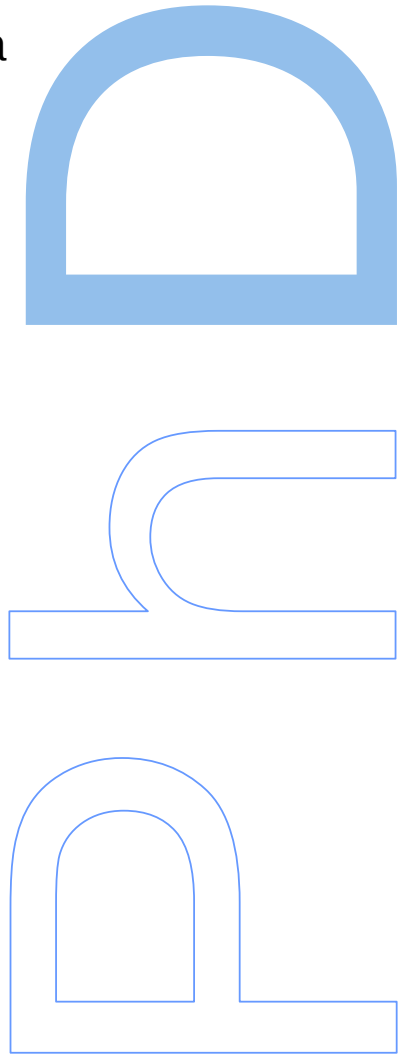
Predictive Analytics for Spatio-Temporal Data

Mariana Rafaela Figueiredo Ferreira de Oliveira
Programa Doutoral em Informática das Universidades do Minho, Aveiro e Porto
Departamento de Ciência de Computadores
2021

Orientador
Luís Fernando Rainho Alves Torgo, Professor Associado
Faculdade de Ciências da Universidade do Porto

Coorientador
Vítor Manuel de Morais Santos Costa, Professor Associado
Faculdade de Ciências da Universidade do Porto



To my family and partner.

Acknowledgements

I would like to express my deepest gratitude to my advisors, Prof. Luís Torgo and Prof. Vítor Santos Costa. I feel very fortunate to have had them as mentors, and cannot thank them enough for all their support and guidance throughout the years. Prof. Luís Torgo was a source of priceless feedback and always helped me stay on track, even from a considerable distance. Prof. Vítor Santos Costa was invariably patient and had consistently insightful suggestions to share. Both were kind and generous with their time, and I am honoured to have worked with and learned from them.

I am also grateful to all my colleagues in Office 1.70. A special word of appreciation should go to Nuno Moniz for the many helpful discussions and for having challenged me to work on the problem that would become the topic of the fifth chapter of this thesis. Many thanks to Paula Branco and Vítor Cerqueira, who shared the office with me for the longest time. It was great to see these three colleagues and friends complete their PhDs.

I would also like to thank all my professors and colleagues at the Department of Computer Science at FCUP, at LIAAD - INESC TEC, and at the MAP-i Doctoral Program. Special thanks to Prof. Rita Ribeiro and Prof. Alípio Jorge, and my colleague Luís Fernandes. Thanks should also go to the administrative staff at DCC, FCUP, and INESC TEC. Special thanks to Ms. Alexandra Ferreira and Ms. Joana Dumas.

Finally, I am extremely grateful to my family, friends, and partner. Without their support, I would not be where I am today.

This work was financially supported by the Portuguese funding agency, FCT – Fundação para a Ciência e Tecnologia, through a MAP-i Ph.D. research grant (PD/BD/128166/2016).

I also acknowledge funding by National Funds through the Portuguese funding agency, FCT – Fundação para a Ciência e a Tecnologia within projects: UID/EEA/50014/2013, UID/EEA/50014/2019, and UIDB/50014/2020. I also acknowledge funding by the ERDF through the COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961.

Abstract

Increasingly widespread sensor networks collect sequences of numerical observations at fixed locations over time, and generate vast amounts of geo-referenced time series. Decision-makers need to make well-informed, data-driven choices, but the volume of data makes it difficult for human experts to glean actionable information. By forecasting the future behaviour of spatio-temporal data, Predictive Analytics (PA) can guide their course of action in a wide range of domains, from transportation to environmental monitoring.

Standard PA faces issues when applied to spatio-temporal data. Most methods assume observations to be independent and identically distributed (i.i.d.), but the autocorrelation in spatio-temporal data breaks this assumption. The implicit spatial and temporal dependencies between observations also cause problems for standard performance estimation used to evaluate these solutions.

Our empirical study of over 15 performance estimation methods showed that standard cross-validation (CV) led to over-optimistic estimates in spatio-temporal settings. We recommend that practitioners evaluate their models using out-of-sample (OOS) methods that respect the temporal order of the data, or CV variants that block observations along time.

Predictive methods explicitly designed for spatio-temporal data can leverage data dependencies to their advantage and improve predictions. We proposed a pre-processing strategy based on a previous proposal to extract features incorporating information from past and neighbouring observations. We tested our proposed method on 17 real-world variables and found that it improved prediction for about half of the tested data and learning model combinations.

In some applications, the target variable follows an imbalanced distribution where the extreme and rare values represent cases of heightened importance (e.g., a spike in air pollution). These values can be particularly difficult to predict, as most standard PA methods optimize for the average case. Once again, the observations' spatio-temporal context can work to our advantage. We proposed new strategies that improved extreme value prediction by introducing a bias into typically random resampling approaches.

Keywords: machine learning; spatio-temporal data; geo-referenced time series; predictive analytics; performance evaluation

Resumo

Redes de sensores que coletam sequências de observações ao longo do tempo em localizações fixas são cada vez mais comuns, gerando grandes quantidades de séries temporais geo-referenciadas. Decisores em muitas áreas têm interesse em tomar decisões bem informadas e baseadas na análise de dados, mas o volume de dados torna impossível para um perito humano obter informação que possa guiar os seus planos de ação. A Análise Preditiva (AP) pode ajudar, prevendo o comportamento futuro de dados espácio-temporais num grande número de domínios, desde os transportes à monitorização ambiental.

Problemas surgem quando os métodos de previsão mais comuns são aplicados a dados espácio-temporais. A maioria destes métodos assume que os dados são independentes e identicamente distribuídos, mas isso não se aplica a dados espácio-temporais. O facto de as observações terem dependências temporais e espaciais implícitas entre si também causa problemas aos métodos habitualmente usados para avaliar os métodos de previsão.

O nosso estudo empírico de mais de 15 métodos de estimação de erro mostram que a validação cruzada (VC) padrão é demasiado otimista, subestimando erros no contexto espácio-temporal. Recomendamos que os analistas avaliem os seus modelos usando métodos "fora da amostra" que respeitam a ordem temporal dos dados, ou então métodos de VC que juntam blocos de observações consecutivas.

Métodos de previsão explicitamente pensados para dados espácio-temporais podem aproveitar as dependências dos dados para melhorar as suas previsões. Propomos um método de pré-processamento, baseado num trabalho anterior, que extrai preditores que incorporam informação de dados passados e de localizações vizinhas. Testamos o nosso método em 17 conjuntos de dados reais, e melhoramos a qualidade de previsões em cerca de metade das combinações de dados e modelos testadas.

Nalgumas aplicações, as variáveis alvo têm uma distribuição desequilibrada onde os valores raros e extremos representam casos de elevada importância (por exemplo, um pico na poluição do ar). Estes valores podem ser particularmente difíceis de prever, tendo em conta que a maior parte dos métodos padrão de AP tentam otimizar o caso médio. Mais uma vez, podemos utilizar as dependências espácio-temporais das nossas observações para nosso

proveito. Propusemos estratégias novas que melhoraram a previsão de valores extremos ao introduzir um termo de enviesamento em abordagens de reamostragem usualmente aleatórias.

Palavras-chave: machine learning; dados espácio-temporais; séries temporais geo-referenciadas; análise preditiva; avaliação de modelos

Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

ANN Artificial Neural Network. 17, 19

AOC Area Over the Curve. 33

AR-HMM Auto-Regressive Hidden Markov Model. 21

ARIMA Auto-Regressive Integrated Moving Average. 18, 19, 78

AUC Area Under the Curve. 95

BN Bayesian Network. 20, 21

CIF Conditional Inference Forest. 94, 95, 102, 104

CIT Conditional Inference Tree. 73, 76, 94, 95

CV cross-validation. vii, 4, 25–30, 37–44, 49, 50, 52, 54, 55, 57–63, 71, 113, 114, 116, 117, 126

FN False Negative. 31

FP False Positive. 31

GWR Geographically Weighted Regression. 19

HMM Hidden Markov Model. 20

i.i.d. independent and identically distributed. vii, 2, 4, 14, 15, 24, 26, 34

ILP Inductive Logic Programming. 16, 20, 21, 24, 35

LLO leave-location-out. 29

LLTO leave-location-and-time-out. 30

LOOCV leave-one-out cross-validation. 26, 27, 29

LTO leave-time-out. 30

MAD Mean Absolute Deviation. 32

MAE Mean Absolute Error. 32, 51

MAPE Mean Absolute Percentage Error. 32

MRF Markov Random Field. 20, 21

MSE Mean Squared Error. 32, 92

NMAE Normalized Mean Absolute Error. 32, 51, 72, 73, 77

OOS out-of-sample. vii, 25, 27, 28, 52–56, 59, 63, 114, 126

PA Predictive Analytics. vii, 1, 9, 14, 15, 25, 34, 113, 118

PCA Principal Component Analysis. 18, 78

PCNM Principal Coordinates of Neighbor Matrices. 17, 78

RAE Relative Absolute Error. 32

REC Regression Error Characteristic. 33

RECS Regression Error Characteristic Surfaces. 33

RROC Receiver Operating Characteristic for Regression. 32

SERA Squared Error-Relevance Area. 34

SMOTE Synthetic Minority Oversampling Technique. 110

STRF Spatio-Temporal Random Field. 21

TN True Negative. 31

TP True Positive. 31

VAR vector autoregression. 19, 78

Chapter 1

Introduction

The last few years have seen the development of widespread sensor networks, that collect extensive data over phenomena such as pollution levels, water quality, or electricity demand. The large amounts of data captured by these networks cannot be processed by humans. Instead, they require pairing with Predictive Analytics (PA) in order to obtain actionable information. To do so, the nature of these data needs to be considered. One first challenge is that sensors are not independent entities as there are dependency structures present in both time and space. That is, a value measured at a given time and location largely depends on values previously measured at that location and its neighbours. The incorporation of spatio-temporal contextual information can be crucial to achieving reliable predictions.

This thesis addresses several key aspects of the problem of spatio-temporal Predictive Analytics. The research aims to identify guidelines for performance estimation that account for spatio-temporal dependency; to design novel methods that improve predictive ability by incorporating spatio-temporal contextual information, while being able to cope with scenarios where it is important to accurately capture rare or extreme events.

In the following sections, we elaborate on the background and motivation, as well as the context for the problems addressed in our research aims and contributions.

1.1 Context and Motivation

Predictive Analytics is one of the main research areas of Data Mining. PA involves obtaining a model that approximates an unknown function which maps a set of predictor variable values into a target variable value. Several important real-world applications require decisions to be taken based on these predictions.

Spatio-temporal data can be categorised into different data types: spatio-temporal event data (e.g., occurrence of a crime), trajectories (e.g., taxi movements in a city), point reference

data (e.g., a weather balloon measuring air temperature along its path), geo-referenced time series (e.g., weather stations measuring wind speed at different fixed locations, or time-varying land value maps), and network data (e.g., traffic in a road network). As shown by these examples, the data in many applications are inherently spatio-temporal. We focus on geo-referenced time series in this thesis.

Geo-referenced time series consist of sequences of measurements generated over time at fixed locations. Locations can be represented by vector data; more often by points (e.g., weather stations represented by points), but they can also be represented by lines, or polygons (e.g., districts represented by polygons). They can alternatively be represented by raster data, which stores data in cells across a regular grid (e.g., a sequence of remote sensing images). Solutions designed for irregularly distributed point locations can often be adapted to accommodate regular raster data, since they can use the centroids of each cell in the raster as the point locations; the reverse may not apply. We focused on data that represent locations as points in space, whether distributed in a grid or not.

1.1.1 Predictive Analytics for Spatio-Temporal Data

Most predictive modelling algorithms make the assumption that data points are independent and identically distributed. However, in many applications, this assumption does not hold, since there is dependency between observations. These dependency structures can be explicit, as is the case of graph or network data where edges represent explicit relationships between recorded data items (e.g., in social networks, chemical compound databases, or a road network); or they can be implicit. Implicit dependencies are not explicitly specified but are known to typically exist in the domain. For example, in a time series there is usually temporal auto-correlation between consecutive observations (or other seasonal dependencies); in geo-referenced data, values measured at a certain location are usually correlated with the observations of its neighbours. Spatio-temporal data typically shows dependencies both in time and space whether data records consist of objects' trajectories or, as in this thesis, measures of a numerical variable over time at several static locations.

When the available data has an associated spatio-temporal context such as a temporal tag and/or geo-referenced location, this contextual information can be used to leverage the implicit dependency between observed values in order to improve predictive performance. The focus of this thesis is the problem of spatio-temporal forecasting, i.e., predicting a future value that is going to be measured at a certain time and location, based on values previously measured at that or other neighbouring locations. This differs from spatio-temporal interpolation whose goal is to fill in values that are missing or unknown based on neighbouring values, past or future.

Ignoring dependency relationships between observed values may seriously hinder the per-

formance of forecasting models on this type of application. However, dealing with these dependencies in forecasting does not come without its challenges. For example, spatial and temporal auto-correlation can obfuscate important insights [Malerba, 2008]; time and space can be seen at multiple levels of granularity and of abstraction [Yao, 2003]; spatial objects can be heterogeneous and have complex geometries. These challenges are compounded by the fact that temporal and spatial dependencies both need to be addressed, but these dimensions have very different characteristics: time is generally considered to be one-dimensional, unidirectional and ordered, while space is three-dimensional (although geographical data often just considers two spatial dimensions, e.g., latitude and longitude).

Solutions to these problems have been proposed that are based on (a) data pre-processing, or (b) designing models that account for the data properties, e.g., by extending a time series model to consider spatial relationships, or by applying a graphical model.

Ohashi and Torgo [2012] proposed spatio-temporal indicators, a method based on data pre-processing that uses statistics summarising past data within a spatio-temporal distance of the target observation as predictors. These spatio-temporal indicators assume that data from distant neighbours becomes less relevant as we look further back in time. But some phenomena may take some time to "travel" from one point to another, e.g., wind may carry high concentrations of pollutants released by a factory at different speeds. Thus, we base our alternative proposal on the assumption that, in some cases, it may be advantageous to include older data from distant neighbours and exclude their more recent observations.

1.1.2 Imbalance in Spatio-Temporal Contexts

An important challenge that often arises when devising solutions for predictive problems, including those in spatio-temporal contexts, is that of imbalanced domains. This is a well-studied problem in classification where the definition of imbalance is quite simple: some categories of the target variable are less frequent than others. When addressing numerical regression problems, as in this thesis, the meaning of imbalance is not as straightforward but, in general terms, it means that there are ranges of values that are much less frequent than others (e.g., spikes in fishing hours within marine protected areas).

Many applications include target variables that follow imbalanced distributions. The imbalance, by itself, does not pose an issue even though most numerical regression algorithms and performance metrics are geared towards, respectively, improving and measuring average predictive performance. Thus, they will be more influenced by the cases around the central tendencies of the distribution. The issue only arises when the user also has a preference bias that does not align with the distribution. That is, there may be applications where rare values correspond to a measurement error that the user is not particularly interested in capturing or identifying, so an algorithm that under-performs in these cases but works

well on average is serviceable. However, if the rare or extreme values signal an important phenomenon that the user needs to be able to predict accurately (e.g., a sudden drop in a patient's blood pressure values), then it may be indeed very important for the model, and the performance metrics used to assess it, to capture these extreme values.

There are many different methods for tackling the problem of imbalanced domains in regression, with re-sampling being a common approach [Branco et al., 2016b]. However, few proposals explicitly account for the potential impact of implicit dependency relationships between observations, and those that do mostly focus on time series data [Moniz et al., 2017a]. The combination of spatio-temporal dependency structures with an imbalanced target domain is, therefore, still an open problem, raising many yet unaddressed research questions.

1.1.3 Evaluation Procedures for Spatio-Temporal Forecasting Methods

It is not only when devising predictive models that it is important to account for dependencies in the data: it is crucial to consider these dependencies in the performance assessment phase as well. That is, when trying to determine whether a proposed learning approach will perform well when applied to data that was previously unseen but will, most likely, maintain dependency relationships with the modelled data, a performance evaluation framework should be in place that accounts for these dependencies so that predictive performance is accurately estimated.

Like many of the predictive models that they aim at evaluating, common evaluation procedures such as cross-validation (CV) also make the assumption that the observations in the data sets are independent and identically distributed (i.i.d.). This can (and often does) lead to overly optimistic estimations of error when dependency structures are present [Arlot and Celisse, 2010].

Previous work has addressed the lack of clear guidelines for error estimation procedures of predictive methods when the data exhibits temporal [Bergmeir and Benítez, 2011, 2012, Bergmeir et al., 2014, Cerqueira et al., 2017, Bergmeir et al., 2018, Mozetič et al., 2018] and spatial dependencies [Roberts et al., 2017]. However, though Roberts et al. [2017] mention the added challenges posed by the simultaneous presence of spatial and temporal dependencies, and a preliminary study show how the dependencies affect CV [Meyer et al., 2018], spatio-temporal forecasting evaluation is yet to be investigated in depth.

1.2 Research Aims and Main Contributions

The main goals of this thesis are to (1) provide guidelines for spatio-temporal forecasting performance estimation, and (2) develop new machine learning methods to (a) forecast the future values of geo-referenced time series, and (b) accurately predict rare and extreme values.

1.2.1 Research Questions

Throughout this thesis we plan to answer the following research questions:

RQ1 What should be the guidelines for performance estimation in spatio-temporal forecasting problems?

RQ2 Can we improve the predictive potential of a previously proposed feature engineering method by changing its definition of spatio-temporal neighbourhood?

RQ3 Can we improve standard resampling approaches to imbalanced domains by incorporating spatio-temporal contextual information?

RQ4 How do domain-specific data properties impact the predictive performance of different approaches?

1.2.2 Main Contributions

The work carried out during this thesis led to the following contributions:

1. Conducted an empirical comparative study of performance estimation methods used to evaluate approaches to spatio-temporal forecasting. We apply them to both artificially generated and real-world data, and provide guidelines for practitioners looking to evalute their models;

2. Proposed and evaluated a method for extracting features based on a variation of a previously proposed spatio-temporal neighbourhood;

3. Proposed resampling strategies that improve extreme value prediction by introducing a spatio-temporal bias weight into the random resampling process. We tested our resampling proposals against standard random resampling strategies using several real-world data sets;

4. Carried out meta-analyses investigating how domain-specific data characteristics influence the efficacy of our feature engineering and resampling proposals;

5. Developed the *R* packages *STEvaluation* and *STResampling*, made freely available online. They include implementations of our feature extraction method, the performance estimation methods, and our biased resampling strategies.

1.3 Thesis Outline

We organised the thesis into the six chapters described below.

Introduction The present chapter contextualised the topics tackled by this thesis, motivated our research, and listed our main contributions, which will be detailed over the following chapters.

Literature Review This chapter details the background necessary to explain and motivate our work. We formalise the problems addressed by the thesis, review existing approaches to spatio-temporal forecasting and imbalanced regression, and discuss evaluation methods in the presence of spatio-temporal dependencies.

Evaluating Spatio-temporal Forecasting This chapter presents an empirical study of performance estimation methods (sometimes called evaluation or validation methods), designed to estimate the predictive ability of modelling approaches. We tested both standard methods and methods designed to cope with temporal and spatial dependencies. We used both artificially generated and real-world data sets.

Extracting Spatio-temporal Indicators This chapter focuses on improving spatio-temporal forecasting through the pre-processing of real-world data. We propose a variant of spatio-temporal indicators that can be extracted from geo-referenced time series to serve as predictors in univariate forecasting tasks.

Resampling Imbalanced Spatio-temporal Data This chapter focuses on improving spatio-temporal forecasting under imbalanced domains and non-uniform user preferences. We propose resampling strategies that can improve prediction of extreme and highly relevant values through pre-processing. Our strategies consider both data dependencies and the target domain's distribution to bias originally random processes.

Conclusions We conclude the thesis by summarising and discussing our findings, pointing to future research directions and open issues in spatio-temporal forecasting.

The thesis also includes an Appendix, providing additional information and details when necessary.

1.4 Bibliographic Note

The thesis includes work that has been previously published. We list the references to those publications below.

- Mariana Oliveira, Luís Torgo, and Vítor Santos Costa. Evaluation Procedures for Forecasting with Spatio-Temporal Data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, volume 11051 LNAI, pages 703–718, 2018. ISBN 9783030109240. doi:10.1007/978-3-030-10925-7_43
- Mariana Oliveira, Nuno Moniz, Luís Torgo, and Vítor Santos Costa. Biased Resampling Strategies for Imbalanced Spatio-Temporal Forecasting. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 100–109. IEEE, 2019. ISBN 9781728144931. doi:10.1109/dsaa.2019.00024
- Mariana Oliveira, Luís Torgo, and Vítor Santos Costa. Evaluation Procedures for Forecasting with Spatiotemporal Data. *Mathematics*, 9(6), 2021b. ISSN 2227-7390. doi:10.3390/math9060691
- Mariana Oliveira, Nuno Moniz, Luís Torgo, and Vítor Santos Costa. Biased resampling strategies for imbalanced spatio-temporal forecasting. *International Journal of Data Science and Analytics*, 12(3):205–228, 2021a. ISSN 23644168. doi:10.1007/s41060-021-00256-2

Chapter 2

Background

2.1 Introduction

The literature on Data Mining and Predictive Analytics (PA) for spatio-temporal data is extensive. The topic has only gained more importance in recent years as the volume of easily accessible spatio-temporal data increases. As early as 1999, Roddick and Spiliopoulou provided a bibliography for temporal, spatial, and spatio-temporal data mining research, and rightfully predicted that the interest in the area would only expand over the next years. Since then, several literature review articles have been published on the wider topic of spatio-temporal data mining [Nanni et al., 2008, Mamoulis, 2009, Vatsavai et al., 2012, Rao et al., 2012, Cheng et al., 2014, Shekhar et al., 2015, Chandola et al., 2015, Wikle, 2015, Atluri et al., 2018, Hamdi et al., 2021], describing data types, methods to address different tasks (including predictive modelling), and their applications.

These surveys review the literature from different perspectives or focus on different tasks. For example, Cheng et al. [2014] present existing solutions to spatio-temporal forecasting, clustering, and visualisation problems, while Atluri et al. [2018] eschew data visualisation but overview other tasks including change detection, frequent pattern mining, and anomaly detection. Wang et al. [2020] cover recent deep learning approaches to some of these (and other) problems. Meanwhile, Wikle [2015] review statistical models, and Shi and Yeung [2018] examine sequence forecasting techniques alone. Vatsavai et al. [2012] and Chandola et al. [2015] highlight spatio-temporal "big data" applications, while Shekhar et al. [2015] build a taxonomy of spatio-temporal data types.

Some reviews restrict their analysis to particular application domains. For example, Koehler and Kuenzer [2020] reviewed work on remote sensing – specifically, land surface forecasts –, while other surveys focused on articles using climate data [Faghmous and Kumar, 2014], predicting urban growth [Aburas et al., 2016, Musa et al., 2017], and crop yield [Rembold

et al., 2013]. Over recent years, several review articles have been written on the topic of traffic forecasting alone [Vlahogianni et al., 2004, 2014, Chang et al., 2016, Ermagun and Levinson, 2018, Lana et al., 2018, Nagy and Simon, 2018, Liu et al., 2021].

This chapter aims not to provide an exhaustive literature review, but to overview the necessary background to the research topics addressed throughout the thesis. Through this selective analysis, we hope to both contextualise and motivate our work.

We start by describing the properties of spatio-temporal data in Section 2.2. In Section 2.3, we define our main problem of spatio-temporal forecasting. The dependence between observations in spatio-temporal data poses particular challenges to predictive analytics; we cover different categories of approaches that try to address them in the literature. We focus especially on solutions based on pre-processing since in Chapter 4 we will explore a feature extraction method that falls under this category.

In Section 2.4, we define a special case of our main problem: spatio-temporal forecasting under an imbalanced target domain. The imbalance raises issues that compound with the questions already posed by data dependence. We discuss common approaches to the problem, which we will later contrast with our proposal in Chapter 5.

Finally, in Section 2.5, we investigate how proposals have been evaluated in the literature, from the metrics used to measure performance to the methods employed to estimate them. Properly evaluating a proposed solution's effectiveness is crucial to achieving progress in any area. We discuss evaluation methods that are common practice in predictive analytics. We will see how, once again, issues may arise due to the data dependence in spatio-temporal data. Later on, in Chapter 3, we will carry out our own empirical study of evaluation methods.

2.2 Spatio-temporal Data

Before we can understand spatio-temporal forecasting, we must know the type of data that will be the focus of our studies. In this section, we present a taxonomy of spatio-temporal data types, describe their properties, and the challenges they pose.

2.2.1 Data Types

We consider five different categories of spatio-temporal data: event data, trajectories, point reference data, geo-referenced time series data, and spatio-temporal network data. This thesis will focus solely on geo-referenced time series, but we provide an overview of other data types as well.

Our categorisation differs slightly from the taxonomies defined by Kisilevich et al. [2010]

and Atluri et al. [2018]. Unlike Kisilevich et al., we do not include separate categories for when data keep only the latest temporal snapshot of spatial data; we see these as special cases of trajectory or geo-reference time series data, with no memory. We also added network data as an additional category; Shekhar et al. [2015] considered spatial networks in their spatial data model taxonomy.

2.2.1.1 Event Data

Event data consists of locations associated with a time-stamp or time period where the event occurred (e.g., occurrence of a crime, disease transmission). There may be associated properties describing the event, such as an event type. Events appear within precise temporal boundaries, and their properties do not change over time, since each instance represents something that has already happened.

Event data can either (a) occur instantaneously, being described by a single time-stamp (e.g., a traffic accident), or (b) persist for a period of time, being described by the time interval between the birth and death time-stamp (e.g., a film festival).

Similarly, an event can either (a) happen at a single location, associated with a point in space (e.g., a robbery), or (b) happen across several points in space, associated with a line or polygon (e.g., the area burned by a forest fire). If the events consist of point data, then the collection of point events is sometimes called a spatio-temporal point pattern [Diggle, 2013].

Event data is often categorised separately from other data types in spatio-temporal data taxonomies [Kisilevich et al., 2010, Atluri et al., 2018, Hamdi et al., 2021]. Shekhar et al. [2015] presented a taxonomy that does not consider events to be a spatio-temporal data type, but a way to model the temporal component of spatio-temporal data instead. From this perspective, events or processes are a model for temporal data, and can be combined with three spatial data models (the object model, the field model, and the spatial network model) to model spatio-temporal data.

2.2.1.2 Trajectories

Trajectories denote the paths taken by an object moving across space. If additional information is included, it usually concerns the moving object itself (e.g., the model and make of a taxi, or a variable indicating whether it is moving to pick up or to drop off a client). The focus is on the object and its movement.

Common tasks using these data include forecasting the object's next position or final destination, clustering similar trajectories, detecting outliers, or classifying different types of trajectories.

Several literature reviews cover trajectory data mining and prediction [Kisilevich et al., 2010, Zheng and Yu, 2015, Mazimpaka and Timpf, 2016, Kong et al., 2018].

Trajectories are often considered separately from point reference data [Atluri et al., 2018, Hamdi et al., 2021]. Kisilevich et al. [2010] categorise them as having the temporal extension of a time series, at dynamic locations. When only the latest position of each object is known at any given moment, Kisilevich et al. [2010] separate it into a separate category called *moving object data*.

2.2.1.3 Point Reference Data

Point reference data comprises measurements of a continuous field at changing reference locations (e.g., a weather balloon measuring atmospheric temperature along its trajectory). Even though the sensor may follow a trajectory, the focus of this type of data is external to the object and its movement. Tasks using these data aim at understanding or reconstructing the behaviour of the continuous spatio-temporal field from the finite sample collected by moving sensors.

2.2.1.4 Geo-referenced Time Series

A geo-referenced time series consists of a collection of observations of a spatio-temporal field at fixed locations over a fixed set of time-stamps. This type of data will be the main focus of the thesis.

Locations in geo-referenced time series can either be distributed irregularly across space, or they can be regularly distributed in a grid of equally spaced locations. Likewise, measurements may be taken at regular time intervals, or at varying frequency.

It may be helpful to distinguish between the following two types of data:

Spatial time series use the object (or entity) spatial data model; observations are associated with fixed spatial points, e.g., a ground sensor network. Each spatial object could be stored as vector data containing the spatial coordinates of points, lines, or polygons.

Raster time series data use the field spatial data model, which views data as if produced by a continuous function that measures a certain property across all space [Lloyd, 2010]. Observations are associated with a cell in an (often regular) grid, e.g., a sequence of remote sensing images [Shekhar et al., 2015, 2017]. Each cell can contain aggregate measurements for its area, e.g., the aggregate population living in each cell's area [Atluri et al., 2018].

Though video can be seen as a raster time series, it may be reasonable to consider it separately from spatio-temporal data mining, since it is the main focus of the related field of computer vision and pattern recognition [Wang et al., 2020]. Unlike Wang et al., we did not add video as a separate category, but, like these authors, we will ignore methods designed specifically to deal with video.

Some authors use the term spatio-temporal raster as the generic term including all data that evolve over time at fixed locations [Atluri et al., 2018, Hamdi et al., 2021]. Like us, Kisilevich et al. [2010] use the term geo-referenced time series as the general term instead. They circumscribe raster spatio-temporal data to only refer to data measured over regular spatial grids.

Shekhar et al. [2015] consider spatial time series and raster time series separately, based on the chosen underlying spatial data model. Unlike Kisilevich et al., these authors consider that raster data may also include data evolving over irregular grids. This inclusion can make the distinction between the two types a bit unclear in some cases. However, it reflects a division that is often found in available methods and tools to handle spatial and spatio-temporal data. For example, in the *R* programming environment, one could use tools like the *sp* [Pebesma and Bivand, 2005, Bivand et al., 2013] and *sf* [Pebesma, 2018] packages to represent spatial time series data, while the *raster* [Hijmans, 2020] package would be better equipped to deal with gridded data. Only more recently, the *stars* [Pebesma, 2020] package has emerged, integrating the representations for what they call raster and vector datacubes.

When only the latest measurement at each location is known at any given moment, Kisilevich et al. [2010] categorise the data as an additional class called *geo-referenced variables*.

2.2.1.5 Time-evolving Network Data

The underlying spatial model for network data is the graph. It consists of a set of nodes, representing locations, connected by edges that represent explicit relationships between the locations, e.g., a road network.

Time-evolving attributes can be associated with each of these elements, e.g., traffic flow in a road network. Network data can also change over time through the addition or removal of nodes or edges in the graph, e.g., after construction of a new road.

We added this category to our taxonomy, even though it was not included in the taxonomy that we mostly follow, presented by Atluri et al. [2018]. Spatio-temporal network data previously appeared in the spatio-temporal data model taxonomy by Shekhar et al. [2015, 2017].

2.2.1.6 Converting Between Data Types

It is often possible to convert between some of these data types [Atluri et al., 2018].

We can model geo-referenced time series as a graph, effectively converting it into network data, by making explicit the implicit spatial relationships between locations. For example, we can connect neighbouring weather stations, represented by nodes, so they become explicit neighbours if they are located within a certain distance of each other.

Trajectory data can also be aggregated to generate network data. For example, vehicle trajectories in a road network can be transformed into a network measuring traffic flow.

We can interpolate or aggregate point reference data to build a raster time series. Similarly, we can convert spatio-temporal event data into geo-referenced time series data by aggregating the data into counts of each type of event at different locations. Vector data can represent the spatial dimension of event data, for example, if we aggregate the events within (polygonal) city boundaries, but it can also be aggregated over a regular grid across space [Atluri et al., 2018].

2.2.2 Properties and Challenges

Some data properties are common between spatio-temporal data types. These properties present challenges to data mining, in general, and to predictive modelling, in particular; they also provide opportunities to improve and design methods that adequately address them. Hamdi et al. [2021] carefully reviews this topic. We highlight a few of the most relevant data properties and challenges:

Autocorrelation Standard PA methods tend to assume data to be i.i.d.. But observations in spatio-temporal data are not independent; they correlate with previous and neighbouring observations. Thanks to this autocorrelation, variables often present a degree of smoothness in their changes across space and along time as there is a tendency for values of close observations to be more similar than distant ones (e.g., surface temperature rarely abruptly changes).

Heterogeneity and non-stationarity Spatio-temporal data often are not identically distributed either. It can simultaneously be the case that the data are autocorrelated, but follow different distributions across space and time (e.g., the processes that govern traffic on a busy street during the week may differ from those in a suburb on the weekend). Heterogeneities may require learning different or dynamic models.

Complex and implicit relationships Autocorrelation can be seen as an implicit dependency relationship. But other complex and implicit relationships can be found between

spatio-temporal instances. For example, a region represented by a polygon may have a complex shape that influences how it interacts with other points or polygons that it contains, overlaps with or touches; the distance between two points may implicitly define them as neighbours with differing degrees of relatedness. Though these relationships are implicit, they can still influence data behaviours.

Aggregation effects Since spatio-temporal data consist of discrete representations of often continuous phenomena, the way we aggregate data may influence their behaviour. Results may vary depending on scale – consider the different temperature change patterns found within the day versus along the year; and zoning decisions – consider how gerrymandering may influence election results by manipulating the boundaries of electoral districts, aggregating votes across space differently.

2.3 Spatio-temporal Forecasting

Within Predictive Analytics, this thesis focuses on spatio-temporal forecasting for numerical geo-referenced time series. In this section, we formalise the problem, and overview some common approaches to tackle it.

2.3.1 Problem Definition

Spatio-temporal forecasting aims at predicting the future values of a target variable at a given location. Consider a set of locations $L = \{l_1, \cdots, l_n\}$, a set of time-stamps $T = \{t_1, \cdots, t_m\}$, and a set of observations

$$\begin{aligned} \mathcal{D} \quad = \quad & \{\{y_{1,1}, < x^1_{1,1}, \cdots, x^k_{1,1} >\}, \cdots, \\ & \{y_{i,j}, < x^1_{i,j}, \cdots, x^k_{i,j} > \}^{j \in \{1,2,\cdots,n\}}_{i \in \{1,2,\cdots,m\}}, \end{aligned}$$

where $y_{i,j}$ and $x^k_{i,j}$ correspond, respectively, to the values of the target variable Y and predictors X^k, at time t_i and geographical location l_j. The goal is to predict the value of Y at a location of interest, l_s ($s \in \{1, 2, \cdots, n\}$), at a future time, t_f, given the observed values $y_{i,j}$ and $\mathbf{x_{i,j}}$, such that $t_m < t_f$.

2.3.2 Approaches to Spatio-temporal Forecasting

As we mentioned, most traditional PA methods assume that data are i.i.d.. They also work on a single table, therefore being categorised as *propositional*. Multi-relational methods try to obtain further insight by explicitly considering the complex relationships in the data.

Often, *relational* approaches simply extend a propositional approach to be able to work on multiple tables from a relational database, keeping the single-table approach as a special case (see Džeroski [2003]). However, both types of approaches have their advantages as well as disadvantages.

Whether we are using a propositional or relational approach, there are several ways of tackling the challenges that the spatio-temporal forecasting task brings, and leveraging the contextual information of both spatial and temporal dimensions to our advantage.

Some proposals focus on adapting or combining context-aware learning models [Barber et al., 2010, Appice et al., 2011, Pravilovic et al., 2013, Zheng et al., 2013, McGovern et al., 2014]. Others use feature engineering to encode spatio-temporal contextual information, while taking advantage of off-the-shelf learning algorithms [Ohashi and Torgo, 2012, Appice et al., 2013b, Oliveira et al., 2016, Ceci et al., 2017].

The main types of existing approaches to spatio-temporal forecasting are summarised in Table 2.1, and the main references will be described in the two following sections.

Table 2.1: Approaches to spatio-temporal forecasting

Propositional	Pre-processing based	[Luk et al., 2000, Ohashi and Torgo, 2012, Appice et al., 2013b, Pravilovic and Appice, 2014, Pravilovic et al., 2017, Ceci et al., 2017]
	Combined temporal and spatial methods	[Li et al., 2003, Cheng and Wang, 2008]
	Integrated spatial and temporal dimensions	[Pace et al., 1998, Lindström et al., 2014]
	Extensions to spatial- or temporal-only methods	[Kamarianakis and Prastacos, 2005, Pravilovic et al., 2013, Appice et al., 2011]
	Deep learning	[Wang et al., 2020][1] [Liang et al., 2018, Yu et al., 2018, Wang et al., 2018]
Relational	Graphical models	[Cano et al., 2004, Madadgar and Moradkhani, 2014, Thompson et al., 2007, Ailliot et al., 2009, Barber et al., 2010, Piatkowski et al., 2013]
	Inductive Logic Programming (ILP) based	[Vaz et al., 2011, McGovern et al., 2014]

[1] This work reviews deep learning approaches to spatio-temporal data mining, including predictive modelling

2.3.3 Propositional Approaches to Spatio-temporal Forecasting

Approaches to spatio-temporal forecasting can be divided into solutions based on (a) pre-processing, (b) the combination of spatial and temporal methods, (c) the integration of the spatial and temporal dimensions into the model, (d) the extension of spatial interpolation or temporal forecasting methods to account for the other dimension.

2.3.3.1 Pre-processing Based

Pre-processing approaches have the advantage of letting the user apply any off-the-shelf learning algorithm, so they can capitalize on novel, efficient learning algorithms. They

achieve this by (a) using lagged temporal and/or spatial inputs, or (b) computing other spatio-temporal features.

Lagged Temporal and/or Spatial Inputs. The simplest pre-processing approach is simply to use historical data, from the same location or from its neighbours to help predict future values.

Luk et al. [2000] transformed rainfall data before feeding it to an Artificial Neural Network (ANN). The rainfall values measured over the past k timestamps (the temporal lag) at M different locations served as input to predict a vector of the future rainfall values measured at N locations. The authors found an apparent trade-off in predictive accuracy between the inclusion of temporal and spatial information: it was more beneficial to include a shorter temporal lag when using more neighbouring spatial inputs.

Bilgili et al. [2007] predicted monthly average wind speed at a target location, using as inputs to an ANN the corresponding month, and neighbouring values at locations that correlated highly with the target. Though this solution was designed for missing data interpolation, the authors note that the method could easily be adapted to predict future values.

Other Spatio-temporal Indicators. Some approaches consider more refined spatio-temporal indicators. For example, Ceci et al. [2017] combined separate spatial and temporal features designed to capture autocorrelation when predicting photovoltaic energy production. The authors found that encoding the hour and day was enough to account for temporal autocorrelation, while Principal Coordinates of Neighbor Matrices (PCNM) [Dray et al., 2006] worked well as a representation of spatial autocorrelation.

Some authors calculate other indicators within neighbourhoods that can be defined using a measure of spatio-temporal distance [Ohashi and Torgo, 2012], or following a clustering step (e.g., [Appice et al., 2013b]).

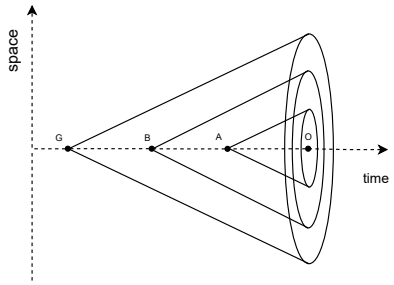


Figure 2.1: Spatio-temporal neighbourhoods of different sizes as defined by Ohashi and Torgo [2012]

Ohashi and Torgo [2012] pre-processed wind speed data to predict future wind speed values based on historical data, using several standard regression models. The method considers three neighbourhoods, defined by a boundary of spatio-temporal distance of increasing size to each observation, O. The neighbourhoods resemble cones (see Figure 2.1), with a base centered on the observation, and a spatial radius that decreases for older observations.

The computed predictors (called indicators) summarize the values within each neighbourhood, and include (a) lagged values from the target location, (b) the average and standard deviation of the values within each of the neighbourhoods, (c) a weighted average of the values, where each observation's contribution is inversely proportional to their spatio-temporal distance to O, and (d) ratios between the indicators of successively larger neighbourhoods.

This approach inspired our work in Chapter 4, where we investigated whether gains could be made by reversing the direction of this spatio-temporal neighbourhood, in the hopes of capturing phenomena that "travel" through time and space differently.

Spatio-temporal Clustering Based Existing approaches using clustering for data pre-processing follow the clustering step by (a) a standard learning technique (e.g., [Appice et al., 2013b]), or (b) a temporal forecasting model (e.g., [Pravilovic et al., 2017]).

Spatio-temporal clustering followed by standard learning technique. Appice et al. [2013b] computed similar indicators to Ohashi and Torgo [2012], and also followed the pre-processing step by a standard regression method. Instead of defining spatio-temporal neighbourhoods based on a boundary of spatio-temporal distance, Appice et al. automatically discovered neighbourhoods through spatio-temporal clustering. To find spatio-temporal clusters, a spatial-aware clustering algorithm first groups together regions of similar values at each temporal snapshot; then, locations classified into the same sequence of spatial clusters over a temporal window are grouped into the same neighbourhood.

A related approach, described by Appice et al. [2013a], was applied to spatio-temporal data stream interpolation instead of forecasting. The method discovers trend clusters online, summarizing observed data. In a later offline stage, inverse distance weighting helps reconstruct observed data and recover missing values.

Spatio-temporal clustering followed by temporal forecasting technique. The method proposed by Pravilovic and Appice [2014] uses a spatio-temporal k-means to cluster temporally similar time series from nearby locations. Then, an algorithm selects Auto-Regressive Integrated Moving Average (ARIMA) model parameters that provide the best fit for all time series in each cluster. Finally, ARIMA models fitted to each time series, using the parameters determined for their cluster, can produce accurate forecasts.

Pravilovic et al. [2017] also included a spatio-temporal clustering step to define neighbourhoods, but the authors post-process the summary statistics using Principal Component

Analysis (PCA) to remove collinearity before modelling the data. Instead of training a global standard regression model, they learn a local vector autoregression (VAR) model for the resulting multi-variate time series at each location.

2.3.3.2 Combined Temporal and Spatial Methods

These approaches build pipelines combining methods designed for temporal forecasting, like the ARIMA models, with spatial prediction methods. An example is the work of Li et al. [2003] who proposed a framework to predict water flow rate comprising three steps. First, build a time series model of the data at each location to produce temporal forecasts. Then, train an ANN to learn spatial correlations between locations, and feed the trained network with the temporal forecasts of neighbouring locations to produce a spatially-influenced forecast for the target location. Finally, combine the temporal forecast with the spatially-influenced forecast through regression to obtain a final forecast for the target location. Cheng and Wang [2008] substituted the static feed-forward ANN in this framework by a dynamic recurrent one with feedback connections to predict wildfire area.

2.3.3.3 Integrated Spatial and Temporal Dimensions

Some authors more explicitly integrate the spatial and temporal dimensions in their models.

Pace et al. [1998] presented a parsimonious model to estimate housing prices which improved on the traditional hedonic pricing model with indicator variables. The proposed method generalized the traditional auto-regressive error model to explicitly account for dependence in both time and space.

Lindström et al. [2014] extended previous work on air pollution forecasting [Szpiro et al., 2010] to include spatio-temporal covariates, such as the output of a deterministic point-prediction air pollution model. The authors' proposed method can also include geographic covariates, and it was implemented in an R package [Lindström et al., 2013].

2.3.3.4 Extension of Spatial- or Temporal-Only Methods.

Instead of combining methods in a pipeline, some choose to extend methods designed for temporal or spatial data and extend them to also consider the other dimension.

Kamarianakis and Prastacos [2005] and Pravilovic et al. [2013] proposed different extensions of the ARIMA model, a time series forecasting technique, to account for spatial information. Meanwhile, Appice et al. [2011] proposed an extension of Geographically Weighted Regression (GWR), a spatial interpolation technique, so that it is able to transfer knowledge across time.

2.3.3.5 Deep Learning

Deep learning has gained attention over the last several years. Many approaches to spatio-temporal data mining, including predictive modelling, have been proposed (e.g., [Liang et al., 2018, Yu et al., 2018, Wang et al., 2018]). Wang et al. [2020] reviews the most important works in the field. The authors separate between methods designed to work on what we defined as spatial time series versus raster time series.

2.3.4 Relational Approaches to Spatio-temporal Forecasting

We divide relational approaches to spatio-temporal forecasting into (a) graphical models, and (b) ILP based solutions.

2.3.4.1 Graphical Models.

We categorize graphical models into Bayesian Networks (BNs), Hidden Markov Models (HMMs), and Markov Random Fields (MRFs).

Bayesian Networks. Probabilistic methods such as BNs have the advantage of producing probability estimates for their forecasts. Examples using BNs for spatio-temporal forecasting include the works by Cano et al. [2004] and Madadgar and Moradkhani [2014].

Cano et al. [2004] introduced a local learning algorithm that leverages the spatial dependencies in meteorological data to reduce the complexity of the search process in a "score and search" BN learning method; thus, reducing computing time.

Madadgar and Moradkhani [2014] used BNs to forecast future drought conditions within a river basin, given historical runoff data. The authors used copula functions to simplify the complex conditional probability density function estimation problem.

Hidden Markov Models (HMMs). Thompson et al. [2007] generalized a multisite daily rainfall generation model, reformulating it using a HMM. The proposal introduces an extra variable describing a local "weather type" at each location, modelling the data using separate HMMs, but capturing spatial dependence using copulas. Applying the general model to real data produced reasonable results, but the lower levels of spatial correlation in the model's predictions show that its spatial dependence model could still be improved.

Ailliot et al. [2009] later proposed a method that modelled rainfall data using a hidden "regional" weather type, common to all locations, to capture temporal dependence in the data via a HMM. Then, conditional on weather type, censored, power transformed, Gaussian

distributions model the spatial dependence. This proposal better captured the spatial dependence of the data, but could not capture local dynamics of rainfall as accurately.

Barber et al. [2010] proposed a missing data tolerant approximate inference method for Auto-Regressive Hidden Markov Model (AR-HMM), built to produce short-term wind forecasts.

Markov Random Fields (MRFs). Similar to BNs, MRFs are graphical probabilistic models that can naturally represent complex spatial and temporal dependencies in data. But these models can be computationally costly.

Piatkowski et al. [2013] focused on efficiency and scalability when designing Spatio-Temporal Random Fields (STRFs), a model based on MRFs. Their proposal included an optimization algorithm that can be run in parallel, and a novel parametrization method that reduces storage space requirements.

2.3.4.2 Inductive Logic Programming (ILP) Based

ILP allows for explicit modelling of relationships between entities in the data, e.g., encoding neighbouring relationships between locations as background knowledge.

In our previous work [Oliveira et al., 2016], we used ILP as a pre-processing step to extract propositional features, so we could apply standard propositional regression algorithms to predict yearly burn fractions (due to wildfires) of Portuguese civil parishes. We will mention this work again in Section 2.4 as it tackled forecasting under imbalanced domains. The approach was not completely relational, as the final predictions were produced by propositional methods.

Previously, Vaz et al. [2011] had used an ILP system paired with a logic-based spatial database to predict fire occurrence. The approach dynamically explores spatial predicates during the ILP search process, reducing computational costs.

McGovern et al. [2014] presented improved versions of their previously proposed spatio-temporal relational probability tree and the spatio-temporal relational random forest, using them to predict hazardous weather events.

2.4 Spatio-Temporal Forecasting with Imbalanced Domains

The problem of imbalanced domains is common to both classification and regression, but much more attention has been paid to solving the problem in classification. Branco et al. [2016b] thoroughly review the state of the art for both imbalanced classification and regression. In this thesis, we focus on numerical problems, so in this section we will define

the imbalanced regression problem in a spatio-temporal setting, provide a brief overview of methods used to solve it in general, and discuss methods previously used specifically in spatio-temporal settings.

2.4.1 Problem Definition

A spatio-temporal forecasting task can be said to be imbalanced when certain ranges of values in the target variable Y are more important to the end-user, but severely under-represented in the training data, e.g. predicting extreme levels of pollution.

The representation bias (imbalance) in the data often leads models to optimize the prediction of values around the central tendencies of the distribution, and to the detriment of accurate predictions of extreme values [Moniz et al., 2017b]. Given that events with extreme values often represent situations of high importance and interest to the end-users, our objective is improving the predictive ability on such cases.

To formalise the prediction task, we need to specify what is meant by "highly important" values of the target variable. In this thesis, we resort to the work by Torgo & Ribeiro [Torgo and Ribeiro, 2007, Ribeiro, 2011]. The authors use a *relevance function* to map the domain of the target continuous variable into a $[0, 1]$ scale of relevance, i.e. $\phi(Y) : \mathcal{Y} \rightarrow [0, 1]$. This notion of a relevance function will be particularly important in Chapter 5, where we propose methods to improve extreme case prediction in spatio-temporal forecasting.

When possible, an expert can determine which ranges of the target variable hold special interest based on their knowledge of the domain. But human experts may be unavailable, and automatically deriving a relevance function becomes necessary. Ribeiro [2011] proposed an automated approach to define the relevance function using box plot statistics. The approach attributes the maximum relevance of 1 to outlier values according to box plot statistics (either extreme high or low), and the minimum relevance of 0 to the target's median value. Then, the relevance of the remaining values is interpolated using a piecewise cubic Hermite interpolation polynomials [Dougherty et al., 1989] (*pchip*) algorithm.

Finally, the user needs to define a relevance threshold, t_R, above which the cases are considered to be extreme. This leads to the formalisation of two subsets of the data set, containing the cases with normal and extreme values, respectively: $\mathcal{D}_N = \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) < t_R\}$ and $\mathcal{D}_R = \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) \geq t_R\}$, where $|\mathcal{D}_R| \ll |\mathcal{D}_N|$. Despite this, the threshold does not serve the purpose of discretisation. It instead allows the authors to propose specialised pre-processing methods, which we will discuss in the next section, and define regression metrics that take into account the predictions' numerical error and not merely their presence in either $\mathcal{D}_N$ or $\mathcal{D}_R$, which we present in Section 2.5.2.2.

2.4.2 Approaches to Imbalanced Regression

An abundance of strategies to deal with imbalanced domains exist, although many of them are geared towards classification. In their survey, Branco et al. [2016b] divided modelling strategies to address imbalanced tasks into four general categories: data pre-processing, special-purpose learning algorithms, prediction post-processing, and hybrid methods.

Data pre-processing strategies have the advantage of allowing practitioners to apply any state-of-the-art algorithm to the transformed data, directly benefiting from any advancements in modelling algorithms. Branco et al. [2016b] groups pre-processing approaches into three subcategories: resampling, active learning, and weighing the data space.

Active learning requires actively selecting the best samples to learn the model, and weighting the data space requires information about the model's misclassification costs. Resampling simply changes the data distribution to force the learner to focus on the least represented cases, but it has proven effective in the classification of rare cases in many imbalanced domains [Sun et al., 2009]. Resampling has long been used to help detect oil spills [Kubat et al., 1998, Chawla et al., 2002], network intrusions [Cieslak et al., 2006], or machine and software failures [Kamei et al., 2007, Bennin et al., 2016]), sort texts [Estabrooks et al., 2004, Moreo et al., 2016], diagnose disease [Mazurowski et al., 2008, Fotouhi et al., 2019], predict bankruptcy and analyse financial data [Zhou, 2013, Le et al., 2018].

Random under-sampling and random over-sampling are two of the simplest existing resampling strategies. In a two-class problem, random under-sampling removes a random set of majority class cases from the training data, while random over-sampling adds a random set of replicas of minority class cases to the data. These methods can remove useful examples or exacerbate over-fitting, respectively. In both cases the ideal target distribution might not be easy to determine. However, these have still proven to be efficient methods of dealing with the imbalance problem [Batuwita and Palade, 2010, Fernández et al., 2008]. Gaussian noise can be added to the rare case replicas during oversampling to prevent model overfitting [Lee, 1999].

In 2013, Torgo et al. extended some common imbalanced learning strategies to numerical prediction for the first time. The techniques work by choosing a relevance function and threshold, as we presented in the previous section, to define the pools from which to randomly select the observations that will be added or removed from the data. Later, Branco et al. [2016a] adapted random over-sampling with added Gaussian noise from classification to regression as well.

Even though more informed methods exist, resampling still works well enough by selecting the observations to add or remove from the data set randomly. However, as we will see, there may be benefits in introducing a bias to the sampling process.

2.4.3 Approaches to Imbalanced Spatio-Temporal Forecasting

Although the relevance of forecasting rare cases or extreme values in spatio-temporal data is considerable [Yang, 2006], only a small fraction of the body of work specifically tackles imbalanced problems [Zheng et al., 2013, McGovern et al., 2014, Oliveira et al., 2016]. Moreover, as we mentioned above, the great majority of work on this topic addresses the prediction of rare cases in classification tasks [Zheng et al., 2013, McGovern et al., 2014]. We will highlight our previous work [Oliveira et al., 2016] and the work of Moniz et al. [2017a], tackling numerical prediction of extreme values.

In our previous work [Oliveira et al., 2016], we applied a pre-processing approach to the prediction of areas burnt by wildfires. As mentioned in Section **??**, we used ILP to extract features encoding spatio-temporal information from the data set. We then used random resampling to improve the predictions of extreme values produced by standard regression algorithms. While the feature engineering step took into account the spatio-temporal context of the problem, the random resampling strategy employed ignored it.

In the work of Moniz et al. [2017a], the authors proposed resampling strategies designed specifically to address imbalanced time series forecasting tasks. They incorporated into random resampling a vector weighting observations according to the relevance of each case and their recency. The focus of this work was solely on the temporal dimension of the data, but it inspired our proposal in Chapter 5, which will take into account both the temporal and spatial dimensions of geo-referenced time series.

2.5 Estimating Performance

As we have seen so far in this chapter, researchers have designed many predictive solutions for geo-referenced time series data. We must assess their effectiveness properly before we decide on which predictive strategy to deploy in a certain application.

An evaluation framework should be able to accurately estimate the performance a predictive approach would have on unseen data. When evaluating a solution, we should select: (a) evaluation metrics that fit the application's goals; and (b) an evaluation method that can estimate them accurately, making the best use of available data.

Some of the most common methods in predictive analytics, such as cross-validation, assume that observations are independent and identically distributed [Arlot and Celisse, 2010]. The dependency between observations in spatio-temporal data can, therefore, negatively impact the accuracy of these evaluation methods.

Evaluation metrics face other issues. Some of the most widely used metrics evaluate performance based on an approach's average behaviour. This can become a problem when distribu-

tions are imbalanced and we are especially interested in accurately predicting rare or extreme values. In that case, we should evaluate approaches using alternative metrics [Branco et al., 2016b].

In this section, we start by exploring existing evaluation methods; then, we briefly discuss appropriate evaluation metrics for different scenarios.

2.5.1 Evaluation Methods

An evaluation method needs to estimate how a model would perform on unseen data. This can be accomplished by following these common steps: (1) divide the available data into training and test sets; (2) build a model based on the training data; (3) use the model to make predictions about the test set; and (4) calculate performance metrics, comparing the predictions to the real observations in the test set. If using more than one test set, then we can estimate the loss by aggregating multiple estimates, e.g., by averaging them.

When the methodology aims to both select an approach and estimate its performance, the evaluation methods should ideally be *nested* [Krstajic et al., 2014]. That is, an internal evaluation method, only having access to the external training set, estimates the performance of a set of candidate models; then, based on the internal estimates, the best model is selected, and the external evaluation method estimates its performance. If there are multiple training and test sets, the process can be repeated for each external test set. The internal and external evaluation methods may be the same, or they can differ. If the candidate models use the same algorithm but different hyper-parameters, we can call this methodology *internal parameter tuning*. Some authors find using nested CV to tune hyper-parameters and select a model to be overzealous in most applications [Wainer and Cawley, 2021].

We start by discussing general methods used in Predictive Analytics, but move on to discuss methods designed for time series, spatial data and, finally, for geo-referenced time series.

2.5.1.1 General Methods

Generally we can think of evaluation methods as belonging to one of two classes: *out-of-sample (OOS)* estimation and *cross-validation (CV)* strategies. We use CV to refer to methods that use each subset in a data partition as a test set at least once [Arlot and Celisse, 2010]. We use OOS to refer to other methods that hold out observations for validation, including hold-out and prequential methods [Cerqueira et al., 2017].

Out-of-sample (OOS) methods. The simplest of OOS estimators is *hold-out validation*. It splits the data into two complementary sets: a training set, and a testing (or validation) set. An algorithm builds a model based on the data from the training set, and

its loss is estimated on the held out, "unseen" data from the test set [Devroye and Wagner, 1979]. We can repeat the process several times, dividing the data randomly each time, and average the results to obtain a more robust estimate. Repeated holdout is sometimes called random subsampling [Kohavi, 1995].

If nesting hold-out methods, it is common to call the internal test set, used to select an approach, the "validation set"; and the external test set, used to estimate the selected approach's performance, the "test set".

Cross-validation (CV) methods. CV splits the data into several similarly sized sets, then uses each set to test a model trained on the remaining data. Averaging the loss across the test sets leads to a final estimate [Stone, 1974]. Arlot and Celisse [2010] review the statistical properties of several CV methods, and their role in model selection.

Data splitting in CV may be exhaustive or partial. Partial splitting is often more computationally viable. We highlight a classic example of each type:

Leave-one-out cross-validation (LOOCV) uses exhaustive splitting. Each observation successively plays the role of test set, with the remaining observations used for training the model.

K-fold CV uses partial splitting. First, it partitions the data into K subsets of approximately the same size (called folds or blocks). Then, it uses each subset to test a model that was trained on the remaining $K - 1$ folds [Geisser, 1975]. It is common practice to randomly shuffle the data before partitioning.

As we discussed in Section 2.2.2, spatio-temporal data often shows autocorrelation. But standard CV methods, such as K-fold CV, require data to be i.i.d., ensuring independence between the training and test sets. Otherwise, the methods may be overly optimistic when estimating the error a model will incur when presented with previously unseen data, which can cause models to overfit and generalize poorly [Roberts et al., 2017]. More than one study has shown how CV overfits the choices of bandwidth for a kernel estimator in regression [Opsomer et al., 2001, Diggle et al., 2002].

Some CV variants have been proposed specifically for dependent data. Though most seem geared toward time series [Chu and Marron, 1991, Burman et al., 1994, Racine, 2000], methods specifically for spatio-temporal data also exist [Meyer et al., 2018]. Next, we present methods designed specifically for data with temporal, spatial and spatio-temporal dependence.

2.5.1.2 Methods for Temporal Dependence

Once the temporal dimension is involved, we restrict OOS methods to refer to methods that inherently respect the temporal order of the data, training models using historical data and testing on more recent data. These approaches are sometimes also called "last-block" methods [Bergmeir and Benítez, 2012].

OOS methods in time series must choose a split point to divide the data between training and testing. Instead of using the whole data to estimate error, they may only train and test the models over a certain window of time. In that case, they must also choose the window size, and whether it slides or grows as time progresses. We mention two approaches:

Repeated time-wise holdout Tashman [2000] recommends practitioners to repeat holdout methods over different periods of time to obtain more robust estimates. The selection of split points for each repetition of time-wise holdout may be randomized, with a window of preceding observations used for training and a fraction of the following instances used for testing. Training and test sets may potentially overlap across repetitions, similarly to random sub-sampling. These are also referred to as Monte Carlo experiments [Torgo, 2016];

Prequential evaluation (or interleaved-test-then-train evaluation) is often used in data stream mining. Each observation (or block of non-overlapping observations) is first used to test and then to train the model [Modha and Masry, 1998] in a sequential manner. The term prequential usually refers to the case where the training window is growing, i.e., a block of observations that is used for testing in one iteration will be merged with all previous blocks and used for training in the next iteration.

We highlight four alternatives to standard CV proposed for time series:

Modified CV (or non-dependent CV [Bergmeir and Benítez, 2012]) Similar to K-fold CV, except that l observations preceding and following the observation(s) in the test set are discarded from the training set after shuffling and fold assignment [Chu and Marron, 1991] Bergmeir and Benítez [2012] refer to it as non-dependent CV;

Block CV A method similar to K-fold CV where each fold is a sequential, non-interrupted time series [Snijders, 1988], instead of each fold containing a random subset of observations;

h-block CV Based on LOOCV, except it removes h observations preceding and following the observation in the test set from the training set [Burman et al., 1994];

hv-block CV A modification of h-block CV that tests the models on a block of v observations preceding and following each observation, instead of single observations. This

causes test sets to overlap, which is uncommon in CV. It still removes h observations from the training set, before and after each testing block [Racine, 2000].

The test sets in all types of block-CV consist of either single observations, or sequences of non-interrupted observations. In contrast, a fold in modified CV will almost certainly include non-sequential observations. If K is set to the number of observations, modified CV is equivalent to h-block CV as long as $l = h$.

Whether we should use blocked CV or an OOS method to evaluate time series models is a contested issue. Bergmeir et al. found non-buffered blocked CV preferable to OOS methods, based on synthetic [Bergmeir and Benítez, 2011] and real-world experiments [Bergmeir and Benítez, 2012, Bergmeir et al., 2014]. Bergmeir et al. [2018] especially argued its benefits for small time series, but Cerqueira [2019] found no evidence of this in their experiments.

Cerqueira et al. [2017] agree that CV may work well on synthetic, stationary time series, but argue that OOS methods like repeated time-wise holdout estimate error better in real-world scenarios, regardless of stationarity. Mozetič et al. [2018] corroborated the notion that respecting the temporal ordering of data is important, but they did not test repeated holdout.

Later, Bergmeir et al. [2018] proved a theorem showing that standard CV estimates error better than holdout, as long as (a) the data is generated by a stationary non-linear auto-regressive process, and (b) the model properly fits it (i.e., errors must be serially uncorrelated); they back this hypothesis using one real-world example.

2.5.1.3 Methods for Spatial Dependence

A major change when switching from temporal dependence to spatial dependence is that there is not a clear unidirectional ordering of data in 2D- or 3D- space as there is in time. This precludes adapting prequential evaluation strategies to the spatial domain, but other strategies can be easily adapted. CV methods are widely used to evaluate spatial data models. We highlight two categories of CV variants that aim at minimizing spatial dependence between training and testing.

Block CV As in the temporal case, blocks can be designed to include neighbouring geographic points, forcing testing on more spatially distant observations, and thus decreasing spatial dependence and mitigating error under-estimation error [Trachsel and Telford, 2016]. Roberts et al. [2017] highlight three ways of grouping observations into folds: (a) **systematic**, assigning observations following a pattern across space (e.g., a checkered pattern); (b) **contiguous**, allocating to the same fold neighbouring observations in contiguous blocks; and (c) **optimized random**, assigning observations

randomly, but choosing the final assignment so that it minimises dissimiliarity between folds;

Buffered CV Buffered CV is a general term commonly used to refer to methods that would correspond to h-block or hv-block CV in the spatial domain, where a geographic vicinity of the testing block is removed from the training set to create a "buffer" breaking the dependence between training and testing sets.

Roberts et al. [2017] empirically tested the validity of some of these methods following a different experimental design from the empirical studies on time series. Where Bergmeir and Benítez [2012] used the results on previously withheld data containing the most recent observations as the "gold standard" error against which error estimates could be compared, Roberts et al. defined that "gold standard" as the average error incurred by predicting onto independent runs of their synthetic data generation process, which simulated artificial species abundances data that depended on spatially autocorrelated "environmental" variables.

Roberts et al. found the best estimator to be block CV with a block size substantially larger than residual autocorrelation, and "buffered" LOOCV, a spatial version of h-block CV, with h equivalent to the distance at which residual autocorrelation is zero.

In this thesis, we work on the forecasting problem, and thus expect that our models might have to extrapolate in their predictions. However, if models are only meant to interpolate, Roberts et al. [2017] recommend that blocks should be no larger than necessary, but models should be trained with as much data as possible, and predictors should be equally represented across blocks or folds. Conservatively large blocks help avoid overly optimistic error estimates, but increase the probability that models will be forced to extrapolate. They suggest the "optimised random" or systematic (patterned) assignment of blocks to folds to minimise the need for extrapolating.

2.5.1.4 Methods for Spatio-Temporal Dependence

To evaluate predictive modelling approaches to spatio-temporal forecasting, authors often resort to one of the methods described in previous sections, effectively treating the data as if it was spatial-only (e.g., [Haberlandt, 2007]) or temporal-only (e.g., [Appice et al., 2013b, Ceci et al., 2017]). Others may treat the problem mostly from a temporal perspective, but break down the results across space (e.g., [Ohashi and Torgo, 2012]), or vice-versa (e.g., [Carroll and Cressie, 1997]), without the evaluation method itself being specifically designed to accommodate this.

Recent work by Meyer et al. [2018] highlights how, for spatio-temporal interpolation problems, the results of conventional CV differ from what they call "target-oriented" CV – versions of CV that address the temporal, spatial or both dimensions, namely, leave-location-

out (LLO), leave-time-out (LTO) and leave-location-and-time-out (LLTO), which work as their name suggests. The authors attribute the lower error estimated by standard CV to spatio-temporal over-fitting of the models, and propose a forward feature selection method to improve data interpolation.

Roberts et al. [2017] did not experiment on spatio-temporal data, but the authors mention that data often shows dependencies in both time and space and provide a general guide to blocking CV. They recommend a five-step workflow: (1) assess dependence structures in the data; (2) determine prediction objectives; (3) block according to objectives and structure; (4) perform CV; and (5) make "final" predictions. The different objectives may be, for example, interpolating data within well-known regions, or extrapolating data to previously unseen locations with different underlying characteristics.

It is worth exploring solutions that consider autocorrelation along the temporal, spatial or both dimensions, but the optimal strategy may indeed depend on the modelling goal. Here, we must once again distinguish between spatio-temporal interpolation and forecasting problems.

In spatio-temporal interpolation, models may not generalise well under different conditions; the goal is moreso to be able to fill in data within well-known regions. Forecasting aims to predict future values, perhaps including new locations, and we expect models to extrapolate if necessary. We depart from the work by Meyer et al. [2018] as we focus on forecasting where they focused on interpolation. Even after that is established, the best evaluation method might still depend on whether the goal is to predict future values in known locations or previously unseen locations.

2.5.2 Performance Metrics

Most metrics designed to measure performance in regression tasks focus on the average loss. We need alternative metrics if our target domain follows an imbalanced distribution, and we consider the rare and extreme cases to be highly relevant, and hope to predict them with increased accuracy [Branco et al., 2016b]. In classification, metrics that can deal with the imbalanced problem are already standard, and some have inspired solutions for regression. In the next sections, we introduce classification metrics to provide context for the regression metrics we will use throughout the thesis.

2.5.2.1 Classification Metrics

Consider a two-class problem, with a positive (or minority) class and a negative class. Though the positive class appears less frequently, it is no less important than the negative class. The confusion matrix of a classifier (see Table 2.2) shows the number of cases correctly

classified as True Positives (TP) and True Negatives (TN). The wrongly classified instances are reported as False Positives (FP), also called Type I errors, and False Negatives (FN), also called Type II errors.

Table 2.2: Confusion matrix for a two-class classification problem

		Predicted	
		Positive	Negative
True	Positive	TP	FN
	Negative	FP	TN

Accuracy (Equation 2.1) is one of the most frequently used metrics in classification that can be extracted from this matrix, but it fails to capture the usefulness of a solution under imbalanced domains.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.1}$$

The less frequent, minority class influences accuracy less, making it an inappropriate metric to evaluate predictive methods under imbalanced domains. An alternative to this metric is the well-known precision-recall framework [Davis and Goadrich, 2006] (see Equations 2.2 and 2.3).

$$\text{precision (positive predictive value)} = \frac{TP}{TP + FP} \tag{2.2}$$

$$\text{recall (true positive rate, sensitivity or hit rate)} = \frac{TP}{TP + FN} \tag{2.3}$$

There is a trade-off between the two measures and monitoring them simultaneously is impractical. The F_β-measure or F_β-score, based on Van Rijsbergen [1979]'s effectiveness measure, combines both these metrics through a harmonic mean, attaching β times as much importance to recall as precision (see Equation 2.4).

$$F_\beta = \frac{(1 + \beta)^2 \cdot \text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \tag{2.4}$$

This precision-recall framework will be the basis for the regression metrics we present on the next section, and use to evaluate our proposals in Chapter 5.

2.5.2.2 Regression Metrics

Standard metrics for regression include Mean Squared Error (MSE) and Mean Absolute Error (MAE), sometimes called Mean Absolute Deviation (MAD), defined in Equations 2.5 and 2.6, where y_i is the ground truth, and $\hat{y}_i$ the model's prediction.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.5}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.6}$$

These metrics can be designed to integrate context about the range of target values, and facilitate comparisons between results on different data. We will see two alternative definitions for the MAE, both sometimes called Normalized Mean Absolute Error (NMAE). The error can be normalised by the mean target value in the test set (Equation 2.7). We can normalize it instead by the error that would be incurred by a naive forecaster, e.g., a forecaster that always predicts the mean value in the training set, $\bar{y}$ (Equation 2.8) – also called Relative Absolute Error (RAE).

$$NMAE = \frac{\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\frac{1}{n}\sum_{i=1}^{n} y_i} \tag{2.7}$$

$$NMAE \text{ (or } RAE) = \frac{\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\frac{1}{n}\sum_{i=1}^{n}|y_i - \bar{y}|} \tag{2.8}$$

Equation 2.9 presents a third alternative, called Mean Absolute Percentage Error (MAPE). The small difference between Equation 2.9 and 2.7 means that MAPE will be more sensitive to outliers, but it cannot compute error when the true values are zero, and it values negative errors more than positive errors.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{2.9}$$

As in classification, these metrics are inadequate when dealing with imbalanced domains given the considerable representation bias between cases with values around the central tendency of the distribution and those with extreme values. These standard, average-based metrics will lead to models optimizating to reduce the models' error on the average case.

One possible solution is to use the Receiver Operating Characteristic for Regression (RROC) [Hernández-Orallo, 2013]. The curve plots the total under-estimation against the total over-

estimation. The Area Over the Curve (AOC) corresponds to the error variance.

Another option would be the Regression Error Characteristic (REC) curves [Bi and Bennett, 2003] that plot the accuracy against the error tolerance of a regression function. Error tolerance is defined as the percentage of points predicted within a certain tolerance ϵ. In this case, the AOC corresponds to a biased estimate of the expected error.

Torgo [2005] proposed Regression Error Characteristic Surfaces (RECS), which add an extra dimension to the REC to represent the cumulative distribution of the target variable. This shows the error's behaviour across different ranges of the target variable domain – particularly important if we expect lower errors within specific ranges of the target domain.

Torgo and Ribeiro [2007] and Ribeiro [2011] introduced the concept of the precision-recall evaluation framework into the context of utility-based regression. Utility is commonly defined as a function combining positive benefits and negative benefits (costs). The authors' proposed utility score considers two factors: (a) the relevance of the true, y, and predicted values, $\hat{y}$, given a user-specified continuous relevance function ϕ, mapping the target variable domain into relevance values within [0,1], and (b) the magnitude of the predictive error, given by a standard loss function and bounded by a maximum admissible loss threshold. Perfect predictions receive non-negative utility, with no costs and benefits equal to the predicted values' relevance. When predictions incur in some error, utility takes into account their magnitude: predictions reasonably close to the true values have non-negative utility; but, as the difference between predicted and true values increases, utility becomes negative, tending to -1. Equation 2.10 defines utility [Ribeiro, 2011],

$$U^p_\phi(\hat{y}, y) = B_\phi(\hat{y}, y) - C^p_\phi(\hat{y}, y) = \phi(y) \cdot (1 - \Gamma_B(\hat{y}, y)) - \phi^p(\hat{y}, y) \cdot \Gamma_C(\hat{y}, y) \tag{2.10}$$

where ϕ^p is the joint relevance function, i.e., a weighted average of the relevance values of y and $\hat{y}$ where the penalisation factor p is the weight of the former, and Γ_B and Γ_C are bounded loss functions in the scale [0,1], with respect to benefit and cost threshold functions.

Based on previous work by Torgo and Ribeiro [2009] and Ribeiro [2011], Branco [2014] proposed the measures of precision and recall for regression defined by Equations 2.11 and 2.12.

$$\text{precision}\phi^u = \frac{\sum\limits_{\phi(\hat{y}_i) > t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(\hat{y}_i) > t_R} (1 + \phi(\hat{y}_i))} \tag{2.11}$$

$$\text{recall}\phi^u = \frac{\sum\limits_{\phi(y_i) > t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(y_i) > t_R} (1 + \phi(y_i))} \tag{2.12}$$

These metrics can be simplified further [Moniz et al., 2019] to match the definitions in Equations 2.13 and 2.14,

$$prec^u_\phi = \frac{\sum\limits_{\phi(\hat{y}_i)\geq t_R, \phi(y_i)\geq t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(\hat{y}_i)\geq t_R} (1 + \phi(\hat{y}_i))} \tag{2.13}$$

$$rec^u_\phi = \frac{\sum\limits_{\phi(\hat{y}_i)\geq t_R, \phi(y)\geq t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(y_i)\geq t_R} (1 + \phi(y_i))} \tag{2.14}$$

where $\phi(y_i)$ and $\phi(\hat{y}_i)$ is the relevance associated with the true value y_i and predicted value $\hat{y}_i$, respectively; t_R is a user-defined relevance threshold, above which cases are signalled as highly relevant for the user, and $u(\hat{y}_i, y_i)$ is the utility of making the prediction $\hat{y}_i$ for the true value y_i, normalized to $[-1, 1]$.

Like in classification, the utility-based F-Score metric F^u_β combines both precision ($prec^u_\phi$) and recall (rec^u_ϕ) with a harmonic mean. We will evaluate our proposals in Chapter 5 using the F^u_β-score based on the simplified definitions of utility-based precision and recall in Equations 2.13 and 2.14. We should stress that, unlike the traditional F-Score metric used in classification tasks, this formulation of the utility-based F-Score is based on the analysis of numerical prediction errors, and not exclusively on whether they were classified into the right "bin" above or below the relevance threshold.

More recently, Ribeiro and Moniz [2020] proposed a new metric, Squared Error-Relevance Area (SERA), which does not require setting a relevance threshold.

2.6 Summary

Spatio-temporal data describe many real-world phenomena. We categorize them into five types: event data, trajectories, point reference data, geo-referenced time series, and time-evolving network data. Data can sometimes be converted into a different type. In this thesis, we focus on geo-referenced time series where numeric variables are measured at regular intervals over a set of fixed locations (e.g., weather stations measuring temperature every hour).

Predictive Analytics (PA) can be used to predict future values at multiple locations – a task called spatio-temporal forecasting. However, PA approaches and the performance estimation methods used to assess them face issues due to the data properties of spatio-temporal data. Most methods assume data to be i.i.d., but spatio-temporal data are often autocorrelated. Other challenges arise from potential heterogeneities and non-stationarities in the data, the

complex and implicit relationships between data instances, and scale and zoning effects.

Methods designed to tackle spatio-temporal forecasting can be divided between propositional approaches, working on a single table, and relational approaches, that can work on multiple, related tables. Propositional approaches can roughly be divided into the following categories: pre-processing based approaches (including methods using spatio-temporal clustering), combined temporal and spatial methods, approaches that integrate the spatial and temporal dimensions, methods that extend spatial or temporal-only methods, and deep learning. Relational approaches can broadly be categorised into graphical models, and ILP based methods. Pre-processing approaches have the advantage of allowing the use of any regression algorithm to learn a model and make predictions.

If the target variable follows an imbalanced distribution and the extreme cases are important to the user, we face an imbalanced spatio-temporal forecasting problem. Methods to deal with the imbalance problem can be divided into four categoories: data pre-processing, special-purpose learning algorithms, prediction post-processing, and hybrid methods. Random resampling strategies are simple, yet effective data pre-processing methods that allow the user to use any off-the-self regression algorithm to learn a model of the data that will focus on predicting rare and extreme cases.

Chapter 3

Evaluating Spatio-temporal Forecasting

3.1 Introduction

Widespread sensor networks generate ever-larger quantities of geo-referenced time series data. Machine learning models can leverage these data and help guide decisions with real-world impact: monitoring air and water quality [Liang et al., 2018], predicting photovoltaic energy production [Ceci et al., 2017]. Forecasting models must accurately predict future numeric values at multiple geographical locations, but their predictive ability can only improve if we can trust our projections of their performance on unseen data.

Performance estimation has two main functions: (a) to help analysts select the best possible forecaster; and (b) to provide end-users with reliable estimates of its future performance. Adequate performance estimation is only possible if we answer two questions: (a) which metrics best serve the application's aims, and (b) which evaluation method best exploits the available data to estimate them accurately. This chapter will focus on the second question in spatio-temporal forecasting.

Most evaluation methods work by (1) partitioning the data into training and test sets, then (2) building models on the training sets, and (3) computing performance metrics' statistics for the test sets. Standard evaluation methods such as CV often assume independence between observations in the training and test sets [Arlot and Celisse, 2010], but spatial and temporal autocorrelation break this assumption, which may lead to overly optimistic loss estimates. Several proposals for variations of CV address this issue by partitioning the data in different ways; most specialise on time series [Chu and Marron, 1991, Burman et al., 1994, Racine, 2000], but some work in spatio-temporal settings [Meyer et al., 2018].

Previous empirical studies on performance estimation methods for dependent data focused

on either temporal [Bergmeir and Benítez, 2012, Bergmeir et al., 2014, Cerqueira et al., 2017, 2020, Mozetič et al., 2018] or spatial data [Roberts et al., 2017], even though Roberts et al. do mention that spatial and temporal dependencies frequently co-exist. In 2018, Meyer et al. compared CV methods for spatio-temporal interpolation. This study was small in scope as it only compared three variations of CV on two data sets, so the question remained: which evaluation methods best estimate model performance on spatio-temporal data? Our study compares the loss estimated by over 15 different methods against the "gold standard" loss measured in previously withheld observations on 192 artificial and 17 real-world data sets, and using four different learning algorithms.

3.2 Materials and Methods

The different estimation methods we compared are presented in Section 3.2.1. We investigate their performance on randomly generated artificial spatio-temporal data (described in Section 3.2.2.1) as a foundation for better understanding their properties; we then test them in real-world case studies (described in Section 3.2.2.2). Section 3.2.3 describes the experimental methodology we followed [1].

3.2.1 Estimation Methods

The estimators tested in this thesis included time-wise holdout methods – one-time holdout *(HO)* and repeated Monte Carlo holdout *(MC)* –, Cross-Validation *(CV)*, and prequential evaluation methods *(Preq)*.

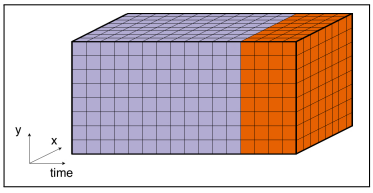
3.2.1.1 Time-wise Holdout Methods

Time-wise holdout methods select one or several split-points across time, train a model on older observations, and test it on more recent data.

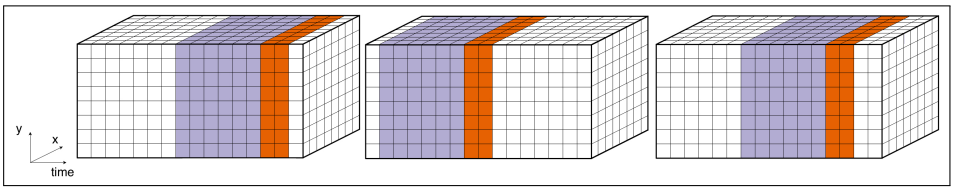
Time-wise Holdout (H): One split-point in time is chosen, with all previous observations used for training and the remaining used for testing (see Figure 3.1a). The split-point is chosen accordinng to a user-selected train/test ratio;

Repeated time-wise Monte Carlo Holdout (MC): Several data-split points are randomly generated; models are trained on a fixed-size window of previous observations and tested on a fixed-size window of the following observations (see Figure 3.1b). The window size should respect the desired train/test ratio and guarantee that enough

[1] All code and data necessary for replication of these experiments is freely available at `https://github.com/mrfoliveira/STEvaluation-MDPI2021`.



(a) Time-wise Holdout (H)



(b) Time-wise Monte Carlo (MC)

Figure 3.1: Time-wise holdout methods. Observations used for training in lighter lilac; observations used for testing in dark orange. Time flows left to right

observations remain outside of the train and test windows at each iteration to make the random generation of split points meaningful.

3.2.1.2 Cross-Validation and Prequential Methods

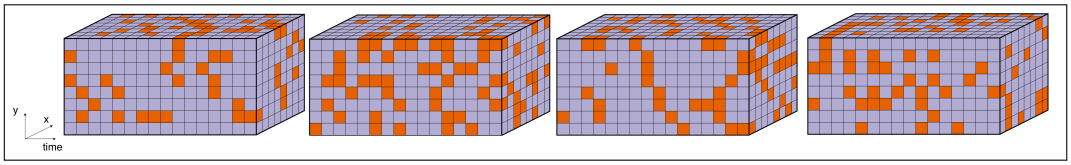
K-fold cross-validation methods split the data into K partitions (also called blocks, or folds), either randomly or according to some criteria; each partition is used as a test set once to estimate the performance of a model trained on the remaining K-1 partitions. Prequential methods split the data according to temporal order ; each partition is used as a test set once to estimate the performance of a model trained on those partitions that contain only older data.

We tested variations of cross-validation with several fold-allocation criteria that do not have a prequential equivalent:

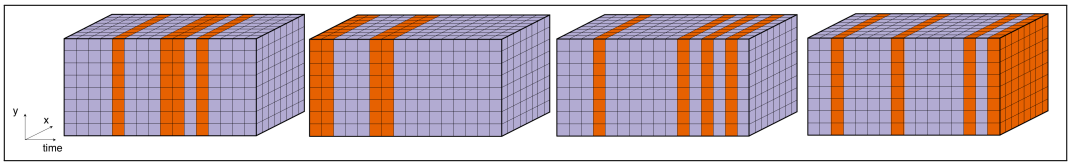
Standard cross-validation (CV): Observations are randomly assigned to folds, irrespective of both temporal and spatial dimensions (see Figure 3.2a);

Time-sliced CV (CV-Tsl): All instances recorded at a given time (time-slices) are randomly assigned to folds, ignoring the spatial dimension (see Figure 3.2b);

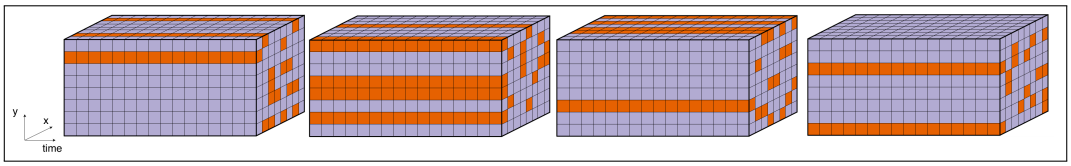
Spatial block CV: Locations are grouped together in folds or spatial blocks, either randomly (**CV-Sb**), in contiguous geographic regions (**CV-Sb-cont**), or in a systematic, checkered pattern (**CV-Sb-sys**), ignoring the temporal dimension (see Figures 3.2c, 3.2d, and 3.2e). Spatial block CV is also referred to as "leave-location-out" CV.



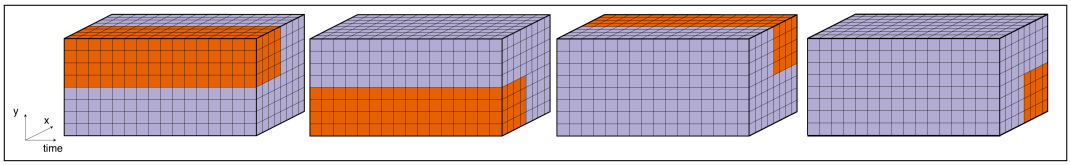
(a) Standard CV (CV)



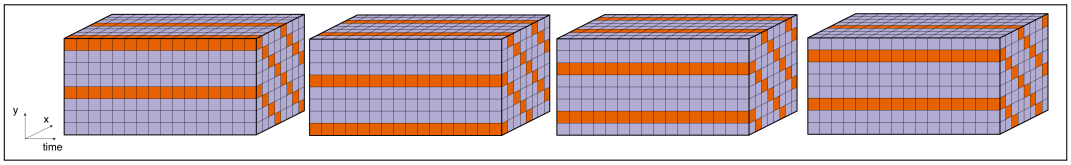
(b) Time-sliced CV (CV-Tsl)



(c) Spatial-block CV (CV-Sb)



(d) Contiguous spatial-block CV (CV-Sb-cont)



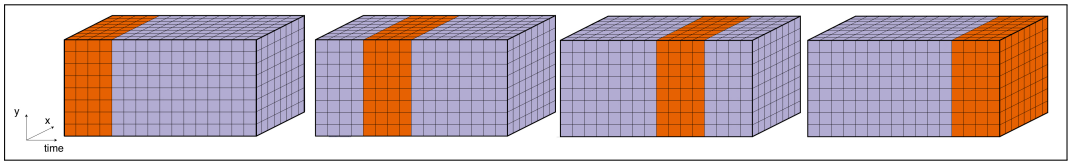
(e) Systematic spatial-block CV (CV-Sb-sys)

Figure 3.2: Cross-validation methods without prequential equivalents. Folds used for training in lighter lilac; folds used for testing in dark orange. Time flows left to right

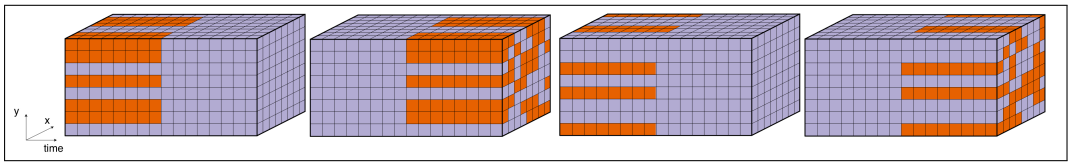
When the fold allocation criteria divides time into blocks of sequential observations, it becomes possible to use both cross-validation and prequential methods. We tested the following variations of cross-validation with several fold-allocation criteria and their prequential equivalents, which respect temporal order:

Time-block CV (CV-Tb): Sequential blocks of time are assigned to each fold, ignoring the spatial dimension. This type of CV is also referred to as "leave-time-out" CV (see Figure 3.3a). For prequential evaluation (**Preq-Tb**), see Figure 3.4a;

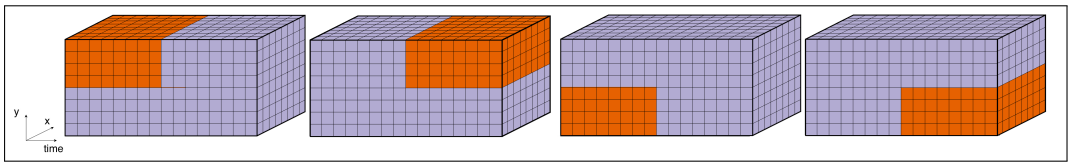
Spatio-temporal-block CV: Time is grouped in sequential blocks and locations are simultaneously grouped together randomly (**CV-STb**), in contiguous geographic re-



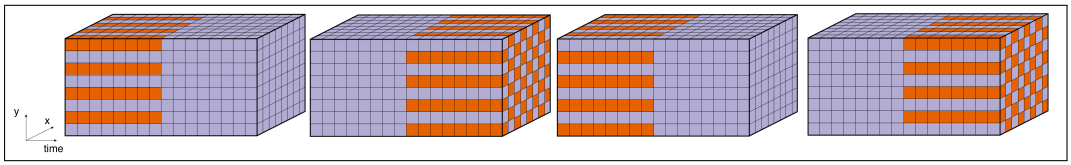
(a) Time block CV (CV-Tb)



(b) Spatio-temporal block CV (CV-STb)



(c) Spatio-temporal contiguous block CV(CV-STb-cont)

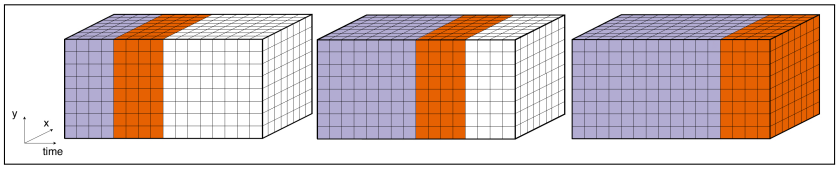


(d) Spatio-temporal systematic block CV (CV-STb-sys)

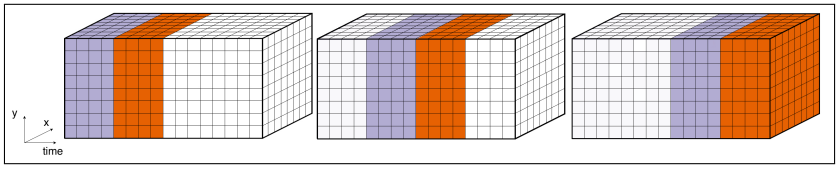
Figure 3.3: Block cross-validation methods that have prequential equivalents. Folds used for training in lighter lilac; folds used for testing in dark orange. Time flows left to right

gions (**CV-STb-cont**), or in a systematic, checkered pattern (**CV-STb-sys**); see Figures 3.3b, 3.3c, and 3.3d. For prequential evaluation, see Figures B.1a (**Preq-STb**), B.1b (**Preq-STb-cont**), and B.1c (**Preq-STb-sys**) in the Appendix.

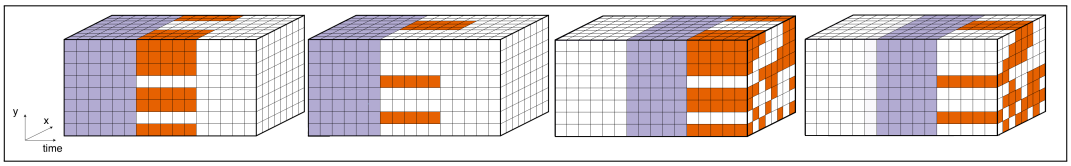
In prequential methods, the spatial region in the test set can optionally be removed from the training set – we use the suffix *-rmS* to indicate this (see Figure 3.4d). The number of blocks in time used for training can increase at each step in time, also called a growing window (suffix *-grW*), as in Figure 3.4a and 3.4d, or it can be fixed at a certain number, named a sliding window (suffix *-slW*), as in Figures 3.4b and 3.4c. The growing window exploits all past data for model training, while the sliding window maintains consistency in training size and, thus, in train/test ratio. In some domains, especially if there is concept drift, using a sliding window to "forget" older data may benefit forecasters.



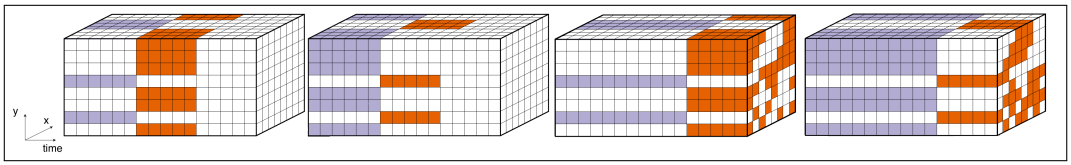
(a) Time block prequential evaluation with growing window (Preq-Tb-grW)



(b) Time block prequential evaluation with sliding window (Preq-Tb-slW)



(c) Spatio-temporal block prequential evaluation with sliding window (Preq-STB-slW)



(d) Spatio-temporal block prequential evaluation with growing window and spatial region removal (Preq-STb-rmS)

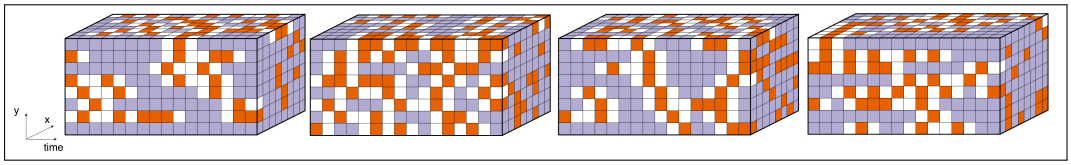
Figure 3.4: Variations of prequential evaluation methods. Blocks of data used for training in lighter lilac; blocks of data used for testing in dark orange. Time flows left to right

3.2.1.3 Buffered Cross-validation

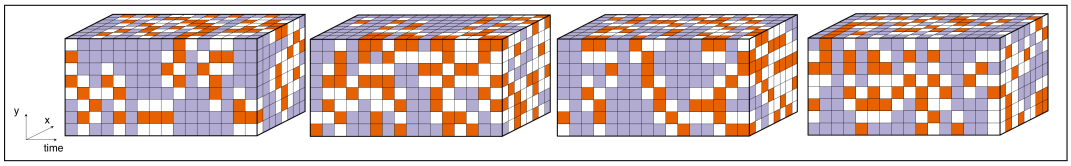
We also tested methods that remove a block of observations in the vicinity of the test set from the training set to try to break dependence between them:

Time-, space-, and space-time-buffered CV: For each instance in the test set, removes from training a number of past and future observations at that location and/or observations within a certain distance from the location (**CV-Tbuf**, **CV-Sbuf**, or **CV-STbuf**; see Figures 3.5a to 3.5c). This is akin to modified CV for time series, but adapted to the spatio-temporal context;

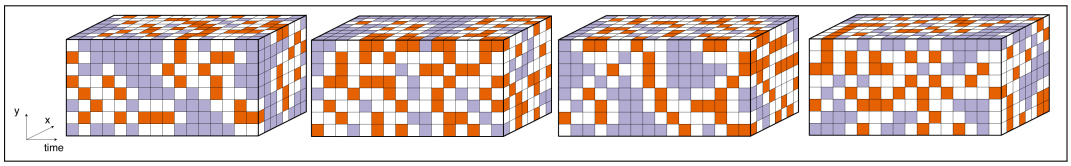
Time-buffered temporal-, and spatio-temporal-block CV: Remove from training a number of past and future observations around the testing block (**CV-Tb-Tbuf**). This is similar to *hv*-block CV for time series, but while *hv*-block CV repeats the method for each instance of the set, causing overlap between test sets, we only repeat the method once for each non-overlapping block of sequential time. The same principle



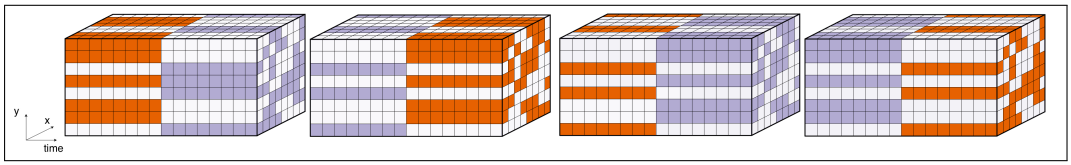
(a) Time-buffered CV (CV-Tbuf)



(b) Space-buffered CV (CV-Sbuf)



(c) Space-time-buffered CV (CV-STbuf)



(d) Leave-location-and-time-out CV (CV-mSTMbuf)

Figure 3.5: Buffered cross-validation. Folds used for training in lighter lilac; folds used for testing in dark orange; buffer observations in white

can be applied to spatio-temporal-block CV since it also blocks data in time (**CV-STb-Tbuf**);

Space-buffered spatial-block CV: Removes from training concurrent observations at locations within a pre-defined spatial distance of the testing set, whether they were allocated randomly (**CV-Sb-Sbuf**), or in contiguous blocks (**CV-Sb-cont-Sbuf**). This is also similar to *hv*-block CV, adapted to the spatial context.

Maximum space-time-buffered spatio-temporal block CV: Removes from training all observations recorded anywhere within the time period of the test set, and all observations recorded at any time at the locations included in the test set – also called "leave-location-and-time-out" CV (**CV-STb-mSTbuf**; see Figure 3.5d).

Table 3.1 summarises the different train/test assignment criteria used for CV and prequential evaluation methods.

Table 3.1: Cross-validation and prequential evaluation fold assignment methods.

Type	Name	Time	Space	Acronym
Cross-validation	Standard	random	random	CV • † ‡
	Time-sliced		all	CV-Ts
	Spatial block	all	random block	CV-Sb •
	Checkered spatial block		systematic	CV-Sb-sys
	Contiguous spatial block		contiguous	CV-Sb-cont •
Cross-validation & Prequential evaluation	Time block	block	all	CV-Tb †
	Spatio-temporal block		random block	CV-STb ‡
	Spatio-temporal systematic block		systematic	CV-STb-sys
	Spatio-temporal contiguous block		contiguous	CV-STb-cont

† Time-buffered CV variation included.
• Space-buffered CV variation included.
‡ Space-time buffered CV variation included.

3.2.2 Data Sets

This study uses both synthetic and real-world data sets. We describe both next.

3.2.2.1 Artificial Data Sets

We generated 192 artificial data sets using the stationary spatio-temporal autoregressive moving average (STARMA) models proposed by Pfeifer and Deutsch [1980] and implemented in the *R* package *starma* [Cheysson, 2016].

The data we generated simulates the output of a regularly spaced sensor network measuring the values of a stationary variable at multiple fixed locations along time. Each artificial observation depends on previous values measured at that location and at its neighbours.

STARMA models. Considering L fixed locations in space, observations of a random variable are generated for T time periods at each location.

A STARMA($p_{\lambda_1\lambda_2\ldots\lambda_p}, q_{m_1m_2\ldots m_q}$) model is specified by Equation 3.1 [Pfeifer and Deutsch, 1980],

$$\begin{aligned}\mathbf{z}(t) = &\sum_{k=1}^{p}\sum_{i=0}^{\lambda_k}\phi_{kl}\mathbf{W}^{(l)}\mathbf{z}(t-k) \\ &-\sum_{k=1}^{q}\sum_{i=0}^{m_k}\theta_{kl}\mathbf{W}^{(l)}\boldsymbol{\epsilon}(t-k)+\boldsymbol{\epsilon}(t)\end{aligned} \tag{3.1}$$

where $\boldsymbol{z}(t)$ is a $L \times 1$ vector of observations at time t, I is the identity matrix, $W^{(l)}$ is a $L \times L$ square matrix of weights where element (i, j) is only non-zero if locations i and j are neighbours of l^{th} order with rows summing to one, p is the autoregressive order, q is the moving average order, λ_l is the spatial order of the k^{th} autoregressive term, m_k is the spatial order of the k^{th} moving average term, ϕ_{kl} and θ_{kl} are parameters, and the $\epsilon_l(t)$ are random normal errors respecting Equations 3.2 and 3.3.

$$E[\epsilon_l(t)] = 0 \tag{3.2}$$

$$E[\epsilon_l(t)\epsilon_j(t+s)] = \begin{cases} \sigma^2 & l = k, s = 0 \\ 0 & otherwise \end{cases} \tag{3.3}$$

Non-linear versions of STAR models can be generated by applying a non-linear function f to $\mathbf{z}(t-k)$ at each autoregressive step – inspired by how Bergmeir and Benítez [2012] obtained non-linear autoregressive time series.

$$\mathbf{z}(t) = \sum_{k=1}^{p} \sum_{l=0}^{\lambda_k} \phi_{kl} \mathbf{W}^{(l)} f(\mathbf{z}(t-k)) + \boldsymbol{\epsilon}(t) \tag{3.4}$$

We considered models where $p_{\lambda_1 \ldots \lambda_p} = q_{\lambda_1 \ldots \lambda_q}$, $p = 0$, or $q = 0$. We will refer to them as follows:

$STARMA(p_{\lambda_1 \ldots \lambda_p})$ denotes a model where the autoregressive and moving average components are of the the same order ($p = q$ and $\lambda_1 \ldots \lambda_p = m_1 \ldots m_q$);

$STAR(p_{\lambda_1 \ldots \lambda_p})$ denotes an autoregressive model ($q = 0$);

$STMA(q_{m_1 \ldots m_q})$ denotes a moving average model ($p = 0$);

$NLSTAR(p_{\lambda_1 \ldots \lambda_p})$ denotes a non-linear autoregressive model ($q = 0$).

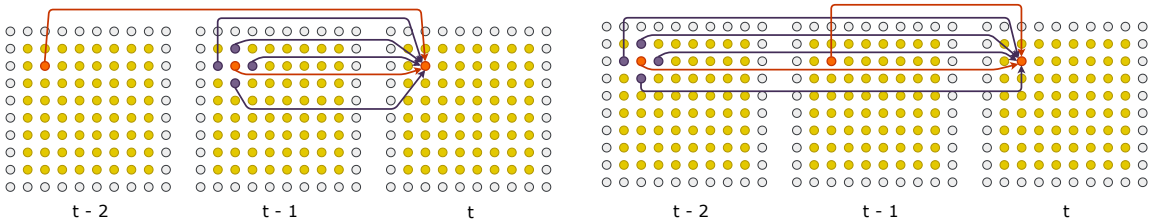
Locations were arranged on equally spaced grids; points directly above, below, and to the right and left of a target location were considered to be first-order neighbours. To avoid dependence on initial conditions, we discarded the first 100 values for each location; to avoid missing data, we discarded data from outer locations in the grid. We generated data with all combinations of parameters shown in Table 3.2.

For each model type and model order, we generated four sets of coefficients that produced stationary models. Coefficients were randomly generated within intervals likely to respect stationary conditions, but they were only kept if they did so; otherwise, they were discarded, and a new set was generated until four different sets were found. To obtain the non-linear

Table 3.2: Parameters used for artificial data sets generation. All combinations of the following parameters were used to generate the data sets: four types of learners, two different grid sizes, two different time series lengths, and four sets of random stationary coefficients for each of three model orders – totalling 192 different artificial spatio-temporal data sets. Outer locations on the grid were discarded to avoid missing data; the first 100 values in the time series were discarded to avoid dependence on initial conditions.

Parameter	Values taken during data generation
Model type	STARMA ($p = q$), STMA, STAR, NLSTAR
Coefficient order (4 sets of each)	2_{10}, 2_{01}, 2_{11}
Grid size (before discarding)	8×8 (10×10), 20×20 (22×22)
Time series length (before discarding)	150 (250), 300 (400)

auto-regressive models (NLSTAR), we randomly selected a non-linear function from a set of five options. Appendix A.1 goes into more detail about the data generation process.



(a) $STAR(2_{10})$ (b) $STAR(2_{01})$

Figure 3.6: Dependence in $STAR(2_{10})$ and $STAR(2_{01})$ data used in the experiments. Circles represent locations; arrows show direct dependence between values for the example orange location at different times; its first-order neighbours are shown in purple. Locations in grey were discarded

Figure 3.6 shows examples of a $STAR(2_{10})$ and a $STAR(2_{01})$. In the case of $STAR(2_{10})$, a value observed at location l and time t will depend directly on values measured at: (a) time $t-1$, at locations l and its first-degree neighbours; (b) time $t-2$, at location l.

Spatio-Temporal Embedding We will be using standard off-the-shelf regression algorithms for data modelling. Before the data can be fed to these algorithms, they need to be transformed into a table where rows correspond to artificial observations and columns contain values that can help predict them. We achieved this through a spatio-temporal embedding that used historical data as predictors. That is, each target value generated for location l and time t we used as predictors the values measured at: (a) the target location l at times $t-1$, $t-2$, and $t-3$, and (b) its first-degree neighbours (top, bottom, left, and right) at times $t-1$ and $t-2$. If we used the same notation for the spatio-temporal embedding as the one used for the STARMA models, the embedding order could be denoted as 3_{110} (see Figure 3.7 for a visual representation).

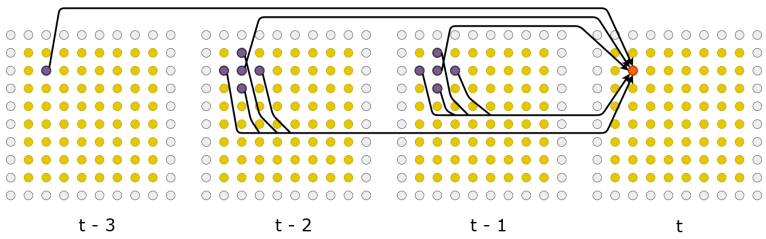


Figure 3.7: Illustration of the 3_{110} spatio-temporal embedding used in the experiments. Circles represent locations; arrows connect values used as predictors for a target observation, pictured in orange at time t. Locations in grey were discarded

3.2.2.2 Real-World Data Sets

We tested the evaluation methods on several real-world data sets, measuring environmental variables, from seven different sources. Each data source includes one or more environmental variables.

Table 3.3 shows a summary of the main characteristics of the real-world data sets used in our experiments. Data consist of series of 105 to over 6.000 values[2] measured at less than 100 different locations, irregularly distributed across space – the only exception is data set 5, with its over 900 locations regularly distributed in a 0.5×0.5 degrees grid. In 9 out of the 16 variables, values are missing for 26-51% of the time-stamp/location pairs, either because sensors were set up at later times, deactivated early, or otherwise failed to acquire data.

Spatio-Temporal Indicators Like artificial data, these data must be transformed into a table format that allows a standard regression algorithm to learn. Unlike artificial data, our real-world data are mostly irregularly distributed across space, so a simple spatio-temporal embedding as the one used above seemed inadequate.

We extracted features following the methods proposed by Ohashi and Torgo [2012], which we will explore further in the next chapter. The features consist of a temporal embedding, and summaries of past values within spatio-temporal neighbourhoods of the target observation. In total, each data set had 20 predictors. Additional details can be found in Appendix A.2.

3.2.3 Experimental Design

In this section, we present the evaluation method used to assess the accuracy of errors estimated by the evaluation methods detailed in Section 3.2.1. We also present the error metrics used in this study and the learning process applied to each training set to obtain a prediction model.

[2]The number in data set 7 was larger in the raw data, but we are presenting the values after we calculated medians per hour per location.

Table 3.3: Description of real-world data sets, including the total number of available observations, and the percentage of all possible combinations of location and time-stamp that they represent

ID	Data source	ID	Variables	Time-stamps	Frequency	Nb. loc.	Network	Nb. obs.	Avail. (%)
1	MESA Air Pollution[1]	10	NO_X conc.	280	bi-weekly	20	irregular	5.6k	100
2	NCDC Air Climate[1]	20	precipitation	105	monthly	72	irregular	7.6k	100
		21	solar energy						
3	TCE Air Climate[1]	30	ozone conc.	330	hourly	26	irregular	8.6k	100
		31	air temperature	360				9.4k	
		32	wind speed	360				9.4k	
4	Cook Agronomy Farm[2]	40	water content	729	daily	40	irregular	22.3k	73
		41	temperature					22.5k	74
		42	conductivity					22.5k	74
5	SAC Air Climate[1]	50	air temperature	144	monthly	900	regular	130k	100
6	airBase[3]	60	PM10 conc.	4382	daily	70	irregular	149k	49
7	Beijing UrbanAir[4]	70	NO_X conc.	6.6k	hourly	36	irregular	152	64
		71	PM_{10} conc.					155k	66
		72	wind speed					161k	68
		73	PM_{25} conc.					162k	68
		74	humidity					162k	69
		75	air temperature					163k	69

[1] From Pravilovic et al. [2018]; Downloaded at: `http://www.di.uniba.it/appice/software/COSTK/data/dataset.zip`, accessed on 12 March 2018;
[2] Loaded from R package GSIF [Hengl, 2017, Gasch et al., 2015] version 0.5-5.1 (`https://cran.r-project.org/web/packages/GSIF/index.html`, accessed on 9 December 2020);
[3] Loaded from R package spacetime [Pebesma, 2012] version 1.2–3 (`https://cran.r-project.org/web/packages/spacetime/index.html`, accessed on 9 December 2020);
[4] From Zheng et al. [2013]; Downloaded at: `https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/Air20Quality20Data.zip`, accessed on 18 October 2017; Since there was more than one measurement for some hours we rounded the time-stamps to the closest hour, and calculated the median values per hour and location.

3.2.3.1 Error Estimation Assessment

Figure 3.8 shows our experimental design, which consists of the following three steps:

1. Each data set was divided into an in-set, consisting of 80% of the oldest observations, and an out-set, consisting of 20% of the most recent ones;

2. A regression model was trained on data from the in-set, and its loss calculated on the test set – this loss was later used as the "gold standard" that evaluation methods should accurately approximate;

3. Each of the evaluation methods described in Section 3.2.1 was used to estimate the error using data from the in-set exclusively – this estimate was compared against the "gold standard" error to assess its usefulness.

The process was repeated for all combinations of data sets (described in Section 3.2.2) and the four different learning algorithms listed below.

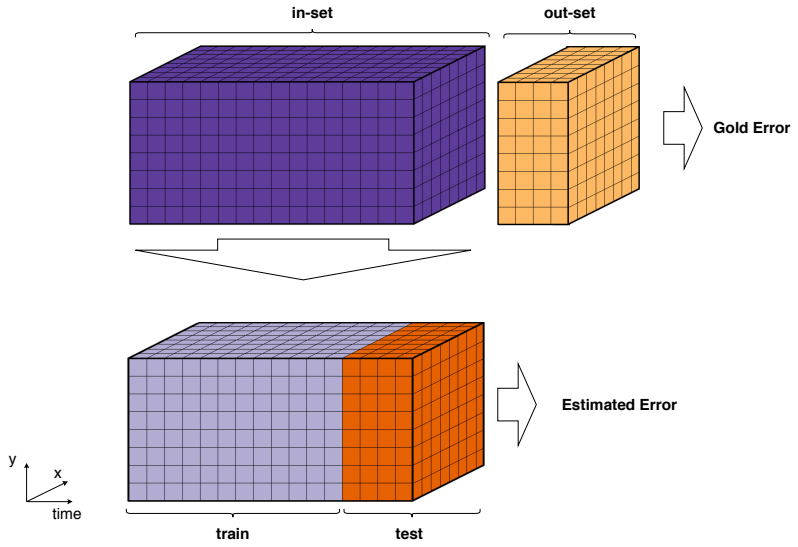


Figure 3.8: Experimental design for spatio-temporal evaluation methods assessment. Data is divided into an in-set and out-set, respecting temporal order. The error on the out-set is considered the "gold standard"; different estimation methods are used to estimate error on the in-set (in this example, time-wise holdout); these values are then compared.

Table 3.4: Training and test set sizes tested for each data set type and evaluation method, as percentages of in-set size. Repeated holdout (MC) involves as many repetitions as the number of folds in CV

Data type	Evaluation method	Train size (%)	Test size (%)	Total (%)	Test/train ratio
Artificial data	CV	94 (15 folds)	6 (1 fold)	100	0.07
	Holdout	94	6	100	0.07
	Holdout	80	20	100	0.25
	MC	55	4	60	0.07
	MC	47	3	50	0.06
Real-world data	CV	89 (8 folds)	11 (1 fold)	100	0.12
	Holdout	89	11	100	0.12
	Holdout	80	20	100	0.25
	MC	53	7	60	0.13
	MC	44	6	50	0.14

Train/test sizing. Table 3.4 summarises the different train and test sizes used for each type of estimation methods and data type as a percentage of the in-set. For CV variants, we used the same number K of folds regardless of blocking type – this means that the blocks in temporal-block CV will span shorter time-intervals than the blocks in spatio-temporal-block CV. For repeated holdout (MC), we used sizes that compare well with the 15/1 and

8/1 train/test ratio in CV, with the same number of repetitions, *nreps*, as the K folds in CV; for one-time holdout, we used one set that compared well with that ratio, and one that matched the 80/20 in-set/out-set ratio.

Ten is a commonly used value for both K and *nreps* in the literature [Cerqueira et al., 2020], but we used $K = nreps = 16$ and $K = nreps = 9$ for artificial and real-world data, respectively, since these values were the closest to $K = 10$ that allowed for simple and equitable divisions along the spatial, temporal, or both dimensions for all the evaluation methods tested in each scenario.

3.2.3.2 Training Workflow

Since we already extracted features before carrying out any experiments, the training workflow only needs two steps: (1) pre-processing to deal with missing data; and (2) training a model using a standard regression algorithm.

Missing data. In the real world, sensors may be set up later or fail to acquire data, leading to *missing data*. We found this problem to be severe for data sets 4, 6, and 7 where a sizeable fraction of all the time-stamps and location pairs were missing (from 49% to 74%). Since our chosen predictors summarise past neighbouring values, missing values can percolate to their neighbours.

Most learning models cannot cope with missing data. There are two main ways to deal with missing values: deleting predictors or observations containing missing values, or filling them in. Missing data can be filled in through imputation, for example, by filling it in with a central value, or through more complex model-based methods [Little and Rubin, 2019]. Discarding incomplete observations causes information loss, but filling in missing data may bias the model – both should be mitigated.

Before training a model, we (1) discarded columns, followed by rows with more than 20% of values missing from the training set, then (2) filled in the remaining missing data in both training sets and test sets with each variable's median value calculated on the training set only.

Learning models. We run the experiments using four different off-the-shelf algorithms:

LM a linear regression model, implemented in *R* package *stats* [R Core Team, 2017];

MARS a multivariate adaptive regression splines model, implemented in *R* package *earth* [from mda:mars by Trevor Hastie and utilities with Thomas Lumley's leaps wrapper., 2018];

RPART a regression tree, implemented in *R* package *rpart* [Therneau et al., 2017];

RF a random forest, implemented in *R* package *ranger* [Wright and Ziegler, 2017]

3.2.3.3 Error Metrics

We measure model error by NMAE, defined in Equation 3.5, where z_i is the observed value, $\hat{z}_i$ is the prediction, and $\bar{z}$ is the mean of Z in the test set[3]. The normalised metric facilitates comparisons across data sets, so it was chosen over the more widely used MAE.

$$NMAE = \frac{\sum_{i=0}^{n} |\hat{z}_i - z_i|}{\sum_{i=0}^{n} |z_i - \bar{z}|} \tag{3.5}$$

$$PAE = Est - Gold \tag{3.6}$$

Estimation error or Predictive Accuracy Error (Equation 3.6) is the difference between (a) the NMAE estimated in the in-set by taking the mean error incurred by the same algorithm across all test sets, as required by a given evaluation method, and (b) the "gold standard" error incurred by a learning algorithm in the out-set.

3.3 Empirical Results

In this section, we will report on estimation errors for the methods described in Section 3.2.1, with some exceptions. We ignored contiguous and systematic spatial blocking in real-world experiments due to their irregular spatial distributions. We also excluded time buffering without time-blocking since it removed too many cases from the training sets in real-world data, making it often impossible to learn adequate models.

We also tested prequential evaluation methods that either had a sliding window for training, removed the spatial region in the test set from the training set, or both. We will not report their results in this chapter because they were consistently out-performed by their growing window counterparts (though this difference was not statistically significant).

Figures 3.9 and 3.10[4] show estimation errors for artificial and real-world data sets, respectively. A positive value indicates the algorithm over-estimated error – a pessimistic estimate; a negative value means the error was under-estimated – an overly optimistic estimate. Each point in the box plot shows the average error incurred by a method for one learning algorithm

[3]Note that this means that we are nomalising by the error incurred by a naive model that is not so naive, as it has access to the whole test set.

[4]Hidden under-estimation outliers in this figure belonged to data set 75 and were similarly distributed across all methods. Hidden over-estimation outliers belonged to data set 41, and were distributed across the two MC methods, Preq-Tb, CV-Tb and CV-Tb-Tbuf.

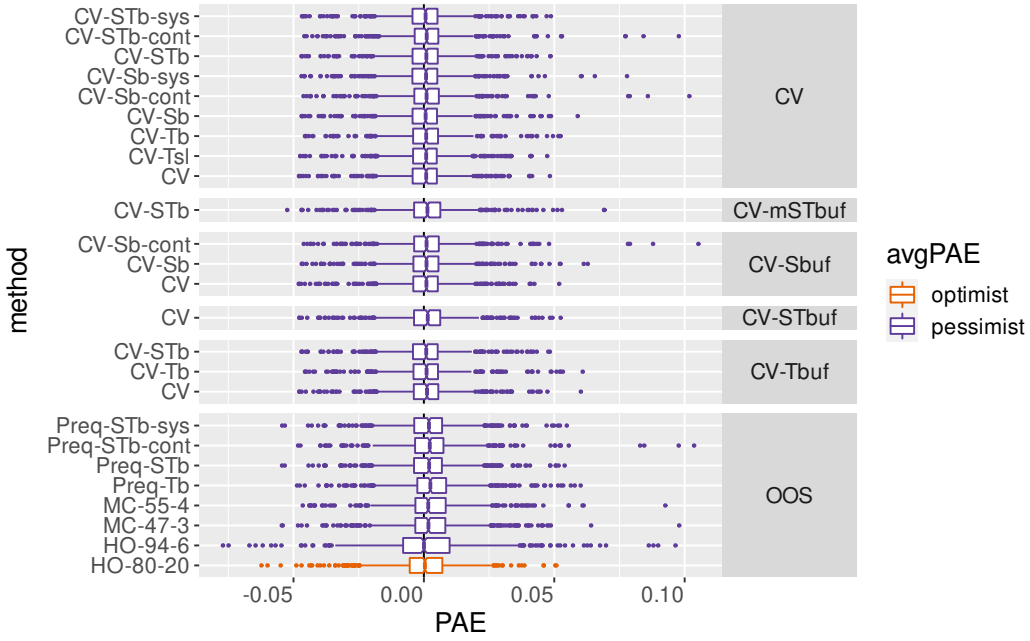


Figure 3.9: Box plots of estimation errors incurred by cross-validation and out-of-sample methods on 192 artificial data sets using four learning algorithms

and data set pair; the boxes colours' reflect whether prediction error is, on average, under-estimated (orange) or over-estimated (purple).

Figure 3.9, shows the results on the artificial data. All methods seem to have a similar spread, centred around zero, but 80-20 holdout, on average, under-estimates error. Holdout 94-6 stands out from the rest, showing the broadest interquartile range (the widest "box") and some of the most prominent outliers in both directions.

Figure 3.10 shows results from real data. They suggest more considerable differences between methods: not all centre around zero, and their spread is more varied and sometimes asymmetric[5]. On average, standard CV under-estimates error – a problem that can be mitigated by temporal blocking, using buffers, or both. OOS methods tend to over-estimate error, except for holdout 80-20, which under-estimates it.

3.3.1 Relative Errors

We turn our attention to the relative absolute predictive accuracy error defined in Equation 3.7 , and the relative predictive accuracy error defined in Equation 3.8. They are shown in Figures 3.12 and 3.11, binned into three categories – low, moderate, and high – of errors

[5]Results for spatial buffers are less reliable as they do not include results for some data sets, due to too many observations having been removed from the training sets when combining buffering with their irregular sensor networks.

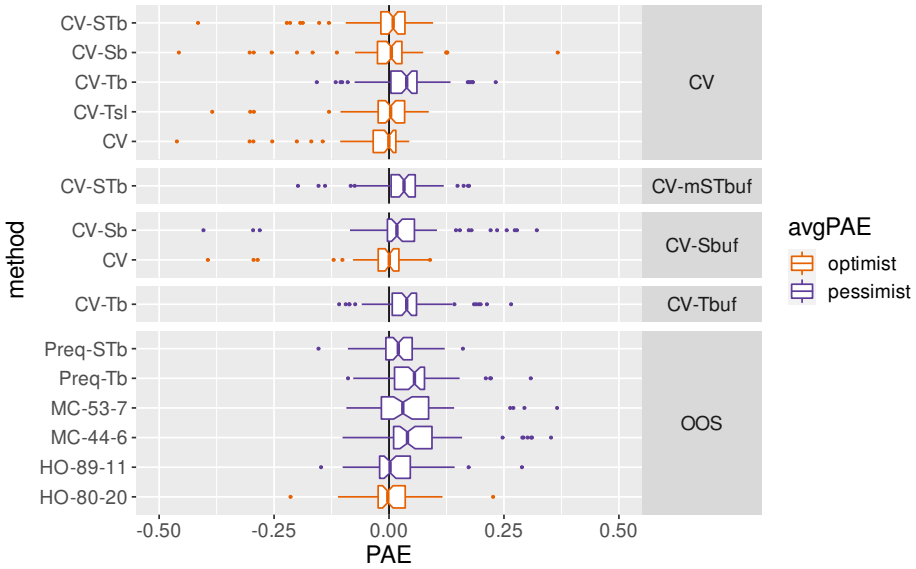


Figure 3.10: Box plots of estimation errors incurred by cross-validation and out-of-sample methods on 17 real-world data sets. A few extreme outliers were hidden to preserve the scale.

and absolute errors; bin intervals were chosen so that moderate absolute errors are roughly those within the interquartile range.

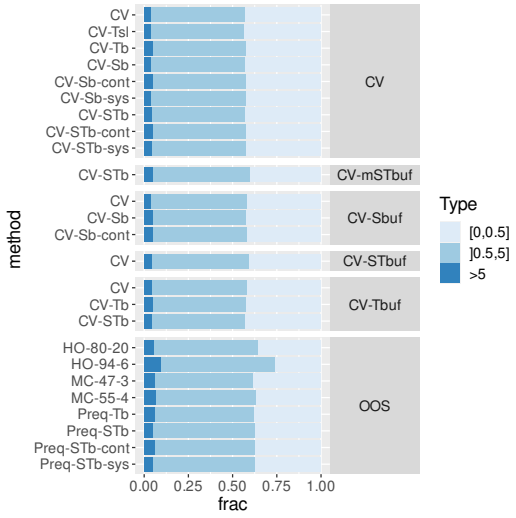
$$RAPAE = |Est - Gold|/Gold \tag{3.7}$$

$$RPAE = (Est - Gold)/Gold \tag{3.8}$$

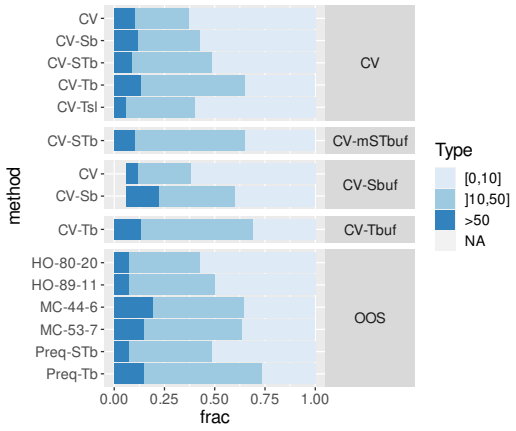
Error estimates for artificial, stationary data are quite accurate. In Figure 3.11a, high relative errors – those differing by more than 5% from the gold standard in absolute terms – represent less than 10% of the results regardless of method; but holdout 94-6 continues to stand out for producing a noticeably larger fraction of high relative errors than others. Figure 3.12a breaks relative errors down into positive (over-estimates, pessimistic) and negative (under-estimates, optimistic) categories. All methods produce a larger number of pessimistic estimates than optimistic ones, but holdout almost balances optimistic and pessimistic estimates.

Estimates for real-world data are much more sensitive to the chosen method; we can see larger differences between them in Figures 3.11b and 3.12b.

OOS methods are very rarely highly optimistic, but this seems to come at the cost of higher rates of (pessimistic) severe error over-estimation, especially so for repeated holdout (MC) and temporal-block prequential evaluation (Preq-Tb). While one-time holdout is comparable to other OOS methods in its rate of severe error over-estimates, it shows a noticeably higher



(a) Artificial



(b) Real-world

Figure 3.11: Bar plots of relative absolute estimation errors incurred by cross-validation and out-of-sample methods on 192 artificial and 17 real-world data sets using four learning algorithms. Note the different legends

percentage of combined low and moderate over-estimates than them.

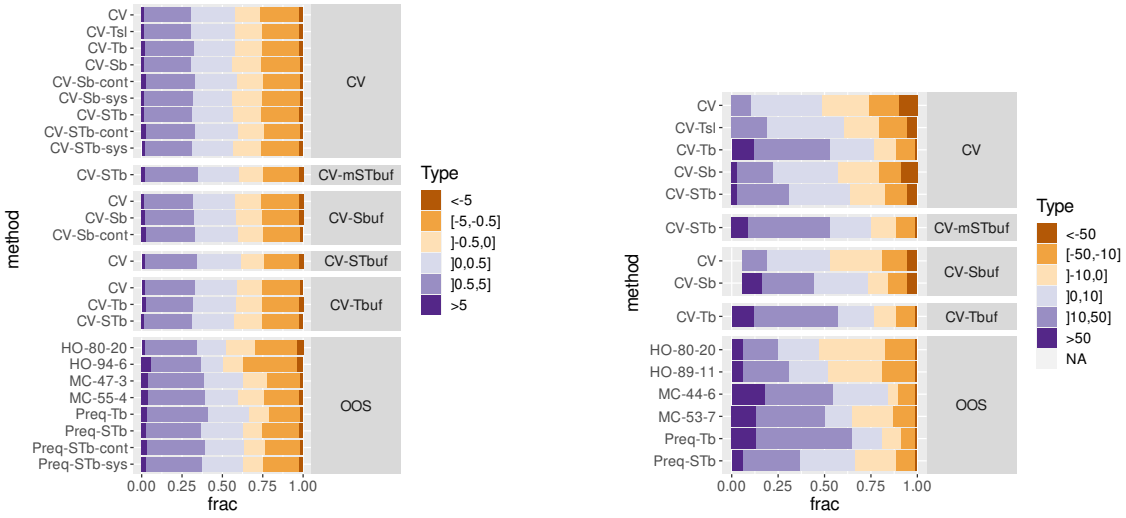
Standard CV completely avoids making highly pessimistic estimates, but there is a trade-off again: it shows similarly high rates of low to moderate optimistic error estimates as holdout, and sizeable rates of highly optimistic error estimates. Only three CV variants can boast rates of severe error under-estimation comparable to OOS methods – namely, temporal-block CV (CV-Tb), buffered temporal-block CV (CV-Tb-Tbuf), and leave-time-and-location-out CV (CV-STb-mSTbuf). Unfortunately, they also severely over-estimate error more frequently than standard CV.

3.3.2 Ranking Relative Absolute Errors

Ranking the methods in terms of how well they approximate the "gold standard" error can help us more directly compare all methods against each other. Using absolute errors allows us to do this, while non-absolute errors would pose the question of how to rank different levels of error under- and over-estimation.

For each data and learning algorithm pair, we ranked methods according to their relative absolute predictive accuracy error (Equation 3.7) on the same data sets and using the same learning algorithms. Figures 3.13 and 3.14 shows the average rank of each method; the lower the rank, the better.

The average rankings differ depending on the type of data, but they have two of their top 5 best estimators in common: time-slice CV (CV-Tsl), the best on artificial data; and spatial



(a) Artificial

(b) Real-world

Figure 3.12: Bar plots of relative estimation errors incurred by cross-validation and out-of-sample methods on 192 artificial and 17 real-world data sets using four learning algorithms. Note the different legends

block CV (CV-Sb), the fourth best on real-world data.

All CV variants, on average, out-rank OOS methods, having lower absolute error on their estimates for artificial data; on real-world data, some OOS methods rank among CV variants – holdout jumps from a low position on artificial data to the top 50% on real-world data.

3.3.2.1 Statistical Significance

To test whether differences across data sets are statistically significant, we apply the Friedman-Nemenyi test as suggested by Demšar [2006]. We used the *R* package *scmamp* [Calvo et al., 2016]). Each method is considered a "treatment" or "classifier", and we narrowed our universe of methods. We only include: (a) standard CV; (b) the best holdout and repeated-holdout variant; (c) the best CV and prequential variants that use, respectively, temporal, spatial, and spatio-temporal blocking; and (d) the best CV variants that use any type of buffer. The best here means the method that achieved the lowest average rank in terms of absolute error.

Figure 3.15 shows examples of critical difference diagrams for both types of data. When using real-world data, methods were not significantly different at 5% confidence level (they are all connected by the bold line). When using artificial data, CV methods performed significantly better than the OOS methods, though they were not significantly different between themselves.

The findings differed when we ran separate significance tests for each learning algorithm.

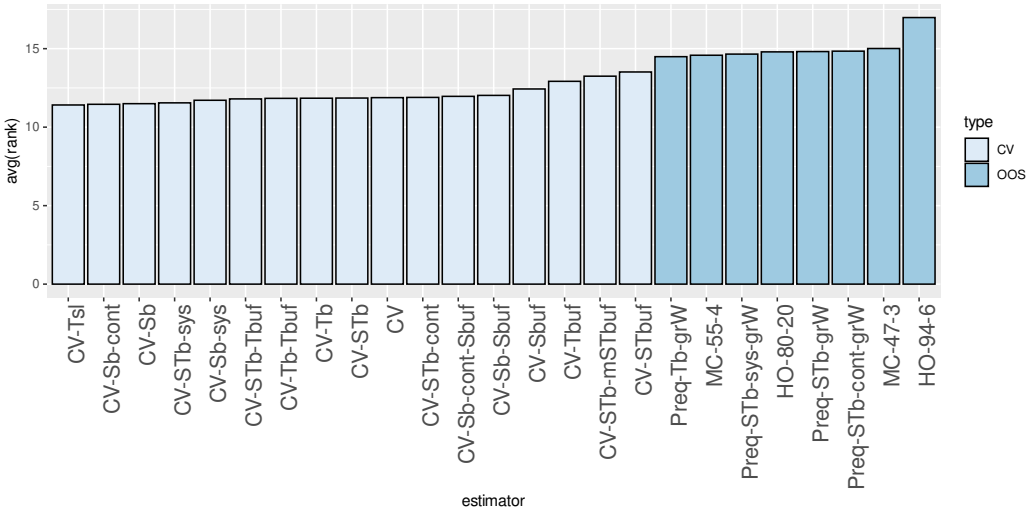


Figure 3.13: Bar plots of absolute error average rank for artificial data.

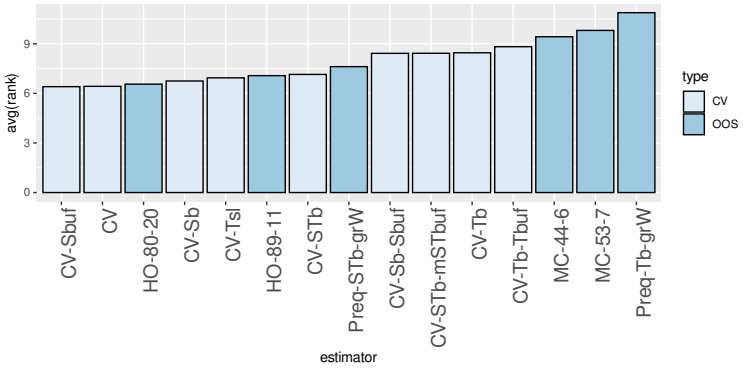


Figure 3.14: Bar plots of absolute error average rank for real-world data.

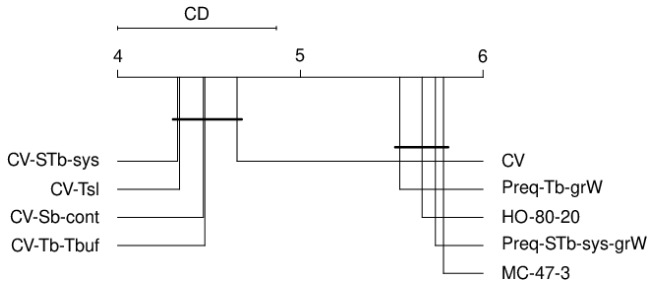
Differences between methods on real-world data were significant when using LM and MARS but not when using tree-based models. On artificial data, we found significant differences when using any of the models individually. To inspect these results, consult Appendix B.2.1.

3.3.3 Accuracy vs Optimism

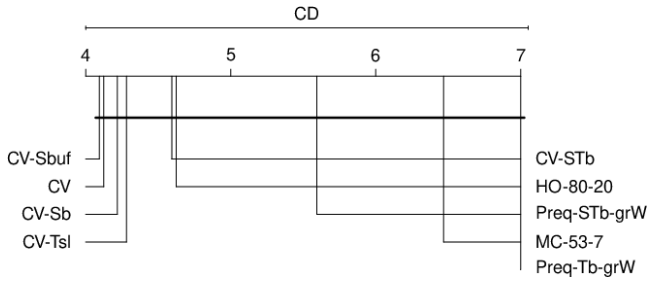
In this section, we investigate whether there was a trade-off between accuracy, as measured by a lower rank in estimates' absolute error, and optimism, that is, a tendency to under-estimate error.

Figure 3.16 shows the average predictive accuracy error incurred by each method against its average rank of absolute errors.

When using artificial data, all OOS methods, except HO-80-20, appear above the dashed line because they over-estimated error, on average; they also appear on the right side of the



(a) Artificial data



(b) Real-world data

Figure 3.15: Critical difference diagram according to Friedman-Nemenyi test (at 5% confidence level) for a subset of estimation methods using artificial and real-world data

plot, since they did not approximate error as well as most CV methods in absolute terms.

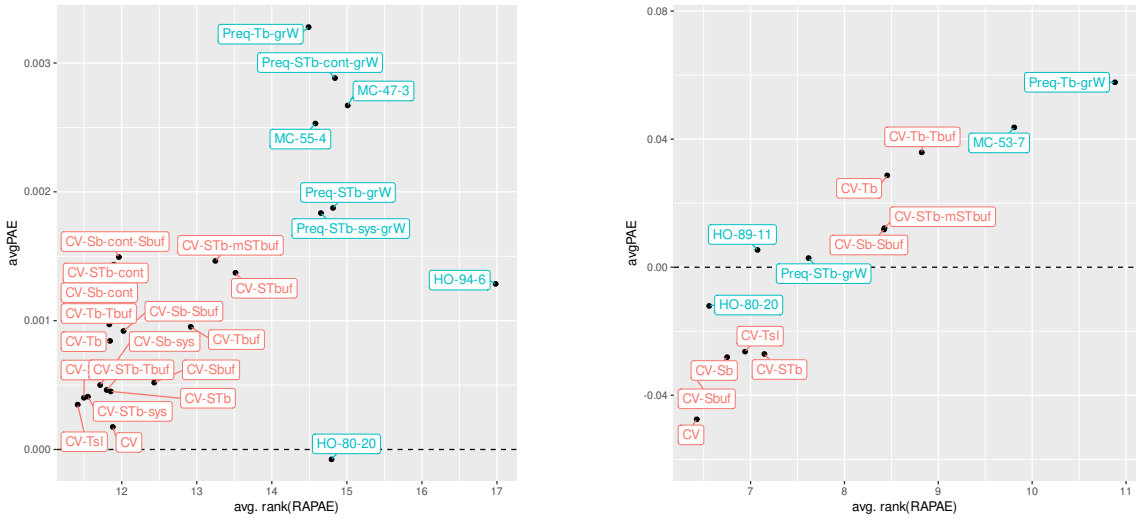
When using real-world data, a trade-off emerged: methods that ranked better in absolute error under-estimated it more severely, on average, on real-world data. The best-ranked methods, on the left side of the plots, appear below the dashed line. For example, standard CV, despite ranking better than almost every other method, under-estimated error, on average, more than most other pessimistic methods over-estimated it, as it suffered from more outliers of severe under-estimation (*cf.* Figure 3.10).

Some methods, such as spatio-temporal block prequential evaluation (Preq-STb-grW) and space-buffered spatial CV (CV-Sb-Sbuf), struck a better balance in real-world scenarios. On average, they approximated error accurately, being only slightly pessimistic (they appear above but close to the dashed line) and not compromising their average rank as much as others (they appear around the middle of the plot).

3.4 Discussion

Our experiments follow the design used to assess performance estimation methods' accuracy in time series by Bergmeir and Benítez [2012] and, later, by other authors [Cerqueira et al., 2017, Mozetič et al., 2018].

We found that all types of evaluation methods estimated error in spatio-temporal numerical



(a) Artificial (b) Real-world

Figure 3.16: Average error against the average rank of absolute errors for (a) artificial; and (b) real-world data sets. Methods below the dashed lined tend to be optimistic in their error estimates. Lower ranks indicate more accurate estimates in terms of absolute error.

prediction accurately when using artificially generated, stationary data. When using real-world data, they approximated error less accurately, and we found more noticeable (though not more statistically significant) differences between methods in their tendencies to under- or over-estimate error.

On average, standard CV under-estimated error, often providing severely over-optimistic projections of performance when using real-world data. These over-optimistic outliers could be doubly deleterious: leading analysts to select models that would perform poorly on unseen data, and to place undue confidence in its predictions. If the model is then used to guide critical decisions in impactful applications, such as public health policy, consequences could be dire. CV's tendency to under-estimate error was mitigated if we (a) divided data into blocks along the temporal dimension, or (b) created a buffer between training and test sets after blocking data along the temporal, spatial, or both dimensions. The more severe error under-estimation outliers only disappeared when data was blocked along the temporal dimension.

In the real-world scenarios, a trade-off between optimism and accuracy emerged. Methods like standard CV and space-buffered CV (CV-Sbuf) often severely under-estimated error, even though, on average, they still approximated error better than other methods applied to the same scenarios. Out-of-sample methods were not as often or as severely over-optimistic in their estimates, but their average accuracy diminished (80-20 holdout being the exception, since it somewhat under-estimated error, on average).

OOS methods mostly avoided the severe error under-estimation outliers that plagued standard CV and most of its variants. For this reason, OOS methods could be preferable to CV (and especially to standard CV), despite their lower absolute accuracy. Time-wise holdout estimates error quite well at a fraction of the computational cost, despite being more optimistic than other OOS methods.

OOS methods also respect temporal order, which can be an advantage in itself. Many analysts frown upon relying on future data to make projections about the past, and the ubiquitous data stream scenario – where data arrive in order as time progresses – naturally privileges prequential evaluation.

Related work on time series. Several reasons could explain the drop in accuracy when using real-world data: (a) some of the real-world data sets contain missing values, which we did not simulate in the artificial data; (b) locations were irregularly distributed in almost all the real-world data, in contrast with the regular grids of artificial data; and (c) real-world data may contain heterogeneities or suffer from concept drift, while artificial data was stationary. In previous work by Bergmeir and Benítez [2012], under-estimation outliers had also appeared more frequently in real-world time series.

Whether we should use blocked cross-validation or an out-of-sample method to evaluate time series models is a contested issue. Bergmeir et al. suggested that non-buffered blocked cross-validation should be used based on experiments using both artificial [Bergmeir and Benítez, 2011] and real-world data [Bergmeir and Benítez, 2012, Bergmeir et al., 2014], especially arguing its benefits for small time series [Bergmeir et al., 2018]. Though Cerqueira et al. [2017] agree that blocked CV may work well for synthetic, stationary time series, they argue that out-of-sample methods, in general (and repeated holdout, in particular) work best for real-world time series. Cerqueira et al. [2020] also claim that OOS methods work best for real-world time series even if they are stationary, though CV also proved competitive in that scenario in their study. These authors found no compelling evidence that CV worked better for small time series. Mozetič et al. [2018] corroborated prequential methods' usefulness, though they excluded repeated holdout from their experiments.

Our results are not directly comparable to work on time series, both because our data include the spatial dimension and because the error metrics used were not consistent across studies. Our experiments did not find temporal block CV or repeated time-wise holdout to have the same advantages as previous studies: while they mitigated issues with severe under-estimation, they had lower absolute predictive accuracy. We also did not find holdout to out-perform standard CV as shown by Bergmeir et al. [2018] for stationary, non-linear auto-regressive time series.

Related work on spatial data. Roberts et al. [2017] generated synthetic spatial data

from complex models of artificially generated variables. We withheld the most recent data to determine the gold-standard error in our artificial experiments, but this is not an option in the absence of the temporal dimension. Roberts et al. resolve this by defining the "gold standard" as the average error incurred by predicting onto independent runs of their simulation. They empirically showed that standard CV under-estimated error on synthetic spatial data, especially when relevant predictors in the test set were within their range in the training set. They recommended using contiguous block CV and buffered leave-location-out CV, which fared better in their experiments.

We did not assess the performance of leave-one-out methods, and only tested the analogous contiguous spatial block CV (CV-Sb-cont) on synthetic data. We found CV-Sb-cont to be on the top three most accurate methods for artificial data.

Buffered CV may have underperformed in our experiments due to the spatio-temporal nature of the data or due to block and buffer size choices. Roberts et al. [2017] warned that buffer sizes needed only to be as large as the distance at which model residuals reached zero, but block sizes needed to be substantially larger than that. We did not vary block and buffer sizes in our experiments. We set the spatial buffer to the distance to first-order spatial neighbours (only first-order neighbours directly influenced the values in the following time-stamp in our data). To reach $K = 16$ folds, we set block sizes to 2×2 and 5×5 for grids of size 8×8 and 20×20, respectively, which may have been too small.

Related work on spatio-temporal data. Meyer et al. [2018] experimented with three variations of cross-validation on real-world spatio-temporal data: leave-location-out CV, which we call spatial block CV (CV-Sb); leave-time-out CV, which we call temporal block CV (CV-Tb); and leave-location-and-time-out CV (CV-STb-mSTbuf). The experimental design differs from ours, as the authors did not compare the results against previously withheld or independent data. They instead explained the noticeably lower errors projected by standard CV as resulting from model over-fitting. Our results support their findings: standard CV under-estimated error more than these three variations on our real-world data.

Strengths and limitations. Experimenting with such a wide range of evaluation methods poses its challenges. It is not trivial to ascertain what constitutes a fair comparison between methods that use available data in such varied ways.

We opted to keep the number of partitions (or repetitions in the case of repeated holdout) constant and, when possible, we kept comparable test/train size ratios. We kept similar test/train ratios for one-time holdout, repeated holdout, and most variants of cross-validation – buffered versions, of course, had a larger ratio due to their reduced training size. Ratios necessarily had to be different for prequential evaluation, whether using sliding or growing training windows. (The 1:15 and 1:8 test:train ratios tested differed from the 2:8 out-set/in-

set ratio, which we only tested in one variation of one-time holdout.)

Keeping the total number of partitions constant meant that the number of partitions along the temporal and spatial dimensions differed between solely temporal or spatial CV and spatio-temporal CV. A spatio-temporal block will span a shorter period and cover a smaller spatial region than its one-dimensional counterparts. It is hard to determine if the comparison would be fairer if we increased the total number of partitions in spatio-temporal block CV to keep the same divisions along each dimension.

Out-of-sample methods (other than one-time holdout), by definition, cannot use all available data (either as part of training or testing) to estimate error at each iteration, which could put them at a disadvantage. Only the last estimate(s) in a prequential block evaluation procedure will have access to the same amount of training data that CV has for all its estimates, and that only happens if using a growing window. In the case of a growing window, the issue may be mitigated by aggregating the estimates differently. We calculated a simple mean of the error incurred in each test set to produce a final estimate, but future research may look into the effects of using a weighted mean that penalises the estimates produced by the older, smaller training windows.

In trying to be fair by using the same number of partitions in prequential evaluation as in CV, we also forced prequential methods to project error based on a lower number of estimates, as they exclude the oldest partitions as test sets.

Repeated holdout performed particularly poorly compared to previous results on time series [Cerqueira et al., 2017, 2020, Cerqueira, 2019]. The chosen train and test sizes could partly explain this underperformance, though other data characteristics may have contributed more to the less than stellar results.

We ignored computational costs and execution times, but they may influence decisions in real scenarios. All methods involve three steps with differing costs: (1) partitioning the data; (2) building models on training sets; and (3) measuring error metrics on test sets. Assuming that we use k data partitions, each method requires training a certain number of models on different fractions of data: holdout trains a single model on the first $k-1$ partitions; CV trains k models on $k-1$ partitions; temporal-block prequential evaluation[6] trains $k-1$ models on at least one partition (if using a sliding window) or an average of $\frac{k}{2}$ partitions (if using a growing window); repeated holdout trains k models on a smaller data fraction, independent from other methods' partitions. Usually, most computational resources will be spent on the learning step (step 2), though data partitioning (step 1) may require computing spatial and temporal distances, which could be quadratic on the number of observations. Suppose that model training costs grow with training set size (which is not necessarily true). In that case, we expect cross-validation to be the slowest and most expensive method, followed

[6]We ignore spatio-temporal block variants of prequential evaluation in these calculations for simplicity.

by prequential evaluation with a growing window. In practice, we often run the learning step in parallel, decreasing execution time by spending more processing power and memory resources.

Another limitation that we share with previous studies on time series is that the experimental design may bias our results since it relies on an "external" 80-20 time-wise holdout. It would be reasonable to assume that its train/test partitions matching the out-set/in-set split could contribute to its higher absolute accuracy (or over-optimism). Indeed, we found 80-20 holdout to be the only out-of-sample method that, on average, under-estimated error, though Cerqueira et al. [2020] did not find the same effect in time series. In future, we suggest testing other experimental designs, such as defining the out-set to contain the most recent observations at until then unobserved locations.

Open issues. We share the concerns of Roberts et al. [2017] over the effects of ignoring irregularities and imbalances when using CV with autocorrelated data.

When data are collected over irregular grids or at irregular intervals, some periods or regions may be poorly covered, complicating proper CV blocking. We avoided this issue in our real-world experiments by (a) ignoring contiguous and systematic spatial and spatio-temporal block CV, and (b) calculating statistics at a constant frequency when needed. Roberts et al. [2017] mention a few potential solutions that have been applied to spatial data by others – namely, distributing consistently shaped blocks irregularly, or using variously shaped blocks. Future work could adapt these solutions to spatio-temporal problems, but accounting for irregularities in both dimensions will be challenging, and may be computationally more exacting.

Even regularly sampled data may display imbalances in target value distribution, causing other complications. Roberts et al. [2017] list some possible solutions to the imbalance issue, but warn that, in trying to address it, methods may become unable to consistently break the dependence between training and test, the root cause of over-optimism in error estimates. Other proposed solutions, like buffered leave-one-out methods, can be prohibitively computationally expensive. Once again, the issue is further complicated if both temporal and spatial autocorrelation occur.

Theoretical analysis, even if under some assumptions that may not hold for real-world data might help guide future empirical work. Additional synthetic experimentation could also provide insights about evaluating data with missing values, imbalanced distributions, irregular sampling, or non-stationarities. It may also be useful to carry out experiments more in line with the work of Roberts et al. on spatial data.

Further research is needed to understand the effects of multiple other factors, including block and buffer sizes, number of partitions or repetitions, alternative error metrics, degrees

of missing data, and other data characteristics. Computational costs also need further study, especially when analysing more complex methods.

Our results were not completely consistent across different learning models and types of data, but they underline that evaluation methods should not ignore the spatial and, especially, the temporal nature of the data. Future work should try to answer the many open issues in spatio-temporal forecasting evaluation.

3.5 Summary

Appropriately evaluating spatio-temporal forecasting approaches is essential, but the temporal, spatial and spatio-temporal autocorrelation in this type of data can cause standard methods like CV to under-estimate error. This chapter describes an empirical study of several evaluation alternatives to standard methods, testing them on both synthetic and real-world spatio-temporal data, and using four different learning algorithms.

Our results show that even though standard CV, on average, produced reasonably accurate estimates, it was over-optimistic, often leading to severe error under-estimation on real-world data. CV variants that blocked observations along the temporal dimension mitigated this issue. Out-of-sample methods estimated error adequately but less accurately than CV, being more pessimistic in their estimates.

We are yet to reach a definitive recommendation for which specific method should be the gold standard for spatio-temporal forecasting, but we can make some general recommendations. When dealing with real-world data, we recommend avoiding standard CV. Variants that block data along time are more suitable because they avoid severe error under-estimation. Against these CV variants, OOS methods are a competitive alternative: they inherently respect temporal order, and provide similarly accurate estimates.

Chapter 4

Extracting Spatio-temporal Indicators

4.1 Introduction

One way to tackle spatio-temporal forecasting problems is by extracting features that capture spatio-temporal dynamics in the data, thus enabling standard regression algorithms to learn from this type of data. Several approaches to building these features, sometimes called *spatio-temporal indicators*, exist in the literature [Ohashi and Torgo, 2012, Appice et al., 2013b, Pravilovic et al., 2017].

Our goal is to generate summary statistics of past values within boundaries of spatio-temporal distance – defined as a linear combination of temporal and spatial distances – to the observation we want to describe. Based on the intuition that the "spatial radius of influence" decreases for older observations, Ohashi and Torgo [2012] propose to only include the most recent observations from more distant neighbours. However, some phenomena may take time to travel from one location to another. For example, consider a blizzard moving from the Arctic to central Europe. In this scenario, in order to predict Berlin's weather it may sometimes be more useful to take measurements from distant locations, such as Barensk in the polar circle, than using Frankfurt measurements. Therefore, we hypothesise that for some applications, including older observations at more distant locations, while excluding their most recent values, may improve prediction. This chapter evaluates an algorithm for feature extraction that follows that principle.

4.2 Materials and Methods

In this section, we present spatio-temporal indicators and introduce a new algorithm. We test and evaluate the two approaches with real-world data sets. We describe our experimental design and meta-analysis methods.

4.2.1 Spatio-temporal Indicators

Standard regression algorithms work on data in table format. When working on univariate forecasting problems, features describing each observation must be extracted so that the model can learn from predictive features.

In 2012, Ohashi and Torgo proposed a set of indicators that can be extracted to describe an observation based on its spatio-temporal context.

4.2.1.1 Ohashi and Torgo's Proposal: The Neighbourhood "Cone"

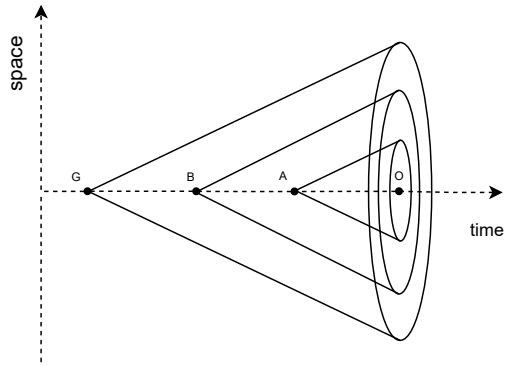


Figure 4.1: Spatio-temporal neighbourhoods with maximum spatio-temporal distance β_1, β_2, and β_3 ($\beta_3 > \beta_2 > \beta_1$)

The spatio-temporal indicators summarise values within spatio-temporal neighbourhoods of an observation, O, whose value we want to predict. Consider that O was measured at time t_O and location l_O. Its spatio-temporal neighbourhood contains all past observations within a boundary, β, of spatio-temporal distance as defined by Equation 4.1.

$$D_A = \alpha \cdot D_A^S + (1 - \alpha) \cdot D_A^T \leq \beta \tag{4.1}$$

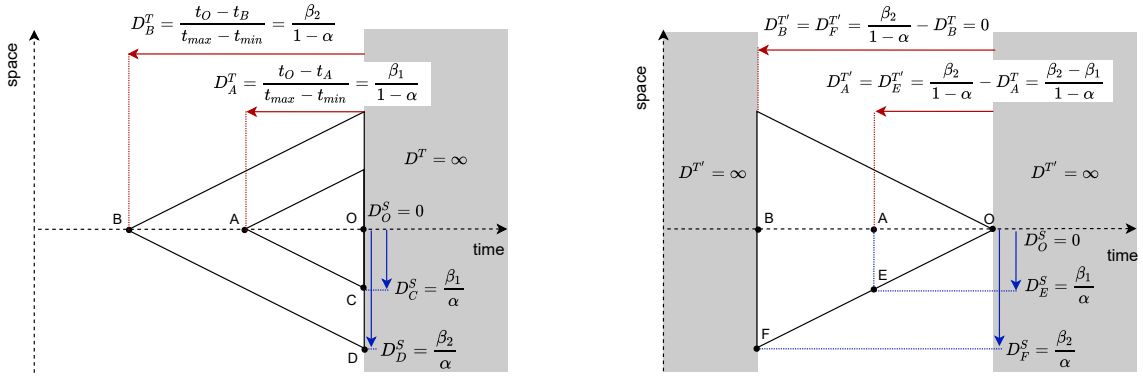
In this equation, D_A^S is the spatial distance between the locations where we observed O and A, D_A^T is the temporal distance between observations O and A, and α is a weighting factor.

The indicators should only be calculated on past observations, D_A^T is defined by Equation 4.2.

$$D_A^T = \begin{cases} \frac{t_O - t_A}{t_{max} - t_{min}}, & \text{if } t_A < t_O \\ \infty, & \text{otherwise} \end{cases} \tag{4.2}$$

Spatial and temporal distances differ within the data set and vary greatly between data sets, we normalised D_A^S and D_A^T to be in the $[0, 1]$ interval before we set *relative* spatio-temporal boundaries. In a deployment scenario, normalising temporal distance may be complicated by an unknown prediction horizon, but this is not a problem in our task.

We must choose β within the interval $]0,\ \min(\alpha,\ \alpha - 1)[$ to ensure that a neighbourhood can be defined within the extension of the data.



(a) Original cone (b) Reversed cone

Figure 4.2: Spatio-temporal neighbourhood cross-sections using (a) the original cone and (b) the reversed cone. Subfigure (a) shows two spatio-temporal neighbourhoods with maximmum distance β_1 and β_2, respectively. Subfigure (b) shows only one spatio-temporal neighbourhood with maximum spatio-temporal distance β_2. Regions where $D^T = \infty$ shown in grey

Figure 4.1 depicts three spatio-temporal neighbourhoods for observation O, which look similar to a cone. If α was set to 1, then instead of a cone, the neighbourhood would be shaped like a cylinder.

Figure 4.2a shows a cross-section of two of the neighbourhoods in Figure 4.1. From Equations 4.1 and 4.2, we can see that for a neighbourhood of maximum distance β_2: (a) the temporal distance D_B^T from observation O to an observation B measured at the same location, at the vertex of the cone, is $\frac{\beta_2}{1-\alpha}$; and (b) the spatial distance to an observation D, measured at the same time as O, at the border of the neighbourhood, is $\frac{\beta_2}{\alpha}$. Observation D would not be used to calculate summary statistics, since its temporal distance D_A^T was defined to be infinite, even though the difference between time-stamps was zero.

4.2.1.2 Our Proposal: The "Reversed Cone"

As we saw in Figure 4.1, the spatio-temporal neighbourhood originally defined resembled a cone; its base centred around the target observation, O, and its radius decreasing until the vertex reached another past observation at the target location.

In this section, we define a variant that reverses the direction of the neighbourhood "cone" in order to propagate older observations from more distant locations.

The reasoning behind the variant we test in this chapter is quite intuitive. In some applications, phenomena may take some time to "travel" from one place to the other, e.g., winds can carry harmful particulate matter generated by wildfires for thousands of kilometers, affecting air quality far away from their source days after ignition [Paulina Firozi, 2021]. Thus, in some cases, reversing the direction of the neighbourhood "cone" may better describe the conditions leading to an observation and, thus, improve prediction.

To achieve this "reversed cone" variant, we redefine temporal distance in Equation 4.3, resulting in the spatio-temporal distance in Equation 4.4.

$$D_A^{T'} = \begin{cases} \frac{\beta}{1-\alpha} - D_A^T, & \text{if } \frac{\beta}{1-\alpha} > D_A^T,\ t_A < t_O \\ \infty, & \text{otherwise} \end{cases} \tag{4.3}$$

$$\begin{aligned} D'_A &= \alpha \cdot D_A^S + (1-\alpha) \cdot D_A^{T'} \leq \beta \\ &= \alpha \cdot D_A^S + \beta - (1-\alpha) \cdot D_A^T \leq \beta \\ &= \alpha \cdot D_A^S - (1-\alpha) \cdot D_A^T \leq 0 \end{aligned} \tag{4.4}$$

Figure 4.2b shows the cross-section for a neighbourhood with the "reversed cone", defined by a maximum distance β_2. In that figure, spatial distance is defined the same as in Figure 4.2a, but the temporal distance $D^{T'}$ was defined to be zero at point B, so that B became the centre of the cone's base instead of its vertex. Any point measured at a time-stamp $t_G < t_B$ was defined to have infinite distance to t_O, so that older points were excluded from the neighbourhood, and no observation was at negative temporal distance from observation O.

4.2.1.3 Calculating Features

Regardless of the direction of the cone, we calculated the same features, including:

- A temporal embedding of values measured at location l and times $t-1,\ t-2, \cdots,\ t-k$;
- Summary statistics (mean, weighted mean, and standard deviation) of values within three data-specific boundaries of spatio-temporal distance, β_1, β_2, and β_3, such that

$\beta_1 < \beta_2 < \beta_3$[1];

- Ratios between summary statistics (mean and weighted mean) within spatio-temporal neighbourhoods of increasing radius, i.e., ratio between the summary statistics for neighbourhoods of boundaries β_3 and β_2, and β_2 and β_1.

4.2.2 Data Sets

We tested our proposal on the real-world data sets presented in Chapter 3, Section 3.2.2.2. Since the aim is to test feature extraction for univariate spatio-temporal forecasting problems, we once again considered the variables separately as if each was an independent, univariate data set.

4.2.3 Experimental Design

In each experiment, we extracted features from the entire univariate data set, before we divided it into training sets and testing sets, according to our chosen performance estimation method. We built models based on the training sets, and then tested them on the respective testing sets.

The experimental design includes the error metric we used to evaluate the experiments, the performance estimation method used to estimate it, the hyper-parameter combinations that defined how we extracted the features, and the learning workflow followed for each training set.

4.2.3.1 Parametrization and Neighbourhood Size

In each experiment, we extract features from three different neighbourhoods (*cf.* Figure 4.1), before learning and testing a model of the data. We used a single weighting factor α to calculate distances for all three neighbourhoods in each experiment, so their different boundaries were specified by a set of three different β values.

For each data set, we tested features extracted with every combination of the parameters in Table 4.1. We tested ten different neighbourhood size combinations, each defined by a set of three β values (out of two different sets of three) in combination with a single α value (out of five different α). Figure 4.3 shows the number of neighbours included in the spatial and temporal neighbourhoods at their maximum extent, for the parameter combinations where $\alpha = 0.25$.

[1]To obtain the weighted mean, the weight of a neighbouring observation was set to be inversely proportional to its spatio-temporal distance to the observation for which statistics were being calculated.

Table 4.1: Parameters used for feature extraction

Parameter	Search space
Neighbourhood type	cone, reversed
Embed size (K)	4, 8
Dimension weight α	0.1, 0.25, 0.5, 0.75, 0.9
Neighbourhood radii β	$\{0.01, 0.02, 0.03\}$, $\{0.02, 0.03, 0.04\}$

We excluded outliers from the number of time-stamps in Figure 4.3b because many of them resulted from initial observations having to draw from shorter spans of past data. We were left with only one maximum number of time-stamps per neighbourhood size for data sets 1 to 6, so we opted to show them as bar plots; only data set 7 showed a range of different, non-outlying number of time-stamps.

The number of neighbours remain roughly within the same order of magnitude for both dimensions across data sets with a few exceptions: data set 5 has a much higher total number of locations distributed on a regular grid and, thus, more spatial neighbours; data sets 6 and 7 have a much higher number of time-stamps and, thus, more temporal neighbours (*cf.* Table 3.3, page 48).

4.2.3.2 Training Workflow

Having extracted features from the whole data set, the training workflow only needs to handle missing data and apply a standard regression algorithm.

Missing data. We followed the steps described in Section 3.2.3.2 to solve the issue posed for model training by the missing data in data sets 4, 6, and 7. Before learning a model on each training set, we discarded columns and rows with 20% or more values missing, and filled in the remaining missing values in both training and test sets with each column's median value in the training set.

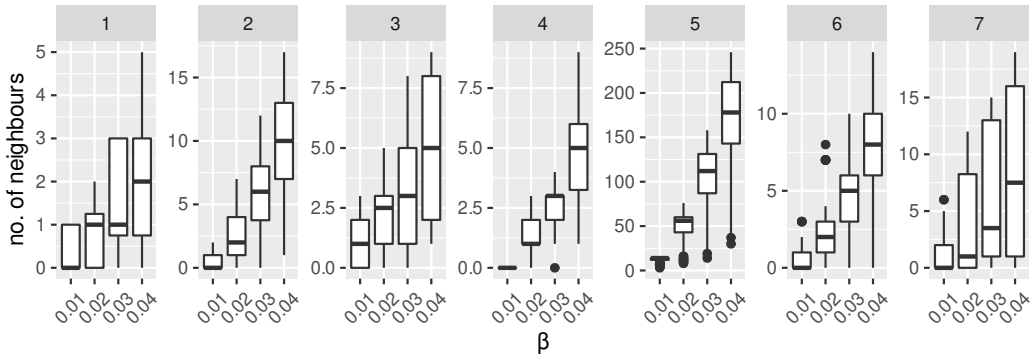
Regression algorithms. We run the experiments using four standard regression algorithms with default parameters, as implemented in free and open-source **R** packages:

LM a linear regression model, implemented in *R* package *stats* [R Core Team, 2017];

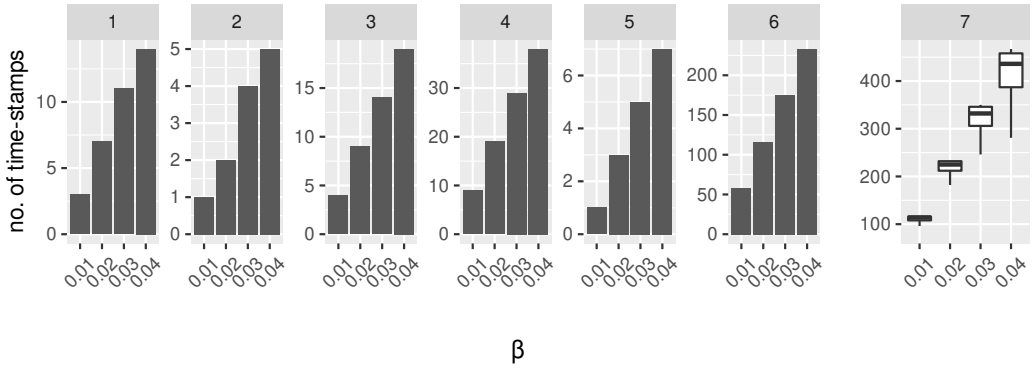
MARS a multivariate adaptive regression splines model, implemented in *R* package *earth* [from mda:mars by Trevor Hastie and utilities with Thomas Lumley's leaps wrapper., 2018];

RPART a regression tree, implemented in *R* package *rpart* [Therneau et al., 2017];

RF a random forest, implemented in *R* package *ranger* [Wright and Ziegler, 2017]



(a) Number of neighbouring locations



(b) Number of neighbouring time-stamps

Figure 4.3: Number of neighbouring locations at maximum spatial radius, and number of time-stamps within maximum temporal distance, calculated with $\alpha = 0.25$. We removed outlying number of time-stamps from subfigure (b); since only one value remained per neighbourhood size for data sets 1 to 6, we show them as bar plots instead of boxplots. Different scales used for each data set

Only the number of trees in the RFs was set to 250, where the default was 500. Reducing the number of trees in RF mitigates its higher computational cost. This number of trees should not severely compromise its performance since Probst and Boulesteix [2018] empirically showed that, for many data sets, "*the biggest performance gain in the out-of-bag curves can be seen while growing the first 250 trees*".

4.2.3.3 Evaluation Framework

To properly evaluate our methods, we need to define an appropriate error metric, and an adequate method to estimate it.

Evaluation Method We established in Chapter 3 that standard estimation methods such as CV are inappropriate in spatio-temporal settings. We opted to use what we call

prequential temporal block evaluation with a growing window (Preq-Tb-grW) with $K = 10$. This evaluation method avoids severely under-estimating error, though it is not as accurate as some other methods. It also respects temporal order, which can be valuable on its own, and it is common in the literature.

Prequential temporal block evaluation divides the data into ten blocks, according to temporal order, and trains models on a growing window: the first block is used to train a model that gets tested on the second block; the first two blocks are used to train a model that gets tested on the third. The process continues until the method tested models on all blocks except the first, which contains the oldest data. The evaluation metric is averaged across the nine testing blocks.

Even though we evaluated predictions in batches, the prediction horizon for our experiments at time t was $t + 1$. That is, when we wanted to predict an observation at a time $t + 1$, we assumed we had access to all observations measured before then. Since we extracted features from the whole data set under this assumption, before dividing it into training and testing sets, choosing this prequential method ensured that information did not spillover between training and testing.

Evaluation Metrics As in the previous chapter, we measure error by NMAE (Equation 3.5), as it facilitates comparisons across data sets.

4.2.4 Meta-Analysis Methodology

After running the experiments, we assess data set characteristics' impact on whether reversing the neighbourhood cone improved performance. This meta-analysis follows four steps: (1) extract features that characterise not a single observation, but the whole data set; (2) label each experiment's results depending on whether results were best with the original cone or its reverse; (3) build a meta-data set that joins the data characteristics with the label and experimental conditions, including the learning model, and spatio-temporal indicator parameters; and (4) train a classification model on this meta data set.

Extracting spatio-temporal data characteristics. We extracted various meta-features that characterise: (a) data size, (b) the prevalence of missing data, (c) the target distribution (e.g., skewness, which measures asymmetry in the distribution; kurtosis, which measures its "tailedness"; percentage of extremes), (d) the spatial distribution of locations (e.g., average normalised distance to the nearest neighbour), (e) spatial auto-correlation (e.g., the mean Moran I index), (f) the time series (e.g., time series acceleration measured by the ratio between simple and exponential moving averages), and (g) temporal autocorrelation (e.g., the Hurst exponent and serial correlation).

The Moran I index is a measure of global spatial autocorrelation. A non-zero Moran I index indicates spatial autocorrelation: it can go from -1 if distinct values cluster together to +1 if similar values do so. An index of zero suggests randomness in the spatial distribution. We computed the Moran I index at each time snapshot using inverse distance as the weight.

The Hurst Exponent measures long-term memory in a time series, while serial correlation measures its short-term memory. We calculated these statistics at each location.

A complete list of the 33 features we used can be found in Table C.8 (page 135).

Meta-labels. As we mentioned, we labelled each experiment depending on which neighbourhood type earned a lower average NMAE, on the same data set and when using the same learning algorithm. We compared experiments using the same remaining parameters (k, α, β_i), on the same data set and using the same learning algorithm.

Meta-learning. We trained a Conditional Inference Tree (CIT), first proposed by Hothorn et al. [2006]. The tree creates an interpretable, visual representation of the most determinant factors affecting our proposals' efficiency, using an unbiased split-rule.

4.3 Empirical Results

We aim at comparing how prediction behaved with varying spatio-temporal indicators, and, especially with different neighbourhood types – the original cone versus the reversed cone. In this section, we present the average performance of different parametrizations, and analyse how results differ across domains.

4.3.1 Average Rank

We start by analysing how the 40 parameter combinations rank against each other. For each data set and learning model pair, we ranked the parameter combinations' NMAE.

Figure 4.4 shows the average percent rank across data sets for each learning model. Similar patterns appear across models, though RPART was much less sensitive to different parameter combinations. It is also apparent that poorer results concentrate on the reverse cone combined with lower values of α.

Aggregating results. Aggregating the results can help us have a clearer picture of what settings led to the best performance, on average. Figure 4.5 shows the results of aggregating average percent rank over all experiments that used each parameter setting in their parameter combination.

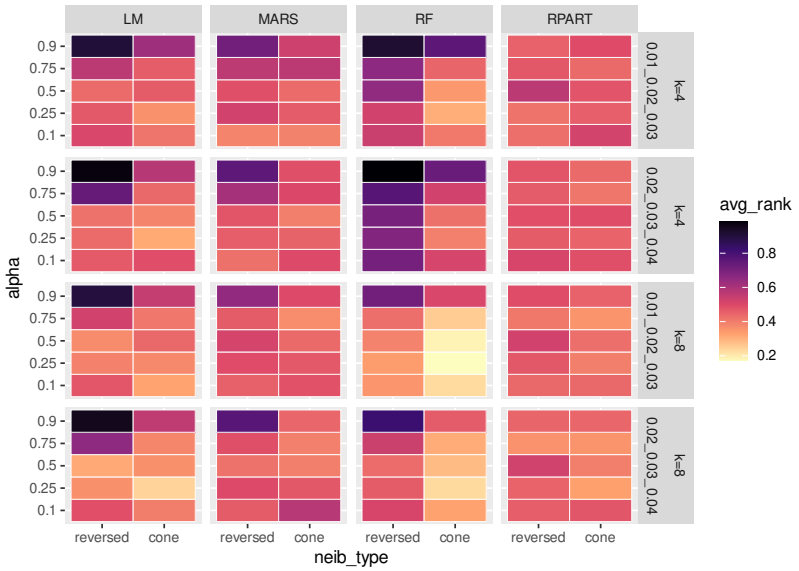


Figure 4.4: Percent rank obtained by each of the 40 parameter combinations, averaged across data sets for each learning model. Lower ranks indicate better performance

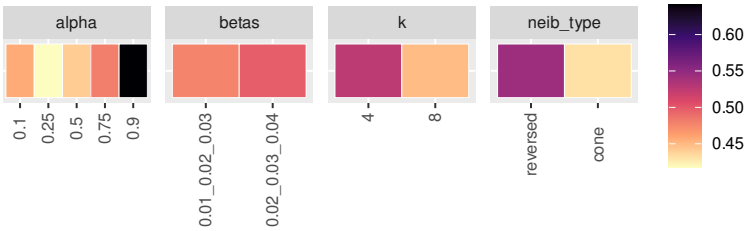


Figure 4.5: Percent rank obtained by each parameter setting, averaged over all experiments that used them in combination with each learning model and across data sets. Lower ranks indicate better performance

We can see that, on average, the original cone proposed by Ohashi and Torgo worked better than the reverse cone that we wanted to test. The remaining parameters led to better average results when, respectively, α was set to 0.25, k was set to 8, and $\beta \in \{0.01,\ 0.02,\ 0.03\}$. Starker differences appeared when we varied the type of neighbourhood (cone or reversed cone) or α, the parameter that balances the temporal and spatial dimension; changing the neighbourhood radii βs produced more similar average ranks.

Results per data set. We understand how each parameter combination works, on average, across data sets. We must also investigate how the results change depending on the domain. Figure 4.6 shows the average rank obtained by each parameter, this time averaged across learning models for each data set.

The reversed cone only performs better than the original cone, on average, for data set 21.

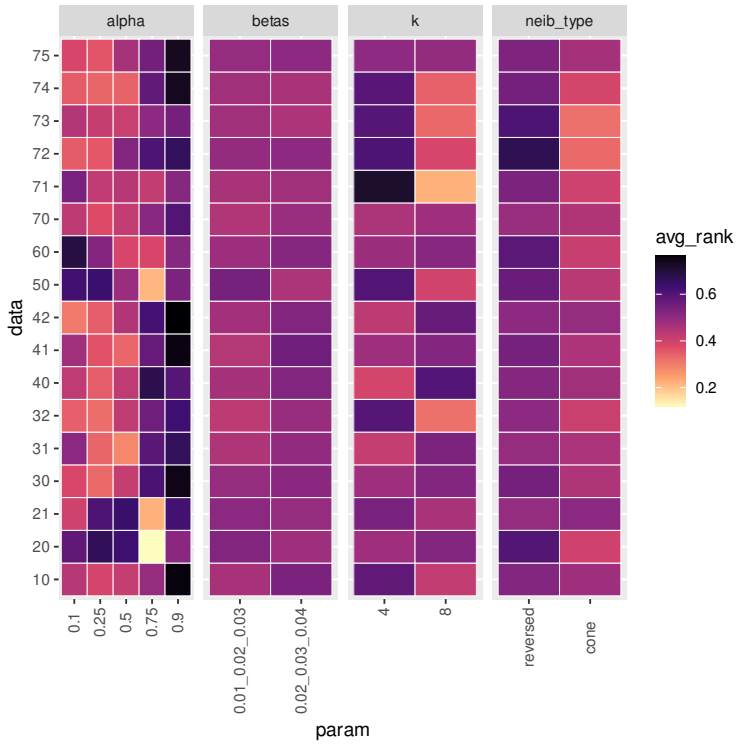


Figure 4.6: Percent rank obtained by each parameter setting, averaged over all experiments that used them across learning models, for each data set. Lower ranks indicate better performance

The two types of neighbourhood result most often in similar ranks; differences are most noticeable for data sets 72 and 73.

The results for each data set seem to be most sensitive to α, though which values work better varies. Most data sets obtain their worst results with higher values of α, which weigh the temporal dimension more. Data sets 20, 21, 50, and 60 contrast with the general trend, performing noticeably worse with lower α values.

The remaining parameters, β and k, also get their best results at different values depending on the data set. However, changing these settings results in less marked differences in performance in most data.

4.3.2 Best Performers

On average, the reversed cone performed rather poorly, but inspecting the top performers for each neighbourhood type could provide a different perspective on the results. Selecting the top performing, "optimal" parameter combinations *a posteriori* means we cannot guarantee that one could achieve these results in real scenarios. Still, this analysis may help uncover the potential benefits of reversing the spatio-temporal neighbourhood.

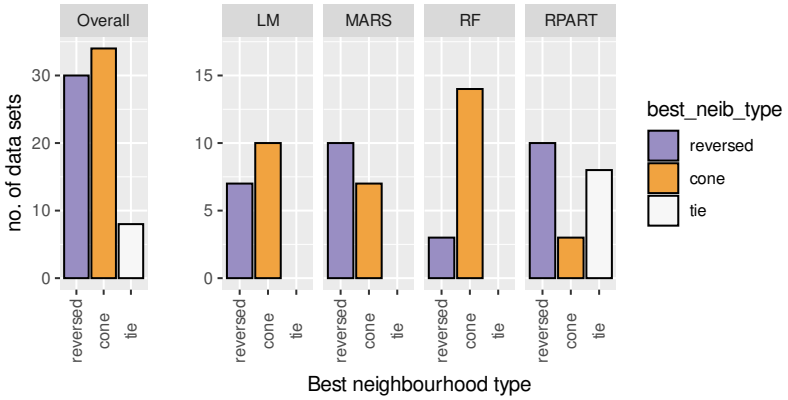


Figure 4.7: Number of data sets achieving their top results with each type of neighbourhood

Figure 4.7 shows the number of data sets that obtained their top results with the original and the reversed cone, respectively, in general and for each learning model.

Overall, the original cone works best in more cases, a result somewhat skewed by the results using RF, where the original cone worked best for 80% of the data. Still, the reversed cone beats or ties with the original half the time, being the best choice for 44% of the data and learning model pairs.

We can see clear differences between models. The reversed cone worked best for more data than the original when using MARS and RPART; the original cone worked best for more data sets when using LM or RF. RF had very imbalanced results; only three data sets got their best results with the reversed cone. RPART was the only model that led to ties.

4.3.3 Impact of Data Set Characteristics

We established that, for some data sets, the reversed cone performed best, as long as the remaining parameters were chosen to be "optimal". It could be helpful to know if these data sets share some common characteristics. With this aim, we trained a Conditional Inference Tree on a meta data set as we explained in Section 4.2.4.

Figure 4.8 shows the tree comparing experiments that used the same parameters, other than neighbourhood type, against each other. We included the parameters as meta-features, and the algorithm even chose parameter α for a split. All the splits, including that one, show statistical significance.

The reversed cone only outperformed the original cone more often than not when we set parameter α to 0.5 or lower, learned the model using MARS or LM, and the data contained fewer outliers, as defined by boxplot statistics. If we used RPART on data with higher average normalised distance between locations, the proportion between the the two types of neighbourhood was almost balanced (with some ties).

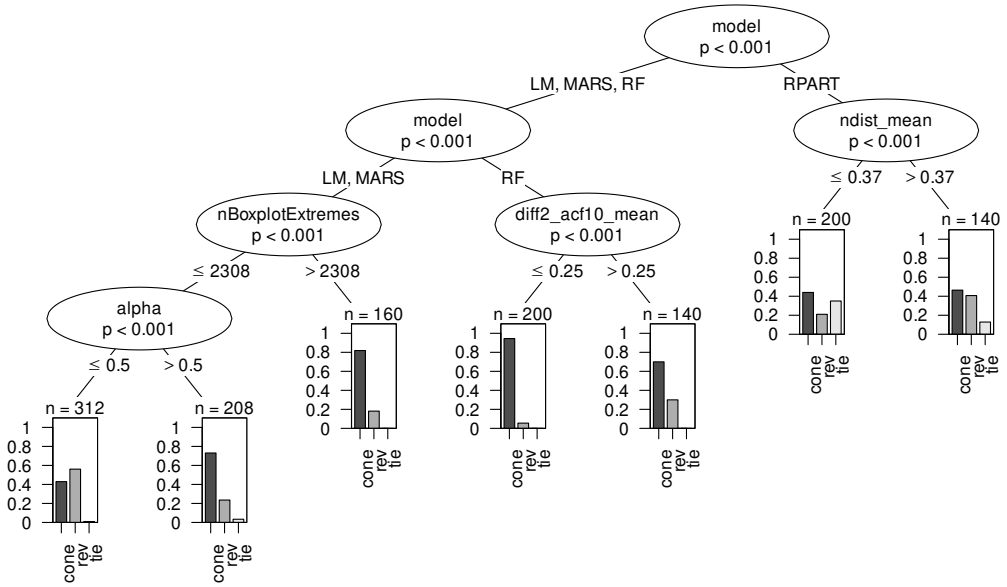


Figure 4.8: Meta-classification trees with fixed remaining parameters

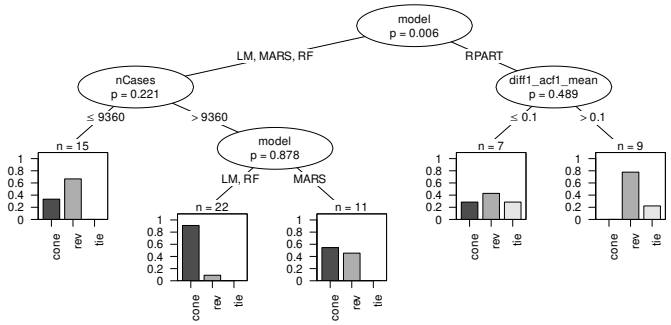


Figure 4.9: Meta-classification trees with "optimal" remaining parameters

In Figure 4.9, we filtered the experiments to only include the ones whose parameter combination was "optimal". That is, we focused on the experiments with the lowest NMAE for each data set and learning model pair. This time, we excluded the parameters of these experiments as meta-features, because (a) we wanted to isolate the data characteristics, and (b) the parameters' distributions may be imbalanced, unlike in the previous tree where all parameter combinations appeared the same number of times. This was a much smaller meta data set, and only the root split shows statistical significance.

We found three sets of circumstances where the reversed cone most often got the best results. Compare this with the previous tree, with non-optimal parameters, where only one leaf contained more instances of the reversed cone winning (*cf.* Figure 4.8). The reversed cone won more often when using RPART. If the average first autocorrelation coefficient of the differentiated time series was higher than 0.1, the original cone never won – only tied

in some cases. If using models other than RPART, the reversed cone also outperformed the original more often on smaller data sets (data sets 4 to 7).

4.4 Discussion

We found that, on average, our "reversed cone" strategy could not improve forecasting, but it worked for some data set and learning algorithm combinations – especially if we chose the remaining hyper-parameters to be optimal.

Our meta-analysis showed that, given optimal parameters, the reversed cone most often beat the original proposal when using a simple regression tree or when training LM and MARS on smaller data sets. All other parameters being equal, including neighbourhood size, the reversed cone seemed to work best when using LM or MARS on data sets with fewer outliers, and features extracted with a lower α. A lower α means the spatio-temporal distance was mostly determined by the temporal distance between observations, translating into a neighbourhood cone with a wider base and a shorter height, including farther spatial neighbours and excluding older historical data.

Related work. Since the publication of the work by Ohashi and Torgo that was the basis for this chapter's proposal, other authors have proposed their own methods to calculate spatio-temporal indicators.

In a similar approach, Appice et al. [2013b] calculate summary statistics within spatio-temporal neighbourhoods determined through a spatio-temporal clustering process. Like us, they then apply a global standard regression algorithm to the transformed data. They favourably compare their results against using historical data from the same location alone. Pravilovic et al. [2017] also include a spatio-temporal clustering step to define useful neighbourhoods, but they post-process the summary statistics using PCA to remove collinearity before modelling the data. Instead of training a global standard regression model, they learn a local VAR model for the resulting multi-variate time series at each location. They compare their results against univariate ARIMA and ARIMA variants, finding that their new proposal works better for phenomena that are stable in time, even if possibly non-stationary (e.g., air temperature and solar radiation).

Ceci et al. [2017] test several combinations of separate spatial and temporal indicators on photovoltaic energy production applications. They suggest that PCNM [Dray et al., 2006] accounts for spatial autocorrelation best, while hour and day scaled to $[0, 1]$ were sufficient to encode temporal autocorrelation.

Strengths and limitations. Ohashi and Torgo [2012] tested their proposal on a single

application, forecasting wind speed in two locations (though they used historical data from additional neighbouring locations to extract the indicators). We applied and tested it against our variant on seventeen real-world data sets.

Choosing optimal parameters *a posteriori* should result in a fair enough comparison, since we treated both strategies – with the original or the reversed cone – equally. However, without an internal evaluation method or validation set, the evaluation framework departs from real-world scenarios.

Future work. Additional experiments with a nested evaluation design, using an internal estimation method to tune parameters, could confirm whether we can expect to attain the benefits suggested by the optimal performance results.

The two cones, in their original and reversed directions, need not be exclusive, but could be complementary instead. Future work could explore whether simultaneously using both cones can capture different aspects of the same phenomena, improving performance.

4.5 Summary

We tackled the problem of numeric spatio-temporal forecasting by extracting features from data so that off-the-shelf regression algorithms can learn models of the data. The extracted features, also called spatio-temporal indicators, incorporate contextual information from spatio-temporal neighbourhoods.

We tested a variant of the spatio-temporal indicators proposed by Ohashi and Torgo [2012]. We reversed the direction of their conic spatio-temporal neighbourhood, so that the cone's vertex was at the observation and the spatial radius widened to contain older observations from more distant locations.

Testing the two variants on seventeen real-world data sets, we concluded that, on average, features using the original proposal ranked better. However, depending on the parametrization, reversing the cone can prove beneficial.

With optimal parameters, the reversed cone beat or tied for best with the original cone for half of the learning model and data pairs; it worked best for more data sets than the original when using decision trees and MARS, and when using LM on smaller data sets.

Future work should use nested validation to confirm that the results of optimal parametrizations can be replicated. This work also opens the possibility to test whether the two cones can complement each other.

Chapter 5

Resampling Imbalanced Spatio-temporal Data

5.1 Introduction

Abnormal weather conditions, pollution level spikes, and fire ignitions are examples of rare or extreme events with severe consequences for the affected communities. They often develop as the effect of uncommon natural factors on spatio-temporal processes or due to the impact of changes in underlying factors [Widmer and Kubat, 1996]. Their characteristics make them difficult to predict, but accurate anticipation often allows action to avoid or mitigate the worst consequences [Baxevani and Wilson, 2018].

Standard learning methods often assume data to follow a Gaussian distribution. But real-world spatio-temporal data often present heavy-tailed behaviours, leading to poor predictions (as shown by [Eklund et al., 2016] for fMRI "spatial" data). Most standard evaluation metrics assume uniform domain preferences, i.e., that users value predictions across the target domain equally. Using such metrics during and after model optimisation specialises the forecasters on predicting average target values, deteriorating accuracy on extreme values [Branco et al., 2016b].

In Section 2.4, we presented the common resampling strategies that compensate for some of these shortcomings. We also discussed how they were extended to work on regression problems [Torgo et al., 2013, Branco et al., 2016a]. In 2017a, Moniz et al. designed resampling strategies that improve time series forecasting by introducing a temporal bias in the random processes. To the best of our knowledge, similar approaches to spatio-temporal data had not been proposed.

Inspired by their work, we propose a novel set of biased resampling strategies for spatio-temporal forecasting of extreme values, based on two proposals for bias: one static and

one relevance-aware. Both consider spatial and temporal implicit relationships between observations and allow the temporal and spatial dimensions to contribute unevenly to the final spatio-temporal weight. The relevance-aware weighing considers the usefulness of obtaining accurate predictions for a particular example.

We empirically compare our proposals against purely random versions of under-sampling and over-sampling. We run experiments using a subset of the real-world data presented in Chapter 3 that had imbalanced domains. We analyse the strategies' parameter sensitivity, the precision-recall trade-off, the results' robustness, and performance impact of data characteristics.

5.2 Materials and Methods

5.2.1 Spatio-Temporal Bias Resampling Strategies

Under-sampling and over-sampling are well-known resampling strategies that shift the target domain distribution by reducing the number of normal cases or increasing rare cases. Often, the strategies randomly select the observations to be removed or replicated, though some proposals introduce a bias in the process.

In this section, we detail our proposal for spatio-temporal bias in resampling strategies.

5.2.1.1 Spatio-Temporal Bias

As emphasised throughout our work, observations in spatio-temporal data have implicit dependence relationships along the temporal and spatial dimensions.

We hypothesise these relationships may affect observations' usefulness when learning a model. Thus, we should be able to leverage spatial and temporal contextual information to improve how we select data during resampling. We propose a sampling weighting function, $W : \mathcal{Y}_{i,j} \rightarrow [0, 1]$, that regulates the probability that random under-sampling will select a regular observation to remain in the training set, or random over-sampling will select a rare observation to replicate. This weight is a combination of a spatial and a temporal component.

We also know that autocorrelation across time and space differs depending on the domain. Therefore, we hypothesise that each dimension should be able to contribute to an observation's weight unevenly. We introduce a parameter α to balance our weight's spatial and temporal components.

We consider two different approaches to calculate both spatial and temporal weights: static and relevance-aware. The static weight does not depend on the observed values, while the

relevance-aware weight considers their relevance as defined in Section 2.5.2.2. In both cases, a higher weight increases the chance that we will find the observation in the training set after resampling.

Static Weights Spatial and temporal static weights are constant across time and space, respectively, and do not depend on observed values.

More recent observations may hold more up-to-date information that could be more useful to the predictive task. Therefore, resampling strategies may benefit from prioritising them. We define the temporal weight, W^T, as a linear function that proportionately attributes higher weight to more recent observations.

Since values measured at neighbouring locations often correlate, models may gain relevant information about a location from its neighbours. It may be beneficial for resampling strategies to prioritise data from isolated locations since, in the absence of neighbouring data, models may experience more difficulty making useful predictions for them. We define the spatial weight, W^L, as a linear function of the shortest spatial distance to any neighbour that proportionately attributes more weight to more isolated locations.

Relevance-Aware Weights Relevance-aware weights prioritise observations according to their distance to relevant observations, as defined by a relevance function, ϕ, and a relevance threshold, t_R. In this chapter, we did not assume access to domain expert knowledge to help us define relevance functions and thresholds. Relevance functions were instead automatically derived using an approach described in Section 2.4.1, proposed by Ribeiro [2011], and the relevance threshold was set to 0.9.

Temporal weights consider time differences to relevant cases at each location — viewing the data as a set of time series. Spatial weights consider spatial distances to relevant cases at each time-stamp.

The relevance-aware temporal weight assumes that:

1. Normal observations that precede extreme cases may hold important information about the processes that lead to extremes in the series, so they should be given higher probability of remaining after under-sampling;

2. The first extreme in a sequence of consecutive extreme values may hold more information about the shift that caused the sequence, so it should be given a higher probability of being replicated during over-sampling.

Temporal weights, W^{T_ϕ} are calculated at each location. We give maximum weight to the first extreme value in a sequence of extreme values (with t_R or higher relevance). The weight

then linearly falls until it reaches zero at the next normal value. From there, it goes up linearly until the next extreme value, which will again have maximum weight.

We cannot access values measured before or after a time series, so we must make assumptions to weigh the first and last observation sequences. We assume the first observation to be the first value in a sequence of its type: if it is an extreme case, it is given maximum weight; otherwise, it is given zero weight. We decided to prioritise the last observations to safeguard the possibility that the next value is extreme. Thus, we treat the last value as extreme, whether or not that is true.

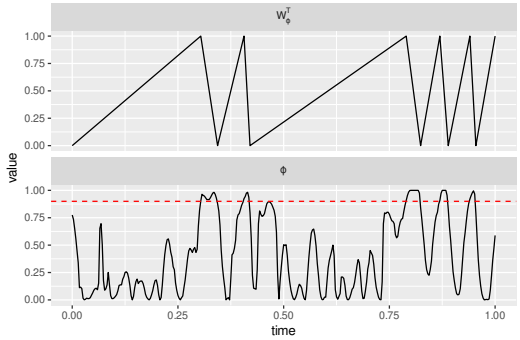


Figure 5.1: Top: relevance-aware temporal weight, W^{T_ϕ} (y-axis) as it changes across time (x-axis; normalized so 1 corresponds to the most recent observation), calculated for a specific station in data set 31. Bottom: relevance function automatically derived for the same station (normalized to [0, 1]). The red dashed line corresponds to the relevance threshold, t_R, at and above which an observation is considered highly relevant

We calculated an example relevance function and temporal weight for a station in data set 31 as an example (see Figure 5.1). The data were previously described in Section 3.2.2.2 and further relevant details will be given in Section 5.2.2.

The relevance-aware spatial weight assumes that:

1. More isolated cases could be more at risk of being filtered out or ignored during the learning process, so they should be given a higher probability of being replicated;

2. Eliminating normal cases that are spatially closer to extreme values could help make borders around extremes easier to learn, so normal cases farther away from extremes should be given a higher probability of remaining.

Spatial weights, W^{L_ϕ}, are calculated at each time snapshot. The weight is directly proportional to the shortest spatial distance to a neighbouring location with an extreme value.

Combining Weights We combine our spatial and temporal weights into a static spatio-temporal weight, W^S, (Equation 5.1) and a relevance-aware weight, W^ϕ, (Equation 5.2). In

both equations, α balances the temporal and spatial components, i is a time-stamp index, j is a location index, and ϵ is a small value added to ensure all observations have non-zero probability of remaining or being replicated during resampling.

$$W^{S}_{i,j} = \alpha \times W^{T_S}_{i,j} + (1 - \alpha) \times W^{L_S}_{i,j} + \epsilon \tag{5.1}$$

$$W^{\phi}_{i,j} = \alpha \times W^{T_\phi}_{i,j} + (1 - \alpha) \times W^{L_\phi}_{i,j} + \epsilon \tag{5.2}$$

We had initially proposed another formulation [Oliveira et al., 2019], which combined a static temporal weight with a relevance-aware spatial weight (Equation 5.3).

$$W_{i,j} = \alpha \times W^{T_S}_{i,j} + (1 - \alpha) \times W^{L_\phi}_{i,j} + \epsilon \tag{5.3}$$

The under- and over-sampling algorithms only randomly select observations of a single type to remain or replicate in the training set, respectively. Therefore, before resampling, we normalise the weights for normal and extreme cases separately.

Figure 5.2 shows an example of the automatically derived relevance function, the spatial, temporal and spatio-temporal weights for data set 31. The heatmap cannot show distances between locations along the y-axis that explain both W^{L_ϕ} and W^{L_S} values. To calculate W^{ϕ}, we deemed values normal if their relevance was below a 0.9 relevance threshold, and we weighed the spatial and temporal components equally (parameter α set to 0.5).

5.2.1.2 Resampling Algorithms

Next, we describe our proposed variants of biased under- and over-sampling algorithms. Algorithm 5.1 shows their pseudocode.

Spatio-temporal Biased Under-Sampling (STRUS) Random under-sampling (RUS) keeps all the extreme observations, and then randomly selects a sub-sample of normal observations to remain in the training set. We propose a method that differs in how it conducts the sub-sampling. In random under-sampling, all normal observations have the same probability of being selected; in our biased random under-sampling, the probability is proportional to their bias weight.

We use random selection without replacement. The parameter u ($0 < u < 1$) defines the

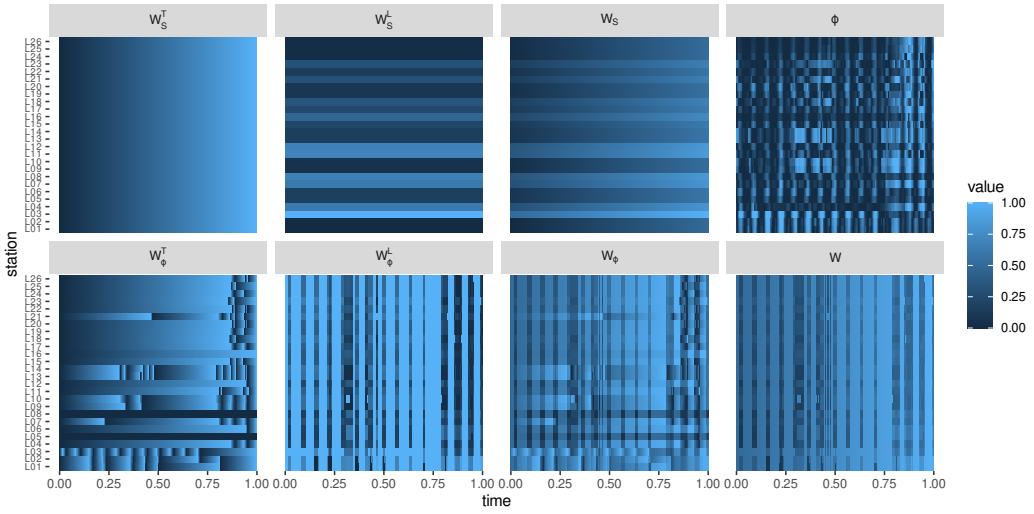


Figure 5.2: Heatmaps of the relevance function, ϕ, and spatio-temporal bias weights calculated for observations captured at each location (y-axis) and time-stamp (x-axis; normalized so 1 corresponds to the most recent observation) for data set 31, as well as their separate spatial and temporal components (α was set to 0.5 so both dimensions are considered equally). The relevance threshold at and above which an observation is considered highly relevant was set to 0.9. The static spatio-temporal bias weight, W^S, is a combination of the static spatial, W^{L_S}, and temporal weights, W^{T_S}. The relevance-aware spatio-temporal bias weight, W^ϕ, is a combination of the relevance-aware temporal, W^{T_ϕ}, and spatial weights, W^{L_ϕ}. The originally proposed bias weight, W, is a combination of W^{T_S} and W^{L_ϕ}.

percentage of normal cases remaining in the training set. Setting $0 < \alpha < 1$ and using a static weight, this strategy will more likely select a normal observation recently recorded at an isolated location. When using a relevance-aware weight, it will more likely select a normal value recorded after a sequence of normal values at a location distant from extreme observations.

Spatio-temporal Biased Over-Sampling (STROS) Like random over-sampling (ROS), this strategy starts with all observations, and then randomly selects a sample of extreme observations to replicate and add to the training set. Biased random over-sampling differs by selecting observations with a probability proportional to their bias weight.

We use random selection with replacement. The parameter o ($o > 0$) specifies the percentage of extreme cases added to the training set; setting $o = 1$ doubles the number of extreme cases. Setting $0 < \alpha < 1$ and using a relevance-aware weight, this strategy will more likely select the first extreme value in a sequence of consecutive extreme values recorded at a location distant from other extreme observations.

Spatio-temporal Biased Over-Sampling with Gaussian noise (STGAUSS) This strategy adds random noise to extreme replicas' predictors and target values before adding them to the training set [Branco et al., 2016a]. The added Gaussian noise has zero mean; its standard deviation is a p fraction of similarly extreme target values' standard deviation in the training set.

Algorithm 5.1 Spatio-temporal bias random under- and over-sampling (with or without added Gaussian noise.)

function STRandReSampling($D, Y, \phi(Y), t_R, B, u, o$)
    ▷ D - A data set
    ▷ Y - The target variable
    ▷ $\phi(Y)$ - User specified relevance function
    ▷ t_R - The threshold for relevance on y values
    ▷ W - Spatio-temporal bias weight
    ▷ B - Variant of biased resampling (STRUS for biased under-sampling; STROS for biased over-sampling; STGAUSS for biased over-sampling with added Gaussian noise)
    ▷ u - Percentage of under-sampling (if $B = STRUS$)
    ▷ o - Percentage of over-sampling (if $B = STROS$ or $B = STGAUSS$)
    ▷ *pert* - Fraction of standard deviation of original data used as the standard deviation of the random Gaussian noise added to selected cases

    $D_R \leftarrow \{D_i : \forall y_i \in Y, \phi(y_i) \geq t_R\}$ ▷ Highly relevant cases
    $D_N \leftarrow \{D_i : \forall y_i \in Y, \phi(y_i) < t_R\}$ ▷ Normal cases
    if $B = STRUS$ **then** ▷ Biased random under-sampling
        $TgtNr \leftarrow |D_N| \times u$
        $newData \leftarrow D_R$ ▷ Highly relevant cases are to be kept
        $selCases \leftarrow$ sample$(tgtNr, D_N, W)$ ▷ Biased random selection of a number of normal cases from D_N
    else if $B \in \{STROS, STGAUSS\}$ **then** ▷ Biased random over-sampling
        $tgtNr \leftarrow |D_R| \times o$
        $newData \leftarrow D$ ▷ All cases kept in the new data set
        $selCases \leftarrow$ sample$(tgtNr, D_R, W)$ ▷ Biased random selection of a number of rare cases from D_R
        if $B = STGAUSS$ **then**
            $selCases \leftarrow$ addGaussianNoise$(selCases, pert)$ ▷ Add Gaussian noise to selected cases
        end if
    end if
    $newData \leftarrow c(newData, selCases)$ ▷ Add selected cases to the new data set
    return $newData$
end function

5.2.2 Data Sets

In this chapter, we use a subset of the real-world data previously described in Section 3.2.2.2. Data was collected from below 200 to more than 6.000 times at 20 to 70 irregularly distributed geo-locations, with varying rates of recorded extreme values (see Table 5.1; *cf.* Table 3.3).

Figure 5.3 shows boxplots of the imbalanced variables used in this chapter and the relevance functions automatically derived from them. In Table 5.1, we show the percentage of extreme cases based on the number of observations with 0.9 or higher relevance, i.e., the shadowed areas in Figure 5.3. The percentages of extreme values recorded range from 2.4% to 8.6% of instances. Like in Chapter 3, we use each of the ten variables as if they were independent and univariate data sets.

Table 5.1: Description of imbalanced real-world data sets used in all experiments. Extremes are those with 0.9 or higher relevance, according to an automatically derived relevance function

Data source name	ID	Variables	Extremes (%)
MESA Air Pollution[1]	10	NO_X conc.	7.3
NCDC Air Climate[1]	20	precipitation	6.0
TCE Air Climate[1]	30	ozone	6.3
	31	air temperature	3.8
	32	wind speed	2.4
airBase[2]	60	PM_{10} conc.	7.5
Beijing UrbanAir[3]	70	NO_X conc.	3.5
	71	PM_{10} conc.	5.5
	72	wind speed	8.6
	73	PM_{25} conc.	3.8

[1] From Pravilovic et al. [2018]; Downloaded at: `http://www.di.uniba.it/appice/software/COSTK/data/dataset.zip`, accessed on 12 March 2018;
[3] Loaded from R package spacetime [Pebesma, 2012] version 1.2–3 (`https://cran.r-project.org/web/packages/spacetime/index.html`, accessed on 9 December 2020);
[4] From Zheng et al. [2013]; Downloaded at: `https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/Air20Quality20Data.zip`, accessed on 18 October 2017; Since there was more than one measurement for some hours we rounded the time-stamps to the closest hour, and calculated the median values per hour and location.

5.2.3 Experimental Design

Our experiments aim at evaluating the following strategies: random under-sampling (*RUS*), over-sampling (*ROS*), and over-sampling with added Gaussian noise (*GAUSS*), and spatio-temporal bias random under-sampling (*STRUS*), over-sampling (*STROS*), and over-sampling with added Gaussian noise (*STGAUSS*). ROS, RUS, and GAUSS are unbiased random versions of STROS, STRUS, and STGAUSS that work as if all observations had the same non-zero weight.

This section describes the experimental design used to test and compare these strategies amongst each other and against a baseline without resampling. Next, we detail the training process, including data pre-processing steps and learning models. The following section describes the evaluation setup, including performance metrics and evaluation methods used to estimate them. Finally, Section 5.2.4 specifies how we analyse data characteristics' effects on our proposals' efficiency.

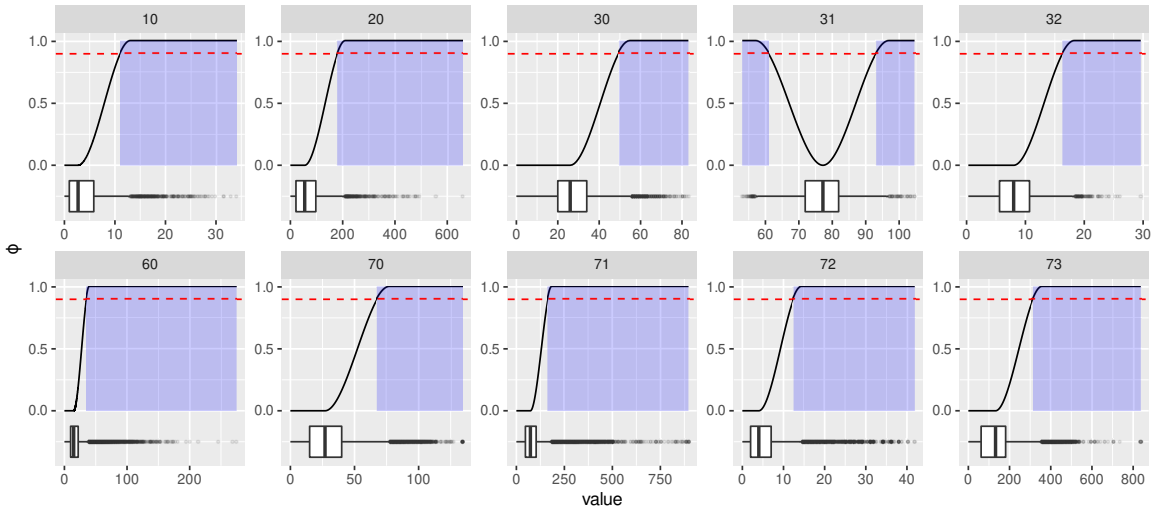


Figure 5.3: Relevance functions automatically derived for imbalanced data from their boxplots. Values are considered extreme at or above the 0.9 relevance threshold – the shadowed areas in the plots. Target domain scales differ

5.2.3.1 Training Workflow

This section explains how we extracted features, how we handled missing data, and which regression algorithms were used.

Feature engineering. As in previous chapters, we need to transform the data so that we can apply standard machine learning algorithms and compare resampling methods' effects. Feature engineering follows the steps detailed in Section 3.2.2.2,[1] as proposed by Ohashi and Torgo [2012].

Handling missing data. Data sets 6 and 7 only include measurements for a fraction of location and time-stamp pairs (from 49% to 68%). We followed the steps described in Section 3.2.3.2 to deal with the issue before applying a resampling strategy or training a model on each data subset we used for training. First, we discarded columns, then rows with 20% or more values missing from the training set; we filled in the remaining missing values in both training and test sets with each column's median value in the training set.

Regression algorithms. We run the experiments using three standard regression algorithms as implemented in free and open-source **R** packages:

MARS a multivariate adaptive regression splines model, implemented in *R* package *earth* [from

[1] There was only one difference in the methodology followed in this chapter: here, we used $\alpha = 0.25$ for all data, including data set 2, for which we had used $\alpha = 0.5$ in Chapter 3.

mda:mars by Trevor Hastie and utilities with Thomas Lumley's leaps wrapper., 2018];

RPART a regression tree, implemented in *R* package *rpart* [Therneau et al., 2017];

RF a random forest, implemented in *R* package *ranger* [Wright and Ziegler, 2017]

The experiments used each model's default parameters, except for RF's number of trees, which was set to 250 (the default was 500). RF is more computationally intensive than other models; reducing the number of trees mitigates its computational cost. We chose to grow 250 trees because Probst and Boulesteix [2018] empirically showed that, for many data sets, "*the biggest performance gain in the out-of-bag curves can be seen while growing the first 250 trees*".

The same study [Probst and Boulesteix, 2018] also suggested that if we measured performance by metrics based on mean quadratic loss, such as mean squared error, performance would continually improve with the number of trees. For this chapter, we used performance metrics based on the concept of utility instead.

5.2.3.2 Parametrization Schemes

We ran two types of experiments with similar design. Both types deploy an evaluation method to estimate performance metrics. One type uses the same set of resampling hyper-parameters for an entire data set – the parameters here can be fixed *a priori* or selected *a posteriori*); the other, deploys an additional, internal evaluation method to determine the best set of hyper-parameters for each training set and learning algorithm.

We will report the empirical results following three different parametrization schemes, which we describe below. We applied them equally to our biased resampling strategies and their random counterparts.

Internal tuning. Internally tuning parameters should accurately approximate the error incurred by using tuned parameters in the real world; it consumes additional computational resources, but it should attain better results than just using default values. We will report the results of running the experiments with an internal evaluation method.

Fixed parameters. In some real scenarios, employing default parameters, regardless of data characteristics, may suffice, especially if strategies are relatively insensitive to parameter change or the defaults were chosen based on past empirical results. Our fixed parametrization scheme approximates this scenario, but it is at a disadvantage against it – we defined parameters *a priori* for all data sets before empirical results could guide our decision. We fixed them at values that sit in the "middle" of our search grid, which may not correspond to our recommendations for future default parameters to use when tuning is infeasible.

Optimal parameters. Our "optimal" scheme *a posteriori* selects the parameters that worked the best for each data set when applied across all testing blocks in the prequential evaluation method. It cannot be replicated in real scenarios, but it can help establish our proposals' full potential.

Parameter grid. Resampling percentages, o and u, are common to all strategies. Parameters α and ϵ are only needed for biased resampling approaches. If Gaussian noise is to be added during over-sampling, then the perturbation size needs to be set as well.

We set the Gaussian perturbation to 0.1 of the standard deviation of similarly extreme values in the training set, and ϵ to 1×10^{-4}. The remaining set of parameters were tested across all combinations of $\alpha \in \{0,\ 0.25,\ 0.5,\ 0.75,\ 1\}$, and $u \in \{0.2,\ 0.4,\ 0.6,\ 0.8,\ 0.95\}$ or $o \in \{0.5,\ 1,\ 2,\ 3,\ 4\}$ – a total of 25 possibilities for each type of biased resampling. The fixed parametrization scheme used $\alpha = 0.5$, $o = 2$, and $u = 0.6$.

5.2.3.3 Evaluation Framework

To properly define our evaluation framework, we need to select an evaluation method to estimate error, and appropriate metrics to evaluate performance. Our resampling strategies also include hyper-parameters that need to be set following a parametrization scheme.

Estimation method. As in Chapter 4, we opted to use, as our main evaluation method, the prequential temporal block evaluation with growing window, described in Section 3.2.1, with $K = 10$.

Besides the reasons for this choice already listed in Section 4.2.3.3 of the previous chapter, we should add that this method also (a) calculates more than just one estimate, which facilitates investigating performance over time, and (b) includes data from every location in all training sets, which may be helpful when we are automatically deriving relevance functions based on the training sets, and we are interested in global extremes.[2]

Once again, even though the error estimation method evaluates predictions in batches, the prediction horizon for our experiments at time t is $t+1$. Therefore, our experiments assume access to all previous true values. Predictors are based on previous neighbouring values and calculated before any train-test divisions.

For the internal evaluation method, we chose time-blocked cross-validation (CV-Tb) because (a) it is similar to the external prequential method, but it uses all the available data at each step, which may be helpful when training windows are small; (b) it mitigates some of the

[2]Note that in this chapter we assume that all events follow the same distribution, thus using a global definition of what constitutes a rare and extreme case, instead of a local one.

severe error under-estimation incurred by standard CV; and (c) it does not completely ignore temporal order, even though it sometimes tests future models on past data.

Performance metrics. As previously mentioned, standard metrics like MSE cannot adequately assess a model's ability to predict under-represented, extreme values. This is precisely the aim of this chapter, so alternative metrics must be used. We opted for the utility-based metrics presented in Section 2.5.2.2, namely the simplified utility-based precision (Equation 5.4) and recall (Equation 5.5) proposed by Moniz et al. [2019], and the F_1^u-score (Equation 5.6) combining them. By using an F_β-score with $\beta = 1$, we weigh precision and recall equally.

$$prec_\phi^u = \frac{\sum\limits_{\phi(\hat{y}_i)\geq t_R, \phi(y_i)\geq t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(\hat{y}_i)\geq t_R} (1 + \phi(\hat{y}_i))} \tag{5.4}$$

$$rec_\phi^u = \frac{\sum\limits_{\phi(\hat{y}_i)\geq t_R, \phi(y)\geq t_R} (1 + u(\hat{y}_i, y_i))}{\sum\limits_{\phi(y_i)\geq t_R} (1 + \phi(y_i))} \tag{5.5}$$

$$F_1^u = 2 \cdot \frac{prec_\phi^u \cdot rec_\phi^u}{prec_\phi^u + rec_\phi^u} \tag{5.6}$$

Even though these metrics require setting a threshold for highly relevant cases (set to 0.9 in this chapter), they are not solely based on categorising cases as above or below the threshold; they incorporate numerical error into the calculation.

Relevance functions were calculated automatically based on each training set's target domain boxplot statistics (examples using the whole data sets in Figure 5.3).

Since distributions may differ as the training window grows, highly relevant observations at one step of the prequential method may not be considered so in the next. The resulting varying percentages of extreme cases in the testing set can reduce our estimates' usefulness. In particular, some testing blocks may contain only normal values, which precludes us from calculating utility-based metrics. We only found this problem in two blocks in data set 32, so they were ignored, and metrics averaged over the remaining blocks.

Repetition. Our resampling strategies have a random component that could cause variability in our results. We repeat every prequential block evaluation method ten times to obtain more robust results for models RPART and MARS, whether or not tuning parameters internally. RF is significantly more computationally expensive, so we only ran experiments with RF once.

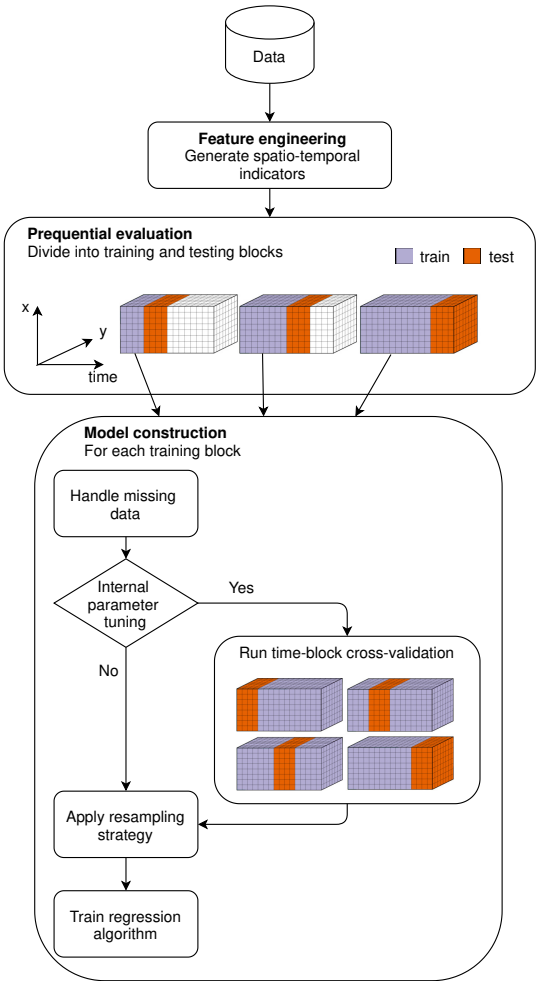


Figure 5.4: Summary of the workflow followed in the experiments. First, we calculate spatio-temporal indicators for the whole data set that will be used as features in the spatio-temporal forecasting task (the prediction of numeric values for $t + 1$, at each time-stamp t and for all locations). Then, we proceed to the prequential block evaluation, where we apply a training process to a growing number of blocks that precede the testing block in time. This training process includes (1) handling missing data, (2) if applicable, running time-block cross-validation to internally tune parameters, (3) applying the chosen resampling strategy, and (4) training a regression algorithm on the resulting training set.

Figure 5.4 shows a summary of the workflow we described. Our proposals are implemented in an **R** package, *STResampling*, available at `https://github.com/mrfoliveira/STResampling-JDSA2020`. All the code and data necessary to replicate our results is also included.

5.2.4 Meta-Analysis Methodology

After running the experiments, we assess data set characteristics' impact on our proposals' efficiency through meta-learning. We followed the same approach described in Section 4.2.4, with an additional step: estimating feature importance.

Extracting spatio-temporal data characteristics. We used the same features extracted in Chapter 4 (described in Section 4.2.4), with two additions – the average percentage of missing predictor values per observation, and the average correlation between the target value and its predictors. In the previous chapter, we were comparing how different spatio-temporal indicators perform as predictors, so we excluded their correlation to the target from the analysis, since it would vary for every experiment. In this chapter, this meta-feature was included since correlation with the predictors means the target values were correlated to past neighbouring values or, more specifically, to their summary statistics. A complete list of the 35 features we used can be found in Table C.8 (page 135).

Meta-labels. We labelled each experiment depending on whether the biased version of resampling, on average, earned a higher average F_1^u-score than its random counterpart, on the same data set and when using the same learning algorithm.

If assessing the internal tuning parametrisation scheme, we only have one experiment to label, per model and data set pair, for each type of resampling (under- or over-sampling). We excluded resampling parameters from the meta-features, given that they can differ across training blocks following this parametrisation scheme.

In experiments without internal tuning, we compared resampling strategies using the same parameters, on the same data set and using the same learning algorithm. Therefore, we had 25 experiments using different parameter combinations to label for each type of resampling strategy.

Meta-learning and feature importance. We trained two types of classification models on the meta data: a Conditional Inference Tree (CIT), and a Conditional Inference Forest (CIF), first proposed by Hothorn et al. [2006]. The CIT creates an interpretable, visual representation of the most determinant factors affecting our proposals' efficiency. The CIF can help discover the meta-features' importance.

Probst et al. [2019] recommends measuring feature importance by conditional permutation importance [Strobl et al., 2008] calculated from CIFs with 0.632 subsampling (without replacement). This measurement is robust to features' differing number of categories [Strobl et al., 2007], and to correlated features [Strobl et al., 2008]. We opted for a variation of this measure that uses Area Under the Curve (AUC) instead of accuracy in its computation, making it more robust to class imbalance [Janitza et al., 2013]. As Janitza et al. [2013], we grew the trees to maximal depth. We set *mtry* to 6, the square-root of the number of predictors, a rule-of-thumb generally accepted as being reasonable [Probst et al., 2019].

Though they each used different importance measures, Lunetta et al. [2004] suggested that multiple thousands of trees may be necessary to get a stable measure of random forest importance, and Nembrini et al. [2018] set their number of trees to 5000. We grew our CIFs to comprise 5000 Conditional Inference Trees grown to maximal depth. This number of trees produced consistent meta-feature rankings across the ten runs we averaged them over.

5.3 Empirical Results

This section is dedicated to the results obtained using our evaluation framework and the three different parametrization methodologies: (a) setting fixed parameters *a priori* for all data sets, (b) internally tuning parameters for each training set, and (c) selecting optimal parameters *a posteriori* for each data set.

We start with an overview of the results and analyse parameter sensitivity, the precision-recall trade-off, the effects of data set characteristics, and the variability of our results.

For the sake of conciseness, we relegated the results of our original, mixed proposal for bias to Appendix C. Under certain conditions, it benefited resampling more than our other bias proposals – for example, it ranked best when over-sampling with optimal or fixed parameters – but our other, more conceptually consistent bias proposals represent our reasoning for the spatio-temporal bias well.

We also omitted from this section the results of adding Gaussian noise to replicas during over-sampling because regular over-sampling consistently outperformed it in both biased and random versions, regardless of parametrisation scheme.

5.3.1 Results per Parametrization Scheme

We start by investigating how our proposals rank against each other, their random counterparts, and the baseline without resampling. Later, we will check if any differences we find are statistically significant. We will also examine the best performers under different experimental conditions.

5.3.1.1 Average Ranks of F_1^uesults

We calculated the average rank determined by F_1^u for each resampling strategy. Table 5.2 shows the average across all learning model and data set pairs (the best results are in bold).

Table 5.2: Average ranks of F_1^u results obtained by each resampling method, across all data set and learning model pairings, for three different types of parametrizations. Best result for each parametrization strategy is in bold; best result per type of resampling strategy is in italics

psampling	tuning	optimal	fixed
BASELINE	6.43	6.93	6.17
ROS	4.43	4.73	3.70
STROS_S	*3.83*	*3.43*	*3.53*
STROSϕ	4.80	4.23	4.70
RUS	3.40	4.27	3.73
STRUS_S	3.10	2.70	3.77
STRUS_ϕ	**2.00**	**1.70**	**2.40**

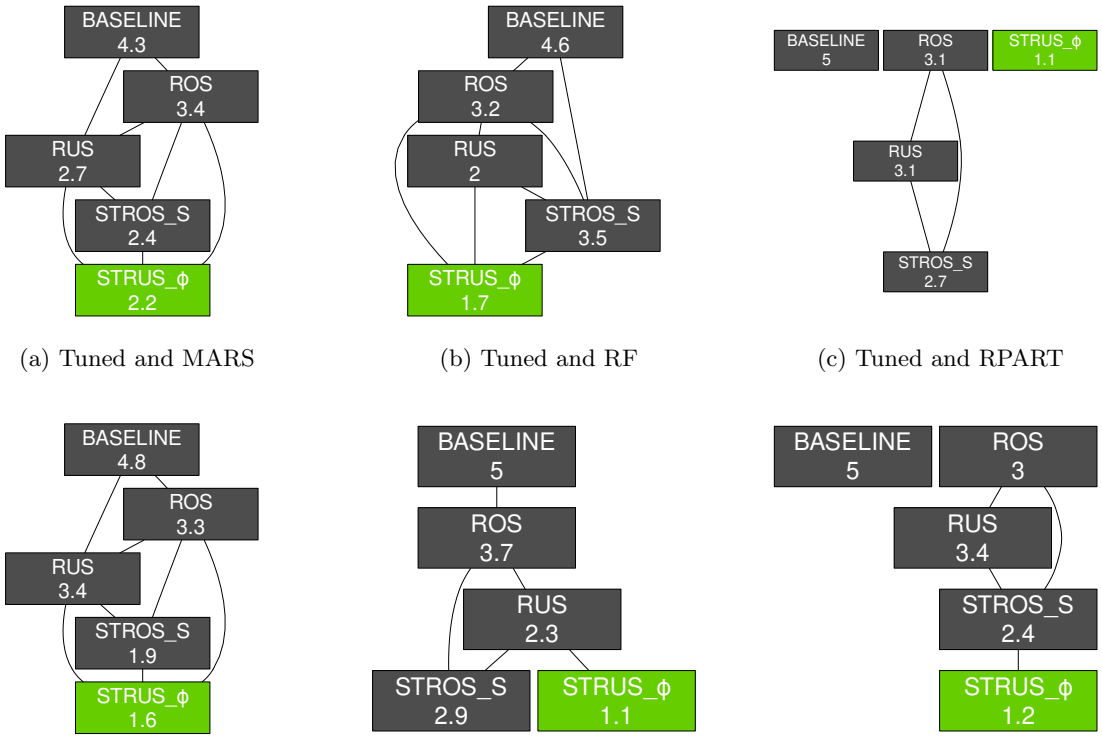
The results show that, on average, all forms of resampling (biased or not) improve the results against the baseline. Regardless of the parametrization scheme, both under- and over-sampling improve when we introduce some kind of bias – either the relevance-aware bias, the static bias, or in both cases.

Under-sampling worked best with the relevance-aware bias – this combination performs best regardless of parametrization scheme –, but the static bias still improved its results if using tuned or optimised parameters. Over-sampling worked best with the static bias[3]; the relevance-aware bias eroded its performance unless parameters were optimised *a posteriri*.

5.3.1.2 Statistical Significance

We selected the best biased strategies and tested the statistical significance of the differences in their performance between them, random resampling, and the baseline across data sets. We applied the Friedman post-hoc test with Bergmann and Hommel's correction. García and Herrera [2008] suggests it to be a more powerful alternative to the Nemenyi correction suggested by Demšar [2006] when multiple data sets and "treatments" are involved. The procedure is computationally expensive, but the implementation we used, available in *R* package *scmamp* [Calvo et al., 2016], allows testing up to nine strategies. We tested the results obtained with each learning model separately so that observations were independent.

[3]Our original bias proposal (excluded from this analysis) actually performed best with optimal or fixed parameters, but static bias outperformed it with tuned parameters. – arguably, the internal tuning scheme simulates common practice in predictive analytics better.



(a) Tuned and MARS (b) Tuned and RF (c) Tuned and RPART

(d) Optimal and MARS (e) Optimal and RF (f) Optimal and RPART

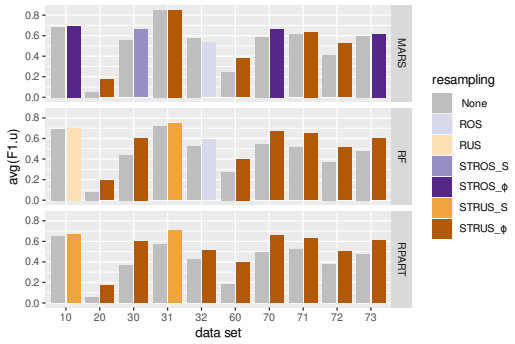
Figure 5.5: Critical difference graphs for selected strategies with internally tuned and optimal parameters, using the three learning algorithms. The best performer appears in a green box. Groups of resampling strategies that are not significantly different according to the Bergmann and Hommel's post-hoc test (at $p = 0.05$) are connected

Figure 5.5 shows critical difference graphs using internally tuned and optimal parameters. Strategies that do not differ significantly according to the test at $p = 0.05$ appear connected in the graph.

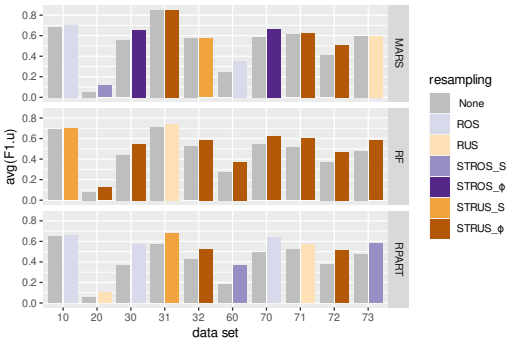
Relevance-aware biased under-sampling (STRUSϕ) beat its random counterpart (RUS) with significance when using RPART, and it beat the baseline even when RUS could not. If we internally tuned parameters, it worked significantly better than any other option.

Static biased random over-sampling (STROS$_S$) may not be significantly better than random over-sampling, but it could always beat the baseline with significance – something that random over-sampling failed to do when using MARS or RF.

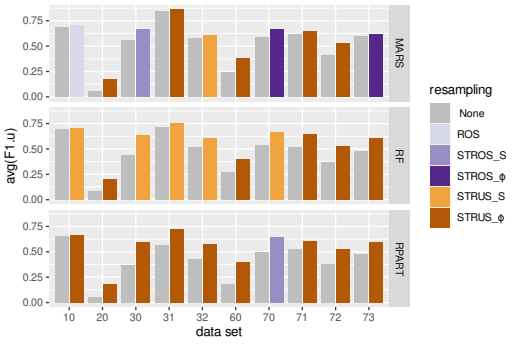
The test often cannot distinguish between our proposals and their random counterparts, but our biased resampling proposals always significantly outperformed the baseline.



(a) Internally tuned parameters



(b) Fixed parameters



(c) Optimal parameters

Figure 5.6: Baseline and best F_1^u result achieved for each data set and learning model pair (top: MARS; middle: RF; bottom: RPART). Each pair of bars correspond to one data set: the baseline, in gray, and the best result in a color indicating the resampling method

5.3.1.3 Best F_1^uesults per Model and Data Set

Now that we understand the strategies' average behaviour, we inspect the best results for each data set and learning algorithm. Figure 5.6 shows the baseline and best average F_1^u-score achieved by any resampling strategy under the three parametrization schemes.

Regardless of model or parametrization scheme, biased strategies got top results for most or all data sets, with one exception — using RPART with fixed parameters produced a tie between random and biased strategies. The relevance-aware biased strategies got top results more often than the static bias.

If we used tuned or optimal parameters, random resampling strategies only beat biased alternatives on two or fewer data sets, depending on the learning algorithm; this number rose to eight if we used fixed parameters instead. Even then, biased resampling still worked best for all data sets when combined with at least one of the three learning algorithms.

For some data sets, a single strategy works best in most situations — e.g., STRUS_ϕ works best for data set 72 regardless of learning algorithm or parametrization. For others, the best

performer varies — e.g., depending on the learning model and parametrization scheme, five of the six different strategies work best for data set 10, the smallest data set.

When using the tree-based models RPART and RF, under-sampling works best for most data sets. Only when using RPART with fixed parameters, do under- and over-sampling tie, working best for five data sets each. When using MARS, under- and over-sampling strategies achieved top results at a similar rate, regardless of parametrization.

These results show us some of the variability behind the average ranks we discussed above. For further details, Section C.1 in the Appendix contains tables of average ranks of F_1^u for each resampling strategy, aggregated by learning algorithm and by data set.

5.3.2 Parameter Sensitivity Analysis

Knowing how the strategies perform on average and when they achieve top results, we investigate further how sensitive they are to the different parameters.

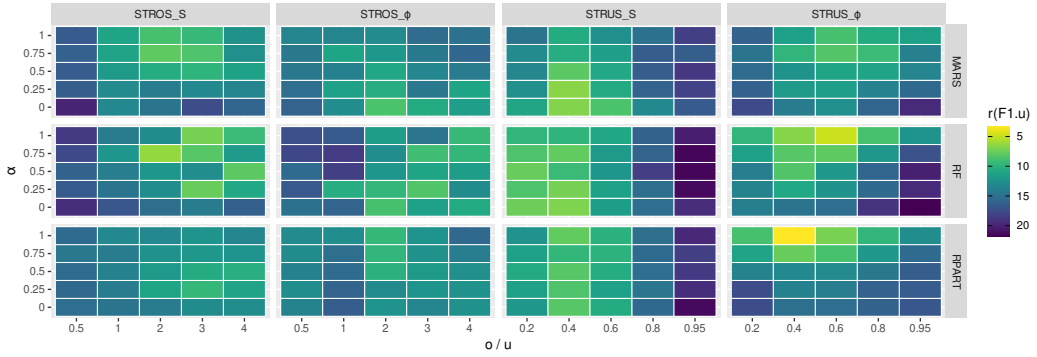


Figure 5.7: Average F_1^u rank obtained by each resampling technique when using 25 different parametrizations averaged over all data sets. Lower ranks correspond to better results

For this purpose, we examine the experiments without internal tuning, where the prequential method applied a set of parameters to all training blocks of each data set. Previously, we analyzed a subset of these experiments with optimal parameters chosen a posteriori. Now, for each data set and learning algorithm, we consider all experiments. We ranked each parameter combination against all others using the same strategy. Figure 5.7 shows these ranks averaged across data sets.

The colour gradient along the X-axis stands out more, though we can discern differences along the Y-axis as well: the average variance in ranks across resampling percentage (10.8) was about double that across α (6.0). This points to the strategies being more sensitive to variations in resampling percentage than in α. Relevance-aware biased under-sampling (STRUS_ϕ) is the only strategy whose performance varies more with α than with resampling percentage (9.6 mean variance across α versus 7.8 across u).

In general, relevance-aware biased strategies are more stable, with a mean variance of 9.5 across parametrizations against the 13.1 of static biased strategies. Likewise, over-sampling strategies are about two times more stable than under-sampling strategies (15.4 versus 7.2 average variance, respectively).

For each strategy, the gradient patterns seem fairly consistent across learning algorithms. Still, MARS and RPART show lower variance than RF (8.3 and 8.7, respectively, versus 16.8) – perhaps partly because we repeated experiments with MARS and RPART ten times, while we ran experiments with RF only once.

Next, we analyze each parameter separately.

5.3.2.1 Parameters o / u

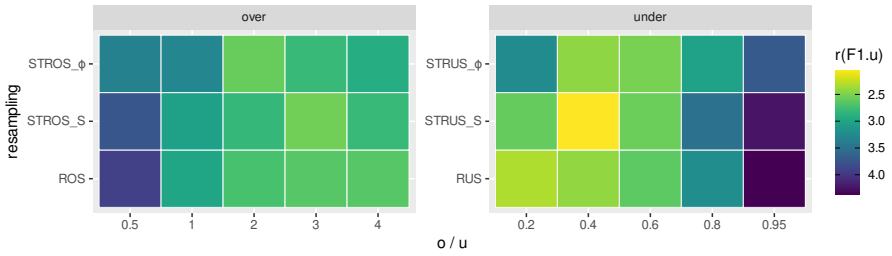


Figure 5.8: Average F_1^u rank obtained by each resampling method for 5 different values of o and u averaged over all data set, learning model, and resampling percentage values. Lower ranks correspond to better results

Figure 5.8 shows resampling percentages' average rank against each other at the same α value, averaged over all data sets and learning algorithms.

We can find somewhat of a "mirror" effect between under- and over-sampling, which can be explained by higher o and lower u values both resulting in less imbalanced distributions.

Static biased under-sampling (STRUS_S) is the most sensitive to resampling percentage at 0.76 mean variance across u, 2.7 times that of the second-most sensitive biased strategy (STRUS_ϕ) and similar to that of random under-sampling (RUS).

5.3.2.2 Parameter α

Only our biased strategies employ parameter α, which balances the temporal and spatial dimensions' influence during resampling. If $\alpha = 1$, they use a purely temporal weight; if $\alpha = 0$, they only consider the spatial dimension.

Figure 5.9 shows the biased resampling percentages' average rank against each other at the

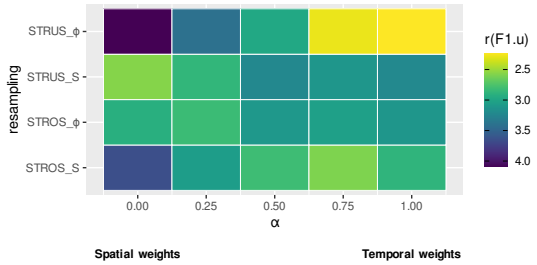


Figure 5.9: Average F_1^u rank obtained by each resampling method for 5 different values of α averaged over all data set, learning model, and resampling percentage values. Lower ranks correspond to better results

same resampling percentage, averaged over all data sets and learning algorithms.

Over-sampling strategies got their best results by combining spatial and temporal weights into a spatio-temporal weight. The static bias got its best results by favouring the temporal dimension; the relevance-aware bias, by favouring the spatial dimension.

Under-sampling strategies worked best by using "pure" weights: the static bias, using spatial weights; the relevance-aware bias, using temporal weights. Combining spatial and temporal weights resulted in their second-best ranks, as long as they favoured those same dimensions.

The rank patterns of STRUS_ϕ appear closer to those of STROS_S, while STRUS_S looks more similar to STROS_ϕ. Under-sampling and over-sampling show opposite tendencies when using the same bias. Whether under- or over-sampling, the static bias reversed the tendencies of the relevance-aware bias.

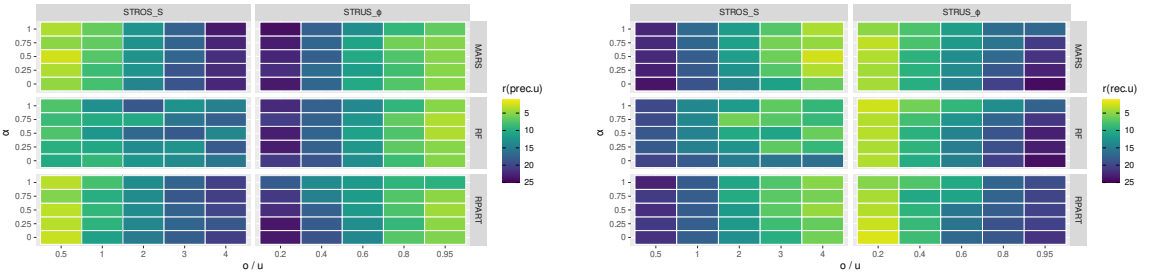
The impact of α is in sharper focus when using STRUS_ϕ. Its ranks varied the most with α, at almost four times the variance of the next most sensitive strategy (STROS_S).

5.3.2.3 Precision and Recall Trade-Off

The F_1^u-score combines utility-based precision ($prec_\phi^u$) and recall (rec_ϕ^u) into a single value – useful when we need to optimize a model or benchmark an approach to imbalanced regression. But a thorough analysis requires that we investigate the behaviour of each of the metrics that comprise it. We expect to find a trade-off between the two.

Figure 5.10 shows how different parametrizations of the best performers of each type, STRUS_ϕ and STROS_S, ranked against each other in terms of $prec_\phi^u$ and rec_ϕ^u across data sets. As expected, we found a trade-off between precision and recall. The figure also shows, once again, a mirroring effect between under- and over-sampling.

Across the x-axis, higher values of u and lower values of o led to lower precision and increased recall. These resampling percentages cause more significant shifts in the target distribution,



(a) $prec^u_\phi$ (b) rec^u_ϕ

Figure 5.10: Average utility-based precision and recall rank obtained by each resampling method for 25 different parametrizations, averaged over all data sets. Lower ranks correspond to better results

decreasing the imbalance and increasing the proportion of extreme values in the training sets. As expected, the higher proportion of extreme values increased the rate of normal cases predicted as highly relevant (with extreme values), lowering precision; the opposite reasoning explains the increase in recall.

Along the y-axis, the variation is more subtle but still present. STRUS_ϕ shows a clearer vertical gradient for recall than precision, suggesting that differences in recall drive α's strong effect on F^u_1-score found in previous sections.

The diagonal gradients show an interplay between parameters α and resampling percentages, and help explain the F^u_1-score rank patterns in Figure 5.7.

5.3.3 Impact of Data Set Characteristics

We now turn our attention to the impact of data set characteristics on our proposals' efficiency, following the analysis framework we outline in Section 5.2.4.

5.3.3.1 Internally Tuned Parameters

We considered one experiment a win in the meta-learning task at hand (the positive class) when a biased resampling strategy got better F^u_1-scores than its random counterpart on the same data set and using the same learning model, with internally tuned parameters.

Table 5.3 shows the average conditional permutation importance in Conditional Inference Forests (CIFs) (normalized to be between -1 and 1) of the top 10 most predictive meta-features, according to their average rank. The most determinant factor is whether under- or over-sampling was carried out. The next four most important factors describe the target distribution. The following three features relate to the locations distribution and the percentage of missing predictors. The remaining features, including the type of bias and

Table 5.3: Feature importance (normalized to be between -1 and 1) of top 10 meta-features as measured by average rank of conditional permutation importance applied 10 times to a meta data set, where the target variable indicates whether results are improved by introducing bias into a random resampling strategy, as measured by the average F_1^u calculated for each data set and learning algorithm pair, and using internally tuned parameters (parameters excluded as meta-features)

	Meta-feature	Avg. importance	Avg. rank	Std. dev. rank
1	type of resampling (under- or over-sampling)	1.00	1.00	0.00
2	extreme ratio at $t_R = 0.9$	0.56	2.50	0.53
3	target's inter-quartile range	0.54	2.50	0.53
4	percentage of target outliers according to boxplot statistics	0.29	4.10	0.32
5	target's Geary's kurtosis	0.15	5.70	0.95
6	0.05 quantile of normalized distance between locations	0.12	6.20	1.48
7	percentage of missing predictors	0.03	9.50	4.30
8	average normalized distance between locations	0.03	9.60	3.72
9	average correlation between target and predictors	0.00	11.90	7.43
10	0.95 quantile of standardized target	0.00	10.60	1.65

learning model, have zero or negative importance.

Feature importance reveals *what* factors determine if our proposals improve random resampling, but it cannot tell us *how* they influence that outcome. Figure 5.11 shows a conditional inference tree trained on the same task, which is more interpretable.

The type of resampling appears at the top node, consistent with its higher importance in Table 5.3, but bias appears next, even though it was near the the bottom in terms of importance. Half of the remaining nodes in the tree appeared on Table 5.3; the extra two included were the overall percentage of time-stamps and location pairs with available observations (Avail), and the average time series acceleration (accel_mean).

Relevance-aware under-sampling (STRUS_ϕ) worked better than random under-sampling (RUS) 95% of the time when the correlation between target and predictors was 0.4 or lower – the leaf with the highest percentage and absolute number of wins. Even at higher correlations, it still improved over RUS over 50% of the time.

Static biased over-sampling (STROS_S) worked better than random over-sampling (ROS) in 78% of cases where the inter-quartile range of normalized distance between locations was higher than 0.28, indicating varying levels of location density – corresponding to the leaf with the second-highest percentage and absolute number of wins. A narrower interquartile range almost halved the win percentage.

5.3.3.2 Parameters Setting

Next, we analyze the effect of data characteristics on the experiments without internal tuning, where the same parameters were used across the training sets. We compare the

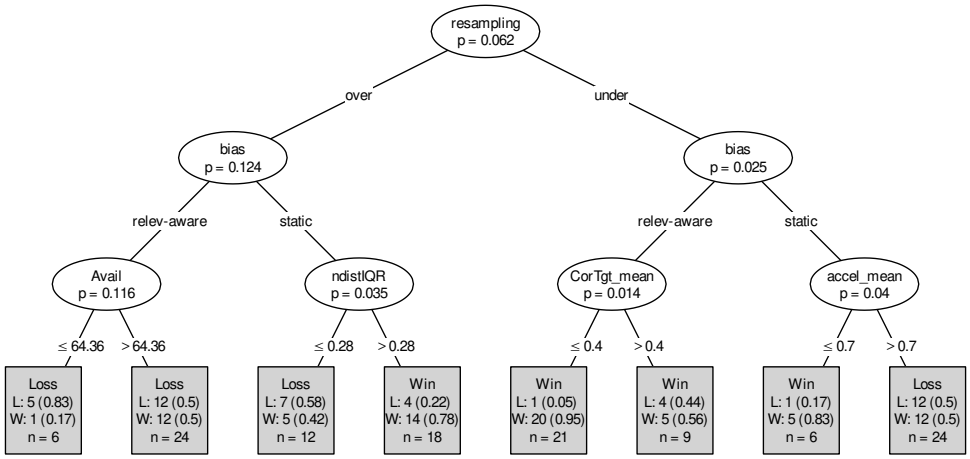


Figure 5.11: Meta-classification tree for internally tuned parameters. Each instance in the training set of a meta decision tree represents the outcome of introducing spatio-temporal bias to a random resampling strategy applied in conjunction with one of three learning algorithms to a data set: if the average F_1^u improves with the bias, the label is Win; otherwise, the label is Loss. Each condition in the tree is true for the left branch. Each node presents the majority class on top; the number of instances of each class, in the middle; and the percentage of meta data instances included in the node, below. The selected resampling parameters were not included since they differ across training blocks

biased strategies against their random counterparts at the same resampling percentage.

Table 5.4 shows the average conditional permutation importance in CIFs (normalized to be between -1 and 1) of the top 10 most predictive meta-features according to their average rank, when fixing resamping percentage. As expected, the most important feature is, once again, the type of resampling. But this feature is now followed by the types of bias and learning model, which were not present in Table 5.3. The average importance falls sharply after the third position, but it remains positive. Most of the remaining features relate to spatial autocorrelation and the spatial distribution of locations, and only three of the seven had previously appeared in Table 5.3. Ranks are much more stable in this table than when parameters were internally tuned (*cf.* Table 5.3).

Figure 5.12 shows a more interpretable conditional inference tree trained on this other task. The p-values in the tree indicate that all the splits were significant, unlike in Figure 5.11.

The first and second most important features according to Table 5.4 appear at the root and the next level of the tree. The features selected to classify under-sampling at the following level were also part of the top 10 most important features, but not the ones used to classify over-sampling – the number of outliers according to boxplot statistics and the average Hurst exponent.

Table 5.4: Feature importance (normalized to be between -1 and 1) of top 10 meta-features as measured by average rank of conditional permutation importance in a conditional inference forest applied 10 times to a meta data set, where the target variable indicates whether results are improved by introducing bias into a random resampling strategy at the same resampling percentage, as measured by the average F_1^u calculated for each data set and learning algorithm pair

	Meta-feature	Avg. importance	Avg. rank	Std. dev. rank
1	type of resampling (under- or over-sampling)	1.00	1.00	0.00
2	type of bias (static or relevance-aware)	0.24	2.00	0.00
3	model (MARS, RF, or RPART)	0.20	3.00	0.00
4	average significant observed Moran I index	0.03	4.00	0.00
5	average normalized distance to closest neighbour	0.02	5.40	0.52
6	average observed Moran I index	0.02	5.60	0.52
7	0.05 quantile of normalized distance between locations	0.02	7.30	0.48
8	percentage of positive observed Moran I indices	0.02	7.70	0.48
9	average correlation between target and predictors	0.01	9.20	0.42
10	average normalized distance between locations	0.01	10.40	1.51

STRUS_ϕ worked best when the average normalized distance to the closest neighbour was very low, beating its random counterpart 83% of the time – the highest win percentage; this number decreased to 65% of the time at higher average distance to the nearest neighbour.

STRUS_S worked better than RUS 60% of the time if the average statistically significant Moran I index was above 0.09, indicating spatial clustering of similar values – corresponding to the second highest absolute number of wins. If the index was below that value, indicating randomness or spatial clustering of dissimilar values, the results were almost reversed, as it lost to RUS 58% of the time.

5.3.4 Variability Analysis

Next, we evaluate the stability of our results from testing block to testing block on each data set and across repetitions.

Across testing blocks. Figure 5.13 shows how performance varies across testing blocks with internally tuned parameters. We could expect that performance would improve as the training window grew, perhaps later deteriorating if there was concept drift. We found instead that performance varied greatly in both directions for most data sets. Data set 10 was the only one to show the curve that we expected, though its performance on the first testing block was extremely low – perhaps because it was the smallest data set, one training block was insufficient to learn adequate models. Data set 10 had relatively consistent ratios of around 10% (corresponding to about 50) extreme cases. Higher variance in both relative and absolute number of extreme cases may help explain performance changes in the remaining data sets. When there were too few extreme cases, failing to capture even a single extreme

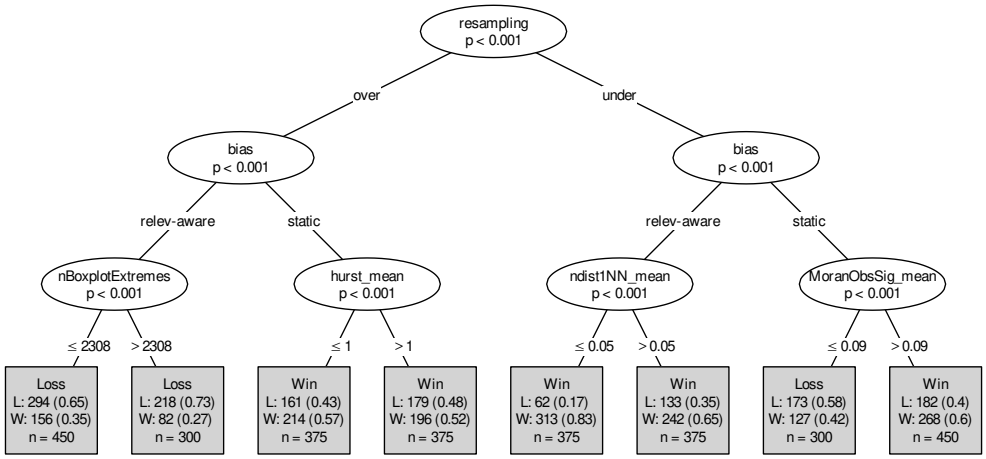


Figure 5.12: Meta-classification tree for pre-set parameters. Each instance in the training set of a meta decision tree represents the outcome of introducing spatio-temporal bias to a random resampling strategy, at a specific resampling percentage, applied in conjunction with one of three learning algorithms to a data set: if the average F_1^u improves with the bias, at that same resampling percentage, the label is Win; otherwise, the label is Loss. Each condition in the tree is true for the left branch. Each node presents the majority class on top; the number of instances of each class, in the middle; and the percentage of meta data instances included in the node, below

case could severely decrease recall, negatively impacting F_1^u.

Across repetitions. All the resampling approaches randomly selected cases to use in training, whether the process was biased or not. We repeated the experiments ten times to get more robust performance estimates, but we expected results to change across repetitions – especially when we internally tuned parameters according to the strategies' performance at each iteration.

Figure 5.14 shows the normalized standard deviation (σ_N) of the average F_1^u across the ten repetitions of the evaluation method, calculated for each data set, learning algorithm, and resampling approach triad.

Biased resampling methods vary less than their random counterparts – expected since introducing a bias should reduce randomness in the resampling process. Still, the normalized variance keeps within the same order of magnitude for every resampling method. The upper hinges of each box do not exceed 0.06, meaning that the standard deviation of F_1^u was most often below 6% of its average. The results kept reasonably stable but still varied due to the strategies' randomness, whether they included a spatio-temporal bias or not.

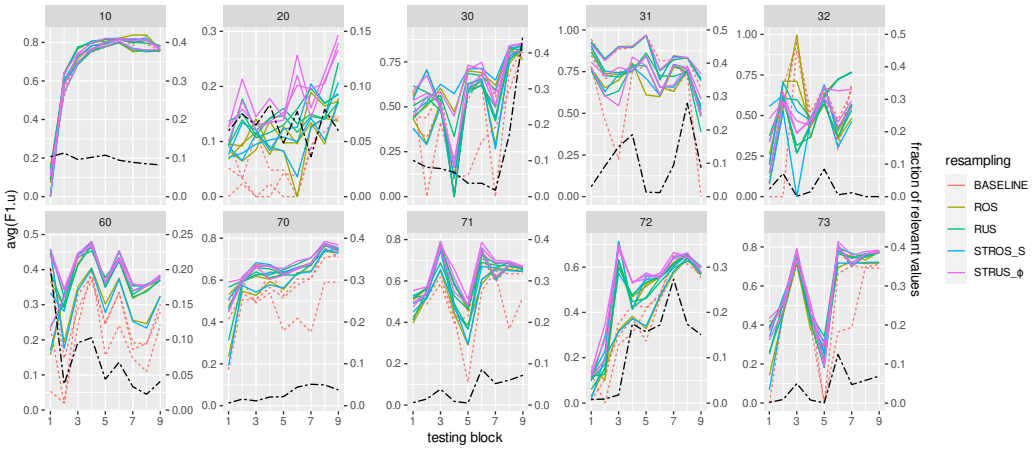


Figure 5.13: F_1^u variation across testing blocks, averaged over repetitions for each data set and learning algorithm. Dashed orange lines show the results without any resampling (one line for each of the three learning models); other colors are used for selected resampling methods (with internally tuned parameters). Dot-dashed black lines show the percentage of relevant cases in each testing block, on the secondary y-axis – note that cases in the testing block are considered relevant if their relevance is 0.9 or above, as defined by the relevance function calculated automatically from each training set

5.4 Discussion

Random resampling strategies address data imbalance issues simply. We injected a spatio-temporal sampling bias into random under- and over-sampling strategies to improve their performance in imbalanced spatio-temporal forecasting. We proposed two weighting schemes: a static, and a relevance-aware bias.

Related work on time series. Previous work had showed that leveraging temporal autocorrelation improved results in imbalanced time series [Moniz et al., 2017a]. We hypothesised and concluded that leveraging both temporal and spatial dependencies yields improvements on spatio-temporal forecasting.

Recommendations. We found that relevance-aware biased under-sampling (STRUS$_\phi$) performed best overall. It favoured the temporal dimension, meaning that it benefited from prioritising observations that preceded extreme cases recorded at a given location when selecting low-relevance cases to keep in the training set. Given that, when under-sampling, we are paring down the data set to a smaller size, it seems reasonable that preserving the dynamics that immediately precede an extreme case would be more important than clearing away the spatial region around an extreme observation. We found that the relevance-aware bias was more stable across parametrisations, but under-sampling was more sensitive to parametrisation than over-sampling.

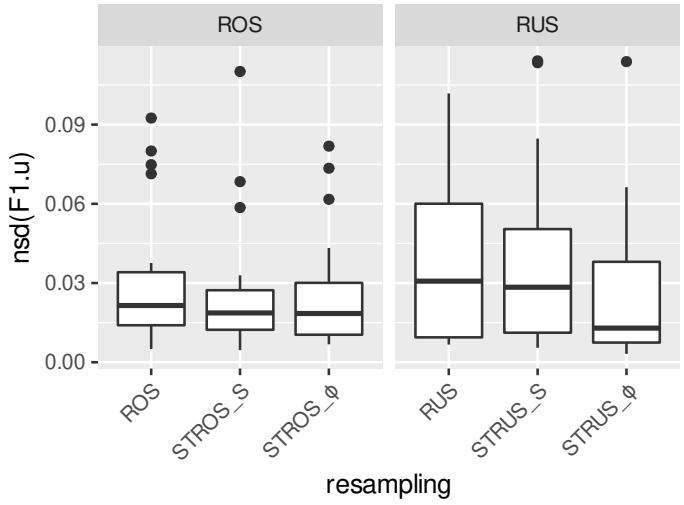


Figure 5.14: Box and whiskers plot of normalized standard deviations of average F_1^u across repetitions, calculated for each data set, resampling strategy, and learning algorithm triad (only MARS and RPART were included, as experiments were only executed once when using RF), using internally tuned parameters

When computational costs are a concern, but tuning parameters (resampling percentage and α) is still feasible, static biased under-sampling (STRUS_S) may be an appropriate choice. Static weights should be easier to compute while still achieving better results than random under-sampling or any type of over-sampling. In contrast with STRUS_ϕ, STRUS_S works best when favouring the spatial dimension instead: preserving observations at spatially isolated locations becomes more important than removing older observations.

If, despite our above considerations, the user still prefers over-sampling (e.g., because the data are too small to under-sample), then they should use the static bias (STROS_S) and favour the temporal dimension. STROS_S often performed worse than under-sampling, but it could be a reasonable choice as long as internal parameter tuning is feasible. When selecting relevant cases to replicate and add to a data set, we should favour observations recorded most recently and at spatially isolated locations – the temporal component once again being more valuable. Given that there are fewer highly relevant cases to take advantage of in the over-sampling process, it is perhaps understandable that favouring more recent cases would be more important than keeping those at spatially isolated locations. Being spatially isolated does not guarantee that the process that led to an extreme value could not be partially explained by dynamics also present in other geographic regions.

Caveats. Though these are our general recommendations for the usage of biased re-sampling strategies (summarised in Figure 5.15), we should acknowledge that there are exceptions to our considerations, and the best strategy and parametrisation for a specific

problem may still depend on the characteristics of the application domain, as well as the chosen learning algorithm. For some data sets, depending on parametrisation, under-sampling with different types of bias or over-sampling strategies achieved the best average results. In fact, when using the MARS learning algorithm, over-sampling with our original (mixed) proposal for bias (STROS) achieved the best overall average rank of F_1^u-score with fixed parameters and second-best with internally tuned or optimal parameters.

Strengths and limitations. The two types of bias, and the two types of resampling show contrasting behaviours in terms of their preferred dimension. As we mentioned, STROS_S and $STRUS_\phi$ worked best when they favoured the temporal dimension, while STRUS_S and STRUS_ϕ. While not entirely predictable, these contrasting patterns point to an interesting interplay between resampling approaches, and spatial and temporal dependency relationships.

We cannot exclude the possibility that the apparent benefit of favouring the temporal dimension, i.e., setting α above 0.5, in the best performers among under- and over-sampling biased strategies that we discussed in the previous paragraphs could be driven by underlying characteristics in most of our real-world data sets – e.g., the (irregular) spatial distribution of the comparatively smaller number of locations in our data sets could be such that spatial autocorrelation was not meaningful enough to carry as much weight when compared to temporal autocorrelation across a much higher number of time-stamps. For these reasons, ideally, internal parameter tuning should be carried out to achieve the best possible results in each application.

To provide a deeper understanding of these issues, we conducted a multidimensional meta-analysis of our results to assess how data set characteristics influence whether introducing bias is beneficial. When we compared strategies using the same resampling percentage, the learning algorithm, the type of resampling strategy and bias were quite influential. If we compared strategies after internal parameter tuning, the type of bias became much less impactful, suggesting that our proposals for spatio-temporal bias may have their merits under different circumstances when properly tuned.

Our results suggest some features affected performance more than other characteristics such as data size, including the distance between locations, correlation between the target and spatio-temporal indicators used as predictors, and the prevalence of missing data. We would expect characteristics such as data set size to influence the outcome of an imbalanced regression problem more. However, our analysis focuses on the differences between addressing the imbalance while accounting for spatio-temporal dependencies (by introducing a resampling bias) versus not considering these relationships while still addressing the imbalance issue (by using random resampling). Thus, these results seem to reinforce that our proposals for spatio-temporal bias are indeed primarily addressing the issues that arise from the spatio-

temporal nature of the data while perhaps compensating for the disruptions caused by missing data.

Another essential issue to consider is the definition of what constitutes a highly relevant case. We assumed all events followed the same distribution, applying a general concept of relevance instead of considering spatial or temporal locality. We recognise that there may be issues with either configuration.

Depending on the application, a general notion of relevance may be the best choice. For example, health risks posed by high levels of pollution affect people regardless of local variables. In other situations, extreme values might need to be determined locally to be helpful. For example, an extremely high influx of customers at a specific bike rental station should be compared to normal consumer behaviour in the neighbourhood at similar periods of the year, not against a station with a much higher or lower average number of customers. Future work may compare our results with experiments that derive relevance functions specific to each location.

Although we saw little to no benefit from the addition of Gaussian noise to highly relevant replicas in this study, we recognise that this may be, in part, due to our chosen amount of noise. A deeper study varying this parameter is needed to investigate the potential benefit of different amounts of noise.

Future work might also look into adapting to the spatio-temporal context techniques such as Synthetic Minority Oversampling Technique (SMOTE) for regression [Torgo et al., 2013] which do not simply replicate or add noise to the already existing highly relevant data, but generate synthetic observations in a more sophisticated manner.

5.5 Summary

We addressed the problem of imbalanced spatio-temporal, numerical forecasting. Spikes in air pollution and other extreme values of high societal impact arise from spatio-temporal processes. We proposed a set of biased resampling strategies that improve the performance of widely used random under-sampling and over-sampling by leveraging the implicit dependencies in spatio-temporal data. We explored two ways of calculating spatial and temporal bias: a static and a relevance-aware weighting scheme. The weights combine into a spatio-temporal bias, which can weigh the spatial and temporal dimensions differently. We found that relevance-aware biased under-sampling performed best, on average. If over-sampling, a static bias would be the better choice as long as internal tuning of hyper-parameters is possible.

Relevance-aware biased under-sampling worked best when favouring the temporal dimension, while static biased under-sampling excelled when favouring the spatial dimension; the

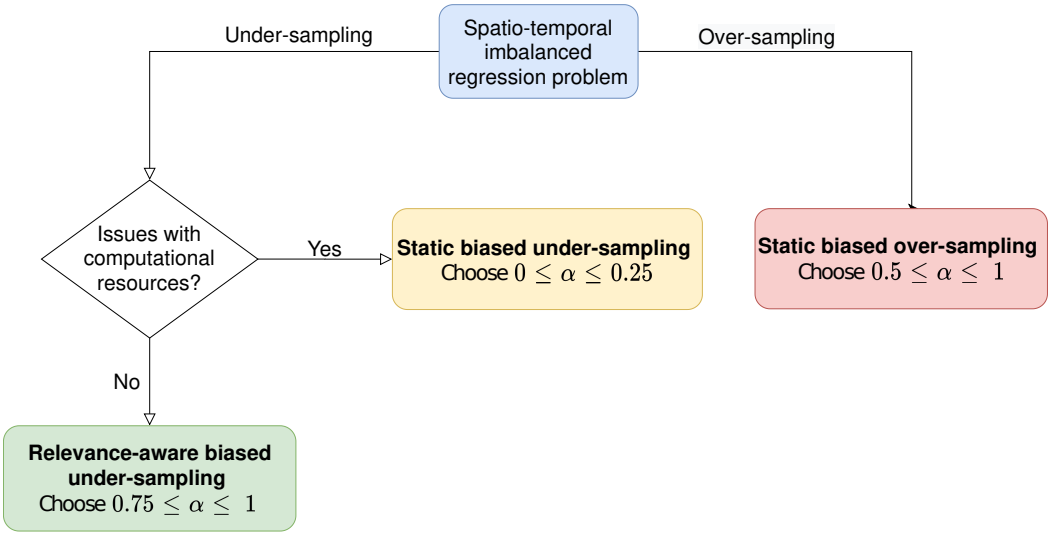


Figure 5.15: General guidelines for the application of spatio-temporal biased resampling approaches. Parameter α regulates the relative importance of the spatial and temporal dimensions in the spatio-temporal bias ($\alpha = 0$ means the bias is spatial-only; $\alpha = 1$ means the bias is temporal-only). The under-sampling approaches generally perform better than over-sampling. These guidelines are based on average results; the best strategy and parameters for a specific problem may vary, depending on characteristics of the application domain and learning algorithm. When possible, parameters should be tuned to achieve improved results for a specific task

opposite holds true for biased over-sampling. Our findings suggest that domain features directly related to spatial dependency relationships in the data influence the capacity of our proposed strategies more than other underlying data set characteristics, such as data set size at the same resampling percentage. If parameters are tuned, then the percentage of missing data and the characteristics of the target distribution affected the results more.

A thorough experimental evaluation supports our claims that our proposed spatio-temporal biased resampling strategies boost the predictive performance of extreme values in real-world spatio-temporal data.

Chapter 6

Conclusions

Spatio-temporal data can describe phenomena on a wide range of application domains, from wind energy production to air quality monitoring. Geo-referenced time series have become widely available as sensors collect more and more data along time, at fixed locations across the globe. These data are often used to prepare for rare but catastrophic events, nature or human caused phenomena that can severely impact human communities. The vast amounts of data make it intractable to human experts, but Predictive Analytics (PA) can guide their decision-making by providing an accurate picture of the future.

This thesis aims at addressing the challenges posed by spatial and temporal dependencies in geo-referenced time series forecasting. We set out to contribute to an experimental-based methodology that can aid geo-temporal PA. We (1) provide guidelines for performance estimation, and, using the corresponding results, (2) propose PA methods that leverage the dependencies to improve prediction of "normal" and extreme numeric values.

Whether we face imbalanced domains or not, we can only improve upon existing spatio-temporal forecasting approaches if we know how to properly assess their performance. The dependencies in spatio-temporal data pose challenges when estimating model performance, causing standard methods like standard cross-validation to be over-optimistic. Our results show that careful fold design is the key to obtaining accurate estimates. Given such a design, we show that we can improve the prediction of extreme values through a biased resampling function that takes into account the spatio-temporal nature of the problem.

This final chapter concludes our work by summarising its main contributions, and discussing possible future research directions.

6.1 Main Contributions

In this thesis, we (1) empirically studied performance estimation methods to make recommendations for the evaluation of spatio-temporal forecasting; (2) proposed and tested pre-processing strategies that aimed at (a) reducing forecasting error by extracting features that incorporate spatio-temporal contextual information, and (b) improving extreme value prediction by considering the observations' spatio-temporal context during resampling; (3) studied how domain-specific data properties impacted our proposals' efficiency; and (4) developed open-source software implementing our solutions, and made it freely available online. Next, we detail each of these contributions.

Guidelines for performance estimation. We conducted an empirical comparative study of over fifteen performance estimation methods, including variants designed or adapted to specifically address the data's spatio-temporal nature (Chapter 3).

Through comparing the methods' estimated error against a "gold standard" error on previously withheld data, we found only slight differences between methods on stationary synthetic data, but more distinct behaviours on real-world data.

We recommend that practitioners avoid standard CV in real-world applications as we confirmed it to be over-optimistic in its estimates, often severely under-estimating error. CV variants could be a suitable alternative as long as they block data in time to mitigate this issue, but OOS methods, which inherently respect temporal order, provide competitive estimates when compared to them.

While we do not recommend a specific method over all others, we selected temporal block prequential evaluation with a growing window to evaluate our proposals in Chapters 4 and 5. This OOS method appears frequently in the literature, and seemed particularly suited to our experimental design as it prevented information from spilling over between testing and training even when we extracted features that summarise historical data from neighbouring locations. To tune parameters internally in Chapter 5, we chose temporal block CV because it works similarly, but it uses the sometimes small training sets more efficiently.

Proposal for extracting spatio-temporal indicators. We proposed a pre-processing strategy based on previous work by Ohashi and Torgo [2012] that extracts features from univariate geo-referenced time series data so that any off-the-shelf regression algorithm can be trained to predict future values (Chapter 4).

The original proposal extracts features that summarise values within a conic spatio-temporal neighbourhood that narrows its spatial radius as it incorporates older historical data. We reversed the direction of the original proposal's conic spatio-temporal neighbourhood, under

the assumption that it could better capture the dynamics of spatio-temporal phenomena that take time to "travel" between locations in some applications.

Our experiments revealed that, on average, the original proposal outranked our variant. However, given the right parametrization, our variant showed potential, performing best for about half the data and learning model combinations.

New biased resampling strategies for imbalanced domains. We proposed new strategies that bias the typically random under-sampling or over-sampling processes by considering the observations' spatio-temporal context (Chapter 5). We proposed two ways of calculating spatial and temporal bias weights: a static scheme, and a relevance-aware weighting scheme.

- **Static bias** This scheme attributes a single weight to each location, and to each time-stamp. It prioritises recent observations at more isolated locations to keep or replicate in a training set, depending on whether we are under-sampling or over-sampling;
- **Relevance-aware bias** This scheme attributes dynamic weights to the observations, depending on how relevant they are when compared to observations at neighbouring locations at each time-stamp, or to previous and subsequent observations at each location.

The temporal and spatial weights combine into a spatio-temporal bias that can balance the the spatial and temporal contributions differently.

A thorough experimental evaluation showed that introducing a spatio-temporal bias improved prediction of extreme cases in real-world applications. We found that the two bias schemes, and the two resampling strategies reached their best results by favouring opposite dimensions.

In general, we recommend that practitioners apply relevance-aware biased under-sampling (STRUS_{ϕ}), attributing more weight to the temporal dimension. If hampered by computational costs, static bias under-sampling (STRUS_{S}) would be the second best choice, favouring the spatial dimension. If other reasons make over-sampling preferable, then we recommend using a static bias (STROS_{S}) and tuning hyper-parameters.

Impact of domain-specific data properties. We carried out meta-analyses investigating how domain-specific data characteristics influence the efficacy of our feature engineering (Chapter 4) and resampling proposals (Chapter 5).

We found that, with optimal parameters, our proposal for feature extraction worked best on smaller data sets, if using linear regression to learn the data model. Besides data set

size, other impactful data properties included the average distance between locations, the number of outliers in the data, and, to a lesser degree, temporal autocorrelation coefficients.

In contrast, when evaluating our biased resampling strategies against their random counterparts at the same resampling percentages, data properties like data set size appeared to affect outcomes less than those directly related to spatial dependency relationships. With tuned parameters, the percentage of missing data and the target distribution's properties affected the results more.

Free and open-source software. We developed two *R* packages, *STEvaluation*[1] and *STResampling*[2], that include ready-to-use implementations of the feature extraction methods used in the thesis, the performance estimation methods, and our biased resampling strategies. We made both packages freely available online.

Final remarks. The implicit dependencies between observations in geo-referenced time series present challenges when forecasting future values, but they also represent opportunities to improve results by leveraging spatio-temporal contextual information. After we determined how to properly estimate forecasting performance, we proposed a feature extraction method to capture the dynamics of some spatio-temporal phenomena, and biased resampling strategies that improved the prediction of rare and extreme values.

6.2 Open Issues and Future Directions

As with any empirical research work, ours has some limitations. Future work can help clarify some of the issues raised, and explore new avenues to improve forecasting and its performance estimation. We list a few potentially interesting research directions.

Evaluating under irregularity and imbalance. Irregularities and imbalances in spatio-temporal data can affect performance estimation methods that rely on grouping data in blocks along the temporal, spatial or both dimensions.

Authors working on spatial data have addressed data irregularity by distributing consistently shaped blocks irregularly, or using differently shaped blocks [Roberts et al., 2017]. Adapting these solutions to work with spatio-temporal data would be challenging, but interesting.

In the absence of strong data autocorrelation, stratified CV is often used to balance the target or predictor variables' distributions if they present imbalances. It is unclear how to achieve balanced training and test sets while avoiding the dependencies between training and

[1] https://github.com/mrfoliveira/STEvaluation-MDPI2021
[2] https://github.com/mrfoliveira/STResampling-JDSA2020

test sets that cause over-optimistic error estimates in geo-referenced time series. Possible solutions like buffered leave-one-out method can be prohibitively computationally expensive, so this is still an open issue.

Theoretical analysis of performance estimation methods. Arlot and Celisse [2010] reviewed a few theoretical studies covering CV with temporal dependence, and we could find several empirical studies of performance estimation on time series [Bergmeir and Benítez, 2012, Bergmeir et al., 2014, Cerqueira et al., 2017, Mozetič et al., 2018]; some issues are still contested in the field. To the best of our knowledge, ours was the first empirical study of performance estimation on geo-referenced time series. A more robust theoretical foundation accounting for simultaneous spatial and temporal autocorrelation could provide more theoretical guarantees and might help guide future empirical work.

Complementary feature extraction. We framed the original proposal for spatio-temporal neighbourhoods by Ohashi and Torgo [2012] as a competitor to our variant that reversed its directions, but they need not be exclusive. Future work could explore whether the two cones could complement each other instead, improving performance by capturing phenomena with different dynamics.

Global vs. local relevance. In Chapter 5, we used a global definition of case relevance, assuming an even distribution across time and space. Depending on the application, this may be the better choice since some effects follow from specific values regardless of context, e.g., water quality conditions leading to algae blooms. Other applications may benefit from considering spatial or temporal locality when calculating relevance, e.g., peaks and troughs in dam water levels during rainy seasons should probably be compared against similar time periods in previous years. Future work could derive relevance functions dynamically along time, or calculate them for each location.

Handling missing data. Throughout the thesis, we handled the missing values in some of our real-world data in the same, simple way. As we mentioned when we first explained our methodology to deal with missing data, other more sophisticated methods exist. It could even be possible to apply methods that leverage the spatial and temporal dependencies in the data to interpolate missing values just as we leveraged them to forecast future values.

Handling seasonality, lack of stationarity, and other heterogeneities. Seasonality is a common feature in many spatio-temporal applications, e.g., weather can vary greatly over the year; data also often contain non-stationarities or otherwise heterogeneous patterns, e.g., soil properties can abruptly change over very short distances. Our proposals worked

well on real-world data, but future work could extend them to explicitly deal with these challenging data properties.

Exploring hyper-parameter spaces and generalisability. We were as exhaustive as possible in our experiments, but expanding hyper-parameter search spaces, and applying our proposals to additional data could provide new insights and further test their generalisability.

For example, in Chapter 3, we followed an experimental design similar to Bergmeir and Benítez [2012], but synthetic experiments akin to the work of Roberts et al. on spatial data could bring forth new insights. Future work on performance estimation could also study the effects of different block and buffer sizes, number of partitions or repetitions; and delve deeper into the effects of missing data, non-stationarities and other data properties.

We analyzed parameter sensitivity on Chapters 4 and 5, but some space for improvement remains. For example, adding our chosen amount of Gaussian noise to highly relevant replicas did not improve results over regular or biased over-sampling, but potential benefits could arise from varying this parameter.

Final remarks. We believe this thesis contributed to advance the field of Predictive Analytics for spatio-temporal data. We investigated the methods we use to evaluate PA approaches, and made some recommendations for how new proposals should be assessed in the future. We also proposed our own PA methods that can improve prediction, even under imbalanced domains. Many questions still remain to be explored, and we hope that future research can build upon the work presented here to expand our knowledge further.

Appendices

Appendix A

Data Sets

A.1 Artificial Data Sets

A.1.1 Stationarity Conditions

Stationarity, meaning that the covariance structure of $\mathbf{z}(t)$ in a STARMA model does not change with time, requires that every x_u that solves Equation A.1 lies inside the unit circle ($|x_u| < 1$).

$$det\left[x_u^q \mathbf{I} - \sum_{k=1}^{q}\sum_{i=0}^{m_k} \theta_{ki}\mathbf{W}^{(i)} x_u^{q-k}\right] = 0 \tag{A.1}$$

Low-order STARMA Stationarity

A STARMA(2_{11}) is defined by the following equation:

$$\begin{aligned}
\boldsymbol{z}(t) &= (\phi_{10} I + \phi_{11}\boldsymbol{W^{(l)}})\boldsymbol{z}(t-1) && \text{(A.2)}\\
&+ (\phi_{10} I + \phi_{21}\boldsymbol{W^{l}})\boldsymbol{z}(t-2) + \boldsymbol{\epsilon(t)} && \text{(A.3)}\\
&+ (\theta_{10} I + \theta_{11}\boldsymbol{W^{(l)}})\boldsymbol{\epsilon}(t-1) && \text{(A.4)}\\
&+ (\theta_{10} I + \theta_{21}\boldsymbol{W^{(l)}})\boldsymbol{\epsilon}(t-2) + \boldsymbol{\epsilon(t)} && \text{(A.5)}
\end{aligned}$$

Stationarity restrictions for STARMA(2_{11}) models can be written as below for the AR component (ϕ_{kl} coefficients) [Pteifer and Deutsch, 1980].

Table A.1: Model coefficients, c_{XY} corresponding to ϕ_{XY} and/or θ_{XY}. Coefficients are fixed or generated within the presented intervals.

Model order	c_{10}	c_{11}	c_{20}	c_{21}
2_{10}	$[-2, 2]$	$[-2, 2]$	$[-1, 1]$	0
2_{01}	$[-2, 2]$	0	$[-1, 1]$	$[-2, 1]$
2_{11}	$[-1.227, 0.733]$	$[0.733, 1.277]$	$[-0.227, 1.773]$	-0.7333
2_{11}	$[-1.755, 0.245]$	$[-1.755, 1.755]$	$[-0.7555, 0.7555]$	0.245

$$
\begin{aligned}
-\phi_{20} + |\phi_{21}| &< 1 \\
|\phi_{10} + \phi_{11}| &< 1 - \phi_{20} - \phi_{21} \\
|\phi_{10} - \phi_{11}| &< 1 - \phi_{20} + \phi_{21}
\end{aligned}
$$

The same set of restrictions apply to the MA terms (θ_{kl}).

A.1.2 Random Coefficient Generation

Coefficients are generally randomly generated within intervals that present reasonable chance of respecting stationarity conditions. In the case of order 2_{11}, one of the coefficients is fixed at a random value first and the remaining three coefficients are generated within intervals informed by this first selection (*cf.* Table A.1). The non-linear function f used to obtain NLSTAR models was randomly selected between $\sin(x)$, $\cos(x)$, $\arctan(x)$, $\tanh(x)$ and $\exp(-\frac{x}{C})$, with $C = 1 \times 10^4$.

A.2 Real-world Data Sets

Figure A.1 shows the spatial distribution of our real-world data sets across the globe.

A.2.1 Spatio-temporal Indicators

We calculated the mean, standard deviation, and weighted mean within each of three spatio-temporal distance boundaries, and the ratios between the mean and weighted mean within neighbourhoods of increasing size. Spatio-temporal distance is a linear combination of temporal and spatial distances, weighted by parameter α (see Equation 4.1).

We set the normalized distance boundaries β_i and the parameter α so that most neighbourhoods were not empty: $\beta \in \{0.0250,\ 0.0375,\ 0.0500\}$; $\alpha = 0.25$ for most data ($\alpha = 0.5$ for NCDC and SAC data, which have a number of unique time-stamps below double the

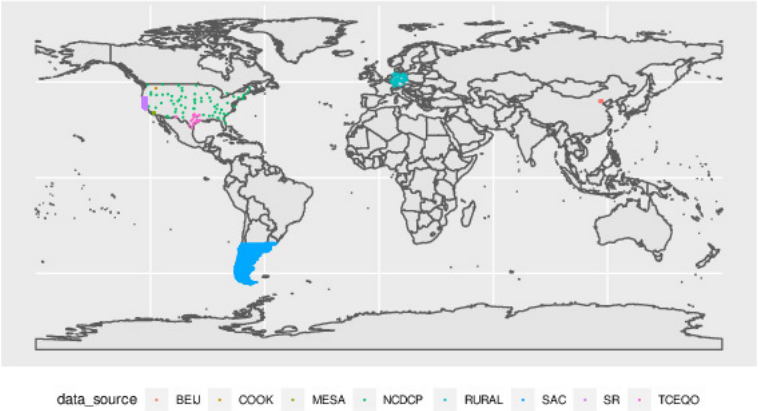


Figure A.1: Global distribution of locations included in each data source.

number of locations). We set the temporal embed size to 7.

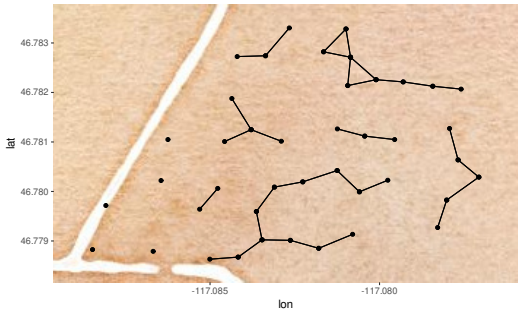
Table A.2: Geo-spatial characteristics of spatio-temporal neighbourhoods. Parameter α and distances between stations (in kilometers), spatial radius of each spatio-temporal neighbourhood at the same time-stamp as the target observation, $r_i = \beta_i/\alpha$, $\beta \in \{0.0250,\ 0.0375,\ 0.0500\}$, average number of neighbours within each radius, and total number of stations.

ID	Data	α	Min. dist.	Max. dist.	r_1	r_2	r_3	AvgNN_1	AvgNN_2	AvgNN_3	N
1	MESA	0.25	0.08	65.11	6.51	9.77	13.02	0.9	1.6	2.4	20
2	NCDC	0.5	1.43	4426.17	221.31	331.96	442.62	1.0	2.3	4.1	72
3	TCE	0.25	4.38	1214.35	121.43	182.15	242.87	2.5	4.6	6.3	26
4	Cook	0.25	0.06	0.90	0.09	0.14	0.18	1.5	4.2	7.4	42
5	SAC	0.5	31.40	2693.96	134.70	202.05	269.40	21.8	46.5	77.2	900
6	airBase	0.25	3.80	814.53	81.45	122.18	162.91	3.5	6.6	12.2	70
7	Bejing	0.25	1.62	128.28	12.83	19.24	25.66	4.9	7.9	11.6	36

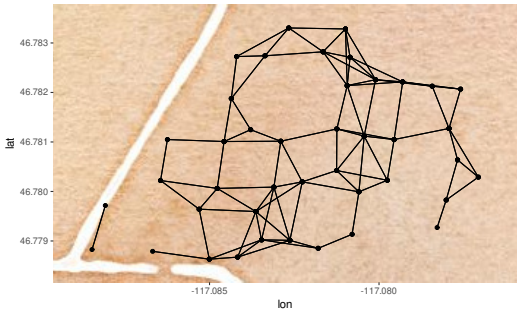
Table A.2 characterises the spatial distribution of data and the defined spatio-temporal neighbourhoods when $\beta \in \{0.0250,\ 0.0375,\ 0.0500\}$, including minimum and maximum distances between locations, maximum spatial radius of each spatio-temporal neighbourhood (at the target time-stamp), and the average number of neighbour locations within them.

Figure A.2 shows neighbours connected by lines within the boundaries defined for data Cook Agronomy Farm (at the target time-stamp).

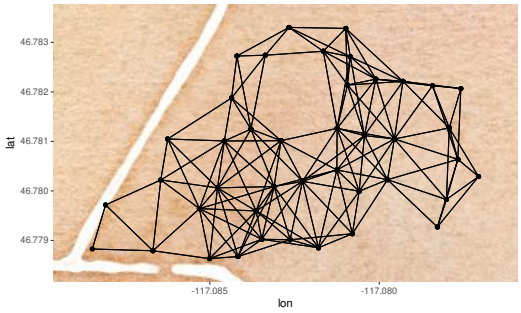
Even though spatio-temporal indicators were calculated using only past data, which excludes observations measured at the same time as the target observation, this is still illustrative of the spatial extent of the spatio-temporal neighbourhoods.



(a) $\beta = 0.025$



(b) $\beta = 0.0375$



(c) $\beta = 0.05$

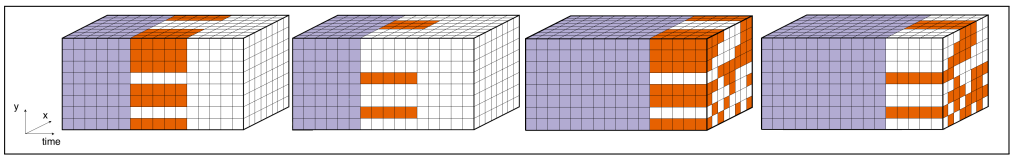
Figure A.2: Spatial neighbours at maximum spatial radius (shown connected by lines) within each spatio-temporal neighbourhood with $\alpha = 0.25$ and different values of β for data set Cook Agronomy Farm.

Appendix B

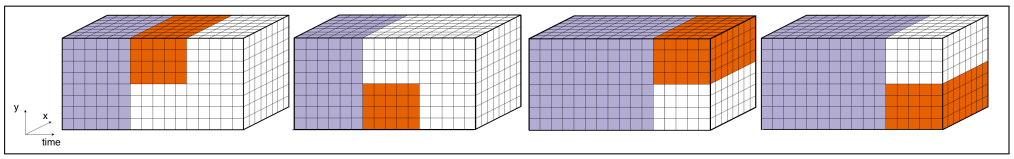
Spatio-temporal Evaluation Methods: Supplementary Results

B.1 Estimation Methods

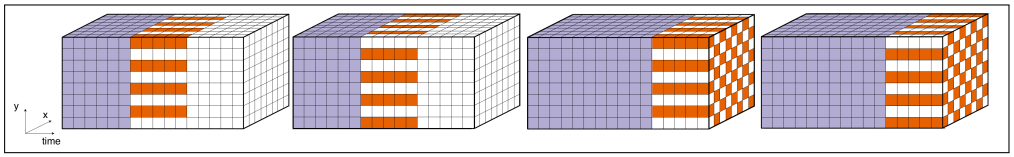
Figure B.1 shows some additional prequential methods with growing window.



(a) Spatio-temporal block prequential evaluation (Preq-STb)



(b) Spatio-temporal contiguous block prequential evaluation (Peq-STb-cont)



(c) Spatio-temporal systematic block prequential evaluation (Preq-STb-sys)

Figure B.1: Additional prequential evaluation methods with growing window. Blocks of data used for training in lighter lilac; blocks of data used for testing in dark orange. Time flows left to right

B.2 Empirical Results

Table B.1: Average ranks of absolute errors, calculated separately for cross-validation and out-of-sample methods when estimating performance on 17 real-world data sets. Best results in bold

Type	Method	MARS	LM	RF	RPART	Overall
CV	CV	**3.41**	**3.47**	6.24	**3.76**	**4.22**
	CV-Sbuf	4.25	4.44	**4.19**	4.06	4.23
	CV-Tsl	4.76	4.65	4.35	4.76	4.63
	CV-Tb	5.53	5.88	4.41	6.18	5.50
	CV-Tb-Tbuf	6.24	6.06	5.12	5.94	5.84
	CV-Sb	3.94	4.12	5.76	4.00	4.46
	CV-Sb-Sbuf	6.44	5.81	4.88	5.19	5.58
	CV-STb	4.59	4.47	4.53	5.00	4.65
	CV-STb-mSTbuf	5.47	5.71	5.06	5.65	5.47
OOS	HO-80-20	**2.59**	**2.59**	**2.94**	3.12	**2.81**
	HO-89-11	3.00	3.00	**2.94**	**2.47**	2.85
	MC-44-6	4.06	4.06	3.65	4.24	4.00
	MC-53-7	3.76	4.12	3.88	3.88	3.91
	Preq-Tb-grW	4.35	4.29	4.18	4.47	4.32
	Preq-STb-grW	3.24	2.94	3.41	2.82	3.10

Tables B.2 and B.1 show detailed results ranking all tested CV and OOS evaluation methods separately, when using each of four learning algorithms and overall.

B.2.1 Statistical Significance per Learning Model

Figures B.2 and B.3 show the results of additional statistical testing for a subset of estimation methods.

Table B.2: Average ranks of absolute errors, calculated separately for cross-validation and out-of-sample methods when estimating performance on 192 artificial data sets. Best results in bold

Type	Method	MARS	LM	RF	RPART	Overall
CV	CV	8.98	9.01	8.48	8.98	8.87
	CV-Tbuf	9.85	9.63	10.17	9.62	9.82
	CV-Sbuf	9.48	9.22	9.29	9.53	9.38
	CV-STbuf	10.32	9.89	10.83	10.08	10.28
	CV-Tsl	8.30	8.83	8.38	8.29	8.45
	CV-Tb	8.90	9.00	8.31	8.94	8.79
	CV-Tb-Tbuf	9.18	8.80	8.79	8.45	8.80
	CV-Sb	8.38	**8.20**	8.56	8.82	8.49
	CV-Sb-cont	8.51	8.66	**8.17**	**8.28**	**8.40**
	CV-Sb-sys	8.56	8.92	8.44	8.64	8.64
	CV-Sb-Sbuf	9.03	8.46	9.15	9.41	9.01
	CV-Sb-cont-Sbuf	9.03	8.85	9.07	8.62	8.89
	CV-STb	8.56	9.05	8.71	9.13	8.86
	CV-STb-cont	8.95	8.91	8.93	8.56	8.84
	CV-STb-sys	**8.19**	8.64	8.37	8.99	8.55
	CV-STb-Tbuf	8.63	9.07	8.65	8.88	8.81
	CV-STb-mSTbuf	10.15	9.86	10.71	9.77	10.12
OOS	HO-80-20	4.51	4.59	4.35	4.82	4.57
	HO-94-6	5.58	5.73	5.12	5.24	5.42
	MC-47-3	4.45	4.32	4.68	4.53	4.50
	MC-55-4	4.62	4.67	4.29	4.49	4.52
	Preq-Tb-grW	**4.19**	**3.93**	4.75	**4.16**	4.26
	Preq-STb-grW	4.20	4.27	4.30	4.40	4.29
	Preq-STb-cont-grW	4.24	4.30	4.35	4.19	4.27
	Preq-STb-sys-grW	4.21	4.19	**4.15**	4.18	**4.18**

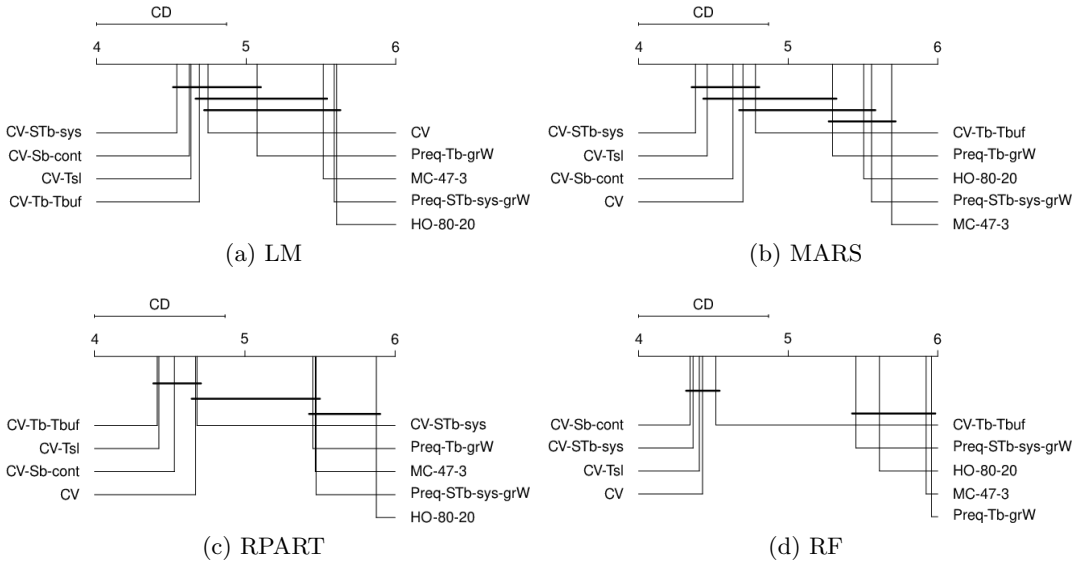
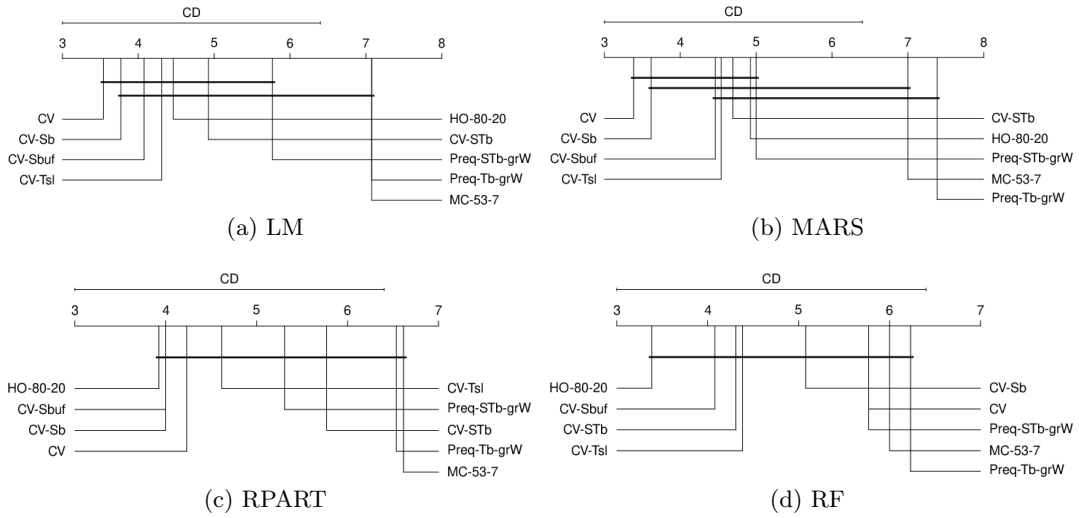


Figure B.2: Critical difference diagram according to Friedman-Nemenyi test (at 5% confidence level) for a subset of estimation methods using 192 artificial data sets



(a) LM (b) MARS

(c) RPART (d) RF

Figure B.3: Critical difference diagram according to Friedman-Nemenyi test (at 5% confidence level) for a subset of estimation methods using real data sets

Appendix C

Biased Resampling Strategies: Supplementary Results

We present more detailed results for all our proposed biased resampling strategies, including strategies using our original mixed proposal for spatio-temporal bias, and the variants of over-sampling with added Gaussian noise.

C.1 Results aggregated per model/data

C.1.1 Internally Tuning Parameters

This subsection describes the results of prequential time-block evaluation with internal parameter tuning using time-block cross-validation.

For each learning model and data set pair, we rank the different strategies according to F_1^u. Table C.1 shows the average ranks of each sampling method aggregated by model. We can see that our relevance-aware biased under-sampling proposal (STRUS_ϕ) is the best performer regardless of model used. Only when using RF, does random over-sampling (ROS) out-perform all types of biased over-sampling; for all other types of resampling and learning models, there is at least one type of bias whose introduction improves the random version of resampling.

Table C.2 shows the breakdown by data set. The baseline always ranks bottom for 4 of the data sets; for the remaining data sets, any type of resampling (biased or not) still improves over the baseline on average, with a few exceptions – five approaches perform worse than no resampling on data set 32; two strategies perform worse than (and one ties with) no re-sampling on data set 31; one non-biased random resampling approach performs worse than none on data set 51. The best results are always obtained by using some type of biased

Table C.1: Average ranks of F_1^u obtained by each resampling method for each learning model, across data sets, with internally tuned parameters. Best result for each learning model is in bold; best result per type of resampling strategy is in italics

type	MARS	RF	RPART
None	9.80	12.10	12.70
GAUSS	9.90	8.90	9.00
STGAUSS	9.40	*7.20*	7.00
STGAUSS$_S$	*9.30*	7.60	*6.60*
STGAUSS$_\phi$	11.30	7.80	10.90
ROS	6.50	*7.40*	7.00
STROS	*4.20*	8.70	7.00
STROS$_S$	4.80	8.50	*5.00*
STROS$_\phi$	5.90	9.30	8.10
RUS	5.20	3.50	6.60
STRUS	4.70	3.50	5.20
STRUS$_S$	5.90	3.40	4.40
STRUS$_\phi$	**4.10**	**3.10**	**1.50**

under-sampling: STRUS$_\phi$ achieves the best rank overall in 7 of the 10 data sets; STRUS$_S$ works best for data sets 10 and 31; STRUS works best for data set 32. When over-sampling with or without Gaussian noise, there is a type of bias that improves the random version for 7 out of 10 data sets in each case; static bias is tied with our original proposal in terms of which type of bias works best for a greater number of data sets if no noise is added; but our original proposal works best for more data sets if using Gaussian noise.

Table C.2: Average ranks of F_1^u obtained by each resampling method for each data set, across three learning models, with internally tuned parameters. Best result for each data set is in bold; best result per type of resampling strategy is in italics

type	10	20	30	31	32	40	50	51	52	53
None	11.00	13.00	13.00	10.00	7.00	12.67	13.00	11.67	13.00	11.00
GAUSS	11.33	10.33	*6.33*	*7.33*	7.67	*8.33*	10.33	12.00	8.33	10.67
STGAUSS	*8.33*	*8.00*	6.67	8.67	*5.00*	9.33	*8.00*	8.67	*7.33*	8.67
STGAUSS$_S$	10.33	8.67	6.67	7.67	*5.00*	9.00	9.00	*8.33*	8.00	*5.67*
STGAUSS$_\phi$	10.00	10.33	8.00	11.00	10.33	12.33	9.00	9.67	9.33	10.00
ROS	5.00	8.33	8.33	7.00	*4.67*	*5.67*	*5.00*	9.33	8.33	8.00
STROS	5.00	8.33	7.00	10.00	5.67	7.67	7.67	*6.00*	*5.33*	*3.67*
STROS$_S$	6.00	*6.00*	7.67	*5.33*	7.67	6.33	5.67	6.33	6.33	*3.67*
STROS$_\phi$	*4.67*	8.00	*5.67*	10.33	11.00	9.00	7.33	7.67	7.33	6.67
RUS	5.67	3.00	8.67	3.67	6.67	3.00	4.33	3.67	6.67	5.67
STRUS	3.33	3.33	6.33	5.00	**5.33**	2.00	5.33	2.67	4.67	6.67
STRUS$_S$	**2.33**	2.67	4.67	**1.67**	8.67	4.33	4.00	4.00	5.33	8.00
STRUS$_\phi$	8.00	**1.00**	**2.00**	3.33	6.33	**1.33**	**2.33**	**1.00**	**1.00**	**2.67**

C.1.2 Fixing Parameters *A Priori*

Next, we show the results obtained by the prequential time-block evaluation procedure when setting parameters to the values in the middle of the grid search ($\alpha = 0.5$, $o = 2$, $u = 0.6$),

regardless of data set or learning algorithm. Tables C.3 and C.4 show the average ranks of F_1^u of each sampling method, aggregated by learning model and data set, respectively.

In Table C.3, we see that when using any of the learning algorithms, the best performance is achieved by using resampling methods with spatio-temporal bias; though unlike with the internally tuned parameters, now the best results with models MARS and RPART are obtained by different types of biased over-sampling (STROS and STROS$_S$, respectively). The combination of the MARS model with any type of over-sampling with Gaussian noise are the only cases where results are worse, on average, than no resampling. We find that, when adding Gaussian noise, no type of bias can beat GAUSS when using RF, and only one type of bias per model improves the performance against the random Gaussian noise resampling (static for RPART; our original proposal for MARS). When over-sampling without Gaussian noise, STROS improves the results over the random variant regardless of learning model. When under-sampling, STRUS and STRUS$_\phi$ perform better than their random counterpart, , while STRUS$_S$ only performed better than the random version when using RF.

Table C.3: Average ranks of F_1^u obtained by each resampling method for each learning model, across data sets, with parameters fixed *a priori*. Best result for each learning model is in bold; best result per type of resampling strategy is in italics

type	MARS	RF	RPART
None	8.60	12.50	11.00
GAUSS	10.10	*5.60*	6.30
STGAUSS	*8.80*	6.60	7.50
STGAUSS$_S$	10.80	7.00	*5.00*
STGAUSS$_\phi$	11.20	8.00	9.50
ROS	4.60	9.00	6.80
STROS	**3.40**	*7.40*	6.20
STROS$_S$	4.40	9.40	**4.20**
STROS$_\phi$	5.80	10.70	8.80
RUS	6.70	5.00	7.30
STRUS	6.20	3.00	6.10
STRUS$_S$	6.70	4.40	7.50
STRUS$_\phi$	*3.70*	**2.40**	*4.80*

In Table C.4, we can see that, even when using parameters that were not specifically selected for each data set, any type of resampling (biased or not) usually improves over the baseline, with some exceptions – there are 4 data sets where one or more types of over-sampling (with or without Gaussian noise, biased or not) do not improve the results against the baseline (these 4 data sets that had a similar issue when internally tuning parameters). Applying a spatio-temporal bias to a form of random resampling still achieves a result that is overall the best for 9 out of 10 data sets. Relevance-aware biased under-sampling works best (or ties for best) overall for 8 out of 10 data sets; only for data set 10, the smallest of them, does a random version of over-sampling beat all others. The random versions of resampling are improved by the introduction of at least one type of bias in almost all other cases –

the exception being that random over-sampling with Gaussian noise is not improved by the introduction of any type of bias for 3 out of 10 data sets (including data set 10).

Table C.4: Average ranks of F_1^u obtained by each resampling method for each data set, across three learning models, with parameters fixed *a priori*. Best result for each data set is in bold; best result per type of resampling strategy is in italics

type	10	20	30	31	32	40	50	51	52	53
None	9.67	13.00	13.00	10.00	6.67	11.67	12.67	8.67	11.67	10.00
GAUSS	*7.00*	6.67	6.33	8.33	7.67	*7.00*	*6.00*	10.33	7.33	6.67
STGAUSS	7.33	5.33	7.67	*8.00*	*8.33*	8.67	7.33	9.33	*6.33*	8.00
STGAUSS$_S$	10.33	*4.00*	**4.67**	9.67	8.67	8.33	7.00	*8.33*	8.67	*6.33*
STGAUSS$_\phi$	8.67	9.00	8.00	10.33	10.00	10.33	9.67	11.33	8.33	10.00
ROS	**3.33**	7.67	6.67	7.00	9.00	6.00	5.67	9.00	6.33	7.33
STROS	6.67	*5.67*	5.67	7.67	*7.33*	6.67	*5.00*	*4.67*	*4.67*	**2.67**
STROS$_S$	5.33	*5.67*	6.33	*6.67*	8.67	*4.00*	5.33	6.00	6.33	5.67
STROS$_\phi$	8.00	9.33	*5.33*	10.67	10.33	6.33	6.67	10.33	9.00	8.33
RUS	5.67	4.33	9.33	4.00	5.67	7.33	7.33	4.33	8.67	6.67
STRUS	4.67	6.00	7.33	**2.33**	3.67	5.33	6.67	4.00	5.00	6.00
STRUS$_S$	*4.00*	11.33	6.00	4.00	2.67	7.33	7.00	3.33	7.67	8.67
STRUS$_\phi$	10.33	**3.00**	**4.67**	**2.33**	**2.33**	**2.00**	**4.67**	**1.33**	**1.00**	*4.67*

C.1.3 Optimal Parametrization

In this section, we present the best results achieved by each resampling strategy for every data set and learning algorithm pairing. These results were obtained by running prequential time-block evaluation with all combinations of parameters applied to all training blocks in each data set – in contrast with internal tuning, where different parametrizations may have been chosen for each training block according to internally estimated performance – and, *a posteriori*, selecting the parameters that produced the best results for each data set and learning algorithm pair, in order to assess the potential of each strategy.

Table C.5 shows the average rank of each resampling approach aggregated by model. The baseline, where no step to address the imbalance problem was taken, is out-performed, on average, by all resampling approaches, regardless of the learning model being used. Including any type of spatio-temporal bias improves the rank of any type of re-sampling in almost all cases – the exceptions being STGAUSS$_\phi$ and STROS$_\phi$ when combined with RPART. STRUS$_\phi$ achieves the best results when using RPART or MARS – which is consistent with the results obtained by internally tuning parameters; however, here, STRUS$_S$ obtains the best rank overall when using RF. We find that, when under-sampling, introducing relevance-aware or static bias is best; while for the other types of resampling, either our original proposal or the static bias can perform better than other types of bias.

In Table C.6, a complementary view is given, aggregating the ranks by data set instead of by model. Here, STRUS$_\phi$ performs the best for 8 out of 10 data sets, with STRUS$_S$ performing best for data sets 10 and 32. When over-sampling (with or without Gaussian noise), static

Table C.5: Average ranks of F_1^u obtained by each resampling method for each learning model, across data sets, with optimal parameters determined *a posteriori*. Best result for each learning model is in bold; best result per type of resampling strategy is in italics

type	MARS	RF	RPART
None	12.00	13.00	13.00
GAUSS	11.50	8.80	7.70
STGAUSS	9.30	*5.40*	6.30
STGAUSS$_S$	*9.10*	6.50	*5.30*
STGAUSS$_\phi$	10.20	6.50	9.70
ROS	6.90	10.50	8.00
STROS	*3.40*	*8.50*	6.40
STROS$_S$	4.30	*8.80*	*5.40*
STROS$_\phi$	4.20	10.10	9.00
RUS	7.50	5.30	7.90
STRUS	4.60	3.20	6.00
STRUS$_S$	4.80	**1.90**	5.10
STRUS$_\phi$	**3.20**	2.50	**1.20**

bias works best for more data sets (7 out of 10 data sets, if adding noise; 5 out of 10 data sets, if not). All types of resampling, random or biased, improve against the baseline with no resampling, on average – finally, with no exceptions. There is a clear advantage afforded by biased resampling regardless of the types of resampling and bias being combined, though there are a few exceptions, more common when using STGAUSS$_\phi$ or STROS$_\phi$.

Table C.6: Average ranks of F_1^u obtained by each resampling method for each data set, across three learning models, with optimal parameters determined *a posteriori*. Best result for each data set is in bold; best result per type of resampling strategy is in italics

type	10	20	30	31	32	40	50	51	52	53
None	13.00	13.00	13.00	11.67	11.67	13.00	13.00	12.33	13.00	13.00
GAUSS	10.33	9.67	7.33	9.33	8.00	10.00	8.33	10.00	10.33	10.00
STGAUSS	*5.33*	*7.33*	*6.00*	7.00	6.00	9.00	8.00	8.67	*6.00*	6.67
STGAUSS$_S$	9.00	9.00	*6.00*	*4.67*	*5.33*	*8.00*	*7.33*	*7.67*	7.00	*5.67*
STGAUSS$_\phi$	7.67	8.00	8.67	8.33	9.00	10.00	9.67	9.33	9.67	7.67
ROS	6.33	10.33	9.33	9.67	*7.00*	7.00	6.33	9.33	9.00	10.33
STROS	*3.00*	7.67	*6.33*	6.33	7.33	7.67	6.00	8.33	*5.00*	*3.33*
STROS$_S$	5.00	*7.00*	6.67	*6.00*	7.33	*6.67*	*4.67*	*6.33*	6.67	5.33
STROS$_\phi$	6.00	8.67	8.00	7.33	9.67	9.33	6.67	*6.33*	7.00	8.67
RUS	10.00	4.33	7.67	10.00	10.67	4.00	5.67	5.00	7.00	4.67
STRUS	7.00	2.67	6.00	5.33	3.33	2.00	6.33	3.33	3.33	6.67
STRUS$_S$	**2.33**	2.33	3.67	4.00	**1.67**	3.00	6.00	3.33	6.00	7.00
STRUSϕ	6.00	**1.00**	**2.33**	**1.33**	4.00	**1.33**	**3.00**	**1.00**	**1.00**	**2.00**

C.1.4 Additional Results

Figures C.1 to C.7 show the results already presented in Section 5.3, now containing all our proposed biased resampling strategies, including our original, mixed proposal for spatio-temporal bias, and the variants of over-sampling with added Gaussian noise.

Table C.7: Average ranks of F_1^u results obtained by each resampling method, across all data set and learning model pairings, for three different types of parametrizations. Best result for each parametrization strategy is in bold; best result per type of resampling strategy is in italics

resampling	tuning	optimal	fixed
None	11.53	12.67	10.70
GAUSS	9.27	9.33	*7.33*
STGAUSS	7.87	7.00	7.63
STGAUSS$_S$	*7.83*	*6.97*	7.60
STGAUSS$_\phi$	10.00	8.80	9.57
ROS	6.97	8.47	6.80
STROS	6.63	*6.10*	*5.67*
STROS$_S$	*6.10*	6.17	6.00
STROS$_\phi$	7.77	7.77	8.43
RUS	5.10	6.90	6.33
STRUS	4.47	4.60	5.10
STRUS$_S$	4.57	3.93	6.20
STRUS$_\phi$	**2.90**	**2.30**	**3.63**

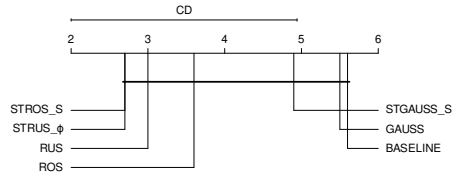
C.2 Meta-Features for Analysis of Impact of Data Set Characteristics

Table C.8 shows meta-features used to study the impact of domain characteristics on our proposals' efficiency.

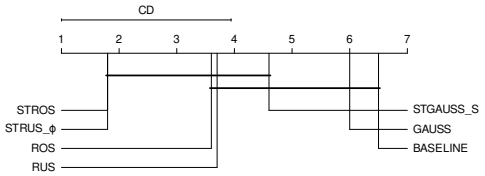
Table C.8: Data meta-features to analyse data characteristics impact on proposals' efficiency

	Meta-features
1	no. of observations
2	no. of time-stamps
3	no. of locations
4	ratio between no. of time-stamps and no. of locations
5	% of time-stamps× locations with available target value
6	avg. missing target values per location
7	avg. ratio between exponential and arithmetic series moving averages of standardised time
8	avg. time series' Hurst exponent
9	avg. 1st auto-correlation coefficient of the time series
10	avg. 1st auto-correlation coefficient of the differentiated time series
11	avg. 1st auto-correlation coefficient of the twice-differentiated time series
12	avg. sum of squares of the first 10 auto-correlation coefficients of the time series
13	avg. sum of squares of the first 10 auto-correlation coefficients of the differentiated time series
14	avg. sum of squares of the first 10 auto-correlation coefficients of the twice-differentiated time series
15	quantile 0.05 of normalised distances
16	avg. normalised distance
17	quantile 0.95 of normalised distances
18	inter-quartile range of normalised distances
19	avg. normalised distance to closest neighbour
20	avg. observed Moran I index
21	avg. observed significant Moran I index
22	% of positive Moran I indices
23	no. of outliers (using boxplot stats.)
24	% of outliers (using boxplot stats.)
25	% of extremes with $t_R = 0.9$ and automatic ϕ function
26	ratio between target standard deviation and mean
27	quantile 0.05 of target values
28	quantile 0.95 of target values
29	inter-quartile range of target values
30	type of outliers (extreme highs, lows, or both)
31	target distribution's Geary's kurtosis
32	target distribution's Pearsons's kurtosis
33	target distribution's skewness
34	avg. % of missing predictors per observation [1]
35	avg. correlation between target and predictors [1]

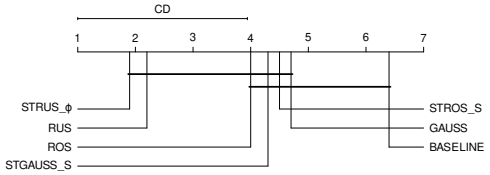
[1] Excluded from meta-analysis in Chapter 4, but included in Chapter 5.



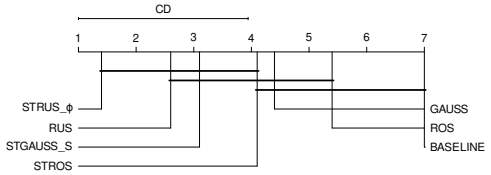
(a) MARS – Internal tuning



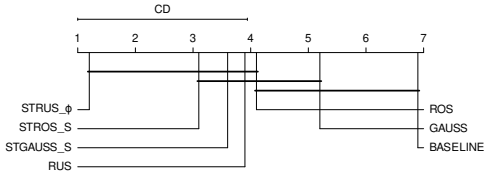
(b) MARS – Optimal parametrization



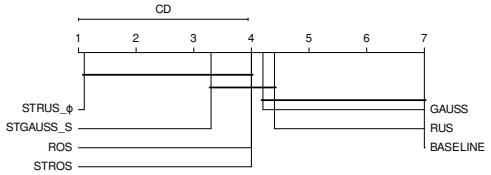
(c) RF – Internal tuning



(d) RF – Optimal parametrization

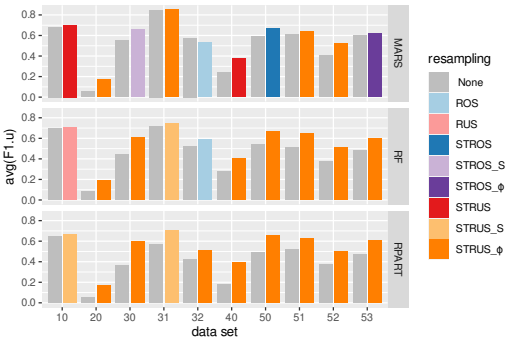


(e) RPART – Internal tuning

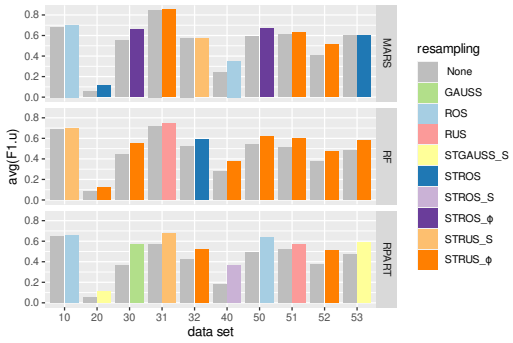


(f) RPART – Optimal parametrization

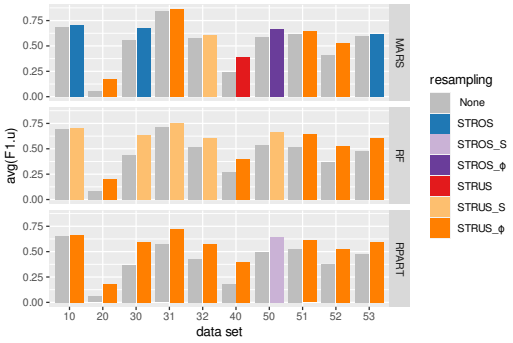
Figure C.1: Critical difference diagrams for selected algorithms with internally tuned and with optimal parametrization determined *a posteriori*, using the three learning algorithms. Groups of resampling strategies that are not significantly different according to the Nemenyi post-hoc test (at $p = 0.05$) are connected



(a) Internally tuned parameters



(b) Fixed parameters



(c) Optimal parameters

Figure C.2: Baseline and best F_1^u result achieved for each data set and learning model pair (top: MARS; middle: RF; bottom: RPART). Each pair of bars correspond to one data set: the baseline, in gray, and the best result in a color indicating the resampling method

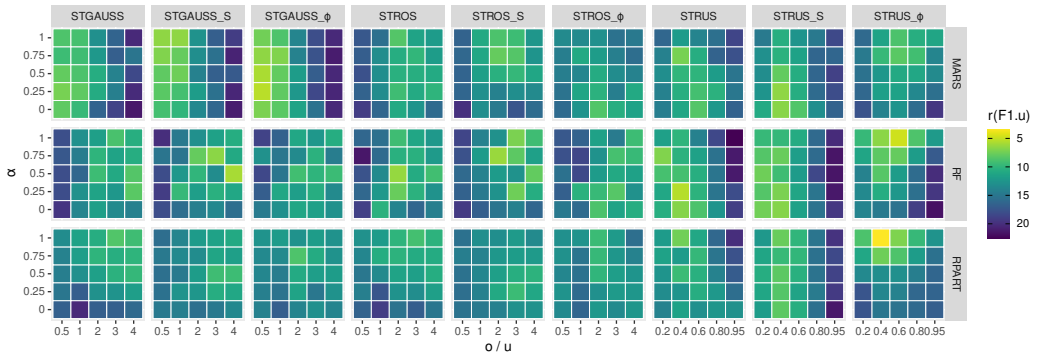


Figure C.3: Average F_1^u rank obtained by each resampling technique when using 25 different parametrizations averaged over all data sets. Lower ranks correspond to better results

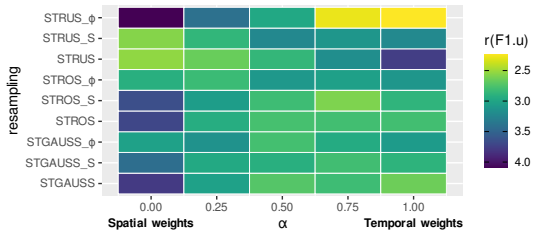
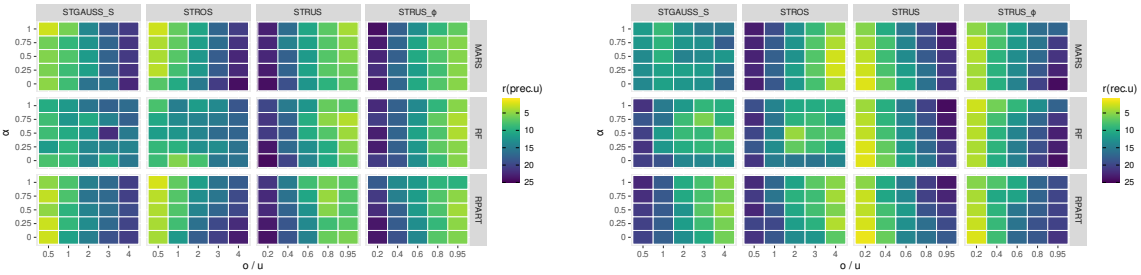


Figure C.4: Average F_1^u rank obtained by each resampling method for 5 different values of α averaged over all data set, learning model, and resampling percentage values. Lower ranks correspond to better results



(a) $prec_\phi^u$ (b) rec_ϕ^u

Figure C.5: Average utility-based precision and recall rank obtained by each resampling method for 25 different parametrizations, averaged over all data set and learning model pairings. Lower ranks correspond to better results

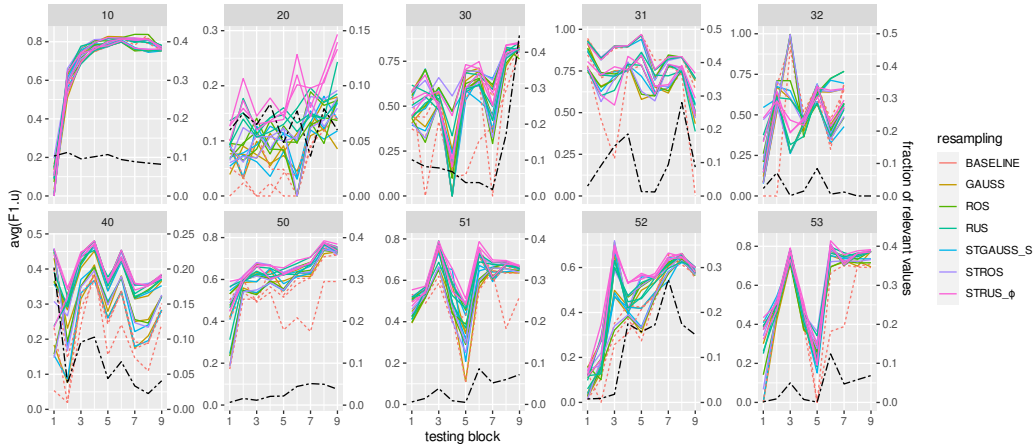


Figure C.6: F_1^u variation across testing blocks, averaged over repetitions for each data set and learning algorithm. Dashed orange lines show the results without any resampling (one line for each of the three learning models); other colors are used for selected resampling methods (with internally tuned parameters). Dot-dashed black lines show the percentage of relevant cases in each testing block, on the secondary y-axis – note that cases in the testing block are considered relevant if their relevance is 0.9 or above, as defined by the relevance function calculated automatically from each training set

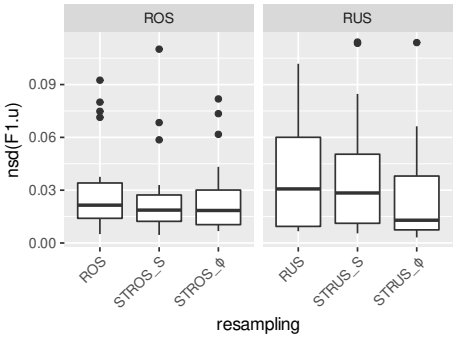


Figure C.7: Box and whiskers plot of normalized standard deviations of average F_1^u across repetitions, calculated for each data set, resampling strategy, and learning algorithm triad (only MARS and RPART were included, as experiments were only executed once when using RF), using internally tuned parameters

References

Maher Milad Aburas, Yuek Ming Ho, Mohammad Firuz Ramli, and Zulfa Hanan Ash'aari. The simulation and prediction of spatio-temporal urban growth trends using cellular automata models: A review. *International Journal of Applied Earth Observation and Geoinformation*, 52:380–389, 2016. ISSN 1872826X. doi:10.1016/j.jag.2016.07.007. URL `http://dx.doi.org/10.1016/j.jag.2016.07.007`.

Pierre Ailliot, Craig Thompson, and Peter Thomson. Space time modelling of precipitation using a hidden Markov model and censored Gaussian distributions. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 58(3):405–426, 2009. ISSN 00359254. doi:10.1111/j.1467-9876.2008.00654.x.

Annalisa Appice, Michelangelo Ceci, Donato Malerba, and Antonietta Lanza. Learning and Transferring Geographically Weighted Regression Trees across Time. In *Modeling and Mining Ubiquitous Social Media (MUSE)*, pages 97–117, 2011. doi:10.1007/978-3-642-33684-3_6.

Annalisa Appice, Anna Ciampi, Donato Malerba, and Pietro Guccione. Using trend clusters for spatiotemporal interpolation of missing data in a sensor network. *Journal of Spatial Information Science (JOSIS)*, 6(6):119–153, 2013a. ISSN 1948-660X. doi:10.5311/JOSIS.2013.6.102.

Annalisa Appice, Sonja Pravilovic, Donato Malerba, and Antonietta Lanza. Enhancing regression models with spatio-temporal indicator additions. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8249 LNAI:433–444, 2013b. ISSN 03029743. doi:10.1007/978-3-319-03524-6_37.

Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010. ISSN 1935-7516. doi:10.1214/09-SS054.

Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. Spatio-Temporal Data Mining: A Survey of Problems and Methods. *ACM Computing Surveys*, 51(4):1–41, 2018. ISSN 0360-0300. doi:10.1145/3161602.

Chris Barber, Joseph Bockhorst, and Paul Roebber. Auto-Regressive HMM Inference with Incomplete Data for Short-Horizon Wind Forecasting. In *Advances in Neural Information Processing Systems*, pages 136–144. Curran Associates, Inc., 2010. ISBN 9781617823800. URL `https://proceedings.neurips.cc/paper/2010/file/242c100dc94f871b6d7215b868a875f8-Paper.pdf`.

Rukshan Batuwita and Vasile Palade. Efficient resampling methods for training support vector machines with imbalanced datasets. In *International Joint Conference on Neural Networks (IJCNN)*, IJCNN'10, pages 1–8. IEEE, 2010. doi:10.1109/IJCNN.2010.5596787.

Anastassia Baxevani and Richard Wilson. Prediction of catastrophes in space over time. *Extremes*, 21:601—-628, mar 2018. ISSN 1572-915X. doi:10.1007/s10687-018-0314-z.

Kwabena Ebo Bennin, Jacky Keung, Akito Monden, Yasutaka Kamei, and Naoyasu Ubayashi. Investigating the Effects of Balanced Training and Testing Datasets on Effort-Aware Fault Prediction Models. In *IEEE Conference on Computer Software and Applications (COMPSAC)*, volume 1, pages 154–163, 2016. ISBN 9781467388450. doi:10.1109/COMPSAC.2016.144.

Christoph Bergmeir and José M Benítez. Forecaster performance evaluation with cross-validation and variants. In *International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 849–854, 2011. ISBN 9781457716751. doi:10.1109/ISDA.2011.6121763.

Christoph Bergmeir and José M Benítez. On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191:192–213, 2012. ISSN 00200255. doi:10.1016/j.ins.2011.12.028.

Christoph Bergmeir, Mauro Costantini, and José M Benítez. On the usefulness of cross-validation for directional forecast evaluation. *Computational Statistics & Data Analysis (CSDA)*, 76:132–143, 2014. ISSN 01679473. doi:10.1016/j.csda.2014.02.001.

Christoph Bergmeir, Rob J Hyndman, and Bonsoo Koo. A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Computational Statistics and Data Analysis*, 120:70–83, 2018. ISSN 01679473. doi:10.1016/j.csda.2017.11.003.

Jinbo Bi and Kristin P Bennett. Regression error characteristic curves. In *International Conference on Machine Learning (ICML)*, pages 43–50. AAAI Press, 2003. URL `https://www.aaai.org/Library/ICML/2003/icml03-009.php`.

Mehmet Bilgili, Besir Sahin, and Abdulkadir Yasar. Application of artificial neural networks for the wind speed prediction of target station using reference stations data. *Renewable Energy*, 32(14):2350–2360, 2007. ISSN 09601481. doi:10.1016/j.renene.2006.12.001.

Roger S Bivand, Edzer Pebesma, and Virgilio Gomez-Rubio. *Applied spatial data analysis with R, 2nd edition*. Springer, NY, 2013. URL `https://asdar-book.org/`.

Paula Branco. *Re-sampling Approaches for Regression Tasks under Imbalanced Domains*. PhD thesis, Universidade do Porto, 2014. URL `https://repositorio-aberto.up.pt/handle/10216/78240`.

Paula Branco, Rita P. Ribeiro, and Luis Torgo. UBL: an R Package for Utility-Based Learning. *Computing Research Repository (CoRR)*, abs/1604.0, 2016a. URL `https://arxiv.org/abs/1604.08079`.

Paula Branco, Luís Torgo, and Rita P Ribeiro. A Survey of Predictive Modelling under Imbalanced Distributions. *ACM Computing Surveys (CSUR)*, 49(May):31:1—-31:50, 2016b. ISSN 0360-0300. doi:10.1145/2907070. URL `http://arxiv.org/abs/1505.01658`.

Prabir Burman, Edmond Chow, and Deborah Nolan. A cross-validatory method for dependent data. *Biometrika*, 81(2):351–358, 1994.

Borja Calvo, Guzmán Santafé, and Guzman Santafe. scmamp: Statistical Comparison of Multiple Algorithms in Multiple Problems. *The R Journal*, 8(1):248–256, 2016. ISSN 20734859. doi:10.32614/rj-2016-017.

R Cano, C Sordo, and José M Gutierrez. Applications of Bayesian networks in meteorology. *Advances in Bayesian Networks*, pages 309–327, 2004. doi:10.1007/978-3-540-39879-0_17.

Steven S Carroll and Noel Cressie. Spatial modeling of snow water equivalent using covariances estimated from spatial and geomorphic attributes. *Journal of Hydrology*, 190 (1-2):42–59, 1997. ISSN 00221694. doi:10.1016/S0022-1694(96)03062-4.

Michelangelo Ceci, Roberto Corizzo, Fabio Fumarola, Donato Malerba, and Aleksandra Rashkovska. Predictive modeling of PV energy production: How to set up the learning task for a better prediction? *IEEE Transactions on Industrial Informatics*, 13(3):956–966, 2017. ISSN 15513203. doi:10.1109/TII.2016.2604758.

Vítor Cerqueira. *Ensembles for Time Series Forecasting*. PhD thesis, Universidade do Porto, 2019. URL `https://hdl.handle.net/10216/125125`.

Vítor Cerqueira, Luís Torgo, Jasmina Smailovic, and Igor Mozetič. A Comparative Study of Performance Estimation Methods for Time Series Forecasting. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 529–538, 2017. ISBN 9781509050048. doi:10.1109/DSAA.2017.7.

Vitor Cerqueira, Luis Torgo, and Igor Mozetič. Evaluating time series forecasting models: an empirical study on performance estimation methods. *Machine Learning*, 109(11):1997–2028, 2020. ISSN 15730565. doi:10.1007/s10994-020-05910-7.

Varun Chandola, Ranga Raju Vatsavai, Devashish Kumar, and Auroop Ganguly. *Analyzing Big Spatial and Big Spatiotemporal Data: A Case Study of Methods and Applications*, volume 33. Elsevier Inc., 1 edition, 2015. ISBN 9780444634924. doi:10.1016/B978-0-444-63492-4.00010-1.

Gang Chang, Shouhui Wang, and Xiaobo Xiao. Review of spatio-temporal models for short-term traffic forecasting. In *IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pages 8–12, 2016. ISBN 9781467390460. doi:10.1109/ICITE.2016.7581298.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002. ISSN 10769757. doi:10.1613/jair.953.

Tao Cheng and Jiaqiu Wang. Integrated Spatio-temporal Data Mining for Forest Fire Prediction. *Transactions in GIS*, 12(5):591–611, 2008. doi:10.1111/j.1467-9671.2008.01117.x.

Tao Cheng, James Haworth, Berk Anbaroglu, Garavig Tanaksaranond, and Jiaqiu Wang. Spatio-Temporal Data Mining. In Manfred M Fischer and Peter Nijkamp, editors, *Handbook of Regional Science*, pages 1173—-1193. Springer Berlin Heidelberg, 2014. ISBN 9783642234309. doi:10.1007/978-3-642-23430-9_68.

Felix Cheysson. starma: Modelling Space Time AutoRegressive Moving Average (STARMA) Processes, 2016. URL `https://cran.r-project.org/package=starma`.

C-K. Chu and James Stephen Marron. Comparison of two bandwidth selectors with dependent errors. *The Annals of Statistics*, 19(4):1906–1918, 1991. doi:10.1214/aos/1176348377.

David A Cieslak, Nitesh V Chawla, and Aaron Striegel. Combating imbalance in network intrusion datasets. In *IEEE International Conference on Granular Computing*, pages 732–737, 2006. ISBN 1424401348. doi:10.1109/grc.2006.1635905.

Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and ROC Curves. In *International Conference on Machine Learning (ICML)*, pages 233–240. ACM, 2006. doi:10.1145/1143844.1143874.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research (JLMR)*, 7(Jan):1–30, 2006. URL `https://www.jmlr.org/papers/v7/demsar06a.html`.

Luc Devroye and Terry Wagner. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory*, 25(5):601–604, 1979. doi:10.1109/TIT.1979.1056087.

Peter J. Diggle. *Statistical Analysis of Spatial and Spatio-Temporal Point Patterns, 3rd edition*. 2013. ISBN 9781466560246. doi:10.1201/b15326.

PeterJ. Diggle, Patrick Heagerty, Kung-Yee Liang, and Scott Zeger. *Analysis of longitudinal data*. Oxford University Press, 2002. ISBN 9780198524847.

Randall L Dougherty, Alan Edelman, and James M Hyman. Nonnegativity-, Monotonicity-, or Convexity-Preserving Cubic and Quintic Hermite Interpolation. *Mathematics of Computation*, 52(186):471–494, 1989. ISSN 00255718. doi:10.2307/2008477.

Stéphane Dray, Pierre Legendre, and Pedro R Peres-Neto. Spatial modelling: a comprehensive framework for principal coordinate analysis of neighbour matrices (PCNM). *Ecological Modelling*, 196(3-4):483–493, 2006. ISSN 03043800. doi:10.1016/j.ecolmodel.2006.02.015.

Sašo Džeroski. Multi-relational data mining: an introduction. *SIGKDD Explorations Newsletter*, 5(1):1–16, 2003. ISSN 1931-0145. doi:10.1145/959242.959245.

Anders Eklund, Thomas E Nichols, and Hans Knutsson. Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates. *Proceedings of the National Academy of Sciences (PNAS)*, 2016. ISSN 0027-8424. doi:10.1073/pnas.1602413113.

Alireza Ermagun and David Levinson. Spatiotemporal traffic forecasting: review and proposed directions. *Transport Reviews*, 38(6):786–814, 2018. doi:10.1080/01441647.2018.1442887.

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A Multiple Resampling Method for Learning from Imbalanced Data Sets. *Computational Intelligence*, 20(1):18–36, 2004. doi:10.1111/j.0824-7935.2004.t01-1-00228.x.

James H Faghmous and Vipin Kumar. Spatio-temporal Data Mining for Climate Data: Advances, Challenges, and Opportunities. In *Data Mining and Knowledge Discovery for Big Data*, pages 83–116. Springer, 2014. doi:10.1007/978-3-642-40837-3_3.

Alberto Fernández, Salvador García, María J del Jesus, and Francisco Herrera. A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems*, 159(18):2378–2398, 2008. ISSN 0165-0114. doi:10.1016/j.fss.2007.12.023.

Sara Fotouhi, Shahrokh Asadi, and Michael W Kattan. A comprehensive data level analysis for cancer diagnosis on imbalanced data. *Journal of Biomedical Informatics*, 90(October 2017):103089, 2019. ISSN 15320464. doi:10.1016/j.jbi.2018.12.003.

Stephen Milborrow. Derived from mda:mars by Trevor Hastie and Rob Tibshirani. Uses Alan Miller's Fortran utilities with Thomas Lumley's leaps wrapper. earth: Multivariate Adaptive Regression Splines, 2018.

Salvador García and Francisco Herrera. An extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008. ISSN 15324435. URL `https://www.jmlr.org/papers/v9/garcia08a.html`.

Caley K Gasch, Tomislav Hengl, Benedikt Gräler, Hanna Meyer, Troy S Magney, and David J Brown. Spatio-temporal interpolation of soil water, temperature, and electrical conductivity in 3D+ T: The Cook Agronomy Farm data set. *Spatial Statistics*, 14:70–90, 2015. ISSN 22116753. doi:10.1016/j.spasta.2015.04.001.

Seymour Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association (JASA)*, 70(350):320–328, 1975. doi:10.2307/2285815.

Uwe Haberlandt. Geostatistical interpolation of hourly precipitation from rain gauges and radar for a large-scale extreme rainfall event. *Journal of Hydrology*, 332(1):144–157, 2007. ISSN 00221694. doi:10.1016/j.jhydrol.2006.06.028.

Ali Hamdi, Khaled Shaban, Abdelkarim Erradi, Amr Mohamed, Shakila Khan Rumi, and Flora D Salim. *Spatiotemporal data mining: a survey on challenges and open problems*. Springer Netherlands, 2021. ISBN 0123456789. doi:10.1007/s10462-021-09994-y.

Tomislav Hengl. GSIF: Global Soil Information Facilities, 2017. URL `https://cran.r-project.org/package=GSIF`.

José Hernández-Orallo. ROC curves for regression. *Pattern Recognition*, 46(12):3395–3411, 2013. doi:10.1016/j.patcog.2013.06.014.

Robert J Hijmans. raster: Geographic Data Analysis and Modeling, 2020. URL `https://cran.r-project.org/package=raster`.

Torsten Hothorn, Kurt Hornik, and Achim Zeileis. Unbiased Recursive Partitioning: A Conditional Interference Framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674, 2006. doi:10.1198/106186006X133933.

Silke Janitza, Carolin Strobl, and Anne Laure Boulesteix. An AUC-based permutation variable importance measure for random forests. *BMC Bioinformatics*, 14(1):1, 2013. ISSN 14712105. doi:10.1186/1471-2105-14-119.

Yiannis Kamarianakis and Poulicos Prastacos. Space-time modeling of traffic flow. *Computers & Geosciences*, 31(2):119–133, 2005. doi:10.1016/j.cageo.2004.05.012.

Yasutaka Kamei, Akito Monden, Shinsuke Matsumoto, Takeshi Kakimoto, and Ken Ichi Matsumoto. The effects of over and under sampling on fault-prone module detection. In *International Symposium on Empirical Software Engineering and Measurement, ESEM*, pages 196–204, 2007. doi:10.1109/ESEM.2007.28.

Slava Kisilevich, Florian Mansmann, Mirco Nanni, and Salvatore Rinzivillo. Data Mining and Knowledge Discovery. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, chapter 4, pages 855–874. Springer, 2010. ISBN 9780387098227. doi:10.1007/978-981-15-8983-6_42.

Jonas Koehler and Claudia Kuenzer. Forecasting spatio-temporal dynamics on the land surface using earth observation data—a review. *Remote Sensing*, 12(21):1–34, 2020. ISSN 20724292. doi:10.3390/rs12213513.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1137—-1143, 1995. URL `https://dl.acm.org/doi/10.5555/1643031.1643047`.

Xiangjie Kong, Menglin Li, Kai Ma, Kaiqi Tian, Mengyuan Wang, Zhaolong Ning, and Feng Xia. Big Trajectory Data: A Survey of Applications and Services. *IEEE Access*, 6 (October):58295–58306, 2018. ISSN 21693536. doi:10.1109/ACCESS.2018.2873779.

Damjan Krstajic, Ljubomir J. Buturovic, David E. Leahy, and Simon Thomas. Cross-validation pitfalls when selecting and assessing regression and classification models. *Journal of Cheminformatics*, 6(1):1–15, 2014. ISSN 17582946. doi:10.1186/1758-2946-6-10.

Miroslav Kubat, Robert C. Holte, and Stan Matwin. Machine Learning for the Detection of Oil Spills in Satellite Radar Images. *Machine Learning*, 30(2-3):195–215, 1998. ISSN 08856125. doi:10.1023/a:1007452223027.

Ibai Lana, Javier Del Ser, Manuel Velez, and Eleni I Vlahogianni. Road Traffic Forecasting: Recent Advances and New Challenges. *IEEE Intelligent Transportation Systems Magazine*, 10(2):93–109, 2018. ISSN 19411197. doi:10.1109/MITS.2018.2806634.

Tuong Le, Mi Young Lee, Jun Ryeol Park, and Sung Wook Baik. Oversampling techniques for bankruptcy prediction: Novel features from a transaction dataset. *Symmetry*, 10(4), 2018. ISSN 20738994. doi:10.3390/sym10040079.

Sauchi Stephen Lee. Regularization in skewed binary classification. *Computational Statistics*, 14(2):277–292, 1999. ISSN 09434062. doi:10.1007/s001800050018.

Zhigang Li, M H Dunham, and Yongqiao Xiao. STIFF: A forecasting framework for spatiotemporal data. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 183–198, 2003. ISBN 3-540-20305-2. doi:10.1007/978-3-540-39666-6_12.

Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, and Yu Zheng. Geoman: Multi-level attention networks for geo-sensory time series prediction. In *International Joint*

Conference on Artificial Intelligence (IJCAI), volume 2018-July, pages 3428–3434, 2018. ISBN 9780999241127. doi:10.24963/ijcai.2018/476.

Johan Lindström, Adam Szpiro, Paul D Sampson, Silas Bergen, and Lianne Sheppard. SpatioTemporal: An R package for spatio-temporal modelling of air-pollution. *Journal of Statistical Software*, 2013.

Johan Lindström, Adam A Szpiro, Paul D Sampson, Assaf P Oron, Mark Richards, Tim V Larson, and Lianne Sheppard. A flexible spatio-temporal model for air pollution with spatial and spatio-temporal covariates. *Environmental and Ecological Statistics*, 21(3): 411–433, 2014. ISSN 13528505. doi:10.1007/s10651-013-0261-4.

Roderick J A Little and Donald B Rubin. *Statistical Analysis with Missing Data*, volume 151. John Wiley & Sons, 3rd edition, 2019. ISBN 9781118596012.

Jin Liu, Naiqi Wu, Yan Qiao, and Zhiwu Li. A scientometric review of research on traffic forecasting in transportation. *IET Intelligent Transport Systems*, 15(1):1–16, 2021. ISSN 1751-956X. doi:10.1049/itr2.12024.

Christopher Lloyd. *Spatial Data Analysis: An Introduction for GIS Users*. Oxford University Press, 2010. ISBN 978-0199554324.

K C Luk, J E Ball, and A Sharma. A study of optimal model lag and spatial inputs to artificial neural network for rainfall forecasting. *Journal of Hydrology*, 227(1-4):56–65, 2000. ISSN 00221694. doi:10.1016/S0022-1694(99)00165-1.

Kathryn L Lunetta, L Brooke Hayward, Jonathan Segal, and Paul van Eerdewegh. Screening large-scale association study data: Exploiting interactions using random forests. *BMC Genetics*, 5(January 2014), 2004. ISSN 14712156. doi:10.1186/1471-2156-5-32.

Shahrbanou Madadgar and Hamid Moradkhani. Spatio-temporal drought forecasting within Bayesian networks. *Journal of Hydrology*, 512:134–146, may 2014. ISSN 00221694. doi:10.1016/j.jhydrol.2014.02.039.

Donato Malerba. A relational perspective on spatial data mining. *International Journal of Data Mining, Modelling and Management (DMMM)*, 1(1):103, 2008. ISSN 1759-1163. doi:10.1504/IJDMMM.2008.022540.

Nikos Mamoulis. Spatio-temporal data mining. In *Encyclopedia of Database Systems*, pages 2725–2730. Springer US, 2009. ISBN 978-0-387-35544-3. doi:10.1007/978-0-387-39940-9_361.

Jean Damascène Mazimpaka and Sabine Timpf. Trajectory data mining: A review of methods and applications. *Journal of Spatial Information Science*, 13(13):61–99, 2016. ISSN 1948-660X. doi:10.5311/JOSIS.2016.13.263.

Maciej A Mazurowski, Piotr A Habas, Jacek M Zurada, Joseph Y Lo, Jay A Baker, and Georgia D Tourassi. Training Neural Network Classifiers for Medical Decision Making: The Effects of Imbalanced Datasets on Classification Performanc. *Neural networks*, 21 (2-3):436–997, 2008. doi:10.1016/j.neunet.2007.12.031.

Amy McGovern, David J. Gagne, John K. Williams, Rodger A. Brown, and Jeffrey B. Basara. Enhancing understanding and improving prediction of severe weather through spatiotemporal relational learning. *Machine Learning*, 95(1):27–50, 2014. ISSN 15730565. doi:10.1007/s10994-013-5343-x.

Hanna Meyer, Christoph Reudenbach, Tomislav Hengl, Marwan Katurji, and Thomas Nauss. Improving performance of spatio-temporal machine learning models using forward feature selection and target-oriented validation. *Environmental Modelling & Software*, 101:1–9, mar 2018. ISSN 13648152. doi:10.1016/j.envsoft.2017.12.001.

Dharmendra S Modha and Elias Masry. Prequential and Cross-Validated Regression Estimation. *Machine Learning*, 33(1):5–39, 1998. ISSN 08856125. doi:10.1023/A:1007577530334.

Nuno Moniz, Paula Branco, and Luís Torgo. Resampling strategies for imbalanced time series forecasting. *International Journal of Data Science and Analytics (JDSA)*, 3(3): 161–181, 2017a. doi:10.1007/s41060-017-0044-3.

Nuno Moniz, Paula Branco, Luís Torgo, and Bartosz Krawczyk. Evaluation of Ensemble Methods in Imbalanced Regression Tasks. *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications (LIDTA)*, 74:129–140, 2017b. URL `http://proceedings.mlr.press/v74/moniz17a.html`.

Nuno Moniz, Rita Ribeiro, Vitor Cerqueira, and Nitesh Chawla. SMOTEBoost for regression: Improving the prediction of extreme values. *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 150–159, 2019. doi:10.1109/DSAA.2018.00025.

Alejandro Moreo, Andrea Esuli, and Fabrizio Sebastiani. Distributional random oversampling for imbalanced text classification. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 805–808, 2016. ISBN 9781450342902. doi:10.1145/2911451.2914722.

Igor Mozetič, Luís Torgo, Vítor Cerqueira, and Jasmina Smailović. How to evaluate sentiment classifiers for Twitter time-ordered data? *PLOS ONE*, 13(3):1–20, 2018. doi:10.1371/journal.pone.0194317.

Sulaiman Ibrahim Musa, Mazlan Hashim, and Mohd Nadzri Md Reba. A review of geospatial-based urban growth models and modelling initiatives. *Geocarto International*, 32(8):813–833, 2017. ISSN 10106049. doi:10.1080/10106049.2016.1213891.

Attila M Nagy and Vilmos Simon. Survey on traffic prediction in smart cities. *Pervasive and Mobile Computing*, 50:148–163, 2018. ISSN 15741192. doi:10.1016/j.pmcj.2018.07.004.

M Nanni, B Kuijpers, C Körner, M May, and D Pedreschi. Spatiotemporal Data Mining. *Mobility, Data Mining and Privacy*, pages 267–296, 2008. doi:10.1007/978-3-540-75177-9_11.

Stefano Nembrini, Inke R König, and Marvin N Wright. The revival of the Gini importance? *Bioinformatics*, 34(21):3711–3718, 2018. ISSN 14602059. doi:10.1093/bioinformatics/bty373.

Orlando Ohashi and Luís Torgo. Wind speed forecasting using spatio-temporal indicators. In *European Conference on Artificial Intelligence*, pages 975–980. IOS Press, 2012. ISBN 9781614990970. doi:10.3233/978-1-61499-098-7-975.

Mariana Oliveira, Luís Torgo, Vítor Santos Costa, and Vítor Santos Costa. Predicting wildfires: Propositonal and relational spatio-temporal pre-processing approaches. In *International Conference on Discovery Science (DS)*, volume 9956 LNAI, pages 183–197, 2016. ISBN 978-3-319-67785-9. doi:10.1007/978-3-319-46307-0_12.

Mariana Oliveira, Luís Torgo, and Vítor Santos Costa. Evaluation Procedures for Forecasting with Spatio-Temporal Data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, volume 11051 LNAI, pages 703–718, 2018. ISBN 9783030109240. doi:10.1007/978-3-030-10925-7_43.

Mariana Oliveira, Nuno Moniz, Luís Torgo, and Vítor Santos Costa. Biased Resampling Strategies for Imbalanced Spatio-Temporal Forecasting. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 100–109. IEEE, 2019. ISBN 9781728144931. doi:10.1109/dsaa.2019.00024.

Mariana Oliveira, Nuno Moniz, Luís Torgo, and Vítor Santos Costa. Biased resampling strategies for imbalanced spatio-temporal forecasting. *International Journal of Data Science and Analytics*, 12(3):205–228, 2021a. ISSN 23644168. doi:10.1007/s41060-021-00256-2.

Mariana Oliveira, Luís Torgo, and Vítor Santos Costa. Evaluation Procedures for Forecasting with Spatiotemporal Data. *Mathematics*, 9(6), 2021b. ISSN 2227-7390. doi:10.3390/math9060691.

Jean Opsomer, Yuedong Wang, and Yuhong Yang. Nonparametric regression with correlated errors. *Statistical Science*, 16(2):134–153, 2001. ISSN 0883-4237. doi:10.1214/ss/1009213287.

R Kelley Pace, Ronald Barry, John M Clapp, and Mauricio Rodriquez. Spatiotemporal autoregressive models of neighborhood effects. *The Journal of Real Estate Finance and Economics*, 17(1):15–33, 1998.

Paulina Firozi. The Western Wildfires Are Affecting People 3,000 Miles Away, 2021. URL `https://www.washingtonpost.com/nation/2021/07/21/bootleg-fire-merge-air-quality-haze/`.

Edzer Pebesma. spacetime: Spatio-Temporal Data in R. *Journal of Statistical Software*, 51 (7):1–30, 2012. ISSN 1548-7660. doi:10.18637/jss.v051.i07.

Edzer Pebesma. Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal*, 10(1):439–446, 2018. doi:10.32614/RJ-2018-009.

Edzer Pebesma. stars: Spatiotemporal Arrays, Raster and Vector Data Cubes, 2020. URL `https://cran.r-project.org/package=stars`.

Edzer J Pebesma and Roger S Bivand. Classes and methods for spatial data in R. *R News*, 5(2):9–13, 2005.

Phillip E. Pfeifer and Stuart Jay Deutsch. A Three-Stage Iterative Procedure for Space-Time Modeling. *Technometrics*, 22(1):35–47, 1980. doi:10.2307/1268381.

Nico Piatkowski, Sangkyun Lee, and Katharina Morik. Spatio-temporal random fields: Compressible representation and distributed estimation. *Machine Learning*, 93(1):115–139, 2013. ISSN 08856125. doi:10.1007/s10994-013-5399-7.

Sonja Pravilovic and Annalisa Appice. Application of Spatio-Temporal Clustering in Forecasting Optimization of Geo-Referenced Time Series. *American Journal of Modeling and Optimization*, 2(1):8–15, 2014. ISSN 2333-1143. doi:10.12691/ajmo-2-1-2.

Sonja Pravilovic, Annalisa Appice, and Donato Malerba. An Intelligent Technique for Forecasting Spatially Correlated Time Series. In *Congress of the Italian Association for Artificial Intelligence*, pages 457–468, 2013. doi:10.1007/978-3-319-03524-6_39.

Sonja Pravilovic, Massimo Bilancia, Annalisa Appice, and Donato Malerba. Using multiple time series analysis for geosensor data forecasting. *Information Sciences*, 380:31–52, 2017. ISSN 00200255. doi:10.1016/j.ins.2016.11.001.

Sonja Pravilovic, Annalisa Appice, and Donato Malerba. Leveraging correlation across space and time to interpolate geophysical data via CoKriging. *Int. J. Geogr. Inf. Sci.*, 32(1): 191–212, 2018. doi:10.1080/13658816.2017.1381338.

Philipp Probst and Anne Laure Boulesteix. To tune or not to tune the number of trees in random forest. *Journal of Machine Learning Research*, 18:1–8, 2018. ISSN 15337928. URL `https://www.jmlr.org/papers/volume18/17-269/17-269.pdf`.

Philipp Probst, Marvin N Wright, and Anne-Laure Laure Boulesteix. Hyperparameters and Tuning Strategies for Random Forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(3):1–19, 2019. ISSN 19424795. doi:10.1002/widm.1301.

Phillip E Pteifer and Stuart Jay Deutsch. Stationarity and invertibility regions for low order starma models. *Communications in Statistics-Simulation and Computation*, 9(5):551–562, 1980. ISSN 15324141. doi:10.1080/03610918008812173.

R Core Team. R: A Language and Environment for Statistical Computing, 2017. URL `https://www.r-project.org/`.

Jeff Racine. Consistent cross-validatory model-selection for dependent data: hv-block cross-validation. *Journal of Econometrics*, 99(1):39–61, 2000. doi:10.1016/S0304-4076(00)00030-0.

K Venkateswara Rao, A Govardhan, and K V Chalapati Rao. Spatiotemporal data mining: Issues, tasks and applications. *International Journal of Computer Science & Engineering Survey (IJCSES)*, 3, 2012.

Felix Rembold, Clement Atzberger, Igor Savin, and Oscar Rojas. Using low resolution satellite imagery for yield prediction and yield anomaly detection. *Remote Sensing*, 5(4): 1704–1733, 2013. ISSN 20724292. doi:10.3390/rs5041704.

R Ribeiro. *Utility-based Regression*. PhD thesis, Dep. Computer Science, Faculty of Sciences - University of Porto, 2011. URL `https://www.dcc.fc.up.pt/~rpribeiro/publ/rpribeiroPhD11.pdf`.

Rita P. Ribeiro and Nuno Moniz. Imbalanced regression and extreme value prediction. *Machine Learning*, pages 1–33, 2020. ISSN 15730565. doi:10.1007/s10994-020-05900-9.

David R. Roberts, Volker Bahn, Simone Ciuti, Mark S. Boyce, Jane Elith, Gurutzeta Guillera-Arroita, Severin Hauenstein, José J. Lahoz-Monfort, Boris Schröder, Wilfried Thuiller, David I. Warton, Brendan A. Wintle, Florian Hartig, and Carsten F. Dormann. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017. doi:10.1111/ecog.02881.

John F Roddick and Myra Spiliopoulou. A bibliography of temporal, spatial and spatio-temporal data mining research. *ACM SIGKDD Explorations Newsletter*, 1(1):34–38, 1999. ISSN 1931-0145. doi:10.1145/846170.846173.

Shashi Shekhar, Zhe Jiang, Reem Y Ali, Emre Eftelioglu, Xun Tang, Venkata M V Gunturi, and Xun Zhou. Spatiotemporal data mining: A computational perspective. *ISPRS International Journal of Geo-Information*, 4(4):2306–2338, 2015. ISSN 22209964. doi:10.3390/ijgi4042306.

Shashi Shekhar, Yan Li, Reem Y Ali, Emre Eftelioglu, Xun Tang, and Zhe Jiang. *Spatial and Spatiotemporal Data Mining*, volume 3. Elsevier, 2017. ISBN 9780128046609. doi:10.1016/B978-0-12-409548-9.09594-4.

Xingjian Shi and Dit-Yan Yeung. Machine Learning for Spatiotemporal Sequence Forecasting: A Survey. *Computing Research Repository (CoRR)*, pages 1–20, 2018. URL `http://arxiv.org/abs/1808.06865`.

Tom A B Snijders. On Cross-Validation for Predictor Evaluation in Time Series. In *Workshop On Model Uncertainty and its Statistical Implications*, pages 56–69, 1988. ISBN 978-3-642-61564-1. doi:10.1007/978-3-642-61564-1_4.

Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society, Series B (Methodological)*, 36(2):111–147, 1974. doi:10.1111/j.2517-6161.1974.tb00994.x.

Carolin Strobl, Anne Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8, 2007. ISSN 14712105. doi:10.1186/1471-2105-8-25.

Carolin Strobl, Anne Laure Boulesteix, Thomas Kneib, Thomas Augustin, and Achim Zeileis. Conditional variable importance for random forests. *BMC Bioinformatics*, 9: 1–11, 2008. ISSN 14712105. doi:10.1186/1471-2105-9-307.

Yanmin Sun, Andrew K C Wong, and Mohamed S Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(4): 687–719, 2009. ISSN 02180014. doi:10.1142/S0218001409007326.

Adam A. Szpiro, Paul D. Sampson, Lianne Sheppard, Thomas Lumley, Sara D. Adar, and Joel D. Kaufman. Predicting intra-urban variation in air pollution concentrations with complex spatio-temporal dependencies. *Environmetrics*, 21(6):606–631, 2010. ISSN 11804009. doi:10.1002/env.1014.

Leonard J Tashman. Out-of-sample tests of forecasting accuracy: an analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000. ISSN 01692070. doi:10.1016/S0169-2070(00)00065-0.

Terry Therneau, Beth Atkinson, and Brian Ripley. rpart: Recursive Partitioning and Regression Trees, 2017. URL `https://cran.r-project.org/package=rpart`.

Craig S. Thompson, Peter J. Thomson, and Xiaogu Zheng. Fitting a multisite daily rainfall model to New Zealand data. *Journal of Hydrology*, 340(1-2):25–39, 2007. ISSN 00221694. doi:10.1016/j.jhydrol.2007.03.020.

Luís Torgo. Regression error characteristic surfaces. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, KDD'05, pages 697–702. ACM, 2005. doi:10.1145/1081870.1081959.

Luís Torgo. *Data mining with R: learning with case studies*. CRC press, 2016. ISBN 1482234890.

Luís Torgo and Rita P. Ribeiro. Utility-based regression. In *European Conference on Principles of Data Mining and Knowledge Discovery in Databases (PKDD)*, PKDD'07, pages 597–604. Springer, 2007. doi:10.1007/978-3-540-74976-9_63.

Luís Torgo and Rita P Ribeiro. Precision and recall for regression. In *International Conference on Discovery Science (DS)*, DS'09, pages 332–346. Springer, 2009. doi:10.1007/978-3-642-04747-3_26.

Luís Torgo, Rita P. Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Portuguese Conference on Artificial Intelligence (EPIA)*, pages 378–389. Springer, 2013. doi:10.1007/978-3-642-40669-0_33.

Mathias Trachsel and Richard J Telford. Estimating unbiased transfer-function performances in spatially structured environments. *Climate of the Past*, 12(5):1215–1223, 2016.

Cornelis J Van Rijsbergen. *Information retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979. ISBN 0408709294.

Ranga Raju Vatsavai, Varun Chandola, Scott Klasky, Auroop Ganguly, Anthony Stefanidis, and Shashi Shekhar. Spatiotemporal data mining in the era of big spatial data: Algorithms and applications. *Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data, BigSpatial*, 1(November):1–10, 2012. ISSN 2313-1799. doi:10.1145/2447481.2447482.

David Vaz, Vítor Santos Costa, and Michel Ferreira. Fire! firing inductive rules from economic geography for fire risk detection. In *International Conference on Inductive Logic Programming (ILP)*, volume 6489 LNAI, pages 238–252, 2011. ISBN 9783642212949. doi:10.1007/978-3-642-21295-6_27.

Eleni I Vlahogianni, John C Golias, and Matthew G Karlaftis. Short-term traffic forecasting: Overview of objectives and methods. *Transport Reviews*, 24(5):533–557, 2004. ISSN 14645327. doi:10.1080/0144164042000195072.

Eleni I Vlahogianni, Matthew G Karlaftis, and John C Golias. Short-term traffic forecasting: Where we are and where we're going. *Transportation Research Part C*, 43(June):3–19, 2014. ISSN 0968-090X. doi:10.1016/j.trc.2014.01.005.

Jacques Wainer and Gavin Cawley. Nested cross-validation when selecting classifiers is overzealous for most practical applications. *Expert Systems with Applications*, 182:1–9, 2021. ISSN 09574174. doi:10.1016/j.eswa.2021.115222.

Senzhang Wang, Jiannong Cao, and Philip Yu. Deep Learning for Spatio-Temporal Data Mining: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, page 1, 2020. ISSN 1041-4347. doi:10.1109/tkde.2020.3025580.

Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S Yu. PredRNN++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International Conference on Machine Learning (ICML)*, volume 80, pages 8122–8131, 2018. ISBN 9781510867963. URL `https://proceedings.mlr.press/v80/wang18b.html`.

Gerhard Widmer and Miroslav Kubat. Learning in the Presence of Concept Drift and Hidden Contexts. *Machine Learning*, 23(1):69–101, 1996. ISSN 0885-6125. doi:10.1023/A:1018046501280.

Christopher K Wikle. Modern perspectives on statistics for spatio-temporal data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(1):86–98, 2015. ISSN 19390068. doi:10.1002/wics.1341.

Marvin N. Wright and Andreas Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17, 2017. ISSN 15487660. doi:10.18637/jss.v077.i01.

Yuhong Yang. Comparing Learning Methods For Classification. *Statistica Sinica*, 16:635–657, 2006. ISSN 1941-0093.

Xiaobai Yao. Research issues in spatio-temporal data mining. Technical report, 2003.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3634–3640, 2018. ISBN 9780999241127. doi:10.24963/ijcai.2018/505.

Zheng and Yu. Trajectory Data Mining: An Overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):29, 2015. ISSN 2157-6904. doi:10.1145/2743025.

Yu Zheng, Furui Liu, and Hsun-Ping Hsieh. U-Air: When Urban Air Quality Inference Meets Big Data. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, page 1436, New York, NY, USA, 2013. ACM. ISBN 9781450321747. doi:10.1145/2487575.2488188.

Ligang Zhou. Performance of corporate bankruptcy prediction models on imbalanced dataset: The effect of sampling methods. *Knowledge-Based Systems*, 41:16–25, 2013. ISSN 09507051. doi:10.1016/j.knosys.2012.12.007.