem we used the elbow method[4], by plotting the sum of the squared differences of the instances, to cluster centre closer to them, also known as the inertia, against the number of clusters. We can then choose the value where the inertia starts to stabilise, even if we were to use a larger number of clusters. This method is usually known as the elbow analysis.

This method is, however, not very reliable since the actual point of stabilisation is not always easy to discern. Therefore we also used the Silhouette score to help to choose the best value [36]. This method analyses the distance between the data points inside each cluster. It is also usually described in a visualisation, requiring human analysis.

The silhouette method represents every data point, with a value varying from -1 to 1. This amounts to the distance from one instance to the other clusters, where a value of 1 represents that the point being analysed is very far from other clusters, meaning that we have a good separation between the different clusters created. A negative value stands for the inverse, meaning that the data point is closer to a neighbouring cluster. We can

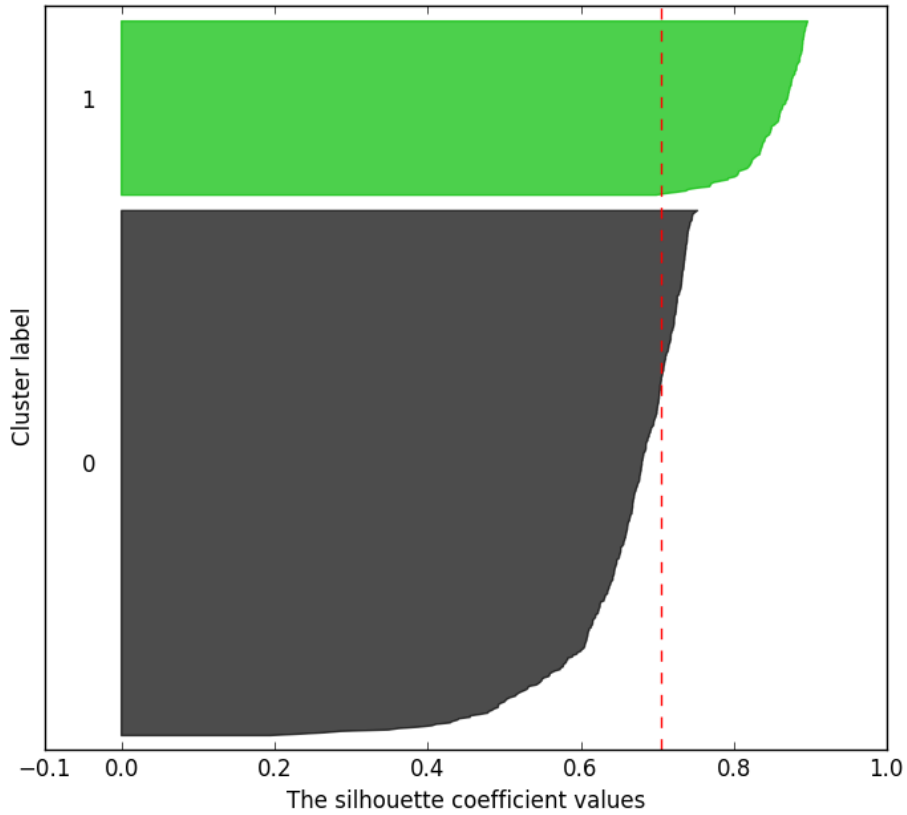


Figure 5.3: Example of a Silhouette analysis plot, where the dashed line represents the average value.

see an example of this plot in Figure 5.3. It is also possible to use the average Silhouette score of all the data points, as a means to assess the separability of our clusters.

5.3 Model Training

Every machine learning algorithm that we used has a set of hyper-parameters that must be specified before training the model. These hyper-parameters can largely impact the performance of the resulting classifier and, as such, must be correctly set for each specific application. In this section, we describe in detail how each of the ML models was trained and parametrised, namely the Random Forest, Gradient Tree Boosting, Deep Neural Network, and Recurrent Neural Network algorithms.

5.3.1 Random Forests

The RF algorithm is widely used due to its performance, ease of use, and parametrisation, as stated in Chapter 3. Even then, many hyper-parameters must be chosen to optimise its use, so that it can extract the most information from our data.

In order to find the optimal values for the parameters we used two algorithms from the

scikit-learn package[1], namely RandomizedSearchCV and GridSearchCV. The first method, given a list of possible values for each of the parameters being tested, tries a predefined number of random permutations on the parameters. This method does not ensure that all the variations are tested, but will give us a good idea for the range of values to use. To better fine-tune these parameters we then used the GridSearchCV method which does test all the possible permutations of the values given to it.

After the initial research using the RandomizedSearchCV we chose the following hyper-parameters and values to test in more detail, where the bold and underlined numbers represent the best value found:

- 1, 2, 3 for the minimum number of samples for a node to be a leaf.
- 2, 3, **6**, 9, 12 for the minimum number of samples to split a node.
- 1000, 1100, 1200, **1300**, 1400, 1500 for the number of trees to be used.

In both of parametrisation phases, we resorted to CV with only 5 folds. This choice was made due to the large number of parameters being tuned, resulting in sizeable time complexity. The implementation of the Random Forest algorithm used in this research was that of the *scikit-learn* package.

5.3.2 Gradient Tree Boosting

To parametrise this algorithm we used the same method described in Section 5.3.1, namely resorting to an initial randomised search, followed by a more specific tuning.

The parameters and values examined in more detail, resorting to the GridSearchCV method, as well as the best values found, are the following:

- 100, **300**, 500 for the number of estimators.
- 0.05, **0.10**, 0.15, 0.20, 0.25, 0.30 for the learning rate.
- 1, 3, 5, **7**, 9 for the minimum child weight.
- 0.0, 0.1, **0.2**, 0.4 for the gamma value.
- 0.4, 0.5, 0.7, **0.8**, 0.9 for the columns sub-sampling by tree.

Again, we utilised a 5 fold CV in the parametrisation of this method. Regarding the implementation of the algorithm using in the research, we resorted to the XGBoost [9] package, for Python.

[1] https://scikit-learn.org/

5.3.3 Deep Neural Networks

The perceptron based algorithms require a different method of optimising the parameters, especially since the architecture of the neural network is, in itself, a hyper-parameter.

As such we tested various architectures:

- Single Dense layer with 32 neurons.
- **Single Dense layer with 64 neurons**.
- Single Dense layer with 128 neurons.
- Single Dense layer with 512 neurons.
- Dense layer with 64 neurons, 0.2 Dropout layer, 16 neuron Dense layer.
- Dense layer with 512 neurons, Dense with 128 neurons, 0.2 Dropout, Dense with 64 neurons, Dense with 64 neurons, 0.2 Dropout, Dense with 32 neurons, Dense with 32 neurons, 0.2 Dropout.

All of these networks were trained using the Rectified Linear Unit (ReLU) activation function for the Dense layers [27]. The last two variants were significantly more complex, made as an attempt to verify if augmenting the complexity of the architecture could improve our results.

These also resort to Dropout layers, in which the output of some random neurons is ignored, during the training. The number of ignored outputs can be parametrised, and this method is used to prevent overfitting. To output the probability of the data being analysed belonging to each of the possible classes, each model had a final Dense layer with a *softmax* activation appended in the end.

During the training, the Adam optimiser was used, with a learning rate of 1^{-5} and a decay of 1^{-6}. We also employed an early stopping method, where the algorithm will stop its training if the results on the validation set do not improve for a predefined number of epochs, i.e. 5 in our research. This is also a method largely used to prevent the final model from overfitting to the training data set.

The best architecture turned out to be a single Dense layer with only 64 neurons. This, in conjunction with the small learning rate and early stopping, resulted in very little overfitting, even after more than 500 epochs of training. Adding more complexity to the network resulted in worse results, probably related with the low amount of data, since neural networks, especially the deeper variants, require a lot of information to correctly learn most problems. All of the methods tested were implemented using the Keras package [11].

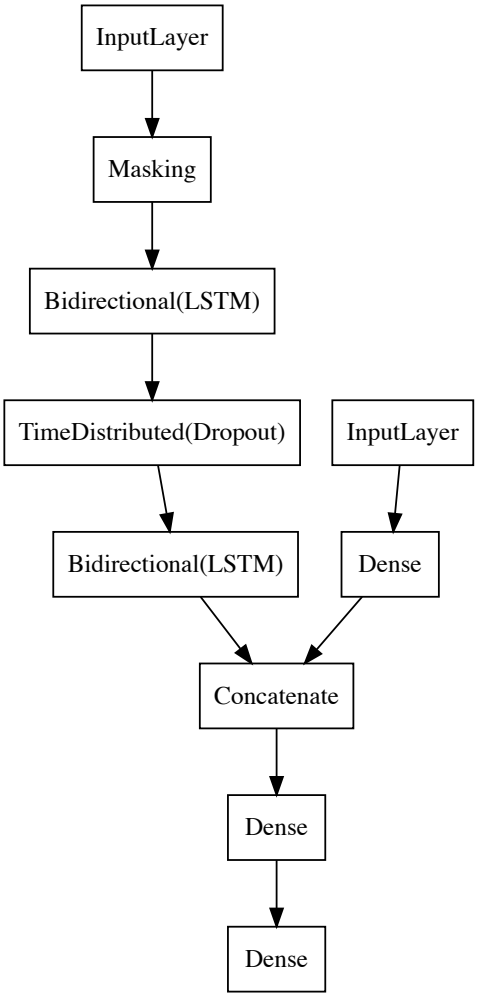


Figure 5.4: Representation of the final architecture that we used for the RNN algorithm.

5.3.4 Recurrent Neural Networks

The RNN algorithm was approached very differently from the methods previously detailed. In those other procedures, we resorted to the data representations that were created in Section 5.1, while for this method we decided to use the data as it was provided. This choice was made due to the sequential nature of the data we have, where the order of the layers certainly impacts not only the other layers but also the final classification.

As stated in Section 5.1, we have two data inputs, i.e. data regarding the whole profile, such as its location, and information about the chemical and physical properties of the layers. Therefore, we decided to use a non-sequential architecture, for which the best network used in our tests can be seen in Figure 5.4, where we have two input layers, one for the profile and the other for the horizons data.

The horizon information then goes through a Masking layer that skips a time-step,

in our case a horizon, if all of its properties are equal to a predetermined value. This lets us use all the layers of each profile, instead of recurring to a previously defined data modelling method that may result in data repetition or even its loss. We can now set the number of input variables to be that of the profile with the most layers while, if another profile does not have enough horizons to fill that input, we create arbitrary horizons where the values for all of its features are set to the placeholder value, so that it will be skipped, not affecting the deeper layers in the network.

This data then goes through two Bidirectional LSTM layers, with a Time Distributed Dropout in between. This makes it so that the network learns not only the relationship between a horizon and the ones previous to it but also the ones following it. The Dropout layer regularises the network, diminishing the effect of overfitting and since it is applied within a Time Distributed layer, it affects each time step of the input, the horizons, independently.

After that, the output from this branch is concatenated with the result of applying a simple Dense layer to the profile information. This data then goes through a Dense layer followed by final *softmax* Dense layer, to extract the probability of the data point being analysed belonging to each of the possible classes. Again, this network was implemented using the Keras package [11].

Many other versions were tested, before reaching this final architecture, namely:

- Without regularisation, meaning no dropout layers.
- Using a regular LSTM layer, without being bidirectional.
- Using a Cyclic Learning Rate [40]
- Using Nested LSTMs [33]
- Using Multiplicative LSTMs [25]
- Using Chrono LSTMs [42]
- Using a IndRNN Layer insted of the LSTMs [28]

Chapter 6

Experimentation and Analysis

After the realisation of all the experiments described in the previous section, we now present their different results. We also go in-depth on the performance of the tested methods and algorithms, providing various visualisations to facilitate the understanding of the results, as well as to extract more information on the reasoning behind them.

In this chapter, we will first present the results of the data modelling tests, followed by the clustering experimentations, and a comparison between the models created. We then analyse the outcome of one of the best classifiers trained and finally present some punctual experimentations made during this research.

6.1 Impact of Data Modelling on the Classifiers

The different data modelling methods were tested only using the Random Forests and the Gradient Tree Boosting classifiers. This is due to the higher time complexity of the Deep Neural Network algorithm, as well as the fact that we use different data modelling techniques to train the RNN algorithm, as previously explained. The totality of the results from the data representation tests is shown in Figure 6.1.

This image presents the various results, for each of the data modelling representations tested. It also shows the performances for the different parameters tested, both on the imputation method, as well as in the data modelling. These results were obtained using the RF and GTB algorithms.

We can observe that the Kappa score ranges from 0.429 to 0.504 and the Accuracy between 0.482 and 0.554. All the representations, in some parameter variation, provide us with Kappa and accuracy values that are very similar to the best results. The variation in our best-performing methods is mostly negligible and, as such, it is hard to say which data modelling provides the best results. However, the standard data modelling was the

Imputation K / CLF

Modelling	Kappa 1 RF	Kappa 1 XGB	Kappa 2 RF	Kappa 2 XGB	Kappa 3 RF	Kappa 3 XGB	Accuracy 1 RF	Accuracy 1 XGB	Accuracy 2 RF	Accuracy 2 XGB	Accuracy 3 RF	Accuracy 3 XGB
Layers 1	0.4290	0.4440	0.4440	0.4510	0.4350	0.4480	0.4820	0.5000	0.4950	0.5070	0.4870	0.5040
Layers 3	0.4690	0.4890	0.4850	0.4910	0.4790	0.4960	0.5200	0.5400	0.5340	0.5420	0.5290	0.5470
Layers 5	0.4780	0.4930	0.4830	0.5040	0.4770	0.4900	0.5280	0.5440	0.5330	0.5540	0.5270	0.5420
Layers 7	0.4720	0.4840	0.4760	0.4900	0.4750	0.4990	0.5220	0.5360	0.5260	0.5420	0.5260	0.5490
Standard	0.4960	0.4950	0.5000	0.5000	0.4980	0.4980	0.5440	0.5460	0.5470	0.5500	0.5460	0.5480
Thickness 10	0.4910	0.4940	0.4880	0.4940	0.4910	0.4950	0.5400	0.5450	0.5370	0.5450	0.5390	0.5460
Thickness 30	0.4820	0.4980	0.4880	0.5000	0.4950	0.4910	0.5310	0.5490	0.5370	0.5500	0.5430	0.5420
Thickness 60	0.4870	0.4920	0.4950	0.4940	0.4960	0.4870	0.5350	0.5440	0.5420	0.5450	0.5430	0.5380
Thickness (W) 10	0.4580	0.4920	0.4620	0.4960	0.4620	0.4910	0.5110	0.5430	0.5140	0.5470	0.5150	0.5430
Thickness (W) 30	0.4730	0.4910	0.4710	0.4830	0.4720	0.4780	0.5240	0.5420	0.5220	0.5360	0.5240	0.5320
Thickness (W) 60	0.4680	0.4720	0.4610	0.4720	0.4620	0.4760	0.5190	0.5260	0.5140	0.5260	0.5140	0.5300

Kappa 0.42900 0.50400 Accuracy 0.48200 0.55400

Figure 6.1: Results from the data modelling tests for the different variations of k-Nearest Neighbours k value, as well as the representation parameters, for Random Forests and Gradient Tree Boosting.

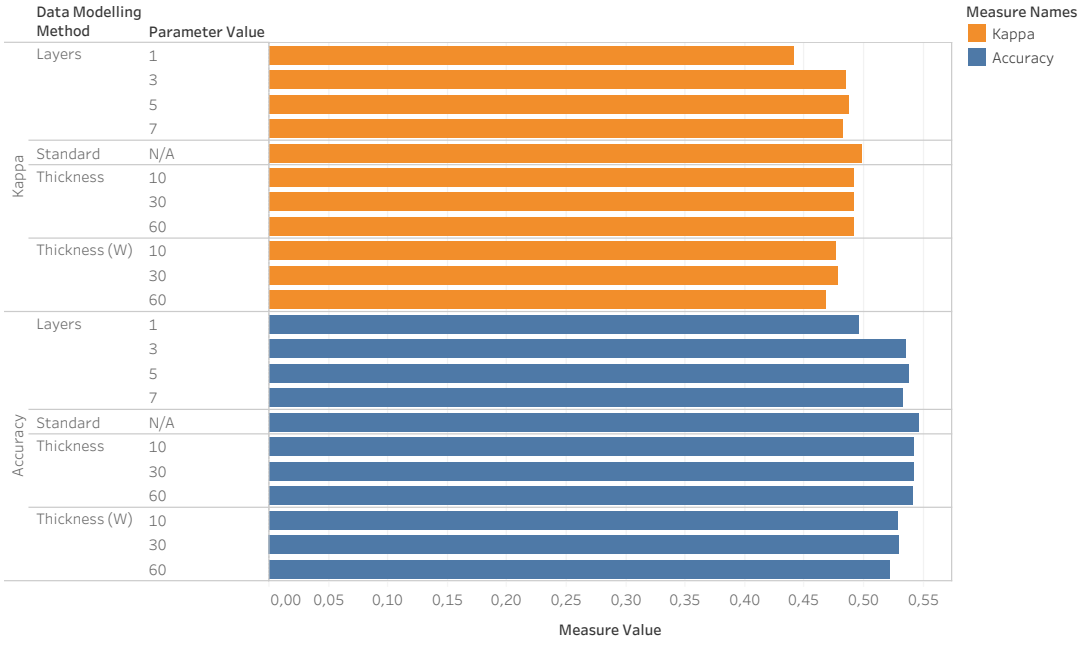


Figure 6.2: Analysis of the different parameters, applied to the representations, averaged over both Random Forests and Gradient Tree Boosting.

representation that provided the best results more consistently, over both algorithms.

Figure 6.2 represents the results for each data modelling variation, as well as the impact of the parametrisation of these methods. We can see that, in the thickness based representation, the t value does not seem to impact the performance by much. On the other hand, on the n first layers method, the number of layers seems to have more impact, where using a single horizon provides substantially lesser information. Even then, using only a single layer achieves over 85% of the Kappa value, when compared with the best

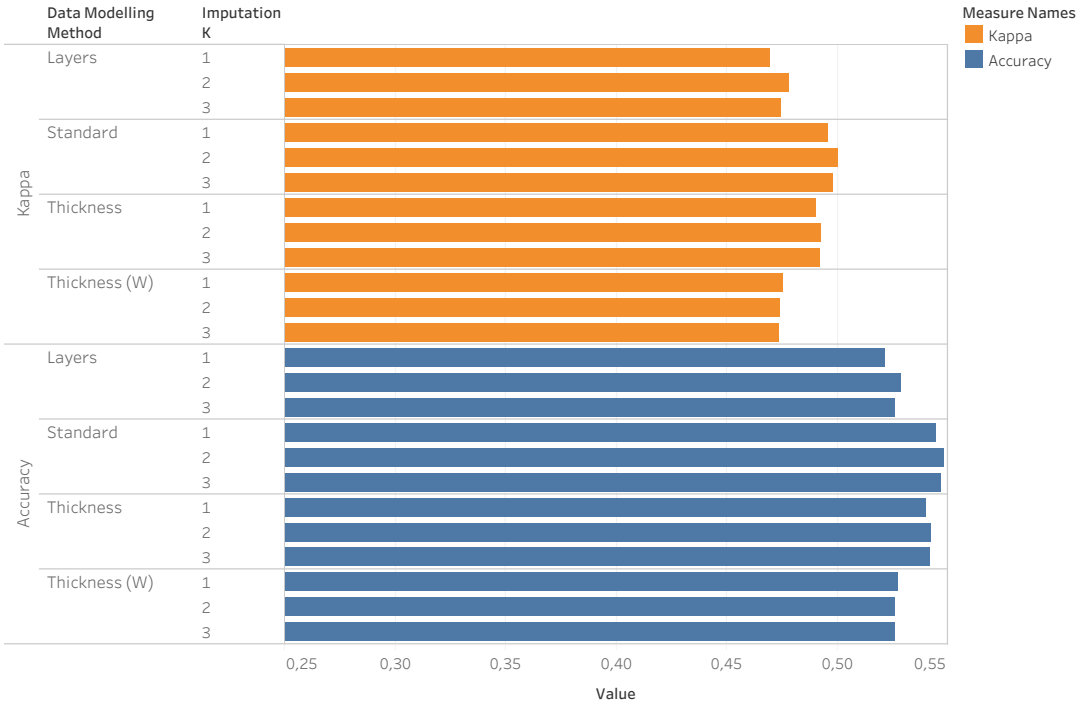


Figure 6.3: Analysis of the k-Means Imputation parameter k, applied to the representations, averaged over both Random Forests and Gradient Tree Boosting.

overall model. With this, we can again confirm that the shallower horizons seem to have the greatest impact in our problem, especially in the first three layers, since the performance of our algorithms does not improve much by using more layers. This is also related to the average number of layers per profile which is around three horizons, as seen in Chapter 4, therefore using data modelling techniques with more than 3 horizons will be mostly filled with repeated information, for the deeper layers.

The weighted thickness data model resulted in the worst results when compared to the other representations. Analysing the regular thickness version, we can see that the weight component hinders the performance of the modelling. This may be due to the loss of abrupt value changes between the layers, a problem that was identified during the creation of this method, in Section 5.1.4. The thickness parameter has a small impact on the performance, although using layers of 30cm showed the best results in this method. On the other hand using thicker layers, namely with 60cm, tends to worsen the predictions.

In Figure 6.3 we present results of varying the parameter k, on the k-Means imputation method, averaged over both algorithms. Regarding the imputation results, these also do not have much of an impact on the performance of our algorithms, with the variance of the Kappa value never reaching the 0.01, between different k values. Even then, the best results consistently belong to an imputation using $k = 2$.

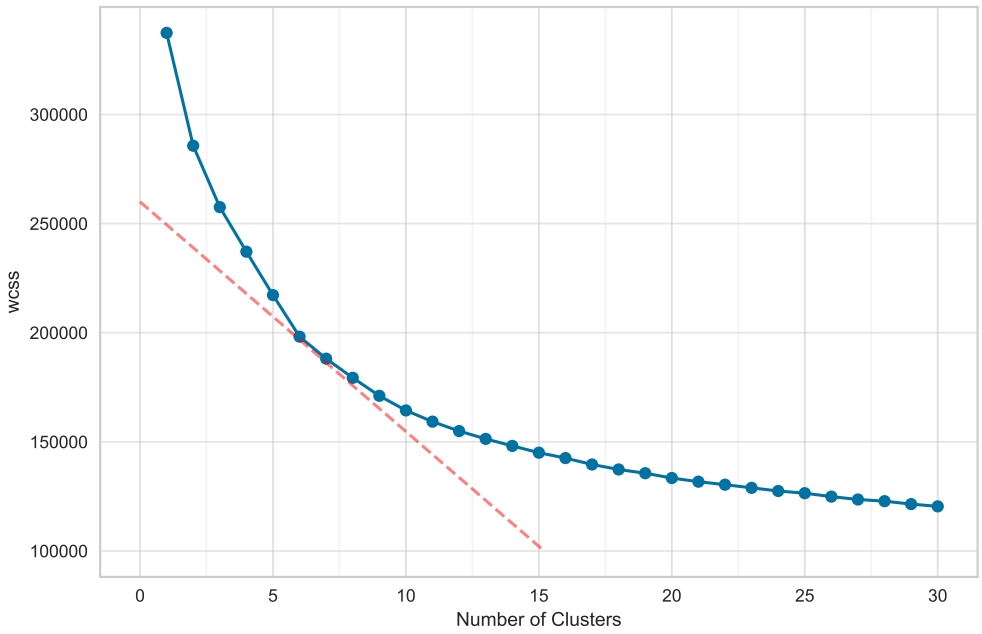


Figure 6.4: Visualisation of number of clusters against the within cluster sum of errors (wcss), to help choose the best value for k, for the k-Means algorithm.

6.2 Clustering

As mentioned in Chapter 5, we used the elbow method to visualise the best k value for our approach. This graph can be seen in Figure 6.4, where we plotted the number of clusters against the within-cluster sum of errors (wcss). Analysing this image we can infer that the best number of cluster to use in our problem is between 5 and 7, where the "elbow" of the line can be seen.

To better analyse this range of possible values, we then resorted to the average Silhouette scores, as seen in Figure 6.5. In this test, the best results come from using 6 as the number of clusters, which is backed up by the graph in Figure 6.4, where we can see an accentuated change in the plot, for that value. Even then this value is very small, only reaching an average score of 0.168, which represents that our data does not seem to be easily separable, using this method.

We then analysed the actual Silhouette plot, for the best k value, as seen in Figure 6.6, on the right. It is possible to observe that, even in the best performing cluster method, we still have a lot of variation on the Silhouette scores, where our clusters usually have a peak, rapidly descending. This means that the centres of our clusters are probably separated by a small distance, meaning that if a data point slightly deviates from the centre of its cluster it will be very close to another cluster, resulting in this fast decrease of the silhouette score, as the points deviate further.

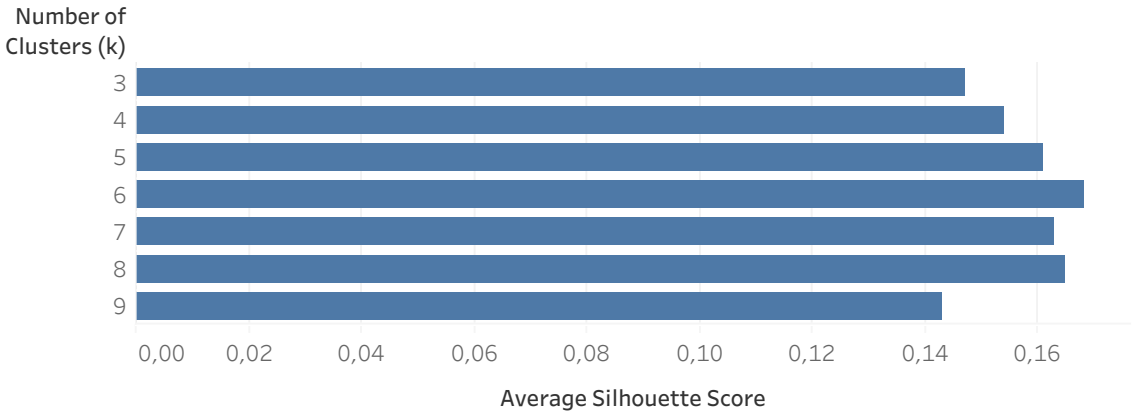
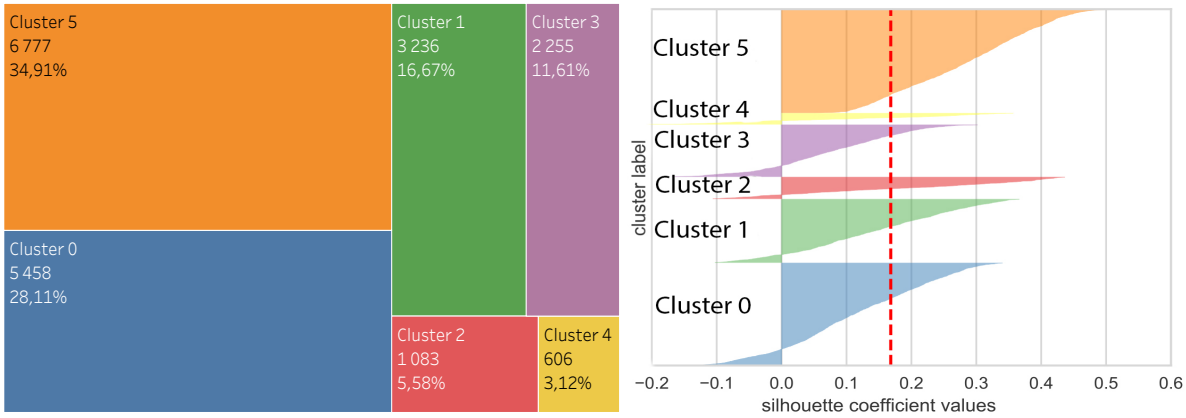


Figure 6.5: Average Silhouette scores for various values of k.



(a) *Cluster sizes*

(b) *Silhouette scores*

Figure 6.6: On the left we have the sizes of the different clusters, using k-Means with $k = 6$. On the right, a visualisation of the Silhouette scores also for $k = 6$, value where the average score was higher, i.e. 0.168. In this last graph, the dashed line represents the average silhouette score.

By performing a more in-depth analysis on the best clustering method, we can see that the groups have very different sizes, being that clusters 5 and 0 have more data points than 4, 2 and 3 combined. This again confirms the lack of balance in the different soil properties, where some combinations tend to happen at a higher quantity, as can bee seen in Figure 6.6, on the left.

Our clustering algorithm had access neither to the class nor the location of the profiles. In Figure 6.8 we can see how the data points, belonging to each of the created clusters, is spatially located. If we analyse the geographical distribution of the different clusters, we will again see that there is a correlation between the location and the soil properties. It is possible to observe that some clusters tend to appear in different areas, such as cluster 1 which is located mostly in the north-western region of the country. Again, this was all taken simply from the physical and chemical properties of the soil.

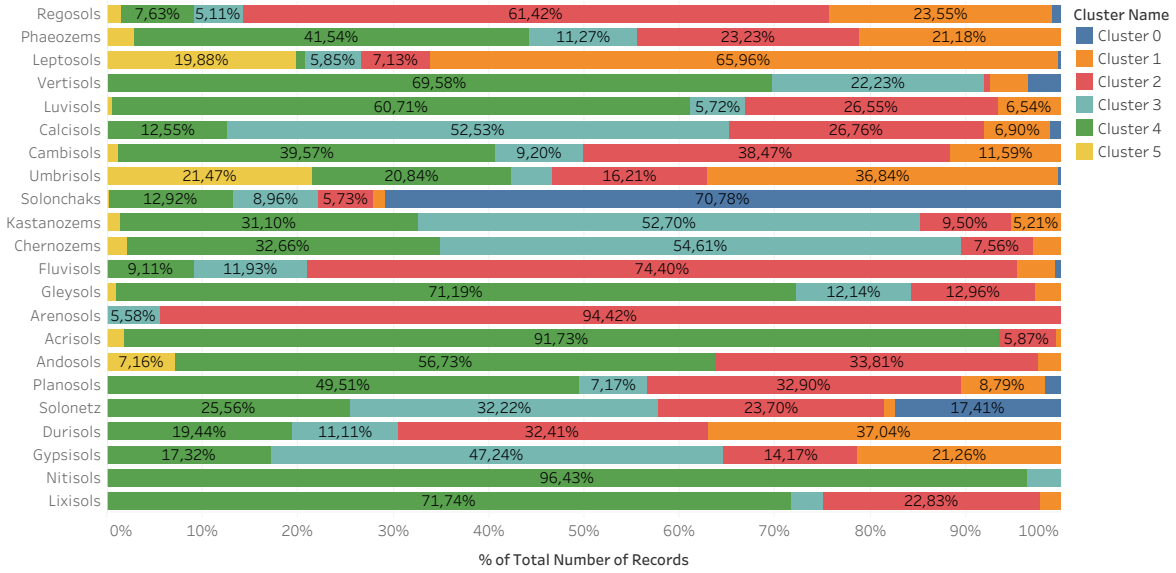


Figure 6.7: Distribution of each class by the different clusters, using k-Means with $k = 6$.

In Figure 6.7 we can see how each class is separated into the different clusters. It is possible to see that the clusters do not cleanly separate the classes existent in our original data. This is due to the fact the most classes have a large number of profiles contained most of the clusters, unlike our initial objective where a single cluster would completely contain more than one class. Therefore, we will not be able to use this method to simplify the classification problem since we can not separate the classes into groups. We could still use these clusters as the label for the training of our algorithms, but we would be losing the connection to the class, instead only classifying the data on the clusters, simply based on the soil properties.

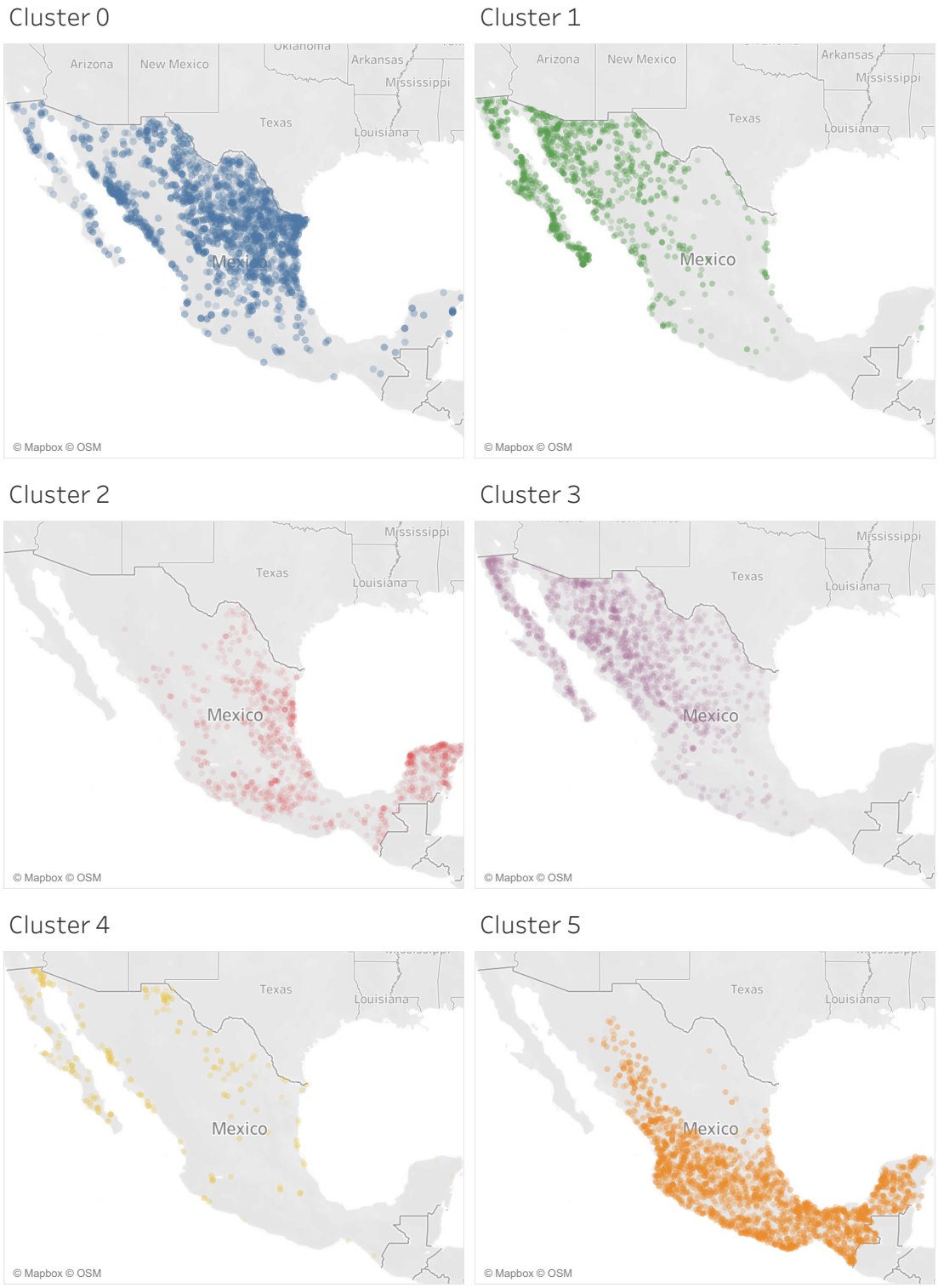


Figure 6.8: Geographical distribution of the different clusters, generated using k-Means with $k = 6$.

6.3 Classification Algorithms Comparison

In Figure 6.9 we can see the various results for each algorithm, for the best data representation and parametrisation. The best algorithm is the GTB with a Kappa value of 0.5 and an accuracy of 0.55. This is closely followed by the RF model with very similar results, where the difference is mostly negligible and, when this value is rounded to two decimal places, they are impossible to distinguish. Analysing Figure 6.3, we can see that, even though they have similar results when comparing their best performances, the GTB can usually extract more information from the various representations than the RF algorithm. This affirmation comes from the fact that the GTB classifier has results near the 0.5 Kappa value in several data representations, unlike the RF algorithm.

After the results we obtained in the previous section, we used the representation that provided the best overall performance to train the DNN algorithm, i.e. the standard approach using an imputation with $k = 2$. On the other hand, the RNN model was trained using the data as is presented in Section 5.3.4, with little treatment, so that the algorithm can extract more information from it.

We can see, again in Figure 6.9, that both of our neuron based algorithms had similar performances, even using very different data representations. These methods had worse results when compared to the decision tree-based models, with a drop of over 15% and 25% on the accuracy and Kappa values, respectively.

In Figure 6.10 we observe can the number of incorrect predictions by our models, separated by the number of classifiers that correctly guessed a specific profile. We can again see that the Gradient Tree Boosting model has a slightly better performance than the Random Forests, both with substantially better performances than the neural network-based algorithms.

To analyse and compare the results of the algorithms, on a per-class basis, we created

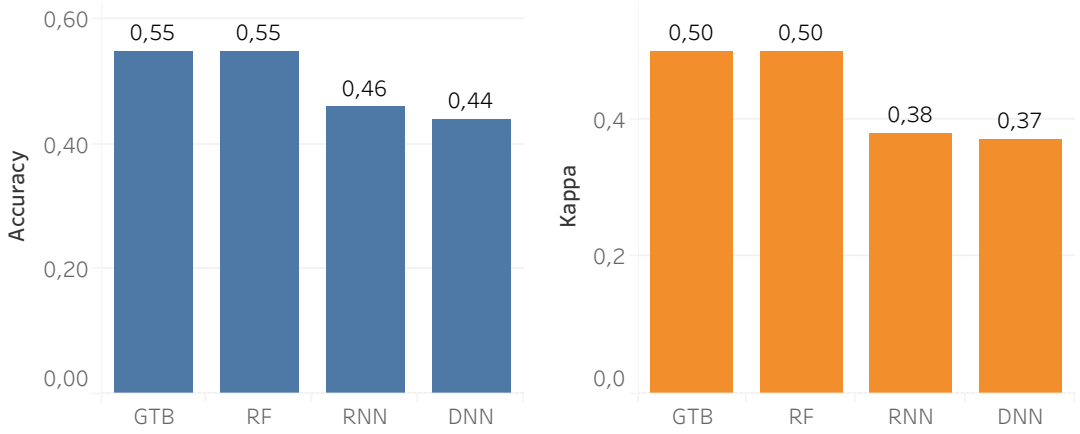


Figure 6.9: Accuracy and Kappa values for the best performing prediction models, obtained by the training of each algorithm.

N of Incorrect Profiles: 0.0 – 424.0

Number of Correct Classifiers	Number of GTB Incorrect	Number of RF Incorrect	Number of RNN Incorrect	Number of DNN Incorrect	Total Number of Profiles
0	417	417	417	417	417
1	102	124	154	148	176
2	54	59	132	123	184
3	16	28	53	51	148
4	0	0	0	0	424
Grand Total	589	628	756	739	1,349

Figure 6.10: Comparison of the number of correct and incorrect predictions for the trained models on the validation set.

Accuracy: 0,000 – 1,000

Reference Soil Group	Gradient Tree Boosting	Random Forests	Recurrent Neural Network	Deep Neural Network	Total Number of Profiles
Regosols	0,587	0,505	0,563	0,615	208,000
Phaeozems	0,531	0,369	0,207	0,218	179,000
Leptosols	0,848	0,795	0,748	0,695	151,000
Vertisols	0,659	0,652	0,616	0,696	138,000
Luvisols	0,577	0,584	0,628	0,569	137,000
Calcisols	0,631	0,631	0,585	0,592	130,000
Cambisols	0,235	0,224	0,153	0,212	85,000
Umbrisols	0,645	0,629	0,290	0,339	62,000
Solonchaks	0,775	0,875	0,625	0,650	40,000
Kastanozems	0,290	0,290	0,000	0,000	31,000
Chernozems	0,185	0,148	0,037	0,037	27,000
Fluvisols	0,080	0,080	0,000	0,040	25,000
Gleysols	0,478	0,609	0,087	0,000	23,000
Arenosols	0,400	0,700	0,600	0,250	20,000
Acrisols	0,421	0,421	0,105	0,263	19,000
Andosols	0,737	0,895	0,316	0,526	19,000
Planosols	0,278	0,222	0,000	0,000	18,000
Solonetz	0,385	0,385	0,000	0,000	13,000
Durisols	0,375	0,375	0,000	0,000	8,000
Gypsisols	0,125	0,375	0,000	0,000	8,000
Lixisols	0,000	0,250	0,000	0,000	4,000
Nitisols	0,250	0,250	0,000	0,000	4,000

Figure 6.11: Comparison of the per class accuracy on the validation set.

the table represented in Figure 6.11 where the accuracy is shown for each possible class, for all of the algorithms used. We can see both the RNN and DNN have very bad results on the classes for which we have the least data, with many instances where they have an accuracy of 0. These algorithms generally require a lot of information to learn the patterns in the data, more than the decision tree-based algorithms [23], which corroborates the results attained. The GTB algorithm tends to have better accuracy on the majority classes, showing better results in these, while the RF model mostly outperforms every other model on the classes for which we have fewer data.

In general, we can see that the Random Forests model is the best when it comes to classifying classes for which we have less information, while the other algorithms can reach better results when more data is present.

In Figure 6.12, the time that each model took to train is represented, in seconds. We

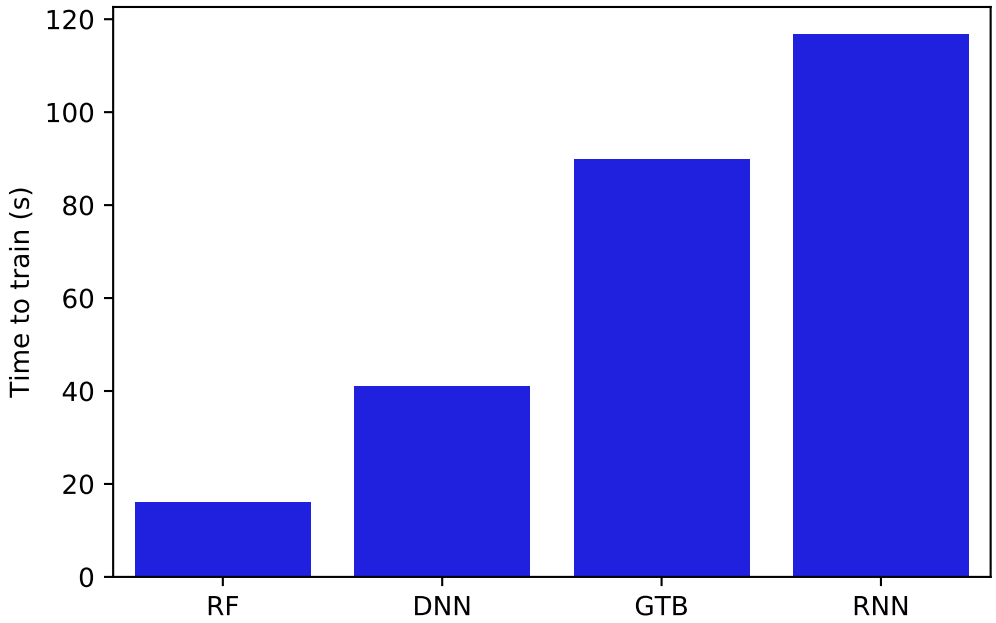


Figure 6.12: Time to train for each of our tested algorithms, for their best model.

can see that overall, our algorithms converge very fast, especially when compared to some of the state-of-the-art methods that utilise remote sensing data, that generally take several hours, or even days, on the training of the algorithms [23]. The RF algorithm was the fastest model to train in our experimentations, taking less than 20 seconds to reach its best results. This method can be fully run in parallel, where every singular random tree is trained in its own thread, explaining these results.

Comparing the neuron based models, the RNN requires a training time that is nearly 3 times that which is required for the DNN. This is to be expected due to the complexity of the RNN, which uses a larger and more complex architecture, with bi-directional LSTM layers, when compared to the simpler dense layers of the DNN model, as well as its smaller network.

All these algorithms were trained on a laptop, using an Intel(R) Core(TM) i5-8259U with four cores, without resorting to a Graphical Processing Unit (GPU), since the training time was already small. If we were to use a GPU we expect that the GTB, DNN and RNN would benefit strongly from it, further reducing the training time on these methods. This would be especially necessary if we were to add more data for our algorithms to train with, or if we were trying to learn a larger area, to provide better scalability of our models.

6.4 Results Examination

After performing a comparison between the various algorithms, as well as the data representation methods used, we now explore the results of one of the best models, namely one created by using the Random Forests algorithm, due to its high performance and faster training time, in combination with the standard depth data modelling, imputed with k-Means with $k = 2$.

In Figure 6.13 we can see the feature importance, as determined by the RF algorithm. From this image, we can infer that the most important information for the classification of the profiles is its location, namely the longitude and latitude. This is to be expected since we saw in Chapter 4 that some WRB classes tend to have a specific region where they are more predominant.

Another fact analysed in previous chapters is that the depth of the profile also has an impact on the class, since some of them are better described by being shallower, for example. This is also present in Figure 6.13, where we have that the depth of the final layer of our profile, i.e. lower_depth_150, has large relative importance.

The remaining features are soil chemical properties, namely at the middle depths of our profile, from the 45 to 150 cm range, where the clay percentage, pH and electrical conductivity seem to outshine other variables, providing more information and distinction on the classes.

As seen throughout various sections of this dissertation, a big problem in the soil classification and the general machine learning areas is the class imbalance. This issue can be further analysed in the confusion matrix, represented in Figure 6.14.

We can see that the least representative classes tend to have worse results since our algorithm did not have enough information to correctly learn these classes. On the other hand, while the majority classes have a generally better performance than the RSG in the minority, we can see that we also have a lot of false positives.

This problem can be clearly seen is by analysing Figure 6.14, where other smaller classes such as *Nitisols*, *Lixisols*, *Acrisols*, between others, tend to be mistaken for a more representative class, i.e. the *Luvisols* . To further determine the causes of this confusion in our model, we decided to compare the range of some of the most important properties, to see if these classes are similar in that regard.

The results of those tests are presented in Figure 6.15 where the 6 most important features are represented for *Luvisols* and the remaining classes that are being classified erroneously. We can see that, although the values are indeed very similar, there is enough variation that can be seen by human analysis. One such example can be seen by comparing *Luvisols* and *Gleysols*, which have a different range of longitude values, and latitude values that, while they intercept, they are more limited and well defined on the *Gleysols*.

This is a pattern that is repeated throughout all the classes, where they have an interception of ranges in all the properties, but we can see that there is a variation between

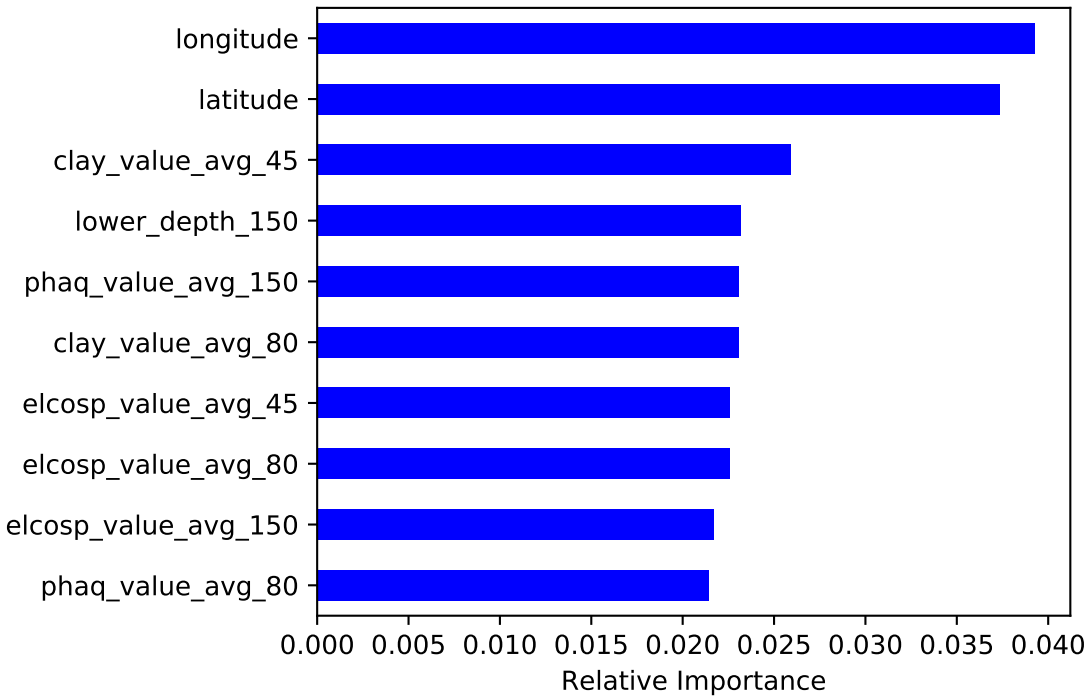


Figure 6.13: Feature importance for the standard depth data representation, with an imputation based on $k = 2$, using the Random Forest classifier. lower depth = lower depth, clay value avg = average clay percentage, phaq value avg = average pH measured in H2O, elcosp value avg = electrical conductivity in saturated paste. The numbers next to the variable names represent the depth at which the measurement was taken.

the values. Therefore, we expect that more data would be required so that the difference between these classes could be reinforced, leading to better results. Another way of improving this problem could be over and under-sampling, where we could reduce the number of samples in the majority class, and introduce some more instances of the minority labels, in order to even the number of profiles in each class, thus reducing the tendency of wrongly classifying the instances as *Luvisols*, i.e. the majority class.

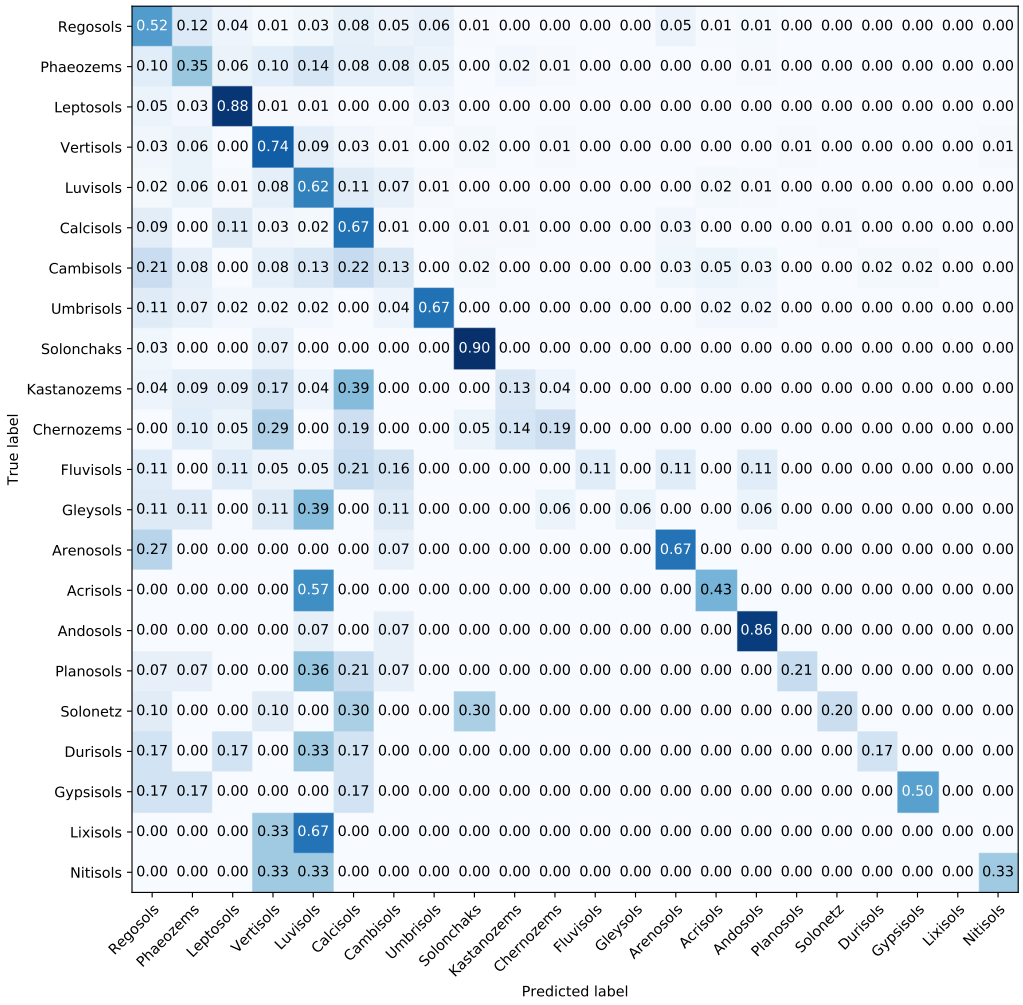


Figure 6.14: Confusion matrix for the standard depth data representation, with an imputation based on $k = 2$, using the Random Forest classifier, ordered by the most representative classes.

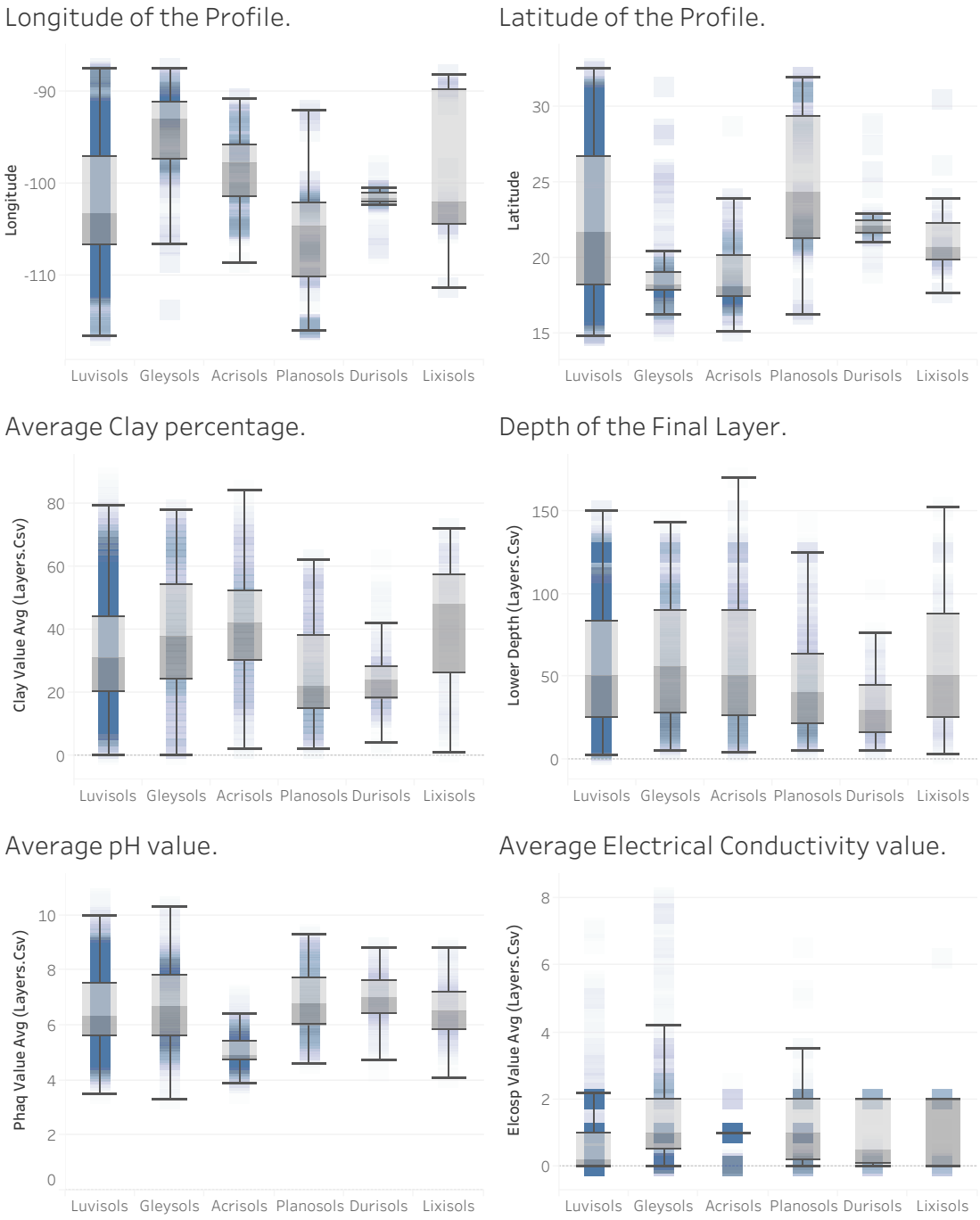


Figure 6.15: Comparison of the 6 most important featured, arranged by their importance. The chemical properties are averaged over all the layers for each class.

6.5 Other Experimentations

We also developed some ideas for possible experimentations, to improve the results, or to test different methodologies. Some of those tests are represented in this section, together with an analysis of their results.

6.5.1 One vs All and Ensemble Models

One of the experimentations, which was developed and tested, is the creation of one-vs-all models. This means that instead of using a single classifier, which can predict all the classes depending on the data it is presented, we create a model that can only distinguish if a profile belongs to one specific class.

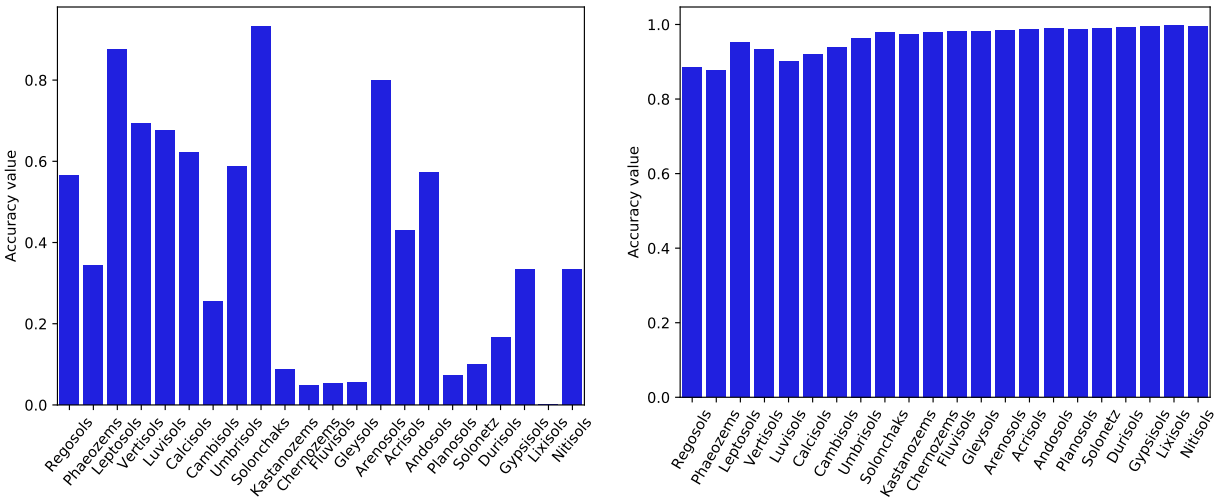
Using this method we reduce the problem to binary classification, either the profile being analysed represents the class that the model has specially trained for, or it belongs to a different unknown class. As we saw in Chapter 3, the number of classes is many times directly related to the performance of the ML algorithms, where a higher number of classes tends to lead to worse results [7].

One model was created for each class present in our data and these tests were made using the standard layers data representation, to train Random Forests models. The results for each model, namely the accuracy value, are shown in Figure 6.16, also with the results obtained with the general model, for each class present in our data. We can see that for the most part the accuracy value is greatly enhanced. This, however, might not be the best metric for this analysis, as described in Chapter 2 and, as such, new validation would be required to analyse these results. Unfortunately, due to time constraints, it was not possible, leaving this approach as possible future work to improve the performance of our algorithms.

This could be especially useful in problems where we do not require to know the exact class but are instead interested in knowing if the profile currently being analysed belongs or not to a class for which we are specifically looking for, due to its properties.

Another possible use case would be to join all these models, creating an ensemble. Since we have a model for each class that can occur in our data, we can simply enquire each of them about the probability, of the instance currently being analysed, belonging to the class in which they specialise. After we have the verdict for all the models in our ensemble, we choose the class that belongs to the model with the higher probability value.

With this method, we again have a multi-class classifier which can give us the information on the precise label of the profiles, given the appropriate data. Our tests regarding this approach generated results very similar to the general model, with Kappa values of 0.49 and accuracy of 0.54. However, this resulted in an increase of both the training time, since we are required to train a singular model for each class, as well as the prediction time since, again, each model in the ensemble needs to be consulted on their results.



(a) *General Model* (b) *One vs All Models*

Figure 6.16: On the left, the accuracy values for each class, as given by the general model. On the right we can see the accuracy values for each specific model created for every class.

6.5.2 Adding Remote Sensing Data

Another experimentation performed was the addition of some remote sensing data, to better inform our models about the profile and its region, thus improving our results. We used some of the features that, in the state of the art methods, provided the most information about the possible soil classification, namely DEM information which informs us about the physical properties of the terrain, and the NDVI, providing us with data about the vegetation of the region where our profiles are located.

For the DEM we extracted several features i.e. elevation, curvature, aspect and slope percentage. All of this information was taken from the *ALOS Global Digital Surface Model*[1] which is a global dataset generated from images collected by the Advanced Land Observing Satellite, from 2006 to 2011. This DEM contains a horizontal resolution of approximately 30 meters and is openly available online, released by the Japan Aerospace Exploration Agency (JAXA)[2].

The NDVI information was taken from the USGS Landsat 8, again with a 30m resolution[3]. To do the required processing of the images to generate the actual vegetation index, namely, on the red and near-infrared bands of the satellite images, we resorted to the Google Earth Engine [4]. We created three features namely the NDVI in the first 4 months of the year (January, February, March and April), the next four, and the final 4 months. With this, we tried not only to introduce the general NDVI of the region but also how it

[1] http://opentopo.sdsc.edu/datasetMetadata.jsp?otCollectionID=OT.112016.4326.2

[2] https://global.jaxa.jp/

[3] https://www.usgs.gov/land-resources/nli/landsat/landsat-8?qt-science_support_page_related_con=0#qt-science_support_page_related_con

[4] https://earthengine.google.com/

varies throughout the year. These values were obtained as an average of the months for which we had information, i.e. from 2013-03 to 2019-07, at the time of testing.

We then appended this information to the standard depth data representation, to each profile, and tested it using the Random Forests algorithm, again due to its results and ease of training. The results do not show any noticeable difference, staying mostly the same with a Kappa value of 0.49 and accuracy of 0.54. When analysing the feature importance, as determined by our model, none of the newly inserted features appears in the top 10 most important variables, for the classification, meaning they do not seem to provide any new information, although a more in-depth exploration of this ideia should be tested, as future work.

Chapter 7

Conclusion and Future Work

After the development of this masters' thesis, we now present the conclusions that result of this work and a discussion of the impact of the tests performed. We also point some possible future work, that could provide improvements to the methods used, and the performances of our prediction models.

7.1 Conclusions

We evaluated the performance of several machine learning algorithms for the soil classification problem, resorting only to its physical and chemical properties. The data treatment was a big part of this project, especially in the modelling of the data, where several representations were created and tested.

Overall, we have achieved comparable results, and many times better, than the current state of the art. However, a direct comparison is not possible, since these methods mostly employ different types of data, different regions and different soil classifications.

We have achieved Kappa values of 0.5 and accuracy of 0.55, for our best models and data representations. Regarding the representations tested, several provided results that are comparable to the best-performing methods. For the most part, the difference in the results while varying the modelling techniques' parameters are negligible, as long as sensible parameters are used for these representations. This is also true for k value, in the imputation method, with very similar results for all the variations tested.mn

The clustering methods did not provide the results we initially expected, showing very little relation with the actual soil classes. This means that, while the properties of the soil are related to the classes, they do not provide a clear separation between classes. Therefore we were not able to use a broader, but better performing, classification system based on the grouping of the WRB classes made by the clustering method.

The algorithms that showed better performance were the Random Forests and the Gradient Tree Boosting. Both of these algorithms showed very similar performance, however, GTB can usually extract more information across different data representations. Regarding the neural network-based algorithms, the performance was very poor both with the RNN and DNN, reaching a Kappa index of 0.27 and 0.31 respectively. This is probably due to the lack of data since these algorithms tend to require a lot more information to learn the problem correctly.

7.2 Future Work

For possible future work, we would like to add more remote sensing variables, that have been used in the state of the art methods. Although some initial tests were made in this direction without an increase of performance of our models, we expect that adding more complex information, and possibly with a better resolution than that currently being used, may provide a significant improvement on the results.

One other method that was tested, although only with a shallow exploration, was the use of one-vs-all classifiers, instead of the general models that were created throughout this research. This method seemed to provide improvements in our results, but a further analysis both on the actual performance of these classifiers, as well as on the optimisation of this method, must be taken into attention. As such, this is another area to research, inside the soil classification.

Another point to explore would be the balancing of the data, namely reducing the larger classes and increasing the information of the minority classes. This could be done by using some of the data balancing methods used in many machine learning problems, some of them even mentioned in Chapter 3.

To improve the results we also could try reducing the number of classes, which is quite excessive for machine learning classification problems. To do this, other than the clustering methods tested, we would like to find similar classes with the help of soil experts, and merge some of them into groups. This would again reduce the specificity of our classification but would provide us with more certainty regarding the predictions of our models.

Bibliography

[1] D. Arrouays et al. *GlobalSoilMap: basis of the global spatial soil information system*. CRC press, 2014.

[2] F. K. Barthold et al. "Land use and climate control the spatial distribution of soil types in the grasslands of Inner Mongolia." In: *Journal of Arid Environments* 88 (2013), pp. 194–205.

[3] B. Bhattacharya and D. P. Solomatine. "Machine learning in soil classification." In: *Neural networks* 19.2 (2006), pp. 186–195.

[4] P. Bholowalia and A. Kumar. "EBK-means: A clustering technique based on elbow method and k-means in WSN." In: *International Journal of Computer Applications* 105.9 (2014).

[5] L. Breiman. "Random forests." In: *Machine learning* 45.1 (2001), pp. 5–32.

[6] L. Breiman. *Classification and regression trees*. Routledge, 2017.

[7] C. W. Brungard et al. "Machine learning for predicting soil classes in three semi-arid landscapes." In: *Geoderma* 239 (2015), pp. 68–83.

[8] T. Chen and C. Guestrin. "Xgboost: A scalable tree boosting system." In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM. 2016, pp. 785–794.

[9] T. Chen and C. Guestrin. "XGBoost: A Scalable Tree Boosting System." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, 2016, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

[10] P. Chivenge et al. "Long-term impact of reduced tillage and residue management on soil carbon stabilization: Implications for conservation agriculture on contrasting soils." In: *Soil and Tillage Research* 94.2 (2007), pp. 328–337.

[11] F. Chollet et al. *Keras*. https://keras.io. 2015.

[12] F. Chollet. *Deep Learning with Python*. 1st. Greenwich, CT, USA: Manning Publications Co., 2017. ISBN: 1617294438, 9781617294433.

[13] R. G. Congalton. "A review of assessing the accuracy of classifications of remotely sensed data." In: *Remote sensing of environment* 37.1 (1991), pp. 35–46.

[14] N. L. Crookston and A. O. Finley. "yaImpute: an R package for kNN imputation." In: *Journal of Statistical Software. 23 (10). 16 p.* (2008).

[15] D. Dias et al. "Soil classification based on physical and chemical properties using random forests." In: *EPIA Conference on Artificial Intelligence*. Springer. 2019, pp. 212–223.

[16] FAO/UNESCO. *FAO/UNESCO Soil Map of the World*. URL: http://www.fao.org/soils-portal/soil-survey/soil-maps-and-databases/faounesco-soil-map-of-the-world/en/ (visited on 01/14/2019).

[17] A. Graves et al. "Speech recognition with deep recurrent neural networks." In: *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE. 2013, pp. 6645–6649.

[18] T. Hengl et al. "SoilGrids250m: Global gridded soil information based on machine learning." In: *PLoS one* 12.2 (2017).

[19] B. Heung et al. "An overview and comparison of machine-learning techniques for classification purposes in digital soil mapping." In: *Geoderma* 265 (2016), pp. 62–77.

[20] S. Hochreiter and J. Schmidhuber. "Long short-term memory." In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[21] K. O. Hounkpatin et al. "Predicting reference soil groups using legacy data: A data pruning and Random Forest approach for tropical environment (Dano catchment, Burkina Faso)." In: *Scientific reports* 8.1 (2018), p. 9959.

[22] R. Isbell. *The Australian soil classification*. CSIRO publishing, 2016.

[23] ISRIC. *SoilGrids*. URL: https://soilgrids.org/ (visited on 01/14/2019).

[24] W. Jeune et al. "Multinomial Logistic Regression and Random Forest Classifiers in Digital Mapping of Soil Classes in Western Haiti." In: *Rev Bras Cienc Solo* 42 (2018), e0170133.

[25] B. Krause et al. "Multiplicative LSTM for sequence modelling." In: *arXiv preprint arXiv:1609.07959* (2016).

[26] A. Krizhevsky et al. "Imagenet classification with deep convolutional neural networks." In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[27] Y. LeCun et al. "Deep learning." In: *nature* 521.7553 (2015), p. 436.

[28] S. Li et al. "Independently recurrent neural network (indrnn): Building a longer and deeper rnn." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5457–5466.

[29] X.-Y. Liu et al. "Exploratory undersampling for class-imbalance learning." In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2 (2009), pp. 539–550.

[30] J. MacQueen et al. "Some methods for classification and analysis of multivariate observations." In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[31] M. Meier et al. "Digital Soil Mapping Using Machine Learning Algorithms in a Tropical Mountainous Area." In: *Revista Brasileira de Ciência do Solo* 42 (2018).

[32] M. Minsky and S. A. Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[33] J. R. A. Moniz and D. Krueger. "Nested lstms." In: *arXiv preprint arXiv:1801.10308* (2018).

[34] A. Mosier et al. "Impact of agriculture on soil consumption of atmospheric CH 4 and a comparison of CH 4 and N 2 O flux in subarctic, temperate and tropical grasslands." In: *Nutrient Cycling in Agroecosystems* 49.1-3 (1997), pp. 71–83.

[35] F. Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), p. 386.

[36] P. J. Rousseeuw. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis." In: *Journal of computational and applied mathematics* 20 (1987), pp. 53–65.

[37] U. Shani et al. "Plant response to the soil environment: An analytical model integrating yield, water, soil type, and salinity." In: *Water Resources Research* 43 (2007).

[38] A. Sharififar et al. "Addressing the issue of digital mapping of soil classes with imbalanced class observations." In: *Geoderma* 350 (2019), pp. 84–92.

[39] P. Shrivastava and R. Kumar. "Soil salinity: a serious environmental issue and plant growth promoting bacteria as one of the tools for its alleviation." In: *Saudi journal of biological sciences* 22.2 (2015).

[40] L. N. Smith. "Cyclical learning rates for training neural networks." In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2017, pp. 464–472.

[41] E. Solos. "Sistema brasileiro de classificação de solos." In: *Centro Nacional de Pesquisa de Solos: Rio de Janeiro* (2013).

[42] C. Tallec and Y. Ollivier. "Can recurrent neural networks warp time?" In: *arXiv preprint arXiv:1804.11188* (2018).

[43] S. S. S. USA. *Soil taxonomy: a basic system of soil classification for making and interpreting soil surveys*. US Government Printing Office, 1999.

[44] P. Werbos and P. J. (Paul John. "Beyond regression : new tools for prediction and analysis in the behavioral sciences /." In: (Jan. 1974).

[45] I. W. G. Wrb. "World Reference Base for Soil Resources 2014, update 2015 International soil classification system for naming soils and creating legends for soil maps." In: *World Soil Resources Reports No. 106* (2015), p. 192.

[46] Z. Zhang et al. "Exploring the clinical features of narcolepsy type 1 versus narcolepsy type 2 from European Narcolepsy Network database with machine learning." In: *Scientific Reports* 8 (Dec. 2018). DOI: 10.1038/s41598-018-28840-w.