D 2019

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

ENSEMBLES FOR TIME SERIES FORECASTING

VITOR MANUEL ARAÚJO CERQUEIRA
TESE DE DOUTORAMENTO APRESENTADA
À FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO EM
ENGENHARIA INFORMÁTICA

Vitor Manuel Araújo Cerqueira

Ensembles for Time Series Forecasting

Dissertation submitted to Faculdade de Engenharia da Universidade do Porto
to obtain the degree of

Doctor Philosophiae in Informatics Engineering

Approved by:

President: Rui Carlos Camacho de Sousa Ferreira da Silva, Ph.D.

Referee: Albert Bifet, Ph.D.

Referee: Paulo Jorge de Sousa Azevedo, Ph.D.

Referee: João Pedro Carvalho Leal Mendes Moreira, Ph.D.

Supervisor: Luís Fernando Rainho Alves Torgo, Ph.D.

_____
(Luís Fernando Rainho Alves Torgo)

Departamento de Engenharia Informática
Faculdade de Engenharia da Universidade do Porto

Dezembro de 2019

Dedicado aos meus pais.

Abstract

A time series represents a collection of data points captured over time. This type of data is actively studied in many domains of application, such as healthcare, finance, energy, or climate. The generalised interest in time series arises from the dynamic characteristics of many real-world phenomena, where events naturally occur and evolve over time.

Uncertainty is a significant issue when analysing time series, which complicates the accurate understanding of their future behaviour. To cope with this problem, organisations engage in forecasting to drive their decision-making process. Forecasting denotes the process of predicting the future behaviour of time series, which allows professionals to anticipate scenarios and take pro-active measures. In this context, the aim of this thesis is to advance the state of the art of the literature in time series forecasting. Particularly, our research goal can be divided into two main parts: (i) forecasting the future numeric values of time series; and (ii) the anticipation of interesting events in time series in a timely manner, a task that is commonly known as activity monitoring. In both parts, we adopt an ensemble learning approach, the field of machine learning that combines different predictive models to address a given predictive task.

The first part is split into two steps. Initially, we study how to estimate the predictive performance of forecasting models. The most appropriate methodology is still an open research question. In this context, we contribute to the literature by presenting an extensive empirical analysis of different methods for estimating the performance of forecasting models. We then develop new methods for time series forecasting. To accomplish this, we leverage the idea

that typically, all predictive forecasting models have strengths and limitations throughout a time series, and we adopt an ensemble learning approach to manage these. As such, several forecasting models are created according to different assumptions about the process generating the time series observations. These models are then weighed over time to cope with the dynamics of the data. We contribute to the literature by developing a meta-learning model designed to estimate the weights of each model in the ensemble. The proposed method is based on a regression analysis of the errors of these models. We argue that the developed method can better adapt to changes in the environment relative to the state of the art approaches. We also contribute to the literature by presenting a novel aggregation framework of forecasting models in an ensemble. This aggregation method explores the idea that, like individual models, different subsets of these models show a varying relative performance throughout a time series. Thus, the idea is to apply state of the art methods for the dynamic combination of forecasting models to these subsets, instead of applying them directly to the original set of models. We show the usefulness of the developed methods in an empirical manner, using a large set of time series from different domains of application.

Regarding the second part, we contribute to the literature of activity monitoring by developing a novel method based on layered learning. Layered learning works by dividing a predictive task into different sub-tasks, or layers, which are in principle easier to solve. A predictive model is then applied to each subtask. These models are then combined to make predictions about the original problem. We apply the proposed method to a case study in healthcare, where the objective is to predict impending critical health events, namely hypotension episodes and tachycardia episodes. These represent a significant cause of mortality in intensive care units, and it is essential to anticipate them. Based on the results in this case study, we conclude that the developed method is competitive with state of the art approaches.

Keywords: machine learning; time series; forecasting; ensemble methods; arbitration; layered learning; performance estimation

Resumo

Uma série temporal representa um conjunto de dados obtidos ao longo do tempo. Este tipo de dados é estudado ativamente em muitos domínios de aplicação, por exemplo em cuidados de saúde, finanças, energia, ou clima. O interesse generalizado em séries temporais está relacionado com as caraterísticas dinâmicas de vários fenómenos observados no mundo real, onde os acontecimentos ocorrem e evoluem naturalmente ao longo do tempo.

A incerteza é um aspeto fundamental na análise de séries temporais, e que dificulta a compreensão exata do comportamento futuro deste tipo de dados. Para lidar com este problema, é comum as organizações aplicarem modelos de previsão para apoiar a sua tomada de decisão. Neste contexto, o objetivo desta dissertação é avançar o estado de arte em previsão em séries temporais. Mais especificamente, o objetivo pode ser dividido em duas partes: (i) na previsão de valores futuros de séries temporais; e (ii) na deteção atempada de eventos interessantes em séries temporais. A abordagem adotada para resolver estes problemas é baseada em métodos *ensemble*, a área de aprendizagem computacional que combina diferentes modelos para resolver uma dada tarefa de previsão.

A primeira parte pode ser dividida em duas fases. Inicialmente, foi estudado como avaliar o desempenho preditivo de modelos de previsão em séries temporais. A forma mais apropriada para resolver este problema ainda é uma pergunta em aberto. Neste sentido, esta tese contribui para a literatura apresentando uma extensa análise empírica de diferentes métodos para avaliar modelos de previsão para séries temporais. Na fase seguinte, foram desenvolvidos novos métodos de previsão. A hipótese base do trabalho é que, normalmente, todos

os modelos de previsão apresentam pontos fortes e pontos fracos ao longo de uma série temporal. Neste sentido, adotamos uma estratégia *ensemble* para gerir este problema. Foram criados vários modelos de previsão de séries temporais, de acordo com diferentes suposições em relação ao processo que gera os dados. Depois, estes modelos são ponderados ao longo do tempo para lidar com as caraterísticas dinâmicas das séries temporais. Esta tese contribui para a literatura com o desenvolvimento de um novo método baseado em aprendizagem *meta*, cujo objetivo é estimar os pesos de cada modelo de previsão no *ensemble* em cada ponto da série temporal. O modelo desenvolvido é baseado numa análise de regressão dos erros dos modelos de previsão de séries temporais. Outra contribuição para a literatura é o desenvolvimento de outra nova abordagem para combinar os modelos de previsão que constituem um *ensemble*. A hipótese base é que, tal como modelos indivíduais, subconjuntos desses modelos apresentam um desempenho relativo que varia ao longo de uma série temporal. Neste sentido, a ideia é aplicar modelos de combinação a estes subconjuntos, em vez de aplicá-los diretamente aos modelos de previsão individuais. A utilidade dos métodos desenvolvidos é demonstrada de forma empírica, usando um grande conjuntos de séries temporais de diferentes domínios de aplicação.

Em relação à segunda parte, esta tese contribui para a literatura de previsão atempada de eventos em séries temporais com o desenvolvimento de um novo métodos baseado em aprendizagem em camadas. Uma abordagem de aprendizagem em camadas divide a tarefa de previsão em diferentes sub-tarefas, ou camadas, que em princípio são mais fáceis de resolver. Um modelo preditivo é treinado para resolver cada uma destas sub-tarefas. Os diferentes modelos são depois combinados para fazer previsões relativas ao problema original. O método proposto foi aplicado a um caso no domínio de cuidados de saúde, cujo objetivo é a previsão de crises de saúde iminentes em unidades de cuidados intensivos, mais especificamente episódios de hipotensão e episódios de taquicardia. Estes representam uma causa considerável de mortalidade em unidades de cuidados intensivos, o que mostra a importância da sua antecipação. Os resultados obtidos sugerem que o método desenvolvido é competitivo com métodos

do estado da arte.

Palavras-chave: aprendizagem computacional; séries temporais; previsão; métodos *ensemble*; arbitragem; aprendizagem em camadas; estimação de desempenho

Financial Support

This work was financially supported by *Fundação para a Ciência e a Tecnologia* (FCT), the Portuguese funding agency that supports science, technology, and innovation, through the Ph.D. grant SFRH/BD/135705/2018.

I also acknowledge funding from project "NORTE-01-0145-FEDER-000036", which is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).

CORAL
Sustainable Ocean
Exploitation

FCT
Fundação
para a Ciência
e a Tecnologia

Acknowledgments

I want to acknowledge everyone that helped me bring this thesis into fruition:

My supervisors, Luis and Carlos, for their guidance and valuable input. Notably, their support on my most useful ideas, and their acute BS detector on my not so useful ones. A special word to Luis, for believing in me and taking me as a student.

My colleagues and professors at LIAAD, INESC, and ProDEI. In particular, Fábio for many insightful discussions that enabled me to become a better researcher; Matias, for the Heidelbergers in Heidelberg and the Chardonnay in Skopje; Cláudio, for the competitions with Outdex; and João Gama for introducing me to scientific research.

The LIAAD crew at DCC for the excellent working environment. Especially Nuno, for the constant company in the office and for taking the time to read this thesis from cover to cover (on a Sunday); Mariana, for lending me her monitor; Paula, for pointing me the right direction of the related work of Chapter 5; and Carlos Leite, for his friendship.

Marcia, for her love and for teaching me how to write a scientific paper.

My family, for their love and support.

My friends, for the pleasant and relaxing times.

Finally, I also want to express my gratitude to the scientific community. The

authors of the many scientific papers I have read, especially those cited in this dissertation, and the anonymous reviewers for their constructive feedback.

Vitor Cerqueira

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

1.1 Context

In the current information age, massive amounts of data are produced continuously. Increasingly, organisations try to make sense of this data and use it to make data-driven decisions. In many cases, the data being collected is a time series, which denotes a set of observations captured over time. This type of data often comprises some degree of temporal dependency among observations, which means that the currently observed value may depend on the values captured previously.

Time series are used to represent the dynamics of many real-world phenomena, and one of the main challenges in data science is to accurately model the process generating the observations of this type of data. Instances appear in a wide range of domains. In healthcare informatics, patients are continually being monitored by devices that provide time series that doctors use to make therapeutic decisions (Ghosh et al., 2016). In finance, professionals track financial instruments over time for economic profits (Graham and Zweig, 2003). Because of the profusion of time series, analysing and learning from these data sources has been one of the most active topics among the research community. The generalised interest in time series arises from the changing nature of many real-world phenomena. Uncertainty is a significant issue in these scenarios, which complicates the accurate understanding of the future behaviour of time series.

Given the uncertainty behind time series, many organisations engage in forecasting. Let $Y = \{y_1, \ldots, y_n\}$ denote a time series. Forecasting denotes the process of estimating the future values of Y, for example, y_{n+1}. Understanding the dynamics of time series is fundamental for decision-makers. It allows them to anticipate scenarios and make proactive decisions which might be invaluable in their domain. For example, intelligent transportation systems rely on short-term traffic flow forecasting to enhance the operational efficiency in the road network (Moreira-Matias et al., 2013).

Forecasting methods have been studied for decades (Trigg, 1964). These have been designed to cope with the temporal dependency among observations. Notwithstanding, some important challenges are still open. Forecasting is an extrapolation process, which considerably increases the uncertainty of the estimations. These estimations produced by a forecasting model are statements conditioned on past assumptions drawn from the collected observations (Chatfield, 2000). Often, however, the underlying process generating a time series changes due to non-stationarities and time-evolving complex structures. This process is commonly known as concept drift (Gama et al., 2014). On top of this, despite having domain expertise, professionals often lack proper time series analysis skills (Taylor and Letham, 2018). Getting the most out of state of the art time series methods requires significant experience, and it is a complex and time-consuming task. In this context, forecasting models should be able to cope with changes in the environment and adapt to new concepts automatically and efficiently.

1.1.1 Forecast Combination

Over the years, many forecasting methods have been proposed, ranging from auto-regressive processes (Hyndman and Athanasopoulos, 2018) to exponential smoothing approaches (Gardner Jr, 1985). Despite this, it is widely accepted that no particular forecasting method is universally applicable (Chatfield, 2000). Different procedures model time series according to a distinct hypothesis, which does not hold at all times. This idea is consistent with the *No Free Lunch* theorem for supervised learning, which states that no learning algorithm is the most appropriate for all predictive tasks (Wolpert, 1996). It is also widely

accepted that even over a single time series, different forecasting methods are better at different times (Aiolfi and Timmermann, 2006).

The idea that every particular predictive model has some limitations is the primary hypothesis behind ensemble learning methods (Brown, 2010a). The goal of these methods is to combine the predictions of different predictive models. Diversity among the individual models is known to be a crucial component in ensemble methods (Brown et al., 2005a). This means that different models should provide overall good but different predictions from one another. Effectively, the idea is to have different models that are better at different parts of the data space to manage the limitations of each one. The predictive advantage of ensemble methods has been demonstrated in several studies, both theoretical and empirical ones (Breiman, 1996; Ueda and Nakano, 1996).

Ensemble methods have been applied to many domains, including forecasting (Newbold and Granger, 1974). By employing models that follow different assumptions regarding the underlying process generating the time series, we expect that individual learners will disagree with each other. This process introduces a natural diversity into the ensemble, which helps handle different dynamic regimes in a time series (Kuncheva, 2004a).

One of the main challenges when applying ensemble learning methods is the determination of which model is stronger in a given point in time (Jacobs, 1995). Several approaches have been proposed to this effect, most of which designed to adapt to concept drift. A common solution is to weigh the available models according to their performance in recent observations. The intuition is that these are the most similar to the one we intend to predict, and thus should be more important. Other approaches include meta-learning (Brazdil et al., 2008) or regret minimisation (Cesa-Bianchi and Lugosi, 2006).

1.1.2 Evaluating Forecasting Models

When developing a predictive model to solve a given problem, such as time series forecasting, one needs a framework to estimate the predictive performance of that model. Performance estimation is a crucial stage in the process of building predictive models, not only in forecasting problems but in machine learning in general. Performance estimation denotes the task of estimating the loss that a

predictive model will incur on unseen data. When the observations in the data are independent and identically distributed, the most common approach to this task is cross-validation (Geisser, 1975). However, the temporal dependency among time series observations raises some issues regarding the application of cross-validation in these scenarios.

In time series, performance estimation is typically carried out using an out-of-sample procedure. These approaches simulate future observations by holding out the last part of the available time series for estimating the loss of forecasting models (Tashman, 2000). Notwithstanding, variants of cross-validation tailored for dependent data have been proposed, for example, blocked cross-validation (Snijders, 1988). Understanding what sort of estimation method is the most appropriate is important because of the need to have reliable estimates about the generalisation ability of predictive models.

1.1.3 Activity Monitoring

As we mentioned before, many organisations rely on forecasting systems to predict the general future of time series as well as possible. This allows them to allocate their resources optimally. However, in some domains of application we are often interested in forecasting, not the general behaviour of the time series (e.g. whether it goes up or down in the next observation, and the respective magnitude), but specific events that require some action from professionals. The predictive task that addresses this type of scenarios is known as activity monitoring (Fawcett and Provost, 1999). The primary goal behind activity monitoring is the timely detection of interesting events, which may be disruptive in the particular domain of application. The accurate and timely prediction of such events can be crucial in decision making because it enables professionals to take appropriate actions to prevent those events or mitigate their consequences.

Activity monitoring systems are relevant in many scenarios. Consider the following example from healthcare. Some infants in neonatal intensive care units are diagnosed with sepsis disease. According to Griffin and Moorman (2001), these infants show abnormal heart beating patterns, up to twenty-four hours before the diagnostic. A system that monitors the heart rate of babies and is designed to capture anomalous events in a timely manner can prevent health

crisis and lead to better healthcare (Ghalwash et al., 2013). Note that the objective is not to predict the value of the heart rate at each point in time for a particular infant. The goal is to forecast a specific event of interest in a timely manner, in this case, abnormal heart beating patterns.

One of the main challenges behind activity monitoring is implicit in the word *timely*. This expression implies that there is an appropriate warning time interval between the point an alarm is issued about an event of interest, and the point the event occurs. This time interval is crucial to professionals, so they can assess the situation and decide the course of action. Another challenge behind activity monitoring is that typically, the events of interest are rare. Therefore, learning the concept behind these events represents an imbalanced learning problem (Branco et al., 2016a). The objective of these predictive tasks is to find patterns in the data that are not consistent with the typical behaviour of an activity. Different approaches have been proposed to address this problem, including supervised and unsupervised learning methods (Fawcett and Provost, 1999).

1.2 Goals and Research Questions

In the previous section, we addressed the importance of forecasting in the decision making of organisations across different domains. We described the main challenges that practitioners face when solving a time series forecasting problem. Within these challenges, we emphasised concept drift, the consequences of the *No Free Lunch* theorem, and the evaluation of predictive models. We also pointed out that in some scenarios, one may be interested in forecasting particular events of interest, and the challenges that this problem entails. In this context, the main goal of this thesis is to:

Develop new machine learning methods for (1) forecasting the future values of a time series, and (2) predicting interesting events in a timely manner.

The proposed methods aim at improving the predictive performance of predictive models in time-dependent domains, and consequently, the quality of the decision-making process of professionals within organisations. Within the scope of forecasting the future values of time series, we aimed at minimising the er-

ror between the predicted values and the values observed. Our work includes the analysis of methods for estimating the predictive performance of forecasting models. Regarding the prediction of interesting events, i.e. activity monitoring, we aimed at maximising the number of events detected in a timely manner while maintaining an adequate number of false alarms.

We decompose the research goal into four research questions:

RQ1 Given the time dependency among observations, what is the most appropriate way of estimating the predictive performance of forecasting models?

RQ2 How can we dynamically combine a set of forecasting models and cope with non-stationary sources of variation that are frequently at play in time series?

RQ3 Can we dynamically aggregate a set of forecasting models using a combination of aggregation functions to achieve a better trade-off between diversity and individual error of the members of the ensemble?

RQ4 How can we better cope with the low frequency of events of interest and detect more of them in a timely manner?

Before developing novel methods for addressing time series forecasting problems, we need a reliable approach to estimate the predictive performance of these methods. While several approaches have been proposed in the literature, there is no consensus regarding the most appropriate one. In this context, to answer the first question, we carried out an extensive empirical experiment to understand what is the most appropriate way of assessing the quality of a forecasting model. We compared several procedures that have been used to estimate the predictive performance of forecasting models (**Chapter 3**).

The second research question was addressed by developing a meta-learning method for dynamically weighting a set of forecasting experts. The idea of the proposed approach is to model the expertise across the time series of each forecasting model. An arbiter is created for each expert that is part of the ensemble. Each arbiter is specifically designed to model how apt its expert is to make an accurate prediction for a given test example. This is accomplished by analysing how the error incurred by a given learning model relates to the characteristics of the time series. At run-time, the experts are weighted according

to their expected degree of competence in the input observation, estimated by the predictions of the arbiters (**Chapter 4**).

To answer the third question, we propose a novel approach to combine a set of forecasting experts, which is dubbed constructive aggregation. We leverage the insight that similarly to individual forecasting models, different subsets of these models show a varying relative performance across a time series. The general idea behind constructive aggregation is to, instead of directly aggregating individual forecasting experts, first rearrange them into different subsets, creating a new set of combined models which is then aggregated into a final decision (**Chapter 5**).

To answer the fourth research question, we developed a layered learning method geared towards activity monitoring problems. Particularly, we split the original predictive task into two separate sub-tasks, which are hopefully simpler to solve. A predictive model is then created to solve each of the sub-tasks. The output of both models is combined to make a final prediction. The proposed method is applied in a particular domain of application: the timely prediction of critical health events in intensive care units of hospitals (**Chapter 6**).

The research questions presented above are addressed empirically. The proposed methods were tested from a different set of perspectives, according to an experimental design developed to this effect. The significance of the main results was assessed according to Bayesian analysis (Benavoli et al., 2017). In support for reproducible science, the code developed for each set of experiments is published online.

1.3 Thesis Contributions

The main research line of this thesis is concerned with learning from time-dependent data. This thesis contributes to the field by proposing novel methods for predicting the future behaviour of time series data. Our contributions reach several fields, including ensemble learning, dynamic model selection, meta-learning, performance estimation, forecasting, and activity monitoring.

The contributions of this thesis can be summarised as follows:

- A meta-learning model for dynamically weighting the predictions of a set

of forecasting models, which is dubbed Arbitrated Dynamic Ensemble;

- A sequential re-weighting strategy for encouraging diversity in dynamic ensembles for time-dependent data, which is based on the recent correlation among the forecasting experts;
- An approach for retrieving out-of-bag predictions from the training data to increase the data to train meta-learning models;
- An extensive set of experiments comparing the proposed approach to state of the art methods for dynamically combining a set of forecasting models. These experiments include an analysis of the impact of additional individual models in the ensemble, or a comparison of different individual models for forecasting;
- A method for combining a portfolio of predictive models in a hierarchical manner termed constructive aggregation;
- The application of constructive aggregation to time series forecasting problems using the concept of out-performance contiguity, which is also formalised in this thesis;
- An extensive empirical comparison of performance estimation methods to time series forecasting. In the experiments we test different types of time series (artificial and from the real-world), control for stationarity, and present a descriptive model that relates the most appropriate estimation method with respect to different time series characteristics;
- A novel layered learning method for the early detection of events in time series data;
- The application of the new layered learning method to the early detection of critical health events, namely acute hypotensive episodes and tachycardia episodes.

1.4 Bibliographic Note

Parts of this thesis have been published in the following articles:

- Cerqueira, V., and Torgo, L. "Combining Forecasters using Arbitration for Water Consumption Forecasting". Doctoral Symposium in Informatics Engineering, pages 149–161, 2017. (**Best Paper Award**)

- Cerqueira, V, Torgo, L, and Soares, C. "Arbitrated Ensemble for Solar Radiation Forecasting." International Work-Conference on Artificial Neural Networks. Springer, 2017;

- Cerqueira, V, Torgo, L, M. Oliveira, and B. Pfahringer. "Dynamic and heterogeneous ensembles for time series forecasting". IEEE International Conference on Data Science and Advanced Analytics (DSAA), pages 242–251, Oct 2017. doi: 10.1109/DSAA.2017.26;

- Cerqueira, V, Torgo, L., Pinto, F., and Soares, C. "Arbitrated Ensemble Time Series Forecasting." Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2017, Springer, Cham. (**Best Student Machine Learning Paper Award**);

- Cerqueira, V, L. Torgo, Smailović, J., and Mozetič, I. "A comparative study of performance estimation methods for time series forecasting". IEEE International Conference on Data Science and Advanced Analytics (DSAA), pages 529–538, 2017;

- Cerqueira, V., Torgo, L., Pinto, F. and Soares, C. Arbitrage of forecasting experts. Machine Learning, 108(6), pages 913-944, 2019;

- Cerqueira, V., Pinto, F., Torgo, L., Soares, C., and Moniz, N. "Constructive Aggregation and Its Application to Forecasting with Dynamic Ensembles." Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2018 pages 620-636. Springer, Cham;

- Cerqueira, V., Torgo, L. and Soares, C. "Layered Learning for Early Anomaly Detection: Predicting Critical Health Episodes." International Conference on Discovery Science. Springer, Cham, 2019.

In collaboration with other researchers on related topics to this thesis, the following additional articles have been published:

- Cerqueira, V., Pinto, F., Sá, C., and Soares, C. "Combining boosted trees with metafeature engineering for predictive maintenance". In International Symposium on Intelligent Data Analysis, pages 393-397, 2016. Springer, Cham;

- Pinto, F., Cerqueira, V., Soares, C., and Mendes-Moreira, J. "autoBagging: Learning to Rank Bagging Workflows with Metalearning", 2017. arXiv preprint arXiv:1706.09367;

- Moniz, N., Ribeiro, R., Cerqueira, V., and Chawla, N. "SMOTEBoost for Regression: Improving the Prediction of Extreme Values". IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA), pages 150-159, 2018;

- Mozetič, I., Torgo, L., Cerqueira, V., and Smailović, J. "How to evaluate sentiment classifiers for Twitter time-ordered data?". PloS one, 13(3), 2018;

- Cerqueira, V., Moreira-Matias, L., Khiari, J., and van Lint, H. "On Evaluating Floating Car Data Quality for Knowledge Discovery". IEEE Transactions on Intelligent Transportation Systems, 19(11), 3749-3760, 2018;

- Cerqueira, V., Torgo, L., and Soares, C. "Machine Learning vs Statistical Methods for Time Series Forecasting: Size Matters." arXiv preprint arXiv:1909.13316, 2019.

1.5 Thesis Outline

This thesis is structured in four parts. The first part serves as an introduction and contains the current and the next chapter. In this chapter (Chapter 1), we introduced and motivated the topic of this thesis. We also stated the main goal and decomposed it into four research questions. Finally, we summarised the contributions of the thesis.

In the next chapter (Chapter 2), we provide more detail on the problems addressed in this thesis. We introduce fundamental concepts behind time series analysis and predictive modelling for this type of data. We also formalise

the main predictive tasks addressed in this thesis and list some of the most important state of the art approaches to solve them.

The main body of the thesis is split into two parts: Part II and Part III. Part II is comprised by three chapters (Chapter 3, 4, and 5), and addresses the problem of forecasting, in which the objective is to predict the next value of time series according to its past values. Chapter 3 explores different methods designed to estimate the predictive performance of forecasting models. In Chapter 4, we present a novel method for dynamically combining a set of forecasting experts. Finally, in Chapter 5, we propose a new method for aggregating predictive models and apply it to forecasting.

Part III is comprised of Chapter 6 and is devoted to activity monitoring. In that chapter, we propose a new method for the timely detection of interesting events in time series. We apply this method to a case study in the healthcare domain.

The final part of the thesis (Part IV) includes Chapter 7, which concludes the thesis. We answer the research questions put forward in this chapter, and point out some open issues and future directions. The thesis also includes an Appendix, where the time series data sets and parameter setting of the learning algorithms are discriminated.

Chapter 2

Background

In the second chapter of the introductory part of this thesis, we provide the essential background related to the topic of our work. In Section 2.1, we start by describing time series data. We define basic concepts, and present state of the art approaches to model this type of data. Afterwards, we overview the three main topics that will be addressed in the subsequent chapters of the thesis. In Section 2.2, we describe the problem of time series forecasting, listing its main challenges and current state of the art approaches. A particular emphasis is given to ensemble learning methods, which represent an important foundation of the work of this thesis. Section 2.3 addresses the issue of evaluating time series forecasting models. We overview several performance estimation methods as well as evaluation metrics to this effect. Finally, in Section 2.4, we describe the activity monitoring predictive task. We describe the main distinctions between the *classical* forecasting task and the activity monitoring one. We also describe the main approaches to tackle this type of problems, listing some important methods in the literature. In summary, the goal of this chapter is to provide the reader with the necessary background and state of the art on the topic of this thesis, which represents a backbone for the remainder of the thesis.

2.1 Time Series

2.1.1 Introduction

A time series Y is a temporal sequence of values $Y = \{y_1, y_2, \ldots, y_n\}$, where y_i is the value of Y at time i. We focus on numeric time series, i.e., $y_i \in \mathbb{R}$, $\forall$ $y_i \in Y$. Throughout this work, we assume that each observation is captured in regular time intervals, e.g. every day.

Time series is an important topic in the literature, and instances of this type of data abound. For example in healthcare informatics (Ghalwash et al., 2013; Ghosh et al., 2016), finance (Graham and Zweig, 2003; Chan, 2004), energy consumption (Asafu-Adjaye, 2000), intelligent transportation systems (Moreira-Matias et al., 2013), or fraud detection (Fawcett and Provost, 1997; Yamanishi and Takeuchi, 2002). As an example, in Figure 2.1 is shown the average, and respective standard deviation, of water consumption levels in a specific area of the city of Oporto for each day of the year. The optimisation of the water pumping schedule and water treatment strategies enables operation planners to reduce energy and water treatment costs while maximising the quality of supplied water.

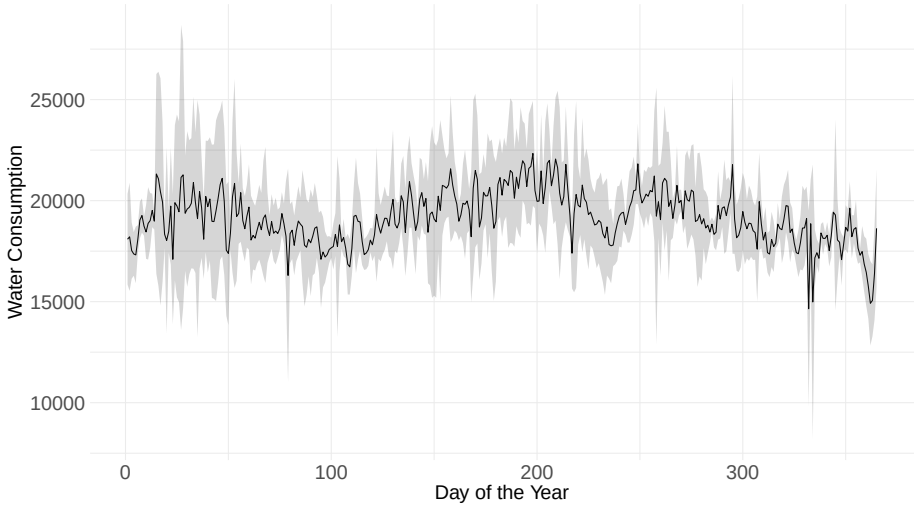


Figure 2.1: Mean and standard deviation of water consumption (in cubic meters) by day of the year

The generalised interest in time series arises from the dynamic characteristics of many real-world phenomena. Uncertainty is a major issue in these problems, which complicates the exact understanding of their future behaviour. This is one key motivation for the study of time series data.

2.1.2 Basic Components

The dynamic nature of time series can be related to the different components comprising time series, namely: the trend, seasonality, cyclic, and irregular components.

The trend component is usually referred to as a long-term change in the mean level of the data. In Figure 2.2 (upper tile) is shown a time series of airline passengers (Box et al., 2015). This graphic illustrates a time series with an upward trend as the mean level of passengers increases over time. If the trend of the time series is of no interest, it can be removed. We describe some methods to this effect in Section 2.1.3. This transformation is shown in the middle tile of Figure 2.2.

When time series experiences regular and predictable changes in fixed periods, it is said to contain a seasonal component. Besides the trend component mentioned above, Figure 2.2 (upper tile) also presents a monthly seasonality. The monthly variation can be estimated, or removed if it is of no interest. In the lower tile of Figure 2.2 is shown the time series without the trend and seasonal components.

Besides seasonal effects, sometimes time series show other predictable oscillations, but which do not have a fixed period. This type of variation is a cyclic pattern. The typical example of a cyclic pattern is a business cycle, in which the economy experiences periods of growth and periods of recession.

After removing the three above components from the time series, the remaining part is known as the irregular component or residuals. An example of this is shown in the lower tile of Figure 2.2. This component is not explainable by any trend, seasonal or cyclic behaviour, but can cause an impact on the dynamics of the time series (Hyndman and Athanasopoulos, 2018).

At a given point in time i, a time series can be decomposed in an additive fashion into the above-mentioned components as follows (Hyndman and

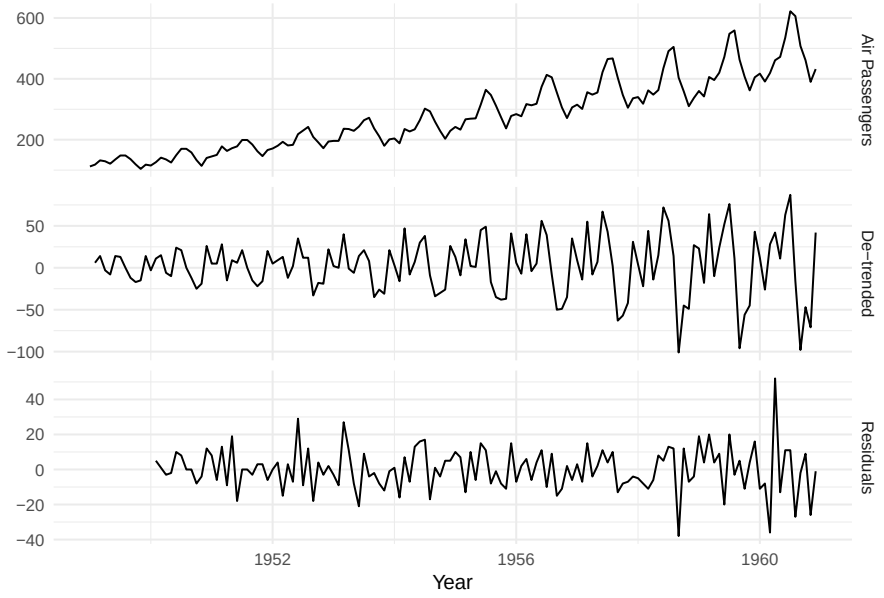


Figure 2.2: Number of international airline passengers per month, from 1949 to 1960. Upper tile shows the raw data, middle tile shows the data de-trended, and the lower tile shows the data de-trended and de-seasonalized.

Athanasopoulos, 2018):

$$y_i = \text{Trend}_i + \text{Seasonal}_i + \text{Cyclic}_i + \text{Residuals}_i,$$

where Trend_i, Seasonal_i, Cyclic_i, and Residuals_i represent the trend, seasonal, cyclic, and residual components of the time series at that point, respectively. This decomposition can also be multiplicative:

$$y_i = \text{Trend}_i \times \text{Seasonal}_i \times \text{Cyclic}_i \times \text{Residuals}_i.$$

Different assumptions lead to an additive or multiplicative decomposition (or a mix of both) of a time series, which are dependent on the problem at hand.

2.1.3 Stationarity

A time series is considered to be stationary if there are no systematic changes in the mean or variance, and if periodic variations have been removed (Chatfield, 2000). This essentially means that the properties of a time series do not depend on the time when the data is observed. It is important to distinguish among

different notions of stationarity. To accomplish this, we follow Chatfield (2000) closely.

A time series is strictly stationary if the joint distribution of $\{y_1, \ldots, y_i\}$ is identical to the joint distribution $\{y_{1+j}, \ldots, y_{i+j}\}$, for all $i, j \in \mathbb{N}$. This means that if we shift the period in which we observe the data, this does not affect the joint distributions.

This definition is often relaxed in practice, giving rise to the notion of second-order stationarity, or weak stationarity. A time series is second-order stationary if it has constant mean, constant variance, and the auto-covariance does not depend on time. Henceforth, when we refer to a time series as stationary, we use the second-order stationarity definition.

Time series can also be regarded as trend-stationary when the mean trend is deterministic. In these scenarios, by estimating and removing the trend, the resulting residuals are stationary.

Tests for Stationarity

Stationarity is an important characteristic of time series data, and several tests were created to verify this. In this section, we outline two of these methods and point the reader towards other widely used ones.

One of the most commonly used methods is the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test (Kwiatkowski et al., 1992). This method tests the null hypothesis that a given time series is stationary in trend, where the alternative is the presence of a unit root. Unit roots are typical sources of non-stationarities in trend. In other words, if a time series comprises a unit root, it is considered non-stationary in trend. However, after removing this component, the residuals are stationary. The KPSS test is typically used for trend inclusion in forecasting models, for example, ARIMA (Auto-Regressive Integrated Moving Average) (Box et al., 2015).

Nason (2013) recently proposed a wavelet spectrum test to estimate whether or not a time series is stationary. Essentially, this test starts by computing an evolutionary wavelet spectral approximation. Then, for each scale of this approximation, the coefficients of the Haar wavelet are computed. Any large Haar coefficient is evidence of a non-stationarity. A hypothesis test is carried out to

assess if a coefficient is large enough to reject the null hypothesis of stationarity. Specifically, multiple hypothesis testing is carried out using a Bonferroni correction and a false discovery rate (Nason, 2013).

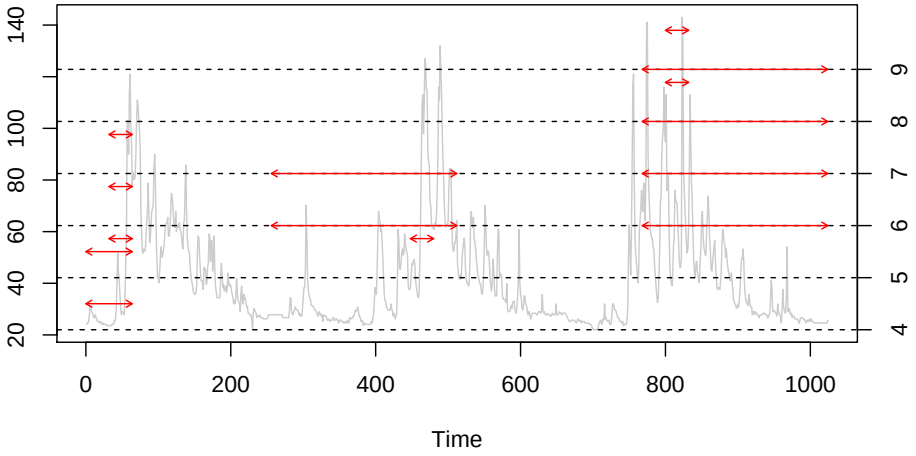


Figure 2.3: Application of the wavelet spectrum test to a non-stationary time series. Each red horizontal arrow denotes a non-stationarity identified by the test.

Figure 2.3 shows an example of the application of the wavelet spectrum test to a non-stationary time series. In the graphic, each red horizontal arrow denotes a non-stationarity found by the test. The left-hand side axis denotes the scale of the time series. The right-hand axis represents the scale of the wavelet periodogram and where the non-stationarities are found. Finally, the length of the arrows denotes the scale of the Haar wavelet coefficient, whose null hypothesis was rejected. For a thorough read on this method, we refer to the work by Nason (2013).

Other well-known methods for testing for stationarity are the following: the Augmented Dickey-Fuller test (Dickey and Fuller, 1981), the Phillips & Perron unit root test (Phillips and Perron, 1988), or the Ljung-Box serial correlation test (Ljung and Box, 1978).

Data Transformations

Time series regular components, such as trend or seasonality, break stationarity. In these cases, there are some transformations we can apply to the time series to make it stationary (Hyndman and Athanasopoulos, 2018). In this section, we outline some widely used transformations: differencing, model fitting, logarithms, and square roots.

Differencing is the process of subtracting the current value of the time series with its previous value:

$$y'_i = y_i - y_{i-1} \tag{2.1}$$

where y'_i is the transformed i-th value of the time series. This approach is typically used to account for a trend component. For example, in Figure 2.2 (middle tile), we difference the original airline passengers data in order to remove trend. Differencing can be applied multiple times to a time series. In particular, the well-known automatic forecasting procedure `auto.arima` (Hyndman et al., 2014) uses the KPSS test described above to estimate the number of differences to make a time series trend-stationary.

We can also use differencing to account for seasonality, by taking the difference between the present value and the previous value from the same season:

$$y'_i = y_i - y_{i-nseason} \tag{2.2}$$

where $nseason$ is the seasonal period. For example, in the lower tile of Figure 2.2, we apply seasonal differencing with $nseason = 12$ since there is a clear monthly pattern in the data.

Another way to account for the trend of a time series is to fit an appropriate model to the data. We can then model the residuals resulting from that fit, thus removing the trend component (Hyndman and Athanasopoulos, 2018).

Sometimes the data shows a non-constant variance, which also breaks stationarity. In these cases, taking the logarithm or square root often helps to stabilise the variance (Croarkin et al., 2006). However, negative data precludes the use of these approaches. In these cases, we can add a suitable constant to make the data positive before applying these transformations. These approaches are particular instances of the Box-Cox transformation (Box and Cox, 1964).

2.1.4 Multiple Variables

So far, we have described time series from a univariate perspective. In this context, only the present and the past values of a time series are available for fitting a given forecasting model. For example, in a univariate scenario, we assume that the future levels of water consumption are essentially dependent on its past values (see Figure 2.1). Often, however, the dynamics of a time series is dependent on additional time series, which are typically referred to as explanatory or predictor variables. In the case of water consumption described before, the future of the time series representing this phenomenon may be dependent on variables other than its past values, for example, weather conditions.

Figure 2.4: Physiological signals of a patient monitored over time

Figure 2.4 shows a multivariate time series. This data represents different physiological signals (HR, SBP, DBP, and MAP) over time collected from a patient being monitored in the intensive care unit of a hospital.

2.2 Forecasting

The main goal behind time series analysis is to predict the future behaviour of the data. This process is commonly referred to as forecasting. Organisations across a wide range of domains rely on forecasting as a decision support tool. For example, Figure 2.1 illustrates the water consumption levels per day of

the year. Water utility systems use short-term water consumption forecasting techniques to plan their operations efficiently. For example, the San Diego Water Department achieved savings of approximately $800.000 US dollars on their first year after introducing a water consumption forecasting system in their short-term planning (Jentgen et al., 2007).

In the last few decades, the research community produced a considerable number of contributions on forecasting methods. Chatfield (2000) refers that these can be split into three groups: subjective, univariate, and multivariate. The first group denotes a qualitative approach that mostly relies on human judgment and domain knowledge, for example, the Delphi method (Murry Jr and Hammons, 1995).

The other two approaches are quantitative. Univariate methods refer to statistical approaches that model future observations of a time series according to its past observations. Multivariate approaches extend univariate ones by considering additional time series that are used as explanatory variables.

The forecasting horizon is another aspect to take into account when addressing these problems. Forecasting methods usually focus on one step ahead forecasting, i.e., the prediction of the next value of a time series (y_{n+1}). Sometimes one is interested in predicting many steps into the future. These tasks are often referred to as multi-step forecasting (Taieb et al., 2012). Higher forecasting horizons typically lead to a more difficult predictive task due to the increased uncertainty (Weigend, 2018).

It is also important to distinguish between point forecasts and prediction intervals (Chatfield, 2000). Point forecast processes represent the intended forecast as a single numeric value. In this thesis, we will focus on this type of forecasts. On the other hand, in some domains, one may be interested in prediction intervals. These represent a lower and upper bound within which the future value is expected to lie. For this type of forecasts, we refer the reader to the work of Chatfield (2000).

2.2.1 Simple Forecasting Models

Several models for time series analysis have been proposed in the literature. These are not only devised to forecast the future behaviour of time series but

also to help understand the underlying structure of the data. In this and the next section, we outline some of the most widely used methods for forecasting.

There are some forecasting models that, although simple, are known to be effective in practice. Here we outline the following ones: the average, naive, and seasonal naive methods (Hyndman and Athanasopoulos, 2018).

As the name implies, the average method predicts the future values of a time series according to the historical mean:

$$\hat{y}_{n+1} = \overline{y} = \frac{\sum_{i=1}^{n} y_i}{n} \tag{2.3}$$

where $\hat{y}_{n+1}$ denotes the prediction of the value of the time series at time $n+1$.

The naive method, also known as the random walk forecast, predicts the next value of the time series according to the last known observation:

$$\hat{y}_{n+1} = y_n \tag{2.4}$$

There is empirical evidence that this method presents a reasonable fit for financial time series data (Kilian and Taylor, 2003).

The seasonal naive model works similarly to the naive method. The difference is that the seasonal naive approach uses the previously known value from the same season of the intended forecast:

$$\hat{y}_{n+1} = y_{n+1-nseason} \tag{2.5}$$

where *nseason* denotes the seasonal period.

2.2.2 ARIMA and Exponential Smoothing

Despite working well in practice, more sophisticated methods than the ones presented above have been proposed in the literature. The ARMA (Auto-Regressive Moving Average) is one of the most commonly used methods to model univariate time series. ARMA(p,q) combines two components: AR(p), and MA(q).

According to the AR(p) model, the value of a given time series, y_n, can be estimated using a linear combination of the p past observations, together with an error term ϵ_n and a constant term c (Box et al., 2015):

$$y_n = c + \sum_{i=1}^{p} \phi_i y_{n-i} + \epsilon_n \tag{2.6}$$

where $\phi_i, \forall\ i \in \{1, \ldots, p\}$ denote the model parameters, and p represents the order of the model. The AR(p) model plays a central part in this thesis since many of the models used in the upcoming chapter are based on this approach. We will return to auto-regressive processes in Section 2.2.3.

The AR(p) model uses the past values of the time series as explanatory variables. Similarly, the MA(q) model uses past errors as explanatory variables:

$$y_n = \mu + \sum_{i=1}^{q} \theta_i \epsilon_{n-i} + \epsilon_n \tag{2.7}$$

where μ denotes the mean of the observations, $\theta_i, \forall\ i \in \{1, \ldots, q\}$ represents the parameters of the models and q denotes the order of the model. Essentially, the method MA(q) models the time series according to random errors that occurred in the past q lags (Chatfield, 2000).

Effectively, the model ARMA(p,q) can be constructed by combining the model AR(p) with the model MA(q):

$$y_n = c + \sum_{i=1}^{p} \phi_i y_{n-i} + \sum_{i=1}^{q} \theta_i \epsilon_{n-i} + \epsilon_n \tag{2.8}$$

The ARMA(p,q) is defined for stationary data. However, many interesting phenomena in the real-world exhibit a non-stationary structure, e.g. time series with trend and seasonality. The ARIMA(p,d,q) model overcomes this limitation by including an integration parameter of order d. Essentially, ARIMA works by applying d differencing transformations to the time series (until it becomes stationary) before applying ARMA(p,q).

The exponential smoothing model (Brown, 1959) is similar to the AR(p) model in the sense that it models the future values of time series using a linear combination of its past observations. In this case, however, exponential smoothing methods produce weighted averages of the past values, where the weight decays exponentially as the observations are older (Hyndman and Athanasopoulos, 2018). For example, in a simple exponential smoothing method, the prediction for y_{n+1} can be defined as follows:

$$y_{n+1} = y_n \zeta_0 + y_{n-1} \zeta_1 + y_{n-2} \zeta_2 + \cdots \tag{2.9}$$

where the $\{\zeta_i\}$ represent the weights of past observations. There are several

types of exponential smoothing methods. For a complete read, we refer to the work by Hyndman and Athanasopoulos (2018).

2.2.3 Auto-Regressive Processes

Auto-regressive approaches are commonly used for time series forecasting. This type of procedures projects a time series into a Euclidean space according to Taken's theorem on time delay embedding (Takens, 1981). Using common terminology in the machine learning literature, a set of observations (x_i, y_i) is constructed (Michalski et al., 2013). In each observation, the value of y_i is modelled based on the past p values before it: $x_i = \{y_{i-1}, y_{i-2}, \ldots, y_{i-p}\}$, where $y_i \in \mathbb{Y} \subset \mathbb{R}$, which represents the vector of values we want to predict, and $x_i \in \mathbb{X} \subset \mathbb{R}^p$ represents the feature vector. The objective is to construct a model for approximating $f : \mathbb{X} \to \mathbb{Y}$, where f denotes the regression function. In other words, the principle behind this approach is to model the conditional distribution of the i-th value of the time series given its p past values: $f(y_i|x_i)$.

In essence, this approach leads to a multiple regression problem. The temporal dependency is modelled by having past observations as explanatory variables. Following this approach, the final representation of the time series is exemplified in the following matrix:

$$Y_{[n,p]} = \left[\begin{array}{ccccc|c} y_1 & y_2 & \cdots & y_{p-1} & y_p & y_{p+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{i-p+1} & y_{i-p+2} & \cdots & y_{i-1} & y_i & y_{i+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{n-p+1} & y_{n-p+2} & \cdots & y_{n-1} & y_n & y_{n+1} \end{array}\right]$$

This auto-regressive approach is at the core of many important forecasting models in the literature, for example, ARIMA (Box et al., 2015)– as we already mentioned in the previous section–, or time-lagged neural networks (Faraway and Chatfield, 1998).

The time delay embedding approach described above involves the determination of p, the embedding dimension. This represents the number of past observations to use, i.e., how far back in time the auto-regressive process should go.

Approaches such as ARIMA typically rely on the partial auto-correlation function to determine the order (number of lags) of the auto-regressive process (p) (Box et al., 2015). The partial auto-correlation function measures the partial correlation of a time series with its lags. The partial part, in this case, means that the correlation is controlled for shorter lags than that under analysis.

Another approach is the False Nearest Neighbours (Kennel et al., 1992), which is based on geometrical construction. This method analyses the behaviour of the nearest neighbours as we increase p. According to Kennel et al. (1992), with a low sub-optimal p many of the nearest neighbours will be false. Then, as we increase p and approach an optimal embedding dimension, those false neighbours disappear. Evolutionary algorithms have also been studied for estimating the embedding dimension, e.g. Lukoseviciute and Ragulskis (2010); Parras-Gutierrez et al. (2014).

2.2.4 Concept Drift

In Section 2.1.3, we introduced the idea of stationarity in time series. As we already mentioned, a time series is stationary if its mean, variance, and auto-correlation structure do not change over time. Time series stationarity is closely related to the phenomenon of concept drift (Schlimmer and Granger, 1986). When sources of non-stationary variation are at play, it is said that concept drift occurs. In other words, concept drift denotes changes in the underlying process generating the time series being observed. This process typically has an impact on the data distribution of the time series, causing it to change.

Types of Concept Drift

There are different types of concept drift. We follow Gama et al. (2014) closely to describe these in some detail. **Abrupt** concept drift occurs when the process generating the data suddenly switches. This type of drift can occur due to, for example, human intervention. **Incremental** and **gradual** concept drift occur when the data distribution changes in a smooth fashion. This arises due to, for example, a trend component in the time series. **Re-occuring** concept drift denotes cyclic changes in the dynamics of the time series. One common phenomenon that causes this type of concept drift is seasonality. In such scenarios,

one can also say that different regimes, or concepts, are causing the underlying time series.

Concept drift is one of the main challenges when learning from time-dependent data. This phenomenon typically happens in dynamically changing environments and complicates the process of learning (Gama et al., 2014). In this context, learning algorithms should be able to cope with the different sources of non-stationary variation. Particularly, according to Gama et al. (2014) predictive models should (1) detect concept drift as soon as possible; and (2) distinguish concept drift from noise, that is, sporadic values that fall outside the typical behaviour of the data. In essence, predictive models should be adaptive to concept drift while robust to noise.

Concept Drift Adaptation

Gama et al. (2014) present a taxonomy for adaptive algorithms that are designed to cope with concept drift. This taxonomy is organized according to different components, namely memory, change detection, learning, and loss estimation. In the interest of conciseness, we will focus on the learning component, which is central to this thesis. We refer the reader to work by Gama et al. (2014) for a complete overview of this taxonomy.

Learning strategies for concept drift adaptation consist of strategies that can forecast the future behaviour of time series and update the predictive model over time. This component can be split into three modules: learning mode, model adaptation, and model management.

Within the learning mode, a predictive model may use a **re-training** or **incremental** strategy. The former consists in regularly (e.g. when a new observation is available) discarding the current model, and re-train a new one from scratch. On the other hand, incremental approaches update the current model using incoming observations.

The model adaptation component can also be split into two types: **blind** adaptation and **informed** adaptation. Essentially, blind adaptation strategies update the predictive models without any explicit detection of concept drift. Incremental algorithms are an example of this approach. Conversely, informed adaptation approaches act on some kind of trigger, for example, an alarm that

concept drift has occurred. Informed adaptation mechanisms are typically followed when using a re-training learning mode.

Finally, model management strategies maintain a pool of different predictive models that make a combined prediction. This type of approaches is commonly known in the literature as ensemble learning (Zhou, 2012). The combined prediction is usually a weighted average of the individual predictions, where the weights reflect the expected performance of the individual models. Ensemble models cope with concept drift by changing these weights over time. These are known in the literature as dynamic ensembles (Kuncheva, 2004a). In the next section, we will delve further into ensemble approaches, focusing on methods used to estimate the weights of each model composing an ensemble.

2.2.5 Ensemble Methods

Despite the wide range of contributions to the forecasting literature, it is widely accepted that there is no method that is applicable to all time series (Chatfield, 2000). This statement is corroborated by experiments performed by Aiolfi and Timmermann (2006) relative to the performance of forecasting models over a time series. They reported systematic evidence that some forecasting models have varying relative performance over time and that some forecasting models are persistently good (or bad) throughout the time series.

The idea that all predictive models have some limitations has been extensively explored in the literature of ensemble learning methods. Several empirical and theoretical studies have shown that combining several individual models leads to better predictive performance (Ueda and Nakano, 1996; Breiman, 1996; Dietterich et al., 2000). Particularly in forecasting, combining different models is a well-studied topic (Bates and Granger, 1969; Armstrong, 1989). For example, Clemen (1989) presented an annotated bibliography comprising over 200 approaches.

Still, it is not clear how we should combine the predictions of a set of models. There are two groups of approaches devised to accomplish this task: static methods and dynamic methods. Static methods assign a weight to each model in the ensemble, which is constant for all observations. The most common static approach in the literature is the simple average (arithmetic mean) of the

predictions of the available models. In this particular approach, all individual models have equal weights.

The main limitation behind the application of static methods in time-dependent domains is that these may fail to capture the evolving dynamics of time series and cope with concept drift. As we described before, there is compelling evidence in the literature that not all models will perform equally well at any given prediction point (Aiolfi and Timmermann, 2006). In these scenarios, it is more common to adopt dynamic methods, in which the weights assigned to an individual model vary over time. This type of approach falls within the scope of online learning. Online learning denotes a learning paradigm in which a predictive model is updated when a new observation, or set of observations, becomes available (Littlestone and Warmuth, 1994). We will overview some dynamic methods used to combine a set of forecasting models. We split these into three dimensions: windowing approaches, regret minimisation approaches, and meta-learning approaches.

Windowing Approaches

Determining the weights of different predictive models at each time step is a difficult task, and several methods have been proposed to accomplish this. Particularly in forecasting, the simple average of the available experts (equal weights) has been shown to be a robust combination method (Clemen and Winkler, 1986) (`Simple`). Its competitive performance relative to approaches using estimated weights is known in the forecasting literature as the "forecasting combination puzzle" (Genre et al., 2013). Using the median value of the available predictions has also been explored (Marcellino, 2004). Nonetheless, more sophisticated approaches have been proposed.

Simple averages are sometimes complemented with model selection before aggregation, also known as trimmed means (`SimpleTrim`). For example, Jose and Winkler (2008) propose trimming a percentage of the worst forecasters in past data and average the output of the remaining experts.

One of the most common and successful approaches to combine predictive models in time dependent data is to weight them according to their performance. Typically the performance is determined on a window of recent data,

or by using some other forgetting mechanism that promotes the importance of recency. The idea is that recent observations are more similar to the one we intend to predict, and thus, they are considered more relevant. For example, Newbold and Granger (1974) use this approach to combine forecasting models (`WindowLoss`). More recently, van Rijn et al. (2015) proposed a method for data streams classification dubbed `Blast`. As opposed to fusing experts, they select the best recent performing one to classify the next observation. Bunn (1975) proposes an approach based on out-performance, where the weights of experts are determined by the number of times they have been the best in the past.

`AEC` (Adaptive Ensemble Combination) is a method for adaptively combining a set of forecasters (Sánchez, 2008). It uses an exponential re-weighting strategy to combine forecasters according to their past performance, including a forgetting factor to give more importance to recent values. Timmermann (2008) argues that models have only short-lived periods of predictability for the prediction of stock returns. He proposes an adaptive combination based on the recent R^2 of forecasters. If all models have a low explained variance (low R^2) in the recent observations, then the forecast is set to the mean value of those observations. Otherwise, the experts are combined by averaging their predictions with the arithmetic mean (`ERP`).

Regret Minimisation

Several strategies have been proposed for aggregating the output of forecasting models, which are based on the idea of regret minimisation. Regret is the average error suffered with respect to the best we could have obtained. Several approaches dynamically combine a set of predictive models by optimising this metric, namely the exponentially weighted average (`EWA`) (Vovk, 1990; Littlestone and Warmuth, 1994), the polynomially weighted average (Cesa-Bianchi and Lugosi, 2003) (`MLpol`), or the fixed share aggregation (Herbster and Warmuth, 1998) (`FixedShare`). For a thorough review of these methods and their theoretical properties, we refer to the seminal work by Cesa-Bianchi and Lugosi (2006). Zinkevich (2003) proposed an online convex programming approach based on gradient descent that also guarantees regret bounds (`OGD`).

Combining by Learning

Meta-learning provides a way of modelling the learning process of a learning algorithm (Brazdil et al., 2008). Several methods use this approach to improve the combination or selection of models (Wolpert, 1992; Todorovski and Džeroski, 2003). We overview some meta-learning approaches that have been designed to combine or select a set of predictive models in time-dependent domains. Some of these approaches are static, for example, stacking, but in practice are often used in time-dependent domains.

A popular and successful approach for dynamically combining experts is to apply multiple regression on the output of the set of forecasting models. For example, Gaillard and Goude (2015) describe a setup in which Ridge regression is used to aggregate experts by minimising the L2-regularised least-squares (Hoerl and Kennard, 1970; Marcellino, 2004). The idea behind these approaches is similar to stacking (Wolpert, 1992), a widely used approach to combine predictive models (`Stacking`).

Rossi et al. (2014) present `MetaStream` for the dynamic selection of regression models in a data stream environment. `MetaStream` works by having a meta-learning model that periodically selects the most appropriate regression method to be used in the next few observations.

Gama and Kosina (2014) present a meta-learning approach designed to cope with concept drift in data streams classification problems. The system proposed by the authors is focused on re-occurring drift. It can be split into two layers: a base layer, where a predictive model is devised to solve the original problem; and a meta layer, which manages the learning process. When concept drift is detected, the meta layer decides whether to train a new model using recent observations or to re-activate a base model trained previously.

2.2.6 Ensemble Diversity

One of the key aspects to take into account when building an ensemble is the diversity among the individual models comprising it. Diversity is related to the degree of disagreement within the ensemble (Brown et al., 2005a). While the generalisation error of a single predictive model can be decomposed using

the bias-variance decomposition (Geman et al., 1992), a regression ensemble can be understood using the bias-variance-covariance decomposition (Ueda and Nakano, 1996). According to the bias-variance-covariance decomposition, the expected mean squared error (MSE) of an ensemble can be split into the following terms:

$$\text{MSE} = \overline{bias}^2 + \overline{var}\frac{1}{s} + (1 + \frac{1}{s})\overline{covar} + \sigma^2 \tag{2.10}$$

where $\overline{bias}^2$, $\overline{var}$, and $\overline{covar}$ represent the average bias, average variance, and average covariance of the s models comprising the ensemble, respectively. σ^2 is a constant irreducible term representing the variance of the noise.

As Brown et al. (2005a) point out, besides the bias and variance of the individual models, the generalisation error of a regression ensemble directly depends on the covariance term between them. Larger values in the covariance term (i.e. lower diversity or disagreement) lead to a greater error. This result shows the importance of encouraging diversity in ensemble methods.

There exists a wide range of methods for encouraging diversity in ensemble methods. Typically, approaches are based on input manipulation (e.g. bagging (Breiman, 1996)), output manipulation (e.g. Error-Correcting Output Coding (Dietterich and Bakiri, 1991)), or the usage of different learning algorithms to build the individual predictive models. Using learning algorithms with distinct inductive biases hopefully leads to individual models occupying different parts of the hypothesis space. We refer to the survey by Brown et al. (2005b) for a comprehensive read on diversity creation approaches. Particularly in forecasting, Oliveira and Torgo (2014) propose a method for encouraging diversity in a bagging ensemble. Essentially, the decision trees composing the ensemble are trained according to an auto-regressive of varying size, i.e. using different values for the parameter p.

2.3 Evaluating Forecasting Models

Performance estimation denotes a task of estimating the loss that a predictive model will incur on unseen data. These procedures are part of the pipeline in every machine learning project and are used for assessing the overall generalisation ability of models. For independent and identically distributed data, one of the

most common approaches is cross-validation. However, the dependency among observations in time series raises some caveats about the most appropriate way to estimate performance in these data sets.

In general, performance estimation methods for time series forecasting tasks are designed to cope with the dependence between observations. This is typically accomplished by having a model tested on observations which occur in the future relative to the ones used for training. These include the out-of-sample testing as well as variants of the cross-validation method.

2.3.1 Out-of-sample

When using out-of-sample (OOS) performance estimation procedures, a time series is split into two parts: an initial fit period in which a model is trained, and a testing period held out for estimating the loss of that model. This simple approach (`Holdout`) is depicted in Figure 2.5. However, within this type of procedure, one can adopt different strategies regarding training/testing split point, growing or sliding window settings, and eventual update of the models. In order to produce a robust estimate of predictive performance, Tashman (2000) recommends employing these strategies in multiple test periods. One might create different sub-samples according to, for example, business cycles (Fildes, 1989). For a more general setting, one can also adopt a randomised approach. This is similar to random sub-sampling (or repeated holdout) in the sense that they consist of repeating a learning plus testing cycle several times using different, but possibly overlapping data samples (`Rep-Holdout`). This idea is illustrated in Figure 2.6, where one (out of *nreps*) iteration of a repeated holdout is shown. A point a is randomly chosen from the available window (constrained by the training and testing sizes) of a time series Y. This point then marks the end of the training set and the start of the testing set.

2.3.2 Prequential

OOS approaches are similar to prequential or interleaved-test-then-train evaluation (Bifet and Kirkby, 2009, Chapter 2.2). Prequential is typically used in data streams mining. The idea is that each observation is first used to test the model and then to train the model. This can be applied in blocks of sequen-

Figure 2.5: Simple out-of-sample procedure: an initial part of the available observations are used for fitting a predictive model. The last part of the data is held out, where the predictive model is tested.

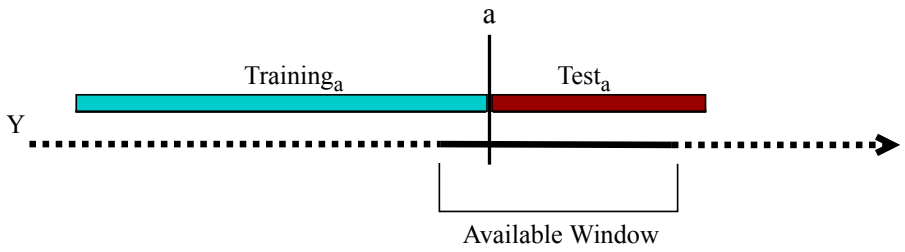


Figure 2.6: Example of one iteration of the repeated holdout procedure. A point a is chosen from the available window. Then, a previous part of observations are used for training, while a subsequent part of observations are used for testing.

tial instances (Modha and Masry, 1998). In the initial iteration, only the first two blocks are used, the first for training and the second for testing. In the next iteration, the second block is merged with the first, and the third block is used for testing. This procedure continues until all blocks are used for testing (`Preq-Bls`). This procedure is exemplified on the left side of Figure 2.7, in which the data is split into five blocks.

A variant of this idea is illustrated in the middle scheme of Figure 2.7. Instead of merging the blocks after each iteration (growing window), one can forget the older blocks in a sliding window fashion (`Preq-Sld-Bls`). This idea is typically adopted when past data becomes deprecated, which is common in non-stationary environments. Another variant of the prequential approach is represented on the right side of Figure 2.7. This illustrates a prequential approach applied in blocks, where a gap block is introduced (`Preq-Bls-Gap`). The rationale behind this idea is to increase the independence between training

and test sets.

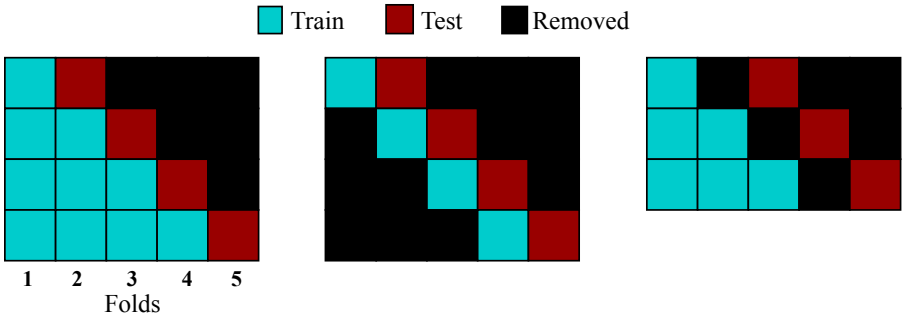


Figure 2.7: Variants of the prequential approach applied in blocks for performance estimation. This strategy can be applied using a growing window (left, right), or a sliding window (middle). One can also introduce a gap between the training and test sets (right).

2.3.3 Cross-validation

Some variants of K-fold cross-validation specially designed for dependent data, such as time series, have been proposed (Arlot et al., 2010). The typical approach when using K-fold cross-validation is to randomly shuffle the data and split it into K equally-sized folds or blocks. Each fold is a subset of the data comprising n/K randomly assigned observations, where n is the number of observations. After splitting the data into K folds, each fold is iteratively picked for testing. A model is trained on $K-1$ folds, and its loss is estimated on the left out fold. The initial random shuffle of observations before splitting into different blocks is not intrinsic to cross-validation (Geisser, 1975). Notwithstanding, the random shuffling is a common practice among data science professionals. This approach to cross-validation is illustrated on the left side of Figure 2.8.

Theoretical problems arise by applying this technique directly to time series data. The dependency between observations is not taken into account since cross-validation assumes that the values of the time series are i.i.d.. During the estimation of predictive performance, a model applied in a cross-validation procedure usually ends up using data from the future to predict past instances, breaking the natural order of observations. This often leads to overly optimistic

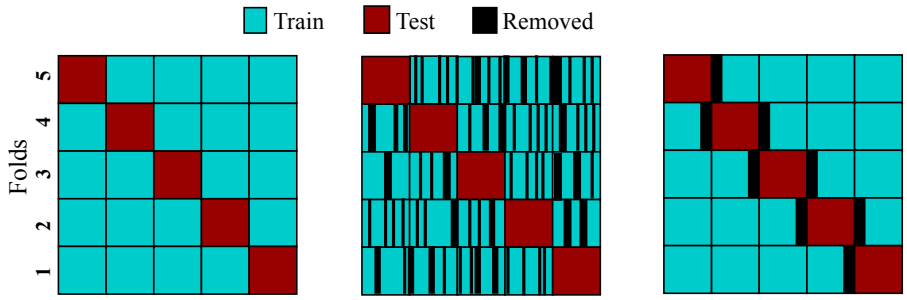


Figure 2.8: Variants of cross-validation estimation procedures

estimations and consequently, poor generalisation ability of models on new observations. For example, prior work has shown that cross-validation yields poor estimations for the task of choosing the bandwidth of a kernel estimator in correlated data (Hart and Wehrly, 1986). To overcome this issue and approximate independence between the training and test sets, several methods have been proposed as variants of this procedure.

The Blocked Cross-Validation (Snijders, 1988) (`CV-Bl`) procedure is similar to the standard form described above. The difference is that there is no initial random shuffling of observations. In time series, this renders K blocks of contiguous observations. The natural order of observations is kept within each block but broken across them. This approach to cross-validation is also illustrated on the left side of Figure 2.8. Since the random shuffle of observations is not being illustrated, the figure for `CV-Bl` is identical to the one shown for `CV`.

The modified cross-validation procedure (McQuarrie and Tsai, 1998), which we denote as `CV-Mod`, works by removing observations from the training set that are correlated with the test set. The data is initially randomly shuffled and split into K equally-sized folds similarly to K-fold cross-validation. Afterwards, observations from the training set within a certain temporal range of the observations of the test set are removed. This ensures independence between the training and test sets. However, when a significant amount of observations are removed from training, this may lead to model under-fit. This approach is also described as non-dependent cross-validation (Bergmeir and Benítez, 2012). The graph in the middle of Figure 2.8 illustrates this approach.

The hv-Blocked Cross-Validation (`CV-hvBl`) proposed by Racine (2000) extends blocked cross-validation to further increase the independence among observations. Specifically, besides blocking the observations in each fold, which means there is no initial randomly shuffle of observations, it also removes adjacent observations between the training and test sets. Effectively, this creates a gap between both sets. This idea is depicted on the right side of Figure 2.8.

2.3.4 Performance Estimation Using Other Non-i.i.d. Data

The problem of performance estimation has also been under research in different scenarios where the observations are somehow dependent (non-i.i.d.).

Spatio-Temporal Data

Geo-referenced time series are becoming more prevalent due to the increase of data collection from sensor networks. In these scenarios, the most appropriate estimation procedure is not obvious as spatio-temporal dependencies are at play. Oliveira et al. (2018) presented an extensive empirical study of performance estimation for forecasting problems with spatio-temporal time series.

Data Streams Mining

Data streams mining is concerned with predictive models that evolve continuously over time in response to concept drift (Gama et al., 2014). Gama et al. (2013) provide a thorough work regarding the evaluation of predictive models for data streams mining. The authors defend the usage of the prequential estimator with a forgetting mechanism, such as a fading factor or a sliding window.

This work is related to performance estimation in time series in the sense that it deals with time-dependent data. The paradigm of data streams mining is in line with sequential analysis (Wald, 1973). As such, the assumption is that the sample size is not fixed in advance, and predictive models are evaluated as observations are collected. In a more classical setting, the objective is, given a time series data set, estimate the loss that a predictive model will incur on unseen observations future to that data set.

2.3.5 Evaluation Metrics

Besides the estimation procedure, in order to quantify the predictive performance of a predictive model, one needs an evaluation metric. In the case of forecasting models, one typically measures the difference between the predicted values provided by the model and the values observed. Several metrics have been devised to accomplish this; for example, the root mean squared error (RMSE), which is computed as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{2.11}$$

Another widely used metric is the mean absolute error:

$$\text{MAE} = \frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{n} \tag{2.12}$$

Hyndman and Koehler (2006) presented the metric mean absolute scaled error (MASE) for evaluating the predictive performance of forecasting models. MASE is computed as follows:

$$\text{MASE} = \frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{\text{Loss}_{naive}} \tag{2.13}$$

where Loss_{naive} represents the average loss of the naive method in the training set used to build the respective forecasting model.

2.4 Activity Monitoring

In the previous subsections of this chapter, we described the problem of forecasting, i.e. the process of predicting the future values of a time series given its history of observations. As we mentioned, this problem is important in several application domains. These domains are concerned with scenarios where professionals rely on periodic short-term estimates of the future behaviour of time series to actively manage organisations in a better way. Often, however, our goal is to anticipate specific events that may occur in a time series. In broad terms, an event consists in a rare behaviour shown by a given time series, such as a sequence of consecutive values above a certain key threshold, which is disruptive in the respective domain of application. In these cases, the goal is not to accurately predict all the values of the upcoming observations, or whether the time

series goes up or down (although these could be part of the solution). Rather, the objective is to predict in a *timely manner* that a specific and typically rare event of interest is about to happen. This task is known in the literature as activity monitoring (Fawcett and Provost, 1999).

2.4.1 Anticipating Interesting Events

Activity monitoring denotes a problem which involves tracking a given activity over time. The goal is to detect, as soon as possible, anomalous and possibly interesting events requiring action (Fawcett and Provost, 1999). Many real-world problems can be framed as activity monitoring predictive tasks.

Griffin and Moorman (2001) found evidence that babies diagnosed with sepsis show abnormal heartbeats in the twenty-four hours before the diagnostic. In this scenario, monitoring the heart rate of babies, and predicting early in time this type of anomalies (abnormal heartbeats) may lead to more efficient diagnostics and an improvement in the healthcare of infants. Note that the goal is not to forecast the values of the time series representing the heart rate, or to classify each point in time as normal or abnormal. The goal is the timely prediction of impending abnormal heartbeats, which may lead to sepsis.

Another example is the case of energy production from renewable sources, such as the wind or the sun. Sometimes there are sudden and unexpected changes in wind speed or solar radiation, which significantly affect the energy output derived from these sources. If such sudden changes are downwards (e.g. lower solar radiation), operators need to take preemptive actions to guarantee enough supply of energy to support the grid. An activity monitoring system can be used in this scenario to support the decision making of operators and anticipate unforeseen scenarios more efficiently.

Sometimes the event of interest can only be captured after it has started, for example, in fraud scenarios. When a cell phone is compromised by fraudsters, the goal is to detect such fraud as soon as possible in order to minimise costs. Walters and Wilkinson (1994) report that telecom frauds in the USA cost hundreds of millions of dollars every year.

These examples illustrate the need for efficient methods that are able to anticipate scenarios and detect anomalies in a timely manner. The ubiquity of

computing applications is a strong motivation for studying this problem. Other examples range from predictive maintenance (Ribeiro et al., 2016a), where early detection of malfunctions enhances the operational efficiency of machines; to behavioural monitoring or assisting systems in healthcare informatics (Roychoudhury et al., 2015).

2.4.2 Problem Definition

In order to formalise the activity monitoring predictive task, we follow Fawcett and Provost (1999) closely. We will resort to an example from the healthcare domain as a motivating example. Activity monitoring typically involves the tracking of a set of entities, which are represented as time series. The specific event of interest may occur in each of these entities. Suppose that we are monitoring a set of patients assigned to the intensive care unit of a hospital. In this scenario, each patient denotes an entity, which is being monitored for a potential health crisis.

Let $\mathcal{D}$ denote a set of entities $\mathcal{D} = \{D_1, \ldots, D_{|\mathcal{D}|}\}$, where $|\mathcal{D}|$ represents the size of this set. Each $D_i \in \mathcal{D}$ denotes a time series $D_i = \{d_{i,1}, d_{i,2}, d_{i,n_i}\}$, where n_i represents the number of observations for entity D_i. The number of observations n_i does not have to be equal to n_k for any two $D_i, D_k \in D$. Finally, each $d_{i,j}$ such that $i \in \{1, \ldots, |\mathcal{D}|\}$ and $j \in \{1, \ldots, n_i\}$ denotes an observation in a given point in time about the entity D_i being monitored. Going back to our example, $d_{i,j}$ may describe a set of physiological signals captured from patient D_i at time step j. This example was illustrated in Figure 2.4, which was shown in Section 2.1.4. This graphic denotes a patient D_i, where each $d_{i,j}$ represents four different physiological signals. In the previous subsections we used the letter Y to denote a numeric and univariate time series. For defining the problem of activity monitoring we use the letter D to this effect. The difference is that D_i is part of a set $\mathcal{D}$ of related time series, and it is not restricted to be either numeric or univariate. Therefore, we use a different notation in the interest of clarity. In our example, each $d_{i,j}$ denotes a vector representing the physiological signals in the j-th time step. Thus, in this case, each D_i is a multivariate time series. If there was a single physiological signal available, D_i would be univariate.

Suppose that our goal is to detect early in time any health crisis that might

occur to a patient. The idea behind activity monitoring is to use the information from D_i (in this case, physiological signals tracked over time) in order to launch alarms about these potential events. These interesting events are also described in the literature as positive activity (Fawcett and Provost, 1999).

2.4.3 Challenges and Relation to Anomaly Detection

The positive activity of each entity (if any) $D_i \in \mathcal{D}$ is usually a small part of the complete activity of that entity. In other words, interesting events are typically rare, and activity monitoring usually represents an imbalanced learning problem (Branco et al., 2016a). In this context, activity monitoring is related to anomaly detection predictive tasks (Chandola et al., 2009). In both cases, the goal is to discover observations which are a small part of the data space. The key distinguishing factor is that activity monitoring is tailored for time-dependent data. The goal of anomaly detection is to classify each observation as normal or abnormal. Conversely, in activity monitoring, the goal is to identify in a timely manner that abnormal behaviour is imminent in a given entity.

Besides the typical rarity of the events of interest, there is another important challenge behind activity monitoring tasks, which is the timeliness of alarms. There should be an appropriate time interval between the time an alarm is issued and the time the event of interest happens (Weiss and Hirsh, 1998). The necessary lead time is dependent on the domain of application and it is important to allow professionals to decide the best course of action. The larger these warning time intervals become, the further into the future we need to forecast interesting events, which typically leads to a more difficult task (Weiss and Hirsh, 1998). Alarming earlier is more useful, and this usefulness ceases when the event becomes apparent.

It should be noted that activity monitoring is different from concept drift detection. The latter is concerned with the detection of changes in the regimes governing the process generating the observations. When regimes change, the distribution of observations typically also changes accordingly. On the other hand, an anomaly represents those observations which deviate significantly from the typical behaviour, where typical behaviour is characterised by the current underlying regime.

Figure 2.9 illustrates this point. It shows the average, and respective standard deviation of global solar radiation per day of the year in Tennessee, USA (data collected by the Oak Ridge National Laboratory (Maxey and Andreas, 2007)). This data shows a strong seasonality, in which the expected level of solar radiation in the summertime is greater than in wintertime. In this context, a day with a total of 2000 watts per square meter of solar radiation would be a normal day in the first days of the year. However, it would present a clear anomaly if it was observed during summertime (e.g. day 180).

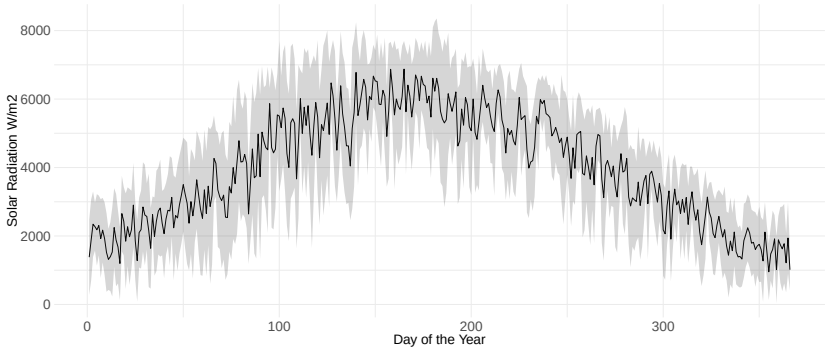


Figure 2.9: Mean and standard deviation of solar radiation per day of the year (in watts per square meter) in Tennessee, USA

2.4.4 Modelling Approaches

According to Fawcett and Provost (1999), there are two classes of methods for activity monitoring:

profiling In a profiling strategy, a model is constructed using only the normal activity of the data, without reference to abnormal cases. Consequently, an alarm is triggered if the current activity deviates significantly from normal activity. This approach may be useful in complex time-dependent data where anomalies do not have a well-defined concept. For example, fraud attempts often occur in different manners. Effectively, by modelling only normal activity, one is apt to detect different types of anomalies, including the ones unknown hitherto.

discriminating A discriminating method constructs a model about anomalies

with respect to the normal activity, handling the problem as a classification one. A system then uses a model to examine the time series and look for anomalies. In this scenario, the recent past dynamics of the data are used as predictor variables. The target variable denotes whether the event of interest occurs.

On top of these two classes, profiling and discriminating, there is a possible second distinction between approaches: uniform or individual. In uniform approaches, a model is built using information from all $D_i \in \mathcal{D}$. On the other hand, individual approaches build a specific model for each D_i. The most appropriate solution is domain-dependent. For example, if each D_i comprises some idiosyncratic signal that should be modelled, it may be worthwhile to use an individual approach.

State of the Art Methods

In this section, we present a small number of methods designed for activity monitoring. Although the list is not comprehensive, to the best of our knowledge, it comprises a reasonable representation of the type of approaches used to tackle activity monitoring problems. One of the pioneering works in activity monitoring is due to Fawcett and Provost (1999), where they formalise the activity monitoring predictive task. They also describe a model, dubbed `DC-1` (Fawcett and Provost, 1997), which was used for detecting fraud in telecom data, or to monitor new stories. `DC-1` works in three main steps. Initially, a set of rules is created, which are designed to indicate possible fraudulent behaviour. Afterwards, an individual profiling approach is carried out as follows. The rules obtained before are used to build profiling monitors for each entity. These are used to model the typical behaviour of the respective entity relative to a rule. In effect, the system can be used to quantify how far the entity deviates from normal activity. The final step of `DC-1` is a weighting mechanism that maximises the performance of the system.

Weiss and Hirsh (1998) presented a method called `Timeweaver` to predict rare events from a sequence of events. They applied the method to predict telecom equipment failure using a set of alarm messages. `Timeweaver` works by using a genetic algorithm to identify prediction patterns from the data. Af-

terwards, a greedy algorithm used the generated patterns to create rules that distinguish normal events from anomalous events.

Other examples of approaches in the literature are the works by Salvador et al. (2004); Vilalta and Ma (2002); Ghosh et al. (2016). Although different methods show some variations in their approaches, the basic idea is similar. A system models whether or not a rare event started recently, or is starting shortly, according to the recent activity of the entity being monitored.

Actionable Forecasting

In some domains of application, the event of interest is defined according to the observed values of a certain numeric variable. For example, a common event of interest in the intensive care unit of hospitals is acute hypotension. An acute hypotensive episode (AHE) is defined as a 30-minute interval in which 90% of the values of mean arterial blood pressure are below 60 millimetres of mercury (Ghosh et al., 2016). The most common approach is to model AHEs as a binary classification problem. The recent values of several physiological signals are used to create the predictor variables. The target variable denotes whether or not there is an impending AHE.

An alternative formulation is to model the underlying numeric variable. Using the same predictor variables as a classification model, a regression algorithm can be used to forecast the future values of the numeric variable used to define the event. A subsequent deterministic function is used to map the forecasted value(s) into a decision (i.e. whether or not the event of interest occurs). This regression-based approach is designated as actionable forecasting (Baía, 2015). In our example, Rocha et al. (2011) take this approach to predict impending AHE. They create a model for forecasting the future values of mean arterial blood pressure, which is the variable used to define an AHE. The decision process about whether or not there is an impending AHE is carried out according to these predicted values.

Baía and Torgo (2017) present a study comparing the two approaches, i.e. a classification-based approach with a regression-based approach, for deciding the correct trading actions in the context of financial trading. In the first approach, a classification model predicts the correct course of action, buy an asset, hold

it, or sell it. In the second approach, a forecasting model first predicts the price variation of an asset. A subsequent deterministic function is applied to decide the correct course of action.

Evaluation

The evaluation of predictive models for activity monitoring tasks is typically constrained by two issues: class imbalance, and time-dependency among consecutive observations. As we mentioned, events of interest are typically rare. This issue has an impact on the evaluation of predictive models (Branco et al., 2016a). Moreover, missing an event of interest does not have the same cost as issuing a false alarm. Recalling the hospital example, failing to anticipate a health crisis in a patient is more costly than launching a false alarm. Cost-sensitive models are often used to cope with this problem (Chan and Stolfo, 1998).

The evaluation of activity monitoring problems also needs to take into account the timeliness of alarms. Suppose that an alarm is issued about an event. A second alarm about the first one adds no information. Moreover, the concept of *true negative* (Flach, 2019) is not well defined in these problems. Because of the continuity of time, there may be "infinitely many *true negatives*" (Fawcett and Provost, 1999).

In order to cope with these issues, Fawcett and Provost (1999) proposed to use the AMOC (Activity Monitoring Operating Characteristic) curve as an evaluation framework for these problems. This approach is similar to ROC (Receiver Operating Characteristic) (Provost et al., 1997) but tailored for time-dependent domains.

Weiss and Hirsh (1998) extended the classical precision and recall metrics to evaluate activity monitoring models. Similarly to AMOC, these metrics, reduced precision and event recall, were designed to accommodate to the time-dependency among observations. In Chapter 6, we will describe these metrics in more detail.

2.4.5 Related Early Decision Systems

The need for early predictions is also important in other predictive tasks which are related to activity monitoring. Time series classification is a well-studied topic, for example, in data streams mining (Bifet and Kirkby, 2009). However, traditional time series classification methods are inflexible for early classification. Typically, a method is trained on the full length of the time series, and the prediction is also made at that time point. Therefore, the main limitation of such methods is that they ignore the sequential nature of data, and the importance of *early* classification (Fawcett and Provost, 1999). The earliness component of classifiers for time series is important so that professionals and decision-makers can take pro-active measures and timely decisions. To overcome this limitation, several models for early classification of time series have been proposed. Some examples are the works of He et al. (2015), Antonucci et al. (2015), or Xing et al. (2011), to name a few. Another example of a type of early decision systems is human motion recognition (Kuehne et al., 2011). This task is fundamental for surveillance systems or human-computer interactive systems.

2.5 Final Remarks

The dynamic and complex structure of time series makes them one of the most researched topics in the literature of knowledge discovery from data. Many models have been proposed that try to explain how the past affects the future. However, predicting the future behaviour of time series is a challenging task.

In this chapter, we presented an overview of the literature on the topic of this thesis. We revised previous work, with a particular emphasis on the topics of dynamic forecast combination, evaluation of forecasting models, and activity monitoring. In the next chapters, we will explore these topics in more detail. We will identify the main limitations of the current state of the art, and propose novel methods for time series forecasting. The ultimate goal behind our work is to help organisations leverage past data to make data-driven decisions regarding the future.

Part II

Forecasting

Chapter 3

Evaluating Forecasting Models

The main goal in this part of the dissertation is to develop new machine learning methods for tackling time series forecasting predictive tasks. To accomplish this, we need a methodology for assessing the predictive performance of these new methods and to compare them with the state of the art approaches. In this chapter, we address the task of estimating the predictive performance of forecasting models.

3.1 Introduction

Machine learning plays an increasingly important role in science and technology, and performance estimation is part of any machine learning project pipeline. This task is related to the process of using the available data to estimate the loss that a predictive model will incur on unseen data. Machine learning practitioners typically use these methods for model selection, hyper-parameter tuning and assessing the overall generalisation ability of the models. In effect, obtaining reliable estimates of the performance of models is a critical issue on all predictive analytics tasks.

Choosing a performance estimation method often depends on the data one is modelling. For example, when one can assume independence and identical

distribution (i.i.d.) among observations, cross-validation (Geisser, 1975) is typically the most appropriate method. This is mainly due to its efficient use of data (Arlot et al., 2010). However, there are some issues when the observations in the data are dependent, such as time series. These dependencies raise some caveats about using standard cross-validation in such data. Notwithstanding, there are particular time series settings in which variants of this approach can be used, such as in stationary or small-sized data sets where the efficient use of all the data by cross-validation is beneficial (Bergmeir et al., 2018).

In this chapter, we present a comparative study of different performance estimation methods for time series forecasting tasks. Several strategies have been proposed in the literature, and currently, there is no consensual approach. We applied different methods in two case studies. One is comprised of 62 real-world time series with potential non-stationarities, and the other is a stationary synthetic environment (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018). The estimation methods under comparison can be broadly split into the following two classes:

- Out-of-sample (OOS): These methods have been traditionally used to estimate predictive performance in time-dependent data. Essentially, out-of-sample methods hold out the last part of the time series for testing. Although these approaches do not make complete use of the available data, they preserve the temporal order of observations, emulating a realistic scenario. This property may be important to cope with the dependency among observations and account for the potential temporal correlation between the consecutive values of the time series.

- Cross-validation (CVAL): These approaches make more efficient use of the available data, which is beneficial in some settings (Bergmeir et al., 2018). They assume that observations are i.i.d., though some strategies have been proposed to circumvent this requirement. These methods have been shown to be able to provide a better estimation ability relative to out-of-sample approaches in some time series scenarios (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018).

A key characteristic that distinguishes these two types of approaches is that

OOS methods always preserve the temporal order of observations meaning that a model is never tested on past data. The objective of this study is to address the following research question: How do OOS estimation methods compare to CVAL approaches in terms of performance estimation ability for different types of time series data?

This chapter is structured as follows. Related work on performance estimation for time series forecasting tasks, which motivated our work, is overviewed in Section 3.2. Materials and methods are described in Section 3.3, including the predictive task, time series data sets, performance estimation methodology, and experimental design. The results of the experiments are reported in Section 3.4. A discussion of our results is carried out in Section 3.5. Finally, the conclusions of our empirical study are provided in Section 3.6. The experiments carried out in this chapter are available online[1].

3.2 Related Work

3.2.1 Methods for Evaluating Forecasting Models

As we mentioned, performance estimation methods for time series forecasting tasks are designed to cope with the time dependence among observations. This is typically accomplished by having a forecasting model tested on observations which occur after the ones used for training that model. Notwithstanding, variants of cross-validation have been proposed and applied to these scenarios. In the Section 2.3 of this thesis, we reviewed several methods for estimating the predictive performance of forecasting models. Within each of these approaches, one can use different variants, for example, prequential with a growing window or with a sliding window, or cross-validation with or without random shuffle of observations.

3.2.2 On the Usefulness of Cross-validation

While the literature in time series analysis typically adopts an OOS approach, recently there has been some work on the usefulness of CVAL procedures for

[1] At `https://github.com/vcerqueira/performance_estimation`

evaluating time series forecasting models. Bergmeir and Benítez (2012) present a comparative study of estimation procedures using stationary time series. Their empirical results show evidence that in such conditions, cross-validation procedures yield more accurate estimates than an OOS approach. Despite the theoretical issue of applying standard cross-validation, they found no practical problems in their experiments. Notwithstanding, the blocked cross-validation (denoted in this work as `CV-Bl`) is suggested for performance estimation when the time series is stationary.

Bergmeir et al. (2014) extended their previous work for directional time series forecasting tasks. These tasks are related to predicting the direction (upward or downward) of the time series. The results from their experiments suggest that the `hv-Blocked CV` procedure provides more accurate estimates than the standard out-of-sample approach. These were obtained by applying the methods on stationary time series.

Finally, Bergmeir et al. (2018) present a simulation study comparing standard cross-validation (`CV`) to the classical OOS evaluation (`Holdout`). They used three data generating processes and performed 1000 Monte Carlo trials in each of them. For each trial and generating process, a stationary time series with 200 values was created. The results from the simulation suggest that cross-validation systematically yields more accurate estimates, provided that the model is correctly specified.

Despite the results provided by these previous works, we argue that they are limited in two ways. First, the used experimental procedure is biased towards cross-validation approaches. While these produce several error estimates (one for each fold), the OOS approach is evaluated in a one-shot estimation, where the last part of the time series is withheld for testing. OOS methods can be applied in several windows for more robust estimates, as recommended by Tashman (2000). By using a single origin, one is prone to particular issues related to that origin.

Second, the results are based on stationary time series, most of them artificial. Time series stationarity is equivalent to identical distribution in the terminology of more traditional predictive tasks. Hence, the synthetic data generation processes and especially the stationary assumption limit interesting

patterns that can occur in real-world time series. Our working hypothesis is that in more realistic scenarios, one is likely to find time series with complex sources of non-stationary variations.

In a related empirical study, Mozetič et al. (2018) compare estimation procedures on several large time-ordered Twitter data sets. They find no significant difference between the best cross-validation and out-of-sample evaluation procedures. However, they do find that standard cross-validation (`CV`) is significantly worse than the blocked cross-validation (`CV-Bl`), and should not be used to evaluate classifiers in time-ordered data scenarios.

In this context, the work in this chapter provides an extensive comparative study using a wide set of methods for evaluating the performance of univariate time series forecasting models. These include several variants of both cross-validation and out-of-sample approaches. The analysis is carried out using a real-world scenario as well as a synthetic case study used in the works described previously (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018).

3.3 Materials and Methods

In this section, we present the materials and methods used in this work. We start by defining the prediction task, in which forecasting the next value of the time series is framed as an auto-regressive problem (2.2.3). Second, we describe the data sets of time series used. These include both synthetic and real-world time series, where half of the latter are non-stationary. We then formalise the methodology employed for applying and evaluating each performance estimation under comparison. Finally, we describe the experimental design.

3.3.1 Predictive Task Definition

As we defined in Chapter 2, a time series represents a temporal sequence of values $Y = \{y_1, y_2, \ldots, y_n\}$, where y_i is the value of Y at time i and n is the length of Y. We remark that we use the term time series assuming that Y is a numeric variable, i.e., $y_i \in \mathbb{R}, \forall\ y_i \in Y$. Time series forecasting denotes the task of predicting the next value of the time series, y_{n+1}, given the previous observations of Y. We focus on an auto-regressive modelling approach, predicting

future values of time series using its past p lags. We describe this process in Section 2.2.3.

3.3.2 Time Series Data

Two different case studies are used to analyse the performance estimation methods: a scenario comprised of real-world time series, and a synthetic setting used in prior work (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018) for addressing the issue of performance estimation for time series forecasting tasks.

Real-World Time Series

The real-world case study includes 62 time series from different domains of application. They have different granularity, size, as well as unknown dynamics. According to a wavelet spectrum test (Nason, 2013), half of the time series are stationary, while the remaining ones are not stationary. A comprehensive description of these time series can be found in Appendix A (Table A.1).

Synthetic Time Series

We use three synthetic use cases defined in previous work by Bergmeir et al. (2014, 2018). The data generating processes are all stationary and are designed as follows:

S1: A stable auto-regressive process with lag 3, i.e., the next value of the time series is dependent on the past 3 observations – c.f. Figure 3.1 for a sample graph.

S2: An invertible moving average process with lag 1 – c.f. Figure 3.2 for a sample graph.

S3: A seasonal auto-regressive process with lag 12 and seasonal lag 1 – c.f. Figure 3.3 for a sample graph.

For the first two cases, S1 and S2, real-valued roots of the characteristic polynomial are sampled from the uniform distribution $[-r; -1.1] \cup [1.1, r]$, where r is set to 5 (Bergmeir and Benítez, 2012). Afterwards, the roots are used to estimate the models and create the time series. The data is then processed by

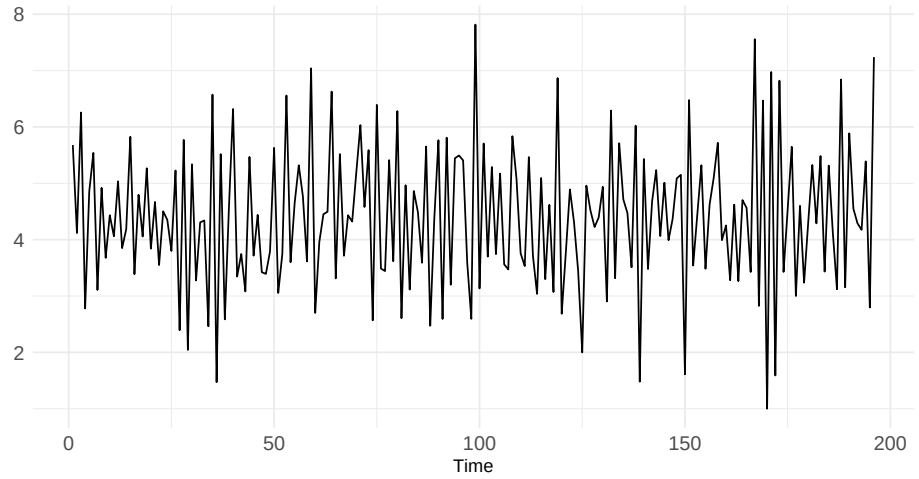


Figure 3.1: Sample graph of the S1 synthetic case.

making the values all positive. This is accomplished by subtracting the minimum value and adding 1. The third case S3 is created by fitting a seasonal autoregressive model to a time series of monthly total accidental deaths in the USA (Brockwell and Davis, 2013). For a complete description of the data generating process, we refer to the work by Bergmeir and Benítez (2012); Bergmeir et al. (2018). Similarly to Bergmeir et al., for each use case, we performed 1000 simulations of each time series. In each repetition, a time series with 200 values was generated.

3.3.3 Performance Estimation Methodology

Performance estimation addresses the issue of estimating the predictive performance of predictive models. Frequently, the objective behind these tasks is to compare different solutions for solving a predictive task. This includes selecting among different learning algorithms and hyper-parameter tuning for a particular one.

Training a learning model and evaluating its predictive ability on the same data has been shown to produce biased results due to overfitting (Arlot et al., 2010). Since then, several methods for performance estimation have been proposed in the literature, which use new data to estimate the performance of models. Usually, new data is simulated by splitting the available data. Part of

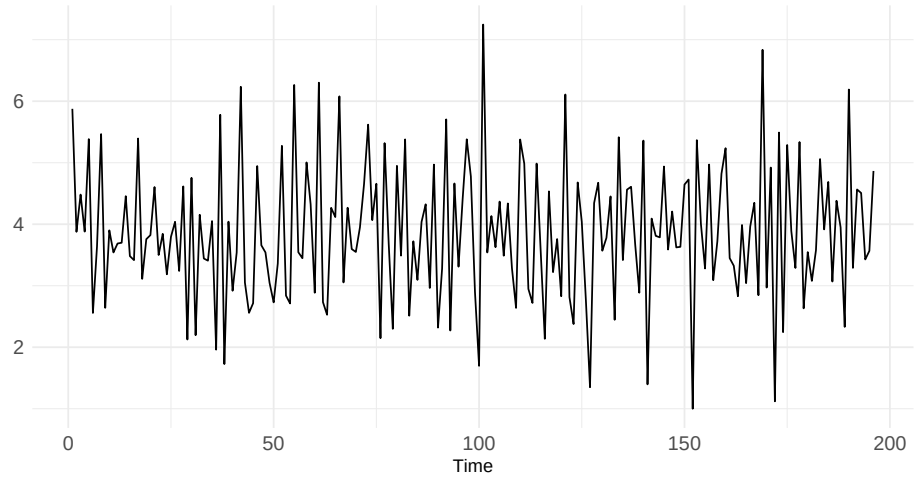


Figure 3.2: Sample graph of the S2 synthetic case.

the data is used for training the learning algorithm, and the remaining data is used to test and estimate the performance of the model.

For many predictive tasks, the most widely used of these methods is K-fold cross-validation (Stone, 1974), which we denote as `CV` and describe in Section 2.3. The main advantages of this method are its universal splitting criteria and efficient use of all the data. However, `CV` is based on the assumption that observations in the underlying data are independent. When this assumption

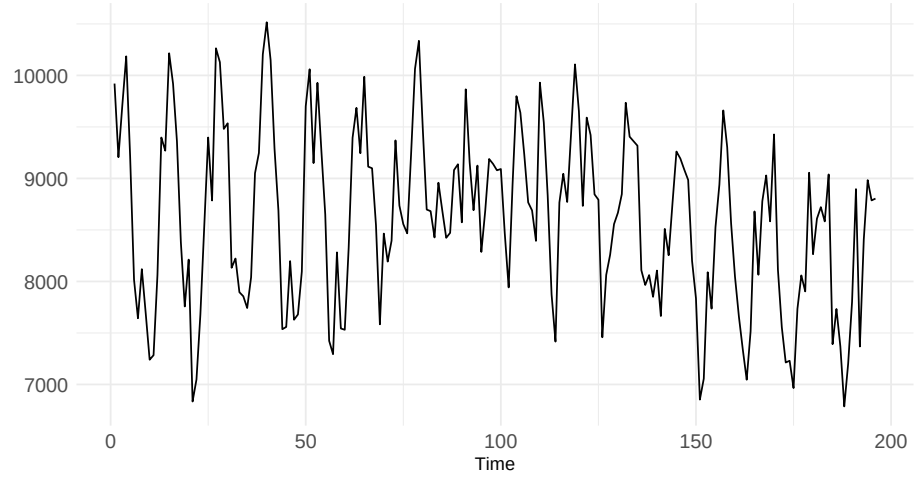


Figure 3.3: Sample graph of the S3 synthetic case.

is violated, for example, in time series data, theoretical problems arise that prevent the use of this method in such scenarios. As we described in Section 2.3, several methods have been developed that cope with this issue, from OOS approaches (Tashman, 2000) to variants of `CV`, e.g., blocked cross-validation (Snijders, 1988).

Our goal in this chapter is to compare a wide set of estimation procedures, and test their suitability for different types of time series forecasting tasks. In order to emulate a realistic scenario, we split each time series data into two parts. The first part is used to estimate the loss that a given learning model will incur on unseen future observations. This part is further split into training and test sets as described before. The second part is used to compute the true loss that the model incurred. This strategy allows the computation of unbiased estimates of error since a model is always tested on unseen observations.

The workflow described above is summarised in Figure 3.4. A time series Y

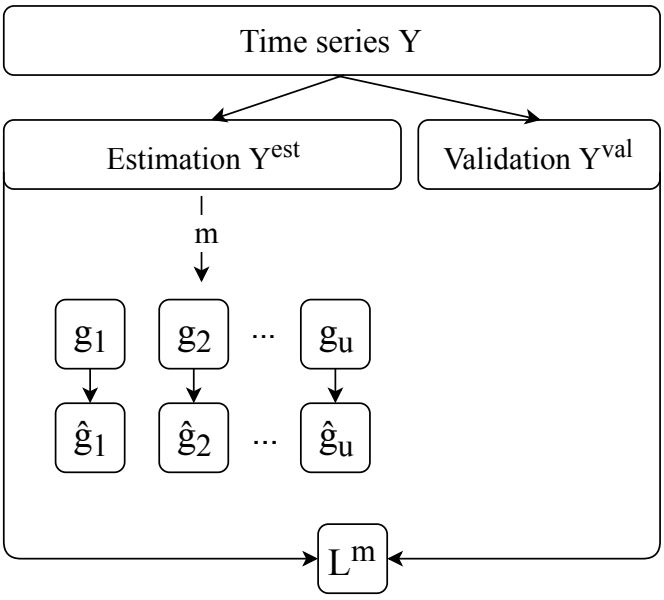


Figure 3.4: Experimental comparison procedure: A time series is split into an estimation set Y^{est} and a subsequent validation set Y^{val}. The first is used to estimate the error $\hat{g}$ that the model m will incur on unseen data, using u different estimation methods. The second is used to compute the actual error L^m incurred by m. The objective is to approximate L^m by $\hat{g}$ as well as possible.

is split into an estimation set Y^{est} and a subsequent validation set Y^{val}. First, Y^{est} is used to compute $\hat{g}$, the estimate of the loss that a predictive model m will incur on future new observations. This is accomplished by further splitting Y^{est} into training and test sets according to the respective estimation procedure g_i, $i \in \{1, \ldots, u\}$. Accordingly, the forecasting model m is built on the training set and $\hat{g}_i$ is computed on the test set.

Second, in order to evaluate the estimates $\hat{g}_i$ produced by the methods g_i, $i \in \{1, \ldots, u\}$, the model m is re-trained using the complete set Y^{est} and tested on the validation set Y^{val}. Effectively, we obtain L^m, the ground truth loss that m incurs on new data. Essentially, the goal of an estimation method g_i is to approximate L^m by $\hat{g}_i$ as well as possible. In Section 3.3.4, we describe how to quantify this approximation.

3.3.4 Experimental Design

The experimental design was devised to address the following research question: How do the predictive performance estimates of CVAL methods compare to the estimates of OOS approaches for time series forecasting tasks?

Existing empirical evidence suggests that CVAL methods provide more accurate estimations than traditionally used OOS approaches in stationary time series forecasting (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018) (see Section 3.2). However, many real-world time series comprise complex structures. These include cues from the future that may not have been revealed in the past. Effectively, our working hypothesis is that preserving the temporal order of observations when estimating the predictive ability of models is an important component.

Trend, Auto-Regressive Order, and Estimation Set Size

We applied a KPSS statistical test (Kwiatkowski et al., 1992) to account for trend in the data. Time series that are not trend-stationary according to this test are differenced until the test is passed. This approach is commonly used for trend inclusion in forecasting models, for example, ARIMA. Specifically, we follow the procedure adopted by the automatic forecasting model `auto.arima` from the forecast R package (Hyndman et al., 2014). The number of differences

applied to each time series is described in the last column of Table A.1.

We estimate the optimal embedding dimension (p) using the method of False Nearest Neighbours (Kennel et al., 1992). This method analyses the behaviour of the nearest neighbours as we increase p (c.f. Section 2.2.3). We set the tolerance of false nearest neighbours to 1%. The embedding dimension estimated for each series is shown in Table A.1. Regarding the synthetic case study, we fixed the embedding dimension to 5. The reason for this setup is to try to follow the experimental setup by Bergmeir et al. (2018).

The estimation set (Y^{est}) in each time series is the first 70% observations of the time series – see Figure 3.4. The validation period is comprised of the subsequent 30% observations (Y^{val}).

Estimation Methods

In the experiments, we apply a total of 11 performance estimation methods, which are divided into CVAL variants and OOS approaches. The cross-validation methods are the following:

`CV` Standard, randomized K-fold cross-validation;

`CV-Bl` Blocked K-fold cross-validation;

`CV-Mod` Modified K-fold cross-validation;

`CV-hvBl` hv-Blocked K-fold cross-validation;

Conversely, the out-of-sample approaches are the following:

`Holdout` A simple OOS approach–the first 70% of Y^E is used for training and the subsequent 30% is used for testing;

`Rep-Holdout` OOS tested in *nreps* testing periods with a Monte Carlo simulation using 70% of the total observations n of the time series in each test. For each period, a random point is picked from the time series. The previous window comprising 60% of n is used for training, and the following window of 10% of n is used for testing;

`Preq-Bls` Prequential evaluation in blocks in a growing fashion;

`Preq-Sld-Bls` Prequential evaluation in blocks in a sliding fashion–the oldest block of data is discarded after each iteration;

`Preq-Bls-Gap` Prequential evaluation in blocks in a growing fashion with a gap block–this is similar to the method above, but comprises a block separating the training and testing blocks in order to increase the independence between the two parts of the data;

`Preq-Grow` **and** `Preq-Slide` As baselines, we also include the exhaustive prequential methods in which an observation is first used to test the predictive model and then to train it. We use both a growing/landmark window (`Preq-Grow`) and a sliding window (`Preq-Slide`).

We refer to Section 2.3 in the background chapter of this thesis for a complete description of these methods. The number of folds K or repetitions $nreps$ in these methods is set to 10, which is a commonly used setting in the literature. The number of observations removed in `CV-Mod` and `CV-hvBl` (c.f. Section 2.3) is the embedding dimension p of each time series.

Evaluation Metrics

Our goal is to study which estimation method provides a $\hat{g}$ that best approximates L^m. Let $\hat{g}_i^m$ denote the estimated loss by the learning model m using the estimation method g on the estimation set, and L^m denote the ground truth loss of learning model m on the test set. The objective is to analyse how well $\hat{g}_i^m$ approximates L^m. This is quantified by the absolute predictive accuracy error (APAE) metric and the predictive accuracy error (PAE) (Bergmeir et al., 2018):

$$\text{APAE} = |\hat{g}_i^m - L^m| \tag{3.1}$$

$$\text{PAE} = \hat{g}_i^m - L^m \tag{3.2}$$

The APAE metric evaluates the error size of a given estimation method. On the other hand, PAE measures the error bias, i.e., whether a given estimation method is under-estimating or over-estimating the true error.

Another question regarding evaluation is how a given learning model is evaluated regarding its forecasting accuracy, that is, how each $\hat{g}_i^m$ or L^m is quantified. In this work, we evaluate models according to RMSE. This metric is traditionally used for measuring the differences between the estimated values and actual values.

Learning Algorithm

The results shown in this work are obtained using a rule-based regression system Cubist (Kuhn et al., 2014), a variant of the *model tree* proposed by Quinlan (1993). This method presented the best forecasting results among several other predictive models in a study that will be presented in the next chapter. Notwithstanding, other learning algorithms were tested, namely the lasso (Tibshirani, 1996; Friedman et al., 2010) and a random forest (Breiman, 2001; Wright, 2015). The conclusions drawn using these algorithms are similar to the ones reported in the next sections.

3.4 Empirical Experiments

3.4.1 Research Questions

The experiments presented in this section are designed to answer the following research questions:

RQ1.1: How do OOS methods compare with CVAL methods for estimating the predictive performance of forecasting models in synthetic stationary time series?

RQ1.2: Similarly to **RQ1.1**, how do the results change when using real-world time series?

RQ1.3: Focusing on the real-world time series, what is the impact of stationarity in the relative performance estimation ability of each method?

RQ1.4: What are the most important time series characteristics when choosing the most appropriate performance estimation method?

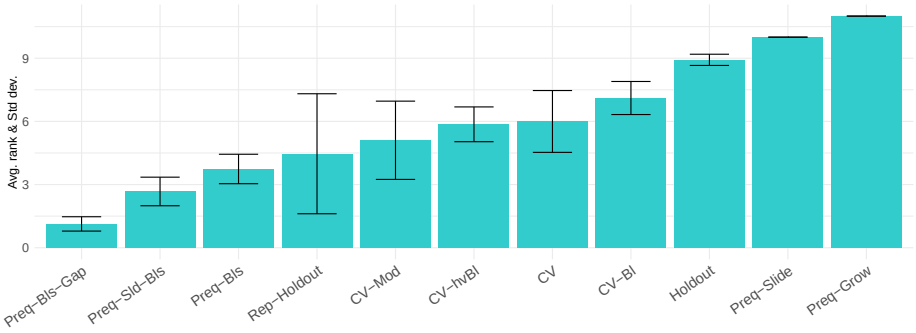


Figure 3.5: Average rank and respective standard deviation of each estimation methods in case study S1

3.4.2 Results with Synthetic Case Studies

In this section, we address the question **RQ1.1**. We start by analysing the average rank, and respective standard deviation, of each estimation method and for each synthetic scenario (S1, S2, and S3), according to the metric APAE. For example, a rank of 1 in a given simulation (c.f. Section 3.3.2) means that the respective method was the best estimator in that repetition. These analyses are reported in Figures 3.5–3.7. This initial experiment is devised to reproduce the results by Bergmeir et al. (2018). Later, we will analyse how these results compare when using real-world time series.

The results shown by the average ranks corroborate those presented by Bergmeir et al. (2018). That is, cross-validation approaches generally perform better (i.e., show a lower average rank) relative to the simple out-of-sample procedure `Holdout`. This can be concluded from all three scenarios: S1, S2, and S3.

Focusing on scenario S1, the estimation method with the best average rank is `Preq-Bls-Gap`, followed by the other two prequential variants (`Preq-Sld-Bls`, and `Preq-Bls`). Although the `Holdout` procedure is a relatively poor estimator, the repeated holdout in multiple testing periods (`Rep-Holdout`) shows a better average rank than the cross-validation procedures (though with a large standard deviation). Among cross-validation procedures, `CV-Mod` presents the best average rank.

Scenario S2 shows a seemingly different story relative to S1. In this problem, the prequential variants present the worst average rank, while the cross-validation procedures show the best estimation ability. Among all, `CV-hvBl` shows the best average rank. Moreover, `Rep-Holdout` presents again a large standard deviation in rank, relative to the remaining estimation methods.

Regarding the scenario S3, the outcome is less clear than the previous two scenarios. The methods show a closer average rank among them, with large standard deviations.

In summary, this first experiment corroborates the experiment carried out by Bergmeir et al. (2018). Notwithstanding, other methods that the authors did not test show an interesting estimation ability in these particular scenarios, namely the prequential variants.

The synthetic scenarios comprise time series that are stationary. However, real-world time series often comprise complex dynamics that break stationarity. When choosing a performance estimation method, one should take this issue into consideration. To account for time series stationarity, in the next section, we analyse the estimation methods using real-world time series. We will also control for time series stationarity to study its impact on the results.

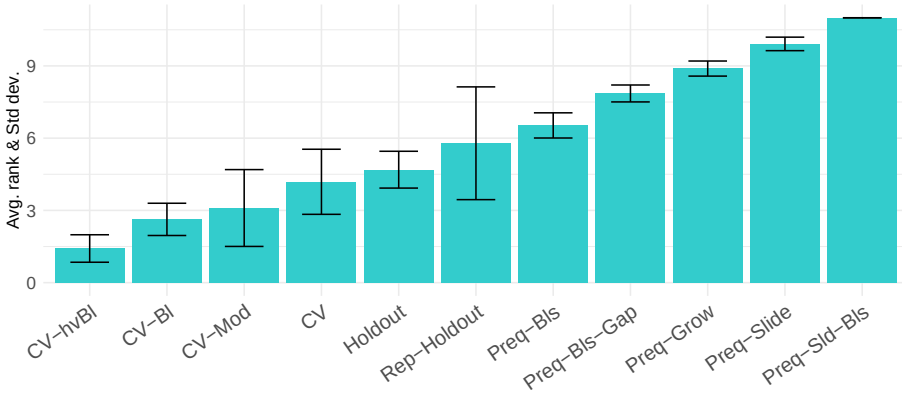


Figure 3.6: Average rank and respective standard deviation of each estimation methods in case study S2

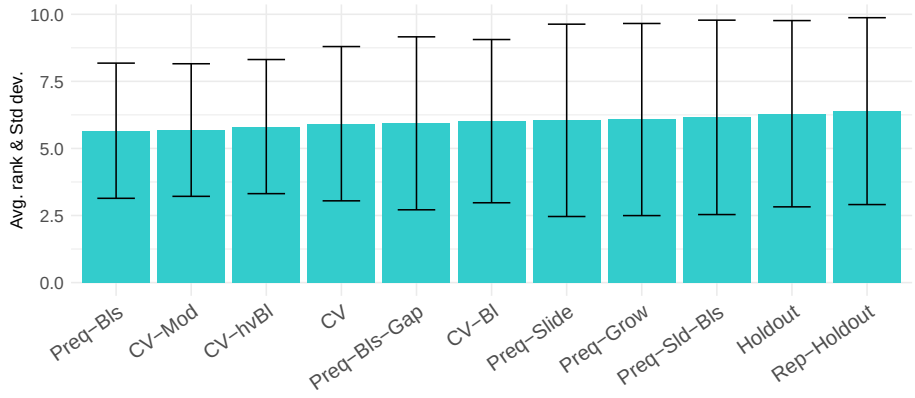


Figure 3.7: Average rank and respective standard deviation of each estimation methods in case study S3

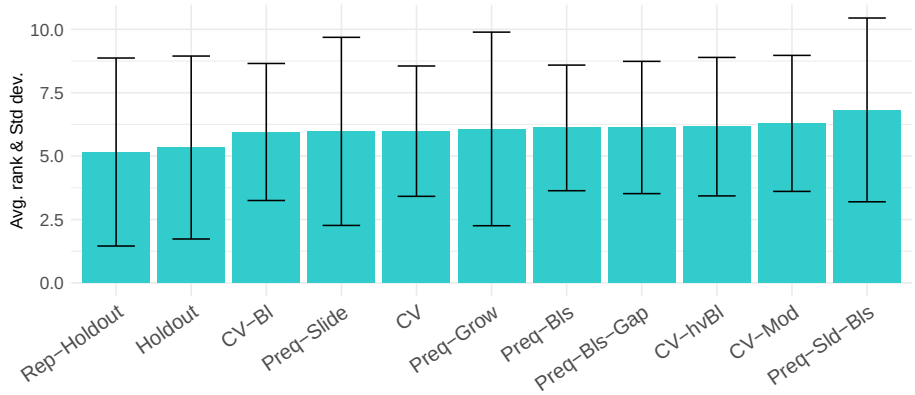


Figure 3.8: Average rank and respective standard deviation of each estimation methods in case study RWTS

3.4.3 Results with Real-world Case Studies

Results using All Real-world Time Series

The research question **RQ1.2** is addressed in this section. We analyse the performance estimation ability of each method using a case study comprised of real-world time series from different domains.

To accomplish this, in Figure 3.8, we start by analysing the average rank,

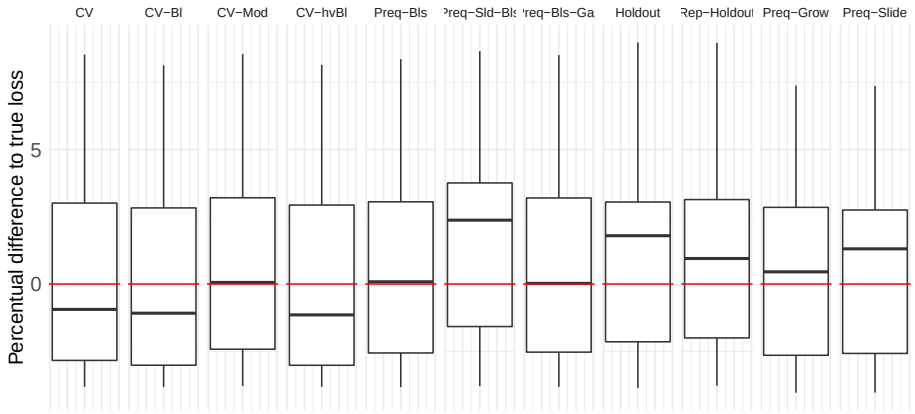


Figure 3.9: Percentual difference of the estimated loss relative to the true loss for each estimation method in the RWTS case study. Values below the zero line represent under-estimations of error. Conversely, values above the zero line represent over-estimations of error.

and respective standard deviation, of each estimation method using the APAE metric. This graphic tells a different story relative to the synthetic case study. Particularly, the `Rep-Holdout` and `Holdout` show the best estimation ability in terms of the average rank. The method `CV-Bl` is the best estimator among the cross-validation procedures.

In order to study the direction of the estimation error, in Figure 3.9 we present for each method the percentual difference between the estimation error and the true error according to the PAE metric. In this graphic, values below the zero line denote under-estimations of error, while values above the zero line represent over-estimations. In general, cross-validation procedures tend to under-estimate the error (i.e. are optimistic estimators), while the prequential and out-of-sample variants tend to over-estimate the error (i.e. are pessimistic estimators).

This result corroborates the results on Twitter time-ordered data (Mozetič et al., 2018). The authors found that all variants of cross-validation procedures tend to under-estimate the errors, while the out-of-sample procedures tend to over-estimate them.

We also study the statistical significance of the obtained results in terms of

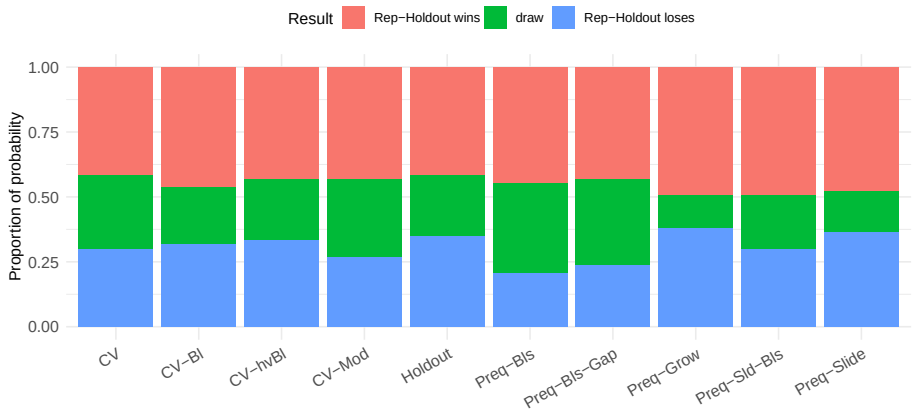


Figure 3.10: Proportion of probability of the outcome when comparing the performance estimation ability of the respective estimation method with the `Rep-Holdout` method. The probabilities are computed using the Bayes sign test.

error size (APAE) according to a Bayesian analysis (Benavoli et al., 2017). Particularly, we employed the Bayes sign test to compare pairs of methods across multiple problems. We define the *region of practical equivalence* (Benavoli et al., 2017) (ROPE) to be the interval [-2.5%, 2.5%] in terms of percentual difference in APAE to the `Rep-Holdout` method. We used the percentual difference value to cope with the different scales of time series. Essentially, this means that two methods show indistinguishable performance if the difference in performance between them falls within this interval. For a thorough description of the Bayesian analysis for comparing predictive models, we refer to the work by Benavoli et al. (2017).

In this experiment, we fix the method `Rep-Holdout` as the baseline since it is the one showing the best average rank (Figure 3.8). According to the illustration in Figure 3.10, the probability of `Rep-Holdout` winning (i.e., showing a significantly better estimation ability) is generally larger than the opposite.

Controlling for Stationarity

After analysing the synthetic case study, we hypothesised that the results were biased due to the stationarity assumption. In this section, we repeat the av-

erage rank experiment in the real-world case study controlling for stationarity (**RQ1.3**). We consider a time series stationary according to a wavelet spectrum test (Nason, 2013). We described this method in Section 2.1.3.

In Figure 3.11, we present the results considering only the real-world time series that are stationary. According to the average rank, `Rep-Holdout` presents the best estimation ability, followed by the typical cross-validation approach `CV`.

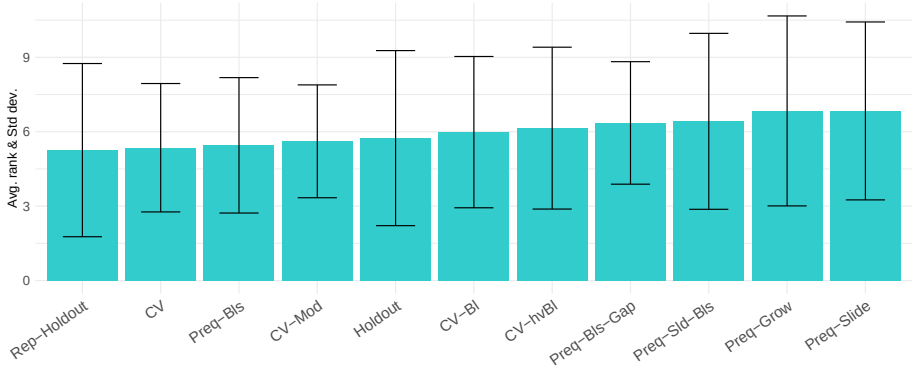


Figure 3.11: Average rank and respective standard deviation of each estimation methods in case study RWTS for stationary time series (31 time series).

In Figure 3.12, we present a similar analysis for the non-stationary time series, whose results are considerably different relative to stationary time series. In this scenario, `CV` is one of the worst estimators according to average rank. The out-of-sample approaches `Holdout` and `Rep-Holdout` present the best estimation ability.

Descriptive model

What makes an estimation method appropriate for a given time series is related to the characteristics of the data. For example, we analysed the impact that stationarity has in terms of what is the best estimation method in the previous section.

The real-world time series case study comprises a set of time series from different domains. In this section, we present, as descriptive analysis, a tree-based model that relates some characteristics of time series according to the most appropriate estimation method for that time series (**RQ1.4**). We create

a predictive task in which the attributes are some characteristics of a time series, and the categorical target variable is the estimation method that best approximates the true loss in that time series. We use CART (Breiman, 2017) (classification and regression tree) algorithm for obtaining the model for this task. The characteristics used as predictor variables are the following summary statistics:

- **Skewness**, for measuring the symmetry of the distribution of the time series;
- 5-th and 95-th Percentiles (**Perc05**, **Perc95**) of the standardized time series;
- Acceleration (**Accel.**), as the average ratio between a simple moving average and the exponential moving average;
- Inter-quartile range (**IQR**), as a measure of the spread of the standardised time series;
- Serial correlation, estimated using a Box-Pierce test statistic;
- Long-range dependence, using a Hurst exponent estimation with wavelet transform;

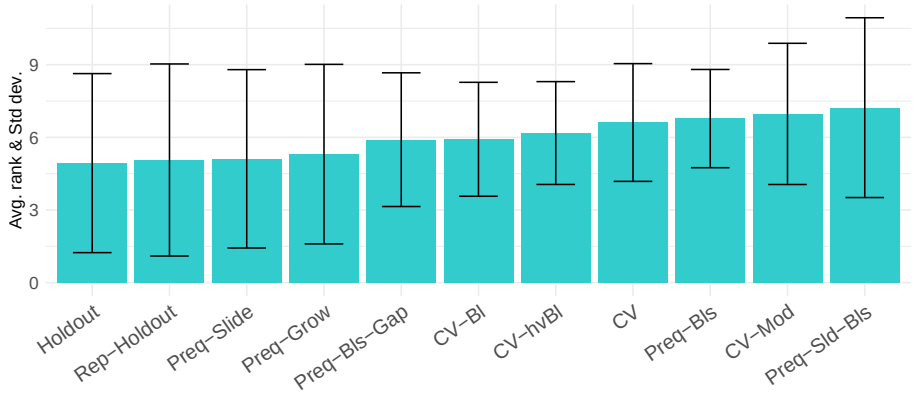


Figure 3.12: Average rank and respective standard deviation of each estimation methods in case study RWTS for non-stationary time series (31 time series).

- Maximum Lyapunov Exponent, as a measure of the level of chaos in the time series;

- a boolean variable, indicating whether or not the respective time series is stationary according to the wavelet spectrum test (Nason, 2013).

The characteristics used in the obtained decision tree are written in boldface. The decision tree is shown in Figure 3.13. The numbers below the name of the method in each node denote the number of times the respective method is best over the number of time series covered in that node.

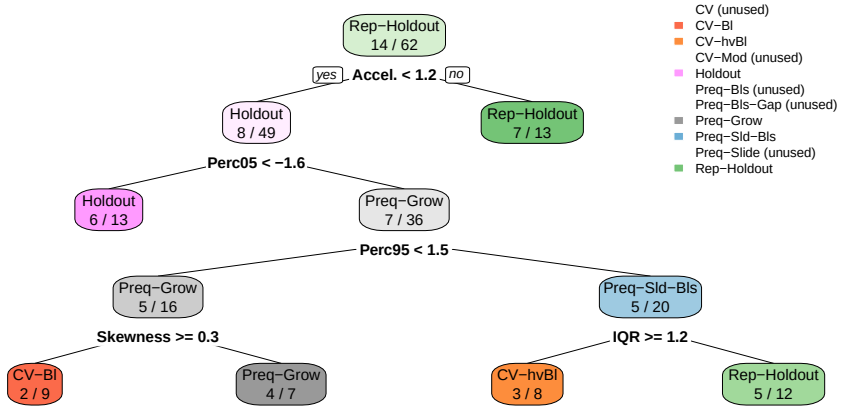


Figure 3.13: Decision tree that maps the characteristics of time series to the most appropriate estimation method. This graphic was created using the *rpart.plot* R package (Milborrow, 2018).

Some of the estimation methods do not appear in the tree model. The tree leaves, which represent a decision, are dominated by the `Rep-Holdout` and `Holdout` estimation methods. The estimation methods `CV-Bl`, `Preq-Grow`, and `CV-hvBl` also appear in other leaves.

The estimation method in the root node is `Rep-Holdout`, which is the best method most of the times across the 62 time series. The first split is performed according to the acceleration characteristic of time series. If acceleration is not below 1.2, the tree leads to a leaf node with `Rep-Holdout` as the most appropriate estimation method. Otherwise, the tree continues with more tests in order to find the most suitable estimation method for each particular scenario.

3.5 Discussion

3.5.1 Impact of the Results

In the experimental evaluation we compare several performance estimation methods in two distinct scenarios: (1) a synthetic case study in which artificial data generating processes are used to create stationary time series; and (2) a real-world case study comprising 62 time series from different domains. The synthetic case study is based on the experimental setup used in previous studies by Bergmeir et al. for the same purpose of evaluating performance estimation methods for time series forecasting tasks (Bergmeir and Benítez, 2012; Bergmeir et al., 2014, 2018).

Bergmeir et al. show in previous studies (Bergmeir and Benitez, 2011; Bergmeir and Benítez, 2012) that the blocked form of cross-validation, denoted here as `CV-Bl`, yields more accurate estimates than a simple out-of-sample evaluation (`Holdout`) for stationary time series forecasting tasks. The method `CV` is also suggested to be "a better choice than OOS (`Holdout`) evaluation" as long as the data are well fitted by the model (Bergmeir et al., 2018). To some extent, part of the results from our experiments corroborate these conclusions. Specifically, this is verified by the APAE incurred by the estimation procedures in the synthetic case studies.

However, according to our experiments, the results from the synthetic stationary case studies do not reflect those obtained using real-world time series. In general, holdout applied with multiple randomised testing periods (`Rep-Holdout`) provides the most accurate performance estimates. Notwithstanding, for stationary time series, `CV` also shows a competitive performance estimation ability.

In a real-world environment, we are prone to deal with time series with complex structures and different sources of non-stationary variations. These comprise nuances of the future that may not have revealed themselves in the past (Tashman, 2000). Consequently, we believe that in these scenarios, `Rep-Holdout` is a better option as a performance estimation method relative to cross-validation approaches.

3.5.2 On the Importance of Data Size

The temporal order preservation by OOS approaches, albeit more realistic, comes at a cost since less data is available for estimating predictive performance. As Bergmeir et al. (2018) argue, this may be important for small data sets, where a more efficient use of the data (e.g. `CV`) may be beneficial. However, during our experimental evaluation, we did not found compelling evidence to back this claim. In the reported experiments, we fixed the data size to 200 observations, as Bergmeir et al. (2018) did. In order to control for data size, we varied this parameter from a size of 100 to a size of 3000, by intervals of 100 (100, 200, ..., 3000). The experiments did not provide any evidence that the size of the synthetic time series had a noticeable effect on the error of estimation methods.

In our experiments, the size of the time series in the real-world case study is in the order of a few thousands. For large scale data sets the recommendation by Dietterich (1998), and usually adopted in practice, is to apply a simple out-of-sample estimation procedure (`Holdout`).

3.5.3 Scope of the Real-World Case Studies

In this work, we centre our study on univariate numeric time series. Nevertheless, we believe that the conclusions of our study are independent of this assumption and should extend for other types of time series. The objective is to predict the next value of the time series, assuming immediate feedback from the environment. Moreover, we focus on time series with a high sampling frequency, specifically, half-hourly, hourly, and daily data. The main reason for this is because high sampling frequency is typically associated with more data, which is important for fitting the predictive models from a machine learning point of view. Standard forecasting benchmark data are typically more centred around low sampling frequency time series; for example, the M competition data (Makridakis et al., 1982). Following our work, a possibly interesting research direction is to study the evaluation of predictive models in other types of dependent data. For example, after the publication of our work (Cerqueira et al., 2017a), Oliveira et al. (2018) presented an extensive study on evaluation

methods for spatio-temporal forecasting problems.

3.6 Conclusions

Choosing the most appropriate model to solve a given predictive task strongly depends on the expectation of how that model will perform on new observations not used during training. Therefore, it is important to obtain reliable estimates of the performance of predictive models. In this chapter, we analyse the ability of different approaches to approximate the loss that a given predictive model will incur on unseen data. We focus on performance estimation for time series forecasting tasks. Since there is currently no settled approach for performance estimation in these settings, our objective is to compare different available methods and test their suitability.

We analyse several methods that can be generally split into OOS approaches and CVAL methods. These were applied to two case studies: a synthetic environment with stationary time series and a real-world scenario with potential non-stationarities. In a stationary setting, the CVAL variants are shown to have a competitive estimation ability. However, when non-stationarities are present, they systematically provide worse estimations than the OOS approaches.

In summary, according to the results of the empirical experiments we recommend the following approach when choosing an estimation method:

- If the data is stationary, we confirm the results of Bergmeir et al. (2018) that standard cross-validation can be applied;
- However, for real-world time series with potential non-stationarities, we conclude that approaches that maintain the temporal order of data systematically provide better error estimations. In particular, we recommend the adoption of `Rep-Holdout`, the holdout approach applied in multiple testing periods.

In the interest of reproducibility, the methods and data sets are publicly available at `https://github.com/vcerqueira/performance_estimation`.

Chapter 4

Arbitrage of Forecasting Models

4.1 Introduction

A considerable number of forecasting methods has been proposed in the literature. Notwithstanding, it is widely accepted that none is the most appropriate for all problems (Chatfield, 2000). Even within a single time series, there is evidence that different hypotheses are better at different times. This is due to concept drift and non-stationary sources of variation, which change the underlying process causing the data (Gama et al., 2014). Accordingly, all predictive models have strengths and limitations that need to be managed. Selecting which model (or models) represent the strongest hypothesis (or hypotheses) at a particular point in time is a difficult task. The work in this chapter is based on this idea. We use a set of different forecasting models and dynamically combine them to predict the future values of time series.

4.1.1 Dynamic Combination of Forecasting Models

The machine learning research field that is devoted to studying "the ability to automatically exploit the strengths and limitations of different learning systems" (Brown, 2010b) is ensemble learning (Kuncheva, 2004b; Brown et al., 2005a; Mendes-Moreira et al., 2012; Krawczyk et al., 2017). The basic idea behind

ensemble methods is the combination of the output of a number of predictive models. Ensemble methods have been shown to provide a superior predictive performance relative to single learning algorithms (Ueda and Nakano, 1996; Breiman, 1996).

As we reported in Section 2.2.5, ensemble methods for time series forecasting are typically dynamic. This means that the weights of each model comprising the ensemble change over time in response to concept drift. The ensemble learning approach to concept drift adaptation falls within the *model management* category, according to the seminal work by Gama et al. (2014). Within this category, in this work, we focus on dynamic combiners (Kuncheva, 2004b). That is, several time series forecasting models are created in advance, and then dynamically combined to adapt to changes in the environment.

The state of the art approaches for dynamically combining forecasting models are mostly based on estimates of predictive performance. The loss of each model is tracked over time and used to combine them adaptively. Some of these approaches have interesting theoretical loss upper bounds based on regret minimisation (Cesa-Bianchi and Lugosi, 2006). Meta-learning approaches are also commonly used (Brazdil et al., 2008). For example, stacking (Wolpert, 1992), which directly models inter-dependencies between experts. This characteristic may be important to take into account the diversity among experts, which is a key component in ensemble learning (Brown et al., 2005a).

4.1.2 Our Approach: Arbitrated Dynamic Ensemble

In this chapter, we present a meta-learning strategy to combine the available forecasting models in a dynamic way. However, contrary to stacking, we separately model the individual expertise of each forecasting model, assuming these to be specialists in different parts of the time series. Consequently, the forecasting models are combined in such a way that they are only selected for predicting examples that they are expected to be good at. Moreover, as opposed to tracking the error on past instances, our combination approach is more proactive as it is based on predictions of future loss of models. This can result in a faster adaptation to changes in the environment.

The motivation for our approach is that different learning models have dif-

ferent areas of expertise across the input space. Moreover, it is common for the underlying process generating the time series to have recurrent structures due to factors such as seasonality (Gama and Kosina, 2009, 2014). In this context, we hypothesise that a meta-learning strategy enables the ensemble to better detect changes in the relative performance of models, or changes between different regimes governing a time series, and quickly adapt itself to the environment.

The proposed meta-learning strategy, hereby denoted as Arbitrated Dynamic Ensemble (`ADE`), is based on arbitrating (Koppel and Engelson, 1996; Ortega et al., 2001a), a method from the family of mixture of experts (Jacobs et al., 1991; Masoudnia and Ebrahimpour, 2014). A meta-learner is created for each forecasting model that is part of the ensemble. Each meta-learner is specifically designed to model how apt its base counterpart is to make an accurate prediction for a given test example. This is accomplished by analysing how the error incurred by a given learning model relates to the characteristics of the data. At test time, the base-learners are weighted according to their expected degree of competence in the input observation, estimated by the predictions of the meta-learners. This is illustrated in Figure 4.1.

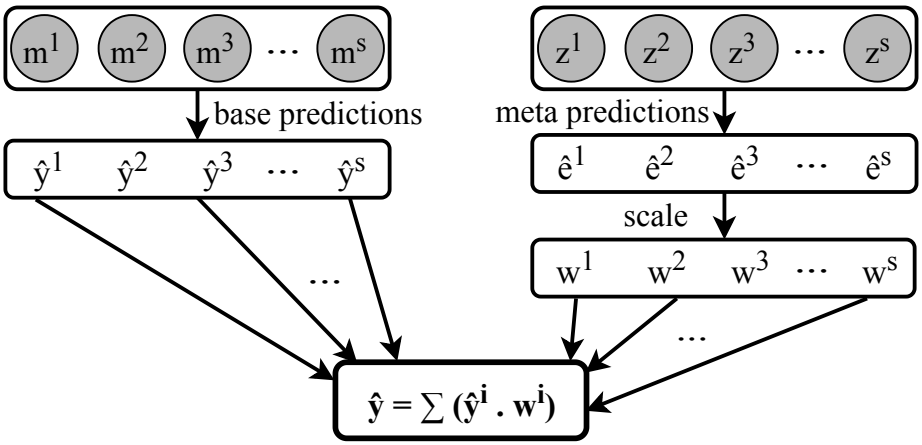


Figure 4.1: Workflow of `ADE` for a new prediction. The base-learners M produce the predictions $\hat{y}^i$, $i \in \{1, \ldots, s\}$ for the next value of the time series. In parallel, the meta-learners Z produce the weights w^i of each base-learner according to the predictions of their error ($\hat{e}^i$). The final prediction $\hat{y}$ is computed using a weighted average of the predictions relative to the weights.

Let M and Z denote a set of s base models and a set of s meta models, respectively. While a given base-learner m^i is trained to model the future values of the time series, its meta-learning associate z^i is trained to model the error of m^i. The model z^i is an arbiter that can make predictions regarding the error that m^i will incur when predicting the future values of the time series. The larger the estimates produced by z^i (relative to the other models in the ensemble), the lower the weight of m^i will be in the combination rule.

Diversity among the experts is a fundamental component in building ensemble methods (Brown et al., 2005b). We start by addressing this issue implicitly, by using experts with different learning strategies, i.e. heterogeneous ensembles. We assume that the ensemble heterogeneity is useful to cope with the different dynamic regimes of time series. Besides heterogeneity, we encourage diversity explicitly during the aggregation of the output of experts. This is achieved by taking into account not only predictions of performance produced by the arbiters but also the correlation among experts in a recent window of observations.

We validate the proposed method in 62 real-world time series. Empirical experiments suggest that our method is competitive with different adaptive methods for combining experts and other meta-learning approaches such as stacking (Wolpert, 1992). In the interest of reproducible research, `ADE` is publicly available as an R software package[1]. Moreover, all experiments reported in the chapter are also reproducible[2].

In summary, the contributions presented in this chapter are the following:

- `ADE`, a novel method based on meta-learning for dynamically combining a portfolio of forecasting models;

- The introduction of a blocked prequential procedure in the arbitrage approach to obtain out-of-bag predictions in the training set in order to increase the data used to train the meta-learning models;

- A sequential re-weighting strategy for controlling the redundancy among the output of the experts using their correlation in a recent window of observations;

[1] **tsensembler**: on CRAN or at `https://github.com/vcerqueira/tsensembler`

[2] Instructions at: `https://github.com/vcerqueira/forecasting_experiments`

- An extensive empirical study encompassing: statistical comparisons with state of the art approaches; analysis on the different deployment strategies of the proposed method; sensitivity analysis on the main parameters of the proposed method; relative scalability analysis in terms of execution time; and a study on the value of increasing the number of experts in the ensemble.

4.1.3 Related Work on Dynamic Ensembles and Arbitration

The models for the dynamic combination of forecasters outlined in Section 2.2.5 are related to our work in the sense that they employ adaptive heuristics to combine forecasting models throughout a time series. However, these heuristics are incremental or sliding summary statistics on relative **past performance**. Our intuition is that these approaches have a short memory and may fail to efficiently capture long-range relationships between changes in the underlying time series and the performance of the experts. Conversely, we explore differences among experts to specialise them across the data space based on a regression analysis. Moreover, we use a more proactive heuristic that is based on the prediction of relative **future performance** of individual forecasting models.

As mentioned before, our proposal follows a meta-learning strategy called arbitrating, which was introduced before for dynamic selection of classifiers by Ortega et al. (2001a). In the original arbitrating methodology, a prediction is made using a combination of different classifiers that are selected according to their expertise concerning the input data. The expertise of a model is learned using a meta-learner, one for each available base classifier, which models a measure of confidence of its respective base model. At run-time, the classifier with the highest predicted confidence is selected to make a prediction. In our work, the idea behind arbitration was reworked and applied to time series forecasting problems. Several of its drawbacks were addressed, such as the inefficient use of the available data, by using out-of-bag samples from the training set; a more robust combination rule by using a committee of recent well-performing models; and the general translation to the time series forecasting tasks, which is fundamentally different than classification tasks.

Arbitration is related to mixture of experts (Jacobs et al., 1991) (ME), in the sense that each expert is specialised in a certain region of the input space. The main difference to ME is the way the weights of the experts are computed. ME estimate the weights using a gating function. The gating function is typically a neural network with as many output units as experts and trained using Expectation-Maximisation (Chen et al., 1999). Our approach uses a set of arbiters that predict the loss of the experts. `ADE` also differs in the training procedure of the experts and how diversity is encouraged in the ensemble. ME are typically comprised by neural network experts built incrementally, and the gating function explicitly controls the patterns each neural network learns according to their relative performance. This results in relatively independent experts. Conversely, `ADE` works as a dynamic combiner approach (Kuncheva, 2004b). Diversity is introduced implicitly by employing a set of heterogeneous experts, which are trained with the whole set of available observations. During expert aggregation, diversity is also encouraged by considering the redundancy among the output of the experts.

The remainder of this chapter is structured as follows. We start by presenting the methodology in Section 4.2, where we formalise `ADE` and our contributions. The experiments and respective results are presented in Section 4.3, which includes the comparisons with the state of the art approaches for the dynamic combination of forecasting models. The results are discussed in Section 4.4, and Section 4.5 concludes the chapter. A brief note on terminology, the following pairs of expressions are used interchangeably throughout the chapter: *expert* and *base learner*, and *arbiter* and *meta-learner*.

4.2 Arbitrated Dynamic Ensemble

We formalise `ADE` in this section. We start by describing the predictive task and then explain the different steps of the methodology.

The predictive task is the same as in the last chapter, i.e., predicting the next value of a univariate time series Y. This process was formalised in Section 2.2.3. Essentially, the objective is to construct a model $f : \mathbb{X} \rightarrow \mathbb{Y}$, where f denotes the regression function. The input domain $\mathbb{X}$ represents the dynamics

of the time series in the past p lags, and the output domain $\mathbb{Y}$ represents the next value of the time series.

The proposed methodology in `ADE` for time series forecasting settles on the following three main steps:

- Training of the base-learners: the set of heterogeneous experts that are used to forecast future values of Y;
- Training the meta-learners: arbiters that model and predict the loss of the base experts;
- Predicting y_{n+1}: Combining the output of the experts according to the output of the arbiters and the correlation among the output of the experts to forecast the next value of the time series.

4.2.1 Training the Experts

The first step of `ADE` is to train s individual base models. Each $m^j \in M, \forall j \in \{1, \ldots, s\}$ is built using the available time series Y. The objective is to predict y_{n+1}, the next value of Y. This is accomplished by having experts approximate the unknown function according to $f : \mathbb{X} \rightarrow \mathbb{Y}$.

M is comprised of a set of s heterogeneous models, for example, decision trees (Quinlan, 1986) and artificial neural networks (Zhang et al., 1998). This type of heterogeneous ensembles is also known as *hybrid ensembles* in the literature (Brown et al., 2005b). The motivation behind combining different models is that these have distinct inductive biases, and thus, different assumptions regarding the process generating the data. Several works in the literature have demonstrated the usefulness of heterogeneous ensembles. For example, Wang et al. (2000) combine decision trees with artificial neural networks. They provide evidence that forming an ensemble with both types of learning algorithms is better than an ensemble composed of models from a single architecture (homogeneous ensemble). Other important works in the literature are those of Woods et al. (1997); Caruana et al. (2004); Langdon et al. (2002); van Rijn et al. (2018), to name a few. Effectively, by adopting a heterogeneous ensemble approach, we expect models to have different expertise across the time series. Later we will

present an approach complementary to ensemble heterogeneity that encourages diversity during the aggregation of the experts (Section 4.2.3).

4.2.2 Training the Arbiters

Meta-learning denotes the process of applying machine learning to model the learning process of learning algorithms (Brazdil et al., 2008). We apply a meta-learning methodology to our problem as follows. In the meta-learning step of `ADE`, the goal is to build models capable of modelling the expertise of each base-learner across the input space. Our assumption is that not all models will perform equally well at any given prediction point. This idea is in accordance with findings reported in prior work (Aiolfi and Timmermann, 2006). Systematic evidence was found that some models have varying relative performance over time and that other models are persistently good (or bad) throughout the time series. Furthermore, in many environments, the dynamic concepts have a recurring nature, due to, for example, seasonality. These findings can be regarded as instances of the *No Free Lunch* theorem presented by Wolpert (2002).

In effect, we use meta-learning to dynamically weigh base-learners and adapt the combined model to changes in the relative performance of the base models, as well as for the presence of different regimes in the time series.

Our meta-learning approach is based on an arbitrating architecture (Ortega et al., 2001a) and mixture of experts (Jacobs et al., 1991). Specifically, a meta-learner $z^j \in Z, \forall\ j \in \{1, \ldots, s\}$ is trained to build the following model:

$$e_i^j = f(x_i) \tag{4.1}$$

where e_i^j is the absolute error incurred by m^j in an observation (x_i, y_i). We formalise the meta-learning problem using the same feature set used by the experts to predict the future values of the time series.

We perform this regression analysis on a meta-level to understand how the error of a given model relates to the dynamics and the structure of the time series. Effectively, we can capitalise on this knowledge by dynamically combining base-learners according to the expectation of how they will perform.

Blocked Prequential for Out-of-Bag Predictions

Typical meta-learning approaches for dynamic model selection or combination, only start the meta-learning layer at run-time. This is the case of, for example, the original arbitrating formulation by Ortega et al. (2001a) or the work of Gama and Kosina (2014). This is motivated by the need for unbiased samples to build reliable meta-learners. However, this means that at the beginning, few observations are available to train the meta-learners, which might result in under-fitting.

`ADE` uses the training set to produce out-of-bag predictions which are then used to compute an unbiased estimate of the loss of each base-learner. By retrieving out-of-bag samples from the training set, we can significantly increase the amount of data available to the meta-learners. We hypothesise that this strategy improves the overall performance of the ensemble by improving the predictive performance of each meta-learner.

We produce out-of-bag samples by running a blocked prequential procedure (Dawid, 1984), a growing window approach. The available embedded time series used for training is split into b equally-sized and sequential blocks of contiguous observations. In the first iteration, the first block is used to train the base-learners M, and the second is used to test them. Then, the second block is merged with the first one for training M, and the third block is used for testing. This procedure continues until all blocks are tested (except the first one). This approach is identical to the `Preq-Bls` estimation method described in the previous chapter. In summary, using out-of-bag samples allows using the available data to train both the experts (as described above) and the arbiters. This results in a more efficient use of the available time series because it is used to fit both the experts and the arbiters. This data efficiency and the preservation of the temporal order of observations was the main motivation for using the blocked prequential with a growing window. The meta-learning phase is described in Algorithm 1.

Algorithm 1: Training the arbiters

input : Y – Time series observations

input : M – Set of base learners

input : b – Number of blocks

output : Set of arbiters Z

foreach m^j *in* M **do**

  $\hat{y}^j \leftarrow$ BlockedPrequential(m^j, Y, b) `// Retrieve out-of-bag predictions of the experts from the available` Y

  $e_i^j = |y_i - \hat{y}_i^j|$ `// Expert absolute loss in out-of-bag samples` $y_i \in Y$

  $z^j \leftarrow e_i^j = f(x_i)$ `// training meta-model` z^j

end

Return Z

4.2.3 Forecasting Upcoming Observations

For predicting the next value of the time series, y_{n+1}, `ADE` combines the output of the experts M according to the output of the arbiters, that is, the predicted loss that the experts will incur. We also take into account the recent correlation among the experts to encourage diversity during the aggregation of the base models.

Committee of Models for Prediction

In the original arbitrating architecture, the expert with the highest confidence (predicted by the arbiters) is selected to make a prediction (Ortega et al., 2001b). Our approach is to combine the output of the experts, as opposed to selecting a single one.

As described earlier, the predictive performance of forecasting models has been reported to vary over a given time series. We address this issue with a committee of models, where we trim recently poor performing models from the combination rule for an upcoming prediction (e.g. trimmed means (Jose and Winkler, 2008)).

As we explain in Section 4.1.3, the state of the art approaches for dynamic

combination in time series rely on past performance to quantify the weight of the experts. Specifically, this is typically used for dynamic selection (e.g. Jose and Winkler (2008)) or dynamic combination (e.g. Newbold and Granger (1974)). Here we use this information for dynamic selection. Formally, we select the $\Omega\%$ base-learners with lowest mean absolute error in the last λ observations ($^{\Omega}M$), suspending the remaining ones. The predictions of the meta-level models ($^{\Omega}Z$) are used to weigh the selected forecasters.

In summary, if we expect m^j to make a large error e^j in a given observation relative to the other experts, we assign it a small weight – or even suspending it – in the final prediction. Conversely, if we expect m^j to incur a small loss relative to its peers, we increase its weight for the upcoming prediction.

Combining the Experts

The weigh of an expert m^j in $^{\Omega}M$ is determined by a simple transformation of the predicted loss by the arbiters $^{\Omega}Z$. This is formalised by the following equation:

$$w^j_{n+1} = \frac{scale(-\hat{e}^j_{n+1})}{\sum_{j \in {}^{\Omega}M} scale(-\hat{e}^j_{n+1})} \tag{4.2}$$

where $\hat{e}^j_{n+1}$ is the prediction made by $z^j \in {}^{\Omega}Z$ for the absolute loss that $m^j \in {}^{\Omega}M$ will incur in y_{n+1}; w^j_{n+1} is the weigh of m^j for observation y_{n+1}; and scale denotes the min–max scaling function used to transform the vector of predicted loss into a 0–1 scale. The normalisation with respect to the summation in Equation 4.2 is performed so that the combination is convex, i.e., the weights sum to 1. The experts that are suspended (Section 4.2.3) are simply assigned a weight of 0.

Sequential Re-weighting of Experts

Most combination approaches, dynamic ones particularly, weigh experts by maximising estimates of predictive performance (c.f. Section 2.2.5). However, in cases where the experts are highly redundant, it is important to model their inter-dependence.

Brown et al. (2005b) stress that the diversity among experts is a critical

component for increasing the ensemble's predictive performance. To address this problem, Jacobs (1995) points out that ensemble methods require:

a. "training procedures that result in relatively independent experts";

b. "aggregation methods that explicitly or implicitly model the dependence among the experts".

We address the first issue (a.) implicitly by focusing on heterogeneous ensembles. These are comprised of experts with different inductive biases. The second issue (b.) is addressed explicitly by re-weighting the experts at each prediction point according to their recent correlation.

For clarity, we have at this point and for a given time instance y_{n+1}:

- the output of the experts $\hat{y}_{n+1}^{M} = \{\hat{y}_{n+1}^{1}, \ldots, \hat{y}_{n+1}^{s}\}$;
- and their respective weights predicted by the arbiters and scaled accordingly:
  $w_{n+1}^{M} = \{w_{n+1}^{1}, \ldots, w_{n+1}^{s}\} : \sum_{i=1}^{s} w_{n+1}^{i} = 1.$

To model the inter-dependence among experts, we frame their aggregation as a ranking task, in which experts are ranked sequentially by their decreasing weight (the one predicted to perform better is ranked first). The intuition for the ranking approach is borrowed from the information retrieval literature. For example, the algorithm Maximal Marginal Relevance (Carbonell and Goldstein, 1998) ranks a list of documents to answer a given query by maximising a function that couples the relevance and redundancy of documents. As such, the value of the second most relevant document (with respect to a given query) also depends on its redundancy to the most relevant document. The point is to emphasise the novelty of information in the document set and enhance their complementarity.

Notwithstanding, time series comprise characteristics that this type of methods need to cope with, e.g. the variance in the relative performance that forecasters show over a time series. We formalise our idea for the dynamic combination of forecasting models in Algorithm 2. We use the correlation among the output of the models to quantify their redundancy. This correlation is computed in a window of recent observations to cope with eventual non-stationarities of time series.

A given expert i is penalised for its correlation to each expert j already ranked. This penalty is determined by the multiplication of the correlation and the weights of expert i and expert j (line 8). The penalty formula takes a multiplication because its elements work on one another: if an expert m^i is fully correlated with other experts already ranked ($m^j \in {}^{\Omega}M \setminus {}^{\Omega}m^i : w^j > w^i$), its weight is absorbed by the latter and the weight of m^i becomes zero. Conversely, if m^i is completely uncorrelated with its ranked peers, m^i is ranked with its original weight. This approach allows the control of redundant information in the output of the experts. A practical advantage of this method is that it requires no parameter tuning, except for the correlation function.

In summary, we propose a method that encourages diversity during the aggregation of experts. This is accomplished by manipulating the experts' weights according to the correlation of their output. To the best of our knowledge, there is no closely related approach in the machine learning literature. However, our approach is inspired by the notions of *diversity* in the context of information retrieval. Particularly, the Maximal Marginal Relevance (Carbonell and Goldstein, 1998) method, which is typically used to rank a list of documents to answer a given query by considering not only the relevance of each document individually but also their redundancy to documents already ranked.

The final prediction is the weighted average of the predictions made by the experts $\hat{y}^j$ with respect to their re-weighted relevance w'^j_{n+1} (Algorithm 3):

$$\hat{y}_{n+1} = \sum_{j \in {}^{\Omega}M} \hat{y}^j_{n+1} \cdot w'^j_{n+1} \tag{4.3}$$

4.3 Empirical Experiments

In this section, we present the experiments carried out to validate `ADE`. We start by describing the overall setup. We compare the proposed method to state of the art approaches for combining the output of experts. Specifically, we focus on approaches designed to cope with temporal dependencies. Afterwards, we perform sensitivity analyses to enhance our understanding of the components of `ADE`. To encourage reproducible research, we published the code used to perform these experiments (c.f. footnote 2 in Section 4.1.2).

Algorithm 2: Sequential re-weighting of experts

input : predictions of experts in the last λ observations: $\hat{y}^{M}_{(n-\lambda):(n+1)}$

input : weight of experts for n+1: W

output: re-estimated weights W'

W $\leftarrow$ Sort(W, decreasing) `// sort weights in decreasing order`

$W' \leftarrow \{\}$ `// List with final weights`

$W'_1 \leftarrow W_1$ `// First element of` W' `is the weight of the predicted to be the best expert`

foreach *remaining expert i in W* **do**

  $W'_i \leftarrow W_i$

  foreach *expert j in* W' **do**

    $cor_{ij} \leftarrow \text{Cor}(\hat{y}^{i}_{(n-\lambda):(n+1)}, \hat{y}^{j}_{(n-\lambda):(n+1)})$ `// Correlation between the predictions of expert i and expert j in the last` λ `observations`

    $\eta_{ij} \leftarrow cor_{ij} \cdot W'_j \cdot W'_i$ `// penalty that expert j applies to expert i`

    $W'_j \leftarrow W'_j + \eta_{ij}$

    $W'_i \leftarrow W'_i - \eta_{ij}$

  end

end

return W'

4.3.1 Research Questions

The experiments were designed to answer the following research questions:

RQ2.1: How does the performance of the proposed method compares to the performance of the state-of-the-art methods for time series forecasting tasks and state of the art methods for combining forecasting models?

RQ2.2: Is it beneficial to use a weighting scheme in our arbitrating strategy instead of selecting the predicted best expert as originally proposed (Ortega et al., 2001a)?

RQ2.3: Is it beneficial to use out-of-bag predictions from the training set to

Algorithm 3: Forecasting $\hat{y}_{n+1}$

input : Time series Y up to time n

input : Experts M

input : Arbiters Z

input : Committee ratio Ω

input : Window size λ

output: $\hat{y}_{n+1}$

1 ${}^{\Omega}M \leftarrow$ Subset(M, λ, Ω)

2 ${}^{\Omega}Z \leftarrow$ Subset(Z, λ, Ω) `// Form the committees` ${}^{\Omega}M$ `and` ${}^{\Omega}Z$ `according to performance on the last` λ `observations`

3 Get loss predictions $\hat{e}^j_{n+1}$ from $z^j \in {}^{\Omega}Z$

4 Compute weights $w^j_{n+1} = scale(-\hat{e}^j) / \sum_{j \in {}^{\Omega}M} scale(-\hat{e}^j)$

5 Get predictions $\hat{y}^j_{(n-\lambda):(n+1)}$ from $m^j \in {}^{\Omega}M$ `// Predictions of selected experts in the last` λ `observations`

6 Apply **Algorithm 2** to weights: $w'^j_{n+1} \leftarrow$ SequentialReweight($\hat{y}^j_{(t-\lambda):(n+1)}$, w^j_{n+1}) `// Calibrate weights according to expert's correlation`

7 Compute final prediction $\sum_{j:m^j \in {}^{\Omega}M} \hat{y}^j_{n+1} \cdot w'^j_{n+1}$

increase the data used to train the meta-learners?

RQ2.4: How does the performance of `ADE` vary by the introduction of a committee, where poor recent base-learners are discarded from the upcoming prediction, as opposed to weighing all the models?

RQ2.5: What is the impact of the sequential re-weighting procedure in `ADE`'s performance?

RQ2.6: How does the performance of `ADE` vary by using different updating strategies for the base and meta models?

RQ2.7: How sensitive is `ADE` to the parameters Ω and λ, and the size of the ensemble in terms of the number of experts?

RQ2.8: How does it scale in comparison to other state of the art approaches for the combination of forecasters in terms of computational effort?

RQ2.9: What is the impact of the sequential re-weighting procedure in state of the art approaches for combining experts? Moreover, how does this approach compare with methods that handle correlation in the feature space (e.g. principal components analysis)?

4.3.2 Experimental Design

To address the research questions, we used 62 real-world time series from several domains. These are briefly described in Table A.1 in the appendix A. We limited the time series portfolio by size: we use time series with size above 750 for having enough data to fit both the experts and the arbiters; and size below 3000 in the interest of computational efficiency. Similarly to the experimental design in the previous chapter, we model the trend in the data according to a KPSS statistical test (Kwiatkowski et al., 1992). Further, we also use the False Nearest Neighbors approach to estimate the optimal order of the auto-regressive process or embedding dimension.

The feature set used by the forecasting models includes the previous p values (embedding vector). In order to enhance the representation of the time series, this approach can be extended by using summary statistics on the embedding vectors, or other external domain-specific knowledge. We compute the following summary statistics in each embedding vector in order to further characterise the recent dynamics of the time series:

- Recent trend, which is estimated according to the ratio between the standard deviation of the embedding vector and the standard deviation of the differenced embedding vectors;
- Skewness, for measuring the symmetry of the distribution of the embedding vectors;
- Mean, as a measure of centrality of the embedding vectors;
- Standard deviation, as a dispersion metric;
- Serial correlation, estimated using a Box-Pierce test statistic;
- Long-range dependence, using a Hurst exponent estimation with wavelet transform;

- Chaos, using the maximum Lyapunov exponent to measure the level of chaos in the system.

These statistics are commonly used to summarise the overall structure of time series (Wang et al., 2009). The meta-learning models use the same feature set used by the base forecasters.

The final representation of the time series is exemplified in the following matrix:

$$Y_{[n,p]} = \left[\begin{array}{cccccccc|c} y_1 & y_2 & \dots & y_{p-1} & y_p & trend_1 & \dots & chaos_1 & y_{p+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{i-p+1} & y_{i-p+2} & \dots & y_{i-1} & y_i & trend_i & \dots & chaos_i & y_{i+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{n-p+1} & y_{n-p+2} & \dots & y_{n-1} & y_n & trend_n & \dots & chaos_n & y_{n+1} \end{array}\right]$$

Taking the first row of the matrix as an example, the target value is y_{p+1}, while the attributes are the previous p values $\{y_1, \dots, y_p\}$ along with the above-mentioned statistics $\{trend, \dots, chaos\}$. In the meta-level, the target value is replaced by the absolute loss of a predictive model in that observation.

Evaluation Procedure

The methods included in the experiments were evaluated using RMSE. Regarding the performance estimation method, we resorted to the approach denoted as `Rep-Holdout` in the previous chapter. We presented evidence that this approach provides competitive estimates of predictive performance relative to other methods. This procedure was applied in 10 testing periods. Each repetition uses 60% of the time series size n for training, while the subsequent 10% observations are used for testing.

Ensemble Setup and Baselines

The set M of s experts forming the ensemble are formed by the following learning algorithms:

`SVR` Support vector regression (Karatzoglou et al., 2004);

`MARS` Multivariate adaptive regression splines (Milborrow, 2016);

`RF` Random forest (Wright, 2015);

`PPR` Projection pursuit regression (R Core Team, 2013);

`RBR` Rule-based regression (Kuhn et al., 2014);

`GBR` Generalised boosted regression (Ridgeway, 2015);

`MLP` Multi-layer perceptron (Venables and Ripley, 2002);

`GLM` Generalised linear model regression (Friedman et al., 2010);

`GP` Gaussian processes (Karatzoglou et al., 2004);

`PCR` Principal components regression (Mevik et al., 2016);

`PLS` Partial least squares (Mevik et al., 2016);

`ARIMA` Auto-Regressive Integrated Moving Average (Hyndman et al., 2014);

`ETS` Exponential smoothing (Hyndman et al., 2014).

Above, we provide the reference for the respective implementation of the methods. These learning algorithms are further described in Table A.3 in appendix A.2. Different parameter settings are used for each of the individual learners, adding up to **50** base models. This choice of the number of experts will be analysed in Section 4.3.4.

As an exploratory analysis, we show in Figure 4.2 the distribution of the rank of each expert across the 62 problems. A rank of 1 means that the respective model was the best performing one in a given data set. Generally, the range of the distribution of rank is large, and even the experts with high median rank are among the best in some of the time series problems. The rule-based model `RBR`, a variant of the *model tree* proposed by Quinlan (1993), presents a remarkable rank distribution. This was the reasoning which led to the choice of `RBR` as the predictive model for the study presented in the previous chapter. This graphic validates the *No Free Lunch* theorem in an empirical way. Finally, we remark that the name of the method in the x-axis is repeated because, as we mentioned above, several parameters are tested for each learning algorithm.

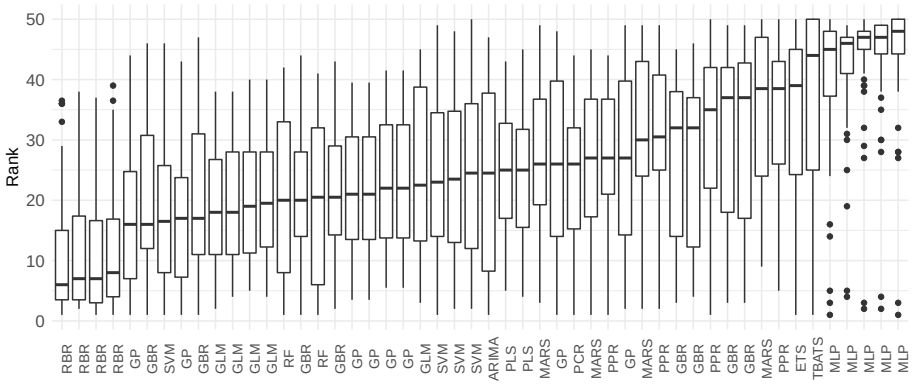


Figure 4.2: Distribution of rank of the base models across the 62 problems

We use a Random Forest as meta-model for each of the experts. We tested different alternatives, and this method provided the best results. The blocked prequential procedure used to obtain out-of-bag samples was run with ten folds ($b = \mathbf{10}$). The committee for each prediction (Section 4.2.3) contains 50% of the forecasters with the best performance in the last 50 observations (Ω and λ values are set to $\mathbf{50}$). We suspend only half the models in the interest of keeping the combined model readily adaptable to changes in the environment. An average performing model may rapidly become important, and the combined model should be able to capture these situations. By setting λ to 50, we strive for estimates of recent performance that renders a robust committee. The sensitivity of `ADE` to different values of Ω and λ is analysed in Section 4.3.4. We used Pearson's method as the correlation function for the sequential re-weighting of experts (Section 4.2.3).

We compare the performance of `ADE` with the following approaches for combining a set of forecasting models:

`Stacking`: An adaptation of stacking (Wolpert, 1992) for times series, where a meta-model is learned using the base-level predictions as attributes. To preserve the temporal order of observations, the out-of-bag predictions used to train the meta-learner (a random forest) are obtained using a blocked prequential procedure (c.f. Section 4.2.2). Different strategies for training the meta-learner (e.g. holdout) were tested and blocked prequential presented the best results;

`Arbitrating`: An approach following the original arbitrating method presented by Ortega et al. (2001a). In Section 4.1.3 we explain the key differences of this approach to `ADE`;

`Simple`: The approach in which the available experts are simply averaged using the arithmetic mean (Timmermann, 2006);

`SimpleTrim`: A variant of the simple average which includes model selection: $\Omega\%$ of the best past performing models are selected and aggregated with a simple average;

`WindowLoss`: Weighted adaptive combination of experts. The weights are computed according to the performance of the experts in the last λ observations (Newbold and Granger, 1974). Particularly, the weights are derived according to the mean squared error of the individual models;

`Blast`: Similar to WindowLoss, but selects the best expert in the last λ observations for prediction. Previous work by van Rijn et al. (2015) showed its competitiveness using streaming data for classification predictive tasks;

`AEC`: The adaptive combination procedure AEC (Sánchez, 2008) which is based on an exponential re-weighting strategy and a fading factor to give more importance to recent observations;

`ERP`: The adaptive combination procedure proposed by Timmermann (2008) which only combines the available models when their explained variance is sufficiently high, otherwise the prediction for the next value is set to be the average of recent observations;

`EWA`: The exponentially weighted average is a widely used online convex aggregation rule which is based on the Weighted Majority Algorithm by Littlestone and Warmuth (1994) – we refer to the seminal work by Cesa-Bianchi and Lugosi for a comprehensive description and theoretical properties. (Cesa-Bianchi and Lugosi, 2006, Section 2.1);

`FixedShare`: The fixed share approach due to Herbster and Warmuth (1998), which is designed for tracking the best expert across a time series (Cesa-Bianchi and Lugosi, 2006, Section 5.2);

`MLpol`: The polynomially weighted average forecast combination (Cesa-Bianchi and Lugosi, 2003). See Cesa-Bianchi and Lugosi for a comprehensive description and theoretical properties. (Cesa-Bianchi and Lugosi, 2006, Section 2.1) ;

`OGD`: An approach based on online gradient descent that provides theoretical loss bound guarantees (Zinkevich, 2003);

`LossTrain`: A baseline combination of experts in which the weights are set according to the performance of experts in the training set. The weights are static, which means that these do not change over time;

`BestTrain`: Another baseline combination approach that selects the individual model in the ensemble with the best performance in the training data. This model is used to predict all the observations in the test set.

For a complete description of these combination methods we refer the reader to the section (2.2) in the Background chapter regarding forecasting methods. For the approaches `EWA`, `MLpol`, `FixedShare`, and `OGD`, we used the software package *opera* (Gaillard and Goude, 2016). We also include the following forecasting baselines in our experimental setup:

`ARIMA`: A state-of-the-art method for time series forecasting. We use the *auto.arima* implementation provided in the forecast R package (Hyndman et al., 2014), which automatically tunes `ARIMA` to an optimal parameter setting;

`Naive` Typical forecasting baseline that uses the value of the previous observation (y_n) for predicting y_{n+1};

`SeasonalNaive`: Another typical forecasting baseline that uses the value of the observation from the previous seasonal period for predicting y_{n+1} (Hyndman et al., 2014). Particularly, for daily time series we use the value from the previous week, and for hourly time series we use the value from the day before;

`ExpSmoothing`: The exponential smoothing state space model typically used for forecasting (Hyndman et al., 2014). We use the *ets* implementation provided in the forecast R package (Hyndman et al., 2014). This implementation automatically tunes `ExpSmoothing` to an optimal parameter setting.

These four methods are not ensembles but are widely used for time series forecasting. Finally, the following variants of `ADE` were also tested:

`ADE-selectbest:` A variant of `ADE` in which at each time point the best model is selected to make a prediction. Here best is the one with the lowest predicted loss. This is in accordance with the original arbitrating architecture (Ortega et al., 2001a);

`ADE-allmodels:` A variant of `ADE`, but without the formation of a committee. In this case, all forecasting models are weighed according to their expertise in the input data;

`ADE-noreweight:` A variant of `ADE` in which there is no reweight of the experts according to the correlation of their predictions (Section 4.2.3);

`ADE-v0:` Early experiments performed with `ADE` indicated that using the softmax function for computing the weights of the individual models lead to a better predictive performance relative to a linear transformation. This softmax function is widely used in the literature of neural networks (Jiang et al., 2018). We include in our experiments its application with `ADE`. Besides using the softmax function, this variant does not reweight the importance of the experts according to their correlation, similarly to `ADE-noreweight`;

`ADE-vanilla:` A baseline variant of `ADE` with a simpler weighting approach: the error $(\hat{y}-y)$ predicted by arbiters is simply added to the output of the respective expert. The final prediction is computed according to the average of the shifted output of experts.

4.3.3 Comparing to the State of the Art

We evaluate the results of the experiments from multiple perspectives. This includes a formal evaluation according to the Bayesian analysis described by Benavoli et al. (2017). Particularly, we employed the Bayesian correlated t-test to compare pairs of models in a single problem and the Bayes sign test to compare pairs of methods across multiple problems. We define the ROPE to be the interval [-0.01, 0.01]. Essentially, this means that two methods show indistinguishable performance if the difference in performance between them

falls within this interval. In this Bayesian analysis of the results, we make a small change to the performance metric. Since RMSE varies according to scale, we normalise this value relative to the RMSE of the `Simple` aggregation approach, which is a standard forecast combination baseline. To be more precise, for each aggregation method `Agg`, we compute the following value:

$$\text{nRMSE}(\texttt{Agg}) = \text{RMSE}(\texttt{Agg})/\text{RMSE}(\texttt{Simple})$$

Figures 4.3 and 4.4 represent the average rank, and respective standard deviation, of `ADE` and its variants, state of the art approaches for forecast combination and other typical forecasting baselines. Figure 4.5 shows the log percentual difference in RMSE of `ADE` relative to other forecasting approaches. For this specific analysis, the initial outliers in the results were removed for better visualisation of the difference in performance. Figure 4.6 show the results of the Bayes sign test. This illustrates the proportion of probability that `ADE` wins, draws (result within the ROPE), or loses with each respective method. Table 4.1 presents the paired comparisons between the proposed method and all other approaches using the Bayesian correlated t-test. The numbers represent wins, draws, and losses of the proposed method. The numbers in parenthesis represent wins/draws/losses with a probability above 95%.

Comparing `ADE` to the State of the Art Approaches

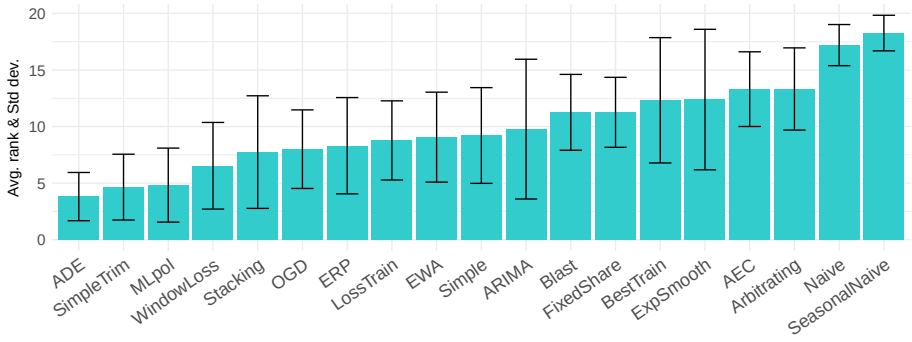


Figure 4.3: Average rank and respective standard deviation of `ADE` and state of the art methods

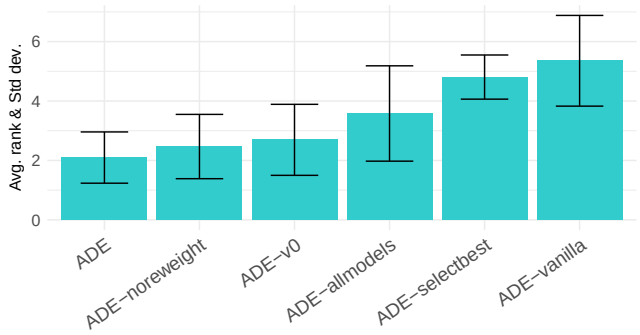


Figure 4.4: Average rank and respective standard deviation of `ADE` and its variants

`ADE` presents the best average rank relative to state of the art aggregation methods. This value is considerably better compared to widely used approaches, including `Stacking`, `Simple`, or `WindowLoss`. From the numbers of Table 4.1, `ADE` wins in most of the problems against other approaches, most of the times in a considerable way (i.e., with a probability above 95%). Among the combination approaches, `BestTrain` presents one of the lowest average ranks, which suggests that the combination of different experts is worthwhile in terms of predictive performance. The simple average aggregation coupled with model selection (`SimpleTrim`) leads to an interesting average rank, which is only topped by that of `ADE`. These results are corroborated by the outcome of the Bayes sign test, which suggests that `ADE` has a higher probability of winning compared to each other approach.

Figure 4.5 is useful for visualising the magnitude in the difference in predictive performance, something which average ranks are blind to. The distribution of the percentual difference varies according to the model under comparison. In general, `ADE` shows a reasonable difference when compared with most of the other approaches. These results answer the research question **RQ2.1** regarding the performance of `ADE` relative to the state of the art approaches for combining forecasting models.

Relative to the original arbitrating architecture denoted as `Arbitrating`, the proposed method shows a considerable improvement, which results in a much better average rank. This proves that the introduced components are

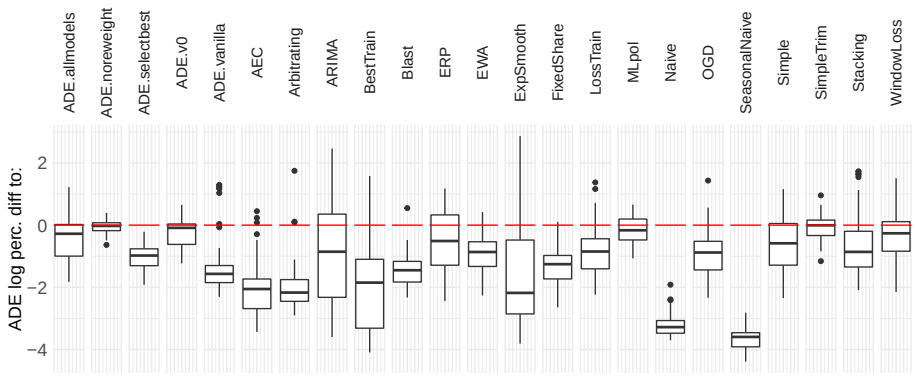


Figure 4.5: Distribution of the log percentual difference in performance of `ADE` relative to other forecasting methods. Negative values denotes better performance by `ADE`.

fundamental for the achieved performance, which answers question **RQ2.2**.

Comparing `ADE` to its Variants

`ADE` shows a consistent advantage over the performance of `ADE-allmodels` (**RQ2.4**). This suggests that indeed, it is worthwhile to prune the ensemble for each prediction (as opposed to combining all the forecasters). The performance of `ADE` is also considerably better relative to `ADE-selectbest`, which gives evidence for the hypothesis that the combination of experts (as opposed to selection) provides better results (**RQ2.3**). `ADE` is also superior to `ADE-vanilla`, which bypasses the weighting scheme, directly adjusting the output of the experts according to the predictions of the arbiters.

`ADE` shows consistent improvement over the variant that does not perform a sequential re-weighting of the experts according to recent correlation, i.e. `ADE-noreweight` (Section 4.2.3) (**RQ2.5**). The magnitude of the difference in performance is small (Figure 4.5), which is corroborated by the high number of draws shown in Table 4.1. However, it is important to note that the sequential re-weighting method does not generally compromise performance (only one loss in 62 problems), and improves it several times. Finally, `ADE` also shows a systematic improvement over its variant `ADE-v0`. Besides not using the sequential re-weighting approach, `ADE_v0` aggregates the output of the experts using

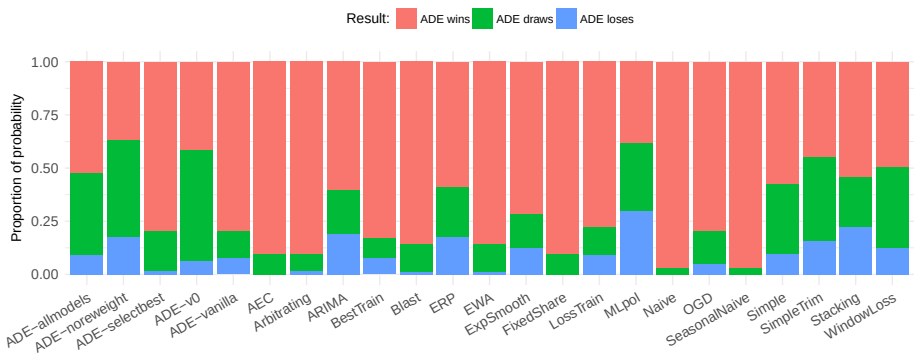


Figure 4.6: Proportion of probability of `ADE` winning/drawing/losing according to the Bayes sign test

a softmax function. We tested this approach in the experimental setup of this work and found that it does not improve the results over a linear transformation. The variant of `ADE` that also does not apply the sequential re-weighting approach (`ADE-noreweight`) also shows a better performance relative to `ADE_v0`. The difference between these two approaches is the weighting function (linear transformation and softmax transformation).

4.3.4 Further Analyses of `ADE`

Following the comparison of `ADE` with the state of the art, in this section, we provide a more detailed analysis of its workflow. The goal is to enhance our understanding of how the method works. This analysis encompasses: (i) an analysis of the different possible deployment strategies; (ii) a sensitivity analysis on the parameters Ω and λ; (iii) a scalability analysis in terms of relative computational time; (iv) a study on the impact of adding additional experts; and (v) additional analysis of the sequential re-weighting method. If not stated otherwise, the experimental setup is the same as described previously.

Analysing Training Strategies

In this section, we address the research questions **RQ2.4** and **RQ2.6**. In a dynamic environment, it is common to update the model over time, either online or in chunks of observations. Time-dependent data is prone to changes in the

Table 4.1: Paired comparisons between ADE and the baselines in the 62 time series. Number in parenthesis represent a probability of win/draw/loss above 95% according to the Bayesian correlated t-test.

Method	ADE **loses**	ADE **draws**	ADE **wins**
Stacking	12 (3)	16 (2)	34 (13)
Arbitrating	1 (0)	2 (0)	59 (41)
Simple	3 (1)	24 (17)	35 (24)
SimpleTrim	4 (1)	45 (32)	13 (10)
LossTrain	3 (1)	21 (8)	38 (25)
WindowLoss	3 (2)	35 (26)	24 (19)
Blast	1 (0)	2 (0)	59 (42)
AEC	0 (0)	6 (4)	56 (47)
ERP	8 (1)	21 (8)	33 (23)
BestTrain	3 (0)	6 (0)	53 (42)
EWA	0 (0)	23 (8)	39 (9)
FixedShare	0 (0)	6 (2)	56 (27)
MLpol	5 (2)	43 (26)	14 (2)
OGD	1 (0)	23 (8)	38 (19)
ARIMA	8 (5)	17 (7)	37 (33)
Naive	0 (0)	0 (0)	62 (61)
SeasonalNaive	0 (0)	0 (0)	62 (62)
ExpSmoothing	6 (5)	10 (4)	46 (45)
ADE-selectbest	1 (1)	11 (2)	50 (24)
ADE-allmodels	3 (1)	34 (22)	25 (16)
ADE-noreweight	1 (0)	53 (47)	8 (6)
ADE-v0	1 (1)	46 (31)	15 (9)
ADE-vanilla	5 (1)	2 (0)	55 (34)

underlying distribution, and continuous training of models ensures that one has an up-to-date model. Since ADE settles on two layers of models, we analysed different approaches for updating these and study their implications in terms of predictive performance.

In the main experiments, `ADE` is trained using only the training data. To understand if and how should `ADE` be updated over time, we tested the following strategies:

`M0_Z0:` both experts (M) and arbiters (Z) are trained in the training set and not updated during test time (`ADE` as reported in the main experiments);

`M0_Z1:` M is trained only in the training data, but Z is re-trained every Δ observations.

`M1_Z0:` M is re-trained every Δ observations, but Z is trained only in the training data.

`M1_Z1:` Both M and Z are re-trained every Δ observations, which is particularly interesting if the models in M are typical online methods (e.g. `ARIMA`);

`ADE-runtime:` A variant of `ADE` in which there is no blocked prequential procedure to obtain out-of-bag samples to increase the data provided to the meta-learners. In this scenario, the arbiters are trained in data obtained only at run-time every Δ observations, which is also in accordance with the original arbitrating strategy and other meta-learning approaches used in time-dependent scenarios (Gama and Kosina, 2014). M is fit only in the training data.

We set Δ to 100. In the interest of robustness, this analysis was carried out using the time series of size 3000 (33 datasets – see Table A.1). Since the predictive models are updated frequently, in this particular analysis, we settled for a simple holdout estimation procedure, where the training consists of the initial 70% of the data. The test set is comprised of the remaining 30% observations.

The results are presented in Figure 4.7, with a barplot representing the average rank and respective standard deviation of each deployment strategy.

`ADE` (also denoted as `M0_Z0` in this particular analysis) shows a better average rank relative to `ADE-runtime`, which suggests that it is better to get out-of-bag predictions from the available data to improve the fit of the meta-learners.

The results also suggest that updating the experts and the arbiters at run-time is better than not updating them. This outcome is expected due to the eventual presence of concept drift (Gama et al., 2014). Particularly, the `M1_Z1`

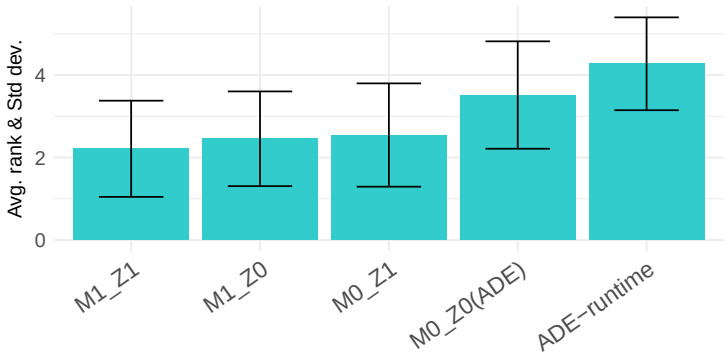


Figure 4.7: Average rank and respective standard deviation of ADE's deployment strategies

approach presents the best average rank. Although the difference in average rank is negligible, the results also suggest that updating the experts and not updating the arbiters (M1_Z0) renders a better average rank than the inverted strategy (M0_Z1).

Sensitivity Analysis on Ω and λ

In this and the next subsection, we answer the research question **RQ2.7** regarding the sensitivity analysis of ADE. Besides the setup of experts and arbiters, ADE has two main parameters: Ω, which represents the ratio of experts selected at each time step for forecasting; and λ, which denotes the window size used to compute the performance of the experts (for selecting which ones to arbitrate).

To some extent, these parameters are dependent not only on the ensemble composition but also on the data itself. In this section, we briefly analyse how the performance of ADE varies as the values of the parameters Ω and λ change. We considered ADE with $\lambda = \{3, 5, 10, 15, 25, 50, 60, 75,\ 100, 150, 300, 450, 600\}$ and $\Omega = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ (values chosen arbitrarily). This renders a total of 117 variants of ADE. This analysis was carried out using the 33 time series of size 3000.

The results are shown in Figure 4.8. The graphic illustrates two heatmaps. These relate the average rank (left heatmap) and respective standard deviation (right heatmap) of each (Ω, λ) combination across the 33 datasets. Higher

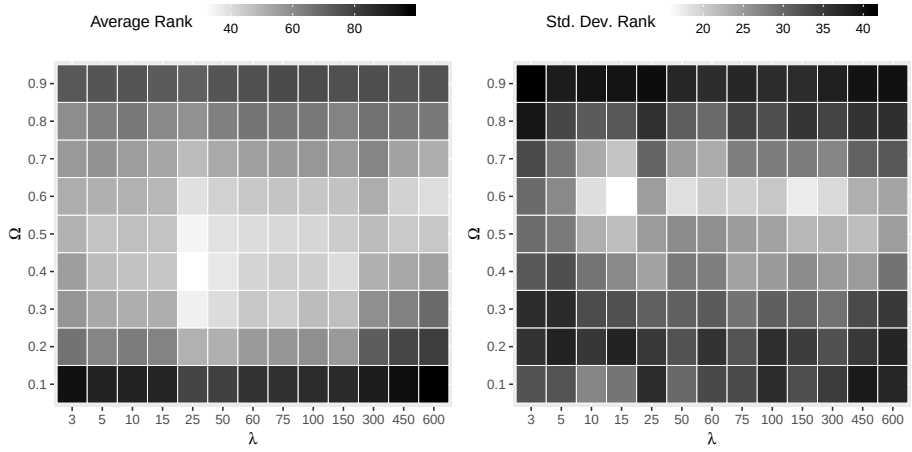


Figure 4.8: Heatmaps illustrating the average rank (left) and respective standard deviation (right) of `ADE` for varying Ω and λ parameters. Darker tiles mean higher values.

average rank (i.e. worse performance) and higher rank standard deviation are denoted by darker tiles.

Regarding Ω, the best performing values are the ones in the middle of the searched distribution. In principle, this parameter depends to a great extent on the number of experts and their predictive ability. The results also suggest that, unless for extremely low λ values, fixing Ω and varying λ renders a relatively stable average rank. The heatmap in the right side suggests that the (Ω, λ) combinations with the lowest rank standard deviation are in the middle of the searched distributions.

In principle and practice, varying the value of λ follows the stability-plasticity dilemma (Carpenter et al., 1991): small values of λ (i.e. small window of recent observations) lead to greater reactiveness, but also makes the model susceptible to outliers. Conversely, higher values lead to greater stability while losing some responsiveness and possibly containing outdated information.

Value of Additional Experts

In the experiments presented in the previous sections `ADE` was employed with 50 experts (Table A.3). In this section, we analyse the sensitivity of `ADE` to

different ensemble compositions. Particularly, we tested ensembles with sizes from 5 to 100 by multiples of 5: $S = \{5, 10, 15, \ldots, 95, 100\}$, rendering a total of 20 different possible ensemble sizes for analysis.

We estimate the predictive performance of each composition using a Monte Carlo approximation. Specifically, for each Monte Carlo repetition and for each considered size $s \in S$, we sampled without replacement s experts from a pool of 100, and compute the performance of `ADE` with this configuration. Afterwards, we measure the relative performance of each size (averaged across 30 Monte Carlo simulations) with respect to the performance obtained when using the complete pool of 100 experts. We tested in 30 Monte Carlo repetition to obtain robust estimates of performance. The pool of 100 experts was created by adding different values to the parameters described in Table A.3.

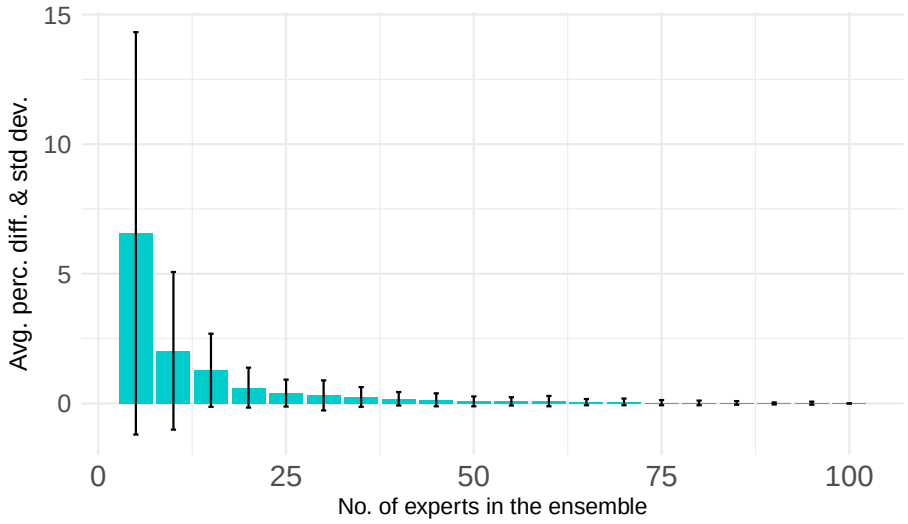


Figure 4.9: Average percentual difference in RMSE, and respective standard deviation, of `ADE` with different ensemble size compositions up to 100 models relative to `ADE` with 100 models.

The result of this analysis is presented in Figure 4.9. Generally, including more experts in the ensemble leads to better performance, and closer to that of the ensemble with 100 models. However, the difference becomes negligible for values above 50. The uncertainty in performance is represented by the vertical bars and is computed according to the standard deviation across the Monte

Carlo repetitions. This value also becomes increasingly small as more base models are included.

Scalability Analysis

In the previous sections, we have analysed `ADE` in terms of predictive performance. We analyse `ADE` in terms of computation time in this section. To accomplish, this we measure the time spent in fitting `ADE` and using it to predict the test set. We use the time spent by `ARIMA` and `SimpleTrim` as references. The first is a state of the art approach to forecasting, while the second is the aggregation approach with the closest average rank to `ADE`. We computed the time spent by `ADE` relative to these two approaches across the 62 time series.

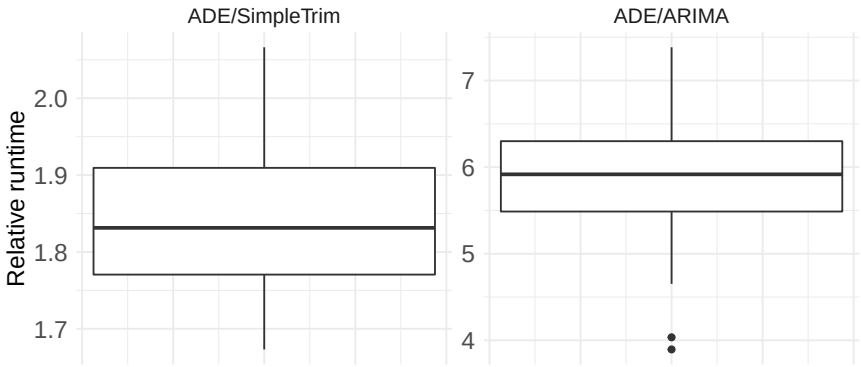


Figure 4.10: Log computational time spent by `ADE` relative to `ARIMA` and `SimpleTrim` approaches across the 62 problems

The results are presented in Figure 4.10 as boxplots. On all problems, `ADE` takes more time to run than `SimpleTrim`. The difference of this method to `ADE` is mostly driven by the fitting and predictions of the arbiters. As expected, `ADE` also takes more time than `ARIMA`. Being a single model (as opposed to an ensemble), `ARIMA` has considerable lower storage requirements when compared to `ADE`.

In summary, `ADE` scales worse than both approaches. Although omitted, it also takes more time than the remaining state of the art approaches used earlier (**RQ2.8**).

Further Analyses of the Sequential Re-weighting Procedure

In Section 4.2.3, we presented an approach for handling the inter-dependencies among experts during their aggregation. The core arbitrage approach does not explicitly model the inter-dependencies among experts, and this approach was designed to overcome this limitation. Particularly, in the previous section, we provided evidence of the benefits of the sequential re-weighting approach when applying it to `ADE`. The results suggest that the magnitude of the impact is not substantial. Notwithstanding, applying this method does not generally decrease performance and improves it several times.

In this section, we analyse the sequential re-weighting method from two more different perspectives, according to research question **RQ2.9**. First, we study the impact of applying this procedure to other state of the art approaches for the dynamic combination of forecasting models. In the interest of fairness, we focused this analysis only on approaches which perform dynamic expert combination using estimated weights. Second, and focusing on `ADE`, we compare sequential re-weighting of models with approaches that handle correlation in the feature space. Specifically and before training the experts, two different approaches are tested: (i) attributes with a correlation above 95% with other features are removed (`ADE-corr-noreweight`); (ii) principal components analysis is applied to the data, keeping 95% of the variance (`ADE-pca-noreweight`). The value of 95% was chosen arbitrarily. In this analysis, we also study `ADE-corr` and `ADE-pca`, where `ADE` is applied with sequential re-weighting and the methods (i) and (ii) described above.

The results of the first analysis are reported in Table 4.2, where each approach in the first column is compared with itself when using the sequential re-weighting approach. Similarly to Table 4.1, this table shows paired comparisons of the respective method with and without the application of the sequential re-weighting method. In parenthesis are denoted the results that happen with at least 95% probability according to the Bayesian correlated t-test.

Besides `ADE`, the results suggest that the approach is also beneficial to `WindowLoss`. However, when applied to the other tested approaches its impact vanishes and is often decreases the predictive performance.

Figure 4.11 shows the results of the second analysis. `ADE` shows the best

Table 4.2: Paired comparisons showing the impact of the sequential re-weighting approach in state of the art methods.

Method	without re-weight wins:	Draw	with re-weight wins:
`WindowLoss`	4 (2)	31 (24)	27 (24)
`AEC`	32 (22)	25 (18)	5 (1)
`EWA`	33 (14)	25 (21)	4 (0)
`FixedShare`	41 (24)	20 (18)	1 (0)
`MLpol`	27 (9)	27 (20)	8 (2)
`OGD`	21 (4)	26 (16)	14 (5)

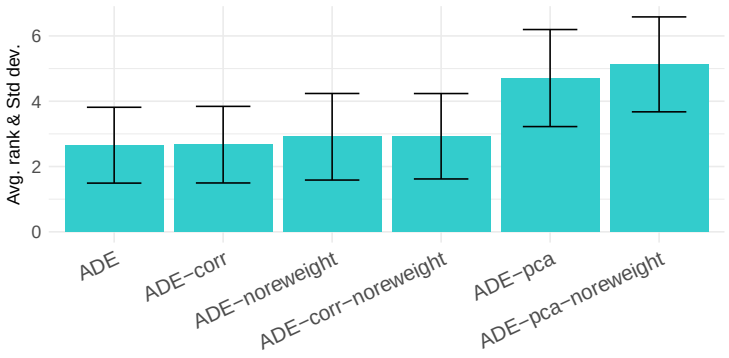


Figure 4.11: Average rank and respective standard deviation of `ADE` and its variants

average rank across the tested approaches. The average ranks suggest that applying the sequential re-weighting procedure improves the predictive performance in the three variants of `ADE`. Even when accounting for correlation in feature space, the sequential re-weighting approach still improves the average rank during expert aggregation.

4.4 Discussion

In this section, we discuss the results presented above. We start by addressing the limitations of `ADE`, where we also outline possible solutions to those

shortcomings. Then we overview some open issues regarding ADE.

4.4.1 On Concept Drift

Some of the design decisions behind ADE are based on prior work regarding the variance in the relative performance of forecasting models over a time series (Aiolfi and Timmermann, 2006) and with potential recurring structures present in the time series. However, there are cases in which time series change into new concepts and both the experts and arbiters may get outdated. Although we do not explicitly cover these scenarios, a possible strategy to address this issue is to track the loss of the ensemble. If its performance decreases beyond some tolerance, new base-learners could be introduced (e.g. Gama and Kosina (2014)) or existing ones re-trained. Since an arbitration approach provides a modular architecture, models can be added (or removed) as needed. Gama et al. (2014) survey several approaches for concept drift adaptation that also could be adopted.

4.4.2 On the Sequential Re-weighting Procedure

The core scheme of arbitrage (c.f. Figure 4.1) does not model the inter-dependency among experts. Each arbiter is responsible for its respective expert without any information regarding the other experts or arbiters. To cope with the inter-dependency among experts, we proposed a sequential re-weighting procedure that controls the redundancy among their outputs by considering their recent correlation. This approach is independent of ADE. However, its application with ADE is particularly interesting because the re-weighting occurs during aggregation and does not withhold ADE's modularity.

Despite the evidence of its benefits, the sequential re-weighting approach has space for improvement. Consider the following (rather extreme) example: one expert producing forecasts with a determined magnitude systematically below the true value, and another expert with similar behaviour but with forecasts above the true value. These two experts are highly correlated but complement themselves greatly. Effectively, using Pearson's correlation as a measure of similarity can be a sub-optimal solution in this case. Future work includes the exploration of better similarity functions. A possibly interesting line of enquiry

is to follow Brown's work on the study of diversity in classifiers from an information theoretic perspective (Brown, 2009). Particularly, instead of measuring the redundancy among experts only according to their outputs, we can also take into consideration the target value, i.e. conditional redundancy.

Finally, in our experiments, the application of the sequential re-weighting approach to other dynamic aggregation methods does not render the same positive effects as seen when it is applied to `ADE`. Notwithstanding, after the publication of `ADE` (Cerqueira et al., 2019), Boulegane et al. (2019) tested the sequential re-weighting approach. They found that this method "considerably improves the overall performance of the dynamic ensemble selection".

4.4.3 On Scalability

In the previous section, we identified the computational effort required by `ADE` relative to other approaches as its main limitation. Notwithstanding, Boulegane et al. (2019) extended `ADE` to a streaming scenario, which often implies many constraints in computational resources. Their approach, called `STREAMING-ADE`, focuses on online learning algorithms such as the Hoeffding Tree (Domingos and Hulten, 2000). They show that `STREAMING-ADE` is able to outperform a number of different alternatives for forecasting while dealing with concept drifts.

Besides using online learning algorithms, another possible solution to reduce the computational demand is to use a single arbiter (instead of one arbiter for each expert) designed for multi-target regression. That is, to have a single regression model that forecasts the errors of all base models. However, for ensembles with a large number of base models this can be cumbersome given the number of target variables we would have.

4.4.4 Model Selection Versus Model Combination

One of the core assumptions behind our work is the case of heterogeneous ensembles. By having an ensemble composed of different learning models, we expect these models to show different specialities across the input space of a given data set. We then dynamically combine the available models to manage the strengths and limitations of each model. As pointed out by Brown et al. (2005b), since models show a better relative performance in different points it may be more

sensible to adopt a selection approach to aggregate the predictions of the models. That is, instead of combining the output of the available models (or part of them) using a weighted average, we should select the model we trust the most. For example, van Rijn et al. (2015) follow this approach in the dynamic aggregation of a set of classifiers applied in a data streams environment (c.f. Section 2.2.5). Notwithstanding, our experiments regarding this issue suggested that a combination approach leads to better predictive performance.

4.4.5 Scope of the Experimental Setup

From a broad perspective, forecasting can be split into different *varieties*. In this work, we focus on univariate time series, assuming that only the variable of interest is available. We also centre our goal on predicting the next value of the time series and assume immediate feedback from the environment. However, in many application domains, one is often interested in predicting multiple steps in the future. Although we do not evaluate the proposed method in this setting, it can be extended to multi-step forecasting using state of the art approaches to this effect. We intend to study the application of `ADE` in these settings in future work.

As we describe in Section 4.3.2, we focus on time series with a high sampling frequency, specifically, half-hourly, hourly, and daily data. Finally, although it is important to test our model in different time series forecasting scenarios (e.g. with multivariate time series, multi-step forecasting), this work is focused on the dynamic aggregation of a set of forecasting models. Particularly, we did not find in the literature any work with such an extensive empirical experiment in terms of aggregation methods used.

4.4.6 Other Research Lines

We plan to address the previous limitations of `ADE` by exploring the described potential solutions. Besides these, there are other interesting open research questions. Specifically, we will study ways of quantifying and leveraging the uncertainty of the arbiters regarding the loss that the experts will incur. For example, one could develop an approach in which, when the uncertainty of the output of the arbiters is high, the weights are smoothed. This could be

accomplished efficiently using, for example, an infinitesimal jackknife (Wager et al., 2014) (provided random forests are used as arbiters).

In the formalisation of ADE, the base-level and meta-level models are trained using the same predictive variables. These are the past p lags of the time series, along with a number of summary statistics that help to characterise the dynamics of the data. In future work, it would be interesting to enrich the attributes used in the meta-level. For example, one could perform auto-regression on the errors of the respective base-model.

After the publication of ADE (Cerqueira et al., 2017b,c, 2019), a variant of this method was published in an article (Wang and Koprinska, 2018). Similarly to ADE, the approach proposed by the authors uses a meta-learning approach that models the loss of an ensemble of neural networks. They use this information to combine the output of the neural networks to predict future values of solar power.

4.5 Conclusions

In this chapter, we presented ADE, a novel method for the dynamic combination of a set of forecasting models. We focused on time series forecasting problems, where the objective is to predict future values of a sequence of observations.

ADE is comprised of a set of forecasting models pre-trained on the available data. A meta-learning approach is used to estimate the weight factors of these models at run-time dynamically. This is accomplished by having a set of arbiters that model the error of each model and predict how well they will perform in future observations. The resulting weights are used for obtaining the aggregated prediction of the ensemble. This aggregation may temporarily assign zero weight to some experts if their current performance is estimated to be too bad. This suspension decision can be revised in future time steps, thus contributing to the robustness of the approach to regime changes.

We argued that the proposed meta-learning approach is useful to better capture recurring changes in the environment. Particularly, long-range temporal dependencies (e.g. seasonal factors) that short-memory windowing approaches may fail to grasp efficiently.

Our proposal also includes a sequential re-weighting approach for modelling the inter-dependencies among experts. Specifically, this approach is designed to control and reduce the redundancy in the output of the experts during their aggregation. Within the proposed arbitrage approach, we also include a procedure for retrieving out-of-bag observations from the training set. These are used to fit the arbiters, significantly improving the data efficiency of the method.

We carried out an extensive empirical study to better characterise the performance of our proposal. This study has provided clear evidence on the competitiveness of our method in terms of predictive performance when compared to state of the art. We also discussed its limitations and provided guidelines for solving them in future work. The main point for improvement is the scalability of the method. We plan to address this issue and potentially adapt `ADE` to streaming or incremental scenarios.

In the interest of reproducible science, all methods are publicly available as a *R* software package.

Chapter 5

Constructive Aggregation

5.1 Introduction

In the previous chapter, we addressed the problem of time series forecasting. Our basic assumption is that all predictive models have some strengths and limitations. Therefore we adopted a dynamic ensemble approach to manage these. Essentially, the idea is to construct a portfolio of s different predictive models $M = \{m^1, m^2, \ldots, m^s\}$. These are then combined to predict the value of an upcoming observation $\hat{y}_{n+1}$ as follows:

$$\hat{y}_{n+1} = \sum_{i=1}^{s} \hat{y}^i \cdot w^i \tag{5.1}$$

where w^i denotes the weight of the i-th model. In this chapter, we continue to study the problem of dynamically estimating the weighting factors for each model in an ensemble.

In time series, the *strengths and limitations* described above are expressed in a variance in relative predictive performance shown by forecasting models (Aiolfi and Timmermann, 2006). We illustrate this as follows. Using an example time series[1] from the set of 62 shown in appendix A, we applied the 50 forecasting models which were used in the previous chapter and are also described in the appendix. We evaluate the rank of the 50 models in the different testing obser-

[1] We use time series id 29. We show the results on a single problem in the interest of conciseness. The same conclusions are valid on the remaining problems.

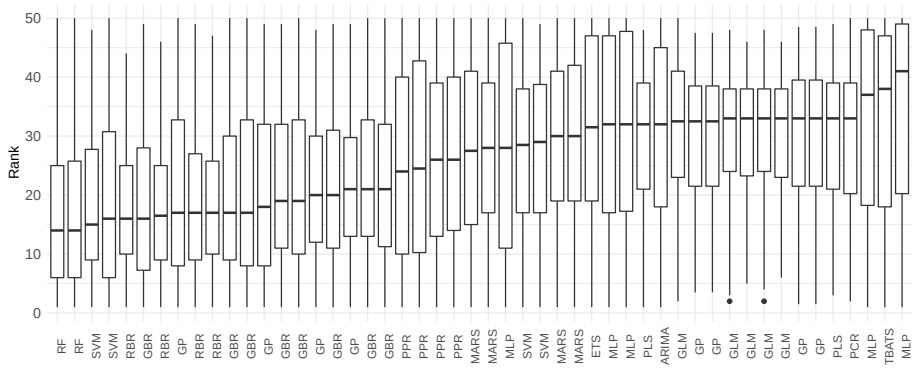


Figure 5.1: Distribution of rank of the forecasting models across the testing observations of a time series

vations of the time series, and show its distribution in Figure 5.1. This graphic has the same intuition as the one shown in Figure 4.2, which was presented in the previous chapter. The difference is that we now apply the predictive models on a single time series, and the rank is computed for each testing point. The results are quite interesting from the perspective of ensemble learning. Although some models show a better average rank, the figure shows that even the model with the lowest average rank is the best model at some time point in the time series. The point of this experiment is to emphasise the motivation behind the dynamic aggregation of different predictive models.

Our working hypothesis in this chapter is that similarly to what we observed above using the set of 50 individual models, different subsets of this portfolio may also show a varying relative performance throughout a time series. If this hypothesis is valid, it may be possible to combine these subsets, which are ensembles themselves, and eventually obtain further gains in predictive performance. Such an approach could be described as an ensemble of ensembles.

5.1.1 Related Work on Combining Different Ensemble Methods

Following the predictive advantage shown by ensemble methods, further gains have been reported by approaches that combine different ensemble learning methods. Webb (2000) developed `Multiboosting`, which combines `AdaBoost`

(Freund and Schapire, 1997) with `Wagging` (Bauer and Kohavi, 1999) (a variant of bagging (Breiman, 1996)). The idea is to use `AdaBoost` as the individual learning algorithm for `Wagging`. In a posterior work, Webb and Zheng (2004) claim that `Multiboosting` and other similar approaches provide a better trade-off between the error of the individual members of the ensemble and the diversity among them. Yu et al. (2007) presented an approach for combining a number of regression ensembles, dubbed `Cocktail Ensemble`, using the ambiguity decomposition. Wu et al. (2001) propose `E-GASEN` (Genetic Algorithm based Selective Ensemble method), a neural network ensemble. Essentially, this approach combines several `GASEN` ensembles using a simple average. A `GASEN` ensemble works by initially building a set of neural network models; afterwards, a genetic algorithm is run to prune the ensemble. In 2000, Pfahringer (2000) won the well-known KDD Cup competition with a combination of bagging and boosting ("bagged boosting"). The `EasyEnsemble` and `BalanceCascade` (Liu et al., 2009) are another two approaches that combine bagging with boosting, focusing on imbalanced learning problems.

A key distinguishing feature of this type of approaches is that they typically combine different ensemble methods during the learning process. We hypothesise that similar effects can be obtained from a portfolio of pre-trained heterogeneous models, like the ones used in the previous chapter. This is the main goal for the work in this chapter. This setting can be advantageous because experts in a portfolio are typically independent and thus easily parallelised (Caruana et al., 2004).

5.1.2 Our Approach: Constructive Aggregation

In this chapter, we propose an aggregation framework for a set of diverse and independently created models M, following the basic principles of constructive induction (Wnek and Michalski, 1994). Constructive induction refers to procedures that modify a set of original attributes, where some of the attributes are removed, others are added, and some of the existing ones are aggregated (Wnek and Michalski, 1994). This leads to a new set of attributes which hopefully provides an overall better description for approximating a regression or classification function f relative to the original set. We follow a similar approach, but

in our work, the attributes refer to predictive models.

The proposed aggregation framework works by rearranging a set of diverse models M into different, overlapping subsets (denoted as $\mathcal{C}_M$). The elements in each of these subsets are then aggregated, leading to a new set of models M' (or sub-ensembles, as denoted by Webb and Zheng (2004)). Similarly to constructive induction approaches, the search for $\mathcal{C}_M$ is done by analysing the individual predictive performance of the original models M in observations not used in the learning process (e.g. a validation set). Essentially, the new models $m' \in M'$ correspond to aggregated subsets of consistently top performing models from M. We refer to this approach as Constructive Aggregation (CA). CA can be thought of a hierarchical approach for aggregating a set of predictive models. Our working hypothesis is that similarly to approaches for combining different ensemble methods (Webb and Zheng, 2004), CA leads to a decrease in the individual error of ensemble members, without overly decreasing the diversity among them.

To illustrate our idea, the workflow of CA is presented in Figure 5.2 with $M = \{m^1, m^2, m^3\}$. After analysing the performance of each model in unseen observations, the set of committees $\mathcal{C}_M = \{\{m^1\}, \{m^1, m^2\}, \{m^2, m^3\}\}$ is created; then the models m^i within each committee are aggregated into models M'. Finally, the new set of combined models M' is aggregated into a final approximation $\hat{y}^{M'}$. Both of these aggregations are done according to a linear combination (Equation 5.1). The construction of the committee set $\mathcal{C}_M$ is carried out by applying the concept of out-performance contiguity (OC), which is also formalised in this chapter.

Although we have not formalised our approach yet, we present a motivating example similar to the one shown in Figure 5.1. As a preliminary analysis, we applied CA to the time series used in the experiment shown in the introduction of this chapter. We used the same portfolio of 50 models, which lead to the creation of 79 different subsets (we will explain how these are obtained in the next section). These subsets were aggregated and evaluated according to their rank in each testing point, whose distribution is illustrated in Figure 5.3. Similarly to the individual 50 predictive models, different groups of models show a varying relative performance in the testing observations of the time series.

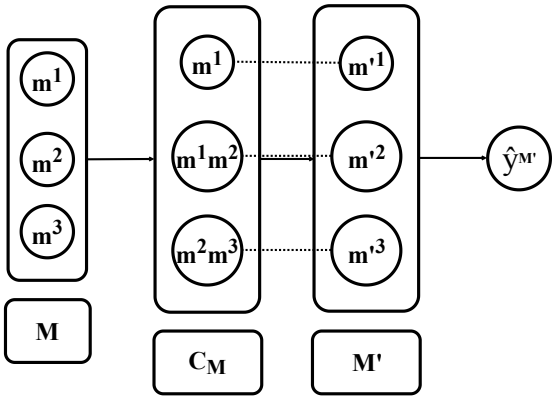


Figure 5.2: Workflow of constructive aggregation: the set of the available models M is rearranged into different subsets $\mathcal{C}_M$. Each subset is aggregated into M', and become hypotheses for approximating f. The models in M' are aggregated into the final decision $\hat{y}^{M'}$.

In experiments using the 62 time series from several domains (c.f. appendix A.1), aggregating a number of forecasting models using CA provides a consistent advantage in terms of predictive performance. That is, applying state of the art aggregation methods to M' leads to a better predictive performance relative to applying them to M. Moreover, we provide results that demonstrate that the constructive aggregation process entails a small execution time overhead. In summary, the contributions of this work are the following:

- Constructive Aggregation (CA), a new concept which consists in rearranging a portfolio of experts M into different subsets $\mathcal{C}_M$, aggregating them, and using them as hypotheses M';
- Out-performance Contiguity (OC), an approach for building $\mathcal{C}_M$ in time-dependent forecasting problems;
- An extensive set of experiments including: paired comparisons that quantify the percentual difference in error between using CA and aggregating M directly (non-CA); execution time analysis of CA using OC; sensitivity analysis of the main parameters behind the approach; and a study of CA in terms of bias-variance-covariance trade-off.

We present CA in the next section and formalise its application to forecasting

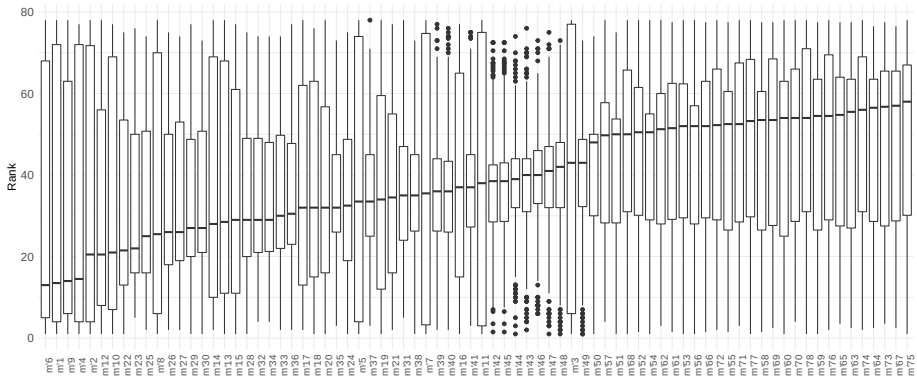


Figure 5.3: Distribution of rank of 79 aggregated groups of forecasting models across the testing observations of a time series

problems via OC. In Section 5.3, we provide an extensive set of experiments to validate the proposal. Afterwards, in Section 5.4, we discuss the results and limitations of our work. Finally, we summarise this chapter in Section 5.5. The proposed method is publicly available as an *R* software package[2].

5.2 Constructive Aggregation

Similarly to the previous chapter, we focus on the problem of forecasting future values of univariate time series. To recapitulate, we formalise the problem as an auto-regressive task, where the target variable is represented by the next value of the time series, and the predictor variables are represented by the past p lags of the data. Our approach is based on heterogeneous ensembles (Caruana et al., 2004): a set of models created in parallel and separated from each other. Diversity in the ensemble, a key ingredient in these methods (Brown, 2010b), is introduced by varying the learning algorithm used to train each model $m \in M$.

5.2.1 Methodology

We propose an aggregation approach for M based on constructive induction (Wnek and Michalski, 1994), and denote this idea as constructive aggregation (CA). CA works by rearranging the models $m \in M$ into different, possibly over-

[2] **tsensembler**: on CRAN or at `https://github.com/vcerqueira/tsensembler`.

lapping, subsets $\mathcal{C}_M$, and aggregating these into a new set of hypotheses M'. Given that forecasting models typically show varying relative performance over time (Aiolfi and Timmermann, 2006), our idea is that there may be different subsets of models that work well in different time intervals. As such, aggregating these subsets into different combined opinions (M') may lead to better representations for the function we want to approximate.

Similarly to related approaches in the literature (c.f. Section 5.1.1), this approach may result in a better trade-off between diversity and the individual error of the ensemble members. Particularly, we hypothesise that aggregating M' decreases the individual error without overly decreasing diversity (relative to aggregating M directly). CA can be split in the following three main steps (c.f. Figure 5.2):

- building $\mathcal{C}_M$ from M;
- aggregating the elements of $\mathcal{C}_M$ into a new set of hypotheses M';
- aggregating M' into a final decision.

In the next subsections, we will address each of these steps in turn.

5.2.2 CA for Forecasting via Out-performance Contiguity

We propose out-performance contiguity (OC) for building $\mathcal{C}_M$ from M. This approach is geared towards time series where observations are time-dependent.

Let $\mathcal{V}$ denote a set of validation observations. OC works by analysing the predictive performance of each model $m^i \in M \; \forall \; i \in \{1, \ldots, s\}$ in $\mathcal{V}$. More precisely, this procedure can be summarised as follows: a subset with size Tm of available models M is aggregated and used as an hypothesis for approximating f, if its elements are the top Tm performers (relative to M) for α contiguous observations. From the example in Figure 5.2, the subset $\{m^1, m^2\}$ is aggregated into an hypothesis $m' \in M'$ because m^1 and m^2 outperform m^3 during a contiguous time interval of size α.

This idea is formalised in Algorithm 4. Initially (lines 1–4), we compute the absolute error of each model in observations of validation data ($\mathcal{V}$). To control for outliers, the loss is smoothed using a moving average of window size

λ. Afterwards (lines 7–8), we compute the rank of each model $m \in M$ across $\mathcal{V}$, using the smoothed error. Then (line 11), for each possible size Tm of the subsets of M (except the size of the full set M), i.e. from 1 to $|M|$-1, we do as follows. All top-ranked Tm models across $\mathcal{V}$ are retrieved (line 12). If the models composing this top are unchanged for α consecutive observations, the respective subset becomes an element of $\mathcal{C}_M$ (lines 13–16). In the extreme case in which $\mathcal{C}_M$ is empty, we revert to using the original set of hypotheses M.

Algorithm 4: Out-performance contiguity for $\mathcal{C}_M$

input : set of hypotheses M;
validation set $\mathcal{V}$;
smoothing window λ;
contiguity window α

1 **foreach** *hypothesis* m^i *in* M **do**
2 **foreach** *observation* (x_j,y_j) *in* $\mathcal{V}$ **do**
3 $e^i_j \leftarrow |\hat{y}^i_j - y_j|$ `// absolute error of` m^i `in observation` j
4 **end**
5 $e'^i_j \leftarrow moving\_average(e^i_j, \lambda)$
6 **end**
7 **foreach** *observation* (x_j,y_j) *in* $\mathcal{V}$ **do**
8 $rnk_j \leftarrow rank(e'_j)$ `// rank of each hypothesis in observation` j
9 **end**
10 $\mathcal{C}_M \leftarrow \{\}$ `// list of committees`
11 **foreach** *size* Tm *from 1 to (*$|M|$*-1)* **do**
12 $TOP_T \leftarrow$ top Tm ranked hypothesis across $\mathcal{V}$
13 **foreach** τ *in* TOP_T **do**
14 **if** τ *is top ranked for at least* α *consecutive points* **then**
15 $\mathcal{C}_M \leftarrow \mathcal{C}_M \cup \tau$
16 **end**
17 **end**
18 **end**
19 return $\mathcal{C}_M$

Effectively, OC searches for groups of models that perform well in consecutive unseen observations to form $\mathcal{C}_M$. The goal of this search is to make a better exploration of the regions of the input space where these groups consistently perform better relative to other models in the available pool. Moreover, these groups may be relevant in the future due to the potential variance in relative performance shown by forecasters (Aiolfi and Timmermann, 2006), or by other recurring concepts that typically characterise time series (Gama et al., 2014).

5.2.3 Aggregation Steps

$\mathcal{C}_M$ into M'

The preceding subsection presents an approach for retrieving the set of subsets $\mathcal{C}_M$ from M. Most of the elements in $\mathcal{C}_M$ are potentially comprised of several models $m \in M$. As such, they need to be aggregated, in order to form a single combined opinion $m' \in M'$.

In this work, we accomplish this by using a windowing approach. Essentially, the elements $m \in M$ in each subset from $\mathcal{C}_M$ are aggregated according to their recent performance (Newbold and Granger, 1974; van Rijn et al., 2018). As we emphasised before in this thesis, the idea behind this method is that recent observations are more similar to the one we intend to predict, and thus, they are considered more relevant. More formally, the weight of each model m^i in a committee $cm \in \mathcal{C}_M$ is given by its relative loss in the last λ observations:

$$w^i = \frac{scale(-\overline{L}^i_\lambda)}{\sum_{i \in cm} scale(-\overline{L}^i_\lambda)} \tag{5.2}$$

where $\overline{L}^i_\lambda$ denotes the average loss of model i in the last known λ observations, and *scale* represents the min–max scaling function.

M' into $\hat{m}'$

The objective of the CA process, from M to M', is to create a new set of hypotheses that present a better approximation to f. Therefore, our working idea is that applying state of the art aggregation methods (e.g. `ADE`) to this new set of hypothesis M' is better than applying them directly to M. We will study this hypothesis in the next section.

5.3 Empirical Experiments

5.3.1 Research Questions

We carried out several experiments to validate CA via OC for forecasting with dynamic ensembles. These address the following research questions:

RQ3.1: How do state of the art approaches for the dynamic combination of forecasters perform when applied using CA relative to non-CA?

RQ3.2: What are the implications of CA in terms of bias, variance, and covariance?

RQ3.3: How sensitive is CA via OC to different values of α and λ?

RQ3.4: How does CA scale in terms of execution time relative to non-CA?

RQ3.5: Is CA simply pruning or avoiding poor models?

To address these questions, we used 62 real-world time series, which are briefly described in Table A.1 in Appendix.

5.3.2 Experimental Design

The experimental design is similar to the one employed in the previous chapter. Forecasting methods are evaluated according to RMSE using a `Rep-Holdout` estimation procedure. In each of a total of 10 iterations of the repeated holdout approach, 60% of the full data size is used for training, while the subsequent 10% of observations are used for testing. The validation data set $\mathcal{V}$, which is described in Algorithm 4 and used to build $\mathcal{C}_M$, is built using the blocked prequential procedure described in Section 4.2.2 of the previous chapter. Note that after building $\mathcal{C}_M$ with the out-of-bag observations, all predictive models are retrained using the complete training data set.

The parameters of OC, α and λ, are set to 30 and 100, respectively. This means that the elements of each subset in $\mathcal{C}_M$ show a better rank than the elements of other subsets of the same size for a contiguous window of 30 points. Each point denotes the average loss of each model in the last 50 observations.

We will present a sensitivity analysis that shows the impact on predictive performance when varying the value of these two parameters. The number (averaged across the 10 testing periods) of subsets comprising $\mathcal{C}_M$ is also described in Table A.1 ($|\mathcal{C}_M|$).

5.3.3 Set of Hypotheses M and Aggregation Approaches

Regarding the ensemble composition, M is comprised of the same 50 individual models used in the previous chapter. These are described in appendix A.3 in more detail, and in Figure 4.2 (page 93), we present a preliminary analysis that shows their relative performance across the 62 problems.

In terms of aggregation procedures we also used the ones studied in the previous chapter, namely `ADE`, `AEC`, `Blast`, `ERP`, `EWA`, `FixedShare`, `MLpol`, `OGD`, `Simple`, `Stacking`, and `WindowLoss`. We refer to Section 2.2.5 for a description of these methods. When the methods are employed using CA, their name is denoted using the prefix "CA" (e.g. `CA.Simple`). We also include the following baselines: `LossTrain`, in which the original hypotheses M are aggregated according to their RMSE in the training data; `CA.SimpleWorst`, a variant of `CA.Simple` in which $\mathcal{C}_M$ is built using the consistently worst performers, as opposed to using the best performers – this is accomplished by searching for the *bottom ranked hypotheses* (see line 9 in Algorithm 4); and `CA.SimpleRandom`, another variant of `CA.Simple` in which $\mathcal{C}_M$ is built randomly – the number of subsets and respective sizes are set according to OC, but these are filled with random models from the available pool. As a forecasting baseline, we include `ARIMA` and `Naive` (Hyndman et al., 2014), two state of the art approaches for time series forecasting.

5.3.4 Results

On Predictive Performance

Our first experiment is used to study the impact of CA on the state of the art forecast combination approaches. In other words, we want to understand if, given an aggregation method, it is better to apply it to the set of hypotheses M' built using CA via OC, or directly to the set of original hypotheses M (non-CA).

We start addressing this question (**RQ3.1**) by exploring the impact of the CA approach by measuring the pairwise difference in performance between each aggregation method with and without the application of CA. This analysis is presented in Figure 5.4, where the boxplots represent the log percentual difference in RMSE between CA and non-CA, for the different aggregation methods and across the 62 time series. Negative values denote better performance when the methods are applied using CA. These results show that for almost all aggregation methods, CA leads to a better predictive performance in the majority of problems. The exception are `ADE` and `Stacking`, although the former is relatively robust to CA. By robust we mean that the application of CA has a small effect on `ADE`, relative to other aggregation methods.

In order to analyse the significance of the previous study, we carried out a Bayesian analysis of the results (c.f. Section 4.3.3 for an overview of Bayesian analysis). We consider two methods to be significantly different if their percentual difference in performance (RMSE) is above 1%. Otherwise, the respective two methods are considered practically equivalent. In other words, the ROPE is the interval [-0.01, 0.01]. Figure 5.5 shows the application of the Bayes sign test which is an analysis of the significance of the impact of the application of CA for each method. The vertical bars, which are illustrated for each

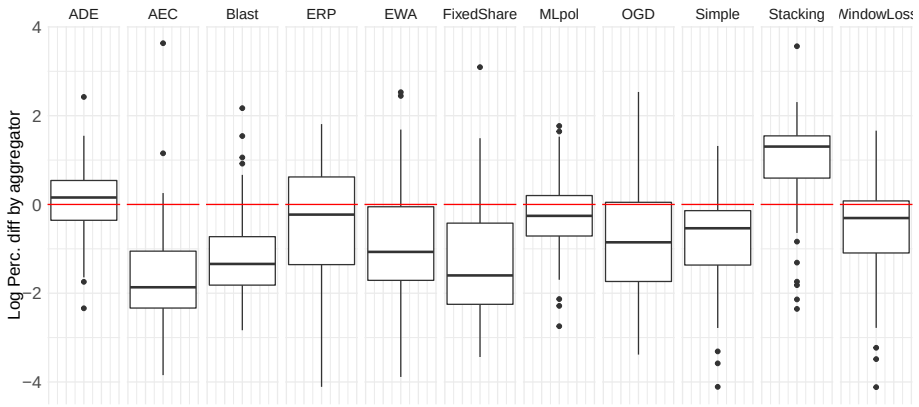


Figure 5.4: Log percentual difference in RMSE between CA and non-CA for each aggregation method. Negative values denote a decrease in RMSE (better performance) when using CA.

method, describe the probability of each outcome: CA winning, drawing (with-ing the ROPE), or losing. In general, the probability that CA wins is larger than the probability that CA loses (except for `Stacking` and `ADE`). For `ADE`, `MLpol`, `Simple`, and `WindowLoss` the most probable outcome is a draw. That is, applying CA leads to a performance difference below 1%.

Figure 5.5: Proportion of probability of CA winning/drawing/losing according to the Bayes sign test for each aggregation method

Finally, we compare all the approaches in Figure 5.6. This graphic illustrates the average rank, and respective standard deviation, of each aggregation method, with and without the application of CA, and the baselines. The main conclusion that we can draw from this graphic is that almost all forecast aggregation methods present a better average rank when applied using CA. The exceptions are `ADE` and `Stacking`, although the results on the first one are comparable according to the Bayesian analysis carried above. `ADE` presents the best average rank among all approaches. That means that the overall most appropriate approach (according to average rank) does not apply the proposed CA method. Nevertheless, the results point to a significant improvement in predictive performance for most of the aggregation methods when these are applied with CA.

Error Decomposition

In order to better understand how CA works, we carried out a decomposition of the error of the ensemble (Ueda and Nakano, 1996; Brown et al., 2005a). As

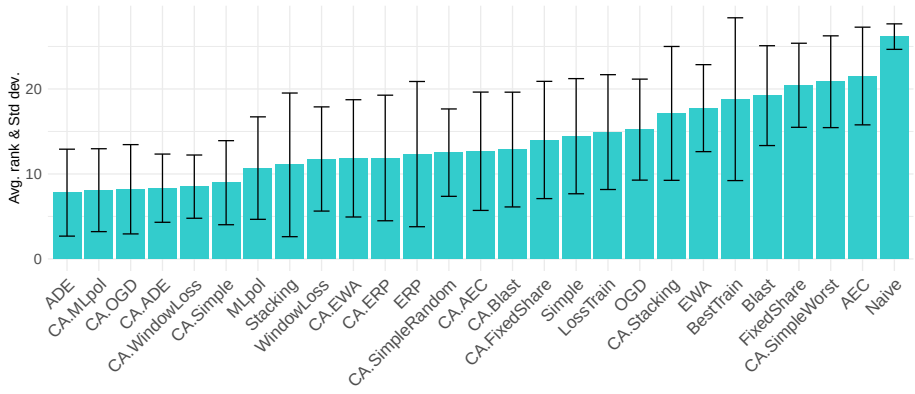


Figure 5.6: Average rank and respective standard deviation of each method across the 62 time series problems

we explained in Section 2.2.6, from a regression perspective, the squared error of a regression ensemble can be decomposed into bias, variance, and covariance terms as follows:

$$\text{Expected Error} = \overline{bias}^2 + \overline{var}\frac{1}{s} + (1 + \frac{1}{s})\overline{covar} + \sigma^2 \tag{5.3}$$

where $\overline{bias}^2$, $\overline{var}$, and $\overline{covar}$ represent the average bias, variance and covariance of the ensemble members, respectively. σ^2 is a constant irreducible term representing the variance of the noise. While a single estimator can be analysed using a bias-variance trade-off, the quadratic error generalisation of a regression ensemble depends on the bias, variance, and covariance of the individual models. The covariance term quantifies the diversity of the ensemble. Increasing values of average covariance denote less diversity. Brown et al. (2005a) provides an interesting and comprehensive read on this topic.

To analyse the impact of CA in this decomposition, we measured the percentual difference in each component relative to non-CA, across the 62 problems. Although the decomposition is valid for non-uniformly weighted ensembles (Brown et al., 2005a), we focus on the simple average aggregation, i.e. the difference between `CA.Simple` and `Simple`. This study is reported in Figure 5.7. Negative values represent a percentual decrease in the respective term when applying CA. On the left part of the figure, we present the decomposition following

the proposed approach. For comparison, we present the same decomposition using the baselines `CA.SimpleRandom` and `CA.SimpleWorst`.

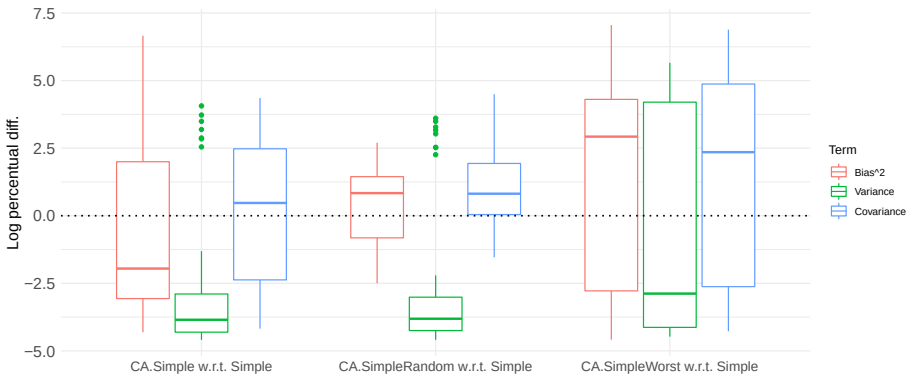


Figure 5.7: Log percentual difference in $\overline{bias}^2$, $\overline{var}$, $\overline{covar}$, and RMSE between `CA.Simple` and `Simple` (left), between `CA.SimpleRandom` and `Simple` (middle), and between `CA.SimpleWorst` and `Simple` (right)

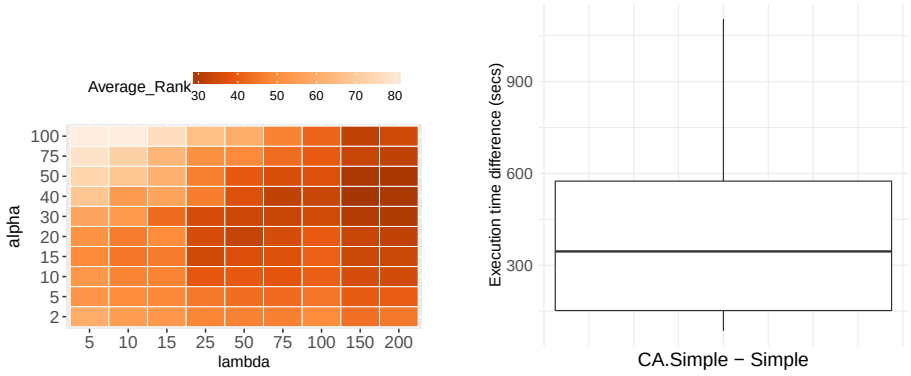
According to the figure, `CA.Simple` shows an average decrease in the bias term relative to `Simple`. This outcome is reasonable since CA focus on searching consistently top performing subsets of models, i.e. regions where some multiple individual models consistently show better rank than other equal-sized groups of models. There is also a considerable decrease in the average variance term. This is expected since most of the models in M' are combinations of models from M, which, when averaged, decrease variance. Oppositely, this leads to an average increase in the covariance term. This is also expected because each model from M can be part of multiple subsets that form the set of hypotheses M', leading to an increase in the redundancy of the ensemble.

`CA.SimpleRandom` and `CA.SimpleWorst` also lead to an average decrease in variance, and an average decrease in diversity (or an increase in the average covariance). This can be justified by the same reasoning presented for `CA.Simple`. The interesting part of this comparison is that the bias term increases for `CA.SimpleRandom` and `CA.SimpleWorst`. This effect is considerable for the latter, which leads to a worse performance relative to `Simple` (Figure 5.6). This outcome suggests, as we hypothesised, that `CA.Simple` improves the performance through an improvement in the average bias of the members of the

ensemble, even though it sacrifices some diversity to this effect (**RQ3.2**).

Parameter Sensitivity and Execution Time

To address question **RQ3.3** we analysed the sensitivity to parameters α and λ (c.f. Algorithm 4). This analysis is presented in Figure 5.8a. This graphic shows a heatmap with the average rank of each combination of (α, λ) across the 62 problems. The set of values tested for α was $\{2, 5, 10, 15, 20, 30, 40, 50, 75, 100\}$, and the respective set for λ was $\{5, 10, 15, 25, 50, 75, 100, 150, 200\}$. The values of both these sets were chosen arbitrarily. For simplicity, we focused on the `Simple` aggregation method. Overall, when α and λ are not set with too low values (from the considered grid), the average rank is relatively stable. In practice, the most appropriate set of values strongly depends on the data and the portfolio of models M.



(a) (b)

Figure 5.8: **a**) Heatmap illustrating the average rank CA for varying α and λ parameters with the `Simple` method. **b**) distribution of difference in execution time (seconds) when using `Simple` aggregation with and without CA.

Regarding question **RQ3.4**, we studied the execution time of CA. Again, in the interest of conciseness, we focused on the `Simple` aggregation method. To carry out this analysis, we compute the time spent to train and aggregate the ensemble. Then, we measure the time difference between CA and non-CA for each time series. The results are reported in Figure 5.8b, demonstrating that, on average, the `Simple` method using CA takes around 300 seconds more than

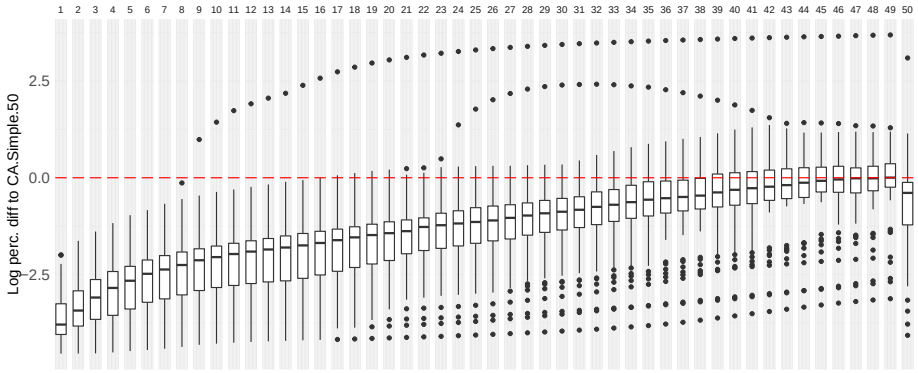


Figure 5.9: Log percentual difference in RMSE of `AvgRank` with a decreasing number of predictors (denoted as sufix) relative to `CA.Simple`. Negative values denote lower RMSE by `CA.Simple`.

when not using CA. This overhead is caused by the creation of $\mathcal{C}_M$ via OC. Notwithstanding, we note that the difference in execution time also depends on the aggregation approach, i.e. how it scales with the number of predictive models in the ensemble.

On Pruning

As we mentioned before, CA via OC builds the set of committees $\mathcal{C}_M$ focusing on consistently top performers. In this context, it might be argued that the improvements in performance are due to avoiding poor predictive models, and it could be reached by simply pruning them from the aggregation rule.

To test this hypothesis, we compared `CA.Simple` with an approach that quantifies the weight of each model according to the average rank in the last λ observations (denoted as `AvgRank`). We focus on the rank because it is the metric we use to build $\mathcal{C}_M$. Moreover, we apply `AvgRank` with a decreasing number of models, where the predictors are dynamically suspended (assigned weight 0) according to their average rank. For example, when using 10 out of the available 50 models and for a given time step, we do as follows. We compute the average rank of each of the 50 models in the last λ observations. Then, we drop the worst 40 models, weighing the remaining ones (with respect to `AvgRank`).

The results of this analysis are presented in Figure 5.9. The number in each

column, from 1 to 50, denotes the number of members in each ensemble. The boxplots represent the percentual log difference in RMSE of each variant of `AvgRank` relative to `CA.Simple`, across the 62 time series. Results show that `CA.Simple` presents a better performance, which is increasing as `AvgRank` is applied with a decreasing number of models. This outcome suggests that CA is not simply pruning poor predictive models (**RQ3.5**) from the aggregation rule.

5.4 Discussion

5.4.1 On the Trade-off Between Individual Error and Diversity

This chapter follows the evidence from previous work by Webb et al. (Webb, 2000; Webb and Zheng, 2004), which showed that combining different ensemble approaches leads to a better trade-off between the individual error of the ensemble members and diversity. The original motivation of Webb with `Multiboosting` (Webb, 2000) was to increase diversity while maintaining a reasonable individual error. Notwithstanding, Webb and Zheng (2004) later report different approaches where the inverse happens: cases where both diversity and individual error decrease, also leading to a lower ensemble error. The results from our experiments follow the second case. However interesting, these incite further investigation. Particularly, research into the mechanisms behind the success of this improved trade-off.

5.4.2 Results

The application of CA did not improve the predictive performance when using `ADE` or `Stacking`. One possible reason for this is that these approaches are the only ones following a metalearning methodology. In effect, it is possible that they do not scale well with a large number of sub-ensembles, which is often the case. We plan to look further into this in future work.

5.4.3 Challenges and Open Issues

We presented OC for retrieving the set of committees $\mathcal{C}_M$ with a small execution time overhead. Despite the results showing a systematic improvement in predictive performance for most of the aggregation methods, our intuition is that this approach can be further improved. For example, OC searches for subsets of M of all sizes (except the full size of M), which can lead to an unnecessary redundancy (c.f. Algorithm 4). We can potentially overcome this problem by introducing a *depth* parameter, which controls how large the subsets should be. Another potential solution to introduce a subset pruning procedure, which prunes some of the created subsets by predictive performance or by the diversity.

Contrary to other approaches, CA combines different sub-ensembles after the learning process, starting from a portfolio of pre-trained experts. This can be advantageous in terms of flexibility and concept drift adaptation: sub-ensembles can be updated, new models can be added to the portfolio M, or obsolete ones removed.

5.5 Conclusions

One of the core assumptions behind the dynamic aggregation of forecasting models is that these typically show a varying relative performance throughout a time series. Likewise, we hypothesised that different subsets of a portfolio of models might behave similarly. In this chapter, we presented a method dubbed constructive aggregation that explores this idea.

Constructive aggregation rearranges a set of independently created hypotheses M into different subsets $\mathcal{C}_M$. This is achieved using out-performance contiguity (OC), which searches for groups of models that outperform other groups of the same size contiguously during some time interval. These subsets are then aggregated into different combined hypotheses M'.

Applying state of the art aggregation approaches for forecasting to M' is demonstrated to provide better performance relative to applying them to M, on average. Moreover, this is accomplished with mild execution time overhead. The results also suggest that the improvement in performance is mainly due to a decrease in the individual error of the members of the ensemble. Although

diversity also decreases, CA leads to a better trade-off between these two factors.

Part III

Activity Monitoring

Chapter 6

Layered Learning for Activity Monitoring

6.1 Introduction

6.1.1 From Forecasting to Activity Monitoring

In the previous part of the thesis, we addressed the problem of time series forecasting. To recapitulate, the objective of this predictive task is to predict the next value of a time series given its information up until the current point. We measured the quality of different forecasting models according to the RMSE metric.

There are many domains where this setting is relevant; for example, solar radiation forecasting (Voyant et al., 2017; Torres et al., 2018). Utility companies use solar radiation forecasting systems to support their decision-making process. They use them to predict if the energy produced by a given solar technology can meet the daily electricity demand. Any predicted surplus of energy produced can be sold in an electricity market. Conversely, any predicted shortage leads to utility companies buying energy, or producing it in an alternative manner. In cases such as this one, the goal is to predict each point in the time series accurately. Any deviation from the observed true value is considered an error (as quantified by RMSE), which is expected to have some impact in the respective

domain.

Sometimes, however, we are interested in a different sort of prediction regarding the future of a given time series. Rather than accurately predicting the upcoming value of the time series, often the goal is to predict a specific event of interest. For example, an unusually low or high value that may be disruptive to the domain. As such, from a machine learning perspective, the output of a predictive model is binary, which represents whether the event of interest happens or not. This task is known as activity monitoring (Fawcett and Provost, 1999). In Section 2.4.1, we described many domains where this predictive task is relevant. In this part of the thesis, we address this problem, and propose a novel method to tackle it. We focus on a case study from the healthcare domain to validate our proposal.

6.1.2 Motivation for Activity Monitoring in Healthcare

Healthcare is one of the domains which has witnessed significant growth in the application of machine learning approaches (Bellazzi and Zupan, 2008). For instance, ICUs evolved considerably in recent years due to technological advances such as the widespread adoption of bio-sensors (Saeed et al., 2002). This lead to new opportunities for predictive modelling in clinical medicine. One of these opportunities is the early detection of critical health episodes (CHE), such as acute hypotensive episode (Ghosh et al., 2016) (AHE) or tachycardia episode (Forkan et al., 2017) (TE) prediction problems. CHEs such as these remain a significant mortality risk factors in ICUs (Ghosh et al., 2016), and their timely anticipation is fundamental for improving healthcare.

AHE or TE prediction can be regarded as a particular instance of early anomaly detection in time series data, or activity monitoring (Fawcett and Provost, 1999). As we mentioned in Section 2.4, the goal behind these problems is to issue accurate and timely alarms about interesting future events requiring action. In the case of CHE, a system should signal physicians about any impending future health crisis.

One of the most common ways to address activity monitoring problems is to view them as conditional probability estimation problems (Fawcett and Provost, 1999; Tsur et al., 2018). Standard supervised learning classification methods can

be used for that purpose. The idea is to approximate a function h that maps a set of input observations U to a binary variable v, which represents whether an anomaly occurs or not. In the context of CHE prediction, the predictor variables (U) summarise the recent physiological signals of a patient assigned to the ICU, while the target (v) represents whether or not there is an impending event in the near future.

6.1.3 Working Hypothesis and Approach

In many domains of application, the anomaly or event of interest is defined according to some rule derived from the data by professionals. In the case of healthcare, CHEs are often defined as events where the value of some physiological signal exceeds a pre-defined threshold. Similar approaches for formalising anomalies can be found in predictive maintenance (Ribeiro et al., 2016a), or wind power prediction (Ferreira et al., 2011). In these scenarios, we can also define pre-conditional events, which are arbitrary but computable relaxed versions of the event of interest. These pre-conditional events occur simultaneously with the anomaly one is trying to model, but are more frequent and, in principle, a good indication for these. To be more precise, a pre-conditional event (i) represents a less extreme version of the anomalies we are trying to detect (main events); and (ii) occur simultaneously with anomalies (i.e. there can not be an anomaly without a pre-conditional event). This concept is illustrated in Figure 6.1 as a Venn diagram for classes.

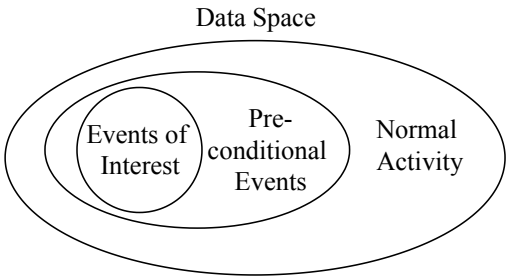


Figure 6.1: Essential scheme of the proposed layered learning approach. Instead of directly modelling events of interest according to normal activity, we first model pre-conditional events which occur simultaneously with the events of interest and are more frequent.

Our working hypothesis in this chapter is that modelling these pre-conditional events can be advantageous to capture the actual events of interest. To achieve this, we adopt a layered learning method (Stone and Veloso, 2000). Layered learning denotes a machine learning approach in which a predictive task is split into two or more layers (simpler predictive tasks) where the learning process within a layer affects the learning process of the next layer. In effect, layered learning is a hierarchical and multi-strategy learning approach.

We propose a layered learning method to address activity monitoring problems by splitting the predictive task into two layers (c.f. Figure 6.1). We first model pre-conditional events relative to normal activity. A subsequent model is applied to distinguish pre-conditional events from the actual anomalies. Effectively, the first layer affects the learning process of the second layer by decreasing the scope of its data space. Since the model in the second layer is created to distinguish the events of interest from pre-conditional events, it does not train on observations of what is designated as normal activity.

Our approach exploits the idea that rare events of interest occur simultaneously with pre-conditional events, which are considerably more frequent. Further, the same type of event of interest can be caused by distinct factors. For example, a particular type of CHE affecting two people may be caused by different diseases, which in turn may cause distinct dynamics in the time series of physiological signals. Therefore, initially modelling a relaxed version of the event of interest may lead to a simplification of the predictive task and a better performance when capturing the actual event.

We apply the proposed approach to tackle the problem of CHE prediction. Our results show that, when comparing to the typical direct classification approach (without layered learning), the layered learning model leads to a significantly better event recall (more CHEs are timely predicted), with a comparable number of false alarms. The proposed method also shows an overall better predictive performance relative to other state of the art methods.

In short, the contributions of this chapter are the following:

- a general layered learning approach to the early detection of events in time series data;
- the application of the proposed approach to AHE and TE prediction;

- a set of experiments validating the approach, including a comparison with state of the art approaches, a scalability analysis in terms of run-time, and a study of the impact of resampling strategies.

This chapter is structured as follows. In the next section, we start by formalising the problem of activity monitoring, both in general terms and using the case study of event prediction in ICUs. In Section 6.3, we present the proposed layered learning approach to activity monitoring. We overview layered learning as proposed by Stone and Veloso (2000), and formalise our proposed adaptation for the early detection of CHE. In Section 6.4, we carry out some experiments using the MIMIC II database (Saeed et al., 2002). In Section 6.5, we overview the related work, and finally conclude the chapter in Section 6.6.

All the work and results presented in this chapter are reproducible. The data is publicly available by Saeed et al. (2002). We also publish our code in an online repository[1].

6.2 Activity Monitoring

We formalise the problem of activity monitoring in time series in this section. We start by formalising the general problem and then the particular instances of AHE and TE prediction.

We follow Weiss and Hirsh (1998) to formalise the predictive task. As we described in Section 2.4.2, let $\mathcal{D} = \{D_1, \ldots, D_{|\mathcal{D}|}\}$ denote a set of time series. For example, $\mathcal{D}$ may represent a set of patients being monitored at the ICU of an hospital. Each $D_i \in \mathcal{D}$ denotes a time series $D_i = \{d_{i,1}, d_{i,2}, d_{i,n_i}\}$, where n_i represents the number of observations for entity D_i, and each $d \in D_i$ represents information regarding D_i in the respective time step (e.g. a set of physiological signals being captured from a patient in the ICU). D_i can also be represented as a set of subsequences $D_i = \{\delta_1, \delta_2, \ldots, \delta_i, \ldots, \delta_{n'-1}, \delta_{n'}\}$, where δ_i represents the *i-th* subsequence. A subsequence is a tuple $\delta_i = (t_i, U_i, v_i)$, where t_i denotes the time stamp that marks the beginning of the subsequence, $U_i \in \mathbb{U}$ represents the input (predictor) variables, which summarise the recent past dynamics of the time series; and $v_i \in \mathbb{V}$ denotes the target variable, which

[1] At `https://github.com/vcerqueira/layered_learning_time_series`

is a binary value ($v_i \in \{0,1\}, \forall\ i \in \{1,\ldots,t\}$) that represents whether or not there is an impending anomaly in the near future in the respective time series. How near in the future is typically a domain-dependent parameter. For each subsequence δ_i, we construct the feature-target pair (U_i,v_i) as follows.

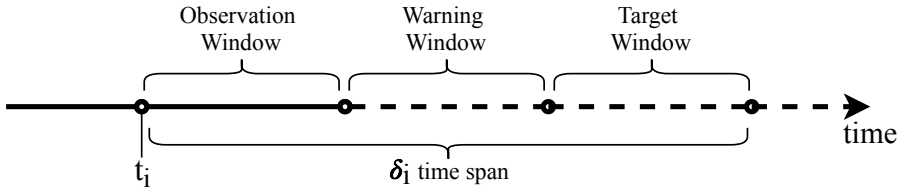


Figure 6.2: Splitting a subsequence δ_i into observation window, warning window, and target window. The features U_i are computed during the observation window, while the outcome v_i is determined in the target window.

As illustrated in Figure 6.2, δ_i has three associated windows: (i) the target window (TW), which is used to determine the value of v_i; (ii) an observation window (OW), which is the period available for computing the values of U_i; and (iii) a warning window (WW), which is the lead time necessary for a prediction to be useful. For instance, in clinical medicine, physicians need some time after an alarm is launched to decide the most appropriate treatment.

The sizes of these windows are domain-dependent. In principle, the problem will be easier as the observation window is closer to the target window, that is, a smaller warning window is required. Weiss and Hirsh (1998) provide evidence for this property when predicting equipment failure, and Lee and Mark (2010a) obtain similar results regarding AHE prediction. Moreover, according to Weiss and Hirsh (1998), larger observation windows generally also lead to better results, but too large observation windows lead to meaningless predictions.

6.2.1 Event Prediction in ICUs

In this chapter, we focus on a particular instance of activity monitoring problems: CHE prediction in ICUs, namely AHE and TE. Ghosh et al. (2016) state that prolonged hypotension leads to critical health damage, from cellular dysfunction to severe injuries in multiple organs. In turn, sustained tachycardia significantly increases the risk of stroke or cardiac arrest. Because CHEs are a

relevant cause of mortality in ICUs, it is fundamental to anticipate them early in time so that physicians can prevent them or mitigate their consequences.

Patients assigned to the ICU are typically monitored constantly, with biosensors capturing several physiological signals, such as heart rate, or mean arterial blood pressure. This is illustrated in Figure 6.3, where the data of a patient is depicted. A subsequence for CHE prediction is given as an example in the shaded area of the graphic. This area is split into three windows (observation, warning, target), as explained before.

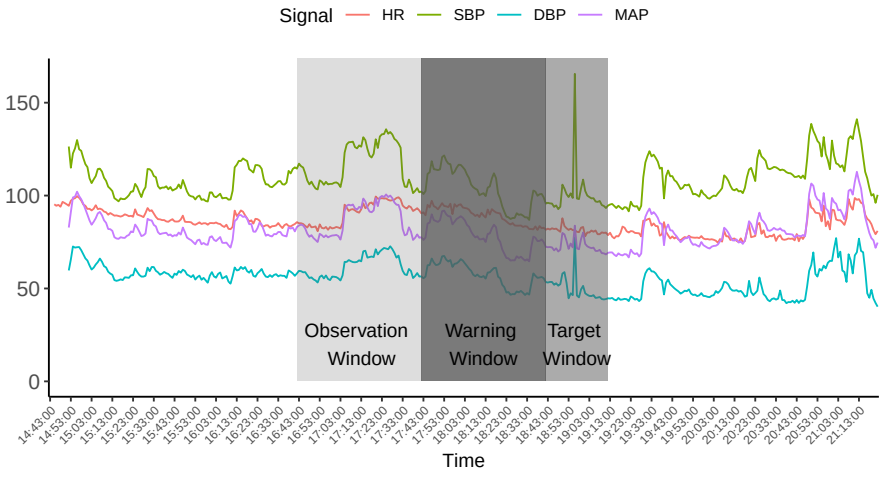


Figure 6.3: The physiological signal of patients are monitored over time. Each subsequence, denoted by the shaded areas, is split in an observation window, a warning window, and a target window.

Acute Hypotensive Episodes

Hypotension episodes are defined as a prolonged drop in the blood pressure. More formally, AHE is an event defined as "a 30-minute window having at least 90% of its mean arterial blood pressure (MAP) values below 60 mmHg [millimetres of mercury]" (Tsur et al., 2018; Lee and Mark, 2010a). In this context, the target variable value is computed as follows:

$$v_i = \begin{cases} 1, & \text{if an AHE happens in } TW_i, \\ 0, & \text{otherwise.} \end{cases}$$

In other words, we consider that the i-th subsequence represents an anomaly if its target window represents an AHE (c.f. Figure 6.3). Since AHEs are rare, the target vector v is dominated by the negative class (i.e. $v = 0$), where a patient shows a normotensive status. For the target window of 30 minutes, we consider an observation window and a warning window of 60 minutes each. These values are typically used in the literature of AHE prediction models (Ghosh et al., 2016).

Tachycardia Episodes

Tachycardia denotes a high heart rate (HR). Generally, an HR over 100 beats per minute (bpm) under a resting state is considered tachycardia. In order to consider a more robust definition for the purpose of discovering tachycardia episodes, we follow a similar intuition to AHEs. We define TE as "a 30-minute window having at least 90% of its HR values above 100 bpm". The respective target variable is computed as follows:

$$v_i = \begin{cases} 1, & \text{if an TE happens in } TW_i, \\ 0, & \text{otherwise.} \end{cases}$$

TEs are defined similarly to AHEs. Moreover, TEs also denote rare events since ICU patients usually show an HR below 100 bpm. We consider identical window sizes (OW, WW, TW) for both problems.

6.2.2 Discriminating Approaches to Activity Monitoring

Naturally, one of the most common approaches to solving the problem defined previously is to view it as a conditional probability estimation problem and use standard supervised learning classification methods (Fawcett and Provost, 1999; Tsur et al., 2018). The idea is to build a model $h : \mathbb{U} \rightarrow \mathbb{V}$, where $U \in \mathbb{U}$ and $v \in \mathbb{V}$. This model can be used to predict the target values associated with unseen feature attributes. In other words, h is a discriminating model that explicitly distinguishes normal activity from anomalous activity.

Notwithstanding the widespread of this approach, activity monitoring problems often comprise complex target variables whose definition is derived from

the data. In such cases, it is possible to decompose the target variable into partial and less complex concepts, which may be easier to model. In this context, our working hypothesis is that we can leverage a layered learning approach to model these partial concepts and obtain an overall better model for capturing the actual events of interest.

6.3 Layered Learning for Activity Monitoring

6.3.1 Layered Learning

Layered learning is designed for predictive tasks whose mapping from inputs to outputs is complex. For example, Stone and Veloso (2000) apply this approach to robotic soccer. Particularly, one of the problems they face is the retrieval and passing of a ball. The authors split this task into three layers: (i) ball interception; (ii) pass evaluation; and (iii) pass selection. This process leads to a more effective decision-making system with a considerably higher success rate than a direct approach.

In essence, layered learning consists of breaking a predictive task into several layers. The approach assumes that the problem addressed in each layer is simpler than the original one. As Stone and Veloso (2000) explain, "the key defining characteristic of layered learning is that each layer directly affects the learning of the next". This effect can occur in several ways. For example, by affecting the set of training examples, or by providing features used for learning the original concept.

6.3.2 Pre-conditional Events

The definition of an anomalous event in time series data is in many cases determined according to some rule derived from the data. As an example from the healthcare domain presented in the previous section, an AHE is defined as a percentage of numeric values which are below some threshold within a time interval (c.f. Section 6.2.1). TEs are defined similarly. This type of approach for defining anomalous events is also common in other domains. For example, in predictive maintenance (Ribeiro et al., 2016a), where numerical information

from sensor readings is transformed into a class label which denotes whether or not an observation is anomalous. Or wind ramp detection, where a ramp event is a rare occurrence that denotes a large percentual change in wind power output in a short time interval (Ferreira et al., 2011).

Since these anomalous events are defined according to the value of an underlying variable, we can also define pre-conditional events: relaxed versions of the actual events of interest, but which are more frequent. A more precise definition can be given as follows. A pre-conditional event is an arbitrary but computable event that is expected to occur with the main event taking place simultaneously. If the main event occurs, the pre-conditional event must occur, but the latter can occur without the main event.

An example can be provided using the case study of AHE prediction. In Section 6.2.1, we defined the main event (AHE) as "a 30-minute window having at least 90% of its mean arterial blood pressure (MAP) values below 60 mmHg". A possible pre-conditional event for this scenario could be "a 30-minute window having at least **45**% of its mean arterial blood pressure (MAP) values below 60 mmHg". Another possibility is "a 30-minute window having at least 90% of its mean arterial blood pressure (MAP) values below **70** mmHg".

In summary, pre-conditional events should have the following two characteristics:

- pre-conditional events should have a higher relative frequency than the main events;
- pre-conditional events always happen when the main events happen. The inverse is not a necessary condition.

6.3.3 Methodology

We can leverage the idea of pre-conditional events and use a layered learning strategy to tackle activity monitoring problems in time series data. Our idea is to decompose the main predictive task into two layers, each denoting a predictive sub-task. Pre-conditional events are modelled in the first layer, while the main events are modelled in the subsequent one.

The intuition behind this idea is given in Figure 6.4. The figure presents

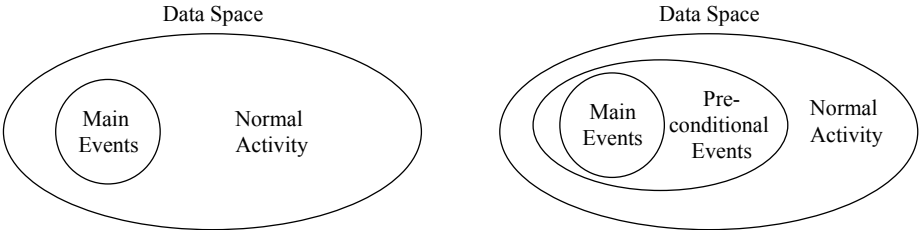


Figure 6.4: Venn diagram for the classes in an activity monitoring problem. The main event represents a small part of the data space; pre-conditional events are more frequent and include the occurrence of the main events.

two Venn diagrams for classes. Focusing on the left-hand side, the anomalies or main events (e.g. AHE) represent a small part of the data space. This is one of the issues that makes them difficult to model. In the typical classification approach, main events are directly modelled with respect to the remaining data space (deemed normal activity).

Our idea is represented on the right-hand side. An initial pre-conditional concept is considered, which is more common than the main target concept, while also including it. The higher relative frequency of the pre-conditional events with respect to the main events helps to mitigate the problem of having an imbalanced distribution, which is the case in activity monitoring tasks. This phenomenon can compromise the performance of learning algorithms (He and Ma, 2013). In effect, we first model the pre-conditional events with respect to normal activity. These pre-conditional events are, in principle, easier to learn relative to the main concept as they are more frequent and thus the classification algorithms will not suffer so much from an imbalanced distribution. Afterwards, the main target events are modelled with respect to the pre-conditional events, which is also a less imbalanced distribution than the original on the left diagram.

In the remainder of this section, we will further formalise our approach using a generic notion of pre-conditional and main events. In the next section, we will apply this formalisation to CHE prediction problems.

Pre-conditional Events Sub-task

Let $\mathcal{S}$ denote a pre-conditional event. The target variable when modelling these events is defined as:

$$v_i^{\mathcal{S}} = \begin{cases} 1 & if\ \mathcal{S} \text{ happens}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

For this task, a subsequence $\delta_i^{\mathcal{S}}$ is a tuple $\delta_i^{\mathcal{S}} = (t_i, U_i, v_i^{\mathcal{S}})$. The difference to the original subsequence δ_i is the target variable, which replaces v with $v^{\mathcal{S}}$. Finally, the goal of this first predictive task is to build a function $h^{\mathcal{S}}$ that maps the input predictors U to the output $v^{\mathcal{S}}$.

Main Events Sub-task

Provided that we solve the pre-conditional events sub-task, in order to predict impending main events the remaining problem is to find out whether or not, when $\mathcal{S}$ happens, the main event also happens.

Let $\mathcal{F}$ be defined as the occurrence: "given $\mathcal{S}$, there is an impending main event in the target window of the current subsequence". Effectively, the target variable for this task is defined as follows:

$$\text{Given } v^{\mathcal{S}} = 1,\ v_i^{\mathcal{F}} = \begin{cases} 1 & \text{if a main event happens in } TW_i, \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

The target variable for this sub-task ($v^{\mathcal{F}}$) is formalised in equation 6.2. Given that the class of $v^{\mathcal{S}}$ is positive (which means that there is an impending pre-conditional event), the class of $v^{\mathcal{F}}$ is positive if a main event also happens in that same target window, or negative otherwise.

The goal of this second predictive task is to build a function $h^{\mathcal{F}}$, which maps U to $v^{\mathcal{F}}$. Formally, a subsequence $\delta_i^{\mathcal{F}}$ is represented by $\delta_i^{\mathcal{F}} = (t_i, U_i, v_i^{\mathcal{F}})$. In this scenario, however, the set of available subsequences D_i is considerably less than in the pre-conditional sub-task because only the sequences for which $v^{\mathcal{S}}$ equals 1 are accounted for. This means that $h^{\mathcal{F}}$ only learns with subsequences that at least lead to a pre-conditional event. Effectively, this aspect represents how the learning in the pre-conditional events sub-task affects the learning on the main events sub-task, i.e., by influencing the data examples used for training. In the main events sub-task, a predictive model is concerned with the distinction between pre-conditional events and main events. Essentially, it assumes that the distinction between normal activity and pre-conditional events is carried out by

the previous layer. Given this independence, the training of the two layers can occur in parallel.

Forecasting Impending Anomalies

To make predictions about impending events of interest we combine the models $h^{\mathcal{S}}$ with $h^{\mathcal{F}}$ with a function $\kappa : \mathbb{U} \times \mathbb{U} \rightarrow \mathbb{V}$.

$$\kappa(U_i) = h^{\mathcal{S}}(U_i) \cdot h^{\mathcal{F}}(U_i) \tag{6.3}$$

Essentially, according to equation 6.3 the function κ predicts that there is an impending main event in a given subsequence δ_i according to the multiplication of the outcome predicted by both $h^{\mathcal{S}}$ and $h^{\mathcal{F}}$.

Ideally, there are three possible outcomes:

- Both event $\mathcal{S}$ and event $\mathcal{F}$ happen, which means there is an impending main event: both $h^{\mathcal{S}}$ and $h^{\mathcal{F}}$ should return 1 so that $h^{\mathcal{S}} \cdot h^{\mathcal{F}} = 1$;
- Event $\mathcal{S}$ happens, but event $\mathcal{F}$ does not happen: $h^{\mathcal{S}} = 1$, but $h^{\mathcal{F}} = 0$, so $h^{\mathcal{S}} \cdot h^{\mathcal{F}} = 0$;
- Event $\mathcal{S}$ does not happen, and consequently, event $\mathcal{F}$ also does not happen: $h^{\mathcal{S}} \cdot h^{\mathcal{F}} = 0$.

6.3.4 Application of Layered Learning to CHE Prediction

We formalise the application of our idea to CHE problems, namely AHE and TE prediction.

AHE Prediction

As mentioned before (c.f. Section 6.2.1), an AHE is defined as a 30-min period where 90% of the blood pressure values are below 60 mmHg. We propose to relax this threshold and define the pre-conditional event $\mathcal{S}$ as:

$\mathcal{S}^{\text{AHE}}$: "a 30-minute window having at least **45**% of its mean arterial blood pressure values below 60 mmHg".

The event $\mathcal{S}$ is consistent with the two above-mentioned characteristics: the frequency of $\mathcal{S}$ across the database is considerably higher than an AHE – note that the blood pressure level can drop below 60 mmHg for some time period without being considered as a hypotensive episode. Consequently, the occurrence $\mathcal{S}$ is simultaneous to the occurrence of an AHE (if 90% of the values are below 60 mmHg, so are 45%).

TE Prediction

We apply the same reasoning to the TE prediction task. In Section 6.2.1, we defined a TE as "a 30-minute window having at least 90% of its HR values above 100 bpm". In order to define $\mathcal{S}^{\text{TE}}$ we again relax the percentage threshold as follows:

$\mathcal{S}^{\text{TE}}$: "a 30-minute window having at least 45% of its HR values above 100 bpm".

Similarly to $\mathcal{S}^{\text{AHE}}$, the events $\mathcal{S}^{\text{TE}}$ also follow the desired characteristics of pre-conditional events. In both situations (AHE and TE), the value of 45% was chosen arbitrarily. Essentially, we attempted to make the pre-conditional events much more frequent relative to the main events. Nevertheless, this parameter can be optimised.

6.4 Empirical Experiments

6.4.1 Case Study: MIMIC II

In the experiments, we used the database Multi-parameter Intelligent Monitoring for Intensive Care (MIMIC) II (Saeed et al., 2002), which is a benchmark for several predictive tasks in healthcare, including CHE prediction.

As inclusion criteria of patients and general database pre-processing steps, we follow Lee and Mark (2010a) closely. For example, the sampling frequency of the physiological data of each patient in the database is one minute. Moreover, the following physiological signals are collected: heart rate (HR), systolic blood pressure (SBP), diastolic blood pressure (DBP), and mean arterial blood pressure (MAP).

As described in Section 6.2.1, the target window size is 30 minutes. For each target window, there is a 60-minute observation window and a 60-minute warning window. For a comprehensive read regarding the data compilation, we refer to the work by Lee and Mark (2010a). Considering this setup, the number of patients is 1,072, leading to a data size of 1,975,936 subsequences. 71,035 of those subsequences represent an AHE (about 3.5%). In turn, 13.6% of the subsequences represent a TE.

We consider HR, SBP, DBP, and MAP values between 10 and 200 (bpm for HR, mmHg for the remaining ones). Values outside of this range are eliminated as "unlikely outliers" (Lee and Mark, 2010a). From the available signals (HR, SBP, DBP, MAP), we compute the values of cardiac output (CO) and pulse pressure (PP).

Regarding feature engineering, we follow previous work in the literature (Lee and Mark, 2010b; Tsur et al., 2018). Using the observation window of each subsequence and of each physiological signal, the feature engineering process was carried out using statistical, cross-correlation, and wavelet functions. The statistical metrics include skewness, kurtosis, slope, median, minimum, maximum, variance, mean, standard deviation, and inter-quartile range. For each observation window, we also compute the cross-correlation of each pair of signals at lag 0. We also use the Daubechies wavelet transform (Percival and Walden, 2006) to perform a 5-level discrete wavelet decomposition and capture the relative energies in different spectral bands. Intuitively, medication data can play an important role. However, Lee and Mark (2010b) reported no predictive advantage in using such information. In effect, we do not include this information in the predictive models.

6.4.2 Experimental Design

The experiments were designed to answer the following research questions:

RQ4.1: how does the proposed layered learning architecture performs relative to state of the art approaches for activity monitoring?

RQ4.2: what is the predictive performance of each layer in the proposed layered learning architecture for CHE prediction?

RQ4.3: how does the layered learning approach scale in terms of run-time compared to other approaches?

RQ4.4: what is the impact of pre-processing the training data using a resampling method for balancing the distribution of classes?

To estimate the predictive performance of each method, we used a 5×10-fold cross-validation, in which folds are split by patients. To be more precise, in each iteration of the cross-validation procedure, one fold of patients is used for validation, another fold of different patients is used for testing, and the remaining patients are used for training the predictive model. Therefore, all subsequences of a given patient are only used for either training, validation, or testing. The set of time series (patients) only comprises a temporal dependency within each patient, and we assume the data across patients to be independent. That is, the probability that a patient suffers a health crisis is independent of another patient also suffering a health crisis. In this context, the application of cross-validation in this setting is valid. Finally, the subsequences of the patients chosen for training are concatenated together to fit the predictive model. This model is tuned using the subsequences of patients chosen for validation and evaluated using the subsequences of patients chosen for testing.

Subsequences used for Training

Given the sizes of OW, WW, and TW (60, 60, and 30 minutes, respectively), the duration of a subsequence is 150 minutes. Since the data is collected every minute, there is considerable overlap between consecutive subsequences. During run-time, a given model is used to predict whether there is an impending CHE in each subsequence. This approach emulates a realistic scenario, where a prediction is produced as more data is available regarding the current health state of a given patient.

Given the redundancy among consecutive subsequences, it is common to sample the subsequences used for training a predictive model (Tsur et al., 2018). For example, Cao et al. (2008) compile subsequences for training according to whether a patient has experienced a CHE. For every patient that did, the latest 120 minutes of data before the onset of the respective CHE are used

to create a training subsequence. If a patient did not experience a CHE, one or more subsequences are sampled at random. Lee and Mark (2010b) collect multiple subsequences in a sliding window fashion, irrespective of whether a patient experienced a CHE. A sliding window with no overlap and of size TW is used to traverse each patient. That is, if a subsequence δ_i starts at time t_i then the next subsequence δ_{t+1} starts at time t_{i+30}. The authors show that this approach leads to better results relative to the approach taken by Cao et al. (2008).

In both cases described above, the authors note that these approaches lead to an imbalanced data set. They recommend under-sampling the majority class to overcome this issue. In this work, we follow the approach by Lee and Mark (2010b). As recommended, we also apply a class balance procedure, which is described below in Section 6.4.3 and analysed in Section 6.4.8.

The Value of a Prediction

The timely prediction of impending CHEs enables a more efficient allocation of ICU resources and a more prompt application of the appropriate treatment. In this context, for a prediction to be useful, it must occur before the onset of the respective CHE. We assume that, after the event starts, any prediction becomes obsolete. Further, predicting too early also leads to meaningless predictions due to the continuity of time. We follow the approach taken in the 10^{th} PhysioNet challenge (Moody and Lehman, 2009) regarding AHE prediction. A CHE (we also extend the definition to TEs) is considered to be correctly anticipated if it starts within 60 minutes after an alarm is launched. We consider the value of an alarm to be binary, where its benefit is 1 if it is issued correctly, and 0 otherwise.

Evaluation Metrics

The goal behind activity monitoring problems is not to classify each subsequence as positive or negative (Fawcett and Provost, 1999). Instead, the main goal is to detect, in a timely manner, when there is an impending anomalous event. In this context, we follow Weiss and Hirsh (1998) regarding the evaluation metrics. Specifically, two measures are computed: Event Recall (ER), and Reduced

Precision (RP). These two metrics follow the same intuition of the widely used Recall and Precision metrics but are tailored for time-dependent data.

Let T denote the total number of events of interest in a test data set, and let $\hat{T}_m$ represent the total number of those events correctly predicted by a model m. The ER for model m is given by the following equation:

$$\mathrm{ER}_m = \frac{\hat{T}_m}{T} \tag{6.4}$$

ER differs from the classical recall metric because a single correct prediction within an observation window leading to an event is enough to consider that event correctly anticipated. As Fawcett and Provost (1999) put it, "alarming earlier may be more beneficial, but after the first alarm, a second alarm on the same sequence may add no value".

The classical precision metric measures the ratio of positive predictions that are correct. Similarly to recall, in a time-dependent domain, the classical precision may be misleading because multiple predictions on the same event are counted multiple times. This idea is intuited in Figure 6.5. This graphic shows a sequence in which predictions are being produced over time. Starting from time t_i, four false alarms are triggered. Performance evaluation should take the first wrong prediction into account as a false positive. However, the subsequent false alarms (as shown in Figure 6.5) are not meaningful since they add no information – assuming some action is taken after the first alarm.

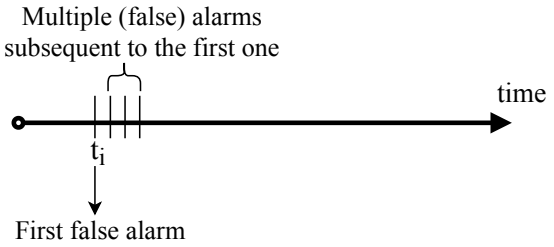


Figure 6.5: A sequence of consecutive false alarms. The first alarm is useful, but the subsequent ones may add no information.

RP overcomes this problem by considering a prediction to be *active* for some time period. Specifically, in this work, we consider a time interval with the same size as the observation window (60 minutes). Notwithstanding, this is usually a

domain-dependent parameter. Effectively, the RP metric replaces the number of false positives with the number of discounted false positives – the number of non-overlapping observation periods associated with a false prediction. This idea is illustrated in Figure 6.6, where each vertical bar in the time line denotes an issued false alarm. There are a total of 6 false positives, but, if taking into account the time interval a prediction is active, there are only two discounted false positives (DFP). Finally, RP also considers the number of target events correctly identified ($\hat{T}_m$), instead of the number of correct predictions (true positives).

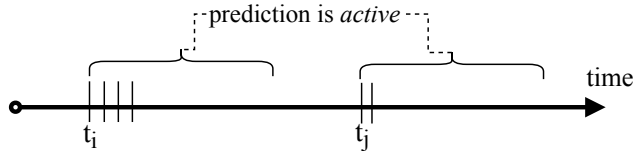


Figure 6.6: False alarms (denoted as vertical bars) over a time interval. There are 6 false positives, but only two discounted false positives.

In effect, RP for model m is given by the following equation:

$$\mathrm{RP}_m = \frac{\hat{T}_m}{\hat{T}_m + \mathrm{DFP}_m} \tag{6.5}$$

ER and RP summarise the predictive performance. To further explore the behaviour of a given method, we also compute the average number of false alarms (FA); and the average anticipation time (AT) (how long in advance an event is predicted).

Learning Algorithms

We tested different predictive models in the experiments, namely a random forest (Wright, 2015), a support vector machine (Karatzoglou et al., 2004), a deep feed-forward neural network (Abadi et al., 2016), and an extreme gradient boosting (`xgboost`) model (Chen et al., 2015). We only show the results of the latter in these experiments, since it provides better performance than the remaining methods for both AHE prediction and TE prediction. This corroborates the experiments by Tsur et al. (2018), stating that `xgboost` gives the best predictive performance for AHE prediction tasks.

The output of the classifiers used in the experiments is a probability. The decision threshold is optimised following previous work in the literature of AHE prediction (Lee and Mark, 2010a; Tsur et al., 2018), which recommends selecting the threshold that maximises the average of *classical* recall and specificity (true negative rate).

6.4.3 State of the Art Methods

We compare the proposed layered learning approach (henceforth denoted as `LL`) with the following four methods.

Standard classification

We compare `LL` with a standard classification method (`CL`) that does not apply a layered learning approach and directly models the events of interest with respect to normal activity (c.f. Figure 6.4). One of the working hypothesis for the application of the proposed layered learning approach is that it helps to mitigate the class imbalance problem. To further cope with this problem, we process the data used for training `CL` and `LL` using a resampling method (Branco et al., 2016b,a). In the case of `LL`, this process was applied to both layers after performing the task decomposition. For the AHE prediction problem, `CL` was applied with random over-sampling, while `LL` was applied using random under-sampling. For TE prediction, both approaches were applied using SMOTE (Chawla et al., 2002). These choices are analysed in Section 6.4.8.

Isolation Forest

An Isolation Forest (`IF`) (Liu et al., 2012) is a state of the art unsupervised model-based approach to anomaly detection. A typical method of this sort typically discards the anomalies within the training data and creates a model for normal activity. Observations that significantly deviate from the typical behaviour are considered outliers. We referred to these approaches as profiling methods (Section 2.4.4). Instead of separating the normal activity, `IF` explicitly models the anomalies in an unsupervised manner using an ensemble of tree-structured models. The core idea behind a `IF` is that the paths resulting from

partitioning the data are shorter for anomalous observations because the regions comprising these anomalies are separated quickly.

Regression Approach

The type of anomalies addressed in this chapter is defined according to the observed values in numeric variables. For example, an AHE is defined according to the distribution of the variable MAP in the target window of an subsequence. In effect, one common alternative modelling approach is to perform a regression analysis with the respective numeric variable as the target variable. Alarms regarding impending anomalies are then triggered using a deterministic function that maps the forecasted values into a decision.

We include a regression-based alternative both for AHE prediction and TE prediction. We apply a multi-step forecasting model to predict the future values of MAP (for AHEs) and HR (for TEs) for the next TW. Regarding the former, and following up on the definition of an AHE (Section 6.2.1), an alarm for an AHE is triggered if 90% of the forecasted values for the MAP variable are below 60 mmHg (Rocha et al., 2011). Likewise, an alarm for a TE is triggered if 90% of the forecasted values for the HR variable are above 100 bpm.

Similarly to Rocha et al. (2011), the multi-step forecasting model follows a *direct* approach (Taieb et al., 2012). This means that a forecasting model is created for each point in the horizon. The horizon in our setup is represented by the target window, which is a 30-minute window of observations with a granularity by the minute. In effect, the regression-based approach is comprised of 30 forecasting models. We denote the regression-based model as `RG`. To train each forecasting model, we also use a `xgboost` learning algorithm, which is tuned for numeric target variables.

Ad-hoc Methods

While there is an increasing number of machine learning applications in healthcare, many of the currently deployed systems still rely on simple *ad-hoc* rules to support the decision-making process of professionals. Taking AHE prediction as an example, a simple rule is to trigger an alarm if the MAP of a patient drops below 60 mmHg in a given time step. A similar approach can be used for

TE prediction, where an alarm is launched if the HR variable exceeds 100 bpm. However simple, these ad-hoc rules often work well in practice. We use these rules as baselines in our experimental design and denote them as `AH`.

6.4.4 Results on AHE Prediction

Table 6.1 presents the average results, and respective standard deviation, for each method across the 50 folds ($5 \times 10-$fold CV) for the AHE prediction problem. Overall, `LL` presents the highest ER, capturing 83% of the AHE. These values are significantly better relative to the remaining methods, including `CL`, which is the typical approach to solve these predictive tasks. Conversely, `LL` shows a comparable RP with `CL`. The RP shown by `LL` is better than `IF`'s, but considerably worse than the ones shown by `AH` and `RG`.

Table 6.1: Average of results for the AHE prediction problem across the 50 folds. Boldface represents the best result in the respective metric

Method	**ER**	**RP**	**Avg. AT**	**Avg. FA**
`AH`	0.625±0.057	0.129±0.022	31.9±3.3	4.9±2.2
`CL`	0.807±0.072	0.089±0.016	44.9±3.3	20.7±6.0
`LL`	**0.830**±0.054	0.090±0.015	**46.9**±3.3	22.9±6.9
`RG`	0.250±0.067	**0.205**±0.044	11.1±3.2	**3.3**±9.1
`IF`	0.700±0.182	0.035±0.009	37.9±11.8	26.6±15.2

In Figure 6.7, we analyse the significance of the results according to the Bayesian correlated t-test (Benavoli et al., 2017). In this test, we consider the ROPE to be the interval $[-0.01, 0.01]$. The results from Table 6.1 are corroborated by the Bayesian analysis. `LL` shows a significantly better ER and a comparable RP with respect to `CL`. Expectedly, there is a trade-off between ER and RP: greater ER leads to lower RP, and vice-versa. Notwithstanding, relative to `CL`, `LL` is able to significantly improve ER while keeping a comparable (i.e., within the region of practical equivalence) RP. Maximising ER in this particular domain of application is important because the events of interest are disruptive. While `LL` shows a significantly worse RP relative to `AH` and `RG`, it compensates with a better ER.

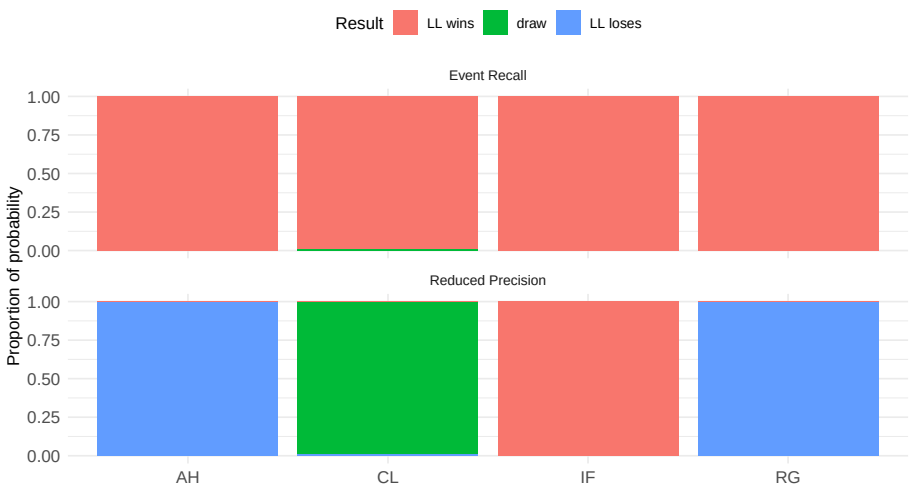


Figure 6.7: Comparing `CL` with `LL` with a Bayesian correlated t-test for ER and RP metrics (AHE prediction)

In Figure 6.8, we show the distribution of the false alarms issued per hour and per patient (upper tile), and the distribution of anticipation time (in minutes) per patient (lower tiles), for each method under comparison. For instance, a value of 10 means that, on average, 10 false alarms are issued for a given patient (upper tile). On the other hand, a value of 30 in the distribution on the right side of the figure means that an AHE was predicted with 30 minutes in advance (before the episode started). The values of these distributions are somehow related to the previous results. `AH` and `RG` show the lowest average false alarms per hour, which correlates with the results of RP. Conversely, `LL` seems to have the most interesting distribution relative to anticipation time (close to the value of 60).

6.4.5 Results on TE Prediction

Similar to Table 6.1, Table 6.2 presents the average of results, and respective standard deviation, across the 50 folds for the TE prediction problem. Overall, similar conclusions can be drawn from the performance metrics. The proposed method `LL` captures 93.8% of the TE, which is a greater value relative to the remaining methods. Although slightly better, the RP value is comparable to `CL`

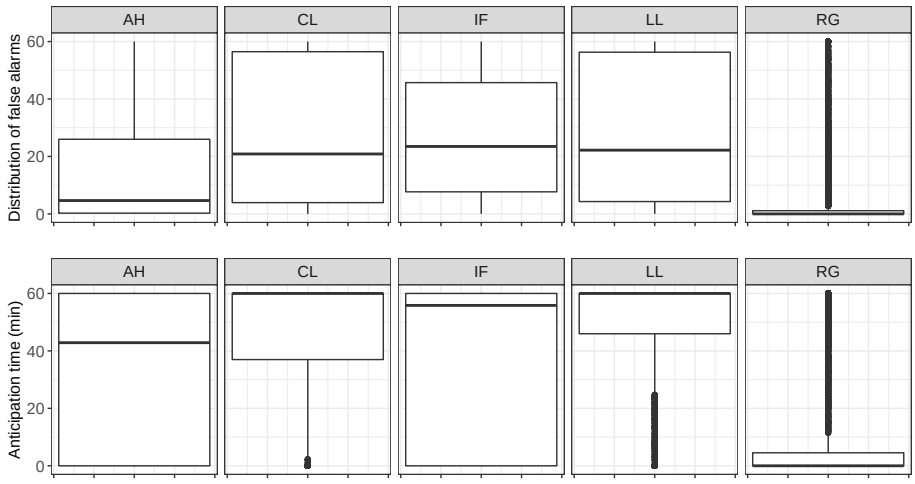


Figure 6.8: Distribution of the false alarms issued per hour and per patient (top), and the distribution of anticipation time (in minutes) per patient (low) for the AHE prediction problem

(but worse than that of `AH` and `RG`).

Table 6.2: Average of results for the TE prediction problem across the 50 folds

Method	**ER**	**RP**	**Avg. AT**	**Avg. FA**
`AH`	0.749±0.051	**0.204**±0.022	37.9±3.1	6.9±2.7
`CL`	0.919±0.024	0.130±0.021	51.6±2.6	31.9±9.7
`LL`	**0.938**±0.027	0.136±0.018	**53.0**±2.2	35.9±8.8
`IF`	0.756±0.317	0.051±0.013	42.5±19.3	35.9±21.7
`RG`	0.646±0.049	0.195±0.025	31.1±2.8	**2.3**±0.9

This conclusion can also be drawn from the results of the Bayesian analysis shown in Figure 6.9. `LL` shows a significantly better ER relative to the `CL` while having a comparable RP. `LL` also shows a better ER relative to the remaining approaches but loses to `AH` and `RG` when analysis RP.

In Figure 6.10, we show the distribution of the false alarms issued per hour and per patient (left), and the distribution of anticipation time (in minutes) per patient (right), for each method under comparison. This analysis also shows a similar result and the same study for the AHE prediction problem shown in

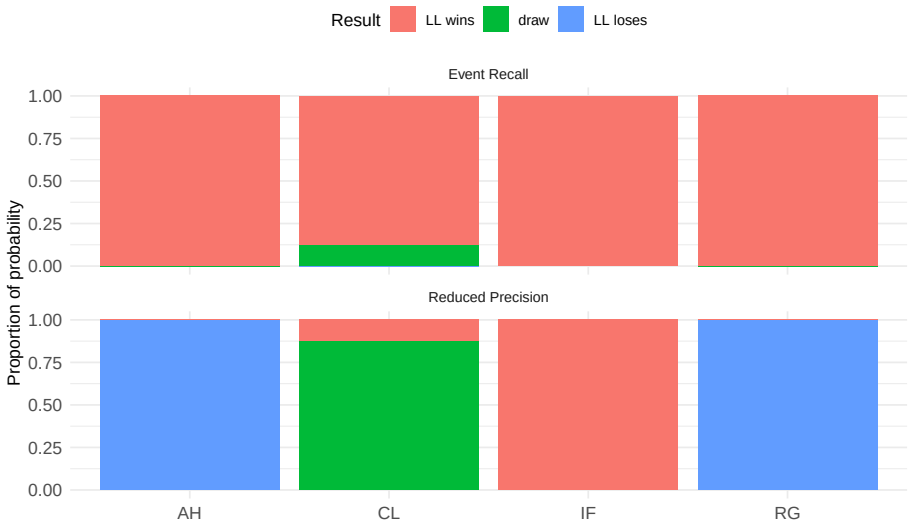


Figure 6.9: Comparing `CL` with `LL` with a Bayesian correlated t-test for ER and RP metrics (TE prediction)

Figure 6.8.

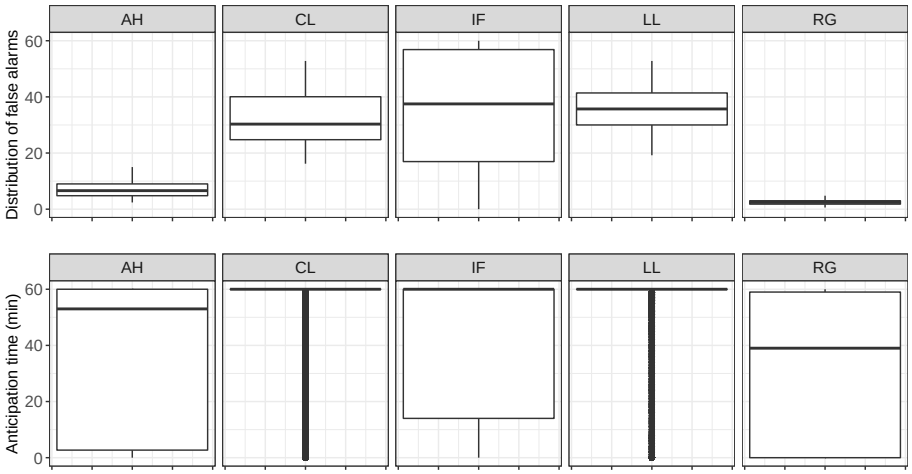


Figure 6.10: Distribution of the false alarms issued per hour and per patient (top), and the distribution of anticipation time (in minutes) per patient (low) for the TE prediction problem

6.4.6 Performance by Layer

We showed the competitiveness of LL relative to state of the art approaches to activity monitoring problems in two case studies: AHE and TE prediction (**RQ4.1**). We address the research question **RQ4.2** in this section and analyse the predictive performance of each layer in the proposed layered learning architecture. We split this evaluation into the following three parts:

RQ4.2.1 What is the performance of the model $h^{\mathcal{S}}$ (pre-conditional events sub-task)? That is, how well does the first layer of LL distinguish normal activity from the pre-conditional events $\mathcal{S}^{\text{AHE}}$ or $\mathcal{S}^{\text{TE}}$ (c.f. right side of Figure 6.4)?

RQ4.2.2 Assuming there is an impending pre-conditional event ($\mathcal{S}^{\text{AHE}}$ or $\mathcal{S}^{\text{TE}}$), what is the performance of the $h^{\mathcal{F}}$ model (main events sub-task)? In other words, how well does the second and final layer of LL is able to distinguish pre-conditional events from main events (AHE/TE)?

RQ4.2.3 Assuming that the first layer wrongly predicts that there is an impending pre-conditional event, i.e., a false positive in the pre-conditional events sub-task issued by $h^{\mathcal{S}}$. What is the performance of $h^{\mathcal{F}}$ (main events sub-task)? To be more precise, how well does the final layer of LL distinguish normal activity (but predicted to be a pre-conditional event by the first layer) from main events.

To answer these questions, we use standard binary classification metrics, namely recall, precision, F-score, and specificity. Note that in the previous subsections, we were concerned with how well each method captured the events of interest (AHE and TE). To evaluate this, we used appropriate measures (ER and RP). In this analysis, however, we want to understand the ability of each layer in LL to distinguish the different scenarios in each subsequence. The results are presented in Table 6.3 for AHE prediction, and Table 6.4 for TE prediction.

The model $h^{\mathcal{S}}$ presents a reasonable performance for capturing pre-conditional events, with an average F-score of 0.517 and 0.618, for AHE prediction and TE prediction, respectively (**RQ4.2.1**). The model $h^{\mathcal{F}}$ presents an average F-score of 0.402 and 0.670 for capturing main events inside the pre-conditional events

Table 6.3: Performance of each component in the proposed layered learning architecture for the AHE prediction problem.

Analysis	Recall	Precision	F-score	Specificity
RQ4.2.1	0.769±0.038	0.391±0.035	0.517±0.030	0.761±0.043
RQ4.2.2	0.683±0.103	0.290±0.062	0.402±0.064	0.595±0.108
RQ4.2.3	NA	NA	NA	0.524±0.126

Table 6.4: Performance of each component in the proposed layered learning architecture for the TE prediction problem.

Analysis	Recall	Precision	F-score	Specificity
RQ4.2.1	0.890±0.039	0.476±0.069	0.618±0.061	0.844±0.034
RQ4.2.2	0.821±0.278	0.616±0.065	0.670±0.163	0.188±0.289
RQ4.2.3	NA	NA	NA	0.183±0.281

data space (**RQ4.2.2**). While the recall of both these components seems adequate, the precision is lower than expected.

One of the main challenges behind using the proposed layered learning architecture is that errors may propagate from layer to layer. We evaluate one scenario where this might occur: when the model for pre-conditional events $h^{\mathcal{S}}$ issues a false alarm. That is, when it predicts that there is an impending pre-conditional event when in fact there is not (data remains as normal activity). We analyse how the second model for capturing main events $h^{\mathcal{F}}$ performs in such conditions. Ideally, $h^{\mathcal{F}}$ should ignore (i.e. classify as negative) all these observations.

The value of specificity averages at 0.524 and 0.183, for AHE prediction and TE prediction, respectively. These values represent the ratio of subsequences, which the model $h^{\mathcal{F}}$ can correctly identify a non-main event, after the first model $h^{\mathcal{S}}$ has issued a false alarm. Note that the true value is always negative, so only the specificity metric makes sense in this case. While the overall layered architecture generally performs better than the typical approach (`CL`), we believe that these results show that there is room for improvement.

6.4.7 Run-time Analysis

In the previous sections, we analysed LL in terms of predictive performance. In this section, we address the research question **RQ4.3** by analysing LL in terms of computation time. To accomplish this, we measure the time spent in fitting each method and using it to predict the test set of a given cross-validation fold.

Table 6.5: Average run-time in seconds, and respective standard deviation, of each method across the 50 folds

Method	AHE prediction	TE prediction
AH	**0**	**0**
CL	15.6±2.5	7.4±0.8
LL	102.3±12.5	70.9±4.9
IF	67.3±7.5	53.3±2.6
RG	416.3±30.1	344.2±21.7

Table 6.5 shows the average run-time in seconds, and respective standard deviation, of each method across the 50 folds. The relative results are similar for both AHE prediction, and TE prediction. LL takes, on average, more than one minute to run. Although this value is not considerable, both CL and IF take less time to compute than LL. The regression-based method RG is the one that takes more time to compute. This is expected since the underlying model is a multi-step forecasting method – a learning model is created for each point in the target window (i.e. 30 models), which significantly drives the run-time of this approach. Finally, AH is not a machine learning method. It is a rule derived from domain expertise which issues alarms when a determined variable exceeds some value. In effect, we consider the run-time of AH to be negligible.

6.4.8 Resampling Analysis

In this section, we address the research question **RQ4.4**. As we mentioned before, the proposed method LL and the state of the art approach CL are applied after pre-processing the data set with a resampling method. Resampling strategies are commonly used to mitigate the class imbalance problem (Branco et al., 2016a), including in CHE prediction problems (Lee and Mark, 2010b; Forkan

et al., 2017). We analyse the impact of several resampling strategies in terms of ER and RP. In this analysis, we focus on the methods `LL` and `CL`. The methods `IF` and `AH` do not require balancing the distribution. To our knowledge, `RG` has been applied to CHE prediction without such procedures (Lee and Mark, 2010b; Rocha et al., 2011).

We tested the following six different strategies:

- No resampling (NR), in which the distribution is left imbalanced;
- Random Under-sample (RU): in this strategy cases from the majority class are randomly removed until the distribution of classes is balanced;
- Random Over-sample (RO): Similarly to RU, in a RO approach random instances from the minority class are replicated the distribution of classes is balanced;
- SMOTE (Synthetic Minority Over-sampling Technique) (Chawla et al., 2002): instead of replicating instances from the minority class, SMOTE generates new synthetic observations similar to these. This is achieved by interpolation from a number of nearest neighbours;
- ADASYN (Adaptive Synthetic) (He et al., 2008): this method is another over-sampling technique which is similar to SMOTE. The core distinction is that ADASYN focuses on instances from the minority class which are more *difficult to learn*, i.e., closer to the decision boundary;
- TOMEK: Tomek links is an under-sampling method (Tomek, 1976). It works by finding pairs of observations which are the nearest neighbour of each other, but of different classes. One can then remove the instance from the majority class, or even the respective pair (Batista et al., 2004). We adopt the latter approach.

Table 6.6 shows the result of the analysis for the AHE prediction problem. Almost all the resampling approaches improve the sensitivity of the methods for the event of interest (i.e. higher ER). The exception is TOMEK for `LL`, and ADASYN for `CL`. However, there is a trade-off with RP, which generally decreases with the application of the resampling methods. Since in this particular domain of application we are focused on preventing CHEs, we choose the

resampling method emphasising ER. This justifies the pick of a RU approach for LL, and the choice for RO for CL.

Table 6.6: Results of the resampling analsys for the AHE problem (average across the 50 folds)

	LL		CL	
Method	**ER**	**RP**	**ER**	**RP**
NR	0.778±0.07	**0.107**±0.02	0.755±0.08	0.108±0.02
RU	**0.830**±0.05	0.090±0.02	0.792±0.07	0.091±0.02
RO	0.828±0.06	0.087±0.02	**0.807**±0.07	0.089±0.02
SMOTE	0.829±0.06	0.083±0.02	0.805±0.06	0.085±0.02
ADASYN	0.788±0.07	0.105±0.03	0.730±0.08	**0.112**±0.03
TOMEK	0.774±0.07	0.101±0.03	0.767±0.09	0.098±0.03

A similar analysis can be made for TE prediction, whose results are shown in Table 6.7. In this case, both methods (LL and CL) present their best results when applied with the SMOTE resampling strategy.

Table 6.7: Results of the resampling analsys for the TE problem (average across the 50 folds)

	LL		CL	
Method	**ER**	**RP**	**ER**	**RP**
NR	0.886±0.04	**0.165**±0.02	0.853±0.04	0.169±0.03
RU	0.924±0.03	0.141±0.02	0.903±0.04	0.138±0.02
RO	0.930±0.03	0.142±0.02	0.900±0.05	0.147±0.02
SMOTE	**0.938**±0.03	0.136±0.02	**0.919**±0.06	0.130±0.02
ADASYN	0.887±0.03	**0.165**±0.02	0.815±0.06	**0.179**±0.03
TOMEK	0.893±0.05	0.149±0.03	0.838±0.05	0.165±0.03

6.4.9 Discussion

On the Experimental Results

In the previous section, we provided empirical evidence for the advantages of using a layered learning approach for CHE prediction problems. We briefly discuss the results in this section. We also discuss the main challenges associated with the proposed approach.

`IF`, a state of the art approach to anomaly detection, shows a significantly worse performance relative to discriminating approaches, namely `LL` and `CL`, for both AHE and TE prediction problems. A regression approach (`RG`) also shows a significantly lower ER with respect to the other methods. It shows the highest RP, which indicates that this type of approach is conservative (low recall of events and low average number of false alarms). In summary, `LL` shows a competitive performance relative to state of the art approaches to solve activity monitoring predictive tasks.

As we mentioned before, the reported experiments were carried out using an extreme gradient boosting learning algorithm (Chen et al., 2015). This algorithm was used to train both layers of our approach (`LL`), and as a stand-alone classifier without layered learning (`CL`). Using this learning algorithm lead to the best overall results relative to other ones such as random forests, or a deep feedforward neural network. Notwithstanding, deep learning approaches, recurrent architectures in particular, have been increasingly applied in the healthcare domain (e.g. Tamilselvan and Wang (2013)). In future work, we will study these methods further, both as baselines and as possible solutions within a layered learning approach.

Challenges Behind Layered Learning

The main challenge behind layered learning is the assumption that the task decomposition is a domain-dependent function. This can be regarded as an opportunity for domain experts to embed their domain expertise in predictive models. Notwithstanding, nowadays there is an increasing interest for end-to-end automated machine learning technologies (Thornton et al., 2013; Feurer et al., 2015), and a manual decomposition can be regarded as a bottleneck. In

this context, future work includes the study of an automated methodology for identifying or learning the pre-conditional events from the data.

Although we focus on CHE prediction problems, our ideas for layered learning can be generally applied to other activity monitoring problems, for example, problems with complex targets, which can be decomposed into partial, simpler targets. While the task decomposition is dependent on the domain, we describe some guidelines which can facilitate its implementation.

We believe that layered learning is a promising area of research. In particular, we showed its competitiveness in a case study for activity monitoring, which is usually a difficult predictive task (Fawcett and Provost, 1999).

6.5 Related Work

6.5.1 Activity Monitoring

The focus of this work is on the timely detection of anomalies. This is a task that is also known in the literature as activity monitoring (Fawcett and Provost, 1999). The goal of this predictive task is to track a given activity over time and launch timely alarms about interesting events that require action. According to Fawcett and Provost (1999), there are two classes of methods for activity monitoring: profiling methods, and discriminating methods. We overview the idea behind these approaches in Section 2.4.4. We focus on the latter strategy, which is the one followed by the proposed layered learning method for activity monitoring. Notwithstanding, we compare our approach to `IF`, which is a method that follows the profiling strategy.

AHE prediction has been gaining increasing attention from the scientific community. For example, the 10^{th} annual PhysioNet / Computers in Cardiology Challenge focused on this predictive task (Moody and Lehman, 2009). While the methods used in this particular challenge are not state of the art anymore, the purpose of the reference is to show the relevance of the predictive task.

Like other activity monitoring problems or anomaly detection tasks, the typical approach to this problem is to use standard classification methods. This is the case of Lee and Mark, which use a feed-forward neural network as predictive model (Lee and Mark, 2010a). Tsur et al. (2018) follow a similar approach

and also propose an enhanced feature extraction approach before applying an extreme gradient boosting algorithm. In turn, Rocha et al. (2011) propose a regression approach by forecasting future values of blood pressure. In their approach, alarms for impending AHE are launched according to a deterministic function which receives as input the numeric predictions. TE prediction also is a relevant task. For example, Forkan et al. (2017) propose a predictive model for detecting several health conditions, including tachycardia and hypotension.

6.5.2 Layered Learning

Layered learning was proposed by Stone and Veloso (2000), and was specifically designed for scenarios with a complex mapping from inputs to outputs. In particular, they applied this approach to improve several processes in robotic soccer.

Decroos et al. (2017) apply a similar approach for predicting goal events in soccer matches. Instead of directly modelling such events, they first model goal attempts as what we call in this chapter as a pre-conditional events sub-task.

Layered learning stems from the more general topic of multi-strategy learning. Layered learning approaches run multiple learning processes to improve the generalisation in a predictive task. This is a similar strategy as ensemble learning methods, which we used in the previous part of this thesis. The main difference is that in layered learning, each layer addresses a different predictive task, while in ensemble learning the predictive task is typically a single one.

6.6 Conclusions

Layered learning approaches are designed to solve predictive tasks in which a direct mapping from inputs to outputs is difficult. In this chapter, we developed a layered learning approach for the early detection of anomalies in time series data. The idea is to break the original predictive task into two simpler predictive tasks, which are, in principle, easier to solve. We create an initial model that is designed to distinguish normal activity from a relaxed version of anomalous behaviour (pre-conditional events). A subsequent model is created to distinguish such pre-conditional events from the actual events of interest.

We have focused on predicting critical health conditions in ICUs, namely hypotension and tachycardia events. Compared to standard classification, which is a common solution to this type of predictive tasks, the proposed model can capture significantly more anomalous events with a comparable number of false alarms. The results also suggest that the proposed approach is better than other state of the art methods.

Part IV

Conclusions

Chapter 7

Conclusions

7.1 Main Conclusions

Time series data can be used to represent a plethora of real-world phenomena from a vast number of domains of application, including finance, healthcare, or transportation, to name a few. The way we expect the future to unfold deeply affects the decision-making process in the present. Therefore, and because the future is rather uncertain, organisations increasingly rely on forecasting mechanisms to make data-driven decisions.

In this context, the aim of this thesis settled on developing new approaches for predicting the future behaviour of time series. That is, our goal was to leverage historical data that is collected over time and help organisations make sense of the future in an accurate and timely manner. In particular, our objective was two-fold: (1) to develop new methodologies for automatically predicting the next values of time series, which are able to cope with potential non-stationarities that often characterise this type of data; and (2) to create new methods for predicting interesting events in a timely manner from a set of activities being monitored over time.

7.1.1 Forecasting

The main body of this thesis was split into two parts according to the objectives set forth. In the first main part (*Forecasting*), we addressed the classical problem

of forecasting, i.e., the prediction of the next value of a numeric time series given its historical values. This represents the "classical" problem of time series forecasting as it has been studied for many decades. In the introductory chapter of this thesis, we divided the first objective into three research questions. We now recapitulate them, and answer them in turn:

RQ1 *Given the time dependency among observations, what is the most appropriate way of estimating the predictive performance of forecasting models?*

In Chapter 3, we carried out an extensive empirical study comparing different methods for evaluating the predictive performance of forecasting models. We broadly split the tested methods into two categories: cross-validation methods, which make efficient use of the available observations but assume them to be independent; and out-of-sample approaches, which preserve the natural order of the data but do not test the respective forecasting model in all observations (less efficient use of data). We designed the experimental setup to control for different scenarios, including data generating processes and stationarity conditions. Overall, the results point out that cross-validation methods have a competitive performance estimation ability when time series are stationary. However, when non-stationary sources of variation are at play, out-of-sample approaches are better for this task. Particularly, the holdout method repeated over several testing periods shows the overall best estimation ability. This method was used to estimate the performance of several time series forecasting approaches in experiments carried out in Chapters 4 and 5.

RQ2 *How can we dynamically combine a set of forecasting experts and cope with their time-varying relative performance?*

We address the problem of time series forecasting using an ensemble learning approach. We create a portfolio of forecasting experts, which are dynamically combined over time to cope with concept drift. The typical set of predictive models applied to time series forecasting includes approaches such as ARIMA or exponential smoothing. In this thesis, we show that standard regression learning algorithms (e.g. rule-based regression, Gaussian processes) are also competitive for these predictive tasks when applied

as auto-regressive processes. The main challenge behind dynamic ensemble methods is how to select the weights of each one of the individual members of the ensemble to the final decision in each time step. State of the art approaches rely on sliding statistics or forgetting mechanisms that summarise the recent relative performance of the ensemble members. Our working hypothesis was that this type of approaches might have a short memory and fail to capture long-range relationships between changes in the environment and the performance of the available experts in an efficient manner. In Chapter 4, we proposed a new meta-learning model, dubbed Arbitrated Dynamic Ensemble, to dynamically combine a set of forecasting models which is based on arbitration. We explore differences among the members of the ensemble and use a regression analysis to specialise them across the data space of a time series. We carried experiments on a large set of time series. The proposed methodology shows a consistent advantage when compared with state of the art methods. On top of this, we also show the importance of the different building blocks comprising the proposed method. We analysed the sensitivity of `ADE` to different parameters, training strategies, and the value of additional experts in the ensemble.

RQ3 *Can we dynamically aggregate a set of forecasting experts using a combination of aggregation functions to achieve a better trade-off between diversity and individual error of the members of the ensemble?*

The core assumption behind the adoption of an ensemble approach to tackle the problem of time series forecasting is variance in relative performance shown by predictive models throughout a time series. We hypothesised that different subsets of models might behave similarly. Moreover, we also hypothesised that, if this happens, aggregating these subsets may lead to better predictive performance due to a better trade-off between diversity and individual error of the members of the ensemble. In order to explore these ideas, in Chapter 5, we propose a combination approach named constructive aggregation and apply it to time series forecasting. The gist of the method is to, instead of directly aggregating forecasting models, first rearrange them into different subsets, creating a new set

of combined models which is then aggregated into a final decision. The results obtained corroborate our hypotheses and suggest that indeed applying state of the art dynamic aggregation methods using CA is better than applying them directly to the original set of forecasting models.

7.1.2 Activity Monitoring

Despite the importance of the task addressed in the *Forecasting* part of the thesis, predicting the future behaviour of time series is a multifaceted problem. In the second main part (*Activity Monitoring*), we were concerned with a different sort of prediction regarding the future of a time series. Rather than predicting the value of upcoming observations, activity monitoring is concerned with the timely detection of specific events of interest. These are typically disruptive in the respective domain and require some action from professionals. Our goal in this part of the thesis can be summarised in the fourth research question formulated in the introduction of the thesis:

RQ4 *How can we better cope with the low frequency of events of interest and detect more of them in a timely manner?*

We focused on interesting events which are based on the values of a numeric variable. For example, an event may occur when a set of consecutive numeric values is below some pre-defined threshold. We propose a new methodology for activity monitoring problems based on a layered learning approach. As such, we split the original problem into two hierarchical layers. The first layer is designed to capture what we define as pre-conditional events: related versions of the actual events of interest, but which are more frequent and, as we hypothesise, easier to model. The subsequent and final layer is designed to distinguish pre-conditional events from the actual events of interest. We validated our approach using two case studies from the healthcare domain, namely acute hypotensive episode prediction and tachycardia episode prediction. In both scenarios, the proposed method is able to outperform state of the art methods, including a standard approach based on classification (i.e., no hierarchical decomposition of the problem), and an Isolation Forest, which is a state of the art approach to

anomaly detection.

7.1.3 Ensemble Methods

Ensemble learning is the field of machine learning in which several predictive models are combined to tackle a given predictive task. Both theoretical and empirical studies have shown the predictive advantage of these methods (Ueda and Nakano, 1996; Breiman, 1996). We adopt an ensemble learning strategy to solve the predictive tasks addressed in this thesis.

In the *Forecasting* part of the thesis, we trained a set of forecasting models with distinct inductive biases (heterogeneous ensembles). The models in the ensemble are dynamically combined to cope with the different regimes causing the time series. In the *Activity Monitoring* part of the thesis, we proposed a layered learning approach to solve activity monitoring problems. Layered learning approaches are related to ensemble learning in the sense that they create and combine several predictive models to address a given predictive task. The key difference is that, in layered learning, each predictive model addresses a different predictive task.

7.2 Open Issues and Future Directions

The methods presented in the thesis can be improved in a number of ways. In this section, we outline some potentially interesting research directions.

7.2.1 Towards an Automated Forecasting Method

Although machine learning plays an increasingly important role in science and technology, applying these methodologies in practice still requires a significant amount of technical expertise. We proposed a method (`ADE`) that dynamically combines a set of forecasting models and automatically adapts itself to the different dynamics of a time series. However, as a forecasting tool, the method is not completely automatic, and there are still important tasks left to the end-user. For example, how many models to form the ensemble, how to set the parameters of the aggregation method, or when to update the available forecasting models. We believe that automating these tasks and developing a

completely automated machine learning tool for forecasting is important for an increased adoption, especially by non-technical professionals (Feurer et al., 2015).

Explainable Machine Learning

Another interesting research line is that of explainable machine learning, which is gaining increasing attention from the scientific community (Ribeiro et al., 2016b). Developing a forecasting method which is able to provide some sort of explanation for its predictions could improve the decision-making process by professionals.

Concept Drift Adaptation

`ADE` is based on a regression analysis on the historical loss of each forecasting model available. When the underlying concept driving a time series drifts to a regime unknown hitherto, the predictive performance of the combined model may decrease significantly. In this context, an interesting research line to follow on the proposed methodology and making it more automatic is the development of efficient methods for concept drift adaptation. Gama et al. (2014) survey several existing methods in the literature to this effect.

7.2.2 Beyond Univariate Time Series and One Step Ahead Point Estimation

In the *Forecasting* part of the thesis, we formalised the predictive task as the prediction of (y_{n+1}) according to $\{y_1, \ldots, y_n\}$. This could be regarded as a simplistic problem from three perspectives. First, often we have other related time series available, which can be used as explanatory variables in the predictive models and significantly improve their performance. Secondly, in the decision-making process, it is crucial to quantify the uncertainty behind predictions. This could be achieved in time series by adopting probabilistic forecasting models. That is, instead of predicting a point estimate $\hat{y}_{n+1}$ to approximate y_{n+1}, it may be more useful to produce a predictive probability distribution over future quantities (Gneiting and Katzfuss, 2014), or a prediction interval (Chatfield, 1993; Torgo and Ohashi, 2011). Finally, one may be interested in predicting more than

one step ahead in the future, for example, due to delayed feedback. Throughout the dissertation, we assumed immediate feedback from the environment, i.e., the n-th observation is known when predicting y_{n+1}. Predicting multiple steps ahead is typically a more difficult task due to the increased uncertainty (Taieb et al., 2012).

In future research, we will study the application of the proposed methodologies in these scenarios. We note that we address a simpler predictive task because we focus on the dynamic aggregation of a set of forecasting models, which is one of the main goals of this part of the thesis.

7.2.3 Data Stream Mining

Data stream mining is a related topic to this thesis. A data stream denotes a time series where there are limited computing and storage capacities. This type of time series is typically recorded in high velocities, leading to a large volume of observations. In this context, data points often need to be processed online. Many learning algorithms have been adapted to work in these scenarios, for example, the Hoeffding Tree (Domingos and Hulten, 2000), Leveraging Bagging (Bifet et al., 2010), among others (Gama, 2010). The methodologies developed in this thesis were devised for offline processes. Given the run-time scalability limitations reported in Chapters 4 and 6, applying the proposed approaches to data streams may be challenging. This represents a potentially interesting research direction. Particularly, after the publication of `ADE` (Cerqueira et al., 2019), Boulegane et al. (2019) proposed a method dubbed `STREAMING-ADE`. As the name suggests, their approach is an extension of `ADE` designed to cope with data streams.

7.2.4 Constructive Aggregation

We presented `CA` as a new methodology for aggregating a set of predictive models. We applied this approach to time series forecasting using dynamic ensemble methods such as `ADE`. In future work, we intend to apply `CA` in other domains of application, for example, standard regression tasks with independent and identically distributed data. We will also study different ways of computing the set of subsets $\mathcal{C}_M$, for example, taking into account not only predictive performance

but also the diversity within and across each subset.

7.2.5 Layered Learning Approaches to Activity Monitoring

In the part of the thesis related to activity monitoring, we focused on two case studies concerning healthcare, specifically the early detection of critical health events. In future work, it would be important to verify the applicability of the method in different domains of application, for example, predictive maintenance or fraud detection.

In the two case studies, the events of interest (AHE and TE) were defined according to the values of a numeric variable. It remains an open challenge to apply the proposed method to other types of events, and perhaps develop a more general definition for pre-conditional events. Another challenge regarding the proposed approach is the manual definition of the pre-conditional events. The automatic identification of events of pre-conditional events could be important for an increased adoption of this method. Finally, the proposed layered learning architecture for activity monitoring settled on two layers and one pre-conditional event. There may be scenarios in which the definition of multiple layers and pre-conditional events may be worthwhile.

We analysed different resampling strategies to balance the class distribution and measured their impact in terms of predictive performance. Overall, these approaches improved the performance of the predictive models. Studying other resampling methods to this type of data may be an interesting research direction. For example, the approach proposed by Moniz et al. (2017), which is designed to cope with temporal dependencies among observations.

7.2.6 On Performance Estimation

The scope of the analysis presented in Chapter 3 regarding performance estimation for time series forecasting is limited by the assumptions we made in Section 7.2.2. Moreover, the data sets used (c.f. Table A.1) are also biased in some properties, for example, the length of the time series or sampling frequency.

While some conclusions were drawn, the results suggested that the most

appropriate estimation method may be dependent on the characteristics of the time series, such as stationarity conditions. In this context, it may be interesting to develop an automatic procedure that selects the most appropriate estimation method according to the characteristics of the data. Such a procedure could be embedded in an automated forecasting system.

Appendix A

Data Sets and Learning Algorithms

A.1 Time Series Data Sets

In this section, we describe the datasets used in Chapters 3, 4, and 5. The data consists of a set of 62 time series from several domains of application. Each one of the 62 time series is a univariate sequence of numeric observations captured at regular intervals. The time series are summarised in Table A.1 and can be described as follows:

1–3 Time series with ID 1–3 represent data about water consumption levels, which was collected from three different locations from the city of Oporto, Portugal (ADDP, 2019). The data was collected from November 11, 2015, to January 11, 2016, with a granularity of half-hour;

4–7 These time series regarding solar radiation were collected by the Oak Ridge National Laboratory in Tennessee, USA (Maxey and Andreas, 2007). The data set includes global horizontal radiation, direct radiation, diffuse horizontal radiation, and average wind speed. The time series are aggregated on an hourly basis and the time period ranges from April 25, 2016, to August 25, 2016;

8–10 Three time series collected from a bike-sharing scenario (Fanaee-T and

Table A.1: Data sets and respective summary. In the interest of conciseness, the meaning of the content of table is detailed in the text.

ID	Time series	Data source	Data characteristics	Size	p	I	S	$\lvert\mathcal{C}_M\rvert$
1	Rotunda AEP	Porto Water Consumption from different locations in the city of Porto	Half-hourly values from Nov. 11, 2015 to Jan. 11, 2016	3000	30	0	0	76
2	Preciosa Mar			3000	9	1	0	224
3	Amial			3000	11	0	0	86
4	Global Horizontal Radiation	Solar Radiation Monitoring	Hourly values from Apr. 25, 2016 to Aug. 25, 2016	3000	23	1	0	20
5	Direct Normal Radiation			3000	19	1	1	77
6	Diffuse Horizontal Radiation			3000	18	1	1	42
7	Average Wind Speed			3000	10	1	0	79
8	Humidity	Bike Sharing	Hourly values from Jan. 1, 2011 Mar. 01, 2011	1338	11	0	0	40
9	Windspeed			1338	12	0	1	173
10	Total bike rentals			1338	8	0	1	39
11	AeroStock 1	Stock price values from different aerospace companies	Daily stock prices from January 1988 through October 1991	949	6	1	1	32
12	AeroStock 2			949	13	1	0	93
13	AeroStock 3			949	7	1	1	31
14	AeroStock 4			949	8	1	1	82
15	AeroStock 5			949	6	1	1	216
16	AeroStock 6			949	10	1	1	36
17	AeroStock 7			949	8	1	1	46
18	AeroStock 8			949	8	1	1	38
19	AeroStock 9			949	9	1	1	31
20	AeroStock 10			949	8	1	1	168
21	CO.GT	Air quality indicators in an Italian city	Hourly values from Mar. 10, 2004 to Apr. 04 2005	3000	30	1	0	44
22	PT08.S1.CO			3000	8	1	0	31
23	NMHC.GT			3000	10	1	0	66
24	C6H6.GT			3000	13	0	1	35
25	PT08.S2.NMHC			3000	9	0	0	31
26	NOx.GT			3000	10	1	1	33
27	PT08.S3.NOx			3000	10	1	0	42
28	NO2.GT			3000	30	1	0	32
29	PT08.S4.NO2			3000	8	0	0	39
30	PT08.S5.O3			3000	8	0	1	26
31	Temperature			3000	8	1	0	33
32	RH			3000	23	1	0	31
33	Humidity			3000	10	1	0	71
34	Electricity Total Load	Hospital Energy Loads	Hourly values from Jan. 1, 2016 to Mar. 25, 2016	3000	19	0	1	51
35	Equipment Load			3000	30	0	1	52
36	Gas Energy			3000	10	1	1	40
37	Gas Heat Energy			3000	13	1	1	27
38	Water heater Energy			3000	30	0	1	49
39	Total Demand	Australian Electricity	Half-hourly values from Jan. 1, 1999 to Mar. 1, 1999	2833	6	0	1	100
40	Recommended Retail Price			2833	19	0	0	79
41	Sea Level Pressure	Ozone Level Detection	Daily values from Jan. 2, 1998 to Dec. 31, 2004	2534	9	0	1	137
42	Geo-potential height			2534	7	0	1	86
43	K Index			2534	7	0	1	46

Table A.2: Continuation of Table A.1

ID	Time series	Data source	Data characteristics	Size	p	I	S	$\lvert\mathcal{C}_M\rvert$
44	Flow of Vatnsdalsa river	Data market	Daily, from Jan. 1, 1972 to Dec. 31, 1974	1095	11	0	0	60
45	Rainfall in Melbourne		Daily, from from 1981 to 1990	3000	29	0	0	56
46	Foreign exchange rates		Daily, from Dec. 31, 1979 to Dec. 31, 1998	3000	6	1	0	255
47	Max. Temp. in Melbourne		Daily, from from 1981 to 1990	3000	7	0	1	120
48	Min. Temp. in Melbourne		Daily, from from 1981 to 1990	3000	6	0	1	128
49	Rainfall in River Hirnant		Half-hourly, from Nov. 1, 1972 to Dec. 31, 1972	2928	6	1	0	28
50	IBM common stock prices		Daily, from Jan. 2, 1962 to Dec. 31, 1965	1008	10	1	0	42
51	Internet traffic data I		Hourly, from Jun. 7, 2005 to Jul. 31, 2005	1231	10	0	1	50
52	Internet traffic data II		Hourly, from Nov. 19, 2004 to Jan. 27, 2005	1657	11	1	0	77
53	Internet traffic data III		from Nov. 19, 2004 to Jan. 27, 2005 – Data collected at five minute intervals	3000	6	1	0	62
54	Flow of Jokulsa Eystri river		Daily, from Jan. 1, 1972 to Dec. 31, 1974	1096	21	0	0	41
55	Flow of O. Brocket		Daily, from Jan. 1, 1988 to Dec. 31, 1991	1461	6	1	0	121
56	Flow of Saugeen river I		Daily, from Jan. 1, 1915 to Dec. 31, 1979	1400	6	0	0	97
57	Flow of Saugeen river II		Daily, from Jan. 1, 1988 to Dec. 31, 1991	3000	30	0	0	36
58	Flow of Fisher River		Daily, from Jan. 1, 1974 to Dec. 31, 1991	1461	6	0	1	64
59	No. of Births in Quebec		Daily, from Jan. 1, 1977 to Dec. 31, 1990	3000	6	1	1	154
60	Precipitation in O. Brocket		Daily, from Jan. 1, 1988 to Dec. 31, 1991	1461	29	0	0	24
61	Min. temperature	Porto weather	Daily values from Jan. 1, 2010 to Dec. 28, 2013	1456	8	0	1	102
62	Max. temperature			1456	10	0	0	34

Gama, 2014). Each observation of each one of the time series contains hourly information about humidity levels, average wind speed, and total bike rentals, respectively. The data was collected from January 1, 2011, to March 1, 2011;

11–20 These time series represent the daily stock prices of 10 aerospace companies (Vlachos, 2000). The data ranges from January 1988 to October 1991;

21–33 The time series with IDs from 21 to 33 represent hourly averages of air quality indicators (Lichman, 2013). These were captured by an air quality chemical multi-sensor device stationed at road level within an Italian city. The data was recorded from March 10, 2004, to April 4, 2005;

34–38 We collected 5 time series related to energy consumption in an hospital (EERE, 2017). This includes energy spent on electricity and on gas. The data was captured on an hourly basis from January 1, 2016, to March 25, 2016;

39–40 The total demand for electricity, and respective recommended retail price from an Australian region (Koprinska et al., 2011). The data was collected from January 1, 1999, to March 1, 1999, with a granularity of 30 minutes;

41–43 These time series are related to the sea level pressure, the K-index, and geo-potential height, respectively (Lichman, 2013). These are important collections of observations for ozone level detection problems;

44–60 We use several time series from the Data Market (Hyndman, 2019), which includes several domains of application such as finance, internet traffic, or meteorology. Some characteristics are described in Table A.1;

61–62 These time series represent daily observations of the minimum and maximum temperatures in the city of Oporto, Portugal (Oliveira and Torgo, 2014). The data was collected from January 1, 2010, to December 28, 2013.

Overall, the sampling frequency of the data sets we used is daily or higher. This type of high-sampling frequency time series is becoming increasingly relevant due to the widespread collection of sensor data (e.g. internet-of-things). An important consequence of the sampling frequency point is data size. Higher sampling rate is usually associated with more data, which is crucial for the regression models we use (and for machine learning in general) to generalise well in unseen observations. In this context, the 62 time series in our portfolio comprise close to at least close to one thousand observations. Moreover, in the forecasting analysis carried out in chapters 3–5, the size of some of the time series was truncated to 3000 observations to speed up computations.

In the table, the column p denotes the embedding dimension of the respective time series. Our approach for estimating this parameter is addressed in Section 3.3.4 of the thesis. Differencing is the computation of the differences between consecutive observations. This process is useful to remove changes in the level of a time series, thus stabilising the mean (Hyndman and Athanasopoulos, 2018). This is important to account for trend and seasonality in time series. The column I represents the number of differences applied to the respective time series in order to make it trend-stationary according to the KPSS test (Kwiatkowski et al., 1992). The column S represents whether or not a time series is stationary (1 if it is, 0 if it is not) according to the wavelet spectrum test proposed by Nason (2013). Finally, the column $|\mathcal{C}_M|$ represents the size of the set of subsets created for the constructive aggregation of forecasting experts using out-performance contiguity, which is addressed in Chapter 5.

A.2 Learning Algorithms

Table A.3 summarises the hyper-parameters of the learning algorithms used in the experiments shown in Chapter 4. We use state of the art methods for time series forecasting, such as ARIMA and exponential smoothing. We also use several standard regression learning algorithms which are applied in an auto-regressive fashion. The list of learning algorithms are the following:

`SVR` Support vector regression (Scholkopf and Smola, 2001) with linear, radial basis function, Laplacian, and polynomial kernels. The parameter for

cost of constraints violation is set to 1 (default), and the epsilon in the insensitive-loss function is set to 0.1 (default). We used the implementation from the R package *kernlab* (Karatzoglou et al., 2004);

`MARS` Multivariate adaptive regression splines (Friedman et al., 1991) with different parameters regarding the maximum degree of interaction (Degree), and the maximum number of model terms before pruning (No. terms). The forward stepping threshold is set up to 0.001 (default). We used the implementation from the R package *earth* (Milborrow, 2012);

`RF` Random forests (Breiman, 2001) with a varying number of trees (100, 250, and 500) (Wright, 2015). The number of variables to possibly split at in each node is set to a third of the number of predictor variables (Breiman, 2001);

`PPR` Projection pursuit regression with different number of terms (2, 5), and two different methods used for smoothing the ridge functions: the Friedman's super smoother (Friedman, 1984) or the smoothing spline (Green and Silverman, 1993) approach (Friedman and Stuetzle, 1981; R Core Team, 2013). We used the implementation from the R package *stats* (R Core Team, 2013);

`RBR` Rule-based regression based on Quinlan's model tree (Quinlan, 1993) with a varying number of boosting iterations (10, 25, 50, and 100). We used the implementation provided in the *cubist* R package (Kuhn et al., 2014);

`GBR` Generalized boosted regression (Elith et al., 2008) with a Gaussian or Laplacian distribution. The number of trees is set to either 500 or 1000. The maximum depth of each tree is set to 5 or 10, and the shrinkage parameter applied to each tree in the expansion is set to 0.1 (default). We used the implementation from the R package *gbm* (Ridgeway, 2015);

`MLP` A multi-layer perceptron (Rumelhart et al., 1985) with a varying number of hidden units (3, 5, 7, 10, 15, and 25), and weight decay set to 0.01 (default). We used the implementation from the R package *nnet* (Venables and Ripley, 2002);

`GLM` Generalized linear model (McCullagh, 2019) regression with a Gaussian distribution and a different penalty mixing. When the penalty is set to 1, the algorithm represents LASSO (Least Absolute Shrinkage and Selection Operator) regression, and Ridge regression when it is set to 0. In between 0 and 1, the algorithm is a linear model with elastic net regularisation. These models are implemented in the R package *glmnet* (Friedman et al., 2010);

`GP` Gaussian processes (Rasmussen, 2003) with linear, radial basis function, Laplacian, and polynomial kernels. The tolerance of termination criterion is set to either 0.01 or 0.001. We used the implementation provided in the R package *kernlab* (Karatzoglou et al., 2004);

`PCR` Principal components regression (Jolliffe, 1982) with a default parameter setting provided in the *pls* R package (Mevik et al., 2016);

`PLS` Partial least squares (Geladi and Kowalski, 1986) regression with two different methods: the kernel method, and the SIMPLS method. These are also provided in the *pls* R package (Mevik et al., 2016);

`ARIMA` The Auto-Regressive Integrated Moving Average (Hyndman and Athanasopoulos, 2018) method. Parameters are optimally set according the the auto.arima function from the *forecast* R package (Hyndman et al., 2014);

`ETS` The exponential smoothing state space model with two methods: The ETS method with automatic parameter setting from the *forecast* R package (Hyndman et al., 2014), and the TBATS method, which includes Box-Cox transformation, ARMA errors, trend and seasonal components (De Livera et al., 2011; Hyndman et al., 2014).

In Figure 4.2, we show the distribution of rank of each method across 62 time series, which shows the competitiveness of the different learning algorithms.

Table A.3: Summary of the learning algorithms

ID	Algorithm	Parameter	Value
SVR	Support Vector Regr.	Kernel	{Linear, RBF Polynomial, Laplace}
		Cost	{1}
		ϵ	{0.1}
MARS	Multivar. A. R. Splines	Degree	{1, 3}
		No. terms	{7, 15}
		Forward thresh.	{0.001}
RF	Random forest	No. trees	{100, 250, 500}
PPR	Proj. pursuit regr.	No. terms	{2, 5}
		Method	{super smoother, spline}
RBR	Rule-based regr.	No. iterations	{10, 25, 50, 100}
GBR	Generalized Boosted regr.	Depth	{5, 10}
		Distribution	{Gaussian,Laplace}
		No. trees	{500, 1000}
		Learning rate	{0.1}
MLP	Multi-layer Perceptron	Hidden units	{3, 5, 7, 10, 15, 25}
		Decay	{0.01}
GLM	Generalised Linear Regr.	Penalty mixing	{0, 0.2, 0.4, 0.6, 0.8, 1}
GP	Gaussian Processes	Kernel	{Linear, RBF, Polynomial, Laplace}
		Tolerance	{0.001, 0.01}
PCR	Principal Comp. Regr.	*Default*	-
PLS	Partial Least Regr.	Method	{kernel, SIMPLS}
ARIMA	ARIMA	*Auto*	-
ETS	Exp. Smoothing	Method	{ETS, TBATS}

References

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

ADDP, "Oporto water consumption," http://addp.pt, 2019, accessed: 2016-11-21.

M. Aiolfi and A. Timmermann, "Persistence in forecasting performance and conditional combination strategies," *Journal of Econometrics*, vol. 135, no. 1, pp. 31–53, 2006.

A. Antonucci, M. Scanagatta, D. D. Mauá, and C. P. de Campos, "Early classification of time series by hidden markov models with set-valued parameters," in *Proceedings of the NIPS Time Series Workshop*, 2015.

S. Arlot, A. Celisse *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

J. S. Armstrong, "Combining forecasts: The end of the beginning or the beginning of the end?" *International Journal of Forecasting*, vol. 5, no. 4, pp. 585–588, 1989.

J. Asafu-Adjaye, "The relationship between energy consumption, energy prices and economic growth: time series evidence from asian developing countries," *Energy economics*, vol. 22, no. 6, pp. 615–625, 2000.

L. Baía and L. Torgo, "A comparative study of approaches to forecast the correct trading actions," *Expert Systems*, vol. 34, no. 1, p. e12169, 2017.

L. C. G. Baía, "Actionable forecasting and activity monitoring: applications to financial trading," Master's thesis, University of Porto, 2015.

J. M. Bates and C. W. Granger, "The combination of forecasts," *Journal of the Operational Research Society*, vol. 20, no. 4, pp. 451–468, 1969.

G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Machine learning*, vol. 36, no. 1-2, pp. 105–139, 1999.

R. Bellazzi and B. Zupan, "Predictive data mining in clinical medicine: current issues and guidelines," *International journal of medical informatics*, vol. 77, no. 2, pp. 81–97, 2008.

A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2653–2688, 2017.

C. Bergmeir and J. M. Benitez, "Forecaster performance evaluation with cross-validation and variants," in *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on.* IEEE, 2011, pp. 849–854.

C. Bergmeir and J. M. Benítez, "On the use of cross-validation for time series predictor evaluation," *Information Sciences*, vol. 191, pp. 192–213, 2012.

C. Bergmeir, M. Costantini, and J. M. Benítez, "On the usefulness of cross-validation for directional forecast evaluation," *Computational Statistics & Data Analysis*, vol. 76, pp. 132–143, 2014.

C. Bergmeir, R. J. Hyndman, and B. Koo, "A note on the validity of cross-validation for evaluating autoregressive time series prediction," *Computational Statistics & Data Analysis*, vol. 120, pp. 70–83, 2018.

A. Bifet and R. Kirkby, *Data stream mining a practical approach.* Citeseer, 2009.

A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2010, pp. 135–150.

D. Boulegane, A. Bifet, S. El-Bouch, and G. Madhusudan, "Arbitrated dynamic ensemble with abstaining for time series forecasting on data streams," in *4th ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, AALTD'19*, 2019.

G. E. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 26, no. 2, pp. 211–243, 1964.

G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

P. Branco, R. P. Ribeiro, and L. Torgo, "UBL: an r package for utility-based learning," *CoRR*, vol. abs/1604.08079, 2016.

P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Computing Surveys (CSUR)*, vol. 49, no. 2, p. 31, 2016.

P. Brazdil, C. G. Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

——, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

——, *Classification and regression trees*. Routledge, 2017.

P. J. Brockwell and R. A. Davis, *Time series: theory and methods*. Springer Science & Business Media, 2013.

G. Brown, *Encyclopedia of Machine Learning*. Boston, MA: Springer US, 2010, ch. Ensemble Learning, pp. 312–320.

——, "An information theoretic perspective on multiple classifier systems," in *International Workshop on Multiple Classifier Systems*. Springer, 2009, pp. 344–353.

——, "Ensemble learning," *Encyclopedia of Machine Learning*, pp. 312–320, 2010.

G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

G. Brown, J. L. Wyatt, and P. Tiňo, "Managing diversity in regression ensembles," *Journal of machine learning research*, vol. 6, no. Sep, pp. 1621–1650, 2005.

R. G. Brown, *Statistical forecasting for inventory control*. McGraw/Hill, 1959.

D. W. Bunn, "A bayesian approach to the linear combination of forecasts," *Journal of the Operational Research Society*, vol. 26, no. 2, pp. 325–329, 1975.

H. Cao, L. Eshelman, N. Chbat, L. Nielsen, B. Gross, and M. Saeed, "Predicting icu hemodynamic instability using continuous multiparameter trends," in *2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2008, pp. 3803–3806.

J. Carbonell and J. Goldstein, "The use of mmr, diversity-based reranking for reordering documents and producing summaries," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1998, pp. 335–336.

G. A. Carpenter, S. Grossberg, and J. H. Reynolds, "Artmap: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network," *Neural Netw.*, vol. 4, no. 5, pp. 565–588, Sep. 1991.

R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 18.

V. Cerqueira, L. Torgo, F. Pinto, and C. Soares, "Arbitrated ensemble for time series forecasting," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2017, pp. 478–494.

V. Cerqueira, L. Torgo, J. Smailović, and I. Mozetič, "A comparative study of performance estimation methods for time series forecasting," in *Data Science and Advanced Analytics (DSAA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 529–538.

V. Cerqueira, L. Torgo, and C. Soares, "Arbitrated ensemble for solar radiation forecasting," in *International Work-Conference on Artificial Neural Networks*. Springer, 2017, pp. 720–732.

V. Cerqueira, L. Torgo, F. Pinto, and C. Soares, "Arbitrage of forecasting experts," *Machine Learning*, vol. 108, no. 6, pp. 913–944, 2019.

N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games*. New York, NY, USA: Cambridge University Press, 2006.

——, "Potential-based algorithms in on-line prediction and game theory," *Machine Learning*, vol. 51, no. 3, pp. 239–261, 2003.

N. H. Chan, *Time series: applications to finance*. John Wiley & Sons, 2004, vol. 487.

P. K. Chan and S. J. Stolfo, "Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection." in *KDD*, vol. 1998, 1998, pp. 164–168.

V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, p. 15, 2009.

C. Chatfield, "Calculating interval forecasts," *Journal of Business & Economic Statistics*, vol. 11, no. 2, pp. 121–135, 1993.

——, *Time-series forecasting*. CRC Press, 2000.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

K. Chen, L. Xu, and H. Chi, "Improved learning algorithms for mixture of experts in multiclass classification," *Neural networks*, vol. 12, no. 9, pp. 1229–1252, 1999.

T. Chen, T. He, M. Benesty, V. Khotilovich, and Y. Tang, "xgboost: Extreme gradient boosting, 2017," *R package version 0.6-4*, 2015.

R. T. Clemen, "Combining forecasts: A review and annotated bibliography," *International journal of forecasting*, vol. 5, no. 4, pp. 559–583, 1989.

R. T. Clemen and R. L. Winkler, "Combining economic forecasts," *Journal of Business & Economic Statistics*, vol. 4, no. 1, pp. 39–46, 1986.

C. Croarkin, P. Tobias, J. Filliben, B. Hembree, W. Guthrie *et al.*, "Nist/sematech e-handbook of statistical methods," *NIST/SEMATECH, July. Available online: http://www. itl. nist. gov/div898/handbook*, 2006.

A. P. Dawid, "Present position and potential developments: Some personal views: Statistical theory: The prequential approach," *Journal of the Royal Statistical Society. Series A (General)*, pp. 278–292, 1984.

A. M. De Livera, R. J. Hyndman, and R. D. Snyder, "Forecasting time series with complex seasonal patterns using exponential smoothing," *Journal of the American Statistical Association*, vol. 106, no. 496, pp. 1513–1527, 2011.

T. Decroos, V. Dzyuba, J. Van Haaren, and J. Davis, "Predicting soccer highlights from spatio-temporal match event streams." in *AAAI*, 2017, pp. 1302–1308.

D. A. Dickey and W. A. Fuller, "Likelihood ratio statistics for autoregressive time series with a unit root," *Econometrica: Journal of the Econometric Society*, pp. 1057–1072, 1981.

T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

T. G. Dietterich and G. Bakiri, "Error-correcting output codes: A general method for improving multiclass inductive learning programs," in *AAAI*, 1991, pp. 572–577.

T. G. Dietterich *et al.*, "Ensemble methods in machine learning," *Multiple classifier systems*, vol. 1857, pp. 1–15, 2000.

P. Domingos and G. Hulten, "Mining high-speed data streams," in *Kdd*, vol. 2, 2000, p. 4.

EERE, "Commercial and residential hourly load profiles tmy3 loc. in the usa," http://en.openei.org/datasets/files/961/pub/, 2017, accessed: 2016-11-21.

J. Elith, J. R. Leathwick, and T. Hastie, "A working guide to boosted regression trees," *Journal of Animal Ecology*, vol. 77, no. 4, pp. 802–813, 2008.

H. Fanaee-T and J. Gama, "Event labeling combining ensemble detectors and background knowledge," *Progress in Artificial Intelligence*, vol. 2, no. 2-3, pp. 113–127, 2014.

J. Faraway and C. Chatfield, "Time series forecasting with neural networks: a comparative study using the air line data," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 47, no. 2, pp. 231–250, 1998.

T. Fawcett and F. Provost, "Adaptive fraud detection," *Data mining and knowledge discovery*, vol. 1, no. 3, pp. 291–316, 1997.

——, "Activity monitoring: Noticing interesting changes in behavior," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 53–62.

C. Ferreira, J. Gama, L. Matias, A. Botterud, and J. Wang, "A survey on wind power ramp forecasting." Argonne National Lab.(ANL), Argonne, IL (United States), Tech. Rep., 2011.

M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in neural information processing systems*, 2015, pp. 2962–2970.

R. Fildes, "Evaluation of aggregate and individual forecast method selection rules," *Management Science*, vol. 35, no. 9, pp. 1056–1065, 1989.

P. Flach, "Performance evaluation in machine learning: The good, the bad, the ugly and the way forward," in *33rd AAAI Conference on Artificial Intelligence*, 2019.

A. R. M. Forkan, I. Khalil, and M. Atiquzzaman, "Visibid: A learning model for early discovery and real-time prediction of severe clinical events using vital signs as big data," *Computer Networks*, vol. 113, pp. 244–257, 2017.

Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, pp. 1–22, 2010.

J. H. Friedman, "A variable span smoother," Stanford Univ CA Lab for Computational Statistics, Tech. Rep., 1984.

J. H. Friedman and W. Stuetzle, "Projection pursuit regression," *Journal of the American statistical Association*, vol. 76, no. 376, pp. 817–823, 1981.

J. H. Friedman *et al.*, "Multivariate adaptive regression splines," *The annals of statistics*, vol. 19, no. 1, pp. 1–67, 1991.

P. Gaillard and Y. Goude, "Forecasting electricity consumption by aggregating experts; how to design a good set of experts," in *Modeling and Stochastic Learning for Forecasting in High Dimensions*. Springer, 2015, pp. 95–115.

——, *opera: Online Prediction by Expert Aggregation*, 2016, r package version 1.0. [Online]. Available: https://CRAN.R-project.org/package=opera

J. Gama, *Knowledge discovery from data streams*. Chapman and Hall/CRC, 2010.

J. Gama and P. Kosina, "Tracking recurring concepts with meta-learners," in *Portuguese Conference on Artificial Intelligence*. Springer, 2009, pp. 423–434.

——, "Recurrent concepts in data streams classification," *Knowledge and Information Systems*, vol. 40, no. 3, pp. 489–507, 2014.

J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine learning*, vol. 90, no. 3, pp. 317–346, 2013.

J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

E. S. Gardner Jr, "Exponential smoothing: The state of the art," *Journal of forecasting*, vol. 4, no. 1, pp. 1–28, 1985.

S. Geisser, "The predictive sample reuse method with applications," *Journal of the American statistical Association*, vol. 70, no. 350, pp. 320–328, 1975.

P. Geladi and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Analytica chimica acta*, vol. 185, pp. 1–17, 1986.

S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

V. Genre, G. Kenny, A. Meyler, and A. Timmermann, "Combining expert forecasts: Can anything beat the simple average?" *International Journal of Forecasting*, vol. 29, no. 1, pp. 108–121, 2013.

M. F. Ghalwash, V. Radosavljevic, and Z. Obradovic, "Extraction of interpretable multivariate patterns for early diagnostics," in *2013 IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 201–210.

S. Ghosh, M. Feng, H. Nguyen, and J. Li, "Hypotension risk prediction via sequential contrast patterns of icu blood pressure," *IEEE journal of biomedical and health informatics*, vol. 20, no. 5, pp. 1416–1426, 2016.

T. Gneiting and M. Katzfuss, "Probabilistic forecasting," *Annual Review of Statistics and Its Application*, vol. 1, pp. 125–151, 2014.

B. Graham and J. Zweig, *The intelligent investor*. HarperBusiness Essentials New York, USA, 2003.

P. J. Green and B. W. Silverman, *Nonparametric regression and generalized linear models: a roughness penalty approach*. CRC Press, 1993.

M. P. Griffin and J. R. Moorman, "Toward the early diagnosis of neonatal sepsis and sepsis-like illness using novel heart rate analysis," *Pediatrics*, vol. 107, no. 1, pp. 97–104, 2001.

J. D. Hart and T. E. Wehrly, "Kernel regression estimation using repeated measurements data," *Journal of the American Statistical Association*, vol. 81, no. 396, pp. 1080–1088, 1986.

G. He, Y. Duan, R. Peng, X. Jing, T. Qian, and L. Wang, "Early classification on multivariate time series," *Neurocomputing*, vol. 149, pp. 777–787, 2015.

H. He and Y. Ma, *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, 2013.

H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1322–1328.

M. Herbster and M. K. Warmuth, "Tracking the best expert," *Machine Learning*, vol. 32, no. 2, pp. 151–178, 1998.

A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

R. Hyndman, "Time series data library," http://data.is/TSDLdemo, 2019, accessed: 2017-12-11.

R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

R. J. Hyndman, with contributions from George Athanasopoulos, S. Razbash, D. Schmidt, Z. Zhou, Y. Khan, C. Bergmeir, and E. Wang, *forecast: Forecasting functions for time series and linear models*, 2014, R package version 5.6.

R. Jacobs, "Methods for combining experts' probability assessments," *Neural computation*, vol. 7, no. 5, pp. 867–888, 1995.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

L. A. Jentgen, H. Kidder, R. Hill, S. Conrad, and A. Papalexopaulos, "Water consumption forecasting to improve energy efficiency of pumping operations," *AwwaRF, Denver*, 2007.

M. Jiang, Y. Liang, X. Feng, X. Fan, Z. Pei, Y. Xue, and R. Guan, "Text classification based on deep belief network and softmax regression," *Neural Computing and Applications*, vol. 29, no. 1, pp. 61–70, 2018.

I. T. Jolliffe, "A note on the use of principal components in regression," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 31, no. 3, pp. 300–303, 1982.

V. R. R. Jose and R. L. Winkler, "Simple robust averages of forecasts: Some empirical results," *International Journal of Forecasting*, vol. 24, no. 1, pp. 163–169, 2008.

A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis, "kernlab – an S4 package for kernel methods in R," *Journal of Statistical Software*, vol. 11, no. 9, pp. 1–20, 2004.

M. B. Kennel, R. Brown, and H. D. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Physical review A*, vol. 45, no. 6, p. 3403, 1992.

L. Kilian and M. P. Taylor, "Why is it so difficult to beat the random walk forecast of exchange rates?" *Journal of International Economics*, vol. 60, no. 1, pp. 85–107, 2003.

M. Koppel and S. P. Engelson, "Integrating multiple classifiers by finding their areas of expertise," in *AAAI-96 Workshop On Integrating Multiple Learning Models*. Citeseer, 1996, pp. 53–58.

I. Koprinska, M. Rana, and V. G. Agelidis, "Yearly and seasonal models for electricity load forecasting," in *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011, pp. 1474–1481.

B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Information Fusion*, vol. 37, pp. 132–156, 2017.

H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2556–2563.

M. Kuhn, S. Weston, C. Keefer, and N. C. C. code for Cubist by Ross Quinlan, *Cubist: Rule- and Instance-Based Regression Modeling*, 2014, R package version 0.0.18.

L. I. Kuncheva, *Multiple Classifier Systems: 5th International Workshop, MCS 2004, Cagliari, Italy, June 9-11, 2004. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, ch. Classifier Ensembles for Changing Environments, pp. 1–15.

——, "Classifier ensembles for changing environments," in *International Workshop on Multiple Classifier Systems*. Springer, 2004, pp. 1–15.

D. Kwiatkowski, P. C. Phillips, P. Schmidt, and Y. Shin, "Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?" *Journal of econometrics*, vol. 54, no. 1-3, pp. 159–178, 1992.

W. B. Langdon, S. Barrett, and B. F. Buxton, "Combining decision trees and neural networks for drug discovery," in *European Conference on Genetic Programming*. Springer, 2002, pp. 60–70.

J. Lee and R. Mark, "A hypotensive episode predictor for intensive care based on heart rate and blood pressure time series," in *Computing in Cardiology, 2010*. IEEE, 2010, pp. 81–84.

J. Lee and R. G. Mark, "An investigation of patterns in hemodynamic data indicative of impending hypotension in intensive care," *Biomedical engineering online*, vol. 9, no. 1, p. 62, 2010.

M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: archive.ics.uci.edu/ml

N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," *Information and computation*, vol. 108, no. 2, pp. 212–261, 1994.

F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, p. 3, 2012.

X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539–550, 2009.

G. M. Ljung and G. E. Box, "On a measure of lack of fit in time series models," *Biometrika*, vol. 65, no. 2, pp. 297–303, 1978.

K. Lukoseviciute and M. Ragulskis, "Evolutionary algorithms for the selection of time lags for time series forecasting by fuzzy inference systems," *Neurocomputing*, vol. 73, no. 10-12, pp. 2077–2088, 2010.

S. Makridakis, A. Andersen, R. Carbone, R. Fildes, M. Hibon, R. Lewandowski, J. Newton, E. Parzen, and R. Winkler, "The accuracy of extrapolation (time series) methods: Results of a forecasting competition," *Journal of forecasting*, vol. 1, no. 2, pp. 111–153, 1982.

M. Marcellino, "Forecast pooling for european macroeconomic variables," *Oxford Bulletin of Economics and Statistics*, vol. 66, no. 1, pp. 91–112, 2004.

S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.

C. Maxey and A. Andreas, "Oak ridge national laboratory (ornl); rotating shadowband radiometer (rsr); oak ridge, tennessee (data)," National Renewable Energy Laboratory (NREL), Golden, CO (United States), Tech. Rep., 2007.

P. McCullagh, *Generalized linear models*. Routledge, 2019.

A. D. McQuarrie and C.-L. Tsai, *Regression and time series model selection*. World Scientific, 1998.

J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, p. 10, 2012.

B.-H. Mevik, R. Wehrens, and K. H. Liland, *pls: Partial Least Squares and Principal Component Regression*, 2016, r package version 2.6-0. [Online]. Available: https://CRAN.R-project.org/package=pls

R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

S. Milborrow, *earth: Multivariate Adaptive Regression Spline Models*, 2012.

——, *earth: Multivariate Adaptive Regression Splines*, 2016, R package version 4.4.4.

——, *rpart.plot: Plot 'rpart' Models: An Enhanced Version of 'plot.rpart'*, 2018, r package version 3.0.6. [Online]. Available: https://CRAN.R-project.org/package=rpart.plot

D. S. Modha and E. Masry, "Prequential and cross-validated regression estimation," *Machine Learning*, vol. 33, no. 1, pp. 5–39, 1998.

N. Moniz, P. Branco, and L. Torgo, "Resampling strategies for imbalanced time series forecasting," *International Journal of Data Science and Analytics*, vol. 3, no. 3, pp. 161–181, 2017.

G. B. Moody and L.-w. H. Lehman, "Predicting acute hypotensive episodes: The 10th annual physionet/computers in cardiology challenge," *Computers in Cardiology*, vol. 36, no. 5445351, p. 541, 2009.

L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

I. Mozetič, L. Torgo, V. Cerqueira, and J. Smailović, "How to evaluate sentiment classifiers for Twitter time-ordered data?" *PLoS ONE*, vol. 13, no. 3, p. e0194317, 2018.

J. W. Murry Jr and J. O. Hammons, "Delphi: A versatile methodology for conducting qualitative research," *The Review of Higher Education*, vol. 18, no. 4, pp. 423–436, 1995.

G. Nason, "A test for second-order stationarity and approximate confidence intervals for localized autocovariances for locally stationary time series," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 75, no. 5, pp. 879–904, 2013.

P. Newbold and C. W. Granger, "Experience with forecasting univariate time series and the combination of forecasts," *Journal of the Royal Statistical Society. Series A (General)*, pp. 131–165, 1974.

M. Oliveira and L. Torgo, "Ensembles for time series forecasting," in *ACML Proceedings of Asian Conference on Machine Learning. JMLR: Workshop and Conference Proceedings*, 2014.

M. Oliveira, L. Torgo, and V. S. Costa, "Evaluation procedures for forecasting with spatio-temporal data," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 703–718.

J. Ortega, M. Koppel, and S. Argamon, "Arbitrating among competing classifiers using learned referees," *Knowledge and Information Systems*, vol. 3, no. 4, pp. 470–490, 2001.

——, "Arbitrating among competing classifiers using learned referees," *Knowledge and Information Systems*, vol. 3, no. 4, pp. 470–490, 2001.

E. Parras-Gutierrez, V. M. Rivas, M. Garcia-Arenas, and M. Del Jesus, "Short, medium and long term forecasting of time series using the l-co-r algorithm," *Neurocomputing*, vol. 128, pp. 433–446, 2014.

D. B. Percival and A. T. Walden, *Wavelet methods for time series analysis*. Cambridge university press, 2006, vol. 4.

B. Pfahringer, "Winning the kdd99 classification cup: bagged boosting," *ACM SIGKDD Explorations Newsletter*, vol. 1, no. 2, pp. 65–66, 2000.

P. C. Phillips and P. Perron, "Testing for a unit root in time series regression," *Biometrika*, vol. 75, no. 2, pp. 335–346, 1988.

F. J. Provost, T. Fawcett *et al.*, "Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions." in *KDD*, vol. 97, 1997, pp. 43–48.

J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

——, "Combining instance-based and model-based learning," in *Proceedings of the tenth international conference on machine learning*, 1993, pp. 236–243.

R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2013.

J. Racine, "Consistent cross-validatory model-selection for dependent data: hv-block cross-validation," *Journal of econometrics*, vol. 99, no. 1, pp. 39–61, 2000.

C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 1135–1144.

R. P. Ribeiro, P. Pereira, and J. Gama, "Sequential anomalies: a study in the railway industry," *Machine Learning*, vol. 105, no. 1, pp. 127–153, 2016.

G. Ridgeway, *gbm: Generalized Boosted Regression Models*, 2015, R package version 2.1.1.

T. Rocha, S. Paredes, P. De Carvalho, and J. Henriques, "Prediction of acute hypotensive episodes by means of neural network multi-models," *Computers in biology and medicine*, vol. 41, no. 10, pp. 881–890, 2011.

A. L. D. Rossi, A. C. P. de Leon Ferreira, C. Soares, B. F. De Souza *et al.*, "Metastream: A meta-learning based method for periodic algorithm selection in time-changing data," *Neurocomputing*, vol. 127, pp. 52–64, 2014.

S. Roychoudhury, M. F. Ghalwash, and Z. Obradovic, "False alarm suppression in early prediction of cardiac arrhythmia," in *2015 IEEE 15th International Conference on Bioinformatics and Bioengineering (BIBE)*. IEEE, 2015, pp. 1–6.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

M. Saeed, C. Lieu, G. Raber, and R. G. Mark, "Mimic ii: a massive temporal icu patient database to support research in intelligent patient monitoring," in *Computers in Cardiology, 2002.* IEEE, 2002, pp. 641–644.

S. Salvador, P. Chan, and J. Brodie, "Learning states and rules for time series anomaly detection." in *FLAIRS conference*, 2004, pp. 306–311.

I. Sánchez, "Adaptive combination of forecasts with application to wind energy," *International Journal of Forecasting*, vol. 24, no. 4, pp. 679–693, 2008.

J. C. Schlimmer and R. H. Granger, Jr., "Incremental learning from noisy data," *Mach. Learn.*, vol. 1, no. 3, pp. 317–354, Mar. 1986.

B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.

T. A. Snijders, "On cross-validation for predictor evaluation in time series," in *On Model Uncertainty and its Statistical Implications.* Springer, 1988, pp. 56–69.

M. Stone, "Cross-validation and multinomial prediction," *Biometrika*, pp. 509–515, 1974.

P. Stone and M. Veloso, "Layered learning," in *European Conference on Machine Learning.* Springer, 2000, pp. 369–381.

S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa, "A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition," *Expert systems with applications*, vol. 39, no. 8, pp. 7067–7083, 2012.

F. Takens, *Dynamical Systems and Turbulence, Warwick 1980: Proceedings of a Symposium Held at the University of Warwick 1979/80.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, ch. Detecting strange attractors in turbulence, pp. 366–381.

P. Tamilselvan and P. Wang, "Failure diagnosis using deep belief learning based health state classification," *Reliability Engineering & System Safety*, vol. 115, pp. 124–135, 2013.

L. J. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International journal of forecasting*, vol. 16, no. 4, pp. 437–450, 2000.

S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: Combined selection and hyperparameter optimization of classification algorithms," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 847–855.

R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

A. Timmermann, "Forecast combinations," *Handbook of economic forecasting*, vol. 1, pp. 135–196, 2006.

——, "Elusive return predictability," *International Journal of Forecasting*, vol. 24, no. 1, pp. 1–18, 2008.

L. Todorovski and S. Džeroski, "Combining classifiers with meta decision trees," *Machine learning*, vol. 50, no. 3, pp. 223–249, 2003.

I. Tomek, "Two modifications of cnn," *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 769–772, 1976.

L. Torgo and O. Ohashi, "2d-interval predictions for time series," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 787–794.

J. F. Torres, A. Troncoso, I. Koprinska, Z. Wang, and F. Martínez-Álvarez, "Deep learning for big data time series forecasting applied to solar power," in *The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications*. Springer, 2018, pp. 123–133.

D. Trigg, "Monitoring a forecasting system," *Journal of the Operational Research Society*, vol. 15, no. 3, pp. 271–274, 1964.

E. Tsur, M. Last, V. F. Garcia, R. Udassin, M. Klein, and E. Brotfain, "Hypotensive episode prediction in icus via observation window splitting," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 472–487.

N. Ueda and R. Nakano, "Generalization error of ensemble estimators," in *Neural Networks, 1996., IEEE International Conference on*, vol. 1. IEEE, 1996, pp. 90–95.

J. N. van Rijn, G. Holmes, B. Pfahringer, and J. Vanschoren, "Having a blast: Meta-learning and heterogeneous ensembles for data streams," in *Data Mining (ICDM), 2015 IEEE International Conference on*, Nov 2015, pp. 1003–1008.

——, "The online performance estimation framework: heterogeneous ensemble learning for data streams," *Machine Learning*, pp. 1–28, 2018.

W. N. Venables and B. D. Ripley, *Modern Applied Statistics with S*, 4th ed. New York: Springer, 2002, iSBN 0-387-95457-0.

R. Vilalta and S. Ma, "Predicting rare events in temporal domains," in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* IEEE, 2002, pp. 474–481.

P. Vlachos, "Statlib project repository," *Carnegie Mellon University*, 2000.

V. G. Vovk, "Aggregating strategies," *Proc. of Computational Learning Theory, 1990*, 1990.

C. Voyant, G. Notton, S. Kalogirou, M.-L. Nivet, C. Paoli, F. Motte, and A. Fouilloy, "Machine learning methods for solar radiation forecasting: A review," *Renewable Energy*, vol. 105, pp. 569–582, 2017.

S. Wager, T. Hastie, and B. Efron, "Confidence intervals for random forests: The jackknife and the infinitesimal jackknife," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1625–1651, 2014.

A. Wald, *Sequential analysis*. Courier Corporation, 1973.

D. Walters and W. Wilkinson, "Wireless fraud, now and in the future: A view of the problem and some solutions," *Mobile Phone News*, vol. 24, pp. 4–7, 1994.

W. Wang, P. Jones, and D. Partridge, "Diversity between neural networks and decision trees for building multiple classifier systems," in *International Workshop on Multiple Classifier Systems*. Springer, 2000, pp. 240–249.

X. Wang, K. Smith-Miles, and R. Hyndman, "Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series," *Neurocomputing*, vol. 72, no. 10, pp. 2581–2594, 2009.

Z. Wang and I. Koprinska, "Solar power forecasting using dynamic meta-learning ensemble of neural networks," in *International Conference on Artificial Neural Networks*. Springer, 2018, pp. 528–537.

G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," *Machine learning*, vol. 40, no. 2, pp. 159–196, 2000.

G. I. Webb and Z. Zheng, "Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 8, pp. 980–991, 2004.

A. S. Weigend, *Time series prediction: forecasting the future and understanding the past*. Routledge, 2018.

G. M. Weiss and H. Hirsh, "Learning to predict rare events in event sequences." in *KDD*, 1998, pp. 359–363.

J. Wnek and R. S. Michalski, "Hypothesis-driven constructive induction in aq17-hci: A method and experiments," *Machine Learning*, vol. 14, no. 2, pp. 139–168, 1994.

D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

——, "The lack of a priori distinctions between learning algorithms," *Neural computation*, vol. 8, no. 7, pp. 1341–1390, 1996.

——, "The supervised learning no-free-lunch theorems," in *Soft computing and industry*. Springer, 2002, pp. 25–42.

K. Woods, W. P. Kegelmeyer, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE transactions on pattern analysis and machine intelligence*, vol. 19, no. 4, pp. 405–410, 1997.

M. N. Wright, *ranger: A Fast Implementation of Random Forests*, 2015, R package.

J.-X. Wu, Z.-H. Zhou, Z.-Q. Chen, J. xin Wu, Z. hua Zhou, and Z. qian Chen, "Ensemble of ga based selective neural network ensembles," 2001.

Z. Xing, J. Pei, P. S. Yu, and K. Wang, "Extracting interpretable features for early classification on time series," in *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM, 2011, pp. 247–258.

K. Yamanishi and J.-i. Takeuchi, "A unifying framework for detecting outliers and change points from non-stationary time series data," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 676–681.

Y. Yu, Z.-H. Zhou, and K. M. Ting, "Cocktail ensemble for regression," in *7th IEEE Int. Conference on Data Mining*, ser. ICDM'2007. IEEE, 2007, pp. 721–726.

G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks:: The state of the art," *International journal of forecasting*, vol. 14, no. 1, pp. 35–62, 1998.

Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC press, 2012.

M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 2003, pp. 928–936.