UNIVERSIDADE FEDERAL DE SANTA CATARINA

CAMPUS REITOR JOÃO DAVID FERREIRA LIMA

PROGRAMA DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Walter Harlinzon Mayorga Olabarrera

DESENVOLVIMENTO DE UM SISTEMA DE PREVISÃO DE DEMANDA DE EXAMES DE TELEDERMATOLOGIA

Florianópolis, Santa Catarina – Brasil

2020

Walter Harlinzon Mayorga Olabarrera

DESENVOLVIMENTO DE UM SISTEMA DE PREVISÃO DE DEMANDA DE EXAMES DE TELEDERMATOLOGIA

Trabalho de Conclusão de Curso submetido ao Programa de Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do Grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Ph.D. Danilo Silva

Florianópolis, Santa Catarina – Brasil

2020

Catalogação na fonte pela Biblioteca Universitária da Universidade Federal de Santa Catarina.
Arquivo compilado às 21:56h do dia 17 de fevereiro de 2020.

Walter Harlinzon Mayorga Olabarrera
Desenvolvimento de um Sistema de Previsão de Demanda de Exames de Teledermatologia / Walter Harlinzon Mayorga Olabarrera; Orientador, Prof. Ph.D. Danilo Silva – Florianópolis, Santa Catarina – Brasil, 21:56, 12 de fevereiro de 2020.
105 p.

Trabalho de Conclusão de Curso – Universidade Federal de Santa Catarina, EEL – Departamento de Engenharia Elétrica e Eletrônica, CTC – Centro Tecnológico, Programa de Graduação em Engenharia Elétrica.

Inclui referências

1. Séries Temporais, 2. Suavização Exponencial, 3. ARIMA, 4. LSTM, 5. GRU, 6. RNN, 7. Redes Neurais, 8. Telemedicina, 9. Telessaúde, 10. Aprendizagem de Máquina, I. Prof. Ph.D. Danilo Silva II. Programa de Graduação em Engenharia Elétrica III. Desenvolvimento de um Sistema de Previsão de Demanda de Exames de Teledermatologia

CDU 02:141:005.7

Walter Harlinzon Mayorga Olabarrera

Desenvolvimento de um sistema de previsão de demanda de exames de teledermatologia

Este Trabalho foi julgado adequado como parte dos requisitos para obtenção do Título de Bacharel em Engenharia Elétrica e aprovado, em sua forma final, pela Banca Examinadora

Florianópolis, 12 de fevereiro de 2020.

Prof. Renato Lucas Pacheco, Dr.
Coordenador do Curso de Graduação em Engenharia Elétrica,
em exercício

Banca Examinadora:

Prof. Danilo Silva, Ph.D.
Orientador
Universidade Federal de Santa Catarina

Prof. Hans Helmut Zürn, Ph.D.
Universidade Federal de Santa Catarina

Alexandre Savaris, Ph.D.
Universidade Federal de Santa Catarina

AGRADECIMENTOS

À minha família, especialmente a Senhora Nelly, minha mãe, que, durante meu percurso pela graduação, sempre esteve ao meu lado com seu amor e carinho.

Aos amigos do Laboratório de Telemedicina[1] e do curso de Engenharia Elétrica, ambos da Universidade Federal de Santa Catarina, que sempre ajudaram no compartilhamento de conhecimento e dicas sobre o trabalho. E ao meu orientador, que, com muita paciência, me guiou para continuar até o fim.

[1] Responsáveis pelo desenvolvimento do STT/SC `http://site.telemedicina.ufsc.br/`.

RESUMO

Nos últimos anos, houve aumento nos gastos públicos com a saúde, devido ao rápido envelhecimento da população. Em virtude desse aumento, a ciência de dados juntamente à Telemedicina e à Telessaúde apresentam-se como alternativa para melhorar a eficiência dos gastos públicos e elevar a produtividade dos atendimentos perante a sociedade. Diante desse contexto, verificou-se a necessidade de propor ferramenta de previsão de demanda, a qual permite que o gestor de saúde pública possa fazer análise da previsão do aumento do uso do sistema público de saúde. Para isso, este trabalho propõe o desenvolvimento de um sistema de previsão de exames de teledermatologia integrado ao Sistema Integrado Catarinense de Telemedicina e Telessaúde (STT/SC), em que seu processo de desenvolvimento se separa em duas etapas: desenvolvimento de modelos preditivos e integração dos modelos preditivos ao STT/SC. Os modelos preditivos utilizados se dividem entre técnicas de abordagem clássica (Autorregresivos, Integrados e de Médias Móveis - ARIMA e Suavização Exponencial) e técnicas de abordagem de aprendizado (*Long Short Term Memory* - LSTM, *Gated Recurrent Unit* - GRU, *Recurrent Neural Network* - RNN). Para medir o desempenho dos modelos, os dados foram divididos em dados de treinamento e dados de teste, em que foi utilizado o método de janela de expansão, no qual o conjunto de treinamento vai aumentando à medida que um erro é calculado. A avaliação dos modelos ocorreu com os indicadores Erro Médio Absoluto (MAE) e Raiz do Erro Quadrático Médio (RMSE). Por fim, para o conjunto de dados analisados, os modelos ARIMA, Suavização Exponencial e RNN apresentaram bons resultados.

Palavras-chaves: Séries Temporais. Suavização Exponencial. ARIMA. LSTM. GRU. RNN. Redes Neurais. Telemedicina. Telessaúde. Aprendizagem de Máquina.

ABSTRACT

In recent years, there has been an increase in public spending on health, caused by the fast aging of the population. This increase led to some possible alternative to improve the efficiency of public spending and increase the productivity of services to society: data science, Telemedicine and Telehealth. Considering this context, there was a need to propose a demand forecasting tool, which allows the public health manager to analyze the forecast of the increase in the use of the public health system. For this, this work proposes the development of a system for forecasting teledermatology exams integrated with the Santa Catarina State Integrated Telemedicine and Telehealth System (STT/SC), in which its development process is divided into two stages: development of predictive models and integration of predictive models to STT/SC. The predictive models used are divided in: classical approach techniques (Autoregressive Integrated Moving Average - ARIMA and Exponential Smoothing) and learning approach techniques (Long Short Term Memory - LSTM, Gated Recurrent Unit - GRU, Recurrent Neural Network - RNN). To measure the performance of the models, the data was divided into training data and test data, using the expansion window method, in which the training set increases as an error is identified. The models were evaluated using the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) indicators. Finally, it is concluded that the analyzed data set, the ARIMA, Exponential Smoothing and RNN models presented good results.

Keywords: Time Series. Exponential Smoothing. ARIMA. LSTM. GRU. RNN. Neural Networks. Telemedicine. Telehealth. Machine Learning.

LISTA DE FIGURAS

Figura 1.1–GISTelemed – análise temporal dos exames de teledermatologia de Santa Catarina 26

Figura 1.2–Diagrama das etapas do processo de registro de um exame teledermatológico 28

Figura 1.3–Estrutura de dados do *Apache Druid* 31

Figura 2.1–Comparação entre série estacionária e série não estacionária 35

Figura 2.2–Estratégias para avaliar o modelo preditivo 37

Figura 2.3–Hierarquia dos modelos preditivos 38

Figura 3.1–Exemplo do modelo Médias Móveis Simples (MMS) para $r = 4$ e $r = 8$. 42

Figura 3.2–Exemplo do modelo SES para $A = 0.1$ e $A = 0.5$ 43

Figura 3.3–Exemplo do modelo SEH 44

Figura 3.4–Exemplo do modelo suavização exponencial de Holt-Winters (HW) para $A = 0.1$, $B = 0.31$ e $C = 0.01$. 45

Figura 3.5–Fluxograma para determinação dos parâmetros ARIMA ou SARIMA . 48

Figura 3.6–Machine Learning – novo paradigma de programação. 50

Figura 3.7–Representação do modelo de regressão linear 52

Figura 3.8–Ajuste dos dados em função do custo 54

Figura 3.9–*Multilayer perceptron* (MLP). 56

Figura 3.10–Funções de ativação 58

Figura 3.11–Arquitetura *Simple* RNN. 60

Figura 4.1–Série original da quantidade de exames de teledermatologia por mês nos últimos 5 anos 66

Figura 4.2–Algoritmo de treinamento para $h = 1$ 67

Figura 4.3–Algoritmo de treinamento para $h = 12$ 68

Figura 4.4–Suavização exponencial – exemplo de implementação para $h = 3$ 70

Figura 4.5–Diferenciação I(d). 71

Figura 4.6–Funções de autocorrelação ACF e PACF 72

Figura 4.7–Resíduos do modelo ARIMA(4,1,0) 72

Figura 4.8–Modelo ARIMA(4,1,0) 73
Figura 4.9–Modelo SARIMA(1,1,0)x$(0,1,0)_{12}$ 73
Figura 4.10–ARIMA/SARIMA – exemplo de implementação para $h = 3$ 74
Figura 4.11–Regressão Linear – etapas de implementação ($h = 3$) 75
Figura 4.12–Etapas da implementação dos modelos de redes neurais 76
Figura 4.13–Curvas da função perda por épocas para cada modelo 78

Figura 5.1–Gráfico com os resultados da predição do conjunto de testes de 2019 – Parte 1 83
Figura 5.2–Gráfico com os resultados da predição do conjunto de testes de 2019 – Parte 2 84

Figura 6.1–Diagrama de processos e fluxo de dados 87
Figura 6.2–*Saiku*: Gráfico de barras da quantidade de exames de teledermatologia para 2020 88
Figura 6.3–Menu principal da ferramenta Saiku. 89
Figura 6.4–*Saiku*: Gráfico com projeção para 2020 da quantidade de exames. 90
Figura 6.5–*Saiku*: Ferramenta permite escolher o modelo preditivo para análise. . . 91
Figura 6.6–*Saiku*: Tabelas de predição. 91

LISTA DE TABELAS

Tabela 3.1–Resumo dos modelos Suavização Exponencial. 46

Tabela 4.1–Tabela de atributos 64

Tabela 4.2–Evolução histórica de exames dermatológicos 65

Tabela 4.3–Parâmetros dos modelos 79

Tabela 5.1–Comparação do desempenho dos modelos 81

Tabela 6.1–Tecnologias utilizadas no desenvolvimento do sistema 85

LISTA DE ABREVIATURAS E SIGLAS

ADF	Augmented Dickey Fuller / Dickey-Fuller aumentado
ARIMA	AutoRegressive Integrated Moving Average / Modelo Autorregressivo Integrado de Médias Móveis
CSV	Comma-Separated Values / Valores Separados por Vírgula
ETL	Extract, Transform and Load / Extração, Transformação e Carga
LSTM	Long Short-Term Memory / Memória de Longo Prazo
MAE	Mean Absolute Error / Erro Médio Absoluto
PACF	Partial Autocorrelation Function / Função de Autocorrelação Parcial
RMSE	Root Mean Square Error / Raiz do Erro Quadrático Médio
GRU	Gated Recurrent Units / Unidade Recorrente Fechada
RNN	Recurrent Neural Network / Redes Neurais Recorrentes
RNA	Redes neurais artificiais
SES/SC	Secretaria de Estado da Saúde de Santa Catarina
STT/SC	Sistema Integrado Catarinense de Telemedicina e Telessaúde
UFSC	Universidade Federal de Santa Catarina

SUMÁRIO

1 INTRODUÇÃO 21

1.1 CONTEXTUALIZAÇÃO 21

1.2 DESCRIÇÃO DO PROBLEMA 23

1.3 MOTIVAÇÃO 25

1.4 OBJETIVOS 27

1.4.1 Objetivo geral 27

1.4.2 Objetivos específicos 27

1.5 CENÁRIO DE APLICAÇÃO 27

1.6 METODOLOGIA 29

1.6.1 Modelagem da predição 29

1.6.2 Desenvolvimento dos modelos de predição 30

1.6.3 Desenvolvimento do sistema de predição integrado com o STT/SC . . 30

1.7 ESTRUTURA DO TRABALHO 31

CAPÍTULO 2 33

2 FUNDAMENTAÇÃO TEÓRICA 33

2.1 SÉRIES TEMPORAIS 33

2.1.1 Estacionariedade 33

2.1.2 Componentes de séries temporais 34

2.2 PREDIÇÃO DE SÉRIES TEMPORAIS 36

2.3 ABORDAGEM DOS MODELOS 38

CAPÍTULO 3 41

3 MODELOS DE PREDIÇÃO DE SÉRIES TEMPORAIS . . . 41

3.1 SUAVIZAÇÃO EXPONENCIAL 41

3.1.1 Médias Móveis Simples (MMS) 41

3.1.2 Suavização Exponencial Simples (SES) 42

3.1.3 Suavização Exponencial de Holt (SEH) 43

3.1.4 Suavização Exponencial de Holt-Winters (HW) 44

3.1.5 Resumo dos Modelos **46**
3.2 ARIMA 47
3.2.1 Autocovariância e autocorrelação **49**
3.3 APRENDIZADO DE MÁQUINA (*MACHINE LEARNING*) 50
3.4 REGRESSÃO 52
3.4.1 Função custo **53**
3.4.2 Gradiente Descendente (GD) **54**
3.4.3 Redes Neurais Artificiais (RNA) **55**
3.4.4 Funções de ativação **57**
3.4.5 Backpropagation **58**
3.4.6 Redes Neurais Recorrentes (RNN) **60**

CAPÍTULO 4 . . . . . **63**

4 DESENVOLVIMENTO DOS MODELOS **63**
4.1 INFORMAÇÕES BÁSICAS 63
4.1.1 Especificação do conjunto de dados **64**
4.1.2 Coleta dos dados **64**
4.1.3 Separação dos dados **66**
4.1.4 Algoritmos de treinamento **66**
4.2 IMPLEMENTAÇÃO DO MODELO SUAVIZAÇÃO EXPONENCIAL 68
4.3 IMPLEMENTAÇÃO DO MODELO ARIMA 69
4.3.1 Encontrar a ordem de diferenciação *I(d)* **70**
4.3.2 Encontrar a ordem do termo AR(p) e MA(q) **71**
4.3.3 Aplicação dos parâmetros **71**
4.4 IMPLEMENTAÇÃO DO MODELO REGRESSÃO LINEAR 75
4.5 IMPLEMENTAÇÃO DOS MODELOS DE REDES NEURAIS 76
4.6 RESUMO DA ESCOLHA DOS PARÂMETROS 79

CAPÍTULO 5 . . . . . **81**

5 RESULTADOS **81**

CAPÍTULO 6 . . . . . **85**

6 INTEGRAÇÃO COM O STT/SC 85
6.1 INTERFACE DE VISUALIZAÇÃO DOS DADOS 88

CONCLUSÃO 92

REFERÊNCIAS 95

ANEXO A – IMAGEM AMPLIADA DO MÓDULO GISTELEMED DO STT/SC 101

ANEXO B – ALGORITMO DE TREINAMENTO APLICADO AO MÉTODO HW PARA $h = 1$ 103

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

Houve, nos últimos anos, oscilação com os gastos públicos em saúde no Brasil; porcentagem de cerca de 8% do PIB (SALDIVA; VERAS, 2018). Uma melhoria na saúde leva ao crescimento econômico por meio de ganhos de longo prazo em capital humano e físico, o que eleva a produtividade e a renda per capita da população. O aumento nos custos na área da saúde deve-se, principalmente, ao rápido envelhecimento da população (ABEDJAN et al., 2019). Portanto, a eficiência dos gastos com tecnologia podem elevar a produtividade dos atendimentos.

A telemedicina é um campo emergente da medicina que utiliza as tecnologias da informação e telecomunicações para o intercâmbio de informações médicas entre os profissionais de saúde e os pacientes. A importância da telemedicina está na redução de custo e tempo, principalmente pela automatização dos processos, que em alguns casos, o paciente não precisa se locomover até o estabelecimento de saúde (EREN; WEBSTER, 2015; WALLAUER et al., 2008). Fica evidente que existe uma relação entre o aumento do investimento em telemedicina e o aumento da eficiência dos serviços da área da saúde pública.

A teledermatologia constitui subcategoria da telemedicina em que há prestação de atendimento especializado à distância por um dermatologista. Devido à natureza orientada visualmente e à crescente importância da imagem digital nessa especialidade, a dermatologia adequa-se à aplicação em telemedicina, especialmente em dermatoscopia, dermatopatologia e microscopia confocal de reflexo (SOYER et al., 2011). Dessa forma, essa especialidade precisa de uma boa infraestrutura de serviços, para o armazenamento dos dados e imagens dos pacientes.

Atualmente, houve crescimento considerável a respeito do número de aplicativos de telemedicina e telessaúde. Ainda assim, pacientes de cuidados intensivos baseados remotamente precisam de monitoração constante, e o teletrabalho doméstico para pacientes que sofrem de doenças crônicas, como doenças cardíacas, está aumentando continuamente, o que tem pressionado recursos médicos escassos (EREN; WEBSTER, 2015). Portanto, há grande caminho a percorrer para direcionar os gastos públicos de forma mais eficiente

nas diferentes áreas da telemedicina.

Não há dúvidas sobre o impacto da ciência de dados para a tecnologia, economia e sociedade. De tal forma que, cada vez mais, as organizações estão sendo incentivadas para melhoria do modelo de negócio com a inclusão de aprendizagem de máquinas, principalmente para solucionar gargalos na área da saúde (CONSOLI; RECUPERO; PETKOVIĆ, 2019). Diante desse contexto, do aumento considerável de informações, do incentivo para aplicação de aprendizagem de máquina, torna-se importante, para este trabalho, a criação de ambiente de ciência de dados para colocar em prática algumas técnicas de predição de séries temporais com o objetivo de melhorar, ainda mais, a eficiência com o investimento na área da saúde.

1.2 DESCRIÇÃO DO PROBLEMA

Com o avanço da Telessaúde e Telemedicina, muitas informações, que antes estavam em documentos impressos ou descentralizados, partiram para sistema de armazenamento permanentemente de dados semi-estruturados. O que permitiu a utilização desses dados para aplicação à algoritmos de aprendizagem de máquina com a finalidade de facilitar o processo de tomada de decisão dos gestores de saúde.

No entanto, muitos sistemas não possuem a arquitetura necessária para que os dados anonimizados sejam acessados por modelos de treinamentos de aprendizagem de máquina. Isso se deve ao fato de que a maioria dos sistemas foram desenvolvidas com arquiteturas monolíticas, sem a preocupação do consumo de grande volume de dados por agentes externos.

Mesmo com essa dificuldade, as organizações sabem da importância da ciência de dados para a área da saúde. Uma delas constitui o uso de modelos preditivos com o objetivo de prever o aumento da demanda de exames médicos, o que permite justificar o aumento relativo ao investimento de alguns setores da telemedicina. Desse modo, a utilização de métodos preditivos pode facilitar a criação de um plano de contingência, tanto de infraestrutura quanto de orçamento para realocar novos servidores para serviços da área da telemedicina.

Diante da falta de análise para o problema de previsão de demanda do Sistema Integrado Catarinense de Telemedicina e Telessaúde (STT/SC), o presente trabalho busca solucionar esse problema por meio do desenvolvimento de um sistema de predição integrado ao sistema de telemedicina STT/SC, em que o produto final será a visualização dos dados preditivos em componentes de visualização habitualmente conhecidas pelos administradores do sistema STT/SC. Dessa forma, os gestores de saúde pública poderão ter mais uma ferramenta ao seu dispor, para tomar decisões mais assertivas.

Parte dos requisitos para o desenvolvimento do sistema de previsão está na escolha dos horizontes de previsão para os modelos preditivos. Pois é preciso limitar o problema a um período definido. A partir de perguntas feitas aos gestores de saúde que usam o sistema STT/SC, delimitou-se o desenvolvimento a horizontes de predição de 1 mês e 12 meses. Os motivos da escolha desses horizontes estão na lista abaixo:

a) planejamento anual do aumento da demanda dos exames de teledermatologia

para um plano de contingência;

b) antecipação de, pelo menos, de 1 a 6 meses para o requerimento de infraestrutura para os equipamentos de saúde;

c) contratação de profissionais de saúde para a diminuição do congestionamento no atendimento; e

d) atualização mensal dos modelos preditivos.

1.3 MOTIVAÇÃO

Desenvolvido desde 2005 em parceria entre a Secretaria do Estado da Saúde (SES/SC) e a Universidade Federal de Santa Catarina (UFSC) (SAVARIS et al., 2017), o Sistema Integrado Catarinense de Telemedicina e Telessaúde (STT/SC) é uma aplicação web com ampla variedades de modalidades de exames, desde exames de alta complexidade à dermatologia e análise clínica, como também educação a profissionais da saúde e serviços de suporte. (BECKHAUSER, 2019).

O STT/SC, atualmente, disponibiliza vários serviços que são utilizados para o cumprimento das atividades realizadas pelos centros de saúde por todo o estado. Entre as funcionalidades disponibilizadas pelo sistema, houve crescente desenvolvimento de ferramentas de geoprocessamento para análise e visualização de informações espacias e temporais (SOUZA INÁCIO, 2016).

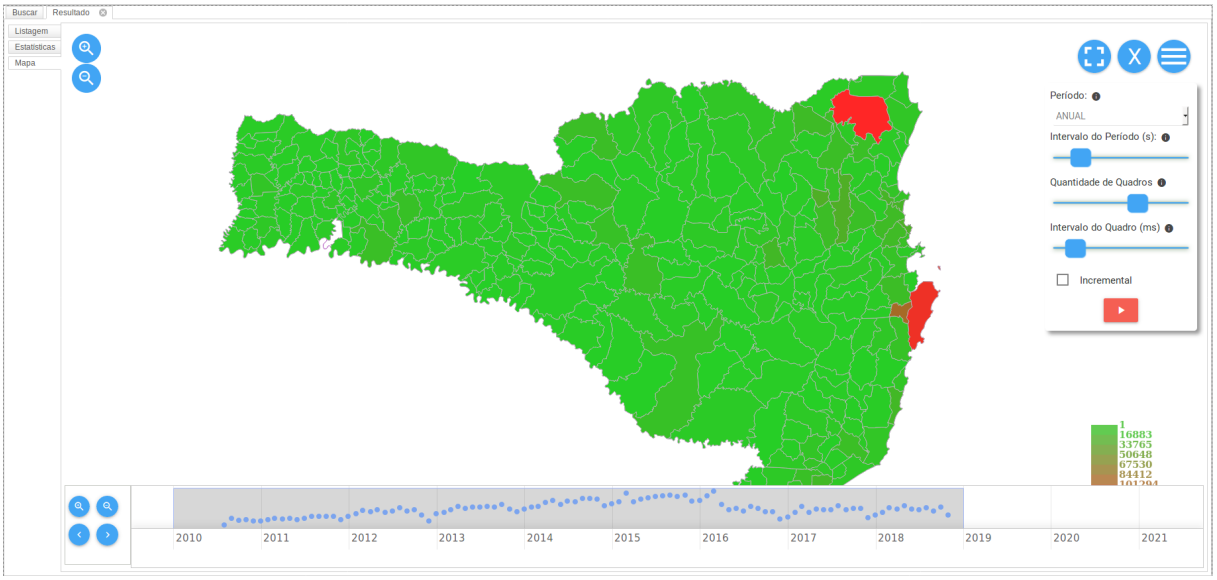
O relatório de janeiro de 2014 a junho de 2018, sobre o impacto nos custos de triagem dos pacientes centrados em teledermatologia no estado de Santa Catarina, observou um total de 83,100 exames teledermatológicos, o que representa significativa diminuição de custos por parte do setor de saúde (WANGENHEIM; NUNES, 2018). O que, mais uma vez, provou a importância das ferramentas de análise para o fortalecimento de tomada de decisões dos gestores de saúde.

A Figura 1.1 exibe informações de *quantidades* de exames dermatológicos entre os anos de 2010 a 2018. Na parte inferior do mapa, é possível observar uma linha do tempo que tem seu ponto mais alto em 2016. Por meio de métodos empíricos, reparamos que há um padrão na linha do tempo, que o padrão se repete durante os anos com pequenas variações, principalmente ao final de novembro. Portanto, a criação de uma ferramenta que possibilite validar esse argumento poderia ajudar os gestores no direcionamento de decisões sobre cenários futuros.

Ainda referente à Figura 1.1, há informação espacial (de latitude e longitude) para a visualização dos dados, que podem ajudar a delimitar o escopo. Esse fator deverá ser considerado para uma análise prévia do pré-processamento e previsão.

Atualmente, as áreas do conhecimento que possuem maior trabalho acadêmico e aplicações para previsibilidade de séries temporais estão direcionadas ao mercado financeiro. Assim, parte deste trabalho incorporará algumas dessas técnicas, como o modelo

Figura 1.1 – GISTelemed – análise temporal dos exames de teledermatologia de Santa Catarina



Fonte – elaborada pelo autor.

autorregressivo integrado de médias móveis (ARIMA) e a suavização exponencial.

1.4 OBJETIVOS

1.4.1 Objetivo geral

Desenvolver um sistema de previsão de demanda de exames de teledermatologia para o Sistema Integrado Catarinense de Telemedicina e Telessaúde (STT/SC).

1.4.2 Objetivos específicos

Os objetivos específicos deste trabalho são os seguintes:

- analisar o comportamento da série histórica da demanda de exames de teledermatologia;
- implementar modelos preditivos com abordagem clássica e abordagem de aprendizado;
- avaliar os resultados obtidos;
- integrar os modelos preditivos com a ferramenta de análise *Saiku* utilizado pelos gestores de saúde do sistema STT/SC.

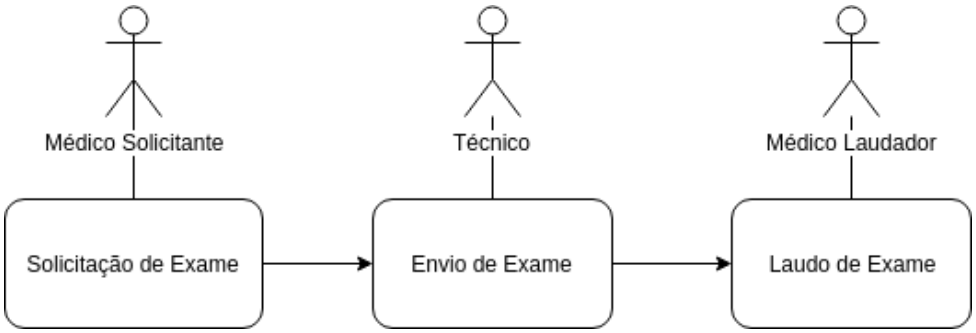
1.5 CENÁRIO DE APLICAÇÃO

Este trabalho possui um cenário de aplicação restrito, apenas, ao contexto de teledermatologia do Sistema Integrado Catarinense de Telemedicina e Telessaúde (STT/SC). Isto é, a **coleta de dados** é delimitado, apenas, para a demanda dos exames de teledermatologia. O processo de armazenamento dos exames de teledermato ocorrem por meio de três etapas: Solicitação de Exame, Envio de Exame (Imagens) e Laudo do Exame.

A primeira etapa constitui o processo no qual o médico solicita o registro de um exame dermatológico para um eventual laudo por outro médico. A segunda etapa é responsável por enviar imagens das lesões por meio de equipamentos médicos. Normalmente, essa etapa é realizado por técnicos. E, por último, um médico especializado finaliza remotamente o diagnostico emitindo laudo como resultado. As etapas podem ser observadas por meio do diagrama da Figura 1.2.

Nesse cenário, no qual os três profissionais de saúde estão envolvidos, o sistema STT/SC armazena informações temporais em cada etapa do processo, que servem co-

Figura 1.2 – Diagrama das etapas do processo de registro de um exame teledermatológico



Fonte – elaborada pelo autor.

mo base para o desenvolvimento dos modelos preditivos para este trabalho. Apesar de existirem três informações de séries temporais, a série temporal escolhida para avaliar a demanda dos exames de teledermatologia está na etapa de *Envio de Exame*.

Portanto, o cenário de aplicação para o desenvolvimento do sistema de demanda de exames de teledermatologia está na coleta de informações provenientes da segunda etapa do processo, no qual a demanda é contabilizada no momento em que um técnico utiliza os equipamentos para registrar as lesões do paciente e enviar as imagens para o sistema de telemedicina STT/SC.

1.6 METODOLOGIA

A metodologia que será aplicada para este trabalho envolve a realização de uma pesquisa exploratória dos seguimentos de *aprendizagem de máquina* e séries temporais voltados para a área da saúde especialmente para o contexto de Telemedicina e Telessaúde.

A pesquisa utiliza variáveis quantitativas da demanda de exames de teledermatologia no período entre 2015 e 2019 para o desenvolvimento dos modelos de predição. E, como metodologia para análise de desempenho dos modelos, foram utilizados as métricas: erro absoluto médio (MAE) e raiz do erro quadrático médio (RMSE).

1.6.1 Modelagem da predição

O processo de criação do modelo de predição ocorrerá de acordo com a metodologia de Montgomery, Jennings e Kulahci (2015), os passos estão enumerados da seguinte forma:

1. **Definição do problema:** determinar, claramente, o escopo de previsão.
2. **Conjunto de dados:** consiste em obter um volume de informações que incorpore informações temporais.
3. **Análise dos dados:** importante passo para seleção do modelo de previsão a ser utilizado. Gráficos de séries temporais dos dados devem ser construídos e inspecionados visualmente para interpretar padrões reconhecíveis. Possíveis *outliers* devem ser identificados e separados para possíveis estudos adicionais.
4. **Seleção dos modelos e ajuste dos parâmetros**: Consiste em escolher um ou mais modelos de previsão e ajustar os modelos aos dados. E também é a etapa de modificação dos parâmetros para diminuição dos erros.
5. **Validação dos modelos:** determina a sua validação perante os dados novos que não fazem parte do treinamento.
6. **Implementação do modelo de previsão:** envolve o uso do modelo pelo cliente, de fácil acesso e interpretação.
7. **Monitoramento do desempenho do modelo de previsão:** nessa etapa o modelo deve estar em constante adaptação, já que as condições mudam com o tempo, garantir

que essas condições não atrapalhem o bom desempenho do processo é o objetivo desta etapa.

1.6.2 Desenvolvimento dos modelos de predição

Os modelos de predição foram desenvolvidos na linguagem de programação *Python*[1], por meio do uso, principalmente, da biblioteca *Keras*[2] para modelar as redes neurais. E, para modelar as técnicas de abordagem paramétrica, utilizou-se a biblioteca *Statsmodels*[3].

1.6.3 Desenvolvimento do sistema de predição integrado com o STT/SC

Os modelos que trabalham com predição de séries temporais necessitam que seus dados estejam atualizados quase em tempo real. Para que, assim, os administradores do sistema possam visualizar o aumento da demanda por meio de gráficos e, consequentemente, tomarem boas decisões com base em dados recentes.

Em virtude da atualização mensal das predições, o desenvolvimento de um sistema que interage com o sistema de telemedicina STT/SC é primordial para o funcionamento da proposta, o que requer a escolha de uma série de tecnologias que permitem a montagem de infraestrutura que engloba: a extração, o treinamento, a predição, a geração e a visualização dos dados.

A escolha dessas ferramentas que auxiliam o desenvolvimento dessa infraestrutura propõe certa complexidade que interage em várias facetas do sistema desenvolvido, entre elas estão: a escolha de um banco de dados, execução de *scripts* de automatização e integração com *frameworks* de visualização.

Para este trabalho, os *scripts* de ETL (*Extract, Transform, Load*) foram desenvolvidas em *python*, e destacamos o uso do *Apache Druid*[4] para o armazenamento dos dados temporais.

A escolha do banco de dados *Apache Druid* se justifica pelo uso de uma variável tempo (*Timestamp*) como coluna prioritária no armazenamento dos seus registros, e

[1] https://www.python.org/

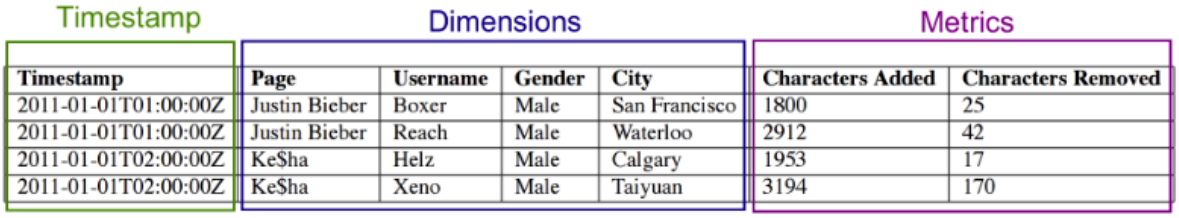
[2] https://keras.io/

[3] https://www.statsmodels.org

[4] https://druid.apache.org/

também pela sua divisão entre dados quantitativos (*metrics*) e dados qualitativos (*dimensions*), como pode ser observado na Figura 1.3

Figura 1.3 – Estrutura de dados do *Apache Druid*



Timestamp	Page	Username	Gender	City	Characters Added	Characters Removed
2011-01-01T01:00:00Z	Justin Bieber	Boxer	Male	San Francisco	1800	25
2011-01-01T01:00:00Z	Justin Bieber	Reach	Male	Waterloo	2912	42
2011-01-01T02:00:00Z	Ke$ha	Helz	Male	Calgary	1953	17
2011-01-01T02:00:00Z	Ke$ha	Xeno	Male	Taiyuan	3194	170

Fonte – *Apache Druid*.

Os dados foram plotados em três ferramentas de visualização. A primeira ferramenta chama-se *matplot* do ambiente *Jupyter*[5], utilizada, apenas, para analisar o desenvolvimento dos modelos preditivos. A segunda ferramenta, denominada de *Saiku*[6], é um *framework* de visualização de dados frequentemente utilizados por usuários do STT/SC. E a última ferramenta, denominada de *GisTelemed*[7], é um módulo do sistema de telemedicina STT/SC para análise de dados geográficos.

1.7 ESTRUTURA DO TRABALHO

A monografia está dividida em seis capítulos. No primeiro, apresenta-se breve contextualização e descrição para a proposta do trabalho, assim como os objetivos geral e específicos.

O segundo e o terceiro capítulos são destinados à fundamentação teórica, necessária para a implementação dos modelos de predição de séries temporais.

O quarto capítulo explica os procedimentos adotados para o desenvolvimento dos algoritmos para cada tipo de modelo preditivo com base na coleta de dados da quantidade de exames de teledermatologia do sistema de telemedicina STT/SC.

O quinto capítulo avalia os resultados obtidos de todos os modelos de acordo com as métricas RMSE e MAE.

[5] https://jupyter.org/

[6] https://github.com/OSBI/saiku

[7] http://site.telemedicina.ufsc.br/gistelemed/

O sexto capítulo apresenta os procedimento adotados para o desenvolvimento do sistema de demanda integrados com o STT/SC.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados os fundamentos de uma série temporal, as características básicas acerca da predição de séries temporais e uma visão geral dos modelos preditivos de séries temporais. Além disso, serão explicadas quais as métricas de desempenho para avaliar esses modelos.

2.1 SÉRIES TEMPORAIS

Uma série temporal é qualquer conjunto de observações ordenadas em intervalos regulares de tempo (dia, mês ou ano), em que cada observação é denotado por y_t, com $t = 1$ sendo a primeira observação e $t = T$ sendo a última observação (DOUGLAS C. MONTGOMERY CHERYL L. JENNINGS, 2015). O conjunto de tempos $t = 1, 2, \ldots, T$ é referido como período de observação dos dados, e o valor de T é referente ao tamanho do conjunto.

Se os valores de uma série temporal puderem ser sintetizados por meio de uma função matemática $y = f(t)$, diz-se que a série é determinística. Em contrapartida, quando a série possuir um termo aleatório ϵ, $y = f(t, \epsilon)$, a série é definida como estocástica ou não determinística (MORETTIN; CASTRO TOLOI, 2018).

As séries temporais possuem algumas características importantes na análise do seu comportamento. Entre elas, podemos citar a estacionariedade e as componentes de uma série temporal. A estacionariedade é importante para o desenvolvimento de modelos de predições estocásticos.

E as componentes de uma série temporal são importantes para isolar o problema em componentes sazonais e de tendência, isolando, assim, as componentes determinísticas de um modelo estocástico. Desse modo, os modelos podem ser aplicados por meio de métodos paramétricos, como o método de suavização exponencial. Nas subseções seguintes, serão abordadas as componentes e a estacionariedade de uma série temporal.

2.1.1 Estacionariedade

Uma série temporal é estacionária quando suas características estatísticas (média e variância) são constantes ao longo do tempo. Segundo (SILVEIRA BUENO, 2011), uma série temporal $\{y_t, t \in \mathbb{Z}\}, \mathbb{Z} = \{0, \pm1, \pm2, \ldots\}$ é *fracamente estacionário* se:

a) $E\left|y_t\right|^2 < \infty$;

b) $E\left(y_t\right) = \mu$, para todo $t \in \mathbb{Z}$; e

c) $E\left(y_t - \mu\right)\left(y_{t-j} - \mu\right) = \gamma_j$

A condição **a** afirma que o segundo momento não centrado deve ser *finito*. E as condições **b** e **c** asseguram, respectivamente, que a média e a variância são constantes para todo período da série, mesmo que a distribuição da variável aleatória se vá alterando ao longo do tempo, em que a variância apenas depende da distância temporal entre as observações.

Outro conceito importante para a série temporal é a *estacionariedade estrita*. Uma série temporal é estritamente estacionária se a função distribuição conjunta de $\{y_{t_i}\}_{i=1}^{T}$ for igual à função de distribuição conjunta de $\{y_{t+h})_{i=1}^{T}$, $h \in \mathbb{Z}$, isto é,

$$f\left(y_{t_1}, y_{t_2}, \ldots, y_{t_T}\right) = f\left(y_{t_1+h}, y_{t_2+h}, \ldots, y_{t_T+h}\right)$$

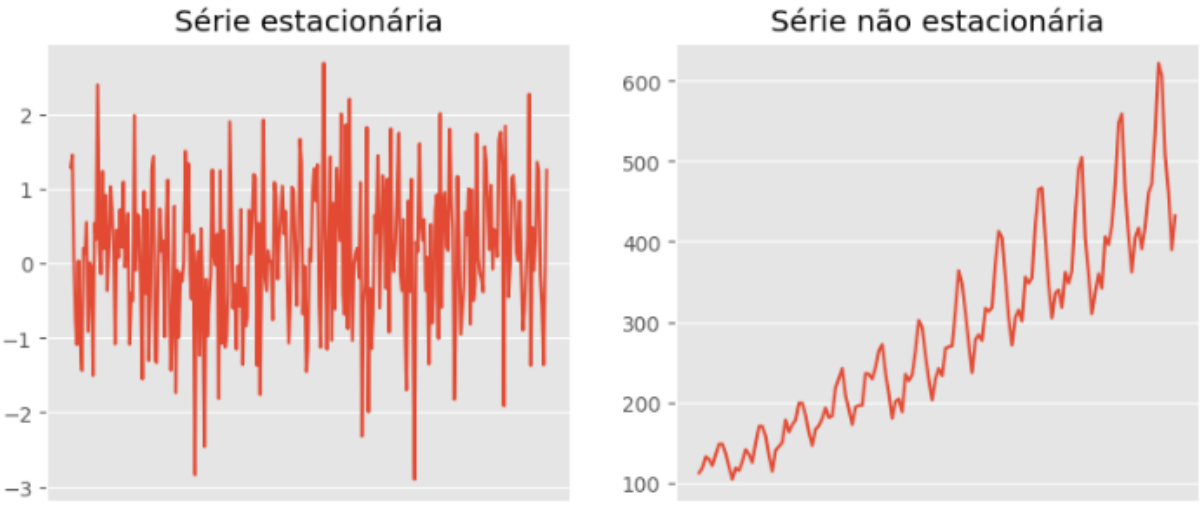
h consiste no horizonte de previsão, e T constitui o total de amostras da série. Portanto, a estacionariedade estrita implica que a função distribuição da série em quaisquer dois intervalos de tempo de igual tamanho exibirá propriedades estatísticas similares. Na literatura, geralmente, escionariedade significa estacionariedade fraca.

A Figura 2.1 apresenta a diferença entre uma série estacionária e uma série não estacionária. Ao lado esquerda da figura, há ruído branco que apresenta as propriedades de média e variância constantes que são características da estacionariedade. Ao lado direito, há uma série que apresenta componentes de tendência e sazonalidade; estes, certamente, não apresentam propriedades de uma série estacionária.

2.1.2 Componentes de séries temporais

As séries temporais podem ser decompostas em um conjunto finito de elementos independentes com a finalidade de entender o comportamento dos eventos de uma série histórica. Entre os componentes, estão: tendência (T), sazonalidade (S) e ruído (N). A tendência e a sazonalidade podem ser sintetizadas de forma determinísticas, o que não ocorre com a componente aleatória (ruído) (DOUGLAS C. MONTGOMERY CHERYL L. JENNINGS, 2015). No entanto, isolar a componente aleatória é importante para modelar a distribuição de probabilidade da série, que é muito útil para o processo estocástico.

Figura 2.1 – Comparação entre série estacionária e série não estacionária



Fonte – elaborada pelo autor.

Em relação à operação matemática entre as componentes, a decomposição se divide em abordagem aditiva e multiplicativa. Na abordagem aditiva, os componentes são somados como apresentado na equação 2.1. Na abordagem multiplicativa, os componentes são multiplicados para formar a série original, como observado na equação 2.2.

$$Z_t = T_t + S_t + N_t \tag{2.1}$$

$$Z_t = T_t \times S_t \times N_t \tag{2.2}$$

Na decomposição aditiva, os componentes possuem a mesma unidade de medida da variável investigada. No entanto, na decomposição multiplicativo, apenas a tendência possui a mesma unidade da variável investigada e o restante geralmente possuem grandezas de magnitude adimensional.

O objetivo de decompor a série temporal é a extração dos componentes determinísticos (T e S) com a esperança de que o componente residual ou de ruído acabe sendo uma série temporal estacionária. Podemos, então, usar a teoria da estacionariedade para encontrar um modelo probabilístico satisfatório. E, portanto, analisar suas propriedades e usá-lo em conjunto com os outros componentes para fins de previsão e simulação (PETER J. BROCKWELL, 2016).

2.2 PREDIÇÃO DE SÉRIES TEMPORAIS

Sabendo que o conjunto de dados tem tamanho T, as predições de séries temporais pretendem encontrar valores desconhecidos nos tempos $T+1, T+2, \ldots, T+h$, em que o cálculo de previsões futuras é delimitado por um *horizonte de previsão* h.

Geralmente, precisamos distinguir entre *erros de previsão* (eq. 2.4) e *resíduos* (eq. 2.3). Apesar de ambos serem erros, conceitualmente, são calculados em etapas diferentes.

$$e_t = y_t - \hat{y}_t, \qquad t = 1, 2, \ldots, T \tag{2.3}$$

$$e_{T+j} = y_{T+j} - \hat{y}_{T+j}, \qquad j = 1, 2, \ldots, h \tag{2.4}$$

Os resíduos são calculados no conjunto de treinamento, enquanto os erros de previsão são calculados no conjunto de testes. Para o resultado do modelo, é mais importante o cálculo do erro de previsão para avaliar os melhores modelos. Já que o modelo treinado não deve conhecer os dados do conjunto de testes.

Visto que os erros de previsão retornam uma lista para horizontes de predição maiores que um ($h > 1$), se faz necessário calcular o erro médio das predições. Para esse fim, utilizam-se, normalmente, as métricas *Raiz do Erro Quadrático Médio* (RMSE) e *Erro médio absoluto* (MAE), ambos representados nas equações 2.5 e 2.6.

$$\text{RMSE} = \sqrt{\frac{1}{h}\sum_{j=1}^{h}(e_{T+j})^2} \tag{2.5}$$

$$\text{MAE} = \frac{1}{h}\sum_{j=1}^{h}|e_{T+j}| \tag{2.6}$$

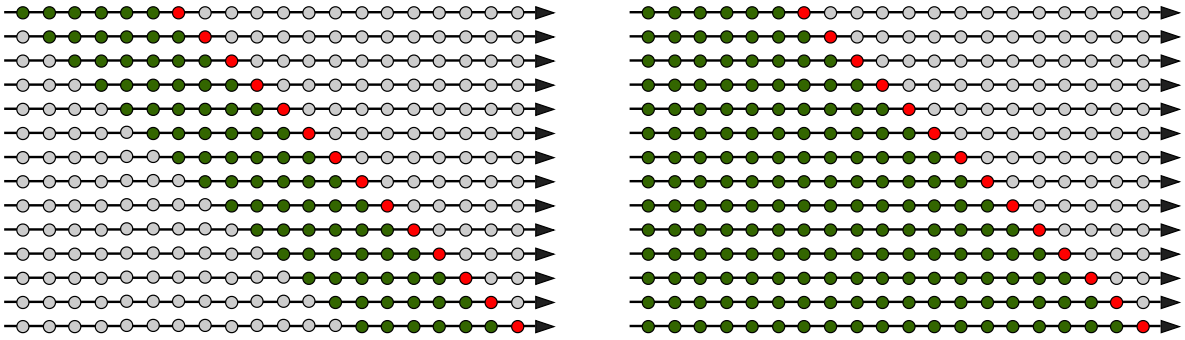
A métrica RMSE é uma medida análoga ao desvio padrão, penaliza valores mais afastados do valor médio. E a métrica MAE mede a magnitude média dos erros em valores absolutos, em que todas as amostras possuem um peso individual igual.

De acordo com Nielsen (2019), não é aconselhável utilizar o processo padrão de validação cruzada para avaliar o desempenho dos modelos preditivos de séries temporais. Pois não se podem selecionar amostras aleatórias de uma série temporal devido a sua

Figura 2.2 – Estratégias para avaliar o modelo preditivo

(a) Janela deslizante

(b) Janela em expansão



Fonte – adaptado de Nielsen (2019).

natureza sequencial. Dessa forma, duas estratégias podem ser adotadas: janela deslizante ou janela em expansão.

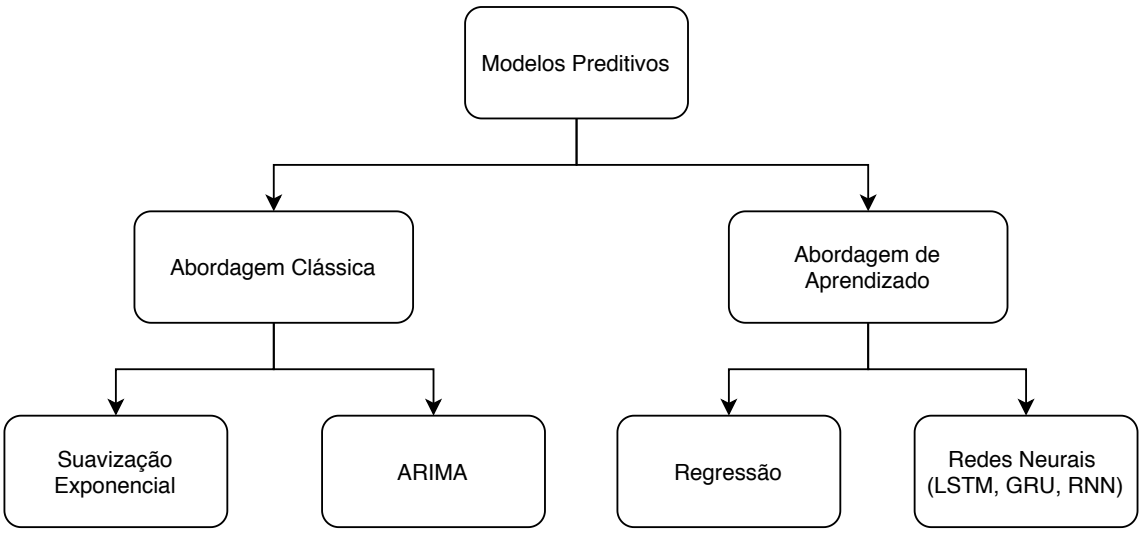
A primeira estratégia chamada de *janela deslizante* (Figura 2.2a) separa o conjunto de treinamento em tamanhos iguais, em que, à medida que o erro de predição for calculado, a amostra mais distante será jogada fora. A segunda estratégia envolve uma *janela em expansão* (Figura 2.2b), no qual as amostras de treinamento vão se expandindo à medida que um erro de predição for calculado.

2.3 ABORDAGEM DOS MODELOS

Os métodos estatísticos se distinguem por conhecer *a priori* a natureza da distribuição de probabilidade dos dados. Em alguns casos, deve-se tomar transformações ou retirar *outliers* das amostras, de forma que se tenha uma distribuição de probabilidade conhecida (GREGORY W. CORDER, 2009).

Diante dessa premissa, podemos dividir os métodos de previsão em uma hierarquia com abordagens clássica e de aprendizado, consoante Figura 2.3. A abordagem clássica pertencente aos métodos de Suavização Exponencial e ARIMA, estes utilizam da estatística paramétrica para transformar seu conjunto de dados numa distribuição de probabilidade conhecida, e, portanto, necessitam conhecer o comportamento da série temporal.

Figura 2.3 – Hierarquia dos modelos preditivos



Fonte – adaptada de Parmezan e Batista (2016).

Por outro lado, a abordagem de aprendizado pertencentes às redes neurais (LSTM, GRU e RNN), em oposição aos modelos clássicos, não dependem do conhecimento prévio das propriedades da série temporal. Esses métodos são mais simples de serem ajustados e demostram considerável desempenho mesmo quando aplicados às séries complexas e altamente não lineares (PARMEZAN; BATISTA, 2016). Portanto, podemos comparar os métodos de aprendizagem de máquina a um modelo de caixa-preta que, em alguns casos,

gera bons resultados preditivos.

3 MODELOS DE PREDIÇÃO DE SÉRIES TEMPORAIS

Neste capítulo será apresentada a base teórica para a modelagem de predição de séries temporais. Apesar de essas séries terem características em comum, são modelados de forma diferentes, com parâmetros diferentes. Na primeira parte, será explicado sobre as técnicas de suavização exponencial e autorregressivo integrado de médias móveis (ARIMA), que estão dentro do contexto de modelos clássicos. E, na segunda parte, serão apresentadas as técnicas de aprendizagem de máquinas com foco em redes neurais.

3.1 SUAVIZAÇÃO EXPONENCIAL

A suavização exponencial é uma das principais classes de modelos usados em previsão de séries temporais. A grande popularidade atribuída aos métodos de suavização deve-se à simplicidade, à eficiência computacional e à sua razoável precisão (SILVEIRA BUENO, 2011; MORETTIN; CASTRO TOLOI, 2018). Serão abordados 4 modelos diferentes. A escolha do modelo dependerá das componentes (nível, tendência e sazonalidade) da séria temporal analisada.

3.1.1 Médias Móveis Simples (MMS)

O modelo de médias móveis simples (MMS) é técnica de predição que consiste em calcular a média aritmética das r observações mais recentes. A ideia é calcular uma estimativa M_T do nível médio que não considera as observações mais antigas. A estimativa da previsão denotada de M_T pode ser calculada da seguinte forma:

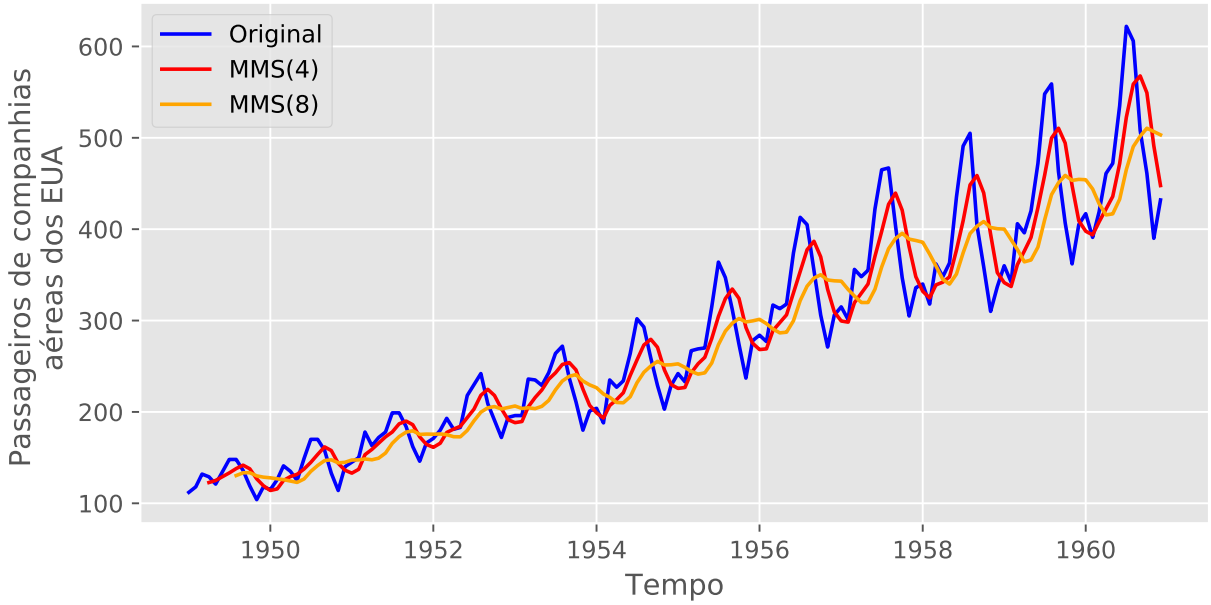
$$\hat{y}_{T+1} = M_T = \frac{y_T + y_{T-1} + \cdots + y_{T-r+1}}{r} = \frac{1}{r} \sum_{t=T-r+1}^{r} y_t \tag{3.1}$$

em que T representa a quantidade total de amostras de treinamento. E r é a quantidade de amostras utilizados para medir a próxima amostra.

Na Figura 3.1, há exemplo do MMS com os parâmetros $r = 4$ e $r = 8$, ambos aplicados a uma série temporal[1] que apresenta as componentes de tendência e sazonalidade.

[1] O conjunto de dados fornece totais mensais de passageiros de uma companhia aérea dos EUA de 1949 a 1960. Esse conjunto de dados foi extraído de `https://www.kaggle.com/chirag19/air-passengers`.

Figura 3.1 – Exemplo do modelo Médias Móveis Simples (MMS) para $r = 4$ e $r = 8$



Fonte – elaborada pelo autor.

3.1.2 Suavização Exponencial Simples (SES)

O método de Suavização Exponencial Simples (SES) calcula a média ponderada de todos os valores anteriores. A ponderação das amostras é calculada com base na atribuição de diferentes pesos que decaem exponencialmente com o tempo. De forma que as amostras mais próximas do valor-alvo possuem maior peso.

Portanto, o método SES utiliza, apenas, o parâmetro A que está limitado entre os valores de 0 a 1 ($0 \leq A \leq 1$). O parâmetro A determina o decaimento exponencial da média ponderada, que é representada pela equação:

$$\hat{y}_{t+1} = \bar{Z}_t = AZ_t + A(1-A)^1 Z_{t-1} + \cdots + A(1-A)^T Z_{t-T}$$

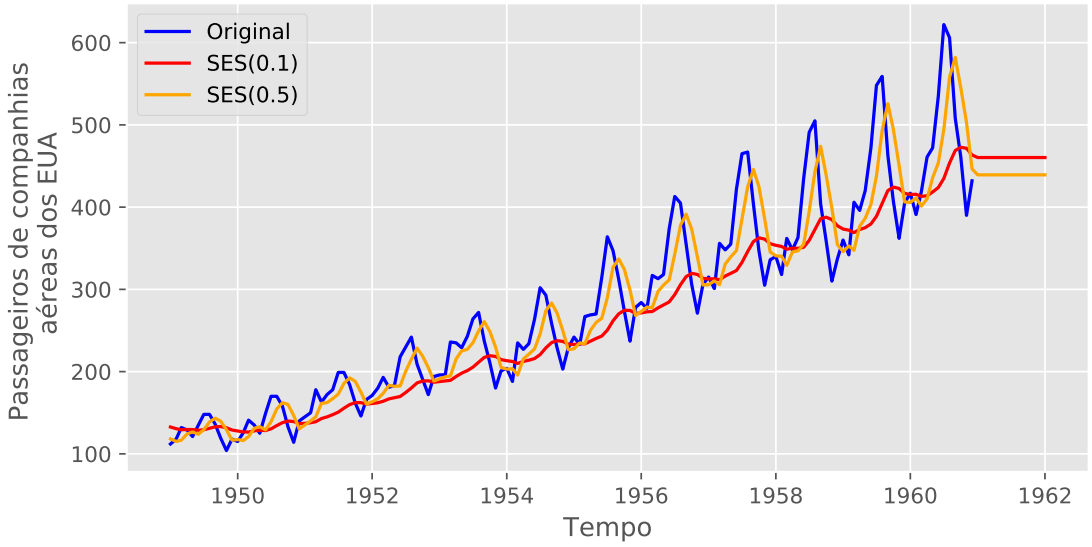
Em que A é chamada de constante de suavização, T define o tamanho de amostras de treinamento e $\bar{Z}_t$ é denominado de valor exponencialmente suavizado.

Ambos os modelos SES (Figura 3.2) e MMS (Figura 3.1) são utilizados para determinar séries localmente constantes. Pois a proposta do modelo é a representação de uma constante mais um ruído aleatório, isto é,

$$Z_t = \mu_t + N_t$$

em que o valor esperado de $E(N_t) = 0$ e $\text{Var}(N_t) = \sigma_N^2$.

Figura 3.2 – Exemplo do modelo SES para $A = 0.1$ e $A = 0.5$



Fonte – elaborada pelo autor.

3.1.3 Suavização Exponencial de Holt (SEH)

A Suavização Exponencial de Holt (SEH) é um modelo aplicado a uma série que apresenta tendência linear positiva (ou negativa). Esse método é similar à SES, a diferença é que, em vez de suavizar, apenas, o nível, o método utiliza nova constante de suavização para modelar sua **tendência**.

Os valores do nível e da tendência da série, no instante t, são estimados por,

$$\hat{T}_t = B\left(\bar{Z}_t - \bar{Z}_{t-1}\right) + (1-B)\hat{T}_{t-1}, \quad 0 < B < 1 \quad \text{e } t = 2, \ldots, T$$

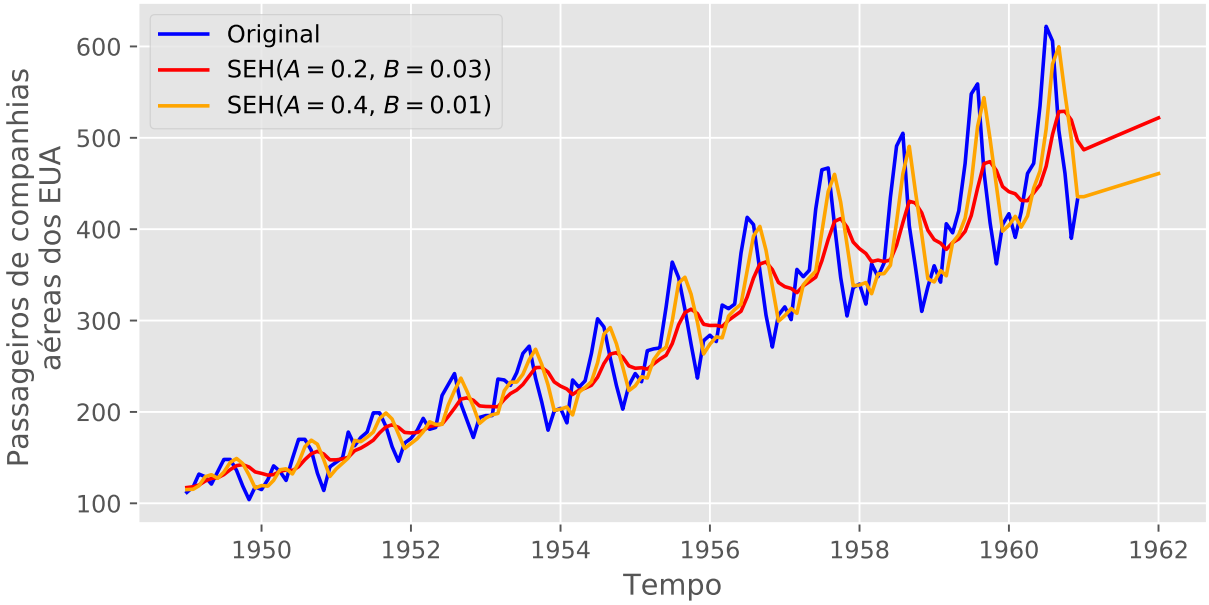
$$\bar{Z}_t = AZ_t + (1-A)\left(\bar{Z}_{t-1} + \hat{T}_{t-1}\right), \quad 0 < A < 1 \quad \text{e } t = 2, \ldots, T$$

$$\hat{y}_{t+h} = \bar{Z}_t + h\hat{T}_t, \quad \hat{T}_1 = y_2 - y_1 \quad \text{e } \bar{Z}_1 = y_1$$

em que, A e B são denominadas constantes de suavização, que representam, respectivamente, a suavização do valor do nível e do valor da tendência.

É presumível que os modelos MMS, SES e SEH possam prever, apenas, o próximo intervalo de tempo. Pois, para horizontes de previsão maiores do que uma unidade de tempo, esses modelos convergem para uma reta.

Figura 3.3 – Exemplo do modelo SEH



Fonte – elaborada pelo autor.

3.1.4 Suavização Exponencial de Holt-Winters (HW)

O modelo de suavização exponencial de Holt-Winters (WH) é o mais completo entre os modelos de suavização exponencial. Além de representar os componentes de nível e de tendência, também representa a componente de sazonalidade. O que permite reproduzir qualquer série que tenha como padrão as componentes de tendência e sazonalidade.

Em virtude de possuir duas componentes fundamentais (tendência e sazonalidade), o modelo pode mesclar a operação matemática dos componentes. Por exemplo, o modelo pode ter uma abordagem de série sazonal multiplicativa com uma série de tendência aditiva. O que permite aumentar sua complexidade.

Sua parametrização depende, apenas, de três variáveis, entre elas estão: o parâmetro A que representa a suavização em relação ao nível; o parâmetro B, em relação à tendência e o parâmetro C, em relação à sazonalidade.

Supondo uma série sazonal aditiva, as estimativas do fator sazonal, do nível e da tendência da série são dadas por

$$\hat{S}_t = C\left(Z_t - \bar{Z}_t\right) + (1 - C)\hat{S}_{t-s}, \quad 0 < C < 1$$

$$\bar{Z}_t = A\left(Z_t - \hat{S}_{t-s}\right) + (1 - A)\left(\bar{Z}_{t-1} + \hat{T}_{t-1}\right), \quad 0 < A < 1$$

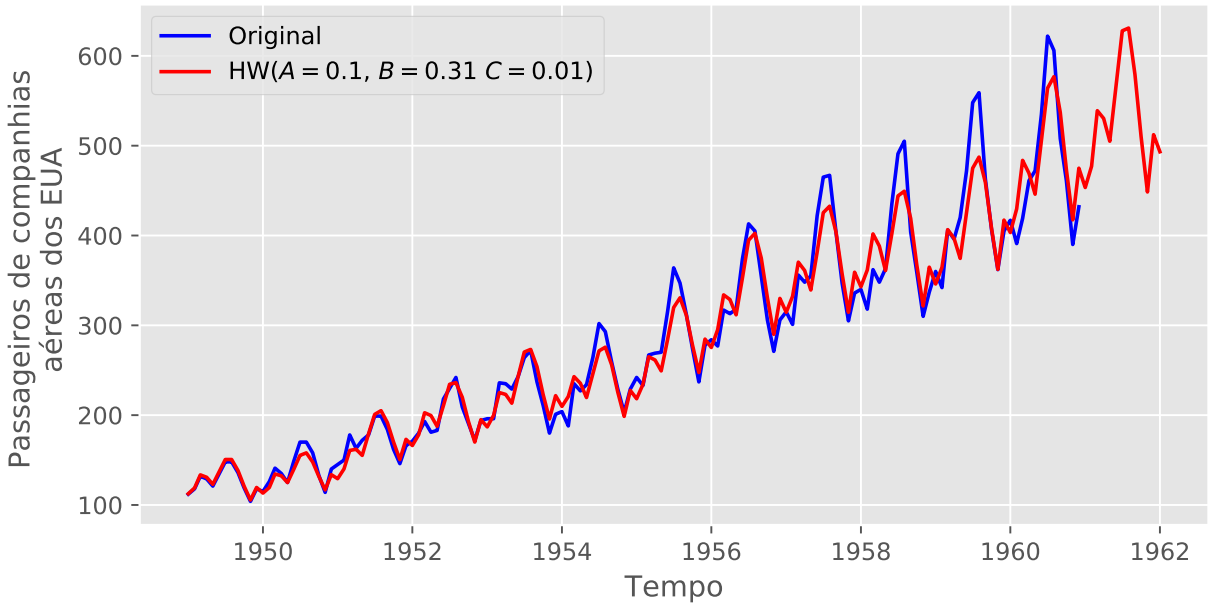
$$\hat{T}_t = C\left(\bar{Z}_t - \bar{Z}_{t-1}\right) + (1-C)\hat{T}_{t-1}, \quad 0 < B < 1$$

$$\hat{y}_{t+h} = \bar{Z}_t + h\hat{T}_t + \hat{S}_{t-s+h}, \quad h = 1, 2, \ldots, s$$

em que A, B e C são as constantes de suavização e s é a periodicidade da sazonalidade.

A Figura 3.4 apresenta a predição de uma série histórica que contém os componentes de tendência e sazonalidade bem definidos. Em vermelho podemos observar que o modelo HW consegue representar sua projeção para o futuro.

Figura 3.4 – Exemplo do modelo suavização exponencial de Holt-Winters (HW) para $A = 0.1$, $B = 0.31$ e $C = 0.01$.



Fonte – elaborada pelo autor.

3.1.5 Resumo dos Modelos

Na Tabela 3.1, apresentamos resumo dos modelos a serem escolhidos em relação a suas componentes de nível, tendência e sazonalidade de uma série temporal.

Tabela 3.1 – Resumo dos modelos Suavização Exponencial.

Modelo	Formato da Série	Hipótese das Componentes	Parâmetros
MMS	Constante	$Z_t = \mu_t + N_t$	r
SES	Constante	$Z_t = \mu_t + N_t$	$0 < A < 1$
SEH	Tendência	$Z_t = \mu_t + T_t + N_t$	$0 < A, B < 1$
HW	Sazonal	$Z_t = \mu_t + T_t + S_t + N_t$	$0 < A, B, C < 1$ s

Fonte – elaborada pelo autor.

3.2 ARIMA

Os modelos autorregressivos, integrados de médias móveis (ARIMA) de ordem (p, d, q), resultam da combinação de três procedimentos estatísticos (BOX et al., 2015): 1 - Autorregressão AR(p); 2 – Ordem de Diferenciação d e 3 – Médias Móveis MA(q).

O modelo Autorregressivo (AR) de ordem p é representado pela equação 3.2, em que Z_t corresponde à observação da série no instante t.

$$Z_t = \delta + \sum_{i=1}^{p} \phi_i Z_{t-i} + e_t \tag{3.2}$$

p denota o número de observações consideradas; ϕ_i indica o i-ésimo coeficiente autorregressivo; δ representa o nível inicial do modelo (constante); e_t compreende o ruído branco em uma distribuição com média zero e variância constante σ_e^2 devido a suas propriedades de estacionariedade.

Da mesma forma que o modelo AR, o modelo de Médias Móveis de ordem q, *MA(q)*, estabelece o valor atual da série por meio de uma combinação linear de valores anteriores. Porém, os pesos $\theta_1, \theta_2, \ldots, \theta_q$ multiplicam os ruídos $e_{t-1}, e_{t-2}, \ldots, e_{t-q}$, conforme 3.3.

$$Z_t = \mu + \sum_{i=1}^{q} \theta_i e_{t-i} + e_t \tag{3.3}$$

E, por fim, a ordem de diferenciação d permite que uma série não estacionária se transforme em uma série estacionária. Para isso, basta aplicar d diferenças na série original. Normalmente, a primeira diferença $\Delta Z_t = Z_t - Z_{t-1}$ é suficiente para torná-la estacionária. De modo geral, a d-ésima diferença de Z_t é:

$$\Delta^d Z_t = \Delta \left[\Delta^{d-1} Z_t \right]$$

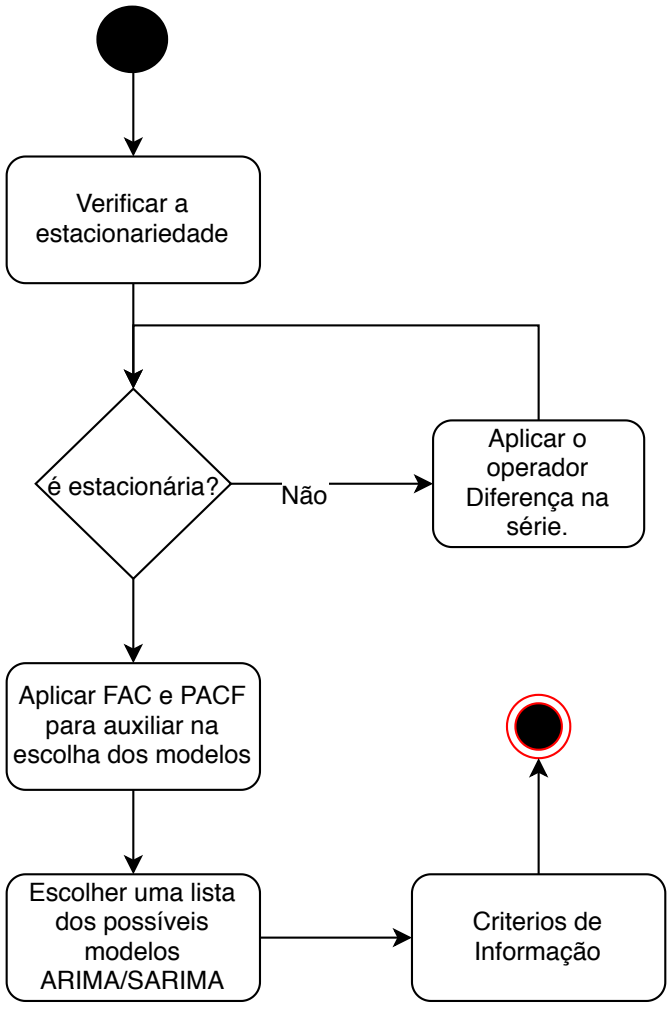
A combinação dos modelos AR, MA e a aplicação das diferenças resultam no modelo ARIMA de ordem (p, d, q) demonstrado na equação 3.4.

$$\Delta^d Z_t = \delta + \sum_{i=1}^{p} \phi_i \Delta^d Z_{t-i} + \sum_{i=1}^{q} \theta_i e_{t-i} + e_t \tag{3.4}$$

Uma variação do modelo ARIMA são os modelos sazonais, também chamados de Autorregressivo Integrado de Médias Móveis Sazonal (SARIMA(p,d,q)(P,D,Q)$_m$). Esse modelo permite separar os parâmetros componente sazonal da série original, para modelar com outros parâmetros (P, D, Q).

A escolha dos melhores parâmetros do modelo ARIMA ou SARIMA depende de um processo interativo que pode ser observado na Figura 3.5. A primeira etapa verifica se a série é estacionária, caso não seja, será aplicado o operador diferença na série original. Na segunda etapa, serão aplicadas as funções de autocorreção **FAC** e **PACF** para auxiliarem a escolha de modelos com distintos parâmetros. E, por fim, serão aplicados os Critério de Informação de Akaike (AIC) ou Critério de Informação Bayesiano (BIC) para determinar o melhor modelo.

Figura 3.5 – Fluxograma para determinação dos parâmetros ARIMA ou SARIMA



Fonte – elaborada pelo autor.

3.2.1 Autocovariância e autocorrelação

A autocovariância pode ser obtida por meio da covariância das variáveis aleatórias y_t, y_{t+k} que são separadas pelo intervalo k. O intervalo k é chamada de atraso (*lag*).

$$\gamma_k = \text{Cov}\left(y_t, y_{t+k}\right) = E\left[\left(y_t - \mu\right)\left(y_{t+k} - \mu\right)\right] \tag{3.5}$$

O valor do coeficiente $\gamma_k, k = 0, 1, 2, \ldots$ é chamado de função de autocovariância. A autocovariância para $k = 0$ é a própria variância da série temporal. Então, o valor de $\gamma_0 = \sigma_y^2$ será uma constante para uma série estacionária. Dessa forma, o coeficiente de autocorrelação ρ_k para um atraso **k** pode ser obtido por meio da razão entre a covariância e a variância demostrada na equação abaixo:

$$\rho_k = \frac{E\left[\left(y_t - \mu\right)\left(y_{t+k} - \mu\right)\right]}{\sqrt{E\left[\left(y_t - \mu\right)^2\right] E\left[\left(y_{t+k} - \mu\right)^2\right]}} = \frac{\text{Cov}\left(y_t, y_{t+k}\right)}{\text{Var}\left(y_t\right)} = \frac{\gamma_k}{\gamma_0} \tag{3.6}$$

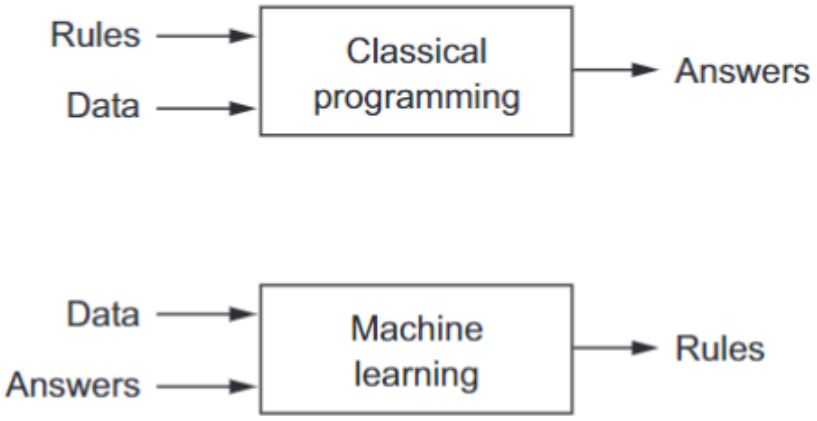
O conjunto de valores de $\rho_k, k = 0, 1, 2, \ldots$ é chamado de **função de autocorrelação** (FAC). De acordo com Montgomery, Jennings e Kulahci (2015), uma boa estimativa necessita de, pelo menos, 50 observações para uma FAC, e poderia ser calculado até o atraso 1/4 do total de observações.

3.3 APRENDIZADO DE MÁQUINA (*MACHINE LEARNING*)

Aprendizado de máquina é um ramo da inteligência artificial que permite que o computador aprenda, a partir de dados e do tratamento estatístico da incerteza, com enforque nas experiências de um grande volume de dados. A estratégia de indução é a mais usada para o contexto de aprendizado de máquina, pois permite obter conclusões com base em um conjunto de exemplos conhecidos. Porém, deve-se ter prudência, porque, se o número de exemplos for insuficiente, as hipóteses obtidas podem ser de pouco valor.

Diferentemente da inteligência artificial simbólica clássica (que se baseia em regras lógicas bem definidas), em aprendizado de máquina, as regras lógicas (programa) são retiradas por meio de conjuntos de dados rotulados (entrada e saída). A partir dessas regras, o programa produzirá novas respostas (saída) para novos dados de entrada.

Figura 3.6 – Machine Learning – novo paradigma de programação.



Fonte – (CHOLLET, 2018).

Portanto, o grau de supervisão (rótulos) presente nos dados é uma parte importante para o treinamento da máquina. Desse modo, dependendo do grau de dados rotulados, o aprendizado de máquina pode ser categorizado de acordo com a seguinte hierarquia: aprendizado não supervisionado, aprendizado semissupervisionado e aprendizado supervisionado.

a) **Aprendizado não supervisionado**: para essa categoria, nenhuma informação possui rótulos, apenas atributos. Seus algoritmos são focados em determinar classes (rótulos), como a partir do agrupamento dos dados de entrada.

b) **Aprendizado semissupervisionado**: apenas uma pequena parte do conjunto de dados estão rotulados, e, portanto, essa pequena parcela auxiliará a determinação dos rótulos no restante dos dados.

c) **Aprendizado supervisionado**: por fim, para essa categoria, todos os dados estão rotulados. Matematicamente, deve-se encontrar uma função hipótese $f : \mathcal{X} \to \mathcal{Y}$ através do conjunto de dados $\mathcal{D}$, que, ao aplicar uma entrada $\mathbf{X}^{(i)}$, uma nova saída rotulada $y^{(i)} = f\left(\mathbf{x}^{(i)}\right)$ é criada.

$$\mathcal{D} = \left\{\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(m)}, y^{(m)}\right)\right\} \tag{3.7}$$

Podem ser aplicados para resolver problemas de classificação (variável de saída discreta) ou regressão (variável de saída contínua).

3.4 REGRESSÃO

A regressão linear é o primeiro passo para entender o funcionamento das redes neurais artificiais, que fazem parte das técnicas de aprendizado supervisionado. O objetivo da regressão é determinar uma função hipótese $f : \mathbb{R}^n \to \mathbb{R}$ a partir de um conjunto de observações de entrada e saída $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$, em que m é o tamanho do conjunto.

Os dados de entrada podem ser generalizados para entradas **n-dimensional**, ou seja, cada elemento pertencente ao vetor $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ também serão vetores, porém de tamanho n, como observado no elemento $\mathbf{x}_m$ da equação a seguir,

$$\mathbf{x}_m = \begin{bmatrix} x_{1,m} \\ x_{2,m} \\ \vdots \\ x_{n,m} \end{bmatrix}$$

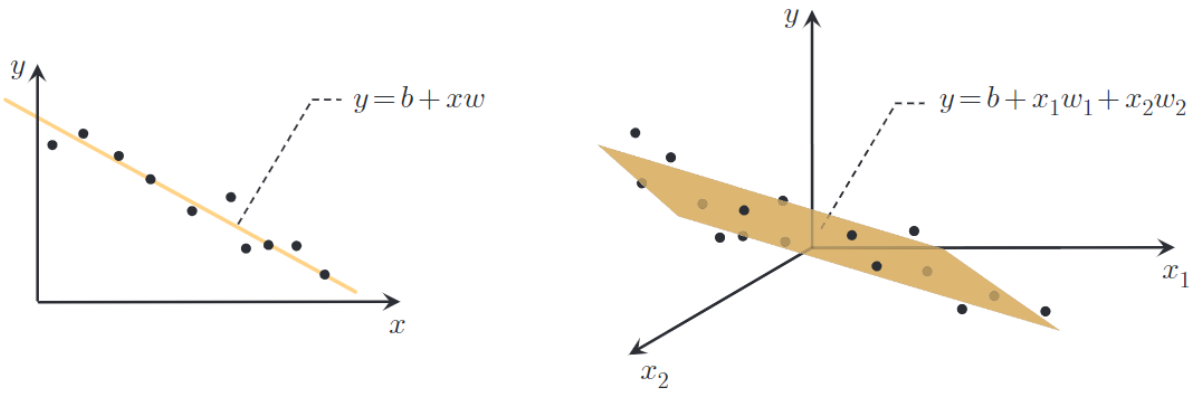
Nesse caso, o resultado será uma matriz de $M^{m \times n}$, e seu ajuste será de um hiperplano.

A função hipótese do modelo pode ser representada pela seguinte equação,

$$\hat{y} = f(\mathbf{x}) = b + w_1 x_1 + \cdots + w_n x_n = \mathbf{w}^T \mathbf{x}$$

em que os parâmetros *pesos* w e *bies* b são responsáveis por aproximar os rótulos por meio de uma linha (para uma dimensão) ou de um hiperplano (para n dimensões), como pode ser visto na Figura 3.7.

Figura 3.7 – Representação do modelo de regressão linear



Fonte – (JEREMY WATT REZA BORHANI, 2016).

A regressão também pode representar outras classes de funções. A regressão polinomial é um exemplo típico da representação de modelos mais complexos. Para isso, será preciso aplicar uma transformação nos valores de entradas por meio de uma função, como demostrado na equação abaixo,

$$\hat{y} = f(\mathbf{x}) = b + \sum_{j=1}^{n-1} w_j \phi_j(\mathbf{x}) = \sum_{j=0}^{n-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

E, portanto, podendo obter uma classe de funções muito mais útil tomando combinações lineares de um conjunto fixo de funções não lineares das variáveis de entrada, conhecidas como **funções de base** $\phi_j(\mathbf{x})$ (BISHOP, 2011).

A fim de criar um modelo a partir do conjunto de observações, é preciso procurar valores para o coeficiente w (peso) e coeficiente b (*bies*) que representem todos os valores de y com o mínimo de error possível. Precisando, assim, minimizar o erro por meio de uma função $J(\mathbf{w})$ chamada de função custo. A criação dessa função custo será apresentada na próxima subseção.

3.4.1 Função custo

Para avaliar o desempenho de um modelo preditivo entre infinitas funções de hipóteses, necessitamos representar, por meio de uma função $J(w)$ (chamada *função custo*), qual o melhor modelo para o conjunto de dados de treinamento. Está função é determinada a partir de uma métrica de erro (chamada de *função perda*) que possa avaliar a diferença entre a predição $\hat{y}$ e o valor-alvo correto y.

Entre as *funções perdas* mais conhecidas, representada por $L(\hat{y}, y)$, temos:

a) erro absoluto (MAE): $L(\hat{y}, y) = |\hat{y} - y|$;

b) erro quadrático (MSE): $L(\hat{y}, y) = (\hat{y} - y)^2$ ou $L(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$.

O erro quadrático é mais adequado, pois é diferenciável em todos os pontos. Em contrapartida, é mais sensível a *outliers*.

Supondo que a escolha for o erro quadrático, temos a seguinte equação da função custo:

$$J(\mathbf{w}) = \frac{1}{2m} \sum_{i=1}^{m} \left(\mathbf{w}^T \mathbf{x}^{(i)} - y^{(i)} \right)^2 \tag{3.8}$$

em que, aplicando a derivada parcial em relação ao peso e a igualando a zero ($\frac{\partial J(\mathbf{w})}{\partial w_j} = 0$), podemos encontrar os pontos mínimos e máximos da função.

Logo, para minimizar a função custo $J(w)$, precisa-se aplicar um dos dois métodos popularmente conhecidos: Equação Normal ou Gradiente Descendente.

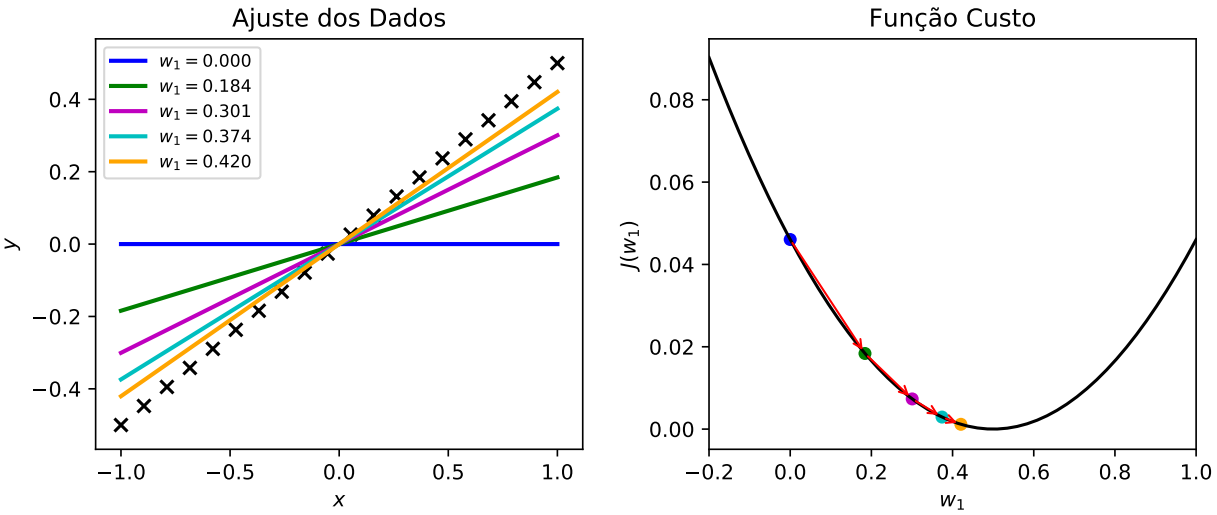
3.4.2 Gradiente Descendente (GD)

Gradiente Descendente (GD) constitui método numérico otimizador que procura os valores mínimos/máximos de uma função. No contexto de aprendizado de máquina, o otimizador procura os valores mínimos do coeficiente peso w, dada uma **função de custo**. Ou seja, percorre a função escolhendo sempre a direção de maior declive.

Frequentemente, minimizamos funções que possuem várias entradas e apenas uma saída (escalar): $f : \mathbb{R}^n \to \mathbb{R}$. E, portanto, funções com múltiplas entradas, devemos fazer uso do conceito de derivadas parciais.

O operador **gradiente** generaliza a noção de derivadas parciais para o caso em que a derivada se refere a um vetor. Logo, ao aplicar **gradiente** à função custo (equação 3.8), encontra-se a **derivada direcional** (equação 3.9) em que a função deve percorrer, conforme Figura 3.8.

Figura 3.8 – Ajuste dos dados em função do custo



Fonte – elaborada pelo autor.

$$\nabla J(\mathbf{w}) = \begin{bmatrix} \frac{\partial J(\mathbf{w})}{\partial w_0} \\ \vdots \\ \frac{\partial J(\mathbf{w})}{\partial w_n} \end{bmatrix} = \frac{1}{m}\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) \tag{3.9}$$

Porém, o método gradiente descendente é um processo interativo que requer etapas de aprendizagem até convergir. Na Figura 3.8 descrevem-se as etapas desse método:

a) Inicializar o valor do peso $\mathbf{w}^{[0]}$

b) A cada iteração $t = 1, 2, \ldots,$ atualize

$$\mathbf{w}^{[t+1]} = \mathbf{w}^{[t]} - \alpha \nabla J\left(\mathbf{w}^{[t]}\right)$$

em que o parâmetro α é chamado de **taxa de aprendizado**

c) Parar quando a diferença entre o valor de $\mathbf{w}^{[t]}$ e $\mathbf{w}^{[t+1]}$ for pequena.

A escolha da taxa de aprendizado é importante para a convergência. Se α é muito pequeno, a convergência pode ser lenta, e se α for muito grande, pode ocorrer *overshoot*. Nesse caso, o método pode não convergir ou até mesmo divergir.

É muito comum que que haja variação de técnicas fundamentadas na tradicional **gradiente descendente**. Entre elas estão: **gradiente descendente estocástica** e **gradiente descendente mini lotes**.

3.4.3 Redes Neurais Artificiais (RNA)

As redes neurais artificiais (RNA) são técnicas populares de aprendizado de máquina que simulam um mecanismo de aprendizagem em organismos biológicos. Pois seu funcionamento é inspirado nos neurônios do cérebro humano. No entanto, apenas servem como inspiração, porque não sabemos como o cérebro realmente funciona (AGGARWAL, 2018).

O componente mais simples desse mecanismo é o **neurônio**, também chamado de *unidade computacional*. As unidades computacionais são conectadas umas às outras por meio de pesos (w), e desempenham o mesmo papel que os pontos fortes das conexões sinápticas em organismos biológicos.

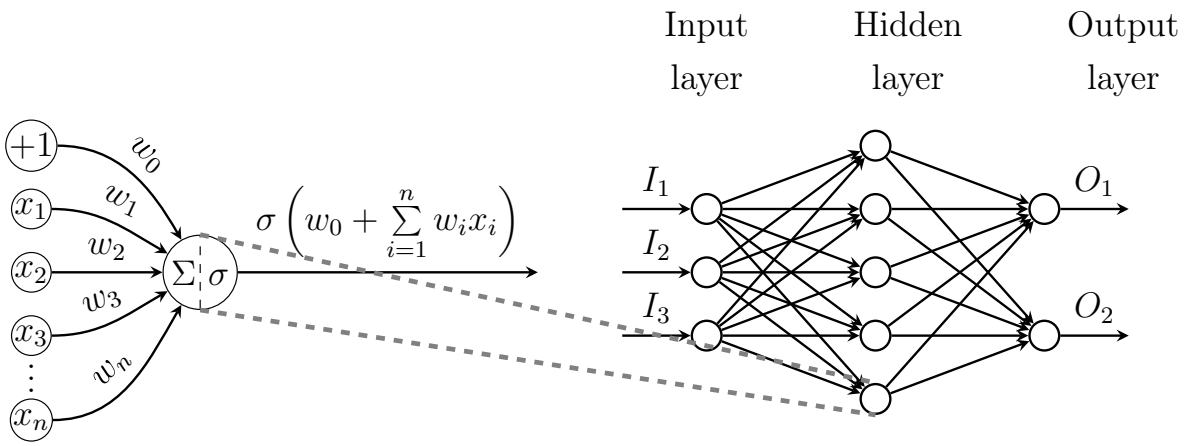
Em termos matemáticos, cada neurônio é escrito por um par de equações:

$$z = w_0 + \sum_{i=1}^{n} w_i x_i \tag{3.10}$$

$$\hat{y} = a = \sigma(z) \tag{3.11}$$

em que $x_1, x_2, \ldots, x_m$ são as entradas do neurônio; $w_1, w_2, \ldots, w_m$ são os pesos; w_0 representa o *bias* do neurônio; e $\sigma(.)$ representa a função de ativação. A ilustração de um neurônio pode ser vista na Figura 3.9.

Figura 3.9 – *Multilayer perceptron* (MLP).



Fonte – elaborada pelo autor.

Na literatura, as redes neurais podem ser explicadas por meio de duas arquiteturas: redes neurais de camada única (***perceptron***) e redes neurais de múltiplas camadas (***multilayer perceptron***), ambos são redes neurais *feedforward* sem realimentação. O objetivo das redes do tipo *perceptron*, inventadas por Rosenblatt (1958), é explicar um classificador binário que mapeia valores de entradas reais para um valor de saída binário. A função de ativação desse classificador é uma função degrau (chamada de função *Heaviside*) que determina valores de 0 e 1.

Em contrapartida, as redes de múltiplas camadas possuem neurônios que são organizados entre as camada de entrada, camadas ocultas e camada de saída. O que aumenta a complexidade do modelo. A função hipótese é construída por meio da composição de L funções vetoriais.

De forma geral, as redes neurais resolvem problemas de classificação e regressão. E sua camada de saída, é responsável por utilizar uma **função perda** diferente. Nas demais camadas, a saída do neurônio é aplicada a uma **função de ativação**, o que implica o bom desempenho para soluções de problemas não lineares.

3.4.4 Funções de ativação

As funções de ativação são normalmente aplicados na saída de cada neurônio, em alguns casos, são também utilizados dentro da arquitetura da rede, como o caso das redes LSTM e GRU, que utilizam a função *sigmoid* dentro da arquitetura.

O cálculo para aprendizagem de cada neurônio requer o conhecimento da derivada da função de ativação. Para que essa derivada exista, exigimos que a função $\sigma(.)$ seja contínua. Em termos básicos, a diferenciabilidade é o único requisito que uma função de ativação deve atender.

A função de ativação na origem para os *perceptron* era a função de *Heaviside*. Porém, a derivada dessa função tem valor nulo para todo $x! = 0$ e não é definida para $x = 0$, o que dificulta o treinamento dos neurônios.

Com o surgimento do algoritmo de treinamento *backpropagation* e o método gradiente, mostrou-se necessária a utilização de funções de ativação com derivadas não nulas em todos os pontos.

- *Heaviside*: inicialmente criada para determinar a função de ativação de um *perceptron* simples.

$$\sigma(x) = \begin{cases} 1 & \text{Se } x \geq 0 \\ 0 & \text{Se } x < 0 \end{cases}$$

- Sigmoide: é a função de ativação mais aplicada na construção das redes neurais. Definida como uma função estritamente crescente que exibe um equilíbrio entre comportamento linear e não linear.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- Tangente hiperbólica: a função $\tanh x$ é preferível à sigmoide quando as saídas dos cálculos são desejadas como positivas e negativas.As funções sigmoide e tanh têm sido as ferramentas históricas de escolha para incorporar a não linearidade na rede neural. Nos últimos anos, no entanto, várias funções de ativação linear por partes se tornaram mais populares, como a função de ativação *ReLU*.

$$\sigma(x) = \tanh x = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

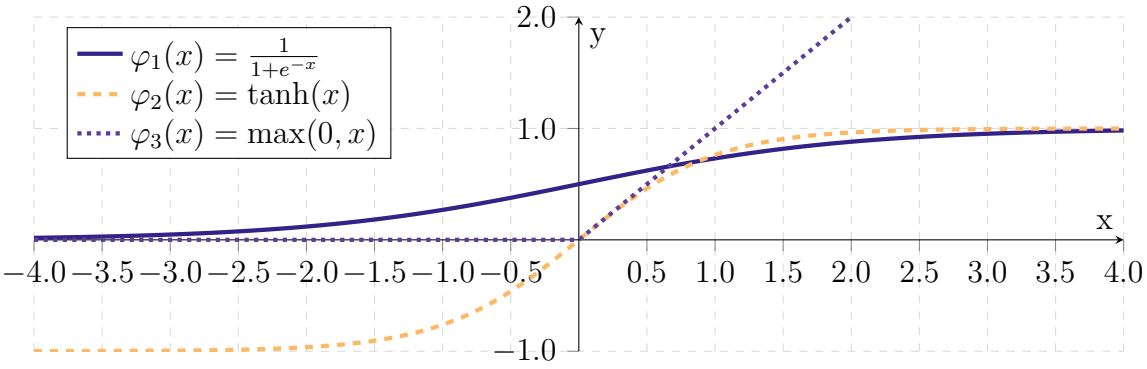
- *Rectified Linear Unit* (ReLU): a função de ativação ReLU foi responsável por substituir a função de ativação sigmoide nas redes neurais modernas devido à facilidade

em treinar redes neurais de várias camadas.

$$\sigma(x) = \max(0, x)$$

Na Figura 3.10, podemos observar todas as funções descritas anteriormente.

Figura 3.10 – Funções de ativação



Fonte – elaborada pelo autor.

3.4.5 Backpropagation

O *Backpropagation* é o algoritmo mais importante para a história das redes neurais. Pois, por meio desse algoritmo, foi possível resolver, computacionalmente, problemas que demorariam anos.

De forma resumida, o algoritmo *backpropagation* consiste em duas etapas fundamentais:

a) inicializar todos os pesos da rede com valores aleatórios;

b) ***forward propagation***: calcular as saídas estimadas para as entradas processadas;

c) ***backward propagation***: calcular os valores dos *gradientes* para cada peso da rede.

Após a inicialização de todos os pesos, o passo de propagação (*forward pass*) obtém os dados de saída $\hat{y}$ por meio da aplicação dos valores de entrada na função hipótese.

E, a partir dos valores encontrados $\hat{y}$, os valores dos pesos são atualizados da última camada até a primeira camada. Esse processo é definido como retro-propagação, na literatura é chamada de *backward propagation*. A atualização dos pesos implica o cálculo

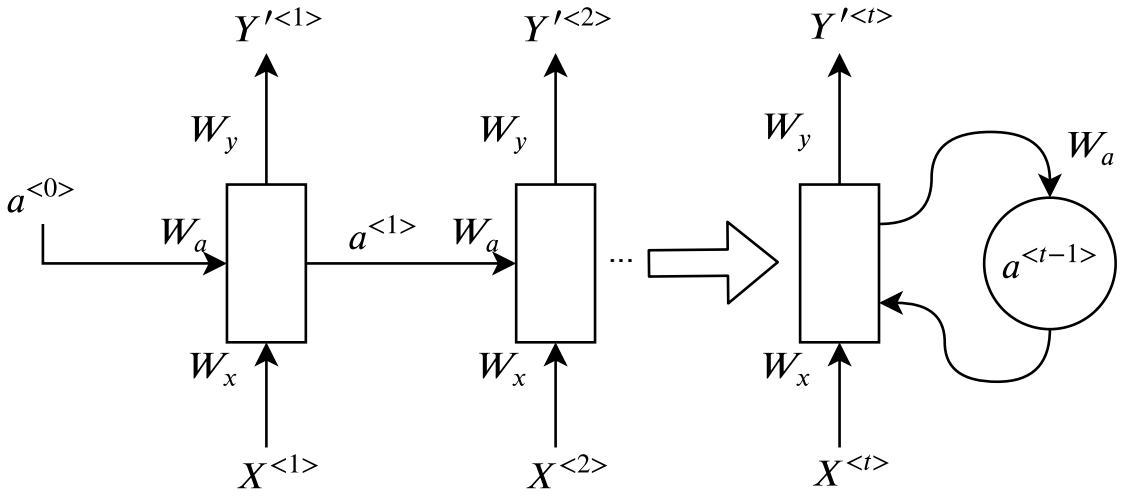
do gradiente em relação aos pesos das camadas mais profundas de uma rede neural. Isso ocorre, pois as redes neurais de multicamadas possuem o conceito de funções aninhadas; é preciso usar a regra da cadeia para calcular o gradiente dos pesos.

3.4.6 Redes Neurais Recorrentes (RNN)

As Redes Neurais Recorrentes (RNN) são arquiteturas de redes neurais mais indicados para processamento de dados sequencias (GOPINATH REBALA AJAY RAVI, 2019). Os dados sequenciais podem ser representados na forma de uma série temporal, um texto ou um vídeo. Onde sua aplicação pode envolver desde processamento de linguagem natural (PLN) para classificação de sentimentos até a predição da demanda de uma série temporal.

Entre os modelos conhecidos estão as arquiteturas: *Long Short Term Memory* (LSTM), *Gated Recurrent Unit* (GRU) e a versão simples do modelo RNN (também chamado de *Simple* RNN). O modelo *Simple* RNN é o mais simplificado entre os modelos, pois sua arquitetura é fundamentada apenas no estado da amostra anterior para a predição do valor-alvo. Para ficar mais claro, na Figura 3.11, apresentamos uma arquitetura simples de redes neurais recorrentes.

Figura 3.11 – Arquitetura *Simple* RNN.



Fonte – elaborada pelo autor.

Na Figura 3.11, a RNN se difere das redes neurais *feedforward* tradicionais, apenas em relação à realimentação dos dados de saída por meio de uma outra função de ativação. Matematicamente, é representado da seguinte forma:

$$a^{<t>} = g\left(W_a a^{<t-1>} + W_x X^{<t>} + b_a\right) \tag{3.12}$$

$$Y'^{<t>} = g\left(W_y a^{<t>} + b_y\right) \tag{3.13}$$

em que

- $a^{<t>}$ é a entrada de ativação calculada pela sequência t;
- $X^{<t>}$ é a entrada da sequência t;
- $Y'^{<t>}$ é a saída calculada da sequência t;
- W_x é o vetor de peso do vetor de entrada;
- W_y é o vetor de peso do vetor de saída;
- W_a é o vetor de peso para a entrada de ativação; e
- g é a função de ativação: *tanh* para os neurônios de entrada e *sigmoid* para a saída.

A RNN aproveita as informações contextuais presentes nos dados da sequência. Intuitivamente, toda entrada da sequência cria um estado de memória que é calculado a partir das entradas atuais e passadas. Essa memória, juntamente ao próximo vetor de entrada, é usada para calcular a saída da próxima sequência.

Os modelos de redes neurais recorrentes *Long Short-Term Memory* (LSTM) são introduzidas para resolver problemas de modelagem de longo prazo nas RNNs. Elas resolvem o problema de dissipação do gradiente (GOPINATH REBALA AJAY RAVI, 2019) projetando abordagem de célula de memória e fornecendo portas que controlam a influência da sequência anterior no estado atual e na saída da sequência atual.

Um bloco de memória contém uma ou mais pequenas memórias moduladas por portas sigmoidais não lineares e são aplicadas multiplicativamente. Para reduzir os parâmetros, as células de memória compartilham as mesmas portas. Essas portas determinam se o modelo mantém os valores nas portas (se as portas avaliadas como 1) ou as descartam (se as portas forem avaliadas como 0), desta forma, a rede pode explorar o contexto temporal de longo alcance. (CHNITI; BAKIR; ZAHER, 2017).

As redes neurais *Gated Recurrent Unit (GRU)* são introduzidas para simplificar os modelos de rede LSTM com menos números de variáveis a serem computadas para

geração do modelo. Os modelos GRU têm sindo eficazes na modelagem de soluções para processamento de linguagem natural (PLN) e são amplamente utilizados para representar modelos de redes neurais recorrentes.

4 DESENVOLVIMENTO DOS MODELOS

Conforme o que foi discutido na descrição do problema, o desenvolvimento do modelo ocorreu com o objetivo de resolver um problema de demanda para estimar a projeção do aumento da quantidade de serviços de teledermatologia (exames dermatológicos) no sistema STT/SC. De acordo com os requisitos pedidos pelos gestores da área da saúde, como parte de um desenvolvimento piloto, o sistema desenvolvido delimitou-se a um horizonte de predição de 1 mês e 12 meses.

Dessa forma, na parte inicial deste capítulo, serão descritos os parâmetros básicos e a ferramenta de coleta de dados que foram aplicados para a montagem da série temporal. E, mo restante do capítulo, será detalhada a implementação de cada um dos modelos preditivos.

Na prática, a diferença no desenvolvimento entre os modelos de treinamento está relacionada a dois aspectos: à transformação da série temporal para um conjunto que o modelo conheça e à escolha dos melhores parâmetros para cada tipo de abordagem. Assim, os modelos foram divididos em quatro processos diferentes de treinamento, entre eles estão: Suavização Exponencial, ARIMA, Regressão Linear e Redes Neurais.

4.1 INFORMAÇÕES BÁSICAS

Listagem de parâmetros básicos utilizados em todos os modelos:

a) horizonte de previsão igual a 12 meses e 1 mês ($h = 12$ e $h = 1$);

b) período do conjunto de dados (data_inicial = 2015/01/01 e data_final = 2019/12/31);

c) granularidade temporal: ***mês*** (freq_group = "1M").

O primeiro passo para o desenvolvimento do sistema de previsão é a especificação do conjunto de dados e o desenvolvimento do algoritmo para coletar os dados. A especificação do conjunto de dados é importante para delimitar as variáveis que servem para solucionar esse problema de demanda. A coleta dos dados descreve como será feita a consulta dos dados e sua transformação para uma série temporal.

4.1.1 Especificação do conjunto de dados

Como o objetivo principal é permitir que os gestores da administração pública possam mensurar a quantidade de exames de teledermatologia atendidos pelo sistema STT/SC. É preciso que esses dados possam ser visualizados por meio de componentes de visualização.

No entanto, a visualização dos dados depende, diretamente, do tipo de dado que será analisado. Abaixo separamos uma relação dos atributos que podem ser visualizados em componentes de visualização distintos:

- **Temporal**: variável dependente do tempo, com a granularidade ano, mês e dia.
- **Espacial**: informação geográfica, permitindo a exibição em mapas temáticos.
- **Numérico**: são valores numéricos que permitem o agrupamento por outros atributos. O agrupamento pode ser do tipo somatório, média ou desvio padrão.

Tabela 4.1 – Tabela de atributos

Atributos	Tipo de Dado	Descrição
exame_data_envio	Temporal	Data na qual o técnico envia as imagens de lesões de pele para o sistema.
qnt	Numérico	A quantidade de envio agrupado pela última granularidade da data.
instituicao_macrorregiao	Espacial	Informação espacial sobre a localização da ocorrência do evento.

Fonte – elaborada pelo autor.

Apenas os atributos *qnt* e *exame_data_envio* da Tabela 4.1 foram utilizados para o desenvolvimento dos modelos.

4.1.2 Coleta dos dados

Os dados foram coletados a partir da biblioteca *PyDruid*[1] do *Python*, que acessa, diretamente, o banco de dados *Apache Druid*[2] com objetivo de consultar informações

[1] https://github.com/druid-io/pydruid

[2] https://druid.apache.org/

temporais. A consulta aos dados podem ser observada no trecho de Código-fonte 4.1.

Código 4.1 – Acesso ao *Apache Druid*

```
client = PyDruid('https://******.telemedicina.saude.sc.gov.br', 'druid/v2/')
client.set_basic_auth_credentials('********','********')

group = client.groupby(
    datasource='cubo_exame_dermato', # Schema da Tabela
    granularity='day', # Granularidade diária
    intervals='2015-01-01/2019-12-31', # Período dos dados
    dimensions=["instituicao_macrorregiao"], # Dimensões.
    aggregations={"qnt": count("qnt")}, # Métricas.
    context={"timeout": 10000}, # Espera até 10 segundos de conexão.
    filter=(Dimension("valido") == "Sim") # Filtro para exames válidos.
)
df = group.export_pandas() # para trabalhar com pandas
```

Com relação à tecnologia *Apache Druid*, as colunas são divididas em dimensões e métricas. As dimensões podem ser avaliadas como atributos qualitativos, e as métricas como atributos quantitativos.

Assim, a execução do trecho de código resulta em uma tabela com três colunas: *timestamp* (data_exame_envio), *instituicao_macrorregiao* e *qnt*. Os atributos *timestamp* e *instituicao_macrorregiao* representam as dimensões, e o atributo *qnt* representa a métrica.

De forma resumida, o que é interessante para o desenvolvimento dos modelos está na Tabela 4.2, que apresenta a quantidade de exames dermatológicos prestados pelo sistema STT/SC nos últimos 5 anos.

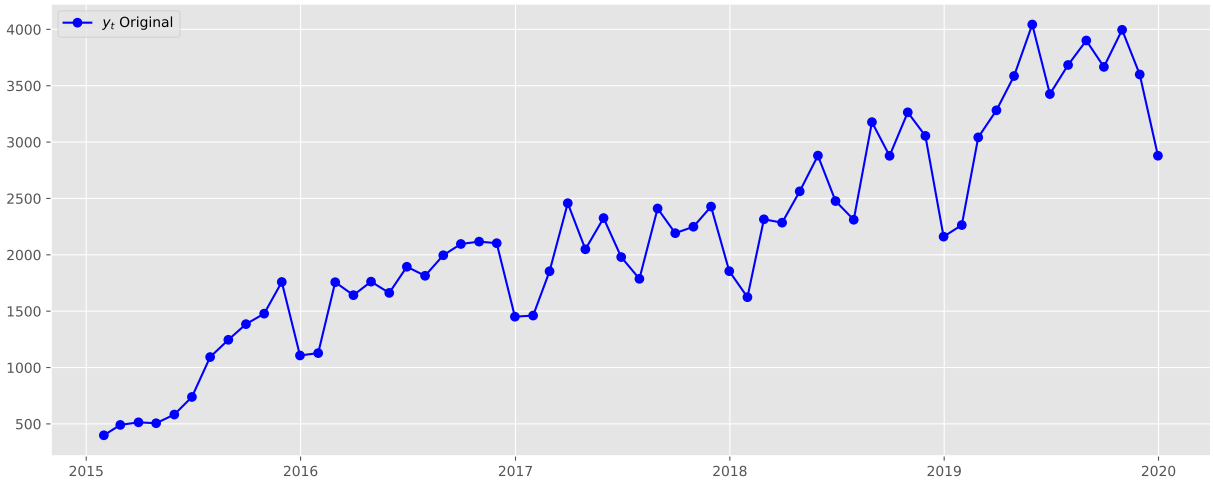
Tabela 4.2 – Evolução histórica de exames de teledermatologia

Ano	**2015**	**2016**	**2017**	**2018**	**2019**
Quantidade	11300	21422	25050	30990	41447
				Total:	130209

Fonte – elaborada pelo autor.

Além disso, na Figura 4.1, demonstra-se que a série histórica possui componente de tendência e componente de sazonalidade anual.

Figura 4.1 – Série original da quantidade de exames de teledermatologia por mês nos últimos 5 anos



Fonte – elaborada pelo autor.

4.1.3 Separação dos dados

Os dados foram separados em conjunto de treinamento e conjunto de teste. A separação ocorreu por meio da execução de um *script* pós-coleta dos dados, com o objetivo de dividir os conjuntos no formato de uma série temporal, ou seja, ambos os conjuntos foram transformados para um *Array* unidimensional de mesmo intervalo temporal.

Dessa forma, foram obtidos como resultados o conjunto de treinamento limitado entre **01/01/2015** até **31/12/2018** e o conjunto de teste limitado entre **01/01/2019** até **31/12/2019**.

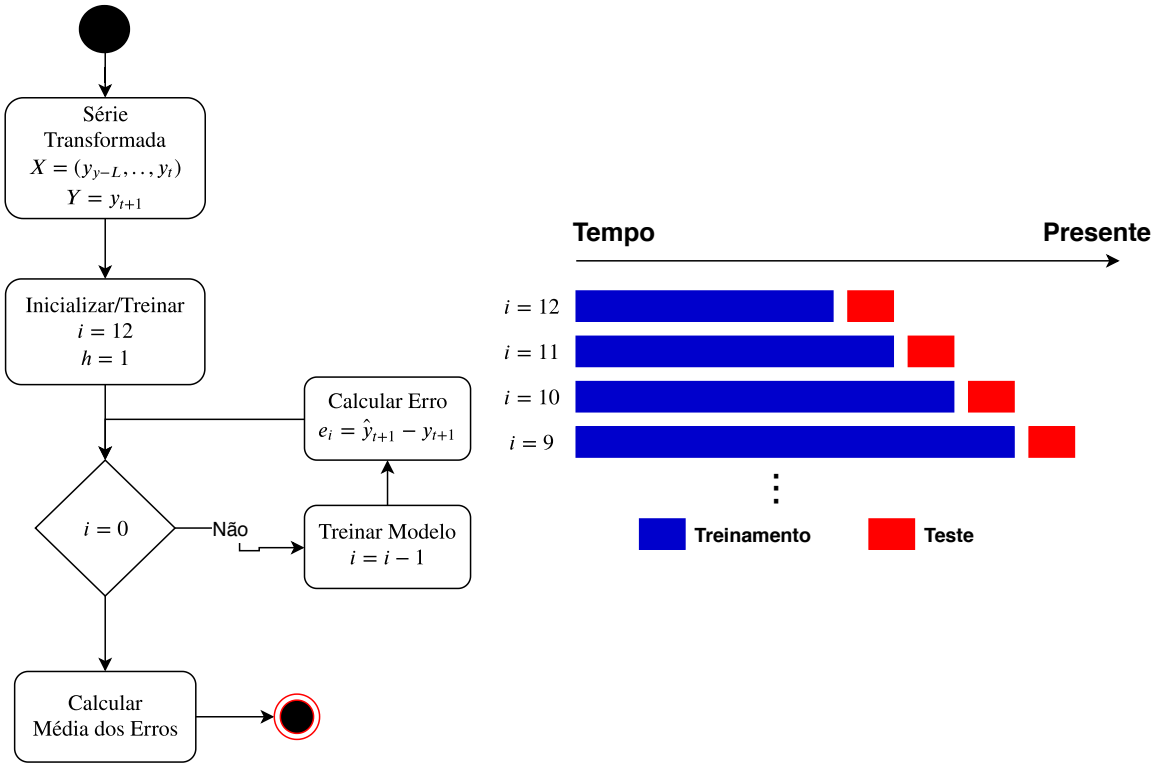
4.1.4 Algoritmos de treinamento

Em relação ao modo de treinamento para avaliar os modelos, escolheu-se a abordagem de treinamento de janela expansiva, em que o conjunto de treinamento vai se expandindo à medida que um modelo é treinado. Portanto, para avaliar diferentes horizontes de previsão, criaram-se dois cenários de implementação. O primeiro cenário envolve o horizonte de previsão de 1 mês e o segundo cenário envolve o horizonte de previsão de 12 meses.

O cenário, para o horizonte de previsão de 1 mês, permite que o modelo possa ser

treinado 12 vezes durante um ano (Figura 4.2). Porém, para o horizonte de previsão de 12 meses, não é possível vários treinamentos, já que o conjunto de teste não comporta mais que 12 meses (Figura 4.3). Então, para $h = 12$, foi possível, apenas, um treinamento.

Figura 4.2 – Algoritmo de treinamento para $h = 1$

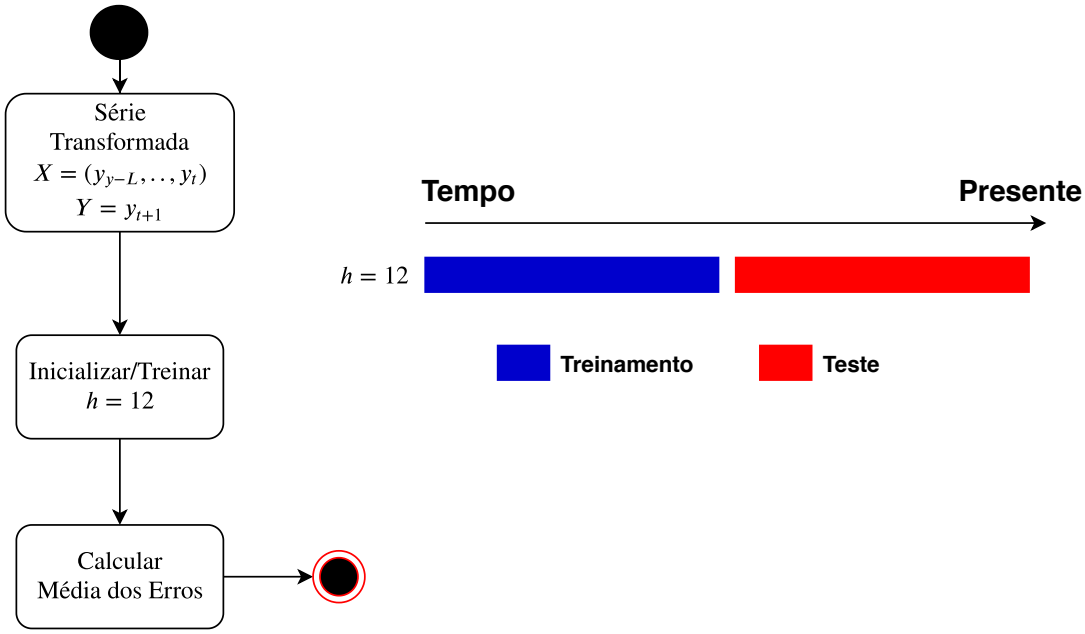


Fonte – elaborada pelo autor.

Na etapa chamada *Série Transformada*, que aparece em ambos os algoritmos da Figura 4.2 e 4.3, é realizada uma pequena transformação dos dados que cada modelo preditivo aplica de formas diferentes. Por exemplo, para regressão linear e redes neurais, os dados precisam ser divididos em conjunto de pares ordenados $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$ (Figuras 4.11 e 4.12). No entanto, para o modelo ARIMA e o modelo de suavização exponencial, os dados não precisam ser divididos, apenas organizados no formato de um *array* unidimensional.

Após a transformação dos dados, o modelo é treinado com os melhores parâmetros encontrados para cada modelo, que será apresentado nas seções seguintes. Na última etapa,

Figura 4.3 – Algoritmo de treinamento para $h = 12$



Fonte – elaborada pelo autor.

os erros MAE e RMSE são calculados da mesma forma para as duas abordagens.

4.2 IMPLEMENTAÇÃO DO MODELO SUAVIZAÇÃO EXPONENCIAL

É possível evidenciar, na Figura 4.1, que a série temporal apresenta as componentes de *tendência* e *sazonalidade*. Por esse motivo, escolheu-se o método de *Suavização Exponencial de Holt - Winters (HT)*. Desse modo, para a criação desse modelo, precisou-se encontrar os melhores valores para os parâmetros A, B e C, ou seja, variáveis que representam, respectivamente, a constante, tendência e sazonalidade.

Além disso, deve-se considerar a abordagem do modelo, se é uma abordagem aditiva ou multiplicativa. Adotou-se abordagem aditiva para a componente de tendência devido ao seu crescimento linear ao longo do tempo. E, no caso da componente de sazonalidade, foi adotada a abordagem multiplicativa devido ao aumento da distância entre picos e vales ao longo do tempo.

O trecho de Código-fonte 4.2 representa todos os parâmetros do modelo suavização

exponencial. Além dos parâmetros A, B e C, também é necessário definir o período de sazonalidade, ao passo que se escolheu o periodicidade de 12 meses devido a evidência empírica retirada do gráfico da Figura 4.1.

Código 4.2 – Modelo Suavizacao Exponencial

```
1 def suavizacao_exponencial(train, A, B, C, h):
2     N = len(train)
3 
4     model = ExponentialSmoothing(train, trend='add', seasonal='mul',
          seasonal_periods=12) # Abordagem sazonalidade multiplicativa
5     model_fit = model.fit(smoothing_level=A, smoothing_slope=B,
          smoothing_seasonal=C) # Parâmetros A, B, C
6     yhat = model_fit.predict(0,N+h-1) # yhat comporta toda a série original + as
          predições
7     return yhat
```

Esta implementação é simples, não precisou fazer nenhuma transformação ou normalização da série temporal. Diante da simplicidade do modelo, que não requer alto desempenho para o treinamento do modelo, implementou-se um algoritmo que procura os melhores parâmetros A, B e C. No qual, encontraram-se os valores 0.23, 0.23 e 0.01, respectivamente, para os parâmetros A,B e C.

A Figura 4.4 exibe um resumo das etapas para 10 amostras ($T = 10$), em que a série temporal recebida não precisou de nenhuma transformação, e, portanto, após a divisão em conjunto de treinamento e teste, foi possível treinar o modelo com base na escolha dos melhores parâmetros para o modelo.

4.3 IMPLEMENTAÇÃO DO MODELO ARIMA

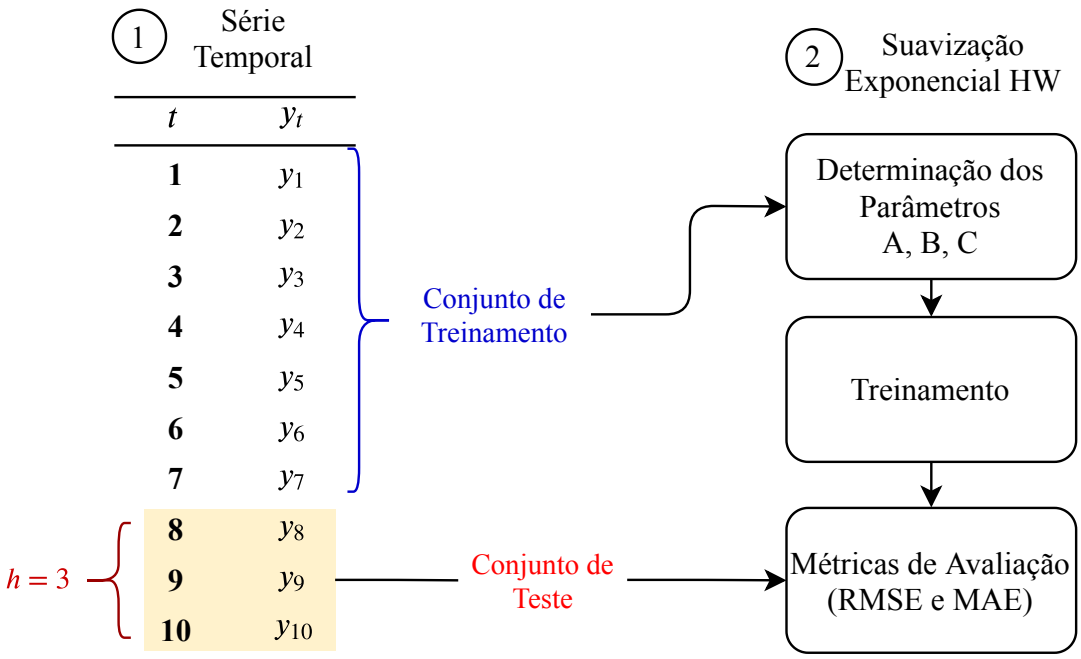
O desenvolvimento do modelo ARIMA segue o fluxograma da Figura 3.5, a busca dos parâmetros ocorreram com ajuda das funções de autocorrelações ACF e PACF. De maneira resumida, o modelo ARIMA é determinado pelos parâmetros *p, d e q*. Logo, é preciso seguir os seguintes passos:

a) encontrar a ordem de diferenciação **d** do modelo ARIMA;

b) encontrar a ordem **p** do termo AR(p);

c) encontrar a ordem **q** do termo MA(q).

Figura 4.4 – Suavização exponencial – exemplo de implementação para $h = 3$



Fonte – elaborada pelo autor.

4.3.1 Encontrar a ordem de diferenciação *I(d)*

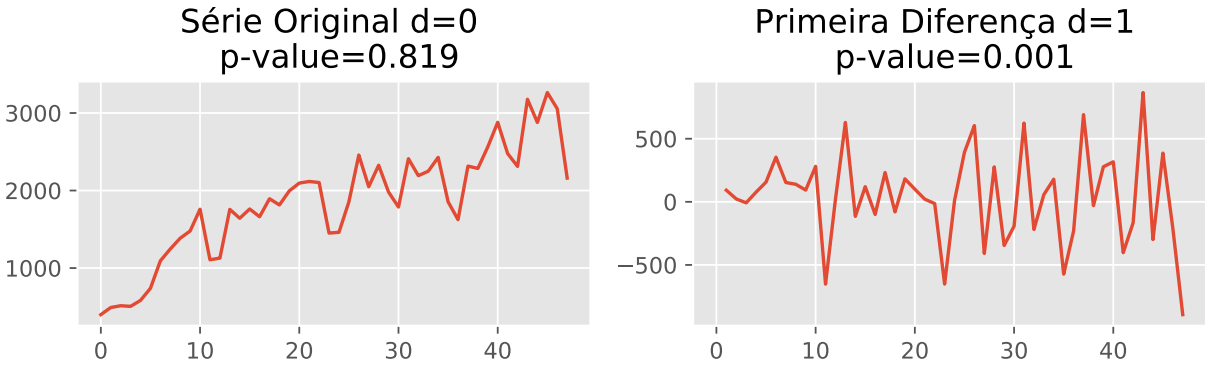
O primeiro passo é encontrar a diferenciação de modo a tornar essa série estacionária. Matematicamente, a diferenciação é calculado como: $y'_t = y_t - y_{t-1}$, aplicando a diferença na série original.

Podemos aplicar a hipótese *Augmented Dickey Fuller (ADF)* para verificar a estacionariedade da série. A hipótese nula do teste define que a série não é estacionária. Se o valor *p-valor* do teste for menor que o nível de significância 0.05, rejeita-se a hipótese nula, o que infere que a série é estacionária.

Para $d = 0$, sem diferenciação da série, o nível de significância ficou acima de 0.05, para ser mais exato p-value = 0.819, o que determina que ela pode não ser estacionária.

Para $d = 1$, com diferenciação de ordem 1, o nível de significância ficou abaixo de 0.05 (p-value = 0.001), o que torna a série estacionária. Visualmente pode ser visto por meio da Figura 4.5.

Figura 4.5 – Diferenciação I(d).



Fonte – elaborada pelo autor.

4.3.2 Encontrar a ordem do termo AR(p) e MA(q)

Os parâmetros p e q foram encontrados plotando suas funções de autocorrelações PACF e ACF. O parâmetro q pode ser encontrado visualizando a função ACF do gráfico da Figura 4.6, em que não possui termos de atraso relevantes acima do limiar.

Do mesmo modo, a função PACF é determinante para AR(p), no qual foi possível observar que existem alguns termos significativos acima do limiar. Resultando, assim, um conjunto de hipóteses de classes de modelos ARIMA(p,1,0), em que p pode variar até encontrar o melhor modelo.

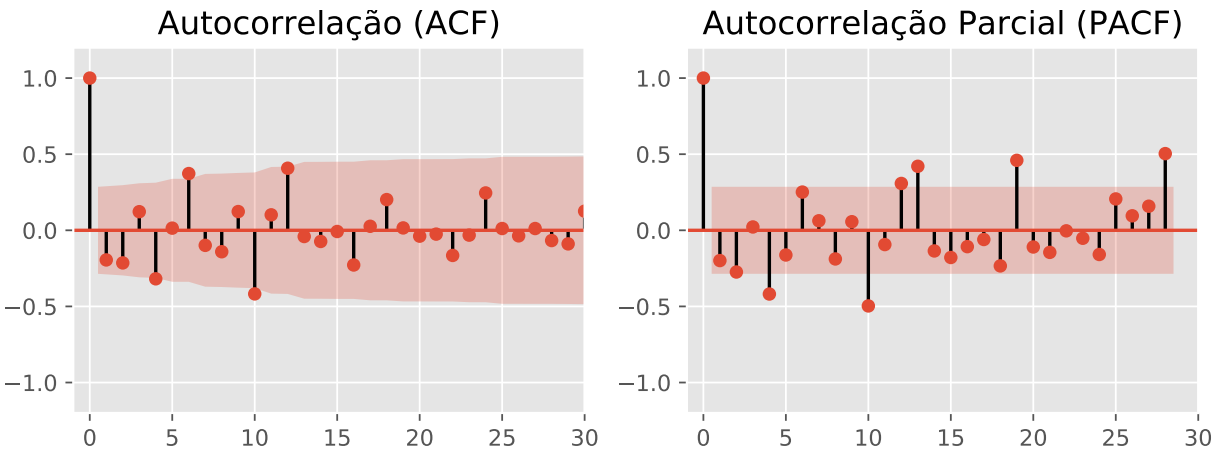
4.3.3 Aplicação dos parâmetros

Uma particularidade dos modelos paramétricos para os modelos não paramétricos é a estimação de um intervalo de confiança dado que a densidade dos resíduos pode se aproximar de uma distribuição normal (Figura 4.7). Desse modo, utilizou-se intervalo de confiança de 95% ($\alpha = 0.05$) .

Por exemplo, como resultado para a escolha do modelo ARIMA(4,1,0) e nível de significância 0.05, temos a Figura 4.8.

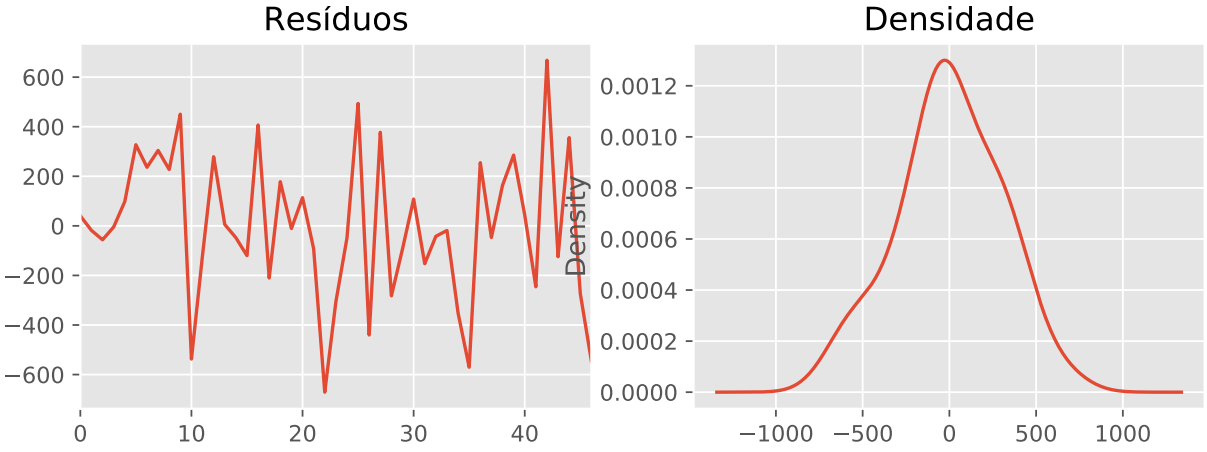
No entanto, o modelo ARIMA não é um bom modelo para séries temporais que apresentam a componente de sazonalidade. Por essa razão, escolheu-se o modelo SARIMA para melhorar a representação da série temporal analisada. Ainda assim, os parâmetros desenvolvidos para o modelo ARIMA serviram como base para a escolha dos parâmetros

Figura 4.6 – Funções de autocorrelação ACF e PACF



Fonte – elaborada pelo autor.

Figura 4.7 – Resíduos do modelo ARIMA(4,1,0)

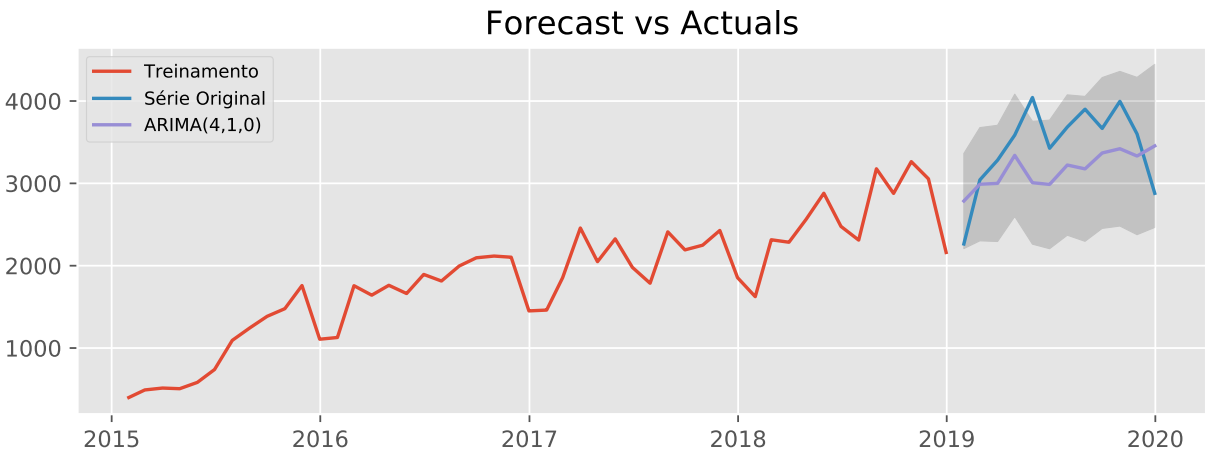


Fonte – elaborada pelo autor.

do modelo SARIMA. O modelo SARIMA combina dois modelos ARIMA's, em que um deles representa a sazonalidade.

A escolha do parâmetros do modelo SARIMA foi gerada de forma automática por meio de uma função do *python* de busca de parâmetros, chamada *pmdarima.auto_arima*(), em que, a partir de informações básicas sobre a série, há recebimento dos melhores parâmetros de acordo com os critérios de informações.

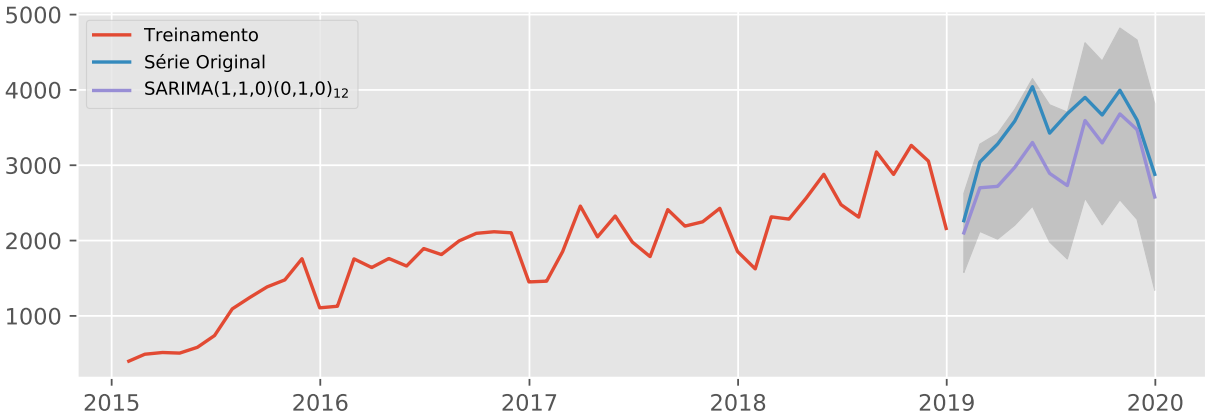
Figura 4.8 – Modelo ARIMA(4,1,0)



Fonte – elaborada pelo autor.

Portanto, como resultado, obteve-se o modelo SARIMAX(1, 1, 0)x(0, 1, 0, 12), e seu resultado de previsão pode ser visto na Figura 4.9.

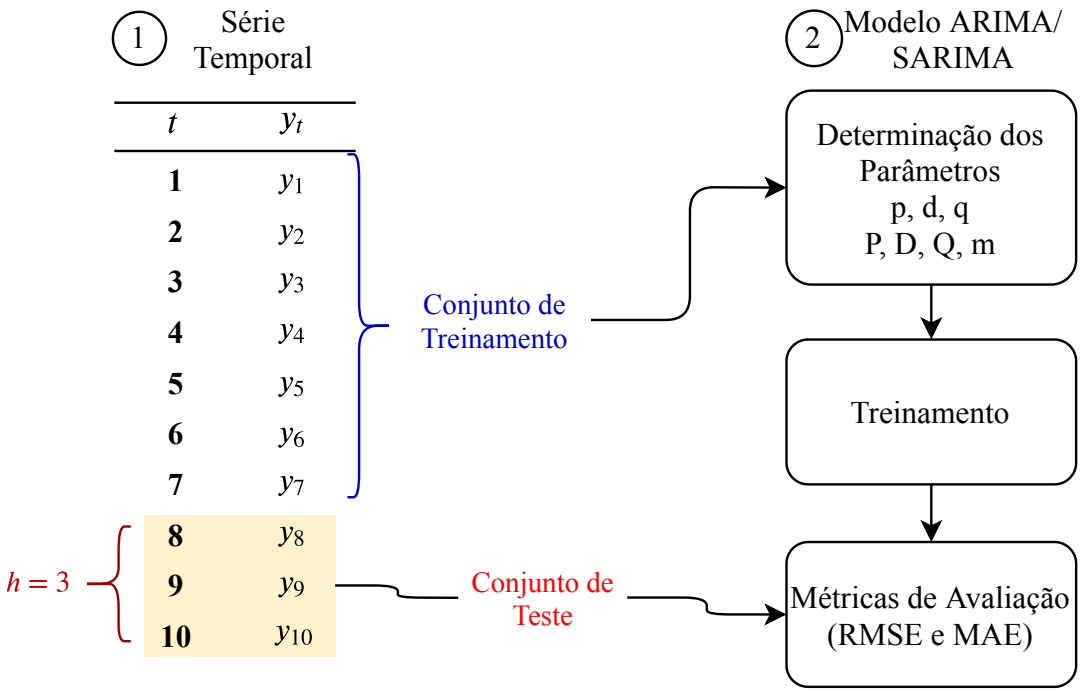
Figura 4.9 – Modelo SARIMA(1,1,0)x(0,1,0)$_{12}$



Fonte – elaborada pelo autor.

Resumindo, na Figura 4.10, apresentam-se, portanto, as etapas de treinamento e avaliação do modelo, assim como a suavização exponencial; esse modelo não requer nenhuma transformação da série temporal para treiná-lo.

Figura 4.10 – ARIMA/SARIMA – exemplo de implementação para $h = 3$



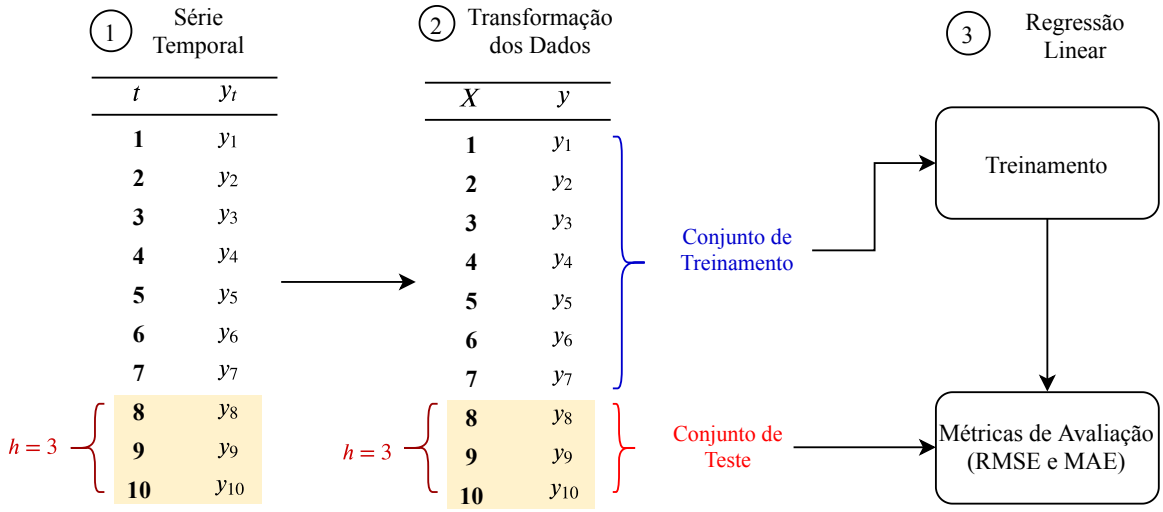
Fonte – elaborada pelo autor.

4.4 IMPLEMENTAÇÃO DO MODELO REGRESSÃO LINEAR

A implementação do modelo de Regressão Linear não precisou encontrar nenhum parâmetro de ajuste. Apesar de ser a base para o entendimento de redes neurais, ela é importante também para conhecer a componente de tendência.

Diferentemente dos métodos paramétricos, os dados dos métodos de aprendizagem de máquina precisam se dividir em conjuntos de entrada (X) e de saída (y). Isto é, a sequencia unidimensional da série temporal foi transformada em conjunto de pares $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_T, y_T)\}$, como observado na Figura 4.11.

Figura 4.11 – Regressão Linear – etapas de implementação ($h = 3$)



Fonte – elaborada pelo autor.

No trecho de Código-fonte 4.3, podemos observar a simplicidade de extrair o modelo de regressão linear.

Código 4.3 – Modelo de Regressão Linear

```
1 def regressao_linear(X_train, y_train):
2     regr = linear_model.LinearRegression()
3     regr.fit(X_train, y_train)
4     return regr
```

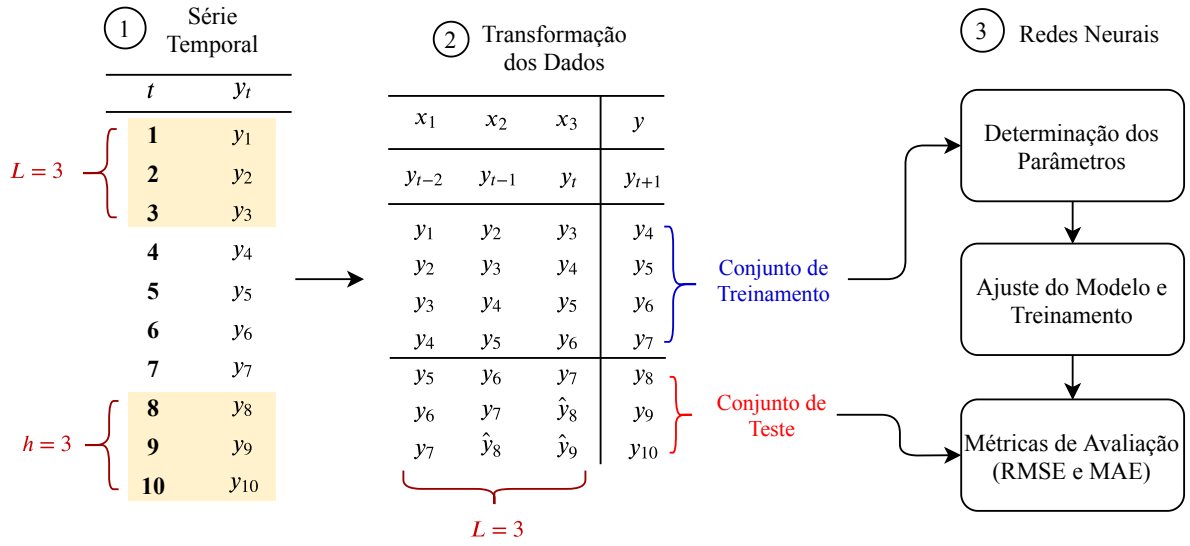
4.5 IMPLEMENTAÇÃO DOS MODELOS DE REDES NEURAIS

Os processos de implementação desses modelos estão descritos na Figura 4.12. A primeira parte do processo está no recebimento da série temporal por parte do banco de dados.

A segunda parte do processo refere-se à transformação dos dados, em que essa etapa executa duas tarefas. A primeira tarefa está na transposição dos dados para aumentar as dimensões de entrada. A segunda tarefa refere-se à criação dos conjuntos de entrada (X) e saída (y) do modelo.

A transposição da série temporal incrementou um novo parâmetro L, que permitiu aumentar o número de dados de entrada. A partir da Figura 4.12, é possível verificar uma janela de tamanho $L = 3$ que aumenta o número de variáveis de entrada, em que $X = (y_{t-2}, y_{t-1}, y_t)$ e $Y = y_{t+1}$.

Figura 4.12 – Etapas da implementação dos modelos de redes neurais



Fonte – elaborada pelo autor.

A última parte do processo constitui a escolha dos hiper-parâmetros das arquiteturas de redes neurais LSTM, RNN e GRU. Esta parte do processo envolve o treinamento das redes de forma interativa, até obter bons parâmetros para cada uma das redes neu-

rais. Em alguns casos, ocorriam problemas de estabilidade da convergência, porém com o incremento da variável *batch size*, as redes se estabilizavam.

A arquitetura, quanto às camadas e número de neurônios, pode ser vistas no trecho de Código-fonte 4.4.

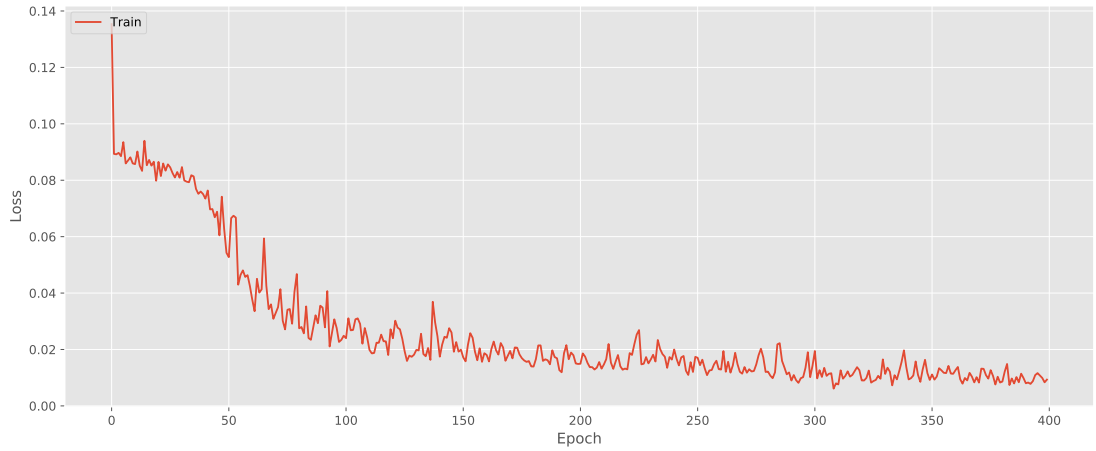
Código 4.4 – Arquitetura das redes neurais

```
def get_lstm(input_shape): # LSTM(Long Short-Term Memory)
    model = Sequential()
    model.add(LSTM(200, input_shape=input_shape, return_sequences=True))
    model.add(LSTM(100))
    model.add(Dense(1))
    return model

def get_rnn(input_shape): # RNN
    model = Sequential()
    model.add(SimpleRNN(200, activation='relu', input_shape=input_shape,
        return_sequences=True))
    model.add(SimpleRNN(200))
    model.add(Dense(1))
    return model

def get_gru(input_shape): # GRU(Gated Recurrent Unit)
    model = Sequential()
    model.add(GRU(64, input_shape=input_shape, return_sequences=True))
    model.add(GRU(64))
    model.add(Dropout(0.2))
    model.add(Dense(1))
    return model
```

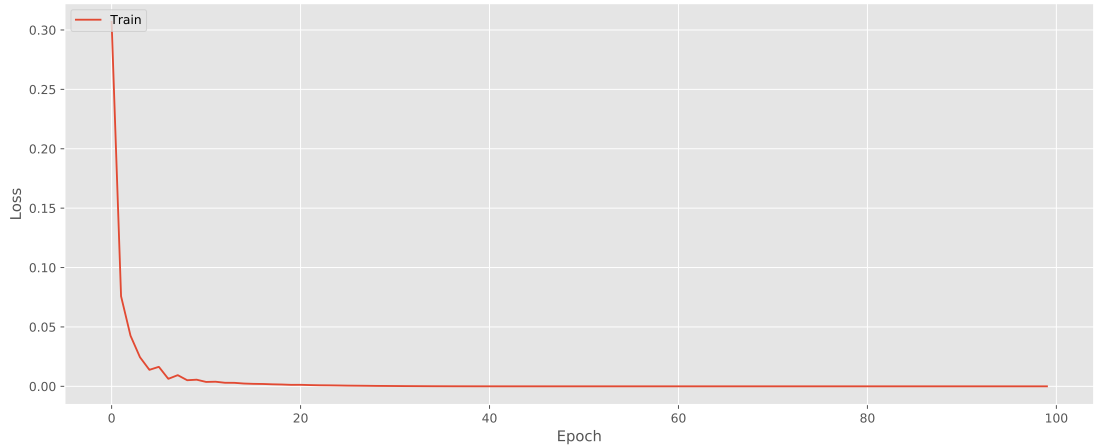
Na Figura 4.13, há o gráfico da função perda por épocas do treinamento de cada uma das arquiteturas LSTM, RNN e GRU. O modelo RNN apresenta estabilidade com poucas épocas de treinamento.

Figura 4.13 – Curvas da função perda por épocas para cada modelo

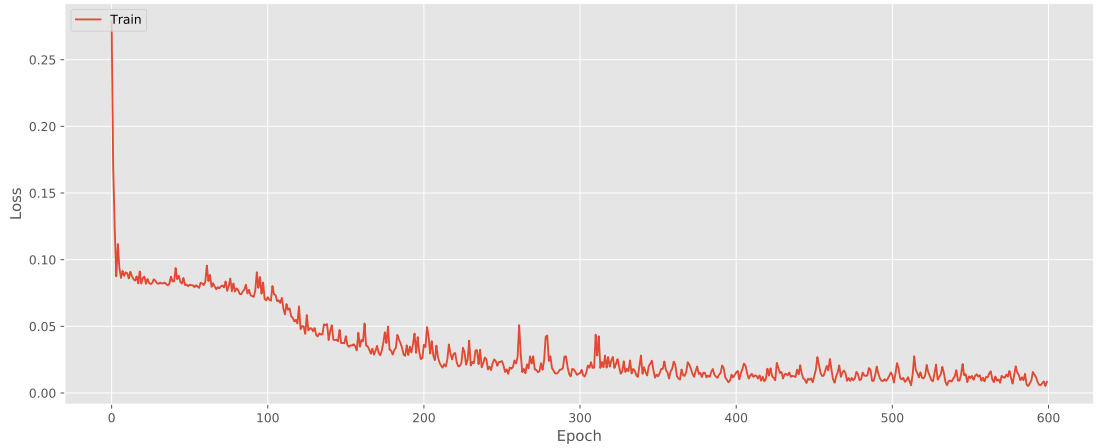
(a) LSTM



(b) RNN



(c) GRU



Fonte – elaborada pelo autor.

4.6 RESUMO DA ESCOLHA DOS PARÂMETROS

Diante dos modelos apresentados nas seções anteriores, a Tabela 4.3 apresenta um resumo dos parâmetros aplicados a esses modelos. Os mesmos parâmetros foram aplicados para horizontes de previsão de 12 meses e 1 mês.

Tabela 4.3 – Parâmetros dos modelos

$h = 12$ **e** $h = 1$	**Parâmetros**
Regressão Linear LR	Sem parâmetros
Suavização Exponencial HW	$A = 0.23$, $B = 0.23$, $C = 0.01$, $s = 12$ (periodicidade)
SARIMA	$p = 1$, $d = 1$, $q = 0$, $P = 0$, $D = 1$, $Q = 0$, $s = 12$ (periodicidade)
GRU	épocas = 600, $L = 24$ batch = 16 função perda = 'mae', algoritmo de treinamento = 'adam'
RNN	épocas = 100, $L = 24$ batch = 32 função perda = 'mae', algoritmo de treinamento = 'adam'
LSTM	épocas = 800, $L = 24$ batch = 16 função perda = 'mae', algoritmo de treinamento = 'adam'

Fonte – elaborada pelo autor.

5 RESULTADOS

Para avaliação dos modelos, foram utilizadas as métricas RMSE e MAE aplicados aos dois cenários descritos na introdução. As métricas de avaliação MAE_1 e $RMSE_1$ referem-se ao cenário que avalia o horizonte de previsão de 1 mês. Da mesma forma, as métricas MAE_{12} e $RMSE_{12}$ avaliam o horizonte de previsão de 12 meses.

Tabela 5.1 – Comparação do desempenho dos modelos

$h = 12$ **e** $h = 1$	$\mathbf{MAE_{12}}$	$\mathbf{RMSE_{12}}$	$\mathbf{MAE_1}$	$\mathbf{RMSE_1}$
Regressão Linear LR	456.09	513.50	415.83	492.39
Suavização Exponencial HW	226.61	310.48	249.56	315.46
SARIMA	446.42	502.87	217.39	242.91
GRU	390.27	472.97	554.11	828.70
RNN	241.96	271.56	198.14	218.55
LSTM	361.69	471.89	393.78	455.91

Fonte – elaborada pelo autor.

Diante dos resultados apresentados na Tabela 5.1, os modelos *RNN* e *Suavização Exponencial HW* produziram bons resultados para horizontes de previsão de 1 mês e 12 meses. Também há o modelo SARIMA, considerando, apenas, o horizonte de previsão de 1 mês.

A regressão linear pode servir de parâmetro para descartar outros modelos, já que é um modelo simples que não modela a sazonalidade da série temporal. Entre os modelos apresentados, podemos descartar o modelo GRU, que obteve um desempenho muito abaixo da regressão linear.

Apesar de os modelos apresentarem dois grupos distintos (clássica e aprendizado), algumas vantagens podem ser consideradas em cada tipo de abordagem. Ambos possuem modelos com bons desempenho, porém, os modelos de abordagem clássica dependem de um maior conhecimento do cientista de dados com as séries temporais. O que não ocorre com os modelos de aprendizagem de máquina.

Os modelos clássicos também permitem a aplicação de inferência estatística, o que permitiria aplicar probabilidade para obtenção de um intervalo de confiança, como no

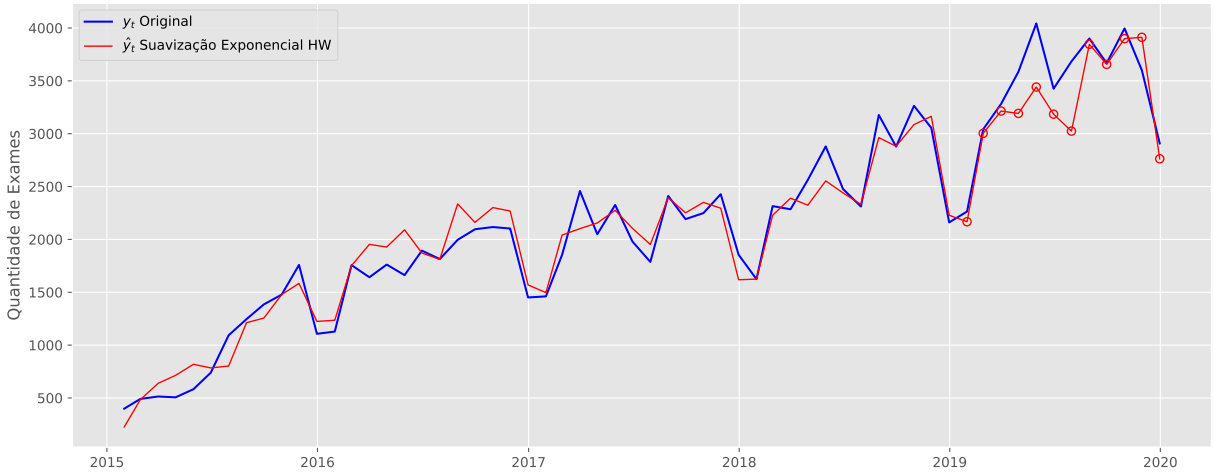
caso do modelo ARIMA.

Já era esperado que os métodos clássicos apresentassem as componentes de sazonalidade e de tendência no período previsto, pois esses componentes são modelados por meio da evidência empírica do cientista de dados em relação à série temporal. No entanto, as redes neurais não possuem tal requisito. Mesmo assim, foi possível perceber o aprendizado das redes neurais em relação a essas componentes.

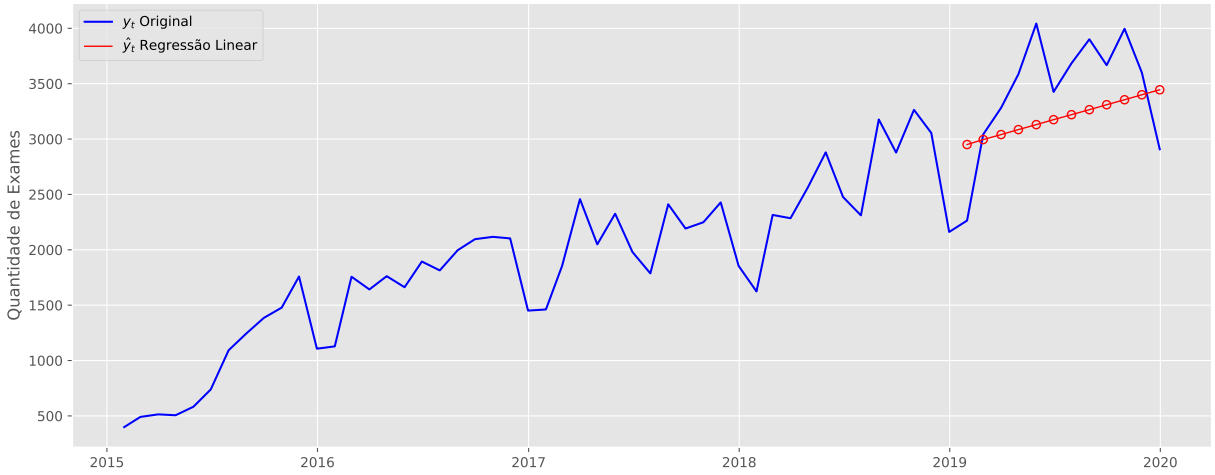
De forma geral, podemos observar as componentes de tendência e sazonalidade em quase todos os modelos da Figura 5.1 e Figura 5.2, exceto para o modelo de regressão linear, que apresenta, apenas, a tendência crescente da série. Também é possível observar que o período sazonal mais acentuado ocorre, principalmente, de dezembro a fevereiro, e esse fenômeno se repete todos os anos.

Figura 5.1 – Gráfico com os resultados da predição do conjunto de testes de 2019 – Parte 1

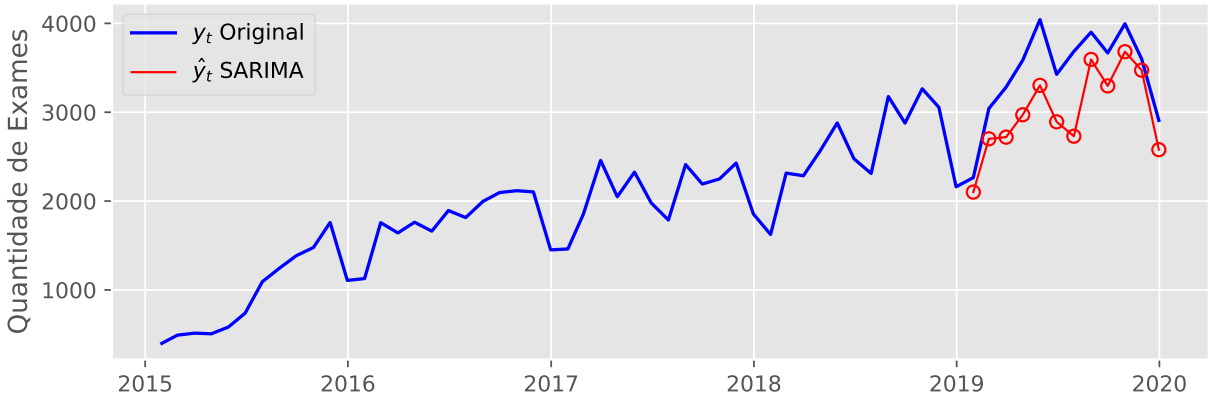
(a) Suavização exponencial HW



(b) Regressão Linear



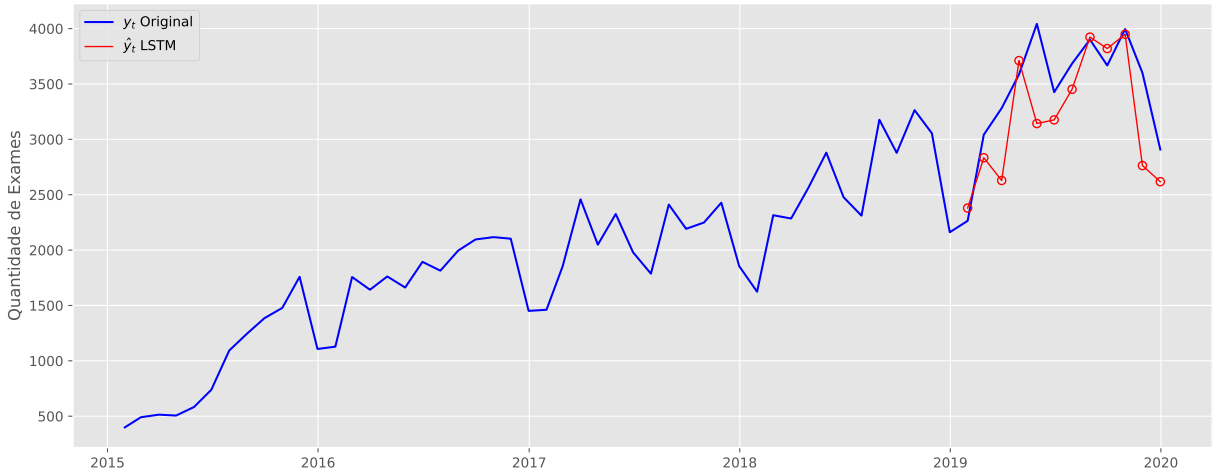
(c) SARIMA



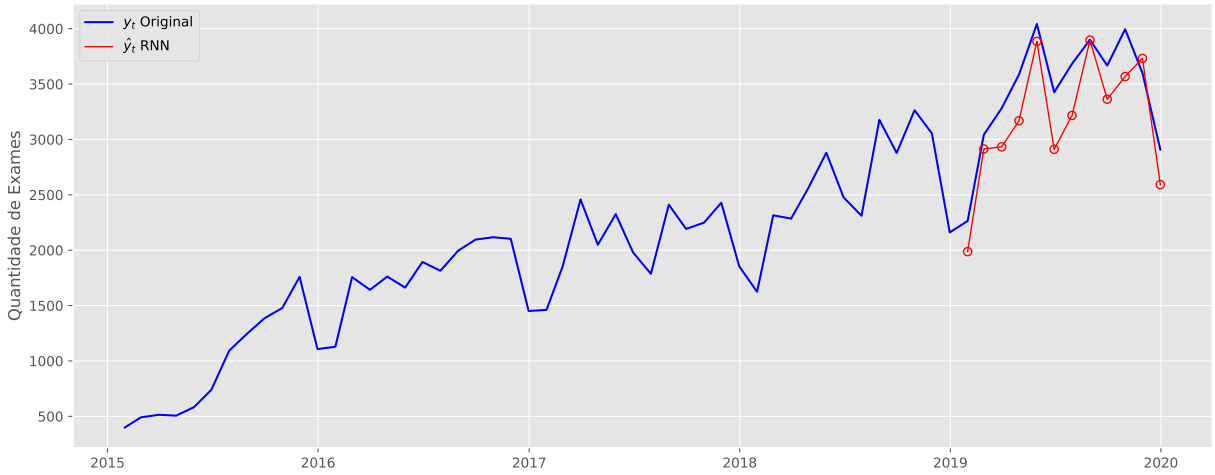
Fonte – elaborada pelo autor.

Figura 5.2 – Gráfico com os resultados da predição do conjunto de testes de 2019 – Parte 2

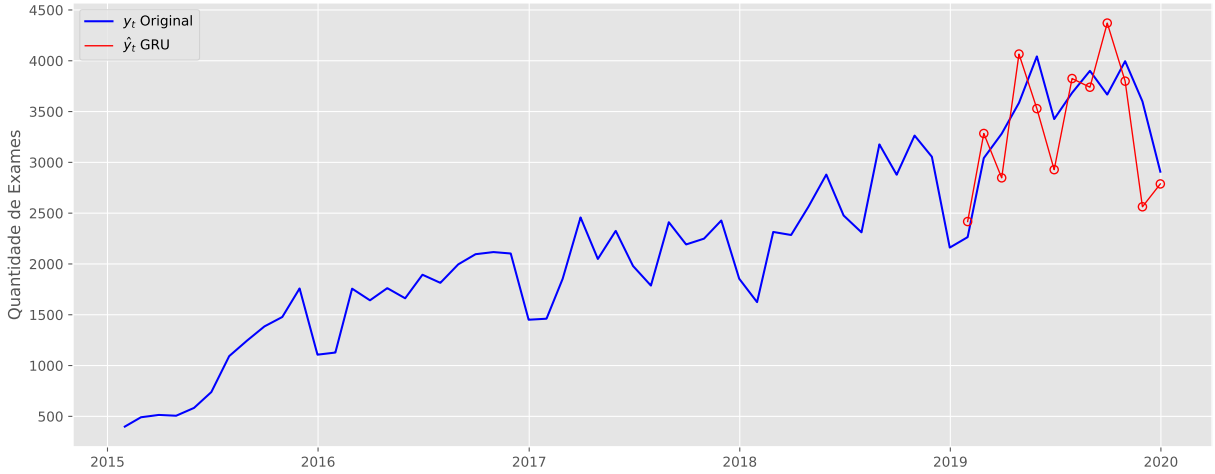
(a) LSTM



(b) RNN



(c) GRU



Fonte – elaborada pelo autor.

6 INTEGRAÇÃO COM O STT/SC

Diante dos resultados obtidos no capítulo 5, algumas conclusões podem ser tomadas em relação aos melhores modelos. Porém, a aplicação na prática desses modelos depende, diretamente, de uma interface gráfica de fácil interação com os dados por parte dos profissionais de saúde. Nesse sentido, é importante a escolha de uma arquitetura de software que possibilite a visualização dos dados em rotinas que já fazem parte dos profissionais que acessam o STT/SC.

De modo geral, a arquitetura escolhida deve realizar 4 tarefas especificas: visualização de dados, armazenamento de dados, geração de dados e execução periódica dos *scripts*. Na Tabela 6.1, temos uma descrição das tecnologias escolhidas requeridas para o desenvolvimento desse sistema.

Tabela 6.1 – Tecnologias utilizadas no desenvolvimento do sistema

Tecnologia	Descrição
Saiku	Visualização dos dados históricos e dados de predição.
Apache Druid	Banco de dados orientada a séries temporais.
Jupyter	Para desenvolvimento dos modelos preditivos e geração dos novos dados temporais.
Script of Script (SoS)	Gerenciamento dos *scripts* desenvolvidas no Jupyter e execução periódica dos *scripts* no ambiente de produção.

Fonte – elaborada pelo autor.

A escolha do visualizador de dados *Saiku*[1] se deve pela familiarização da ferramenta pelos profissionais que analisam dados no sistema STT/SC. No caso do *Apache Druid*[2], a escolha relaciona-se ao tipo de dado armazenado, pois esse banco de dados trabalha com dados temporais. Em relação à tecnologia *Jupyter*, a escolha passa pela quantidade de bibliotecas de aprendizagem de máquinas e de estatísticas disponíveis na comunidade de software livre.

[1] https://github.com/OSBI/saiku.

[2] https://druid.apache.org/.

E, para finalizar, a preferência no uso da ferramenta *Script of Script*[3] (SoS) está relacionado à tecnologia *Jupyter*[4], pois sua principal tarefa é a transformação dos algoritmos desenvolvidos no Jupyter em *scripts* que podem ser gerenciados por meio de um agendador. Ou seja, os algoritmos desenvolvidos em Jupyter passaram a ser executados periodicamente por meio da ferramenta SoS.

Diante das tecnologias apresentadas, o diagrama de blocos da Figura 6.1 demonstra as responsabilidades de cada tecnologia perante o ciclo completo do processo de predição. No primeiro momento, os dados são consultados no banco de dados Apache Druid (Consulta de dados) com o objetivo de serem *transformados* para que os modelos possam ser treinados. E, após o treinamento, os dados são gerados a partir do bloco de "Geração de Dados"e armazenados no banco de dados *Apache Druid*.

Nesse caso, a integração entre o *Saiku* e o *Apache Druid* ocorre de forma simples. É preciso configurar um arquivo XML que define as colunas que referenciam as dimensões (Ano, Mes, Tipo de Modelo, Erro) e as colunas referentes à métrica (quantidade de exames).

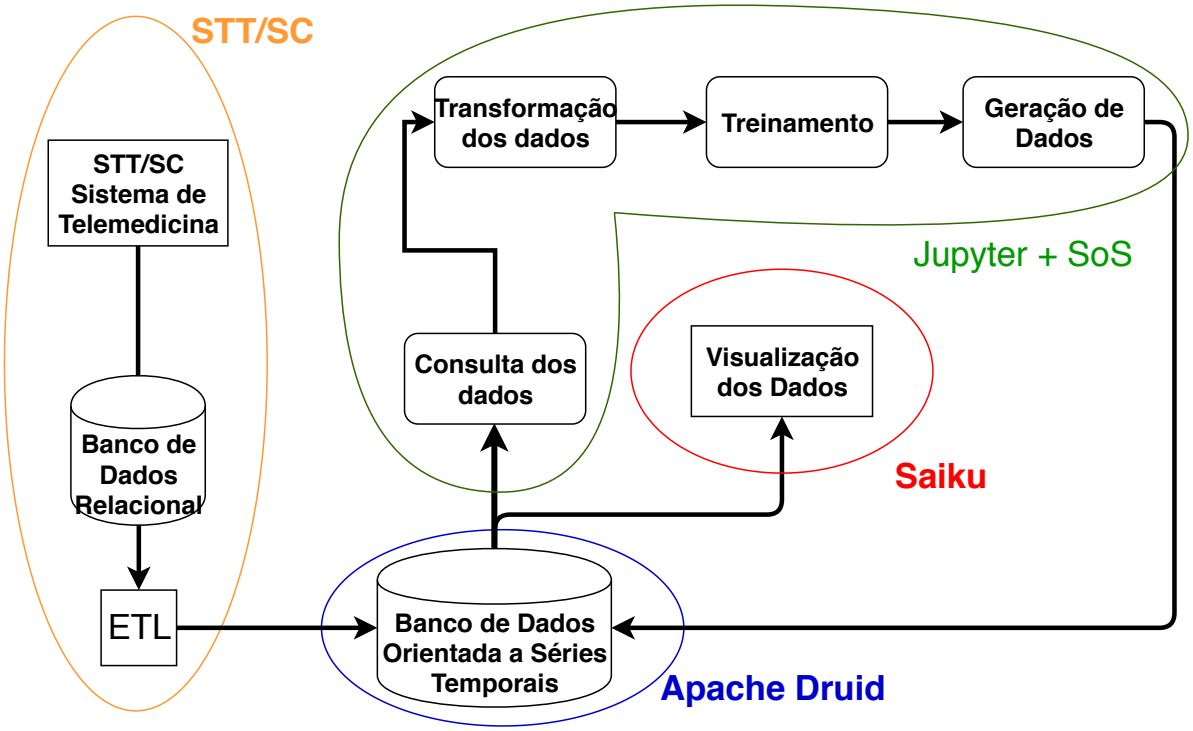
Em virtude dessa integração com o STT/SC, o trabalho desenvolve um sistema que interage com *aplicações* que fazem parte do STT/SC. Entre essas aplicações destaca-se o software livre *Saiku*, que é uma ferramenta de análise de dados, em que é utilizado por administradores para analisar dados históricos da demanda dos exames de teledermatologia.

Em relação aos *scripts* de automatização, os processos são definidos da seguinte forma: 1 – ETL (Extração, Transformação e Carregamento) para atualização do banco de dados orientada a séries temporais. 2 – Consulta de Dados para o Jupyter. 3 – Transformação dos dados para o treinamento. 4 – Geração de dados (predição) para o armazenamento no banco de dados.

[3] https://vatlab.github.io/sos-docs/.

[4] https://jupyter.org/.

Figura 6.1 – Diagrama de processos e fluxo de dados



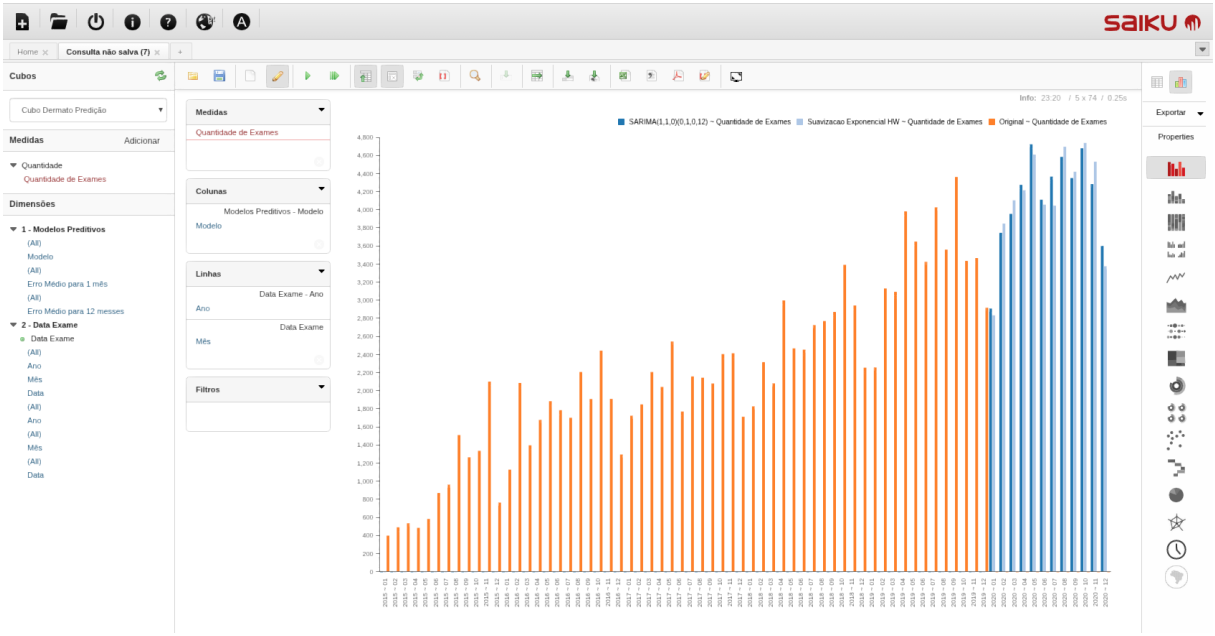
Fonte – elaborada pelo autor.

6.1 INTERFACE DE VISUALIZAÇÃO DOS DADOS

Um dos critérios para a integração com o sistema de telemedicina STT/SC é o reaproveitamento da interface de visualização *Saiku*. A seguir há alguns exemplos de telas elaboradas com os dados gerados para 2020.

A primeira tela de demostração, exibida na Figura 6.2, é um gráfico de barras que na cor laranja representa a série original e em azul representa a projeção dos dados para o ano de 2020 por meio do modelo de suavização exponencial.

Figura 6.2 – *Saiku*: Gráfico de barras da quantidade de exames de teledermatologia para 2020

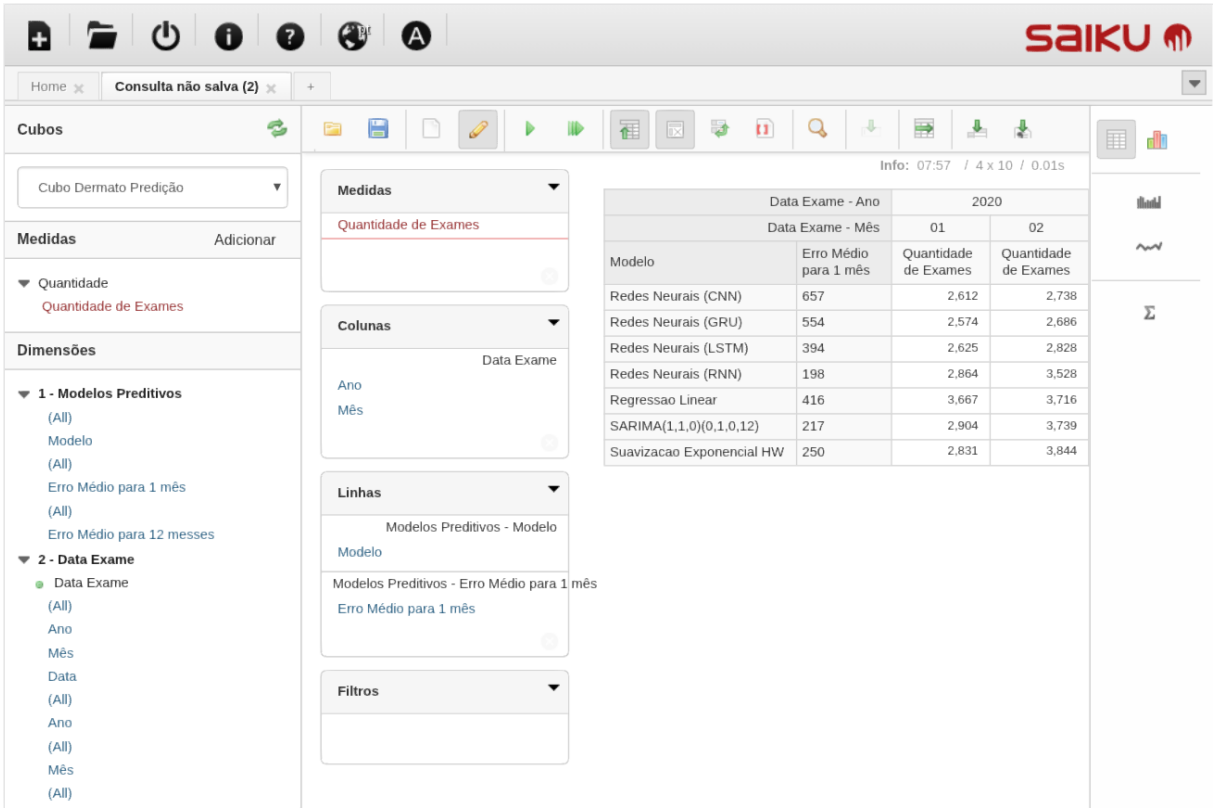


Fonte – elaborada pelo autor.

Na tela da Figura 6.3, há o menu principal da ferramenta *Saiku*. Ao lado esquerdo, é possível selecionar atributos de medidas (Quantidade de Exames) e atributos de dimensões (Modelo, Data do Exame e Erro Médio). Ao escolher, pelo menos, dois atributos, é possível gerar uma tabela pelo cruzamento desses atributos.

A Figura 6.4 exibe um gráfico de projeção da quantidade de exames para 2020, considerando-se, apenas, os modelos de Regressão Linear, Redes Neurais (RNN), Suavização Exponencial HW e SARIMA.

Figura 6.3 – Menu principal da ferramenta Saiku.



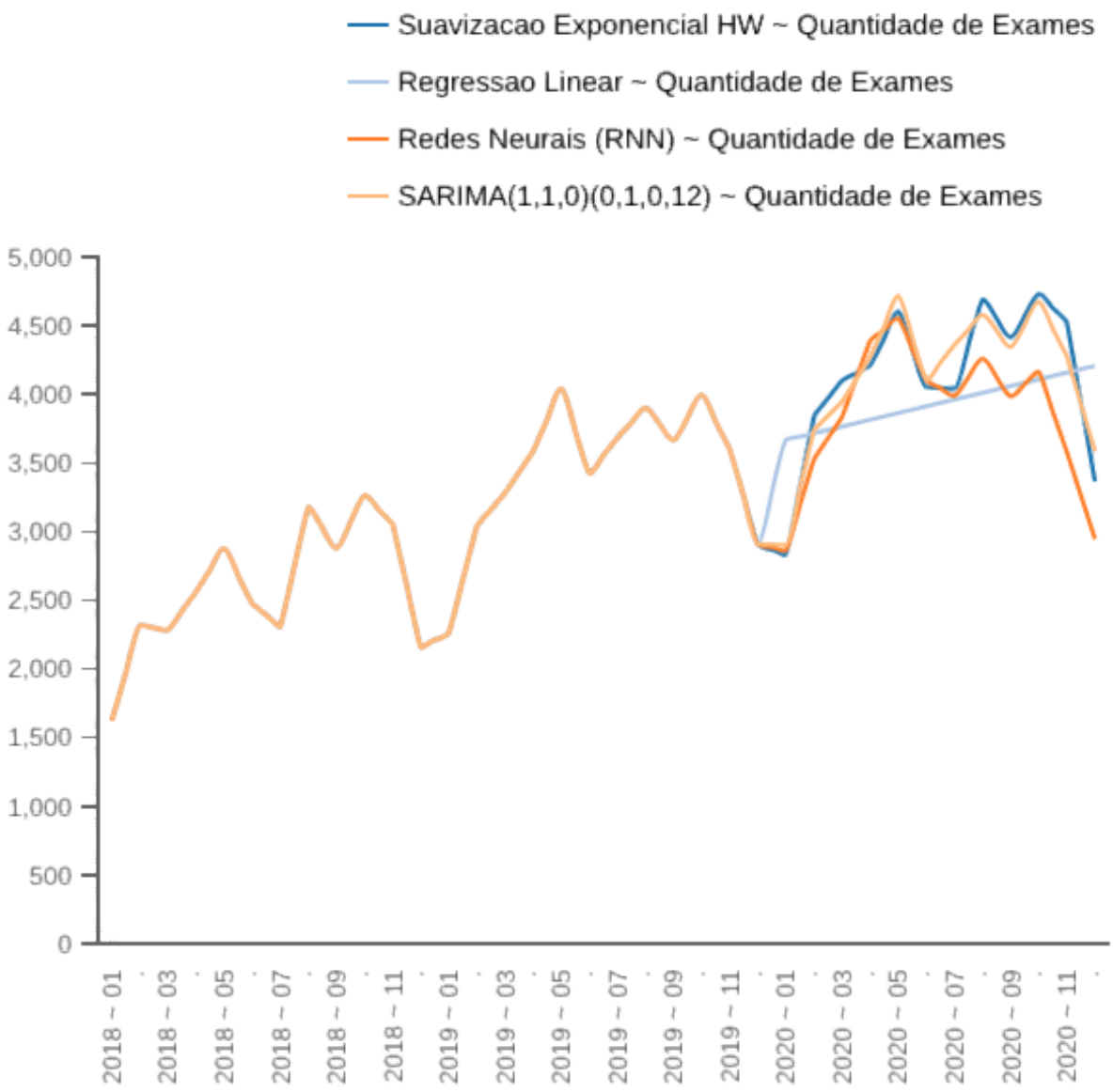
Fonte – elaborada pelo autor.

Por meio da funcionalidade de filtro do *Saiku*, é possível escolher um ou mais modelos preditivos a serem exibidos usando qualquer interface gráfica pertencente à ferramenta *Saiku* (Figura 6.5).

Na Figura 6.6, há duas tabelas com o prognóstico de tempo diferente. O lado esquerdo da imagem exibe o prognostico para 2020 da quantidade total de exames que serão executados. E, ao lado direito, demonstra a predição mensal dos dois próximos meses.

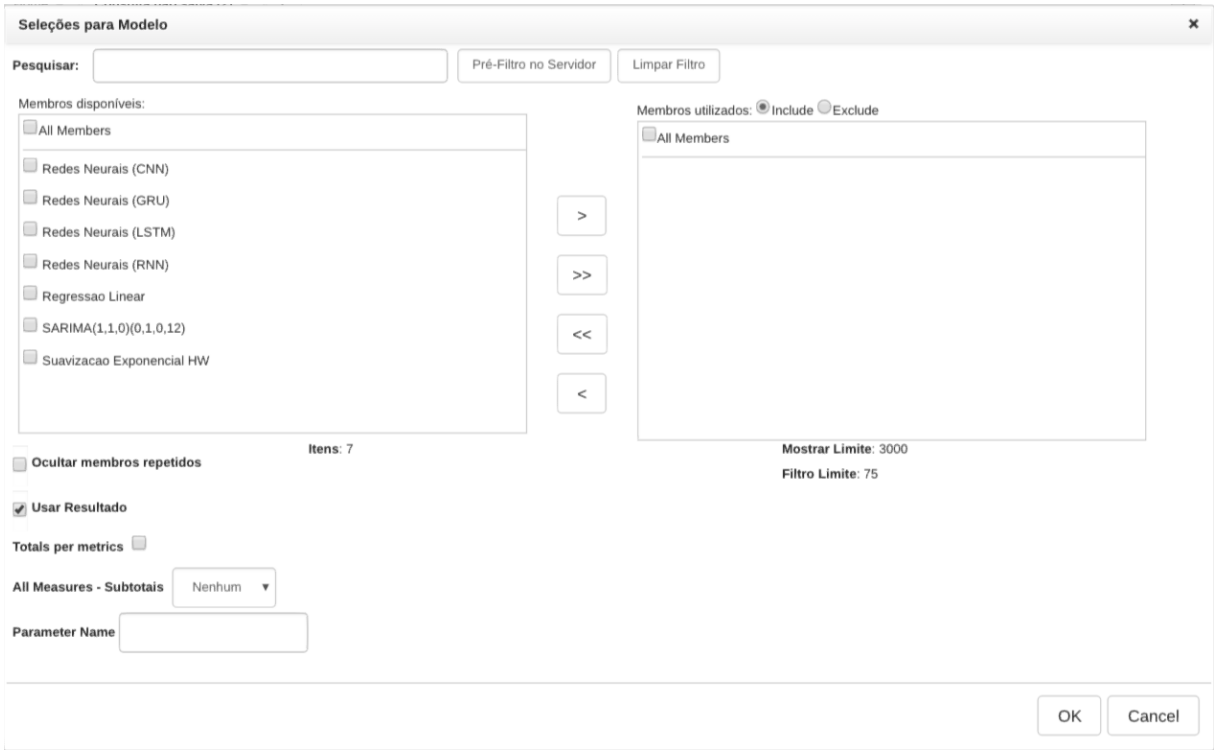
Outra possibilidade que a ferramenta permite é a exportação da tabela para os formatos *.csv* ou *xls* que podem ser manipulados por outras ferramentas de análise.

Figura 6.4 – *Saiku*: Gráfico com projeção para 2020 da quantidade de exames.



Fonte – elaborada pelo autor.

Figura 6.5 – *Saiku*: Ferramenta permite escolher o modelo preditivo para análise.



Fonte – elaborada pelo autor.

Figura 6.6 – *Saiku*: Tabelas de predição.

(a) Próximo Ano

Data Exame - Ano	2020
Modelo	Quantidade de Exames
Redes Neurais (CNN)	42,973
Redes Neurais (GRU)	42,595
Redes Neurais (LSTM)	42,328
Redes Neurais (RNN)	46,196
Regressao Linear	47,247
SARIMA(1,1,0)(0,1,0,12)	49,501
Suavizacao Exponencial HW	49,403

(b) Próximos 2 meses

Data Exame - Ano	2020	
Data Exame - Mês	01	02
Modelo	Quantidade de Exames	Quantidade de Exames
Redes Neurais (CNN)	2,612	2,738
Redes Neurais (GRU)	2,574	2,686
Redes Neurais (LSTM)	2,625	2,828
Redes Neurais (RNN)	2,864	3,528
Regressao Linear	3,667	3,716
SARIMA(1,1,0)(0,1,0,12)	2,904	3,739
Suavizacao Exponencial HW	2,831	3,844

Fonte – elaborada pelo autor.

CONCLUSÃO

Entre os modelos de redes neurais desenvolvidos para este trabalho, apenas o modelo RNN apresentou bons resultados, no entanto, para os métodos de abordagem clássica (ARIMA e suavização exponencial), todos apresentaram bons resultados.

É possível verificar que os modelos que apresentaram bons resultados não possuem valores discrepantes em relação aos horizontes de predição diferentes. Ou seja, o mesmo modelo que apresentou bons resultados para a predição de 12 meses também apresentou bons resultados para 1 mês.

É muito importante que os profissionais de saúde utilizem a mesma ferramenta para visualização dos dados preditivos. Visto que eles não são técnicos, apenas usuários do sistema.

É muito importante desenvolver arquitetura de software com foco na automatização de processos. Pois possibilita melhora contínua nos modelos preditivos, em que o cientista de dados poderá acrescentar novas técnicas de predição de forma mais rápida. E, dessa forma, isso facilita a integração com o sistema de telemedicina STT/SC.

Diante das predições para o conjunto de teste de 2019, podemos confirmar um crescimento anual da demanda dos exames de teledermatologia. E, com base nesses resultados, o prognóstico de crescimento continuará para 2020, como visto nos componentes de visualização do *Saiku*.

Com base no trabalho desenvolvido, os métodos com abordagem clássica precisaram de maior conhecimento matemático sobre séries temporais, o que não ocorreu com a abordagem de aprendizado. Porém, as redes neurais, que apesar de serem modelos de caixa preta, apresentaram resultados equiparáveis aos modelos de suavização exponencial e ARIMA.

TRABALHOS FUTUROS

a) Acrescentar um *dashboard* como alternativa de visualização conjunta de dados históricos e dados preditivos, como a ferramenta *Apache Superset*[5], que integra nativamente o *Apache Druid*.

[5] https://github.com/apache/incubator-superset

b) Criar um ambiente integrado do *Jupyter* com o sistema STT/SC para resolver problemas de ciência de dados. Centralizar as fontes de informações de diversas bases de dados por meio de um banco de dados que trabalhe com um grande volume de informações para o processamento estatístico. Como a adição da ferramenta *Apache Spark*[6] na arquitetura do presente trabalho.

c) Acrescentar extensões de visualização de dados conectados nativamente ao *Apache Druid*, como: *Metatron*[7], *Turnilo*[8], *Apache Superset*[9] e *Imply*[10].

d) Incrementar outras ferramentas de *notebook* para fazer análise de dados de forma simples e rápida, como o caso do *Apache Zeppelin*[11]. Que internamente já possui ferramenta que agenda a execução de *notebooks* para a automatização dos treinamentos e geração dos dados.

[6] https://spark.apache.org/

[7] https://github.com/metatron-app/metatron-discovery

[8] https://github.com/allegro/turnilo

[9] https://github.com/apache/incubator-superset

[10] https://imply.io/

[11] https://zeppelin.apache.org/

REFERÊNCIAS

ABEDJAN, Ziawasch et al. Data Science in Healthcare: Benefits, Challenges and Opportunities. In: **Data Science for Healthcare: Methodologies and Applications**. Edição: Sergio Consoli, Diego Reforgiato Recupero e Milan Petković. Cham: Springer International Publishing, 2019. P. 3–38. ISBN 978-3-030-05249-2. DOI: 10.1007/978-3-030-05249-2_1. Disponível em: <https://doi.org/10.1007/978-3-030-05249-2_1>. Citado na p. 21.

AGGARWAL, Charu C. **Neural Networks and Deep Learning. A Textbook**. [S.l.]: Springer, 2018. ISBN 978-3-319-94463-0. Citado na p. 55.

BECKHAUSER, Eduardo. **UMA METODOLOGIA PARA A ESTRUTURAÇÃO DE LAUDOS TEXTUAIS: COM FOCO NA ROTINA DE LAUDOS EM SISTEMAS DE TELEMEDICINA**. 2019. Diss. (Mestrado) – Universidade Federal de Santa Catarina, Florianópolis, Brasil. Citado na p. 25.

BISHOP, Christopher M. **Pattern Recognition and Machine Learning**. Hardcover. [S.l.]: Springer, 2011. ISBN 0387310738,9780387310732. Citado na p. 53.

BOX, George E. P. et al. **Time Series Analysis: Forecasting and Control**. 5. ed. [S.l.]: Wiley, 2015. (Wiley Series in Probability and Statistics). ISBN 1118675029,9781118675021. Citado na p. 47.

CHNITI, Ghassen; BAKIR, Houda; ZAHER, Hédi. E-commerce Time Series Forecasting Using LSTM Neural Network and Support Vector Regression. In: PROCEEDINGS of the International Conference on Big Data and Internet of Thing. London, United Kingdom: ACM, 2017. (BDIOT2017), p. 80–84. ISBN 978-1-4503-5430-1. DOI: 10.1145/3175684.3175695. Disponível em: <http://doi.acm.org/10.1145/3175684.3175695>. Citado na p. 61.

CHOLLET, François. **Deep Learning with Python**. [S.l.]: Manning, 2018. ISBN 9781617294433. Citado na p. 50.

CONSOLI, Sergio; RECUPERO, Diego Reforgiato; PETKOVIĆ, Milan. **Data Science for Healthcare: Methodologies and Applications**. 1st ed. [S.l.]: Springer International Publishing, 2019. ISBN 978-3-030-05248-5,978-3-030-05249-2. Citado na p. 22.

DOUGLAS C. MONTGOMERY CHERYL L. JENNINGS, Murat Kulahci. **Introduction to Time Series Analysis and Forecasting**. 2. ed. [S.l.]: Wiley, 2015. (Wiley Series in Probability and Statistics). ISBN 1118745116,9781118745113. Citado nas pp. 33, 34.

EREN, H.; WEBSTER, J.G. **The E-Medicine, E-Health, M-Health, Telemedicine, and Telehealth Handbook (Two Volume Set)**. [S.l.]: Taylor & Francis, 2015. ISBN 9781482236552. Disponível em: <https://books.google.com.br/books?id=jOjNoQEACAAJ>. Citado na p. 21.

GOPINATH REBALA AJAY RAVI, Sanjay Churiwala. **An Introduction to Machine Learning**. [S.l.]: Springer, 2019. ISBN 978-3-030-15729-6. Citado nas pp. 60, 61.

GREGORY W. CORDER, Dale I. Foreman. **Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach**. 1. ed. [S.l.]: Wiley, 2009. ISBN 047045461X,9780470454619. Citado na p. 38.

JEREMY WATT REZA BORHANI, Aggelos Katsaggelos. **Machine Learning Refined: Foundations, Algorithms, and Applications**. [S.l.]: Cambridge University Press, 2016. ISBN 1107123526,9781107123526. Citado na p. 52.

MONTGOMERY, Douglas C.; JENNINGS, Cheryl L.; KULAHCI, Murat. **Instroduction to Time Series Analysis and Forecasting**. Hoboken, New Jersey: Wiley, 2015. P. 643. Citado nas pp. 29, 49.

MORETTIN, P.A.; CASTRO TOLOI, C.M. de. **Análise de séries temporais**. [S.l.]: Edgard Blucher, 2018. (ABE - Projeto Fisher). ISBN 9788521213512. Citado nas pp. 33, 41.

NIELSEN, Aileen. **Practical Time Series Analysis: Prediction with Statistics and Machine Learning**. 1. ed. [S.l.]: O'Reilly Media, 2019. ISBN 1492041653, 978-1492041658. Citado nas pp. 36, 37.

PARMEZAN, Antonio; BATISTA, Gustavo. **Descrição de Modelos Estatísticos e de Aprendizado de Máquina para Predição de Séries Temporais**. [S.l.: s.n.], ago. 2016. DOI: 10.13140/RG.2.2.14938.85440. Citado na p. 38.

PETER J. BROCKWELL, Richard A. Davis (auth.) **Introduction to Time Series and Forecasting**. 3. ed. [S.l.]: Springer International Publishing, 2016. (Springer Texts in Statistics). ISBN 978-3-319-29852-8,978-3-319-29854-2. Citado na p. 35.

ROSENBLATT, F. The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. **Psychological Review**, p. 65–386, 1958. Citado na p. 56.

SALDIVA, Paulo Hilário Nascimento; VERAS, Mariana. Gastos públicos com saúde: breve histórico, situação atual e perspectivas futuras. pt. **Estudos Avançados**, scielo, v. 32, p. 47–61, abr. 2018. ISSN 0103-4014. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-40142018000100047&nrm=iso>. Citado na p. 21.

SAVARIS, A. et al. Integrating a PACS Network to a Statewide Telemedicine System: A Case Study of the Santa Catarina State Integrated Telemedicine and Telehealth System. In: 2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS). [S.l.: s.n.], jun. 2017. P. 356–357. DOI: 10.1109/CBMS.2017.128. Citado na p. 25.

SILVEIRA BUENO, Rodrigo De losso da. **Econometria de Séries Temporais**. [S.l.]: Cengage Learning, 2011. P. 360. ISBN 9788522111572. Citado nas pp. 33, 41.

SOUZA INÁCIO, Andrei de. **Estratégia de Recuperação e Análise de Informações Epidemiológicas com Visualização Georreferenciada sobre Dados Médicos Heterogêneos**. 2016. Diss. (Mestrado) – Universidade Federal de Santa Catarina, Florianópolis, Brasil. Citado na p. 25.

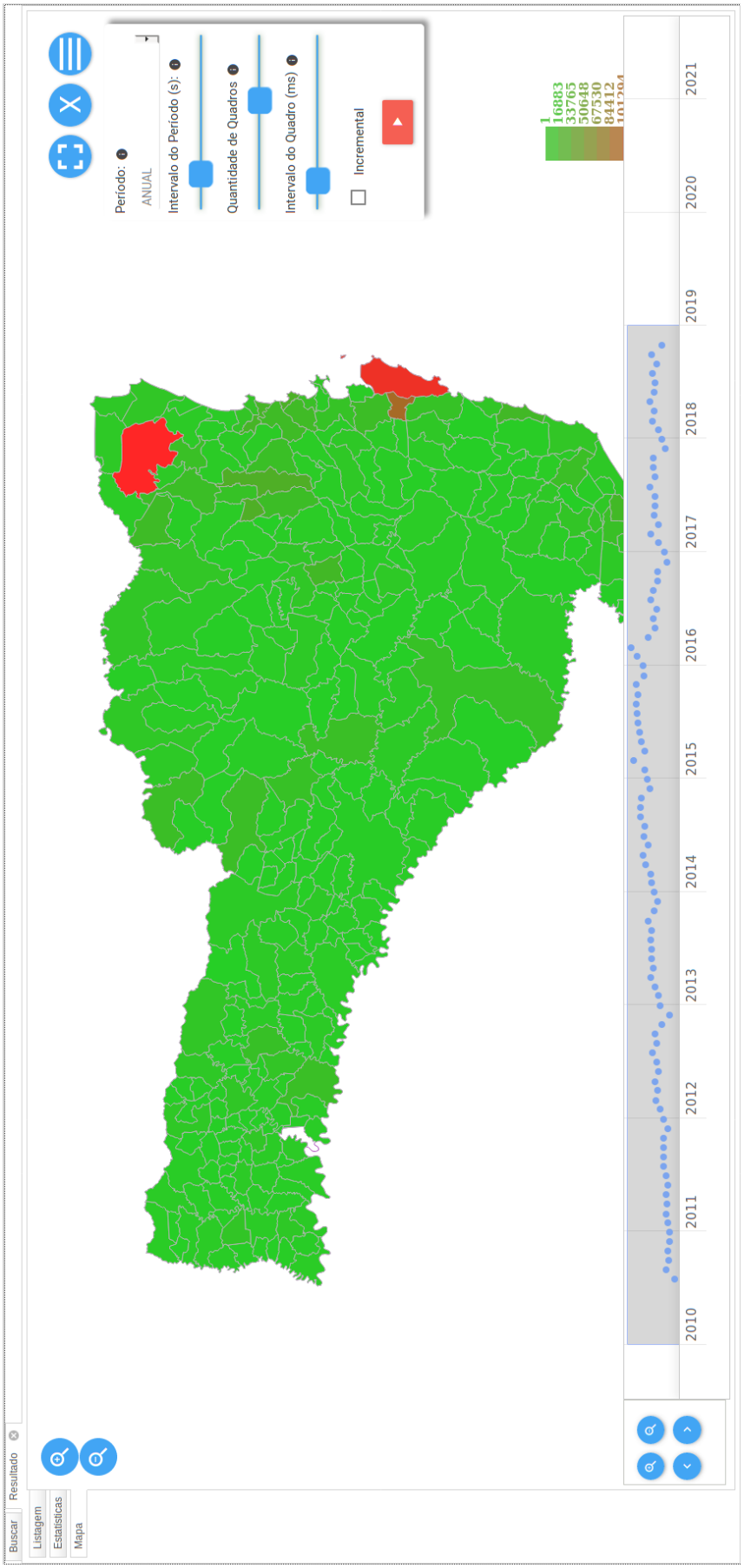
SOYER, H.P. et al. **Telemedicine in Dermatology**. [S.l.]: Springer Berlin Heidelberg, 2011. ISBN 9783642208010. Disponível em: <https://books.google.com.pr/books?id=fskuB4T-yWwC>. Citado na p. 21.

WALLAUER, J. et al. Building a National Telemedicine Network. **IT Professional**, v. 10, n. 2, p. 12–17, mar. 2008. ISSN 1941-045X. DOI: 10.1109/MITP.2008.23. Citado na p. 21.

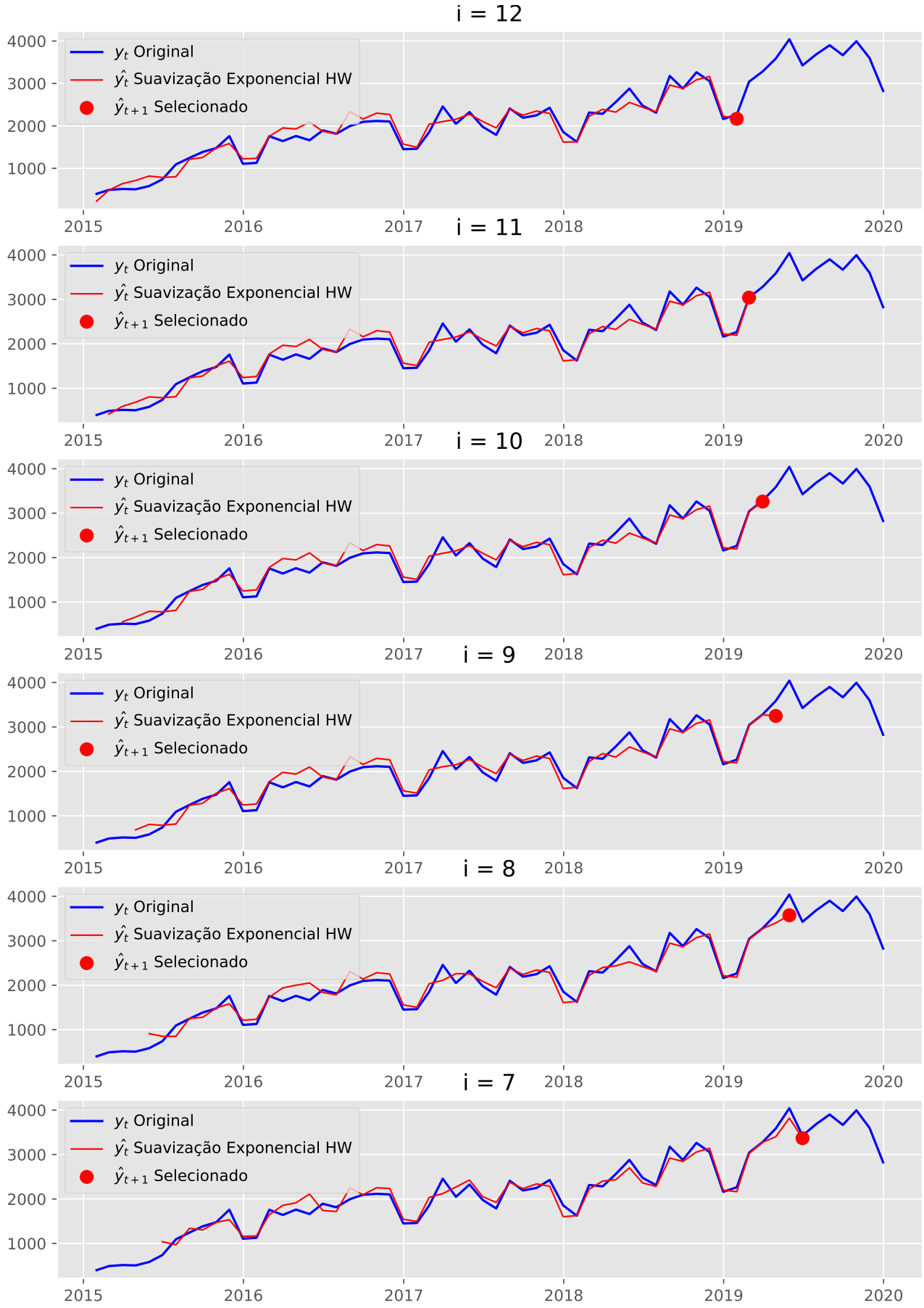
WANGENHEIM, A. von; NUNES, D. Holthausen. **Direct Impact on Costs of the Teledermatology-Centered Patient Triage in the State of Santa Catarina: Analysis of the 2014-2018 data**. [S.l.], jun. 2018. Citado na p. 25.

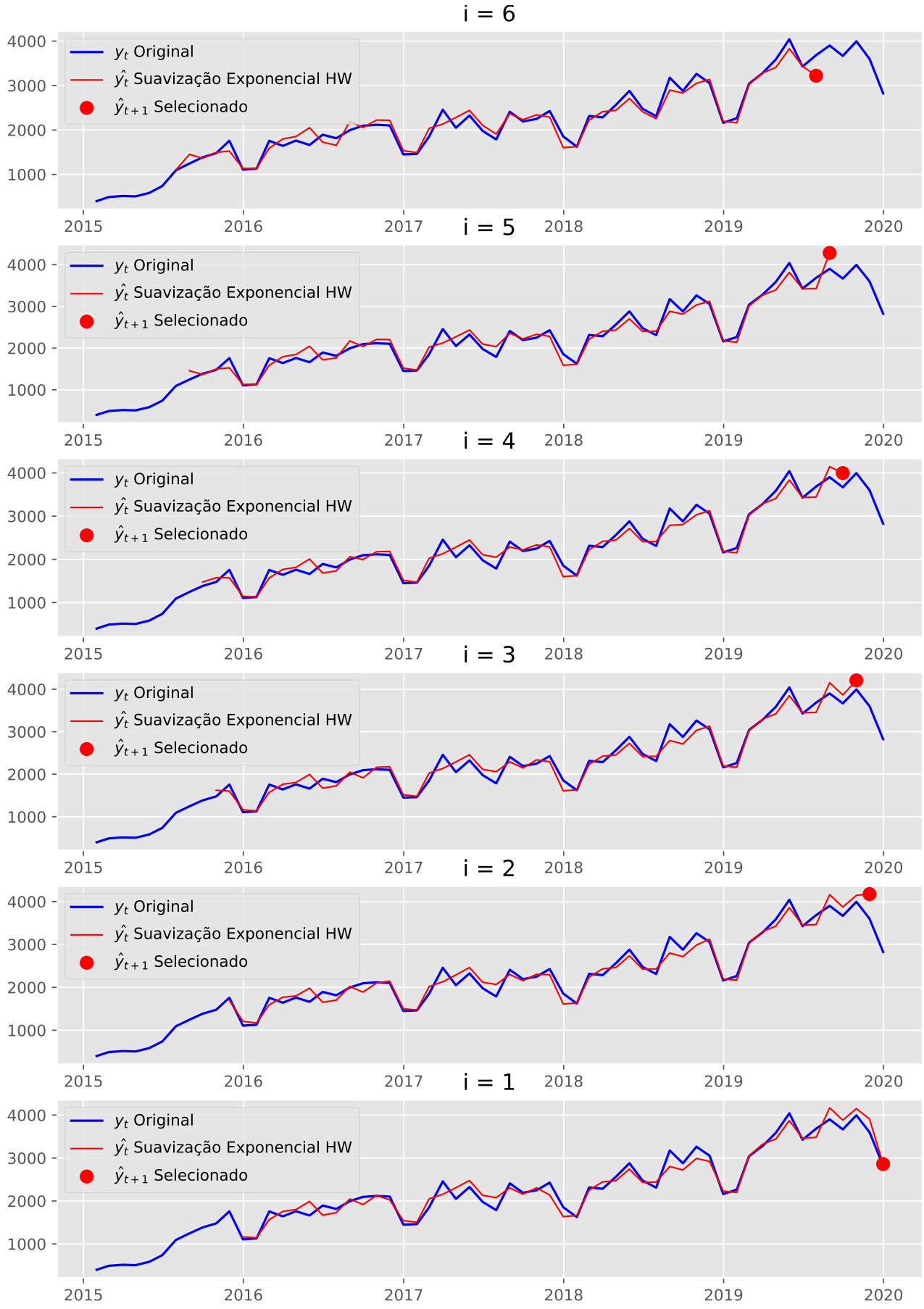
Anexos

ANEXO A – IMAGEM AMPLIADA DO MÓDULO GISTELEMED DO STT/SC


Período:
ANUAL
Intervalo do Período (s):
Quantidade de Quadros
Intervalo do Quadro (ms)
Incremental
1
16883
33765
50648
67530
84412
101294
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
Buscar
Resultado
Listagem
Estatísticas
Mapa

ANEXO B – ALGORITMO DE TREINAMENTO APLICADO AO MÉTODO HW PARA $h = 1$


i = 12
i = 11
i = 10
i = 9
i = 8
i = 7
y_t Original
$\hat{y}_t$ Suavização Exponencial HW
$\hat{y}_{t+1}$ Selecionado
4000
3000
2000
1000
2015
2016
2017
2018
2019
2020


i = 6
i = 5
i = 4
i = 3
i = 2
i = 1
y_t Original
$\hat{y}_t$ Suavização Exponencial HW
$\hat{y}_{t+1}$ Selecionado
4000
3000
2000
1000
2015
2016
2017
2018
2019
2020