Universidade Federal do Rio Grande do Sul
Instituto de Matemática e Estatística
Programa de Pós-Graduação em Estatística

mFFORMS: multi-level Feature-based FORecast Model Selection

Bruno Grillo de Bermúdez

Porto Alegre, maio de 2023.

Dissertação submetida por Bruno Grillo de Bermúdez como requisito parcial para a obtenção do título de Mestre em Estatística pelo Programa de Pós-Graduação em Estatística da Universidade Federal do Rio Grande do Sul.

Orientador:

Dr. Flávio A. Ziegelmann (PPGEst - UFRGS)

Comissão Examinadora:

Dr. Hudson da Silva Torrent (DEST - UFRGS)

Dr. Taiane Schaedler Prass (PPGEst - UFRGS)

Dr. Diego Nascimento (UDA - Chile)

Data de Apresentação: 06 de junho de 2023

Agradecimentos

Agradeço:

- à minha mãe Edite, ao meu pai Gabriel e à minha irmã Camila pelo amor, carinho e apoio que sempre me deram.
- Ao meu orientador, Dr. Flávio Ziegelmann, pelo empenho e sugestões para que essa dissertação fosse concluída.
- Aos membros da banca, pela disponibilidade e sugestões.
- A todos membros do PPGEst pela dedicação ao programa e pelo auxílio provido aos discentes.

Abstract

Montero-Manso et al. [2020] proposed a meta-learning combination method that outputs weights for each candidate forecast method given the time series features, named FFORMA. It reached the second overall place in the M4 competition, a contest that received a special edition at the International Journal of Forecasting. The competition's relevance stems from the vast number of diverse time series (100,000) that reflect the characteristics of the time series often encountered by researchers and practitioners in their challenges. Inspired by recent developments in forecast combination literature, we propose mFFORMS, a meta-learner to select a forecast combination method of time series models based on time series features. This novel approach consists of two phases. First, we use a collection of time series to train a meta-learner that selects the forecasting combination method that minimises an error measure. Each combination method uses cross-validated data to estimate the weights of each forecasting model in the pool. The inputs for the meta-learner are the time series features, and the response is the label of the combination method that produces the lowest error. In the second phase, we use the previously trained meta-learner to select the combination method given the time series features. Afterwards, we employ the selected combination method to assign weights to the forecasts produced by the forecasting methods. In our comparison, we consider the same pool of forecasting models and the same information available throughout the competition. In the M4 settings, our approach provides better results than the candidate combination methods; however, it underperforms FFORMA and is more intensive computationally

Resumo

Montero-Manso et al. [2020] propôs um método de combinação de meta-aprendizagem, denominado FFORMA, que fornece pesos para cada método de previsão candidato, dado as características da série temporal. Esse método obteve o segundo lugar geral na competição M4, concurso que contou com edição especial no International Journal of Forecasting. A relevância da competição M4 advém do grande número de séries temporais (100.000) com características variadas que são encontradas por pesquisadores e profissionais em seus desafios. Inspirados por desenvolvimentos recentes na literatura de combinação de previsão, propômos o mFFORMS, um *meta-learner* que seleciona um método de combinação de previsão de modelos de séries temporais com base em características de séries temporais. Esta abordagem consiste em duas fases. Na primeira, usamos uma coleção de séries temporais para treinar um *meta-learner* que seleciona o método de combinação de previsão que minimiza uma medida de erro. Cada método de combinação usa dados de validação cruzada para estimar os pesos de cada modelo de previsão presentes no pool de candidatos. As entradas para o *meta-learner* são as características da série temporal e a resposta é o rótulo do método de combinação que produz o menor erro esperado. Na segunda fase, usamos o *meta-learner* previamente treinado para selecionar o método de combinação dado as características da série temporal. Em seguida, empregamos o método de combinação selecionado para atribuir pesos às previsões produzidas pelos métodos de previsão. Em nossa comparação, consideramos o mesmo conjunto de modelos de previsão e as mesmas informações disponíveis ao longo da competição. Nas configurações M4, nossa abordagem fornece melhores resultados do que os métodos de combinação candidatos; no entanto, tem desempenho inferior ao FFORMA e é mais intensivo computacionalmente.

Índice

1 Introdução **1**

2 Contexto e Objetivo **2**

3 Resumo da Abordagem **4**

4 Artigo **6**

5 Conclusão **29**

1 Introdução

Em seu artigo seminal, Bates and Granger [1969] introduziram um método para combinar previsões de diferentes modelos estatísticos com o intuito de reduzir o erro de previsão. O método consiste em estimar pesos para as previsões dos distintos modelos. A ideia subjacente é atribuir maiores pesos aos modelos cujos erros apresentam menor variância. Desde a publicação desse artigo, um vasto corpo de pesquisa teórica e empírica deu suporte à afirmação de que as previsões geradas por combinação de *forecast* geralmente apresentam erros menores do que as dos modelos individualmente. Essa redução de erros pode ser atribuída a diferentes aspectos dos modelos candidatos, tais como a forma que as informações/variáveis são utilizadas pelos modelos, os distintos mas desconhecidos viéses de má-especificação e o comportamento na presença de quebras estruturais. Clemen [1989] e Timmermann [2006] fornecem uma extensa revisão da literatura anterior, enquanto Wang et al. [2022] fornece uma revisão atualizada.

Em sua revisão da literatura inicial, Clemen [1989] fez duas perguntas: "Por que a média simples funciona tão bem¿' e "Sob quais condições outros métodos específicos funcionam melhor¿'. Stock and Watson [2004] cunhou o termo *Forecast Combination Puzzle* para se referir à observação em vários estudos empíricos de que a combinação de pesos iguais (média simples) das previsões candidatas geralmente supera métodos mais complexos e sofisticados, apesar de sua sub-otimalidade teórica, visto que ela é teoricamente ótima apenas quando as variâncias dos resíduos de todos os métodos candidatos são iguais.

Alguns estudos demonstram o efeito de erros de estimativa de pesos ótimos. Smith and Wallis [2009] demonstrou que a alta variância dos estimadores dos pesos devido a amostras pequenas resulta em desvios maiores em relação ao ótimo do que a média simples. Timmermann [2006] indica situações em que o ganho dos pesos ótimos é pequeno em relação à combinação de pesos iguais devido à proximidade das estimativas. Outra explicação para o desempenho instável da estimativa de peso ótimo é a presença de quebras estruturais no processo de geração de dados e especificação incorreta do modelo. Hendry and Clements [2004] demonstram que pesos iguais geralmente superam uma estratégia de estimativa de pesos ótimos sob quebras estruturais e especificações incorretas do modelo, pois os dados históricos não dão suporte à estimativa do conjunto de pesos ótimos.

Para explicar o *Forecast Combination Puzzle*, Yang [2004] sugere a existência de dois cenários. O primeiro é a combinação para adaptação (CFA, na sigla em inglês), onde um dos candidatos é o melhor ou porque captura o processo gerador de dados ou porque os outros candidatos adicionam apenas informações redundantes. Nesse cenário, o objetivo de um método de combinação de previsões deve ser imitar o desempenho da melhor previsão individual, que é desconhecida. Isso é análogo a atribuir peso um ao melhor modelo e zero aos demais candidatos. O segundo é a combinação para melhoria (CFI, na sigla em inglês), onde nenhum método é individualmente melhor e há, portanto, um potencial ganho ao combinar as previsões. Nesse cenário, o objetivo de um método de combinação de previsão deve ser encontrar os pesos ótimos.

Partindos dos cenários sugeridos por Yang [2004], Qian et al. [2019] propõem uma estratégia de combinação adaptativa que funciona bem em ambos cenários. A estratégia consiste em duas etapas, na primeira, emprega-se diferentes métodos de combinação de previsão para gerar novos candidatos. Esses novos candidatos são passados para a segunda etapa, que consiste em atribuir peso um a algum dos candidatos e zero aos demais. No artigo, Qian et al. [2019] consideram os seguintes métodos de combinação para a primeira etapa: (a) pesos iguais, (b) regressão linear Granger and Ramanathan [1984] e (c) AFTER Yang [2004], um método de seleção adequado para o cenário CFA. Na segunda etapa, eles usam AFTER novamente nas previsões combinadas em vez de usar nas previsões originais, por esse motivo o nome AFTER multinível. Nas simulações geradas, eles descobriram que o método pode se adaptar ao cenário subjacente contornando o *Forecast Combination Puzzle*.

Recentemente, métodos de combinação de previsão que partem da ideia de meta-aprendizado baseado em características de séries temporais ganharam atenção devido ao excelente desempenho na competição de séries temporais M4, uma disputa que recebeu edição especial no International Journal of Forecasting, e é de grande importância, pois traz um número expressivo de séries temporais (100.000) com características variadas. O FFORMA (*Feature-based FORecast-Model Averaging*) proposto por Montero-Manso et al. [2020] combina previsões empregando uma abordagem de meta-aprendizagem baseada em características de séries temporais. Esse método alcançou o segundo lugar geral na competição M4 para estimativas pontuais e intervalos de previsão. O FFORMA é inspirado no FFORMS (*Feature-based FORecast-Model Selection*) do Talagala et al. [2018]. A distinção conceitual entre FFORMS e FFORMA reside em seus respectivos *meta-learners*. Enquanto o FFORMS escolhe um modelo singular de um pool de candidatos, o FFORMA gera um conjunto de pesos que coletivamente somam um.

Esta dissertação propõe uma abordagem de combinação de previsões baseada em meta-aprendizado para lidar com uma quantidade extensa de séries temporais. Inspirados na proposta de Qian et al. [2019], avaliamos se um FFORMS multinível supera o FFORMA nas configurações da competição M4. Chamamos de mFFORMS, ou FFORMS multinível, porque nós, assim como Qian et al. [2019], usamos um método de seleção para selecionar previsões geradas por combinações de previsões, com a diferença de que o FFORMS não seleciona um modelo de previsão na primeira etapa.

O restante deste trabalho está organizado da seguinte forma: na Seção 2, descreve-se o contexto e objetivos da dissertação, na Seção 3 descreve-se a abordagem proposta através de um fluxograma, na Seção 4 está o artigo da dissertação e, finalmente, na Seção 5, a conclusão.

2 Contexto e Objetivo

O contexto parte de qualquer profissional que enfrenta o desafio de prever milhares de séries temporais recorrentemente e em intervalos curtos. As séries encontradas apresentam comprimentos variados devido a diversos fatores, entre eles estão, por exemplo, o início da coleta de dados, mudança na política comercial, e as diferentes datas de lançamento de produtos. Algumas séries podem apresentar processos geradores de dados com diferentes características, como sa-

Tabela 1: Quantidade Séries M4

period	Demographic	Finance	Industry	Macro	Micro	Other	Total
Daily	10	1,559	422	127	1,476	633	4,227
Monthly	5,728	10,987	10,017	10,016	10,975	277	48,000
Quarterly	1,858	5,305	4,637	5,315	6,020	865	24,000
Weekly	24	164	6	41	112	12	359
Yearly	1,088	6,519	3,716	3,903	6,538	1,236	23,000
Hourly	0	0	0	0	0	414	414
Total	8,708	24,534	18,798	19,402	25,121	3,437	100,000

zonalidade, tendências e ciclos. É improvável que um único modelo de previsão lide com séries temporais que exibem padrões tão distintos. Uma alternativa é combinar as previsões de um conjunto de métodos de previsão de forma a lidar com as diversas características da série temporal.

No entanto, atribuir os pesos a cada método de previsão candidato para cada uma das milhares de séries temporais é uma tarefa complexa, especialmente para séries temporais curtas onde uma avaliação residual adequada é impossível. Uma solução possível para esse problema é empregar meta-aprendizagem para combinar previsões. A ideia-chave é aprender com outras séries que apresentaram características semelhantes no passado. O meta-aprendizado envolve o uso de algoritmos de aprendizado de máquina para aprender com dados históricos quais métodos de previsão são mais eficazes para séries com um determinado conjunto de características. O modelo de meta-aprendizagem pode identificar padrões e relações entre diferentes métodos de previsão e características de séries temporais ao analisar uma grande quantidade de dados históricos.

A competição M4 visa fornecer séries temporais que reflitam a realidade; por isso, emprega um conjunto de dados que consiste em uma amostra aleatória de 100.000 séries temporais extraídas do FoReDeck, um banco de dados disponibilizado pela National Technical University of Athens contendo cerca de 900.000 séries temporais contínuas de múltiplas fontes socioeconômicas de dados gerados por interações humanas. O conjunto de dados possui seis categorias principais: dados demográficos, financeiros, industriais, macroeconômicos, microeconômicos e outros tipos. Inclui dados de indústrias, serviços, turismo, importações e exportações, demografia, educação, trabalho e salários, governo, famílias, títulos, ações, seguros, empréstimos, imóveis, transporte, etc. O conjunto de dados tem seis frequências de tempo distintas: horária, diária, semanal, mensal, trimestral e anual. No entanto, a proporção em que aparecem não é uniforme; 95% das séries são de periodicidade mensal (48%), trimestral (24%) e anual (23%). Em relação às observações disponíveis, o teste de treinamento mais curto é 13 para anual, 16 para trimestral, 42 para mensal, 80 para semanal, 93 para diário e 700 para série horária. O número de passos previstos à frente também variou de acordo com a frequência da série. Os participantes da competição foram solicitados a prever seis próximas observações para a série temporal anual, oito para trimestral, 18 para mensal, 13 para semanal, 14 para diária e 48 para horária. Observa-se na Tabela 1 a quantidade de séries temporais presentes na competição de acordo com a periodicidade e origem. Na Tabela 2 estão algumas estatísticas da quantidade de observações das séries disponíveis.

Tabela 2: Quantidade Observações M4

Period	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Daily	93	323	2,940	2,357	4,197	9,919
Hourly	700	700	960	854	960	960
Monthly	42	82	202	216	306	2 794
Quarterly	16	62	88	92	115	866
Weekly	80	379	934	1,022	1,603	2,597
Yearly	13	20	29	31	40	835

O objetivo é desenvolver um *framework* de combinação de previsões altamente automatizado que consiga utilizar informações históricas para determinar padrões que auxiliem na decisão da maneira mais efetiva de prever cada uma das séries temporais distintas originadas de um grupo heterogêneo e de volume expressivo. O meta-aprendizado é um artifício importante para permitir essa previsão em larga escala. As entradas não ficam restritas às *features* de séries temporais, pois é possível inserir outras variáveis que possam trazer informação relevante (ex: categoria de produto). O método proposto é idêntico ao FFORMS, exceto na parte em que o alvo não é um método de previsão, mas um método de combinação de previsões. Inspirou-se em Qian et al. [2019] para ajustar o FFORMS para fazer seleção do método de combinação. Verifica-se seu desempenho em dados reais tanto usando os métodos de combinação de previsão individuais quanto comparando ao FFORMA em condições mais parecidas possíveis, ou seja, usando os mesmos dados, as mesmas informações, os mesmos métodos de previsão, o mesmo conjunto de *features* e considerando, de certa forma, a mesma função de perda.

3 Resumo da Abordagem

A implementação do mFFORMS é realizada em duas fases: (1) a fase offline, na qual o *meta-learner* é treinado para selecionar o melhor método de combinação de previsões e (2) a fase online, na qual o *meta-learner* pré-treinado é usado para identificar a melhor combinação de previsões para uma nova série com base em suas *features* de séries temporais. O fluxograma Figura 1 descreve os passos para prever usando o mFFORMS, que precisa das seguintes entradas: (1) as séries temporais **Y** a prever, (2) as funções **F** para calcular as *features*, (3) os métodos de previsão **M**, (4) os métodos de combinação **C** e (5) o conjunto de **Dados Referência**. Se a resposta para a pergunta 'É necessário treinar o *meta-learner*?' for sim, começa-se a fase offline do mFFORMS, caso a resposta seja não, pois já foi calculado anteriormente, inicia-se a fase online.

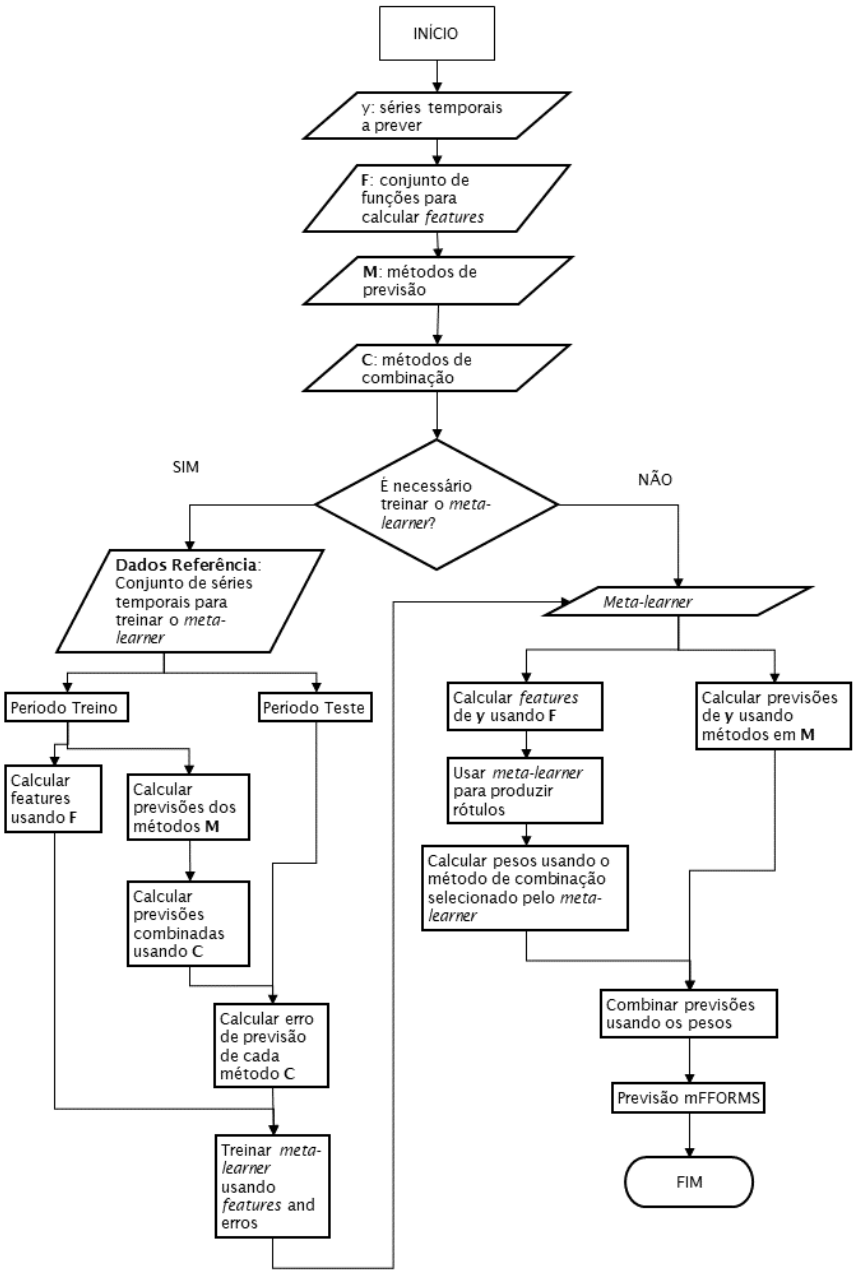

Figura 1: Fluxograma mFFORMS

4 Artigo

Autores: Bruno Grillo de Bermúdez, Flávio A. Ziegelmann

Título: mFFORMS: multi-level Feature-based FORecast Model Selection

Ano: 2023

mFFORMS: multi-level Feature-based FORecast Model Selection

de Bermúdez, B. G.[1], Ziegelmann, F.[1]

Abstract

Montero-Manso et al. (2020) proposed a meta-learning combination method that outputs weights for each candidate forecast method given the time series features, named FFORMA. It reached the second overall place in the M4 competition, a contest that received a special edition at the International Journal of Forecasting. The competition's relevance stems from the vast number of diverse time series (100,000) that reflect the characteristics of the time series often encountered by researchers and practitioners in their challenges. Inspired by recent developments in forecast combination literature, we propose mFFORMS, a meta-learner to select a forecast combination method of time series models based on time series features. This novel approach consists of two phases. First, we use a collection of time series to train a meta-learner that selects the forecasting combination method that minimises an error measure. Each combination method uses cross-validated data to estimate the weights of each forecasting model in the pool. The inputs for the meta-learner are the time series features, and the response is the label of the combination method that produces the lowest error. In the second phase, we use the previously trained meta-learner to select the combination method given the time series features. Afterwards, we employ the selected combination method to assign weights to the forecasts produced by the forecasting methods. In our comparison, we consider the same pool of forecasting models and the same information available throughout the competition. In the M4 settings, our approach provides better results than the candidate combination methods; however, it underperforms FFORMA and is more intensive computationally

Keywords: Forecast Combination, Time Series Features, Meta-learning, M4 Competition

1. Introduction

Forecasting practitioners are often faced with the challenge of frequently predicting thousands of time series in short intervals. The time series usually display varying lengths due to several factors, such as the beginning of the data collection and the different release dates of products. Some series may display data-generating processes with different characteristics, such as seasonality, trends, and irregular patterns. A single forecasting model is unlikely to deal with time series exhibiting such distinct patterns. An alternative is to combine the predictions of a pool of forecasting models to deal with the varied features of the time series.

In their seminal paper, Bates and Granger (1969) proposed a method to reduce prediction error by combining predictions from different models by estimating the weights of each candidate method. The underlying idea was to attribute greater weights to models whose errors had lower variance. Since their seminal paper, a vast body of theoretical and empirical research has supported the claim that combined predictions generally have lower errors than individual ones. This improvement can be attributed to different aspects of the candidate models, such as the form they use the information, the distinct but unknown misspecification bias, and the different behaviour in the presence of structural breaks. Clemen (1989) and Timmermann (2006) provide extensive earlier literature review while Wang et al. (2022) provide an up-to-date review.

In his review of early literature, Clemen (1989) posed two questions: "Why does the simple average work so well?" and "Under what conditions do other specific methods work better?". Stock and Watson (2004) coined the term Forecast Combination Puzzle to refer to the observation in several empirical studies that the equal weights combination of candidate forecasts often outperforms more complex and sophisticated

methods despite its theoretical sub-optimality. The simple average combination is theoretically optimal only when all forecast candidates' error variances are equal.

Some studies demonstrate the effect of optimal-weights estimation errors. Smith and Wallis (2009) demonstrated that the high variance of weight estimators due to a small sample causes the estimated weights to deviate more from the optimum than the simple average. They also conclude that their simulations, together with a large-sample approximation to the variance of the combining weight, support the recommendation to ignore forecast error covariances in estimating the weights. Timmermann (2006) indicates situations where the gain of the optimal weights is small relative to the combination of equal weights due to the proximity of the estimates.

Another explanation for the unstable performance of the optimal weight estimation is the presence of structural breaks in the data-generating process and model misspecification. Hendry and Clements (2004) demonstrate that equal weights usually outperform an optimal weight estimation strategy under structural breaks and model misspecification. Under these scenarios, the simple average generally outperforms strategies based on weight estimation, as historical data do not support estimating the set of optimal weights.

Yang (2004) considers two scenarios. The first is combining for adaptation (CFA), where one of the candidates is the best because it captures the actual data generating process or the other candidates add only redundant information. Under this scenario, the goal of a forecast combination method should be to target the performance of the best individual forecast, which is unknown. This is analogous to attributing weight one to the best model and zero to all other candidates. The second is combining for improvement (CFI), where no method is alone the best, so there is a potential significant accuracy gain over all the individual candidates. Under this scenario, the goal of a forecast combination method should be to find the optimal set of weights. Qian et al. (2019) suggest an adaptive combining strategy that performs well in both scenarios, named mAFTER. The strategy consists of two steps. First, it creates new candidates by combining the original forecasts with different methods. The second step selects one of the previously created combined forecasts. In their paper, Qian et al. (2019) considers the following combination methods: (a) equal weights, (b) linear regression Granger and Ramanathan (1984) and (c) AFTER Yang (2004), a selection method suited for the CFA scenario. In the second step, they use AFTER again on the combined forecasts instead of the original forecasts, hence the name, multi-level AFTER. In the simulation settings, they found that the method can adapt to the underlying scenario by circumventing the puzzle caused by improperly chosen combining methods. Under the CFI, it performs closer to the simple average when the degree of estimation error is high and closer to the optimal weight estimation when information is sufficient to estimate it.

In the M4 competition, Makridakis et al. (2020) observed that the candidates that employed combination methods had more accurate predictions than those who did not. Also, the top three methods utilised information from multiple time series to decide the most effective way of forecasting or selecting the weights for combining methods.

However, assigning the weights to each candidate forecasting method for each of the thousands of series is a complex task, especially for short time series where proper residual evaluation is impossible. One possible solution to this practitioner's problem is to employ meta-learning to combine predictions. The key idea is to learn from other series that displayed similar characteristics in the past. Meta-learning involves using machine learning algorithms to learn from historical data about which forecasting methods are most effective for series with a set of characteristics. The meta-learning model can identify patterns and relationships between different forecasting methods and time series characteristics by analysing a large amount of historical data.

Montero-Manso et al. (2020)'s FFORMA (Feature-based FORecast-Model Averaging) combines predictions using a meta-learning approach based on time series features. It is an 'averager' in the sense that the output is a set of weights for each candidate forecast model. It reached second place overall in the M4 competition for point estimates and prediction intervals. It is inspired by Talagala et al. (2018)'s FFORMS (Feature-based FORecast-Model Selection). The conceptual distinction between FFORMS and FFORMA lies in their respective meta-learners. Whereas FFORMS chooses a singular model from a candidate pool, FFORMA generates a set of weights that collectively sum to one. FFORMS is a times series forecast model selection based on meta-learning to identify the best forecast method using time series features as inputs.

It employs machine learning to estimate the meta-learner. It requires a large pool of time series whose features and labels of the 'best model' are already known. The new time series to be forecasted must be additional draws from the same population to avoid applying the classification rules to a set of time series whose features were not observed in the sample employed to calculate the meta-learner. It is useful when the number of series to predict is extensive.

Inspired by the proposal of Qian et al. (2019), we evaluate whether a two-step FFORMS can outperform FFORMA in the M4 settings. We call it mFFORMS, or multi-level FFORMS, because we, as Qian et al. (2019), use a selection method to select previously combined forecasts, with the difference that FFORMS does not select a forecasting model in the first step. The M4 competition is a rich playground since it provides 100,000 time series from different socio-economical interactions, described in subsection 3.1. The underlying mFFORMS idea is to create new forecast candidates by combining the candidate forecasts with different strategies that are suited for the CFA or the CFI scenario. For the CFA scenario, we use a naïve approach that selects the forecasting method that produces the lowest cross-validated mean squared error. For the CFI scenario, the candidates will be partially-egalitarian LASSO, or peLASSO, which has approximately equal weights on pooled methods, equal weights and LASSO (optimal weights that might work as a forecasting method selection).

The remainder of this work is organised as follows. In section 2, we describe the mFFORMS methodology providing a flowchart of the steps and choices involved; alternatively, we also offer a more detailed description through a pseudo-algorithm. In section 3, we provide details of the implementation in the M4 settings, containing a description of the problem space, the time series features, the candidate forecasting methods, the combination methods employed to generate new forecasts, the error metric and the mapping function (the meta-learner) to classify the labels according to the time series features. In section 4, we conduct an ex-ante evaluation of the candidate combination methods against mFFORMS. This evaluation considers only information available to the competitors throughout the competition, simulating the competition settings for mFFORMS. We also describe the selected meta-learner results, which produced lower errors than the candidate combination strategies alone, suggesting that the information used by the meta-learner helps reduce prediction errors by identifying the underlying scenario. In section 5, we compare the predictions generated by the mFFORMS with the ones submitted by FFORMA.

2. Methodology

One early proponent of the meta-learning idea was Rice (1976), which he called the algorithm selection problem (ASP) and whose four main components were the problem space, the feature space, the algorithm space and the performance metric. Talagala et al. (2018) employs the ASP framework to describe the forecasting selection problem.

Let P be the problem space that represents the available data set with a large collection of time series, F the time series features space that represents the measures that characterise the problem space P, A the algorithm space that contains a list of suitable candidate algorithms which can be used to find solutions to the problem in P, and L the metrics to evaluate the algorithm performance.

> **Algorithm selection problem**. For a given time series $y \in P$, with features $f(y) \in F$, find the selection mapping $S(f(y))$ into the algorithm space A, such that the selected algorithm $\alpha \in A$ minimises forecast accuracy error metric $l(\alpha(y)) \in L$ on the test set of the time series.

In our context, the algorithm space contains the candidate forecasting methods that are suited to predict the time series and the candidate forecast combination methods that assign weights to the predictions generated by each candidate forecasting method. In the recent M4 competition, a method that employs this framework achieved achieved second overall place. This framework, FFORMA, is inspired by Talagala et al. (2018)'s FFORMS. The conceptual distinction between FFORMS and FFORMA lies in how the meta-learner is specified. Whereas FFORMS chooses a singular model from a candidate pool, FFORMA generates a set of weights that collectively sum to one. The meta-learner estimation reflects this distinction.

FFORMS and FFORMA can employ the same pool of forecasting methods and time series features. They may utilise the same loss function. They both consist of two distinct phases: an offline phase and an online phase. In the offline phase, both require a reference set, a collection of time series divided into training and test period. The reference set must be representative of the new series to be predicted. For the training period, they calculate a set of time series features employed as inputs for the meta-learner, estimated by machine learning techniques. The observations in the training period are used to generate forecasts for the test period, and forecast errors are then computed.

In FFORMS, the forecasting method that produces the lowest error is the target for the meta-learner. The meta-learner estimation is a classification problem with defined classes. Among the candidate configurations of meta-learner, the one that achieves the lowest loss is selected. The configurations could be, for example, estimating the meta-learner with balanced and unbalanced classes.

In FFORMA, the target is the set of weights that minimises the expected loss. The meta-learning estimation is a problem of finding a function that assigns weights to each forecasting method to minimise the expected loss that would have been produced if the methods had been picked at random using these weights as probabilities.

In the online phase, the previously trained meta-learner is employed to generate the response. In FFORMS, it is the label of the selected forecasting model. In FFORMA, it is a set of weights for the candidate forecasting models. For each new series, the time series features are calculated and then passed to the meta-learner to output the class or set of weights that minimise the expected loss.

The mFFORMS is exactly like FFORMS, except that the target is not an individual forecasting model but a combination strategy that estimates weights to combine the predictions generated by the pool of forecasting methods. It is inspired by Qian et al. (2019)'s finding that a multi-level approach can deal with the different combination scenarios, CFA or CFI, without knowing the underlying scenario beforehand. In the following subsections, there is a detailed description of mFFORMS.

To evaluate whether this multi-level approach outperforms FFORMA, we aim to conduct the fairest comparison by employing the same forecasting methods and the time series features.

2.1. Algorithmic Description

The implementation of mFFORMS is conducted in two phases: (1) the offline phase, in which the meta-learner is trained to select the best combination method and (2) the online phase, in which the pre-trained meta-learner is used to identify the best combination for a new series based on its features. The following algorithms present the pseudo-code of the proposed combination selection. The algorithm 1 describes the steps for estimating the set of weights, while the algorithm 2 describes the mFFORMS approach, both online and offline phases.

In the offline phase, the weights for the series are calculated based on the cross-validation errors of each candidate forecasting method in the same series. This approach has some apparent drawbacks; it requires more observations than FFORMA since there is a cross-validation step for the the meta-learner and the estimation of the optimal set of weights. Also, it requires running the forecasting methods twice in the online phase unless the selected combination approach is the simple average.

A flowchart of the mFFORMS process can be seen in Figure 1. The online phase implementation was optimised to calculate only the essential steps to output the combined prediction to speed up the computations. For example, if the meta-learner suggests the best method is equal weights, estimating the weights with out-of-sample data is unnecessary, thus reducing the number of fitted forecasting methods. In cases where the combination method assigns weights to a few forecasting models, it only estimates these forecasts.

3. Implementation to M4 Settings

In this section we describe the components of the algorithm selection problem.

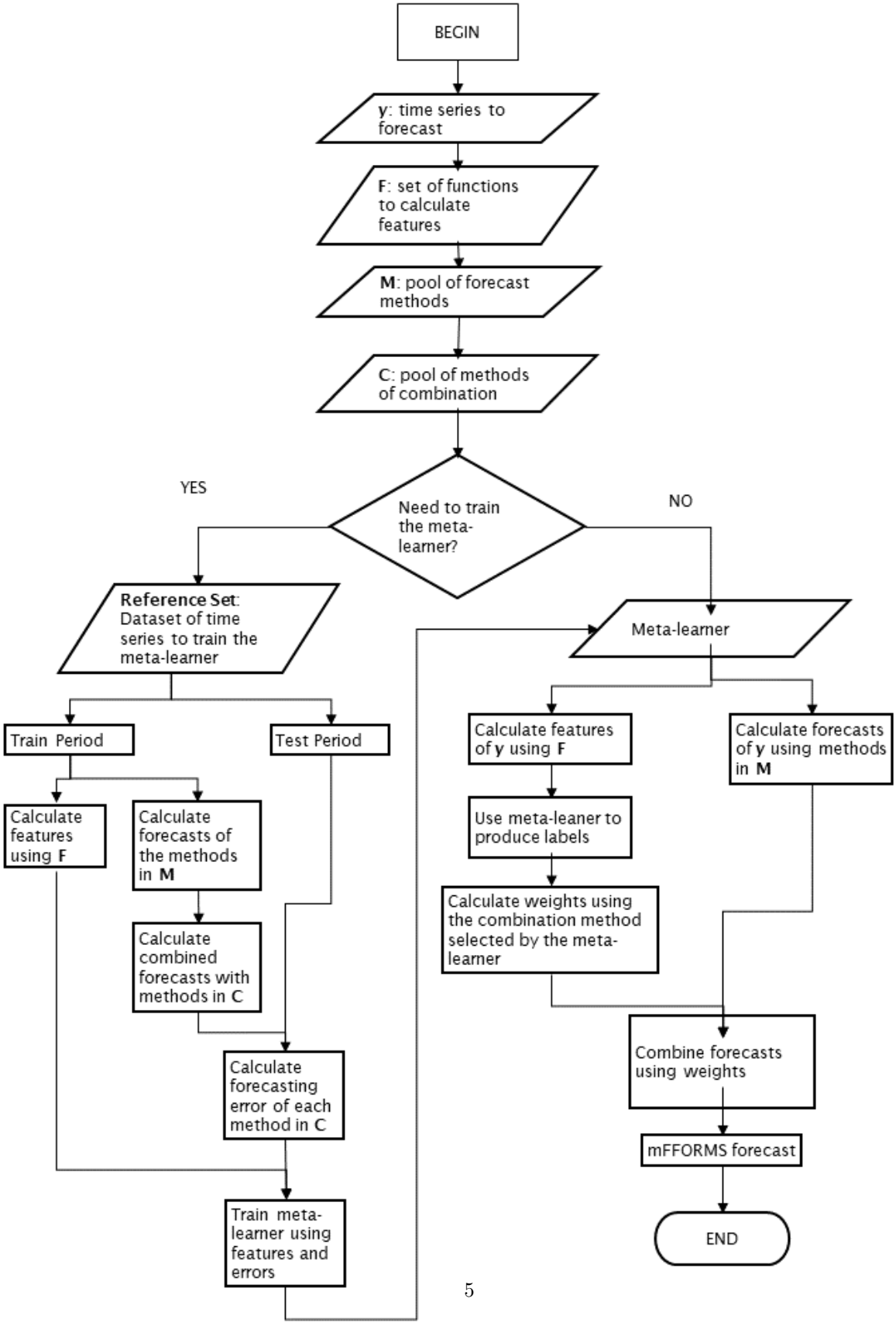

Figure 1: mFFORMS Flowchart

Algorithm 1 Estimate out-of-sample weights

ESTIMATING OPTIMAL WEIGHTS FOR COMBINATION

Inputs:

y: a time series.
h: the number of steps ahead to predict.
M: a set of forecasting methods in the pool.
C: a set of combination methods.

Output:

the set of weights for each forecast combination method

1: Split y into a training period and test period of size h.
2: Fit each forecasting method $m \in M$ over the training period and generate the forecasts over the test period.
3: Calculate the set of weights for each combination method $c \in C$ on the observations of test period.

3.1. Problem Space

The M4 dataset includes 100,000 time series of different frequencies. For each time series, we split it into a training and test period, whose length was set equal to the forecast horizon set by the competition. The dataset is limited to time series that are not constant over the training period and with a minimum number of observations relative to the frequency of the series and the number of steps ahead to predict since there would be two holdout periods, one to compare the results and the other to estimate the weights for the combination methods. The amount of time series included in the *ex ante* evaluation is 94,966. One fundamental assumption is that the new series comes from a similar data-generating process to the reference set.

The M4 aims to provide time series that reflects reality; thus, it employs a dataset consisting of a random sample of 100,000 time series extracted from the FoReDeck, a database made available by the National Technical University of Athens containing around 900,000 time series from multiple socioeconomic sources of data generated by human interactions. The dataset has six main categories: demographic, finance, industry, macroeconomic data, microeconomic data and other types. It includes data from industries, services, tourism, imports exports, demographics, education, labour and wage, government, households, bonds, stocks, insurance, loans, real estate, transportation, et cetera.

The dataset has six distinct time frequencies: hourly, daily, weekly, monthly, quarterly and yearly. However, the proportion in which they appear is not uniform; 95% of the series comes from monthly (48%), quarterly (24%) and yearly (23%) frequency. Regarding the available observations, the shortest training test is 13 for yearly, 16 for quarterly, 42 for monthly, 80 for weekly, 93 for daily and 700 for hourly time series. The number of predicted steps ahead also varied by the series' frequency. The contestants had to predict six steps for the yearly time series, eight for quarterly, 18 for monthly, 13 for weekly, 14 for daily and 48 for hourly.

The organisers of the M4 competition scaled the time series to prevent negative observations and values lower than 10. The scaling was performed by adding a constant to the series so that their minimum value was equal to 10 (only 29 occurrences across the whole dataset). In addition, the organisers removed any information that could lead to identifying the original series to ensure the results' objectivity. For example, the series' starting dates were only available once the M4 competition had ended.

3.2. Time Series Features

The features used should provide information about the structure of the time series, such as trend, seasonality, cycle, and so on. The features for seasonal time series are set to zero for non-seasonal time series, as in the FFORMA implementation. Also, the features are calculated for the scaled time series because

Algorithm 2 The mFFORMS framwork: Forecast combination selection based on meta-learning

OFFLINE PHASE: TRAIN THE CLASSIFIER

Inputs:

$\{y_1, y_2, \ldots, y_N\}$: N observed time series forming the reference set.
F: a set of functions for calculating time series features.
M: a set of forecasting methods in the pool.
C: a set of combination methods.

Output:

mFFORMS meta-learner: a classifier that returns the label of the combination method with lowest expected loss given the time series features input.

Prepare the meta-data

for $n = 1$ to N: **do**
  Split y_n into a training period and test period.
  Calculate the set of features $f \in F$ over the training period.
  Estimate out-of-sample weights (algorithm 1)
  Fit each forecasting method $m \in M$ over the training period and generate forecasts over the test period.
  Combine the forecasts for the test period using the set of weights (4).
  Calculate the forecast error of each combination over the test period.
  Select the combination with the lowest forecast error .
  Meta-data: input features (2) and label of best combination (8).
Bind the meta-data from each time series.

Train the meta-learner classifier

Train a xgboost classifier based on the meta-data
Return the mFFORMS meta-learner

ONLINE PHASE: FORECAST A NEW TIME SERIES

Input:

mFFORMS meta-learner from offline phase.

Output:

Forecast the new time series y_{new}.

for each y_{new}: **do**
  Calculate features y_{new} by applying F.
  Use the meta-learner to produce the label of the combination method.
  Estimate out-of-sample weights Algorithm 1
  Fit each forecasting method $m \in M$ over all data and generate forecasts
  Combine the forecasts using the set of weights to generate final forecasts

some features are scale dependent. The features provide information about the dynamic structure of the time series, such as trend, seasonality, autocorrelation, nonlinearity, heterogeneity, and so on. Furthermore, interpretability, robustness to outliers, and scale and length independence should also be considered when selecting features for this classification problem.

The $tsfeatures$ R-package by Hyndman et al. (2022) was employed to calculate the time series features. The Table 1 was extracted from Montero-Manso et al. (2020). For a more detailed definition of the time series features, see Talagala et al. (2018).

Table 1: Time Series Features in mFFORMS framework

Feature		**Description**	**Non-seasonal**	**Seasonal**
1	T	length of time series	✓	✓
2	trend	strength of trend	✓	✓
3	seasonality	strength of seasonality	-	✓
4	linearity	linearity	✓	✓
5	curvature	curvature	✓	✓
6	spikiness	spikiness	✓	✓
7	e_acf1	first ACF value of remainder series	✓	✓
8	e_acf10	sum of squares of first 10 ACF values of remainder series	✓	✓
9	stability	stability	✓	✓
10	lumpiness	lumpiness	✓	✓
11	entropy	spectral entropy	✓	✓
12	hurst	Hurst exponent	✓	✓
13	nonlinearity	nonlinearity	✓	✓
13	alpha	ETS(A,A,N) $\hat{\alpha}$	✓	✓
14	beta	ETS(A,A,N) $\hat{\beta}$	✓	✓
15	hwalpha	ETS(A,A,A) $\hat{\alpha}$	-	✓
16	hwbeta	ETS(A,A,A) $\hat{\beta}$	-	✓
17	hwgamma	ETS(A,A,A) $\hat{\gamma}$	-	✓
18	ur_pp	test statistic based on Phillips–Perron test	✓	✓
19	ur_kpss	test statistic based on KPSS test	✓	✓
20	y_acf1	first ACF value of the original series	✓	✓
21	diff1y_acf1	first ACF value of the differenced series	✓	✓
22	diff2y_acf1	first ACF value of the twice-differenced series	✓	✓
23	y_acf10	sum of squares of first 10 ACF values of original series	✓	✓
24	diff1y_acf10	sum of squares of first 10 ACF values of differenced series	✓	✓
25	diff2y_acf10	sum of squares of first 10 ACF values of twice-differenced series	✓	✓
26	seas_acf1	autocorrelation coefficient at first seasonal lag	-	✓
27	sediff_acf1	first ACF value of seasonally differenced series	-	✓
28	y_pacf5	sum of squares of first 5 PACF values of original series	✓	✓
29	diff1y_pacf5	sum of squares of first 5 PACF values of differenced series	✓	✓
30	diff2y_pacf5	sum of squares of first 5 PACF values of twice-differenced series	✓	✓
31	seas_pacf	partial autocorrelation coefficient at first seasonal lag	✓	✓
32	crossing_point	number of times the time series crosses the median	✓	✓
33	flat_spots	number of flat spots, calculated by discretizing the series into 10 equal-sized intervals and counting the maximum run length within any single interval	✓	✓
34	nperiods	number of seasonal periods in the series	-	✓
35	seasonal_period	length of seasonal period	-	✓
36	peak	strength of peak	✓	✓
37	trough	strength of trough	✓	✓
38	ARCH.LM	ARCH LM statistic	✓	✓
39	arch_acf	sum of squares of the first 12 autocorrelations of z^2	✓	✓
40	garch_acf	sum of squares of the first 12 autocorrelations of r^2	✓	✓
41	arch_r2	R^2 value of an AR model applied to z^2	✓	✓
42	garch_r2	R^2 value of an AR model applied to r^2 ✓	✓	✓

3.3. Algorithm Space

3.3.1. Combination Methods

The first step of mFFORMS consists of generating different combinations of candidate forecasting methods. Consider y_t a univariate time series selected from P and let $\hat{y}_{m,t+h|t}$ be an h-step-ahead forecast of y_{t+h} made at time t using the information set available at the moment t by the candidate forecast $m \in M$, where M is the set of forecasting methods in the pool. The combination $c \in C$, where C is the set of combination methods, without imposing a sum-to-one constraint is given by

$$c_{t+h|t} = \beta_1 \hat{y}_{1,t+h|t} + \beta_2 \hat{y}_{2,t+h|t} + \cdots + \beta_M \hat{y}_{M,t+h|t}$$

where β_m is the non-negative weight associated to the forecast $\hat{y}_m$. To reduce the notational clutter in this section, the $t+h|t$ will be dropped, considering it will be clear in the context that the t notation will refer to cross-validated observations.

The first candidate is the equal-weights combination, a simple average of all the candidate forecasts. A second candidate is a naïve approach that assigns weight one to the forecasting method that produces the lowest mean squared cross-validated error and zeroes to the others. The other candidates are penalised methods that overcome some limitations of the Bates-Granger weights estimation method. First, BG cannot be estimated when the number of candidate forecasts exceeds the number of observations. Second, when there are highly correlated forecasts, the variance inflation factor makes the variance of the estimated weights larger than they would otherwise be. Also, when the forecasts are redundant or not advantageous for the prediction combination, they are still considered in the combination. The penalised combination is helpful in these scenarios. The considered methods are LASSO and peLASSO.

Ridge-based methods are not considered because they cannot produce a parsimonious combination since they keep all the candidates in the combination. Elastic Net, proposed by Zou and Hastie (2005), is a convex combination of LASSO and Ridge. It overcomes LASSO limitations when the number of predictors is much larger than the number of observations. However, it also exhibits the grouping effect, where strongly correlated predictors tend to be in or out of the model together; furthermore, given that the elastic net penalty is strictly convex, a group of highly correlated variables tend to be equal. In the forecast combination setting, the grouping effect keeps redundant models.

LASSO. The primary reference for LASSO methods is Tibshirani (1996), it is a penalized least squares method that imposed a L_1-penalty on the regression coefficients; due to the nature of the L_1-penalty, the LASSO simultaneously shrinks and selects the coefficients. Tibshirani (1996) points out that LASSO does not uniformly dominate the ridge regression. Despite its success in many applied situations, it has some limitations. Despite dealing with scenarios where the number of candidates exceeds the number of observations, it selects at most n variables before it saturates because of the nature of the convex optimization problem. If there is a group of variables with a very high pairwise correlation, LASSO tends to select only one and does not care which one is selected (since it is not strictly convex, it does not have a unique solution). In the combination setup, the LASSO is given by

$$\widehat{\boldsymbol{\beta}}_{LASSO} = \arg\min_{\beta} \left(\sum_{t=1}^{T} \left(y_t - \sum_{i=1}^{M} \beta_i \hat{y}_{i,t} \right)^2 + \lambda \sum_{i=1}^{M} |\beta_i| \right),$$

where $t = 1$ refers to the first out-of-sample observation and $T = t + h$

peLASSO. Diebold and Shin (2019) proposed the partially-egalitarian LASSO, a LASSO-based procedure that sets some combining weights to zero and shrinks the survivors toward equality. The LASSO select and shrink the coefficients; however, the direction of the selection and shrinkage is towards zero. In its concept, peLASSO is a two-penalty function given by

$$\hat{\boldsymbol{\beta}}_{peLASSO} = \arg\min_{\beta} \left(\sum_{t=1}^{T} \left(y_t - \sum_{i=1}^{M} \beta_i \hat{y}_{i,t} \right)^2 + \lambda_1 \sum_{i=1}^{M} |\beta_i| + \lambda_2 \sum_{i=1}^{M} \left| \beta_i - \frac{1}{p(\beta)} \right| \right),$$

where $p(\beta)$ is the number of elements different from zero in β. The first penalty term is the standard LASSO that selects and shrinks towards zero, while the second penalty selects and shrinks towards equality. Optimizing the objective function in one step is difficult due to its discontinuity when $\beta_i = 0$; therefore, the authors propose a two-step estimation.

In the first step, the goal is to select the m predictions within the complete set of forecasts M. For such an end, they employ the standard LASSO with one penalty. The method can handle situations where $M > T$, in other words, where the number of predictions exceeds the number of observations. In the second step, the goal is to shrink towards equality the m forecasts that survive the first step. For this purpose, the authors employ the egalitarian Ridge (eRidge), which shrinks the coefficients towards equality. It is estimated using standard Ridge regression with a modification in the response variable, which consists of the difference between the actual value and the simple average of the surviving forecasts.

3.3.2. Forecasting Methods

The pool of nine candidate forecasting methods was implemented in the *forecast* package in R by Hyndman and Khandakar (2008). The choice of parameters is the same as in the implementation available at robjhyndman's github repository M4metalearning, where most of the methods were run with the default parameters. The methods are:

1. naïve (naive);
2. random walk with drift (rwf). The parameter *drift* is set to TRUE;
3. seasonal naïve (snaive);
4. theta method (thetaf);
5. automated ARIMA algorithm (auto.arima). The parameter *stepwise* is set to FALSE, so it searches over all methods, which can be relatively slow to seasonal series;
6. automated exponential smoothing algorithm (ets). The parameter *opt.crit* is set to *'mae'*, which uses the mean of absolute errors as optimisation criteria;
7. TBATS model (tbats);
8. STLM-AR seasonal and trend decomposition using loess with AR modelling of the seasonally adjusted series. The parameter *modelfunction* is set to *stats::ar* to model the seasonally adjusted series. Furthermore, if the stlm produces an error, an automated ARIMA algorithm is used instead.
9. neural networks autoregressive (nnetar).

A brief description of each forecasting model is available at Appendix Appendix A.

3.4. Error Metric

Hyndman and Koehler (2006) proposed a measure named MASE, which has a defined mean and finite variance, is scale-independent and can be computed for a single forecast horizon, being less than one if it provides a better forecast than the benchmark and more than one otherwise. Franses (2016) points out that the absolute scale error has the same moment properties as the absolute error and thus has moment properties that match the assumptions underlying the asymptotic theory of the Diebold-Mariano test. In the M4 competition, a modified version of MASE was proposed to evaluate, given by

$$\textbf{MASE} = \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} \left| Y_t - \widehat{Y}_t \right|}{\frac{1}{n-m} \sum_{t=m+1}^{n} |Y_t - Y_{t-m}|}.$$

In the Hyndman and Koehler (2006) version, the m is set to one regardless of the frequency of the data. Also, the M4 is scaled by the in-sample errors of the benchmark, which might generate a MASE above one when comparing the benchmark against itself.

3.5. Meta-Learner: the mapping function

The candidate approaches to estimating the optimal hyperparameters for the meta-learner are: (i) minimising the cross entropy to calculate the candidate meta-learners specifications and selecting the one with the lowest cross-validated loss, a two-step procedure as in FFORMS, and (ii) minimising a custom objective function depending on the class labels, a single-step procedure as in FFORMA. As in FFORMA, the learning model implementation utilised the gradient tree boosting model of XGBoost, a supervised machine-learning algorithm that implements regularised gradient boosting implemented by Chen and Guestrin (2016) in the *xgboost* R package. However, instead of selecting the hyperparameters that minimise a custom loss, OWA (the M4 evaluation score), we consider the optimisation approach as in FFORMS, where the meta-learner is trained as a classification problem minimising the cross-entropy, then selecting the configuration that minimises the error measure, in this case, the MASE. We are not interested in accurately predicting the class as in a classification problem but in minimising overall error given the output labels. The configurations vary on the class imbalance, where some classes are significantly more prevalent than others in the available data, and down-sampling might be required. Another configuration choice for the meta-learner is calculating a single model or one model for each period (monthly, yearly, and so on).

The selection of the meta-learner strategy is the one that provides the

$$\min \sum_{n=1}^{N} \sum_{k=1}^{K} I_{c_k = \hat{S}(f_n)} L_{nk},$$

where K is the length of C, the set of candidate combination methods mapped in the algorithm selection problem, $C = \{c_1, c_2, ..., c_K\}$, N is the length of the reference set considered in the meta-learning estimation, L_{nk} is the loss associated to the k-th combination for the n-th time series, $\hat{S}(f_n) = \hat{C}$ is the estimated class produced by the meta-learner given the time series features f_n, and $I_{c_k = \hat{C}}$ is the indicator function for when the k-th combination is equal to the predicted combination.

The performance of XGBoost relies on the choice of hyperparameters, which cannot be estimated directly from the data. We consider the race ANOVA algorithm proposed by Kuhn (2014) to efficiently select the set of hyperparameters that minimise the cross entropy on cross-validated data. The hyperparameters are essential to prevent the trees to growing too much and having end-points that contain very few observations, which can, in turn, lead to overfitting. Estimating the best subset of hyperparameters based on cross-validated error avoids the eventual problem of learning patterns available only in the training data.

4. Ex Ante Evaluation

In this section, we conduct an evaluation of the results of the historical cross-validated data, limited to the length of series available to the competitors throughout the competition.

4.1. Evaluating the Results

For each series and combination method, the errors are scaled by the in-sample loss of the naive prediction, as proposed in the competition guidelines. From now on, we will call it the competition MASE or cMASE. However, to aggregate and compare the results, we scale it by the cMASE of the seasonal naïve as the benchmark. We then compute the ratio between the cMASE of each candidate method and the cMASE of the seasonal naïve. When it is equal to one, the proposed method had the same cMASE as the benchmark; when it is below one, the combination provided more accurate forecasts; when it is above one, the seasonal naive provided better predictions. From now on, this ratio will be named eMASE, or evaluation MASE.

$$eMASE = \frac{cMASE}{cMASE_{benchmark}}.$$

4.1.1. Frequency Each Method Provides Better Results

In this subsection, we evaluate how frequently each combination method should be selected to provide the lowest loss among the candidate combination methods for each time series in consideration. Overall, the combination for adaptation (CFA) scenario represents 40% of the series in consideration and the combination for improvement (CFI) scenario the remaining 60%, as displayed in Table 2. For 40% of the series, a single forecasting method better describes the data-generating process. For the series in the CFI scenario, the better strategy is to employ the equal weights combination in 59% of these cases. It is worth estimating the optimal weights using LASSO or peLASSO only in 41% of the CFI scenario cases. We consider the relative frequency CFI for Min RMSE to be NA, since it belongs to the CFA scenario.

Table 2: How often each combination method provides better forecasts

Scenario	Combination	Rel. Frequency	Rel. Frequency CFI
CFA	Min RMSE	39.97%	NA
CFI	1/N	35.51%	59.07%
CFI	LASSO	9.86%	16.40%
CFI	peLASSO	14.75%	24.53%

In the Figure 2, we observe this scenario is not homogeneous throughout the series when we evaluate the frequency in which each combination method provides the lowest loss by the time series frequency. However, it is essential to note that most of the M4 data set is composed of monthly, yearly and quarterly series.

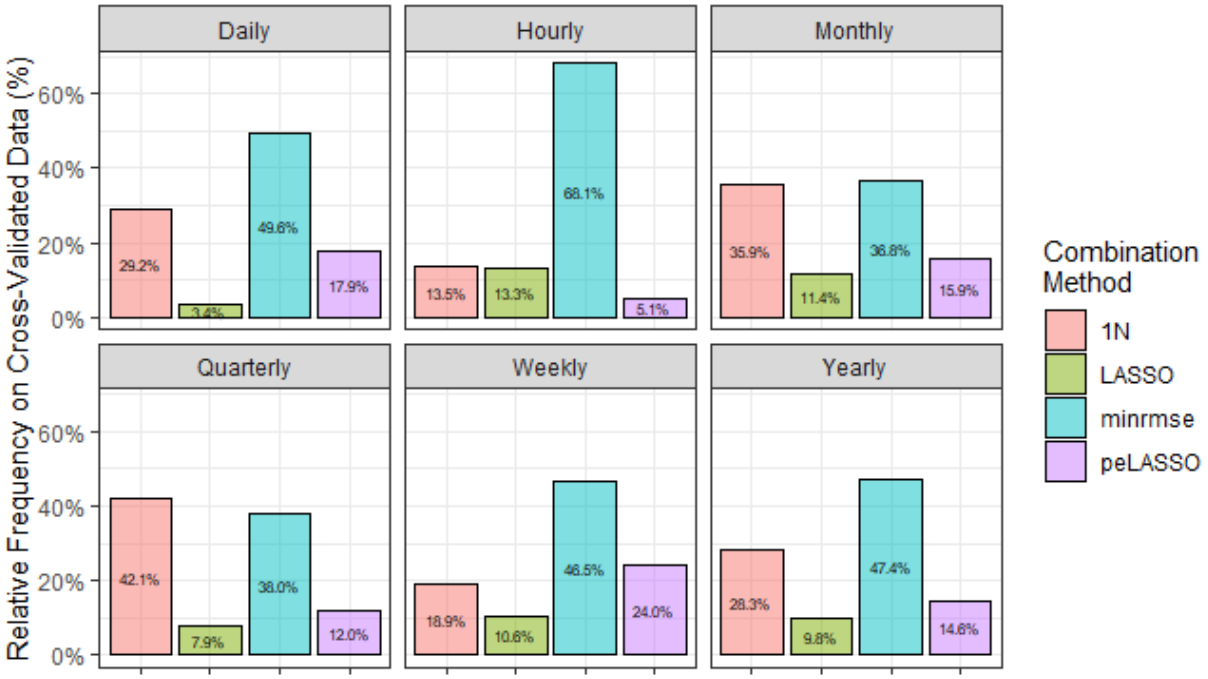

Figure 2: How often each combination method provides better forecasts per Period

For hourly series, the CFA scenario happens more often, around 68% of the time. For daily, yearly and weekly series, the CFI scenario appears around 46-49% of the time. For monthly and quarterly series, the

most frequent in the M4 dataset, only 36-38% of the scenarios are CFI. Within the CFA scenario, the simple average combination is more frequent in quarterly and monthly series; for the other periods, there is a more balanced distribution of how frequent each method is the best.

4.1.2. Combination Errors Dispersion

In this subsection, we compare the dispersion of the loss associated to each combination method. The idea is to evaluate whether the misclassification affects the overall loss considerably. In the Figure 3, we observe how the errors for each combination approach relate.

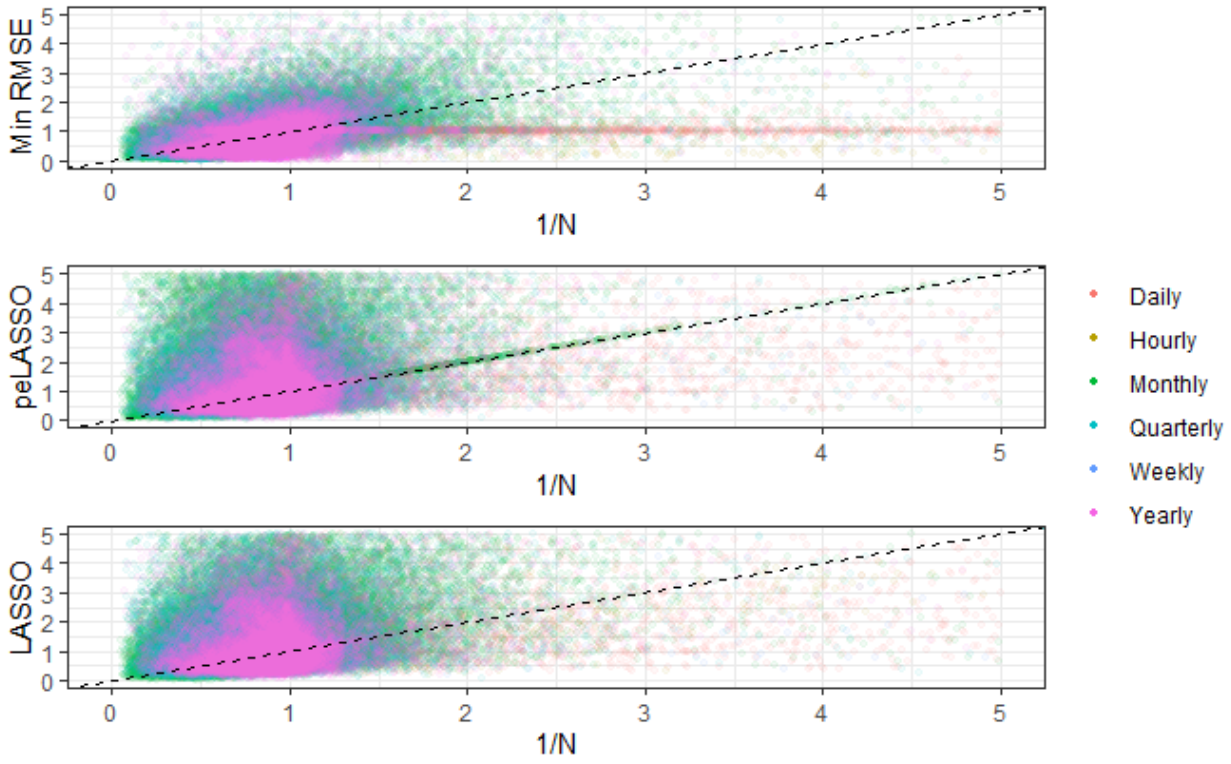

Figure 3: eMASE of the combination strategies

We observe that peLASSO and LASSO combination methods provide errors that are considerably higher than the equal weights combination. Furthermore, it appears that for a considerable amount of time series the seasonal-naive forecasting is the best method.

4.1.3. How Often Each Strategy Defeats the sNaive?

This subsection evaluates how often each combination method beats the seasonal naive benchmark, a simple approach that requires few computational resources.

The Table 3 shows the frequency each combination strategy provides lower cross-validated loss than the seasonal naive benchmark. The clear winner is the simple average, which beats the seasonal naive benchmark in 71.5% of the time series evaluated. The second best strategy is to select a method with the lowest cross-validated RMSE. The LASSO and peLASSO combination provide worse predictions more than half the time, except for the yearly series. However, the gain is marginal, demonstrating its inefficiency for this application due to its computing-intensive requirements and poor performance.

Table 3: How often each combination method provides better forecasts

Period	1/N	LASSO	Min RMSE	peLASSO
Daily	34.5%	27.9%	39.6%	27.4%
Hourly	27.5%	22.9%	62.6%	9.7%
Monthly	74.4%	44.1%	64.0%	40.6%
Quarterly	74.6%	41.7%	61.2%	41.3%
Weekly	43.7%	42.9%	60.4%	42.3%
Yearly	69.8%	51.4%	65.6%	51.3%
Total	71.5%	44.1%	62.5%	42.1%

The Figure 4 contains the distribution of the eMASE for each combination strategy. From the distribution, we observe that the minimum cross-validated error strategy beats the benchmark by a wide margin more frequently than the combination of equal weights, despite beating less frequently.

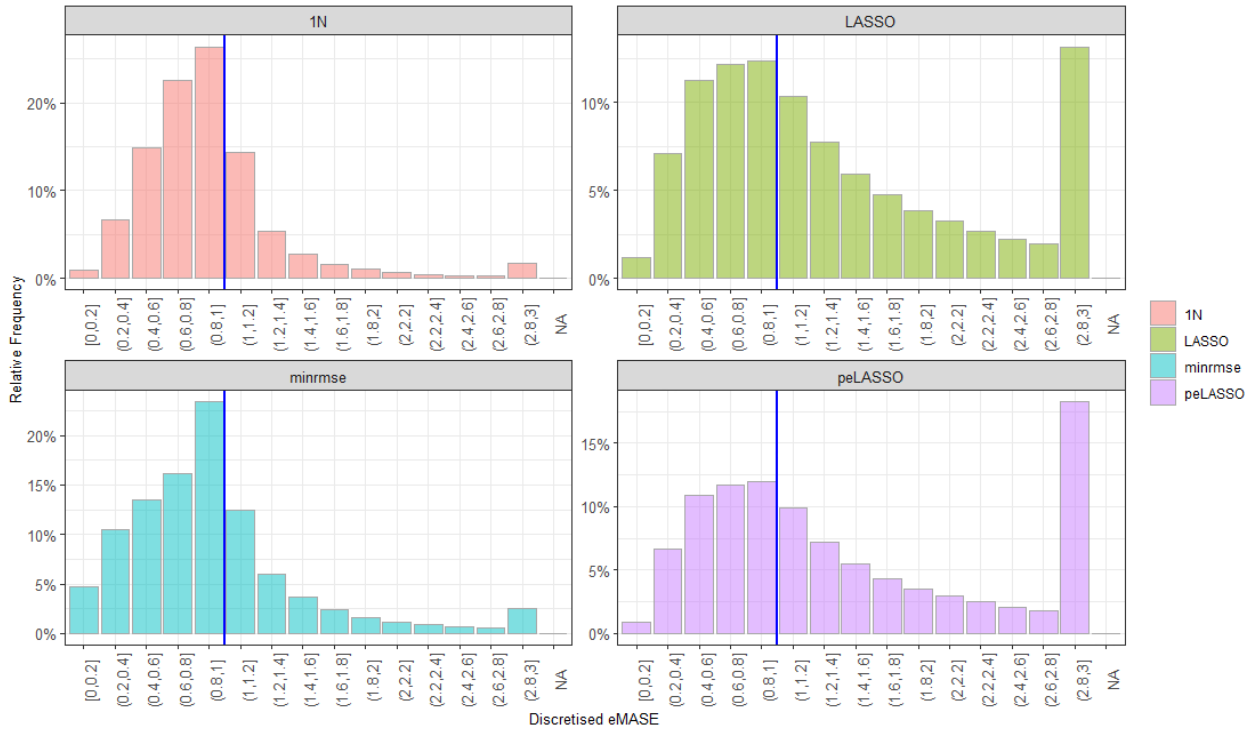

Figure 4: In the x-axis we display the discretised eMASE with intervals of size 0.2 and truncated at 3. The y-axis provide the relative frequency they occur within the displayed combination method. The relative frequencies sum to one within each facet. The colours and facets represent each distinct combination strategy. The blue line represents the point where the eMASE is equal to one.

4.2. Selecting the Meta-Learner

In this subsection, we evaluate the implications of the choices regarding the possible parametrisations of the meta-learner. We consider three candidate approaches. The first consists of a single meta-learner and unbalanced classes. The second consists of a single meta-leaner and balanced classes by down-sampling the most frequent ones. The third consists of multiple meta-learners, one for each period available, and unbalanced classes. We randomly select a third of the available time series of the *ex ante* section to estimate the meta-learner and the remaining two-thirds to compare the results.

Table 4: The candidate meta-learners cMASE

Period	Multiple Unbalanced	Single Balanced	Single Unbalanced
Daily	3.3432	4.0977	3.3001
Hourly	0.9937	1.4661	0.9937
Monthly	0.9515	1.2469	0.9274
Quarterly	1.2116	1.3781	1.2028
Weekly	2.6841	3.9763	2.6473
Yearly	3.4196	4.0303	3.3766

Table 5: How often each combination method provides better forecasts

Period	mFFORMS	1/N	LASSO	Min RMSE	peLASSO
Daily	48.2%	34.0%	27.7%	44.7%	27.2%
Hourly	72.5%	29.9%	23.4%	72.9%	9.6%
Monthly	73.4%	74.2%	43.7%	69.4%	40.3%
Quarterly	74.1%	74.9%	41.8%	67.2%	41.5%
Weekly	68.1%	41.4%	40.6%	67.7%	39.8%
Yearly	72.1%	69.8%	51.3%	71.8%	51.2%
Total	72.2%	71.4%	43.9%	68.2%	42.0%

The results in Table 4 demonstrate that the clear winner is the single meta-learner with unbalanced classes to predict the combination algorithm. It is marginally better than multiple meta-learners, except in hourly time series, where there is a tie. The meta-learner with balanced classes provides a worse performance due to predicting LASSO and peLASSO more often, even in scenarios where they perform considerably worse, thus making the overall loss considerably higher, as observed in Figure 4.

From now on, mFFORMS refers to the single meta-learner with unbalanced data.

4.3. mFFORMS Results

In this subsection, we compare the mFFORMS results with the candidates, but we display only three combination methods graphically: mFFORMS, simple average and minimum cross-validated RMSE. We discard the other methods because they provide a worse loss and make the plots harder to read.

In Table 5, we observe the mFFORMS beats overall the benchmark more often than any other method. A surprising observation is that the simple average combination beats the benchmark more often than mFFORMS in the monthly and quarterly series despite having a worse average loss. The explanation is that when the mFFORMS beat the benchmark, it beats by a wider margin. The Figure 5 provides a visual demonstration.

Table 6: $E[eMASE|eMASE < 1]$

Period	mFFORMS	1/N	LASSO	Min RMSE	peLASSO
Daily	0.86	0.75	0.68	0.89	0.68
Hourly	0.64	0.81	0.76	0.64	0.75
Monthly	0.65	0.66	0.64	0.65	0.65
Quarterly	0.65	0.69	0.64	0.64	0.64
Weekly	0.73	0.80	0.68	0.75	0.68
Yearly	0.60	0.78	0.59	0.58	0.59
Total	0.64	0.69	0.63	0.64	0.63

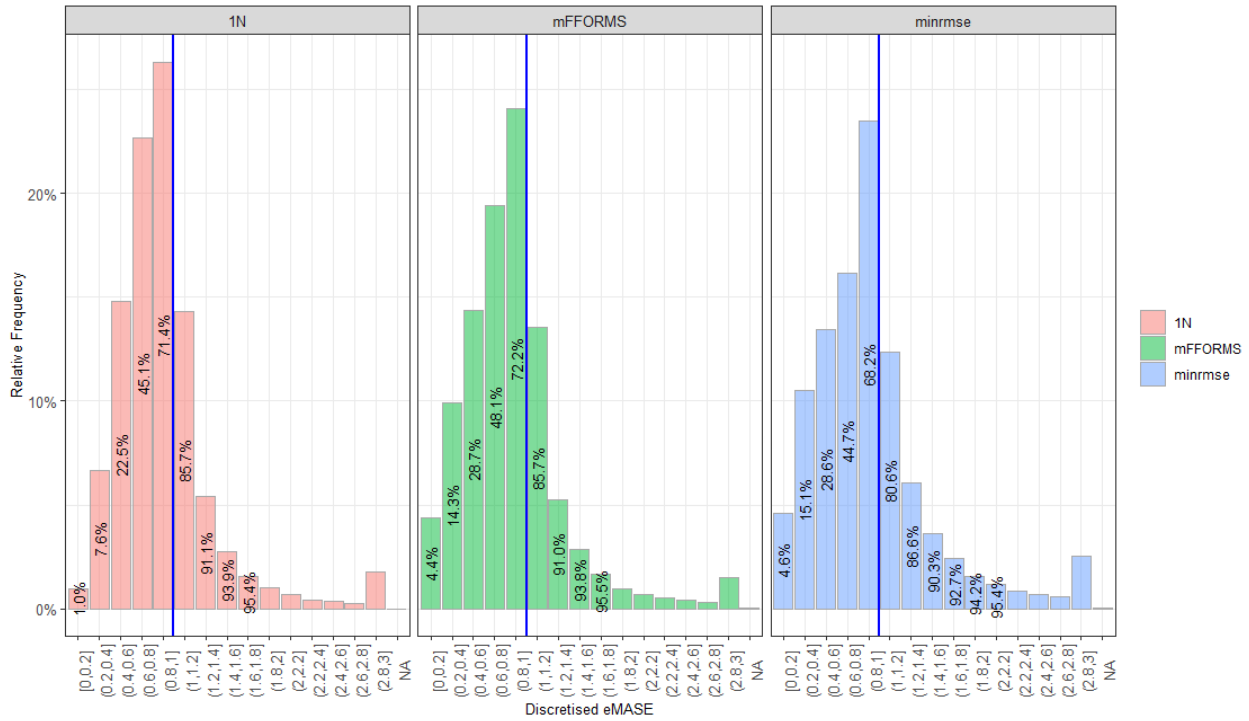

Figure 5: In the x-axis we display the discretised eMASE with intervals of size 0.2 and truncated at 3. The y-axis provide the relative frequency they occur within the displayed combination method. The relative frequencies sum to one within each facet. The colours and facets represent each distinct combination strategy. The blue line represents the point where the eMASE is equal to one. The text inside the bars represent the cumulative relative frequency.

The Table 6 displays the mean eMASE conditioned on being lower than one, so the lower the value, the higher the gain over the benchmark. We observe that the LASSO has the most gain when it beats the benchmark, despite not beating it often. Also, we observe that the mFFORMS approach has little to no gain over the minimum cross-validated RMSE and a smaller gain over the simple average, except on the daily time series.

4.3.1. Variables Importance

To understand the variables that help the most at identifying the most suited forecast combination method, we employ the Gain as a measure of variable importance. The Gain represents the improvement in the model's loss function achieved by splitting a tree node based on a particular feature. It measures how much each feature contributes to the overall reduction in the error of the model. The higher the value the metric assumes, the more influential the feature is in making accurate predictions. Since the measure is

relative, the sum of the Gain provided by the variables sum up to one.

From Figure 6, we observe that the time series features that help the most with prediction are the seasonal strength and trend, followed by the curvature, spike and linearity.

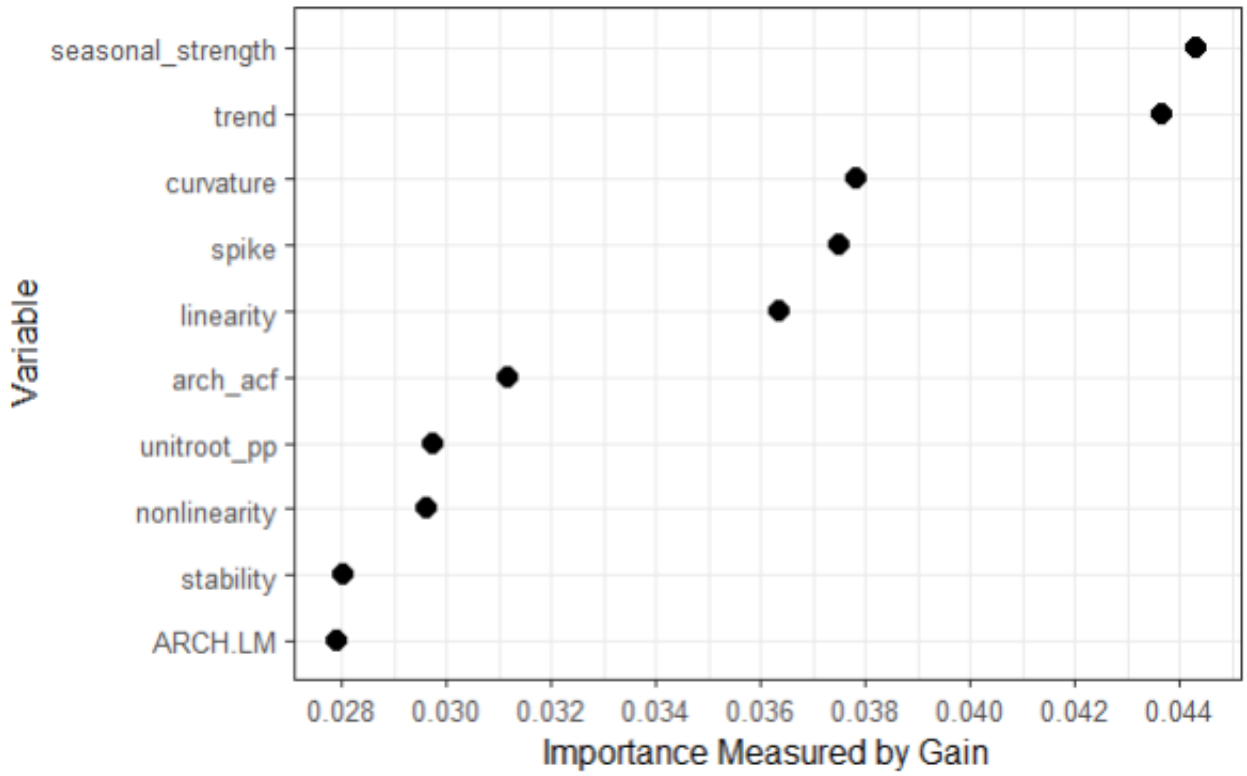

Figure 6: Variable Importance

For a detailed treatment of the gain, see Chen and Guestrin (2016).

5. Ex Post Evaluation

In this section, we compare the mFFORMS results with the ones obtained from FFORMA for the M4 competition submission. To compare the results, we calculate the MASE of the candidates considering FFORMA as the benchmark. So when it is equal to one, the proposed method had the same errors as the FFORMA; when it is below one, the candidate method provided more accurate forecasts; when it is above one, the FFORMA provided better predictions. In Table 7 we observe that the mFFORMS underperforms FFORMA for all time series frequencies. We also notice that the mFFORMS performs consistently worse than FFORMA for all time series lengths.

6. Conclusion

In conclusion, the results of our analysis showed that the forecast combination strategy under evaluation did not outperform the FFORMA in the M4 settings. FFORMA provides better results and is less intensive computationally since it provides a set of weights given the time series features and requires the series to be forecasted only once. In contrast, mFFORMS requires calculating some forecasting methods twice (except when it selects the equal weights combination method) due to its optimal weights estimation on cross-validated data. We observe that the multi-level approach provides more accuracy than the combination

Table 7: MASE with FFORMA as benchmark

Period	Mean	5th Percentile	1st Quartile	Median	3rd Quartile	95th Percentile
Total	1.120	0.590	0.908	1.029	1.222	2.196
Daily	1.197	0.730	0.945	1.018	1.136	2.080
Hourly	1.130	0.624	0.969	1.012	1.157	1.833
Monthly	1.166	0.638	0.916	1.024	1.191	2.008
Quarterly	1.171	0.605	0.910	1.036	1.233	2.085
Weekly	1.103	0.756	0.924	1.010	1.125	1.670
Yearly	1.304	0.472	0.871	1.039	1.320	2.759

strategies in the algorithm space on the historical cross-validated setting; however, the gains are marginal, and the excess complexity is very high. Considering the results, We infer that FFORMA exploits better the dependencies between time series and that FFORMA already deals with the CFA and CFI scenarios in the M4 competition settings. Another potential reason for the mFFORMS underperformance is that any classification approach considers only the best combination method, regardless of the performance gap between the first and second methods. Conversely, the approach taken by FFORMA considers the magnitude of the error of each candidate forecast while attributing weights. Furthermore, Montero-Manso et al. (2020) points out that FFORMA provides a diverse profile of weights, ranging from simple average to an almost classification weights set (most of the weight to only a single method).

Appendix A. Forecast Methods

The **naïve** forecasting method consists of replicating the last observation available, and it is equivalent to an ARIMA(0,1,0). The **seasonal naïve** replicates the last seasonal observation given the time series frequency, equivalent to an ARIMA(0,0,0)(0,1,0)m, where m is the time series frequency. If the frequency is one, it is analogous to a naïve approach. The **random walk with drift** is equivalent to an ARIMA(0,1,0) model with a drift coefficient.

The **ets** method is based on an extended range of exponential smoothing methods. The acronym 'ets' stands for error type, trend type and seasonality type, where each can assume different forms: none, additive, multiplicative. So, for example, "ANN" is simple exponential smoothing with additive errors, "MAM" is multiplicative Holt-Winters' method with multiplicative errors, and so on. We consider the fully automated selection method, which chooses the type of error, trend and seasonality. The parameters are optimised for all appropriate models, selecting the best type according to the AIC or mean absolute error.

The **auto.arima** estimation follows the algorithm proposed by Hyndman and Khandakar (2008), which combines unit root tests and the minimisation of AICc to obtain an ARIMA model. The parameters passed to the algorithm affect the solution path. As in FFORMA implementation, we consider the non-stepwise approach that searches over all models and returns the one with the lowest AIC. Some parameters set boundaries to the search space; they limit the maximum orders each (p, P, q, Q) may take and the maximum sum of these orders. The order of the first-differencing is estimated by the KPSS unit root test, and the order of seasonal-differencing is selected by a measure of seasonal strength computed from an STL decomposition.

The **theta method** of Assimakopoulos and Nikolopoulos (2000) is equivalent to simple exponential smoothing with drift, as demonstrated in Hyndman and Billah (2003). The series is tested for seasonality, where the criterion is the t-test value for the auto-correlation function value with the lag to previous seasonal observation compared to the t-statistic value for 0.1 probability. If the series is seasonal, it is seasonally adjusted using a classical multiplicative decomposition before applying the theta method. The resulting forecasts are then reseasonalized.

The **TBATS** method is the Exponential smoothing state space model that was developed for forecasting time series with complex seasonal patterns developed by Livera et al. (2011). It stands for **T**rigonometric **B**ox-Cox transformation, **A**RMA errors, **T**rend and **S**easonal components. The default parameters evaluate

whether to use the Box-Cox transformation, whether to include a trend or not, and whether to include a dumping parameter in the trend or not. Also, it considers ARMA errors.

The **NNETAR** is a feed-forward neural network with a single hidden layer and lagged values of the time series as inputs. By default, the number of nodes in the hidden layer is half of the number of input nodes plus one. Also, it estimates 20 networks with different random starting weights that are averaged when producing forecasts.

The **STLM** consists of applying a non-seasonal forecasting method to the seasonally adjusted data by LOESS (locally estimated scatterplot smoothing) and re-seasonalizing using the last full period of the seasonal component.

Appendix B. Error Metric

Error measures can be used to select models and estimate their parameters. Some of these measures are inadequate to optimise the model's parameters due to some drawbacks, and others are not adequate to compare the performance of forecasting methods for series with different scales.

Appendix B.1. Point Forecasts

Typically, point forecasts are evaluated by evoking a loss function (usually the quadratic loss), and the average loss is employed to compare candidate methods' performance. Regardless of the loss function choice, comparing the performance of different models for the same series or multiple series on the same scale is pretty straightforward. However, the problem arises when one wants to assess the performance of the candidate forecast methods in series measured on different scales. For example, the mean squared error might be misleading in this scenario despite providing unbiased forecasts. The mean squared errors lead to unbiased forecasts; however, they are sensitive to very high errors. It is calculated by

$$\mathbf{MSE} = \frac{1}{h}\sum_{t=n+1}^{n+h}\left(Y_t - \widehat{Y}_t\right)^2.$$

Some error measures, such as the ones based on percentage errors, are scale-independent and intuitive to compare the performance of the models on time series of different scales. The mean absolute percentage error, MAPE, tends to be biased downwards due to its asymmetric treatment of errors since the under-forecast is limited to 100

$$\mathbf{MAPE} = \frac{1}{h}\sum_{t=n+1}^{n+h}\frac{\left|Y_t - \widehat{Y}_t\right|}{|Y_t|} \times 100\,(\%)\,.$$

$$\mathbf{sMAPE} = \frac{2}{h}\sum_{t=n+1}^{n+h}\frac{\left|Y_t - \widehat{Y}_t\right|}{|Y_t| + \left|\widehat{Y}_t\right|} \times 100\,(\%)\,.$$

Goodwin and Lawton (1999) point out that the symmetric MAPE (sMAPE) is a flawed symmetrised version of MAPE. Despite being symmetric for interchanged values of actuals and forecasts, it introduces an asymmetry in treating negative and positive errors. They also demonstrate that, under some circumstances, a non-monotonic relationship can occur between the sMAPE and the absolute forecast errors. It is defined if forecasts and actuals are not both zero, but if there is a realisation of zero, the contribution is two independent of the point forecast.

Another group of interest is the relative error measures, which compare the error of a forecast relative to a benchmark. The relative error measure will be undefined if the benchmark forecasts the actual without error, which is more likely to happen on a period-by-period evaluation but not over a forecasting horizon. Hyndman and Koehler (2006) proposed a measure named MASE, which has a defined mean and finite variance, is scale-independent and can be computed for a single forecast horizon, being less than one if it

provides a better forecast than the benchmark and more than one otherwise. Franses (2016) points out that the absolute scale error has the same moment properties as the absolute error and thus has moment properties that match the assumptions underlying the asymptotic theory of the Diebold-Mariano test. In the M4 competition, a modified version of MASE was proposed to evaluate, given by

$$\textbf{MASE} = \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} \left| Y_t - \widehat{Y}_t \right|}{\frac{1}{n-m} \sum_{t=m+1}^{n} |Y_t - Y_{t-m}|}.$$

In the Hyndman and Koehler (2006) version, the m is set to one regardless of the frequency of the data.

Appendix B.2. Interval

Winkler (1972) proposed a score method to allow comparisons between intervals that consider the coverage rate and the width of the intervals. The Winkler (1972) score is defined by the mean of

$$W\left(L_t, U_t, Y_t\right) = \begin{cases} \left(U_t - L_t\right); L_t < Y_t < U_t \\ \left(U_t - L_t\right) + \frac{2}{\alpha}\left(L_t - Y_t\right); Y_t < L_t \\ \left(U_t - L_t\right) + \frac{2}{\alpha}\left(Y_t - U_t\right); Y_t > U_t \end{cases}.$$

This score penalizes both wide ranges (since $U_t - L_t$ will be wide) and non-coverage, as observations that are further from the range are penalized more severely. Although this metric was proposed in 1972, it has only recently received more attention when a scaled version proposed by Gneiting and Raftery (2007) was used as the M4 competition metric; however, Hyndman (2020) points out that it seems rather *ad hoc* and its properties are unknown. It is calculated as

$$\textbf{MSIS} = \frac{1}{h} \times \frac{\sum_{t=n+1}^{n+h} \left(U_t - L_t\right) + \frac{2}{\alpha}\left(L_t - Y_t\right) I\left(Y_t < L_t\right) + \frac{2}{\alpha}\left(Y_t - U_t\right) I\left(Y_t > U_t\right)}{\frac{1}{n-m} \sum_{t=m+1}^{n} |Y_t - Y_{t-m}|}.$$

Askanazi et al. (2018) argue that comparisons of interval forecasts are problematic in several ways and should be abandoned for density forecasts, because density forecasts convey more information than interval forecasts and can be computed by simulation. Furthermore, density forecasts can be readily compared using known proper scoring rules like the log predictive score, whereas interval forecasts cannot.

Appendix C. XGBoost Hyperparameters

The hyperparameters in consideration are:

- **mtry**: the number of predictors (features) that are randomly sampled at each split when creating the tree models
- **trees**: the number of trees contained in the ensemble
- **min_n**: the minimum number of data points in a node that is required for the node to be split further
- **tree_depth**: the maximum depth of the tree (i.e. the number of splits)
- **learn_rate**: a number for the rate at which the boosting algorithm adapts from iteration-to-iteration. It is also referred to as the shrinkage parameter
- **loss_reduction**: the reduction in the loss function required to split further

References

P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, T. S. Talagala, Fforma: Feature-based forecast model averaging, International Journal of Forecasting 36 (2020) 86–92. URL: https://www.sciencedirect.com/science/article/pii/S0169207019300895. doi:https://doi.org/10.1016/j.ijforecast.2019.02.011, m4 Competition.

J. M. Bates, C. W. J. Granger, The combination of forecasts, OR 20 (1969) 451–468. URL: http://www.jstor.org/stable/3008764.

R. T. Clemen, Combining forecasts: A review and annotated bibliography, International Journal of Forecasting 5 (1989) 559–583. URL: https://www.sciencedirect.com/science/article/pii/0169207089900125. doi:https://doi.org/10.1016/0169-2070(89)90012-5.

A. Timmermann, Forecast Combinations, in: G. Elliott, C. Granger, A. Timmermann (Eds.), Handbook of Economic Forecasting, volume 1 of *Handbook of Economic Forecasting*, Elsevier, 2006, pp. 135–196. URL: https://ideas.repec.org/h/eee/ecofch/1-04.html.

X. Wang, R. J. Hyndman, F. Li, Y. Kang, Forecast combinations: An over 50-year review, International Journal of Forecasting (2022). URL: https://www.sciencedirect.com/science/article/pii/S0169207022001480. doi:https://doi.org/10.1016/j.ijforecast.2022.11.005.

J. Stock, M. Watson, Combination forecasts of output growth in a seven-country data set, Journal of Forecasting 23 (2004) 405–430. doi:10.1002/for.928.

J. Smith, K. F. Wallis, A simple explanation of the forecast combination puzzle*, Oxford Bulletin of Economics and Statistics 71 (2009) 331–355. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0084.2008.00541.x. doi:https://doi.org/10.1111/j.1468-0084.2008.00541.x.

D. Hendry, M. Clements, Pooling of forecasts, The Econometrics Journal 7 (2004) 1 – 31. doi:10.1111/j.1368-423X.2004.00119.x.

Y. Yang, Combining forecasting procedures: Some theoretical results, Econometric Theory 20 (2004) 176–222. URL: http://www.jstor.org/stable/3533509.

W. Qian, C. Rolling, G. Cheng, Y. Yang, On the forecast combination puzzle, Econometrics 7 (2019). doi:10.3390/econometrics7030039.

C. W. J. Granger, R. Ramanathan, Improved methods of combining forecasts, Journal of Forecasting 3 (1984) 197–204. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3980030207. doi:https://doi.org/10.1002/for.3980030207.

S. Makridakis, R. J. Hyndman, F. Petropoulos, Forecasting in social settings: The state of the art, International Journal of Forecasting 36 (2020) 15–28. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301876. doi:https://doi.org/10.1016/j.ijforecast.2019.05.011, m4 Competition.

T. S. Talagala, R. J. Hyndman, G. Athanasopoulos, Meta-learning how to forecast time series, Monash Econometrics and Business Statistics Working Papers 6/18, Monash University, Department of Econometrics and Business Statistics, 2018. URL: https://ideas.repec.org/p/msh/ebswps/2018-6.html.

J. R. Rice, The algorithm selection problem, volume 15 of *Advances in Computers*, Elsevier, 1976, pp. 65–118. URL: https://www.sciencedirect.com/science/article/pii/S0065245808605203. doi:https://doi.org/10.1016/S0065-2458(08)60520-3.

R. Hyndman, Y. Kang, P. Montero-Manso, T. Talagala, E. Wang, Y. Yang, M. O'Hara-Wild, tsfeatures: Time Series Feature Extraction, 2022. URL: https://CRAN.R-project.org/package=tsfeatures, r package version 1.1.

H. Zou, T. Hastie, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society. Series B (Statistical Methodology) 67 (2005) 301–320. URL: http://www.jstor.org/stable/3647580.

R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society: Series B (Methodological) 58 (1996) 267–288. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1996.tb02080.x. doi:https://doi.org/10.1111/j.2517-6161.1996.tb02080.x.

F. X. Diebold, M. Shin, Machine learning for regularized survey forecast combination: Partially-egalitarian lasso and its derivatives, International Journal of Forecasting 35 (2019) 1679–1691. URL: https://www.sciencedirect.com/science/article/pii/S0169207018301596. doi:https://doi.org/10.1016/j.ijforecast.2018.09.006.

R. J. Hyndman, Y. Khandakar, Automatic time series forecasting: the forecast package for R, Journal of Statistical Software 26 (2008) 1–22. doi:10.18637/jss.v027.i03.

R. J. Hyndman, A. B. Koehler, Another look at measures of forecast accuracy, International Journal of Forecasting 22 (2006) 679–688. URL: https://www.sciencedirect.com/science/article/pii/S0169207006000239. doi:https://doi.org/10.1016/j.ijforecast.2006.03.001.

P. H. Franses, A note on the mean absolute scaled error, International Journal of Forecasting 32 (2016) 20–22. URL: https://www.sciencedirect.com/science/article/pii/S0169207015000448. doi:https://doi.org/10.1016/j.ijforecast.2015.03.008.

T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 785–794. URL: https://doi.org/10.1145/2939672.2939785. doi:10.1145/2939672.2939785.

M. Kuhn, Futility analysis in the cross-validation of machine learning models, 2014. URL: https://arxiv.org/abs/1405.6974. doi:10.48550/ARXIV.1405.6974.

V. Assimakopoulos, K. Nikolopoulos, The theta model: a decomposition approach to forecasting, International Journal of Forecasting 16 (2000) 521–530. URL: https://www.sciencedirect.com/science/article/pii/S0169207000000662. doi:https://doi.org/10.1016/S0169-2070(00)00066-2, the M3- Competition.

R. J. Hyndman, B. Billah, Unmasking the theta method, International Journal of Forecasting 19 (2003) 287–

290. URL: https://www.sciencedirect.com/science/article/pii/S0169207001001431. doi:https://doi.org/10.1016/S0169-2070(01)00143-1.

A. M. D. Livera, R. J. Hyndman, R. D. Snyder, Forecasting time series with complex seasonal patterns using exponential smoothing, Journal of the American Statistical Association 106 (2011) 1513–1527. doi:10.1198/jasa.2011.tm09771.

P. Goodwin, R. Lawton, On the asymmetry of the symmetric mape, International Journal of Forecasting 15 (1999) 405–408. URL: https://www.sciencedirect.com/science/article/pii/S0169207099000072. doi:https://doi.org/10.1016/S0169-2070(99)00007-2.

R. L. Winkler, A decision-theoretic approach to interval estimation, Journal of the American Statistical Association 67 (1972) 187–191. URL: http://www.jstor.org/stable/2284720.

T. Gneiting, A. Raftery, Strictly proper scoring rules, prediction, and estimation, Journal of the American Statistical Association 102 (2007) 359–378. doi:10.1198/016214506000001437.

R. J. Hyndman, A brief history of forecasting competitions, International Journal of Forecasting 36 (2020) 7–14. URL: https://www.sciencedirect.com/science/article/pii/S016920701930086X. doi:https://doi.org/10.1016/j.ijforecast.2019.03.015, m4 Competition.

R. Askanazi, F. X. Diebold, F. Schorfheide, M. Shin, On the comparison of interval forecasts, Journal of Time Series Analysis 39 (2018) 953–965. URL: https://ssrn.com/abstract=3261535. doi:hhttp://dx.doi.org/10.1111/jtsa.12426.

5 Conclusão

Em conclusão, os resultados de nossa análise mostraram que a estratégia de combinação de previsão proposta não superou o FFORMA nas configurações M4. O FFORMA fornece melhores resultados e é menos intensivo computacionalmente, pois fornece um conjunto de pesos dado as *features* das série temporal e requer que a série seja prevista apenas uma vez. Em contraste, o mFFORMS requer o cálculo de alguns métodos de previsão duas vezes (exceto quando ele seleciona o método de combinação de pesos iguais) devido à sua estimativa de pesos ótimos em dados de validação cruzada. Observamos que a abordagem multinível apresenta menores erros do que as combinações de previsão que compõe o grupo de candidatos individualmente nos dados de validação cruzada histórica; no entanto, os ganhos são marginais e o excesso de complexidade é muito alto. Inferimos que o FFORMA explora melhor as dependências entre as séries temporais. Outra razão potencial para o baixo desempenho do mFFORMS é que qualquer abordagem de classificação considera apenas o melhor método de combinação, independentemente da diferença de desempenho entre o primeiro e o segundo métodos. Por outro lado, a abordagem adotada pelo FFORMA considera a magnitude do erro de cada previsão candidata ao atribuir pesos. Além disso, Montero-Manso et al. [2020] aponta que o FFORMA fornece um perfil diversificado de pesos, variando de média simples a quase um conjunto de pesos de classificação (a maior parte do peso a apenas um único método).

Referências

J. M. Bates and C. W. J. Granger. The combination of forecasts. *OR*, 20(4):451–468, 1969. ISSN 14732858. URL http://www.jstor.org/stable/3008764.

Robert T. Clemen. Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting*, 5(4):559–583, 1989. ISSN 0169-2070. doi: https://doi.org/10.1016/0169-2070(89)90012-5. URL https://www.sciencedirect.com/science/article/pii/0169207089900125.

Clive W. J. Granger and Ramu Ramanathan. Improved methods of combining forecasts. *Journal of Forecasting*, 3(2):197–204, 1984. doi: https://doi.org/10.1002/for.3980030207. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3980030207.

David Hendry and Michael Clements. Pooling of forecasts. *The Econometrics Journal*, 7:1 – 31, 06 2004. doi: 10.1111/j.1368-423X.2004.00119.x.

Pablo Montero-Manso, George Athanasopoulos, Rob J. Hyndman, and Thiyanga S. Talagala. Fforma: Feature-based forecast model averaging. *International Journal of Forecasting*, 36(1):86–92, 2020. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2019.02.011. URL https://www.sciencedirect.com/science/article/pii/S0169207019300895. M4 Competition.

Wei Qian, Craig Rolling, Gang Cheng, and Yuhong Yang. On the forecast combination puzzle. *Econometrics*, 7, 05 2019. doi: 10.3390/econometrics7030039.

Jeremy Smith and Kenneth F. Wallis. A simple explanation of the forecast combination puzzle*. *Oxford Bulletin of Economics and Statistics*, 71(3):331–355, 2009. doi: https://doi.org/10.1111/j.1468-0084.2008.00541.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0084.2008.00541.x.

James Stock and Mark Watson. Combination forecasts of output growth in a seven-country data set. *Journal of Forecasting*, 23:405–430, 09 2004. doi: 10.1002/for.928.

Thiyanga S Talagala, Rob J Hyndman, and George Athanasopoulos. Meta-learning how to forecast time series. Monash Econometrics and Business Statistics Working Papers 6/18, Monash University, Department of Econometrics and Business Statistics, 2018. URL https://ideas.repec.org/p/msh/ebswps/2018-6.html.

Allan Timmermann. Forecast Combinations. In G. Elliott, C. Granger, and A. Timmermann, editors, *Handbook of Economic Forecasting*, volume 1 of *Handbook of Economic Forecasting*, chapter 4, pages 135–196. Elsevier, 2006. URL https://ideas.repec.org/h/eee/ecofch/1-04.html.

Xiaoqian Wang, Rob J. Hyndman, Feng Li, and Yanfei Kang. Forecast combinations: An over 50-year review. *International Journal of Forecasting*, 2022. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2022.11.005. URL https://www.sciencedirect.com/science/article/pii/S0169207022001480.

Yuhong Yang. Combining forecasting procedures: Some theoretical results. *Econometric Theory*, 20(1):176–222, 2004. ISSN 02664666, 14694360. URL http://www.jstor.org/stable/3533509.