POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Multi-task Feature Selection for Drought Monitoring: a conditional mutual information approach

Master of Science in
Computer Science and Engineering

Author: **Lorenzo Nardi**

Student ID: 939517
Advisor: Prof. Marcello Restelli
Co-advisors: Prof. Andrea Castelletti, Prof. Matteo Giuliani, Dott. Alberto Metelli, Dott. Paolo Bonetti
Academic Year: 2021-2022

Abstract

The issue of climate change is one of the major topics of discussion in recent years, given the ecosystem effects that threaten the future of the next generations. Rising temperatures, mainly driven by humans' growing need for resources, have increased the frequency of observations of drought periods. This phenomenon is typically associated with a lack of water and contributes to the reduction of land to grow food, with significant economic and social consequences. Several studies have been proposed in the literature to study and monitor the drought phenomenon through indices calculated from remotely sensed data. Among them, the Normalized Difference Vegetation Index (NDVI) has been widely adopted in different Machine Learning techniques for monitoring and predicting drought periods. Specifically, this thesis aims to advance the FRIDA (FRamework for Index-based Drought Analysis) framework for monitoring drought events on a larger geographical scale, by employing multivariate feature selection based on information theory concepts and the Multi-Task Learning paradigm. The proposed approach selects the most informative subset of climatological features from sub-regions that differ in geographic and demographic characteristics, but whose drought conditions might depend on correlated drivers. The different classes of learning model choices use candidate features to produce results in two different contexts: reconstructing the drought index in a regression problem and monitoring cultivable soil conditions using a classification approach. The analysis of the first regression setting shows that an autoregressive model obtains better results than models using extracted data; in the classification case, Multi-Task models demonstrate satisfactory performances considering the limitations imposed by noisy input measurements and the scarcity of samples for the training phase.

Keywords: Drought, Machine Learning, Feature Selection, Conditional Mutual Information, Multi-Task Learning, Knowledge Sharing, Artificial Neural Networks, Autoregression, Classification

Sommario

Il tema del cambiamento climatico è uno dei principali argomenti di discussione degli ultimi anni, visti gli effetti sugli ecosistemi che minacciano il futuro delle prossime generazioni. L'innalzamento delle temperature, principalmente guidato dal crescente bisogno di risorse da parte dell'uomo, ha incrementato la frequenza di periodi di siccità. Questo fenomeno è tipicamente associato alla mancanza d'acqua e contribuisce alla riduzione dei terreni per la coltivazione del cibo, con notevoli conseguenze economiche e sociali. In letteratura sono stati proposti diversi studi per studiare e monitorare il fenomeno della siccità attraverso indici calcolati da dati telerilevati. Tra questi, il Normalized Difference Vegetation Index (NDVI) è stato ampiamente adottato in diverse tecniche di Machine Learning per il monitoraggio e la previsione dei periodi di siccità. In particolare, questa tesi si propone di progredire il framework FRIDA (FRamework for Index-based Drought Analysis) per il monitoraggio degli eventi di siccità su una scala geografica più ampia, impiegando una selezione multivariata delle variabili basata sui concetti della teoria dell'informazione e sul paradigma del Multi-Task Learning. L'approccio proposto seleziona il sottoinsieme più informativo di *features* climatologiche da sottoregioni che differiscono per caratteristiche geografiche e demografiche, ma le cui condizioni di siccità potrebbero dipendere da fattori correlati. Le classi di modelli di apprendimento scelte utilizzano le variabili selezionate per produrre risultati in due contesti diversi: la ricostruzione dell'indice di siccità in un problema di regressione; il monitoraggio delle condizioni del suolo coltivabile utilizzando un approccio di classificazione. L'analisi del primo problema di regressione ha mostrato che un modello autoregressivo ottiene risultati migliori rispetto a quelli impiegati, che invece utilizzano esclusivamente le variabili scelte; nel caso della classificazione, i modelli Multi-Task dimostrano prestazioni soddisfacenti, considerando le limitazioni imposte dagli input rumorosi e dalla scarsità di dati per la fase di addestramento.

Parole chiave: Siccità, Machine Learning, Selezione di Variabili, Mutua Informazione Condizionata, Multi-Task Learning, Condivisione della Conoscenza, Reti Neural Artificiali, Autoregressione, Classificazione

Contents

List of Figures

List of Tables

1 | Introduction

In recent years, there has been a lot of discussion about *climate change*. The media have contributed in making public opinion particularly aware of environmental issues, spreading studies and public events that highlighted the catastrophic consequences that a phenomenon of this magnitude could have on planet Earth and the survival of mankind.

Among the effects of climate change, *global warming* is particularly relevant due to its impact on the ecosystem. Even if changes in the global climate can happen for natural causes, researches believes that the increasing trends in global temperatures are mainly driven by human activity. Earth's atmosphere is composed by different types of gases, such as oxygen and nitrogen, and the so-called *greenhouse gases*, among which carbon dioxide, nitrous dioxide and methane. Incoming sunlight hits the planet's surface, that is partially absorbed, while the rest gets reflected. The reflected energy is partially dispersed in space, but a percentage remains trapped in the Earth's atmosphere due to the effect of greenhouses gases. This contributes to increase the temperature of the planet's surface even more, leading to an effect also known as *greenhouse effect*. Although this effect occurs naturally, an high concentration of greenhouse gases in the atmosphere contributes trapping growing amounts of solar energy, which translates on increasing global temperatures at an accelerating rate.

Increasing temperatures have devastating long-term effects on the planet climate patterns, which involves the occurrence of extreme climatic events more and more frequently. One of such effects are the *droughts*.

A drought is described as a prolonged condition of water supply deficit, whether it is referred as rainfalls, surface water or ground water. The impacts of droughts can be distressing for biodiversity and food production, but not all droughts are the same. Droughts can vary in the size of the area affected, their intensity and their duration. There are several types of droughts:

- *Meteorological drought*: this type of drought relates directly to rainfall deficiency. Continued low rainfall can determine less runoff and lead to situation of soil temperature above the average. For these reason, this is typically the first drought event

that occurs and that can lead to the observation of other types of droughts.

- *Agricultural drought* refers to the altered state of the crops caused by water shortage, that reflects on lower productivity and have mainly an economic impact.
- *Hydrological drought* is caused by the deficit of surface or ground water, which is derived from rainfall deficiency. The frequency and severity of these events is often defined on basin scale and its impacts are particularly relevant since the periods of recovery of water levels can be particularly long.
- *Operational drought* has a socioeconomic meaning, and it is caused by water supply systems failing to satisfy water demands of the society. This drought is typically associated with deficiencies in water supplies, that may be caused by other types of droughts, and excessive demand of water supplies. This effects are furtherly intensified in the case of insufficient design and maintenance of water distribution network.

Droughts have been a part of the earth natural cycle for a long time, but climate change has exacerbated the problem. Higher temperatures, greater water evaporation, and less plant coverage all contribute to a more frequent and severe droughts.

Drought has various impacts on ecosystems and society, producing a complex web of effects for which it is important to assess the consequences [1]. Assessing droughts impact is fundamental to provide critical information to support decision making processes, with the objective of deriving mitigation strategies [2] that may help in containing its devastating effects on the long term. In fact, a review of major studies in this area highlights that the majority of deaths associated with natural hazards are attributed to droughts. This is mainly caused by the consequences that this phenomena have on the ecosystem, such as agricultural crisis, livestock losses, water scarcity as well as epidemic diseases carried by mosquitoes breeding in stagnant water [3].

A potential solution to mitigate and counteract droughts effects is studying the repercussions that this climatic effects have on trophic levels [4]. Various studies have assessed the efficacy of satellite data in ecology researches [5] [6], among which the NDVI index is particulary suitable to detect or predict droughts [7] as its high correlation with parameters that indicates the amount of energy absorbed by solar radiation from vegetation [8], [9].

1.1. Objective of the Thesis

Motivations Traditional indicators used for drought detection may fail, especially in the case of highly regulated water systems, where the effects of meteorological and operational droughts are not necessarily overlapping [10]. As a starting point of this thesis, various researches that employed Machine Learning techniques in order to derive basin-specific droughts indicators have been reviewed [10]-[11]. The approach was the evaluation of drought indicators from context-specific hydrological data, by selecting a subset of relevant and non-redundant features for the chosen target variable. A further step is taken in [12], in which the previous framework was extended by using as target variable NDVI remote sensed data, that allows to analyze drought condition by assessing the state of the vegetation over a wider area. However, drought indicators in the cited analyses are used to study a single hydrological area.

Goals This thesis further extends previously introduced studies by employing multivariate feature selection techniques, in order to evaluate drought conditions on a particularly extensive hydrological area, such as the Po Valley. Starting from meteorological data of the analyzed study area, the goal is the reconstruction of the NDVI signal anomaly to predict the occurrence of droughts phenomena. The intermediate results induced us to consider investigating the drought state by classifying the data samples into three labels, identifying as many target signal conditions derived from the output analysis. In this way, we aim to improve the learning models generalization capability for the identification of a framework that allows the detection of droughts on any geographical area.

Contributions As a novelty approach, feature selection techniques are based on evaluating the *Conditional Mutual Information* [13] between the candidate variables, in order to find the subset of relevant and non-redundant features most suited to support the learning process of the selected target. A dataset of extracted candidate drought drivers and target variables of each sub-region in the study area has been created, which has been used to train and test model instances from several classes. The research was conducted both in a single-task setting, evaluating each of the identified hydrological sub-basins in the study area individually, and a multi-task setting, in which the input variable selection has been performed by considering as a target the anomalies of all the areas simultaneously. Finally, the obtained results highlight that the reconstruction of the NDVI index has not produced satisfactory results, as the anomaly signal demonstrated to be predictable only by looking at its historical values. However, the performances obtained when considering a multi-class classification problem show that this setting could be particularly fruitful

for evaluating drought conditions on multiple geographical areas.

1.2. Outline of contents

In this section, the overall structure of this thesis is presented, that includes Chapters 2, 3 that contain a theoretical overview of Machine Learning concepts extensively adopted in the experimental work, while Chapters 4 and 5 are oriented to the experimental results obtained on the extracted data.

Chapter 2 reviews the state of the art of feature selection algorithm, with a focus on information theoretical based techniques, that involves the use of the concepts of Mutual Information (MI) and Conditional Mutual Information (CMI) to identify the most *informative* variables to explain a given target.

In Chapter 3 Machine Learning algorithms employed to tackle Supervised Learning tasks are presented, such as Regression and Classification. Specifically, Linear Regression model is firstly introduced, due to its simplicity, then some of the most well-known ensemble algorithms are presented, that are Random Forests and Extremely Randomized Trees. In the final part, two Deep Learning models are then discussed, which are Feed-Forward Neural Networks and Recurrent Neural Networks. Then, an overview of the Multi-Task Learning paradigm is given, what are its purposes and benefits, and some methodologies that enable learning based on knowledge sharing among different targets.

After introducing the theoretical concepts of the experimental work, in Chapter 4 how data for the analysis has been extracted is showed, as well as the choices on the aggregation and the dimensionality reduction performed on the created dataset. Furthermore, this chapters also presents the results of the feature selection algorithm, based on variables CMI scores in both single-task and multi-task approach.

In Chapter 5 the performances of the previously introduced models on the generated dataset are showed. In particular, the results in the single-task setting are firstly reviewed, in which models deal with a Regression task for the NDVI anomaly reconstruction, and then a Classification approach to detect the anomaly behavior. In the final part, multi-task setting results are presented to evaluate the performance of the chosen models when the target is a vector of anomalies.

Finally, in Chapter 6 the conclusions that the results suggest and possible improvements are summarized.

2 | Feature Selection

In the recent years we have assisted to a significant amount of high-dimensional data across different fields. This growth poses a challenge to machine learning techniques as algorithms struggle to deal with a large amount of input functions, that leads to severe repercussions on processing times.

The problem was addressed as the *curse of dimensionality*, expression attributed to Richard Bellman[14] to highlight the difficulty of optimizing a learning function with too many input variables using brute force. Having a larger number of features rather than observations poses a risk of overfitting, which translates to poor performance of a learning model on unseen data.

Appropriate data preprocessing methods are then required to overcome the limitations that high-dimensionality imposes on machine learning techniques. Among these, the most important one is *feature selection*, whose relevance has grown so much that has become part of the learning process.

In general, with feature selection we mean a series of techniques aimed at identifying the subset of non redundant and relevant features, to learn a given task. The goal of this process is then reducing the dimension of the input variables of the learning model, which can help in increasing the performances and reducing the computational cost. A choice criterion is applied to discriminate which feature is relevant, thus eliminating the rest. This approach is one of the options for obtaining dimensionality reduction. Another way is obtained by finding a suitable combination of variables to summarize the informative content, which is the goal of *Principal Component Analysis* (PCA) [15]. In fact, PCA reduces dimensionality of the input data by applying a transformation on the original feature set: correlated variables are transformed into a smaller number of uncorrelated variables named *principal components*. While the feature selection approaches are different, each having a different computational complexity, the PCA algorithm is efficient, at the cost of reduced intepretability of the selected components.

Once a choice criterion is selected, the process of how to discover the set of relevant features is different among various techniques in literature. The scientific community has proposed

over the years a broad categorization of available feature selection techniques [16]. Usually, they are grouped in three categories [17]: *filters*, *wrappers*, *embedded methods*.

2.1. Wrapper feature selection

Wrapper methods assess input variables quality based on the prediction performance of a learning model.

Each subset of selected features is used as input for a learning algorithm, until some stopping criterion is met. Typically, this iteration is ended when it is obtained the highest prediction performance, or after selecting the desired number of inputs.

Since the detection of the relevant features is done by directly assessing how the performance of the model changes with different variables, this approach can lead to optimal prediction results. However, this process heavily relies on the model class used in the training process, meaning that the set of selected inputs could change when adopting a different predictor. In addition to that, this approach has a high computational cost for a large number d of inputs, since it would imply evaluating 2^d subsets.

There are several search algorithms for finding the optimal subset of features. Mainly, they can be divided into two classes [16]:

- **Selection Algorithms**, that sequentially add or remove features in the subset over which prediction performance of the model will be evaluated.
- **Heuristic Search Algorithms**, that use heuristic criteria to speed up the subset selection. In particular, they try to find the minimal subset of features that lead to good learning performance.

These methods have several drawbacks. As already discussed, if the number of samples is large, most of the time is spent in training the model. Moreover, some Heuristic Search Algorithms can evaluate the same subset of features multiple times as there is no history of past accuracy values obtained. Another disadvantage regards the trend to overfitting of the training model [17], since using the predictive performance as the only criterion for selection may result in picking the subset that works best only on the train data. Finally, the results of such methods depends on the model class used for evaluating the inputs, for this reason if the learning model were to change during the model selection, there would be needed to repeat the feature selection process.

2.2. Filters

Filter methods select features by ranking them on the basis of a criterion. The ranking allows to identify the best variables based on the evaluation criterion used, and to establish a threshold to remove features above/below it.

Unlike wrapper methods, filters are applied prior to the learning process, thus ensuring that the model only exploits relevant features for improving its generalization capability.

Because of this behavior and of the avoidance of repeatedly evaluating a model's predictions against a certain subset of the initial features, these methods represent the most efficient alternative for variable selection in terms of computational cost.

Furthermore, since filter methods does not rely on the underlying model class, they might lead to worse prediction performance on a single model, if compared to wrappers, however using only the input data allows to obtain a better generalization capability.

2.3. Embedded methods

Embedded methods represent a combination of filter and wrapper methods, in attempting to derive the benefits of both of them, by including variable selection in the training process while simultaneously trying to reduce the computation time.

This characteristic makes them more efficient than Wrappers, since there is no more the need of trying every combination of the input variables to improve the predictive performance of the model. Furthermore, the computation time is considerably faster since they avoid training a predictor all over again for every subset of features evaluated.

Regularization models (e.g., Ridge or Lasso regression [18] [19]) are an example of embedding method, as their objective is fitting a training model to minimize the prediction error and simultaneously forcing feature to assume only small, or even null, values, to contrast the impact of high data dimensionality on evaluation time.

However, the main disadvantage of this approach is due to the embedding of feature selection into learning process, which highlights a great dependence of these methods on the characteristics of the underlying learning model.

2.4. Data perspective

In this thesis, a filter-type feature selection strategy is employed to identify the subset of relevant and non-redundant variables with respect to the analyzed target, in order to be independent from the learning model by evaluating statistical dependencies on the data at disposal.

A more comprehensive categorization of feature selection methods is presented in [20] and is focused on the type of data to which existing feature selection algorithms are applied.

Following a top-down approach, the authors identify two types of data:

- *static* data, which refers to the traditional case where the input data have fixed dimensions that are already known before the learning process;
- *streaming* data, which includes the case of data whose size is not known a priori, as they are continuously generated with possibly new features uncovered.

Beginning with this distinction, more complex categories are then illustrated, which tend to group data according to their structure, source, or the links that can be found between different features. In the same paper, the authors tend to give an overview of the main feature selection algorithms used on a case-by-case basis, depending on the type of data, as shown in Figure 2.1.

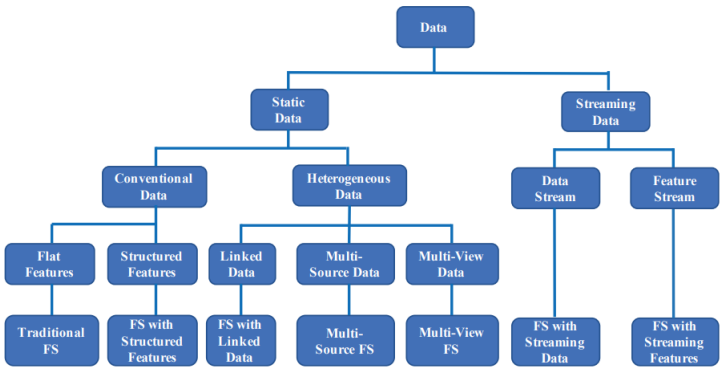


Figure 2.1: Different types of data, for each of them a category of feature selection algorithms is given. Picture from [20].

The most common case, which is the one of interest for this thesis, is represented by traditional feature selection algorithms, that work on the assumption of having flat features. They are unstructured data, whose samples are considered to be independent from each others (*i.i.d.*). These algorithms falls into one of the following four groups:

1. **Similarity based**, that is a set of methods for assessing the importance of a feature based on its ability to maintain data similarity. Having n input samples and d features, an affinity matrix of dimension $\mathbb{R}^{n \times n}$ encodes pairwise similarity between instances. Then, a way to select the k most relevant features in a subset S, is to maximize their utility: $\max_S U(S)$. Typically, the feature selection process happens by greedily selecting the top k features whose individual utility is maximized. The differences among algorithms in this category is how the affinity matrix is built. Examples are Laplacian Score [21], where the $(i, j) - th$ element of the affinity matrix is not null only if x_i is among the $p-$nearest neighbors of x_j; SPEC [22] that extends Laplacian score to be used both in unsupervised and supervised learning. The main drawback of these approaches is that they are mostly univariate, that is, each feature is evaluated independently and thus they cannot handle feature redundancy.

2. **Information theoretical based**, a family of algorithms that use filter criteria to measure the importance of features, typically by evaluating some metric that indicates how relevant is a given feature for the target label. The majority of information theory concepts applies only to discrete variables and that is why algorithms of this family usually employ discretization techniques when working with continuous feature values.

3. **Sparse learning**, an embedded method that leverages sparse regularizers to force many feature coefficients to be small or exactly zero, while attempting to minimize the fitting error. An example of sparse learning method is the feature selection using a ℓ_p-norm regularizer: a penalization term is added to the model's loss function, with a coefficient that regulates the sparsity of the weights W. LASSO [19] regularization uses a ℓ_1-norm, introducing more sparsity in the weight matrix as small weights are set to zero. Therefore, this methods performs both parameter shrinkage and variable selection, ad it allows to select features whose corresponding weights are large. Being embedded methods, their drawbacks have been already discussed in subsection 2.3.

4. **Statistical based**, in which features relevance is assessed by looking at some statistical measures. Most of them are filter-based methods, since they don't rely on learning algorithms to evaluate features' relevancy. A first approach is eliminating features whose variance is below a user-defined threshold, since these cannot help the learning model in discriminating samples from different classes. Another examples are: T-score [23], that evaluates the statistical difference between the means of two classes and selects features having a high t-score, that is an indicator on how is im-

portant an input for discriminating instances from different classes; Chi-square (χ^2) score [24] is a measure of independence between two events and it selects features whose χ^2 is higher. These methods are simple and they do not require a significant computational effort. However, as similarity-based variable selection, features are evaluated individually and they do not take into account feature redundancy.

2.4.1. Information theoretical based Feature Selection

The feature selection approach adopted in this thesis falls into the category of information-based methods, as the ranking, and subsequently the selection process, of features are made on the basis of how much informative they are with respect to the given target. The choice to use this approach is primarily dictated by the fact that information theory concepts, including *Mutual Information* and *Conditional Mutual Information*, are able to capture nonlinear dependencies between tested variables [25], with the ultimate goal of finding the subset of the most relevant and non-redundant features.

The next sections will introduce concepts and definitions that will simplify the presentation of the feature selection algorithm used.

Feature Relevance

As mentioned in Subsection 2.2 , the objective of filter methods is to filter out non-relevant features with respect to a target. Therefore, it is important to define the concept of relevance.

Given a set of n training samples, we define each training instance as a tuple $\langle X, Y \rangle$, where $X \subseteq \mathbb{R}^d$, and we denote with $P(X, Y)$ the probability distribution, defined over $X \times Y$, from which the n i.i.d. instances are drawn. Every further consideration does not make any assumption on both features, that can be either discrete or continuous and show some sort of unique structure (e.g., a graph or decision tree), and label, which can have an arbitrary dimension.

Various definitions of *relevance* have been proposed in literature; particularly interesting is the one proposed in [26], that identifies three disjoint relevance classes of features.

Let S_i the set containing all features but X_i , and s_i a value assignment to S_i, then:

Definition 2.4.1 (Strong Relevance)**.** X_i is strongly relevant if and only if there exist x_i, y, and s_i $P(X_i = x_i, S_i = s_i) > 0$ such that:

$$P(Y = y|X_i = x_i; S_i = s_i) \neq P(Y = y|S_i = s_i).$$

Definition 2.4.2 (Weak Relevance)**.** X_i is weakly relevant if and only if is not strongly relevant and there exists a subset of features S'_i of S_i for which there are some x_i, y, and s'_i with $P(X_i = x_i, S_i = s_i) > 0$ such that:

$$P(Y = y|X_i = x_i; S'_i = s'_i) \neq P(Y = y|S'_i = s'_i).$$

Definition 2.4.1 implies that feature i is relevant whenever the probability of occurrence of the label changes if we discard the knowledge about X_i value, given all the remaining features. Definition 2.4.2 instead states that the occurrence of a weakly relevant features can *sometimes* improve prediction accuracy. Therefore, if a weakly relevant feature is discarded from X, then the prediction accuracy *might* get worse, while the absence of a strongly relevant one will undoubtedly lead to performance degradation.

All strongly relevant features should be included in every optimal subset of features, but also weakly relevant features may be included. That is why a further objective in the selection of variables becomes identifying a non-redundant set of relevant features.

Feature Redundancy

The notion of redundancy among features it is commonly linked to their correlation: two completely correlated features are redundant. However a more formal notion of redundancy is devised in [27]. The following definition in given in [28].

Definition 2.4.3 (Markov Blanket)**.** Given a feature X_i, let $M_i \subset X$ $(X_i \notin M_i)$, M_i is said to be a Markov blanket for X_i if and only if:

$$P(X - M_i - X_i, Y|X_i, M_i) = P(X - M_i - X_i, Y|M_i).$$

Definition 2.4.3 states that M_i contains all the information that feature X_i has about the target, as well as the information about every other features. A procedure to obtain an optimal subset is checking if for the current set of features there exists a Markov blanket for X_i [28]. If that's the case, then X_i can be removed, thus leading to a backward elimination until only non-redundant features are left.

Since variable selection aims at removing irrelevant features and it can be proved that there are no Markov blankets for strongly relevant features [27], then the definition of redundancy follows.

Definition 2.4.4 (Feature Redundancy)**.** Let S be the current set of features, a feature is redundant and thus it should be discarded if and only if it is weakly relevant and has

a Markov blanket M_i within S.

2.4.2. Ranking criteria

There are several criteria that can be used to rank feature based on their relevance with respect to the target. This subsection introduced some criteria to use for evaluation and selection of the set of predictors for the target variable.

Linear correlation

The simplest criterion is to establish a ranking based on the linear correlation between each variable and the target.

The linear correlation of the i-th variable can be evaluated through the Pearson correlation coefficient [29], which has the following form:

$$R(i) = \frac{cov(X_i, Y)}{\sqrt{var(X_i)var(Y)}}. \tag{2.1}$$

However, this ranking criterion can only capture linear dependencies between the evaluated features with the target and it only focuses on relevance, without looking at the relationship between variables, that can lead to redundancy.

Mutual Information

A category of filter methods that measure the importance of features is based on information theoretical ranking [30], that leverages the concept of *Mutual Information* (MI).

Mutual Information can be considered as a measure of how much information is possible to get about a random variable by observing another one.

To understand the concept of MI, it is appropriate to introduce some preliminary notation that helps to clarify its mathematical meaning.

Definition 2.4.5 (Entropy)**.** It is defined *entropy* H of a random variable X that has p as probability density function the equation:

$$H(x) = \mathbb{E}_X[-\log(p(X))] = -\int p(x)\log p(x)dx. \tag{2.2}$$

Equation 2.2 describes the measure of uncertainty of the random variable in terms of the probability of its occurrence.

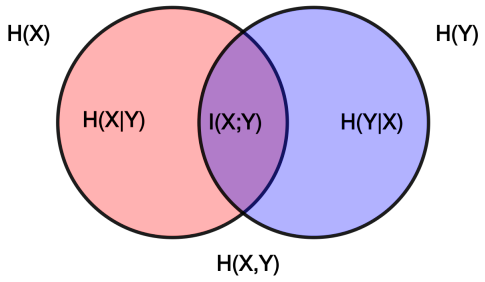


Figure 2.2: Relationships between entropies H and mutual information I of two random variables X and Y.

Definition 2.4.6 (Kullback-Leibler Divergence)**.** The Kullback-Leibler (KL) divergence of two distributions p and q is defined as:

$$D_{KL}(p\|q) = \mathbb{E}_{p(X)}[\frac{p(X)}{q(X)}] = \int p(x) \log \frac{p(x)}{q(x)} dx. \tag{2.3}$$

Then, we can define the Mutual Information between two random variables X and Y.

Definition 2.4.7 (Mutual Information)**.**

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) \\ &= D_{KL}(p(X,Y)||p(X)p(Y)) \\ &= \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy. \end{aligned} \tag{2.4}$$

MI measures how much information the random variable X carries about Y, and viceversa. In feature selection terms, this allows to quantify the importance of a feature subset in relation to the target vector.

Figure 2.2 shows the relationship between MI and entropy: $I(X;Y)$ represent the reduction of the uncertainty of Y occurrence after observing X.

Conditional Mutual Information

The definition of Mutual Information can be extended when taking into consideration the occurrence of a third random variable Z [31]. We can define the *Conditional Mutual*

Information (CMI) between X, Y giving the observation of Z as follows.

Definition 2.4.8 (Conditional Mutual Information)**.**

$$\begin{aligned} I(X;Y|Z) &= \mathbb{E}_Z[D_{KL}(p(X,Y|Z), p(X|Z)p(Y|Z))] \\ &= \int p(z) \iint p(x,y|z) \log \frac{p(x,y|z)}{p(x|z)p(y|z)} dxdydz. \end{aligned} \tag{2.5}$$

The CMI measures the information of two variables in the context of a third, and it can be rewritten in terms of entropies as:

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z). \tag{2.6}$$

This measures how much information X carries about Y that is not already contained in Z.

It is shown in [13] that the criterion that ranks features based on the highest CMI scores is equivalent to maximize the conditional likelihood between the inputs and the target.

The novelty of the application of CMI in this thesis is to apply for the first time an algorithm of feature selection based on conditional mutual information [32] in the context of droughts monitoring.

2.5. Selecting features via Conditional Mutual Information

In the past years Mutual Information has been employed into filter-based feature selection methods, in order to evaluate relevancy and redundancy [13]. In [32], authors also illustrate that the theoretical learning errors are linked with conditional mutual information of the considered subset of features.

Specifically, feature selection is applied by leveraging greedy sequential search algorithms and imposing a stopping condition that guarantees that the ideal regression/classification error, obtained using the evaluated subset of features, stays below a user-defined threshold.

2.5.1. Ideal regression error bound

For regression problems, in [32] it is assumed the existence of $B \in R$ such that $|Y| \leq B$. Considering $X_{\bar{A}}$ the subset of features of X that includes only the features in $\bar{A}$ and $\mathcal{G}_{\bar{A}} = \{g : X_{\bar{A}} \to Y\}$ the space of all functions that maps $X_{\bar{A}}$ to Y, then the ideal regression error, using the *mean squared error* (MSE) as criterion, can be defined as:

$$\inf_{g \in \mathcal{G}_{\bar{A}}} \mathbb{E}_{\mathbb{X}}[(Y - g(\boldsymbol{X}_{\bar{A}}))^2] \tag{2.7}$$

where the expected value is taken considering the set of all features and the target. To simplify the notation, given the set of indices A, we indicate the CMI between features X_A and Y, conditioned to $X_{\bar{A}}$ as:

$$\nu(A) = I(Y; X_A | X_{\bar{A}}). \tag{2.8}$$

Then, defining the irreducible error as $\sigma^2 = \mathbb{E}_{X,Y}[Y - \mathbb{E}[Y|X]^2]$, the first result found in [32] is that the *ideal* regression error, after removing features X_A, is bounded as:

$$\inf_{g \in \mathcal{G}_{\bar{A}}} \mathbb{E}_{\mathbb{X}}[(Y - g(\boldsymbol{X}_{\bar{A}}))^2] \leq \sigma^2 + 2B^2\nu(A), \tag{2.9}$$

which formalizes the fact that weakly relevant or highly redundant features can be safely removed without impacting on the prediction error significantly.

This results holds for an ideal error, that is the error obtained by a model of infinite capacity. In practice, learning models have limited capacity and thus the bound of equation 2.9 may be optimistic as there is the chance that the estimated CMI leads to the elimination of features that might be useful in practice.

2.5.2. Ideal classification error bound

The same considerations can be made when evaluating a classification problem, in which case the goal becomes to find the optimal function $g^* \in \mathcal{G}_{\bar{A}}$ that minimizes the ideal prediction error:

$$\inf_{g \in \mathcal{G}_{\bar{A}}} \mathbb{E}_{\mathbb{X}}[\mathbb{1}_{\{Y \neq g(X_{\bar{A}})\}}], \tag{2.10}$$

where $\mathbb{1}$ indicates the characteristic function over an event E. In this case, considering

$\epsilon = \mathbb{E}_{\mathbb{X}}[\mathbb{1}_{\{Y \neq \arg\max_{y \in Y} p(y|X)\}}]$ the *Bayes error*, the bound on the ideal prediction error obtained after removing features X_A has the following form:

$$\inf_{g \in \mathcal{G}_{\bar{A}}} \mathbb{E}_{\mathbb{X}}[\mathbb{1}_{\{Y \neq g(X_{\bar{A}})\}}] \leq \epsilon + \sqrt{2\nu(A)}. \tag{2.11}$$

This result, that is very similar to the previous one, remarks that the minimum ideal classification error, that can be achieved by a model using only the subset of features $X_{\bar{A}}$, is limited by the score $\nu(A)$.

2.5.3. Selection algorithms

To select the best subset of features, the authors in [32] propose the adoption of two search algorithms, introducing a stop condition to ensure that the ideal prediction error remains below a specific threshold. In the same paper, theoretical results ensure that the given threshold will be actually observed.

Given $\delta \geq 0$ the maximum error that a subset of features is allowed to introduce, the proposed stopping condition, respectively for regression and classification, are:

$$\sum_{h=1}^{t} I(Y; X_{i_h} | X_{\bar{A}_h} \backslash X_{i_h}) \geq \frac{\delta}{2B^2}, \tag{2.12}$$

$$\sum_{h=1}^{t} I(Y; X_{i_h} | X_{\bar{A}_h} \backslash X_{i_h}) \geq \frac{\delta^2}{2}. \tag{2.13}$$

Backward and forward selection algorithms are applied to select features via CMI score, using the stopping conditions 2.12 and 2.13 to control the theoretical prediction error.

Backward selection The backward selection algorithm [13] starts by considering the entire set of available features. At each step t, the removed feature is the one with the lowest CMI score relative to the target, conditioned on the remaining features in the set.

The stopping criteria of equation 2.12, 2.13 allow the error, added by the feature selection process to the ideal error, to be bounded below the specified threshold.

As highlighted in [13], this procedure is a greedy maximization of the selected features' conditional likelihood, with respect to the given target Y.

Forward selection In the forward selection algorithm, the search for the optimal subset of features starts from an empty set. Then, at each step t the feature that obtains the

highest score in terms of CMI between the target and the variable, is added, conditioned on the features already included within the resulting set.

The stopping conditions are the same of the backward approach, the main difference is faster execution time, as forward selection allows the CMI score to be evaluated on a smaller feature set.

2.5.4. Estimation of Conditional Mutual Information

So far, no assumptions have been made about the nature of the data on which to calculate CMI scores. However, most of the concepts of information theory are applicable only on discrete variables [20]. For this reason, when dealing with continuous values, there is a need for appropriate discretization techniques, which consequently do not allow an exact calculation of the mutual information value between two variables.

Several alternatives are proposed in literature to estimate mutual information [13] [33] [34], but the estimator used to validate the results, as well as the one used in the case study under consideration, is proposed in [35], which allows to estimate mutual information between variables and targets of different nature (e.g., discrete target but continuous features).

Taking advantage of the chain rule:

$$I(X;Y|Z) = I(X;Y,Z) - I(X;Z), \tag{2.14}$$

it is possible to employ the mutual information estimator to calculate the individual terms that concur in the definition of CMI.

In conclusion, the concept of feature selection and the rationale regarding its use to support the learning process was introduced. Next, a general overview of the main distinction of feature selection techniques was presented, with the related advantages and disadvantages of each. Then, in order to present the feature selection technique adopted for this thesis, some notions were presented to introduce the concept of Mutual Information, which allows the identification of the most informative features with respect to the target considered. At this point, a feature selection algorithm was presented that adopts a filter-type approach to rank features according to the CMI score between them and the target, given the other observed variables. This approach aims at selecting only relevant and non-redundant features, ensuring that the ideal prediction error is kept under control.

3 | Supervised Learning

Machine Learning is a branch of computer science and it is an inference process based on the paradigm of *induction*, that is the process of inferring general rules from observed data. This is opposite to the typical approach of computer science, that is based on *deduction*, in which rules are the starting point to generate new knowledge.

Machine learning has three main sub-fields [36]:

- **Supervised learning**: given a dataset $D = \{(x_i, y_i) \mid i = 1, 2, ..., n\}$, where x_i is the $i-$th input and x_i the $i-$th desired output, the goal is estimating the unknown model that maps each x_i to y_i, so that it is possible to estimate output values for unseen data. Supervised learning problems can be divided into three main categories, depending on the target.
  - *Classification*, where y_i is discrete and the goal is finding a model that correctly assigns a data sample x_i to one of the K discrete classes available.
  - *Regression*, where y_i is continuous and aims at finding the mathematical relationship that correctly maps the input samples to each target.
  - *Probability estimation*, if y_i is a probability distribution: similar to regression, but now the objective is finding mathematical relationship that is able to map each x_i to a probability distribution over possible events.
- **Unsupervised learning**: given a dataset $D = \{x_i \mid i = 1, 2, ..., n\}$, the goal of unsupervised machine learning algorithms is learning a more efficient representation of these unlabeled inputs. The following are examples of algorithms that fall into this category:
  - *Clustering* algorithms group unlabeled inputs that share similarities or differences into distinct categories (clusters). This allows to discover structures, or patterns, in the data and to classify samples based on their input properties. These algorithms may assign data points only to a single cluster (*hard clustering*), or providing a probability of a data point to be assigned in one

of the available clusters (*soft clustering*). It is important to notice that, differently from Classification, the output category is not defined by the given target, instead is provided by the Clustering algorithm itself only based on the discovered patterns in the inputs.

  - *Dimensionality reduction*, that aims at transforming the inputs from a high-dimensional space into a low-dimensional one, while keeping the significant properties of the original data points. The transformation can either be linear and nonlinear. Principal Component Analysis (PCA) is an example of linear transformation that projects input data into a lower dimensional space, on the direction of maximum variance within the data.

- **Reinforcement learning** aims at realizing autonomous agents capable of choosing actions, to pursue certain goals through the interaction with their environment. The action to be taken depends on the current state of the system and determines its future state. A numerical reward evaluates the quality of an action, so that higher rewards encourage correct behaviors of an agent. The goal of this class of tasks is learning the optimal policy π^* that maximizes the sum of future rewards. This learning task is usually modeled through Markov Decision Process [37] and can be carried out with different types of algorithms. These algorithms can be classified according to the use of a model describing the environment, the way the experience is collected (first-person or third-party), the type of representation of the system states and the actions to be performed (discrete or continuous).

In this thesis we apply supervised learning techniques, in which predictive models see both the input and the desired output as training examples.

In this chapter some key concepts will be introduced to frame the problem addressed in this thesis work.

As it will be explained in Chapter 4, one of the objectives of this work is the reconstruction of the NDVI index, originally obtained from satellite measurements, through more easily observable variables (features). This will be addressed as a regression problem since all the variables and the target are real-valued quantities.

3.1. Linear Regression

The simplest regression method is by leveraging a linear combination of the input variables [36]. The output of a linear regression model is then:

$$Y(X) = \hat{W}_0 + \hat{W}_1 X_1 + ... + \hat{W}_d X_d, \tag{3.1}$$

where d is the number of input features.

Linear Regression model aims to find the line that best fits the input data through the optimal combination of weights W, which minimizes the residual sum of squares between the true targets in the training data and the predictions of the linear model [19]:

$$\min_W \|XW - Y\|_2^2. \tag{3.2}$$

The main limitation of this approach is that it assumes a linear dependency between X and Y [36]. To extend the capabilities of this model, input variables can be transformed using a linear combination of a fixed number of nonlinear functions, known as *basis functions*. In this way, it is possible to make the prediction of such model a non-linear function of X. However, this does not eliminate the linearity with respect to the model parameters W, therefore the urge to use more complex models arises.

3.2. Ensemble models

The intuition behind ensemble learning is to combine multiple models rather than using only one, so that it is possible to find a trade-off between bias and variance error.

The models used in the ensemble can be of any type (e.g., linear regression, decision trees etc.) and ultimately the goal is compensate the error of an individual predictor by others.

Various explanations have been proposed in literature [38] [39] of the reason why the predictive performances often improves when employing ensemble methods:

- in the event of small amount of data available, these methods have a considerable advantage allowing to make an average between different hypotheses and reducing the risk of overfitting if only a single model is employed;
- using multiple predictors allows to reduce the risk of a falling into local optima during training phase thanks to the combination of multiple optimization processes, one for each available learner;
- extending the searching space by combining different models results in a predictor that fits better on the data space;
- balancing the variance and bias error by using proper techniques such as *bagging*

[40] and *boosting* [41].

An ensemble learning model ϕ uses an aggregation function G to aggregate the predictions of K learning models $\{f_1, ..., f_K\}$ into a single output:

$$\hat{y}_i = \phi(x_i) = G(f_1, f_2, ..., f_K). \tag{3.3}$$

For an ensemble model to perform well on a given task, the predictors f_i must be sufficiently diverse, so that their prediction errors are less correlated. In fact, it has been shown that this condition leads to an improvement in ensemble prediction performances [42]. The techniques usually adopted to differentiate the models are [43]:

- *input manipulation*: each predictor is trained using a different training subset. This method is particularly useful when small changes in the dataset may have a strong impact on the performances of a single learning model;
- *manipulated learning algorithm*: this approach consists in manipulating how each predictor's learning function converges to a solution. This can be achieved in different way, for example by employing the same model with different hyperparameters;
- *partitioning*: it provides diversity by dividing the dataset into subsets and use them to train a different predictor. Partitioning can either be horizontal, if the subsets contain the whole set of input features, or vertical, in which each predictor uses the same samples but with different input features;
- *ensemble hybridization* allows to achieve diversity by leveraging a combination of the previous techniques.

Another question arises when it must be considered which will be the output of the ensemble model. For example, when dealing with classic problems as binary classification and regression, the learning model is expected to produce as output a single value. Being composed of K learners, appropriate techniques must be selected in order to join each of the K predictions to a unique one.

The main approach is using *weighting methods*, that combine the outputs of the ensemble's predictor into a single one by assigning a weight to each individual model. Examples that falls into this category are *majority voting* for classification, which selects the most voted class as the output of the ensemble, and *averaging* the output of each predictor in the case of a regression problem.

Finally, the last distinction that helps categorize even more ensembles is whether the

output of a single model may guide the learning of other predictors of the ensemble or not. Most of the ensemble methods provide independent predictors in their architecture. In this thesis we deal with climatological data, which are known to have a huge noise component. Therefore, we will consider Random Forest and Extremely Randomized Trees, which are ensemble methods designed to reduce the variance of the model.

3.2.1. Random Forest

Random Forest[44] is one of the most popular ensemble algorithms [43] and comprises independent unpruned decision trees. This algorithm tries to overcome the limitation of decision trees, that is their high sensitivity to the input data, leading to high variance and possibly poor generalization results [45]. At each node of a tree, N input variables determine which is the outcome of the decision. To ensure each tree is less correlated to the others, such that the ensemble effectively reduces variance, random forests perform an ensemble hybridization process that comprises two randomization techniques:

- each tree is fitted on different samples of the input data, that includes a random number of *samples with replacement* from the training data (i.e., the same sample may appear multiple times when selecting data), leveraging the *bootstrap aggregation* (*bagging*) technique to generate the input dataset for each predictor;
- each decision tree is trained independently on each of the bootstrapped dataset, but with a random subset of input features on which perform the training process. Generally, each subset contains a selected number of features that is equal to the square root of the total number of variables in the training dataset, although it is possible to modify this parameter during cross-validation.

Bootstrap ensures that each predictor is trained on less correlated input datasets, so that the ensemble model is more robust to changes in the training dataset. This helps to reduce variance in a noisy dataset [40]. The process of selecting a random subset of features at each split allows to avoid trees in the ensemble to be highly correlated with each other: if one or few features are highly relevant for the target, many of the K trees will select those features, causing them to be highly correlated. The selection of the cut points for each of the subsets of features is performed by choosing the *optimum split* with respect to a selected measure (e.g., entropy). Finally, to merge the outputs of the K predictors into a single one, the algorithm leverages proper weighting techniques based on the type of problem.

Random forests tend to not overfit and the injection of proper randomness helps in achieving good prediction performances, although they tend to show better results in classifica-

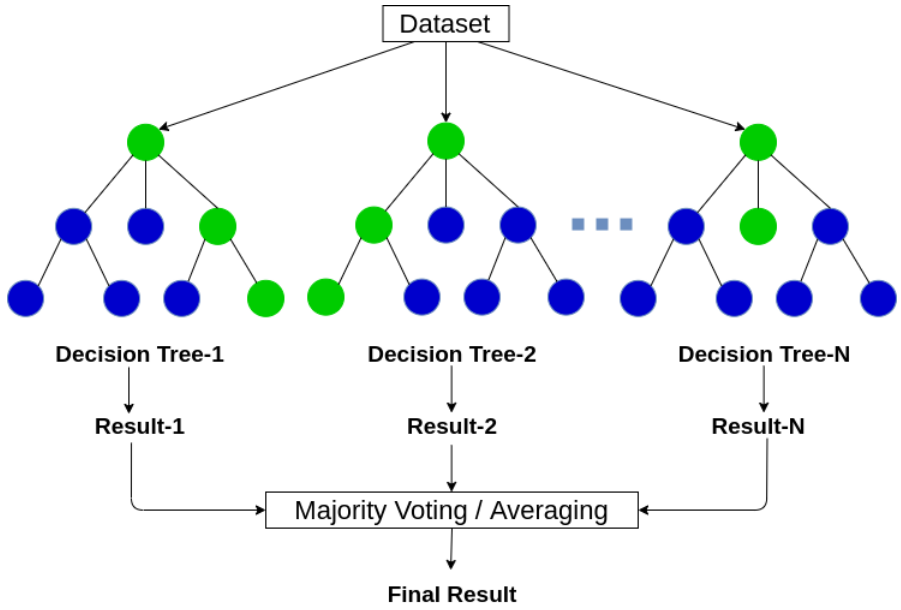


Figure 3.1: Graphical representation of random forest algorithm. Each tree of the ensemble is fitted to a random sample of the dataset and then, at each node, a subset of features is evaluated until the model is able to produce an output. Picture from [46].

tion problems, rather than regression [44]. Nevertheless, a recent found shows that this ensemble seems to be more robust and stable than other popular machine learning models such as neural networks and extreme learning machines, especially if, as in the case study we will consider, the number of samples in the training set is not very large [47].

3.2.2. Extremely Randomized Trees

Extremely Randomized Trees trees [48], or Extra Trees in short, are a generalization of Random Forest ensembles, the difference is that this algorithm leverages a different randomness technique during the training process to generate distinct predictors in the ensemble: each tree uses only a subset of the original features during the training phase. However, extremely randomized trees algorithm chooses the cut points in a complete random way. This allows the ensemble to gain an advantage in terms of computational cost with respect to Random Forest and it reduces the correlation between different trees.

From bias/variance analysis, this algorithm leads to a reduction in variance thanks to the random splits. Moreover, given the simplicity of the node splitting procedure, the computational cost is much smaller than in other ensemble based methods, including Random Forests.

The main parameters of the ExtraTrees algorithm are K, $nmin$ and M, where [48]:

- K is the number of feature to randomly select at each node;
- *nmin* is the minimum number of samples for splitting a node;
- M is the number of estimators, that is, the number of trees in the ensemble. This parameter determines the variance reduction.

In this thesis we only show the results obtained with this ensemble approach in Chapter 5,due to its better efficiency in terms of computational costs, compared to Random Forests.

Although, both Extremely Randomized Trees and Random Forest have been shown similar performances when feature selection process is nearly optimal [48], the randomization processes employed to build the trees in the Extra Trees algorithm allow them to be less dependent on each other, reducing the variance.

3.3. Deep Learning

Machine learning algorithms have been employed in various research fields and they have been widely used to address distinct tasks such as text mining, image classification, stock market prediction and so forth. With the increasing availability of complex data, a branch of machine learning known as *deep learning* has been shown to perform better than its predecessors [49].

The intuition behind deep learning algorithms is to implement complex model whose structure and functioning attempt to emulate those of the human brain. These complex models are referred as *Artificial Neural Networks* (ANN).

In the simplest case, a neural network can be imagined as a black-box that receives a certain of number of inputs and returns an output. Going deeper and exploring what's inside the black-box, these models can be represented by a graph structure, whose nodes are called *neurons*, that are grouped into several *layers*. Neurons in two consecutive layers interact with each other through *weighted connections*, which transform the output of neurons in a layer and give it as input to the neurons in the next layer. Each neuron consists of a *bias* term and an *activation function*, which processes the input and establishes the output value.

The complexity of a neural network depends on the number of the *hidden layers*, that are those between the input and the output layers, the direction of the connections and the activation functions used in each node. Complex models allows to discover and learn complicated relationships between input variables and targets, at the cost of increased computational complexity especially when dealing with high-dimensional datasets.

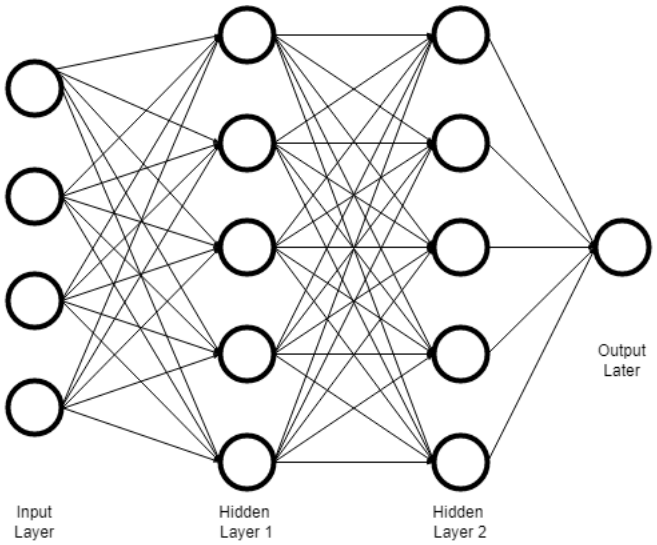


Figure 3.2: General architecture of a Feed-forward Neural Network, with two hidden layers.

The topology of the network's graph depends on the type of the neural network we are considering, but in the next sections the discussion will be limited to two ANN models: *Feed-Forward* neural networks and *Recurrent* neural networks. The first one is usually applied on *independent and identically distributed* (*i.i.d*) unstructured data, while the latter is specifically designed for time-series.

3.3.1. Feed-forward Networks

Feed-forward Neural Networks (FFNN) are the earliest and simplest form of neural networks, whose general architecture is represented in Figure 3.2.

The input data is fed forward to one layer to the next, until the output layer is reached. Each layer contains one or more neurons, that receive inputs from all the units in the previous layer and send output to all the units in the next layer.

The goal of such models is to approximate a function f^* that describes the relationship between the input data X and the target data Y:

$$Y = f^*(X). \tag{3.4}$$

A FFNN model learns a mapping:

$$Y = f(X; \theta), \tag{3.5}$$

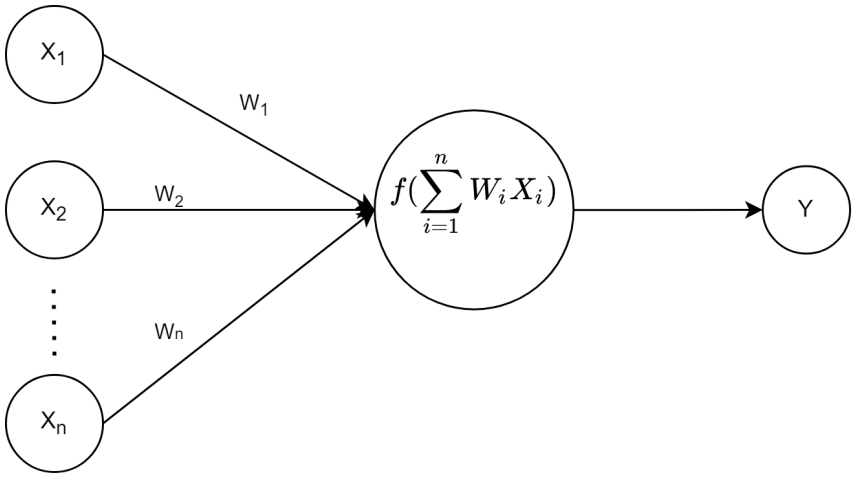


Figure 3.3: Basic structure of a single neuron.

where θ are the learnable parameters whose value is tuned during training phase to learn the best function that approximates $f*$.

The structure of a single neuron is highlighted in Figure 3.3, and its output is computed as:

$$Y = f(\sum_{i=1}^{n} W_i X_i + W_0). \tag{3.6}$$

The term W_0 is the bias, that is used to adjust the output so that it helps the model fit best for the given input. Function f is the activation function, which decides whether the neuron will fire its output or not.

The activation functions can be divided into two types:

1. Linear activation functions.
2. Nonlinear activation functions.

Typically, f takes the form of a nonlinear activation function to better model the complex relationships between the given data. The following are among the most frequently used nonlinear activation functions in practical applications.

Sigmoid Function The Sigmoid function can be considered the differentiable version of the Step function, since its values are between 0 and 1 and it is used especially in the case of having a probability prediction as output.

$$f(z) = \frac{1}{1 + e^{-z}}. \tag{3.7}$$

Its main disadvantage is related with the *vanishing gradient* problem. In fact, during the process of updating the network weights, it is necessary to employ the first derivative of the function as a multiplicative term, whose values range from 0 to 0.25, which hinders the learning process.

Hyperbolic Tangent The Hyperbolic Tangent is a scaled version of the Sigmoid:

$$f(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}, \tag{3.8}$$

since its values range between $[-1, 1]$ is particularly suited for binary classification. As the previous function is continuous and differentiable in all points, the difference being that negative inputs will be mapped to strongly negative, while zero inputs will be mapped around zero. However, being the scaled versions of the previous function it does not solve the vanishing gradient issue.

Rectified Linear Unit (ReLU) Function The ReLU function takes the form:

$$f(z) = \begin{cases} z, & \text{if } z \geq 0 \\ z, & \text{if } z \geq 0. \end{cases} \tag{3.9}$$

The function takes values between $[0, \infty)$, thus is unbounded and non-differentiable at zero. The advantages with respect to the previous alternatives is the fact that, due to the mapping of negative values to 0, it leads to sparse activation of the neurons, making the computation efficient and easy. However, since gradients for negative inputs are zero, this may lead to areas of *dead neurons*, that are never activated during the training process.

Softmax Function This function is different from all the previous ones due to the fact that it produces multiple outputs for input data.

$$f(z_i) = \frac{e_i^z}{\sum_j e_j^x}. \tag{3.10}$$

For this peculiarity, it is employed in multiclass classification problems, when the network's output is a probability distribution over predicted output labels.

The choice of activation function is typically made based on the type of problem for which the network is built. It has been shown that sigmoid and tanh are rarely used to due the vanishing gradient problem, while ReLU is the most popular choice for hidden neurons'

activation function [50].

Feed-Forward networks are capable of approximating a huge number of nonlinear functions, hence the justification for their use. This result is stated in the *universal approximation theorem*[51], that specifies that a Feed-Forward network with at least one hidden layer can approximate any continuous function on a closed and bounded subset of $\mathbb{R}$. However, this does not mean that the learning algorithm employed is guaranteed to learn any function the network is able to represent. This may happen for several reasons, such as failing to compute the right parameters value to approximate the desired function or overfitting of the train data. This insight suggests that there is no definitive machine learning model, as it is formalized in the so-called *no free lunch* theorem [52].

In [53], some bounds are provided on the network's hidden layer size to be able to approximate a wide class of functions. In the worst case, the number of hidden units is exponential, that is the case of one hidden unit for any input configuration to evaluate. Then, a Feed-Forward neural network with a single hidden layer would be enough to represent any function, but the size of such single hidden layer would lead to poor generalization capacity of the model. That is why is preferable to use *deep models*, since having large number of hidden layers with few hidden neurons can reduce the computational effort required for the training process as well as the cumulative generalization error.

Focusing on supervised learning tasks, the training process implies updating the value of the weights given the error computed on the network's output, with respect to the target's true value. Since the nonlinearity of the activation functions causes the loss function to become non-convex, the network is trained by leveraging iterative optimization algorithms whose goal is minimizing the gradient of the loss function. The application of such optimization algorithms is also known as **backpropagation**, since the calculation of the gradient goes backward through the network: the gradient is partially computed in one layer and the result is reused in the computation of the previous ones.

3.3.2. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) [54], are neural network models specialized in processing sequential data, typically of fixed length, although some models are also capable of processing sequences of variable length.

The main difference with FFNNs is that different parts of the model share the same

parameters. This allows the network to generalize across input sequences, since the same piece of information may reappear in different positions among the sequences that the model receives as input. This sequential data is basically ordered data in which the values of samples' sequence are related to each other. A simple example would be time series data, in which each sample represents the value taken by the signal at a given time instant.

It is particularly hard to generalize about this type of data by exploiting a normal FFNN, since the information goes only in one direction and the model has no memory of the previously seen inputs to predict future values of the target.

In RNNs instead, the information is processed through a loop, which makes these model to consider both the current input and the information learned from the samples seen before.

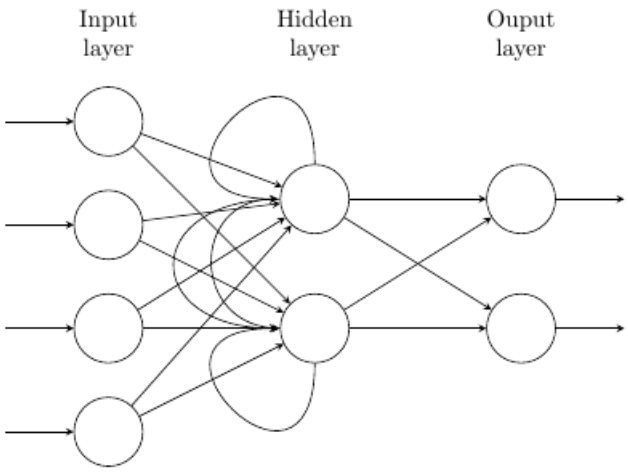


Figure 3.4: General architecture of a RNN [36].

Different RNN topologies can be identified depending on the number of input sequences the model receives and the number of outputs returned:

- **one-to-one** architecture provides that the model receives in input one sequence and produces a single time step prediction;
- **one-to-many**, in which the input is a single sequence and the model is required to output predictions for more than one time step;
- **many-to-one**, where the input are a certain number of sequence from which the model is required to make a single step prediction;
- **many-to-many**, in which from many input sequences, the model is required to

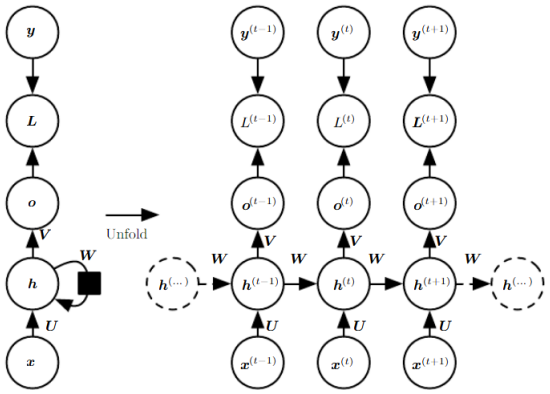


Figure 3.5: Design architecture of a recurrent network that takes as input a sequence of x values and returns a prediction at each time step [36].

output multi step predictions.

Further differences come from design pattern for the RNN hidden architecture. All of the next considerations, will be made assuming a recurrent network that produces an output at each time step, with connections between hidden nodes, as shown in Figure 3.5.

Considering $h^{(t)}$ the value of the hidden units which represents the state of the network, defined as

$$h^{(t)} = f(h^{(t-1}, x^{(t)}; \theta), \tag{3.11}$$

the network uses $h^{(t)}$ as a memory of what learned during the current iteration, and uses it to make predictions in the next time steps. Training process starts from a random initialization of $h^{(0)}$, then for each time step from 1 to τ, hidden neurons receive as input the memory of the previous iteration and the current sample of the sequence. Then, it is evaluated the loss between the current prediction and the target's true value and finally the weights are accordingly updated.

The learning process again exploits the Backpropagation algorithm, since the unfolded version of the recurrent network reminds the structure of a Feed-Forward neural network, as the same layer is repeated an number of τ times, where τ indicates the maximum length of the input sequence.

However, in case of long sequence input, this type of RNN are particularly prone to the vanishing gradient problem. In fact, in backpropagation the gradients are computed going from the last time step of the sequence until the very first step. At the early layers, the gradient would be calculated as the product of all the gradients of the previous time steps, which can lead to extremely small values that do not allow the value of the weights to

be updated correctly. The dual situation that can occur is that of *exploding gradients*, in which the gradients values will be so large that weights' values become unstable. More advanced techniques for sequences exist, that are designed to solve some of the issues of RNN (e.g., LSTM) but, given the small amount of training samples available for the considered problem, they will not be applied in this work.

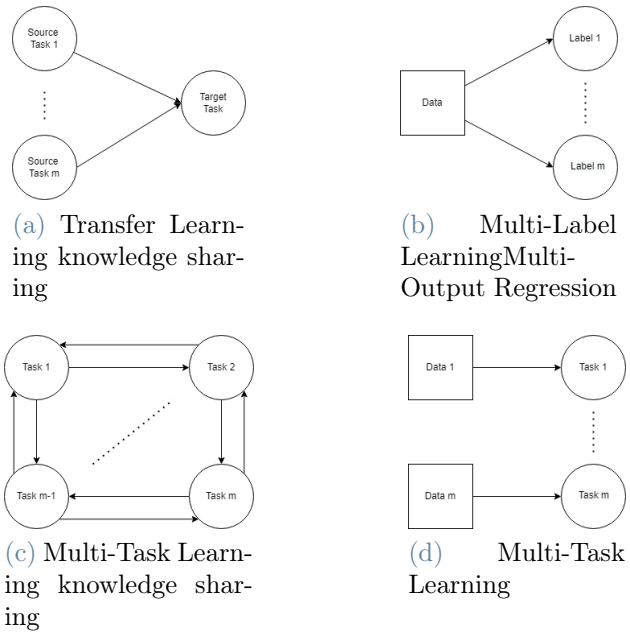
3.4. Multi-Task Learning

Machine Learning algorithms are traditionally focused on learning individual tasks, even though there can be situations in which it is useful to consider the information provided by training samples from other tasks in the same domain. This intuition resembles the idea that the human brain can reuse some form of knowledge learned in a task when facing a new problem. **Multi-Task Learning** (MTL) [55] is a paradigm that aims to improve generalization by leveraging knowledge from one task to another.

A major advantage of MTL is that it allows to leverage trainining data from multiple task during the training process, which mitigates data sparsity in the case the number of samples of a single task is not sufficient for obtaining good generalization performances. This learning paradigm has been remarkably successful in applications of several research fields, including some examples such as Natural Language Processing [56] or Computer Vision [57].

The use of data from multiple tasks is not new in Machine Learning algorithms. **Transfer Learning** (TL) is an example of a class of problems that aims at improving model performances on a target task by leveraging knowledge coming from source tasks [58]. Although this may suggests that the setting of TL is identical to the one of Multi-Task, the substantial difference is that in MLT all targets are equally important and the learning objective is *sharing* the knowledge of relevant information from tasks to help the model learn better.

Other examples involve models that are required to provide multiple outputs, such as **Multi-Label Learning** and **Multi-Output Regression**. These class of problems might be considered particular cases of Multi-Task Learning where all the target tasks share the same input data. However this consideration is no longer applicable in the case where different tasks require to use a different data. For example, in medicine applications it is expected to consider different biological characteristics to evaluate the probability of contracting a disease [59].



(a) Transfer Learning knowledge sharing

(b) Multi-Label LearningMulti-Output Regression

(c) Multi-Task Learning knowledge sharing

(d) Multi-Task Learning

Figure 3.6: Graphical comparisons of Multi-Task Learning paradigm with other settings that leverage multiplicity of targets.

This section introduces the results of various researches that have led to the development of MTL models for various Machine Learning applications.

3.4.1. Multi-Task Supervised Learning Models

At the beginning of this Chapter it has been introduced the concept of Supervised Learning, in which for a given task $\mathcal{T}$ a model has sees in input a dataset $\mathcal{D} = \{x_j, y_j\}_{j=1}^{n}$, comprised of n training samples and labels.

In a multi-task setting where the number of task is m, for a task $\mathcal{T}_i$ are available n_i samples: $\mathcal{D}_i = \{x_j^i, y_j^i\}_{j=1}^{n_i}$. A first classification of MTL models can be made based on the dimensionality of the available datasets:

- let d_k the number of input features for task k. If if $d_i = d_j, \forall i \neq j$, this means that different tasks lay in the same feature space and this is referred as **homogeneous features MTL**;
- if the above condition is not verified, then this is the setting of **heterogeneous features MTL**.

A further distinction refers to the type of the analyzed m tasks: if all of them fall in the same category of problems (e.g., Supervised Learning), then this is the situation of

homogeneous MTL; viceversa, this is referred as **hetereogeneous MTL**.

At this point, having introduced a generic overview on Multi-Task Learning, it is clear that this paradigm leverages on the concept of **sharing** to learn the given m tasks jointly, with the goal of improving generalization over all of them. Hence, some issues need to be addressed [60].

1. When a MTL model can be adopted to address a multi-task problem.
2. What it needs to be shared to allow knowledge sharing among tasks.
3. How the sharing is concretely implemented.

The answer to the first question is trivially through a choice made by human expert, although it may be attempted to formulate this decision as model-selection problems and then adopt techniques available in this case, such as cross-validation. However, this approach weighs heavily on the computational costs and it needs a large amount of training data. An advanced approach is using a multi-task model that can be traced back to its single-task version by means of some model parameters.

To enable the knowledge sharing among task, it is necessary to identify the elements that can allow a model to reuse the knowledge learned from a task i, during the learning process of the task j. With this in mind, there are three alternatives to achieve this result [60]:

- **Feature-based MTL** enables knowledge sharing by learning the set of common features that describe better the different tasks.
- **Parameter-based MTL** shares model parameters learned in a task to support the learning process of other tasks.
- **Instance-based MTL** shares knowledge between tasks by identifying useful data instances for other tasks when training on a given task.

As for the third point to be discussed, given the amount of alternatives existing in the literature, the next section introduces some of the most significant techniques for implementing the first two sharing methodologies described above, which are the most popular alternatives [60].

Feature-based MTL

This methodology of knowledge sharing is based on a simple intuition, that is: if the given tasks are somehow related, it is reasonable to suppose that they share a common feature

representation. Therefore, the aim of this methodology is learning a more powerful feature representation for all the given tasks that can boost the overall performances.

The relationship between the learned feature representation and the original features allow to further classify feature-based approaches into two categories.

Feature Transformation This approach involves a linear (or nonlinear) transformation of the original feature representation. One of the earliest examples of multi-task models are single layer Feed-Forward Neural Networks, receiving as input training samples of the m tasks and having as many m outputs from the last layer. Indeed, the single hidden layer output can be considered what the model is able to achieve by evaluating a combination of the input features.

Another example of MTL model in this category is **Multi-task feature learning** (MTFL) [61], a regularization framework which aims at minimizing the training loss on each task samples, while enforcing model parameters matrix to be row-sparse via a regularization term.

Lastly, deep learning models represent once again a viable alternative in this setting, as they are able to learn nonlinear feature representations for multiple tasks. This is achieved in various ways, for example sharing several layers of the model architecture between targets. More advanced solutions imply using **Generative Adversarial Networks** (GAN) [62] to minimize training loss of every task while maximizing the cross-entropy loss, in order to make it harder for the discriminative network to associate that representation to one of the targets. One last example of deep learning models in MTL are **Cross-Stitch Networks** [63], that are Convolutional Neural Networks able to share knowledge among models with the same architecture, but trained on different tasks, by using the *cross-stitch operation* to learn hidden features, which are combinations of different input variables. Figure 3.7 clarifies how this typology of networks allows the joint learning of multiple tasks, since cross-stitch units allow the layers of each network to receive a transformation of incoming features from both targets.

Feature Selection Techniques that fall into this category aims at learning a common feature representation by selecting a subset of the original features. In Chapter 2 are introduced several feature selection algorithms that are also suitable in the case of evaluating multiple targets. This approach can also be considered a special case of the previous one, where the transformation matrix is a sparse diagonal matrix, with non-null elements only in correspondence of the selected feature.

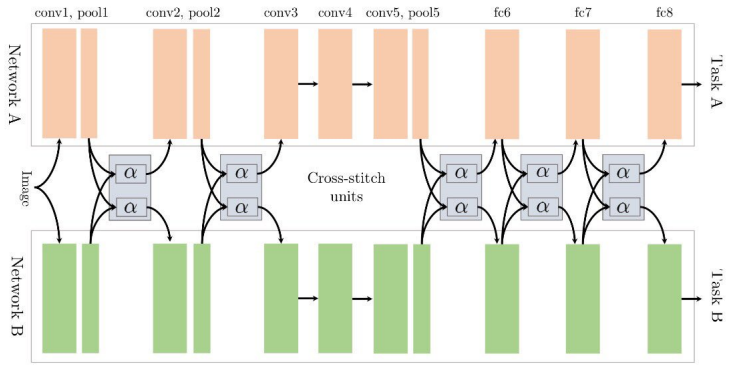


Figure 3.7: Example of architecture of a Cross-stitch Convolutional Network [63].

From a comparison between the two categories it emerges that the first approach is focused on improving the performance of the models on the objective tasks, while the second method allows to identify the set of features to generalize better on all the targets. For this reason, the second approach is more interpretable.

Parameter-based MTL

Parameter-based approach allows joint learning of tasks by sharing model parameters during the training process. This intuition relies on the fact that those parameters have been tuned to perform well on a given task, therefore they can be tweaked even more if the same are reused again to minimize the empirical loss of the other targets.

Several methods can be considered to be part of this MTL category, whose differences consist in which actual parameter is shared between the different tasks.

Low Rank Approach This methods aims at finding a low-rank parameter matrix for a model, based on the intuition that related tasks share the same model parameters. This is accomplished in [64], where the Frobenius norm induces a regularization on the model parameters matrix.

Another suitable approach is performing a regularization with the parameters matrix trace norm, that induces low-rank for the given matrix and it shows to perform well on MTL tasks such as gene expression pattern analysis [65].

Task Clustering To share model parameters among similar tasks, these methods first use clustering techniques to identify clusters of task that share some information.

In [66], authors separate the task-clustering from model-learning process, by grouping into clusters those task that share same parameters tuned in a single-task setting. Then,

after cluster are defined, data from each of these cluster tasks are pooled to allow learning a better model able to generalize better on them. Although this method is fairly simple to reproduce, its results may be not optimal due to inaccurate parameters learned in the single-task setting. Alternatively, a regularizer proposed in [67] groups tasks into clusters based on the variance of tasks within a cluster, as well as the variances between different clusters.

Finally, [68] explain how to extend the MTFL method in the case of multiple task clusters. In this setting, the squared trace norm regularizer is applied on each individual cluster to learn its structure. The number of clusters can be determined automatically, such as in [68], employing a structurally sparse regularizer to penalize all pairs of parameters not merged with each other.

Task Relation Learning A model is able to generalize on joint group of tasks by learning the relatedness between them. These relations can either come from model assumptions, given by a priori information or learned from data. This approach is particularly interesting, because it allows to develop models that can be easily replicated on different set of tasks, whenever information on task relatedness in not available.

In [69] is adopted a multi-task Gaussian process to define a prior for the functional value of each task training data. These functional values are weighted with the covariance between the considered pair of tasks. Other approaches are available, for example using a normal prior on the weight matrix [70], but the majority assumes that clusters are not overlapping, thus possibly restricting the generalization capability of the resulting multi-task model.

Decomposition This last approach is based on the parameter matrix decomposition into two or more sub-matrices. Generally, this implies the optimization of a generic objective function, comprised of the minimization of the empirical loss over each target task, given a set of constraints defined on the sub-matrices, and a decomposable regularization parameter with respect to each component matrix. This framework can be reduced to previously introduced approaches, such as MTFL or low-rank, that is why it is typically considered to be an improved version of those.

Having introduced a general overview of the methods that enables multi-task learning via parameter sharing, some conclusions can be drawn by analyzing their main characteristics. Low-rank is a powerful approach, as it can learn the subspace of the parameter matrix either directly or indirectly, in this case employing a kind of regularizer. However, its

main limitation is that it is mostly applicable to linear models [60].

Task clustering allows to identify similar tasks, its main limits being that it can ignore the negative correlation among tasks in different clusters and that the number of clusters is typically determined with model selection methods that introduce additional computational costs.

Task relation approach seems to be highly interesting, due to the fact that, in this setting, a model can learn both its parameters and the task relations, which improve interpretability of the results.

Finally, the decomposition approach can be considered as an extended version of other methods, employing multi-level parameters to model complex structures among tasks. However, the number of components must be carefully selected, since performances and computational costs highly depends on it.

Optimization for MTL Models

After providing some examples of MTL models, the next question to address is which are the optimization techniques that enable the joint learning of multiple tasks. This section briefly discusses the options available in literature for optimization in a multi-task setting.

In [60], three main classes of algorithms are presented.

- **Gradient Descent** algorithms and their variants can be used also in the case it is needed to optimize when multiple objective functions. In fact, a convenient way to represent the the objective function in a multi-task setting is to formulate it as a combination of each task objective function, such as a weighted sum [55]. Therefore, a gradient descent step, allows to optimized the shared parameters with respect to every task. This approach is fairly simple and allows to find a closed form solution in the case the objective functions are convex. However, this approach is not ideal in the case of deep MTL models, since the process of weights update can be undermined by conflicting task gradients, or higher order of magnitude of a task-specific gradient compared to the other task gradients [71].
- **Block Coordinate Descent** method supports scenarios in which the model parameters are partitioned. Each single block consists of a group of parameters shared by all tasks, while a few are only related to a specific task. This algorithms works by alternatively optimizing each partition, keeping this process independent from other blocks as they are kept fixed. Compared with the previous case, this method allows to reduce the computational complexity as each blocks represents an individual

optimization sub-problem [60].

- **Proximal method** [72] enables the optimization of nonsmooth objective functions in MTL models, that consists of smooth and nonsmooth functions. This optimization algorithms allow to solve such functions by leveraging a proximal problem, that replace smooth functions with quadratic terms that can boost the convergence rate and simplify the design of distributed optimization algorithms [60].

In conclusion, in this section the main components of Supervised Learning have been discussed. A first introduction on the problem class has been given, then various models to address Regression or Classification problems have been introduced. First the Linear Regression model has been reviewed, highlighting the fact that it is limited despite being easily interpretable. Then, Random Forests ad Extremely Randomized Trees have been discussed, which are among the most famous examples of Ensemble models, an attempt of boosting predictive performances by training several learning algorithms and combining their outputs. Next, two main Deep Learning algorithms have been overviewed, which are Feed-Forward Neural Networks and Recurrent Neural Networks. These are extremely powerful models, that are able to learn any continuous functions in a closed and bounded subset, the differences being that Recurrent networks are particularly suited to treat sequential data. These models have been used for experimental work. In the final part, the Multi-Task Learning setting has been briefly introduced, so that this section has outlined the theoretical foundations on which the experimental work is based. These models, in both the single and multi task setting, will be employed to obtain the results presented in Chapter 5.

4 | Dataset

This section presents the data used in this thesis. Specifically, the aim is to reconstruct the NDVI index through a combination of meteorological variables, which are easier to obtain than target satellite measurements, especially in case the satellite images are obscured by the presence of clouds [7]. This regression problem is tackled through the use of the learning models described in the previous section, aiming to identify the existence of a relationship between the input data and the target variable that allows to reconstruct the NDVI time-series signal.

In Section 4.1 the historical series of NDVI index is described. Then, in Sections 4.2,4.3 and 4.4, metereological data used in this work are discussed together with the feature extraction process applied for this case study. Finally, in Section 4.6 are presented motivations and techniques employed for dimensionality reduction of the input variables, which lead to the final dataset that is the starting point of all the Machine Learning approaches described in next sections.

4.1. NDVI Index

Long-term shifts in temperature and weather patterns are described as *climate change*. The causes of these deviations may be natural, such as variations in the solar cycle, even though from the end of the 19th century human activities have been among the main drivers of the rapid shifts we are observing nowadays [73]. The main effect of climate change is *global warming*, that is characterized, in general, by an increase in global average temperature and it is associated with weather phenomena such as floods, droughts, desertification, melting of ice, overheating of seas and rising oceans levels.

To establish strategies for preventing and containing climate effects on biodiversity, a possible approach is understanding the repercussions these effects have on trophic levels and interactions [4]. In latest years, several studies have suggested that satellite data could be pivotal in ecology researches [5] [6], one of them being the NDVI index.

The **Normalized Difference Vegetation Index** (NDVI) [74] is computed from red

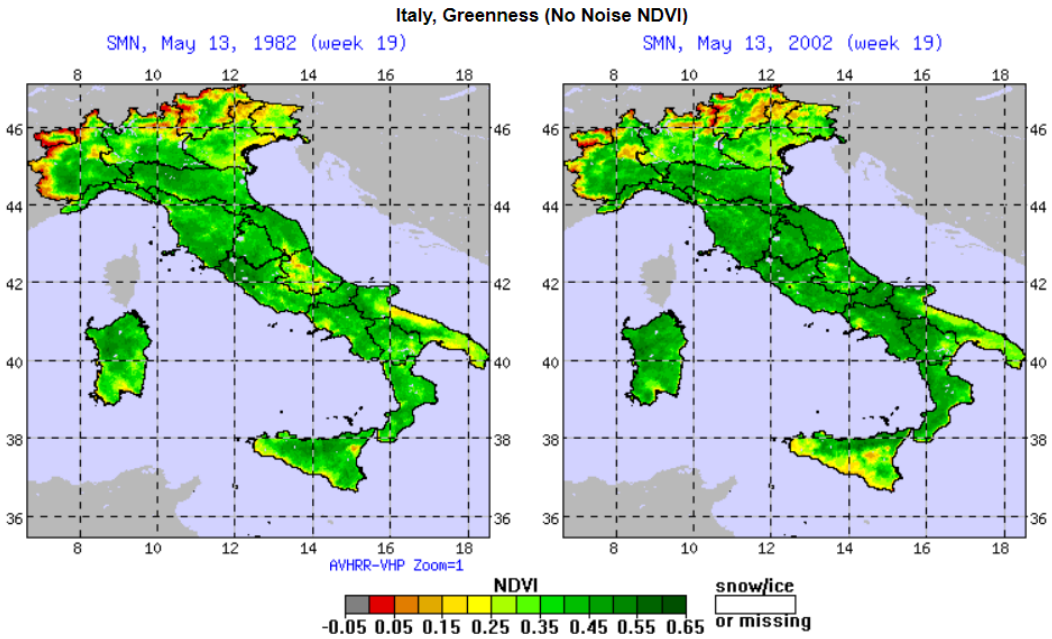


Figure 4.1: Comparison of smoothed NDVI values of the Italian territory over 20 years, during spring period. The higher values in the Alpine regions in 2000s highlight the reduction of the snow cover in the same seasonal period. Graphs obtained using [78].

and near-infrared band reflectance, and it has been proved to be an efficient indicator for monitoring vegetation health [75]:

$$NDVI = \frac{\rho_{NIR} - \rho_{RED}}{\rho_{NIR} + \rho_{RED}}. \tag{4.1}$$

Green leaves strongly absorb solar radiation in the blue and red spectral regions, while in the near-infrared spectral region are highly reflective and thus there is no radiation absorption [76], which leads to positive NDVI values. Other elements such as clouds, snow and bare soil have NDVI values that are almost zero, while negative values are showed by water [77]. A graphical example of NDVI data is shown in Figure 4.1, where the map shows the NDVI over the Italian territory in the same reference week. Because NDVI is highly correlated with some widely used measures to assess the fraction of energy foliage absorbs from solar radiations, such as Photosynthetically Active Radiation (PAR) or Leaf Area Index (LAI) [8] [9], it can be employed to detect or predict extreme climate events such as droughts, floods etc. [7].

Historical NDVI data is available over a wide time range from different providers with global coverage and distributed in several spatial resolution [79], which make it a convenient choice for evaluating vegetation health.

Disadvantages of the NDVI are its nonlinearity, being a ratio-based index, and its lagged response to drought phenomena when evaluating the impact of rainfall deficits on vegetation health, due to the residual moisture in the soil [80].

NDVI historical values have been extracted firstly from National Oceanic and Atmospheric Administration (NOAA) satellites [81], which provide data measured by Advanced Very High Resolution Radiometer (AVHRR) sensors from July 1981, with a spatial resolution of 4 kilometers. However, this data suffers from several problems, such as orbit drift of satellite NOAA-7 [82], post-launch sensor calibrations [83] and inconsistency between the sensors AVHR/2 (1981-2000) and AVHRR/3 (2000-present) [84]. For these reasons, as well as to employ data at a higher resolution for extracting the NDVI values only for the cultivable areas in the case study, as described in Section 4.5.1, NDVI data have been obtained from Aqua Moderate Resolution Imaging Spectroradiometer (MODIS) (2002–present) sensor [85]. This sensor is designed to have more stable orbits, improved spectral configurations for monitoring vegetation health and higher image resolution, since each pixel covers an area of 250 meters. Although in this way temporal depth is lost, since data is available on a shorter time frame, this choice was made to avoid the use of NOOA data that exhibits non-stationary trend due to systematic errors, which would not make sense to consider for this case study.

4.2. Temperature and Precipitation

Air temperature and precipitation can have major impacts on natural processes, and their data are critical for climate analysis. Changes in temperature and precipitation can substantially damage crops, influence the frequency and intensity of severe weather events, and affect the quality and quantity of water available for civil and industrial use.

Examples in literature [86] prove that these two meteorological variables have a direct association with the phenomenon of drought:

- increasing temperature trends are associated with growing drought coverage and persistency;
- decreasing precipitation mitigate the effects and the coverage of droughts.

Precipitation and temperature data are obtained from European Climate Assessment & Dataset (ECA&D) database [87]. Specifically, for this thesis, daily average temperature and daily precipitation sum have been extracted from a gridded dataset with spatial resolution of 0.1×0.1 lat-lon degrees over a grid of the European continent.

In Figure 4.2 it is shown an overview, over different time intervals, of average yearly trends in temperature and precipitation data in the European continent. It is evident that yearly temperatures exhibit an increasing trends in the latest sixty years, while precipitations are decreased over time.

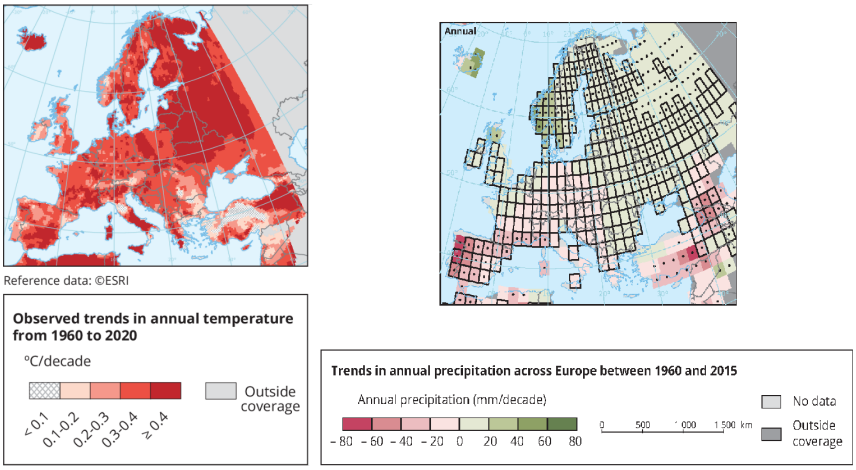


Figure 4.2: Annual temperature and precipitation trends over the European continent, over different time intervals [87].

4.3. Snow Depth

Snow is an important factor in hydrology as it determines the availability of water supplies that accumulate during winter and get released in spring and summer, affecting the conditions of crops. Due to climate change, shifts in temperature and precipitation patterns have had a particular impact on the snow cover depth in the Alps region [88], which is the one that affects aquifers level in the study area of this thesis. Measurements from 1971 to 2019 have shown the overall reduction of snow cover depth [89]: about 85% of locations show decreasing trends of snow cover during this period, which are most evident in spring and on the Italian side of the Alps. Furthermore, snow cover duration also shows a decreasing trend: the length of the snow season has been decreasing for an average of a month per year. Thus, snow is slow to accumulate in the fall and melts more rapidly in the spring, leading damages to ecosystems, hydrological regimes and the economy associated with winter sports.

Authors in [89] highlight that the causes of snow cover reduction are the rising temperatures, which lead to rainfalls rather than snowfalls, especially at low altitudes, and accelerate snow melting. Although the overall trend shows the reduction of snow presence in the Alpine region, extreme snow episodes are still possible from time to time, as shown

in Figure 4.3.

Snow depth data comes from the dataset [90]. Where not directly available, data were retrieved from web portals provided by the entities that manage data recording and distribution (ARPA Piemonte, ARPA Valle d'Aosta, ARPA Lombardia).

Figure 4.3: Time series of monthly mean of snow depth, averaged by 500 meters elevation bands. Values are showed only if more than five registration stations are active. Image by [89].

4.4. Lake Level

The increased variability of rainfalls and the increasing trend of temperatures are among the main drivers of droughts, which are becoming more frequent and severe [91]. Hydrological cycles propagate climate water deficit derived by droughts, causing groundwater levels, streamflows and lake levels to reduce. These effects may be separated both temporally and spatially from drought drivers, that is why this phenomenon is usually referred as hydrological drought [92].

In the study area of this thesis, presented in Section 4.5.1, some of the major lakes that contribute to the water supply of areas of high interest from both agricultural and tourism perspectives were selected. Since drought phenomena impact water availability of basins [93], lake levels on a given time span were considered among the candidate predictors for modeling the NDVI index.

The following list briefly presents the four lakes whose height levels are employed as candidate variables for the target, while Figure 4.4 shows their geographical location with

respect to the study area.

- **Como Lake**: third largest lake in Italy, with an area of 146 km^2, it is part of the pre-Alpine belt and it is of glacial origin. The lake reaches at most a width of just over 4 km and it is the Italian lake with the largest perimeter, approximately 170 km.
- **Maggiore Lake**: the second largest lake in Italy, although the northern part is on Swiss territory, with an area of 212 km^2; it is part of the lakes of the pre-Alpine belt and is of glacial origin, the maximum depth is 372 m. Lake Maggiore has a rather tapered and narrow shape with an overall length of 54 km, while the distance between its shores averages only 4 km.
- **Iseo Lake**: is the seventh largest lake in Italy and the fourth largest in Lombardy region. The basin is located at an altitude of 186 m above sea level and has an area of 65.3 km^2, with a maximum depth of 251 m.
- **Lugano Lake**: is located on the border between Italy and Switzerland. Its shores are divided between the Swiss region Ticino and the Italian region Lombardy. The lake has an area of 48.7 km^2, and a maximum depth of 288 m.

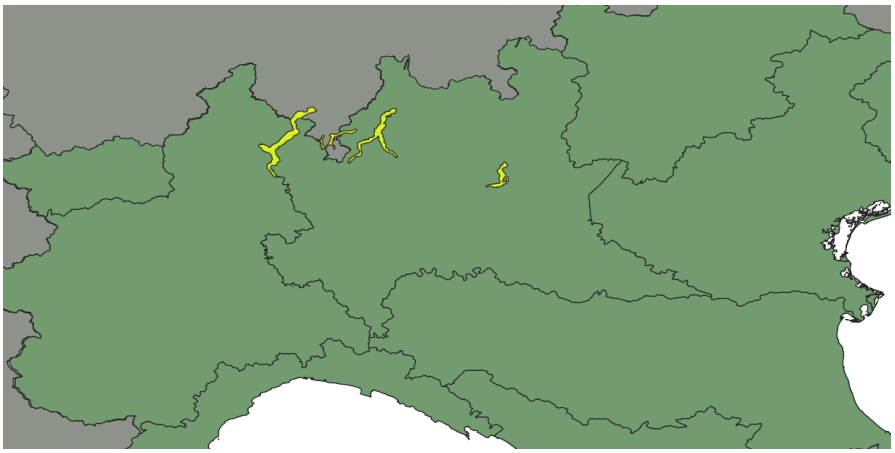


Figure 4.4: Geographical location of the considered lakes (in yellow) on the Italian territory (in green). Image obtained using [94].

4.5. Feature Extraction

This section introduces the study area considered for this thesis work and the techniques used to extract, aggregate and process NDVI and meteorological variables on this region. Due to the high dimensionality of the obtained dataset, as well as the high correlation

that variables exhibit among themselves, in Section 4.6 it is introduced the dimensionality reduction approach adopted and the resulting dataset is presented.

Since original NDVI data provided by NOAA are weekly averaged, each input variable has been extracted considering the weekly mean. From now on, when we refer to the value of an input or target variable, we will always refer to is weekly averaged value, unless otherwise specified.

4.5.1. Study area and novelty approach

For the study of NDVI index modeling, a step forward is taken from researches on drought index modeling that are the basis of this thesis. In [10], the Spanish basin of Jucar was the first test case for a framework aimed at reconstructing a drought index from more readily available variables than those used by national regulators. The good results obtained prompted the reuse of the same framework to evaluate its replicability on the Lake Como basin [11], whose hydrological characteristics are markedly different from the first case analyzed. However, this methodology involves employing target variables that are specific to the target area of the case study. Only in [12] NDVI index is used as the target variable, moreover evaluated over a larger geographic region, suggesting the possibility of replicating the experiment over a study area no longer tied to a specific hydrologic basin. For this reason, the approach of modeling a drought index based on NDVI is employed for the first time on an extended region, that is the Italian region of the Po Valley, mainly included in the hydrological basin of the Po River. Furthermore, previous studies only used a single-task approach, meaning that the learning goal was a single drought indicator. In this thesis, it is proposed a framework that evaluates both single and multi-task approaches for drought index modeling.

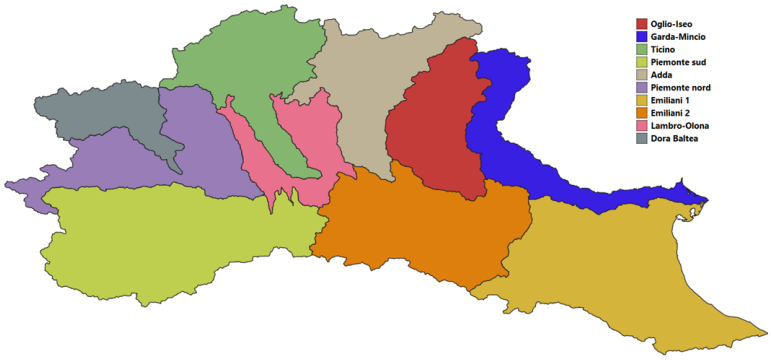


Figure 4.5: Study area considered for extraction of input and target variables. Shapefiles of the area have been created using QGIS software [94].

The Po Valley enjoys a robust diversified economy thanks also to the good distribution of the population in several medium-large urban areas in the eastern part, that is where there is the highest concentration of agricultural countryside. The western side has a much more impacting demographic weight, because the population is mainly concentrated in the large areas between the industrial cities of Milan, Turin and Genoa.

Given the size of the study area under analysis, as well as the diversity among the different geographic areas under consideration (e.g., a mountainous region such as Aosta Valley compared with a predominantly flat area such as the region of Emilia Romagna), ten sub-areas are defined for the feature extraction process, which correspond to the main hydrological sub-basins of Po River.

Although the ten sub-basins each have specific geographical characteristics, all are crossed by Po river, therefore it is reasonable to assume that they are correlated. For this reason, the study area configured in this way is particularly interesting to be evaluated in a Multi-Task setting.

4.5.2. Extraction of NDVI

NDVI data are available in NetCDF format, which also contains the geographic coordinates of the points to which each grid value refers. To crop the gridded NDVI dataset, so that it is possible to extract values only for points within a region, it has been employed QGIS software [94] to obtain the geographical coordinates of each sub-basin in the study area of Figure 4.5. An example of the resulting extraction is shown in Figure 4.6, that presents the average NDVI value of a given week extracted from NOAA AVHRR data [81] on the whole area examined.

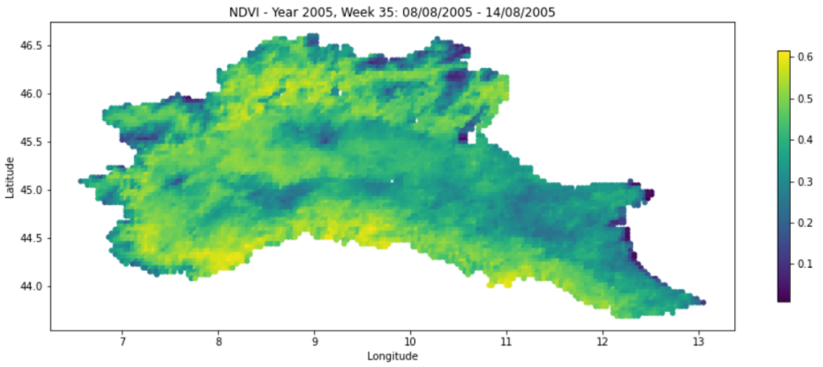


Figure 4.6: Sample of extracted NDVI over the study area, obtained using [94].

As discussed in Section 4.1, this data source has some data consistency problems, and given the low resolution of the NDVI grid, values extracted over the different study areas

often end up overlapping with each other. Therefore, it was preferred to use the data from the Aqua-MODIS sensor. A comparison of the trend of the extracted target over the study area from the years 2002 to 2020 can be found in Figure 4.7, where it can be noticed that there is practically no difference in the NDVI signal obtained for each study area. In fact, the only differences are notable in the time frame corresponding to years 2005, 2011 – 2012, 2017 – 2020.

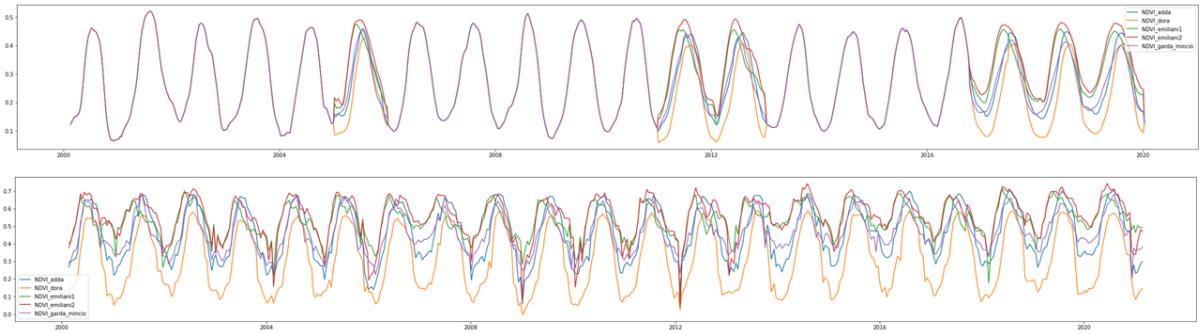


Figure 4.7: Comparison between AVHRR NDVI data (above) and Aqua-MODIS NDVI data (below) of the considered study area. In order not to compromise the visibility of the graph, we report the comparison of values on 5 zones only.

In the continuation of the thesis work, NDVI extraction was then limited to only cultivable areas in the Po Valley region. In this way, it is possible to obtain the average NDVI on regions not directly influenced by large urban centers or the prevalence of mountainous areas, confining the study of drought periods on areas of agricultural interest. Figure 4.8 shows the cultivable areas in the case study region, that are then considered to extract the NDVI values.

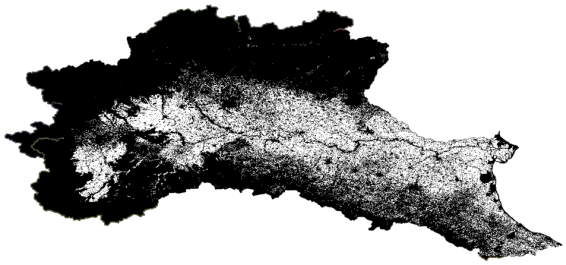


Figure 4.8: Cultivable areas in the Po Valley region. White pixels represent cultivable areas whose coordinates are used for NDVI extraction. The raster images with cultivable areas are obtained from [95].

.

As highlighted in Figure 4.7, the target signals have sinusoidal shapes and, if they were

considered as they are, the models would end up learning the seasonality of the NDVI indices. In this context, the goal is learning a target's peculiarity of a specific week in a given year, which is captured by how much the signal deviates from the average value it has in that week. This allows to obtain a true and easily interpretable drought index: 0 indicates a normal situation, positive values are associated with higher availability of water, while negative values mean water shortage. This setting is particularly interesting to evaluate as a classification problem, as it is shown in Subsection 5.1.4. In fact, in this case the models are required to determine the condition of the NDVI anomaly, encoded into three labels (i.e., *positive*, *negative* and *normal*), given the chosen inputs variables. For this reason, we focus on NDVI anomaly, which has been proved to be a more accurate index for monitoring droughts phenomena [96].

NDVI anomaly is obtained as follows.

$$a(t) = x(t) - M(t), \tag{4.2}$$

where:

- $a(t)$ is the anomaly of the mean NDVI signal in week t;
- $x(t)$ is the extracted NDVI signal in week t;
- $M(t)$ is the average NDVI in week t, obtained by considering all the extracted samples in the availability time window and averaging the target values of week t for every year.

Thanks to the NDVI anomaly, it is possible to remove the deterministic correlation of the target signals, as highlighted in Figure 4.9, due to the fact that NDVI has an increasing trend in summer, while a decreasing one during winter.

Lastly, NDVI anomalies obtained from the considered sub-basins show a positive linear trend with respect to the year. This behavior conditions the feature selection algorithm to choose a temporal indication, such as Year or Week variables, as the most informative input for the given target. This is not desirable, since once this type of variable is selected, the CMI scores of subsequent iterations are significantly lower by several orders of magnitude, or even negative, leading to an early stopping of the searching algorithm. In fact, this temporal indicators exhibit a linear correlation with the targets, while we are interested in capturing the nonlinear dependencies among NDVI and our chosen variables using the CMI. For this reasons, the NDVI anomaly signals are subject to a detrend operation, which involves fitting a linear model over each target and subtracting its values

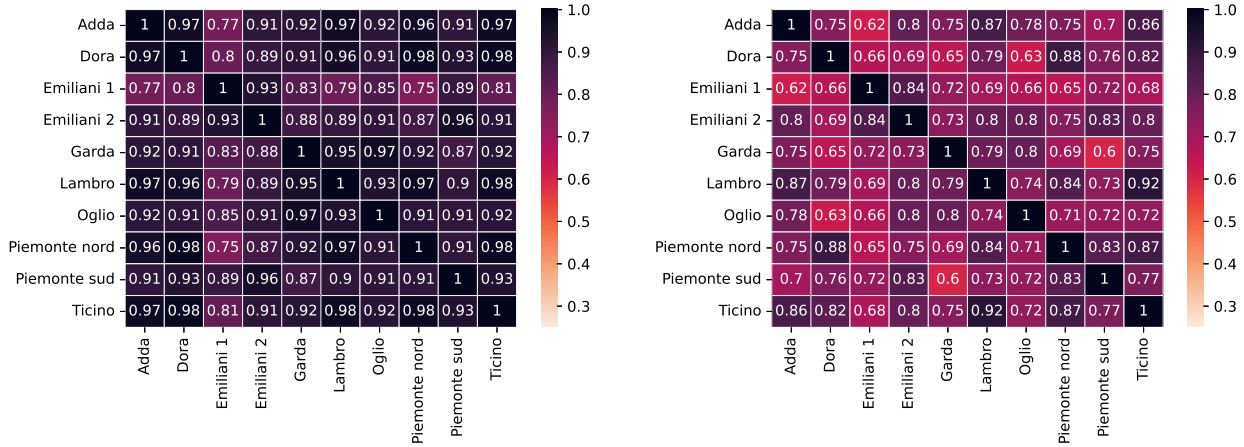


Figure 4.9: Cross-correlation between raw NDVI signals (left), NDVI anomalies (right).

from the original series. An example of detrended anomaly is shown in Figure 4.10, in which it is immediately evident that detrending brings signal values more centered around zero, without changing the shape of the curve.

4.5.3. Extraction of temperature and precipitation

Gridded values of temperature and precipitations provided by [87] are cropped using the shapefile of each of the sub-basins in Figure 4.5. In this way it is possible to extract temperature and precipitation values for the entire sub-basin from 1950 to 2021. For these variables, the geographical coordinates of cultivable areas were not employed for the dataset clipping. In fact, the extracted values over cultivable areas would not be statistically different from the southern regions where there is the highest concentration of crop lands, while for northern regions it is interesting to take into account temperature and precipitation fluctuations also in mountainous area.

4.5.4. Extraction of snow depth

The extraction of snow depth data is fairly simple, as there have been considered only data produced by recording stations that fall into the considered study area. The main challenge was the lack of consistency between registered values, due to inactive recording stations in summer periods and different time intervals for availability of data. To overcome these difficulties, we considered and aggregated data from recording stations in the study area, that remained active for at least 25 years, discarding all those that recorded measurements for short periods. Then, snow depth data from every region in the study area has been aggregated, to have an index of weekly accumulated snow as candidate

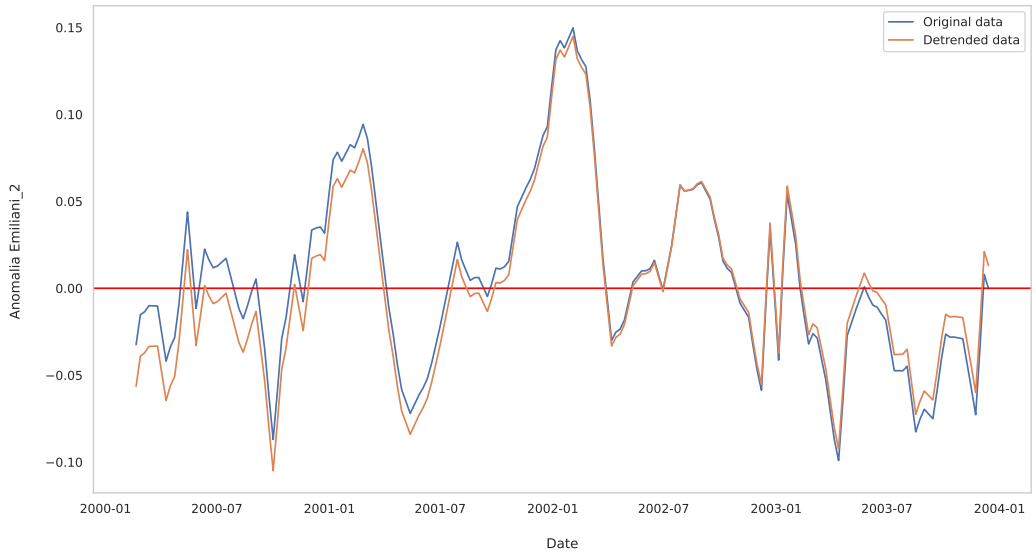


Figure 4.10: Comparison of NDVI anomaly and detrended NDVI anomaly signals of *Emiliani_2* area. Output is truncated to the first four years of the available time interval, to improve visibility.

predictor to model the chosen drought index.

4.5.5. Extraction of lake levels data

Lake levels data are immediately available and ready to use. Data come from ARPA Lombardia, the regional body responsible for controlling, monitoring and issuing opinions on environmental topics. The availability of data is not homogeneous across the considered basins, for this reason it has been decided to consider samples only up to 2019, as they were the latest available for Como Lake when experimental test have been carried out for this thesis.

4.5.6. Data aggregation

Extracted input variables are available over differing time horizons. As already discussed in Section 4.5, candidate features have been resampled to weekly mean values to be temporally consistent with the target values.

It is reasonable to assume that meteorological phenomena, such as increasing temperatures or reducing snow cover, may have a long-term impact on drought conditions. For this reason, rather than exploring the link between weekly values of candidate features and

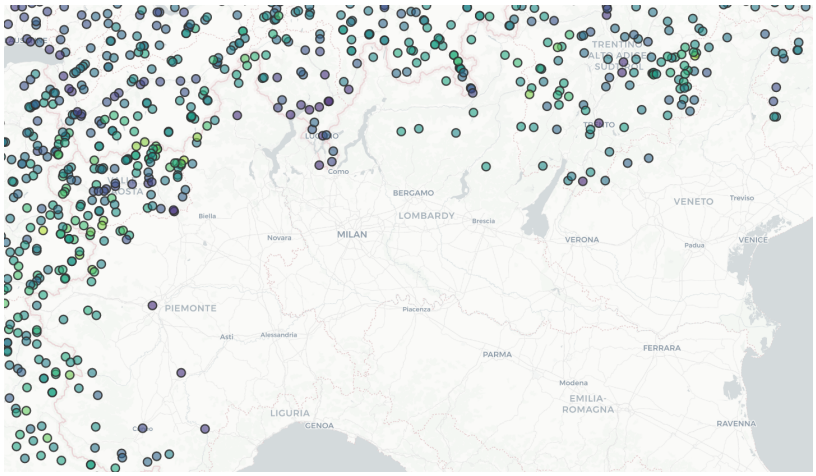


Figure 4.11: Overview of recording stations of snow depth (in cm) in the Alpine region [90]. Notice that this picture includes more stations than those considered for extracting the data, as detailed in Subsection 4.5.4.

the target, variables have been aggregated over $4, 8, 12, 16, 24$ weeks. In this way, it is possible to evaluate, for example, the availability of water during spring, that originates from melting snow cover that has accumulated over winter.

At this point, inputs were merged together on week and year values of their recording, together with the targets. Since candidate features are available on different observation intervals, it was decided to consider a time span of 20 years as a time interval for the considered area, in which there are measurements for all the selected data. With the selected aggregation, the dataset at this stage consists of 154 features.

Variable name	**Description**
PrecNw_AREA (1950-2021)	Average precipitation over AREA in the last N weeks.
TempNw_AREA (1950-2021)	Average temperature over AREA in the last N weeks.
AREA_SnowDepth_Mean_Nw (1981-2019)	Average snow cover depth over AREA in the last N weeks.
LAKE_AltezzaNw (various-2019)	Average LAKE level in the last N weeks.

Table 4.1: Extracted variables names with relative description. As already mentioned, the weekly aggregation varies as $N \in [4, 8, 12, 16, 24]$ weeks.

The obtained result is the first dataset that has been processed by the feature selection algorithm described in Chapter 2.

	Date	Year	Week	anomalia_adda	anomalia_dora	anomalia_emiliani1	anomalia_emiliani2	anomalia_garda	anomalia_lambro	anomalia_oglio
0	2000-02-20	2000	7	-0.040373	-0.048006	-0.017890	-0.032433	-0.076673	-0.068592	-0.052193
1	2000-02-27	2000	8	-0.033085	-0.050458	-0.008446	-0.015238	-0.068462	-0.063635	-0.037494
2	2000-03-05	2000	9	-0.037256	-0.066148	-0.015465	-0.013517	-0.074402	-0.069215	-0.028342
3	2000-03-12	2000	10	-0.040029	-0.056693	-0.016502	-0.010025	-0.062694	-0.063294	-0.023070
4	2000-03-19	2000	11	-0.044534	-0.057331	-0.014892	-0.010137	-0.054540	-0.063500	-0.025200

Prec4w_emiliani1	Prec4w_emiliani2	Prec4w_garda	Prec4w_lambro	Prec4w_oglio	Prec4w_piemonte nord	Prec4w_piemonte sud	Prec4w_ticino	Temp4w_adda
0.721117	0.197088	0.088006	0.094740	0.072905	0.341252	0.075852	1.582182	-0.363536
0.405419	0.095687	0.093497	0.063442	0.093190	0.604173	0.132934	1.945730	-0.414453
0.358700	0.089066	0.092217	0.063442	0.081667	0.505016	0.123336	1.469526	0.560761
0.643750	0.537006	0.319403	0.075714	0.235952	0.929535	0.330885	2.128494	0.331739
0.535984	0.537006	0.312260	0.075260	0.236381	0.781421	0.308807	1.919876	1.365427

Figure 4.12: Overview of the header of the first dataset. To preserve readability, only a part of the dataset is shown.

The results obtained by selecting input variables through the CMI score is shown in Figure 4.13 for different values of the user-specified threshold δ, in the case where each target is evaluated individually. This parameter is an indicator of how much information we are willing to lose to reduce the dataset dimensionality, since the smaller it is, the smaller would be the subset of inputs selected by the searching algorithm. Variable selection has been performed with several values of the parameter δ, although we found that the algorithm selects the same set of inputs for values below or above 0.05.

Figure 4.14 shows instead the results obtained when evaluating the vector of anomalies as the target for the feature selection process.

In the single-task case, the choice of candidate predictors is surprisingly unrelated to the geographical area used as a target.

When analyzing the results obtained from forward feature selection search, it is interesting to notice that the first selected variable is the one obtaining the highest MI score with the specific target. Therefore, it is worth trying to investigate the confidence intervals of MI estimates produced by each candidate feature, in order to verify if the difference among the mean MI scores is statistically significant. The results showed that the confidence intervals are all overlapped, which is predictable given the high correlation of the variables, as discusses in Section 4.6. For this reason, in a completely data driven approach like the one adopted, the feature selection process is carried out without any information about the fact that we are considering a distinct geographical region. This is the reason why often the chosen inputs correspond to different regions than the considered target.

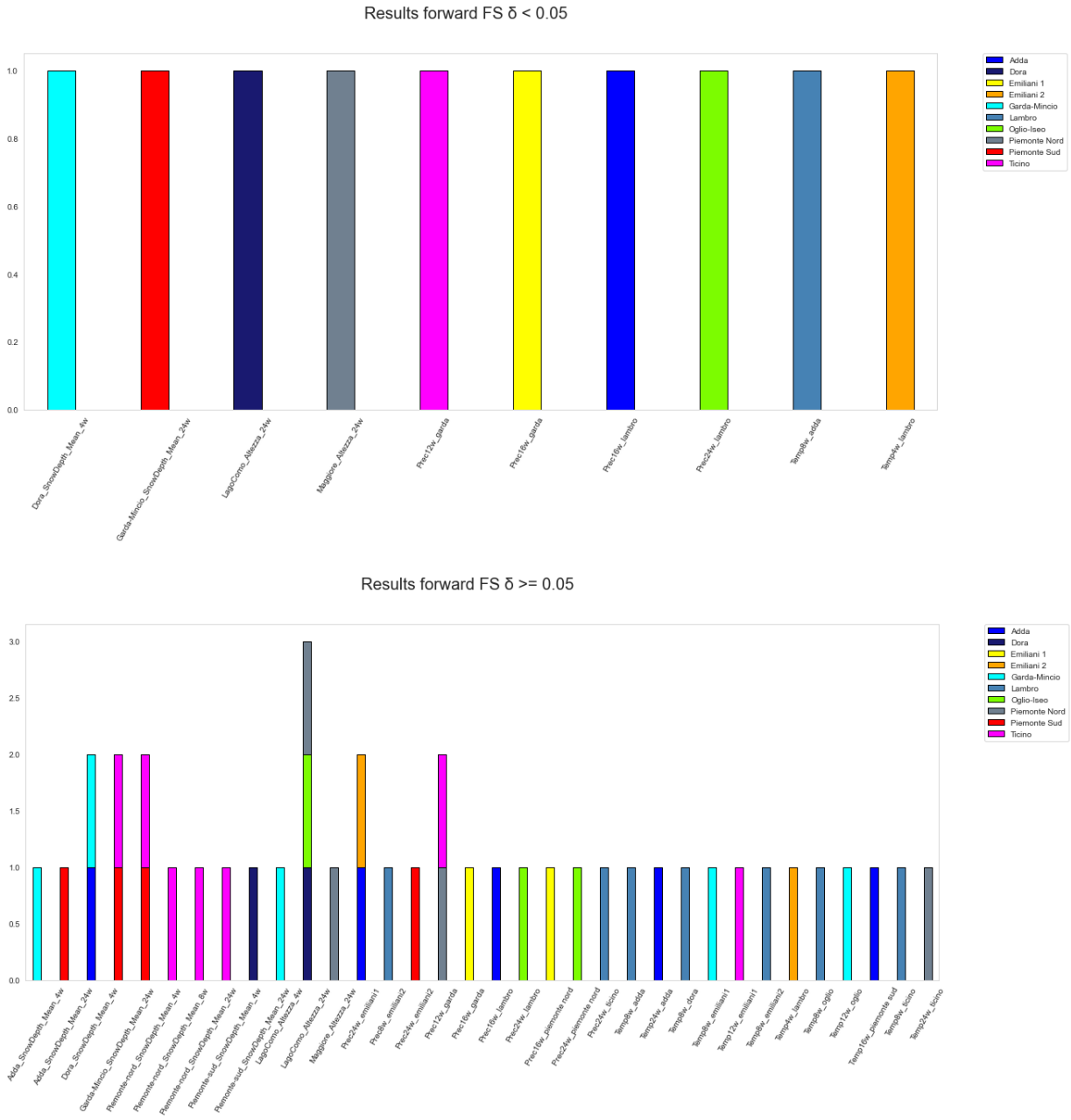


Figure 4.13: Forward feature selection results considering single-task problem. It can be noticed that the selected variables for each task are not always related to the target's geographical location.

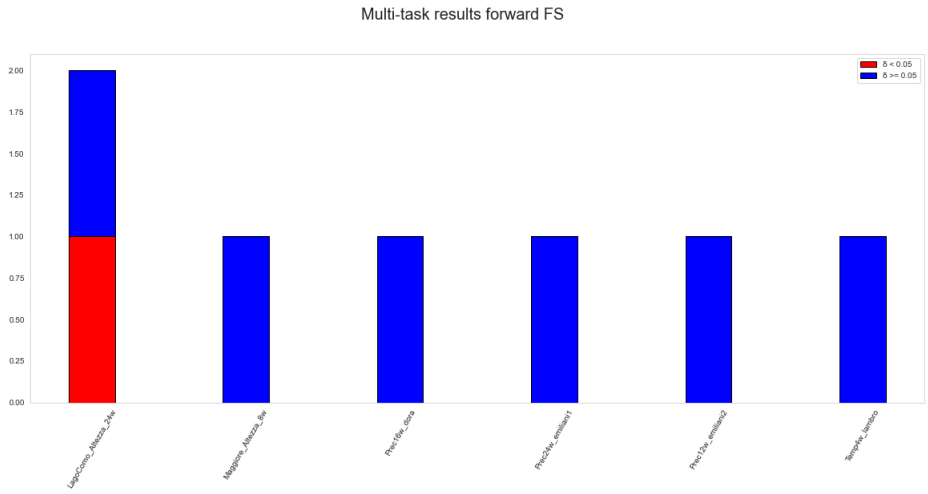


Figure 4.14: Forward selection results considering multi-task problem, for different values of the user-defined threshold parameter δ. It appears that the temperature values is the most informative one for describing the vector of anomalies.

4.5.7. Input anomalies

The previous work to model the NDVI index [12] used the original time series data as objective of a regression problem, that shows a sinusoidal trend. In this thesis, the anomaly of NDVI signal is the target to reproduce, that is why it is reasonable trying to consider the anomalies of the input variables as candidate predictors for our target. In fact, anomalous behaviors of target signals should correspond to an anomaly in the observed features. This intuition is also supported by [97], in which authors found correlation between temperature and precipitation anomalies with severe droughts occurred in California region during 2012-2015 period.

Variable name	**Description**
anomalia_PrecNw_AREA (1950-2021)	Anomaly of the average precipitation over AREA in the last N weeks.
anomalia_TempNw_AREA (1950-2021)	Anomaly of the average temperature over AREA in the last N weeks.
anomalia_AREA_SnowDepth_Mean_Nw (1981-2019)	Anomaly of the average snow cover depth over AREA in the last N weeks.
anomalia_LAKE_AltezzaNw (various-2019)	Anomaly of the average LAKE level in the last N weeks.

Table 4.2: Anomalies of input variables names with relative description. Weekly aggregation is the same as the previous dataset version $N \in [4, 8, 12, 16, 24]$ weeks.

	Date	Year	Week	anomalia_adda	anomalia_dora	anomalia_emiliani1	anomalia_emiliani2	anomalia_garda	anomalia_lambro	anomalia_oglio
0	2000-02-20	2000	7	-0.040373	-0.048006	-0.017890	-0.032433	-0.076673	-0.068592	-0.052193
1	2000-02-27	2000	8	-0.033085	-0.050458	-0.008446	-0.015238	-0.068462	-0.063635	-0.037494
2	2000-03-05	2000	9	-0.037256	-0.066148	-0.015465	-0.013517	-0.074402	-0.069215	-0.028342
3	2000-03-12	2000	10	-0.040029	-0.056693	-0.016502	-0.010025	-0.062694	-0.063294	-0.023070
4	2000-03-19	2000	11	-0.044534	-0.057331	-0.014892	-0.010137	-0.054540	-0.063500	-0.025200

anomalia_Adda_SnowDepth_Mean_4w	anomalia_Adda_SnowDepth_Mean_8w	anomalia_Adda_SnowDepth_Mean_12w	anomalia_Adda_SnowDepth_Mean_16w
-32.861493	-16.242731	-12.667229	-9.466587
-39.786733	-21.721446	-16.684856	-12.423759
-44.656067	-30.039898	-19.556670	-15.446634
-48.491140	-35.947579	-22.655174	-18.446586
-50.433368	-41.647430	-27.639610	-22.108764

Figure 4.15: Truncated header of the second dataset, which uses as inputs the anomalies of analyzed meteorological variables. For readability purposes, only the first 5 inputs are shown.

The anomalies of the inputs were calculated on the same period used for the previous analysis, with the same formula to obtain the NDVI anomaly presented in Subsection 4.5.2. Figure 4.15 shows the truncated header of the obtained dataset using input anomalies. Table 4.2 describes the variables of the resulting dataset, which again consists of 154 features.

Figure 4.16 shows the selected features via CMI ranking in both the single-task and multi-task analysis. Results for the multi-task approach are significantly different with respect to the previous dataset, as they show that, when considering input anomalies, different meteorological variables are more informative when evaluating the anomalies of all the study areas together.

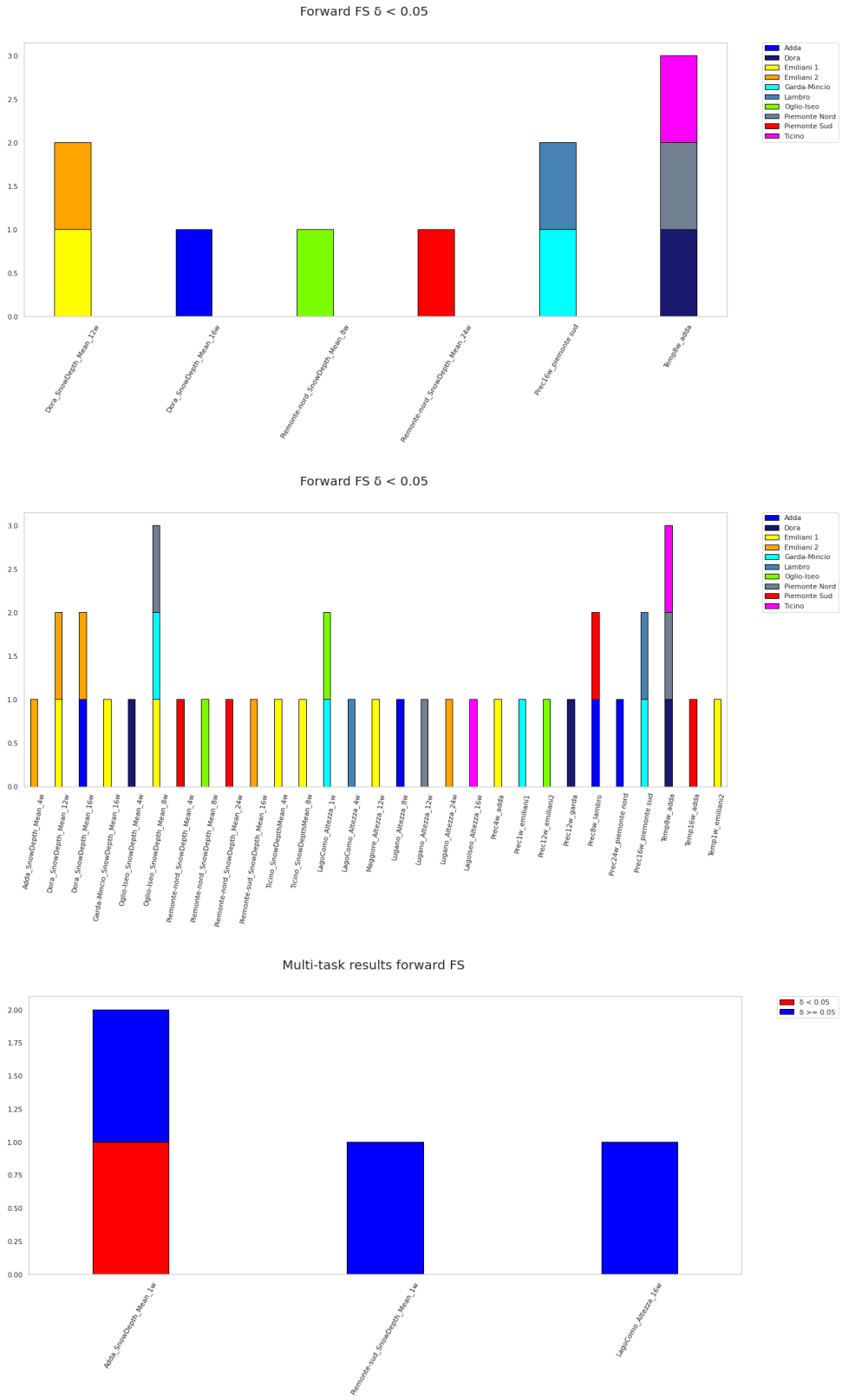


Figure 4.16: Forward feature selection results considering both single-task and multi-task settings on the dataset with anomalies both in the features and the target. It can be noticed a mismatch between selected input and target's geographical location for many basins.

4.6. Dimensionality reduction with PCA

The dataset using input anomalies has an overall of 154 candidate features for 10 target variables. The considered time interval covers a period of almost 20 year of weekly NDVI anomalies values, that result in a dataset of dimension 1038×164. A quick inspection of the cross-correlation matrix of the inputs shows that metereological data over different regions are extremely correlated. This not only suggests that it is possible to aggregate variables across multiple areas, but also explains why the feature selection algorithm selects inputs that are not geographically significant with respect to the target region.

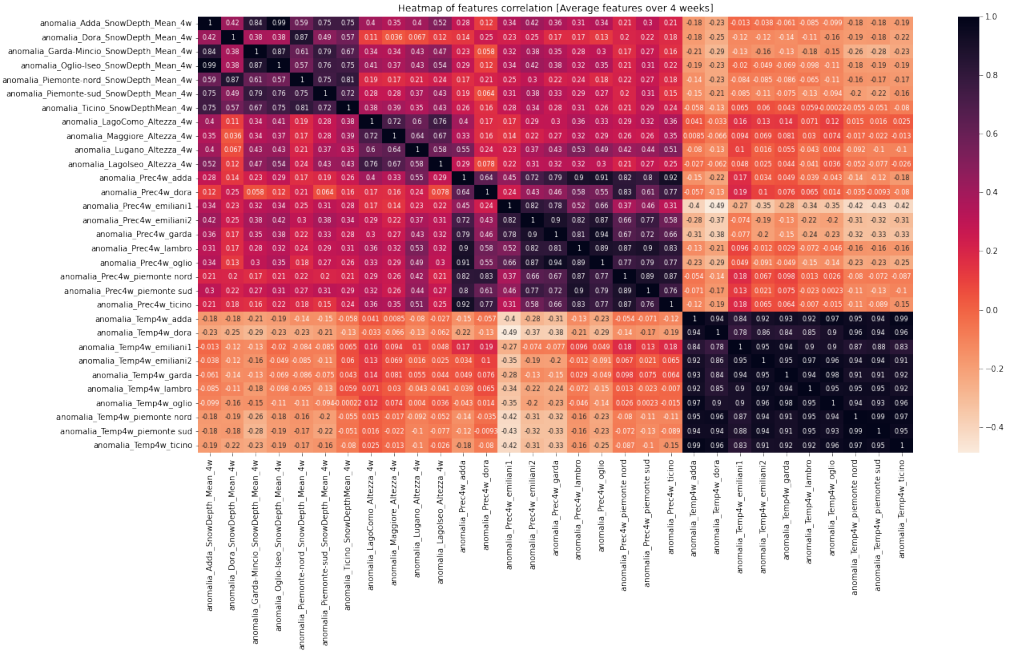


Figure 4.17: Cross correlation among candidate predictors. To ease readability of the graph, only the variables aggregated over 4 weeks are considered. The results over the other available aggregations show a similar trend.

The PCA algorithm [98] is used in order to reduce the dimensionality of the dataset, in order to obtain models able to generalize better both in the single-task and in the multi-task case.

Dimensionality reduction has been performed by employing a "local" version of the PCA algorithm, keeping the different types of variables distinct and applying the component analysis on each of the defined aggregations (i.e., for every variable aggregated on $4, 8, 12, 16, 24$ weeks). Dimensionality reduction has been performed by employing a "local" version of the PCA algorithm, keeping the different types of variables distinct and applying the component analysis on each of the defined aggregations (i.e., for every

variable aggregated on $4, 8, 12, 16, 24$ weeks).The PCA results show that the number of components is the same for all available aggregations, as illustrated in Figure 4.19. Figure 4.18 shows that the variables obtained with the PCA algorithm are much less correlated.

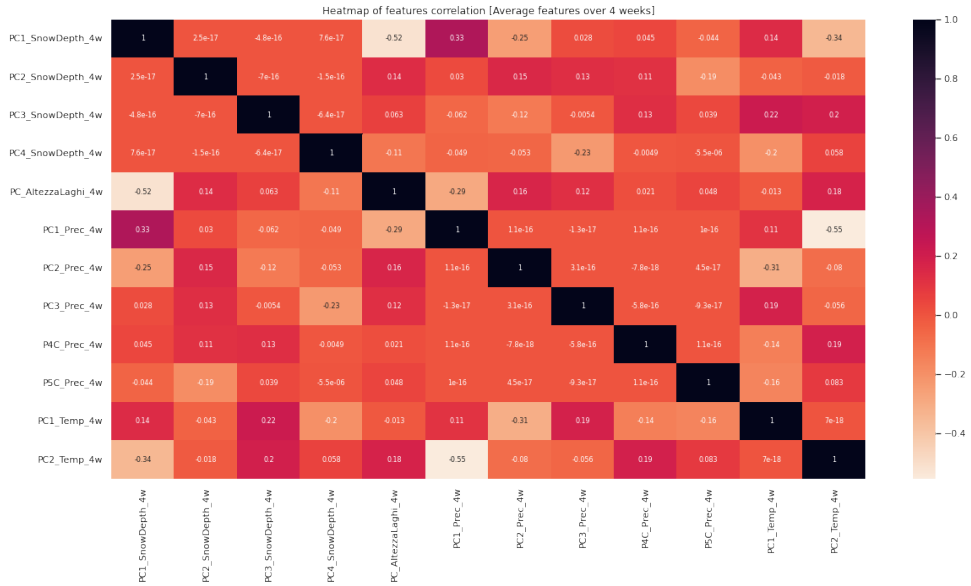


Figure 4.18: Cross correlation among the new variables introduced by PCA algorithm. To ease readability of the graph, only the components of the 4 weeks averages are considered.

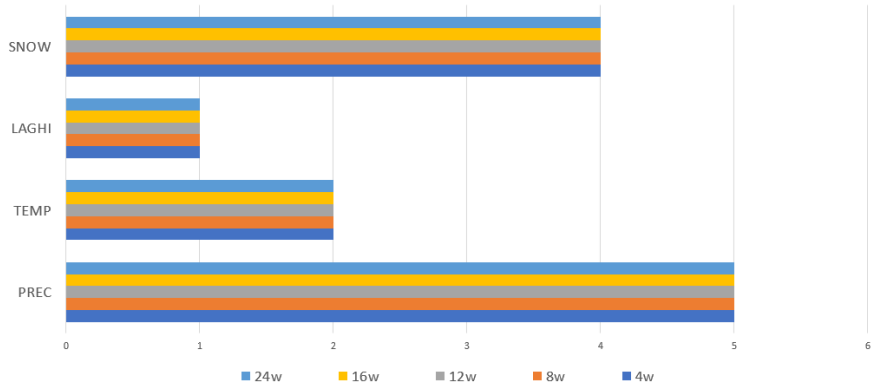


Figure 4.19: Selected number of PCA components for each meteorological variable needed to explain at least 95% of variance.

The results of the feature selection algorithm are show in Figure 4.21. Although we are sacrificing the geographical interpretability when looking at the selected features of each single basin, the overall interpretability is not much affected. Indeed, we have preserved the groups of variables (i.e., snow, lakes, precipitations and temperature) as well as the temporal aggregations.

	Date	Year	Week	anomalia_adda	anomalia_dora	anomalia_emiliani1	anomalia_emiliani2	anomalia_garda	anomalia_lambro	anomalia_oglio
0	2000-02-20	2000	7	-0.040373	-0.048006	-0.017890	-0.032433	-0.076673	-0.068592	-0.052193
1	2000-02-27	2000	8	-0.033085	-0.050458	-0.008446	-0.015238	-0.068462	-0.063635	-0.037494
2	2000-03-05	2000	9	-0.037256	-0.066148	-0.015465	-0.013517	-0.074402	-0.069215	-0.028342
3	2000-03-12	2000	10	-0.040029	-0.056693	-0.016502	-0.010025	-0.062694	-0.063294	-0.023070
4	2000-03-19	2000	11	-0.044534	-0.057331	-0.014892	-0.010137	-0.054540	-0.063500	-0.025200

PC1_SnowDepth_4w	PC2_SnowDepth_4w	PC3_SnowDepth_4w	PC4_SnowDepth_4w	PC1_SnowDepth_8w	PC2_SnowDepth_8w	PC3_SnowDepth_8w
-65.424254	-13.632711	3.784050	-8.230311	-37.034736	-12.309155	9.237081
-78.009271	-9.506963	-1.257922	-8.229785	-45.410477	-9.863076	6.408196
-88.948296	-6.739065	-6.611822	-10.602557	-60.688607	-9.040278	3.573041
-97.077745	-2.462017	-10.627163	-13.214360	-72.090111	-6.604201	-1.397034
-101.687791	0.692878	-12.431400	-16.248750	-83.255853	-5.090734	-3.379941

Figure 4.20: Truncated header of the last version of the employed dataset. The number of predictor candidates becomes one third of the original one, from 154 inputs to 60.

Figure 4.21 shows the results of the application of FS based on CMI for the dataset obtained with the PCA algorithm. Single-tasks results shows that the first component of temperature and snow depth are selected as most informative variables for the majority of basins, which is in line with what was obtained in the second version of the dataset in Figure 4.16. The results of the multi-task setting, on the other hand, demonstrate a more varied selection of variables compared to the previous cases.

This dataset version has been used to evaluate the learning models described in Chapter 5. Specifically, the chosen features are used in two different settings: a Single-Task approach, in which only the features of the analyzed task are evaluated; the Multi-Task approach, in which the variables in the last plot of Figure 4.21 are subjected to a further supervised selection as better explained in 5.2.

In conclusion, in this Chapter it has been introduced how candidate predictors and the target variables have been extracted for the analyzed area. Data were aggregated over different weekly intervals, to model the long-term effects of meteorological variables chosen to model drought index anomalies. The results obtained in this first case highlighted an unusual situation: selected features were in the most cases not belonging to the target area. This suggested considering the anomalies of the inputs. An inspection on the input variables showed high cross-correlation among them, which partially justified the choices of the feature selection algorithm. For this reason, (local) PCA has been employed to reduce the dimensionality of the dataset and to obtain the number of components that could explain at least 95% of the variance. In this way, candidate predictors have been further aggregated to improve generalization of the selected models, even if this comes at the cost of reducing the interpretability of the input selection process. Finally, the results of FS reported in Figure 4.21 identify the relevant and non-redundant principal

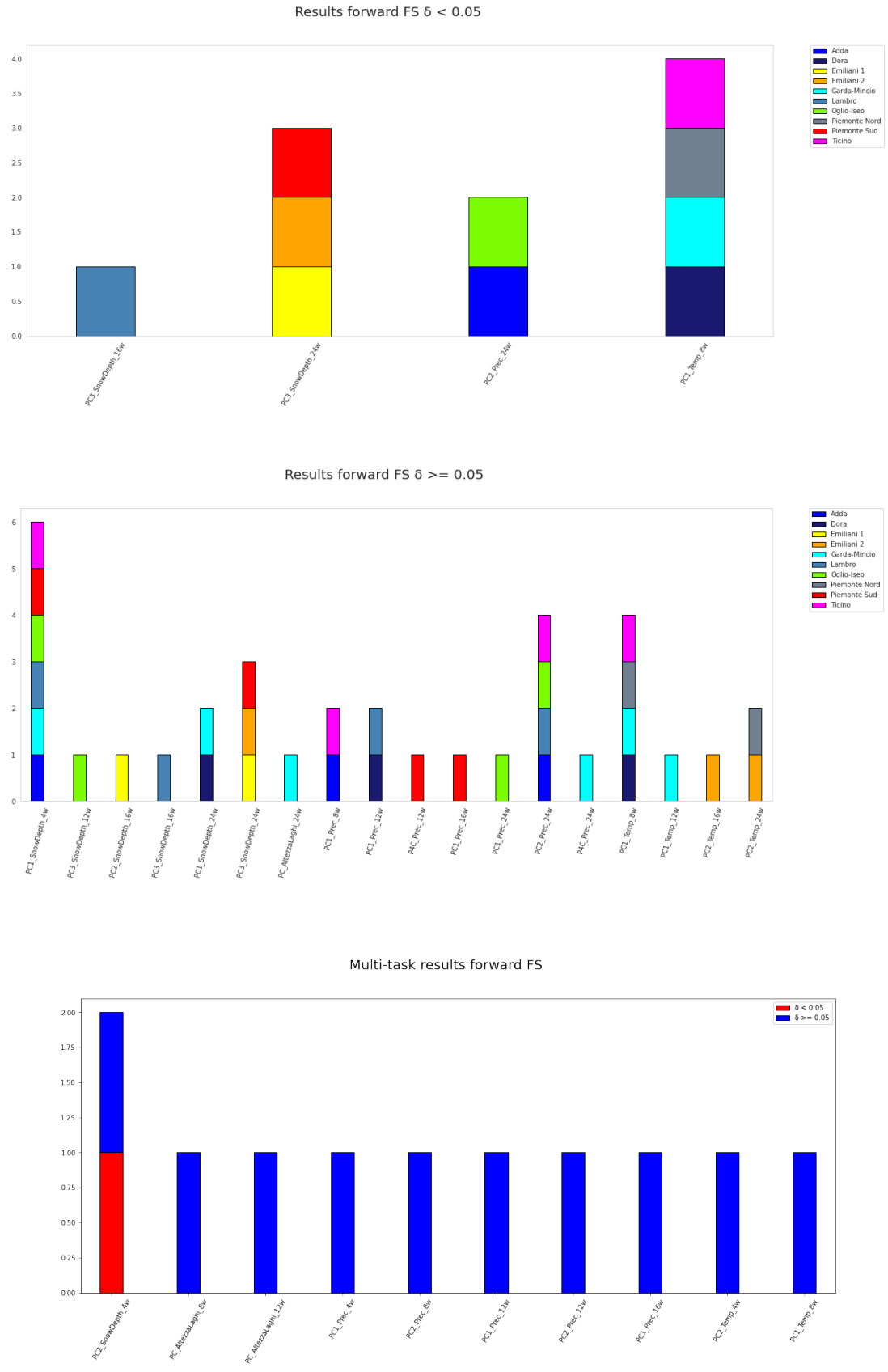


Figure 4.21: Forward selection results applied on the dataset resulting from PCA results. The selected inputs are less interpretable than the previous cases in the single-task settings.

components for each basin and for the vector of anomalies and they will be the starting point for the application of the methods of next Section.

5 | Results

This chapter shows the main results of the thesis, which rely on the application of the models discussed in Chapter 3 on the data introduced in Chapter 4. In particular, we consider the dataset shown in Figure 4.20, which has been obtained from the raw data following the methodology described in Section 4.6. The first subsection discusses the results obtained in the single-task setting, where an autoregressive approach is proposed to analyze if the selected models could benefit from evaluating the historical time series of the candidate predictors obtained by the feature selection algorithm. This investigation shows that the autoregression is sufficient to explain the phenomenon, and the residual analysis does not lead to anything significant. Still, we are interested in obtaining a good approximation of the target signal using only the selected features. However, we could not achieve promising results due to the noise in features and target measurements and the small number of samples. Nevertheless, the Recurrent network's prediction trend suggests that this model class has quite accurately captured the target signal's correct sign; therefore, this intuition has guided us to consider a classification problem. This analysis shows that the variables help explain the sign of the NDVI anomaly rather than its exact value, which is still a very relevant result, since it let us capture the status of the basins and to identify drought conditions. Finally, the results obtained in the multi-task setting are presented. Specifically, two of the approaches described in Section 3.4.1 are used to produce predictions in a classification approach for all the sub-basins at the same time.

All the experimental work has been carried out using *Python 3.7.13* as programming language, *Tensorflow 2.8.2* and *PyTorch 1.11.0+cu113* as libraries for deep learning models.

5.1. Single-Task

The discussion of the results obtained by evaluating a single area as a target is divided into two parts. The first analysis is focused on training the models in a regression setting, whose objective is to reconstruct the anomaly of the NDVI signal with the variables selected using the CMI feature selection approach, as explained in Chapter 2. The second

part illustrates the results obtained on a Classification problem, where three classes are identified, corresponding to as many crop conditions.

As an example, the results are related to the area in the dataset called *Emiliani_2*, which is shown in Figure 5.1. This choice has been made because, by analyzing the target signals of the NDVI anomalies, it appears to be the one containing the most significant percentage of cultivable areas, meaning that the resulting NDVI signal results from the average of a more extensive set of points.

Figure 5.1: Target area for evaluating the results in the Single-Task setting, highlighted in yellow.

5.1.1. Input Data

Since all results in this subsection are produced on the single area *Emiliani 2*, Table 5.1 shows the chosen variables using the CMI feature selection approach, for the considered target.

Selected Variable	Description
PC3_SnowDepth_24w	Third component selected by the PCA algorithm of the average snow cover depth in 24 weeks.
PC2_Temp_24w	Second component selected by the PCA algorithm of the average temperature in 24 weeks.
PC2_Temp_16w	Second component selected by the PCA algorithm of the average temperature in 16 weeks.

Table 5.1: Feature selection results for the analyzed area.

During the discussion of the results, we explicitly mention if the model are using as input the features at the current time-step t, or their historical time-series. In fact, to tackle

the problem of noisy input measurements, we try to augment the chosen variables by investigating the behavior of the selected models when they receive past input values for up to 6 months. In this way, we attempt to make the models find a relationship between input and the NDVI anomaly so that we are able to reproduce the drought index using only our extracted data.

5.1.2. Analysis of Prediction Residual

This analysis is configured to verify if the prediction process actually benefits from the selected features or if all the information can be obtained only by looking at the historical data of the target itself.

An autoregressive (AR) model represents a random process in which the output value has a linear dependency only on its previous values and an unpredictable stochastic term. The following equation gives an example of an autoregressive model of order k [99]:

$$y(t) = \beta_0 + \sum_{i=1}^{k} \beta_i y(t-i) + \epsilon(t). \tag{5.1}$$

The residuals [100] of a model's predictions are the differences between the true output values y and the estimates $\hat{y}$:

$$r(t) = y(t) - \hat{y}(t). \tag{5.2}$$

Residuals can be considered as the error that is not explained by the fitted line, and they carry helpful information to understand if the model was able to capture the pattern of the data:

- if the residuals are correlated, there is still room for improving the model since it does not fully explain the relationship between data;
- if the residuals have a non-zero mean, the predictions are biased.

These two conditions identify if the forecasting model can be improved to obtain better predictive performance. Therefore, this analysis aims to verify if the residuals of AR models of the NDVI anomaly are indeed uncorrelated and have zero mean, that is if the residuals are *white noise*. This verification is carried out using **Ljung-Box test** [101].

Definition 5.1.1 (Ljung-Box test)**.** The test is defined to verify the following two hypotheses:

- $\boldsymbol{H_0}$: residuals are independently distributed.
- $\boldsymbol{H_1}$: there is serial correlation between residuals, therefore the data is not i.i.d.

Let r_i be the estimated autocorrelation between forecasts separated by i periods. The test involves computing the Q statistic as critical value, defined as:

$$Q = n(n+2)\sum_{i=1}^{k}\frac{r_i^2}{n-i}, \tag{5.3}$$

where n is the number of samples considered and k the number of tested lags. The obtained critical value is compared against a chi-squared distribution [102]:

$$Q > \chi^2_{1-\alpha,k} \tag{5.4}$$

where $\chi^2_{1-\alpha,k}$ is the $(1-\alpha)$-quantile of the chi-squared distribution with k degrees of freedom. If the $p-$values based on the Q statistics are small than 0.05, the null hypothesis H_0 can be rejected with a statistical confidence of 95%; thus, the series is not white noise. Otherwise, $p-$values larger than 0.05, mean that we cannot reject the null hypothesis; therefore, the series is a white noise.

The dataset is divided into the following splits: 60% for training samples, and the remaining 20% equally divided between validation and test. Cross-validation for time-series is used to select the best model's hyperparameters. The hyperparameters used to produce the presented results are reported in each subsection for each model.

Autoregressive model of first order

The first step is to employ an AR model of the first order, whose output, according to Equation 5.1, is:

$$y(t) = \beta_0 + \beta_1 y(t-1) + \epsilon(t). \tag{5.5}$$

In this way, the forecasts of NDVI anomalies at time t are obtained by evaluating only the NDVI anomalies at time $t-1$. Then, once the model is fitted to training data, the residuals are computed as shown in Equation 5.2. Figure 5.2 shows the autocorrelation plot for the $k = 50$ lags tested to evaluate whether the residuals are actually white noise. It can be seen that there are few lag values for which the autocorrelation value is statistically significant, evaluated in a confidence interval of 95%.

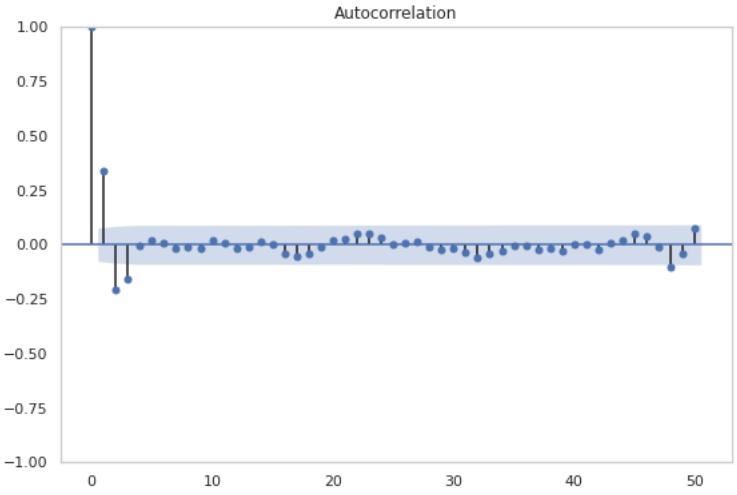


Figure 5.2: Autocorrelation plot of the residuals of AR(1) model's forecasts. There are several spikes that lie outside the 95% confidence interval.

The results of the cross-validation procedure are reported in Table 5.2, while Table 5.3 presents the results of the Ljung-Box test, as well as the statics obtained on the residuals of the predictions in the test split.

Model	**Tuned hyperparamters**
Extremely Randomized Trees	• *bootstrap=True* • *max_depth=10* • *max_features=sqrt* • *min_samples_leaf=2* • *n_estimators=400* • *random_state=0*
Feed-Forward Neural Network	• *epochs=100* • *EarlyStopping val_loss = 5* • *shuffle=False* • *batch_size=32*

Table 5.2: Tuned hyperparameters reproduce the residual of the AR(1) model's prediction. Linear Regression model does not have hyperparamters to be tuned, therefore it is omitted in this table.

Since the results show that the series of residuals is not white noise, it is reasonable to try if the selected models can improve the performances of the regression by learning the residuals of the prediction, using the variables from the feature selection process.

Q statistic	$p-$value	Is white noise
152.3638	$2.79e^{-12}$	False

Table 5.3: Ljung-Box test results on residuals of AR(1) model forecasts on the test split.

Let $\hat{y}$ the prediction of the AR(1) model, r the residuals of the predictions. We try to use the previous model classes to learn $\hat{r}$, so $\hat{y} + \hat{r}$ is a better prediction of the true output y, compared to $\hat{y}$.

The models employed for this analysis use as inputs the historical time-series $[t-24, ..., t]$ of chosen variables, presented in Table 5.1, to predict the residual $\hat{r}$ to add to the autoregressive model's prediction.

Figure 5.3 shows the results on the test set, zoomed on the first 50 samples. The forecast signal is lagged with respect to the observed data, suggesting that the anomaly of the NDVI is an autoregressive signal, and that is why the features are not useful for adding informativity on its prediction.

Table 5.4 presents numeric measurements that help in assessing if the quality of the autoregressive model predictions has been effectively improved. These statistics highlight that despite the series of the residuals of the forecasts is not white noise, the trained models using only the chosen input features do not help obtain better predictive performances.

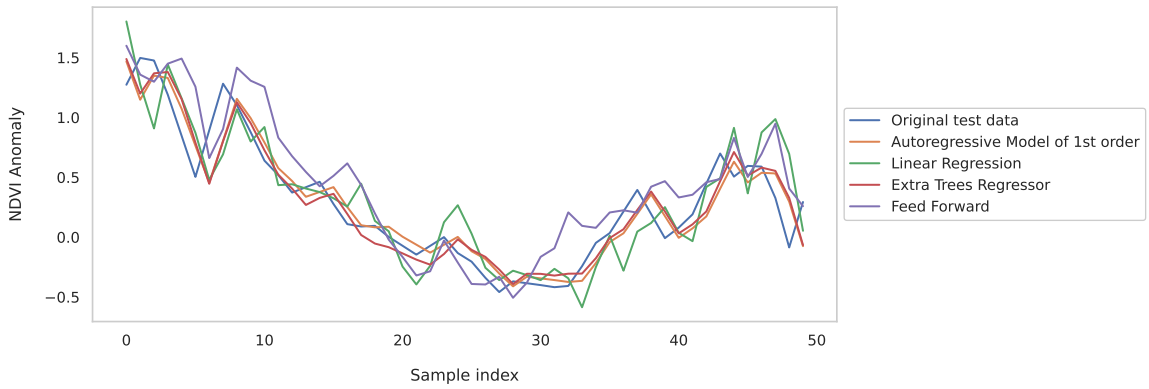


Figure 5.3: Comparison of the predictions obtained by the AR(1) models and those obtained by adding to it the residual predicted by the other model classes. The output is truncated to the first 50 samples of the test data.

It can be noticed that the Recurrent network has been excluded from this analysis. In fact, given the implemented architecture described in Section 3.3.2, this kind of network should

Model	MAE	MSE	RMSE
Autoregressive first order	0.16	0.040	0.20
Linear Regression	0.20	0.07	0.26
Extra Trees Regressor	0.172	0.047	0.21
Feed-Forward	0.26	0.11	0.33

Table 5.4: Models performances in terms of Mean Absolute Error, Mean Squared Error, Root Mean Squared Error when comparing the true output value y with the prediction of AR(1) model $\hat{y}$ and the predictions obtained by adding the predicted residual $\hat{r}$.

already be able to detect autoregressive patterns, since it receives in input a sequence of i time steps to predict the $i+1-$th step.

Autoregressive model of third order

The results obtained using an autoregressive model of the first order suggest that the features do not help to reconstruct the NDVI anomaly signal. Moreover, the forecasts appear to be lagged with respect to the true output. This behavior does not seem to be mitigated by the models that use only the chosen variables to predict the residual of the prediction. Therefore, it is reasonable to look ad autoregressive models of order greater than 1. The analysis is repeated by employing an AR(3) model, whose output is given by setting $k = 3$ in Equation 5.1:

$$y(t) = \beta_0 + \beta_1 y(t-1) + \beta_2 y(t-2) + \beta_3 y(t-3) + \epsilon(t). \tag{5.6}$$

Once the model has been fitted to the training samples, the residuals are computed for the forecasts of the test set, and the autocorrelation plot is shown in Figure 5.4. It is evident that the residual series does not have a correlation with its delays.

Then, using again the historical time-series of data in Table 5.1, the hyperparameters' values tuned with cross-validation are the same of those in Table 5.2, while Table 5.6 presents the Ljung-Box test outcome, which confirmes that the residuals are white noise. This implies that the AR(3) model cannot be further improved.

This is also confirmed by running the same learners again to infer the residuals of the predictions of the AR(3) model. As shown in Table 5.6, the metrics confirm that models trained only on the variables cannot learn anything more than what can be done with the autoregression. For completeness, Figure 5.5 reports the predictions compared to the true values of the NDVI anomaly, zoomed in on the first 50 samples of the test set. As

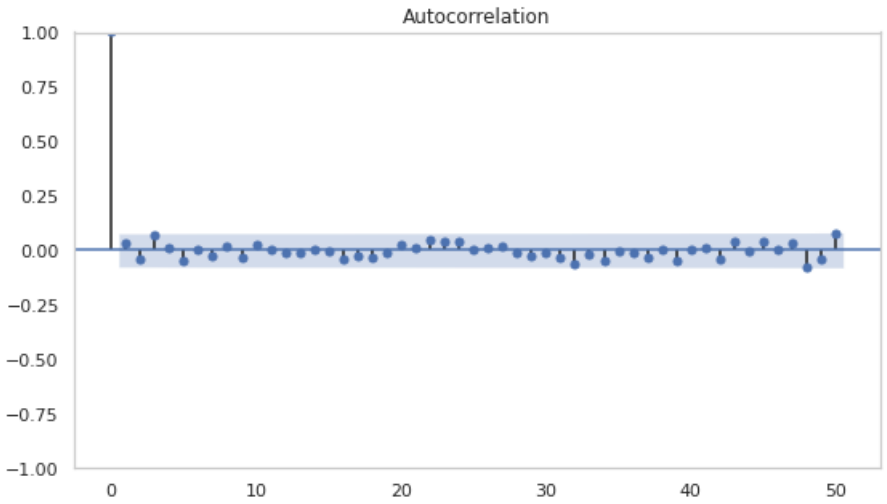


Figure 5.4: Autocorrelation plot for the residuals of an AR(3) model predictions, for $k = 50$ lags. All the spikes lie in the 95% confidence interval, except the sample autocorrelation at lag 0. This indicates that residuals are uncorrelated.

Model	**Tuned hyperparamters**
Extremely Randomized Trees	• *bootstrap=True* • *max_depth=10* • *max_features=sqrt* • *min_samples_leaf=2* • *n_estimators=400* • *random_state=0*
Feed-Forward Neural Network	• *epochs=100* • *EarlyStopping val_loss = 5* • *shuffle=False* • *batch_size=32*

Table 5.5: Tuned hyperparameters for reproducing the residual of AR(3) model's prediction. It can be noticed that the results of the cross-validation are the same of the previous case.

Q statistic	***p*−value**	**Is white noise**
36.84	0.917	True

Table 5.6: Ljung-Box test results on residuals of AR(3) model forecasts on the test split.

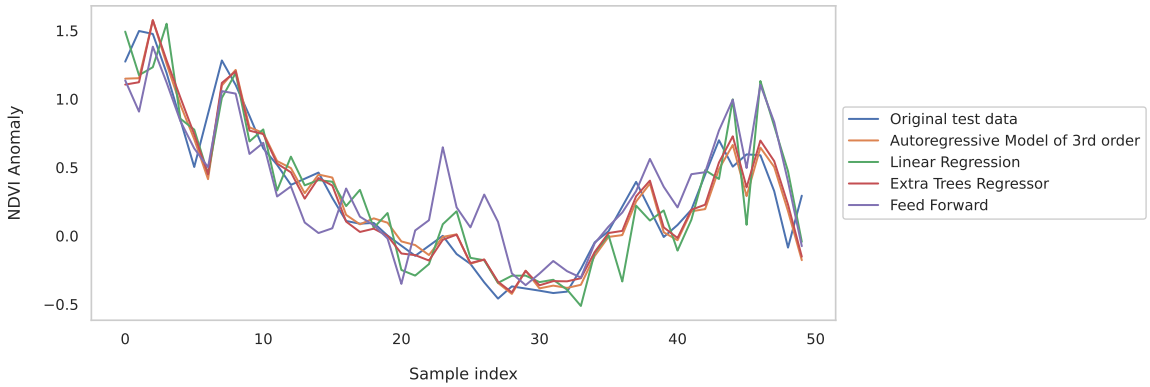


Figure 5.5: Comparison of the predictions obtained with the AR(3) models and the ones obtained by adding to it the residuals predicted by other model classes. The output is truncated to the first 50 samples of the test data.

in the previous case, the predictions of every employed model appear to be lagged with respect to the target, due to the noise that affects the data at our disposal. Therefore the final result of this analysis is that the best prediction of the anomaly at time t is given by its past values from $t-1$ to $t-3$.

Model	MAE	MSE	RMSE
Autoregressive first order	0.13	0.029	0.17
Linear Regression	0.17	0.052	0.228
Extra Trees Regressor	0.14	0.033	0.18
Feed-Forward	0.19	0.06	0.33

Table 5.7: Models performances in terms of Mean Absolute Error, Mean Squared Error, Root Mean Squared Error when comparing the true output value y with the prediction of AR(3) model $\hat{y}$ and the predictions obtained by adding the predicted residual $\hat{r}$.

5.1.3. Regression approach: identification of the NDVI anomaly from features

The previous analysis concluded that the NDVI anomaly signal is autoregressive since the residual of the prediction of an AR(3) model results in white noise. This section investigates if it is possible to obtain a good approximation of the target by only using the variables selected with the CMI feature selection approach.

This discussion will be divided into two Subsections, which distinguish the two cases in which the model's input data are features at the current time step t or if the output is obtained by evaluating the historical series up to time-step $t-24$. In both cases, the

dataset split is kept coherent with the previous analysis, meaning that $60\% - 20\% - 20\%$ are the percentage reserved for training, validation, and test, respectively. Moreover, since we are treating time-series data, it has been ensured that shuffling samples is not allowed so that models correctly see data in the temporal order. This approach tries to eliminate the possibility that a model can learn the past by looking at future signal values: the training set includes samples from February 2000 to October 2011; the validation set spans from October 2011 to August 2015; finally, the test set goes from August 2015 to July 2019.

The validation set is used to verify which model's hyperparameter obtains the best performances according to the selected loss function, that is, the *Mean Squared Error*. Once the hyperparameters are found, the model instance is re-trained on a new split, including samples starting from the first 20% of the train data and going until the validation split end. This allows a new model class instance, initialized with the found hyperparameters, to be trained on 60% of the available samples and tested on the remaining 20%. Therefore, the training phase of the final models is also consistent with the data size employed for cross-validation.

Variables at time-step t

The first approach to reconstructing the NDVI anomaly, using only the variables selected by the CMI feature selection approach, is to consider the inputs at time-step t for predicting the target at t. Table 5.1 shows the results of the feature selection algorithm, for the basin *Emiliani_2*.

Cross-validation results are reported in Table 5.8, except for the Linear Regression model since it has no hyperparameter to be tuned.

An important consideration is done when employing a Recurrent Network, using the described architecture in Section 3.3.2. In this way, a signal from previous inputs and prediction of the time-series can be exploited for subsequent ones.

The results of the predictions obtained on the test set are shown in Figure 5.6. To better understand the performance of the models, Table 5.9 reports some metrics to evaluate the predictions with respect to the actual target value.

From the numerical results, it is possible to conclude that the first three models performed better than the Recurrent Network. However, it must be remembered that this last type of model suffers from a lack of data. Compared with the others, the recurrent network must learn a more complex sequential model that needs more data to learn meaningful

Model	Tuned hyperparamters
Extremely Randomized Trees	• *bootstrap=True* • *max_depth=10* • *max_features=sqrt* • *min_samples_leaf=2* • *n_estimators=200* • *random_state=0*
Feed-Forward Neural Network	• *epochs=50* • *EarlyStopping val_loss = 5* • *shuffle=False* • *batch_size=32*
Recurrent Neural Network	• *epochs=50* • *Input sequences time-steps = 5* • *Number of stacked RNN layers=1* • *Output step = 1*

Table 5.8: Tuned hyperparameters for the applicable models receiving in input the features at current time-step. Recurrent Neural Network has been implemented by scratch using *PyTorch* library, therefore hyperparameters names are user-defined and may not reflect the nomenclature of the network provided by default by deep-learning libraries.

Model	MAE	MSE	RMSE
Linear Regression	0.467	0.345	0.587
Extra Trees Regressor	0.44	0.293	0.54
Feed-Forward	0.423	0.3	0.55
Recurrent Network	0.5	0.416	0.645

Table 5.9: Models performances in terms of Mean Absolute Error, Mean Squared Error, Root Mean Squared Error.

relationships. Furthermore, the metrics suggest that there is no actual benefit in adopting models more complex than the Linear Regression.

Figure 5.6: Test results of the selected models. Each model receives as input the variables selected via CMI FS and the goal is to reproduce the target NDVI anomaly.

Variables at time-steps $[t-24, ..., t]$

Learners exhibit unpromising results in the previous subsection; the following approach tries to tackle this problem by considering a historical series of the selected variables. Specifically, the setting is:

- the target is to predict the NDVI anomaly at time step t;
- for each input feature, the historical data series from t to $t-24$ are considered.

This analysis aims to understand whether a model can obtain better predictive performances by providing a large number of past values from the most informative variables.

The results of cross-validation process on the chosen model classes can be reviewed in Table 5.10.

Model	Tuned hyperparamters
Extremely Randomized Trees	• *bootstrap=True* • *max_depth=35* • *max_features=sqrt* • *min_samples_leaf=2* • *n_estimators=200* • *min_samples_split=2* • *n_estimators=200* • *random_state=0*
Feed-Forward Neural Network	• *epochs=50* • *EarlyStopping val_loss = 5* • *shuffle=False* • *batch_size=28*
Recurrent Neural Network	• *epochs=50* • *Input sequences time-steps = 5* • *Number of stacked RNN layers=1* • *Output step = 1*

Table 5.10: Tuned hyperparameters for the applicable models when using as input the historical series of input features. Recurrent Neural Network has been implemented by scratch using *PyTorch* library, therefore hyperparameters names are user-defined and may not reflect the nomenclature of the network provided by default by deep-learning libraries.

Figure 5.7 shows the results of the same models in this setting. It is evident the difference in the Recurrent Network prediction with respect to the previous case. The signal shape recalls those of a classification problem. Indeed, when evaluating the accuracy between the sign of the original signal and that of the prediction, the model exhibits an accuracy of approximately 70%. This seems to suggest that the network has learned the sign of the anomaly. The metrics to evaluate the predictions are reported in Table 5.11.

Although the numbers do not seem to indicate an improvement compared to the previous situation, what can be seen from the prediction graphs is the increased variability in the output signal. In fact, this result allows us to say that the model seems to better reproduce the target particularly, at least for the first half of the test set.

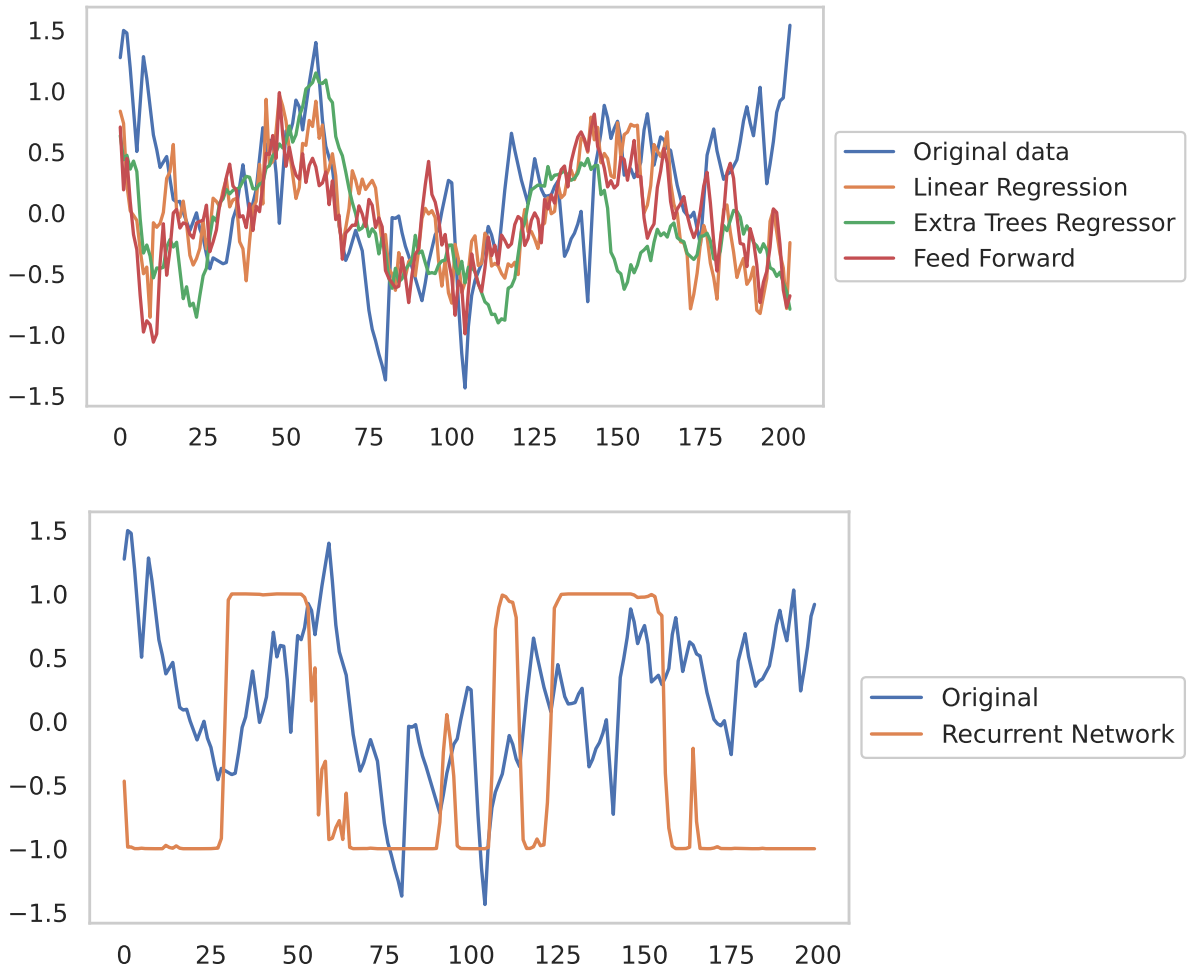


Figure 5.7: Results of the selected models when providing as input the selected features from t up to $t-24$. The goal is to predict the NDVI anomaly at time t. Note that, in this context, a single time step represents a week.

Model	MAE	MSE	RMSE
Linear Regression	0.51	0.443	0.66
Extra Trees Regressor	0.55	0.453	0.67
Feed-Forward	0.597	0.567	0.753
Recurrent Network	0.98	1.118	1.058

Table 5.11: Models performances in terms of Mean Absolute Error, Mean Squared Error, Root Mean Squared Error when predicting from the historical series of input variables.

5.1.4. Classification of NDVI signal

The previous approaches show that the NDVI anomaly is autoregressive, and features have difficulties in reconstruct this very non-linear signal with a small amount of data.

Section 5.1.3 illustrates that the models cannot obtain a good approximation of the target using only the input variables. However, the Recurrent Neural Network in the setting of Subsection 5.1.3 shows that the prediction shape can somehow distinguish the sign of target values. Since the sign would be able to give us insights on a drought situation, this section shows the models performances in evaluating a classification problem.

In this setting, we identify three conditions in which the anomaly of the NDVI can be found, which describe as many drought conditions:

- *Normal* indicates that the anomaly value is average; therefore, no drought conditions are detected.
- *Good* is assigned to anomaly values above average, which indicates and is an indicator of healthy vegetation.
- *Bad* identifies the situation in which anomaly values are below the average for a given week, meaning that there is a water shortage and, consequently, a drought condition.

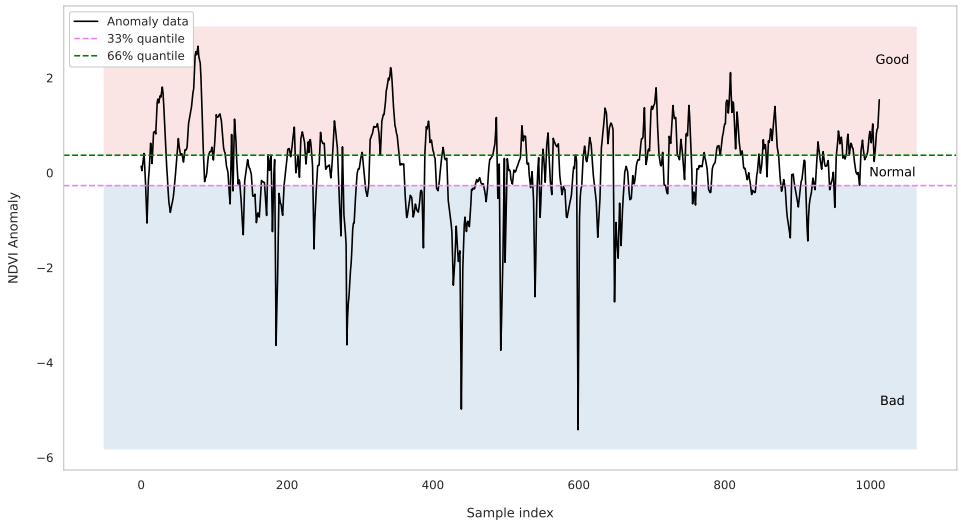


Figure 5.8: Identification of the thresholds to label NDVI anomaly values. The anomaly signal is the one of *Emiliani_ 2* area, but quantiles are computed only on training samples.

We computed the tertiles using the defined training split to assign a label to each sample of the NDVI anomaly. Values above the 66% quantile are classified as *Good*, below the 33% quantile as *Bad*, while the middle region identifies the *Normal* values. Figure 5.8 shows a graphical representation of the three identified regions.

At this point, except for the Linear Regression, the other three classes of models are trained on the selected feature history, as shown in the last step of Subsection 5.1.3, using again the variables in 5.1. Figure 5.9 shows that the labels are not uniformly distributed across the three defined splits of the dataset; therefore, we prefer to employ more complex models to discover the non-linear dependencies between data.

It is worth mentioning that labels are balanced by construction in the training set. Therefore it is not necessary to employ imbalanced classification approaches.

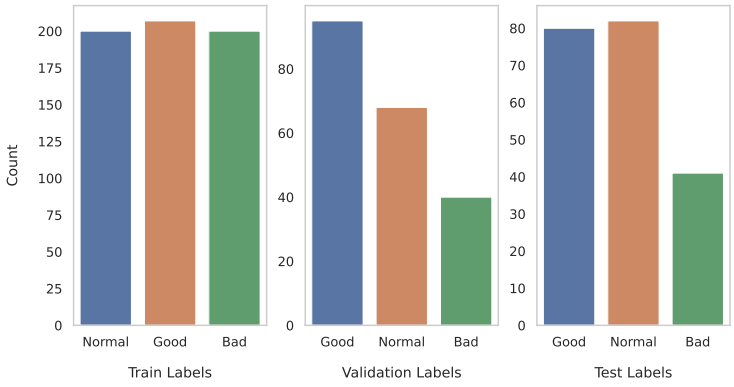


Figure 5.9: Histogram showing the classes distribution for each split.

As in the previous approaches, the dataset on which the results are obtained uses the same $60-20-20$ split for training, validation and testing. The models' hyperparameters are firstly tuned using the validation set, then once we obtain their optimal value new instances of the same model class are trained using samples from the 20% of the training subset up to the final part of the validation set. In this way, the new models are trained on the same percentage of data. The tuned hyperparameters are presented in Table 5.12.

To better understand what a model has learned, Figure 5.10 shows each model's confusion matrix, that is placed side by side with a graph showing the prediction shape compared to the actual output trend.

Firstly, Figure 5.10a shows the ExtraTrees performances on the test split. It can be seen that the overall accuracy is not particularly good, since it obtains a good prediction only in approximately 38% of the cases. It seems that this classifier is struggling to distinguish "Good" samples from "Normal" or "Bad". By looking at the prediction trend plot, the ExtraTrees model seems to optimistically assign the "average label" to samples of weeks that oscillate between a normal condition to an extreme one. However, this behavior seems to drastically change in the last part of the test set, in which the classifier erroneously assigns "Good" to weeks that are, in reality, between average and bad

Model	Tuned hyperparamters
Extremely Randomized Trees	• *bootstrap=True* • *max_depth=10* • *max_features=sqrt* • *min_samples_leaf=2* • *n_estimators=140* • *random_state=0*
Feed-Forward Neural Network	• *epochs=35* • *shuffle=False* • *batch_size=32*
Recurrent Neural Network	• *epochs=50* • *Input sequences time-steps = 5* • *Number of stacked RNN layers=1* • *Output step = 1*

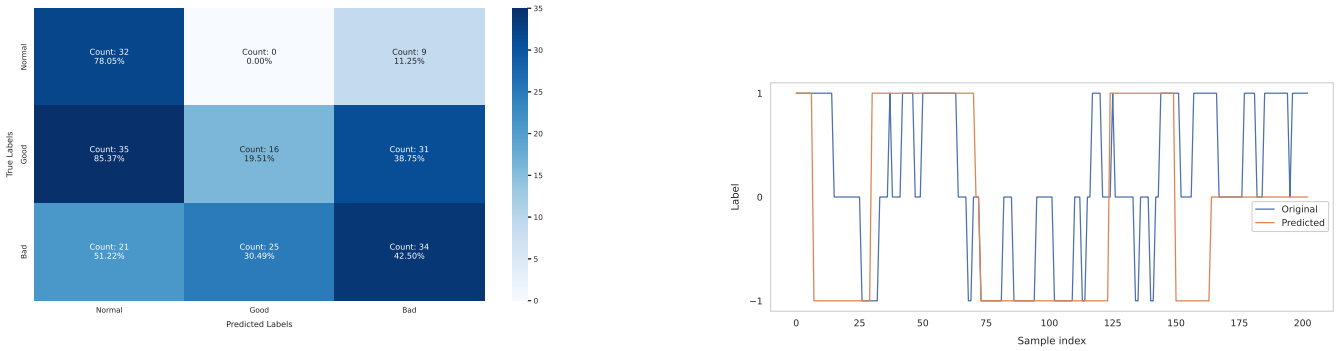
Table 5.12: Hyperparameters tuned in the Single-Task Classification setting. Recurrent Neural Network has been implemented by scratch using *PyTorch* library, therefore hyperparameters names are user-defined and may not reflect the nomenclature of the network provided by default by deep-learning libraries.

conditions. Furthermore, in the final segment, the average is pessimistic, as it identifies the samples as "Normal" even if the signal oscillates more towards "Good". In most cases, the model persistently predicts the average label for consecutive weeks even though it has no indication of what is the current week, which is a satisfactory robustness in the result.

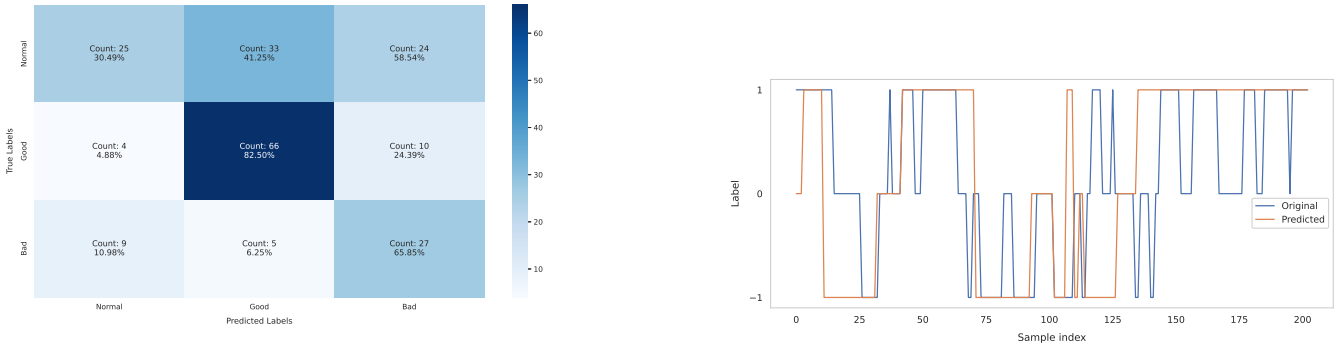
The Feed-Forward performances are shown in Figure 5.10b. The model achieves an overall accuracy of 59%, and it seems to perform better than ExtraTrees when identifying "Good" conditions. The prediction trend's graph suggests that the Feed-Forward network also predicts an average condition over several weeks, especially evident in the last segment of the test split. However, it differs from the previous case as it tends to be more optimistic, always assigning "Good" even if the signal oscillates more towards the "Normal" label. By looking at the model's accuracy on the "Good" label, it can be highlighted that the Feed-Forward is working well in predicting a favorable soil condition.

Lastly, Figure 5.10c. presents the result in the case of the Recurrent Neural Network. The accuracy is slightly worse than the previous architecture since its overall score is 45%. The predictions' trend plot suggests that the network behaves differently than other models. In fact, rather than giving an average prediction, it tends to oscillate more, especially in the first half of the test set, while it seems to start giving average predictions towards the end.

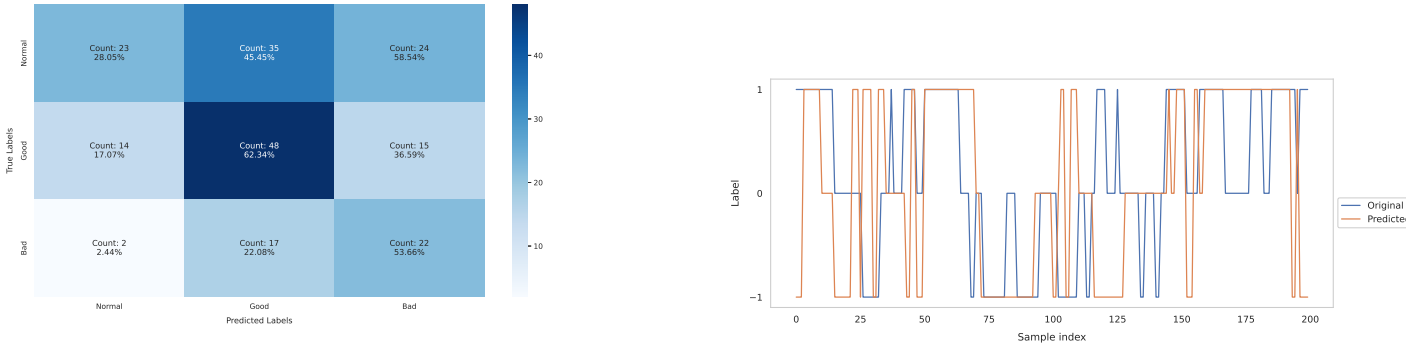
To sum up, the switch to a classification setting seems to have improved model performances in recognizing a drought condition encoded into three labels derived from data. The analysis shows that complex neural network models obtain discrete performances even if they suffer from few available data and noisy input values. To address the first problem, we tried to export the last approach seen during the Regression analysis, using the historical data of the selected features to augment the training data for each classifier. The results show that the Feed-Forward network performs well in recognizing the target signal's good condition. As an additional indicator of robustness, the predictions are kept stable for consecutive weeks even though the model has no indicator of what is the current week.



(a) Results on the test set of the Extremely Randomized Trees ensemble model



(b) Results on the test set of the Feed-Forward Neural Network



(c) Results on the test set of the Recurrent Neural Network

Figure 5.10: Results on the test set in the classification setting. For each model, it is shown the confusion matrix (on the left) and the prediction trend plot (on the right). The confusion matrix allows visualizing a summary of the classification results, stating every possible combination between predicted labels and true labels. The prediction trend allows us to interpret how the model works when giving a prediction. From this figure, it is evident that the first two models give an average label prediction over multiple weeks, while the Recurrent network's prediction tends to oscillate more.

5.2. Multi-Task

This Section presents the results of the analysis in the Multi-Task setting. The goal of this learning paradigm, as described in Section 3.4, is to allow tuning a model's parameter on multiple tasks so that it is able to generalize more by optimizing multiple loss functions.

The residual analysis in the Single-Task highlighted that the target signal for that problem is autoregressive. Therefore, variables are not more helpful than past values of the NDVI anomaly for the target reconstruction. Then, when trying to obtain an approximation of the NDVI anomaly signal using only the input variables chosen by the CMI feature selection algorithm, the models could not fully capture the signal's trend, except for the Recurrent Network. In this case, the model's output demonstrated to be able to reconstruct the target's sign in most cases, an intuition that led us to consider a classification problem. Models in the classification setting obtained better results by considering only the input features, especially in the Feed-Forward Neural Network case. In fact, this model achieved an accuracy of 80% when classifying a target's sample as *good* for a given week.

Starting from the good results of Subsection 5.1.4, the analysis in the Multi-Task setting is performed by training the models using the historical data of the selected features. In this case, the number of selected variables is higher than in the Single-Task setting (i.e., 10 inputs rather than 3 for most of the single targets). Using the whole number of inputs would not allow us to consider the series of their past values again, mainly because the resulting dataset would have a number of features bigger than the dataset processed by the PCA algorithm. That is why we selected the subset of the most relevant features based on their CMI scores by keeping only the first 5 variables, whose scores are larger by at least an order of magnitude than the remaining. The resulting variables are presented in Table 5.13 and those are used for training and evaluating the multi-task models.

5.2.1. Classification of multiple anomalies

This experimental analysis aims to verify whether a model in a Multi-Task setting is able to generalize over the ten sub-basins defined in the Po Valley region. In particular, the goal is to output a label for every area in the target vector using the historical time-series of the subset of selected features presented in Table 5.13.

The results presented in this Section are obtained with two techniques mentioned in Chapter 3 for the Multi-Task learning paradigm. The first enables a knowledge transfer among tasks by sharing a part of the model's architecture, while the second assigns shared

Name	Description
PC1_Prec_16w	First component of the average precipitation in 16 weeks.
PC2_Prec_12w	Second component of the average precipitation in 12 weeks.
PC1_Prec_4w	First component of the average precipitation in 4 weeks.
PC1_Temp_8w	First component of the average temperature in 8 weeks.
PC2_SnowDepth_4w	Second component of the average snow cover depth in 4 weeks.

Table 5.13: Subset of selected features, all having comparable CMI scores. The remaining variables chosen by the feature selection algorithm have obtained CMI scores whose order of magnitude is smaller than those listed in this table.

layers to tasks grouped in clusters.

Feature Transformation based approach

The feature transformation approach provides the sharing of knowledge among tasks by learning a transformed representation of the feature suited to explain all the targets simultaneously. This condition is accomplished using Neural Networks with a shared hidden layer among tasks, with the intuition that this enables a mapping from the original space of features to one joint for all the targets.

This analysis is performed using only Neural Networks, since we can adapt their architecture to make the models learn a common representation of the input features. The dataset splitting for training, validation, and test is the same as the one used to provide Single-Tasks results. The target is no longer a single area but rather a vector that includes all the targets in the case study. Moreover, the models are trained by optimizing multiple loss functions, one for each component of the output vector. A general overview of this architecture is illustrated in Figure 5.11, while Table 5.14 shows the result of the

hyperparameters tuning on the validation set.

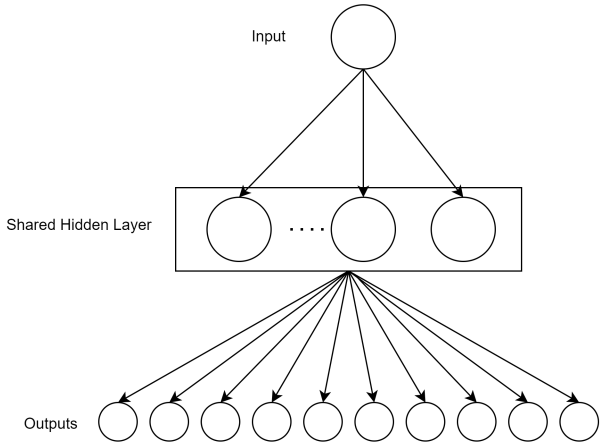


Figure 5.11: Simplified diagram of a Neural Network model used in this Multi-Task analysis. The single shared hidden layer has connection to every output node. The difference with the Single-Task setting is that every node's output is used to evaluate that node's loss function.

Model	**Tuned hyperparamters**
Feed-Forward Neural Network	• *Hidden units = 64* • *epochs=50* • *shuffle=False* • *batch_size=32*
Recurrent Neural Network	• *Hidden units = 10* • *epochs=35* • *Input sequences time-steps = 10* • *Number of stacked RNN layers=1* • *Output step = 1*

Table 5.14: Hyperparameters tuned in the Multi-Task Classification setting.

Figure 5.12 shows the results on the test set for the Feed-Forward Neural Network. The confusion matrices on the areas highlight that the model has been able to generalize on the test set. The overall accuracy is 46%, a positive result since it was achieved despite limited data and noisy input measurements, and on a classification problem with three labels. It

is interesting to evaluate the difference in performance obtained by the Multi-Task model on the area analyzed in the Single-Task setting. In this way, we can compare the two results and verify whether there have been improvements on the signal classification.

Figure 5.14a presents the plot to evaluate how the model predicts the label of the target signal. In this case, the visual interpretation is more complex since the model does not seem to maintain a stable prediction for several consecutive weeks. Probably, having several tasks to optimize causes the classifier to learn to change its prediction more frequently. However, it should still be remembered that the model is not receiving any temporal input on the week for which it is being called to classify.

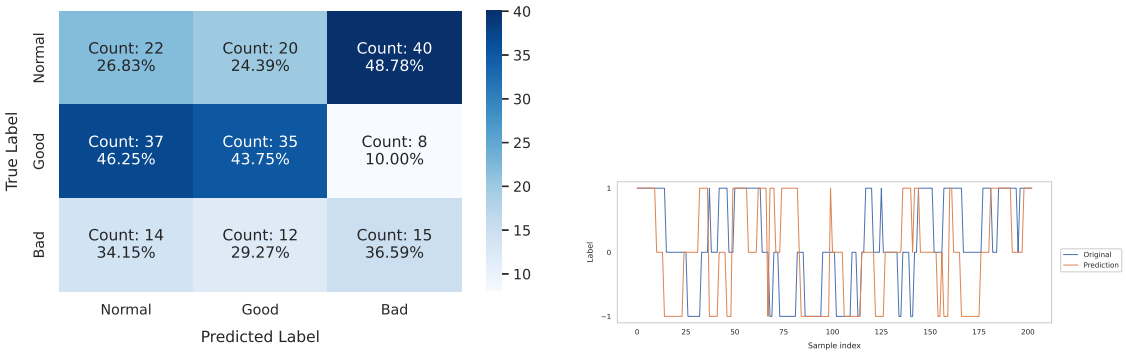
Compared to the Single-Task Feed-Forward Neural Network, the performances obtained on the same target area have worsened, especially when classifying "Good samples". Furthermore, the model seems to struggle in distinguishing "Normal" samples from "Bad" ones. In fact, the number of misclassified samples, in this case, is particularly high.

Despite a deterioration in performance when compared to the Single-Task case, the Multi-task model manages to achieve good generalization among targets. In fact, a single network is able to obtain discrete results on all observed areas, although the accuracy on a single task is worse than that obtained by a network trailed only on this one.

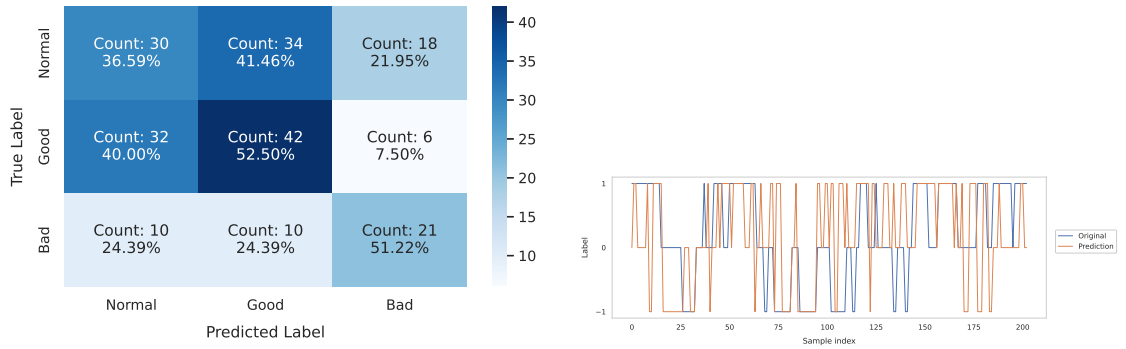
Figure 5.13 presents the results obtained with the recurrent neural network in the multi-task case. The model obtains an average accuracy of about 37%, a worse result than in the previous setting, most likely due to the scarcity of available data. Again, Figure 5.14b demonstrates the trend of the model's predictions. The visual interpretation is more complex than in the Single-Task setting, especially in the central part of the test set, where the prediction signal frequently oscillates between "Normal" and "Bad" frequently. This is reflected in the confusion matrix results, where the model accuracy in the case of the previously mentioned labels is slightly higher than the Feed-Forward model.

Figure 5.12: Confusion matrices of the multi-task Feed-Forward network's prediction on the test set.

Figure 5.13: Confusion matrices of the multi-task Recurrent network's prediction on the test set.



(a) Feed-Forward Neural Network model forward



(b) Recurrent Neural Network model

Figure 5.14: Results on the test set of the multi-task Neural Networks models' predictions for *Emiliani 2* area.

Task clustering based approach

The Task Clustering approach is based on grouping similar tasks to facilitate learning. As discussed in Section 3.4, clustering allows a subset of parameters to be defined, which is then shared exclusively among the tasks in the cluster.

This experiment identified clusters based on Mutual Information scores among the target anomalies. From this analysis, four groups emerged, as illustrated in Table 5.15.

Cluster name	**Areas included**
Cluster 1	Adda, Lambro, Ticino
Cluster 2	Dora, Piemonte nord
Cluster 3	Emiliani 1, Emiliani 2, Piemonte sud
Cluster 4	Garda, Oglio

Table 5.15: Identified clusters based on their Mutual Information scores.

The shared architecture defined for the two Neural Networks under analysis consists of a common hidden layer to all the targets. At the same time, each cluster has a fully connected layer among the joined tasks. The choice of selecting a fully connected layer also for the Recurrent Network has been made using the validation set, while tuning the hyperparameters.

The models in this Subsection use again the historical data of features in Table 5.13, and the dataset splitting percentages are coherent to all the previous problems discussed in this Chapter. The result of the hyperparameters tuning on the validation set are showed in Table 5.16.

The result of the task-clustering approach for Feed-Forward Neural Network is presented in Figure 5.15. The model achieves an overall accuracy of 38%, meaning that grouping tasks into clusters based on their MI scores has not been beneficial in improving the predictive performance. In light of this result, the Feed-Forward without task clustering is preferred since it is a simpler network with the same performance. Again, in Figure 5.17a it is reported the result on the *Emiliani 2* area for comparing it with what we achieved so far. In this case, the task clustering method allows the models to classify better samples belonging to the "Normal" class. The prediction's trend, again, is not easily interpretable, but it seems that there are no notable differences from the previous case.

Figure 5.16 illustrates the results obtained with the Recurrent Neural Network. In this case, the task clustering has boosted the model accuracy, obtaining an overall 50% of cor-

Model	Tuned hyperparamters
Feed-Forward Neural Network	• *Hidden units = 64* • *Hidden units of cluster-specific layers = 64* • *epochs=50* • *shuffle=False* • *batch_size=32*
Recurrent Neural Network	• *Hidden units = 10* • *Hidden units of cluster-specific layers = 64* • *epochs=35* • *Input sequences time-steps = 10* • *Number of stacked RNN layers=1* • *Output step = 1*

Table 5.16: Hyperparameters tuned in the Multi-Task Classification setting, using the task clustering approach for knowledge sharing.

rectly classified samples across all areas. Figure 5.17a focuses on the model's performance on the usual *Emiliani 2* area, for which the classification of "Good" samples is considerably better than in the previous method. Indeed, the prediction is almost comparable to the Single-Task Feed-Forward Network, that obtained a classification accuracy of about 60%: this model's overall accuracy is 46% on the considered area, but the its almost as accurate as the other one when identifying "Good" labels. However, the Recurrent network's ability to correctly identify "Bad" labels seems to be much worse than in the previous approach. This time, the prediction trend allows us to say that the classifier is keeping its prediction stable over consecutive weeks, as it happened in the Single-Task setting.

Figure 5.15: Confusion matrices of the multi-task Feed-Forward's prediction on the test set, where the targets are clustered.

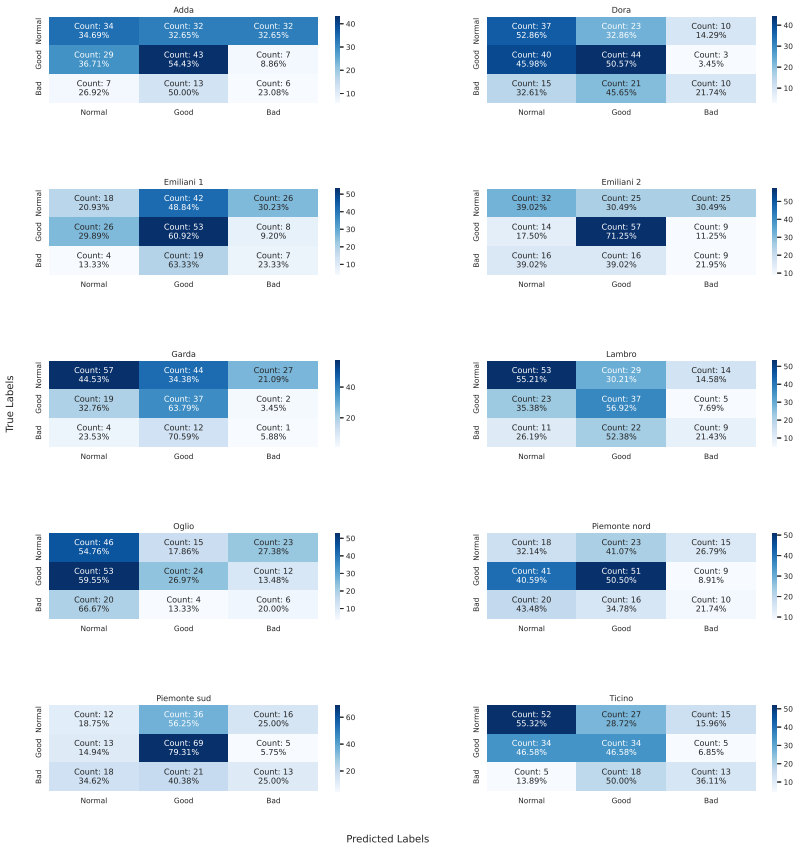
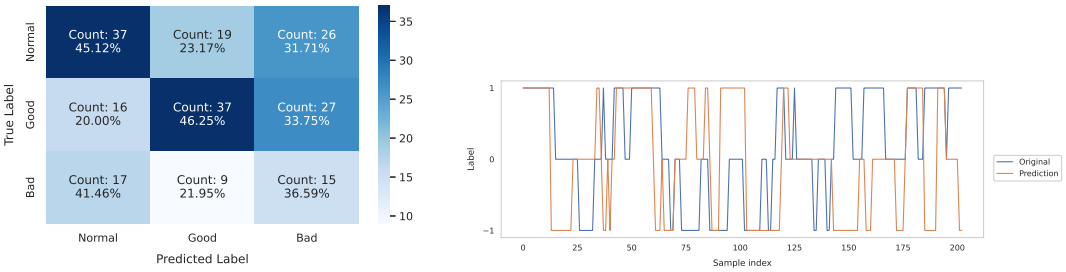
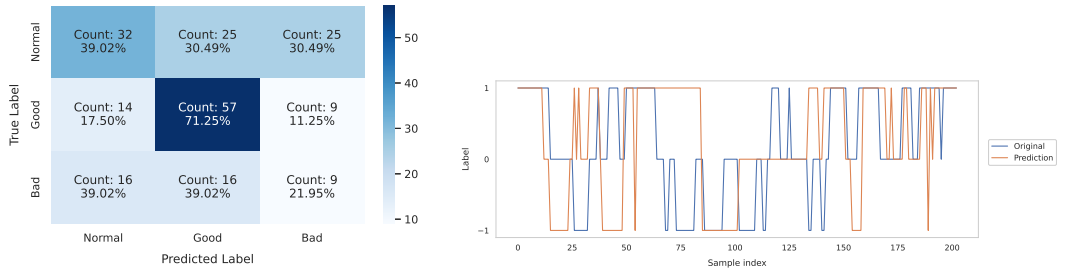


Figure 5.16: Confusion matrices of the multi-task Recurrent network's prediction on the test set, where the targets are clustered.



(a) Feed-Forward Neural Network model



(b) Recurrent Neural Network model

Figure 5.17: Results on the test set of the multi-task Neural Networks models' predictions for *Emiliani 2* area, using the task-clustering approach.

To summarize the content of this Section devoted to the presentation of results in a Multi-Task setting, it can be seen that this target shift resulted in a decrease in classification accuracy in the case of the area monitored in the Single-Task case. However, the models also achieve discrete performance on other areas, allowing to have satisfactory results training one single neural network for all the ten basins. Looking at the trends of the predictions a direct interpretation of the graphs appears more complex since there is more significant variability of the models in predicting the class of a test sample.

Evaluating the performance obtained in the two different cases of Multi-Task learning approaches, we can state that the model that seems most promising is the Feed-Forward Neural Network having a single hidden layer shared by multiple targets. At the same time, task clustering did not benefit when applied to the same type of network. Task clustering does not consider the likely negative cross-correlations between tasks belonging to different clusters that may result in updating the model parameters differently. In contrast, in the case of the Recurrent Neural Network, grouping targets result in a boost in predictive performance across the considered areas.

Despite the deterioration of predictions in the transition to the Multi-Task case, this analysis verified that this approach still yields decent results in the presence of noisy input measurements and few data. In this regard, we considered the historical series of features to allow the model to have an overview of the values of the most informative variables over the past six months. Computations with this set of features are not a problem from a computational point of view, especially due to the small number of variables selected by the CMI-based feature selection algorithm. Another advantage of this approach is the ability to analyze multiple areas in parallel to assess their drought conditions, with the only need to train a single neural network.

6 | Conclusions and future developments

This last chapter summarizes the main results obtained from this thesis work and some possible improvements to the methodology proposed.

The data used for this thesis work, whose sources and procedures for extraction are discussed in Chapter 4, are averages of meteorological data obtained on the sub-basins identified on the study area, the Po Valley. The unsupervised PCA algorithm first processed the obtained dataset to reduce the dimensionality, considering the high number of input features.

At this point, analyses were carried out in parallel for two settings: Single-Task and Multi-Task. In both cases, the feature selection algorithm based on Conditional Mutual Information, introduced in Chapter 2, was performed on the entire dataset. In this way, we have identified the relevant and non-redundant subset of features for the target considered.

Next, with the feature selection results, we trained the model classes presented in Chapter 3 in three different problems. Initially, we verified whether the features add information to the NDVI anomaly prediction obtained with an autoregressive model. The analysis of the prediction's residuals led us to discover that the NDVI anomaly signal is autoregressive. Therefore, we focused on obtaining a good approximation of the target using only the chosen features. This approach obtained better results when we extended the input data seen by each model by considering the input historical values in the past six months. However, the reconstructed signal appeared lagged with respect to the original data, most likely due to the noisy values the model received in input. Nevertheless, the Recurrent Neural Network prediction trend has shown that the model has learned to recognize the sign of the target signal, obtaining an accuracy of about 70% in this context. This result prompted us to try a classification approach, where the purpose of the models is to assign a label to each sample. The three labels identified represent three different conditions of the signal, which we have interpreted with as many drought conditions.

In classification, the models performances have definitely improved, especially in the case

of the Single-Task setting. Taking as reference the *Emiliani 2* sub-basin, chosen since it is the area with the highest concentration of arable areas, the Feed-Forward Neural Network is able to correctly classify the test samples with an accuracy of almost 60%. By analyzing the trend of the predictions, it can be seen how the model maintains the prediction for several consecutive weeks, even though it does not receive any information on the current week as input. These results are very positive since they imply that the chosen models were able to extrapolate information from the input data, in a context with limited number of samples and noisy data.

In the Multi-Task case, on the other hand, the results seem to be worse when comparing the trend of the predictions on the same area used in the previous analysis. The overall accuracy obtained from the predictions across all the sub-basins of the study area does not exceed 50% (which is still a good result since the classification problem is with three labels). Recurrent Neural Network using the task clustering approach achieves performances on the *Emiliani 2* sub-basin that are almost comparable to the good results obtained in the Single-Task setting, with the advantage that a single neural network fed with the shared input features is able to produce all the ten outputs, one for each basin, at the same time.

The results obtained on the classification problem suggest that this setting may be particularly fruitful for predicting future drought conditions. The multi-task models demonstrated how it is possible to monitor the trend of the drought index on multiple geographically distinct areas in parallel. This analysis was conducted using a dataset common to all sub-basins, obtained from the same type of meteorological data extracted for different coordinates. As a future step, it may be interesting to add local variables specifically suited for the peculiar characteristics of each basin together with the very general variables considered in this work.

An additional level of complexity could be introduced by variables that are directly influenced by the area demography, to evaluate how much the urbanization affects the classification of the state of drought.

A further step could be to target different drought indices beyond the NDVI. In fact, in this thesis, we used different models to reconstruct the link between the variables we have chosen and the anomaly of the NDVI signal. However, the Multi-Task approach could also be used for the simultaneous reconstruction, or classification, of several indices on the state of vegetation. NDVI values are more accurate mid-season, during the intermediate growth phase, while other indices are more significant for monitoring different stages. For example, the Normalized Difference Red Edge Vegetation Index (NDRE) [103] is

indicated for crops that have reached a state of maturity, while the Modified Soil-Adjusted Vegetation Index (MSAVI) [104] is helpful in the early stage of growth.

Finally, a further improvement to the approach could be a different way of constructing the dataset in which more advanced techniques than simple arithmetic averaging are employed to obtain the data value for given geographic coordinates. In addition, in this thesis, the ten study areas analyzed result from a selection of hydrologically already distinct regions. An advancement could be the use of data-driven techniques to create the target basins.

Bibliography

[1] Anne F Van Loon, Kerstin Stahl, Giuliano Di Baldassarre, Julian Clark, Sally Rangecroft, Niko Wanders, Tom Gleeson, Albert IJM Van Dijk, Lena M Tallaksen, Jamie Hannaford, et al. Drought in a human-modified world: reframing drought definitions, understanding, and analysis approaches. *Hydrology and Earth System Sciences*, 20(9):3631–3650, 2016.

[2] Ya Ding, Michael J Hayes, and Melissa Widhalm. Measuring economic impacts of drought: a review and discussion. *Disaster Prevention and Management: An International Journal*, 2011.

[3] Regina Below, Emily Grover-Kopec, and Maxx Dilley. Documenting drought-related disasters: A global reassessment. *The Journal of Environment & Development*, 16 (3):328–344, 2007.

[4] Joël M Durant, Dag Ø Hjermann, Tycho Anker-Nilssen, Grégory Beaugrand, Atle Mysterud, Nathalie Pettorelli, and Nils Chr Stenseth. Timing and abundance as key mechanisms affecting trophic interactions in variable environments. *Ecology Letters*, 8(9):952–958, 2005.

[5] Jeremy T Kerr and Marsha Ostrovsky. From space to species: ecological applications for remote sensing. *Trends in ecology & evolution*, 18(6):299–305, 2003.

[6] Woody Turner, Sacha Spector, Ned Gardiner, Matthew Fladeland, Eleanor Sterling, and Marc Steininger. Remote sensing for biodiversity science and conservation. *Trends in ecology & evolution*, 18(6):306–314, 2003.

[7] Nathalie Pettorelli, Jon Olav Vik, Atle Mysterud, Jean-Michel Gaillard, Compton J Tucker, and Nils Chr Stenseth. Using the satellite-derived ndvi to assess ecological responses to environmental change. *Trends in ecology & evolution*, 20(9):503–510, 2005.

[8] Ranga B Myneni, Forrest G Hall, Piers J Sellers, and Alexander L Marshak. The interpretation of spectral vegetation indexes. *IEEE Transactions on Geoscience and Remote Sensing*, 33(2):481–486, 1995.

[9] Quan Wang, Samuel Adiku, John Tenhunen, and André Granier. On the relationship of ndvi with leaf area index in a deciduous forest site. *Remote sensing of environment*, 94(2):244–255, 2005.

[10] Marta Zaniolo, Matteo Giuliani, Andrea Francesco Castelletti, and Manuel Pulido-Velazquez. Automatic design of basin-specific drought indexes for highly regulated water systems. *Hydrology and Earth System Sciences*, 22(4):2409–2424, 2018.

[11] Marta Zaniolo, Matteo Giuliani, and Andrea Castelletti. Data-driven modeling and control of droughts. *IFAC-PapersOnLine*, 52(23):54–60, 2019.

[12] Davide Cananzi. Improving drought monitoring via machine learning: a new impact-based drought index. 2021.

[13] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *The journal of machine learning research*, 13(1):27–66, 2012.

[14] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[15] Ethem Alpaydin. *Introduction to Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 3 edition, 2014. ISBN 978-0-262-02818-9.

[16] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[17] Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324, 1997.

[18] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[19] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[20] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[21] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. *Advances in neural information processing systems*, 18, 2005.

[22] Zheng Zhao and Huan Liu. Spectral feature selection for supervised and unsu-

pervised learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1151–1157, 2007.

[23] Davis John C. Robert J. Sampson. *Statistics and Data Analysis in Geology*. John Wiley & Sons, Inc., USA, 1973. ISBN 0471198951.

[24] Huan Liu and Rudy Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, pages 388–391. IEEE, 1995.

[25] Jacek Biesiada and Wlodzislaw Duch. Feature selection for high-dimensional data—a pearson redundancy based filter. In *Computer recognition systems 2*, pages 242–249. Springer, 2007.

[26] George H John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *Machine learning proceedings 1994*, pages 121–129. Elsevier, 1994.

[27] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224, 2004.

[28] Daphne Koller and Mehran Sahami. Toward optimal feature selection. Technical report, Stanford InfoLab, 1996.

[29] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[30] Roberto Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on neural networks*, 5(4):537–550, 1994.

[31] Aaron D Wyner. A definition of conditional mutual information for arbitrary ensembles. *Information and Control*, 38(1):51–59, 1978.

[32] Mario Beraha, Alberto Maria Metelli, Matteo Papini, Andrea Tirinzoni, and Marcello Restelli. Feature selection via mutual information: New theoretical insights. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2019.

[33] Liam Paninski. Estimation of entropy and mutual information. *Neural computation*, 15(6):1191–1253, 2003.

[34] Alkiviadis Tsimpiris, Ioannis Vlachos, and Dimitris Kugiumtzis. Nearest neighbor estimate of conditional mutual information in feature selection. *Expert Systems with Applications*, 39(16):12697–12708, 2012.

[35] Weihao Gao, Sreeram Kannan, Sewoong Oh, and Pramod Viswanath. Estimating mutual information for discrete-continuous mixtures. *Advances in neural information processing systems*, 30, 2017.

[36] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007. ISBN 0387310738.

[37] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

[38] Thomas G Dietterich et al. Ensemble learning. *The handbook of brain theory and neural networks*, 2(1):110–125, 2002.

[39] Robi Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45, 2006.

[40] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[41] Harris Drucker, Corinna Cortes, Lawrence D Jackel, Yann LeCun, and Vladimir Vapnik. Boosting and other ensemble methods. *Neural Computation*, 6(6):1289–1301, 1994.

[42] Kamal M Ali and Michael J Pazzani. On the link between error correlation and error reduction in decision tree ensembles. 1995.

[43] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[44] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[45] Yali Amit and Donald Geman. Randomized inquiries about shape: An application to handwritten digit recognition. Technical report, CHICAGO UNIV IL DEPT OF STATISTICS, 1994.

[46] Abhishek Sharma. A simple analogy to explain decision tree vs. random forest, 2020. [Online].

[47] Te Han, Dongxiang Jiang, Qi Zhao, Lei Wang, and Kai Yin. Comparison of random forest, artificial neural networks and support vector machine for intelligent diagnosis of rotating machinery. *Transactions of the Institute of Measurement and Control*, 40(8):2681–2693, 2018.

[48] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[49] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[50] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *towards data science*, 6(12):310–316, 2017.

[51] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[52] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[53] Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

[54] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[55] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[56] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[57] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[58] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[59] Max Little, Patrick McSharry, Eric Hunter, Jennifer Spielman, and Lorraine Ramig. Suitability of dysphonia measurements for telemonitoring of parkinson's disease. *Nature Precedings*, pages 1–1, 2008.

[60] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[61] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine learning*, 73(3):243–272, 2008.

[62] Yusuke Shinohara. Adversarial multi-task learning of deep neural networks for robust speech recognition. In *Interspeech*, pages 2369–2372. San Francisco, CA, USA, 2016.

[63] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003, 2016.

[64] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 137–144, 2009.

[65] Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20(6):3465–3489, 2010.

[66] Sebastian Thrun and Joseph O'Sullivan. Discovering structure in multiple learning tasks: The tc algorithm. In *ICML*, volume 96, pages 489–497, 1996.

[67] Laurent Jacob, Jean-philippe Vert, and Francis Bach. Clustered multi-task learning: A convex formulation. *Advances in neural information processing systems*, 21, 2008.

[68] Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *ICML*, 2011.

[69] Edwin V Bonilla, Kian Chai, and Christopher Williams. Multi-task gaussian process prediction. *Advances in neural information processing systems*, 20, 2007.

[70] Yu Zhang and Dit-Yan Yeung. A regularization approach to learning task relationships in multitask learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(3):1–31, 2014.

[71] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[72] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[73] Intergovernmental Panel On Climate Change. Climate change 2007: the physical science basis. *Agenda*, 6(07):333, 2007.

[74] Compton J Tucker. Red and photographic infrared linear combinations for monitoring vegetation. *Remote sensing of Environment*, 8(2):127–150, 1979.

[75] SD Prince. A model of regional primary production for use with coarse resolution satellite data. *International Journal of Remote Sensing*, 12(6):1313–1330, 1991.

[76] Amir AghaKouchak, A Farahmand, FS Melton, J Teixeira, MC Anderson, Brian D Wardlow, and CR Hain. Remote sensing of drought: Progress, challenges and opportunities. *Reviews of Geophysics*, 53(2):452–480, 2015.

[77] Christopher SR Neigh, Compton J Tucker, and John RG Townshend. North american vegetation dynamics observed with multi-resolution satellite data. *Remote Sensing of Environment*, 112(4):1749–1772, 2008.

[78] National Oceanic and Atmospheric Administration. Star - global vegetation health products : Browse archived image of selected country.

[79] Beatrice Monteleone, Brunella Bonaccorso, and Mario Martina. A joint probabilistic index for objective drought identification: the case study of haiti. *Natural Hazards and Earth System Sciences*, 20(2):471–487, 2020.

[80] Prasad Srinivasa Thenkabail and MSDN Gamage. *The use of remote sensing data for drought assessment and monitoring in Southwest Asia*, volume 85. Iwmi, 2004.

[81] Eric Vermote. Program. 2019. noaa climate data record (cdr) of avhrr normalized difference vegetation index (ndvi), version 5. noaa national centers for environmental information, 2019.

[82] J Pizon. Satellite time series correction of orbital drift artifacts using empirical mode decomposition. *Hilbert-Huang transform: introduction and applications*, pages 167–186, 2005.

[83] CR Nagaraja Rao and J Chen. Inter-satellite calibration linkages for the visible and near-infared channels of the advanced very high resolution radiometer on the noaa-7,-9, and-11 spacecraft. *International Journal of Remote Sensing*, 16(11):1931–1942, 1995.

[84] Jorge E Pinzon and Compton J Tucker. A non-stationary 1981–2012 avhrr ndvi3g time series. *Remote sensing*, 6(8):6929–6960, 2014.

[85] Kamel Didan. Myd13q1 modis/aqua vegetation indices 16-day l3 global 250m sin grid v006 (2015), 2015.

[86] David R Easterling, Trevor WR Wallis, Jay H Lawrimore, and Richard R Heim Jr. Effects of temperature and precipitation trends on us drought. *Geophysical Research Letters*, 34(20), 2007.

[87] Richard C Cornes, Gerard van der Schrier, Else JM van den Besselaar, and Philip D

Jones. An ensemble version of the e-obs temperature and precipitation data sets. *Journal of Geophysical Research: Atmospheres*, 123(17):9391–9409, 2018.

[88] Andreas Gobiet, Sven Kotlarski, Martin Beniston, Georg Heinrich, Jan Rajczak, and Markus Stoffel. 21st century climate change in the european alps—a review. *Science of the Total Environment*, 493:1138–1151, 2014.

[89] Michael Matiu, Alice Crespi, Giacomo Bertoldi, Carlo Maria Carmagnola, Christoph Marty, Samuel Morin, Wolfgang Schöner, Daniele Cat Berro, Gabriele Chiogna, Ludovica De Gregorio, et al. Observed snow depth trends in the european alps: 1971 to 2019. *The Cryosphere*, 15(3):1343–1382, 2021.

[90] Michael Matiu, Alice Crespi, Giacomo Bertoldi, CM Carmagnola, S Morin, S Kotlarski, and V Weilguni. Snow cover in the european alps: Station observations of snow depth and depth of snowfall. data set. Technical report, -, 2020.

[91] Aiguo Dai. Increasing drought under global warming in observations and models. *Nature climate change*, 3(1):52–58, 2013.

[92] Hege Hisdal, Lena M Tallaksen, Bente Clausen, Elizabeth Peters, Alan Gustard, and H VanLauen. Hydrological drought characteristics. *Developments in water science*, 48(5):139–198, 2004.

[93] K Martin Perales, Catherine L Hein, Noah R Lottig, and M Jake Vander Zanden. Lake water level response to drought in a lake-rich region explained by lake and landscape characteristics. *Canadian Journal of Fisheries and Aquatic Sciences*, 77 (11):1836–1845, 2020.

[94] QGIS Software. Qgis.org, 2022. qgis geographic information system. qgis association. http://www.qgis.org.

[95] SINAnet Groupware. Uso, copertura e consumo di suolo, 2018.

[96] Rui Li, Atsushi Tsunekawa, and Mitsuru Tsubo. Index-based assessment of agricultural drought in a semi-arid region of inner mongolia, china. *Journal of Arid Land*, 6(1):3–15, 2014.

[97] Lifeng Luo, Deanna Apps, Samuel Arcand, Huating Xu, Ming Pan, and Martin Hoerling. Contribution of temperature and precipitation anomalies to the california drought during 2012–2015. *Geophysical Research Letters*, 44(7):3184–3192, 2017.

[98] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space.

The London, Edinburgh, and Dublin philosophical magazine and journal of science, 2(11):559–572, 1901.

[99] Hirotugu Akaike. Autoregressive model fitting for control. In *Selected Papers of Hirotugu Akaike*, pages 153–170. Springer, 1998.

[100] R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

[101] Greta M Ljung and George EP Box. On a measure of lack of fit in time series models. *Biometrika*, 65(2):297–303, 1978.

[102] Karl Pearson F.R.S. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900. doi: 10.1080/14786440009463897.

[103] EM Barnes, TR Clarke, SE Richards, PD Colaizzi, J Haberland, M Kostrzewski, P Waller, C Choi, E Riley, T Thompson, et al. Coincident detection of crop water stress, nitrogen status and canopy density using ground based multispectral data. In *Proceedings of the Fifth International Conference on Precision Agriculture, Bloomington, MN, USA*, volume 1619, page 6, 2000.

[104] Jiaguo Qi, Abdelghani Chehbouni, Alfredo R Huete, Yann H Kerr, and Soroosh Sorooshian. A modified soil adjusted vegetation index. *Remote sensing of environment*, 48(2):119–126, 1994.

Acknowledgements

This work has been supported by the CLINT research project funded by the H2020 Programme of the European Union under Grant Agreement No. 101003876.