CONRADO MATTEI DE CABANE OLIVEIRA

# META-REGRESSÃO PARA A PREVISÃO DE ERROS EM SÉRIES TEMPORAIS

(*versão pré-defesa, compilada em 29 de julho de 2021*)

Documento apresentado como requisito parcial ao exame de qualificação de Mestrado no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Prof. Luiz Eduardo Soares de Oliveira.

CURITIBA PR

2021

## RESUMO

A presença de séries temporais em governos, pesquisas e empresas cresceu devido ao aumento da captura, processamento e armazenamento de dados. Por outro lado, existem vários modelos preditivos para cada série temporal. Assim, a tarefa de previsão e escolha do melhor modelo para uma determinada série temporal pode ser custosa. O objetivo deste artigo é criar um meta-regressor para prever erros em previsões de séries temporais para facilitar a escolha do melhor modelo dada uma série temporal e um modelo de previsão. Para ajustar este meta-regressor, usamos 60 características de extraídas de cerca de 100.000 séries temporais. Nossos resultados experimentais mostram que o método proposto supera todos os quinze regressores usados neste trabalho e produz uma soma de erros 20% menor que o melhor regressor.



Palavras-chave: Séries Temporais. Meta-Regressor. Aprendizado de Máquina.

**ABSTRACT**

The presence of time series in governments, research, and companies has grown due to increased data capture, processing, and storage. On the other hand, there are several predictive models for every time-series. Thus, the prediction task and choosing the best model for a given time series can be costly. The purpose of this paper is to create a meta-regressor for predicting errors in time series predictions to facilitate the choice of the best model given a time series and a forecast model. To fit this meta-regressor, we have used 60 time-series features extract from about 100,000 time-series. Our experimental results show that the proposed method surpasses all fifteen regressors used in this work and produces a sum of errors 20% smaller than the best regressor.

Keywords: Time Series. Meta-Regressor. Machine Learning.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ACRÔNIMOS

| | |
|---|---|
| ANN | Rede Neural Artificial |
| AR | Modelo Auto-Regressivo |
| ARIMA | Modelo Autorregressivo Integrado de Médias Móveis |
| ARMA | Modelo Autorregressivo e de Médias Móveis |
| BACEN | Banco Central do Brasil |
| CNN | Convolutional Neural-Network |
| EAM | Erro Absoluto Médio |
| EQM | Erro Quadrático Médio |
| ETS | Modelo de Erro, Tendência e Sazonalidade |
| GARCH | Heteroscedasticidade Condicional Auto-Regressiva Generalizada |
| GBM | Gradient Boosting Machines |
| GBRT | Gradient Boost Regression Tree |
| I | Integração |
| IAP | Instituto Ambiental do Paraná |
| KNN | K-Nearest Neighbourhood |
| KPSS | Teste de Hipótese Kwiatkowski-Phillips-Schmidt-Shin |
| LASSO | Least Absolute Shrinkage And Selection Operator |
| LSTM | Long-Short Term Memory |
| MA | Média Móvel |
| REQM | Raiz do Erro Quadrático Médio |
| REQN | Raiz do Erro Quadrático Normalizado |
| RNN | Redes Neurais Recorrentes |
| SES | Suavizamento Exponencial Simples |
| SMAPE | Symetric Mean Absolute Percentage Error |
| SVR | Support Vector Regressor |

**SUMÁRIO**

# 1 INTRODUÇÃO

Uma série temporal é uma sequência de observações medidas ao longo do tempo. Segundo Zagger Zeger et al. (2006), elas naturalmente surgem quando uma população ou fenômeno é monitorado por um período de tempo. Desta forma, as séries temporais são onipresentes onde há dados (Mills, 2019).

Atualmente, com a facilidade da captura, do armazenamento e do processamento de dados proporcionado pelo avanço da computação, as empresas, os governos e os pesquisadores têm a possibilidade de rastrear diversos fenômenos que se organizam como séries temporais. Desta forma, problemas com séries temporais estão difundidos em diversos campos da ciência, como na biologia, geografia, medicina, economia e meteorologia, na maioria das empresas e governos.

No campo médico, alguns fenômenos podem ser monitorados no paciente, como o nível glicêmico no sangue, os batimentos cardíacos, os níveis hormonais, as funções cerebrais, a pressão arterial, o peso, além de medidas auto-reportadas, como o nível de bem-estar, o nível de estresse e a qualidade de vida do paciente. As séries temporais também possibilitam analisar os impactos de intervenções médicas, farmacológicas e ambientais no paciente, através das medidas citadas. Na epidemiologia, os pesquisadores podem estar interessados em como as partículas de poluição podem influenciar em doenças respiratórias, na admissão e mortalidade diária de um hospital e no número de pacientes infectados por uma determinada doença.

Um banco privado, por exemplo, tem em sua rotina diversas previsões que envolvem séries temporais, como a demanda e oferta de crédito, a taxa de juros futura, os preços de ativos, o nível de acessos em seu aplicativo bancário, os fluxos de caixas de empresas tomadoras de créditos e outras atividades. Da mesma maneira, bancos centrais devem prever taxas de desemprego da população, o produto interno bruto do país, a taxa de juros futura, o preço do dólar futuro, níveis de atividades econômicas, finanças públicas etc.

As empresas privadas necessitam prever a flutuação do preço de suas matérias primas, bem como a oferta e demanda de seus produtos, o volume de ligações em seu terminal telefônico, o acesso ao seu *site*, controle de qualidade e a evolução de suas despesas. Os governos devem prever a demanda por necessidades básicas, como energia, saneamento e água potável e a demanda por serviços públicos, como educação, saúde, justiça e segurança, além de questões demográficas, como a evolução populacional de uma região e seus componentes demográficos.

Na meteorologia, as séries temporais estão presentes na observação dos níveis pluviométricos, nos níveis de umidade do ar, na velocidade dos ventos, na temperatura de determinada região, no número de avalanches em determinada localidade, na previsão de condições climáticas em geral, a fim de prever incêndios florestais, tufões, ressacas marítimas e outros desastres naturais.

A análise de séries temporais tem dois propósitos (Cryer e Chan, 2008). O primeiro propósito é uma análise descritiva estatística do fenômeno observado. Na análise descritiva, para (McElroy, 2017), são utilizados conceitos básicos de estatística como média, variância e covariância. O objetivo desta análise é entender e sumarizar as características externas (observáveis) da série temporal. O segundo propósito de uma análise de série temporal é compreender os mecanismos internos de geração de dados, com o objetivo de predizer valores futuros (McElroy, 2017). O foco desta dissertação é tratar do problema de predição da série temporal.

Obter a melhor solução para o problema de série temporal reduz as incertezas e é determinante para que governos tomem as melhores decisões em relação a políticas públicas, empresas tomem melhores decisões nos negócios, ampliando margens de mercado em relação a seus concorrentes e pesquisadores tenham as melhores conclusões em suas pesquisas.

Dado um conjunto de séries temporais, para Talagala et al. (2018), há duas alternativas para lidar com a demanda de todas as previsões deste conjunto. A primeira é utilizar uma metodologia única para prever as próximas observações de todas as séries temporais; a segunda alternativa é utilizar uma metodologia diferente para cada série temporal deste grupo. Entretanto, é improvável que um único método seja o melhor para diversas séries temporais, inviabilizando a primeira estratégia.

Escolher o melhor método para uma única série temporal também não é tão simples. Primeiramente, há diversos modelos que podem ser utilizados para a previsão de uma série temporal, como o Modelo Autorregressivo Integrado de Médias Móveis (ARIMA) (Box et al., 1976), Heteroscedasticidade Condicional Auto-Regressiva Generalizada (GARCH) (Engle e Engle, 1995), Bayesianos (Barber et al., 2011), Exponenciais (Gardner, 2006), Holt-Winters (Chatfield, 1978) e Redes Neurais Artificiais (ANN) (Ho et al., 2002), além de outros modelos. Além da escolha de um modelo, deve-se, em alguns casos, escolher quais hiper-parâmetros este modelo deve ter, como é o caso do número de camadas da rede neural (Metcalfe e Cowpertwait, 2009), se a sazonalidade é aditiva ou multiplicativa no caso do Holt Winter (Hyndman e Athanasopoulos, 2014) ou a ordem dos parâmetros do modelo ARIMA (Hyndman e Khandakar, 2008), multiplicando o número de candidatos para o modelo ótimo.

Para Prudêncio e Ludermir (2004), há três abordagens para solucionar um problema de série temporal. O primeiro é testar através de um *hold-out* diversos tipos de modelos e aplicar aquele que obteve o melhor resultado no teste. O problema desta abordagem, segundo os autores, é que este processo pode ser custoso computacionalmente e demandar muito tempo, devido ao grande número de métodos disponíveis.

A segunda abordagem para o problema de série temporal é contratar um especialista. Apesar de ser uma solução mais eficiente que a primeira abordagem Prudêncio e Ludermir (2004), especialistas podem ser caros financeiramente e nem sempre estar disponíveis. A fim de mitigar os problemas das duas abordagens acima, a terceira solução proposta por Prudêncio e Ludermir (2004) é criar meta-classificadores para séries temporais.

A metodologia de meta-aprendizagem é um subcampo da aprendizagem de máquina (Schmidhuber, 1987). O objetivo de uma metodologia de meta-aprendizado, no contexto das séries temporais, é criar um algoritmo para classificar a melhor solução para o problema de previsão da série temporal de acordo com os meta-dados (características) da respectiva série temporal. Isto é, a partir das características da série temporal de interesse, indicar qual o modelo que terá o menor erro possível ao predizer futuras observações.

Ademais, uma metodologia de meta-aprendizado de séries temporais pode ajudar a responder a provocação elaborada por Hyndman (Ord, 2001), nos comentários da Competição M-3, *"agora temos que identificar por que alguns modelos funcionam enquanto outros não"*. Aqui, o papel da metodologia de meta-aprendizado é identificar quais características que a série temporal deve ter para funcionar em determinados modelos.

## 1.1 DEFINIÇÃO DO PROBLEMA

Com o aumento da presença das séries temporais nos negócios e na ciência, devido a facilidade de captura, registro e armazenamento de dados (Talagala et al. (2018)) aumenta também a importância de previsões de séries temporais. Como apontou Wang et al. (2009), um único modelo de previsão de séries temporais é incapaz de ser o melhor ajuste a todos os dados. A escolha do modelo de previsão de séries temporais deve considerar alguns requisitos (Shukla, 2019).

A acurácia do modelo de previsão é um dos requisitos a ser considerado. Deseja-se selecionar o modelo que minimizará a diferença entre o valor observado e o valor predito[1] (Talagala et al., 2018). Para Shukla (2019), o tempo de treinamento do modelo e de previsão também devem ser critérios de escolha do modelo. No caso das séries temporais, cada nova observação altera a geração de dados internos da série temporal. Deste modo, um novo modelo deve ser treinado após a inserção de uma nova observação na série temporal (Skander Hannachi, 2019) e algumas aplicações como no caso da previsão de preços de ações em bolsa de valores, a resposta do sistema deve ser mais rápida que a do mercado (Li et al., 2016).

Outro fator a ser considerado na escolha do modelo são os as restrições de recursos computacionais. O modelo de previsão que obteve a melhor acurácia na fase de experimento pode não ser suportado pelo sistema no qual ele será integrado. O sistema pode não ter memória e/ou processamento suficiente para suportar o modelo escolhido, além disto, nem todos os modelos estão disponíveis em todas as linguagens de programação.

Segundo James et al. (2014), há um *trade-off* entre a interpretabilidade e a flexibilidade dos modelos. Modelos mais restritivos, como Modelo Autorregressivo e de Médias Móveis (ARMA), são modelos mais interpretáveis que modelos mais flexíveis, como métodos *boosting* (James et al., 2014). No contexto das séries temporais, a interpretabilidade está relacionada em

[1] Presume-se que não há diferença entre o erro devido à superestimação e a subestimação.

como os dados internos são gerados e pode-se estar interessado tanto nos valores não observados quanto na geração interna de dados.

Uma busca exaustiva através de um processo *hold-out*, ou seja, separar a série temporal em duas bases, uma de treino e outra de teste, para verificar a partir da base de treino qual é o melhor modelo para prever a base de teste, pode não ser recomendado. O volume de séries temporais a ser preditas multiplicado pela variedade dos modelos de previsões disponíveis podem aumentar o tempo de execução da tarefa de predição de séries temporais.

## 1.2 DESAFIOS

A proposta de uma metodologia de meta-aprendizado para séries temporais é economizar recursos para o usuário que busca a melhor previsão para sua série temporal, sem que este usuário perca a qualidade na solução final do seu problema. Com isto, o usuário diminui a dependência da contratação de um especialista, economizando tempo e dinheiro, além de não precisar testar diversos modelos, economizando novamente tempo e processamento computacional.

Entretanto, para que a diminuição da dependência do especialista ocorra, a metodologia de meta-aprendizado tem que ter um bom nível de acurácia, para gerar confiança nas suas respostas. A metodologia também deve ser eficiente, ela deve ser capaz de extrair as características da série temporal e predizer qual é o erro de determinados modelos para aquele problema em um tempo menor que o usuário testaria o seu conjunto de modelos.

Por se tratar de um problema de aprendizado de máquina[2], outros dois desafios surgem. O primeiro é como a série temporal é representada para o modelo de regressão, isto é, quais características extrair de cada série temporal que ajudará na previsão do erro das futuras séries temporais. Além de quais características melhor representaram as séries temporais, como elas serão representadas (por categoria ou por valores contínuos). Segundo Bengio et al. (2013), o desempenho de um modelo de aprendizado de máquina é influenciado em como os dados (no presente caso, séries temporais) são representados. Um segundo desafio está na escolha do algoritmo de aprendizagem de máquina.

## 1.3 OBJETIVO

O objetivo desta dissertação é a criação de um meta-regressor, que indique o erro esperado caso a série temporal dada pelo usuário seja modelada por cada um dos seguintes modelos de série temporal: ARIMA, Modelo de Erro, Tendência e Sazonalidade (ETS), *Gradient Boosting Machines* (GBM), Holt-Winters, *Long-Short Term Memory* (LSTM), ANN, Suavizamento Exponencial Simples (SES) e um modelo *naïve*.

[2]No caso desta dissertação, o problema de aprendizado de máquina é do tipo supervisionado.

## 1.4 CONTRIBUIÇÕES

A principal contribuição desta dissertação é a criação de um meta-regressor que prevê qual é o erro esperado dado uma série temporal e um modelo de previsão. A primeira diferença em relação às demais metodologias de meta-aprendizagem é que estes são classificadores. Como contribuições marginais, criou-se este meta-regressor a partir de cem mil séries temporais e iniciou-se a análise com 60 variáveis. A literatura mostra que o classificador com o maior histórico de séries temporais foi de 3.803 séries e o máximo de características preditoras foi de 25 características.

## 1.5 ORGANIZAÇÃO DA DISSERTAÇÃO

Esta dissertação está organizada em seis capítulos. O Capítulo 2, fundamentação teórica apresenta as definições necessárias para o entendimento desta dissertação. O Capítulo 3, o estado-da-arte, apresenta os trabalhos já realizados na área de metodologia de meta-aprendizado em série temporal. O Capítulo 4 apresenta a proposta do metodologia de meta-aprendizado da dissertação e o Capítulo 5 apresenta os resultados e discussões do metodologia de meta-aprendizado. O último Capítulo é a conclusão da dissertação.

# 2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos fundamentais para o entendimento tanto do problema de meta-aprendizagem para séries temporais quanto a sua solução. A entrada do método de meta-aprendizagem de séries temporais são as características das séries temporais e a saída deste método são os erros de previsão de cada série temporal para cada tipo de modelagem abordada, neste caso ARIMA, SES, Holt-Winters, GBM, ANN e LSTM. Com isto, este Capítulo apresenta os conceitos de meta-aprendizagem, série temporal, modelos de séries temporais, *random forest* (o meta-regressor) e como foi calculado o erro de previsão.

## 2.1 META-APRENDIZAGEM

O aprendizado de máquina refere-se a um conjunto de métodos computacionais e estatísticos que utilizam dados históricos observados para melhorar o desempenho de um sistema ou realizar predições acuradas (Mohri et al., 2012).

Para Talagala et al. (2018), John Rice foi um dos pioneiros no uso de métodos de meta-aprendizagem, no qual ele denominou como *problema da seleção de algoritmo*. No conceito de John Rice, o problema de seleção de algoritmo tem quatro componentes, como visto na Figura 2.1. O primeiro componente é o espaço do problema $P$, que representa os conjuntos de dados utilizados no estudo. O componente $F$ representa as características dos conjuntos de dados $P$. O conjunto $A$ representa a lista de algoritmos candidatos que podem ser a solução do problema $P$ e o último componente, $Y$, é a métrica utilizada para avaliar os algoritmos da lista $A$ em relação aos conjuntos de dados $P$.

No caso das séries temporais, o espaço do problema $P$ são as $N$ séries temporais que compõe a base histórica, o espaço $F$ representa as características que podem ser usadas para descrever as séries temporais (elas serão apresentadas mais adiante neste capítulo), os algoritmos do espaço $A$ são os modelos preditivos de séries temporais, como o ARIMA, LSTM, ANN, GBM, Exponenciais, Holt-Winters etc. e o desempenho dos modelos, para este artigo, é o erro de previsão do próximo dado não observado de cada um destes modelos preditivos em relação a cada série temporal, representado pelo *Symetric Mean Absolute Percentage Error* (SMAPE).

A partir da definição dada por John Rice e com o desempenho de cada algoritmo mapeado para cada conjunto de dados, pode-se utilizar um meta-classificador como sugerido por Prudêncio e Ludermir (2004) ou um meta-regressor como proposto por Guerra et al. (2008). No primeiro caso, o método de meta-classificação é definido com um sistema automático de geração de conhecimento que relaciona o desempenho dos modelos preditivos com as características do problema (no caso, dos conjuntos de dados) e este processo é desempenhado por um algoritmo de aprendizado de máquina que indica qual dos modelos preditivos é o melhor indicado, dado um novo conjunto de dados não-observado pelo sistema (Prudêncio e Ludermir, 2004). No caso

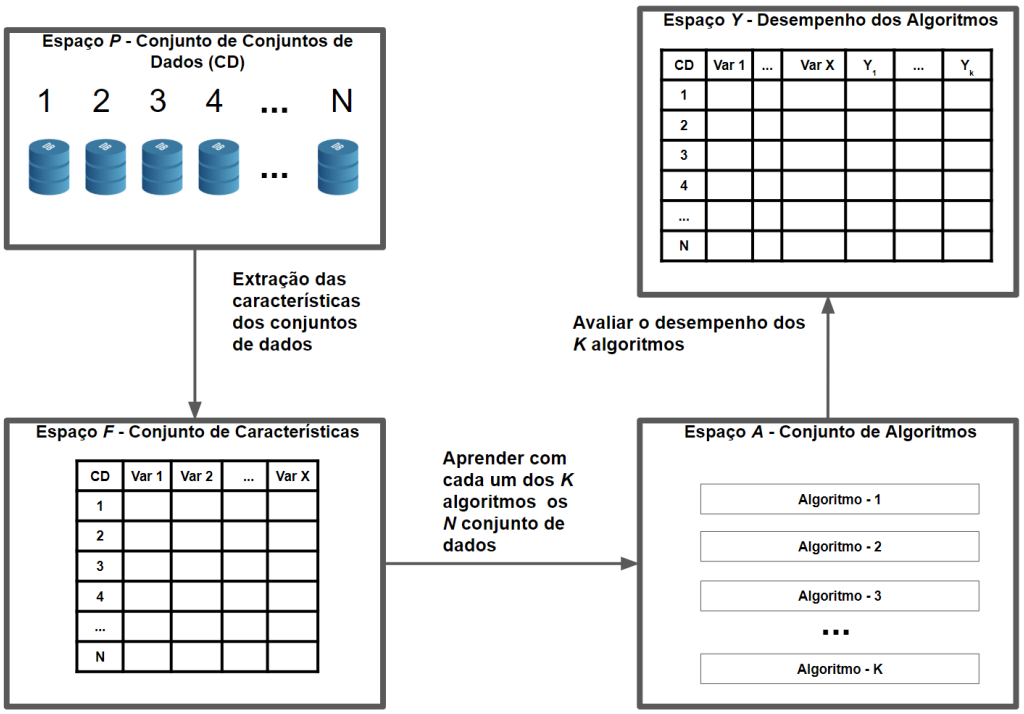


Figura 2.1: Representação do método de meta-aprendizagem

do meta-regressor, o sistema é idêntico ao meta-classificador, contudo a resposta ao invés de ser uma categoria (o melhor modelo preditivo para dado conjunto de dados) é o valor numérico do desempenho de cada modelo preditivo dado um conjunto de dados (Guerra et al., 2008).

## 2.2 SÉRIES TEMPORAIS

Uma série temporal é um caso particular de um processo estocástico (Mills e Markellos, 2008). Knill (2009) define um processo estocástico como um conjunto de variáveis aleatórias, X = {X; $t \in T$}, definidas em um espaço de probabilidade comum, no qual os valores são provenientes de um conjunto comum S (o espaço de conjunto) e estes valores são indexados por um conjunto T, frequentemente definido no conjunto dos naturais ou reais [0, $\infty$) e considerado como tempo, discreto ou contínuo, respectivamente. De uma forma mais objetiva, (Montgomery et al., 2008) define uma série temporal como sendo uma sequência cronológica ou orientada pelo tempo de observações sobre uma variável de interesse.

A Figura 2.2 é um exemplo de uma série temporal. Ela apresenta a cotação semanal do dólar em relação ao real, de acordo com o Banco Central do Brasil (BACEN), entre janeiro e abril de 2020. Ou seja, cada ponto no gráfico representa uma semana entre o período coletado do fenômeno de interesse (cotação do dólar) indexado por um tempo $t$ (cada semana).

A Figura 2.3 é um segundo exemplo de série temporal. Ela apresenta o número de partes por milhão de monóxido de carbono por dia coletado pelo Instituto Ambiental do Paraná (IAP) na cidade de Curitiba, entre os períodos de 15 de março de 2020 até 26 de março de 2020.

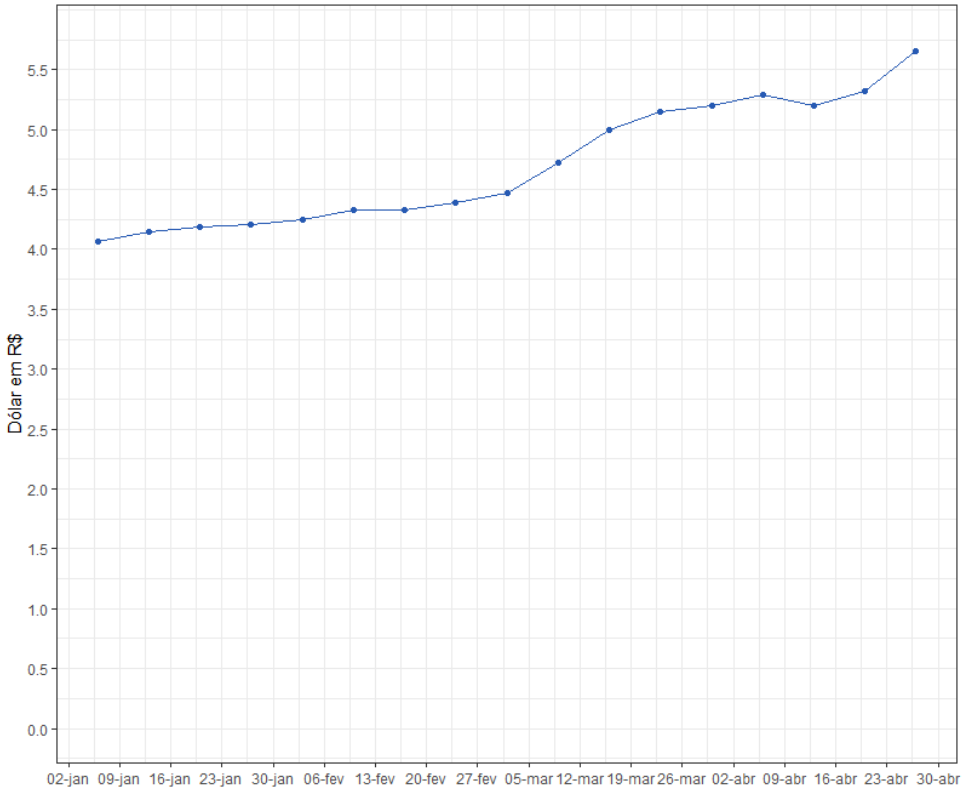


Figura 2.2: Cotação diária do dólar em relação ao real - De jan/2020 até abr/2020 - Fonte: BACEN

A variável de interesse é o número de partes de monóxido de carbono na cidade de Curitiba e o índice é cada dia de coleta.

Uma maneira de se contextualizar os dados observados dentro de um fenômeno é através de uma análise descritiva dos dados (Andersen e Risør, 2014). Esta análise descritiva resume os dados observados em uma maneira mais simples e objetiva (Trochim, 2020). Segundo este autor, ela é feita através de gráficos, medidas centrais (como média, mediana e moda) e medidas de dispersão (como desvio-padrão e variância).

No caso da Figura 2.2, a média da cotação do dólar no período foi de R$4,70 e o desvio-padrão foi de 50 centavos. O dólar se valorizou frente ao real em 30%, em um contexto que segundo Valim (2020) se deve à fuga de capitais estrangeiros por conta da baixa de juros brasileiros, turbulências políticas e além da crise da saúde e da economia provocado pelo Corona vírus. No caso da poluição em Curitiba, o número de partes por milhão de monóxido de carbono caiu pela metade desde a publicação de calamidade pública no Estado do Paraná, ocorrida no dia 19 de março de 2020 (Brodbeck, 2020).

Uma segunda maneira de contextualização dos dados dentro de um fenômeno é através de um modelo estatístico. O modelo estatístico representa o processo de geração dos dados observados. A identificação do processo de geração de dados pode ter dois objetivos: estimação ou previsão (Adèr et al., 2008). Na estimação, segundo o autor, o objetivo é não só identificar se existe a associação de duas variáveis ou mais, assim como o grau de associação entre estas

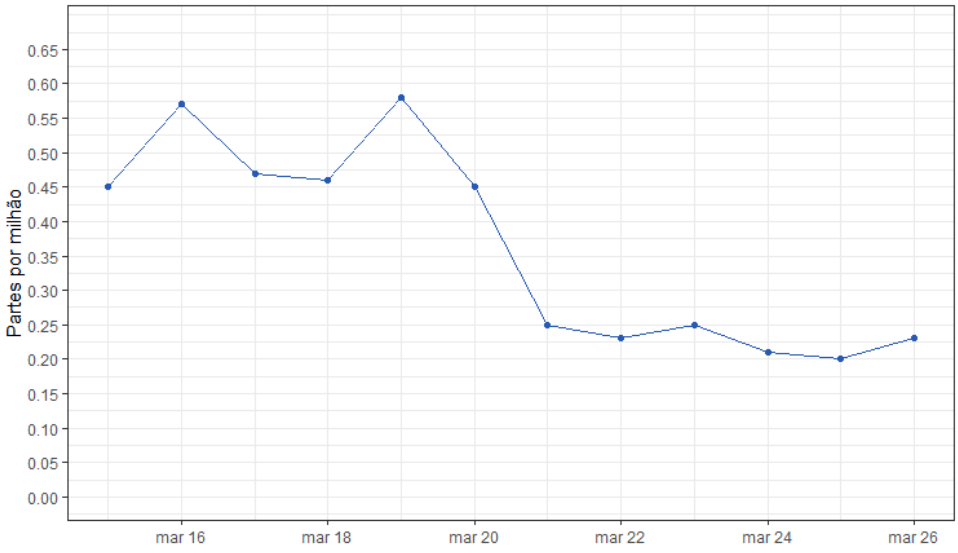


Figura 2.3: Partes por milhão de monóxido de carbono na cidade de Curitiba - De 15/03/2020 até 26/03/2020 - Fonte: IAP

variáveis. E na previsão, deseja-se, através do modelo estatístico obter um valor futuro (Shmueli, 2011).

No exemplo do dólar, uma empresa poderia se beneficiar de uma previsão da moeda, pois segundo (Brigham e Ehrhardt, 2016), a variação cambial afeta preços (tanto de matéria prima quanto de produtos finais) e o comércio (oferta e demanda). Além disto, para (Zani et al., 2010), a variação cambial afeta as demonstrações de resultado não só de empresas que estão na cadeia global de produção (importando ou exportando produtos), bem como aquelas com empréstimo em moeda estrangeira. Para estes autores, a variação cambial afeta o fluxo de caixa e o lucro das empresas, como os demais indicadores financeiros da empresa (balanço e demonstrações de resultado). Para o governo, prever o dólar é importante, pois esta variável segundo (Barbosa-Filho et al., 2011), impacta no crescimento econômico do país, para (Marconi, 2011) impacta na industrialização e composição setorial econômica do país, (Gala et al., 2011) cita a influência da moeda na poupança interna e externa do país e para (Teixeira e Mattos, 2010), a variação cambial afeta na arrecadação municipal.

A previsão de valores da poluição de uma localidade pode auxiliar na previsão meteorológica (li Wang et al., 2004), na privação social (Namdeo e Stringer, 2008), no status socio-econômico de uma população (Neidell, 2004), no nível alérgico e de bronquite em crianças (Janssen et al., 2003), na mortalidade pós-natal (Woodruff et al., 1997), na mortalidade entre adultos (Kan e Chen, 2003) e na demanda hospitalar (Atkinson et al., 2014).

### 2.2.1 Sazonalidade e Tendência

A sazonalidade e a tendência são duas fontes de variabilidade de uma série temporal. A sazonalidade é um padrão que se repete a cada determinado período de tempo na série temporal,

atrelado ao calendário (Rose et al., 2002). A tendência é padrão de crescimento ou decrescimento de uma variável dentro de um período de tempo (Montgomery et al., 2008). A Figura 2.2 apresenta uma série temporal com tendência e a Figura 2.4 apresenta uma sazonalidade e uma tendência de alta.

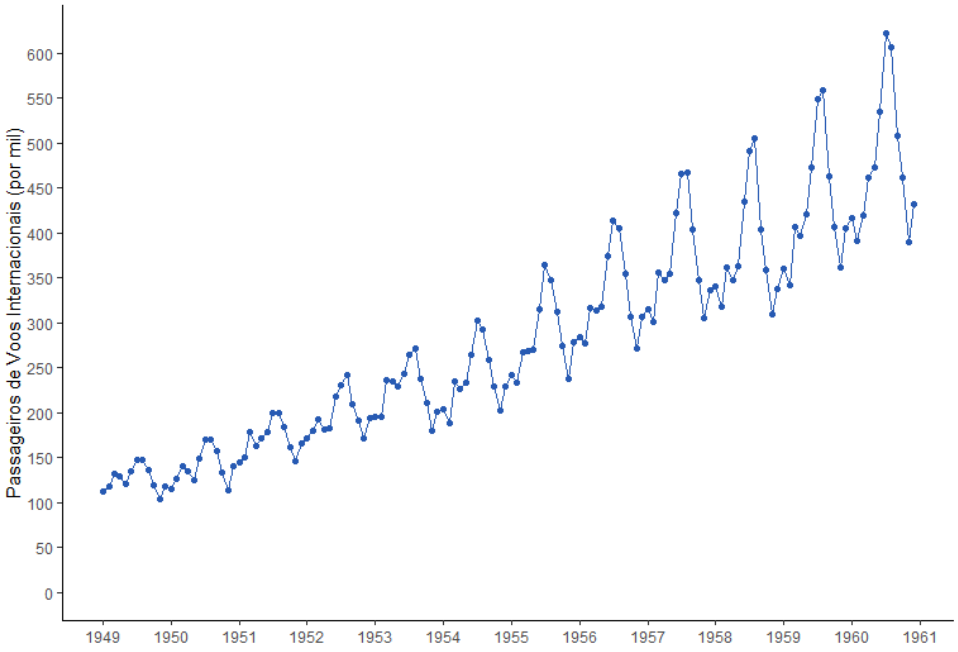


Figura 2.4: Número de Passageiros de Voos Internacionais Por Mês - Entre 1949 e 1960 - Fonte: R Core Team (2020)

## 2.3 MODELO NAÏVE

O modelo mais simples de predição de séries temporais é o modelo *naïve*, (Shmueli e Lichtendahl, 2016). Segundo os autores, a previsão de um modelo naïve é igual ao valor mais recente da série, como apresentado na equação 2.1, no qual o valor $x_{t+1}$ é o valor a ser predito e $x_t$ é o valor mais recente.

$$x_{t+1} = x_t \tag{2.1}$$

Caso a série temporal seja sazonal, o valor da previsão será aquele referente a última observação do período idêntico. A equação 2.2 apresenta o cálculo de previsão de uma série temporal com sazonalidade semanal. Como a sazonalidade semanal tem uma cobertura de sete observações (uma para cada dia da semana), a predição do valor $x_{t+1}$ é igual a sete observações atrás, $x_{t-7}$ (Shmueli e Lichtendahl, 2016).

$$x_{t+1} = x_{t-7} \tag{2.2}$$

O pressuposto do modelo *naïve*, segundo Shmueli e Lichtendahl (2016), é que o próximo valor tende a ser mais próximo do valor anterior do que dos demais. Para os autores, este modelo tem dois propósitos. O primeiro, é ser usado como uma predição, pela sua simplicidade e fácil interpretabilidade. E o segundo propósito é ser usado como uma referência para os demais modelos.

## 2.4 MODELOS EXPONENCIAIS

Os métodos de suavização exponencial são uma família de modelos de previsão de séries temporais. Eles usam médias ponderadas de observações anteriores para prever novos valores (Daitan, 2019). De acordo com os autores, a lógica destes modelos é dar mais importância aos valores mais recentes da série e à medida que as observações envelhecem, a importância destes valores fica exponencialmente menor.

### 2.4.1 SES

O SES é um modelo é flexível, de fácil automação, computacionalmente barato e tem um bom desempenho, de acordo com Shmueli e Lichtendahl (2016). De acordo com estes autores, ele é um modelo similar a médias móveis, contudo, ao invés de ele fazer médias aritméticas simples, este modelo utiliza médias ponderadas de todos os valores passados.

Para predizer a observação $x_{t+1}$ com o modelo SES, deve-se utilizar a equação 2.3:

$$x_{t+1} = \alpha x_t + \alpha(1-\alpha)x_{t-2} + \alpha(1-\alpha)^2 x_{t-3}... \tag{2.3}$$

em que $\alpha$ é uma constante entre 0 e 1, conhecida como coeficiente de suavização. Este coeficiente controla a influência de cada observação na resposta final e decai exponencialmente a cada observação passada. O valor de $x_{t+1}$ é a observação que deseja-se predizer, e os demais $x$'s são as observações subsequentemente passadas de $x_{t+1}$. A formulação acima exibe a suavização exponencial como uma média ponderada de todas as observações anteriores, com pesos decrescentes exponencialmente (Shmueli e Lichtendahl, 2016). Quanto maior o valor de $\alpha$, mais importante são as variáveis mais recentes. Para Hyndman e Athanasopoulos (2014), este método é ideal para séries temporais sem tendência e sazonalidade.

### 2.4.2 Holt-Winters

O modelo Holt-Winters é uma extensão do modelo SES, criada para lidar com tendência e sazonalidade (Hyndman e Athanasopoulos, 2014). De acordo com os autores, o modelo de previsão do Holt-Winters envolve três equações, sendo a primeira conhecida como equação de nível e a segunda equação conhecida como equação de tendência e a última equação é conhecida

como a equação de previsão. A equação 2.4 é usada para a previsão da *h-ésima* observação pelo método de Holt-Winters.

$$\begin{aligned} l_t &= \alpha x_t + (1-\alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta(l_t - l_{t-1}) + (1-\beta) b_{t-1} \\ x_{t+h} &= l_t + h b_t \end{aligned} \tag{2.4}$$

O valor de $\beta$ na equação 2.4 é uma taxa de suavização da tendência e é uma constante entre 0 e 1. O $\alpha$ representa nesta equação a mesma função que na equação 2.3. O resultado da equação $b_t$ e $l_t$ indicam a inclinação (tendência) e nível no tempo $t$, respectivamente (Hyndman e Athanasopoulos, 2014).

### 2.4.3 Sazonalidade no Modelo Holt-Winters

Para prever a sazonalidade de uma série temporal, o método de Holt-Winters estima três equações[1]. A primeira equação é a de nível, a segunda equação refere-se à tendência e a terceira equação é conhecida como equação de sazonalidade. Este método utiliza duas variações para estimar a sazonalidade. A primeira variação é conhecida como sazonalidade aditiva e a segunda variação é a sazonalidade multiplicativa (Hyndman e Athanasopoulos, 2014). Segundo os autores, a sazonalidade aditiva ocorre quando quando a sazonalidade é constante ao longo da série e a sazonalidade multiplicativa ocorre quando a sazonalidade varia proporcionalmente de acordo com o nível da série (nível da série é o resultado da equação $\ell_t$). Para prever a observação $x_t + 1$, com a variante sazonalidade aditiva no modelo Holt-Winters, usa-se a equação 2.5 (Hyndman e Athanasopoulos, 2014).

$$\begin{aligned} x_{t+1} &= \ell_t + h b_t + s_{t+h-m(k+1)} \\ \ell_t &= \alpha(x_{t-1} - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^(\ell_{t-1} - \ell_{t-2}) + (1-\beta^) b_{t-2} \\ s_t &= \gamma(x_t - \ell_{t-2} - b_{t-2}) + (1-\gamma) s_{t-m} \end{aligned} \tag{2.5}$$

Segundo Hyndman e Athanasopoulos (2014), a equação sazonal mostra uma média ponderada entre o índice sazonal atual e o índice sazonal da mesma temporada do ano passado

[1] A quarta equação é o resultado da previsão, que é a soma das três equações no caso da sazonalidade aditiva e um produto no caso da multiplicativa.

(ou seja, $m$ períodos de tempo atrás). A constante $\gamma$ varia entre 0 e 1. Para prever a observação $x_{t+1}$ com sazonalidade multiplicativa, utiliza-se a seguinte equação:

$$\begin{aligned} x_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\ \ell_t &= \alpha\frac{x_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^(\ell_t - \ell_{t-1}) + (1-\beta^)b_{t-1} \\ s_t &= \gamma\frac{x_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m} \end{aligned} \tag{2.6}$$

Nas equações 2.5 e 2.6, $s$ representa a equação sazonal, o $\gamma$ é a taxa de suavização do fator sazonal e o $m$ é a periodicidade da sazonalidade. Os demais símbolos nestas equações têm as mesmas funções que na equação 2.4. Para Shmueli e Lichtendahl (2016), no método aditivo, o componente sazonal é expresso em termos absolutos na escala das séries observadas e a equação de nível é ajustada subtraindo o componente sazonal. No método multiplicativo, o componente sazonal é expresso em termos relativos e a série de nível é ajustada pela divisão do componente sazonal.

### 2.4.4 ETS

Além da tendência e sazonalidade, outro componente pode ser adicionado nos modelos exponenciais: o erro. Desta forma, o modelo exponencial fica conhecido como ETS, acrônimo do inglês *Error, Trend and Seasonality* (Daitan, 2019) ou Erro, Tendência e Sazonalidade na tradução em português. Há dois tipos de erros nos modelos ETS, aditiva e multiplicativa. Segundo Hyndman e Athanasopoulos (2014), no erro aditivo, assume-se que os erros do treinamento têm distribuição normal e são dados por:

$$\varepsilon_t = y_t - \ell_{t-1} - b_{t-1} \tag{2.7}$$

Com erros aditivos, a previsão da observação $x_{t+1}$ no modelo de ETS é dada por:

$$\begin{aligned} x_{t+1} &= \ell_t + b_t + \varepsilon_{t+1} \\ \ell_t &= \ell_{t-1} + b_{t-1} + \alpha\varepsilon_{t+1} \\ b_t &= b_{t-1} + \beta\varepsilon_{t+1}, \end{aligned} \tag{2.8}$$

No caso de os erros serem multiplicativos, assume-se que eles também têm distribuição normal e seguem a equação 2.9 (Hyndman e Athanasopoulos, 2014).

$$\varepsilon_t = \frac{y_t - (\ell_{t-1} + b_{t-1})}{(\ell_{t-1} + b_{t-1})} \tag{2.9}$$

Neste caso, a observação $x_{t+1}$ é dada por:

$$\begin{aligned} x_{t+1} &= (\ell_t + b_t)(1 + \varepsilon_{t+1}) \\ \ell_t &= (\ell_{t-1} + b_{t-1})(1 + \alpha\varepsilon_{t+1}) \\ b_t &= b_{t-1} + \beta(\ell_{t-1} + b_{t-1})\varepsilon_{t+1} \end{aligned} \tag{2.10}$$

Por ser uma extensão dos modelos exponenciais, os termos das equações ETS são os mesmos das equações Holt-Winters, com a adição do $\epsilon$, que significa o erro da predição das observações anteriores. O modelo ETS é representado por ETS(E,T,S), no qual E representa se o erro é aditivo ou multiplicativo, T se a tendência é aditiva, multiplicativa ou nula e S, se a tendência é aditiva, multiplicativa ou nula. Desta forma, há 18 variações do modelo ETS para a mesma série temporal.

## 2.5 ARIMA

O modelo ARIMA é o modelo mais utilizados na modelagem de séries temporais (Hyndman e Athanasopoulos, 2014). O objetivo do modelo ARIMA é descrever a autocorrelação da série temporal. Este modelo é dividido em três partes: a autorregressão, a integração e a média móvel.

### 2.5.1 Autocorrelação e Autocorrelação Parcial

O coeficiente de correlação entre duas observações individuais de uma mesma série temporal é chamada de autocorrelação. Ela mede a dependência linear entre uma observação no índice $t$ para com os seus vizinhos da mesma série (Venables e Ripley, 2002), com uma diferença $p$ entre o índice destas observações. Segundo os autores, o coeficiente varia de -1 até 1[2], sendo uma autocorrelação de 1, uma correlação perfeita, -1 uma anti-correlação perfeita. Caso este coeficiente for zero, não há autocorrelação entre as duas observações. A autocorrelação é dado por, na qual $y_t$ é a variável de interesse:

$$Corr(y_t, y_{t-p}) = E[(y_t - \mu)(y_{t-p} - \mu)], p = 1, 2, ... \tag{2.11}$$

O conjunto de estatísticas de autocorrelações da série temporal como um todo é a função de autocorrelação. Para Tattar et al. (2017), ela é uma importante métrica para entender a natureza de dependência interna da série temporal e é dado pela equação:

$$c_k = T^{-1} \sum_{t=1}^{T-p} E[(y_t - \mu)(y_{t-p} - \mu)], p = 1, 2, ... \tag{2.12}$$

[2]Há outras formas de calcular uma autocorrelação. Especificamente, a autocorrelação utilizada neste trabalho tem o intervalo -1 e 1.

Na equação 2.12 e 2.11, $c_k$ é a média das autocorrelações de todos os $y_t$ (observações referentes ao tempo de referência) com $y_{t-p}$ (observações referentes aos tempos que estão sendo comparados) e $T$ é o número de relações calculadas na série temporal e $\mu$ é a média da série temporal. Em ambas as equações, caso $p$ (diferença entre o tempo de referência e o tempo a ser calculado) seja igual a zero, por definição o resultado é igual a 1. Na Figura 2.5[3], nota-se que o grau de dependência médio das observações $y_t$ em relação às observações $y_{t-1}$ é de 0,98 e das observações $y_{t-2}$ é de 0,95 e assim sucessivamente.

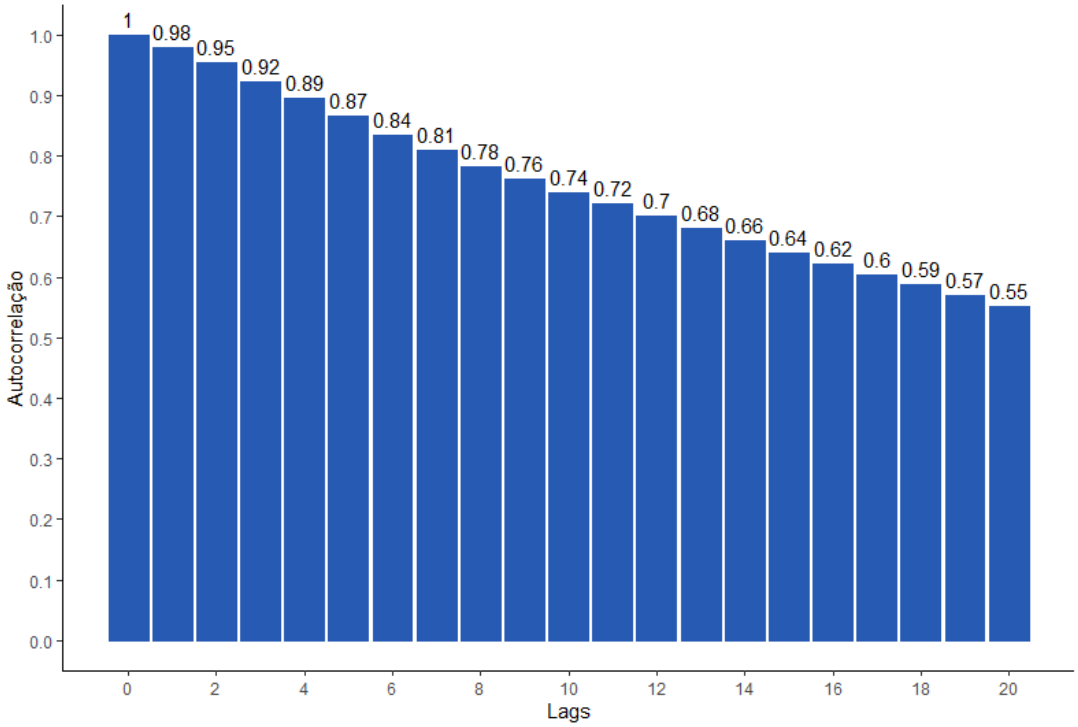


Figura 2.5: Função de Autocorrelação da Cotação do Dólar em Relação ao Real - De jan/2020 até abr/2020

A função de autocorrelação parcial calcula a correlação entre os resíduos das observações, isto é, esta função calcula a dependência de uma observação sobre uma segunda observação que ainda não foi explicada pelas observações entre elas. Por exemplo, a autocorrelação parcial da observação $y_t$ em relação a observação $y_{t-2}$ exclui a influência que $y_{t-1}$ pode ter sobre $y_t$, pois ela já foi explicada na autocorrelação parcial entre $y_t$ e $y_{t-1}$ (Mills e Markellos, 2008). Para os autores, a $p$-ésima autocorrelação parcial é o coeficiente $\phi_{pp}$, dado pela autorregressão linear:

$$x_t = \phi_{p1}x_{t1} + \phi_{p2}x_{t-2} + ... + \phi_{pp}x_{t-p} + \epsilon_t \tag{2.13}$$

Na equação 2.13, cada coeficiente $\phi$ é a autocorrelação parcial do $x_t$ de interesse e o $\epsilon_t$ é o erro da autorregressão (Mills e Markellos, 2008). A autocorrelação para $p$ igual a zero é 1 e para $p$ igual a 1 é similar a autocorrelação de mesma ordem. Para McElroy (2017), a função de autocorrelação parcial ajuda a entender o comportamento de geração de dados interno da série temporal. A Figura 2.6 é a função de autocorrelação parcial da cotação do dólar.

[3]O nome do eixo-x é o termo em inglês *lags*, que significa a diferença entre o tempo $t$ e o tempo $p$.

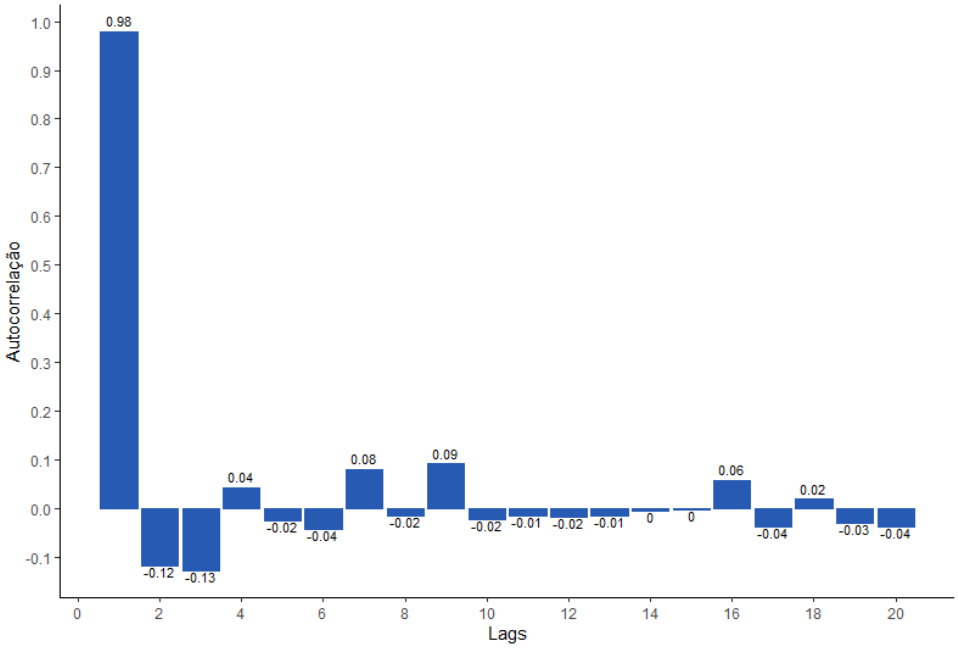


Figura 2.6: Função de Autocorrelação Parcial da Cotação do Dólar em Relação ao Real - De jan/2020 até abr/2020

A equação 2.13 é conhecida como autorregressão de ordem $p$ ou simplesmente $AR(p)$, pois a variável de interesse é uma combinação linear dos seus próprios $p$ termos passados, indicando que a regressão acontece internamente na variável (Hyndman e Athanasopoulos, 2014).

### 2.5.2 Modelos de Médias Móveis

Outro componente dos modelos ARIMA é a possibilidade de adição de modelos de médias móveis. Os modelos de médias móveis usam os erros de predições dos modelos autorregressivos em uma regressão de $x_t$ Mills (2019).

$$x_t = \mu + \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + ... + \theta_q\epsilon_{t-q} \tag{2.14}$$

A equação 2.14 é conhecida como modelo de média móvel de ordem $q$ ou $MA(q)$, pois são utilizados os erros de $q$ equações autorregressoras (Montgomery et al., 2008). Neste caso, $x_t$ pode ser interpretado como a média móveis dos erros de predição passados ponderados. Na equação 2.14, $\mu$ é o intercepto da autoregressão, $\epsilon$ representa o valor dos erros das predições dos modelos autoregressores (no tempo $t$) e $\theta$ representa os coeficientes de inclinação (coeficiente angular) dos erros de predição em relação à média móvel.

### 2.5.3 Integração

Os modelos ARIMA são uma generalização dos modelos ARMA. Um dos pressupostos dos modelos ARIMA é a obrigação que a série temporal seja estacionária. Uma série temporal estacionária se desenvolve em torno de uma média e variância constante, Mills e Markellos

(2008). A ideia da estacionaridade é que a distribuição de probabilidade da série temporal não se altere ao longo do tempo Cryer e Chan (2008).

Para transformar uma série temporal não-estacionária em estacionária calcula-se a diferença entre as observações consecutivas (Hyndman e Athanasopoulos, 2014):

$$x'_t = x_t - x_{t-1} \tag{2.15}$$

A equação 2.15 é conhecida como diferenciação de primeira ordem e ela cria uma nova série temporal, com uma observação a menos, dado que é impossível diferenciar a primeira observação. O processo inverso da diferenciação é a integração, neste caso a integração seria de primeira ordem, ou $I(1)$ Cryer e Chan (2008).

Em alguns casos, a série temporal diferenciada em primeira ordem pode não ser estacionária e mais um processo de diferenciação precisa ser feito, criando uma série temporal de segunda ordem Cryer e Chan (2008). Esta diferenciação é obtido por:

$$x''_t = x'_t - x'_{t-1} = (x_t - x_{t-1}) - (x_{t-}1 - x_{t-2}) = x_t - 2x_{t-1} + x_{t-2} \tag{2.16}$$

No caso da equação 2.16, a nova série temporal tem duas observações a menos que a original e pode ser interpretada como a mudança na mudança. Na prática, quase nunca é necessário ir além da diferenciação de segunda ordem (Hyndman e Athanasopoulos, 2014).

Se a série temporal apresentar sazonalidade, a primeira diferenciação é dado por,

$$x'_t = x_t - x_{t-m} \tag{2.17}$$

no qual o índice $m$ é o tamanho da sazonalidade (Hyndman e Athanasopoulos, 2014). Por exemplo, se a série temporal tem uma sazonalidade semanal, $m = 7$. A diferenciação sazonal de segunda ordem de uma série temporal é dado por:

$$x''_t = x'_t - x'_{t-1} = (x_t - x_{t-m}) - (x_{t-1} - x_{t-m-1}) = x_t - x_{t-1} - x_{t-m} + x_{t-m-1} \tag{2.18}$$

O teste de hipótese Kwiatkowski-Phillips-Schmidt-Shin (KPSS) verificar se a série temporal necessita ser diferenciada (Kwiatkowski et al., 1992). Neste teste, a hipótese nula é que os dados são estacionários.

### 2.5.4 Estimação do Modelo ARIMA

O modelo ARIMA, como o nome indica, é a combinação de modelos autorregressivos e modelos de médias móveis, ambos integrados. O modelo completo pode ser escrito como (Hyndman e Athanasopoulos, 2014):

$$y'_t = \mu + \phi_1 y'_{t-1} + \ldots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_{t-q} + \epsilon_t \tag{2.19}$$

Na equação 2.19, $y'_t$ indica que a ordem de integração é um, mas ela pode ser integrada mais de uma vez, caso seja necessário e se a série temporal for estacionária, não há necessidade de integração. Este modelo é representado por $ARIMA(p, d, q)$, no qual $p$, $d$ e $q$ são as ordens de autorregressão, integração e média móvel, respectivamente. Segundo Bisgaard e Kulahci (2011), estas ordens dificilmente são maiores que 2.

Uma vez identificado a ordem do modelo desejado, deve-se estimar os parâmetros $\mu$, $\phi_1$, ..., $\phi_p$, $\theta_1$, ..., $\theta_q$. Há duas formas de estimação destes parâmetros. O primeiro método é pela máxima verossimilhança, técnica que encontra os parâmetro que maximiza a probabilidade de obter os dados observados. O segundo método é o método dos mínimos quadrados, função que minimiza a soma dos quadrados do erro de previsão (Hyndman e Athanasopoulos, 2014).

## 2.6 REDES NEURAIS ARTIFICIAIS

Uma Rede Neural Artificial (RNA) é um algoritmo de processamento de dados que tem como objetivo imitar como o cérebro humano processa os dados (estímulos externos) (Haykin, 2007). Ela é composta por pelo menos três tipos de camadas de neurônios. O primeiro tipo de camada de neurônios, conhecido como camada de entrada, é a camada responsável pela entrada dos dados que a rede-neural trabalhará a partir de então. O segundo tipo de camada, denominada de camada intermediária ou escondida, é responsável pelo processamento dos dados e dependendo da aplicação pode ser composta por mais de uma camada de neurônios. A última camada de neurônios, ou a camada de saída, é onde se situa a resposta que deseja-se obter com a rede neural. Os neurônios da camada de entrada representam as variáveis independentes, os neurônios de saída representam a resposta que o usuário deseja e os neurônios das camadas intermediárias processam os dados. Não há limites máximo para o número de neurônios (Haykin, 2007). Na Figura 2.7, é uma generalização de uma aplicação de redes neurais para séries temporais, no qual a camada de entrada recebe a observação mais recente $x_t$ e quantas outras observações forem necessárias, até $k$. A camada intermediária processa os dados e a camada de saída prevê a próxima observação $x_{t+1}$. Os neurônios são representados pelos círculos.

A conexão entre as camadas ocorre através das sinapses entre os neurônios. Na Figura 2.7, as sinapses são representadas pelas setas entre os neurônios. Cada neurônio da camada atual tem uma sinapse para se conectar com cada neurônio da próxima camada (Siami Namini e Siami Namin, 2018). Segundo os autores, cada sinapse contém um peso (ou parâmetro) que deve ser estimado. O peso é multiplicado pelo valor de saída do neurônio anterior e será um dos componentes do somatório do valor de entrada do neurônio de destino.

Na Figura 2.8, o neurônio é composto por $N$ terminais de entrada que enviam um valor $a$ para este neurônio, representado por $a_1$, $a_2$, ..., $a_N$. Estes valores de saídas dos terminais são

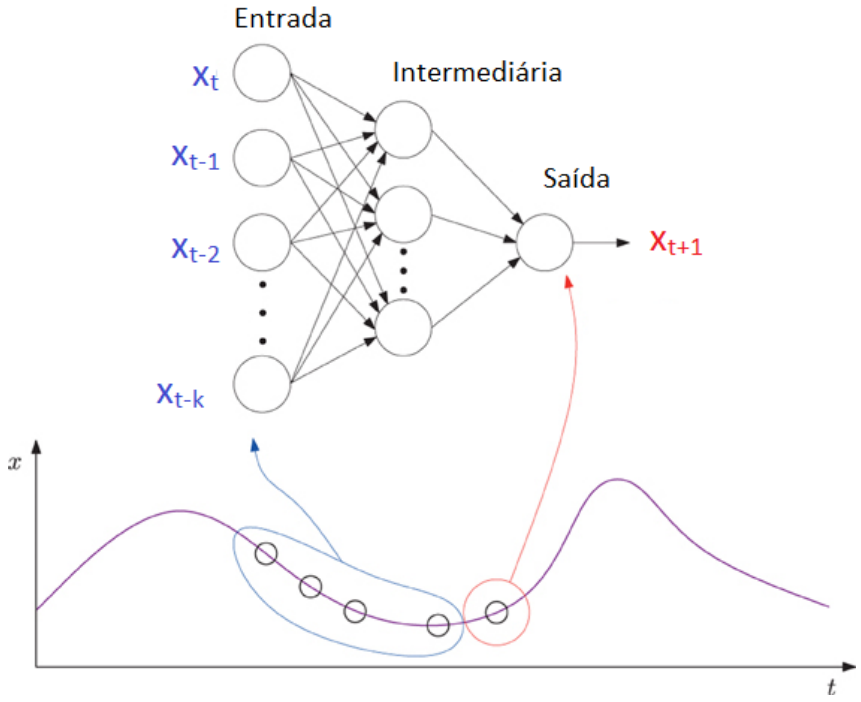


Figura 2.7: Generalização de Uma Rede Neural Para Séries Temporais. Fonte: Yoo et al. (2014)

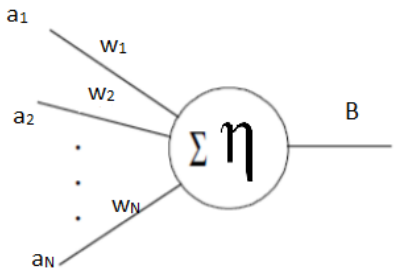


Figura 2.8: Representação de um neurônio.

multiplicados pelos pesos $w$, representados por $w_1$, $w_2$, ..., $w_N$. O valor de entrada da função de ativação deste neurônio $\eta$ é dado pelo somatório definido pela equação:

$$\sum_{i=1}^{N} a_i w_i \tag{2.20}$$

A função de ativação é responsável pelo valor de saída do neurônio do qual ela pertence (Siami Namini e Siami Namin, 2018) e será o valor de entrada da próxima camada ou a resposta final da rede neural. A importância desta função de ativação é injetar uma não-linearidade na rede neural, caso contrário, a rede-neural seria uma combinação linear. Este valor é representado por $B$, na Figura 2.8. O valor de saída representa se o neurônio está ligado (ativado) ou desligado (desativado). O processo de aprendizagem da rede neural ocorre através da estimação dos pesos da rede neural. Para a estimação dos pesos da rede neural é usado um algoritmo, como por

exemplo o *backpropagation* (Nielsen, 2018). Este algoritmo calcula o gradiente descendente da função custo do conjunto de pesos da rede neural .

## 2.7 LSTM

Uma das características do cérebro humano é a capacidade de memória, atividade que auxilia no aprendizado. Em redes neurais tradicionais, como a apresentada na Figura 2.7, assume-se que há independência entre todos os dados de entrada (Britz, 2016). Por exemplo, a observação $x_t$ não tem correlação com a observação $x_{t-1}$. Contudo, como indicam as Figuras 2.12 e 2.13, no caso das séries temporais, pode haver dependência entre as observações de uma mesma série temporal. As Redes Neurais Recorrentes (RNN) computam a previsão de uma observação com base em informações computadas no passado, que estão gravadas em uma *memória* na RNN (Britz, 2016). Para Kang (2017), por conter esta característica de memória (dependência entre as observações), as RNN tornam-se aptas a processar e predizer séries temporais.

A LSTM é um tipo especial de RNN, capaz de aprender dependências de longo-prazo (Olah, 2015). Ela foi proposta por Hochreiter e Schmidhuber (1997), para evitar o problema que as RNN's tinham do desaparecimento do gradiente (Zhou et al., 2019). A Figura 2.9 mostra uma representação de uma aplicação do LSTM em uma série temporal, para a previsão da observação $ht$, a partir do vetor $x(t)$, composto pelas observações anteriores ao momento $t$ da série temporal e a parte em verde é o LSTM em si, no momento $t$, conhecido como $C_t$.

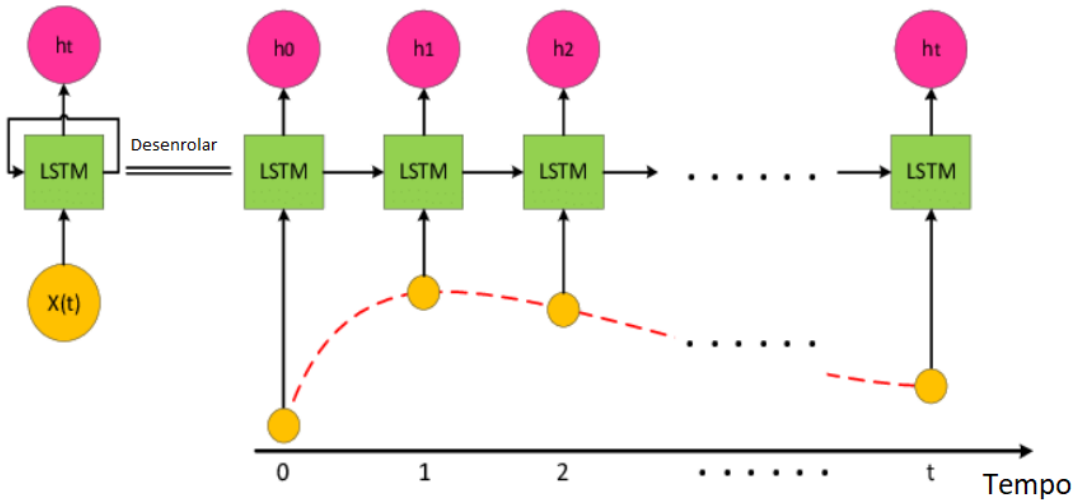


Figura 2.9: Representação de um LSTM para Séries Temporais. Fonte: Zhou et al. (2019)

Para Olah (2015), a ideia central do LSTM é o estado da célula, ou a linha horizontal percorrendo a parte superior do diagrama, como visto na Figura 2.10. O estado da célula $C_{t-1}$ percorre a célula $C_t$ e é transferido para a próxima célula, com poucas interações na $C_t$. Esta célula tem a capacidade de remover ou adicionar informações oriundas da $C_{t-1}$ através de estruturas chamadas portas. Estas portas deixam as informações passarem opcionalmente, sendo compostos de uma camada sigmoide e uma multiplicação. O resultado da sigmoide varia no

intervalo $[0, 1]$ e controla o percentual de informações que vai interagir com o estado da célula $C_{t-1}$.

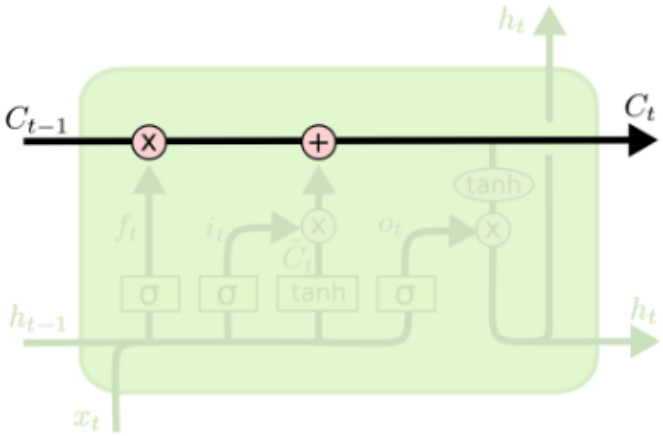


Figura 2.10: Representação da célula $C_t$ de uma LSTM. Fonte: Olah (2015)

O primeiro passo da LSTM é conhecido como *forget gate* e está representado pela 2.11. O objetivo desta porta é remover as informações inúteis de observações passadas. Ela tem duas entradas, $h_{t-1}$ e $x_t$ e uma saída, $f_t$. Na equação 2.21, $h_{t-1}$ representa a observação passada e o $x_t$ é o vetor de observações referentes ao estado presente, $W_f$ é o peso multiplicativo (como ocorre nas ANN's), $b_f$ é o viés e $\sigma$ representa a função sigmoide (Mittal, 2019).

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.21}$$

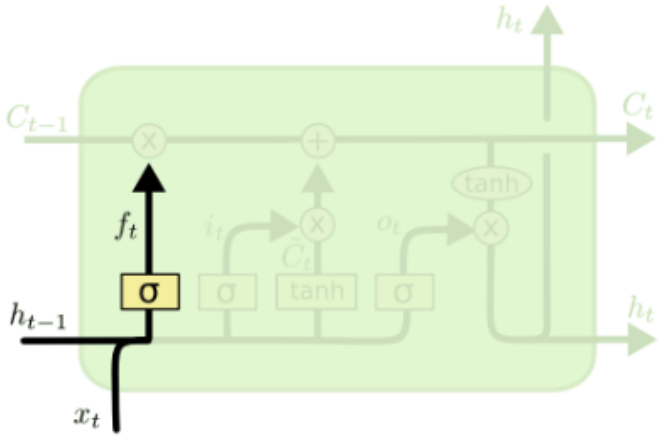


Figura 2.11: Representação do *forget gate* de um LSTM. Fonte: Olah (2015)

O segundo componente da LSTM é conhecido como *input gate* e é demonstrado na Figura 2.12. A responsabilidade desta porta é atualizar o estado da célula atual. Primeiramente, passa-se, novamente, a observação anterior e o vetor de observações referentes ao estado presente em uma função sigmoide, que vai decidir o percentual de informação que será atualizado. Este

processo é apresentado na equação 2.22, no qual $W_i$ é o peso desta sinapse, $b_i$ é o viés desta relação, $h_{t-1}$ e $x_t$ têm o mesmo significado que na equação 2.21 (Phi, 2020).

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.22}$$

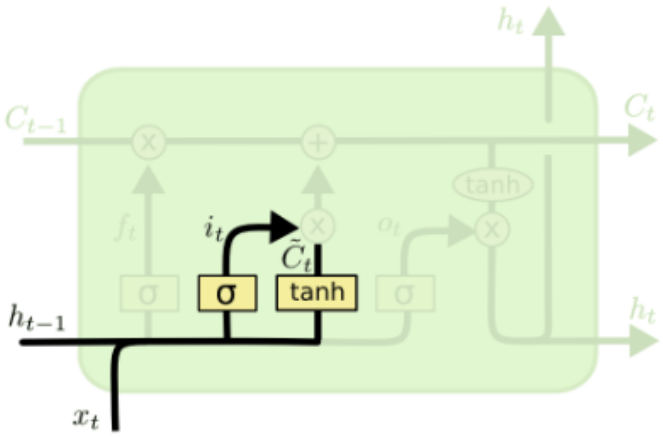


Figura 2.12: Representação do *input gate* de um LSTM. Fonte: Olah (2015)

Segundo Phi (2020), a saída do *input gate* é multiplicada pela saída do estado da célula atual, $\tilde{C}_t$, dado pela equação 2.23. Neste processo, uma função tangente hiperbólica, que tem como entrada os pesos da sinapse multiplicado pela observação anterior ($h_{t-1}$) e o vetor de observações referentes ao estado atual ($x_t$) mais o viés desta relação ($b_c$). Esta equação gera um número entre -1 e 1, e tem como objetivo informar o grau de importância do estado atual.

$$\tilde{C}_t = tahn(W_c[h_{t-1}, x_t] + b_c) \tag{2.23}$$

O último passo (componente) a ser calculado é o *output gate*, representado na Figura 2.13. Há dois processos nesta porta. O primeiro, representado pela equação 2.24 é calcular o percentual de informação da observação anterior e do vetor de observações referentes ao estado atual será utilizado para interagir com o resultado da célula atual ($C_t$). Nesta equação, $W_o$ representa o peso da sinapse e o $b_o$ representa o viés da sinapse (Olah, 2015).

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.24}$$

O valor $o_t$ é multiplicado pela tangente hiperbólica do estado da célula atual para prever a observação referente ao tempo $t$ ($h_t$), como demonstrado na equação 2.25 (Phi, 2020).

$$h_t = o_t * tanh(f_t * C_{t-1} + i_t * \tilde{C}_t) \tag{2.25}$$

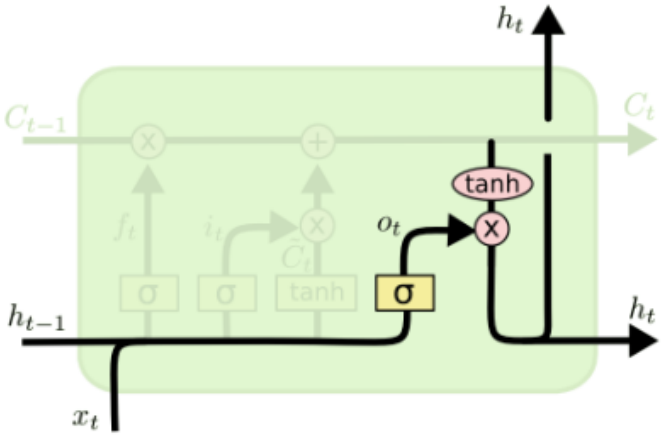


Figura 2.13: Representação do *output gate* de um LSTM. Fonte: Olah (2015)

## 2.8 ÁRVORE DE DECISÃO

O algoritmo de árvore de decisão foi utilizado nesta dissertação como base de dois algoritmos de aprendizagem *ensembles*, GBM e *random forest*. Em aprendizado de máquina, os modelos *ensembles* utilizam múltiplos modelos de aprendizagem para obter um resultado preditivo melhor do que se obteria a partir de um algoritmo individual (Polikar, 2006).

Para aprender, a árvore de decisão estratifica o espaço das variáveis independentes em regiões simples, através de cortes paralelos aos eixos (James et al., 2014). Visualmente, ele pode ser visto como um gráfico de fluxo, em que a cada nódulo de decisão (representado por quadrados nas árvores da Figura 2.15) o algoritmo toma uma decisão do tipo verdadeiro/falso e ramifica a árvore, até a decisão final, representada pelo nódulo de decisão (os círculos, na Figura 2.15). A resposta final do modelo encontra-se nos nódulos de decisão (Koning e Smith, 2017), representada pela predição de cada árvore.

Para fazer as estratificações do espaço das variáveis independentes, o algoritmo divide cada variável em diversos pontos cada uma e a que apresentar o menor o erro quadrático médio (EQM) é escolhido para ser o ponto de divisão da primeira estratificação (na representação visual, esta estratificação é representada pelo nódulo de divisão). A equação 2.26 apresenta o EQM de cada subdivisão feita. O $N_t$ representa o número de observações de cada subdivisão, $D_t$ é a subdivisão, o $i$ representa cada uma das observações presentes em $D_t$, $y$ é a observação apresentada e $\hat{y}$ é a predição da observação em questão (Li, 2019).

$$EQM(t) = \frac{1}{N_t} \sum_{i \in D_t} (y^{(i)} - \hat{y}_t)^2 \tag{2.26}$$

O usuário, de acordo com Koning e Smith (2017), pode atribuir um critério de parada de estratificação. Este critério pode ser um número mínimo de observações por subconjunto, quando a árvore atinge um valor mínimo de EQM ou um número máximo de nódulos. Ou,

se desejar, pode-se esperar que cada nódulo de decisão seja puro, contudo, esta última opção, segundo os autores, pode incorrer em uma superestimação.

## 2.9 GRADIENT BOOSTING MACHINES

O modelo GBM foi utilizado nesta dissertação como um modelo de série temporal. Segundo Singh (2018), a ideia principal do *boosting* é transformar algoritmos de aprendizado de máquina francos[4] em algoritmos de aprendizado fortes. Para os autores, a essência do GBM[5] é começar com um modelo fraco e sequencialmente aumentar o seu desempenho construindo novas árvores onde a anterior cometeu os maiores erros. Isto é, a nova árvore se concentrará nas linhas de treinamento onde a árvore anterior cometeu os maiores erros de previsão (Boehmke e Greenwell, 2019). A Figura 2.14 apresenta uma representação do GBM com árvores de decisão, na qual é criado um modelo inicial a partir dos dados, identificado os erros e criando um segundo modelo, concentrando-se onde o modelo errou mais. A partir do segundo modelo, cria-se mais modelos em sequência até atingir um critério de parada, como número máximo de árvores ou se o erro não diminui mais. A resposta final do GBM é a resposta do último modelo da sequência.

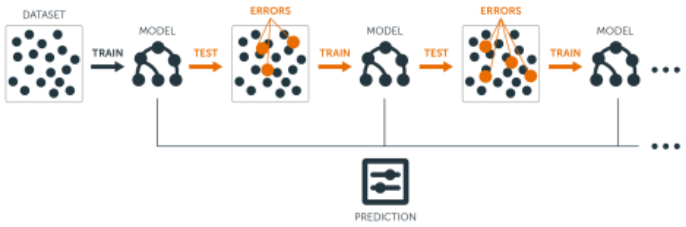


Figura 2.14: Representação do *GBM*. Fonte: Boehmke e Greenwell (2019)

Para construir o modelo GBM através de um processo sequencial, utiliza-se o seguinte passo-a-passo:

- Entrada: Dados $\{(x_i, y_i)\}_{i=1}^{n}$, e uma função custo diferencial $L(y_i, F(x_i))$.
  1. Inicia-se o modelo com uma constante: $F_0(x) = \text{argmin}_\gamma \sum_{i=1}^{n} L(y_i, \gamma)$
  2. Para árvore $m = 1$ até $M$:
     (a) Computar o resíduo: $r_{im} = -[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}]_{F(x)=F_{m-1}(x)}$
     (b) Ajustar uma árvore de decisão para os valores dos resíduos $r_{im}$ e criar nódulos de respostas chamadas $R_{jm}$ para $j = 1, ..., J$.
     (c) Para cada $j = 1, ..., J$, calcular $\gamma_{jm} = \text{argmin}_\gamma \sum_{x_i \in R_{ij}} L(y_i, F_{m-1}(x_i) + \gamma)$
     (d) Atualiza a próxima árvore $m$, $F_m(x) = F_{m-1}(x) + \nu \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

[4]Um modelo de aprendizado de máquina franco é aquele que desempenha ligeiramente melhor do que uma escolha aleatória, como uma árvore de decisão com poucas divisões, por exemplo.

[5]O GBM permite acoplar diversos tipos de algoritmos de aprendizagem. Porém, o mais comum deles é a árvore de decisões (James et al., 2014) e elas serão a base desta explicação.

- Saída: $F_M(x)$

No passo-a-passo acima, $x$ representa o conjunto de variáveis de entrada, $y$ a variável resposta, $n$ o número de observações, $i$ qual observação do conjunto de dados refere-se o cálculo, $\gamma$ representa o valor observado da resposta, $M$ representa o número de árvores total construída (ou a ser construída pelo modelo), $m$ o índice da árvore em questão e $j$ refere-se aos nódulos de resíduos. O GBM exige como entrada um conjunto de dados, nos uma coluna representa os dados a serem preditos e uma função custo, que geralmente é dada pela seguinte equação:

$$Custo = \frac{1}{2}(Valor\ Observado - Valor\ Predito)^2 \tag{2.27}$$

## 2.10 RANDOM FOREST

Para a previsão dos erros dos modelos de séries temporais, foi escolhido o regressor *random forest*. Para Breiman (2001), o regressor *random forest* é um conjunto de árvores de decisão regressoras independentes, nas quais cada uma delas prediz um valor numérico e o resultado final do *random forest* é a média aritmética de todas as árvores de decisão. A Figura 2.15 mostra uma simplificação do *random forest*.

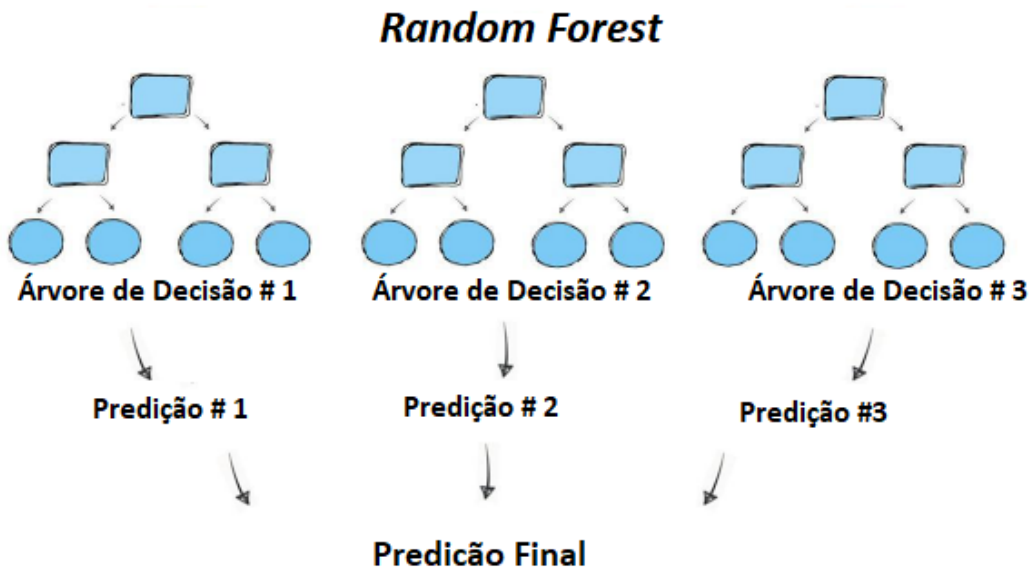


Figura 2.15: Representação de um *random forest*. Fonte: Koning e Smith (2017)

Cada árvore do *random forest* é construída, segundo Koning e Smith (2017), usando o seguinte processo:

1. Cria-se aleatoriamente um subconjunto dos dados (linhas do conjunto de dados original);
2. Seleciona-se aleatoriamente variáveis independentes deste subconjunto de dados (colunas do conjunto de dados original);

3. Treina-se a árvore;

Destaca-se que este processo se repete $n$ vezes, no qual $n$ é o número de árvores desejadas pelo usuário. A ideia do *random forest* é que cada árvore de decisão é treinada com um subconjunto de observações e atributos, aprendendo este subespaço do conjunto de dados original (Breiman, 2001).

Para selecionar quais observações serão usadas em cada árvore de decisão, é usado um *bootstrapping*, pois, desta forma cada observação tem a mesma chance de ser considerada para cada árvore de decisão e uma mesma observação pode estar em mais de uma árvore de decisão, dado que é feito uma reposição de observações para cada árvore (Koning e Smith, 2017). O mesmo ocorre com os atributos, mas antes deve-se restringir o número de atributos que cada árvore deve ter. Para a predição, considera-se a média aritmética de cada árvore de decisão, dado pela equação 2.28 (Koning e Smith, 2017).

$$\hat{f} = \frac{1}{T}\sum_{b=1}^{T} f_b(x') \tag{2.28}$$

Na equação 2.28, $\hat{f}$ é o valor previsto pelo *random forest*, $T$ é o número de árvores de decisão, $b$ representa cada árvore de decisão e $f_b$ representa a predição individual da árvore $b$, dado um subconjunto $x'$.

## 2.11 SMAPE

O critério de avaliação do desempenho da predição dos modelos de série temporal usado foi o SMAPE. Este critério é definido por:

$$SMAPE = \frac{\sum_{t=1}^{n} |A_t - F_t|}{\sum_{t=1}^{n} |A_t + F_t|} \tag{2.29}$$

Na equação 2.29, o $n$ é o número de observações previstas pelo modelo de séries temporais, o $A_t$ é o valor observado e o $F_t$ é o valor previsto pela série temporal.

## 2.12 RAIZ DO ERRO QUADRÁTICO NORMALIZADO

Uma das etapas do desenvolvimento do meta-regressor é encontrar os hiper-parâmetros ótimos do *random forest*. Para avaliar qual dos modelos de *random forest* desempenhou melhor na fase de otimização, foi adotado o critério raiz do erro quadrático normalizado (REQN). Segundo Guerra et al. (2008), esta métrica não tem escala e não é comparável com meta-regressores que utilizaram outros conjuntos de dados, contudo, valores de REQN menores que 1 informam que o meta-regressor tem um desempenho melhor que predizer o valor médio (uma espécie de modelo *naïve*). A equação 2.30 define o REQN. Nesta equação, $A_n$ é o valor observado da n-ésima

observação, $F_n$ é o valor predito desta observação e $\bar{A}$ é a média de todas as observações que foram previstas pelo modelo de interesse.

$$REQN = \frac{\sum_{n=1}^{N}(A_n - F_n)^2}{\sum_{n=1}^{N}(A_n - \bar{A_n})^2} \tag{2.30}$$

## 2.13 ERRO ABSOLUTO MÉDIO

De acordo com Willmott e Matsuura (2005), há duas principais métricas para a avaliação do desempenho de uma regressão, o erro absoluto médio (EAM) e a raiz do erro quadrático médio (REQM). A primeira métrica, segundo os autores, é um critério mais natural para avaliar o erro de um modelo do que a segunda métrica, além disto, o erro absoluto médio não é ambíguo, como o REQM. A equação 2.31 demonstra como é calculado o EAM, sendo o $N$ é o número de observações preditas, $F_n$ é o valor da predição da *n-ésima* observação e $A_n$ é o valor verdadeiro desta observação.

$$EQM = \frac{1}{N}\sum_{n=1}^{N}|A_n - F_n| \tag{2.31}$$

## 2.14 CARACTERÍSTICAS DAS SÉRIES TEMPORAIS

A extração das características (variáveis independentes do meta-classificador) de cada série temporal foi feita através do pacote *tsfeatures* (Hyndman et al., 2019), da linguagem de programação *R*. No total, foram extraídas 60 variáveis das séries temporais.

Para a extração das características das séries temporais, o primeiro passo foi normalizar todas as séries temporais em um mesmo intervalo, entre [0, 1]. Deste modo, todas as séries temporais ficam com a mesma magnitude, assim como suas respectivas características. Esta transformação foi feita através da equação *min-max*:

$$x_{transformado} = \frac{x - min}{max - min} \tag{2.32}$$

Na equação 2.32, $x$ é cada observação da série temporal, $min$ é o menor valor da respectiva série temporal e $max$ é o maior valor da respectiva série temporal.

Abaixo, segue a lista de como estas características estão representadas no pacote *tsfeatures* e seus respectivos significados.

1. **embed2_ incircle_1:** retorna a proporção de observações dentro de um limite circular de tamanho 1. Saída da função: reais, [0, 1].

2. **embed_incircle_2:** retorna a proporção de observações dentro de um limite circular de tamanho 2. Saída da função: reais, [0, 1].

3. **ac_9:** informa a autocorrelação de ordem 9 da série temporal. Saída da função: reais, [-1, 1].

4. **firstmin_ac:** calcula qual é a ordem do primeiro mínimo da autocorrelação da série temporal. Saída da função: inteiros, [1, ∞].

5. **trev_num:** mede a autocorrelação não-linear normalizada da série temporal, na ordem 1. Saída da função: reais, [-1, 1] .

6. **motiftwo_entro3**: calcula a entropia de ordem 3 da série temporal binarizada pela média. Saída da função: reais, [0, ∞].

7. **walker_propcross**: simula um hipotético *(*passeio) (*walker*) pela série temporal desejada, criando uma nova série temporal. Cada observação da série temporal simulada é um resultado da observação anterior da nova série mais dez porcento da observação de mesmo índice da série original (a nova série temporal começa no zero). A função retorna o percentual de vezes que a série temporal simulada cruza com a original. Saída da função: reais, [0, 1].

8. **localsimple_mean1:** retorna a primeira ordem da autocorrelação dos resíduos de uma previsão simples local que cruza com o zero. Esta previsão é feita através da média. Saída da função: inteiros, [1, ∞].

9. **localsimple_lfitac:** retorna a primeira ordem da autocorrelação dos resíduos de uma previsão simples local que cruza com o zero. Esta previsão é feita através de um preditor linear. Dominio: inteiros, [1, ∞].

10. **sampen_first:** retorna a primeira entropia da amostra da série temporal, no qual a dimensão é 5 e o limite é 0.3. Saída da função: reais, [0, ∞].

11. **std1st_der:** informa o desvio padrão da primeira diferença [6] da série temporal. Saída da função: reais, [0, 1].

12. **spreadrandomlocal_meantaul_50:** cem séries temporais são criadas aleatoriamente com tamanho equivalente à metade do tamanho da série temporal original e a função retorna a média da primeira autocorrelação que cruza com o zero das nova série temporais. Saída da função: reais, [0, ∞].

13. **spreadrandomlocal_meantaul_33:** cem séries temporais são criadas aleatoriamente com tamanho equivalente a um terço do tamanho da série temporal original e a função retorna a média da primeira autocorrelação que cruza com o zero das nova série temporais. Saída da função: reais, [0, ∞].

[6] A primeira diferença significa que é criada uma nova série temporal, com observação menos a observação anterior, ignorando a primeira observação da série temporal original.

14. **spreadrandomlocal_meantaul_25:** cem séries temporais são criadas aleatoriamente com tamanho equivalente a um quarto do tamanho da série temporal original e a função retorna a média da primeira autocorrelação que cruza com o zero das nova série temporais. Saída da função: reais, [0, ∞].

15. **spreadrandomlocal_meantaul_20:** cem séries temporais são criadas aleatoriamente com tamanho equivalente a quinto do tamanho da série temporal original e a função retorna a média da primeira autocorrelação que cruza com o zero das nova série temporais. Saída da função: reais, [0, ∞].

16. **spreadrandomlocal_meantaul_ac2:** cem séries temporais são criadas aleatoriamente com tamanho de duas vezes a primeira autocorrelação que cruza com o zero da série temporal original e a função retorna a média da primeira autocorrelação que cruza com o zero das nova série temporais. Saída da função: reais, [0, ∞].

17. **histogram_mode_10:** informa o centro da barra com mais observações de um histograma divido em dez barras. Saída da função: reais, [0, 1]

18. **outlierinclude_mdrmd:** mede a mediana conforme valores aberrantes [7] são incluídos na série. O número de valores aberrantes incluídos na série temporal segue a regra do máximo desvio padrão encontrado na série temporal, dividido por 0,01. A cada inclusão, é calculado a intensidade da mudança na mediana da série temporal. O resultado final é a mediana desta mediana. Saída da função: reais, [-1, 1].

19. **fluctanal_prop_r1:** retorna o intervalo do ajuste de polinômio de ordem 1, correspondente a análise flutuação. Saída da função: reais, [0, 1].

20. **length:** informa o número de observações da série temporal. Saída da função: inteiros, [2, ∞].

21. **periodogram:** calcula a periodicidade da série temporal de acordo com a análise de Fourier. Saída da função: inteiro, [1, ∞].

22. **entropy:** calcula a entropia de Shannon da série temporal. Saída da função: reais, [0, 1].

23. **stability:** mede a variância das médias das janelas não-sobrepostas da série temporal. Uma janela[8] em é um corte consecutivo feito na série temporal. Saída da função: reais, [0, ∞].

24. **lumpiness:** retorna a média das médias das janelas não-sobrepostas da série temporal. Saída da função: reais, [0, ∞].

[7] *Outlier.*

[8] O tamanho da janela em todos as variáveis são iguais à frequência da série temporal. A frequência da série temporal é igual ao número de observações em uma unidade de tempo.

25. **crossing_points:** informa a proporção de vezes em que a série temporal cruza a média dela mesma. Saída da função: reais, [0, 1].

26. **flat_spots:** cria-se dez intervalos na série temporal de tamanhos iguais, dividindo a diferença entre o valor máximo da série temporal e o valor mínimo dela por 10. O próximo passo corresponde a classificar cada observação da série temporal em um intervalo de 1 a 10. A função retorna o número da maior sequência de observações presentes no mesmo intervalo dividido pelo tamanho da série temporal. Saída da função: reais, [0, 1].

27. **hurst:** calcula o coeficiente de memória de longo prazo da série temporal de acordo com o coeficiente de Hurst. Ele é dado por $H = DF + 0,5$, no qual $H$ é o coeficiente de Hurst e $DF$ é a decomposição fractal da série temporal. Saída da função: reais, [0,5 , 1].

28. **unitroot_kpss:** retorna o valor do KPSS. O KPSS testa (neste caso) sob a hipótese nula se a série temporal é estacionária em relação a uma tendência linear e a hipótese alternativa é se ela é uma raiz unitária. A série temporal testada é primeira diferença da série temporal original. Saída da função: reais, [0, $\infty$].

29. **unitroot_pp:** calcula a estatística do teste Phillips-Perron. A hipótese nula deste teste é que a série é integrada em ordem 1, isto é, ela se torna estacionária na primeira diferença. Saída da função: reais, [-$\infty$, +$\infty$].

30. **arch_stat:** retorna o valor do coeficiente de determinação de um modelo autorregressivo de ordem 13. Saída da função: reais, [0, 1].

31. **binarize_mean:** calcula a proporção de obervações acima da média. Saída da função: reais, [0, 1].

32. **max_level_shift:** mede a maior diferença entre a média de duas janelas sobrepostas da série temporal. Saída da função: reais, [0, 1].

33. **time_level_shift:** retorna o índice da observação em que ocorre o *max_level_shift* dividido pelo tamanho da série temporal. Saída da função: reais, [0, 1].

34. **max_var_shift:** calcula a maior diferença de variância entre duas janelas sobrepostas da série temporal. Saída da função: reais, [0, 1].

35. **time_var_shift:** retorna o índice da observação em que ocorre o *max_varl_shift* dividido pelo tamanho da série temporal. Saída da função: reais, [0, 1].

36. **max_kl_shift:** calcula a maior divergência de Kullback-Leibler entre duas janelas sobrepostas. Isto é, a função retorna a maior entropia relativa. Saída da função: reais, [0, $\infty$].

37. **time_kl_shift:** retorna o índice da observação em que ocorre o *max_kl_shift* dividido pelo tamanho da série temporal. Saída da função: reais, [0, 1].

38. **arch_acf:** o primeiro passo é criar uma segunda série temporal a partir da série temporal original, removendo a média, tendência e a informação de autorregressão. Então, é modelado com a nova série temporal um modelo ***GARCH*(1,1)**. A função retorna o somatório do quadrado das doze primeiras autocorrelações. Saída da função: reais, [0, $\infty$].

39. **arch_r2:** calcula o coeficiente de determinação do modelo ***GARCH*(1,1)** modelado na função *arch_acf*. Saída da função: reais, [0, $\infty$].

40. **garch_acf:** esta função é um segundo passo da função *arch_acf*. Ela modela através de um outro modelo ***GARCH*(1,1)** os resíduos da originados na modelagem do ***GARCH*(1,1)** e retorna o somatório do quadrado das doze primeiras autocorrelações. Saída da função: reais, [0, $\infty$].

41. **garch_r2:** calcula o coeficiente de determinação do modelo ***GARCH*(1,1)** da função *garch_acf*. Saída da função: reais, [0, 1].

42. **alpha_holt:** retorna a constante de suavização do nível médio da série temporal, pelo método *Holt's Linear*. Saída da função: reais, [0, 1].

43. **beta_holt:** mede a constante de tendência da série temporal, pelo método *Holt's Linear*. Saída da função: reais, [0, 1].

44. **x_pacf5:** mede a soma dos quadrados dos primeiros cinco primeiros coeficientes das autocorrelações parciais da série temporal. Saída da função: reais, [0, $\infty$].

45. **diff1x_pacf5:** calcula a soma dos quadrados dos primeiros cinco primeiros coeficientes das autocorrelações parciais da série temporal diferenciada em primeira ordem. Saída da função: reais, [0, $\infty$].

46. **diff2x_pacf5:** calcula a soma dos quadrados dos primeiros cinco primeiros coeficientes das autocorrelações parciais da série temporal diferenciada em segunda ordem. Saída da função: reais, [0, $\infty$].

47. **x_acf1:** retorna a primeira autocorrelação da série temporal. Saída da função: reais, [-1, 1].

48. **diff1_acf1:** informa a primeira autocorrelação da série temporal diferenciada em primeira ordem. Saída da função: reais, [-1, 1].

49. **diff2_acf1:** informa a primeira autocorrelação da série temporal diferenciada em segunda ordem. Saída da função: reais, [-1, 1].

50. **x_acf10:** calcula a soma dos quadrados das dez primeiras autocorrelações da série temporal. Saída da função: reais, [0, 10].

51. **diff1_acf10:** calcula a soma dos quadrados das dez primeiras autocorrelações da série temporal diferenciada em primeira ordem. Saída da função: reais, [0, 10].

52. **diff2_acf10:** calcula a soma dos quadrados das dez primeiras autocorrelações da série temporal diferenciada em segunda ordem. Saída da função: reais, [0, 10].

53. **trend:** mede o quão forte é a tendência da série temporal. Saída da função: reais, [0, 1].

54. **spike:** calcula a variância das variâncias do processo *one-leave-out*. Isto é, a cada iteração é retirada uma observação diferente da série temporal e é calculada a variância. Após todas as variâncias calculadas, calcula-se a variância de todas elas. Saída da função: reais, [0, 1].

55. **nperiods:** variável lógica. Se retorna um, significa que a série temporal não é sazonal e zero, caso contrário. Saída da função: *booleano*.

56. **linearity:** retorna a linearidade baseada nos coeficientes da regressão ortogonal quadrática. Saída da função: reais, [-∞, +∞].

57. **curvature:** retorna a curvatura baseada nos coeficientes da regressão ortogonal quadrática. Saída da função: reais, [-∞, +∞].

58. **e_acf1:** retorna a primeira autocorrelação do ruído após uma decomposição da série temporal. Saída da função: reais, [-1, 1].

59. **e_acf10:** calcula a soma dos quadrados das dez primeiras autocorrelações do ruído após uma decomposição da série temporal. Saída da função: reais, [0, 10].

60. **nonlinearity:** mede a não linearidade da série temporal. Valores próximos de zero indicam que a série temporal não é linear. Saída da função: reais, [0, +∞].

# 3 ESTADO-DA-ARTE

Este capítulo está dividido em três seções. A primeira seção refere-se aos trabalhos nos quais a solução de previsão de séries temporais foi utilizado somente um tipo de modelo, isto é, a solução foi através de uma abordagem de especialista. A segunda seção reúne trabalhos que utilizaram mais de um tipo de modelo de série temporal e a comparação entre os modelos foi feita através de um *hold-out*. E a última seção reúne trabalhos que utilizaram a metodologia de meta-aprendizado.

## 3.1 ESPECIALISTAS

Uma das maneiras mais simples de prever uma série temporal é por meio de SES, que assume que o futuro será mais ou menos igual ao passado recente. Papacharalampous et al. (2018) e Jere et al. (2017), aplicaram esta metodologia para modelar a temperatura e precipitação mensal (a nível global) e o investimento estrangeiro direto na Zâmbia, respectivamente. Outros dois trabalhos usaram modelos exponenciais, mas no caso, como havia sazonalidade, a metodologia usada foi o Holt-Winter. Yan-ming Yang et al. (2017) modelaram a taxa de falhas de aeronaves e Dantas et al. (2017) criaram um modelo para a previsão de demanda aérea.

Choi et al. (2015) utilizaram modelos ARIMA para prever a produção de petróleo em sete condados do estado da Dakota do Norte, nos Estados Unidos. Os autores utilizaram os dados da produção mensal destes condados entre os anos de 1970 e 2014 (ou 528 observações) com o objetivo de prever a produção dos próximos 5 anos ou 60 observações. O erro absoluto médio dos modelos variou entre 8% e 10%, aproximadamente.

Gonzalez e Yu utilizaram duas séries temporais não-lineares sintéticas para analisar qual o desempenho de um LSTM em um problema de série temporal não-linear. Em cada série temporal havia 180 mil observações, sendo 80% delas na base de treino e 20% delas na base de teste. No estudo, foram testadas entre uma e cinco camadas escondidas na rede neural. O modelo de LSTM com 3 camadas teve o melhor desempenho pelo critério raiz do erro quadrático médio (Gonzalez e Yu, 2018).

Para prever a qualidade da água do Lago Taihu, China, Wang et al. (2017) ajustaram modelos de LSTM. Os autores experimentaram quais eram os melhores hiper-parâmetros de um LSTM (número de neurônios na camada escondida e o épocas) para prever a quantidade de oxigênio dissolvido e o total de fósforo no lago[1]. Foram testados 10, 13, 15 e 17 neurônios e 10, 20, 25 e 30 épocas. A base treino de cada série temporal de interesse tinha 657 observações e a base de teste tinha 65 observações. Utilizando o critério da menor REQM, os autores concluíram que para prever o oxigênio dissolvido a melhor configuração de LSTM era 15 neurônios e 25

[1]Em ambos os casos, a medida utilizada foi miligramas por litro.

épocas, enquanto que para prever o nível de fósforo na água a melhor configuração era 15 neurônios e 20 épocas. Os LSTM também foram utilizadas para prever o tempo de viagem nas auto-estradas britânicas, com 66 séries temporais (Duan e Yisheng, 2016). A RQEM no experimento foi de 7%.

Adebiyi et al. (2014) modelaram as ações da empresa *Nokia Corporation* e do *Zenith Bank* através de um modelo ARIMA. No caso da *Nokia*, a base de treino foi a cotação diária da ação no período entre 25 de abril de 1.995 e 25 de fevereiro de 2.010, totalizando 3.990 observações e no caso do *Zenith Bank* a base de treino compreendeu o período entre 3 de janeiro de 2.006 até 25 de fevereiro de 2.011, ou 1.296 observações diárias. A base de teste em ambos os casos foi o mês de março de 2010. Para as duas séries temporais, os autores testaram todas as combinações do modelo ARIMA da ordem zero até a ordem dois, para os parâmetros $p$, $d$ e $q$ na base de treino e utilizaram o *Bayesian Information Criterion* como critério de escolha do modelo a ser utilizado na base de teste. Para a fase de teste, os modelos escolhidos foram ARIMA(2, 1, 0) e ARIMA(1, 0, 1), para a *Nokia* e *Zenith Bank*, respectivamente. Como resultado, os autores obtiveram um erro absoluto médio entre a previsão e o valor observado de 1,27% no caso da previsão da cotação das ações da *Nokia* e um erro de previsão de 7,2% no caso da previsão das cotações das ações do *Zenith Bank*. No segundo caso, as cotações da empresa cresceram em um ritmo mais rápido do que o modelo pode acompanhar.

Outro estudo, elaborado por Moayedi e Masnadi-Shirazi, os autores utilizaram modelos ARIMA para prever o tráfego de rede, em uma série temporal sintéticas com 900 observações (Moayedi e Masnadi-Shirazi, 2008). Para os autores, o modelo ARIMA foi capaz de detectar o padrão de comportamento da série temporal sintética.

## 3.2 HOLD-OUT

Sagheer e Kotb (2018) fizeram um experimento para prever a produção de um campo de petróleo em Huabei, China e em um outro campo em Cambay, Índia. Foram utilizados sete modelos de previsão de séries temporais, entre eles ARIMA, LSTM (duas configurações diferentes) e redes neurais recorrentes (quatro configurações diferentes). Os autores selecionaram o melhor modelo através do critério REQM, sendo 80% de cada série temporal a base treino e os restantes 20% a base teste. No quadro geral, o modelo LSTM ficou com o menor erro em ambos os estudos de caso. O modelo ARIMA ficou em último no estudo de caso do campo de petróleo chinês e em segundo no estudo de caso indiano.

Cai et al. (2019) testaram o desempenho dos modelos ARIMA, *convolutional neural-network* (CNN)[2] e ANN para prever o consumo de energia elétrica em três prédios tanto em um

[2]Conforme descritos pelos autores, a CNN desenvolvida para o artigo funciona de forma similar a uma ANN para séries temporais. A diferença entre as duas redes-neurais é que para cada saída na CNN, há dois tipos de entrada. A primeira entrada é o vetor da série temporal, com as observações anteriores (no caso do artigo, foi utilizado somente a observação anterior, $x_{t-1}$) e a segunda entrada são as observações anteriores transformadas por um filtro convolucional de uma dimensão, isto é, o filtro se move em apenas um eixo, no caso, no eixo do tempo. Este filtro tem como objetivo extrair as características de $k$ observações. O resultado deste filtro é utilizado em

período frio quanto em um período não-frio, gerando seis diferentes séries temporais. O critério de escolha do melhor modelo se baseou na REQM na previsão da base treino, que correspondia a 5% do total das séries temporais (a base de treino correspondia a 90% e a base de validação correspondia a 5%). Das seis aplicações, o modelo ARIMA teve melhor desempenho em uma delas, enquanto que o modelo CNN teve o melhor desempenho nas demais.

Para avaliar qual é o melhor modelo para prever a velocidade dos ventos na região da Mongólia Interior, China, (Chen et al., 2018) testaram seis modelos diferentes: ARIMA, LSTM, *support vector regressor* (SVR), *gradient boosting*, ANN e *k-nearest neighbors* (KNN). O experimento foi dividido em dois estudos de caso, no primeiro estudo de caso a série temporal era composta de observações coletadas a cada dez minutos, enquanto que no segundo estudo de caso a série temporal era composta por observações coletadas por hora. O tamanho da série temporal do primeiro estudo era de 738 observações e do segundo estudo era de 720 observações e em ambos os casos a base de treino correspondeu a 70% das observações e a base de teste correspondeu a 30% das observações. O critério de desempenho dos modelos foi a REQM e o LSTM ficou com a melhor pontuação nos dois casos enquanto que o ARIMA ficou na última posição também nos dois casos.

Os modelos ARIMA, SVR, LSTM e ANN foram utilizados para prever o tráfego real da rede *Géant*[3]. A série temporal era composta de 2.190 observações, coletadas a cada uma hora. O critério de seleção do melhor modelo de previsão de séries temporais foi a REQM. O pior modelo neste experimento foi o ARIMA, com um erro aproximadamente de 7,8% e o melhor modelo foi o LSTM, com um erro de 4,7% (Hua et al., 2018).

Em um estudo de previsão de tráfego de trânsito, Fu et al. (2016) compararam o desempenho de uma LSTM e um ARIMA para prever o tráfego de quatro áreas do Estado da Califórnia, Estados Unidos. Estas quatro regiões somam 15.000 sensores de trânsito e para o estudo foram selecionados 50 sensores aleatoriamente. Cada sensor produziu sua própria série temporal. Como resultado, o modelo LSTM obteve uma melhora de 5% considerando a raiz do erro quadrático médio melhor segundo o critério utilizado, a raiz do erro quadrático médio. Utilizando os dados da mesma região[4], Liu et al. (2017) utilizaram além do LSTM e ARIMA, regressão linear, *least absolute shrinkage and selection operator* (LASSO) e *ridge regression*. O modelo com a menor raiz do erro quadrático médio foi o LSTM, seguido da regressão linear, LASSO, *ridge regression* e ARIMA.

Na área financeira, Siami Namini e Siami Namin (2018) testaram o desempenho do ARIMA e do LSTM em 12 séries temporais financeiras, da bolsa de valores americana. A base de

uma função sigma e o resultado desta função é um dos fatores do produto tensorial - o outro fator deste produto é a primeira entrada da CNN. A predição do CNN é o resultado do produto tensorial é transformado linearmente junto com o peso do respectivo neurônio e um *bias*. Contudo, ressalta-se que a extração das características não é semelhante ao proposto nesta dissertação, e sim, fazer um processo convolucional, isto é, transformar um *kernel* e fazer um produto de ponto deslizante com os sinais da série.

[3]Géant é uma rede pan-européia de pesquisa e educação, que tem objetivo interligar diversos institutos europeus de pesquisa.

[4]Não fica claro, neste caso, qual o número de sensores utilizados no estudo.

treino para todas as séries temporais correspondeu a 70% e de teste 30% do total de observações. A menor série temporal tinha ao todo 368 observações e a maior série temporal tinha 1.698 séries temporais. Segundo os autores reportaram, a LSTM reduziu em média 87% a REQM em relação ao ARIMA, no caso das séries temporais testadas. Em outra aplicação no mercado financeiro, Yan e Ouyang (2017) modelaram o principal índice da Bolsa de Valores de Xangai (China) usando quatro diferentes técnicas: LSTM, SVR, ANN e KNN. A série temporal tinha 1.347 observações, sendo que 80% destas observações foram separadas para a base de treino e 20% para a base de teste. O com a menor REQM foi a LSTM, seguida pelo ANN, SVR e por último KNN.

Considerando os quatorze experimentos analisados nas seções Especialista e *Hold-Out*, apenas três contavam com uma série temporal, dos outros onze experimentos, ou o problema exigia mais de uma série temporal (como o problema da produção de petróleo em sete condados (Choi et al., 2015) ou cinquenta sensores de tráfego urbano (Fu et al., 2016)). Além disto, dentro de uma rotina de previsão, a observação a ser prevista em um dado momento, $x_{t+1}$ se torna parte da série temporal para a previsão da observação subsequente, $x_{t+2}$. Esta adição de mais uma observação, pode mudar a estrutura de geração interna de dados da série temporal, e a melhor solução para prever $x_{t+1}$ pode não ser a mesma para prever a observação subsequente. Com isto, a rotina de predição de séries temporais pode ser volumosa, seja pelo número de séries a serem previstas, seja pela rotina de escolha da série temporal ou ambas. Assim, um meta-aprendizado ajudaria na escolha, sem que os experimentos sejam repetidos periodicamente, podendo o processo ser, inclusive, automatizado. Outra vantagem da aplicação de uma metodologia de meta-aprendizado para os dois casos acima, especialista e *hold-out*, é que um meta-aprendizado pode ampliar a perspectiva do usuário em relação ao método de predição, ao sugerir um modelo que estava fora do escopo inicial, mas pode ser incluído no processo, além de facilitar para o usuário a escolha do método, oferecendo aquele que provavelmente terá o menor erro.

## 3.3 METODOLOGIAS DE META-APRENDIZADO

Segundo Lemke e Gabrys (2010), a primeira vez que o termo meta-aprendizagem foi utilizado no contexto de séries temporais foi no trabalho de Prudêncio e Ludermir (2004) apresentando um novo conceito na forma de descrever o processo de aprendizagem automática para a previsão de séries temporais através de seus atributos.

Neste trabalho, os autores apresentaram dois estudos de caso usando a meta-aprendizagem. No primeiro caso, através de dez atributos das séries temporais, sendo eles o número de observações, a media absoluta dos valores das cinco primeiras autocorrelações, o teste de significância das autocorrelações, as significâncias da primeira, segunda e terceira autocorrelação, o coeficiente de variação, os valores absolutos de curtose e assimetria e o teste de mudança de pontos, o estudo classificou cada série como sendo melhor predita através de um modelo exponencial suavizado ou uma rede-neural. Ao todo, foram 99 séries temporais em um

processo iterativo de *one hold-out* no qual 98 eram treinadas com ambos os modelos e através dos atributos classificava a série temporal sobressalente utilizando uma árvore de decisão C4.5.

A taxa de erro, isto é quantas vezes o classificador errou dividido pelo total de predições, deste classificador de séries temporais é de 37,37%, enquanto o intervalo de confiança de 95% do erro é entre 27,84% e 46,90%. Isto é, espera-se que com 95% de confiança que o classificador erre entre um quarto e aproximadamente metade das vezes e na média, espera-se que o classificador erre um terço das vezes, com apenas dois modelos se contrapondo.

No segundo estudo de caso, foram utilizadas 3.003 séries temporais provenientes da Competição-M3 (Makridakis e Hibon, 2000). Elas foram modeladas com modelos auto-regressivo (AR), Holt-Winter Linear e *random walks*. As variáveis que descreviam as séries foram o número de observações, a tendência, o primeiro coeficiente de autocorrelação, o percentual de mudanças direção e o tipo de série temporal (micro, macro, indústria, finanças, demografias e outros). Como classificador das séries temporais, foi utilizado uma ANN[5]. O objetivo deste segundo caso de estudo foi testar a classificação entre dois modelos de cada vez, ou seja, classificar a série temporal entre AR e Holt Winter, depois entre AR e random walks e por fim Holt Winter e random walk. A média da taxa de erro das três classificações foi de 33,5%.

Shah (1997) classificou 203 séries temporais trimestrais da Competição-M3 em três tipos de modelo de previsão: Holt-Winters, Exponencial Suavizado Simples e Séries Temporais de Estrutura Simples. Para descrever cada série temporal, foram utilizados 24 variáveis. Para selecionar o melhor modelo, o autor utilizou análise discriminante como forma de aprendizado. No total, foram testados 16 tipos de classificadores, variando o número de variáveis independentes e o tipo de modelo discriminante (Análise Discriminante Linear, Análise Discriminante Quadrática, *Kernel Group* e *Kernel Pool*). A taxa de erro dos classificadores variou de 34% até 60%.

Lemke e Gabrys (2010) classificaram dois bancos de dados com 111 séries temporais cada um, provenientes das Competição-NN3 (Crone et al., 2011) e Competição-NN5 (Ben Taieb et al., 2011). Os meta-classificadores foram: rede neural, árvore de decisão e *support vector machine*. Neste estudo, foi extraído de cada série temporal 24 características e os modelos de previsão das séries temporais foram ARIMA, modelos exponenciais suavizados e redes-neurais. Como métrica de desempenho, os autores utilizaram a média dos SMAPE de cada sugestão do meta-classificador. Isto é, o para cada classificação de modelo de série temporal na base teste, calculava-se o SMAPE da série temporal. O melhor SMAPE do banco de dados da Competição-NN3 foi de 17% e para o banco de dados da Competição-NN5 foi de 24%. Isto quer dizer que, espera-se que com as sugestões do melhor classificador, o usuário espera ter um erro médio de 17%, se o meta-classificador for ensinado com o banco de dados da Competição-NN3. Cada aplicação tem o seu próprio limite de erro aceitável, contudo, é raro a aplicação que aceite um erro médio, se tratando de séries temporais, de 17%.

[5]Não foi especificado a quantidade de camadas, as funções de ligação e nem se era *backpropagation* a função otimizadora do erro

Wang et al. (2009) extraíram nove características de 315 séries temporais, sendo elas tendência, sazonalidade, correlação serial, não linearidade, curtose, assimetria, autocorrelação, entropia e periodicidade. Para a classificação da melhor série, foi utilizado uma métrica chamada percentual de melhora média, que mede a acurácia dos três modelos de previsão - modelo exponencial suavizado, ARIMA, rede-neural *feed-foward* - em relação a um modelo de *random walk*. O meta-classificador de séries temporais foi a árvore de decisão. Os autores não evidenciam se o meta-classificador de fato aumentou a acurácia das classificações das séries temporais.

Kück et al. (2016) usaram uma rede neural do tipo *feed-foward* com uma camada escondida como metodologia de meta-aprendizagem para selecionar quatro tipos diferentes de série temporal: sazonal, sazonal com tendência e exponencial suavizada. O conjunto de séries temporais era composto por 111 séries temporais provenientes da Competição-NN3 (Crone et al., 2011). Para caracterizar cada série temporal, os autores utilizaram sete diferentes conjuntos de variáveis. Como critério de escolha para o melhor meta-classificador, os autores utilizaram o SMAPE para doze observações da base de teste de cada série temporal. Ou seja, as doze últimas observações foram previstas por cada um dos modelos de série temporal e calculou-se para cada modelo o SMAPE. O melhor resultado encontrado foi um SMAPE médio de 11%.

Talagala et al. (2018) propuseram um meta-classificador chamado *Feature-based Forecast-Model Selection*, no qual um *random forest* seleciona, baseado entre 25 (séries temporais anuais) e 30 variáveis (séries temporais sazonais), qual o melhor modelo de previsão dado uma série temporal. Para construir este meta-classificador, foram utilizadas 3.830 séries temporais provenientes da Competição-M1 e Competição-M3, além de outras séries temporais criadas artificialmente pelos autores, processo conhecido como *data augmentation*.

As séries temporais foram classificadas nos seguintes tipos de modelos: ruído branco, ARIMA, *random walk with drift*, *random walk*, Theta, ETS sem tendência e sem sazonalidade, ETS com tendência e sem sazonalidade, ETS com tendência suavizada e sem sazonalidade, SARIMA, ETS sem tendência e com sazonalidade, ETS com tendência e com sazonalidade, ETS com tendência suavizada, com sazonalidade e método *naïve* sazonal. O estudo foi dividido em seis partes: cada uma das duas base de dados foi divido em três categorias, séries temporais anuais, séries temporais trimestrais e séries temporais mensais. O meta-classificador proposto quanto comparado com o ARIMA, em termos de erro absoluto médio ganhou em três situações: nas séries temporais anuais na Competição-M1 e nas séries anuais e trimestrais da Competição-M3.

Os sete meta-classificadores desta seção apresentaram um índice alto de erro. Como um usuário poderia incorporar na sua rotina um meta-classificador que erra em média um terço das vezes o melhor modelo preditivo de série temporal? Como é o caso dos meta-classificadores do Prudêncio e Ludermir (2004) e do Shah (1997). Ou no caso dos meta-classificadores do Kück et al. (2016) e do Lemke e Gabrys (2010), que espera-se um erro de aproximadamente 11% e 17% nas previsões propostas pelos modelos sugeridos pelos respectivos meta-classificadores.

Este alto índice de imprecisão pode ser consequência do baixo número de séries temporais usadas para treinar os meta-classificadores. O estudo com o maior número de séries

temporais utilizou 3.830 séries temporais e os demais utilizaram menos de 500 séries temporais ao todo. Assim como há uma grande variedade de aplicações de séries temporais há também uma grande variedade de séries temporais que dificilmente é representado por uma amostra limitada de séries temporais. Na falta de séries temporais reais, séries temporais artificiais poderiam ter sido criadas, como no caso do Talagala et al. (2018).

Saeed et al. (2017) criaram um meta-regressor para auxiliar na predição de geração de energia em cinco parques eólicos europeus. As séries temporais consistiam em três anos na produção de energia destes parques, além das condições climáticas ligadas ao vento, como direção e força do vento. Devido a mudanças abruptas de condições climáticas e de o sistema elétrico requerer predições rápidas, os meta-regressores tinham o objetivo de auxiliar, dentro de um sistema automatizado de predição, qual modelo preditivo seria utilizado. Para este estudo, os preditores de séries temporais eram o SVR e o ARIMA e o meta-regressor utilizado foi um ANN. A mediana do EAM dos regressores presentes no meta-regressor proposto foi de 0,083.

Os oito artigos analisados nesta seção não informaram se os métodos de meta-aprendizados passaram por um processo de otimização de hiper-parâmetros ou quais foram os hiper-parâmetros escolhidos para cada meta-classificador. Por exemplo, o número de camadas das redes-neurais, o número de árvores das *random forests* ou o critério de divisão das árvores de decisão. Estes hiper-parâmetros poderiam ser o ponto de partida na criação de outros métodos de meta-aprendizados.

Outro ponto negativo a ser destacado em relação a estes artigos sobre os meta-classificadores é a ausência de análises de dados e análises exploratórias em relação aos resultados encontrados. Por exemplo, os erros entre dois modelos preditivos de séries temporais são tão diferentes assim?

Em relação ao meta-regressor, ele foi criado para um tipo específico de séries temporais, geração de energia para parques eólicos europeus. Desta maneira, possivelmente, ele dificilmente conseguiria generalizar com séries temporais provenientes de outras origens, como uma série temporal proveniente da microeconomia, por exemplo, restringindo a sua aplicabilidade para o usuário.

# 4 MÉTODO PROPOSTO

O objetivo deste capítulo é apresentar o meta-regressor proposto para a predição de erros de previsão dos modelos de séries temporais. A Seção Meta-Regressor descreve o funcionamento do meta-regressor. A Seção Base de dados informa a origem das séries temporais utilizadas para o treinamento do meta-regressor e a descrição destas séries temporais.

## 4.1 META-REGRESSOR

Como afirmado anteriormente, a proposta deste trabalho é criar um algoritmo de meta-aprendizagem para séries temporais para ajudar o usuário a escolher um modelo preditivo para as séries temporais escolhidas, sendo um meta-regressor a metodologia escolhida. Um meta-classificador pode induzir um usuário a escolher um modelo mais complexo sem necessidade, em detrimento de um modelo mais simples e com um erro similar. Além disto, o modelo escolhido pelo meta-classificador pode não estar no escopo das soluções, seja porque o modelo não é interpretável (como uma rede neural), por não ter suporte na ferramenta utilizada pelo usuário e/ou pela falta de conhecimento do usuário na modelagem escolhida pelo meta-classificador. Uma terceira vantagem do meta-regressor sobre o meta-classificador é a versatilidade ao adicionar um novo modelo preditivo. Neste caso, todo o processo de treinamento do meta-classificador tem que ser refeito, pois um novo modelo altera a resposta final do algoritmo, enquanto que no caso do meta-regressor, deve-se treinar apenas a resposta para o novo modelo preditivo e agregá-lo junto das *n*-respostas finai do algoritmo, como um *plug-and-play*.

Em nosso estudo, consideramos 15 modelos diferentes de séries temporais, que são descritos na Tabela 4.1. Por uma questão de reprodutibilidade, adicionamos a função R e os parâmetros usados. Os regressores LSTM foram treinados usando o *framework Keras* (Chollet et al., 2015).

O modelo ARIMA é o mais utilizado na modelagem de séries temporais, conforme Hyndman e Athanasopoulos (2014). De acordo com Monfared et al. (2014) o modelo SES é simples, computacionalmente eficiente, razoavelmente preciso, fácil de ajustar e é um modelo competitivo em relação a outros métodos de previsão mais complicados. Os modelos ETS e Holt-Winters são generalizações dos modelos SES. Na perspectiva dos modelos de *deep learning*, LSTM é a arquitetura mais avançada para o aprendizado de processos estocásticos (Fischer e Krauss, 2017). Os modelos *ensemble* são representados pelo GBM. As ANN's são capazes de aprender tanto de séries temporais lineares quanto não lineares. O modelo Naïve é uma linha de base e fácil de calcular. De acordo com o Hyndman e Athanasopoulos (2014), o modelo Naïve é um passeio aleatório com modelo de deriva aplicado às séries temporais.

A primeira etapa do método proposto consiste na criação do conjunto de treinamento do meta-regressor. Para fazer isso, temos que treinar cada modelo básico e usá-los para prever o erro

Tabela 4.1: Modelos utilizados para treinar o meta-regressor

| Nome | Função R | Parâmetros |
|---|---|---|
| ARIMA | forecast::auto.arima | default |
| SES | forecast::ses | default |
| ETS | forecast::ets | default |
| GBM | gbm::gbm | n.trees = 100,000; distribution = "gaussian" |
| Holt-Winters | forecast::holt | default |
| Holt-Winters Aditivo | forecast::hw | seasonal = "additive" |
| Holt-Winters Multiplicativo | forecast::hw | seasonal = "multiplicative" |
| LSTM - 1 Camada Escondida | keras::keras_model_sequential | 1 Camada Escondida |
| LSTM - 2 Camadas Escondidas | keras::keras_model_sequential | 2 Camadas Escondidas |
| LSTM - 3 Camadas Escondidas | keras::keras_model_sequential | 3 Camadas Escondidas |
| ANN - 1 Neurônio | forecast::nnetar | repeats = 200; lambda = "auto"; size = 1 |
| ANN - 10 Neurônios | forecast::nnetar | repeats = 200; lambda = "auto"; size = 10 |
| ANN - 20 Neurônios | forecast::nnetar | lambda = "auto"; size = 20 |
| ANN - Auto Neurônios | forecast::nnetar | repeats = 200; lambda = "auto" |
| Naïve | forecast::snaive | default |

no conjunto de treinamento. O SMAPE foi a métrica adotada neste trabalho. No caso de séries temporais, a primeira vantagem desta métrica, em relação às métricas mais tradicionais como o EQM, é que ele, por ser um erro percentual, tem os erros de todas as séries temporais na mesma magnitude, facilitando a comparação dos erros no meta-aprendizado. A segunda vantagem deste método é que ele é simétrico, isto é, erros que subestimam o valor observado tem valor que erros que superestimam o valor observado Flores (1986). A terceira vantagem do SMAPE é que ele evita que valores observados próximos de zero tenham erros infinitos (ou que tendem ao infinito). Portanto, o rótulo de cada regressor é o SMAPE do respectivo modelo.

Para ajustar o modelo ARIMA, usou-se a função *auto.arima* (Hyndman e Khandakar, 2008). Esta função fornece a autorregressão, integração e ordens de média móvel (incluindo a parte sazonal do modelo, se a série temporal apresentar sazonalidade) para o melhor ajustar aos dados. Para a modelagem dos modelos Naïve, SES, ETS e Holt-Winters, utilizou-se as respecitvas funções do mesmo pacote *forecast* (Hyndman et al., 2020). Para Holt-Winter Additive and Multiplicative, o argumento apropriado teve que ser adicionado.

Para encontrar os hiperparâmetros dos modelos de aprendizado de máquina, usou-se um conjunto de validação composto por 10.000 amostras selecionadas aleatoriamente do conjunto de treinamento. Para o GBM, o menor SMAPE foi alcançado com 10.000 árvores. As outras configurações testadas foram 1.000, 2.000, 5.000 e 15.000 árvores. As redes neurais foram

treinadas com 20, 40, 80, 100 e 200 épocas usando os hiperparâmetros padrão do pacote R. O menor SMAPE obtido foi com 100 épocas.

Os LSTMs foram treinados usando o *framework Keras* e a mesma validação de 10.000 amostras. Os melhores resultados foram alcançados treinando os modelos para 80 épocas usando o otimizador Adam. O número de neurônios em cada camada é relatado na Tabela 4.1.

Antes da extração de características, todas as séries temporais foram normalizadas usando a função min-max(equação 4.1). A normalização é necessária devido às diferenças de magnitude das séries temporais. Ressalta-se que a normalização considerou apenas as observações de teste e não a observação a ser predita.

$$x_{transformada} = \frac{x - min}{max - min} \tag{4.1}$$

no qual $x$ é a observação da série temporal, $max$ é a maior observação daquela série temporal e $min$ é o menor valor dela. No total, sessenta características foram usadas para descrever cada série temporal. Elas foram extraídos usando o pacote *tsfeatures* do R (Hyndman et al., 2019), exceto os recursos de periodograma (pacote TSA) e comprimento (pacote base).

O modelo de aprendizado de máquina usado como meta-regressor foi o *random forest* disponível no *Scikit-learn* 0.23.1 (Pedregosa et al., 2011). Para encontrar os hiper-parâmetros ótimos deste modelo, dividiu-se o conjunto de dados de treinamento em 80% e 20% para treinamento e validação, respectivamente. O número de estimadores foi fixado em 500 e o MSE (Mean Squared Error) foi utilizado como critério de otimização, pois esta métrica pune modelos com erros maiores. A tabela 4.2 demonstra os parâmetros e valores usados no procedimento de validação.

Tabela 4.2: Hiper-parâmetros testados e valores usados no *hold-out* do *random forest*

| Parâmetro | Intervalo dos Valores |
|---|---|
| Máximo Profundidade da Árvore | [None, 5, 10, 15] |
| Amostra Mínima Por Nó | [2, 100, 200, 1000, 2000] |
| Máximo de Atributos | [auto, sqrt, log2] |
| Bootstrap | [True, False] |

Ao final, treinou-se mais de 1.800 *random forests* e seleciou-se as melhores configurações para cada um dos modelos, aquelas obtiveram o menor SMAPE para compor o meta-regressor. Os hiperparâmetros desses regressores de *random forest* são apresentados na tabela 4.3.

Então, para qualquer série temporal, o meta-regressor prevê o SMAPE esperado para todos os modelos de base. Assim, o usuário pode analisar o erro de cada modelo, deixando a escolha do modelo final a seu critério. Se o usuário não tem conhecimento prévio de modelagem de série temporal e/ou deseja automatizar a seleção, ele pode simplesmente escolher o modelo com o menor erro.

Tabela 4.3: Hiper-parâmetros das *random forests*

| Modelo | Prof. Árvore | Amostra Mín. | Atributos | Boostrap |
|---|---|---|---|---|
| ARIMA | None | 2 | sqrt | FALSE |
| SES | None | 2 | log2 | FALSE |
| ETS | 15 | 2 | log2 | FALSE |
| GBM | None | 2 | sqrt | FALSE |
| HW | None | 2 | log2 | FALSE |
| Holt-Winters Aditivo | None | 2 | log2 | FALSE |
| Holt-Winters Multiplicativo | 15 | 2 | log2 | FALSE |
| LSTM - 1 Camada Escondida | None | 2 | sqrt | FALSE |
| LSTM - 2 Camadas Escondidas | None | 2 | sqrt | FALSE |
| LSTM - 3 Camadas Escondidas | None | 2 | log2 | FALSE |
| ANN - 1 Neurônio | None | 2 | sqrt | FALSE |
| ANN - 10 Neurônios | None | 2 | sqrt | FALSE |
| ANN - 20 Neurônios | None | 2 | log2 | FALSE |
| ANN - Auto Neurônios | None | 2 | log2 | FALSE |
| Naïve | 15 | 2 | log2 | FALSE |

A figura 4.1 mostra como o meta-regressor pode ser implantado para o teste. Primeiro, o usuário fornece a série temporal e, em uma primeira etapa, o meta-regressor extrai suas 60 características. A segunda etapa é a estimativa SMAPES para cada um dos 15 modelos de série de tempo em que o meta-regressor foi treinado. Essa estimativa foi realizada usando um *random forest* para cada modelo de série temporal. Como saída do algoritmo, o usuário recebe os valores esperados de SMAPES de cada modelo de série temporal.

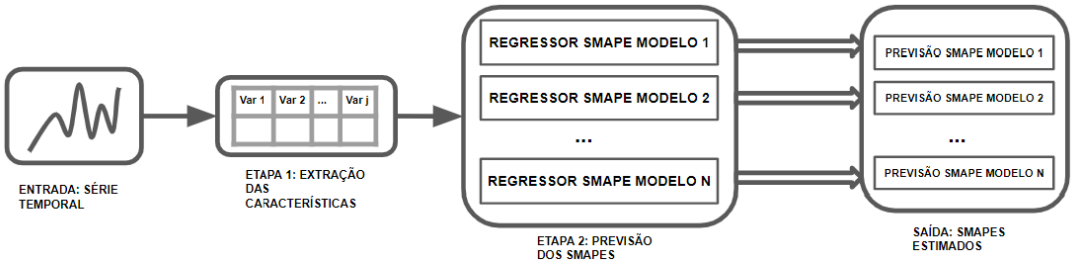


Figura 4.1: Representação do Meta-Regressor

## 4.2 BASE DE DADOS

O conjunto de dados utilizado neste trabalho foi extraído da M4-Competition (Makridakis et al., 2018). De acordo com a publicação oficial da competição, todas as 100.000 séries temporais da competição são reais e contínuas e envolvem dados de frequência alta (semanal, diário e horário) e dados de baixa frequência (anual, trimestral e mensal). A fonte da série temporal provém de domínios relevantes, como indústrias, serviços, turismo, importações e exportações,

demografia, educação, trabalho e salários, governo, famílias, títulos, ações, seguros, empréstimos, imóveis, transporte e meio ambiente. A Tabela 4.4 mostra o número de séries temporais por frequência de dados e domínio.

Tabela 4.4: Número de séries temporais por frequência e domínio

| Frequência | Microecon. | Indústria | Macroecon. | Finanças | Demografia | Outros | Total |
|---|---|---|---|---|---|---|---|
| Anual | 6.538 | 3.716 | 3.903 | 6.519 | 1.088 | 1.236 | 23.000 |
| Trimestral | 6.020 | 4.637 | 5.315 | 5.305 | 1.858 | 865 | 24.000 |
| Mensal | 10.975 | 10.017 | 10.016 | 10.987 | 5.728 | 277 | 48.000 |
| Semanal | 112 | 6 | 41 | 164 | 24 | 12 | 359 |
| Diária | 1.476 | 422 | 127 | 1.559 | 10 | 633 | 4.227 |
| Horária | 0 | 0 | 0 | 0 | 0 | 414 | 414 |
| Total | 25.121 | 18.798 | 19.402 | 24.534 | 8.708 | 3.437 | 100.000 |

O tamanho das séries temporais variam entre 17 a 4.752 observações, a mediana e o comprimento médio são de 104 e 246 observações, respectivamente. A Figura 4.2 mostra o histograma do comprimento da série temporal do conjunto de dados.

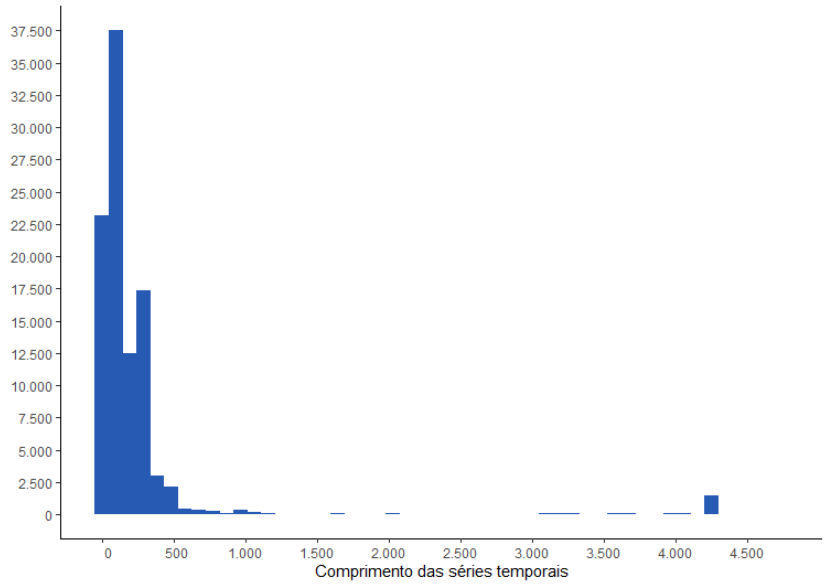


Figura 4.2: Histograma do comprimento das séries temporais

Das 100.000 séries temporais disponíveis, 7.249 contêm características com valores nulos e, portanto, foram removidos do conjunto de dados. Os 92.751 restantes foram usados em nossos experimentos. A tabela 4.5 mostra a distribuição final por periodicidade[1] Nota-se que com o filtro, não há mais séries temporais com frequência horária.

[1]Não há um rótulo para o domínio (microeconomia, indústria, finanças...) de cada série temporal, somente a informação oficial. Deste modo, não foi possível computar o número final de séries temporais de cada domínio.

Tabela 4.5: Número final de séries temporais por frequência

| Frequência | Número de Séries Temporais |
|---|---|
| Anual | 17.781 |
| Trimestral | 23.974 |
| Mensal | 47.967 |
| Semanal | 294 |
| Diária | 2.735 |
| Total | 92.751 |

# 5 RESULTADOS EXPERIMENTAIS

As 92.722 observações apresentadas na Seção 4.2 foram divididas em treinamento (80%) e teste (20%), garantindo-se uma representatividade similar das variáveis em ambas as bases. A tabela 5.1 mostra os quantis e a média dos SMAPE's para todos os modelos de base no conjunto de teste. A coluna "Melhor Modelo"indica quantas vezes o modelo base foi a melhor opção (menor erro) para uma determinada série temporal. As demais colunas, nos indicam medidas de posição dos SMAPE's de cada modelo. Com estes dados, podemos inferir a distribuição de erros de cada modelo. Como por exemplo, apesar do modelo Holt-Winters Aditivo ter menos séries temporais em que foi o melhor modelo do que o modelo *naïve*, observa-se na tabela 5.1 que na média (e também na mediana), o modelo Holt-Winters foi melhor que o modelo *naïve*. Além disto, a diferença interquartil (diferença entre o terceiro e o primeiro quartil) do modelo naïve é a maior de todas, indicando uma maior variância dos erros deste modelo.

Tabela 5.1: Resultado das previsões de séries temporais

| Modelo | Melhor Modelo | 1° Quartil SMAPE | Mediana SMAPE | Média SMAPE | 3° Quartil |
|---|---|---|---|---|---|
| ARIMA | 1.047 | 0,0052 | 0,0161 | 0,0531 | 0,05 |
| ETS | 1.072 | 0,0038 | 0,0119 | 0,0561 | 0,0406 |
| GBM | **2.077** | 0,0077 | 0,0226 | 0,0546 | 0,0615 |
| Holt-Winters | 1.368 | **0,0037** | 0,0120 | 0,0562 | 0,0407 |
| Holt-Winters Aditivo | 685 | 0,0038 | 0,0123 | 0,0563 | 0,0411 |
| Holt-Winters Multiplicativo | 989 | 0,0039 | 0,0125 | 0,0564 | 0,0416 |
| LSTM - 1 Camada Escondida | 1.427 | 0,0041 | 0,0118 | 0,0353 | **0,0347** |
| LSTM - 2 Camadas Escondidas | 1.363 | 0,0038 | **0,0116** | 0,0363 | 0,0354 |
| LSTM - 3 Camadas Escondidas | 1.700 | 0,0038 | 0,0118 | 0,0368 | 0,0357 |
| ANN - 1 Neurônio | 1.618 | 0,0040 | 0,0118 | **0,0352** | 0,0348 |
| ANN - 10 Neurônios | 784 | 0,0038 | 0,0117 | 0,0363 | 0,0357 |
| ANN - 20 Neurônios | 1,414 | 0,0038 | 0,0118 | 0,0368 | 0,0357 |
| ANN - Auto Neurônios | 1.370 | 0,0038 | 0,0118 | 0,0368 | 0,0362 |
| SES | 898 | 0,0045 | 0,0141 | 0,0565 | 0,042 |
| Naïve | 768 | 0,0231 | 0,0703 | 0,1614 | 0,198 |

A Figura 5.1 mostra a distribuição do SMAPE de três modelos: ARIMA, LSTM - 1 Camada Escondida e *naïve*. Como podemos observar, os dois primeiros modelos têm uma distribuição semelhante entre si. Elas têm assimetria à direita, com grande concentração nos valores menores. Contudo, a distribuição do SMAPE do *naïve* se difere de ambas, seus valores estão mais distribuídos ao logo de 0 e 1. Isso quer dizer que pode-se esperar de erros maiores desta última distribuição do que das demais.

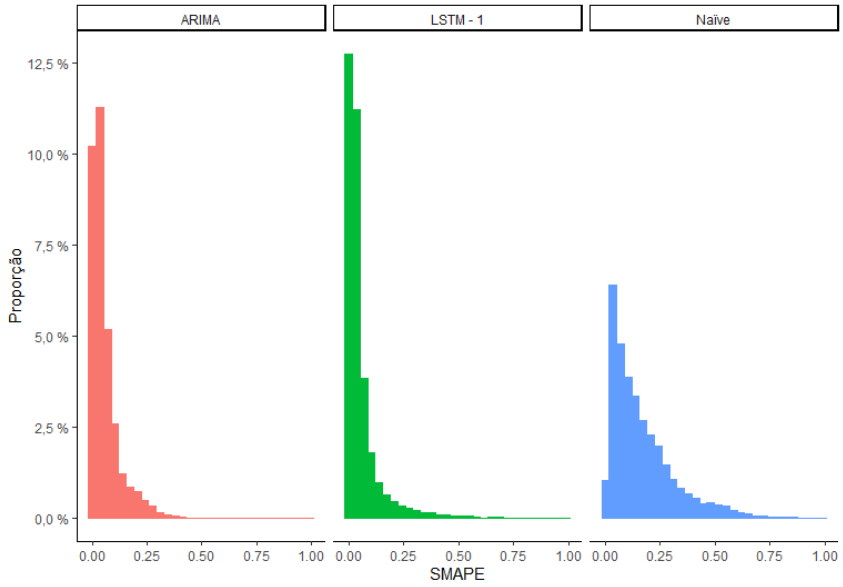


Figura 5.1: Exemplo da distribuição do SMAPE's da base de dados de teste

A dificuldade de criar um meta-classificador que preveja o melhor modelo reside no fato de que a diferença entre o SMAPE de dois regressores pode ser muito pequena e espúria. A tabela 5.2 mostra os quantis da diferença entre a menor SMAPE e a segunda menor SMAPE. Observa-se que o valor mínimo é zero, porque há 661 casos de empate.

Tabela 5.2: Quartis da diferença entre o menor e o segundo menor SMAPE

| Quartil | Diferença |
|---|---|
| 0% | 0 |
| 25% | 0,00022 |
| 50% | 0,001 |
| 75% | 0,004 |
| 100% | 0,7 |

Usando a metodologia apresentada na seção anterior, treinamos os *random forests*, que compuseram o meta-regressor proposto. A tabela 5.3 demonstra o REQM alcançado para cada regressor que compõe o meta-regressor. A mediana do REQM é cerca de 1,7%, que é cinco vezes menor que o meta-regressor proposto em (Saeed et al., 2017). Isso corrobora o bom poder de discriminação do conjunto de recursos adotado neste trabalho.

O melhor modelo para a respectiva série temporal foi o mesmo com o menor SMAPE previsto pelo meta-regressor em 11% das vezes, como visto na figura 5.2, categoria Top 1. Como mostrado na figura 5.3, na maioria das situações, a diferença dos modelos é espúria, assim, considerando o número de vezes que o melhor modelo encontra-se entre os dois modelos com o menor SMAPE predito é de 20% e entre os três menores SMAPES preditos é de 27,67%.

Tabela 5.3: A REQM dos regressores que compõe o meta-regressor

| Modelo | REQM |
|---|---|
| ARIMA | 1,682 |
| SES | 1,797 |
| ETS | 1,830 |
| GBM | 1,647 |
| Holt-Winters | 1,930 |
| Holt-Winters Aditivo | 1,887 |
| Holt-Winters Multiplicativo | 1,866 |
| LSTM - 1 Camada Escondida | 1,774 |
| LSTM - 2 Camadas Escondidas | 1,729 |
| LSTM - 3 Camadas Escondidas | 1,693 |
| ANN - 1 Neurônio | 1,500 |
| ANN - 10 Neurônios | 1,532 |
| ANN - 20 Neurônios | 1,603 |
| ANN - Auto Neurônios | 1,583 |
| Naïve | 1,556 |

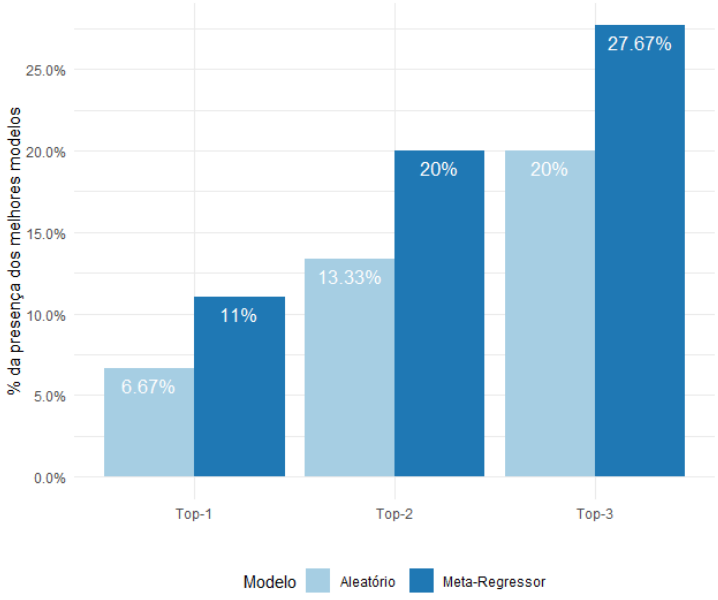


Figura 5.2: Presença do melhor modelo nas escolhas principais do meta-regressor em comparação com uma escolha aleatória

Quando o meta-regressor não prevê o melhor modelo, a mediana da diferença entre o SMAPE do melhor modelo e o modelo previsto com o menor SMAPE pelo meta-regressor é 0,005. Se o meta-regressor não sugerir o melhor modelo, a diferença mediana esperada é meio ponto percentual de SMAPE. Ressalta-se que um erro baixo, segundo Flores (1986), a qualidade da predição dependerá da aplicação da série temporal e dos riscos envolvidos. Em algumas aplicações, um erro de 5% pode ser considerado alto enquanto que em outras aplicações pode ser considerado baixo. A figura 5.3 mostra a distribuição dessas diferenças.

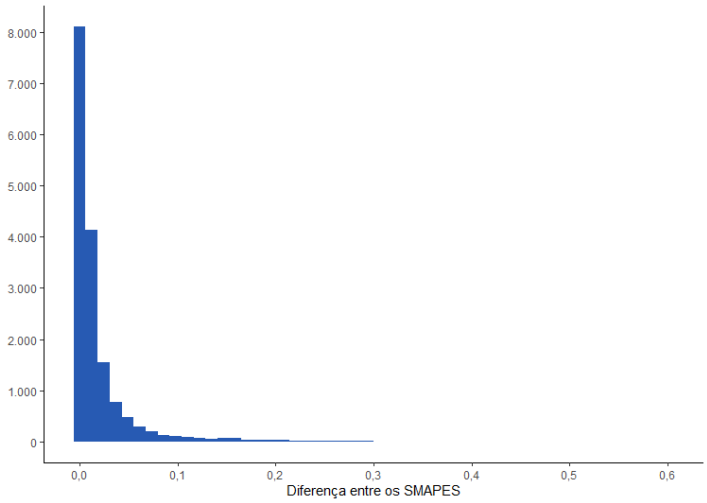


Figura 5.3: Distribuição das diferenças dos SMAPES do melhor modelo observado e do modelo sugerido pelo meta-regressor

Tabela 5.4: Soma das SMAPE de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia

| **Método** | **Soma dos SMAPES** |
|---|---|
| Oráculo | 270 |
| Método Proposto (Top 1) | 584 |
| ANN - 1 Neurônio | 702 |
| ANN - 10 Neurônios | 718 |
| ANN - Auto Neurônios | 726 |
| ANN - 20 Neurônio | 735 |
| LSTM - 3 Camadas Escondidas | 962 |
| LSTM - 2 Camadas Escondidas | 990 |
| LSTM - 1 Camadas Escondidas | 1.025 |
| ARIMA | 1.051 |
| GBM | 1.073 |
| ETS | 1.121 |
| Holt-Winters Aditivo | 1.129 |
| SES | 1.129 |
| Holt-Winters | 1.130 |
| Holt-Winters Multiplicativo | 1.134 |
| Naïve | 3.109 |

Para melhor avaliar os resultados do método proposto, calculamos o resultado do oráculo, que é um modelo abstrato que sempre seleciona o regressor da base que previu o menor erro. Esse é o limite superior do nosso meta-regressor. A tabela 5.4 compara a soma

de todas as séries temporais SMAPE no conjunto de dados de teste para o oráculo, o método proposto (considerando Top 1) e todos os modelos básicos (caso todas as séries temporais fossem modeladas pelo respectivo modelo). Como podemos observar, o meta-regressor é o mais próximo do oráculo e supera o melhor modelo de base em 20%, em termos de erro acumulado.

Tabela 5.5: Soma dos SMAPES de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia divido pela tamanho da série temporal

| **Método** | **Séries Curtas** | **Séries Longas** |
|---|---|---|
| Oráculo | 171,74 | 97,50 |
| Método Proposto (Top 1) | 365,63 | 219,22 |
| ARIMA | 659,55 | 391,80 |
| ETS | 572,88 | 548,21 |
| GBM | 613,23 | 460,14 |
| Holt-Winters | 582,82 | 547,63 |
| Holt-Winters Aditivo | 582,28 | 547,22 |
| Holt-Wintes Multiplicativo | 585,01 | 549,20 |
| LSTM - 1 Camada Escondida | 596,75 | 428,49 |
| LSTM - 2 Camadas Escondidas | 591,90 | 398,87 |
| LSTM - 3 Camadas Escondidas | 578,85 | 383,36 |
| ANN - 1 Neurônio | **448,51** | 254,40 |
| ANN - 10 Neurônios | 466,21 | **253,05** |
| ANN - 20 Neurônios | 481,90 | 253,41 |
| ANN - Auto Neurônios | 462,54 | 264,10 |
| SES | 581,57 | 547,95 |
| Naïve | 1891,99 | 1217,24 |

A tabela 5.5 mostra a soma dos SMAPES utilizando uma única metodologia dividindo as séries temporais em dois grupos: séries temporais curtas e séries temporais longas. As séries temporais curtas são as séries temporais que tem o seu tamanho menor do que a mediana do tamanho de todas as séries temporais do conjunto de teste (a mediana desta base é de 108 observações) e as séries temporais longas tem o seu tamanho maior que esta mediana. Nota-se que em ambos os grupos, o método proposto é o mais próximo do Oráculo, com um desempenho aproximadamente superior a 15% do ANN - 1 Neurônio em ambos os casos. Nota-se que todas as metodologias desempenharam melhor (no acumulado) em séries longas do que curtas, uma evidência de que quanto mais observações (informações), melhor a predição de um modelo de séries temporais.

O coeficiente de Hurst mede a memória de longo prazo da série temporal. Quanto maior esta memória, maior a previsibilidade da série temporal. A tabela 5.6 compara a utilização de um único método para todas as séries da base de teste, dividindo as séries em dois grupos: séries temporais com coeficiente de Hurst menor do que a mediana do coeficiente de Hurst de todas as séries temporais da base de teste (grupo denominado memória longa)[1] e as séries temporais com o coeficiente de Hurst maior que esta mediana (grupo denominado memória curta). Novamente,

[1] A mediana do coeficiente de Hurst para a base de teste é de 0,99.

a soma do SMAPE do método proposto é a mais próxima do Oráculo, comparando com os modelos individualmente. Nota-se também, que o grupo que contém as séries com o coeficiente de Hurst maior que a mediana têm um SMAPE médio menor em todos os casos. Nota-se também que os resultados vão ao enontro da teoria. Quanto maior a memória de longo prazo, menor o erro em todas as metodologias.

Tabela 5.6: Soma dos SMAPES de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia divido pelo coeficiente de Hurst da série temporal

| **Método** | **Memória Curta** | **Memória Longa** |
|---|---|---|
| Oráculo | 207,63 | 61,61 |
| Método Proposto (Top 1) | 449,62 | 135,23 |
| ARIMA | 812,50 | 238,85 |
| ETS | 779,09 | 342,00 |
| GBM | 738,01 | 335,36 |
| Holt-Winters | 788,32 | 342,13 |
| Holt-Winters Aditivo | 787,22 | 342,28 |
| Holt-Wintes Multiplicativo | 789,82 | 344,39 |
| LSTM - 1 Camada Escondida | 821,29 | 203,95 |
| LSTM - 2 Camadas Escondidas | 794,99 | 195,78 |
| LSTM - 3 Camadas Escondidas | 766,16 | 196,06 |
| ANN - 1 Neurônio | **539,72** | **163,20** |
| ANN - 10 Neurônios | 554,99 | 163,96 |
| ANN - 20 Neurônios | 569,81 | 165,50 |
| ANN - Auto Neurônios | 556,98 | 169,65 |
| SES | 782,12 | 347,40 |
| Naïve | 2257,38 | 851,85 |

Como dito anteriormente, a tendência da série temporal é uma fonte de variação da mesma. A tabela 5.7 mostra a soma dos SMAPES de dois grupos, com baixa tendência e alta tendência. Como nos casos anteriores, as séries de baixa tendência são aquelas com a tendência abaixo da mediana da base de teste e alta tendência caso contrário (a mediana da tendência foi de 0,97). Novamente, em ambos os grupos o método proposto ficou mais próximo do oráculo. Nota-se que em todos os métodos, a soma dos SMAPES é menor no grupo de baixa tendência. Todas as metodologias apresentaram um erro menor no grupo de alta tendência do que no grupo de séries temporais de baixa tendência.

Além da tendência, a sazonalidade também é uma fonte de variabilidade da série temporal. Dentro do conjunto de dados de teste, 3.553 séries temporais podem ser consideradas sazonais, o que representa 19,15% do conjunto. A tabela 5.8 mostra a média[2] dos SMAPES de dois grupos, séries temporais com sazonalidade e sem sazonlidade. Novamente, em ambos os

[2]Nas Tabelas 5.5, 5.6 e 5.7 foi usado a soma pois a coluna destas tabelas tinha o mesmo número de séries temporais, dado que foi utilizado a mediana (50% em cada coluna). No caso da Tabela 5.8, não foi dividido em medianas e sim na presença ou não da sazonalidade. Assim, para que as colunas ficassem comparáveis entre si (dado que elas não tem o mesmo número de séries temporais entre si), utilizou-se a média. O mesmo ocorre para a Tabela 5.9, que no caso foi divido pela periodicidade.

Tabela 5.7: Soma dos SMAPES de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia divido pela tendência da série temporal

| **Método** | **Alta Tendência** | **Baixa Tendência** |
|---|---|---|
| Oráculo | 56,57 | 212,67 |
| Método Proposto (Top 1) | 119,71 | 465,14 |
| ARIMA | 212,24 | 839,11 |
| ETS | 288,20 | 832,88 |
| GBM | 303,19 | 770,18 |
| Holt-Winters | 289,71 | 840,74 |
| Holt-Winters Aditivo | 289,55 | 839,96 |
| Holt-Wintes Multiplicativo | 291,62 | 842,59 |
| LSTM - 1 Camada Escondida | 178,59 | 846,66 |
| LSTM - 2 Camadas Escondidas | 169,68 | 821,09 |
| LSTM - 3 Camadas Escondidas | 168,71 | 793,50 |
| ANN - 1 Neurônio | **147,24** | **555,67** |
| ANN - 10 Neurônios | 148,26 | 570,70 |
| ANN - 20 Neurônios | 150,16 | 585,15 |
| ANN - Auto Neurônios | 151,36 | 575,28 |
| SES | 300,14 | 829,39 |
| Naïve | 800,17 | 2309,05 |

grupos, o método proposto ficou mais próximo do oráculo. Todas as metodologias tiveram um erro menor no grupo Não-Sazonal, com exceção das ANN's, que ou tiveram um erro menor no grupo Sazonal ou a média dos dois grupos ficou igual.

Considerando a base de teste e dividindo as séries temporais pela sua periodicidade, o método proposto consegue ter um SMAPE médio menor do que os demais modelos em todas as periodicidades, perdendo somente para o Oráculo, como visto na tabela 5.9.

Tabela 5.8: Média dos SMAPES de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia divido pela presença de sazonalidade na série temporal

| Método | Série Sazonal | Série Não-Sazonal |
|---|---|---|
| Oráculo | 0,015 | 0,014 |
| Método Proposto (Top 1) | 0,032 | 0,029 |
| ARIMA | 0,057 | 0,053 |
| ETS | 0,067 | **0,036** |
| GBM | 0,059 | 0,055 |
| Holt-Winters | 0,067 | 0,037 |
| Holt-Winters Aditivo | 0,066 | 0,038 |
| Holt-Wintes Multiplicativo | 0,067 | 0,038 |
| LSTM - 1 Camada Escondida | 0,059 | 0,042 |
| LSTM - 2 Camadas Escondidas | 0,056 | 0,042 |
| LSTM - 3 Camadas Escondidas | 0,054 | 0,044 |
| ANN - 1 Neurônio | **0,038** | 0,0388 |
| ANN - 10 Neurônios | 0,039 | 0,040 |
| ANN - 20 Neurônios | 0,039 | 0,042 |
| ANN - Auto Neurônios | 0,039 | 0,039 |
| SES | 0,070 | 0,040 |
| Naïve | 0,165 | 0,180 |

Tabela 5.9: SMAPE médio de todas as séries temporais no conjunto de dados de teste usando apenas uma metodologia divido pela periodicidade da série temporal

| Método | Anual | Trimestral | Mensal | Semanal | Diária |
|---|---|---|---|---|---|
| Oráculo | 0,01 | 0,01 | 0,02 | 0,01 | 0,00 |
| Método Proposto (Top 1) | 0,03 | 0,03 | 0,03 | 0,02 | 0,005 |
| ARIMA | 0,05 | 0,05 | 0,06 | 0,05 | 0,01 |
| ETS | 0,04 | 0,04 | 0,08 | 0,08 | 0,01 |
| GBM | 0,05 | 0,05 | 0,07 | 0,07 | 0,02 |
| Holt-Winters | 0,04 | 0,04 | 0,08 | 0,08 | 0,01 |
| Holt-Winters Aditivo | 0,04 | 0,04 | 0,08 | 0,08 | 0,01 |
| Holt-Wintes Multiplicativo | 0,04 | 0,04 | 0,08 | 0,08 | 0,01 |
| LSTM - 1 Camada Escondida | 0,04 | 0,05 | 0,07 | 0,04 | 0,01 |
| LSTM - 2 Camadas Escondidas | 0,04 | 0,05 | 0,06 | 0,04 | 0,01 |
| LSTM - 3 Camadas Escondidas | 0,04 | 0,05 | 0,06 | 0,04 | 0,01 |
| ANN - 1 Neurônio | 0,04 | 0,04 | 0,04 | 0,03 | 0,01 |
| ANN - 10 Neurônios | 0,04 | 0,04 | 0,04 | 0,03 | 0,01 |
| ANN - 20 Neurônios | 0,04 | 0,04 | 0,04 | 0,03 | 0,01 |
| ANN - Auto Neurônios | 0,04 | 0,04 | 0,04 | 0,03 | 0,01 |
| SES | 0,04 | 0,04 | 0,08 | 0,07 | 0,01 |
| Naïve | 0,18 | 0,16 | 0,18 | 0,12 | 0,03 |

# 6 CONCLUSÃO

Neste trabalho, foi proposto um meta-regressor para auxiliar na previsão de séries temporais. Espera-se que os ganhem mais agilidade no processo de previsão, economizando recursos como tempo e dinheiro e ganhando mais independência de processos computacionais caros e especialistas.

Como resultado, o meta-regressor obteve uma mediana dos erros absolutos médios de aproximadamente 1,7%. Este resultado é 5 vezes menor que o meta-regressor proposto por (Saeed et al., 2017). O meta-regressor também reduziu o erro do melhor modelo de série temporal em 18% no conjunto de dados de teste e foi a melhor opção em várias segmentações (tamanho, coeficiente de Hurst, tendência e periodicidade). Além de prever corretamente o melhor modelo 11% contra 6% de escolher o melhor modelo aleatoriamente. Se o meta-regressor falhar em prever o melhor modelo, a mediana da diferença SMAPE do modelo Top-1 e o melhor modelo é 0,005.

As próximas etapas para melhorar este meta-regressor incluem adicionar novos modelos de previsão de série temporal, dando ao usuário mais opções. Outro ponto que deve ser investigado diz respeito a novas variáveis que ajudam a explicar melhor o processo de regressão dos erros do modelo e o ajuste das variáveis já existentes. Muitas variáveis têm parâmetros padrões que podem ser calibrados para uma melhor precisão do regressor. Além disso, outros hiper-parâmetros da *random forest* e LSTM devem ser testados, além de outros modelos de regressão, como uma redes neurais. Neste trabalho, foi considerado somente a última observação de cada série temporal. Assim, adicionar mais observações de cada série temporal poderia enriquecer a base de treinamento. Ademais, outro ponto de melhoria poderia ser o enriquecimento da base de treinamento com séries temporais de outras fontes de dados.

Por ser uma aplicação computacional, quando possui diversos modelos disponíveis, o usuário pode escolher apenas os modelos de séries temporais dentro do seu escopo de trabalho, agilizando o processo de resposta do modelo. Além disso, em vez de ter todos os SMAPES, o usuário pode escolher como resposta final apenas o modelo que apresenta o menor erro de previsão.

Outra aplicação do meta-regressor é que ele pode fazer parte de um sistema. O meta-regressor é responsável por decidir qual sistema usará o modelo de previsão de série temporal. Neste caso, um nível de tolerância pode ser configurado para o uso de um modelo de série temporal mais complexo, como, por exemplo, apenas usando LSTM se o erro ARIMA for 5% maior (esta tolerância dependerá de fatores como o risco de erro, recursos computacionais e tempos de resposta, por exemplo).

Um benefício extra deste meta-regressor, além de economizar tempo, dinheiro e processamento computacional, ao reduzir a dependência de *hold-out* e contratar um especialista,

é que quando há um maior número de modelos de séries preditivas de tempo, o usuário saberá qual erro esperar de sua série temporal, reduzindo a pressão interna e externa por resultados que podem ser improváveis.

# REFERÊNCIAS


Adebiyi, A., Adewumi, A. e Ayo, C. (2014). Stock price prediction using the arima model. *Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014*.

Adèr, H., Hand, D. e Mellenbergh, G. (2008). *Advising on Research Methods: A Consultant's Companion*. Johannes Van Kessel Publishing.

Andersen, R. S. e Risør, M. B. (2014). The importance of contextualization. anthropological reflections on descriptive analysis, its limitations and implications. *Anthropology & Medicine*, 21(3):345–356. PMID: 24484056.

Atkinson, R. W., Kang, S., Anderson, H. R., Mills, I. C. e Walton, H. A. (2014). Epidemiological time series studies of pm2.5 and daily mortality and hospital admissions: a systematic review and meta-analysis. 69(7):660–665.

Barber, D., Cemgil, A. e Chiappa, S. (2011). *Bayesian Time Series Models*. Bayesian Time Series Models. Cambridge University Press.

Barbosa-Filho, N., Silva, J., Goto, F. e Silva, B. (2011). *Crescimento econômico, acumulação de capital e taxa de câmbio*, páginas 1–30.

Ben Taieb, S., Bontempi, G., Atiya, A. e Sorjamaa, A. (2011). A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition. *Expert Systems with Applications*, 39.

Bengio, Y., Courville, A. e Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Bisgaard, S. e Kulahci, M. (2011). *Time Series Analysis and Forecasting by Example*. Wiley Series in Probability and Statistics. Wiley.

Boehmke, B. C. e Greenwell, B. M. (2019). Hands-on machine learning with r.

Box, G., Jenkins, G. e Day, H. (1976). *Time Series Analysis: Forecasting and Control*. Holden-Day series in time series analysis and digital processing. Holden-Day.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Brigham, E. F. e Ehrhardt, M. C. (2016). *Administração Financeira: teoria e prática*. Cengage Learning.

Britz, D. (2016). Recurrent neural networks tutorial, part 1 – introduction to rnns.

Brodbeck, P. (2020). Coronavírus: Poluição do ar em curitiba cai após medidas de isolamento social.

Cai, M., Pipattanasomporn, M. e Rahman, S. (2019). Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. *Applied Energy*, 236:1078–1088.

Chatfield, C. (1978). The holt-winters forecasting procedure. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 27(3):264–279.

Chen, J., Zeng, G.-Q., Zhou, W., Du, W. e Lu, K.-D. (2018). Wind speed forecasting using nonlinear-learning ensemble of deep learning time series prediction and extremal optimization. *Energy Conversion and Management*, 165:681–695.

Choi, J., Roberts, D. e Lee, E. (2015). Forecasting oil production in north dakota using the seasonal autoregressive integrated moving average (s-arima). *Natural Resources*, 6:16–26.

Chollet, F. et al. (2015). Keras.

Crone, S., Hibon, M. e Nikolopoulos, K. (2011). Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction. *International Journal of Forecasting*, 27:635–660.

Cryer, J. e Chan, K. (2008). *Time Series Analysis: With Applications in R*. Springer Texts in Statistics. Springer.

Daitan (2019). Exponential smoothing methods for time series forecasting. *Medium*.

Dantas, T. M., Cyrino Oliveira, F. L. e Varela Repolho, H. M. (2017). Air transportation demand forecast through bagging holt winters methods. *Journal of Air Transport Management*, 59:116 – 123.

Duan, Y. e Yisheng, L. (2016). Travel time prediction with lstm neural network. páginas 1053–1058.

Engle, R. e Engle, C. (1995). *ARCH: Selected Readings*. Advanced texts in econometrics. Oxford University Press.

Fischer, T. e Krauss, C. (2017). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270.

Flores, B. E. (1986). A pragmatic view of accuracy measurement in forecasting. *Omega*, 14(2):93–98.

Fu, R., Zhang, Z. e Li, L. (2016). Using lstm and gru neural network methods for traffic flow prediction. páginas 324–328.

Gala, P., Araújo, E. e Bresser-Pereira, L. (2011). *Poupança doméstica e externa e a taxa de câmbio*, páginas 77–101.

Gardner, E. S. (2006). Exponential smoothing: The state of the art—part ii. *International Journal of Forecasting*, 22(4):637 – 666.

Gonzalez, J. e Yu, W. (2018). Non-linear system modeling using lstm neural networks. *IFAC-PapersOnLine*, 51(13):485 – 489. 2nd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems MICNON 2018.

Guerra, S., Prudencio, R. e Ludermir, T. (2008). Predicting the performance of learning algorithms using support vector machines as meta-regressors. 5163:523–532.

Haykin, S. (2007). *Redes Neurais: Princípios e Prática*. Artmed.

Ho, S., Xie, M. e Goh, T. (2002). A comparative study of neural network and box-jenkins arima modeling in time series prediction. *Computers  Industrial Engineering*, 42(2):371 – 375.

Hochreiter, S. e Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hua, Y., Zhifeng, Z., Li, R., Chen, X., Liu, Z. e Zhang, H. (2018). Deep learning with long short-term memory for time series prediction.

Hyndman, R. e Athanasopoulos, G. (2014). *Forecasting: principles and practice*. OTexts.

Hyndman, R., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E. e Yasmeen, F. (2020). *forecast: Forecasting functions for time series and linear models*. R package version 8.12.

Hyndman, R., Kang, Y., Montero-Manso, P., Talagala, T., Wang, E., Yang, Y. e O'Hara-Wild, M. (2019). *tsfeatures: Time Series Feature Extraction*. R package version 1.0.1.

Hyndman, R. e Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software*, 26.

James, G., Witten, D., Hastie, T. e Tibshirani, R. (2014). *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated.

Janssen, N. A. H., Brunekreef, B., van Vliet, P., Aarts, F., Meliefste, K., Harssema, H. e Fischer, P. (2003). The relationship between air pollution from heavy traffic and allergic sensitization, bronchial hyperresponsiveness, and respiratory symptoms in dutch schoolchildren. *Environmental Health Perspectives*, 111(12):1512–1518.

Jere, S., Kasense, B. e Chilyabanyama, O. (2017). Forecasting foreign direct investment to zambia: A time series analysis. *Open Journal of Statistics*, 07:122–131.

Kan, H. e Chen, B. (2003). Air pollution and daily mortality in shanghai: A time-series study. *Archives of environmental health*, 58:360–7.

Kang, E. (2017). Long short-term memory (lstm): Concept.

Knill, O. (2009). Probability and stochastic processes with applications.

Koning, M. e Smith, C. (2017). *Decision Trees and Random Forests: A Visual Introduction for Beginners*. Amazon Digital Services LLC - Kdp Print Us.

Kwiatkowski, D., Phillips, P. C. B., Schmidt, P. e Shin, Y. (1992). 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*, 54(1-3):159–178.

Kück, M., Crone, S. F. e Freitag, M. (2016). Meta-learning with neural networks and landmarking for forecasting model selection an empirical evaluation of different feature sets applied to industry data. *2016 International Joint Conference on Neural Networks (IJCNN)*, páginas 1499–1506.

Lemke, C. e Gabrys, B. (2010). Meta-learning for time series forecasting and forecast combination. *Neurocomputing*, 73:2006–2016.

Li, L. (2019). Classification and regression analysis with decision trees.

Li, X., Xie, H., Wang, R., Cai, Y., Cao, J., Wang, F., Min, H. e Deng, X. (2016). Empirical analysis: stock market prediction via extreme learning machine. *Neural Computing and Applications*, 27:67–78.

li Wang, J., hang Zhang, Y., Shao, M., lin Liu, X., min Zeng, L., lan Cheng, C. e feng Xu, X. (2004). Chemical composition and quantitative relationship between meteorological condition and fine particles in beijing. *Journal of Environmental Sciences*, 16:860–864.

Liu, Y., Wang, Y., Ma, W. e Zhang, L. (2017). Short-term travel time prediction by deep learning: A comparison of different lstm-dnn models. páginas 1–8.

Makridakis, S. e Hibon, M. (2000). The m3-competition: results, conclusions and implications. *International Journal of Forecasting*, 16:451–476.

Makridakis, S., Spiliotis, E. e Assimakopoulos, V. (2018). The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*.

Marconi, N. (2011). *MARCONI, N. ; BARBI, F. . Taxa de câmbio e composição setorial da produção. In: Marcio Holland; Yoshiaki Nakano. (Org.). Taxa de câmbio no Brasil - estudos de uma perspectiva do desenvolvimento econômico. Rio de Janeiro: Elsevier Editora, 2011, v. , p. 31-75.*

McElroy, T. S. (2017). Time series econometrics klaus neusser springer international publishing, 2016, xxiv + 409 pages, hardcover isbn: 978-3-319-32861-4. *International Statistical Review*, 85(1):3.

Metcalfe, A. V. e Cowpertwait, P. S. (2009). *Introductory Time Series with R*. Springer New York.

Mills, T. C. (2019). Chapter 1 - time series and their features. Em Mills, T. C., editor, *Applied Time Series Analysis*, páginas 1 – 12. Academic Press.

Mills, T. C. e Markellos, R. N. (2008). *The Econometric Modelling of Financial Time Series*. Cambridge University Press, 3 edition.

Mittal, A. (2019). Understanding rnn and lstm.

Moayedi, H. e Masnadi-Shirazi, M. (2008). Arima model for network traffic prediction and anomaly detection. *Proceedings of the International Symposium on Information Technology*, 4:1 – 6.

Mohri, M., Rostamizadeh, A. e Talwalkar, A. (2012). *Foundations of Machine Learning*. The MIT Press.

Monfared, M., Ghandali, R. e Esmaeili, M. (2014). A new adaptive exponential smoothing method for non-stationary time series with level shifts. *Journal of Industrial Engineering International*, 10:209–216.

Montgomery, D., Jennings, C. e Kulahci, M. (2008). *Introduction to Time Series Analysis and Forecasting*.

Namdeo, A. e Stringer, C. (2008). Investigating the relationship between air pollution, health and social deprivation in leeds, uk. *Environment international*, 34:585–91.

Neidell, M. (2004). Air pollution, health, and socio-economic status: The effect of outdoor air quality on childhood asthma. *Journal of health economics*, 23:1209–36.

Nielsen, M. A. (2018). Neural networks and deep learning.

Olah, C. (2015). Understanding lstm networks.

Ord, K. (2001). *International Journal of Forecasting*, 17:537–584.

Papacharalampous, G., Tyralis, H. e Koutsoyiannis, D. (2018). Predictability of monthly temperature and precipitation using automatic time series forecasting methods. *Acta Geophysica*, 66:807–831.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. e Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Phi, M. (2020). Illustrated guide to lstm's and gru's: A step by step explanation.

Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6(3):21–45.

Prudêncio, R. e Ludermir, T. (2004). Meta-learning approaches to selecting time series models. *Neurocomputing*, 61:121–137.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rose, C., Richard A. Davis, P., Brockwell, P., Davis, R., Smith, M., Davis, R., Calder, M. e (Firm), S. (2002). *Introduction to Time Series and Forecasting*. Número v. 1 em Introduction to Time Series and Forecasting. Springer.

Saeed, A., Khan, A., Zameer, A. e Usman, A. (2017). Wind power prediction using deep neural network based meta regression and transfer learning. *Applied Soft Computing*, 58.

Sagheer, A. e Kotb, M. (2018). Time series forecasting of petroleum production using deep lstm recurrent networks. *Neurocomputing*, 323.

Schmidhuber, J. (1987). Evolutionary principles in self-referential learning. on learning now to learn. Diploma thesis, Technische Universitat Munchen, Germany.

Shah, C. (1997). *International Journal of Forecasting*, 13:489–500.

Shmueli, G. (2011). To explain or to predict? *Statistical Science*, 25.

Shmueli, G. e Lichtendahl, K. (2016). *Practical Time Series Forecasting with R: A Hands-On Guide [2nd Edition]*. Practical Analytics. Axelrod Schnall Publishers.

Shukla, L. (2019). "part i-choosing a machine learning model".

Siami Namini, S. e Siami Namin, A. (2018). Forecasting economics and financial time series: Arima vs. lstm.

Singh, H. (2018). Understanding gradient boosting machines.

Skander Hannachi, P. (2019). 3 facts about time series forecasting that surprise experienced machine learning practitioners.

Talagala, T. S., Hyndman, R. J. e Athanasopoulos, G. (2018). Meta-learning how to forecast time series. *Monash Business School*.

Tattar, P., Ojeda, T., Murphy, S. P., Bengfort, B. e Dasgupta, A. (2017). *Practical Data Science Cookbook - Second Edition: Data Pre-Processing, Analysis and Visualization Using R and Python*. Packt Publishing, 2nd edition.

Teixeira, L. e Mattos, E. (2010). Câmbio e arrecadação municipal no Brasil: uma análise empírica de 2004 a 2007. (251).

Trochim, W. M. (2020). *Research Methods Knowledge Base*. Conjointly.

Valim, C. E. (2020). Até onde o dólar sobe?

Venables, W. e Ripley, B. (2002). *Modern Applied Statistics with S*. Springer-Verlag New York, 4th edition.

Wang, X., Smith-Miles, K. e Hyndman, R. (2009). Rule induction for forecasting method selection: Meta-learning the characteristics of univariate time series. *Neurocomputing*, 72:2581–2594.

Wang, Y., Zhou, J., Chen, K., Wang, Y. e Liu, L. (2017). Water quality prediction method based on lstm neural network. páginas 1–5.

Willmott, C. e Matsuura, K. (2005). Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate Research*, 30:79.

Woodruff, T. J., Grillo, J. e Schoendorf, K. C. (1997). The relationship between selected causes of postneonatal infant mortality and particulate air pollution in the united states. *Environmental Health Perspectives*, 105(6):608–612.

Yan, H. e Ouyang, H. (2017). Financial time series prediction based on deep learning. *Wireless Personal Communications*, 102:1–18.

Yan-ming Yang, Hui Yu e Zhi Sun (2017). Aircraft failure rate forecasting method based on holt-winters seasonal model. Em *2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, páginas 520–524.

Yoo, J., Kwon, J. e Choe, Y. (2014). Predictable internal brain dynamics in eeg and its relation to conscious states. *Frontiers in neurorobotics*, 8:18.

Zani, T. B., Zanini, F. A. M. e Zani, J. (2010). Governança corporativa e conflito de agência: Estudo de caso sobre a utilização de derivativos cambiais por cinco grandes empresas brasileiras. *EnANPAD*. Acessado: 2020-05-01.

Zeger, S. L., Irizarry, R. e Peng, R. D. (2006). On time series analysis of public health and biomedical data.

Zhou, H., Zhang, Y., Yang, L., Liu, Q., Yan, K. e Du, Y. (2019). Short-term photovoltaic power forecasting based on long short term memory neural network and attention mechanism. *IEEE Access*, PP:1–1.