UNIVERSIDADE FEDERAL DO ABC

TRABALHO DE GRADUAÇÃO EM
ENGENHARIA DE INFORMAÇÃO

Estudo comparativo entre modelos de Aprendizado de Máquina em Séries Temporais Hierárquicas: uma aplicação na previsão de demanda.

Victor de Oliveira

Santo André, SP
2021

Victor de Oliveira

Estudo comparativo entre modelos de Aprendizado de Máquina em Séries Temporais Hierárquicas: uma aplicação na previsão de demanda.

Monografia apresentada ao curso de Engenharia de Informação da Universidade Federal do ABC como parte dos requisitos para a obtenção do grau de Engenheiro de Informação.

UNIVERSIDADE FEDERAL DO ABC

Orientador:
Prof. Dr. Ricardo Suyama

Santo André, SP
2021

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Previsão em séries temporais é um tópico de estudo amplamente trabalhado na comunidade científica, variando seu campo de atuação entre economia, saúde, automação, energia, etc., sendo de extrema importância para planejamento e antecipação de eventuais demandas nestes setores. Todavia, ao modelar séries temporais surgem desafios não aparentes em outros tipos de conjuntos de dados como aspectos sazonais, correlação entre pontos adjacentes, estacionariedade e tendência. A dificuldade se torna ainda maior em séries temporais hierárquicas, onde temos diferentes níveis de agregação e múltiplas séries necessitam de previsões coerentes. Modelos paramétricos foram desenvolvidos e amplamente explorados nas últimas décadas a fim de capturar tais propriedades e gerar previsões confiáveis. Aprendizado de Máquina, por sua vez, tem ganhado adoção cada vez maior na comunidade científica em áreas como reconhecimento de imagem, tradução automática e sistemas de recomendação, e mais recentemente na área de previsão em séries temporais. Esta crescente utilização vem de sucessos obtidos ao mapear problemas complexos e gerar resultados não vistos antes, principalmente na área de reconhecimento de imagem e texto. Neste contexto, este trabalho busca aprofundar a utilização de modelos de Aprendizado de Máquina eficientes como *Gradient Boosting* e *Random Forests* em séries hierárquicas realizando um estudo comparativo com métodos clássicos na área como *ARIMA* e *ETS* com dois conjuntos de dados na área de previsão de demanda.

Abstract

Time series forecasting is of extreme importance for fields such as economy, health, automation, energy, etc., being fundamental for planning demand in these sectors. However, time series has properties such as seasonal effects, stationary, trend, autocorrelation that are difficult to model properly. The challenge becomes even bigger when the topic is hierarchical forecasting where each aggregation level requires coherent predictions. Standard approach in forecasting comes from generating parametric models that were built specifically to deal with those intrinsic properties in time dependent data. Machine Learning, however, has increased its adoption through areas such as image recognition and text processing, and only in previous years its use has being widening for time series data. The reason for this increasing adoption comes majorily from recent sucess in image recognition. In this scenario, this work develop a study comparing Machine Learning and statistical models in hierarchical forecasting with two datasets from demand prediction.

CAPÍTULO 1

Introdução

1.1 Motivação

Aprendizado de Máquina (AM) tem ganhado cada vez mais espaço na indústria nos últimos anos, devido a sucessos em diversas aplicações reais como Reconhecimento de Voz e Imagem, Robôs Inteligentes, Tradução Automática, Veículos Autônomos e Sistemas de Recomendação [2]. Algoritmos de AM trabalham com reconhecimento de padrões a partir de conjuntos de dados, variando suas técnicas entre estimar parâmetros a partir de um conjunto de dados ruidosos, mapeamento de estruturas complexas lineares e não lineares, inferência de distribuição de probabilidades, e previsão de categorias ou valores reais, isto é, tarefas de classificação e regressão, respectivamente [3]. Porém, quando o conjunto de dados fornece uma característica temporal, isto é, existem propriedades intrínsicas relacionados ao tempo, abordagens adequadas precisam ser empregadas para modelar estas propriedades, bem como avaliar a eficácia de métodos estatísticos e de AM uma vez que o fator tempo entra como parte do treinamento do modelo.

Uma série temporal é um conjunto de dados coletados ao longo do tempo, onde assume-se que a sua estrutura apresenta uma correlação entre pontos adjacentes ou apresentam forte dependência de valores passados, além de propriedades como sazonalidade, tendência e estacionariedade. Tudo isto acaba impondo dificuldades em métodos estatísticos tradicionais que normalmente assumem dados coletados de forma independente e identicamente distribuídos [4]. Entre os métodos mais utilizados para previsão em séries temporais estão ARIMA (*Autoregressive Integrated Moving Average*, do inglês), suavização exponencial de *Holt-Winters*, passeio aletório com drift e ES (exponential smoothing, em inglês) [5].

Métodos estatísticos e algoritmos de AM buscam resolver o mesmo problema: minimizar uma função de custo baseado em algum critério de otimização, com alguns exemplos claros de intersecção entre os campos como Regressão Linear Múltipla [6], para exemplificar. Entretanto, aplicações de algoritmos de AM podem tratar melhor problemas com relações não-lineares e exigem maior capacidade computacional. Mas a verdadeira eficácia de cada método não fica claro. Assim, é necessário a criação de estudos comparativos entre métodos clássicos em estatística com aqueles mais recentes atribuídos a área de AM, e mais precisamente, com os sucessos atuais de algoritmos de Aprendizagem Profunda (Deep Learning, em inglês) como arquiteturas LSTM (*Long-Short Term Memory*, em inglês), principalmente na área de previsão [7].

Insipirado então pelos trabalhos em [7], [8] e [9], este projeto pretende utilizar métodos estatísticos (*ARIMA* e *Holt-Winters*) e de AM (*Gradient Boosting* e *Random Forests*) para previsão de séries temporais hierárquicas, especificamente na previsão de demanda em diferentes tipos de estabelecimentos com dados publicados online. Séries temporais hierárquicas surgem a partir de agregações possíveis em uma cadeia de previsões, por exemplo, em grandes lojas varejistas temos vendas regionais, que podem ser divididas em categorias dos produtos e/ou lojas. Assim, o desafio é gerar previsões que sejam coerentes ao longo da hierarquia, isto é, que a soma dos nós inferiores reflita no valor predito para os nós superiores. Previsão de demanda, seja de produtos, ou pessoas, tem importância vital para a indústria e sobrevivência das empresas em um mercado altamente competitivo, sendo assim essencial uma estimativa aproximada de quanto um estabelecimento irá vender para evitarmos problemas de estoque. Portanto, queremos determinar, utilizando conjuntos de dados com características diferentes, quais métodos são mais apropriados em cada caso e avaliar a acurácia das técnicas empregadas bem como questões relacionadas a facilidade de implementação, complexidade computacional do algoritmo utilizado e interpretabilidade dos mesmos.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo deste trabalho consiste em realizar um estudo comparativo entre técnicas tradicionais aplicadas a séries temporais e aquelas obtidas no campo de AM para uma previsão precisa de demanda em séries temporais hierárquicas. Os modelos estatísticos utilizados são *ARIMA* e *Holt-Winters*, enquanto os modelos de AM são *Gradient Boosting* e *Random Forests*.

1.2.2 Objetivos Específicos

Abaixo são citados os objetivos específicos que este trabalho visa alcançar para atingir o objetivo geral.

- Aplicação de quatro modelos (*ARIMA*, *Holt-Winters*, *Gradient Boosting* e *Random Forests*) em séries temporais hierárquicas para previsão de demanda em dois conjuntos de dados distintos.
- Avaliar quais métodos de reconciliação tem resultados melhores na geração de previsões coerentes.
- Realização de análises exploratórias no conjunto de dados a fim de entender especifidades nos dados obtidos e o quanto conseguimos generalizar o estudo a partir dos mesmos.

1.3 Organização do trabalho

Este trabalho está organizado da seguinte maneira: O Capítulo 2 traz uma revisão da literatura na área de previsão em séries temporais, com foco na previsão de demanda em estruturas temporais hierárquicas. O Capítulo 3 desenvolve a fundamentação teórica apresentando os modelos a serem empregados, bem como os tipos existentes de estruturas hierárquicas, finalizando a discussão tratando sobre seleção de modelos e métricas a serem avaliadas. O capítulo 4 apresenta os resultados e simulações computacionais, e por fim, no capítulo 5 temos a conclusão do trabalho.

CAPÍTULO 2

Revisão da Literatura

Técnicas para previsão de demanda tem uma série de aplicações práticas, variando desde o setor da saúde, consumo de energia elétrica e água em áreas residenciais, vendas de produtos perecíveis em mercados, vendas de varejo online e físico, entre outros. Nesta seção, iremos tratar e revisar alguns artigos que trabalham o tema e utilizam tanto técnicas de AM, como técnicas estatísticas para resolver o problema em diferentes frentes.

Em [10] os autores realizam um estudo comparativo de técnicas conhecidas em análise de séries temporais para previsão de demanda no setor de Comida e Bebida. O principal objetivo é predizer a quantidade demandada de um produto específico em uma região. O estudo avalia três modelos estatísticos: Análise de tendência, decomposição e método de Holt-Winters (HW) otimizando uma métrica de acurácia baseado no erro percentual do modelo de regressão (MAPE, *Mean Absolute Percentage Error*, em inglês), definido na seção 3.4.2. Os dados coletados tem um intervalo de três anos (2013-2015) e todos os modelos são treinados com dados de 2013-2014 e sua desempenho avaliada nos dados de 2015. A tabela 2.1 resume a acurácia obtida para cada modelo e produto oferecido. Os valores da métrica variam no intervalor entre 0 e 100%.

Ainda segundo os autores, os resultados são satisfatórios para modelos de Decomposição e HW, pois estes são capazes de capturar características em séries com tendência, ciclos e sazonalidades ao mesmo tempo, enquanto que modelos de tendência só conseguem capturar a direção de movimento da série.

A realização de previsão de demanda tem uma aplicação vital também na indústria de geração de energia. Em um mundo cada vez mais competitivo e com uma dependência do estilo de vida humano baseado na adoção e uso de energia elétrica, se torna essencial saber quanto e quando teremos picos de requisição de carga. Realizar previsão de carga, isto é, de energia, tem sido

Tipo da Bebida	Análise de Tendência	Decomposição	HW
Black Mulb. J.	2.77	1.22	2.29
Lemonade	2.88	0.47	0.59
Blue Berry Sh.	2.46	1.32	1.50
Raspberry J.	2.20	1.80	1.72
Currant Sh.	2.38	0.47	1.53
Black Mulb. Sh.	2.26	1.58	1.50
Blue Berry Jam	2.36	2.58	2.34

Tabela 2.1: Valores de acurácia (MAPE) para os métodos utilizados.

amplamente estudado na literatura, principalmente com a utilização de modelos baseados em séries temporais. Porém, características não-lineares dos dados e a influência de diversos fatores distintos, como clima, horário do dia, tempo, etc., acabam por dificultar a tarefa destes modelos, altamente baseados em premissas de linearidade [11]. Com uma perspectiva então de contornar essas adversidades, os autores em [12] propuseram um ensemble de modelos de *Deep Learning* com uma camada de SVR (*Support Vector Regression*, em inglês) para prever a quantidade requisitada de energia em um determinado dia.

O método envolve em treinar uma Rede Neural em forma de um *autoencoder*, isto é, a arquitetura da rede é configurada para copiar a entrada até sua saída, impondo um *estrangulamento* na representação dos dados na camada intermediária, gerando um vetor compactado da sua entrada original. Utilizamos então esta camada compactada como dados de entrada para o modelo a ser utilizado. Porém, uma das dificuldades ao se utilizar redes neurais é sua convergência. Para contornar a situação, é proposto então um modelo de *ensemble*, isto é, combinam-se modelos distintos para uma predição mais confiável e estável. São gerados vinte modelos diferentes, com inicialização dos pesos das redes mudando a cada iteração, com cada vetor de um modelo distinto, gera-se uma matriz X. Essa matriz serve então como entrada para a SVR que gerará o resultado final. Por fim calcula-se o RSME (*Root Mean Squared Error*, em inglês) e o MAPE para diferentes conjuntos de dados de demanda coletados.

As tabelas 2.2 e 2.3 ilustram os resultados obtidos em dois tipos de dados testados, e uma comparação direta com outros modelos comumente utilizados, como o próprio SVR e Redes Neurais *Feed-Forward*.

Métrica	SVR	FNN	DBN	Proposto
CV RMSE	75.5476	99.4513	90.4974	**59.8561**
RMSE	74.3053	95.8105	90.2061	**72.2545**
CV MAPE	2.25%	3.00%	2.73%	**1.79%**
MAPE	2.83%	4.12%	3.50%	**2.71%**

Tabela 2.2: Resultados obtidos para demanda de carga em South Wales.

Métrica	SVR	FNN	DBN	Proposto
CV RMSE	40.6467	36.8863	33.1023	**26.6159**
RMSE	44.6742	38.8585	35.9375	**30.5989**
CV MAPE	3.64%	4.38%	3.74%	**3.35%**
MAPE	5.30%	6.22%	5.70%	**4.98%**

Tabela 2.3: Resultados obtidos para demanda de carga no Sul da Australia.

Os resultados mostrados acima evidenciam a eficácia da metodologia proposta. Ao combinar a predição de diversos modelos, acurácia aumenta em todas as métricas, mostrando que se temos o único objetivo de diminuir erros, o método de *ensemble* prevalece.

Outra aplicação de previsão amplamente implementada na literatura é na área de vendas. Previsão de vendas tem uma ligação direta com planejamento de demanda, principalmente no ramo varejista. Podemos citar, por exemplo, na área de gerenciamento de suprimentos, aplicações para antecipação de reposição de estoque, campanhas de marketing mais eficientes e impacto de produtos novos no mercado [13]. Os autores de [13] realizam um estudo comparativo de diferentes técnicas de AM aplicados à previsão de vendas no ramo varejista.

Cinco modelos são avaliados pelo autores, sendo eles: (*i*) Média Móvel Simples (*Simple Moving Average*, em inglês); (*ii*) Média Móvel Exponencial (*Exponential Moving Average* ou EMA, em inglês); (*iii*) K-Vizinhos mais Próximos (*K-Nearest Neighbors*, em inglês); (*iv*) Redes Neurais e por fim, (*v*) Ensemble de todos os modelos anteriores. Para coleta dos resultados, um conjunto genérico de dados é descrito, tendo entre os principais fatores a temperatura, dia, se é ou não feriado e vendas de produtos alternativos. Os autores argumentam que com estes fatores conseguem descrever uma forma mais geral do problema, e assim avaliar a eficácia de métodos alternativos.

Para comparação dos modelos, quatro métricas conhecidas são calculadas: (*i*) Mean Absolute Erro (MAE); (*ii*) Mean Absolute Percentage Erro (MAPE); (*iii*) Mean Squared Error (MSE); e (*iv*) Root Mean Square Error (RMSE). O trabalho conclui então que o método que retorna o melhor valor para as medidas é o ensemble baseado em todos os modelos. Apesar de significativo o resultado, não fica claro em quais dados realmente os modelos foram testados, e pouco se discute da interpretabilidade do modelo obtido e o quão impactante isto seria para o negócio em que este seria aplicado.

Um estudo de revisão é realizado em [14] sobre aplicações de AM e Séries Temporais para Previsão de Demanda e Vendas com uma perspectiva de tendências futuras e aplicações na indústria. O trabalho começa por estabelecer a dificuldade atual em sistemas preditivos devido ao grande volume de dados coletados, influência de variáveis exógenas, e características e relações não lineares entre dados de entrada e saída. Devido à estes fatores, cresce-se então o interesse em modelos baseados em Inteligência Artificial, porém levanta-se a questão se de fato, a utilização de modelos mais complexos traz benefícios.

Os trabalhos revisados incluíram dez artigos selecionados com tags específicas como, por exemplo, ”Supply Chain Management”, ”Demand Forecasting”, ”Sales Forecasting”, ”Machine Learning”, entre outros. Destas análises, algumas conclusões se destacam, como:

- Aplicações de dados de venda de produtos diferentes para a previsão de produtos novos.
- Grande utilização de técnicas de pré-processamento para redução de dimensionalidade do problema, gerando economia de recursos computacionais e diminuição no tempo de treino e previsão do modelo.
- Crescente utilização de variáveis exógenas em modelos.
- Utilização de técnicas fuzzy ao tratar de informações imprecisas.

Por fim, os autores concluem o trabalho dizendo que a adoção de modelos de inteligência artificial tem mostrado-se razoável dentro das novas características de negócio, ao realizar previsões mais adequadas, ao mesmo tempo que criam novas possibilidades ao diminuir efeitos indesejados ao gerenciar uma cadeia de suprimentos. Porém, apesar do estudo ter detalhado uma comparação em um número variado de casos, pouco é citado sobre a perda de interpretação que se obtém ao escolher estes modelos.

Já dentro da área de previsão em série temporais, os autores introduzem o problema tratando da dificuldade em gerar previsões denominadas coerentes, isto é, valores que ao serem agregados ao longo da hierarquia não produzem previsões enviesadas. Os autores apresentam cinco técnicas para reconciliação da estrutura hierárquica, sendo elas: Bottom-Up, três variações de Top-Down e Reconciliação Ótima. Os três métodos estão definidos em 3.3. Em [**9**] os autores expandem os estudos dentro da área de séries hierárquicas incluindo modelos de AM como *Gradient Boosting*, Redes Neurais e SVR para realização de previsão de vendas. Eles argumentam que em dados com dinâmicas afetadas por variáveis exógenas como promoções podem ser melhor modelado através do uso de modelos de AM. Os resultados mostram que modelos de *Boosting* apresentam melhores resultados ao realizar previsão dos nós mais afetados por promoções, isto é, os nós inferiores da hierarquia, como a quantidade de vendas em uma determinada loja / região.

CAPÍTULO 3

Fundamentação Teórica

3.1 Séries Temporais

Uma série temporal é uma sequência de observações de uma variável dispostas no tempo [15]. Formalmente, temos que x_t é um conjunto de informações onde cada valor é obtido no tempo t. Se o intervalo t é contínuo, dizemos que trata-se de uma série contínua, caso contrário denominamos como uma série temporal discreta. É um campo abrangente onde muitos conjuntos de dados dentro de diversas áreas surgem como série temporais, por exemplo:

1. valores diários da quantidade de chuva por dia em determinada região;
2. índices mensais de inflação de um país;
3. quantidade demandada de um produto;

A figura 3.1 ilustra três séries temporais distintas. Enquanto as duas primeiras são originalmente discretas, e a terceira contínua.

Ao analisar série temporais, temos os seguintes objetivos: *(i)* criar previsões de valores futuros; *(ii)* testar hipóteses das causas do caminho que a série temporal percorreu para que mudanças sejam sugeridas; *(iii)* criar simulações para possíveis caminhos que a série pode percorrer dependendo dos parâmetros que foram utilizamos para inferi-la [16]. Para atingir este objetivo, pode-se descrever a série temporal em termos estatísticos como média, variância e função de autocorrelação, essenciais para criação de modelos paramétricos da mesma.

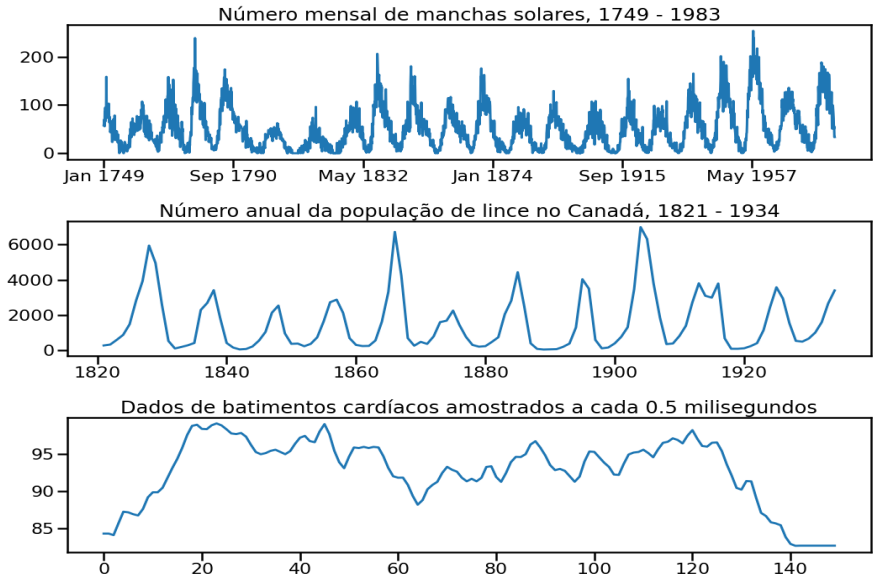

Figura 3.1: Três séries temporais com períodos distintos.

3.1.1 Processos Estocásticos Estacionários

Uma série temporal é determinada por sua trajetória, entre muitas possíveis, sendo esta o resultado do processo físico que cria uma variável aleatória ao longo do tempo. Este caminho é denominado de realização dentro um conjunto de possibilidades chamado de processo estocástico [15]. Isto é, quando analisamos uma série temporal, tratamos ela como uma realização deste processo aleatório ao longo do tempo.

Uma classe especial de processos estocásticos são denominados de processos estacionários, isto é, eles são baseados na hipótese que a série temporal apresenta um equilíbrio estatístico. Eles podem ser classificados como estritamente e fracamente estacionários. Um processo é denominado estritamente estacionário se suas propriedades não são afetadas pelo momento que a estatística é calculada, em termos matemáticos, seja um conjunto de m observações $x_{t_1}, x_{t_2}, x_{t_3}, \ldots, x_{t_m}$, e seja $Z(t_m)$ sua probabilidade conjunta, a série será estacionária se $Z(t_m)$ for igual a $Z(t_{m+h})$, onde h é um inteiro fixo. Em outras palavras, dizemos que as propriedades estatísticas da série temporal não mudam se deslocarmos a série temporal em h instantes para frente ou para trás. Séries estritamente estacionárias são difíceis de serem encontradas em problemas reais, portanto utilizaremos como base séries fracamente estacionárias.

Para que uma série temporal seja fracamente estacionária, temos que ter as seguintes condições:

- Média μ_t é igual para todo tempo t.
- Variância σ_t^2 é igual para todo tempo t.
- Covariância entre x_t e x_{t-h} é igual para todo tempo t e depende apenas do atraso entre

estes dois pontos.

Onde a média e variância são dados por:

$$\mu_t = \frac{1}{T}\sum_{t=1}^{T} x_t$$

$$\sigma_t^2 = \frac{1}{T}\sum_{t=1}^{T} (x_t - \mu_t)$$

Onde T é o período utilizado para cálculo das estatísticas. A covariância será dada então por:

$$Cov[x_t, x_{t+h}] = E[(x_t - \mu_t)(x_{t+h} - \mu_t)] \tag{3.1}$$

Ela também é denominada de autocovariância já que estamos calculando a covariância da série com ela mesmo deslocada em k. Similarmente, podemos definir o coeficiente de autocorrelação:

$$\delta_h = \frac{E[(x_t - \mu)(x_{t+h} - \mu)]}{\sigma_t^2}$$

Estes valores são importantes pois ao assumir que uma série é fracamente estacionária, ela mantém constantes certos momentos estatísticos ao longo de toda sua trajetória, implicando que ao determinar sua média, covariância e autocorrelação conseguimos criar modelos paramétricos que consigam prever a série com grande precisão. Vamos tratar na próxima seção de modelos que utilizam estes parâmetros para inferência e previsão de séries temporais.

3.2 Modelos aplicados à Séries Temporais

Nesta seção iremos abordar os modelos e técnicas empregados neste trabalho para previsão de demanda em séries temporais. O grande desafio é trabalhar com modelos capazes de lidar com estruturas intrínsicas de séries temporais ou criar transformações adequadas nos dados para aqueles tradicionalmente aplicados em dados de corte transversal como Regressão Multivariada, por exemplo. Pontos como correlação entre tempos adjacentes, média e desvio padrão constantes, sazonalidade, tendência e estacionariedade da série devem ser verificados a priori para que não exista conclusões equivocadas baseadas em dados com correlação espúria, por exemplo.

3.2.1 ARIMA

Modelos de regressão não capturam informações essenciais em uma série temporal como: a correlação na variável independente em tempos adjacentes, bem como a não estacionariedade da mesma, levando a problemas na modelagem e introdução de viéses [4]. Portanto, é interessante

que em uma série temporal, a variável dependente seja influenciada por seus próprios valores passados e valores da variável independente. O modelo ARIMA (*Autoregressive Integrated Moving Average*) é uma representação de dois modelos, o *Autoregressive* (AR) e o *Moving Average* (MA), mais o termo de diferenciação da série que lida com a não estacionariedade da mesma. Nesta seção, iremos detalhar ambos os métodos visando modelar séries temporais através de seus valores passados para realização de previsão.

Modelos AR

Em um modelo de auto regressão, utilizamos seu próprio valor passado como um preditor para seu valor futuro, isto é, uma combinação linear nos moldes de uma regressão clássica. A equação 3.2 demonstra o modelo $AR(p)$, pois sua ordem é determinada pelos últimos p valores.

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + w_t \tag{3.2}$$

Onde x_t é estacionário, w_t é um ruído branco com média 0 e variância σ_w^2, e $\phi_1, \phi_2, \phi_3, ..., \phi_p$ são parâmetros com $\phi_p \neq 0$.

Por se tratar de uma auto regressão, isto é, os preditores são valores passados da série temporal, a notação é facilitada ao se utilizar o operador de *backshift*, a equação 3.2 pode ser reescrita através da equação 3.3.

$$\phi(B)x_t = w_t \tag{3.3}$$

O operador de *backshift* é uma ferramenta utilizada ao se trabalhar com valores defasados em séries temporais, por exemplo, podemos representar o valor x_{t-1} utilizando x_t pelo operador B descrito pela equação 3.4.

$$Bx_t = x_{t-1} \tag{3.4}$$

Ou seja, ao aplicar o operador em x_t estamos atrasando o valor em um período de tempo. Com duas aplicações, atrasamos duas vezes, conforme mostrado em 3.5.

$$B(Bx_t) = B^2 x_{t-1} = x_{t-2} \tag{3.5}$$

De uma maneira geral, podemos definir $\phi(B)$ então como representado em 3.6.

$$\phi(B) = 1 - \phi B - \phi_2 B^2 - ... - \phi_p B^p \tag{3.6}$$

Modelos MA

Em um modelo de média móvel, assume-se que a variável independente é obtida através da combinação linear de uma série de ruído branco. A equação 3.7 demonstra o modelo $MA(q)$, onde a sua ordem é determinada pelos últimos q valores.

$$x_t = \theta_1 w_{t-1} + \theta_2 w_{t-2} + ... + \theta_q w_{t-q} \tag{3.7}$$

Onde w_t é um ruído branco com média 0 e variância σ_w^2, e $\theta_1, \theta_2, \theta_3, ..., \theta_q (\theta_q \neq 0)$ são parâmetros. A equação 3.7 pode ser reescrita com operador de *backshift* da seguinte forma:

$$x_t = \theta(B) w_t \tag{3.8}$$

Entretanto, como não conseguimos observar o ruído de fato para estimar MA, temos que o mesmo modelo pode ser não único, isto é, apresenta mais de uma representação. Além do mais, uma propriedade importante é que conseguimos converter modelos $MA(\infty)$ em modelos AR(p) e vice-versa, contanto que algumas restrições sejam impostas aos coeficientes θ do modelo MA. Este modelo é denominado invertível. Isto é importante, pois conseguimos converter modelos MA em modelos AR, no qual conseguimos de fato observar. Porém, como estamos tratando de modelos infinitos, as restrições citadas acima são tal que $|\theta| < 0$, criando modelos onde observações mais recentes ganham peso maior ao determinar o valor do futuro da variável independente, no caso x_t.

Diferenciação

Uma série estacionária é aquela nos quais suas propriedades não dependem do tempo em que a série é observada [17]. Portanto, séries com sazonalidades ou tendência não são estacionárias. Existem duas definições formais de estacionariedade: estritamente e fracamente. Iremos abordar o segundo caso, pois para maioria das aplicações reais a primeira não é realizável [4].

Seja x_t uma série temporal, ela é dita fracamente estacionária se trata-se de um processo com variância finita tal que

1. O seu valor médio, μ_t é constante, e não depende do tempo t.
2. A função de autocovariância 3.1, $Cov[x_t, x_{t+h}]$ depende somente de h e t através de seu atraso $|t+h|$.

Uma abordagem ao tratar séries não estacionárias é o método da diferenciação, que consiste em basicamente computar as diferenças entre obervações consecutivas, denominado de modelo de integração. Por exemplo, em muitas situações uma série temporal pode ser tratada através de um componente estacionário e outro não estacionário. Seja o modelo dado pela equação 3.9,

onde $v_t = \beta_0 + \beta_1 t$ e y_t é estacionário. A diferenciação com relação a x_t irá gerar um processo estacionário, conforme ilustrado na equação 3.10.

$$x_t = v_t + y_t \tag{3.9}$$

$$\nabla x_t = x_t - x_{t-1} = \beta_1 + y_t - y_{t-1} = \beta_1 + \nabla y_t \tag{3.10}$$

No geral, definimos o operador de diferenciação ∇ de ordem d através da equação 3.11.

$$\nabla^d x_t = (1-B)^d x_t \tag{3.11}$$

Portanto, temos que a combinação do modelo $AR(p)$, $MA(q)$ e o modelo de Integração de ordem d, o método ARIMA pode ser descrito por estes três coeficientes, com notação $ARIMA(p,q,d)$. Em geral, se $d = 0$, dizemos que x_t é estacionário, ou simplesmente temos um modelo ARMA.

O modelo $ARIMA(p,q,d)$ é descrito pela equação 3.12.

$$\phi(B)(1-B)^d x_t = \theta(B) w_t \tag{3.12}$$

Para estimar os parâmetros, utilizamos os valores obtidos graficamente através das Funções de Autocorrelação (Aucorrelation Function ou ACF, em inglês) e Autocorrelação Parcial (Partial Aucocorrelation Function ou PACF, em inglês), o critério é minimizar a métrica de interesse. Na seção 3.4.3 mostraremos qual métrica utilizamos para cada modelo.

Modelos Sazonais

Por fim, vamos expandir o modelo ARIMA ainda mais, incluindo agora componentes sazonais. Por exemplo, dados econômicos apresentam fortes componentes sazonais, que se repetem a cada ano. Isto é, além de assumir que conseguimos modelar a série temporal x_t através de seus valores passados e seus erros de predição, também podemos assumir que está fortemente correlacionado com um padrão sazonal. Assim, definimos um modelo $ARMA(P,Q)$ com seus próprios coeficientes através da equação 3.13.

$$\Phi_P(B^S) x_t = \Theta_Q(B^S) w_t \tag{3.13}$$

Onde os operadores de auto regressão e média móvel sazonais de periodo s, e de ordem P e Q, respectivamente são definidos por 3.14 e 3.15 e B^S é o operador de *backshift* sazonal, definido por 3.16.

$$\Phi_P(B^S) = 1 - \Phi_1(B^s) - \Phi_2(B^{2s}) - \ldots - \Phi_P(B^{Ps}) \tag{3.14}$$

$$\Theta_Q(B^S) = 1 - \Theta_1(B^s) - \Theta_2(B^{2s}) - ... - \Theta_Q(B^{Qs}) \tag{3.15}$$

$$B^S x_t = x_{t-S} \tag{3.16}$$

Como definido anteriormente, séries sazonais não apresentam estacionariedade, portanto, torna-se necessário definir um operador de diferenciação sazonal e um modelo integrado de ordem D para que tenhamos um modelo ARIMA sazonal, ou SARIMA. De forma geral, o modelo é denotado por $ARIMA(p,q,d) \times (P,D,Q)_s$, e é definido através da equação 3.17.

$$\Phi_P(B^S)\phi(B)\nabla_S^D\nabla^d x_t = \delta + \Theta_Q(B^S)\theta(B)w_t \tag{3.17}$$

Determinar os coeficientes de um modelo ARIMA sazonal não é uma tarefa fácil, ainda mais quando estamos trabalhando com múltiplas séries temporais. Na seção 4.1.1 estimamos os parâmetros para a série quadrimestral de visitantes noturnos da Austrália a fim de demonstrar o passo a passo na geração de um modelo completo e sua utilização na previsão de novos valores.

3.2.2 Suavização Exponencial

Começaremos nosso estudo em cima dos modelos denominados suavização exponencial. Estes são amplamente empregados através de softwares de análise estatísticas dos mais variados, por sua facilidade de interpretação e adaptação aos dados, não necessitando a priori um tratamento muito rigoroso, apesar de sempre ser recomendado que o analista faça uma análise descritiva de seus dados.

Suavização Exponencial Simples

A técnica de suavização exponencial simples (SES) [5] utiliza uma média ponderada de valores passados para previsão de valores futuros com pesos associados a um decaimento exponencial, isto é, quanto mais para o passado a variável caminha, menos importância ela tem em uma nova previsão. São mais indicados para séries com nenhuma tendência ou sazonalidades claras. Seja y_t uma série temporal onde queremos prever o valor futuro y_{t+h} e onde $h \geq 1$, temos que pelo modelo SES, a previsão de um passo futuro é dada pela equação:

$$\hat{y}_{t+1|t} = \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + ... \tag{3.18}$$

Esta equação é denominado de nível ℓ_t no instante t, onde $0 < \alpha < 1$ é denominado de fator de suavização, y_t é o último valor observado e $y_{t+1|t}$ é a previsão no tempo futuro um passo a frente. O único hiperparâmetro a ser determinado é o fator de suavização, onde geralmente emprega-se

uma busca em grade onde vários valores de α são testados e métricas de otimização são escolhidas para o melhor modelo. Por exemplo, podemos escolher o fator a fim de minimizar o soma do erro quadrático médio de previsões nos dados amostrados. Para facilitar a implementação do mesmo, usualmente considera-se sua versão recursiva, onde o modelo é definido por:

$$\hat{y}_{t+1|t} = \ell_t \tag{3.19}$$

$$\ell_t = \alpha y_t + \alpha(1-\alpha)\ell_{t-1} \tag{3.20}$$

O modelo simples então serve como uma boa base ao se comparar e tratar de métodos mais robustos, pontos favoráveis ao SES incluem fácil interpretação dos parâmetros, pouco ou nenhuma preparação prévia dos dados, fácil implementação e razoável precisão.

A figura 3.2 ilustra uma série temporal do número de nascimento de crianças do sexo feminino na califórnia no ano de 1959.

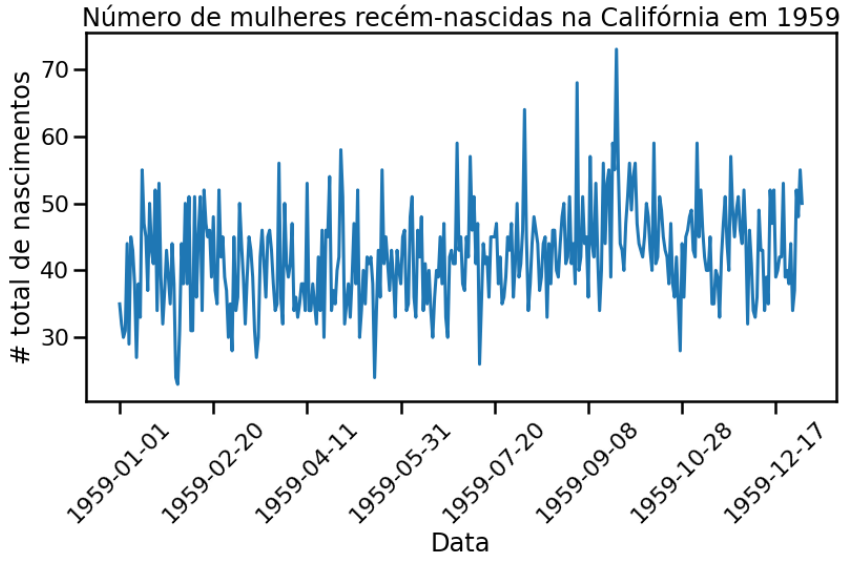

Figura 3.2: Série temporal sem tendência ou sazonalidade.

Vemos que trata-se de uma série sem tendências ou sazonalidades claras, a figura 3.3 ilustra vários modelos SES variando alfa. Podemos ver claramente o efeito ao variar o valor do parâmetro, sendo que valores menores utilizam mais dados do passado, portanto apresentam uma previsão mais suave, enquanto números maiores dependem de valores mais próximos, gerando maior volatilidade nas previsões.

Suavização Exponencial de Holt (HES)

O método SES não é recomendado para séries com tendência, para contornar este problema, Holt propôs a utilização de uma nova constante que suavização para tendência denominada $\beta*$

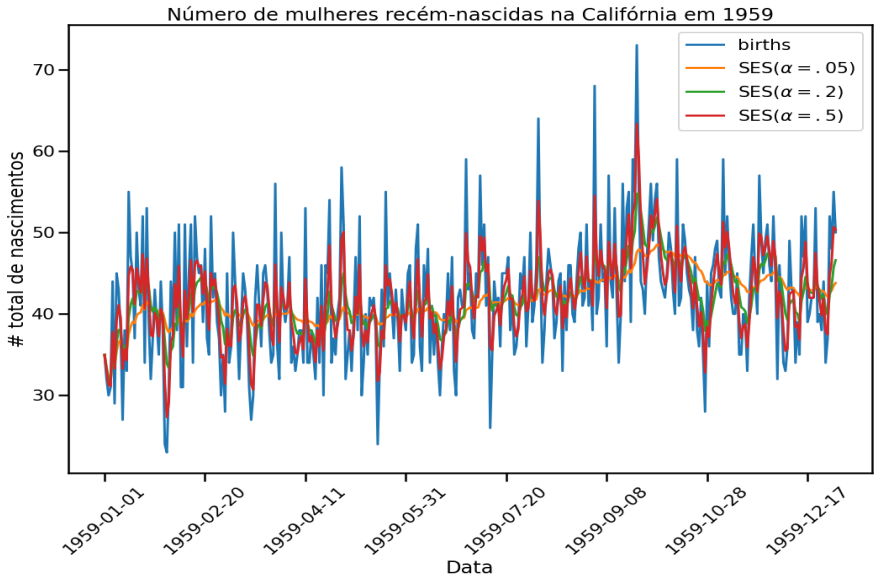

Figura 3.3: Série com modelos aplicados variando alfa.

[18]. As novas equações se tornam então:

$$\hat{y}_{t+h|t} = \ell_t + hb_t \tag{3.21}$$

$$\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1}) \tag{3.22}$$

$$b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \tag{3.23}$$

Onde a nova equação 3.23 é chamada de equação de tendência e b_t é a estimativa do fator de tendência da série no instante t e $\hat{y}_{t+h|t}$ refere a uma previsão h passos a frente, isto é, estamos prevendo h pontos futuros a partir do ponto t. Assim, para cada novo valor adicionado a série, temos novos valores de tendência, bem como a própria previsão muda. A figura 3.4 ilustra o método aplicado na série temporal do volume negociado diariamente do índice do *Google* na bolsa NASDAQ. Trata-se de uma série que só apresenta tendência, e vemos que o modelo é capaz de capturar esta propriedade e gerar previsões que estão de acordo com o que a série temporal representa.

Suavização Exponencial Sazonal de Holt-Winters

Quando a série apresenta características típicas de sazonalidade e/ou tendência, utilizamos o modelo de suavização exponencial sazonal de Holt-Winters, ou denominados métodos de Holt-Winters (HW). Novamente temos o emprego de uma equação extra em nosso modelo levando em consideração o fator sazonal que o dado apresenta, sendo que estes podem ter sazonalidade aditiva ou multiplicativa. O novo fator de suavização exponencial é dado por γ e as equações do modelo de suavização exponencial são reajustas da seguinte forma:

Figura 3.4: Previsão e valores observados da série temporal com tendência.

$$\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)} \tag{3.24}$$

$$\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \tag{3.25}$$

$$b_t = \beta^*(\ell_t - \ell_t) + (1-\beta^*)b_{t-1} \tag{3.26}$$

$$s_t = \gamma(y_t - \ell_t - b_{t-1}) + (1-\gamma)s_{t-m} \tag{3.27}$$

A equação 3.27 retrata o modelo HW aditivo. Temos que s_t é o valor da equação sazonal no instante t, m o período do dado, i.e., $m = 4$ para dado quadrimestral, $m = 12$ para dados com frequência anual, etc. A única mudança para o modelo multiplicativo ocorre na equação 3.27, neste caso ela será corrigida a cada iteração por:

$$s_t = \gamma\frac{y_t}{(\ell_t - b_{t-1})} + (1-\gamma)s_{t-m} \tag{3.28}$$

A grande diferença do modelo multiplicativo para o modelo aditivo é que ao invés de considerarmos o valor absoluto para gerar o novo valor de suavização sazonal, consideramos sua porcentagem. Ele é comumente indicado quando a variância ou amplitude da sazonalidade da série aumenta com sua tendência. A figura 3.5 ilustra um modelo de Holt-Winters com sazonalidade aditiva aplicada à uma série temporal para previsão de demanda de energia elétrica. Na figura temos os valores observados, as previsões *in-sample*, isto é, os valores preditos para os dados de treino, e a previsão para o conjunto de teste, no qual o modelo não foi treinado. Vemos como neste caso, além da tendência, o modelo consegue capturar o componente sazonal na hora de realizar suas previsões fora da amostra de treino.

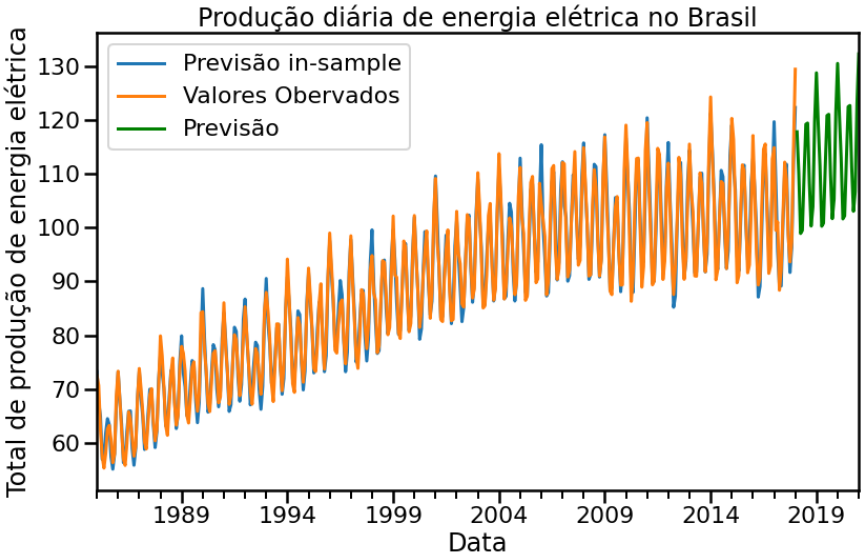

Figura 3.5: Previsão e valores observados da série temporal com tendência e sazonalidade

A escolha dos parâmetros para todos os modelos citados continua sendo da mesma forma: escolhemos uma função de otimização para ser minimizada, por exemplo, soma dos erros quadráticos, e escolhemos o conjunto de parâmetros que melhor obtém valor nesta métrica por uma busca em grade. Na seção de métricas e critérios de avaliação de modelos trataremos mais sobre o assunto.

3.2.3 *Random Forests*

Um dos modelos mais utilizados na área de AM são as árvores de decisão, por sua alta acurácia nas mais diversas aplicações, bem como simplicidade de conceito. Neste trabalho, vamos explorar o algoritmo *Random Forests*, que utiliza uma combinação de árvores preditoras amostrada de forma independente e com a mesma distribuição, agregando um poder de generalização que aumenta a acurácia da mesma comparada com uma única árvore [19].

Algoritmos baseados em árvores buscam particionar o espaço de variáveis em regiões ou blocos de decisão onde o valor obtido em cada região determinará o valor de saída para aquela entrada, seja ele um classificador ou um regressor. Nesta seção, iremos abordar árvores de decisão aplicadas à problemas de regressão, isto é, cujo objetivo é a previsão de uma variável contínua.

Seja y a variável resposta do nosso problema, cuja as variáveis de entrada são x_1 e x_2, começamos por criar regiões R_m de tal forma que minimizamos algum critério de erro, por exemplo, a soma dos erros residuais, dada por

$$\sum_{j=1}^{J} \sum_{i \in R} (y_i - \hat{y}_{R_j})^2 \tag{3.29}$$

onde $\hat{y}_{R_j}$ é o valor médio obtido nos dados de treino para a j-ésima região. Entretanto, é computacionalmente infactível considerar todas as regiões para se determinar a divisão ótima, por isso temos que considerar uma abordagem top-down gulosa, também denominado árvore binária recursiva. Top-down pois começamos no topo da árvore e gulosa pois a cada particionamento, escolhemos a variável que melhor minimiza o critério de erro, não sendo necessariamente aquela que gera a melhor árvore no geral [20]. Vamos ilustrar a construção de uma árvore genérica.

Por exemplo, ao começar por separar a reta $x_1 = a_1$ em duas regiões R_1 e R_2, construímos o primeiro critério de decisão da nossa árvore, onde cada nova instância terá seu valor de saída determinado a partir do nó raiz sendo particionado em a_1. A figura 3.6 ilustra esta divisão inicial nos dados.

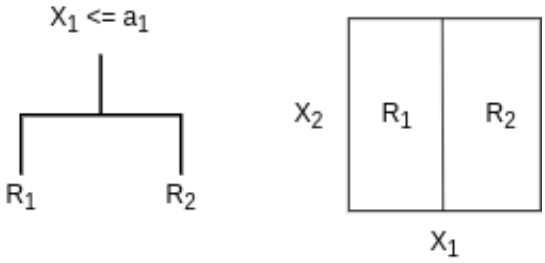

Figura 3.6: À esquerda temos a raiz da árvore criada ao particionar os dados; à direita temos as regiões criadas pela regra inicial.

Continuando o processo de divisão das regiões, podemos criar um segundo particionamento, porém agora utilizamos a variável x_2 seguindo a hierarquia da árvore. Assim, seja a_2 o novo ponto de particionamento, dividimos a região R_1 em duas novas regiões, R_3 e R_4. A figura 3.7 mostra as novas regiões.

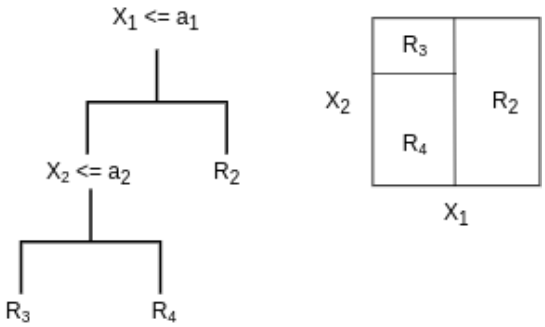

Figura 3.7: Representação de uma segunda partição adicionada através do ponto de corte a_2 na variável x_2.

No fim, o modelo prediz um valor utilizando a média dos valores nos dados de treino para cada região, ou seja, se a média obtida no treino do modelo para R_2 é 20, o valor de x_i que cair na região R_2, será 20 também. De forma geral, nós primeiros selecionamos um preditor x_j e um ponto de corte a onde a divisão causará a maior redução possível na função objetivo, como a soma residual dos quadrados, ou RSS (*Residual Sum of Squares*, do inglês), dada por 3.30.

$$RSS = \sum_{i=1}^{n} (y_i - f(x_i))^2 \tag{3.30}$$

Em seguida, iteramos nos próximos preditores para continuarmos minimizando o erro nas novas regiões, até que um processo de parada termine a iteração. Conforme ilustrado no exemplo anterior, nossa predição será dada pelo valor médio da região em que o ponto de teste cair.

Bagging

Árvores de decisão são algoritmos simples e poderosos na construção de modelos de aprendizado de máquina, porém são extremamente suscetíveis a alta variância devido a hiperespecialização nos dados de treino, o que denominados de *overfitting*. Para contornar este problema, utilizamos uma técnica da área de amostragem estatística denominada *bagging* ou *boostrap aggregation*.

Bootstrap é uma técnica de amostragem utilizada para estimar parâmetros de uma população nos quais a realização de múltiplos experimentos ou simulações não é realizável. Vamos supor que gostaríamos de estimar um parâmetro ζ de uma população qualquer. Poderíamos simplesmente calcular o valor de ζ diretamente, porém não teríamos informações sobre sua distribuição ou variância. O método de bootstrap consiste em amostrar N vezes com reposição do conjunto de dados criando diferentes conjuntos denominados M^{*i}, então cada amostra terá um valor estimado de ζ sendo ζ^{*i}, portanto temos N parâmetros estimados a partir dos dados amostrados. Por fim, podemos calcular o desvio padrão pela equação 3.31.

$$SE_N(\hat{\xi}) = \sqrt{\frac{1}{N-1} \sum_{r=1}^{N} (\hat{\xi}^{*r} - \frac{1}{N} \sum_{r'=1}^{N} \hat{\xi}^{*r'})^2} \tag{3.31}$$

Sabemos do teorema do limite central [21] que dado um conjunto n de observações distintas, a variância da média gerada pelos n conjunto de dados é dado por σ^2/n, isto é, conseguimos reduzir a variância de um estimador a partir da média individual destes. Mais precisamente, criamos N conjunto de dados a partir dos dados de treino, e geramos N árvores diferentes a partir destas amostras $\hat{f}^{*1}(x), \hat{f}^{*2}(x), ..., \hat{f}^{*N}(x)$. O estimador final será dado por 3.32, denominado *bagging*.

$$\hat{f}_{bag} = \frac{1}{N} \sum_{n=1}^{N} \hat{f}^{*n}(x) \tag{3.32}$$

Random Forests trata-se de um conjunto de estimadores baseados em árvores de decisão com *bagging*, com uma diferença que ao invés de cada estimador utilizar todos as variáveis disponíveis para realizar as divisões, escolhemos um número de variáveis k tal que $k < p$ pois assim criamos modelos não correlacionados, aumentando a acurácia do algoritmo.

Por fim, estes parâmetros como k, número de variáveis por estimador, N número total de

estimadores, entre outros como profundidade da árvores, número de amostras por folha da árvores são denominados hiperparâmetros. Eles são definidos a partir de métodos de validação cruzada. Como sempre, séries temporais tem seus próprios métodos de validação cruzada, discutiremos a fundamentação teórica dos mesmos na seção Métodos de validação para séries temporais. Os hiperparâmetros definidos para cada conjunto de dados será detalhado na seção de resultados.

3.2.4 Gradient Boosting

Outro método que é capaz de aumentar a capacidade de previsão de modelos de árvores é denominado *boosting*. Neste trabalho utilizaremos o modelo de *Gradient Boosting*. Assim como bagging, esta técnica pode ser empregada para qualquer método estatístico que envolva construção de algoritmos a partir de dados, porém ela é comumente empregada em modelos de árvore. Relembramos que ao treinar um modelo de *bagging*, as árvores são construídas paralelamente, a partir de um conjunto de dados amostrados com *bootstrap*. Boosting, ao contrário, cria árvores de forma sequencial, isto é, a cada iteração uma nova árvore é criada a partir da anterior, onde a variável alvo nada mais é do que o valor residual da árvore anterior. O algoritmo 1 detalha o passo a passo para a construção de um modelo de árvore baseado em boosting para tarefas de regressão.

A ideia por trás deste algoritmo envolve o conceito de aprendizado lento, isto é, construímos árvores menores a cada iteração que aprende com o erro da anterior. Este erro é diminuído a cada iteração através de uma função de otimização diferenciável, isto é, ele calcula um gradiente para otimizar o erro, dando nome ao modelo. Aumentamos a capacidade de previsão a cada pequeno passo controlando o parâmetro denominado de taxa de aprendizado, μ, onde valores altos podem convergir mais rápido, porém com altas chances de overfitting do modelo, enquanto valores menores necessitam de maiores iterações para convergência. Assim, de forma resumida, os três principais parâmetros de um modelo de boosting são:

1. O número de árvores N. Ao contrário de bagging, ao aumentar o número de árvores, existe chance de overfitting.

2. Taxa de aprendizado υ.

3. O número de folhas, ou nós terminais da árvore.

Todos estes parâmetros são selecionados então utilizando validação cruzada.

Algorithm 1 *Gradient Boosting* em árvore

1. **Entrada:** Seja o conjunto de dados (x_i, y_i), e uma função diferenciável $L(y_i, F(x))$
2. Inicialize o modelo com um valor constante: $F_0(x) = \arg\min_{\varphi} \sum_{i=1}^{n} L(y_i, \varphi)$
3. Para $n = 1$ até N:
   (a) Calcule $r_{im} = -[\frac{\partial L(y_i, F(x))}{\partial F(x)}]_{F(x)=F_{m-1}(x)}$
   (b) Cria uma árvore de regressão com os valores residuais r_{im} e regiões terminais (folha da árvore) R_{jm} para $j = 1..J_m$
   (c) Para $j = 1..J_m$ calcule $\varphi_{jm} = \arg\min_{\varphi} \sum_{x \in R_{ij}} L(y_i, F_{m-1}(x_i) + \varphi)$
   (d) Atualize $F_m(x) = F_{m-1}(x) + \upsilon \sum_{j=1}^{J_m} \varphi_m I(x \in R_{jm})$
4. **Saída:** $F_M(x)$

3.3 Estruturas Temporais Hierárquicas

Conforme descrito na introdução deste trabalho, algumas séries temporais apresentam propriedades hierárquicas devido a divisões geográficas, por segmento de produto, ou qualquer divisão de negócios que gere uma hierarquia no qual a nó pai depende da agregação dos nós filhos [1]. Séries hierárquicas requerem previsões que se somam de forma **coerente**, isto é, não podemos gerar previsões individuais para cada nó e esperar que elas façam sentido ao longo das ramificações. A figura 3.8 mostra duas ramificações de uma estrutura hierárquica comum.

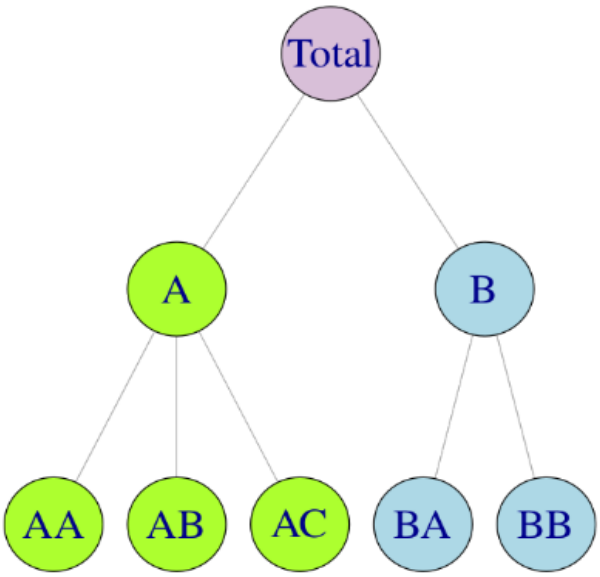

Figura 3.8: Hierarquia de dois níveis. Fonte: [1]

Vemos pela imagem que o nó A tem como previsão as somas dos nós filhos AA, AB e AC.

O mesmo ocorre para o nó raiz, isto é, o total da série. Maneiras diferentes de agregação são denominadas de **reconciliação**, nesta seção vamos descrever três métodos mais utilizados e que serão empregados nos conjuntos de dados a serem experimentados.

3.3.1 Bottom-Up

Este é um método simples que consiste em gerar previsões para todas as séries nos nós folhas, isto é, no último nível da hierarquia e então somar os nós de acordo com as ramificações para gerar previsões nos nós superiores.

De acordo com [1], por exemplo, para a hierarquia da figura 3.8, geramos previsões h passos à frente para todos os nós folhas: $\hat{y}_{AA,h}$, $\hat{y}_{AB,h}$, $\hat{y}_{AC,h}$, $\hat{y}_{BA,h}$ e $\hat{y}_{BB,h}$.

Geramos então as previsões para os nós pais somando os nós filhos:

$$\tilde{y}_h = \hat{y}_{AA,h} + \hat{y}_{AB,h} + \hat{y}_{AC,h} + \hat{y}_{BA,h} + \hat{y}_{BB,h} \tag{3.33}$$

$$\tilde{y}_{A,h} = \hat{y}_{AA,h} + \hat{y}_{AB,h} + \hat{y}_{AC,h} \tag{3.34}$$

$$\tilde{y}_{B,h} = \hat{y}_{BA,h} + \hat{y}_{BB,h} \tag{3.35}$$

Que podem ser reescritas através utilizando notação de matrizes:

$$\begin{bmatrix} \tilde{y}_h \\ \tilde{y}_{A,h} \\ \tilde{y}_{B,h} \\ \tilde{y}_{AA,h} \\ \tilde{y}_{AB,h} \\ \tilde{y}_{AC,h} \\ \tilde{y}_{BA,h} \\ \tilde{y}_{BB,h} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{y}_{AA,h} \\ \hat{y}_{AB,h} \\ \hat{y}_{AC,h} \\ \hat{y}_{BA,h} \\ \hat{y}_{BB,h} \end{bmatrix}. \tag{3.36}$$

A vantagem deste método é sua simplicidade, pois não perdemos informações através da propagação da reconciliação. A grande desvantagem gira em torno de que nós folhas da hierarquia são geralmente mais difíceis de prever com alta confiança.

3.3.2 Top-Down

Ao contrário do método *bottom-up*, no método *top-down* geramos primeiro a previsão para os nós pais e então desagregamos os valores para a séries dos nós filhos de forma coerente. Para ocorrer a reconciliação, denominamos variáveis de proporção p_m que irão ditar como as previsões do nó pai se distribuem através das séries subsequentes. Por exemplo, continuando no

exemplo anterior, temos que

$$\tilde{y}_{AA,t} = p_1\hat{y}_t, \quad \tilde{y}_{AB,t} = p_2\hat{y}_t, \quad \tilde{y}_{AC,t} = p_3\hat{y}_t, \quad \tilde{y}_{BA,t} = p_4\hat{y}_t \quad \text{and} \quad \tilde{y}_{BB,t} = p_5\hat{y}_t.$$

Utilizando novamente notação matricial, temos

$$\tilde{y}_h = Sp\hat{y}_t.$$

Onde p é um vetor m-dimensional com as proporções a serem utilizadas para reconciliação e S representa a matriz de soma que irá desagregar as previsões de acordo com os níveis hierárquicos da série. A questão que fica é como produzir este vetor de forma ótima. O método descrito em [22] utiliza proporções baseadas nas próprias previsões para cada nó da hierarquia, vamos descrever o algoritmo agora.

Seja uma hierarquia de um nível, primeiro produzimos previsões h passos à frente para toda a série. Utilizamos então estes valores como fator de multiplicação que gerará as proporções para cada nó. Mais genericamente, seja um hierarquia com K níveis, vamos gerar as previsões para cada nó h passos à frente, indo do nó no topo da hierarquia até o mais baixo nível. A equação 3.37 define as proporções de cada nó:

$$p_j = \prod_{\ell=0}^{K-1} \frac{\hat{y}_{j,h}^{(\ell)}}{\hat{S}_{j,h}^{(\ell+1)}} \tag{3.37}$$

onde $j = 1, 2, \ldots, m$, $\hat{y}_{j,h}^{(\ell)}$ é a previsão h passos a frente da série que corresponde ao nó ℓ níveis acima de j, e $\hat{S}_{j,h}^{(\ell)}$ é a soma das previsões h passos a frente correspondente ao nó que está ℓ níveis acima do nó j e está diretamente conectado àquele nó. Então, desagregamos o nó total através destas proporções para gerar previsões coerentes em toda hierarquia.

Notação Matricial

Ambos os métodos apresentados acima podem ser escritos utilizando notação matricial. Vamos supor que temos todas as previsões obtidas de forma independente ao longo da hierarquia, denominados $\hat{y}_h$, sendo h o horizonte de previsão. Então, todas as abordagens podem ser representadas pela seguinte equação:

$$\tilde{y}_h = SG\hat{y}_h, \tag{3.38}$$

onde G é uma matriz que mapeia as previsões base para os nós inferiores, e S é uma matriz de soma que agrega toda a estrutura para produzir previsões coerentes, denominadas $\tilde{y}_h$.

A matriz G é definida de acordo com o método implementado. Por exemplo, no método bottom-up, temos

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.39}$$

Enquanto que no top-down:

$$G = \begin{bmatrix} p_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ p_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.40}$$

No fim, podemos reescrever a equação 3.38 da seguinte forma:

$$\tilde{y}_h = P\hat{y}_h, \tag{3.41}$$

onde $P = SG$ é uma projeção ou matriz de reconciliação. Como parte do trabalho tem como objetivo implementar tais métodos na linguagem de programação *python*, a notação matricial torna a tarefa mais clara e direta, facilitando o processo de codificação das funções que geram previsões reconciliáveis.

3.3.3 Reconciliação Ótima

Por fim, finalizamos mostrando o método de reconciliação ótima [23]. Basicamente, ele busca gerar previsões coerentes a partir da matriz G que minimiza o erro das previsões no grupo de nós da hierarquia.

Vamos supor que tenhamos gerado uma série de previsões coerentes utilizando a equação 3.38. Primeiro, queremos gerar previsões sem vieses, ou seja, se $\hat{y}_h$ é enviesada, então as previsões $\tilde{y}_h$ serão enviesadas dado que $SGS = S$ [24], onde S é a matriz de soma definida em 3.38. Portanto, gera-se uma restrição na matriz G.

Em seguida, queremos encontrar o erro gerado pelas previsões h passos a frente. No trabalho [23], os autores mostram que a matriz de variância-covariância das previsões é dada por:

$$V_h = \text{Var}[y_{T+h} - \tilde{y}_h] = SGW_hG'S' \tag{3.42}$$

Na equação 3.42, $W_h = \text{Var}[(y_{T+h} - \hat{y}_h)]$ é a matriz de variância-covariância dos erros obtidos

a partir das previsões geradas para cada nó.

Portanto, queremos encontrar a matriz G que minimiza as variâncias dos erros obtidos pelas previsões coerentes. Estes erros se encontram na diagonal da matriz V_h, sendo assim a soma dos erros será dada pelo traço da matriz V_h. Em [23] os autores mostram que a matriz G que minimiza este traço é dada pela equação 3.43.

$$G = (S'W_h^{-1}S)^{-1}S'W_h^{-1}. \tag{3.43}$$

Portanto as previsões ótimas serão dadas pela equação 3.44.

$$\tilde{y}_h = S(S'W_h^{-1}S)^{-1}S'W_h^{-1}\hat{y}_h. \tag{3.44}$$

Em resumo, como em qualquer outro método, a reconciliação ótima utiliza toda informação gerada dentro de uma hierarquia, isto é, alguns ramos da hierarquia podem ser impactados por variáveis totalmente diferentes, mas ainda assim o método tentará suavizar as previsões para todo o ramo, produzindo erros ao longo de um nó maiores do que em outros, ou como se estivessem sido produzidos de forma independente.

3.4 Seleção de Modelos

Ao construir modelos estatísticos estamos interessados em aprender uma função f que relaciona um certo sinal de entrada $x_1, x_2, x_3, ..., x_p$ representado pela matriz X_p, com sua saída y, da seguinte forma:

$$y = f(X_p) + \varepsilon \tag{3.45}$$

Na equação 3.45, ε é o erro intrínseco do processo, também denominado de erro irredutível. Portanto, o aprendizado de modelos estatísticos envolve em encontrar técnicas adequadas para estimar f [20]. Isto se torna importante para dois tipos de aplicações: inferência e predição. Este trabalho tratará de criar um modelo para uma tarefa preditiva, mas sempre que pertinente, iremos comentar sobre a inferência que o modelo gera em determinado momento.

Uma passo importante ao criar modelos de previsão é verificar a acurácia fora dos dados de treino, isto é, simular como os dados se comportam para diferentes horizontes de tempo não usados pelo modelo ao estimar parâmetros. Esta definição é essencial pois um *fit* perfeito sempre pode ser estimado dado um número suficiente de parâmetros [1].

Como f trata-se da verdadeira função que relaciona X_p com y, quase nunca temos a sua verdadeira forma para realizar o treino do modelo. Portanto, nós queremos aprender uma função $\hat{f}$ que se aproximará de f tanto quanto possível. Na equação 3.46, $\hat{f}$ é tratado como uma caixa preta, isto é, não sabemos a sua verdadeira forma, sendo necessário então abordagens para

estimar se estamos indo na direção correta.

$$\hat{y} = \hat{f}(X_p) + \varepsilon \tag{3.46}$$

Nosso objetivo será minimizar o valor esperado quadrático médio entre o valor predito e o valor real, dado pela equação 3.47. Vemos que a primeira parte da equação comporta aquilo que denominamos de *erro redutível*, isto é, no qual o aprendizado dos modelos ocorre. A segunda parte mostra o erro irredutível, que comentamos acima. Trata-se de um erro intrínseco do sistema a ser estudado no qual não temos controle, por isto o nosso foco principal se torna a primeira parte da equação 3.47.

$$E[(y - \hat{y})^2] = [f(X_p) - \hat{f}(X_p)]^2 + Var(\varepsilon) \tag{3.47}$$

Todavia, para estimar o verdadeiro valor de f não podemos utilizar somente os dados de treino como uma métrica confiável de desempenho. Para contornar o problema, utilizamos dados de teste, que não foram usados para treino para avaliação de resultados. Como o conjunto de dados geralmente é limitado, e não temos acesso a informações no futuro, utilizamos o conceito de validação cruzada para simularmos situações reais do dia a dia. Na próxima seção iremos detalhar um pouco mais os conceitos e aprofundar suas implicações em séries temporais.

3.4.1 Métodos de Validação para séries temporais

Parte de todo projeto que envolve algum tipo de modelagem inclui estimativas de desempenho de um modelo, seja ele clássico ou não. Validação cruzada é uma técnica empregada principalmente na área da computação para gerar tais estimativas, sendo utilizada por praticantes para seleção de modelos, seleção de hiperparâmetros e avaliação de previsibilidade para amostras fora do treino, isto é, no conjunto de teste [25]. Todavia, métodos clássicos assumem que as variáveis de interesse são aleatórias, independentes e identicamente distribuídas (*iid*, independent and identically distributed, do inglês), enquanto que séries temporais tem erros dependentes com o tempo. Apesar de diversos trabalhos terem empregado variações do método de validação cruzada para serem aplicados a séries temporais, vamos limitar o escopo deste trabalho para métodos de validação que incluam amostras denominadas *out of sample*, isto é, periodo posteriori ao treino, para mais detalhes de outros métodos, consultar [25].

Abordagens *out-of-sample* consistem em dividir o conjunto de dados em um ponto gerando duas séries: treino e teste, este método também é denominado de *holdout*. Ele é indicado para estimar o desempenho final do modelo escolhido em teoria com dados não vistos. Para criar uma alternativa para seleção de modelos com maior número de combinações de dados de treino e teste, criamos a partir dos dados de treino conjuntos de treino e validação, treinamos diversos

modelos com diferentes hiperparâmetros, e avaliamos seu desempenho médio por configuração treino / validação. Estas configurações podem ser de duas maneiras: janela deslizante (*sliding window*, do inglês) ou janela de rolamento (*rolling window*, do inglês). As figuras 3.10 e 3.9, ilustram os dois métodos, respectivamente.

Figura 3.9: Validação janela de rolamento (*rolling window*).

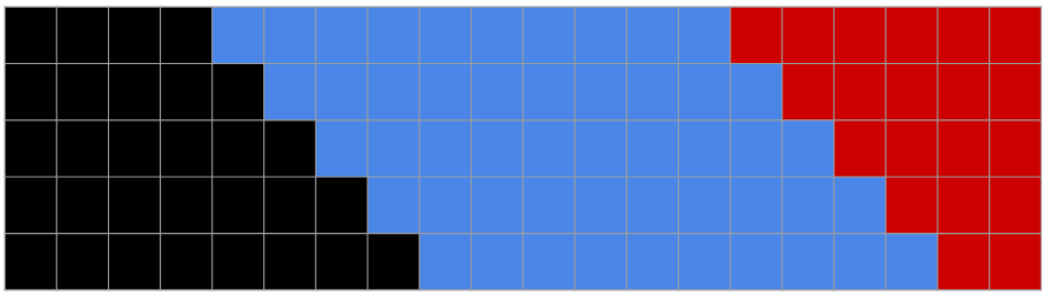

Figura 3.10: Validação janela deslizante (*sliding window*).

Na figuras, as linhas caracterizam as iterações, onde azul significa conjunto de treino, vermelho conjunto de validação e preto dados não selecionados. Conseguimos verificar claramente que no caso da janela de rolamento, aumentamos o conjunto de treino a cada iteração, enquanto que na janela deslizante utilizamos um comprimento fixo denominado tamanho da janela. O método de janela deslizante requer um período fixo de dados a serem utilizados no treino, devido a isso, neste trabalho iremos utilizar o método de rolamento, por ser mais simples e deixar que os modelos aprendam com todos os dados disponíveis. A configuração final então é uma separação entre treino e teste, e então com o conjunto de treino criamos iterações de treino e validação para selecionarmos o melhor modelo.

3.4.2 Medidas de Acurácia

Iremos detalhar nesta seção os métodos mais comuns utilizados ao analisar modelos de regressão em séries temporais e tratarei ao mesmo tempo de uma breve discussão de quais métodos iremos focar ao selecionar a técnica com melhor desempenho nos dados de treino e validação.

Como vimos, é importante diferenciarmos os valores estimados dentro da amostra de treino (*in-sample*) para aqueles obtidos nos dados de teste (*out-of-sample*), sendo os primeiros denominados valores residuais e são obtidos através de predições uma unidade de tempo à frente do dado até o instante t, conforme equação 3.48:

$$r_t = y_t - \hat{y}_t \tag{3.48}$$

Residuais não são valores verdadeiros de previsão pois são obtidos *in-sample*, porém são utilizados para verificar se a série não contém erros autocorrelacionados, o que indicaria informação extra não capturada pelo modelo nos dados; adicionalmente, residuais tem que apresentar valores de média zero, pois caso contrário é uma indicação que o modelo tem um viés [1].

Erro Absoluto Médio (MAE)

Erro Absoluto Médio (*Mean Absolute Error, ou MAE*) é uma métrica dependente da escala em que estamos trabalhando ao fazer previsões para determinada série temporal. Dito isto, elas irão variar conforme a aplicação e somente fazem sentido dentro de um contexto específico. São amplamente empregadas e intuitivas, a equação 3.49 abaixo detalha o cálculo da mesma.

$$MAE = \sum_{t=1}^{n} \frac{|y_t - \hat{y}_t|}{n} \tag{3.49}$$

Erro Percentual Médio Absoluto (MAPE)

Erros percentuais são utilizados ao comparar modelos aplicados a diferentes conjuntos de dados, pois não tem escalas definidas. É dado por 3.50.

$$MAPE = \frac{100}{n} \sum_{t=1}^{n} \frac{|y_t - \hat{y}_t|}{y_t} \tag{3.50}$$

Perceba que uma dificuldade ao lidar com esta métrica é para valores pequenos de y_t o que torna a medida instável em alguns momentos o que torna por dificultar a sua adoção dependendo do problema. A próxima métrica consegue contornar este problema.

Erro Escalado Médio Absoluto (MASE)

Em [26] os autores apresentam MASE (*Mean Absolute Scaled Error*, em inglês), uma medida capaz de lidar com os três tipos possíveis de previsão definidos da seguinte forma:

- (i) Podemos realizar previsões para uma sequência de horizontes de $t = n$ até $t = n + h$, utilizando dados de $t = 1..n$.
- (ii) Podemos variar os horizontes conforme a validação a ser utilizada (*rolling* ou *window*).

- (iii) Previsões a partir de múltiplos conjuntos de dados com diferentes níveis de agregação (região, produto, item) e combiná-los de uma forma coerente.

Assim, os autores argumentam que a métrica MASE é a mais adequada para o último caso, e ao mesmo tempo ela consegue capturar os dois primeiros. Como este trabalho tem como intenção explorar técnicas em séries temporais hierárquicas com diferentes níveis de venda dependendo do produto / departamento / região, iremos utilizá-la ao fazer comparação entre modelos. Outro benefício indireto da métrica, como veremos abaixo, é que ela já é empregada comparando-se com um método *naive*, isto é, ela é escalada pelo modelo mais simples que podemos gerar ao tratar de séries temporais: utilizar o último valor visto na série, ou o último valor da série sazonal. A equação 3.51 detalha os dois casos.

$$MASE = \frac{\frac{1}{J}\sum_j \left|y_{t+j} - \hat{y}_{t+j}\right|}{\frac{1}{T-m}\sum_{t=m+1}^{T} |y_t - \hat{y}_{t-m}|} \tag{3.51}$$

Na equação acima, temos:

- m é o parâmetro ideal para o modelo *naive*, por exemplo, m = 1 para uma série sem sazonalidade, m = 12 para uma série anual ou m = 4 para uma série quadrimestral.
- j é o horizonte em que a previsão foi realizada.
- T é o tamanho dos dados de treino, isto é, os dados utilizados pelo modelos para estimar os seus parâmetros ou curva de aprendizado.

Intuitivamente então, vemos que trata-se de uma medida que é escalada pelo seu erro médio obtido dentro dos dados de treino, o que mostra o quanto um modelo qualquer $f(x)$ pode realizar acima do que um simples *baseline*.

3.4.3 Seleção de Parâmetros

Por fim, dado os critérios de validação e métricas discutidos, vamos descrever a seleção dos hiperparâmetros para cada modelo testado no trabalho.

Modelos estatísticos

Vamos começar tratando de estimar os parâmetros dos modelos de Holt-Winters e ARIMA. Bibliotecas que implementam modelos estatísticos utilizam uma alternativa na seleção de parâmetros do modelo denominado estimador de máxima verossimilhança. A verossimilhança nada mais é do que a probabilidade dos dados serem resultados de um modelo específico, isto é, qual modelo teria gerado a série que estamos utilizando para treiná-lo? Para um modelo

de erro aditivo, isto é o mesmo que minimizar a soma dos quadrados do erro. A função de verossimilhança L é definida por 3.52.

$$L(\theta; x_1, \ldots, x_n) = f(x_1; \theta) \times \ldots \times f(x_n; \theta) = \prod_{i=1}^{n} f(x_i; \theta). \tag{3.52}$$

Onde $f(x, \theta)$ é função densidade de probabilidade, θ o parâmetro que queremos determinar e $x_1, x_2, ..., x_n$ os valores observados. O valor $L(\theta; x_1, \ldots, x_n)$, é dado pela distribuição conjunta 3.53 que quantifica a probabilidade de obter as amostras x_n dado o parâmetro θ [27].

$$P(X_1 = x_1, X_2 = x_2, ..., X_n = x_n | \theta) \tag{3.53}$$

Por fim, utiliza-se o Critério de Informação de Akaike (AIC) para seleção dentre todos os modelos possíveis gerados por hiperparâmetros distintos tanto para o modelo de Holt-Winters $(\alpha, \beta, \gamma, \phi)$, como para o ARIMA (p, d, q, P, D, Q, m). O critério para o modelo de Holt-Winters é definido como:

$$\text{AIC} = -2\log(L) + 2k$$

Onde L é a função de verossimilhança e k o total número de parâmetros e estados iniciais estimados. Já para o modelo ARIMA, o critério é dado por:

$$\text{AIC} = -2\log(L) + 2(p + q + k + 1)$$

Para o modelo de Holt-Winters utilizamos a biblioteca *statsmodels* [28], já para o modelo ARIMA, utilizamos a biblioteca *pmdarima* [29]. Assim, todos os parâmetros foram definidos utilizando o menor valor de AIC obtido nos dados de treino utilizando a validação de janela de rolamento para os dois conjunto de dados obtidos.

Modelos Não Paramétricos

Modelos baseado em árvores são denominados não paramétricos, isto é, não assume-se a priori a distribuição dos dados, estas são aprendidas durante o treinamento do modelo. Assim, eles contém somente hiperparâmetros a serem aprendidos, estes são controlados usualmente para evitar *overfitting* e *underfitting* conforme detalhada na seção de validação cruzada 3.4.1.

O mesmo critério foi utilizado para seleção de modelos conforme descrito anteriormente, dado o conjunto de treino, a validação cruzada é realizada e selecionamos o modelo que contém os melhores valores de MASE 3.51 na validação para ser avaliado no conjunto de teste.

CAPÍTULO 4

Simulações Computacionais e Discussão

4.1 Análise Descritiva e Preparação dos Dados

Conforme detalhado na seção 3.3, iremos trabalhar com séries temporais que apresentam uma estrutura hierárquica. Sendo assim, estudaremos a eficácia das técnicas apresentadas em duas bases de dados distintas: *visnights* e uma base extensa disponibilizada pelo Walmart em uma competição famosa na área de séries temporais, a *M5* [30]. Vamos detalhar um pouco a estrutura dos dados, bem como dar uma breve análise descritiva dos mesmos. O link para os notebooks com todo o código para reprodução pode ser encontrado no repositório público do github. [1]

4.1.1 Número de turistas noturnos para diversas regiões da Austrália - *vistnights*

Este conjunto de dados trata-se de uma base coletada pelo governo australiano que mostra o número total de visitantes noturnos (em milhões) por quadrimestre no período de 1998-2016 para mais de vinte regiões dentro de seis estados diferentes [1].

A série é divida entre 20 nós no ramo inferior, quatro nós representadas pelos estados e um nó representando a agregação de todos os níveis, ou seja, o total. Ao todo são 76 séries temporais. O grande desafio é a geração de previsões que sejam reconciliáveis. Para ilustração dos desafios de se trabalhar com múltiplas séries temporais, vamos trabalhar no passo a passo para treinarmos um modelo SARIMA com o nó raiz da hierarquia, isto é, queremos prever para os próximos dois

[1] https://github.com/vtoliveira/hierarchical-time-series

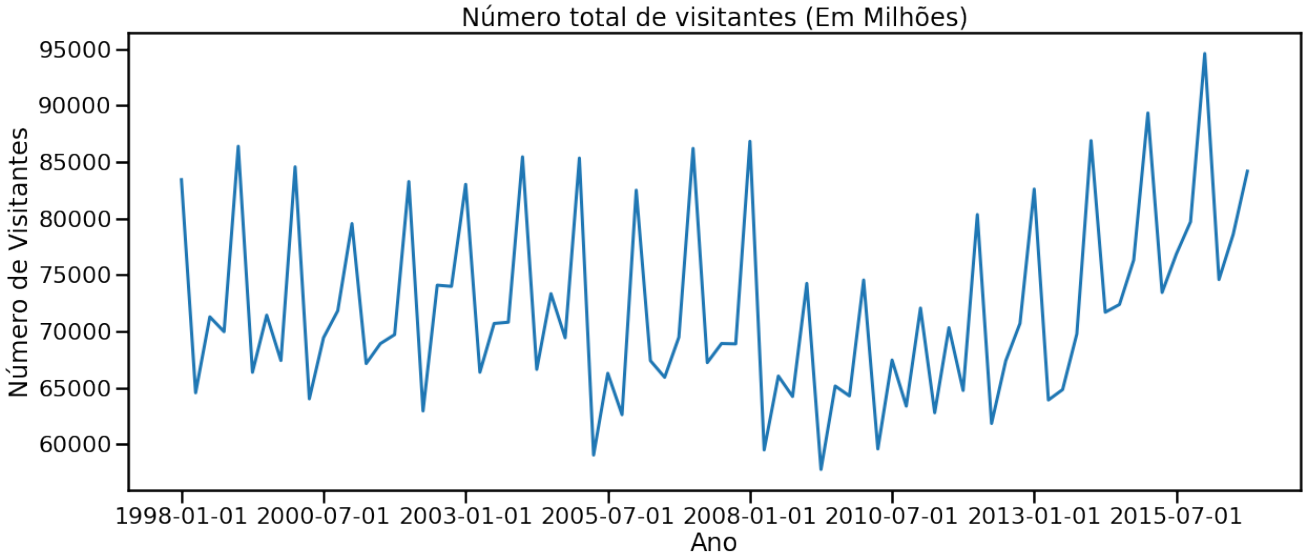

Figura 4.1: Série total do número de turistas (em milhões) visitando a Austrália, 1998-205.

anos o número total de visitantes noturnos que passarão pela Austrália. A figura 4.1 mostra a série total.

Antes de construirmos de fato o modelo, é importante notar algumas características fundamentais de séries temporais:

- A série não aparenta ter nenhuma tendência visível, porém vemos que em torno de 2008 houve uma queda e nos últimos anos um aumento e leve tendência, o que pode atrapalhar os resultados.
- Existe uma sazonalidade forte nos dados.
- Não aparenta ter *outliers*, isto é, pontos fora da normalidade da curva.
- Nenhum ciclo aparente.
- Variância da série aparenta ser constante, porém uma análise mais detalhada precisa ser feita utilizando gráficos de autocorrelação e autocorrelação parcial.

Como discutido na seção de modelos SARIMA, temos que determinar se a série é estacionária ou não, e aplicar as correções necessárias para o modelo. Vamos primeiro plotar os gráficos de PACF e ACF para a série total, conforme ilustrado na figura 4.2.

Nos gráficos acima, a sombra em azul representa o intervalo de confiança de 95% dos coeficientes obtidos ao se computar ambas as funções. Notamos que no gráfico de autocorrelação os pontos significativos estão espaçados a cada 4 intervalos de tempo, indicando que trata-se de uma série com forte sazonalidade quadrimestral, o que era esperado. Portanto, temos que usar

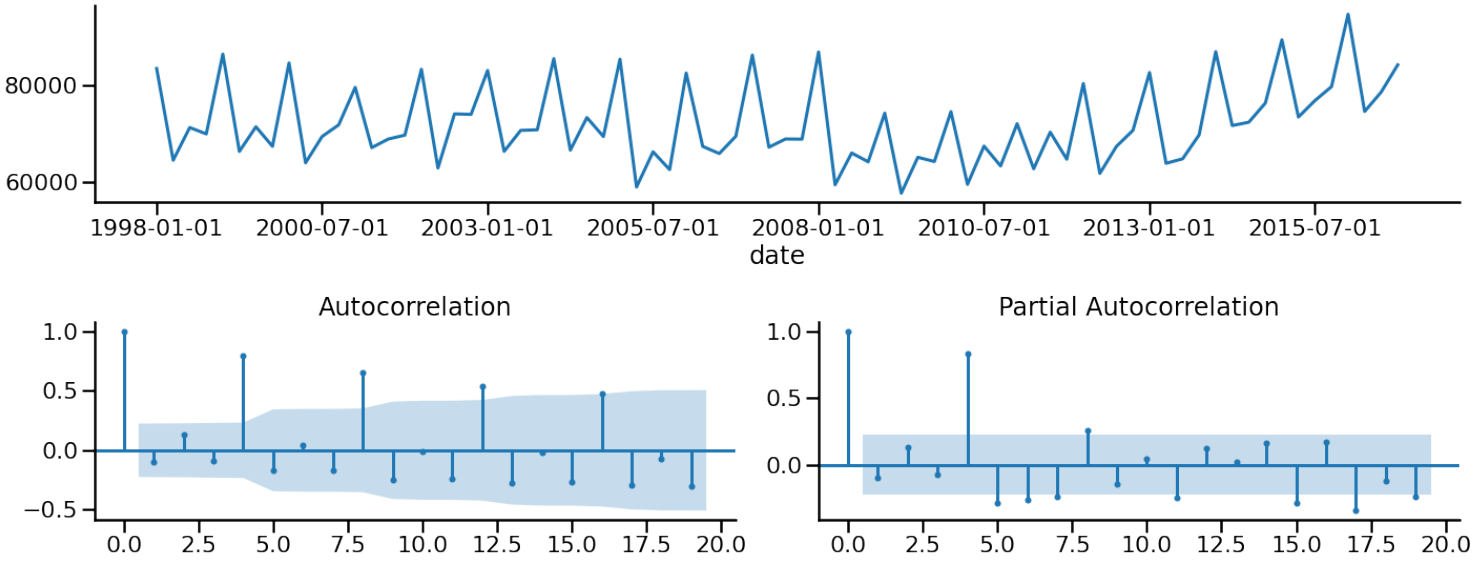

Figura 4.2: ACF e PACF gráficos para a série total de visitantes.

como coeficiente de diferenciação no modelo SARIMA, $D = 1$ e $s = 4$, indicando que temos que fazer uma diferenciação da série com intervalo de quatro períodos para que ela seja estacionária.

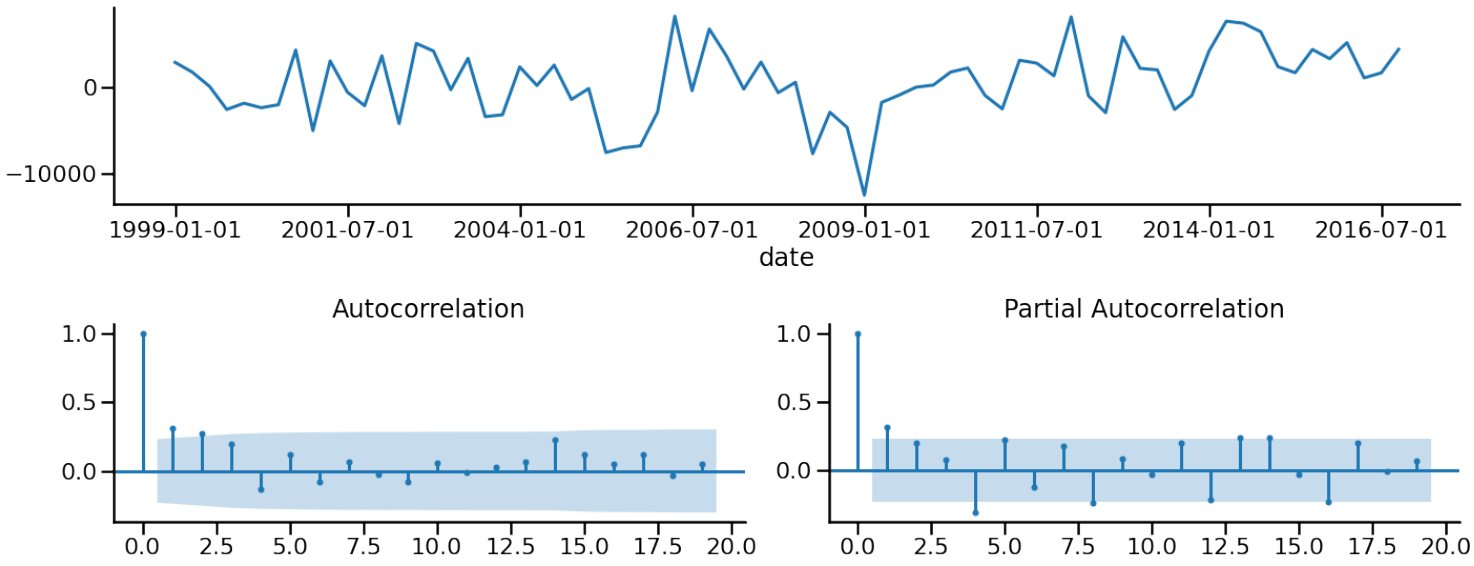

Figura 4.3: ACF e PACF gráficos para a série total de visitantes diferenciada para d = 4.

Vemos pela 4.3 que a série não apresenta mais nenhuma tendência de pontos fora do intervalo de confiança, exceto pelo primeiro ponto no gráfico de autocorrelação e o primeiro no gráfico de autocorrelação parcial. Portanto, determinamos os coeficientes da seguinte forma:

- p é provavelmente 1, devido ao gráfico PACF.
- d é igual a 0, pois não há nenhuma tendência nos dados.

- q é provavelmente 1, devido ao gráfico ACF.
- P pode ser 2, pois temos dois pontos significativos para o gráfico PACF em múltiplos de quatro, isto é, nos pontos 4 e 8.
- D será 1, pois determinamos uma diferenciação com $s = 4$.
- Q será 0, pois não vemos nenhum poto acima do intervalo de confiança no gráfico de ACF em múltiplos de 4, isto é, o período da série.

Portanto, temos um modelo $ARIMA(1,0,1) \times (2,1,0)_4$. Para treinar o modelo, foram separados 20% do conjunto de dados para teste e os restantes para treino. Importante lembrar que eles foram separados respeitando o aspecto temporal dos dados. Para testar de fato a eficácia do modelo, comparou-se o resultado obtido com um modelo denominado *naive*, que determina o valor predito como sendo igual ao último valor daquele quadrimestre.

	Mean Absolute Error	Mean Absolute Percentage Error
SARIMA	7813.24	10.47
Naive	7848.04	10.57

Temos que o modelo SARIMA apresenta desempenho ligeiramente melhor que o modelo *naive*, porém a tendência notada no final da série que não está representada nos dados de treino, torna a tarefa de previsão mais complexa conforme ilustrado pela figura 4.4. Este exemplo ilustra a dificuldade de se modelar uma única série temporal, por isso torna-se necessário alternativas para se modelar todas as séries de uma vez e sua reconciliação hierárquica. Na próxima seção utilizaremos os modelos descritos em 3.2 de forma a comparar quais produzem melhores resultados.

4.1.2 Dados de vendas de varejo do Walmart

Os dados disponibilizados pelo Walmart para a competição M5 envolve a quantidade de diversos produtos vendidos nos Estados Unidos organizados na forma de grupos. Esta estrutura de grupos é composta da seguinte forma: no nível inferior da série, temos 3.049 produtos, estes são classificados em 3 categorias, e para cada uma temos 7 departamentos. Os produtos são vendidos então em dez lojas espalhadas por três estados. A tabela 4.1 abaixo contabiliza o número de séries por cada nível de agregação.

A figura 4.5 ilustra o começo da hierarquia para este conjunto de dados.

Os dados ainda trazem um histórico de 5,4 anos, de um intervalo desde 29/01/2011 até 19/06/2016. Além do mais, temos algumas variáveis exógenas para nos ajudar na tarefa de previsão:

- **Calendário:** Informações sobre datas que o produto foi vendido.

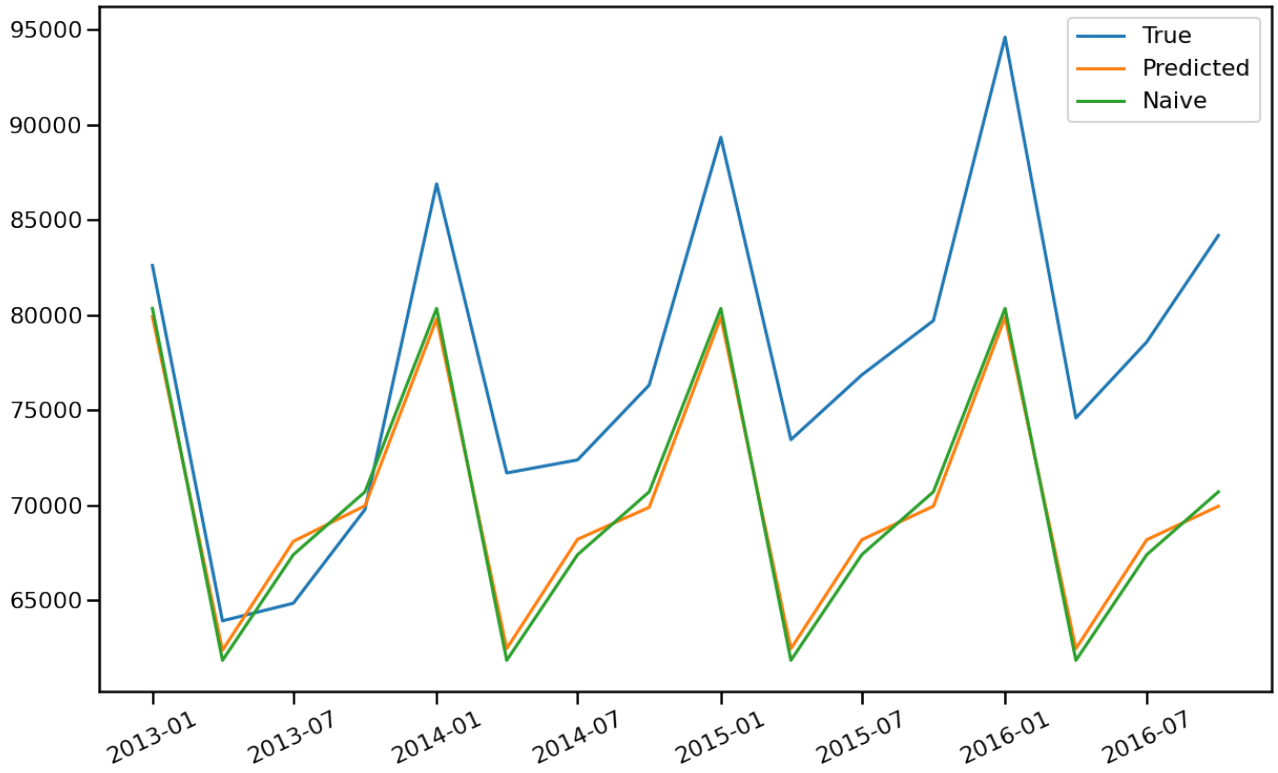

Figura 4.4: Comparação entre modelo SARIMA, Naive e a curva original.

- **Preço de Venda:** Informações sobre preços de produtos vendido por loja e data.

Apesar de ser uma base com um grande volume de dados, devido a restrições computacionais e de tempo, neste trabalhos vamos utilizar uma amostra dos dados composto pelos seguintes níveis: total, estados e lojas. Portanto, temos um total de trinta séries, 1 do agregado de vendas de todos os produtos, três por estado e 10 por loja. Estes dados são então transformados em séries mensais para diminuir o tamanho do dado de treino e suavizar ruídos gerados por vendas diárias.

4.2 Análise dos resultados

Vamos abordar agora os resultados obtidos com os experimentos em ambos conjuntos de dados, bem como os modelos descritos na fundamentação teórica. A figura 4.6 é um diagrama ilustrando o passo a passo realizado nas simulações.

Nível de Agregação	**# de séries**
Vendas de todos os produtos, agregado por todas as lojas/estados	1
Vendas de todos os produtos, agregado por estado	3
Vendas de todos os produtos, agregado por loja	10
Vendas de todos os produtos, agregado por categoria	3
Vendas de todos os produtos, agregado por departamento	7
Vendas de todos os produtos, agregado por estado e categoria	9
Vendas de todos os produtos, agregado por estado e departamento	21
Vendas de todos os produtos, agregado por loja e categoria	30
Vendas de todos os produtos, agregado por loja e departamento	70
Vendas de todos os produtos x, agregado por todas as lojas/estados	3049
Vendas de todos os produtos x, agregado por estado	9147
Vendas de todos os produtos x, agregado por loja	30490
Total	42840

Tabela 4.1: Número de séries por nível de agregação.

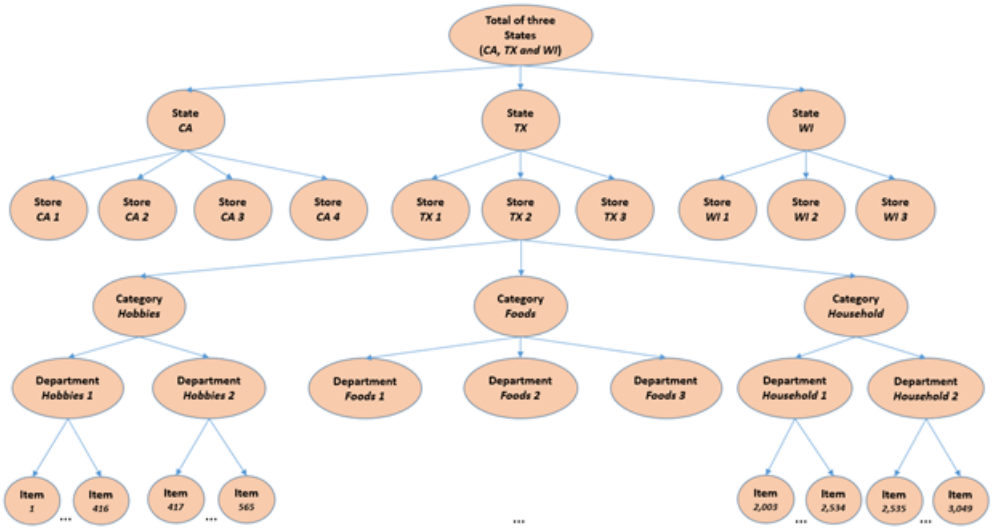

Figura 4.5: Estrutura Hierárquica de vendas nos dados do M5.

4.2.1 Separação treino e teste

Para começar, nosso objetivo neste trabalho é comparar como modelos de categorias diferentes se comportam ao aprender padrões em séries temporais, assim é importante separar os dados a priori em dois conjuntos: treino e teste. Conforme descrito [25], com o primeiro conjunto de dados, iremos inferir e estimar todos os parâmetros dos modelos e no segundo conjunto, o verdadeiro erro obtido será utilizado para avaliar seu desempenho final. Este passo é importante para que se evite qualquer viés ao trabalhar com todos os dados disponíveis. A tabela 4.2 resume as informações obtidas para ambos os dados. Os períodos de teste foram escolhidos para refletir um cenário real de projeto. No caso dos dados de visitantes noturnos da Austrália, saber antecipadamente 8 quadrimestres, ou dois anos à frente, é parte fundamental

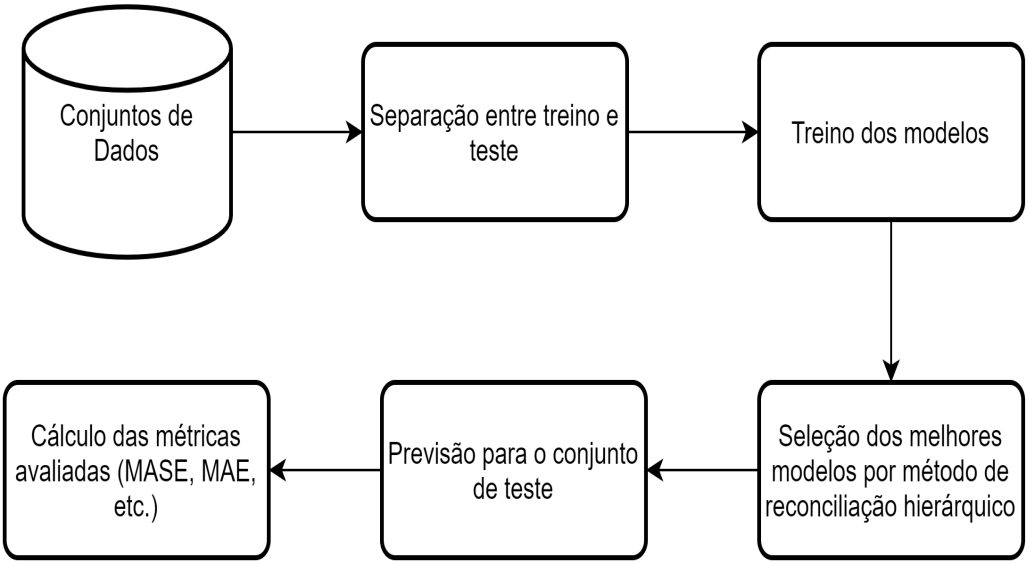

Figura 4.6: Diagrama com passo a passo realizado nas simulações para ambos conjuntos de dados.

do planejamento de qualquer ministério turístico. Já nos dados de vendas do *walmart*, saber a quantidade a ser demandada em certas regiões em um período de 28 dias torna-se essencial para toda a cadeia logística da empresa.

Conjunto de Dados	Periodo	# amostras treino	# amostras teste
visnights	01-1998/10-2014	68	8
walmart	02-2011/04-2015	51	12

Tabela 4.2: Informações dos dados de treino e teste para os conjuntos de dados a serem explorados.

4.2.2 Experimentos

Nesta seção vamos tratar dos resultados empíricos obtidos com os quatro modelos a serem testados, sendo eles: Random Forests, ARIMA, HW e Gradient Boosting. Para cada um deles, os seguintes métodos de reconciliação foram utilizados: Bottom-Up, Top-Down e Combinação Ótima. Para os modelos baseados em árvores, conforme descrito na seção 3.4.1 utilizamos validação cruzada variando o tamanho da janela bem como os parâmetros dos modelos. A tabela 4.3 mostra para cada parâmetro dos modelos de árvores os valores que foram utilizados na validação cruzada.

Os experimentos foram implementados utilizando a linguagem de programação Python [31],

Parâmetro	**Valores**
Número de árvores	[100, 200, 300]
Máxima profundidade	[3, 5, 7, Sem limite]
Porcentagem de variáveis amostradas para divisão	[50%, 80%, 100%]
Taxa de aprendizado (υ)	[0.001, 0.01, 0.1]

Tabela 4.3: Hiperparâmetros explorados na validação cruzada.

junto com as bibliotecas de cunho científico altamente qualificadas: *pandas*, *matplotlib*, *seaborn*, *numpy*, *scikit-learn*, *sktime* e *scikit-hts*.

Experimentos com dados *visnights*

Reconciliação	**Modelos**	**MAE**	**MAPE**	**MASE**
BU	Random Forest	9782.98	12.74	2.99
	Gradient Boosting	10023.15	13.05	3.06
	HW	6426.76	8.28	1.96
	ARIMA	8524.06	11.02	2.60
Recon. Ótima	Random Forest	9740.69	12.55	2.98
	Gradient Boosting	8109.66	10.33	2.48
	HW	**4604.64**	**5.89**	**1.40**
	ARIMA	4709.05	6.01	1.44
TD	Random Forest	15890.67	21.80	4.86
	Gradient Boosting	18047.61	25.14	5.52
	HW	13778.68	18.66	4.21
	ARIMA	17935.18	24.84	5.48

Tabela 4.4: Resultados para o conjunto de dados *visnights*.

A tabela 4.4 resume os resultados obtidos nas simulações. As métricas mostradas são médias simples de todos os nós ao longo da hierarquia. Vemos que a reconciliação ótima produz os melhores resultados para todos os modelos testados. Isto está de acordo com as expectativas pois a recombinação ótima em teoria trabalha de forma a minimizar o erro médio ao longo da hierarquia e isto consequentemente afeta o nó raiz. Notamos também que os modelos estatísticos performam substancialmente melhor que os modelos baseado em aprendizado de máquina, isto pode acontecer pois estas séries apresentam propriedades como sazonalidade e tendência que seriam facilmente capturados pelo modelos clássicos, enquanto que modelos de árvore necessitam de uma engenharia de atributos mais elaborado do que simplesmente utilizar dados passados.

Ao tratar com séries hierárquicas, não estamos somente interessados na série geral, mas sim como os algoritmos aproximam os dados ao longo de toda a cadeia. A figura 4.7 mostra os resultados dos valores previsto comparados com a base de teste para cada região da Austrália.

Comparar nó a nó não seria eficiente, a figura 4.8 mostra o resultado de todos os nós utilizando a métrica MASE. Conforme discutido na seção 3.4.2, trata-se de uma métrica escalonada pelos dados de treino para cada nó, sendo então uma métrica ideal para comparação de resultados em séries hierárquicas. Neste gráfico também ilustramos a linha teórica onde o MASE é igual a 1, onde valores maiores resultam numa série com desempenho pior do que um simple modelo naive, que neste caso é prever o valor futuro como sendo o último valor no periodo previsto, isto é, prevemos Janeiro de 2016 e 2017 como sendo o valor de Janeiro de 2015 obtido nos dados de treino. Apesar de alguns nós estarem acima de 1, os autores proponentes da métrica argumentam que para séries longas isto é esperado [**26**].

Vemos que há um balanço entre desempenho ao longo da hierarquia, enquanto alguns nós como a região *Metro* dentro do estado SAU tem bons resultados, o mesmo não ocorre para a região *Coast* no mesmo estado. Estes valores ilustram a dificuldade ao lidar com séries temporais hierárquicas, pois além do desafio de realizar previsões confiáveis, temos que criar métodos de reconciliação que sejam capazes de capturar as peculiaridades dos dados e agregar os nós de forma coerente.

Experimentos com dados *walmart*

A tabela 4.5 resume os resultados obtidos com os dados de vendas da varejista *walmart* para nó raiz. É interessante notar que neste conjunto de dados, as abordagens *top-down* apresentam resultados melhores, enquanto que nos dados visnights, todos os métodos de reconciliação que tinham melhores estimativas eram baseado em combinação ótima, aqui vemos que reconciliação e modelo se diferem caso a caso. Para os modelos de árvores, Random Forest tem *bottom-up* como melhor método, enquanto que o Gradient Boosting tem *top-down*. No geral, o melhor modelos continua sendo um simples HW, porém com uma abordagem *top-down*. Também vale notar que neste caso os modelos baseado em Aprendizado de Máquina chegam mais perto de performar melhor que os modelos clássicos, com destaque para o modelo de Gredient Boosting com reconciliação *top-down*.

Os resultados diferem do conjunto de dados anterior provavelmente pela sua natureza. Por ser uma série temporal diária com reamostragem mensal, além de ser dados de uma varejista, isto é, diversos componentes podem afetar a demanda em certo local como a ocorrência de feriados, problemas na cadeia logística, falta de produtos, ou simplesmente alguma mudança física nas lojas, acaba por causar choques nos dados que os modelos clássicos tem maiores dificuldades para capturar. Possivelmente ao utilizar a série agregada semanalmente ou diariamente produziria resultados ainda mais próximos de favorecer modelos de aprendizado de máquina.

A figura 4.9 mostra o resultado por estado e algumas lojas para o modelo de Gradient Boosting.

Vemos como em alguns nós da hierarquia como no estado do Texas (TX), e na loja represen-

Reconciliação	Modelos	MAE	MAPE	MASE
BU	Random Forest	78957.21	6.78	0.79
	Gradient Boosting	95790.62	8.31	0.95
	HW	83318.29	7.22	0.83
	ARIMA	92719.34	8.03	0.92
Recon. Ótima	Random Forest	94192.25	8.17	0.94
	Gradient Boosting	81769.05	7.09	0.81
	HW	79026.80	6.85	0.79
	ARIMA	101530.04	8.82	1.01
TD	Random Forest	94942.48	8.24	0.95
	Gradient Boosting	77839.57	6.75	0.77
	HW	**75659.50**	**6.55**	**0.75**
	ARIMA	100114.21	8.69	1.00

Tabela 4.5: Resultados para o conjunto de dados *walmart*.

tada como 1 na Califórnia (CA_1) o modelo é capaz de antecipar tendências sazonais.

Por fim, temos na figura 4.10 o resultado obtido para o modelo HW com *top-down*.

Outra diferença substancial dos modelos obtidos nestes dados é a melhora nos resultados gerais obtidos na métrica MASE. Conforme discutido, esta série apresenta uma irregularidade maior, o que torna difícil a tarefa de um modelo simples, como utilizar o valor do último mês, ser no fim melhor do que um modelo estatístico.

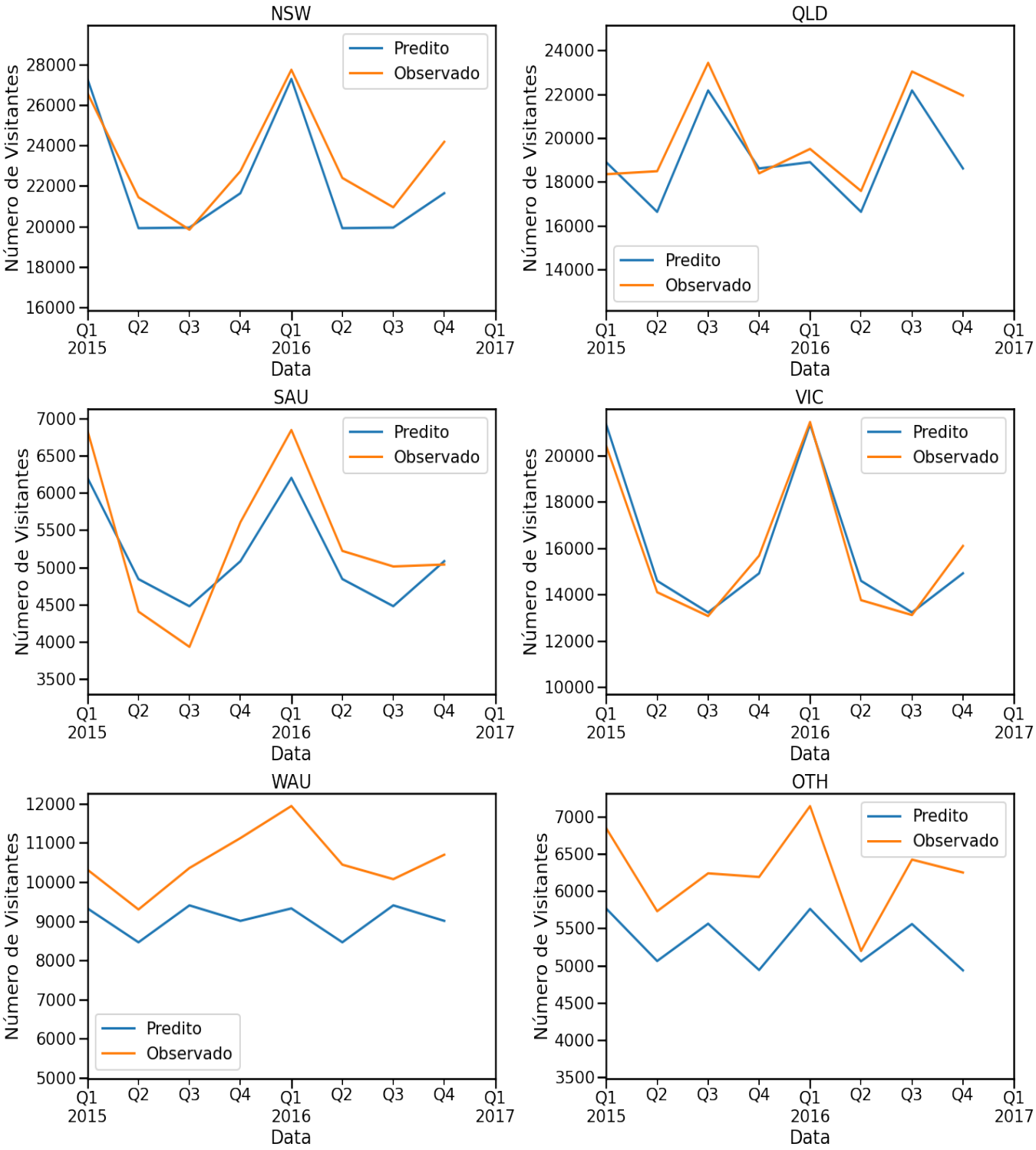

Figura 4.7: Previsto versus Observado das regiões australianas utilizando o melhor modelo obtido.

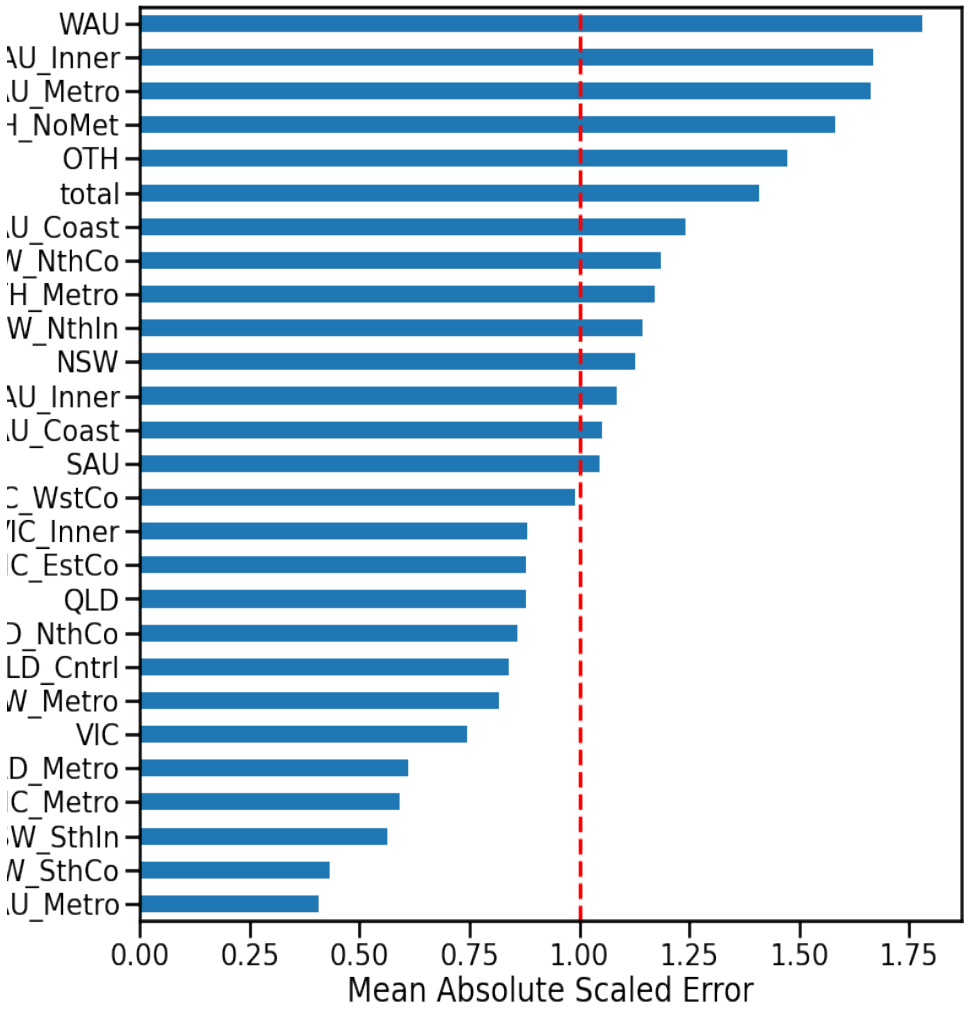

Figura 4.8: Métrica MASE para todos os nós na hierarquia.

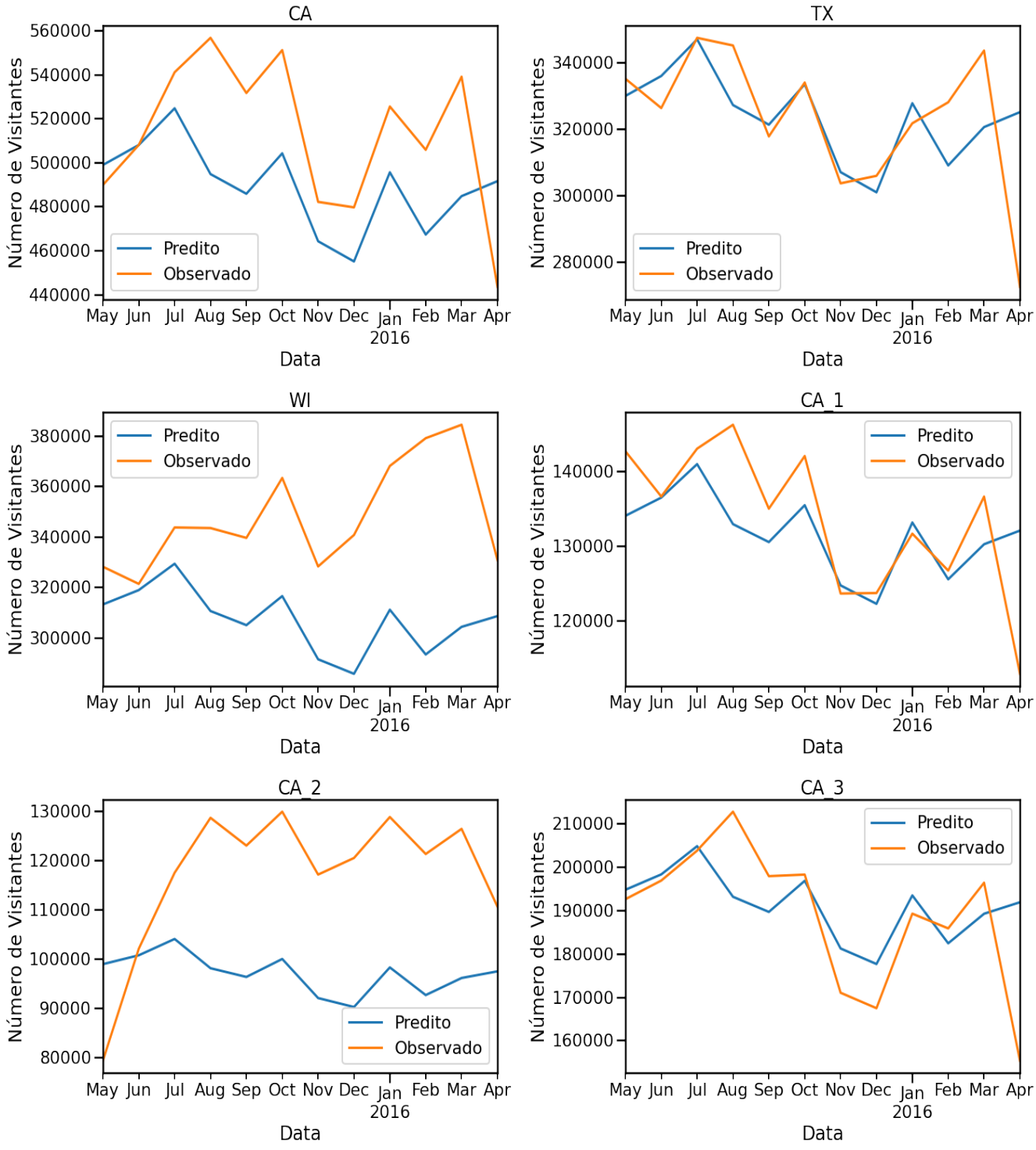

Figura 4.9: Previsto versus Observado dos estados e algumas lojas da varejista walmart nos EUA para o modelo de Gradient Boosting.

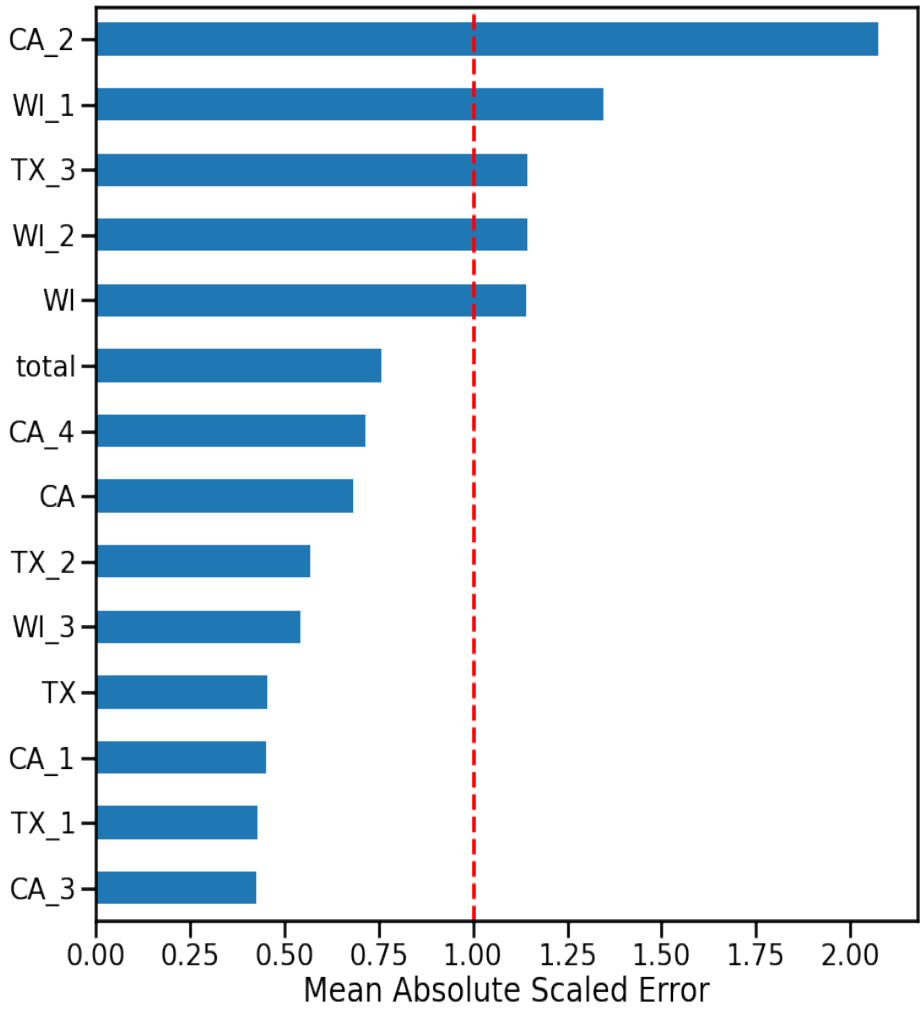

Figura 4.10: Métrica MASE para todos os nós na hierarquia para os dados do walmart.

CAPÍTULO 5

Conclusão

Este trabalho investigou a eficácia de modelos paramétricos e não paramétricos aplicados à previsão de demanda em estruturas temporais hierárquicas em dois conjuntos de dados distintos. Três técnicas distintas de reconciliação foram avaliadas: top-down, bottom-up e Combinação Ótima. Para cada classe de reconciliação e modelo os resultados foram avaliados utilizando como base a métrica MASE, uma medida adequada quando estamos trabalhando com grupos de séries temporais com escalas distintas. Os principais resultados mostram que nem sempre o método mais sofisticado é o melhor, tanto na modelagem como na reconciliação, e que as previsões são dependentes do problema e da estrutura do dado de treinamento.

Os experimentos realizados cumpriram o objetivo de explorar diferentes técnicas de modelagem, como a estatística clássica e a não paramétrica baseada em modelos de árvore, ao tratar séries temporais, bem como as dificuldades ao tratar dados com estruturas temporais como estacionariedade, sazonalidade e autocorrelação. O melhor modelo para o conjunto *visnights* foi o modelo HW com reconciliação ótima, enquanto para o conjunto do *walmart*, HW com top-down obteve os melhores resultados. A grande conclusão aqui é que modelos de árvore necessitam de uma engenharia de feature para que eles consigam aprender padrões que modelos clássicos capturam mais facilmente como sazonalidade, por exemplo.

Os principais desafios do projeto foram a implementação de parte do código que calcula os diferentes tipos de reconciliação na linguagem Python e o tratamento dos dados para que estes fossem adequados ao problema que queríamos investigar. Treinar cada modelo para o nó da hierarquia é custoso computacionalmente e a paralelização das tarefas é essencial.

Para trabalhos futuros é interessante investigar a aplicação destes modelos em séries com amostragem maiores como séries diárias de demanda e com a utilização de variáveis exógenas

para verificar como os modelos utilizam essa informação extra na hora de estimar um valor futuro.

Bibliografia

[**1**] R. J. Hyndman e G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[**2**] S. Makridakis, "The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms," *Futures*, vol. 90, pags. 46–60, 2017.

[**3**] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[**4**] R. H. Shumway e D. S. Stoffer, *Time series analysis and its applications: with R examples*. Springer, 2017.

[**5**] E. S. Gardner Jr, "Exponential smoothing: The state of the art," *Journal of forecasting*, vol. 4, n^{o} 1, pags. 1–28, 1985.

[**6**] M. Tranmer e M. Elliot, "Multiple linear regression," *The Cathie Marsh Centre for Census and Survey Research (CCSR)*, vol. 5, n^{o} 5, pags. 1–5, 2008.

[**7**] S. Makridakis, E. Spiliotis, e V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PloS one*, vol. 13, n^{o} 3, pag. e0194889, 2018.

[**8**] N. K. Ahmed, A. F. Atiya, N. E. Gayar, e H. El-Shishiny, "An empirical comparison of machine learning models for time series forecasting," *Econometric Reviews*, vol. 29, n^{o} 5-6, pags. 594–621, 2010.

[**9**] M. Abolghasemi, R. J. Hyndman, G. Tarr, e C. Bergmeir, "Machine learning applications in time series hierarchical forecasting," *arXiv preprint arXiv:1912.00370*, 2019.

[**10**] G. Tirkes, C. Guray, e N. Celebi, "Demand forecasting: A comparison between the holt-winters, trend analysis and decomposition models/predvidanje potraznje: usporedba izmedu

holt-winters modela, analize trenda i modela dekompozicije," *Tehnicki Vjesnik-Technical Gazette*, vol. 24, nº S2, pags. 503–510, 2017.

[**11**] J. C. Sousa, H. M. Jorge, e L. P. Neves, "Short-term load forecasting based on support vector regression and load profiling," *International Journal of Energy Research*, vol. 38, nº 3, pags. 350–362, 2014.

[**12**] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan, e G. Amaratunga, "Ensemble deep learning for regression and time series forecasting," em *2014 IEEE symposium on computational intelligence in ensemble learning (CIEL)*. IEEE, 2014, pags. 1–6.

[**13**] D. Delen, "A comparative analysis of machine learning techniques for student retention management," *Decision Support Systems*, vol. 49, nº 4, pags. 498–506, 2010.

[**14**] J. P. U. Cadavid, S. Lamouri, e B. Grabot, "Trends in machine learning applied to demand & sales forecasting: A review," 2018.

[**15**] G. E. Box, G. M. Jenkins, G. C. Reinsel, e G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[**16**] P. J. Brockwell, P. J. Brockwell, R. A. Davis, e R. A. Davis, *Introduction to time series and forecasting*. Springer, 2016.

[**17**] D. Kwiatkowski, P. C. Phillips, P. Schmidt, Y. Shin *et al.*, "Testing the null hypothesis of stationarity against the alternative of a unit root," *Journal of econometrics*, vol. 54, nº 1-3, pags. 159–178, 1992.

[**18**] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *International journal of forecasting*, vol. 20, nº 1, pags. 5–10, 2004.

[**19**] L. Breiman, "Random forests," *Machine learning*, vol. 45, nº 1, pags. 5–32, 2001.

[**20**] G. James, D. Witten, T. Hastie, e R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[**21**] M. Rosenblatt, "A central limit theorem and a strong mixing condition," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 42, nº 1, pag. 43, 1956.

[**22**] G. Athanasopoulos, R. A. Ahmed, e R. J. Hyndman, "Hierarchical forecasts for australian domestic tourism," *International Journal of Forecasting*, vol. 25, nº 1, pags. 146–166, 2009.

[**23**] S. L. Wickramasuriya, G. Athanasopoulos, e R. J. Hyndman, "Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization," *Journal of the American Statistical Association*, vol. 114, nº 526, pags. 804–819, 2019.

[**24**] R. J. Hyndman, R. A. Ahmed, G. Athanasopoulos, e H. L. Shang, "Optimal combination forecasts for hierarchical time series," *Computational statistics & data analysis*, vol. 55, nº 9, pags. 2579–2589, 2011.

[**25**] V. Cerqueira, L. Torgo, e I. Mozetic, "Evaluating time series forecasting models: An empirical study on performance estimation methods," *arXiv preprint arXiv:1905.11744*, 2019.

[**26**] R. J. Hyndman *et al.*, "Another look at forecast-accuracy metrics for intermittent demand," *Foresight: The International Journal of Applied Forecasting*, vol. 4, nº 4, pags. 43–46, 2006.

[**27**] B. S. Kim, S. G. Park, Y. K. You, e S. I. Jung, "Probability & statistics for engineers & scientists," 2011.

[**28**] S. Seabold e J. Perktold, "Statsmodels: Econometric and statistical modeling with python," em *Proceedings of the 9th Python in Science Conference*, vol. 57. Austin, TX, 2010, pag. 61.

[**29**] T. G. Smith *et al.*, "pmdarima: Arima estimators for Python," 2017–, [Online; accessed ¡today¿]. [Online]. Available: http://www.alkaline-ml.com/pmdarima

[**30**] M. O. F. Center. (2019) The m5 competition. [Online]. Available: https://mofc.unic.ac.cy/m5-competition/

[**31**] G. Rossum, "Python reference manual," 1995.