Multi-target Learning and Prediction: Novel Methods, and Applications

Ekaterina Antonenko

Supervised by Jesse Read

A Ph.D. thesis manuscript submitted for review

Laboratoire d'Informatique

École Polytechnique

Institut Polytechnique de Paris

France

31 August 2023

Abstract

A multi-output machine learning predictive task is characterized by the need to predict multiple numerical or categorical outputs for each instance. While a straightforward approach involves modeling each output separately, employing joint modeling techniques often enhances prediction performance and yields superior results due to analyzing and exploiting the interdependencies between the various target variables.

In the context of regression, when addressing joint modeling, several challenges emerge. One common issue is that many methods tend to assume a single-modal Gaussian distribution, while the ground-truth target distribution does not necessarily correspond to this assumption. To tackle this issue, we propose a novel solution based on Regressor Chains, which are basically chains of single-output models incorporating already predicted targets to the modeling of the subsequent ones. The proposed approach, Multi-Modal Ensemble of Regressor Chains, offers a mechanism to effectively handle multi-modal target distributions, enhancing the model's predictive capabilities while maintaining flexibility with regard to base estimators of Regressor Chains.

Second, we study multi-target regression in the scenarios when some of the target values are known in the prediction phase and can be leveraged to predict the unknown ones without re-training the model. This may be the case if, for example, the training data is restricted or not available

anymore. To this end, we develop an approach for backward inference in Regressor Chains which incorporates the information about the fixed values and includes them into the joint modeling of the other values regardless of the chain order and positioning of the known targets in it. Additionally, the proposed solution provides the distribution for each instance instead of a mean value.

In the classification domain, we introduce a novel application of multi-output Random Forests. We propose using them in an Autoreplicative fashion to perform missing value imputation or, in other words, denoise the data. The proposed method is evaluated across a range of various datasets, demonstrating its efficacy. Moreover, we develop a general framework that unifies different imputation methods and makes it possible to select a method by tuning hyperparameters. We make an important distinction by telling apart the procedural and iterative methods. The procedural methods are optimized on the observed values and impute missing ones only once. Oppositely, the iterative methods update imputed values iteratively in cycles until convergence criteria are reached. We add the newly proposed method, Autoreplicative Random Forests to the general framework both in procedural and iterative versions. Additionally, we extend it with distributional iterative Autoreplicative Random Forests that incorporate the model's confidence of imputation on each iteration to the modeling in subsequent cycles and output a distribution for each imputed value in the end.

Finally, we extend the applicability of Autoreplicative Random Forests to

genomic high-dimensional data, namely Single Nucleotide Polymorphisms datasets, by imputing missing values in separate windows and incorporating the processed windows into the modeling of the subsequent ones in a chained fashion. Through a comprehensive experimental analysis, our approach showcases competitive and even outperforming results when compared to other reference-free methods.

Résumé en Français

Une tâche prédictive de l'apprentissage automatique à sorties multiples se caractérise par le besoin de prédire plusieurs sorties numériques ou catégorielles pour chaque instance. Alors qu'une approche directe consiste à modéliser chaque sortie séparément, l'utilisation de techniques de modélisation conjointe améliore souvent les performances de prédiction et produit des résultats supérieurs en raison de l'analyse et de l'exploitation des interdépendances entre les différentes variables cibles.

Dans le contexte de la régression, lorsqu'il s'agit de la modélisation conjointe, plusieurs défis émergent. Premièrement, un problème courant est que de nombreuses méthodes ont tendance à supposer une distribution gaussienne unimodale, alors que la distribution cible réelle ne correspond pas nécessairement à cette hypothèse. Pour résoudre ce problème, nous proposons une nouvelle solution basée sur les chaînes de régresseurs, qui sont essentiellement des chaînes de modèles à une seule sortie incorporant les cibles déjà prédites comme variables d'entrée dans la modélisation des cibles suivantes.

Nous proposons l'ensemble multi-modal de chaînes de régresseurs offrant un mécanisme pour gérer efficacement les distributions cibles multimodales et qui améliorent les capacités de la prédiction du modèle, tout en maintenant la flexibilité des estimateurs de base composant les chaînes de régresseurs.

Deuxièmement, nous étudions la régression multi-cibles dans les scénarios où certaines des valeurs cibles sont connues dans la phase de prédiction et qui sont exploitées pour prédire les valeurs inconnues sans ré-entraîner le modèle. Cela est interessant si, par exemple, les données d'entraînement sont restreintes ou ne sont plus disponibles.

À cette fin, nous développons une approche pour l'inférence régressive intègrant les informations sur les valeurs connues dans la modélisation conjointe des autres inconnues, indépendamment de l'ordre des chaînes et de la position des cibles connues dans celle-ci. De plus, la solution proposée fournit la distribution pour chaque instance au lieu d'une valeur moyenne.

Dans le domaine de la classification, nous introduisons une nouvelle application des forêts aléatoires à sorties multiples. Nous proposons de les utiliser de manière autoréplicative pour effectuer l'imputation des valeurs manquantes ou, en d'autres termes, pour débruiter les données. La méthode proposée est évaluée sur un ensemble de jeux de données différents pour démontrer son efficacité dans des applications du monde réel. De plus, nous développons un cadre général qui unifie les différentes méthodes d'imputation et permet de sélectionner une méthode en ajustant les hyperparamètres. Nous faisons une distinction importante en différenciant les méthodes procédurales et itératives. Les méthodes procédurales sont optimisées sur les valeurs observées et imputent les valeurs manquantes une seule fois. En revanche, les méthodes itératives mettent à jour les valeurs imputées de manière itérative jusqu'à ce que les critères de convergence soient atteints.

Nous ajoutons la méthode nouvellement proposée, les forêts aléatoires autoréplicatives, au cadre général dans les versions procédurales et itératives. De plus, nous l'étendons avec des forêts aléatoires autoréplicatives itératives distributionnelles qui intègrent la confiance du modèle envers l'imputation à chaque itération dans la modélisation des cycles suivants et produisent une distribution pour chaque valeur imputée à la fin.

Enfin, nous étendons l'applicabilité des forêts aléatoires autoréplicatives aux données génomiques de haute dimension, notamment aux ensembles de données de polymorphismes mononucléotidiques, en imputant les valeurs manquantes dans des fenêtres séparées et en incorporant les fenêtres traitées dans la modélisation des suivantes de manière enchaînée. Notre approche présente des résultats compétitifs, voire supérieurs, par rapport à d'autres méthodes sans référence.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	Autoencoder
ARF	Autoreplicative Random Forest
aRRMSE	Average Relative Root Mean Squared Error
CC	Classifier Chain
DAE	Denoising Autoencoder
DAG	Directed Acyclic Graph
ditARF	Distributional iterative Autoreplicative Random Forest
DT	Decision Tree
ECC	Ensemble of Classifier Chains
EM	Expectation-Maximization
ERC	Ensemble of Regressor Chains
GWAS	Genome-Wide Association Studies
itAE	Iterative Autoencoder
itARF	Iterative Autoreplicative Random Forest
itPCA	Iterative Principal Component Analysis

MAE	Mean Absolute Error
MAR	Missing At Random
MCAR	Missing Completely At Random
mhsERC	Metropolis-Hastings Sampled Ensemble of Regressor Chains
mhsRC	Metropolis-Hastings Sampled Regressor Chain
MICE	Multiple Imputation by Chained Equations
MLE	Maximum Likelihood Estimation
mmERC	Multi-Modal Ensemble of Regressor Chains
MNAR	Missing Not At Random
MSE	Mean Squared Error
mtRF	Multi-target Random Forest
pAE	Procedural Autoencoder
pARF	Procedural Autoreplicative Random Forest
PCA	Principal Component Analysis
pPCA	Procedural Principal Component Analysis
RC	Regressor Chain
RF	Random Forest
SNP	Single Nucleotide Polymorphism
stRF	Single-target Random Forest
UCF	Uniform Cost Function
WD	Wasserstein Distance

Chapter 1

Introduction

Information and data in today's world are everywhere. It is natural for humankind to collect and analyze this information and make inferences and decisions based on interconnections within the data. Automated machine learning has gained huge popularity in recent years and has successfully helped people in data processing. In the case of labeled data, a machine learning algorithm is able to explore the connections and interactions between the properties of a particular data instance and its label and hence to predict labels of new incoming unlabeled instances. Frequently, an instance may be associated with a set of multiple outputs, also known as targets or labels, instead of a single one. This scenario is known as *multi-output learning* and is encountered in a wide variety of domains. While it is possible to build a separate model for each output, the relations and dependencies between the data and outputs can be captured in a fundamentally more profound way if the outputs are modeled jointly by a single model.

1.1 Real-world applications of multi-output modeling

In this section, we discuss a few examples of application areas for multi-output machine learning methods. However, we want to emphasize that this is just a very small subset of domains where the perspective of multi-output modeling is relevant. In the examples below, the targets are expected to impact each other, and the joint modeling can reveal these interactions.

- **Medical Diagnosis and Treatment design**

  Multi-output methods can be used to predict various medical outcomes simultaneously, such as predicting the progression of different diseases, suggesting personalized treatment plans, and predicting patient outcomes (e.g. survival rates, and recovery times) influenced by the previous factors.

- **Environmental Monitoring and Analysis**

  Multi-output models can predict multiple environmental variables like air quality, temperature, humidity, and pollution levels, enabling a more comprehensive understanding of environmental conditions and their impact.

- **Climate Modeling**

  Climate models often involve predicting various climate-related variables like temperature, precipitation, sea levels, and ocean currents, which are interconnected and impact each other.

- **Vegetation Forecasting**

  Vegetation models may, for example, describe the ratios of vegetation types per earth unit. By altering, e.g. the climatic or soil variables, we can query potential vegetation distribution in the changing environment.

- **Bioinformatics and Genomics**

  Multi-output methods can be employed to predict various properties of biological molecules, such as predicting the function of proteins, gene expression levels, and interactions between different molecules.

- **Image Analysis and Computer Vision**

  Multi-output models can simultaneously predict different attributes in images, such as object detection, segmentation, and recognition of multiple objects within a single image.

- **Social Media Analysis**

  In social media, multi-output methods can be used to predict multiple user engagement metrics, sentiment scores for different aspects of a text, and user behaviors across various social platforms.

- **Energy Consumption Forecasting**

  Predicting multiple energy consumption variables, such as electricity, gas, and water usage, can help optimize energy distribution and management in smart grids and buildings.

In this thesis, we discuss in more detail multi-output modeling for potential vegetation prediction (e.g. in the absence of human and urban activity) as well as missing value imputation in genomics data (which may be further used for detecting associations between the genotype and phenotypes).

1.2 Goals and objectives

The overall goal of this thesis is to develop and enhance machine learning methods capable of uncovering and exploiting potential connections and relationships between the outputs. We aim to have an impact in the field

of multi-output modeling by enlarging the area where the multi-output approaches are beneficial and thus promoting their usage in general.

Generally, the fundamental objectives of this thesis include the adaptation of the existing machine learning multi-output techniques in the cases where they perform sub-optimally. More specifically, we begin with throwing new light on the modeling of multi-modal target distributions where existing multi-target approaches typically do not capture such specific data structures and tend to predict non-relevant values that are in fact rarely observed.

After that, we bring up the problem when some of the target values are known after the training and before the prediction and are to be incorporated into the joint inference of unknown targets.

We hypothesize and further prove that in both scenarios the multi-output methods known in the literature as Regressor Chains can be generalized to perform more refined joint modeling and thus address both aforementioned challenges.

Finally, we point out that the imputation of missing values in features can be viewed as a multi-output predictive problem where, consequently, joint multi-output modeling may achieve superior performance. We discuss the related work from this perspective and uncover new insights on how multi-output methods such as Random Forests can be successfully adapted for missing value imputation, or denoising, in the data.

1.3 Summary of contributions

The first part of this thesis is dedicated to the challenges of multi-target regression, i.e. when the outputs are continuous. One of the classic multi-target methods is Regressor Chains which order targets into chains and incorporate information about the previous targets into the predictions of

the subsequent ones. We propose a new approach, Multi-Modal Ensembles of Regressor Chains (mmERC), which improves the performance of Regressor Chains when the data distribution is multi-modal, e.g. consists of several clusters. In this case, traditional models including Regressor Chains tend to put the predictions in between actual clusters, where the data is unlikely to exist in practice. Our approach is better adapted to such data distributions and successfully models multi-modal distributions.

Another challenge of multi-target Regression Chains is how to include information about target values that come known in the prediction phase. While one may argue that it is possible to predict all targets and modify the known ones, the question of interest is the joint distribution of these targets, when modifying one target affects the predictions of the others. From a certain point of view, this problem may be reformulated as missing value imputation in the target space, where some values are observed and others need to be predicted, or imputed. To this end, we design Metropolis-Hastings sampled Regressor Chains (mhsRC) and Metropolis-Hastings sampled Ensembles of Regressor Chains (mhsERC) which allow backward inference and modeling of the outputs together while incorporating the given information about the targets. We apply this approach to both synthetic and real-world vegetation distribution data and study potential vegetation distribution when the urban activity is removed.

The second part of this thesis is dedicated to missing value imputation which we consider as a multi-output problem, where features become targets to predict, or impute. We propose Autoreplicative Random Forests in procedural (pARF), iterative (itARF), and distributional iterative (ditARF) versions. Autoreplicative Random Forests receive the same set of features as inputs and outputs and successfully train to denoise, or impute, data. Additionally, we compare different existing methods for missing value imputation and incorporate them into a common framework where a method can be selected via hyperparameter tuning. In an experimental

study, we show that Autoreplicative Random Forests are very efficient for missing value imputation while maintaining low computational complexity.

We also propose an extension of ARFs, Chains of Autoreplicative Random Forests (ChARF), for missing value imputation in Single Nucleotide Polymorphism (SNP) genomic data. The important characteristics of such data are that the data is high-dimensional but low-sampled and ordering of the features is important. The proposed solution, ChARF, takes into account these properties and outperforms the baselines in most experiments.

1.4 Thesis organization

The remainder of the thesis is organized as follows. Chapter 2 presents the background required for understanding the specifics and challenges of multi-output modeling together with a general overview of the related work. In Chapter 3, we present Multi-Modal Ensembles of Regressor Chains. Chapter 4 discusses backward inference in probabilistic Regressor Chains and introduces Metropolis-Hastings sampled [Ensembles of] Regressor Chains. Chapter 5 describes missing value imputation as a multi-label task and presents Autoreplicative Random Forests which successfully solve this problem. In Chapter 6, we present Chains of Autoreplicative Random Forests which impute missing values in high-dimensional and low-sampled Single Nucleotide Polymorphisms data. Finally, Chapter 7 concludes the thesis by providing an overall summary of our contributions, and by presenting potential future work to extend the research explored in this thesis.

1.5 List of works appearing in this thesis

The contributions in this thesis are available as published articles or preprints.

- E. Antonenko and J. Read. Multi-modal Ensembles of Regressor Chains for Multi-output Prediction. *Advances in Intelligent Data Analysis XX*, IDA 2022.

- E. Antonenko and J. Read. Chains of Autoreplicative Random Forests for missing value imputation in high-dimensional datasets. *ArXiv e-prints*, 2023. This preprint was presented at the Multi-Label Learning workshop, ECML PKDD 2022, and received the Best Paper award.

- E. Antonenko, A. Carreño and J. Read. Autoreplicative Random Forests for missing value imputation. *preprint*, 2023. Preprint.

- E. Antonenko, R. Beigaitė, M. Mechenich, J. Read and I. Žliobaitė. Backward inference in probabilistic Regressor Chains with distributional constraints. *preprint*, 2023. Preprint.

Chapter 2

Background

This thesis is focused on multi-output predictive models. In this chapter, we provide a general introduction to this domain. First, we describe the concept of machine learning and explain the key ideas behind this field. Second, we introduce multi-output models, followed by marginal and joint modeling in multi-output settings. Further, we proceed with giving background on some particular multi-output methods. Then, we describe missing value imputation which may be also seen as a multi-output prediction problem. We conclude this chapter by summarizing the notation used throughout the thesis.

2.1 Machine learning

Machine learning is a fast-developing discipline that investigates the way computers can automatically search for the best model to explain the data via an optimization process. The targeted model automatically captures

complex patterns in data and makes intelligent decisions based on the extracted information.

In supervised learning, models are built with an input set of feature vectors X and an output set of target targets Y (comprising one or several targets), each represented by a number of instances. Each instance $\boldsymbol{x} \in X$ is described by a feature vector $\boldsymbol{x} = [x_1, \ldots, x_p]$ and associated with an output vector $\boldsymbol{y} = [y_1, \ldots, y_L]$, a dataset $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}_{i=1}^{N}$ consists of N instances. If the output is represented by categorical variables, the task is called classification. If the outputs are continuous, the task is called regression. The objective of supervised learning is to build a function $h : X \to Y$ which can predict outputs Y from input features X and is defined by a set of parameters $\boldsymbol{\theta}$.

Supervised models learn to optimize a loss function $L(\hat{Y}, Y)$ measuring the error between the predictions $\hat{Y} = h(X)$ and the true values Y. A loss function, also known as a cost function or an objective function, provides a measure of the quality of the model's predictions. The choice of an appropriate loss function depends on the specific learning task. Training a model corresponds to optimizing a loss function. A metric is a function used to measure the predictive performance of your model. Metric functions are similar to loss functions, except that the results from evaluating a metric are not used when training the model. Any loss function can be used as a metric.

Among popular classification loss functions are, for example, Zero-One (0/1) Loss, Binary Cross-Entropy (Log Loss), and Categorical Cross-Entropy. Zero-One Loss measures the fraction of incorrect predictions among all instances,

$$\mathsf{Loss}_{01} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{\boldsymbol{y}}_i \neq \boldsymbol{y}_i).$$

Binary cross-entropy is commonly used in binary classification problems. It measures the dissimilarity between the predicted probabilities $L(p(\hat{Y}), Y)$

and the true binary labels,

$$\mathsf{LogLoss} = \frac{1}{N}\sum_{i=1}^{N} -(y_i \log(p(\hat{y}_i)) + (1 - y_i)\log(1 - p(\hat{y}_i))).$$

Binary Cross-Entropy encourages the model to output high probabilities for the correct class and low probabilities for the incorrect class. Categorical Cross-Entropy extends Binary Cross-Entropy to multiple classes. The loss quantifies the divergence between the predicted class probabilities and the true class labels. While these are examples of common loss functions that are used quite extensively, there are many other specialized loss functions designed for specific tasks or domains. The choice of the loss function depends on the learning task, the characteristics of the data, and the desired behavior of the model.

Exact Match and Hamming Score are among popular classification metrics [Tsoumakas and Katakis, 2007]. Exact Match, or simply Accuracy in single-output machine learning tasks, measures the fraction of correctly predicted instances, i.e. instances with *all* labels predicted correctly, and can be presented as $\mathsf{Exact\ Match} = 1 - \mathsf{Loss}_{01}$. Hamming Score counts the fraction of per-output labels predicted correctly,

$$\mathsf{Hamming\ Score} = \frac{1}{N} \cdot \frac{1}{L}\sum_{i=1}^{N}\sum_{j=1}^{L} \mathbb{I}(\hat{y}_{ij} = y_{ij}).$$

In this thesis, we will refer to Exact Match and Hamming Score as *joint accuracy* and *marginal accuracy* respectively.

In regression tasks, Mean Squared Error (MSE) and Mean Absolute Error (MAE) are widely used. MSE computes the average squared difference between the predicted and true values,

$$\mathsf{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i)^2.$$

The squared term gives higher weight to larger errors, making it sensitive to outliers. The goal is to minimize MSE, which leads to estimating the mean or expected value of the target variable. Mean Absolute Error calculates the average absolute difference between the predicted and true values,

$$\mathsf{MAE} = \frac{1}{N}\sum_{i=1}^{N} |\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i| \,.$$

Compared to MSE, it is less influenced by the magnitude of errors. MAE is more robust to outliers but can be less sensitive to subtle differences. Minimizing MAE implies estimating the median of the target variable. Another possible choice is Uniform Cost Function (UCF) [Burger and Lucka, 2014] as an approximation of 0/1 Loss for regression setting,

$$\mathsf{UCF}(\delta) = \frac{1}{N}\sum_{i=1}^{N} \begin{cases} 0 \text{ if } \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_2 < \frac{\delta}{2}, \\ 1 \text{ otherwise,} \end{cases}$$

where δ is an adjustable parameter defining the size of the neighborhood of ground-truth points.

2.2 Multi-output models

In traditional machine learning tasks, such as classification or regression, a single label y is assigned to each data point. However, in multi-output learning, an instance can be associated with multiple labels $\boldsymbol{y} = [y_1, \dots y_L]$ simultaneously. In this setting, an algorithm is trained to predict multiple labels or targets for each instance. This task is called multi-target prediction in [Waegeman et al., 2019]. Multiple outputs may often be correlated with each other, and the use of this information may boost predictive quality.

For example, an email classification system may illustrate a multi-label problem. In a traditional email classification task, the goal is to assign a single label to each email, such as "spam" or "not spam". This is a binary classification problem where each email is either classified as spam or not. In multi-label learning, the email classification task is extended to allow assigning multiple labels. For example, an email can be classified as both "not spam" and "urgent" simultaneously. In this case, the algorithm needs to learn to predict multiple labels for each email.

Another well-known example is image classification where some labels may be correlated and have a higher chance to explain each other, e.g. an object looking similar to a palm tree is likely to be in pair with an object looking like a beach chair but much less likely to be coupled with an office chair. Multi-target regression may be used, for example, in predicting affinity of different drugs to proteins, or gene expression under multiple scenarios. Physical characteristics of a plant such as height, weight, fertility, protein concentration, etc., predicted from characteristics of the surrounding earth may be considered as another example of multi-target prediction.

The main challenge in multi-output learning is dealing with the inherent complexity of the output space and dependencies between the outputs. Each target can be treated as a separate single-output problem, but the presence of multiple targets introduces interactions and correlations between them. The algorithm needs to capture these dependencies and make accurate predictions for each output.

In [Kocev et al., 2013], multi-target prediction is discussed with a focus on ensembles of predictive clustering trees and the presence of specific structures in the target space, e.g. hierarchies between the targets. In [Waegeman et al., 2019], authors present a more recent unifying view on multi-target prediction and discuss similarities and differences between related problems and methods.

As in traditional single-output machine learning, multi-output settings may fall into two subcategories, multi-label classification (where the outputs take categorical values) and multi-target regression (where the outputs take continuous values). An extensive review of the multi-label classification setting is given in [Tsoumakas and Katakis, 2007; Zhang and Zhou, 2014] along with discussion of recent trends and open issues in [Mylonas et al., 2023], and multi-target regression has been recently discussed in [Waegeman et al., 2019].

2.3 Marginal and joint modeling

Independent learning also known as marginal learning, a straightforward approach for multi-output modeling, refers to an approach where each task or variable is learned independently, without explicitly considering the dependencies or relationships with other tasks. In the classification context, this approach is known as the *binary relevance* method [Tsoumakas and Katakis, 2007; Godbole and Sarawagi, 2004]. Given an instance $\boldsymbol{x}$, the prediction is obtained as

$$\hat{\boldsymbol{y}} = [\hat{y}_1, ..., \hat{y}_L] = [h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), ..., h_L(\boldsymbol{x})].$$

This means that L models are trained to maximize the marginal probabilities $p(y_j \,|\, \boldsymbol{x})$ separately for each target y_j, $j = 1, ..., L$. An important advantage of this approach is flexibility as each model h_j can be optimized separately, allowing for different algorithms or hyperparameters to be used for each model, based on specific characteristics of each task. However, the interdependencies between the targets are completely ignored which can lead to suboptimal performance or prediction of impossible combinations [Park and Fürnkranz, 2008; Elisseeff and Weston, 2001; Godbole and Sarawagi, 2004].

Joint modeling refers to a learning approach where multiple tasks or variables are learned simultaneously. In joint learning, a model considers all tasks or variables together, capturing the dependencies and interactions between them and maximizing the joint probability $p(\boldsymbol{y} \mid \boldsymbol{x})$. The goal is to leverage the shared information across the tasks to improve overall performance. Such an approach allows exploiting shared structure as joint learning can leverage the relationships and dependencies between tasks to improve predictive accuracy. So far, there is a consensus among the researchers from the domain that output interdependencies have to be incorporated into the modeling [Luaces et al., 2012; Tsoumakas and Katakis, 2007; Guo and Gu, 2011; Alvares-Cherman et al., 2012].

2.4 Maximum Likelihood Estimation

One of the common approaches for estimating the joint probability of a dataset is Maximum Likelihood Estimation (MLE). This strategy treats the problem as an optimization problem, where a set of parameters that results in the best fit for the joint probability of the data sample is estimated. In a machine learning setting, this corresponds to maximizing the probability of observing the targets $\boldsymbol{y}$ from the joint probability distribution $P(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta})$ given unknown parameters $\boldsymbol{\theta}$ of a classfication or regression model.

The Expectation-Maximization (EM) algorithm is an approach for Maximum Likelihood Estimation in the presence of latent variables (or, in particular, missing values or labels), that is to say, that not all variables related to the problem are observed [Bishop, 2006]. The EM algorithm is an iterative approach consisting of two repeating steps. In the first estimation step (E-step) the algorithm attempts to estimate the missing or latent variables. The second maximization step (M-step) tries to optimize the parameters of the model to best explain the data. The EM algorithm has a wide

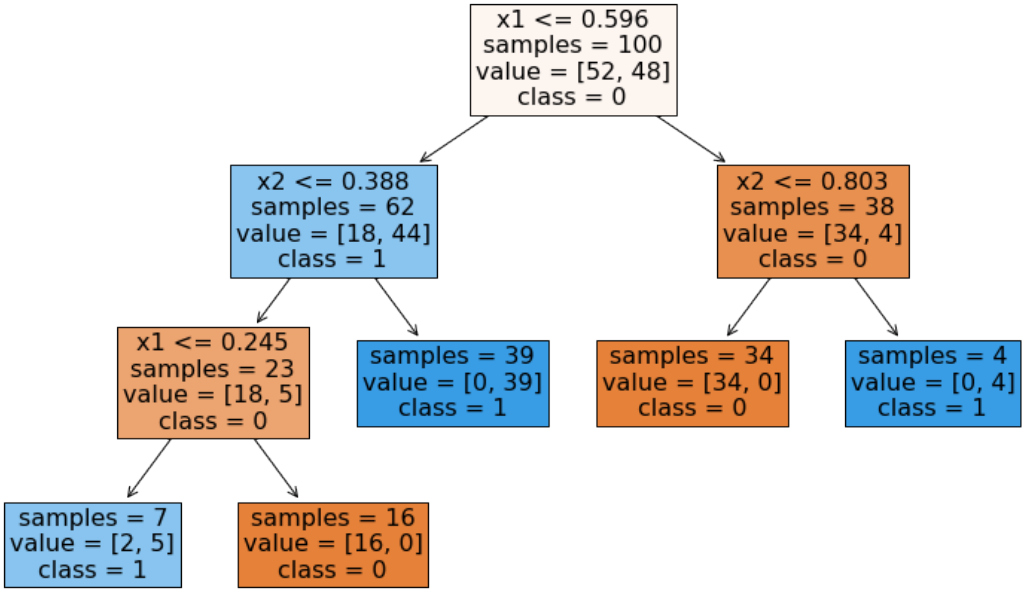
range of applications, although is perhaps most well-known for its use in unsupervised learning problems, such as density estimation and clustering.

2.5 Tree-based methods

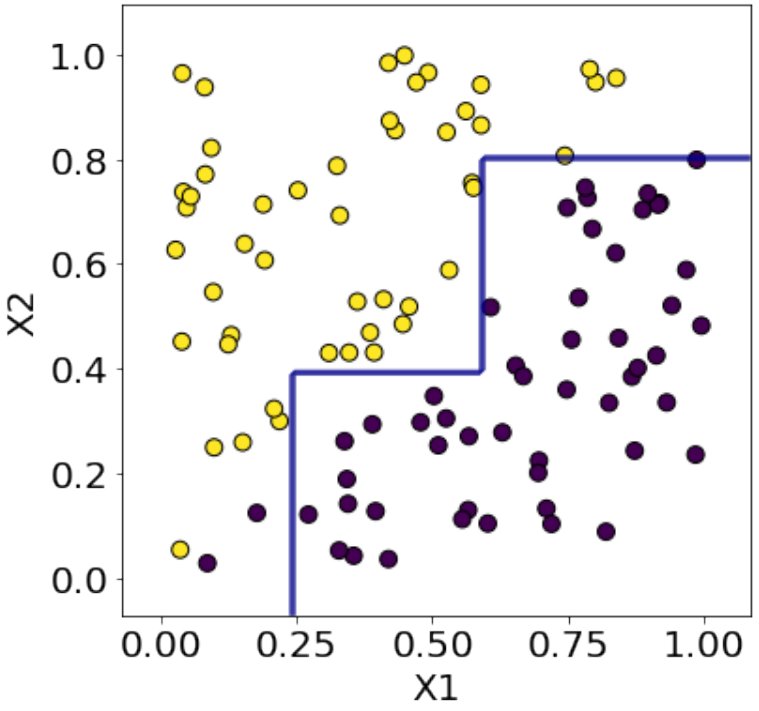
Tree-based models are among the most popular machine learning approaches which can be used both for regression and classification. In a multi-output setting, tree-based methods can be included into the *algorithm adaptation* family of methods. Such approaches address multi-output problems directly by adapting some existing learning algorithms to a multi-output scenario.

A Decision Tree (DT) [Breiman et al., 2017], an underlying structure in this methods family, is a tree graph structure and consists of nodes connected by directed edges. Every node may have outgoing edges connecting it to its children. The final nodes with no output edges are called leaves. The top node which has only outgoing edges (and no ingoing ones) is called the root. Fig. 2.1 illustrates a decision tree and its decision boundaries in a single-output setting.

Predictive Clustering Trees [Blockeel et al., 2000; Kocev et al., 2013] are a well-established generalization of Decision Trees, where each node represents a data cluster and each edge corresponds to a decision rule. Starting from the root which contains all data points, all nodes are split recursively by applying a decision rule to one of the features. The task of a machine learning algorithm is to identify the optimal split using the split quality criterion. The tree-growing process is stopped when the stopping criterion is reached, and each terminal node, i.e. leaf, is associated with an output value. In the prediction phase, each incoming instance traverses the tree from the root to a leaf and is assigned the output value of the leaf. The PCT method has been also extended to predict multiple output variables in [Kocev et al., 2013]. In multi-output prediction, each leaf of a tree is assigned a combination of output values.



(A)



(B)

FIGURE 2.1: Single-output decision tree (A) and its decision boundaries (B) on a dataset with two variables and one binary label.

Random Forest (RF) [Breiman, 2001] is an ensemble of multiple decision trees. To introduce diversity in the learning process, each tree is built on a random subset of features and bootstrap replicates of the training instances. The final prediction is a major vote for all tree predictions in the classification scenario or their mean average in the regression case. This holds equally for single- and multi-output Decision Trees [Kocev et al., 2013].

Random Forests are often preferred over single Decision Trees as they typically show better performance. While building multiple trees instead of a single one may seem more computationally expensive, taking a random subsample of features per tree alleviates this drawback. Also, the building process is straightforward to parallelize as trees are built independently. Though Decision Trees are considered easier to interpret as a consequence of decision rules, Random Forests may still provide feature importances computed as a total reduction of the criterion brought by each feature. Fig. 2.2 illustrates a Random Forest[1].

In general, multi-output tree-based methods may be considered as algorithm adaption approaches as they adapt to handle multi-dimensional outputs. Both multi-output Decision Trees and multi-output Random Forests typically outperform their single-target versions applied to all features separately one by one. Also, they are a frequent choice in problem transformation approaches described in more detail in the next section.

2.6 Problem transformation approaches

Another family of multi-output methods is *problem transformation* methods that transform a multi-target problem into multiple single-target problems.

[1]Created with BioRender.com

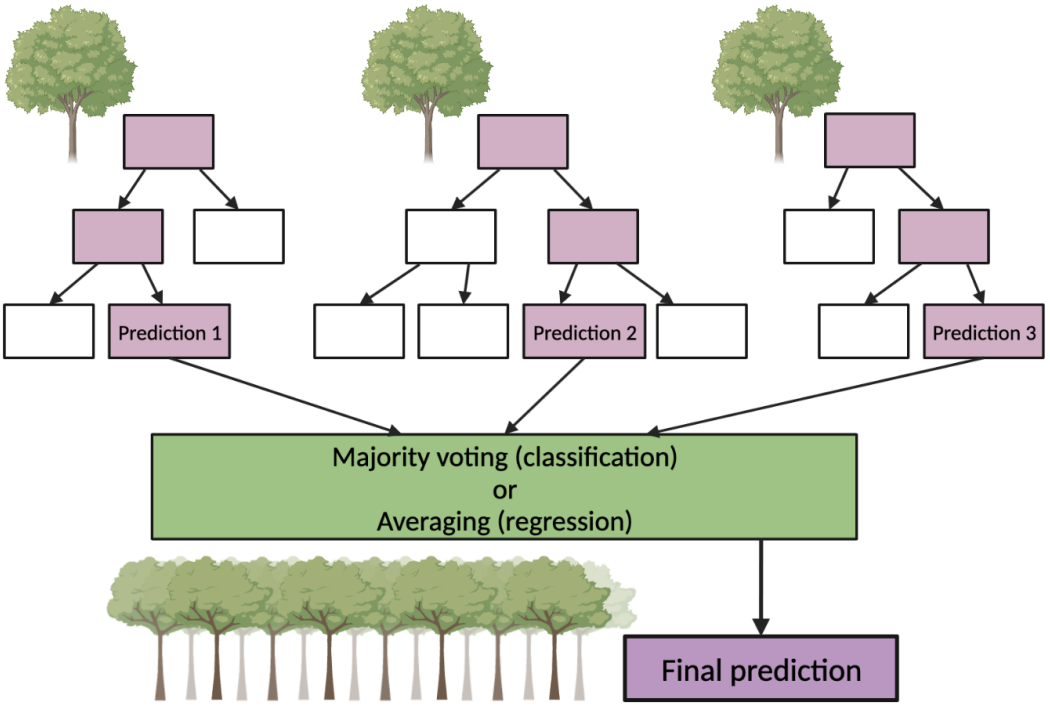


FIGURE 2.2: Illustration of a Random Forest as an ensemble of trees.

For example, the Binary Relevance (BR) [Tsoumakas and Katakis, 2007; Godbole and Sarawagi, 2004] approach straightforwardly transforms a multi-output problem into a set of independent single-output problems while completely ignoring correlations and interdependencies between the targets.

The Label Powerset (LP) [Tsoumakas and Katakis, 2007] method considers each combination of output values into a unique class, and thus can be exploited in a classification setting but not in a regression problem. A popular Random k-Labelsets (RAkEL) [Tsoumakas and Vlahavas, 2007] approach generalizes the LP method by considering a small random subset of labels and learning a single-label classifier for the prediction of each element in the powerset of this subset.

Another well-known representative of the problem transformation family of methods is a chaining approach implemented, e.g. in Regressor and Classifier Chains. The initial idea of the chaining approach, for classification [Read et al., 2011], was to arrange per-target models in a chain, such that the previous labels are used to train each next model in the training phase and the output prediction of one model becomes an additional feature for the subsequent models in the prediction phase. That is, given an instance $\boldsymbol{x}$, we obtain a prediction as

$$\hat{\boldsymbol{y}} = [\hat{y}_1, ..., \hat{y}_L] = [h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, \hat{y}_1), ..., h_L(\boldsymbol{x}, \hat{y}_1, ..., \hat{y}_{L-1})].$$

Each estimator h_j in the chain takes $[x_1, \ldots, x_p, y_1, \ldots, y_{j-1}]$ as feature space and is trained to predict $\hat{y}_j$. In the testing phase, the algorithm begins with y_1 and propagates predictions $\hat{\boldsymbol{y}}$ along the chain, on each step augmenting the feature space by the predictions of the previous estimators.

As opposed to independent modeling (binary relevance in classification, Fig. 2.3a), the chaining approach allows the model to capture the dependencies and interactions between the target variables. An important characteristic of Regressor Chains is that they can utilize various base regression

models including non-differentiable ones, e.g. linear regression, tree models, or support vector machines, for each target variable in the chain. The choice of the base model depends on the characteristics of the problem and the desired performance. The main advantage of Regressor Chains is their ability to benefit from the interdependencies between target variables by leveraging the predictions of the previous models. It allows the models to exploit the relationships among the targets, potentially leading to improved predictive performance.

The order of the chain clearly has an impact on the model's ability to learn interdependencies between the targets and thus predictive performance. Different approaches have been suggested to optimize chain order including evolutionary algorithms [Moyano et al., 2017] and using correlation to build the best structure [Melki et al., 2017]. Another way to obtain better results for Classifier and Regressor Chains is using Ensembles of Classifier or Regressor Chains (ECC and ERC) with random chain orders [Read et al., 2011; Spyromitros-Xioufis et al., 2016].

Classifier Chains have proved to have high predictive performance and are widely known as one of the state-of-the-art techniques for multi-label modeling [Dembczyński et al., 2012; Read et al., 2021]. Although they seem naturally extendable to a regression setting, Regressor Chains are less robust and may be more sensitive to, e.g., suboptimal chain order or errors propagating along the chain [Read and Martino, 2020]. However, Regressor Chains are still a popular model choice and have been successfully applied to a number of problems, e.g. [Wu and Lian, 2020; Poonawala-Lohani et al., 2021; D'hondt et al., 2023].

The original Regressor and Classifier Chains definition [Read et al., 2011] referred to a fully-connected cascade, Fig. 2.3b; nevertheless, this view may be extended to the chains with any Directed Acyclic Graph (DAG) order [Read et al., 2021], for example, Markov chains (Fig. 2.3c), trees (Fig. 2.3d and [Ramírez-Corona et al., 2014]), arbitrary DAGs (Fig. 2.3e

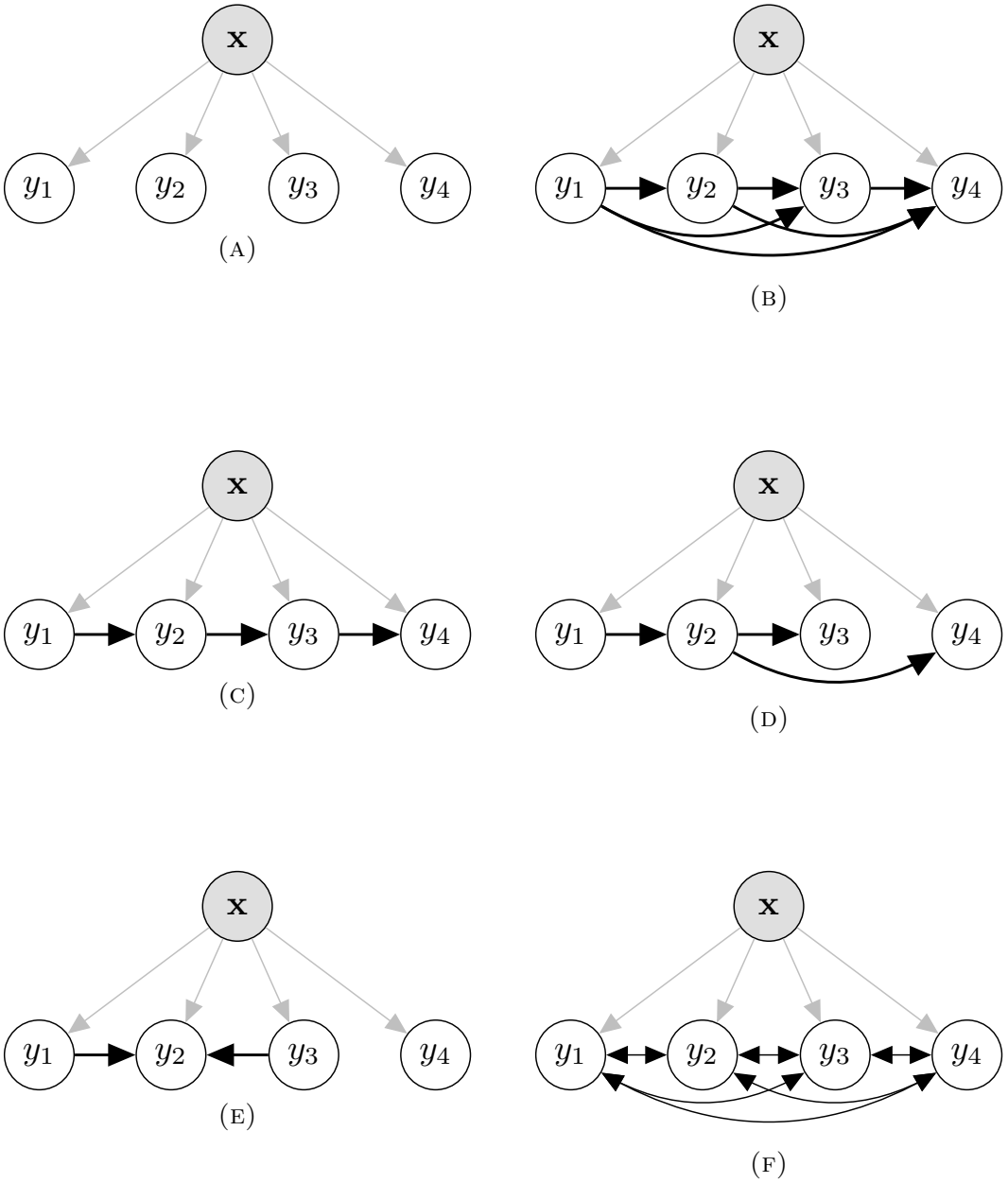


FIGURE 2.3: Different chain structures for a problem with $L = 4$ targets.

and [Zhang and Zhang, 2010]). The chaining approach may be also adapted to an undirected and cyclic framework, i.e. beyond the aforementioned DAG formulation. For example, [Guo and Gu, 2011] proposed a fully-connected bi-directional classifier graph, a network shown in Fig. 2.3f. The advantage of such a model is the full connectedness: a prediction for any y_j can influence a prediction for any y_k and vice versa. The inference phase of this method is inherently more expensive compared to a single greedy pass used by Classifier Chains because many iterations through the graph, or cycles, are needed to arrive at the convergence of an estimate. In [Read and Martino, 2020], Regressor Chains were further developed into a probabilistic framework.

A Classifier or Regressor Chain may be also seen as a particular implementation of a Bayesian Network, specifically a hybrid Bayesian network [Salmerón et al., 2018] in the regression case. It is worth noting that in classic Bayesian Networks training is extremely costly and inference options are limited, normally corresponding to linear-Gaussian models or approximate methodologies based on sampling, and variational inference.

An alternative non-chaining approach in regression is Regressor Stacking [Spyromitros-Xioufis et al., 2016; Santana et al., 2017] which includes predictions of single-target regressors as new features for the next rounds of training. The Multi-Target Regressor Stacking [Spyromitros-Xioufis et al., 2016] method consists of separately training single-target models for each output and using their prediction as additional features for the second round of training. Considering a dataset composed by $X = \{x_1, x_2, ..., x_p\}$ input features and $Y = \{y_1, y_2, ..., y_L\}$ target outputs, this approach uses the single-target predictions $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_L\}$ as new features, forming a new training dataset $X' = \{x_1, x_2, ..., x_p, \hat{y}_1, \hat{y}_2, ..., \hat{y}_L\}$. The transformed input is used to train the second regressors' layer of L single-target models, whose outputs are the final predictions. In Deep Regressor Stacking [Santana et al., 2017] a similar idea is used, but several layers of re-prediction

are performed. The authors show that the predictive error may lower with a rise of number of layers.

2.7 Missing value imputation

Missing values are abundant and remain a very important issue in real-world data in all domains. They refer to the absence of data for a particular feature and instance in a dataset. Further, we denote by $\tilde{X}_{i,j}$ a random variable that corresponds to a missing value in the i-th instance and j-th feature and has to be estimated, i.e. imputed, and by $\tilde{X} = \{\tilde{X}_{i,j}\}$ the set of all missing value in the data.

Missing values can occur for various reasons, such as data entry errors, data loss during collection or storage, non-response in surveys, and many others. Handling missing values is important as their mistreatment may bias predictions, affect statistical analyses negatively, and impact the performance of machine learning models.

Missing values may be traditionally classified into three types with regard to the pattern of missingness. These are Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR) [Santos et al., 2019]. When values are Missing Completely at Random, the missing values occur randomly, and their presence or absence does not depend on observed or unobserved data. In the Missing at Random case, the missingness is related to observed variables but not to the missing values themselves. When values are Missing Not at Random, the probability of missingness depends on the values that are missing.

Missing values can lead to biased estimates, especially if the missingness is related to the variable being measured, i.e. values are missing not at random. In particular, ignoring missing values can reduce the sample size and subsequently decrease the statistical power of the analysis, making it

difficult to detect significant effects or relationships. Additionally, missing values can distort the relationships between variables, leading to inaccurate interpretations and conclusions.

For example, let us consider a study examining the effects of a new teaching method on students' academic performance. The researchers collect data on students' test scores before and after implementing the new teaching method. However, they encounter a missing values problem for the post-method scores, as some students were absent on the day of the test or failed to complete the test for various reasons. To address this issue, the researchers exclude the students with missing data from their analysis and only analyze the data from those students who provided complete information on both pre-method and post-method scores. After analyzing the available data, they find a significant improvement in the average test scores. Based on this result, they infer that the new teaching method is effective in enhancing students' academic performance. However, this conclusion may be biased due to the exclusion of students with missing post-method scores. For instance, the absent students might have had lower motivation or struggled more academically, leading to a potential underestimation of the effects of the teaching method. By excluding students with missing post-method scores, the researchers unintentionally introduce bias into their analysis. The excluded students may have different outcomes compared to those included, and this can impact the observed improvement in test scores.

Some machine learning algorithms can handle missing values inherently (e.g. variations of decision trees) and can work with datasets containing missing values without additional preprocessing. However, most off-the-shelf machine learning methods are not able to do this. This increases the necessity for imputation methods, i.e. filling in missing values with estimated values based on observed data. Various imputation techniques aim

at replacing missing values with plausible values, maintaining the integrity of the dataset.

One of the simplest though common approaches is imputation with some statistics, e.g. mean, mode, or median of the observed values [Little and Rubin, 2019]. Though simple and fast, this approach does not consider any relationships between variables and may not accurately represent the true values.

Another popular approach is hot deck imputation which replaces missing values with values from similar instances in the dataset. It involves finding the nearest neighbors based on a similarity measure and imputing the missing values with values from those neighbors. For example, this approach is implemented in [Schwender, 2012] by using the k-Nearest Neighbors algorithm. Hot deck imputation is better adapted to preserve the relationships between variables.

A more complex approach is building a prediction model for each variable using the complete or randomly pre-imputed cases and using that model to predict the missing values [Montiel et al., 2018; van Buuren and Groothuis-Oudshoorn, 2011; Stekhoven and Bühlmann, 2011]. For each target variable, a model utilizes all other variables or a subset of them as predictors to estimate the missing values. In [Montiel et al., 2018] the available complete values of other features are used to train a model, while [van Buuren and Groothuis-Oudshoorn, 2011; Stekhoven and Bühlmann, 2011] first fill the missing values randomly or with some statistics, and then iteratively update the values until a convergence criterion is met. This family of methods may remind the binary relevance method from multi-output prediction as an independent estimator is built for each feature.

Finally, advanced machine learning algorithms, such as deep learning and in particular Denoising Autoencoders [Vincent et al., 2008], can be used for missing value imputation. These algorithms learn patterns and relationships from the available data and use that knowledge to predict missing

values. These techniques can handle complex data structures and capture non-linear relationships, but they may be computationally expensive and require careful tuning. Autoencoders are a type of neural network designed to learn efficient representations or compressed versions of input data. Autoencoders consist of an encoder network that maps the input data to a typically lower-dimensional latent space, and a decoder network that reconstructs the original input data from the latent representation. The encoder may consist of one or more hidden layers, which gradually reduce the dimensionality of the input data. A latent space, or encoding, is a compact representation of the input data learned by the encoder. It captures the most important features or patterns in the data. The decoder takes the encoded representation from the latent space and reconstructs the original input data. Similar to the encoder, the decoder consists of one or more hidden layers that gradually increase the dimensionality of the data to match the original input dimensions. The performance of an autoencoder is evaluated based on the loss function of the original input data and its reconstruction from the latent representation. Autoencoders are widely used to denoise corrupted or noisy input data as well as impute missing data. By reconstructing the original clean data from inputs presenting missing values, they effectively learn to replace these gaps with reasonable values.

While missing values are widely seen in real-world datasets in all application domains and are typically imputed in a pre-processing step before further data analysis, some studies show that it might be beneficial to impute missing values in the feature space and model targets simultaneously [Le Morvan et al., 2021; Perez-Lebel et al., 2022].

On the other hand, in particular problems we may observe incomplete data in the target space when values of some of the targets are observed while others are missing [Beigaitė et al., 2022]. In this case, instead of just predicting the unobserved targets it is beneficial to include information from the observed values to explore possible correlations between the targets

with observed and unobserved values. As a motivating example of such problem, we study possible vegetation in the absence of human activity. In this setting, we may observe only data when urban types are present and aim to predict shift of vegetation distribution when urban types are 'provided', i.e. set to zero.

2.8 Single Nucleotide Polymorphisms

Single Nucleotide Polymorphisms (SNPs) represent genetic variation among individuals given by single nucleotide differences at specific positions in the DNA sequence (Fig. 2.4). For example, at a specific position, one person might have an adenine (A), while another person might have a guanine (G). These single nucleotide differences can be used as genetic markers to track genetic variation across populations. Single Nucleotide Polymorphisms genotyping typically involves using high-throughput genotyping technologies to determine the genotype (i.e. variant of the SNP) at specific SNP positions for each individual. This information is then used to assess the association between SNP genotypes and the trait of interest, widely known as Genome-Wide Association Study (GWAS) [Manolio, 2010; Uffelmann et al., 2021]. It involves examining the entire genome of individuals to detect association of particular SNPs with one or several traits of interest. GWAS studies have contributed to significant advancements in our understanding of the genetic basis of complex traits and diseases. They have identified numerous SNPs associated with various traits and provided insights into the underlying biology. These findings can have implications for personalized medicine, risk prediction, and the development of targeted therapies.

As happens frequently in real-world data, SNP datasets are prone to the presence of missing values. These can arise due to various reasons, including technical limitations in genotyping platforms, sample quality issues,

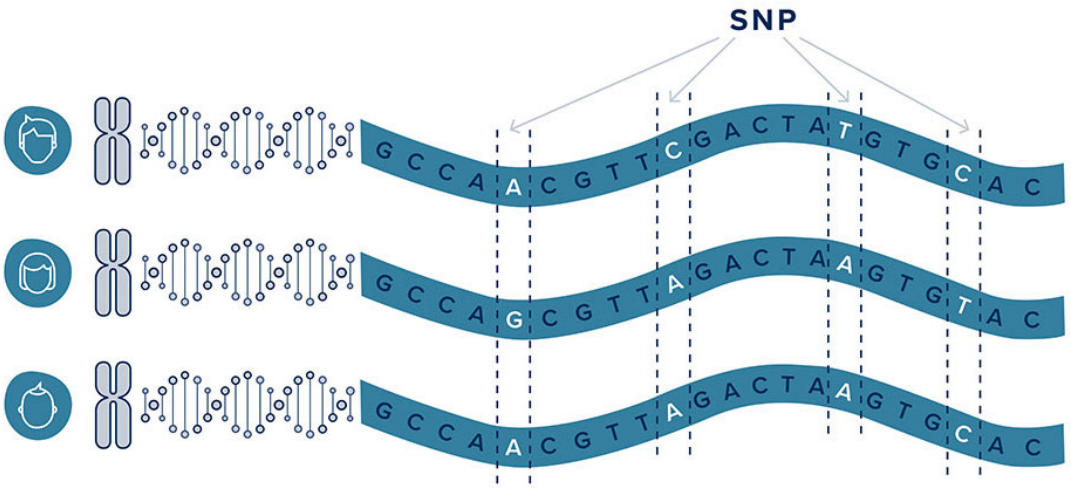


FIGURE 2.4: Single Nucleotide Polymorphisms. Copyright: Scientific DX GmbH, 2020

biological reasons such as genetic variation and deviations from Hardy-Weinberg dysequilibrium, or combining different datasets with unequal sets of features in meta-studies [Das et al., 2018]. Imputation methods are usually split into two subgroups, reference-based and reference-free methods. Reference-based approaches require a reference panel of large size and high quality and remain state-of-the-art methods in human genome studies [Das et al., 2018]. However, in less-studied species such as most animals and plants, these reference panels are not available, and thus a need for methods based only on the data available arises [Davies et al., 2016]. These are called reference-free and remind us about traditional missing value imputation techniques. However, a very important challenge in SNP data refers to their curse of dimensionality. The SNP datasets are typically high-dimensional ($10^5 - 10^6$ features) and low-sampled ($10^2 - 10^3$ instances) and these characteristics raise issues for many machine learning methods [Johnstone and Titterington, 2009]. In this thesis, we propose a method that handles effectively missing values in SNP data and typically outperforms other baseline methods.

2.9 Notation

We use the following notation in this thesis:

- X is an input set of p-dimensional feature vectors;
- $\boldsymbol{x} \in X$ is an instance, described by a feature vector $\boldsymbol{x} = [x_1, \ldots, x_p]$;
- Y is an output set of L-dimensional target vectors;
- Each instance $\boldsymbol{x} \in X$ is associated with an output vector $\boldsymbol{y} = [y_1, \ldots, y_L]$, $\boldsymbol{y} \in Y$;
- $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}_{i=1}^{N}$ is a dataset of N samples;

- $h : X \to Y$ is a predictive model (regressor or classifier);
- $\boldsymbol{\theta}$ are parameters of the model h;
- $\hat{\boldsymbol{y}} = h(\boldsymbol{x})$ is a prediction of multi-output model h for instance $\boldsymbol{x}$, $\hat{\boldsymbol{y}} = [\hat{y}_1, \ldots, \hat{y}_L]$;
- $L(\hat{Y}, Y)$ is a loss function measuring error between the predictions $h(X)$ and the true values Y;
- $p(\boldsymbol{y} \mid \boldsymbol{x})$ is a conditional probability of the output $\boldsymbol{y}$ given the instance $\boldsymbol{x}$;
- $\tilde{X}_{i,j}$ is a random variable corresponding to a missing value in the i-th instance and j-th feature;
- $\tilde{X}$ is a set of random variables $\{\tilde{X}_{i,j}\}$, $i = 1, \ldots, N$, $j = 1, \ldots, p$.

Chapter 3

Multi-Modal Ensembles of Regressor Chains

Classifier Chains are widely known as a technique that successfully models the outputs together in the domain of multi-label classification. Although this approach should be naturally extendable to the multi-target regression task (as Regressor Chains) and seems to be straightforward to adapt to the regression setting, large improvements over independent models (as seen already in the multi-label classification context over the recent decade) have not as of yet been obtained from Regressor Chains. One of the reasons for the unsatisfying performance of Regressor Chains is the adoption of squared-error-based loss metrics which do not require consideration of joint-target modeling. In this chapter, we consider cases where the predictive distribution can be multi-modal. Such a scenario, which easily manifests in real-world tasks involving uncertainty, motivates a different loss metric and, thereby, a different approach. We thus present a new method for multi-target regression: Multi-Modal Ensemble of Regressor Chains (mmERC),

which performs competitively on datasets exhibiting a multi-modal distribution, both against independent regressors and state-of-the-art Ensembles of Regressor Chains. We argue that such distributions are not sufficiently considered in the regression and particularly multi-target regression literature.

3.1 Introduction

Multi-target prediction algorithms can be a solution to the nowadays extensively growing number of multi-output data science problems across academy and industry areas [Waegeman et al., 2019; Xu et al., 2019]. Multi-label classification, which refers to the multi-output case with binary variables, has made significant progress in the previous decade. Within this area, Classifier Chains is a family of methods that have proved to have high predictive performance [Dembczyński et al., 2012; Read et al., 2021]. Compared to the naive approach with an independent classifier per label (known in the literature as binary relevance), advanced methods such as Classifier Chains outperform with regard to most metrics. This has been widely attributed to their ability to extract and exploit the dependencies between the targets, as well as other factors linked to multi-target modeling [Read et al., 2021; Waegeman et al., 2019].

Chaining methods can be adapted in a straightforward way to the regression context, known as Regressor Chains. Alongside many multi-label methods, Classifier Chains are known to perform invariably better than independent classifiers under empirical study [Madjarov et al., 2012; Bogatinovski et al., 2022]. However, the performance of Regressor Chains shows relatively few advantages compared to individual regression models.

There has been recent work attempting to unravel some of the explanations for Regressor Chains' underperforming [Read and Martino, 2020]. It has been identified that Classifier Chains perform well with respect to the 0/1

Loss, i.e. modeling the labels jointly and, in the probabilistic sense, seeking out a posterior *mode*. However, in the case of Regressor Chains, an almost-ubiquitous choice of loss metric is the Mean Squared Error (MSE) or its variants; as also for regular regression problems. By definition, minimizing MSE is the same as maximizing the likelihood of a Gaussian distribution; it will thus correspondingly incur a posterior *mean-seeking* behavior. This may be inadequate if the posterior is bi-modal or more generally multi-modal; a model which optimizes MSE may place the prediction between two modes of a hypothetical posterior – a place that will not correspond to the ground truth and maybe is not even observed in data at all. This situation is illustrated in Fig. 3.1 for a bi-modal distribution in a single-target setting. While $p(y_1)$ is bi-modal and a mean is visibly distinguishable from two modes, a Random Forest aims to model a uni-modal Gaussian-like distribution $p(\hat{y}_1 \mid x)$ especially when a feature x is not highly informative. Minimizing the Mean Absolute Error (MAE) is similar as it assumes a [uni-modal] Laplacian rather than a Gaussian [Qi et al., 2020].

There are plentiful real-world examples of multi-modal outputs; these include, e.g. cases from agriculture [Vasconcelos et al., 2021], evolution biology [Hendry et al., 2008], and gene expression [Paliwal et al., 2007]. For instance, [Hendry et al., 2008] considers a finch (*Geospiza fortis*) population that shows bi-modality in beak size, an important trait in this taxon, while [Paliwal et al., 2007] studies bi-modality in gene expression for certain pheromones, which allows a cell population to diversify its transcriptional response. One more famous example of multi-modal data is described in [Pearson, 1894], where normal mixture model analysis of the ratios of forehead breadth to body length for 1000 crabs sampled at Naples reveals the presence of two distinct crab species. In such cases, an estimate under MSE and under uncertainty can be inappropriate.

Naturally, this discussion of multi-modality relates to regression tasks in general, but it becomes particularly crucial in many multi-target regression

problems due to the effect of error propagation [Read and Martino, 2020] and the potential presence and complexity of modes.

This chapter introduces a novel method, Multi-Modal Ensemble of Regressor Chains (mmERC), which combines an ensemble approach for Regressor Chains [Spyromitros-Xioufis et al., 2016] and a novel mechanism designed to recognize the multi-modality and to produce the predictions taking it into account. We argue that multi-modal scenarios are not widely studied in machine learning research (as opposed to statistics) while taking them into account can significantly boost the power of machine learning methods. Our experimental results show an improvement in the performance of Regressor Chains with the novel technique. In particular, we show that mmERC can outperform independent regressors.

The rest of the chapter is organized as follows. After summarizing the background and related work in Section 3.2, we present our method in Section 3.3. We describe our implementation and the setup for comparison to independent regressors and standard Regressor Chains in Section 3.4. The results and their discussion are in Section 3.5. In Section 3.6, we draw the conclusions.

3.2 Background and Related Work

Following notation from Section 2, we are given a dataset $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}_{i=1}^{N}$ of N samples, each instance $\boldsymbol{x} = [x_1, ..., x_p]$ is associated with a vector $\boldsymbol{y} = [y_1, ..., y_L]$ of real numbers. Opposite to a straightforward binary relevance approach [Tsoumakas and Katakis, 2007; Godbole and Sarawagi, 2004], Fig. 2.3a, where

$$\hat{\boldsymbol{y}} = [\hat{y}_1, ..., \hat{y}_L] = [h_1(\boldsymbol{x}), ..., h_L(\boldsymbol{x})],$$

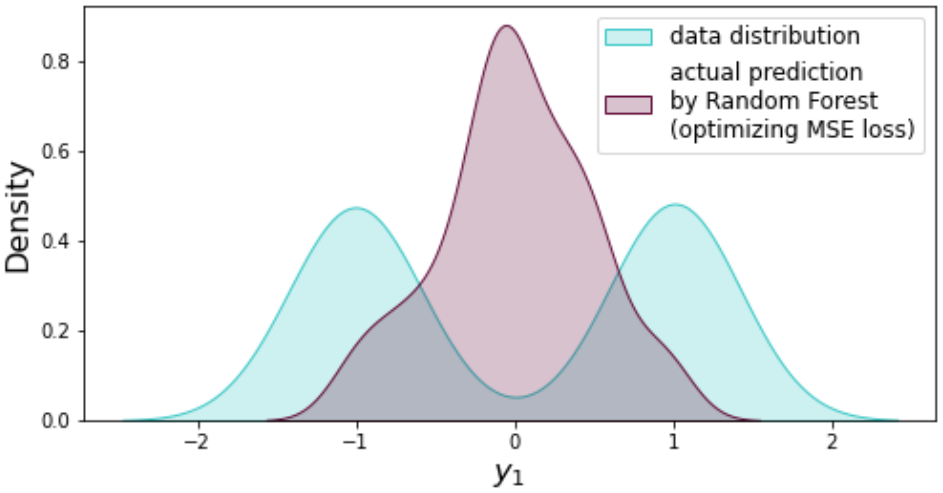


FIGURE 3.1: A ground-truth distribution $p(y_1)$ vs a distribution of predictions by Random Forest $p(\hat{y}_1 \,|\, x)$; both provided via a KDE estimate. Most predictions – when provided under uncertainty (input x is poorly informative here) – are in the space highly likely to be incorrect. The model aimed to minimize MSE (Random Forest here) puts predictions of most instances close to zero, between two modes of the real posterior distribution.

the method of Classifier Chains [Read et al., 2011] arranges per-target (base) models in a chain, such that the prediction of one model becomes an additional feature for the subsequent models. That is, for an instance $\boldsymbol{x}$, we obtain a prediction as

$$\hat{\boldsymbol{y}} = [\hat{y}_1, ..., \hat{y}_L] = [h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, \hat{y}_1), ..., h_L(\boldsymbol{x}, \hat{y}_1, ..., \hat{y}_{L-1})].$$

This approach is demonstrated in Fig. 2.3b. It is observed that the performance of Regressor Chains can suffer from sensitivity to the chain order. Different approaches have been suggested to optimize chain order including evolutionary algorithms [Moyano et al., 2017] and using correlation to build the best structure [Melki et al., 2017]. One of the state-of-the-art solutions to overcome this issue is using an Ensemble of Regressor Chains (ERC) [Spyromitros-Xioufis et al., 2016], where n random chains are trained independently. Then the final predictions are obtained as the means of the n estimates for each target. The same mechanism is used, for example, in Random Forests [Ho, 1995; Breiman, 2001], that output the average mean of a number of Decision Trees. However, we observe that while Ensembles of Regressor Chains work on average better than standard Regressor Chains, they may produce inadequate results in the case of multi-modal distributions, and the improvement is not as significant as in the classification scenario. This brings our interest to multi-modal regression.

By taking a squared-error loss metric such as MSE, conventional regression models predict their estimated mean of the distribution. This approach may produce inadequate results if the data distribution is bi-modal or multi-modal (recall the example in Fig. 3.1) or whenever the mode is not close to the mean. Modal regression (e.g. [Yao and Li, 2014]) is to model a mode of distribution. The advantages of this approach are that modal regression is more likely to capture a mode; which corresponds to values that are – in those settings – more likely to occur in practice. Multi-modal regression has been approached previously due to its properties of robustness to outliers and heavy tail distributions [Feng et al., 2020]. In [Read and Martino,

2020] a probabilistic approach allows to explicitly model the distribution and take samples to find an approximation of the mode.

The questions related to understanding and modelling multi-modal distributions stand close to predicting multiple hypotheses, e.g. in image classification or future step prediction [Rupprecht et al., 2017], where a framework reformulating existing single-prediction models as multiple hypotheses prediction models is proposed. By optimizing a new meta loss, the proposed solution outperforms a single-hypothesis approach where averaging over hypotheses, or mean, is used.

Mode estimation has been studied in the Bayesian statistics literature and, in particular, maximum a posteriori probability (MAP) estimation [Burger and Lucka, 2014; Bassett and Deride, 2018]. These methods suggest, in particular, optimizing the Uniform Cost Function,

$$\mathsf{UCF}(\delta) = \frac{1}{N}\sum_{i=1}^{N}\begin{cases}0 \text{ if } \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_2 < \frac{\delta}{2},\\ 1 \text{ otherwise,}\end{cases} \tag{3.1}$$

as an approximation of 0/1 Loss within δ-neighborhood of ground-truth points. We recall that Classifier Chains are a natural choice if the 0/1 Loss is to be used, yet this metric cannot be directly optimized in the regression context where an exact match is unlikely to be obtained on the continuous spectrum.

Another approach to a regression problem may be to discretize continuous output space into bins and thus to adapt the problem for a classification algorithm [Dougherty et al., 1995; Phan-Minh et al., 2020; Spyromitros-Xioufis et al., 2020]. In [Phan-Minh et al., 2020], this is done for predicting trajectories of a self-driving engine. In [Spyromitros-Xioufis et al., 2020], a framework for solving multi-target regression problems via output space quantization is proposed. Though such an approach allows joint modelling by well-performing classification methods, the outputs continuity is lost,

while we should solve a computationally difficult multi-class multi-label problem with big number of classes per label. Also, finding a good discretization may be itself a complex task [Sokolovska et al., 2018].

The similarities and differences of regression and classification metrics, as well as regimes when they are better to be used, are discussed, for example, in [Muthukumar et al., 2021] for overparameterized models. In any case, these mentioned works do not consider multi-target regression settings. Multi-modality was considered in the context of multi-target regression in [Read and Martino, 2020], but specifically to probabilistic models, therefore their study could not include methods such as tree-based methods; and results were not strong. In our experiments, Decision Trees and Random Forests show competitive performance both as independent methods and as base models for Regressor Chains.

In this case, we suggest the UCF as a useful alternative for comparing model performance. In the following Section 3.3, we present our novel approach to minimize this loss, Multi-Modal Ensembles of Regressor Chains (mmERC) that adapt the ERC method to datasets with multi-modal distribution and do not require an explicit probabilistic analysis, allowing the application of more diverse base classifiers regressors such as decision trees.

3.3 Multi-Modal Ensembles of Regressor Chains

We present our novel method, Multi-Modal Ensemble of Regressor Chains (mmERC), which aims at providing successful outputs in the context of multi-modal distributions. The new approach is based on Ensembles of Regressor Chains while targeting the Uniform Cost Function as a loss function. However, as Regressor Chain-based methods bear a significant advantage of being very flexible with regard to a choice of per-target base estimators, we

want to maintain this flexibility and thus do not target the loss function of base estimators directly. Instead of this, we simulate minimizing the UCF function by two mechanisms that can be used with any base estimator.

3.3.1 Mechanism 1: one base estimator training

Using UCF as a loss function promotes mode-seeking by entailing a uniform penalty when the correct mode is not found; unlike MSE which entails a quadratic penalty. We select correntropy [Feng et al., 2015],

$$\mathsf{corr}(y_{ij}, \hat{y}_{ij}) = 1 - e^{-(y_{ij}-\hat{y}_{ij})^2}, \quad i = 1, \ldots, N, \quad j = 1, \ldots, L, \tag{3.2}$$

as a smooth approximation of UCF, allowing fine-grained threshold selection. The MSE, UCF, and correntropy metrics are compared in Fig. 3.2. The UCF and correntropy errors significantly increase when the prediction does not fall to a small neighborhood of the truth point but stays nearly constant when the prediction is far from this neighborhood.

In the vein of Regressor Chains, we train one target y_j (corresponding to one base estimator) at a time, $j = 1, \ldots, L$. Initially, we train the first regressor on the entire dataset $\mathcal{D}$. After that, we measure the performance of the predictions of the trained model under correntropy. We select a subset of instances $\{x_i\}$ of $\mathcal{D}$ of size $s \cdot N$, $0 < s < 1$, with the lowest correntropy $\mathrm{corr}(y_{ij} - \hat{y}_{ij})$ and train the second regressor on this reduced dataset. By using this mechanism, we aim at cutting off the instances with too much uncertainty and improve the optimization process of the regressor. The parameter s is a hyperparameter of the proposed method and is later evaluated in Section 3.4.

This process bears some resemblance to iteratively reweighted least squares or Expectation Maximization (EM) as mentioned in [Yao and Li, 2014] for the context of single-target regression; however, here we only take a

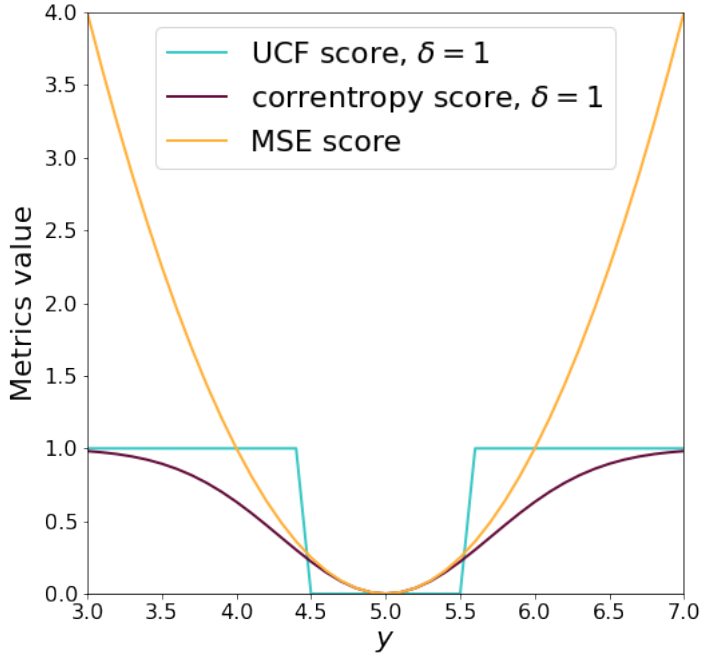


FIGURE 3.2: Comparison of Mean Squared Error (MSE), Uniform Cost Function (UCF), and correntropy; for single target estimate where true $y = 5$.

single step rather than an iterative EM-like procedure. The mechanism is summarized as pseudocode in Algorithm 1.

3.3.2 Mechanism 2: ensemble mode prediction

We train an ensemble of Regressor Chains, each with a random order. Most of the previously developed ensemble methods (e.g. Random Forests [Breiman, 2001] and Ensembles of Regressor Chains [Spyromitros-Xioufis et al., 2016]) use mean averaging to obtain the final predictions. However, we would like to fit our models on datasets with a multi-modal distribution and have them identify a mode. Since in this setting, the mean does not necessarily coincide with a mode, we develop an approach to search for a mode of distribution. Therefore, instead of averaging, we first apply K-means clustering [Lloyd, 1982] in order to identify modes, and then produce the mean of the largest cluster as an estimate of the mode of the predictive distributions.

An example with two modes is given in Fig. 3.3. An average of the bigger cluster of predictions better corresponds to a ground-truth value than an average of all predictions. We select 10 Regressor Chains in an ensemble as a standard trade-off between the accuracy of prediction and computation time [Spyromitros-Xioufis et al., 2016].

3.4 Experiments

3.4.1 Methods

Table 3.1 summarizes the methods used in the experiments; all of which as implemented in Scikit-Learn [Pedregosa et al., 2011]. We experimented with different base estimators for multi-target methods (as indicated in the table).

Algorithm 1 mmERC: training h_j for target y_j (done for $j = 1, \ldots, L$)

procedure FIT(h_j, $\{\boldsymbol{x}, y_j\}$) ▷ Train Base Estimator h_j for target y_j on $\{\boldsymbol{x}, y_j\}$
  $\widetilde{h_j} \leftarrow$ clone of h_j
  fit $\widetilde{h_j}$ on $\{(\boldsymbol{x}, y_j)\}$ ▷ First training phase (full training set)
  $\hat{y}_j \leftarrow \widetilde{h_j}(\boldsymbol{x})$ ▷ Prediction of $\widetilde{h_j}$ on $\boldsymbol{x}$
  corr $\leftarrow 1 - e^{-(y_j - \hat{y}_j)^2}$ ▷ Correntropy; See Eq. 3.2
  $\{\boldsymbol{x}', y'_j\} \subset \{\boldsymbol{x}, y_j\}$ ▷ Top s-instances wrt (lowest) corr, $0 < s < 1$
  fit h_j on $\{(\boldsymbol{x}', y'_j)\}$ ▷ Second training phase
  return h_j ▷ Return the trained model

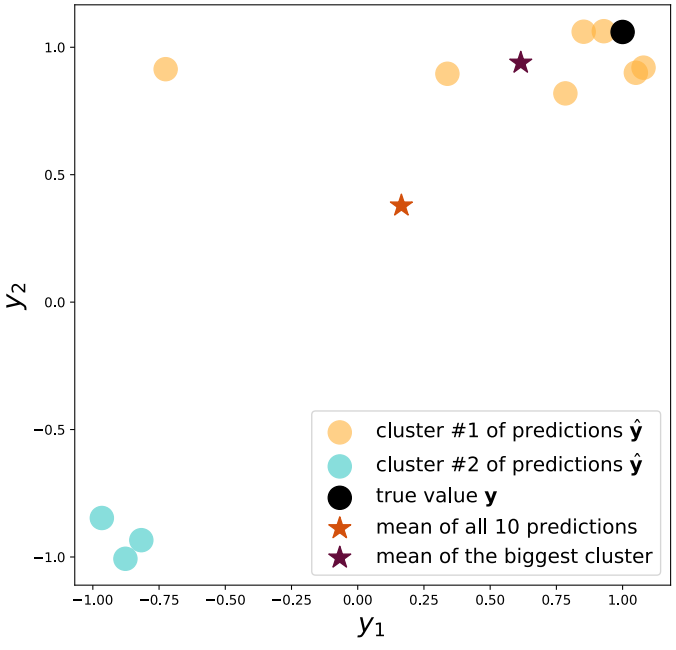


FIGURE 3.3: The mmERC method, mechanism 2: average of the largest cluster (#1) gives a more precise prediction of $\boldsymbol{y} = \{y_1, y_2\}$ which is closer to the true value.

TABLE 3.1: Regression methods compared in the experiments.

	(Meta) Method	Base estimator
DT	Multi-output Decision Tree	–
RF	Multi-output Random Forest	–
IR (dt)	Independent Regressors	Decision Tree
IR (rf)	Independent Regressors	Random Forest
IR (svr)	Independent Regressors	Support Vector Regression
RC (dt)	Regressor Chain	Decision Tree
RC (rf)	Regressor Chain	Random Forest
RC (svr)	Regressor Chain	Support Vector Regression
ERC (dt)	Ensembles of Regressor Chains	Decision Tree
ERC (rf)	Ensembles of Regressor Chains	Random Forest
ERC (svr)	Ensembles of Regressor Chains	Support Vector Regression
mmERC (dt)	Multi-Modal Ensembles of Regressor Chains	Decision Tree
mmERC (rf)	Multi-Modal Ensembles of Regressor Chains	Random Forest
mmERC (svr)	Multi-Modal Ensembles of Regressor Chains	Support Vector Regression

3.4.2 Evaluation

We used two evaluation metrics: average Relative Root Mean Squared Error (aRRMSE), which is common to use in multi-target regression,

$$\mathsf{aRRMSE} = \frac{1}{L}\sum_{j=1}^{L}\sqrt{\frac{\sum_{i=1}^{N}(y_{ij}-\hat{y}_{ij})^2}{\sum_{i=1}^{N}(y_{ij}-\overline{y}_j)^2}},$$

(where $\bar{y}_j$ is the mean value of the j-th target in the training data); and UCF [Burger and Lucka, 2014] – an analog of the 0/1 Loss for regression problems within given neighbourhood δ of the true values, see Eq. 3.1. For the experiments, we take $\delta = 1.0$ for the targets scaled normally.

All the methods were evaluated using a 10-fold cross-validation.

3.4.3 Datasets

We evaluated our algorithm on 40 synthetic datasets and one real-world dataset.

We generated $40 = 8 \cdot 5$ synthetic datasets as pairwise combinations of 8 distributions for target variables $\boldsymbol{y} = \{y_1, y_2\}$, and 5 distributions for a feature variable x. Total number of instances varies from 200 to 600. The clusters $c = 0$ and $c = 1$ are generated with Bernoulli distribution $c \sim \mathcal{B}(0.5)$. The distributions of targets $\boldsymbol{y}$ are Gaussian mixtures forming two clusters and presenting a variety of shapes, illustrated in Fig. 3.4. They vary in standard deviation, number of instances, shape, rotation, and proximity to each other. The feature variable x is designed to provide little information about the targets and thus invoke high predictive uncertainty so that the dependencies between the targets are even more useful for the model than

the feature. Different distributions of x reflect different degrees of uncertainty about which cluster the model should choose for a particular sample, summarized in Table 3.2. In the scenarios A and E, the feature x does not provide information to which of two clusters $\boldsymbol{y}$ belongs. In the scenarios B, C, and D, the feature x is generated taking the cluster of $\boldsymbol{y}$ into account. In each generated synthetic dataset, one of 8 distributions of targets $\boldsymbol{y}$ and one of 5 distributions of feature x are combined.

A real-world dataset (432 instances) was taken from the R package *agricolae* [de Mendiburu and de Mendiburu, 2019] and refers to a native plant of the Peruvian Andes called *yacon (Smallanthus sonchifolius)*. The data belongs to the International Potato Center in Lima (Peru). As targets, we consider two multi-modally distributed features from the dataset: *degrees brix* (density or sugar concentration) and *height* of the plant. We add feature $x \sim \mathcal{N}(0, 1)$ which again invokes a big amount of predictive uncertainty. The distributions of the targets are demonstrated in Fig. 3.5.

3.5 Results and Discussion

An initial investigation indicates that mmERCs achieve generally the best performance with a parameter value $s = 0.5$ in Algorithm 1, i.e. taking half of the training dataset in the second training phase (see Fig. 3.6). The subsequent experiments in this chapter were conducted with $s = 0.5$.

The experimental results for the UCF metrics (Table 3.3) show that our method, mmERC, on average outperforms the independent regressors as well as standard Regressor Chains with a sequential cascade order. This is already an important result not found in other Regressor Chains implementations. Moreover, our proposed mechanism to deal with multi-modal distributions improves the performance of Ensembles of Regressor Chains for all base estimators in most of the scenarios. These results are also illustrated by the Friedman-Nemenyi diagram shown in Fig. 3.7a, where

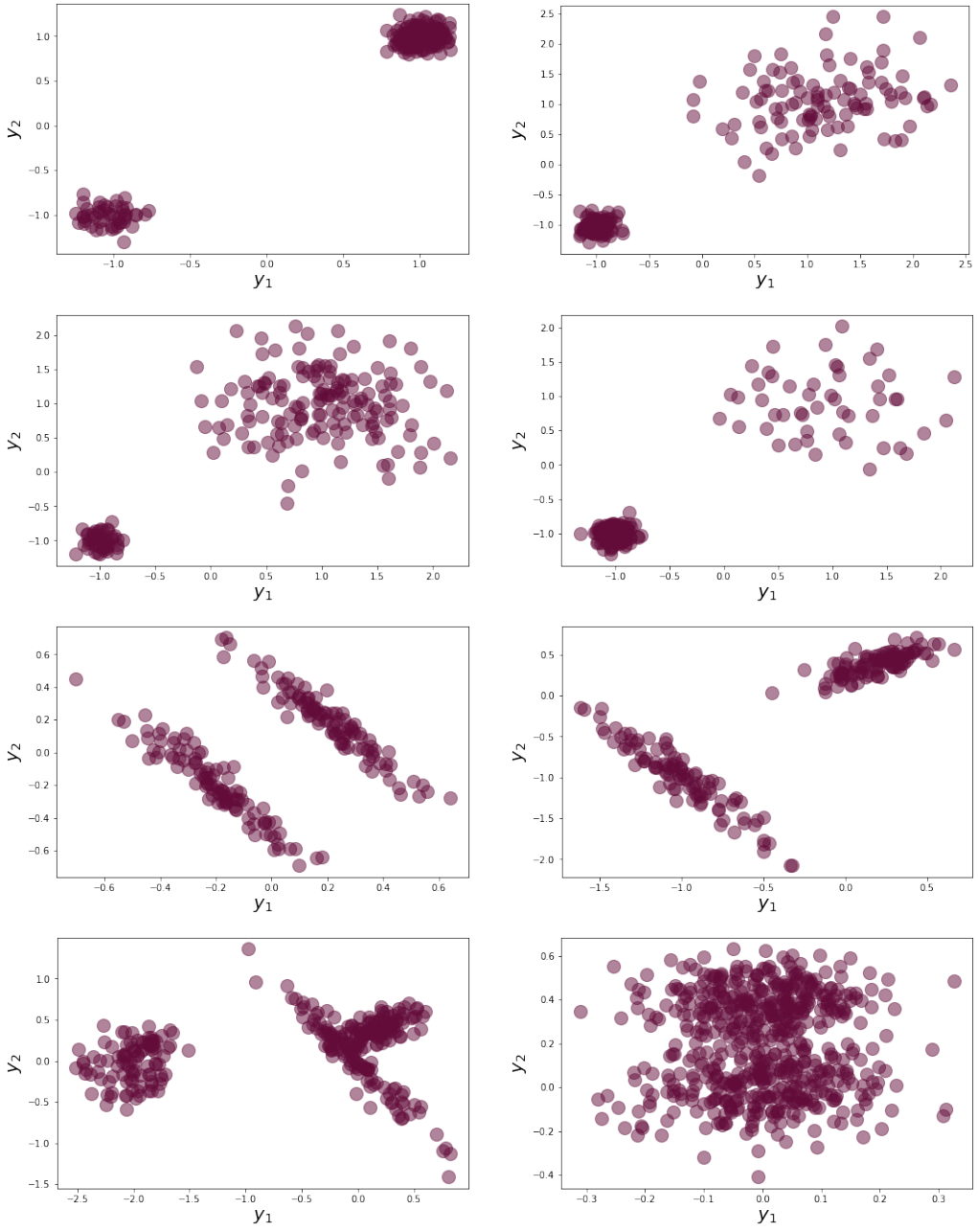


FIGURE 3.4: Points drawn from eight distributions of the targets $\boldsymbol{y} = \{y_1, y_2\}$ in generated synthetic datasets.

TABLE 3.2: Distributions of the feature x in generated synthetic datasets.

Group	Distribution
A:	$\sim U(0,1)$ where U stands for uniform distribution
B:	$\begin{cases} 0 \text{ if } c = 0, \\ 1 \text{ if } c = 1 \end{cases}$
C:	$\begin{cases} \sim U(0,1) \text{ if } c = 0, \\ \sim U(1,2) \text{ if } c = 1 \end{cases}$
D:	$\begin{cases} \mathcal{N}(0,1) \text{ if } c = 0, \\ \mathcal{N}(1,1) \text{ if } c = 1 \end{cases}$
E:	$\sim \mathcal{N}(0,1)$

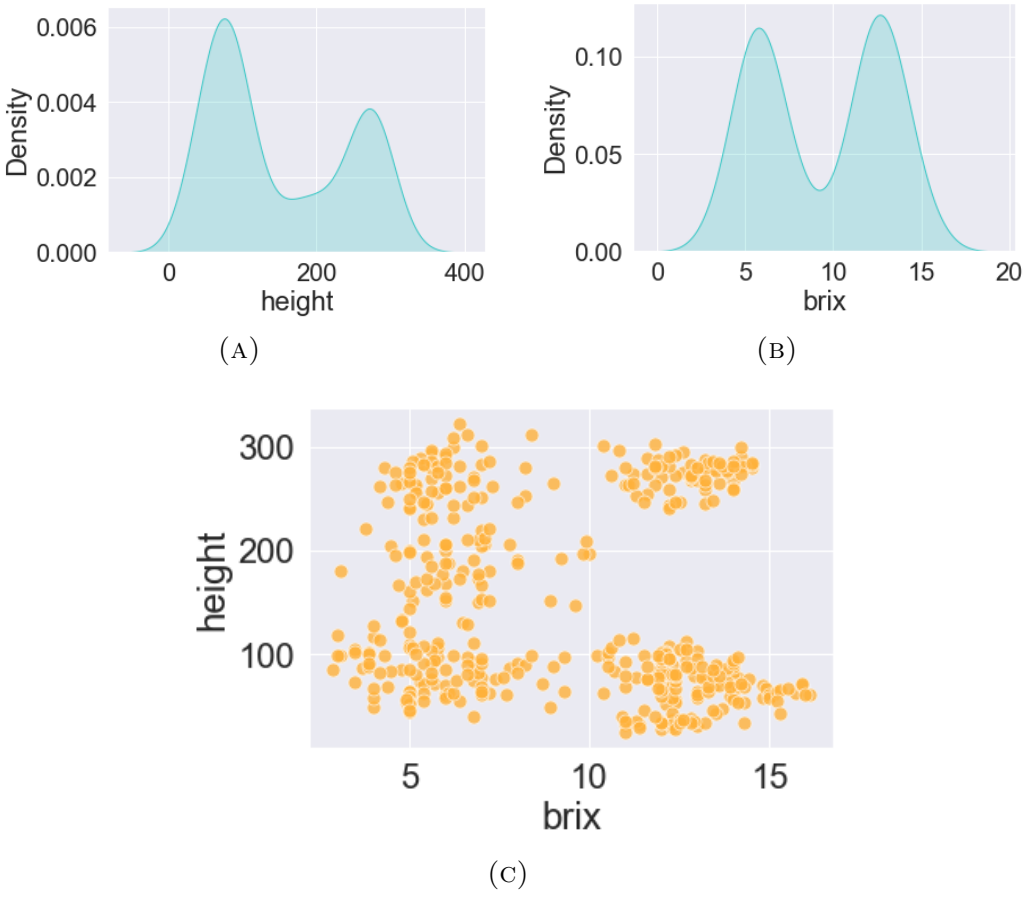


FIGURE 3.5: Distributions of the targets height and brix in the *yacon* dataset shown in (A) and (B) have both bi-modal structures. Samples of the dataset shown in (C) form visibly separated clusters.

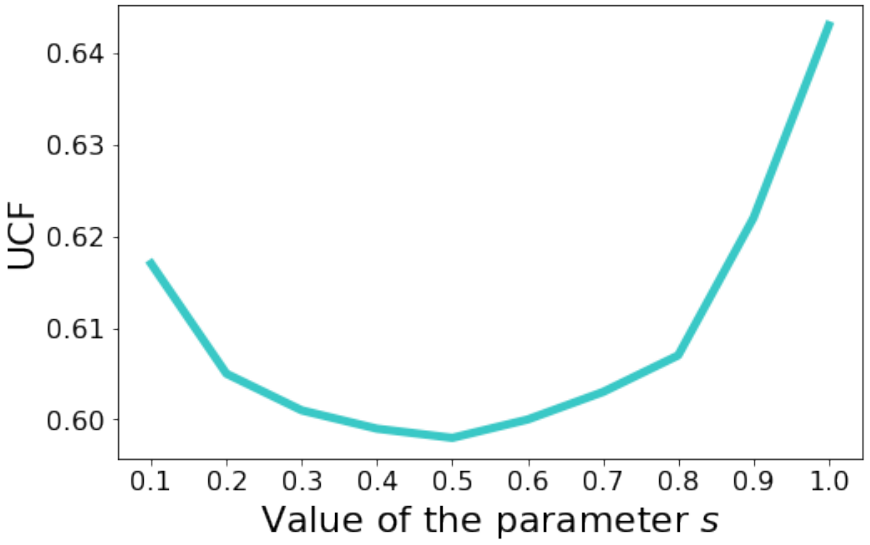


FIGURE 3.6: Averaged UCF metric for the mmERC method, measured across all synthetic datasets and grouped by the value of the s parameter used in mechanism 1, $s \in (0, 1]$.

the rank of the mmERC method is significantly better than for all other tree-based methods.

As expected, the results under aRRMSE are less optimistic, see Table 3.4 and Friedman-Nemenyi diagram in Fig. 3.7b. However, in Fig. 3.8 we show that mmERCs recognize clustered distributions better than ERCs both for Decision Trees and Random Forests as base estimators. The same situation is observed for the other datasets and other base estimators. We propose the following explanation: MSE-based metrics penalize choosing the wrong cluster more than putting estimations in-between of actual clusters since the distance between prediction and the true value is bigger in the former case. Thus, when a model recognizes a multi-modal distribution but fails to choose the right cluster for some points, it can perform worse under MSE-based metrics than models fitting to a single Gaussian distribution. We, therefore, argue that this standard choice of the aRRMSE metrics may be inappropriate in the case of multi-modal distributions and requires further investigation.

In general, Decision Trees and DT-based models successfully recognize clustered distributions, but in the lack of informative features, they assign clusters randomly. This can be seen in Random Forests (which are an average of a number of random Decision Trees) results: all models, based on Random Forests, put the predictions between the real clusters. Furthermore, Decision Trees are formed as sets of decision boundaries and thus are not smooth. Random Forests should be able to solve this issue, but, as we mentioned above, do not work well for recognizing multi-modal nature. Our method, mmERC, improves the performance of Random Forests methods and outputs a smooth function at the same time.

In the *Yacon* dataset, we observe the best predictive performance under UCF for the mmERC models, see Table 3.5. Fig. 3.9a and 3.9b illustrate the performance of the two models, mmERC (based on Random Forests)

TABLE 3.3: UCF results for the synthetic datasets. For simplicity of presentation, the results are grouped and averaged by type of x feature distribution as they reflect different degrees of uncertainty. This simplification does not affect the average values of metrics and average ranks. The best value per group is in **bold**. The results are rounded to 2 decimal points to display, so minor differences may be not seen in this representation.

Regressor	A	B	C	D	E	Average	AvgRank
DT	0.71	0.50	0.50	0.70	0.73	0.63 ± 0.01	7.9
RF	0.84	0.47	0.45	0.78	0.84	0.67 ± 0.04	10.2
IR (dt)	0.79	0.50	0.52	0.74	0.78	0.66 ± 0.02	11.1
IR (rf)	0.86	0.47	0.47	0.79	0.87	0.69 ± 0.04	11.0
IR (svr)	0.72	**0.40**	0.52	0.70	0.72	0.61 ± 0.02	6.0
RC (dt)	0.74	0.50	0.51	0.70	0.72	0.63 ± 0.01	8.6
RC (rf)	0.81	0.45	0.45	0.75	0.82	0.66 ± 0.03	8.8
RC (svr)	0.70	**0.40**	0.51	0.67	0.71	0.60 ± 0.02	4.2
ERC (dt)	0.78	0.50	0.49	0.72	0.76	0.65 ± 0.02	8.6
ERC (rf)	0.83	0.44	**0.44**	0.76	0.83	0.66 ± 0.04	8.6
ERC (svr)	0.71	**0.40**	0.50	0.67	0.72	0.60 ± 0.02	5.0
mmERC (dt)	0.72	0.50	0.51	0.69	0.71	0.63 ± 0.01	8.2
mmERC (rf)	0.69	0.43	**0.44**	**0.63**	**0.67**	$\mathbf{0.57 \pm 0.02}$	**2.2**
mmERC (svr)	**0.69**	**0.40**	0.52	0.67	0.68	0.59 ± 0.02	4.6

TABLE 3.4: aRRMSE results for the synthetic datasets. For simplicity of presentation, the results are grouped and averaged by type of x feature distribution as they reflect different degrees of uncertainty. This simplification does not affect the average values of metrics and average ranks. The best value per group is in **bold**. The results are rounded to 2 decimal points to display, so minor differences may be not seen in this representation.

Regressor	A	B	C	D	E	Average	AvgRank
DT	1.46	0.64	0.67	1.38	1.48	1.13 ± 0.19	12.4
RF	1.17	0.55	0.55	1.11	1.17	0.91 ± 0.11	6.2
IR (dt)	1.47	0.64	0.69	1.38	1.47	1.13 ± 0.18	12.8
IR (rf)	1.17	0.55	0.55	1.11	1.17	0.91 ± 0.11	6.8
IR (svr)	**1.10**	0.46	0.60	**1.02**	**1.10**	$\mathbf{0.86 \pm 0.09}$	2.8
RC (dt)	1.46	0.64	0.69	1.40	1.47	1.13 ± 0.18	12.6
RC (rf)	1.29	0.53	0.54	1.21	1.30	0.97 ± 0.16	7.4
RC (svr)	1.13	0.46	0.60	1.05	1.12	0.87 ± 0.10	3.6
ERC (dt)	1.35	0.63	0.64	1.28	1.36	1.05 ± 0.14	10.0
ERC (rf)	1.17	0.51	**0.52**	1.09	1.17	0.90 ± 0.12	4.8
ERC (svr)	1.12	**0.46**	0.59	1.03	1.11	$\mathbf{0.86 \pm 0.10}$	**2.6**
mmERC (dt)	1.42	0.63	0.71	1.40	1.45	1.12 ± 0.17	12.2
mmERC (rf)	1.20	0.49	0.53	1.11	1.20	0.91 ± 0.13	5.8
mmERC (svr)	1.16	0.47	0.61	1.06	1.16	0.89 ± 0.11	5.0

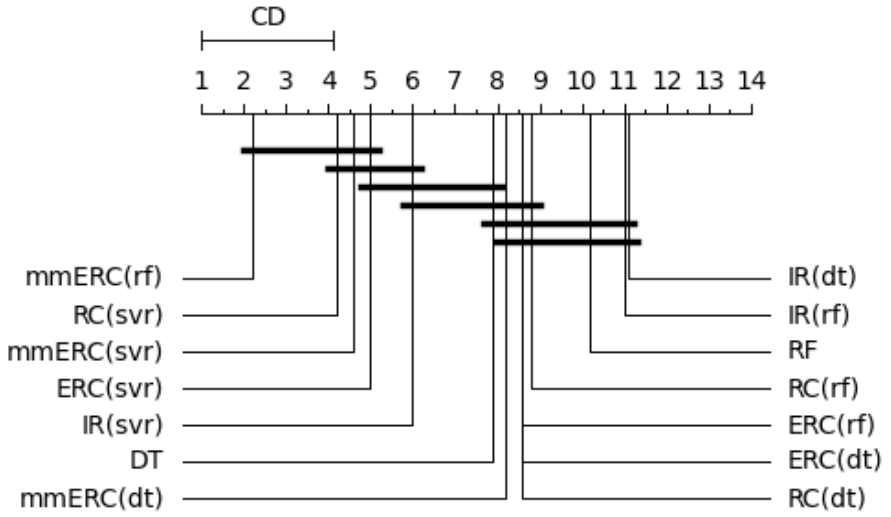
and Decision Trees, respectively. Though graphically it seems that Decision Trees better mimic the distribution of the clusters, from the UCF comparison we imply that they assign these clusters in a more random way. Fig. 3.9c compares the precision of predictions of these two models per sample. It shows that our method is more precise on some of the clusters. Though we have not observed a significant advantage of our approach on real-world datasets, we argue that it performs well on some datasets with explicit multi-modality, particularly on some subsets of samples.

3.6 Conclusions and Future Work

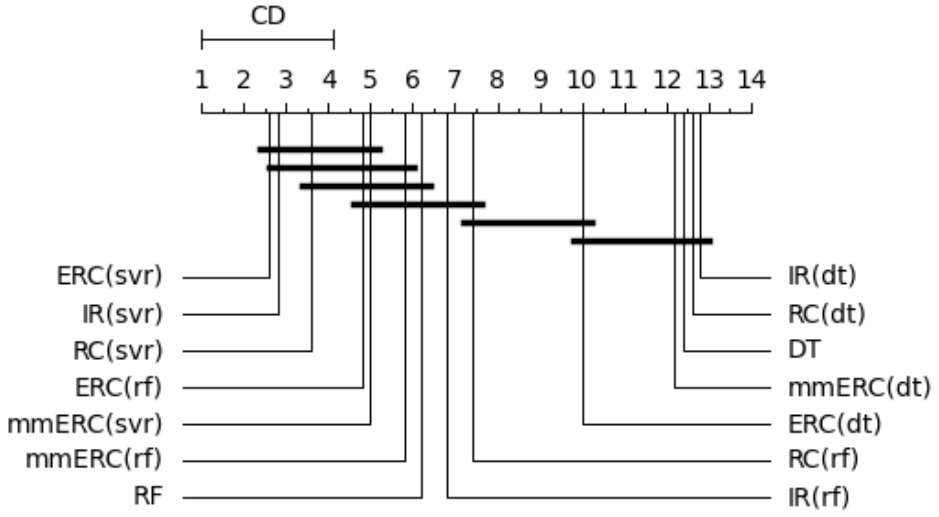
In this work, we have developed a new method, Multi-Modal Ensembles of Regressor Chains (mmERC), for multi-target regression. As opposed to the conventional approaches assuming a uni-modal predictive distribution approximating Gaussians, our approach is better able to capture the modes of the distribution. The experimental study compares the performance of the proposed method, independent regressors, standard Regressor Chains, and Ensembles of Regressor Chains on 40 multi-modal synthetic and one real-world datasets.

In empirical evaluation under the UCF metrics, mmERC achieves important performance improvement across the multi-modal distributed datasets, outperforming baseline and state-of-the-art methods. This is unlike the vast majority of multi-target (and standard single-target) regression approaches which target squared-error-based metrics. Our study hints that a choice of UCF metric for measuring performance can be more adequate than using standard errors and that this metric deserves further investigation.

In future work, we consider looking at additional evaluation schemas, such as allowing multiple multi-output predictions (hypotheses) for a single instance. This would allow a greater chance of capturing the true mode, even when uncertainty is high. Furthermore, a more sophisticated structure of

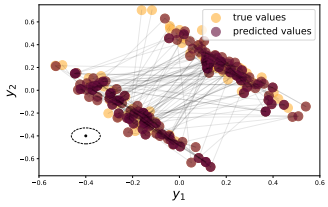


(A) UCF

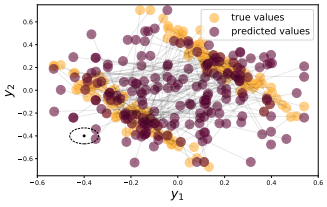


(B) aRRMSE

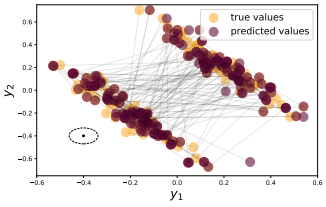
FIGURE 3.7: Friedman-Nemenyi diagrams comparing the ranking of the experimentally tested methods. A lower rank is better, statistically undistinguishable methods are connected by a horizontal line.



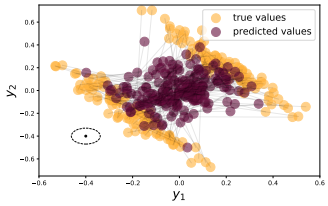
(A) DT



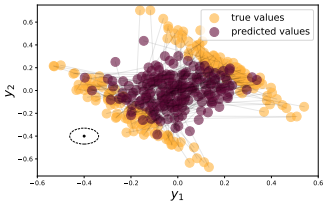
(B) IR (dt)



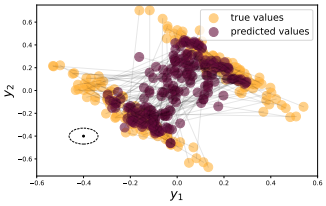
(C) RC (dt)



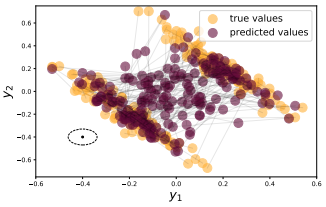
(D) RF



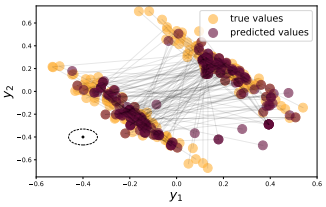
(E) IR (rf)



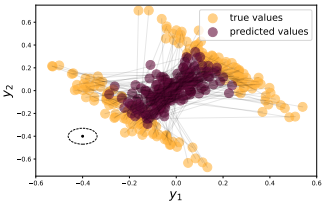
(F) RC (rf)



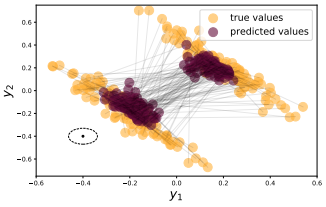
(G) ERC (dt)



(H) mmERC (dt)



(I) ERC (rf)

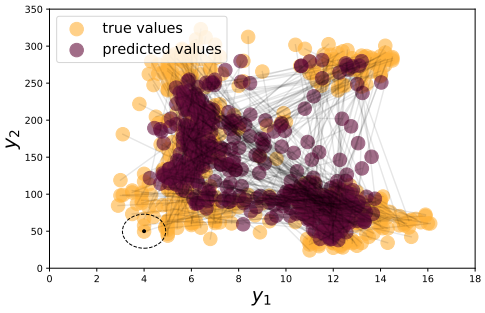


(J) mmERC (rf)

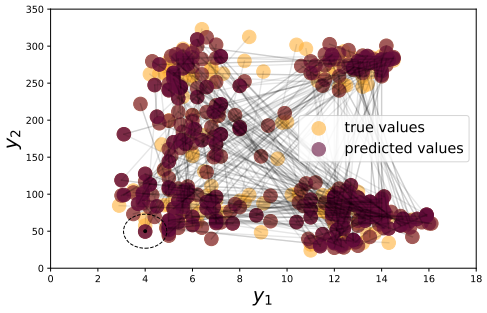
FIGURE 3.8: Models based on DTs and RFs on one of the synthetic datasets with a feature $x \sim \mathcal{N}(0, 1)$. Black lines connect pairwise true and predicted values. Black dashed ellipses represent the size of δ-neighborhood used in UCF metrics.

TABLE 3.5: UCF results for the *Yacon* dataset.

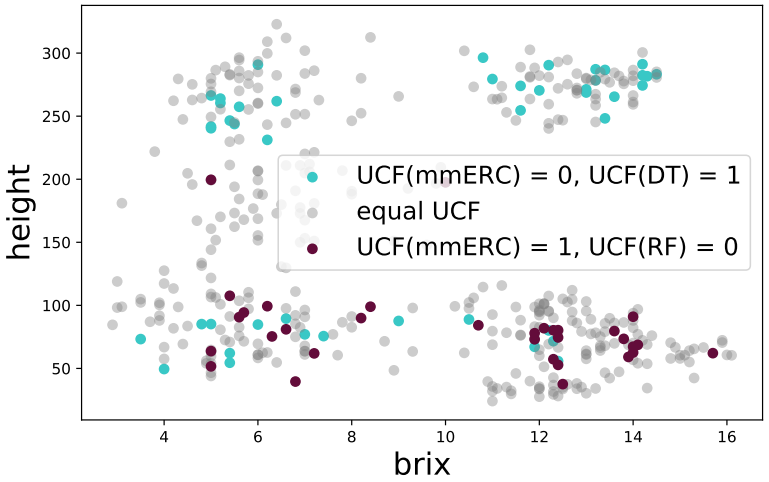
DT	RF	IR (dt)	IR (rf)	IR (svr)	RC (dt)	RC (rf)	RC (svr)	ERC (dt)	ERC (rf)	ERC (svr)	mmERC (dt)	mmERC (rf)	mmERC (svr)
0.92	0.94	0.92	0.94	0.91	0.92	0.94	0.92	0.92	0.94	0.94	0.89	0.89	**0.83**



(A) mmERC (rf)



(B) DT



(C) The mmERC (rf) method performs better (w.r.t. UCF) on blue dots and worse on violet dots. On grey dots, both methods have the same UCF values.

FIGURE 3.9: Comparison of (A) mmERC (rf) and (B) DT models on *yacon* dataset. Predictions of both methods together in (C).

the chains in the ensembles in order to better exploit dependencies between the targets and achieve better predictive results.

Additionally, the proposed approach may be evaluated in the concept drift machine learning tasks, where the relationships between inputs and outputs in the underlying problem change over time, thus provoking multi-modality of the target distribution as well as predictive uncertainty.

Chapter 4

Backward inference in probabilistic Regressor Chains

As discussed earlier in this thesis, state-of-the-art approaches for multi-target prediction, such as Regressor Chains, can exploit interdependencies among the targets and model the outputs jointly. However, these models are often too inflexible to answer queries under constraints such as when targets jointly comprise a distribution and/or when certain target values are fixed prior to inference and cannot be incorporated into the modeling of the other targets. These limitations complicate the practical usage of such models, particularly in applications where targets are highly dependent and must be modeled as such. In this chapter, we present a solution to the aforementioned problem as a backward inference algorithm for Regressor Chains via Metropolis-Hastings sampling. We evaluate the proposed approach via different metrics using both synthetic and real-world data. We show that it is able to solve the issue with much lower error than marginal inference (i.e. ignoring joint modeling). Furthermore, we show that the proposed method

can provide useful insights into a real-world problem, namely in predicting the distribution of potential natural vegetation.

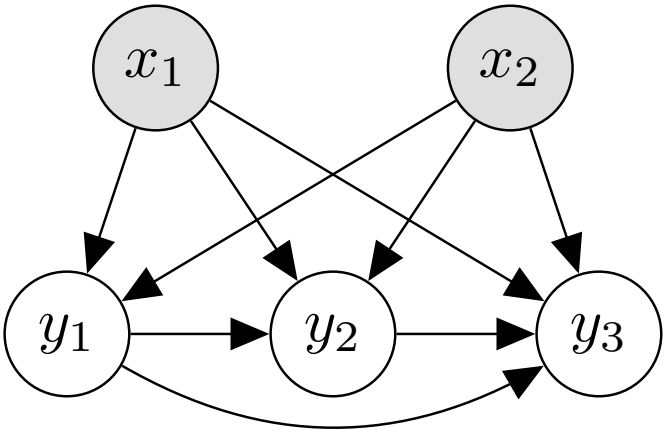
4.1 Introduction

In Regressor Chains, increasingly used for multi-output prediction, predictions are cascaded across the outputs. This means that the predictions of the first labels in the chain are used as input features to model the rest of the chain. Fig. 4.1a illustrates the standard setting of Regressor Chains for 3 targets. It has been well used in the context of multi-label classification (as Classifier Chains, for binary outputs [Read et al., 2021]). There are recent successes also in the multi-target regression context with continuous outputs: for example, Regressor Stacking [Santana et al., 2017], Ensembles of Regressor Chains [Spyromitros-Xioufis et al., 2016; Antonenko and Read, 2022] (see also Chapter 3), and probabilistic frameworks [Read and Martino, 2020]. A variety of applications can be targeted with such methods [Poonawala-Lohani et al., 2021].

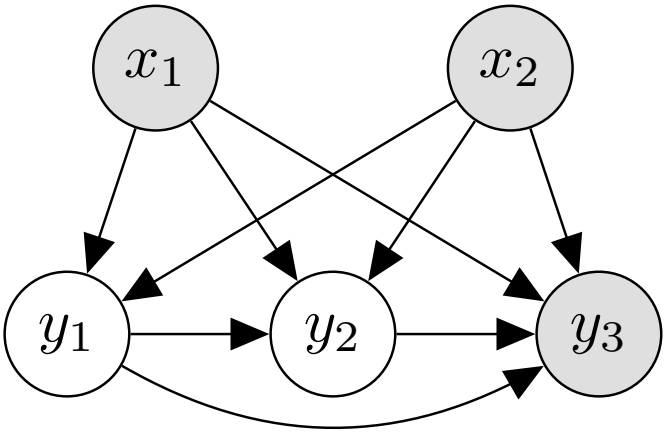
However, the above-cited works make a number of standard assumptions: all outputs are to be predicted, each can be predicted individually, and those models can be retrained at will and with relative ease. We consider a new setting that breaks these assumptions, by imposing the following constraints:

1. Any output may be imputed/fixed prior to prediction;
2. Base regression models cannot be retrained;
3. Outputs satisfy a joint constraint.

We aim at inferring a joint posterior distribution over labels, i.e. probabilistic Regressor Chains, under these constraints.



(A)



(B)

FIGURE 4.1: Illustration of a Regressor Chain (depicted as a Bayesian network, where shaded nodes indicate fixed observations) for inputs $\boldsymbol{x} = \{x_1, x_2\}$ and outputs $\boldsymbol{y} = \{y_1, y_2, y_3\}$. (A) demonstrates the standard setting where forward inference is possible; (B) demonstrates challenges we address: how to propagate imputed label information (label y_3) 'backward' while maintaining a joint constraint and without training a new structure.

While training a model with the constrained targets included in the features set (i.e. ignoring constraint 2.) may be considered a simpler solution, we argue that training a new model may not always be easily accessible. Oppositely, we want to extract as much information as possible from the model obtained earlier, trained on provided sets of features and targets. This can be the case, for example, if due to ethical concerns, the data is not accessible after training the model or computational resources are limited. Nowadays, more and more datasets are restricted, and, for example, in federated learning, it is typical to transmit the model while keeping the data characteristics hidden. This is similar to transfer learning when a pre-trained model is used for a new task to avoid excessive training or model tuning.

This problem may be also reformulated as a missing value problem but with missing labels in the output space. However, we do not discuss here standard missing value imputation methods as these would not satisfy constraint 2., using the previously trained model which predicts the outputs $\boldsymbol{y}$ from the inputs $\boldsymbol{x}$.

In this work, we consider the following motivating example of a problem satisfying the constraints 1.-3. We solve a problem of estimation of the hypothetical vegetation and land-cover types based on climatic conditions supposing that no urban activity was present while only data with urban activity observed is available for the model training. In this setting, we must fix the proportion of urban to 0 (constraint 1.) in order to query the model on what types would be present under such conditions. Further, we suppose that models cannot be retrained (constraint 2.) since, for example, access to human expertise and/or to the training data used to build the model has expired, or there is insufficient time or computational resources to retrain and re-test models (e.g. re-validate for robustness, etc.). Additionally, since this is an example of compositional data, the outputs comprise a categorical

distribution, and thus their value, for any given input, must sum to 1 (i.e. constraint 3.).

We remind the reader that $\boldsymbol{y} = [y_1, \ldots, y_L]$ represents the L target components of the data. As a categorical distribution (i.e. outputs represent compositional data) it should be that $\sum_{l=1}^{L} y_l = 1$, and $y_l \geq 0$. Our goal is to answer queries of the form

$$p(\boldsymbol{y}_{\neg F} \,|\, \boldsymbol{x}, \boldsymbol{y}_F), \tag{4.1}$$

where $F \subset \{1, \ldots, L\}$ is a set of fixed/observed outputs, and $\neg F = \{1, \ldots, L\} \setminus F$ are the remaining outputs to predict; e.g. $\boldsymbol{y}_{\neg F} = [y_1, y_2]$ and $\boldsymbol{y}_F = [y_3]$ in Fig. 4.1b.

A Regressor Chain $\mathcal{H} = [h_1, ..., h_L]$ involves a model (regressor) h_l for each of the outputs $y_1, \ldots, y_L$ providing prediction

$$\hat{y}_l = h_l(\boldsymbol{x}, y_1, \ldots, y_{l-1})$$

which is, typically, a function of probability density function (pdf)

$$p(y_l \,|\, \boldsymbol{x}, y_1, \ldots, y_{l-1}), \tag{4.2}$$

e.g. the expected value

$$\hat{y}_i = E_{y_i \sim p(y_i \,|\, \boldsymbol{x}, y_1, \ldots, y_{i-1})}[y_i].$$

This allows us to provide a prediction for all outputs,

$$\hat{\boldsymbol{y}} = [\hat{y}_1, \ldots, \hat{y}_L] = [h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, \hat{y}_1), \ldots, h_L(\boldsymbol{x}, \hat{y}_1, \ldots, \hat{y}_{L-1})].$$

Recall that each prediction becomes a feature for the following model in the 'chain'. By this mechanism, Regressor Chains (as well as Classifier Chains, e.g. [Read et al., 2011]) aim to model the outputs together, or jointly.

At the same time, independent models will inherently make an assumption of independence as they model each $\hat{y}_l = h_l(\boldsymbol{x})$ separately, and thus the joint constraint is not satisfied, so independent models are not applicable in the given setting.

If the pdf is explicitly modeled (which is the case of Probabilistic Regressor Chains [Read and Martino, 2020]), Regressor Chains also provide the joint posterior distribution:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{l=1}^{L} p(y_l \mid \boldsymbol{x}, y_1, \ldots, y_{l-1}). \tag{4.3}$$

Returning to the example model pictured in Fig. 4.1, the corresponding joint distribution is given by $p(\boldsymbol{y} \mid \boldsymbol{x}) = p(y_1 \mid \boldsymbol{x}) \cdot p(y_2 \mid \boldsymbol{x}, y_1) \cdot p(y_3 \mid \boldsymbol{x}, y_1, y_2)$. However, here we face the challenge posed by the interaction of constraints 1. (fixed output) and 3. (joint constraint): if y_3 is a fixed observation, forward inference along the chain cannot be completed while respecting the other constraints; specifically the term $p(y_3 \mid \boldsymbol{x}, y_1, y_2)$. A naive approach of simply predicting $\hat{y}_1$ and $\hat{y}_2$ and then normalizing them to meet the constraint $\sum_{l=1}^{L} \hat{y}_l = 1$ is not valid, because it answers the query $p(\boldsymbol{y}_{\neg F} \mid \boldsymbol{x})$ but not the target query$p(\boldsymbol{y}_{\neg F} \mid \boldsymbol{x}, \boldsymbol{y}_F)$.

We propose a method, Metropolis-Hastings sampled Regressor Chains (mhsERC), which is able to provide a solution to the aforementioned problem by combining Regressor Chains and Metropolis-Hastings sampling for backward inference in the prediction step. We apply our approach, mhsERC, on synthetic and real-world datasets and find that:

- In the case of synthetic data, the resulting distribution provided by mhsERC is very close to the ground-truth, for a given imputation. The model naturally provides a distribution for each instance in addition to a predicted mean value;

- In three multi-target regression datasets, we provide values for one target explicitly and compare predictions of the other targets. We conclude that mhsERC successfully infers missing targets, given other targets held fixed as observations, and reaches significantly better performance than the baseline methods;
- In the real-world climate and land-cover data, we were able to extract insights regarding the potential distribution of vegetation, after fixing the percentage of urban cover to zero.

The rest of the chapter is organized as follows. After summarizing the background and related work in Section 4.2, we present our approach in Section 4.3. Data and evaluation metrics are presented in Section 4.4. The results and their discussion as well as complexity analysis are in Section 4.5. In Section 4.6, we draw conclusions and describe future work.

4.2 Related work

Although there exist a variety of methods for multi-target regression (for example, Predictive Clustering Trees [Blockeel et al., 2000], Regressor Chains and Ensembles of Regressor Chains [Spyromitros-Xioufis et al., 2016], Regressor Stacking [Santana et al., 2017], etc.), these approaches are typically used in a standard predictive setting and do not directly target joint prediction of targets when some of the output values are provided before the prediction.

In [Read and Martino, 2020], Regressor Chains were further developed into a probabilistic framework, however only ancestral, or forward, inference along the chain is available which does not respond to the set constraints 1.–3. Also, the authors did not propose how to use non-probabilistic base classifiers such as Decision Trees.

The given problem (recall example in Fig. 4.1b) can be represented as a Bayesian Network, specifically a Hybrid Bayesian Network [Salmerón et al., 2018] which can handle continuous variables during inference as opposed to classic Bayesian Networks. However, learning the structure of a Bayesian Network is extremely costly and inference options are limited, normally corresponding to linear-Gaussian models or approximate methodologies based on sampling, and variational inference.

The problem setting involving the distributional constraint, with missing value, has been called 'structurally incomplete' by [Beigaitė et al., 2022]; but authors here use a neural network approach that can be built arbitrarily. Oppositely, we develop a probabilistic inference approach under Regressor Chains. The general setting for predicting a composition of outputs is known in the statistics literature as 'compositional data analysis' [Aitchison, 2005].

As an example of a real-world problem involving constraints 1.–3, we consider the prediction of potential vegetation distribution in the absence of urban activity. A similar setting was considered in [Beigaitė et al., 2022]. In our work, we also face the issue of the evaluation of ground-truth distribution for comparison, since the goal is to explore alternative hypotheses. Our solution is to study the probabilistic challenge of deriving a joint distribution directly; whereas the authors of [Beigaitė et al., 2022] focus on the accuracy of predicting dominant vegetation types. Another important difference is that we consider the additional constraint of tackling the problem at *inference* time, rather than selecting different training regimes.

In ecology and biogeography, a related research question concerns the inference of potential natural vegetation; the anticipated state of mature vegetation under specific environmental conditions, without any human intervention [Chiarucci et al., 2010]. In recent years, statistical and machine-learning techniques have gained popularity for their application in constructing such models [Hemsing and Bryn, 2012; Hengl et al., 2018; Raja

et al., 2018] but, in these works, the focus is on exploring the relationship from climatic input observations to the targets, rather than the probabilistic relationship among the targets, as we do.

4.3 Our method: Metropolis-Hasting sampled Regressor Chains

As mentioned in Section 4.1, we target inferring the probability defined by Eq. (4.1). In other words, we want to evaluate the probability

$$\pi(\hat{\boldsymbol{y}}) = p(\hat{\boldsymbol{y}}_{\neg F} \mid \hat{\boldsymbol{y}}_F, \hat{\boldsymbol{x}})$$

for any particular $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{x}}$, where $\hat{\boldsymbol{y}}_F$ are fixed and $\hat{\boldsymbol{y}}_{\neg F}$ may vary. By the definition of condition probability,

$$\begin{aligned}\pi(\hat{\boldsymbol{y}}) = \frac{p(\hat{\boldsymbol{y}}_{F\cup\neg F}, \hat{\boldsymbol{x}})}{p(\hat{\boldsymbol{y}}_F, \hat{\boldsymbol{x}})} &= \frac{\Pi_{l=1}^{L} p(\hat{y}_l \mid \hat{y}_{l-1}, ..., \hat{y}_1, \hat{\boldsymbol{x}})}{p(\hat{\boldsymbol{y}}_F, \hat{\boldsymbol{x}})} \\ &\propto \Pi_{l=1}^{L} p(\hat{y}_l \mid \hat{y}_{l-1}, ..., \hat{y}_1, \hat{\boldsymbol{x}}),\end{aligned}$$

The first important question is how to evaluate probabilities in the right-hand side of the last expression, $p(\hat{y}_l \mid \hat{y}_{l-1}, ..., \hat{y}_1, \hat{\boldsymbol{x}})$. For this goal, we assume that for each base estimator h_l, the corresponding distribution may be presented as a normal distribution, and it is possible to obtain its parameters, the mean μ and standard deviation σ (see Algorithm 2). These parameters may be inferred, for example,

- directly from the model for Bayesian regression models;
- by bootstrap [Abdar et al., 2021] (non-probabilistic, but ensembled base models, like Random Forest) with an empirical distribution;
- by Monte Carlo Dropout [Abdar et al., 2021];

– by perturbation of input (shallow Monte Carlo Dropout).

In our work, we use Random Forests and calculate the mean and the standard deviation for the predictions of individual trees from the ensemble.

Another important issue is that while we assume that we are able to evaluate this probability for any point $\hat{\boldsymbol{y}}$, we need an explicit method for sampling from this distribution for modeling it. Here we propose to use the Metropolis-Hastings (MH) sampling [Metropolis et al., 1953; Hastings, 1970] as a simple standard sampling method, though other sampling approaches may be applied. In the MH algorithm (summarized as pseudocode in Algorithm 3), a random walk $\{\boldsymbol{y}^{[t]}\}_{t=0...T}$, each iteration proposes a new estimate $\boldsymbol{y}'$ by adding random normal noise to the previous estimate $\boldsymbol{y}^{[t]}$. The new estimate $\boldsymbol{y}'$ is evaluated by the distribution probability function $\pi(\boldsymbol{y}')$ and transition function $q(\boldsymbol{y}^{[t]}, \boldsymbol{y}')$ and accepted as a new step $\boldsymbol{y}^{[t+1]}$ if a randomly generated $r \sim U(0,1)$ is smaller than the acceptance ratio

$$\alpha = \min\left(1, \frac{\pi(\boldsymbol{y}') \cdot q(\boldsymbol{y}^{[t]}, \boldsymbol{y}')}{\pi(\boldsymbol{y}^{[t]}) \cdot q(\boldsymbol{y}', \boldsymbol{y}^{[t]})}\right).$$

In the scope of this work, we assume that the transition function q may be considered symmetrical and thus may be eliminated from the last formula.

To summarize our approach, we generate a targeted distribution for each instance by following these steps:

1. An initial first point of a random walk $\boldsymbol{y}^{[0]}$ is selected randomly (e.g. equal to $\mathbf{0}$);
2. On each iteration, a new estimate $\boldsymbol{y}'$ is proposed by adding random noise to the previous step $\boldsymbol{y}^{[t]}$ and fixing $\boldsymbol{y}'_F$ to the corresponding values;
3. Acceptance ratio $\alpha = \min\left(1, \frac{\pi(\boldsymbol{y}')}{\pi(\boldsymbol{y}^{[t]})}\right)$ is calculated, where $\pi(\hat{\boldsymbol{y}}) = \Pi_{l=1}^{L} p(\hat{y}_l \mid \hat{y}_{l-1}, ..., \hat{y}_1, \hat{\boldsymbol{x}})$, each probability in the product is calculated

as pdf of the normal distribution obtained from base estimators of the Regressor Chain;

4. The proposed estimate $\boldsymbol{y}'$ is accepted as $\boldsymbol{y}^{[t+1]}$ if a randomly generated $r \sim U(0,1)$ is smaller than the acceptance ratio α; otherwise, $\boldsymbol{y}^{[t+1]} = \boldsymbol{y}^{[t]}$;

5. The steps 2.-4. are repeated T times.

Note that the proposed method, Metropolis-Hastings sampled Regressor Chains (mhsRC), is not specific to any particular chain order and can be applied to a Regressor Chain of any order with any set of fixed outputs. If an Ensemble of Regressor Chains was given as a prior trained model, then we can perform the procedure with all individual chains in the ensemble and then average the predictions. We will further call this approach Metropolis-Hastings sampled Ensembles Regressor Chains (mhsERC)

4.4 Experiments

We remind the reader that our goal is to estimate Eq. (4.1). To evaluate possible solutions of this problem, we perform the following experiments. First, we generate synthetic data where all distributions (joint, marginals), including Eq. (4.1), are fully known; therefore we compare the distributions directly. Then, we perform experiments on real-world data where we take values of fixed targets $\boldsymbol{y}_F$ directly from data and evaluate predictions for the other targets $\boldsymbol{y}_{\neg F}$. Finally, we use real-world data and expert intuition to make conclusions with regard to Eq. (4.1), given hypothetical (that is, not from the data) values of $\boldsymbol{y}_F$.

Algorithm 2 Evaluate the probability of the proposed estimate

procedure π(proposed estimate $\boldsymbol{y}'$)
  for l in $1, ..., L$ **do**
    $\mu, \sigma \leftarrow$ mean and std of $h_l(\hat{x}, \hat{y}_1, ..., \hat{y}_{l-1})$
    $p_l \leftarrow pdf(y'_l)$ for $\mathcal{N}(\mu, \sigma)$
  $\pi = \Pi_{l=1}^{L} p_l$

Algorithm 3 Metropolis-Hastings sampled Regressor Chains

procedure MHSRC(T iterations)
  $\boldsymbol{y}^{[0]} \leftarrow 0$ ▷ First step of random walk
  for $0 < t < T$ **do**
    $\boldsymbol{y}' \leftarrow \boldsymbol{y}^{[t]} + \mathcal{N}(0, \sigma_{pr})$ ▷ Propose new $\boldsymbol{y}'$
    ▷ ($\sigma_{pr} = 0.01$ for normalized data)
    $\alpha = \min\left(1, \frac{\pi(\mathcal{H}, \boldsymbol{y}')}{\pi(\mathcal{H}, \boldsymbol{y}^{[t]})}\right)$ ▷ Calculate acceptance ratio
    $r \sim U(0, 1)$ ▷ Sample random number from $[0, 1)$
    if $r < \alpha$ **then**
      $\boldsymbol{y}^{[t+1]} \leftarrow \boldsymbol{y}'$ ▷ Accept proposed point
    else
      $\boldsymbol{y}^{[t+1]} \leftarrow \boldsymbol{y}^{[t]}$ ▷ Refuse proposed, keep previous

4.4.1 Data

Synthetic dataset

In this work, we present a synthetic dataset with one feature x and targets $\boldsymbol{y} = \{y_1, y_2, y_3\}$, where $\boldsymbol{y}_F = \{y_3\}$ and the ground-truth distribution $p(\boldsymbol{y}_{\neg F} \,|\, \boldsymbol{y}_F, x)$ is known. Dependencies between the parameters and variables are illustrated by a Bayesian diagram in Fig. 4.2. We first sample three target variables of interest, $y_1^{y_3=0} \sim \alpha$, $y_2^{y_3=0} \sim 1-\alpha$, and $y_3 = 0$, where α is a bi-modal mixture of normal distributions, see Fig. 4.3a, and y_1, y_2 are normalized afterwards so that $0 \leq y_1^{y_3=0}, y_2^{y_3=0} \leq 1$ and $y_1^{y_3=0} + y_2^{y_3=0} = 1$. This gives us $P(y_1 \,|\, y_3 = 0)$ and $P(y_2 \,|\, y_3 = 0)$.

After that, the joint distribution is generated: for each instance,

$$\begin{aligned} y_1 &= y_1^{y_3=0} \cdot (1-p), \\ y_2 &= y_2^{y_3=0} \cdot (1-q), \\ y_3 &= y_1^{y_3=0} \cdot p + y_2^{y_3=0} \cdot q, \end{aligned}$$

where $p \sim \mathcal{N}(0.1, 0.1)$ and $q \sim \mathcal{N}(0.5, 0.2)$, respectively (taking the absolute value if negative is generated). This significantly shifts the distribution of the y_2 variable when compared to $y_2^{y_3=0}$, see Fig. 4.3b. The x feature is generated by adding noise to the parameter α and further linear transformation: $x = -20 \cdot (\alpha + \varepsilon) + 10$, $\varepsilon \sim \mathcal{N}(0, 0.1)$.

This synthetic dataset may be illustrated by a vegetation distribution example, where y_1, y_2, and y_3 variables correspond to snow, grass, and urban, respectively; each instance presents a tile on the earth's surface. As urban activity is more likely to be settled in grass type, snow and grass are not equally affected by the presence of humans. In this setting, we observe y_1 and y_2 in the presence of urban and are interested in evaluating

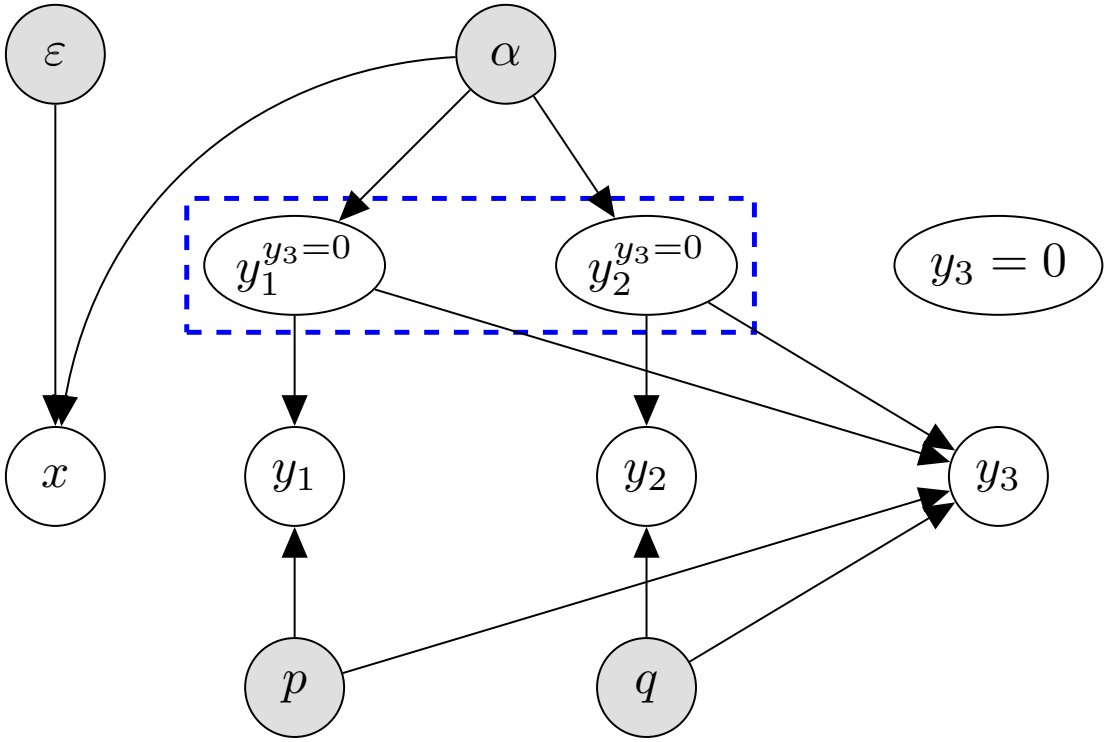
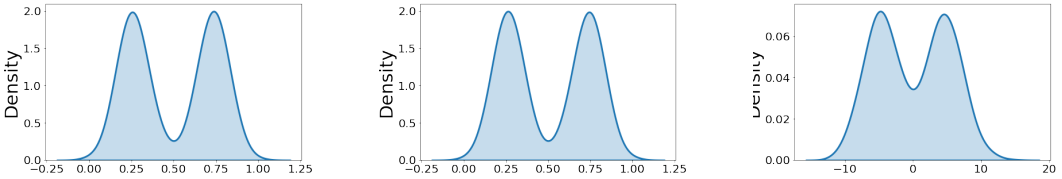
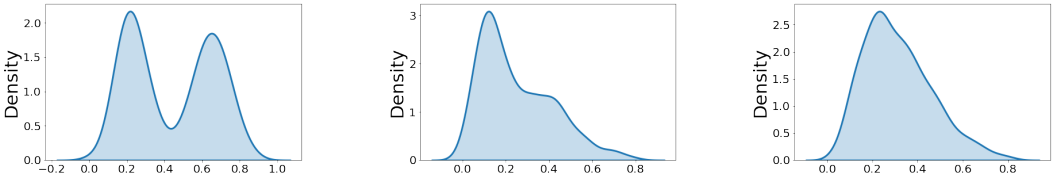


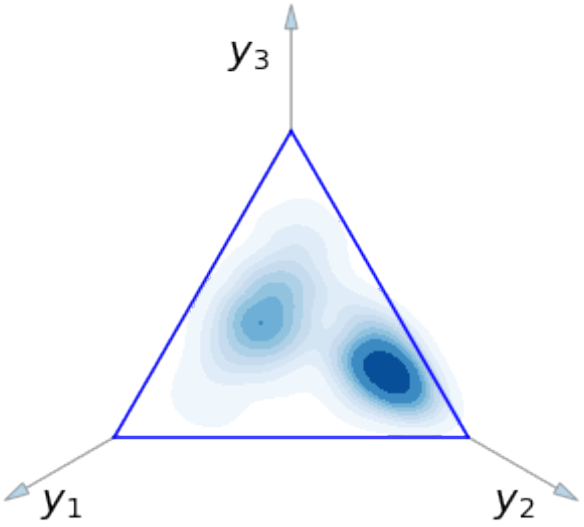
FIGURE 4.2: A diagram representing the generation of synthetic data. Grey nodes correspond to parameters with known distributions, and white nodes correspond to generated variables. While variables x, y_1, y_2, and y_3 are given for model training, the goal is to query $y_1^{y_3=0}$ and $y_2^{y_3=0}$ (highlighted with a dashed line).



(A) Ground-truth distributions of snow $P(y_1 \mid y_3 = 0)$ and grass $P(y_2 \mid y_3 = 0)$ types before adding urban; temperature $P(x)$.



(B) Observed distributions of snow $P(y_1)$, grass $P(y_2)$, and urban $P(y_3)$.



(C) Distribution of all vegetation types $P(y_1, y_2, y_3)$ on simplex, $y_1 + y_2 + y_3 = 1$

FIGURE 4.3: Synthetic dataset.

$P(y_1 \mid y_3 = 0)$ and $P(y_2 \mid y_3 = 0)$, i.e. vegetation distribution in the absence of urban.

Real-world benchmark data

Subsequently, we use three real-world multi-target regression datasets. A compositional Arctic lake dataset [Aitchison, 2005] describes the distribution of sand, clay, and silt (3 targets) in 39 water samples, on different depths (1 feature). The Concrete Slump dataset [Yeh, 2007] has three targets that describe three properties of concrete (slump, flow, and compressive strength) and seven features presenting concrete ingredients in 103 samples. The Energy Building dataset (Enb) [Tsanas and Xifara, 2012] has two targets, heating load and cooling load requirements of buildings (i.e. energy efficiency), and eight features presenting building parameters for 768 instances.

For evaluation on all three datasets, we split the data into train and test subsets (80:20, 5-fold cross-validation), and in the prediction phase provide explicitly the values of the first target $\boldsymbol{y}_F = [y_1]$. The metrics are calculated for the predicted targets $\boldsymbol{y}_{\neg F} = [y_2, \dots]$.

Vegetation data

Finally, we apply our method to a dataset describing the distribution of land cover globally to infer a possible vegetation distribution in the absence of urban activity (i.e. force the corresponding classes to 0 explicitly). The set of land cover classes we aim at predicting is derived from the Moderate Resolution Imaging Spectroradiometer MCD12Q1 dataset [Friedl et al., 2019] for the year 2018 and represents land cover as defined by the International Geosphere-Biosphere Programme cover classification scheme [Loveland and Belward, 1997].

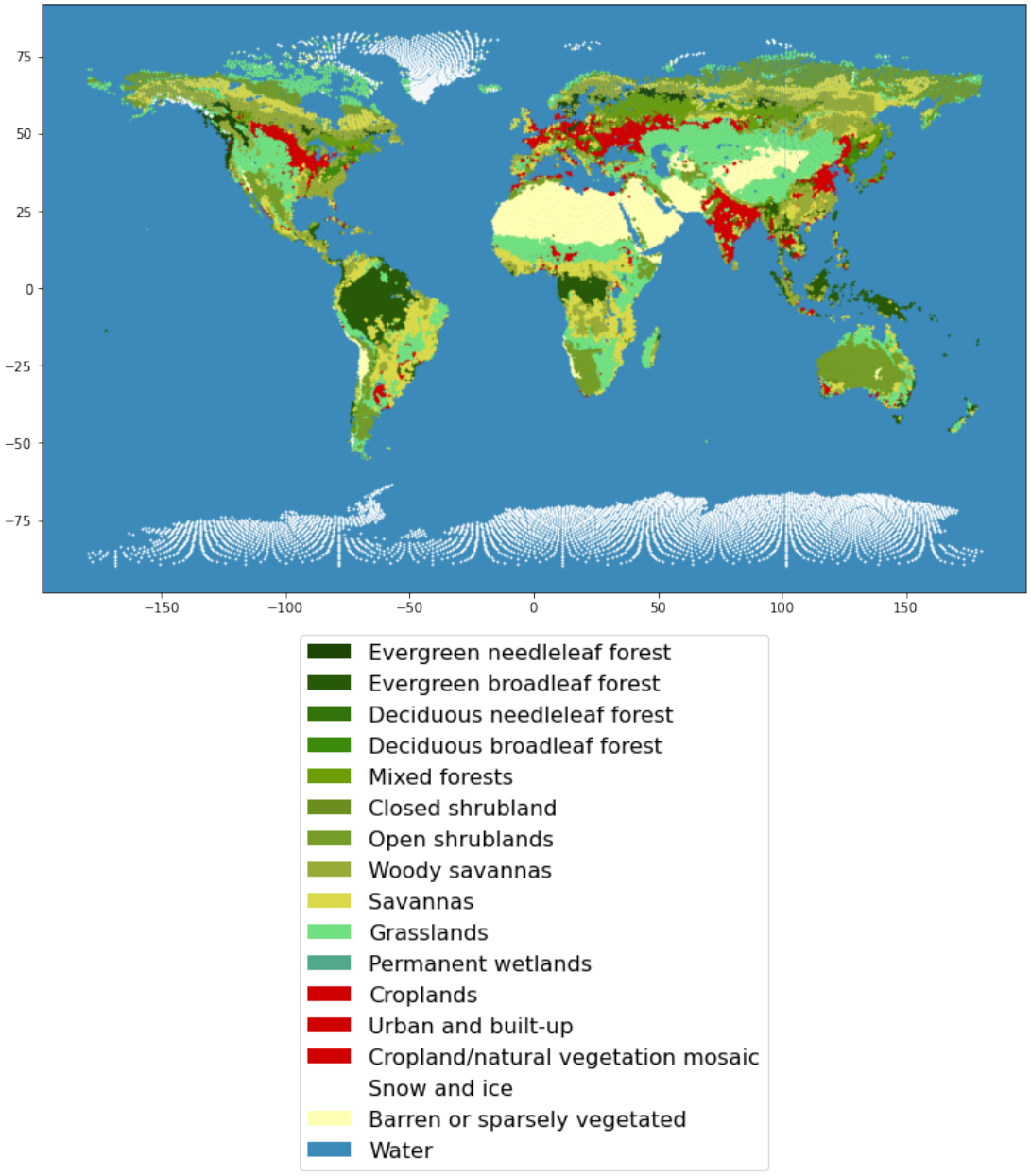


FIGURE 4.4: The predominant land cover class per global grid cell. In our prediction problem, each cell is represented by a categorical distribution over all types; only the maximum type within each cell is mapped.

We use 19 bioclimatic variables as predictive features, derived from the WorldClim database v2.0 [Fick and Hijmans, 2017]; these 19 variables represent ecologically relevant means, minima, and maxima in temperature and precipitation, averaged for the period 1970–2000. Both land cover targets and bioclimatic features were obtained from the Eco-ISEA3H database [Mechenich and Žliobaitė, 2023], a compilation of publicly-available Earth observation (EO) datasets characterizing global climate and biogeography.

The database is built on a geodesic discrete global grid system [Sahr et al., 2003], a systematic spatial framework of equal-area hexagonal cells. We used resolution 3H09, in which cells measure approximately 2600 km^2, and retained only terrestrial cells in our analysis (56,821 instances or approximately 28% of total cells globally). The proportions of each grid cell covered by each of the 16 land cover classes (summing to 1.0) serve as model output. The predominant cover class within each grid cell is mapped in Fig. 4.4; the 16 terrestrial classes (as well as water cover) are listed in the map legend.

We are interested in inferring the fractional distribution of natural land cover classes in the absence of three human-modified cover classes, namely croplands, urban and built-up lands, and cropland/natural vegetation mosaics (mapped together in red in Fig. 4.4).

4.4.2 Evaluation

In the synthetic dataset both ground-truth distributions $P(y_1 \mid y_3 = 0)$ and $P(y_2 \mid y_3 = 0)$ and observed distributions $P(y_1)$, $P(y_2)$, and $P(y_3)$ are known. The task of a model is to reconstruct the ground-truth distribution from the observed distribution and measure the distance between them. In the multi-target regression datasets, we provide values for the first target explicitly and evaluate the prediction of the other targets.

While the proposed algorithms, mhsRC and mhsERC, naturally provide a distribution per instance, we will use the means of these distributions to compare these approaches with the baselines.

As evaluation metrics, we use Mean Squared Error (MSE), Uniform Cost Function (UCF), and Wasserstein Distance (WD). Mean Squared Error calculates averaged squared loss, and, for the ground-truth $\{\boldsymbol{y}_i\}$ and predicted $\{\hat{\boldsymbol{y}}_i\}$, $i = 1, \ldots, N$,

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i)^2.$$

Uniform Cost Function [Burger and Lucka, 2014] may be considered as an analog of the 0/1 Loss for regression problems within a given neighborhood δ of the true values:

$$UCF(\delta) = \frac{1}{N}\sum_{i=1}^{N}\begin{cases}0 \text{ if } \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|_2 < \frac{\delta}{2},\\ 1 \text{ otherwise,}\end{cases}$$

where δ is an adjustable parameter. For the experiments, we take $\delta = 0.5$. Wasserstein distance is a distance function defined between probability distributions and may be seen as the minimum amount of "work" required to transform one distribution to another,

$$WD = \int_{-\infty}^{+\infty} |\hat{\boldsymbol{Y}} - \boldsymbol{Y}|,$$

where $\hat{\boldsymbol{Y}}$ and $\boldsymbol{Y}$ are cumulative distribution functions for the vectors $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$, respectively.

We compare the proposed inference method built on Regressor Chains with Random Forests as base estimators to several baselines. First, we compare to a Regressor Chain with direct order $[y_1, ..., y_L]$, when the target with fixed values (y_1) comes first in the chain, and thus it is possible to cascade the fixed values directly, without backward inference. Second, we evaluate several marginal models that don't take the joint constraint into account:

– The fixed targets are set to corresponding values, other targets are re-normalized, no trained model is used (redistrib.) – applicable only for the synthetic dataset;

– Ensembles of Regressor Chains (ERC);

– Multi-target Random Forests (mtRF);

– Individual single-target Random Forests (stRF);

where we plug in the fixed values *after* prediction and re-normalize the targets so that their sum is equal to one.

4.5 Results and discussion

Synthetic data

Table 4.1 shows the comparison of different methods for the synthetic data, where a model should 'uncover' the ground-truth distribution without urban activity. First, to support the choice of Regressor Chains for a predictive task, we evaluate performance of all methods in a standard setting when prediction from $\boldsymbol{x}$ to $\boldsymbol{y} = \{y_1, y_2, y_3\}$ is required. To this end, we perform 5-fold cross-validation and observe that Regressor Chains and Ensembles of Regressor Chains outperform single- and multi-target Random Forests.

Second, we compare empirically the predictions of $\hat{y}_1, \hat{y}_2$ when $y_3 = 0$ by the models listed above and ground-truth y_1, y_2 when $y_3 = 0$. We observe that the metrics values differ significantly for different chain orders and a chain [3,1,2] shows the best result: this is unsurprising as we plug in directly the constraint to the first model of the chain and further propagate its inference. For some of the orders ([2,1,3], [2,3,1]) the task is more difficult.

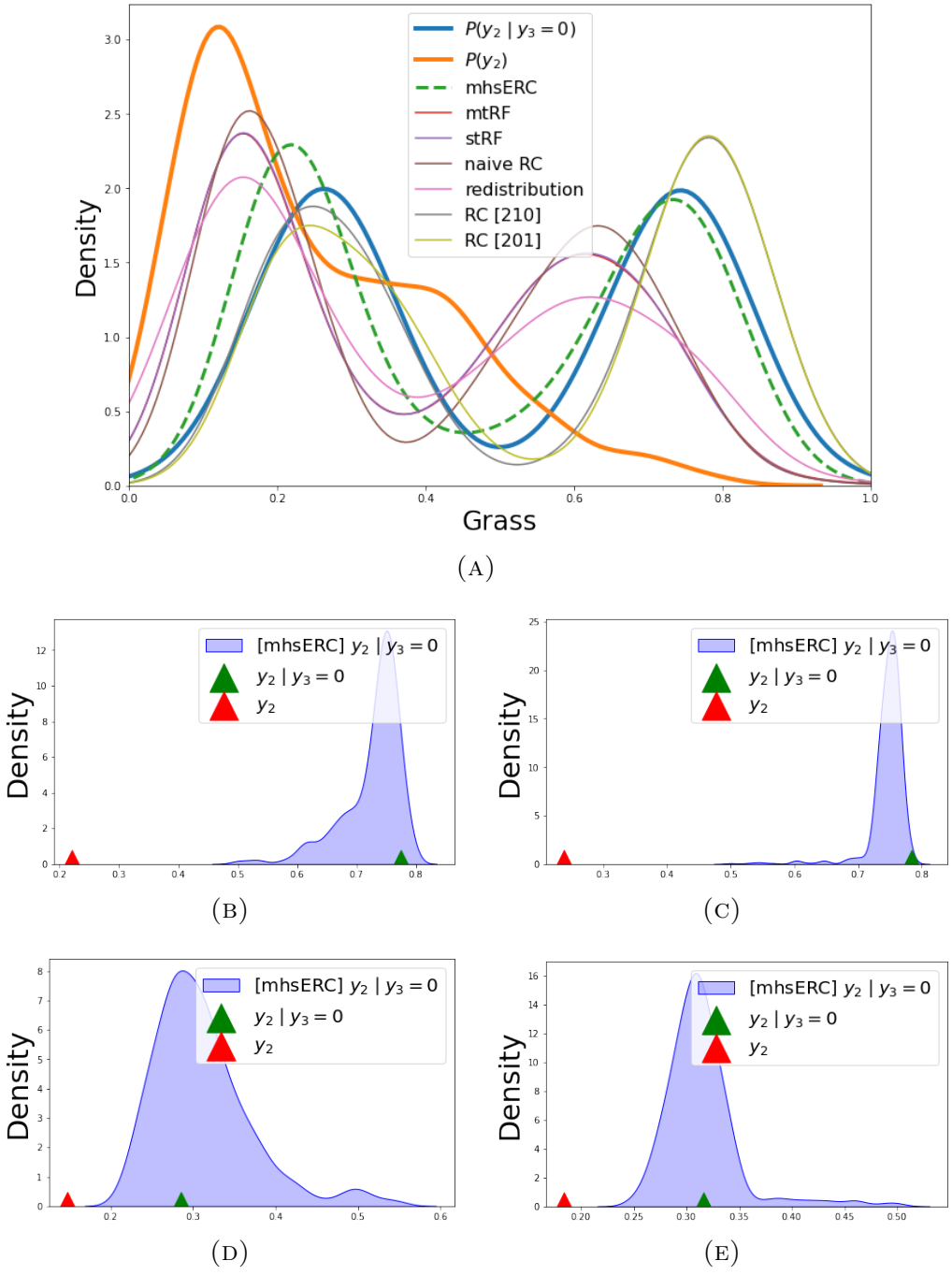


FIGURE 4.5: (A) Re-discovering of ground truth $P(y_2 \mid y_3 = 0)$ distribution in synthetic data. Note, that $y_1 \mid y_3 = 0$ is equal to $1 - y_2 \mid y_3 = 0$ (by nature of compositional data) so technically we are evaluating distributions of both targets. (B-E) Predicted per-instance distributions for four individual instances.

However, Metropolis-Hastings sampled Ensemble of Regressor Chains (mhsERC) containing all possible 6 chain orders shows high-performing results when compared to 'naive' models without joint inference. Fig. 4.5a also demonstrates graphically the resulting distributions. The inference of the mhsERC model is very close to the original bi-modal symmetric distribution of grass and snow. Examples of individual per-instance distributions of $\hat{y}_2 \,|\, y_3 = 0$ (for given $\boldsymbol{x}$) are presented in Fig. 4.5b–4.5e. Again, we see that the predicted distributions tend to center around the desired value of ground-truth $y_2 \,|\, y_3 = 0$.

Multi-target datasets with one target provided explicitly

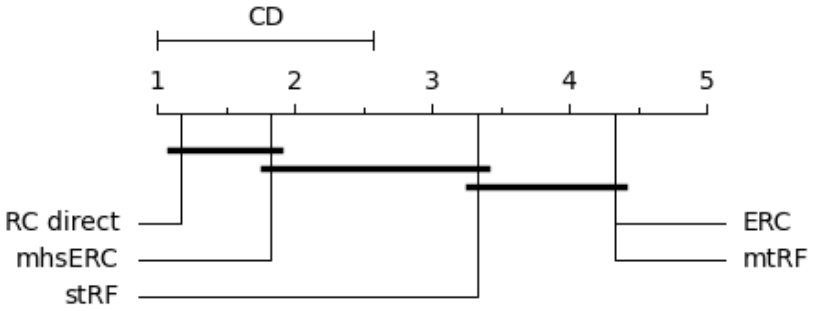
For three multi-target regression datasets, Arctic Lake [Aitchison, 2005], Slump [Yeh, 2007], and Enb [Tsanas and Xifara, 2012], the values of the target y_1 are provided explicitly, and other targets are to be predicted. The experiments are performed in a 5-fold cross-validation setting: models are trained on 80 percent of the data, and for 20 percent of the data values of the target y_1 are provided in the prediction phase. Table 4.2 shows the comparison of the newly proposed method, mhsERC, Regressor Chain with direct order $[y_1, ..., y_L]$ (when fixed values of y_1 are simply propagated via chain), and three marginal methods (ERC, mtRF, stRF). We observe that mhsERC obtains significantly better results than the marginal methods and close to the ones of Regressor Chains with direct orders with regard to all three metrics. The statistical significance is illustrated by the Friedman-Nemenyi diagrams in Fig. 4.6 for all three metrics, the mhsERC method ranked along with Regressor Chains with direct order and significantly higher than other methods.

TABLE 4.1: Synthetic data. First, results for classic cross-validated regression ($\boldsymbol{x} \to \boldsymbol{y}$) are provided to compare the performance of all models in a standard setting. Then, the models are evaluated for prediction with $y_3 = 0$ vs. ground truth $y_1, y_2 \,|\, y_3 = 0$; a smaller value is better. Best value in **bold**, second best value underlined. Orders of Regressor Chains (permutations of $1, 2, 3$) are given in square brackets.

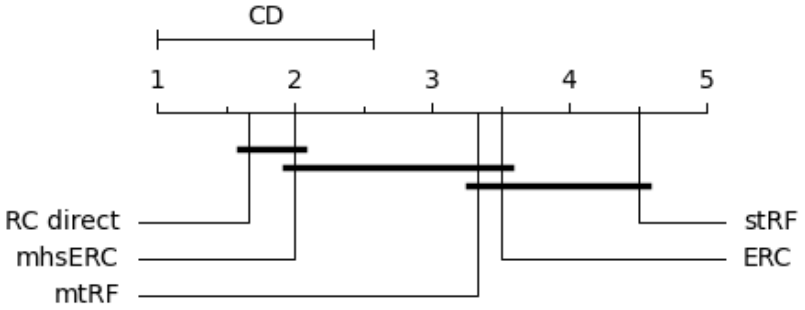
	$\boldsymbol{x} \to \boldsymbol{y}$	$\boldsymbol{x} \to y_1, y_2 \,\|\, y_3 = 0$		
Model	MSE	MSE	WD	UCF
mhsRC [1,2,3]	0.016	0.019	0.052	0.099
mhsRC [1,3,2]	**0.015**	0.011	0.038	0.073
mhsRC [2,1,3]	0.017	0.169	0.132	0.281
mhsRC [2,3,1]	0.017	0.029	0.114	0.272
mhsRC [3,1,2]	0.017	**0.007**	**0.016**	0.054
mhsRC [3,2,1]	0.017	0.011	0.039	0.085
mhsERC	**0.015**	0.010	0.027	**0.037**
stRF	0.018	0.018	0.115	0.172
mtRF	0.018	0.019	0.115	0.176
ERC	0.016	0.016	0.108	0.117
redistrib.	–	0.024	0.122	0.213
RC [3,1,2]	0.017	0.009	0.038	0.053
RC [3,2,1]	0.017	0.009	0.036	0.055

TABLE 4.2: Multi-target regression data, predictions when the first target is provided explicitly; smaller value is better. Best value in **bold**, second best value underlined.

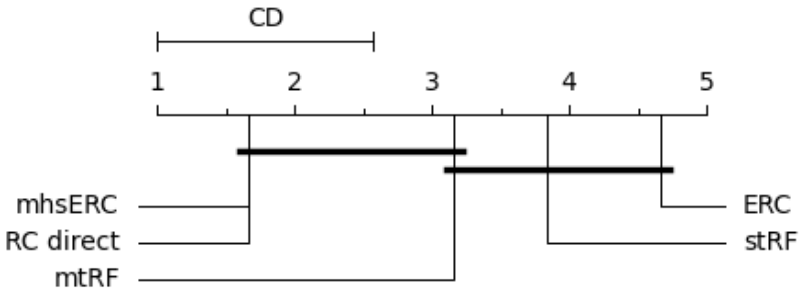
	Arctic Lake			Slump			Enb		
Model	MSE	WD	UCF	MSE	WD	UCF	MSE	WD	UCF
mhsERC	**0.002**	0.030	**0.000**	0.177	0.205	**0.806**	0.016	0.065	0.152
RC direct	**0.002**	**0.029**	**0.000**	**0.173**	**0.188**	0.825	**0.015**	0.068	**0.132**
mtRF	0.008	0.039	0.150	0.373	0.446	0.951	0.025	**0.063**	0.187
stRF	0.008	0.041	0.150	0.322	0.415	0.951	0.020	0.069	0.174
ERC	0.008	0.036	0.125	0.333	0.408	0.971	0.027	0.069	0.193



(A) MSE



(B) WD



(C) UCF

FIGURE 4.6: Friedman-Nemenyi diagrams comparing the ranking of the experimentally tested methods for multi-target regression data, predictions when the first target is provided explicitly. A lower rank is better, statistically undistinguishable methods are connected by a horizontal line.

Vegetation data

Prediction of vegetation from climate. First, we point out that Ensemble of Regressor Chains is a well-performing model for the prediction of vegetation types from climate. Land areas worldwide are highly imbalanced in terms of dominant land cover types and their mixtures. For example, deciduous needleleaf forest, the least represented of the 16 land cover types, is dominant over less than 0.0005% of the globe, whereas grassland is the most common, being the dominant land cover type on more than 20% of the planet's tiles; i.e. cells dominated by these two types of land cover have errors which vary up to 1000%. A standardized squared error metric would encourage the model to ignore the former since the squared error would be relatively minuscule. We thus choose Mean Absolute Error (MAE) as a metric to prevent the model from ignoring minority cover types and not penalize the model too hard for making non-zero estimates on minority cover types. The experiments are done under 10-fold cross-validation, and splits are designed to account for spatial correlations between neighboring grid cells and avoid information leakage between train and test partitions.

While single- and multi-target Random Forests show the best performance in cross-validation experiments with regard to MAE, we are not aware if it is possible to force these models to modify particular targets in the prediction phase. Ensembles of Regressor Chains run only slightly worse, and we propose a natural mechanism to impute particular targets with particular values for any chain in the ensemble, while other targets take this value into account.

No urban activity. We set the values of three variables (croplands, urban and built-up, cropland/natural vegetation mosaic) to zero and apply the newly designed method, mhsERC. Fig. 4.7 demonstrates the predicted vegetation distribution in the absence of human activity for two densely populated large areas, Europe and South Asia, as well as for four selected

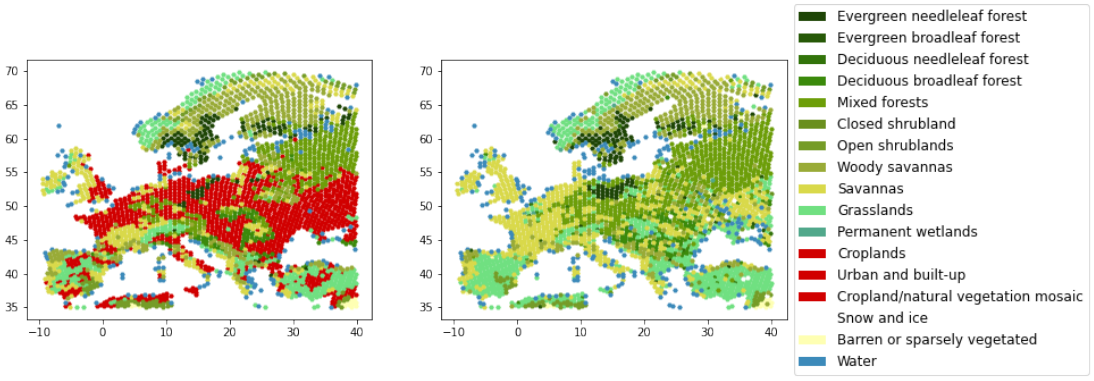
TABLE 4.3: 10-fold cross-validation, comparing different multi-label models for vegetation prediction. Best value in **bold**.

Model	MAE	std
stRF	0.047	5.07e-05
mtRF	**0.045**	**4.55e-05**
RC (with RFs)	0.061	6.12e-05
ERC (with RFs)	0.050	4.87e-05

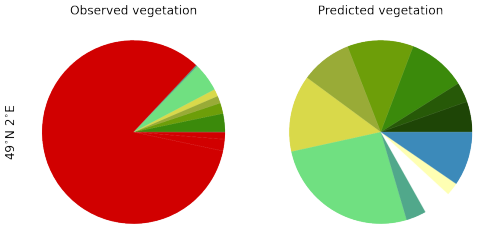
small areas within these two. Subjectively, the results appear visually plausible. There are no noticeable anomalies. This adds support to our claim that our method can be used flexibly for real-world tasks. Although, inherently, there can be no ground-truth evaluation for such a task, we can take confidence in the relatively high performance on the synthetic and non-hypothetical real-world tasks investigated earlier.

4.6 Conclusion

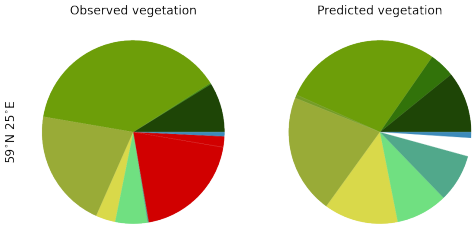
In this chapter, we introduce a particular challenge in the multi-output setting: some targets come observed in the inference phase, and the option to re-train models is not available, yet the outputs must be nevertheless provided under a joint constraint. We study this setting in the context of Regressor Chains and adapt them to the given scenario by employing Metropolis-Hastings backward inference hereby enabling to leverage pre-trained models without re-training under different chain orders. Naturally, the proposed method provides an empirical distribution for each prediction instead of a single expected value. We show that the result can be adapted and evaluated in several use cases. It performed out-competitively on synthetic data (with controlled ground truth) as well as in a real-world challenge of predicting potential vegetation in the absence of human activity. We conclude that the proposed method successfully solves the task, allowing flexibility and applicability of Regressor Chains algorithms beyond their predictive performance in standard multi-target regression settings.



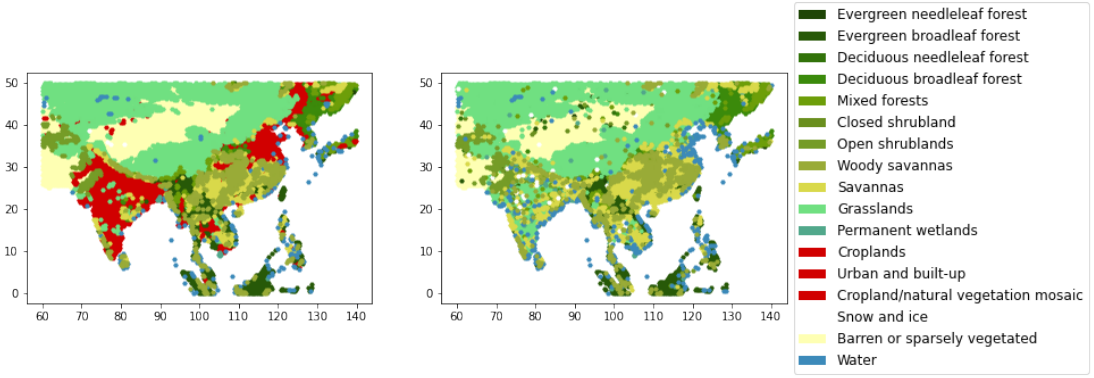
(A) Europe



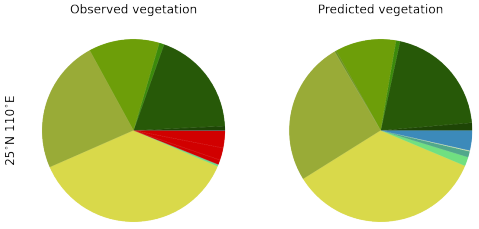
(B) Europe, 49°N 2°E



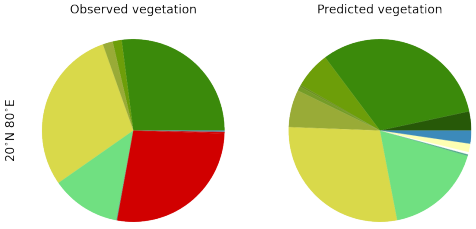
(C) Europe, 59°N 25°E



(D) Asia



(E) Asia, 25°N 110°E



(F) Asia, 20°N 80°E

FIGURE 4.7: Plots 4.7a and 4.7d show the prevalent vegetation type per global grid cell when human activity is present (left) and when it is hypothetically absent (right; where we fixed urban to 0 and re-predicted). Plots 4.7b, 4.7c, 4.7e, and 4.7f show vegetation distribution per grid cell, with (observed) and without (predicted) human activity.

Chapter 5

Missing value imputation as a multi-label task

Missing values are a common problem in data science and machine learning. Removing instances with missing values is a straightforward workaround, but this can significantly hinder subsequent data analysis, particularly when features outnumber instances ($p \gg N$). There are a variety of methodologies proposed in the literature for imputing missing values, most of them proceeding iteratively in a coordinate-ascent scheme. Denoising Autoencoders, for example, have been leveraged efficiently for imputation. But neural-network approaches have been relatively less effective on smaller training sets. To this end, we propose Autoreplicative Random Forests (ARF) via a multi-output learning approach, which we introduce in the context of a framework that may impute via either an iterative or procedural process. Experiments on several low- and high-dimensional datasets show that ARF exhibits better imputation performance than its competitors. We also propose ARF in a probabilistic framework, where the confidence values are provided over different imputation hypotheses, therefore

maximizing the utility of such a framework in a machine-learning pipeline targeting predictive performance.

5.1 Introduction

Missing values are a common problem and an important issue in the domain of data science and machine learning. Most off-the-shelf statistical and machine learning methods cannot learn from data containing missing values, and such values must be imputed, or entire instances removed, prior to analysis. When many values are missing, considering only instances with complete information (no missing values) can lead to a significant loss of information or even an empty dataset.

Indeed, a special challenge is when missing values occur in many or most training samples. This is more likely to occur when there are sufficiently more features (p) than samples (N), i.e. when $p \gg N$, which means that removing samples amplifies the imbalance. Examples of this scenario include medical and bioinformatics arrays, classification problems in astronomy, tool development for finance data, and weather prediction [Johnstone and Titterington, 2009].

Denoising Autoencoders (DAE) is a state-of-the-art method for missing value imputation [Vincent et al., 2008] which treats missing values as 'noise'. They may be trained only on complete data (ignoring missing values) or with missing values randomly imputed and then iteratively re-trained, and data re-imputed, successively until convergence, e.g. [Seo et al., 2022; Wright, 2015]. PCA has been used in a similar way [Dray and Josse, 2014]; indeed, PCA can be seen as a special (linear case) of auto-encoding.

Another well-known approach is Multiple Imputation with Chained Equations (MICE) [van Buuren and Groothuis-Oudshoorn, 2011]. MICE also imputes the missing values randomly, so as to have a complete 'training set',

then successively trains and maps values (imputing) in a column-wise leave-one-out scheme; and repeats this process until some convergence criterion is met. This whole process can be repeated to obtain multiple candidate values for imputation (hence the 'multiple imputation'; but this ensemble-like approach can also be applied generally with other model-based imputation algorithms). Indeed, the well-known MissForest imputer [Stekhoven and Bühlmann, 2011] can be seen as a special case of MICE (with Random Forest chosen as a base learner to do the column-wise training and imputation).

In this work, we combine the best of these two worlds: what we call the *procedural* approach (e.g. DAE) which is a one-shot imputation (each value imputed only once); vs what we call the *iterative* approach, (e.g. MICE) in which values are successively re-imputed until convergence. We first propose a unifying framework for missing value imputation within which to set these two strategies.

Building in this framework, we produce a novel method, Autoreplicative Random Forests (ARF) that can be carried out in either a procedural or iterative fashion. To the best of our knowledge, using multi-label models in an autoreplicative way without explicit encoding has not been widely studied in the literature. We empirically demonstrate the advantages of this method, including specifically its ability to effectively impute data of a small sample size.

Furthermore, while there exist many methods for missing value imputation, we observe from the literature that the overwhelming majority do not keep the information about the uncertainty of imputation. This means, that once imputed, all values are treated equally and independently from the fact if they were observed or imputed. This may bias the forthcoming analysis. To this end, we propose a probabilistic imputation method, distributional iterative Autoreplicative Random Forests (ditARF), that not only imputes the missing values but also provides their uncertainty.

In this work, we consider specifically categorical features as a multi-label multi-output classification problem. In general, we expect that this strategy may be generalized to continuous outputs as well. However, preliminary results show that some deeper studies for regression imputation are required. This is not surprising as this is often seen in multi-output literature: classification methods are not always straightforward to transfer to regression prediction. As a compromise solution, we may suggest a discretization technique where the continuous values are split into bins and a problem is further tackled as a classification task. Despite the apparent loss of information, such an approach is well known for its accuracy improvement [Dougherty et al., 1995]. Like the methods we compare to, we assume that data is Missing Completely At Random (the 'missingness' occurs entirely independently from feature and class values) [Santos et al., 2019].

To sum up, we contribute to the state-of-the-art with the following:

- We propose an efficient method, Autoreplicative Random Forests (ARF), in three different variants, for missing value imputation;
- We propose a general framework to be used to illustrate and compare ARF vs a variety of competitors; namely one of procedural vs iterative imputation strategies;
- We propose Distributional ITerative ARF (ditARF); a probabilistic approach that may provide confidence over imputation hypotheses, both under the assumptions of marginal (individual) vs joint (combination) imputations;
- We show the effectiveness of ARF (in particular vs deep learning methods) on both standard-dimensional and high-dimensional low-sampled data ($p \gg N$).
- We provide a comprehensive experimental study in which we assess the capabilities of the missing value imputation technique. Moreover,

we also compare the difference, in terms of accuracy, between a generic classifier learned from a ground-truth dataset and an imputed dataset.

The rest of the chapter is organized as follows. Together with summarizing the background and related work, we present an imputing framework unifying different methods in Section 5.2 and then expand it with a group of new methods, pARF, itARF, and ditARF, in Section 5.3. The results and their discussion as well as complexity analysis are described in Section 5.4. In Section 5.5, we draw conclusions and future work.

5.2 A General Framework for Missing Value Imputation

In this section, we describe a general framework unifying different approaches to missing value imputation; namely the procedural vs iterative strategies.

5.2.1 Preliminaries

In the scope of this chapter, we focus on missing values in the features, and to this end, omit targets Y and define a dataset $\mathcal{D} = \{X \cup \tilde{X}\}$, consisting of N rows (instances) and p columns (features); with missing values denoted as variables $\tilde{x}_{i,j}$. For example $(N = 5, p = 3)$,

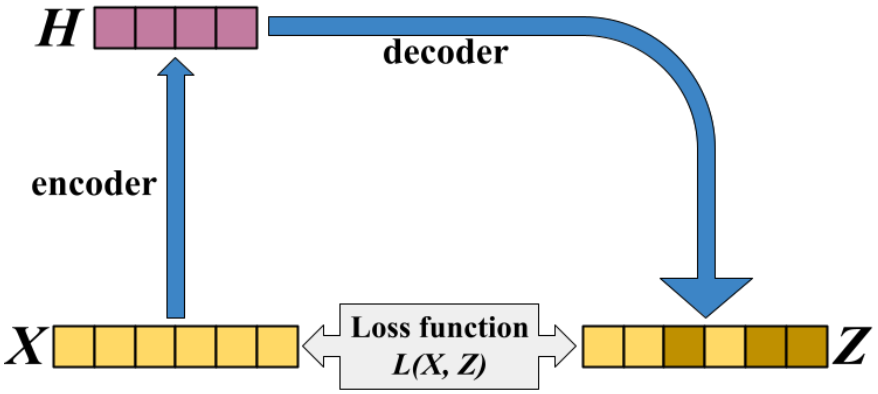
$$\mathcal{D} = \begin{bmatrix} \tilde{x}_{1,1} & \tilde{x}_{1,2} & x_{1,3} \\ x_{2,1} & \tilde{x}_{2,2} & x_{2,3} \\ \tilde{x}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix}$$

where we have $x_{i,j}$ to stand for an existing value in the dataset (realization of a random variable $X_{i,j}$), and $\tilde{x}_{i,j}$ to imply that such a value is not yet known/realized (i.e. it is missing). When there is no need for this distinction (e.g. referring to a particular instance of p features with both missing and observed values), we use $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^{N}$. We denote $\dot{\boldsymbol{x}}$ specifically for instances where missing values have been imputed. A model h (e.g. Autoencoder, Random Forest, ...) is parametrized by $\boldsymbol{\theta}$, and $p_t(\dot{\boldsymbol{x}}_i^{[t]} \mid \dot{\boldsymbol{x}}_i^{[t-1]}, \boldsymbol{\theta})$ is the probability that random vector $\tilde{\boldsymbol{x}}_i$ takes value $\dot{\boldsymbol{x}}_i^{[t]}$ at iteration t.

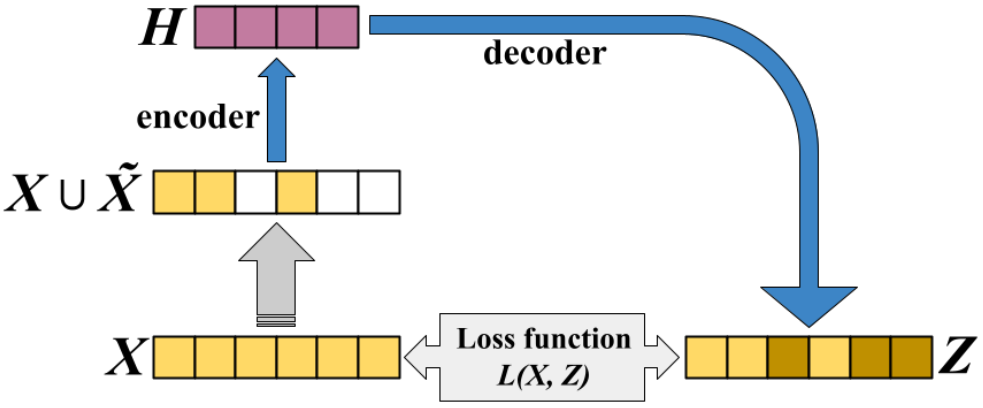
5.2.2 Procedural vs Iterative approaches

In this work, we particularly distinguish between the ways how the missing values may be imputed: procedurally or iteratively. **Procedural** methods impute values only once, based on the observed values. **Iterative** methods first impute values randomly and then update these imputations until some convergence criterion is met. Note, that a method belonging to one of these families, might be easily adaptable for another one.

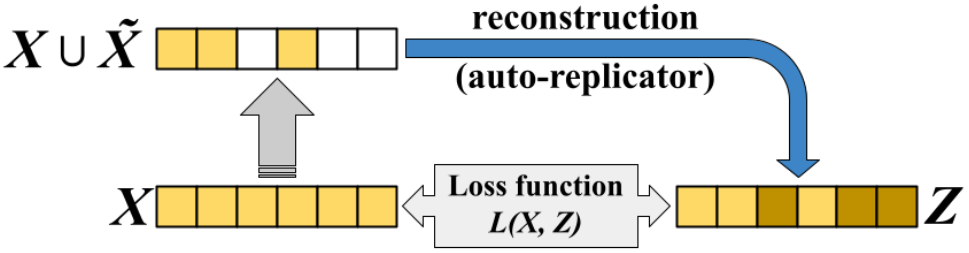
The procedural methods range from rather simple ones such as replacement with the mean, mode, or median statistics [Little and Rubin, 2019] to more sophisticated machine learning techniques such as k-Nearest Neighbours (kNN) [Schwender, 2012] and Cascade Imputation (CIM) [Montiel et al., 2018]. While the kNN method processes the data row-wise, i.e. extracting information from the k instances that are most similar to the one whose missing values need to be replaced, CIM first rearranges data, so that missing values may be imputed block by block. In the deep learning domain, the aforementioned Denoising Autoencoders may be successfully used for imputation after training on the complete instances. Classical Autoencoders implemented within neural networks architecture consist of Encoder and Decoder structures as illustrated in Fig. 5.1a. While the inner structure of hidden layers can be very different, the typical common property is having at least one narrow middle layer H to restrict the model to learning only



(A) Classic Autoencoder



(B) Denoising Autoencoder



(C) Autoencoder without an explicit encoding

FIGURE 5.1: An illustration of (A) Classic Autoencoder (with hidden representation H), (B) Denoising Autoencoder (input is corrupted with noise or missing values as $X \cup \tilde{X}$ before encoding), and (C) Autoencoder without an explicit encoding (as we use in our work). In all cases, the goal is to minimize the difference between input X and its reconstruction Z.

important information from the data. Optimizing hidden layers implies a search for some inner patterns in the data. For Denoising Autoencoders, a widely-used step in the learning phase is inducing artificial missingness into complete rows and training the model to reproduce the original input (see Fig. 5.1b). A general schema for the procedural methods is given in Algorithm 4. The following example below illustrates one-shot row-wise procedural imputation (blue represents training samples):

$$\begin{bmatrix} \tilde{x}_{1,1} & \tilde{x}_{1,2} & x_{1,3} \\ x_{2,1} & \tilde{x}_{2,2} & x_{2,3} \\ \tilde{x}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix} \Rightarrow \begin{bmatrix} \dot{x}^{[1]}_{1,1} & \dot{x}^{[1]}_{1,2} & x_{1,3} \\ x_{2,1} & \dot{x}^{[1]}_{2,2} & x_{2,3} \\ \dot{x}^{[1]}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix}$$

The approach of iterative imputation takes the general schema of coordinate ascent, with special cases including expectation maximization (EM) and classification maximization (CM, i.e. 'hard EM') [MacKay, 2003; Dempster et al., 1977] where the expectation step E is replaced by a hard classification step (an actual value is imputed), and the maximization step M refers to training. Inspired by this idea, many missing value imputation algorithms start from a random or data-driven (mode, mean, median, ...) initial imputation so that any classifier can be trained on the entire dataset. Next, the missing values are predicted and the model is relearned after every imputation. This process is repeated until some convergence criteria are met. Algorithm 5 summarizes this schema and the following example illustrates the approach:

$$\begin{bmatrix} \tilde{x}_{1,1} & \tilde{x}_{1,2} & x_{1,3} \\ x_{2,1} & \tilde{x}_{2,2} & x_{2,3} \\ \tilde{x}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix} \Rightarrow \begin{bmatrix} \dot{x}^{[0]}_{1,1} & \dot{x}^{[0]}_{1,2} & x_{1,3} \\ x_{2,1} & \dot{x}^{[0]}_{2,2} & x_{2,3} \\ \dot{x}^{[0]}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix} \Rightarrow \ldots \Rightarrow \begin{bmatrix} \dot{x}^{[t]}_{1,1} & \dot{x}^{[t]}_{1,2} & x_{1,3} \\ x_{2,1} & \dot{x}^{[t]}_{2,2} & x_{2,3} \\ \dot{x}^{[t]}_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix}$$

Algorithm 4 General framework for procedural imputation

procedure PROCEDURAL IMPUTATION($\mathcal{D} = \{X \cup \tilde{X}\}$)
  $h \leftarrow \mathsf{Train}(\{\boldsymbol{x}\})$ ▷ Train the model with complete data
  $\{\dot{\boldsymbol{x}}\} \leftarrow h(\{\tilde{\boldsymbol{x}}\})$ ▷ Predict the missing values with h

Algorithm 5 General framework for iterative imputation

procedure ITERATIVE IMPUTATION($\mathcal{D} = \{X \cup \tilde{X}\}$, ε)
  $\{\dot{\boldsymbol{x}}\}^{t=0} \leftarrow \mathcal{R}(\tilde{x})$ ▷ Random imputation of missing values
  while $\Delta_{imp} > \varepsilon$ **do**
    $h^t \leftarrow \mathsf{Train}(\{\boldsymbol{x}\} \cup \{\dot{\boldsymbol{x}}\}^{t-1})$
    $\{\dot{\boldsymbol{x}}^t\} \leftarrow h^t(\{\dot{\boldsymbol{x}}\}^{t-1})$
    $\Delta_{imp} \leftarrow \mathsf{convergence}(\{\dot{\boldsymbol{x}}\}^t, \{\dot{\boldsymbol{x}}\}^{t-1}, h^t)$ ▷ Compute convergence criteria

TABLE 5.1: Some example methods as specific parametrizations of a general framework where Strategy CW = column-wise, RW = row-wise, BW = block-wise; p = procedural, it = iterative; SO = single-output, MO = multi-output. SI = single(standard)-imputation, MI = multiple imputation where '-' indicates that it could be implemented, but we are not aware of any reference doing so; with Ensemble (MICE) or via a predictive posterior Distribution ($\tilde{\boldsymbol{x}} \sim p(\cdot \mid \dot{\boldsymbol{x}})$) being options.

Method	Type	SO/MO	Strategy	MI	Reference
Mode	p	SO	CW	No	[Little and Rubin, 2019]
kNN	p	MO	RW	No	[Schwender, 2012]
MICE	it	SO	CW	Ens.	[van Buuren and Groothuis-Oudshoorn, 2011]
CIM	p	SO	BW	-	[Montiel et al., 2018]
DAE	p	MO	RW	-	[Vincent et al., 2008]
DAE	it	MO	RW	-	[Seo et al., 2022]
PCA/SVD	it	MO	RW	-	[Dray and Josse, 2014; Troyanskaya et al., 2001]
ARF	p	MO	RW	-	This work
ARF	it	MO	RW	-	This work
ditARF	it	MO	RW	Dist.	This work

Denoising Autoencoders, mentioned before in a procedural setting, may be also implemented as an iterative method, starting from randomly imputed missing values and then iteratively re-imputing the missing data until convergence is reached, e.g. [Dray and Josse, 2014; McCoy et al., 2018]. Principal Component Analysis (PCA) is also widely used as an imputation method and may be considered as a version of Autoencoder with a linear activation function. PCA may be also considered as a special case of Single Value Decomposition (SVD) [Troyanskaya et al., 2001]. The latter calculates the k most significant eigenvectors and then imputes the missing values using a low-rank SVD approximation estimated by an Expectation-Maximization algorithm.

All iterative methods listed above (DAE, SVD, PCA) may be considered multi-output predicting models, as they impute all missing values simultaneously. Oppositely, Multivariate Imputation by Chained Equations (MICE) is also iterative but single-output, as it processes features consequently in a leave-one-out manner. The MICE method is very flexible with regard to the base model, i.e. any per-feature estimator is possible. The MICE method is commonly used for different types of data and, in particular, clinical data, and can be considered state-of-the-art for missing value imputation, but we have not found substantial evidence of using the MICE method for high-dimensional datasets, as the computational cost drastically increases in this setting. The CIM method mentioned above may be considered a procedural version of MICE. Table 5.1 summarizes the characteristics of the methods discussed.

We would also like to mention the work [Wolputte and Blockeel, 2020] which uses a Random-Forest-based predictor to perform imputation in the prediction phase assuming complete data in the training phase. We think that with some extra effort, one could adapt this method to the setup described above and thus incorporate it into the framework, we leave this question for future research.

We need to mention also the idea of applying Gaussian Processes for missing value imputation proposed in [Jafrasteh et al., 2023]. Gaussian Processes are non-parametric models and output predictive distributions for target variables, which can be also considered as uncertainty estimates. Sparse Gaussian Processes have been applied for missing value imputation leveraging the idea of the MICE method where the features are processed one by one in a cascaded fashion. While obtaining promising results, the proposed method has a much higher computational cost both for training and prediction than other baselines including computationally expensive MICE. Also, in [Jafrasteh et al., 2023], Gaussian Processes are applied for continuous variables and assume a normal distribution for each which may not always be the case.

Obviously, multiple variations of the methodologies shown above can be discussed. Although we believe that these are out of the scope of this work, we think that is worth sharing some insights. In procedural imputation, the main characteristic is that the imputation is only done once for each missing value. Following this idea, we could introduce the *row-wise imputation* in which we increasingly impute the missing values through time and we relearn the imputation model after each imputation. Following this idea, we would add more true values to the training dataset after every imputation, but, at the same time, we might supply incorrect imputations to the model as ground truth. Similar to the previous approach, there is the *column-wise* imputation that matches the MICE imputation strategy.

5.2.3 Other Framework parameters

Estimator

For the MICE method, any single-output estimator can be used. As a default parameter, we use Random Forests (of 10 trees each) as they proved to be a robust and stable method, though any other classifier may be provided

manually to the framework. Among multi-output methods, we propose including Autoencoders and PCA as a standard choice and Autoreplicative Random Forests as a novelty (see Section 5.3).

Initial imputation

For iterative methods, an initial pre-starting imputation is needed. There are several possibilities for that: impute with modes of the values or impute randomly with the observed values with a uniform distribution or distribution taken from the observed values. We use random imputation with uniform distribution over the observed values as a default parameter.

Number of iterations

For iterative methods, the re-predictive process stops when the convergence is reached. We measure convergence as a difference between new and preceding imputations. If this difference is smaller than a provided parameter ε, set as default to 0.005, then the last imputed dataset is returned as the final estimation. However, to keep the overall complexity feasible, we provide a maximum number of iterations parameter, set as default to 10.

5.3 Proposed Approach: Autoreplicative Random Forests

Following the description of the general imputation framework, we want first to introduce a new imputation approach, Autoreplicative Random Forests, and second to propose its distributional extension.

5.3.1 Autoreplicative Random Forests

Although apparently largely overlooked in the literature, we have noticed that any other model designed for multi-output, multi-label, prediction can be used instead of a neural network as an Autoencoder. One such example is a combination of Decision Trees [İrsoy and Alpaydin, 2016] where the first Decision Tree is used as an encoder, and the second one is used in a vice versa manner as a decoder. Meanwhile, this idea can be simplified even more: in our approach, we will use a multi-output Random Forest as an estimator. Random Forests have been selected since they naturally are multi-label and multi-class classifiers and they proved to be competitive and robust classifiers in several works [Wood et al., 2023]. Such an approach can facilitate the optimization process for the model on data containing a small number of samples, and at the same time, Decision Trees and Random Forests are both efficient and simple to understand and interpret. To the best of our knowledge, this simple but efficient idea has not been well studied in the literature. We argue, that however it deserves attention and can be further investigated. Applying this idea, we suggest further Autoreplicative Random Forests.

It is worth noting, that while we choose Random Forests as a well-known and stable multi-label method with good performance, this idea may be developed by using other multi-label methods, such as e.g. Classifier Chains [Read et al., 2011], Multilabel k Nearest Neighbours [Zhang and Zhou, 2007], Random k-Labelsets [Tsoumakas and Vlahavas, 2007], Conditional Dependency Networks [Guo and Gu, 2011].

The setting is illustrated in Fig. 5.1c. In the procedural approach, we first select complete cases X of the entire dataset $\mathcal{D} = \{X \cup \tilde{X}\}$, corrupt it manually with missing values (uniformly distributed, following the proportion of missing values in the original dataset), and train an Autoreplicative Random Forest to reproduce $Z \sim X$, i.e. fill missing values by minimizing loss

function between Z and X. Then the fitted model can be used to replace actual missing values. In the iterative version, values should be first imputed randomly, then a Random Forest is re-trained in an iterative manner, on iteration t receiving $\mathcal{D} = \{X \cup \tilde{X}\}$ as an input, learning to reproduce $Z \sim \dot{X}^{t-1}$ as output and storing prediction $\dot{X}^t$ as a new imputation.

5.3.2 Distributional Iterative ARF (ditARF)

Missing value imputation does not always result in optimal accuracy when that imputed dataset is used to train a predictive model, because imputation accuracy and predictive accuracy are separate goals. The inevitable imperfections introduced by the imputation process and, more importantly, the inability to distinguish between imputed (estimated) and true values at training time are two important reasons for this.

To address this issue, methods such as MICE were proposed with the technique of 'multiple imputation', that is repeating the imputation several times independently (essentially, bootstrapping) in order to obtain multiple plausible values and run further analysis on these datasets.

Here we propose a distributional variant of ARF (ditARF) which provides a probability distribution instead of a single imputed value, for each missing value. Thus, encapsulating and expressing the uncertainty regarding such an imputation.

As occurs in virtually every ensemble method, the votes of the different base classifiers can be treated as probabilities for the predicted instance. We leverage this idea and introduce ditARF. Similar to itARF introduced in Section 5.3, ditARF learns and predicts missing values iteratively. However, at every iteration, the instances are weighted by the output joint probability. The main goal of ditARF is to provide statistical information along with the predicted missing value. To that end, we approximate the true joint distribution.

Formally, a multi-output Random Forest predicts $p(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) = \prod_{j=1}^{p} p(\tilde{x}_j \mid \boldsymbol{x})$. The imputed missing value can be produced equivalently as $\dot{x}_j = \text{argmax}_j\, p(\tilde{x}_j \mid \boldsymbol{x})$. However, this makes the assumption that each imputed value is conditionally independent of the others for a given instance; which may not be the case. In certain application domains such as medicine, it may be a critical mistake to make this assumption [Gerych et al., 2021].

Consider the example of Fig. 5.2, where the joint distribution $p(\tilde{\boldsymbol{x}})$ only gives nonzero probability to two values ($x_1x_2 = 00$ and $x_1x_2 = 11$), yet the marginal probability (as would be estimated by Random Forest) indicates the equal probability for all combinations (00, 01, 10, 11). This means that even though the dataset does not contain values for two of the possible combinations, a Random Forest would produce value combinations that could never exist in real data.

For example, the Label Powerset (LP) method [Tsoumakas and Katakis, 2007] transforms each combination of output values into a unique class and thus naturally models the labels jointly. However, such an approach could not be applied in the iterative setting as initial imputation creates value combinations that may not exist in the data while closing the opportunity to learn other possible combinations in future iterations. We can imagine adapting, for example, a less strict and more generalized version of the LP approach, the Random k-Labelsets method [Tsoumakas and Vlahavas, 2007], to tackle this issue, as well as inducing some randomness at each iteration. However, these possible solutions are out of the scope of this work and we leave them for future research.

In the discrete case that we study, we are implicitly providing estimates of a posterior mode; either on the marginal distribution,

$$\tilde{x}_j = \underset{\tilde{x}_{i,j}}{\text{argmax}}\, p(\tilde{x}_j \mid \boldsymbol{x}) \tag{5.1}$$

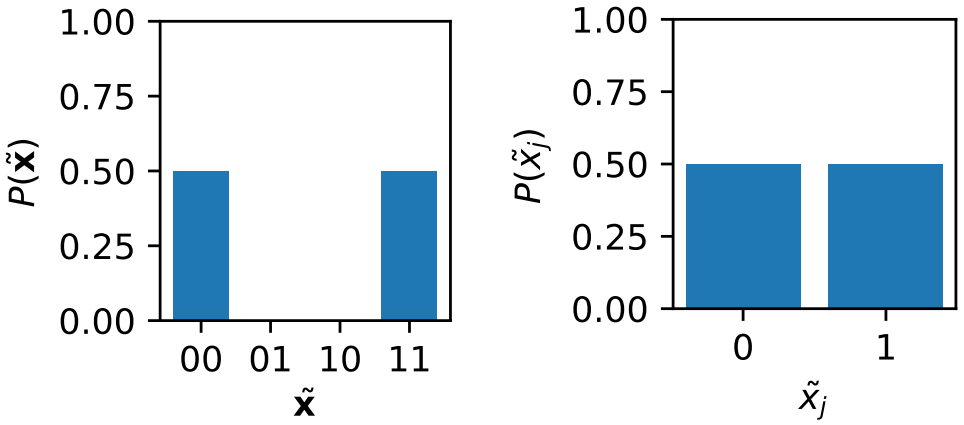


FIGURE 5.2: Illustrating the difference between the joint and marginal distributions of two binary missing-value variables $\tilde{\boldsymbol{x}} = \{\tilde{x}_1, \tilde{x}_2\}$ of an instance. The marginal distribution (the same distribution covers both variables, having been marginalized from the joint) indicates that all combinations of values 0 and 1 are equally likely, even though only two such combinations would occur. This indicates the potential importance of joint modeling.

and/or the joint distribution $\tilde{\boldsymbol{x}} \sim P$,

$$\tilde{\boldsymbol{x}} = \underset{\tilde{\boldsymbol{x}}}{\operatorname{argmax}}\, p(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}). \tag{5.2}$$

In the following, we develop a probabilistic framework for ARF, which we call ditARF (distributional iterative Autoreplicative Random Forests), which provides an estimate of these distributions.

The proposed solution is closely related to other well-known iterative methods such as the Expectation Maximization (EM) algorithm [Dempster et al., 1977] and to more general coordinate-ascent methods [Wright, 2015]. Such methods find the maximum likelihood parameters for the corresponding model based on data. In the case of the EM, it can be used to fit a mixture of Gaussian distribution models while the coordinate-ascent method just performs a linear optimization in the log-likelihood function by iteratively learning and predicting data. DitARF also maximizes the log-likelihood after each iteration, which is computed as

$$\log \mathcal{L}(\boldsymbol{\theta} \mid \mathcal{D}) = \sum_{i=1}^{N} \max_{j=1}^{p} \log(p(\tilde{x}_{i,j} \mid \boldsymbol{x}_i)). \tag{5.3}$$

Similar to the EM algorithm, ditARF considers a set of weights $\mathbf{w}$ for each instance. This corresponds to

$$w_i = \prod_j \max p(\tilde{x}_{i,j} \mid \boldsymbol{x}_i).$$

These weights are used when learning a Random Forest classifier as weights for each instance, thus giving higher weights for instances where the model is more confident about the imputation. Following the strategy of the iterative version of ARFs, a Random Forest is iteratively re-trained until it reaches convergence in terms of likelihood. When this occurs, an estimate of

the joint posterior distribution $\tilde{\boldsymbol{x}}^{[t]} \sim P$ is obtained and hence, we provide $p(\tilde{x}_{i,j} \mid \boldsymbol{x}_i)$ as a measure of uncertainty along with the imputed missing value $\dot{x}_{i,j}$.

5.4 Experimental Study

In order to compare the performance of the proposed solution, we perform several experiments on real-world datasets obtained from the UCI repository [Dua and Graff, 2017] as well as on three Single Nucleotide Polymorphism (SNP) datasets which we have truncated to 1000 features for the sake of computation memory. These datasets contain categorical multinomial variables. The datasets used in the experiments are summarized in Table 5.2. So as to properly simulate missing values in real-world situations, we followed the MCAR strategy by corrupting a percentage of the data values. These percentages range from 1% to 30%. We refer to this parameter as the Missing Value Ratio (MVR) throughout the text.

For the purpose of evaluating the proposed solution, we consider marginal accuracy, which is also known as Hamming Score, among the imputed values; and joint accuracy also referred to as Exact Match in the literature. Formally, marginal accuracy can be defined as

$$\frac{1}{N_m}\frac{1}{p_m}\sum_{i=1}^{N_m}\sum_{j=1}^{p_m}\mathbb{1}(\dot{x}_{i,j}, x_{i,j}), \tag{5.4}$$

where N_m and p_m refer to the number of instances and the number of features with missing values, respectively. Similarly, joint accuracy can be defined as

$$\frac{1}{N_m}\sum_{i=1}^{N_m}\mathbb{1}(\dot{\boldsymbol{x}}_i, \boldsymbol{x}_i). \tag{5.5}$$

Finally, since missing value imputation is usually a preprocessing step for further classification tasks, we compare the classification accuracy obtained

TABLE 5.2: Datasets used in experiments, p features, N samples.

Name	p	N	Reference
Mushroom	22	8,124	[Dua and Graff, 2017]
Soybean	35	307	[Dua and Graff, 2017]
Primary Tumor	17	339	[Dua and Graff, 2017]
Lymphography	18	148	[Dua and Graff, 2017]
Congressional Voting Records	16	435	[Dua and Graff, 2017]
Financial Well-Being Survey	212	6,394	[Consumer Financial Protection Bureau, 2017]
SNP Maize	1,000	247	[Negro et al., 2019]
SNP Eucalyptus	1,000	970	[de Lima et al., 2019]
SNP Colorado Beetle	1,000	188	[Crossley et al., 2017]

with a random forest classifier trained on full data, and on imputed data. The experiments have been run 5 times and the average of the scores of each run is used.

We compare our method against a variety of well-known literature approaches. Autoencoder and PCA methods are implemented using the scikit-learn [Pedregosa et al., 2011] package. We tested the performance of both procedural and iterative Autoencoders in three modifications: with one hidden layer of $0.1p$ neurons, one hidden layer of $0.2p$ neurons, or three hidden layers of $0.2p$, $0.1p$, and $0.2p$ neurons respectively, where p is the number of features. The model with one hidden layer of $0.1p$ neurons has shown slightly better performance, although the difference was not significant. The results of this model are further presented. The PCA method was also realized as a neural network with one hidden layer of $0.1p$ neurons but with an identity activation function. The kNN method is presented with the number of neighbors $k = 2$ selected during inner validation.

In order to select the best-performing parameters, we have internally run a grid search over the parameters of Autoreplicative Random Forests. As a result, we opted to use 20 trees (base classifiers) per forest (no significant difference compared to other values), each tree trained on all provided features (better performance than with default parameter), a minimal number of samples per split equal to 5. Criterion (gini/entropy) has not shown an influence on the method's performance.

5.4.1 Results and discussion

Imputation performance

Table 5.3 summarizes the performance of all methods measured by the marginal accuracy, i.e. percentage of correctly imputed values out of the

missing ones. Table 5.4 shows joint accuracy, i.e. percentage of the instances, where *all* values were imputed correctly. MICE results are not shown for the datasets with a large number of features because of excessive computation time.

In low-dimensional datasets, the MICE method remains very competitive. Its time consumption is significantly higher than for all other methods but stays feasible when the number of features is relatively small. The procedural and iterative ARFs show competitive performance. For the Mushroom dataset, pARF shows the best results when the missing value ratio is small but fails when it is big and thus there is not enough data to train a reliable model. In most cases, the itARF method along with its ditARF modification runs second best. In high-dimensional datasets, procedural methods cannot be used when all instances come affected by missing values. The itARF and ditARF methods systematically outperform other methods available. The Friedman-Nemenyi diagrams demonstrate the statistical significance of the methods' performance difference in Fig. 5.3, confirming that three ARF-based methods lie in the high spectrum of methods ranking along with the MICE method.

The ditARF method computes the probabilities of imputed values $p(\tilde{\boldsymbol{x}} \mid \boldsymbol{x})$ on every iteration and uses these to provide a measure of confidence per instance as sample weights of the model on the next iteration. To understand better its behavior, we illustrate probabilities of having a '1' class changing through iterations on Fig. 5.4. We observe that after several iterations each probability 'converges' to a certain level and continues oscillating around it. From this evidence, we conclude that the model is not overfitting (otherwise we would expect converging to 0 or 1) and indeed can provide a distribution for possible values for imputation.

Fig. 5.5 shows the difference in classification accuracy of a Random Forest classifier learned on ground-truth data, without missing values, and a

TABLE 5.3: Marginal accuracy. The best accuracy per column is in **bold**. The second best accuracy is underlined. All results are rounded to 3 dp. For [it]erative (includes MICE) and [p]rocedural versions of methods.

MVR	0.01	0.05	0.1	0.2	0.3	0.01	0.05	0.1	0.2	0.3	0.01	0.05	0.1	0.2	0.3
			Mushroom					Soybean					Tumor		
Complete cases	80.1%	32.3%	10.1%	0.7%	0.04%	69.7%	13.7%	1.0%	0%	0%	83.8%	38.9%	15.0%	1.2%	0%
MICE	0.649	0.698	0.730	**0.753**	**0.767**	**0.867**	**0.873**	**0.875**	**0.838**	**0.823**	**0.775**	**0.749**	**0.778**	**0.749**	0.725
ditARF	0.748	**0.763**	0.747	0.727	0.699	0.830	0.845	0.818	0.768	0.755	0.639	0.660	0.679	0.653	0.658
itARF	0.741	0.742	0.746	0.727	0.684	0.809	0.844	0.829	0.776	0.777	0.604	0.653	0.665	0.643	0.645
pARF	**0.767**	**0.763**	**0.764**	0.684	0.514	0.774	0.780	0.656	–	–	0.572	0.654	0.705	0.687	0.698
itAE	0.605	0.587	0.595	0.563	0.566	0.667	0.718	0.699	0.682	0.673	0.702	0.708	0.742	0.726	**0.727**
pAE	0.574	0.517	0.500	0.530	0.525	0.667	0.725	0.642	–	–	0.702	0.701	0.743	0.683	0.606
itPCA	0.613	0.611	0.612	0.607	0.596	0.710	0.742	0.721	0.688	0.692	0.702	0.712	0.739	0.727	**0.727**
pPCA	0.609	0.585	0.571	0.532	0.490	0.686	0.729	0.662	–	–	0.702	0.610	0.717	0.588	0.522
kNN	0.642	0.659	0.678	0.670	0.569	0.731	0.774	0.768	0.729	0.697	0.526	0.549	0.594	0.559	0.507

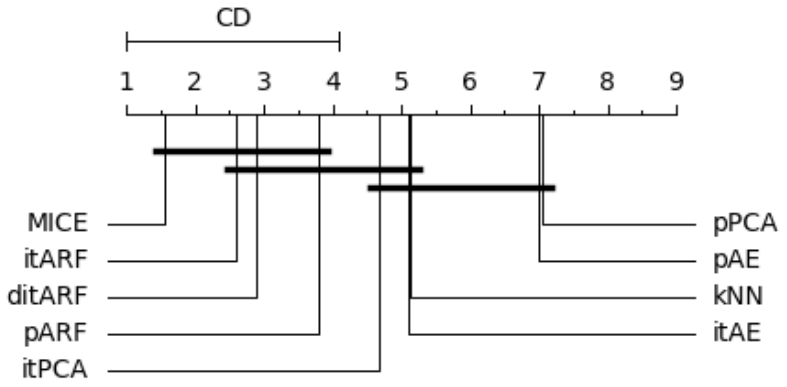
			Votes				Lymphography					Financial Survey			
Complete cases	85.3%	42.2%	18.5%	1.3%	0%	81.8%	40.5%	14.9%	2.7%	0%	11.8%	0%	0%	0%	0%
MICE	**0.888**	**0.774**	**0.772**	**0.758**	**0.768**	0.562	**0.677**	**0.621**	**0.633**	**0.643**	–	–	–	–	–
ditARF	0.712	0.708	0.697	0.684	0.703	0.669	0.556	0.608	0.590	0.610	**0.687**	**0.674**	0.670	0.663	0.655
itARF	0.765	0.703	0.696	0.682	0.689	0.677	0.546	0.609	0.583	0.606	0.676	0.670	**0.673**	**0.664**	**0.656**
pARF	0.653	0.722	0.720	0.712	–	**0.708**	0.586	0.610	0.513	–	0.668	–	–	–	–
itAE	0.612	0.634	0.591	0.531	0.553	0.462	0.480	0.538	0.461	0.465	0.622	0.618	0.616	0.606	0.589
pAE	0.594	0.616	0.537	0.596	–	0.462	0.466	0.553	0.465	–	0.518	–	–	–	–
itPCA	0.676	0.634	0.620	0.603	0.556	0.431	0.517	0.535	0.466	0.462	0.649	0.647	0.645	0.636	0.623
pPCA	0.629	0.528	0.537	0.548	–	0.446	0.493	0.550	0.472	–	0.625	–	–	–	–
kNN	0.824	0.615	0.667	0.625	0.641	0.346	0.406	0.436	0.425	0.447	0.490	0.489	0.491	0.490	0.487

			SNP Maize				SNP Eucalyptus					SNP Wheat				
Complete cases	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
MICE	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
ditARF	0.837	0.819	0.798	0.755	0.694	**0.936**	0.918	0.908	0.848	0.782	0.923	0.931	0.920	0.904	0.920
itARF	**0.857**	**0.846**	**0.835**	**0.825**	**0.817**	0.935	**0.933**	**0.929**	**0.915**	**0.901**	**0.942**	**0.940**	**0.937**	**0.933**	**0.934**
pARF	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
itAE	0.724	0.724	0.717	0.717	0.715	0.715	0.723	0.720	0.715	0.706	0.931	0.929	0.931	0.931	0.931
pAE	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
itPCA	0.725	0.694	0.672	0.645	0.624	0.832	0.850	0.854	0.849	0.831	0.895	0.890	0.883	0.876	0.856
pPCA	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
kNN	0.758	0.753	0.749	0.746	0.736	0.912	0.912	0.909	0.903	0.897	0.914	0.914	0.913	0.912	0.911

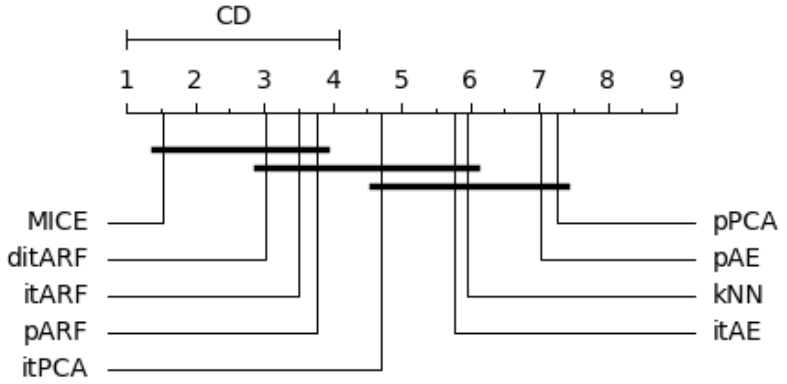
TABLE 5.4: Joint accuracy. The best accuracy per column is in **bold**. The second best accuracy is underlined. All results are rounded to 3 dp. For [it]erative (includes MICE) and [p]rocedural versions of methods.

MVR	0.01	0.05	0.1	0.2	0.3	0.01	0.05	0.1	0.2	0.3	0.01	0.05	0.1	0.2	0.3
			Mushroom					Soybean					Tumor		
Complete cases	80.1%	32.3%	10.1%	0.7%	0.04%	69.7%	13.7%	1.0%	0%	0%	83.8%	38.9%	15.0%	1.2%	0%
MICE	0.622	0.563	0.480	0.307	**0.191**	**0.845**	**0.765**	**0.622**	**0.339**	**0.177**	**0.763**	**0.663**	**0.606**	**0.436**	**0.264**
ditARF	0.731	0.666	0.541	**0.321**	0.154	0.802	0.714	0.545	0.266	0.106	0.619	0.584	0.501	0.289	0.180
itARF	0.725	0.643	0.534	0.306	0.130	0.777	0.719	0.552	0.251	0.113	0.581	0.579	0.488	0.271	0.170
pARF	**0.752**	**0.669**	**0.565**	0.280	0.039	0.742	0.614	0.290	–	–	0.548	0.556	0.531	0.305	0.208
itAE	0.582	0.462	0.349	0.136	0.046	0.613	0.538	0.378	0.141	0.048	0.685	0.622	0.557	0.370	0.261
pAE	0.553	0.390	0.253	0.114	0.038	0.613	0.543	0.264	–	–	0.685	0.612	0.563	0.313	0.111
itPCA	0.591	0.492	0.374	0.179	0.069	0.662	0.559	0.396	0.149	0.084	0.685	0.628	0.546	0.370	0.261
pPCA	0.587	0.463	0.328	0.114	0.031	0.637	0.550	0.289	–	–	0.685	0.508	0.525	0.211	0.087
kNN	0.620	0.527	0.420	0.240	0.100	0.700	0.594	0.427	0.188	0.068	0.500	0.434	0.378	0.205	0.098

			Votes					Lymphography					Financial Survey		
Complete cases	85.3%	42.2%	18.5%	1.3%	0%	81.8%	40.5%	14.9%	2.7%	0%	11.8%	0%	0%	0%	0%
MICE	**0.885**	**0.701**	**0.631**	**0.470**	**0.376**	0.527	**0.560**	**0.424**	**0.262**	**0.105**	–	–	–	–	–
ditARF	0.703	0.636	0.551	0.363	0.272	0.655	0.453	**0.424**	0.201	0.086	**0.454**	**0.038**	0.001	0.000	0.00
itARF	0.770	0.633	0.554	0.343	0.229	0.645	0.437	0.416	0.215	0.088	0.442	0.035	**0.002**	0.000	0.000
pARF	0.642	0.646	0.573	0.400	–	**0.682**	0.473	0.421	0.168	–	0.432	–	–	–	–
itAE	0.600	0.538	0.403	0.230	0.096	0.455	0.389	0.336	0.105	0.019	0.384	0.021	0.001	0.000	0.000
pAE	0.582	0.535	0.388	0.248	–	0.455	0.374	0.336	0.125	–	0.270	–	–	–	–
itPCA	0.667	0.551	0.457	0.287	0.136	0.418	0.431	0.339	0.110	0.022	0.412	0.029	0.001	0.000	0.000
pPCA	0.618	0.428	0.355	0.213	–	0.455	0.409	0.336	0.137	–	0.383	–	–	–	–
kNN	0.818	0.528	0.495	0.269	0.237	0.318	0.297	0.248	0.103	0.034	0.259	0.006	0.000	0.000	0.000

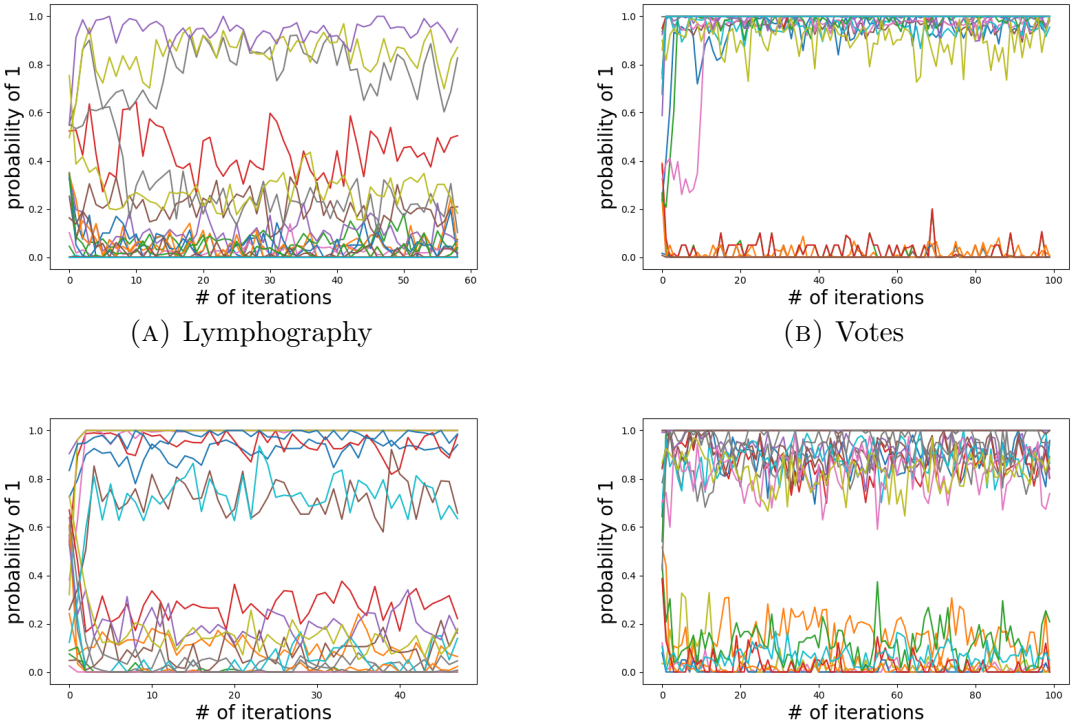


(A) Marginal accuracy



(B) Joint accuracy

FIGURE 5.3: Friedman-Nemenyi diagrams comparing the ranking of the experimentally tested methods. A lower rank is better, statistically undistinguishable methods are connected by a horizontal line.



(A) Lymphography (B) Votes

(C) Tumor (D) Soybean

FIGURE 5.4: Probabilities of predicting a '1' class for the first 20 missing values on 4 different datasets. One line corresponds to one imputed missing value.

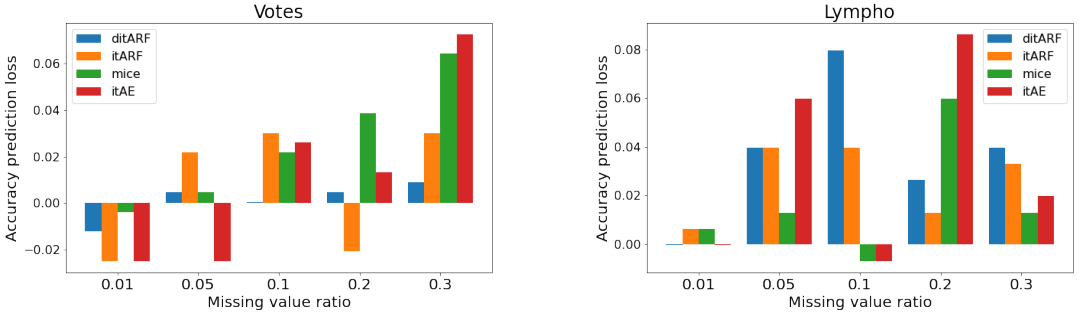


FIGURE 5.5: Classification accuracy gain/loss when compared to a complete dataset (smaller value = better).

Random Forest learned on an imputed dataset. First, we observe that imputation quality and further classification quality do not strictly correlate. This poses the question if the best strategy would be to do imputation and classification simultaneously to optimize the performance of both. Second, in some cases, the proposed ARF method facilitates classification compared to the MICE method even when imputation accuracy is lower. Third, we see in the Votes dataset that the ditARF method in some cases provides significantly better accuracy even when compared to the itARF method, which supports the further need of considering prediction confidence during the imputation step.

Time complexity analysis

The complexity of a Decision Tree is $\mathcal{O}\left(pN \log N\right)$ with regard to the number of features p and the number of instances N. If all the trees in a Random Forest are trained on all features, the total complexity of the forest remains the same. In the MICE method, a separate model is trained per feature, thus for one iteration, the complexity of the MICE method with Random Forest base estimator becomes quadratic $\mathcal{O}\left(p^2 N \log N\right)$. At the same time, with a multi-label Random Forest, the total complexity remains linear. Thus, both the methods itARF and pARF provide linear complexity with regard to the number of features, as the complexity of one forest is only multiplied by the number of iterations which typically is low as convergence is reached soon. These theoretic estimations are well supported in the simulation study, see Fig. 5.6. We empirically compare the time complexity of the imputation methods on subsets of the Eucalyptus dataset under the MCAR scenario with 10% missing values. The subsets are selected as the first p_s features of the original dataset, $10 \leq p_s < 100$.

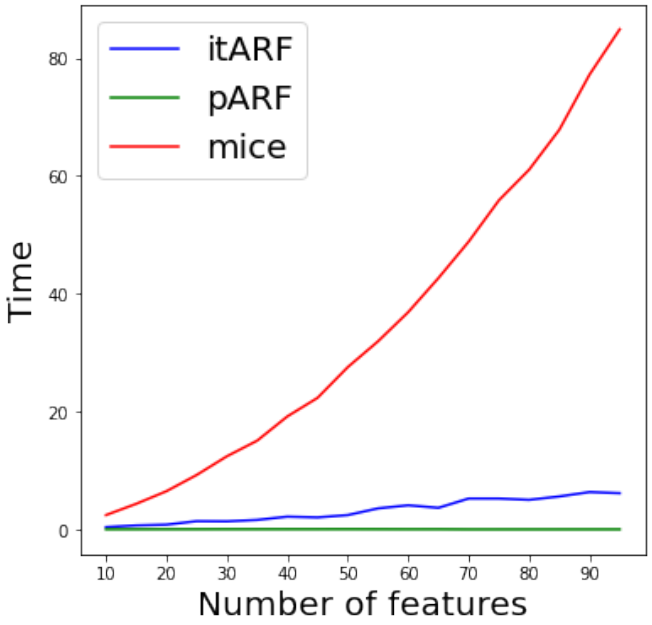


FIGURE 5.6: Empirical results on time complexity (in seconds) for imputation methods. Here, ditARF is not specifically included since it is already covered by iterative ARF (itARF).

5.5 Conclusions and future work

In this work, we describe a general framework for missing value imputation and we deeply analyze the literature on missing value imputation schemes.

In order to accurately impute missing values, we propose multi-output, multi-label, Autoreplicative Random Forests (ARFs) in three different variants. First, we propose procedural ARF (pARF) that leverages the idea of DAEs for missing value imputation that only impute once the missing values. Second, we propose iterative ARF (itARF). The proposed itARF approach works as a deterministic iterative imputation method that not only obtains competitive results to the state-of-the-art methods but also drastically outperforms them in terms of computational time. Moreover, we focused on the necessity of providing a measure of uncertainty with respect to the imputed missing values, and we proposed the distributional itARF (ditARF) which works similarly to the EM algorithm and estimates the posterior distribution.

To evaluate the proposed solution, we have performed an extensive evaluation of the proposed and previously existing methods on low- and high-dimensional datasets in which we included a variety of datasets from the UCI repository plus three SNP datasets. As can be seen, the proposed solutions drastically outperform existing literature approaches when $p \gg N$. Finally, we have also tested the difference between training a Random Forest classifier for an imputed dataset and ground-truth data. The results show that the obtained accuracy with the classifier learned in ARF methods are good estimates since they obtain similar results to the classifier learned with ground-truth data.

Chapter 6

Missing value imputation in genomics data

Single Nucleotide Polymorphisms (SNP) data is essential in genetic studies. Typically, such data is prone to missing values, and removing instances with missing values can adversely affect the quality of further data analysis, thus imputation methods are required. While in human studies a reference genome panel of high quality may be an efficient solution, in non-human settings such panels are often not available. While deep learning is a state-of-the-art approach for imputation in high-dimensional data, existing methods still require enough complete cases to be trained on, which is often unavailable in real-world problems. In this work, we propose Chains of Autoreplicative multi-label Random Forests which impute missing values based only on the information extracted from the presented data, are computationally effective, and work well for high-dimensional and low-sampled data. Experiments on several SNP datasets show that our algorithm effectively imputes missing values and exhibits better performance than standard algorithms that do not require additional reference panels. In this

work, the algorithm is implemented specifically for SNP data. Still, it can easily be adapted for other cases of missing value imputation in biological data, e.g. gene expression arrays.

6.1 Introduction

Genome-wide association studies (GWAS) allow the detection of associations between genetic variants and traits. Typically they study associations between Single-Nucleotide Polymorphisms (SNPs) and complex phenotypes such as traits and/or diseases. GWAS investigate the whole genome, as opposed to methods that study a small number of pre-selected genetic regions. A standard coding for values in SNP datasets is 0, 1, and 2 for variants AA, Aa, and aa respectively, where allele A corresponds to the prevalent variant in the population and allele a to the minor one. Due to linkage disequilibrium [Browning and Browning, 2007; Chen and Shi, 2019], neighboring features can correlate to each other, and taking such dependencies into account is helpful for missing value imputation. At the same time, some long-distance correlations (across the genome) are also possible, though rare.

As discussed earlier in Chapter 5, missing values are a common problem in the domain of data science, and SNP datasets are also prone to a missing value problem due to a variety of reasons, such as deviations from the Hardy-Weinberg equilibrium, an abundance of rare variants, and missing features in combining different datasets in meta-studies [Song et al., 2020]. Within this study, we assume that missing data is Missing Completely At Random as it depends on external factors rather than observed/unobserved values. A typical SNP dataset contains a number of features (positions on the genome) greatly exceeding the number of samples (individuals in the population of the study), and all or most of the samples may be affected by missing values. Most off-the-shelf statistical and machine learning methods

cannot handle missing values, and such values must be imputed, or the whole instance or row removed, before the actual data analysis. When many values are missing, considering only instances with complete information can lead to a loss of necessary information and can yield a very poor or even empty dataset.

For SNP data, imputation methods are traditionally split into reference-based and reference-free methods. Reference-based techniques require a reference panel based on whole-genome sequencing samples and show the advantage of using large datasets with complete data as well as additional information such as linkage patterns, mutations, and recombination hotspots [Das et al., 2018]. While reference-based methods may be considered a state-of-the-art approach in well-studied species, e.g. humans, these reference panels are often not available in many other cases. The differences between the populations within one panel should also always be taken into account. These facts necessitate the search for reference-free methods that incorporate only information from the data itself.

Reference-free methods include common statistical imputation techniques such as replacement with mode statistics [Little and Rubin, 2019], k Nearest Neighbours (kNN) [Schwender, 2012], Singular Value Decomposition (SVD) [Troyanskaya et al., 2001], MissForest [Stekhoven and Bühlmann, 2011], and logistic regression. Recently developed deep learning techniques have also been applied for imputation, e.g. Denoising Autoencoders [Chen and Shi, 2019] method. Below we present the listed methods in more detail.

Mode [Little and Rubin, 2019]. For each feature, a mode of non-missing values, i.e. the most frequent value, is estimated, and the missing values are imputed with this mode.

k-Nearest Neighbors (kNN) [Schwender, 2012]. The imputation procedure is based on the weighted k-Nearest Neighbors algorithm. The algorithm looks for the k samples that are most similar to the one whose missing values need to be replaced and uses these k neighbors to impute

the missing values. For experiments, we used the knncatimpute function implemented in scrime R package.

Singular Value Decomposition (SVD) [Troyanskaya et al., 2001]. This method calculates the k most significant eigenvectors and then imputes the missing values using a low-rank SVD approximation estimated by an Expectation-Maximization algorithm. For experiments, we used IterativeSVD function implemented in fancyimpute [Rubinsteyn and Feldman, 2016] python package.

MissForest [Stekhoven and Bühlmann, 2011]. The MissForest method works in an iterative manner, similar to the MICE method discussed in Chapter 5, by predicting missing values by Random Forests trained on the observed features. The MICE and missForest methods are commonly used for different types of data and, in particular, clinical data, and can be considered state-of-the-art for missing value imputation, but we have not found big evidence of using these methods for high-dimensional datasets, as they become very costly with the rise of the number of features. In our empirical study, we try to adapt the MICE method for SNP data but do not obtain promising results (see Section 6.3).

Denoising Autoencoders (DAE) [Vincent et al., 2008; Chen and Shi, 2019]. Recently developed deep learning techniques have also been applied for genotype imputation, e.g. *Sparse Convolutional Denoising Autoencoder (SCDA)* [Chen and Shi, 2019] method. We remind the reader that the basic idea of both Autoencoders and Denoising Autoencoders is discussed in Chapter 5 and illustrated in Fig. 5.1a and Fig. 5.1b, respectively. Denoising Autoencoders have been successfully applied to address the missing data problems in various fields [Vincent et al., 2008]. In [Chen and Shi, 2019] the authors suggest Sparse Convolutional Denoising Autoencoders (SCDA) to impute missing values in SNP datasets. Sparsity is required due to the high dimensionality of the features space which largely exceeds the number of instances, and convolution layers are used because neighboring

features have a bigger chance to explain each other in SNP data. The main limitation of the SCDA method is that it requires training data of complete cases, which is usually very limited in SNP data. For this reason, we don't include the SCDA method in an experimental setting where all or almost all cases are affected by missingness.

To overcome the lack of complete data in the entire dataset, we propose treating SNP data by splitting it into windows of consequent features and incorporating information from already imputed previous windows. We treat each window as data that can be given to an Autoencoder, however, noticing that we do not explicitly need a hidden-layer representation, we use multi-label Autoreplicative Random Forests instead of neural network architecture, as we propose earlier in Chapter 5.

In this work, we study imputation for SNP data as it exhibits all the aspects we are interested in tackling: high-dimensional data (such that $p \gg N$), the possibility of a significant proportion of missing values, and no reference panel but at least some local correlations in the feature space. However, it is important to note that the same approach can be adapted to any data exhibiting these characteristics. For high-dimensional and low-sampled datasets, the ChARF method was shown to be very competitive in terms of both imputation quality and time complexity. At the same time, even for low-dimensional data, we can adapt this approach with personalized splitting for blocks depending on missing patterns.

The rest of the chapter is organized as follows. We present our method in Section 6.2. The results and their discussion as well as complexity analysis are in Section 6.3. In Section 6.4, we draw conclusions and describe future work.

6.2 Method

Our method consists of two main novelties. First, we use multi-label classifiers (e.g. Autoreplicative Random Forests) for imputation as described in Chapter 5. We use multi-label predictive models that are not based on neural networks, as we want to efficiently process relatively low-sampled (compared to a number of features) datasets, where complex neural networks are prone to overfitting and get stuck in optimizing parameters. Furthermore, as discovering the inner structure of the data itself is out of the scope of this task, we do not explicitly need hidden layers of the neural network. In this chapter, we use procedural Autoreplicative Random Forests (pARF), though the iterative version of the method (itARF) may be also applied and will be implemented in the future.

Second, Chains of subsequent windows of Autoreplicative models allow adapting the idea of Autoencoders to real-world high-dimensional scenarios when there is no complete data available for training, as explained in the following subsection. We call the resulting method Chains of Autoreplicative Random Forests (ChARF).

6.2.1 Ensemble of Chains of Autoencoders

As we use Autoreplicative models where input and output represent the same dataset, both of them are high-dimensional, and thus training process has rather high memory consumption. At the same time, in SNP datasets local neighbors have a much higher chance to be inherited from the same distant ancestor and thus to be informative for imputing missing values within the neighborhood. This effect is called linkage dysequilibrium [Das et al., 2018].

In this work, we use procedural ARFs as they have only one cycle of imputation, and thus modeling takes less time and computational resources.

However, they require complete data for training which may be difficult or impossible to obtain in data with abundant missing values. This is especially the case for high-dimensional data: with a large number of features, it is likely not feasible to select a reasonable number of rows without missing values, even for a small ratio of missingness. This is the second reason to process SNP datasets by small-sized windows where it is possible to select a training subset of reasonable size with full information. We fit the model on the selected subset and then predict values to impute missing ones in the remaining subset.

Note that this window approach may serve for other types of ordered data, such as e.g. gene expression arrays, time series, images, and sound fragments.

Fig. 6.1 shows the average size of available training data in simulation with uniformly distributed missing values. As can be seen, it decreases dramatically with the growth of window size. To increase the method's power to catch and use dependencies between the features, we suggest chains of imputation models, similar to the Classifier Chains methodology [Read et al., 2011], i.e. stacking of already processed features as new features for the consequent estimators (see Fig. 6.2). To keep the complexity of the algorithm feasible and reduce computation time, we do not incorporate all previous windows but select only the ν last ones.

Again, as consequent features have a higher chance to be shared between some individuals, it makes sense to include neighboring windows in chains. To this end, we select one forward and one backward chain, as well as several (up to 3) random chains, to incorporate possible long-term interactions. Selection of previously imputed windows can be generalized as, for example, sampling from a normal distribution with a mean equal to the current window number (Fig. 6.3a) or other kinds of distributions for different kinds of data. In the ensemble of chains, we average predictions (i.e. take a major vote) for each missing value.

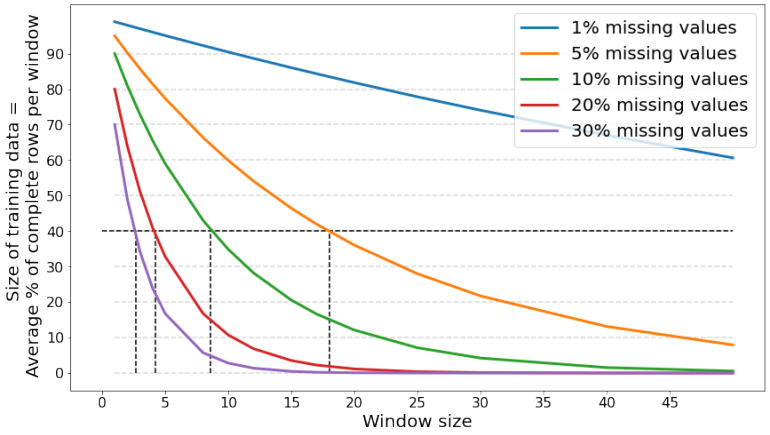


FIGURE 6.1: Average complete training size (i.e. rows without missing values) according to window size Δ. Missing values are simulated for the MCAR scenario with a uniform distribution. Dashed black lines show examples of possible window sizes for $\tau = 0.4$.

TABLE 6.1: Example window sizes Δ according to desired training samples, via Eq. 6.1

	% of missing data				
Size of training data	1%	5%	10%	20%	30%
20% of original data	160	31	15	7	4
30% of original data	120	23	11	5	3
50% of original data	69	14	7	3	2

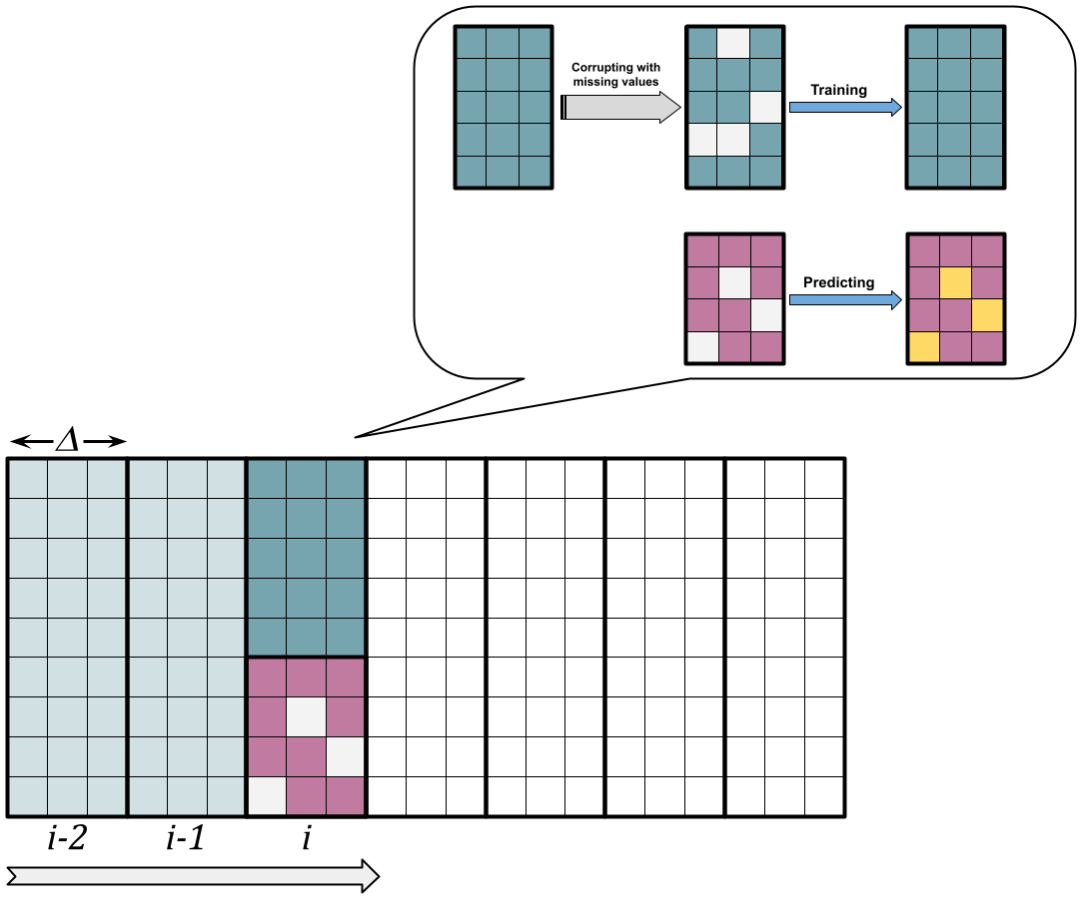
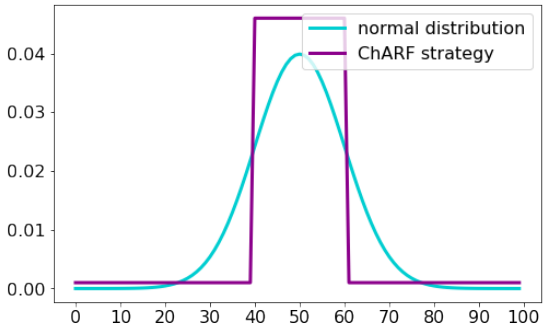
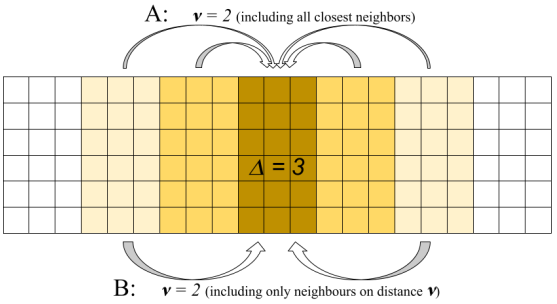


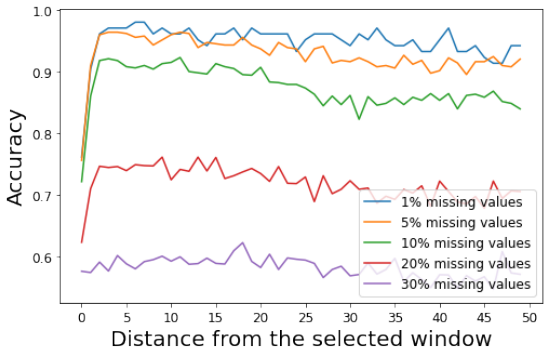
FIGURE 6.2: Model processes windows in a chain, incorporating windows with already imputed values as additional features. At one step, we split the window of size Δ into the training part with complete data and the testing part with missing values. After fitting on training data corrupted with missing values, we impute missing values in the testing part.



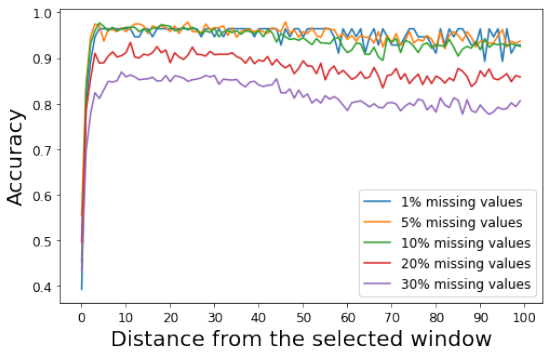
(A) Probabilities to take already predicted windows into chain at position 50; 100 windows; 10 previous windows for each chain; 3 chains: forward, backward, and random.



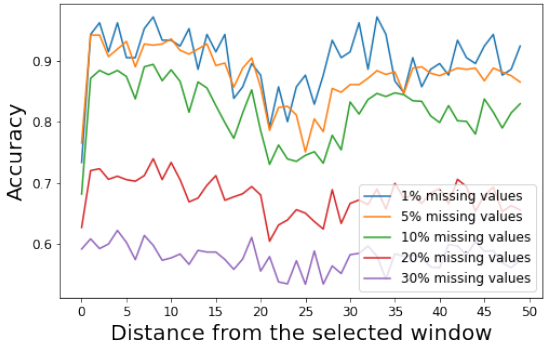
(B) Two strategies to include distant windows into analysis



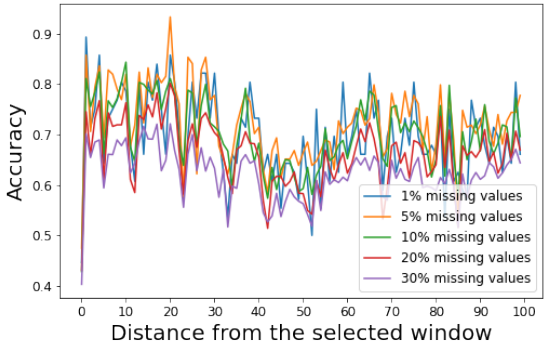
(C) Window size $\Delta = 10$, strategy A



(D) Window size $\Delta = 5$, strategy A



(E) Window size $\Delta = 10$, strategy B



(F) Window size $\Delta = 5$, strategy B

FIGURE 6.3: Including the 2ν closest neighbor windows as additional features (strategy A) significantly increases the accuracy while including only 2 windows on distance ν (strategy B) has occasional and unstable improvement.

Our method is summarised as pseudocode in Algorithm 6. We compare the performance of the models with hyperparameters Δ and ν in Section 6.3. We estimate theoretically the maximum size Δ of one window according to the desired size of training data τ. As $\tau \sim (1-f)^{\Delta}$, then

$$\Delta(\tau) \sim \log_{1-f}\tau = \frac{\ln\tau}{\ln(1-f)}, \quad \tau \in (0,1), \tag{6.1}$$

where f is a fraction of missing values and τ is a desirable threshold for a ratio of complete rows in the training subset. The empirical results of the simulation (Fig. 6.1) correspond to this estimation. We see that with the growth of window size, the size of training data decreases dramatically. As a consequence, the window size should be selected carefully by taking the missing value ratio into account. We suggest possible window sizes according to the desired size of training data in Table 6.1.

Algorithm 6

procedure CHARF($X_{N\times p}$, window size Δ, # of previous steps ν, # of chains K)
  Split features into Δ-wide windows ▷ Last window has size $p \pmod\Delta$
  Generate K permutations of $(1, 2, ..., n = \lceil \frac{p}{\Delta} \rceil)\}$
  for each permutation $\{\sigma(1), ..., \sigma(n)\}$ **do**
    for each window $X_{\sigma(i)}$ **do**
      $X_{ext} \leftarrow X_{\sigma(i)} \oplus X_{\sigma(i-1)} \oplus ... \oplus X_{\sigma(i-\nu)}$ ▷ Stack last ν processed windows as additional features
      $X_{train} \leftarrow X_{ext}^{complete}$ ▷ Select complete cases for training
      $\tilde{X}_{train} \leftarrow X_{train}$ corrupted with m.v. ▷ Uniformly distributed, % of m.v. as in $X_{\sigma(i)}$
      $X_{test} \leftarrow X_{ext}^{missing}$
      Fit model on $(\tilde{X}_{train}, X_{train})$
      $X_{pred} \leftarrow$ predictions of fitted model on X_{test}
      replace m.v. in X_{test} with corresponding values from X_{pred}
  Take major vote for all K predictions per missing value

6.3 Results and discussion

We test Chains of Autoreplicative Random Forests (of 10 trees each) on several high-dimensional SNP datasets ($p \gg N$), briefly summarized in Table 6.2. We simulate the missing values by masking true values in the data under a uniform distribution, with the proportion of missing values 1%, 5%, 10%, 20%, and 30%.

To support the hypothesis that neighboring features have a higher chance of explaining each other, in Fig. 6.3b-6.3f we include experiments for using all neighboring (strategy A) or only two distant (strategy B) windows on distance ν on one of the SNP datasets. We can see that including very close neighbors significantly increases the quality of imputation, while with including distant neighbors the improvement may present (this fact corresponds to possible long-term correlations), but is very unstable and cannot be guaranteed.

We evaluate the performance of our method by imputation marginal accuracy, i.e. the percentage of correctly imputed values out of missing ones. The masking procedure is performed 5 times to produce independent incomplete datasets containing missing values. Values are imputed for each of the datasets, and average accuracy is shown. The empirical study has shown a significant improvement when both the features and the targets are one-hot encoded (paired t-test statistics 3.442, df=29, p-value=0.0018, see Fig. 6.4). While performing one-hot encoding is not common for Random Forests training and typically does not lead to a big improvement, we guess that it can be beneficial when a multi-label multi-class Random Forest is used, i.e. encoding in the target space may be essential.

For ChARF, we first evaluate hyperparameters: window size $\Delta \in [3, 5, 8, 10, 15]$, and number of previous windows in the chain to take as new features $\nu \in [0, 1, 3, 5, 10]$. To reduce the computation time, we first search for the best values of Δ and ν on reduced datasets (first 1000

TABLE 6.2: SNP datasets used in experiments, p features, N samples.

Name	p	N	Reference
Maize	44,729	247	[Negro et al., 2019]
Eucalyptus	33,398	970	[de Lima et al., 2019]
Colorado Beetle	34,186	188	[Crossley et al., 2017]
Arabica Coffee	4,666	596	[Fanlli Carvalho, 2021]
Wheat (Zuchtwert study)	9,763	388	[Reif, 2020]
Coffea Canephora	45,748	119	[Ferrão et al., 2018]

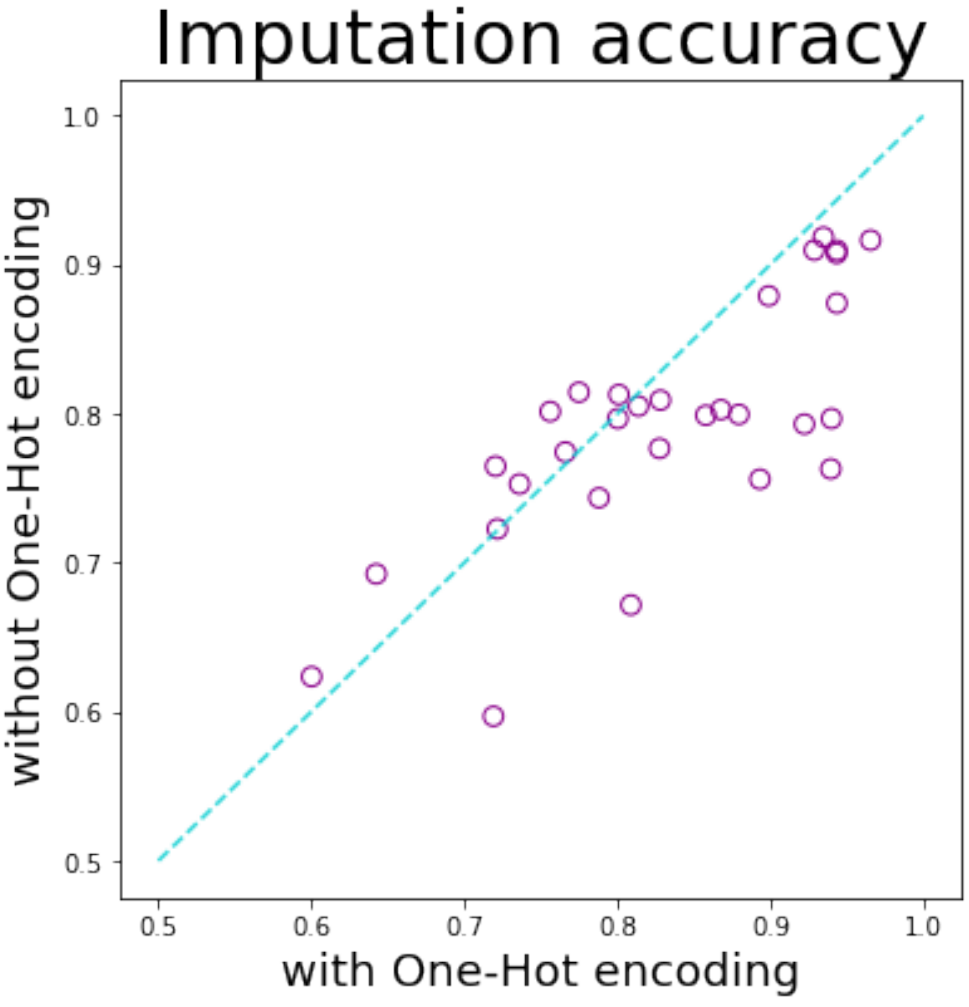


FIGURE 6.4: One-Hot encoding may significantly improve the imputation power of ChARF method in SNP datasets.

features) and then use these values for computation on the entire datasets. The grid-search results are presented in Fig. 6.5. As expected (from estimation in Subsection 6.2.1, Fig. 6.1 and Table 6.1), from Fig. 6.5 we see that the most effective window size decreases with the growth of a number of missing values (since a bigger part of instances gets corrupted).

For the MICE method, with the default settings, each estimator considers all other variables, which makes the total complexity at least quadratic and thus requires huge computational and time resources in the case of high-dimensional data (in our experiments, the machine ran out of memory). The original R package suggests a pre-processing quickpred function, which selects the predictive features based on pairwise correlation, but in this case, quadratic complexity is required in this step. With the intuition that the neighboring features in SNP data have the highest chance to explain each other, for the experiments we select windows of 10 neighbors for each feature. Such an approach is computationally feasible, but the imputation still leaves some missing values in the data (around 10-20%). The possible explanation is the collinearity between features [van Buuren and Groothuis-Oudshoorn, 2011]. This approach worked for smaller SNP datasets (Arabica Coffee and Wheat), but for the other ones, the computations still failed.

In most cases, the experiments show better or competitive performance with regard to benchmark methods (Table 6.3). At the same time, we see that with the rise of the missing value ratio, the accuracy of imputation diminishes. This is explained by the very small size of training data even on small windows for a big number of missing values. However, for a moderate missing value ratio, our method consistently outperforms its alternatives.

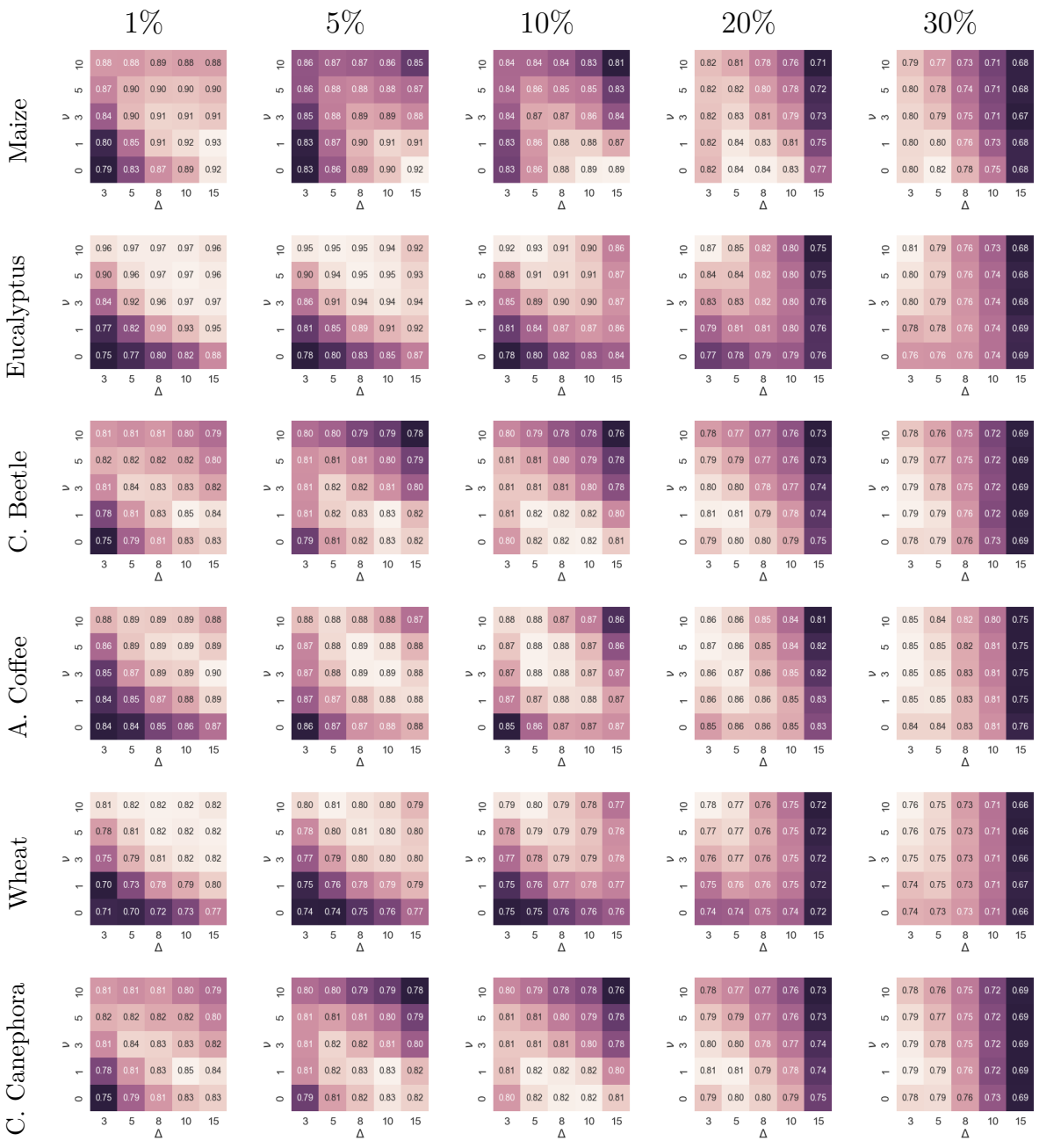


FIGURE 6.5: Accuracy of imputation for SNP datasets for different ratios of missing values (indicated in column headers). Better accuracy lighter/higher value (shown in cells).

TABLE 6.3: Accuracy. An asterisk (*) indicates optimal hyperparameters estimated via internal validation. For kNN we selected the best of $k \in \{3, 5, 10, 20, 50\}$ (shown in brackets) in a similar way. Likewise for rank $\in \{10, 20, 50, 100, 200, 300, 500\}$ for the SVD method. The missForest method was run for the first 100 features only as it is not possible to run it for a whole dataset. The best accuracy per column is in **bold**. The second best accuracy is underlined. All results rounded to 3 dp.

MVR	0.01	0.05	0.1	0.2	0.3	0.01	0.05	0.1	0.2	0.3
	Maize					Eucalyptus				
	$\Delta = 15^*$ $\nu = 1^*$	$\Delta = 15^*$ $\nu = 1^*$	$\Delta = 10^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 10^*$ $\nu = 5^*$	$\Delta = 5^*$ $\nu = 10^*$	$\Delta = 5^*$ $\nu = 10^*$	$\Delta = 3^*$ $\nu = 10^*$	$\Delta = 3^*$ $\nu = 10^*$
ChARF	**0.952**	**0.935**	**0.916**	**0.882**	**0.845**	**0.970**	**0.950**	**0.926**	**0.866**	0.810
kNN (5/10)	0.803	0.802	0.801	0.798	0.794	0.851	0.849	0.847	0.843	**0.839**
mode	0.727	0.727	0.726	0.727	0.726	0.725	0.732	0.731	0.730	0.729
SVD (50/500)	0.647	0.648	0.645	0.643	0.636	0.788	0.788	0.788	0.785	0.780
MICE	–	–	–	–	–	–	–	–	–	–
missForest	0.662	0.650	0.622	0.593	0.580	0.684	0.673	0.626	0.564	0.521
	Colorado Beetle					Arabica Coffee				
	$\Delta = 10^*$ $\nu = 1^*$	$\Delta = 10^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 3^*$ $\nu = 1^*$	$\Delta = 15^*$ $\nu = 3^*$	$\Delta = 10^*$ $\nu = 3^*$	$\Delta = 5^*$ $\nu = 5^*$	$\Delta = 3^*$ $\nu = 10^*$	$\Delta = 3^*$ $\nu = 3^*$
ChARF	**0.835**	**0.824**	**0.818**	**0.805**	**0.792**	**0.897**	**0.886**	**0.878**	**0.866**	0.854
kNN (50/10)	0.765	0.763	0.765	0.765	0.764	0.867	0.866	0.866	0.865	**0.864**
mode	0.761	0.760	0.762	0.761	0.761	0.807	0.804	0.805	0.805	0.804
SVD (50/100)	0.740	0.737	0.737	0.735	0.734	0.693	0.694	0.696	0.692	0.690
MICE	–	–	–	–	–	0.757	0.741	0.724	0.689	0.664
missForest	0.352	0.349	0.361	0.326	0.335	0.497	0.480	0.533	0.541	0.586
	Wheat					Coffea Canephora				
	$\Delta = 8^*$ $\nu = 10^*$	$\Delta = 5^*$ $\nu = 10^*$	$\Delta = 5^*$ $\nu = 10^*$	$\Delta = 3^*$ $\nu = 10^*$	$\Delta = 3^*$ $\nu = 10^*$	$\Delta = 10^*$ $\nu = 1^*$	$\Delta = 10^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 5^*$ $\nu = 1^*$	$\Delta = 3^*$ $\nu = 1^*$
ChARF	0.821	0.808	0.795	0.777	0.762	**0.799**	**0.781**	**0.761**	0.731	0.717
kNN (10/10)	**0.823**	**0.819**	**0.818**	**0.815**	**0.811**	0.737	0.739	0.737	**0.734**	**0.731**
mode	0.729	0.727	0.729	0.729	0.727	0.691	0.693	0.692	0.692	0.691
SVD (200/50)	0.622	0.618	0.609	0.600	0.594	0.456	0.453	0.450	0.449	0.450
MICE	0.641	0.635	0.621	0.585	0.545	–	–	–	–	–
missForest	0.614	0.736	0.746	0.756	0.755	0.377	0.449	0.442	0.395	0.383

6.4 Conclusions and future work

Continuing the idea of Chapter 5, we support the idea of using Autoreplicative Random Forests for missing value imputation and adapt it as Chains of Autoreplicative Random Forests for high-dimensional and low-sampled datasets, e.g. Single Nucleotide Polymorphisms data. This newly proposed method splits data into windows of consequent features and imputes missing values window by window while incorporating information from already processed features.

The empirical study performed on several SNP datasets under different missing value ratios demonstrates a very competitive predictive power. Our method requires neither reference panels nor complete feature-wide data for training and thus can be used in a variety of real-world scenarios when the imputation of missing data is required. Our approach consists of two main novelties: it is model agnostic (we used using Random Forests in experiments) in regards to the Autoreplicator; essentially an Autoencoder with no explicit encoding; and operates on windows of data. Our approach proved competitive and is promising for further investigation.

In future work, we are going to improve algorithm performance on datasets with a large number of missing values and make it more stable with regard to high missing value ratios. As preliminary experiments show that this approach works for the MAR scenario as well, we will further analyze the performance of the ChARF method for other patterns of missingness.

We plan to test the imputation performance with the iterative version of the ARF method applied in the same chaining manner as it potentially may increase effective window size and hence improve the imputation quality. It is also essential to adapt distributional iterative ARFs as the distributional imputation may provide information about the model's confidence of imputation per instance and per feature which may be useful for further analysis of dependencies between genome and phenotype, e.g. GWAS.

Chapter 7

Conclusion

This thesis addresses multi-output modeling, where each instance may be associated with multiple outputs, traditionally called labels in the classification case and targets in the regression case. In recent years, multi-output tasks have arisen more and more frequently and are associated with a wide range of different domains.

In this chapter, we summarize the contributions of the conducted research and discuss future work on the subject.

7.1 Contributions

Research in this thesis considers both classification and regression multi-output settings and, more precisely, is focused on the following:

- We improved the existing solution in the multi-target regression domain, Ensembles of Regressor Chains, particularly for the cases of the

multi-modal target distribution, where previous solutions may produce inadequate results not corresponding to the data distribution, especially when the features are not highly informative (in Chapter 3);

- We considered a non-standard multi-output task where some target values are provided in the inference phase and are to be incorporated into the joint modeling of the other targets without re-training of the model, and we proposed a well-performing probabilistic solution based on Regressor Chains for this setting (in Chapter 4);
- We proposed a novel method for missing value imputation that treats the problem as a multi-output task; using multi-output Random Forests as Autoreplicative imputing models allowed us to reach statistically significant improvement while keeping computational time relatively slow (in Chapter 5);
- We extended the proposed missing value imputation method to Single Nucleotide Polymorphisms datasets which are typically high-dimensional and low-sampled, and obtained the results outperforming the benchmarks in most of the cases (in Chapter 6).

Below we elaborate on the achieved results and discuss them in more detail.

Multi-modal Ensembles of Regressor Chains

In Chapter 3, we discussed the possible reasons for Regressor Chains' underperformance in multi-target regression tasks. As one of the possible reasons for their insufficient power, we suggested an inappropriate choice of the loss function. Oppositely to, e.g., the well-known classification 0/1 Loss function, which estimates the percentage of instances where all labels have been predicted correctly, most regression methods aim at optimizing MSE or MAE loss functions, which are decomposable in multi-output settings and thus practically do not perform joint modeling of the targets.

This may be inadequate, for example, if targets have multi-modal distribution, and MSE-optimizing algorithms model Gaussian distribution, putting the predictions between the clusters, in the place when ground-truth targets are rarely observed. As a solution for this issue, we proposed optimizing UCF (a regression analog of 0/1 Loss), mode-seeking loss function, instead of MSE or MAE, mean-seeking loss function.

To this end, we developed a method based on Ensembles of Regressor Chains. As they are model-agnostic with regard to the base estimator, and we wanted to maintain this flexibility, we forced the ERC model to optimize UCF loss on the upper level, instead of forcing each possible base estimator.

The results showed that the proposed method successfully learns the target distribution and obtains outperforming results when compared to the state-of-the-art methods.

In addition to proposing this new method, we would like to underline that the choice of UCF metric both for optimization and for measuring performance is still underlooked but deserves further attention and investigation.

Metropolis-Hastings sampled Regressor Chains

In Chapter 4, we introduced a non-standard multi-output challenge with additional constraints coming in the inference phase. We were interested in querying the target predictions when some target values are provided before the prediction and have to be incorporated into the joint modeling of the other targets while re-training of the model is not available. This may be the case in, for example, federated learning, transfer learning, or simply when the training data is not available anymore due to ethical or computational reasons.

As Regressor Chains and Ensembles of Regressor Chains are in fact popular choices in multi-target modeling, we proposed Regressor Chain-based methods, Metropolis-Hasting sampled [Ensembles of] Regressor Chains for prediction that include the mentioned constraints in joint modeling of the targets regardless of the order of the chain and in particularly of the positions of the observed values in the chain.

We evaluated the newly proposed methods in several use cases and obtained very promising results. Again, as in Chapter 3, we aimed at maintaining the flexibility of the method with regard to base estimator choice which allows us to apply the method in different domains.

Autoreplicative Random Forests

In Chapter 5, we analyzed the literature on missing value imputation approaches and described a general framework where all methods can be selected via hyperparameter tuning. As an important distinction between methods, we separated procedural and iterative groups.

Procedural methods perform imputation once, training models on all or a subset of observed values. Iterative methods train on all data including pre-imputed missing data and update imputed missing values in multiple cycles until convergence between iterations is reached. We also discussed that while some methods exist in literature only in one of the versions, procedural or iterative, typically they can be easily adapted to both approaches.

As a novel method within the framework, we proposed multi-output multi-label Autoreplicative Random Forests (ARF) for missing value imputation in three different variants. Procedural ARF and iterative ARF follow the same ideas, performing imputation only once or iteratively, respectively. Additionally, we proposed distributional iterative ARF which, first, takes into account the model's confidence of imputation for each instance and,

second, estimates the posterior distribution as a measure of imputation uncertainty.

We showed that Autoreplicative Random Forests in three proposed versions obtain competitive results when compared to state-of-the-art methods while also having significantly lower computation complexity if compared to the main competitor, the MICE method. This makes Autoreplicative Random Forests a very nice imputation alternative with high performance and reasonable computation time.

We also would like to highlight the importance of joint modeling, which is not by default done by multi-label Random Forests as such models maximize probability per-label and as a consequence may provide label combinations that are not typically observed in the data. While making the first step in this direction within the ditARF method, we are interested in further research in the future.

Another important observation was that imputation performance and further classification/regression performance are separate goals and the first does not necessarily imply the second. We concluded that this raises a necessity for designing algorithms that both model the targets and impute the missing values in the features jointly.

Chains of Autoreplicative Random Forests

In Chapter 6, we continued studying previously proposed in Chapter 5 Autoreplicative Random Forests and proposed their adaption for very high-dimensional and low-sampled categorical datasets, in particular Single Nucleotide Polymorphisms (SNP) data. Our approach splits the data into the windows of consequent features and imputes missing values in the windows separately. To increase the method's imputing power, we included already processed windows to the modeling on the non-processed missing values, in a chained fashion, similar to the Classifier and Regressor Chains discussed

above. Performing several chains with different window orders optimizes the result.

The empirical study performed on several SNP datasets demonstrated outstanding results. We concluded that Autoreplicative Random Forests are a promising and flexible novel method that may be adapted and applied to different settings and scenarios.

7.2 Future work

In this thesis, we several times bring up the subject of joint modeling of multiple outputs. While, for example, multi-output Random Forests remain one of the state-of-the-art methods in both classification and regression domains, the loss functions optimize the outputs separately in both settings. As a consequence, the predictions may appear to be impossible or very rare combinations of the outputs and thus not correspond to the actual output distribution.

Intrigued by this problem, we plan to further investigate joint loss function optimization in general, and in particular for multi-output Decision Trees and Random Forests. As we have seen, the importance of such research is observed in both classification and regression as well as both target and feature modeling.

In this thesis, we discuss regression and classification problems mostly separately. First, we identify problems and develop possible solutions in the regression domain, for multi-modal distributions of multiple targets and for joint inference with provided fixed target values. Second, we develop a new multi-label imputation method in the classification domain.

From the first glance, all developed methods may be transferred to the opposite domain, from regression and classification and vice versa. However, initial research uncovers that the straightforward adaptations of our

methods do not perform as well as one may expect. While this is not very surprising and such limitations are often discussed in the literature, we are very interested in further research on the aforementioned limitations, the specific reasons behind them, and possible solutions.

Also, in general, we consider it important to not only provide the mean predicted optimal value of targets, but also provide a measure of uncertainty. In our view, this is particularly important in missing value imputation, where the imputed values typically are not distinguishable from the observed ones in further modeling, and this may bias the analysis outcomes. We plan to further investigate better ways to measure and output models' confidence as well as the ways to take it into account in classification/regression modeling.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U.R., Makarenkov, V., Nahavandi, S., 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. Information Fusion 76, 243–297. doi:10.1016/j.inffus.2021.05.008.

Aitchison, J., 2005. A concise guide to compositional data analysis, in: Compositional Data Analysis Workshop. URL: http://www.leg.ufpr.br/lib/exe/fetch.php/pessoais:abtmartins:a_concise_guide_to_compositional_data_analysis.pdf.

Alvares-Cherman, E., Metz, J., Monard, M.C., 2012. Incorporating label dependency into the binary relevance framework for multi-label classification. Expert Systems with Applications 39, 1647–1655. doi:10.1016/j.eswa.2011.06.056.

Antonenko, E., Read, J., 2022. Multi-modal ensembles of regressor chains for multi-output prediction, in: Advances in Intelligent Data Analysis XX: 20th International Symposium on Intelligent Data Analysis, IDA 2022, Rennes, France, April 20–22, 2022, Proceedings, Springer. pp. 1–13. doi:10.1007/978-3-031-01333-1_1.

Bassett, R., Deride, J., 2018. Maximum a posteriori estimators as a limit of bayes estimators. Mathematical Programming 174, 129–144. doi:10.1007/s10107-018-1241-0.

Beigaitė, R., Read, J., Žliobaitė, I., 2022. Multi-output regression with structurally incomplete target labels: A case study of modelling global vegetation cover. Ecological Informatics 72, 101849. doi:10.1016/j.ecoinf.2022.101849.

Bishop, C.M., 2006. Pattern recognition and machine learning. Springer. URL: https://link.springer.com/book/9780387310732.

Blockeel, H., De Raedt, L., Ramon, J., 2000. Top-down induction of clustering trees doi:10.48550/ARXIV.CS/0011032.

Bogatinovski, J., Todorovski, L., Džeroski, S., Kocev, D., 2022. Comprehensive comparative study of multi-label classification methods. Expert Systems with Applications 203, 117215. doi:10.1016/j.eswa.2022.117215.

Breiman, L., 2001. Random forests. Machine Learning 45, 5–32. doi:10.1023/a:1010933404324.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J., 2017. Classification And Regression Trees. Routledge. doi:10.1201/9781315139470.

Browning, S.R., Browning, B.L., 2007. Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. The American Journal of Human Genetics 81, 1084–1097. doi:10.1086/521987.

Burger, M., Lucka, F., 2014. Maximum a posterior estimates in linear inverse problems with log-concave priors are proper bayes estimators. Inverse Problems 30. doi:10.1088/0266-5611/30/11/114004.

van Buuren, S., Groothuis-Oudshoorn, K., 2011. MICE: Multivariate imputation by chained equations in R. Journal of Statistical Software 45. doi:10.18637/jss.v045.i03.

Chen, J., Shi, X., 2019. Sparse convolutional denoising autoencoders for genotype imputation. Genes 10, 652. doi:10.3390/genes10090652.

Chiarucci, A., Araújo, M.B., Decocq, G., Beierkuhnlein, C., Fernández-Palacios, J.M., 2010. The concept of potential natural vegetation: an epitaph? Journal of Vegetation Science 21, 1172–1178. doi:10.1111/j.1654-1103.2010.01218.x.

Consumer Financial Protection Bureau, 2017. Financial well-being survey data. URL: https://www.consumerfinance.gov/data-research/financial-well-being-survey-data/.

Crossley, M.S., Chen, Y.H., Groves, R.L., Schoville, S.D., 2017. Landscape genomics of colorado potato beetle provides evidence of polygenic adaptation to insecticides. Molecular Ecology 26, 6284–6300. doi:10.1111/mec.14339.

Das, S., Abecasis, G.R., Browning, B.L., 2018. Genotype imputation from large reference panels. Annual Review of Genomics and Human Genetics 19, 73–96. doi:10.1146/annurev-genom-083117-021602.

Davies, R.W., Flint, J., Myers, S., Mott, R., 2016. Rapid genotype imputation from sequence without reference panels. Nature Genetics 48, 965–969. doi:10.1038/ng.3594.

Dembczyński, K., Waegeman, W., Hüllermeier, E., 2012. An analysis of chaining in multi-label classification, in: ECAI: European Conference of Artificial Intelligence, IOS Press. pp. 294–299. URL: http://hdl.handle.net/1854/LU-3132158.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical

Society: Series B (Methodological) 39, 1–22. doi:10.1111/j.2517-6161.1977.tb01600.x.

D'hondt, R., Moylett, S., Goris, A., Vens, C., 2023. A binning approach for predicting long-term prognosis in multiple sclerosis, in: Artificial Intelligence in Medicine. Springer Nature Switzerland, pp. 25–34. doi:10.1007/978-3-031-34344-5_3.

Dougherty, J., Kohavi, R., Sahami, M., 1995. Supervised and unsupervised discretization of continuous features, in: Machine Learning Proceedings 1995. Elsevier, pp. 194–202. doi:10.1016/b978-1-55860-377-6.50032-3.

Dray, S., Josse, J., 2014. Principal component analysis with missing values: a comparative survey of methods. Plant Ecology 216, 657–667. doi:10.1007/s11258-014-0406-z.

Dua, D., Graff, C., 2017. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Elisseeff, A., Weston, J., 2001. A kernel method for multi-labelled classification. Advances in neural information processing systems 14. URL: https://proceedings.neurips.cc/paper_files/paper/2001/hash/39dcaf7a053dc372fbc391d4e6b5d693-Abstract.html.

Fanlli Carvalho, H., 2021. Arabica coffee - IAC/EMBRAPA - BRAZIL.

Feng, Y., Fan, J., Suykens, J.A., 2020. A statistical learning approach to modal regression. Journal of Machine Learning Research 21, 1–35. URL: https://www.jmlr.org/papers/volume21/17-068/17-068.pdf.

Feng, Y., Huang, X., Shi, L., Yang, Y., Suykens, J., 2015. Learning with the maximum correntropy criterion induced losses for regression. Journal of Machine Learning Research 16, 993 – 1034. URL: https://www.jmlr.org/papers/volume16/feng15a/feng15a.pdf.

Ferrão, L.F.V., Ferrão, R.G., Ferrão, M.A.G., Fonseca, A., Carbonetto, P., Stephens, M., Garcia, A.A.F., 2018. Accurate genomic prediction of coffea canephora in multiple environments using whole-genome statistical models. Heredity 122, 261–275. doi:10.1038/s41437-018-0105-y.

Fick, S.E., Hijmans, R.J., 2017. WorldClim 2: new 1-km spatial resolution climate surfaces for global land areas. International Journal of Climatology 37, 4302–4315. doi:10.1002/joc.5086.

Friedl, M., Gray, J., Sulla-Menashe, D., 2019. Mcd12q2 modis/terra+aqua land cover dynamics yearly l3 global 500m sin grid v006. doi:10.5067/MODIS/MCD12Q2.006.

Gerych, W., Hartvigsen, T., Buquicchio, L., Agu, E., Rundensteiner, E.A., 2021. Recurrent bayesian classifier chains for exact multi-label classification, in: Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 15981–15992. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/859bf1416b8b8761c5d588dee78dc65f-Paper.pdf.

Godbole, S., Sarawagi, S., 2004. Discriminative methods for multi-labeled classification, in: Advances in Knowledge Discovery and Data Mining. Springer Berlin Heidelberg, pp. 22–30. doi:10.1007/978-3-540-24775-3_5.

Guo, Y., Gu, S., 2011. Multi-label classification using conditional dependency networks, in: IJCAI Proceedings-international joint conference on artificial intelligence, p. 1300. URL: http://people.scs.carleton.ca/~yuhongguo/research/papers/ijcai11guo.pdf.

Hastings, W.K., 1970. Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57, 97–109. doi:10.1093/biomet/57.1.97.

Hemsing, L.Ø., Bryn, A., 2012. Three methods for modelling potential natural vegetation (PNV) compared: A methodological case study from south-central norway. Norsk Geografisk Tidsskrift - Norwegian Journal of Geography 66, 11–29. doi:10.1080/00291951.2011.644321.

Hendry, A.P., Huber, S.K., León, L.F.D., Herrel, A., Podos, J., 2008. Disruptive selection in a bimodal population of darwin's finches. Proceedings of the Royal Society B: Biological Sciences 276, 753–759. doi:10.1098/rspb.2008.1321.

Hengl, T., Walsh, M.G., Sanderman, J., Wheeler, I., Harrison, S.P., Prentice, I.C., 2018. Global mapping of potential natural vegetation: an assessment of machine learning algorithms for estimating land potential. PeerJ 6, e5457. doi:10.7717/peerj.5457.

Ho, T.K., 1995. Random decision forests, in: Proceedings of 3rd International Conference on Document Analysis and Recognition, pp. 278–282. doi:10.1109/ICDAR.1995.598994.

Jafrasteh, B., Hernández-Lobato, D., Lubián-López, S.P., Benavente-Fernández, I., 2023. Gaussian processes for missing value imputation. Knowledge-Based Systems 273, 110603. doi:10.1016/j.knosys.2023.110603.

Johnstone, I.M., Titterington, D.M., 2009. Statistical challenges of high-dimensional data. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 367, 4237–4253. doi:10.1098/rsta.2009.0159.

Kocev, D., Vens, C., Struyf, J., Džeroski, S., 2013. Tree ensembles for predicting structured outputs. Pattern Recognition 46, 817–833. doi:10.1016/j.patcog.2012.09.023.

Le Morvan, M., Josse, J., Scornet, E., Varoquaux, G., 2021. What's a good imputation to predict with missing values? Advances in Neural Information Processing Systems 34, 11530–11540.

URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/5fe8fdc79ce292c39c5f209d734b7206-Paper.pdf.

de Lima, B.M., Cappa, E.P., Silva-Junior, O.B., Garcia, C., Mansfield, S.D., Grattapaglia, D., 2019. Quantitative genetic parameters for growth and wood properties in eucalyptus "urograndis" hybrid using near-infrared phenotyping and genome-wide SNP-based relationships. PLOS ONE 14, e0218747. doi:10.1371/journal.pone.0218747.

Little, R.J., Rubin, D.B., 2019. Statistical analysis with missing data. volume 793. John Wiley & Sons. URL: https://books.google.fr/books?id=BemMDwAAQBAJ.

Lloyd, S., 1982. Least squares quantization in PCM. IEEE Transactions on Information Theory 28, 129–137. doi:10.1109/TIT.1982.1056489.

Loveland, T., Belward, A., 1997. The international geosphere biosphere programme data and information system global land cover data set (DISCover). Acta Astronautica 41, 681–689. doi:10.1016/s0094-5765(98)00050-2.

Luaces, O., Díez, J., Barranquero, J., del Coz, J.J., Bahamonde, A., 2012. Binary relevance efficacy for multilabel classification. Progress in Artificial Intelligence 1, 303–313. doi:10.1007/s13748-012-0030-x.

MacKay, D.J., 2003. Information theory, inference and learning algorithms. Cambridge university press. URL: https://books.google.fr/books?id=AKuMj4PN_EMC.

Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S., 2012. An extensive experimental comparison of methods for multi-label learning. Pattern Recognition 45, 3084–3104. doi:10.1016/j.patcog.2012.03.004.

Manolio, T.A., 2010. Genomewide association studies and assessment of the risk of disease. New England Journal of Medicine 363, 166–176. doi:10.1056/nejmra0905980.

McCoy, J.T., Kroon, S., Auret, L., 2018. Variational autoencoders for missing data imputation with application to a simulated milling circuit. IFAC-PapersOnLine 51, 141–146. doi:10.1016/j.ifacol.2018.09.406.

Mechenich, M.F., Žliobaitė, I., 2023. Eco-ISEA3h, a machine learning ready spatial database for ecometric and species distribution modeling. Scientific Data 10. doi:10.1038/s41597-023-01966-x.

Melki, G., Cano, A., Kecman, V., Ventura, S., 2017. Multi-target support vector regression via correlation regressor chains. Information Sciences 415-416, 53 – 69. doi:10.1016/j.ins.2017.06.017.

de Mendiburu, F., de Mendiburu, M., 2019. Package 'agricolae'. URL: https://cran.r-project.org/package=agricolae.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E., 1953. Equation of state calculations by fast computing machines. The Journal of Chemical Physics 21, 1087–1092. doi:10.1063/1.1699114.

Montiel, J., Read, J., Bifet, A., Abdessalem, T., 2018. Scalable model-based cascaded imputation of missing data, in: Advances in Knowledge Discovery and Data Mining. Springer International Publishing, pp. 64–76. doi:10.1007/978-3-319-93040-4_6.

Moyano, J.M., Gibaja, E.L., Ventura, S., 2017. An evolutionary algorithm for optimizing the target ordering in ensemble of regressor chains, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE. doi:10.1109/cec.2017.7969548.

Muthukumar, V., Narang, A., Subramanian, V., Belkin, M., Hsu, D., Sahai, A., 2021. Classification vs regression in overparameterized regimes: Does the loss function matter? The Journal of Machine Learning Research 22, 10104–10172. URL: https://dl.acm.org/doi/abs/10.5555/3546258.3546480.

Mylonas, N., Mollas, I., Liu, B., Manolopoulos, Y., Tsoumakas, G., 2023. On the persistence of multilabel learning, its recent trends, and its open issues. IEEE Intelligent Systems 38, 28–31. doi:10.1109/mis.2023.3255591.

Negro, S.S., Millet, E.J., Madur, D., Bauland, C., Combes, V., Welcker, C., Tardieu, F., Charcosset, A., Nicolas, S.D., 2019. Genotyping-by-sequencing and SNP-arrays are complementary for detecting quantitative trait loci by tagging different haplotypes in association studies. BMC Plant Biology 19. doi:10.1186/s12870-019-1926-4.

Paliwal, S., Iglesias, P.A., Campbell, K., Hilioti, Z., Groisman, A., Levchenko, A., 2007. MAPK-mediated bimodal gene expression and adaptive gradient sensing in yeast. Nature 446, 46–51. doi:10.1038/nature05561.

Park, S.H., Fürnkranz, J., 2008. Multi-label classification with label constraints, in: ECML PKDD 2008 Workshop on Preference Learning, pp. 157–171. URL: http://ecmlpkdd2008.org/files/pdf/workshops/pl/11.pdf.

Pearson, K., 1894. Contributions to the mathematical theory of evolution. Philosophical Transactions of the Royal Society of London. A 185, 71–110. URL: https://www.jstor.org/stable/90667.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830. URL: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf.

Perez-Lebel, A., Varoquaux, G., Morvan, M.L., Josse, J., Poline, J.B., 2022. Benchmarking missing-values approaches for predictive models on health databases. GigaScience 11. doi:10.1093/gigascience/giac013.

Phan-Minh, T., Grigore, E.C., Boulton, F.A., Beijbom, O., Wolff, E.M., 2020. Covernet: Multimodal behavior prediction using trajectory sets, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 14074–14083. URL: https://openaccess.thecvf.com/content_CVPR_2020/html/Phan-Minh_CoverNet_Multimodal_Behavior_Prediction_Using_Trajectory_Sets_CVPR_2020_paper.html.

Poonawala-Lohani, N., Riddle, P., Adnan, M., Wicker, J., 2021. A novel approach for time series forecasting of influenza-like illness using a regression chain method, in: Biocomputing 2022, WORLD SCIENTIFIC. doi:10.1142/9789811250477_0028.

Qi, J., Du, J., Siniscalchi, S.M., Ma, X., Lee, C.H., 2020. On mean absolute error for deep neural network based vector-to-vector regression. IEEE Signal Processing Letters 27, 1485–1489. doi:10.1109/lsp.2020.3016837.

Raja, N.B., Aydin, O., Çiçek, İ., Türkoğlu, N., 2018. A reconstruction of Turkey's potential natural vegetation using climate indicators. Journal of Forestry Research 30, 2199–2211. doi:10.1007/s11676-018-0855-7.

Ramírez-Corona, M., Sucar, L.E., Morales, E.F., 2014. Chained path evaluation for hierarchical multi-label classification, in: Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference, p. 502–507. URL: https://d1wqtxts1xzle7.cloudfront.net/76385587/7896-libre.pdf?1639662221=&response-content-disposition=inline%3B+filename%3DChained_Path_Evaluation_for_Hierarchical.pdf&Expires=1692806096&Signature=OzHdz4ZbydU-gXyuJl-54PE2~hodpVUfNJJhU2~Xvy8aqKxRwbcf46zY1dY2DADKdl-EDmTfXkp_&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA.

Read, J., Martino, L., 2020. Probabilistic regressor chains with Monte Carlo methods. Neurocomputing 413, 471–486. doi:10.1016/j.neucom.2020.05.024.

Read, J., Pfahringer, B., Holmes, G., Frank, E., 2011. Classifier chains for multi-label classification. Machine Learning 85, 333–359. doi:10.1007/s10994-011-5256-5.

Read, J., Pfahringer, B., Holmes, G., Frank, E., 2021. Classifier chains: A review and perspectives. Journal of Artificial Intelligence Research 70, 683–718. doi:10.1613/jair.1.12376.

Reif, J., 2020. Genotyping information for diverse european bread wheat genotypes based on the zuchtwert project. URL: 10.5447/ipk/2020/12.

Rubinsteyn, A., Feldman, S., 2016. fancyimpute: An imputation library for python. URL: https://github.com/iskandr/fancyimpute.

Rupprecht, C., Laina, I., DiPietro, R., Baust, M., Tombari, F., Navab, N., Hager, G.D., 2017. Learning in an uncertain world: Representing ambiguity through multiple hypotheses, in: Proceedings of the IEEE international conference on computer vision, pp. 3591–3600. URL: https://openaccess.thecvf.com/content_iccv_2017/html/Rupprecht_Learning_in_an_ICCV_2017_paper.html.

Sahr, K., White, D., Kimerling, A.J., 2003. Geodesic discrete global grid systems. Cartography and Geographic Information Science 30, 121–134. doi:10.1559/152304003100011090.

Salmerón, A., Rumí, R., Langseth, H., Nielsen, T.D., Madsen, A.L., 2018. A review of inference algorithms for hybrid bayesian networks. Journal of Artificial Intelligence Research 62, 799–828. doi:10.1613/jair.1.11228.

Santana, E., Mastelini, S., Jr., S., 2017. Deep regressor stacking for air ticket prices prediction, in: Anais do XIII Simpósio Brasileiro de Sistemas

de Informação, SBC, Porto Alegre, RS, Brasil. pp. 25–31. doi:10.5753/sbsi.2017.6022.

Santos, M.S., Pereira, R.C., Costa, A.F., Soares, J.P., Santos, J., Abreu, P.H., 2019. Generating synthetic missing data: A review by missing mechanism. IEEE Access 7, 11651–11667. doi:10.1109/access.2019.2891360.

Schwender, H., 2012. Imputing missing genotypes with weighted k nearest neighbors. Journal of Toxicology and Environmental Health, Part A 75, 438–446. doi:10.1080/15287394.2012.674910.

Seo, B., Shin, J., Kim, T., Youn, B.D., 2022. Missing data imputation using an iterative denoising autoencoder (IDAE) for dissolved gas analysis. Electric Power Systems Research 212, 108642. doi:10.1016/j.epsr.2022.108642.

Sokolovska, N., Chevaleyre, Y., Zucker, J.D., 2018. A provable algorithm for learning interpretable scoring systems, in: International Conference on Artificial Intelligence and Statistics, PMLR. pp. 566–574. URL: https://proceedings.mlr.press/v84/sokolovska18a.html.

Song, M., Greenbaum, J., Luttrell, J., Zhou, W., Wu, C., Shen, H., Gong, P., Zhang, C., Deng, H.W., 2020. A review of integrative imputation for multi-omics datasets. Frontiers in Genetics 11. doi:10.3389/fgene.2020.570255.

Spyromitros-Xioufis, E., Sechidis, K., Vlahavas, I., 2020. Multi-target regression via output space quantization, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE. doi:10.1109/ijcnn48605.2020.9206984.

Spyromitros-Xioufis, E., Tsoumakas, G., Groves, W., Vlahavas, I., 2016. Multi-target regression via input space expansion: treating targets as inputs. Machine Learning 104, 55 – 98. doi:10.1007/s10994-016-5546-z.

Stekhoven, D.J., Bühlmann, P., 2011. MissForest — non-parametric missing value imputation for mixed-type data. Bioinformatics 28, 112–118. doi:10.1093/bioinformatics/btr597.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B., 2001. Missing value estimation methods for DNA microarrays. Bioinformatics 17, 520–525. doi:10.1093/bioinformatics/17.6.520.

Tsanas, A., Xifara, A., 2012. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. Energy and Buildings 49, 560–567. doi:10.1016/j.enbuild.2012.03.003.

Tsoumakas, G., Katakis, I., 2007. Multi-label classification. International Journal of Data Warehousing and Mining 3, 1–13. doi:10.4018/jdwm.2007070101.

Tsoumakas, G., Vlahavas, I., 2007. Random k-labelsets: An ensemble method for multilabel classification, in: Machine Learning: ECML 2007. Springer Berlin Heidelberg, pp. 406–417. doi:10.1007/978-3-540-74958-5_38.

Uffelmann, E., Huang, Q.Q., Munung, N.S., de Vries, J., Okada, Y., Martin, A.R., Martin, H.C., Lappalainen, T., Posthuma, D., 2021. Genome-wide association studies. Nature Reviews Methods Primers 1. doi:10.1038/s43586-021-00056-9.

Vasconcelos, J.C.S., Cordeiro, G.M., Ortega, E.M.M., Édila Maria de Rezende, 2021. A new regression model for bimodal data and applications in agriculture. Journal of Applied Statistics 48, 349–372. doi:10.1080/02664763.2020.1723503.

Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A., 2008. Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning - ICML '08, ACM Press. doi:10.1145/1390156.1390294.

Waegeman, W., Dembczyński, K., Hüllermeier, E., 2019. Multi-target prediction: a unifying view on problems and methods. Data Mining and Knowledge Discovery 33, 293–324. doi:10.1007/s10618-018-0595-5.

Wolputte, E.V., Blockeel, H., 2020. Missing value imputation with MERCS: A faster alternative to MissForest, in: Discovery Science. Springer International Publishing, pp. 502–516. doi:10.1007/978-3-030-61527-7_33.

Wood, D., Mu, T., Webb, A., Reeve, H., Lujan, M., Brown, G., 2023. A unified theory of diversity in ensemble learning. arXiv:2301.03962.

Wright, S.J., 2015. Coordinate descent algorithms. Mathematical Programming 151, 3–34. doi:10.1007/s10107-015-0892-3.

Wu, Z., Lian, G., 2020. A novel dynamically adjusted regressor chain for taxi demand prediction, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE. doi:10.1109/ijcnn48605.2020.9207160.

Xu, D., Shi, Y., Tsang, I.W., Ong, Y.S., Gong, C., Shen, X., 2019. Survey on multi-output learning. IEEE transactions on neural networks and learning systems 31, 2409–2429. doi:10.1109/tnnls.2019.2945133.

Yao, W., Li, L., 2014. A new regression model: Modal linear regression. Scandinavian Journal of Statistics 41, 656–671. doi:10.1111/sjos.12054.

Yeh, I.C., 2007. Modeling slump flow of concrete using second-order regressions and artificial neural networks. Cement and Concrete Composites 29, 474–480. doi:10.1016/j.cemconcomp.2007.02.001.

Zhang, M.L., Zhang, K., 2010. Multi-label learning by exploiting label dependency, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, p. 999–1008. doi:10.1145/1835804.1835930.

Zhang, M.L., Zhou, Z.H., 2007. ML-KNN: A lazy learning approach to multi-label learning. Pattern Recognition 40, 2038–2048. doi:10.1016/j.patcog.2006.12.019.

Zhang, M.L., Zhou, Z.H., 2014. A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering 26, 1819–1837. doi:10.1109/tkde.2013.39.

İrsoy, O., Alpaydin, E., 2016. Autoencoder trees, in: Holmes, G., Liu, T.Y. (Eds.), Asian Conference on Machine Learning, PMLR, Hong Kong. pp. 378–390. URL: http://proceedings.mlr.press/v45/Irsoy15.