



**Universidade Federal da Bahia
Escola Politécnica
Colegiado do Curso de Eng. Elétrica**



Bernardo Perrone De Menezes Bulcão Ribeiro

Previsão de Irradiância Solar a partir de Dados Meteorológicos Utilizando Técnicas de Aprendizado de Máquina

Orientadora: Profa. Dra. Luciana Martinez

Salvador-BA – Brasil
2022

Bernardo Perrone De Menezes Bulcão Ribeiro

**Previsão de Irradiância Solar a partir de Dados
Meteorológicos Utilizando Técnicas de Aprendizado de
Máquina**

Trabalho apresentado ao Curso de Graduação em Engenharia Elétrica da Universidade Federal da Bahia como parte dos requisitos para a obtenção do grau de Engenheiro(a) Eletricista.

Orientadora: Profa. Dra. Luciana Martinez

Salvador-BA – Brasil

2022

Bernardo Perrone De Menezes Bulcão Ribeiro

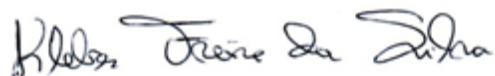
Previsão de Irradiância Solar a partir de Dados Meteorológicos Utilizando Técnicas de Aprendizado de Máquina

Trabalho apresentado ao Curso de Graduação em Engenharia Elétrica da Universidade Federal da Bahia como parte dos requisitos para a obtenção do grau de Engenheiro(a) Eletricista.

Trabalho aprovado. Salvador-BA – Brasil, 2022:

Luciana Martinez

Orientadora: Profa. Dra. Luciana Martinez



Prof. Dr. Kleber Freire da Silva

Karla Patrícia Santos Oliveira Rodriguez Esquerre

Profa. Dra. Karla Patrícia Santos Oliveira
Rodriguez Esquerre

Salvador-BA – Brasil
2022

Este trabalho é dedicado a todas as pessoas que se esforçam para criar um mundo melhor.

Agradecimentos

Agradeço à minha família e sobretudo meus pais, por todo apoio, suporte e educação que sempre me foram dados. Aos amigos, que marcaram e estiveram presentes durante toda minha trajetória de vida.

Agradeço também a universidade pública, e em especial a UFBA e seus professores e funcionários, por terem me proporcionado uma educação gratuita e de altíssimo nível, além de inestimáveis experiências de vida. Aos meus colegas de curso, em especial aos que iniciaram a jornada junto comigo, conhecidos como “Sobreviventes”. Agradeço também aos colegas Thiago e Renan, que tiveram participação fundamental na minha formação como engenheiro eletricista e compuseram comigo numerosas equipes e duplas “dinâmicas”.

Agradeço aos professores do curso que, durante todo o percurso acadêmico, transferiram as habilidades e conhecimentos que compuseram a minha formação. Em especial, agradeço encarecidamente à minha orientadora, Luciana Martinez, por todo suporte, disponibilidade e atenção durante toda a síntese deste trabalho final.

*“Nada tem tanto poder para ampliar a mente que a capacidade de investigar sistemática e
realisticamente tudo o que é observável na vida.”*
(Marco Aurélio)

Resumo

A potência de saída das usinas de geração fotovoltaica depende diretamente da irradiação solar no nível do solo, sendo esta dotada de volatilidade devido à natureza estocástica da cobertura das nuvens, gases e níveis de aerossol na atmosfera. A variabilidade e intermitência da energia injetada na rede no horizonte intradia geram um desafio para o atendimento do equilíbrio da oferta e demanda de energia, sugerindo que o desenvolvimento de modelos de previsão de irradiação solar intradia pode ser de altíssimo valor para operadores e participantes do mercado. Este trabalho pretende o desenvolvimento de uma ferramenta computacional para previsão de irradiação solar horária intradia a base de algoritmos de Aprendizado de Máquina a partir de entradas de medição de irradiação solar horária e variáveis meteorológicas de modelos numéricos de previsão. Iniciou-se com uma fundamentação teórica explorando todos os tópicos necessários para o entendimento da solução proposta. Em seguida, os dados de interesse foram extraídos do modelo numérico MERRA-2 e de uma estação meteorológica do Instituto Nacional de Meteorologia, sendo então explorados e devidamente tratados. Propôs-se um modelo de *benchmark* implementando uma solução trivial, denominada de persistência e os dados foram estruturados e condicionados para um problema de Aprendizado de Máquina supervisionado, com codificação de variáveis cíclicas, normalização de dados, separação em conjuntos de treino, teste e validação e por fim a geração de sequências temporais para cada algoritmo de aprendizado. Foram então propostos 6 modelos, baseados em três arquiteturas de redes neurais artificiais, *Random Forest*, *Support Vector Regression* e *Gradient Tree Boosting*, cujos hiperparâmetros foram otimizados utilizando o conjunto de validação. Finalmente, os resultados foram gerados para um cenário de testes, permitindo a identificação do modelo de *Random Forest* e da arquitetura de rede neural *Encoder-Decoder Long Short-Term Memory* como os mais performantes, e também a verificação das limitações e possibilidades para aperfeiçoamento dos resultados, atendendo o objetivo de desenvolvimento de modelos capazes de auxiliar na tomada de decisão referente aos dados de irradiação solar intradia.

Palavras-chave: Previsão de Irradiância. Energia Solar. Aprendizado de Máquina. Redes Neurais Artificiais. Ciência de Dados. Análise de Dados.

Abstract

Power output from PV systems is directly dependent on the solar irradiance reaching the surface, which is volatile due to the stochastic nature of cloud cover, gasses, and aerosol levels in the atmosphere. Therefore, the variability and intermittence of the injected energy to the grid in the intra-day range creates a challenge for the supply and demand of energy, suggesting that the development of intra-day solar irradiance models could be extremely valuable for energy operators and market players. This academic work aims to develop software for hourly intra-day solar irradiance based on Machine Learning algorithms using solar irradiance measurements and meteorological variables generated by numerical weather prediction models as inputs. Firstly, a theoretical base was built exploring all key topics needed and used throughout this work. Then, input data was gathered from the numerical weather model MERRA-2 and a local weather station from the National Institute of Meteorology of Brazil, explored, and adequately treated. A persistence model was proposed as a benchmark, implementing a trivial solution. Moreover, input data was structured and conditioned to be used in a supervised machine learning algorithm, encoding cyclical features, normalizing data, creating train, test, and validation sets, and ultimately generating temporal sequences suited for each approach. Afterward, 6 models were proposed, implementing 3 neural network architectures and also Random Forest, Support Vector Regression and Gradient Tree Boosting, whose hyperparameters were then optimized using the validation set. Finally, every model was tested following a test scenario, where the Random Forest model and the Encoder-Decoder neural network model showed the best results overall. Although some limitations and potential improvements were identified, it could be concluded that the overall goal was reached, generating informational models capable of supporting decision-making dependent on intra-day solar irradiance.

Keywords: Solar Irradiance Forecasting. Solar Energy. Machine Learning. Artificial Neural Networks. Data Science. Data Analysis.

Listas de ilustrações

Fig. 1 – O presente e futuro dos sistemas elétricos	27
Fig. 2 – Esquemático do Funcionamento de uma Célula Fotovoltaica	30
Fig. 3 – Perfil da Irradiância Solar ao longo do dia	31
Fig. 4 – Dados de Micro e Mini-Geração Distribuída no Brasil	32
Fig. 5 – Exemplo de Piranômetro	33
Fig. 6 – Lucro Trimestral por Ação da Johnson & Johnson	34
Fig. 7 – Resumo da relação entre Horizontes, Modelos e Atividades	37
Fig. 8 – A etapa de <i>Feature Engineering</i> no fluxo de trabalho do Aprendizado de Máquina	38
Fig. 9 – Comparação entre modelos polinomiais para um problema gerando <i>overfitting</i> , <i>underfitting</i> e sem estes problemas	43
Fig. 10 – Aplicação do <i>Dropout</i> numa rede neural	44
Fig. 11 – Modelo não-linear de um neurônio artificial	46
Fig. 12 – Esquemático de uma rede <i>feedforward</i> multicamada totalmente conectada	47
Fig. 13 – SVM de uma dimensão (linear)	51
Fig. 14 – Gráficos dos dados brutos do MERRA-2 e INMET	57
Fig. 15 – Dados de irradiância solar após tratamento	58
Fig. 16 – Gráficos dos dados após tratamento em um recorte de 1 mês	59
Fig. 17 – Diagrama de Caixa das Variáveis	60
Fig. 18 – Correlação entre as Variáveis	61
Fig. 19 – Gráficos de Dispersão entre as entradas e a irradiação solar	61
Fig. 20 – Gráfico de Dispersão entre os alvos e as previsões de persistência	62
Fig. 21 – Gráfico de linha para os alvos e previsões de persistência durante 5 dias .	62
Fig. 22 – Gráfico de linha a <i>feature</i> usando o número cardinal da hora	63
Fig. 23 – Gráfico de dispersão do cosseno e seno da hora	64
Fig. 24 – Diagrama temporal dos conjuntos da geração de sequência para LSTM .	66
Fig. 25 – Diagrama dos conjuntos da geração de sequência para algoritmos tradicionais	67
Fig. 26 – Diagrama da Arquitetura LSTM Comum (<i>Vanilla</i>)	71
Fig. 27 – Diagrama da Arquitetura <i>Encoder-Decoder LSTM</i>	72
Fig. 28 – Diagrama da Arquitetura <i>Encoder-Decoder CNN-LSTM</i>	73
Fig. 29 – Disposição do repositório do projeto na plataforma <i>GitHub</i>	75
Fig. 30 – <i>Loss</i> com a evolução do treinamento da rede LSTM comum	78
Fig. 31 – <i>Loss</i> com a evolução do treinamento da rede <i>Encoder-Decoder LSTM</i> .	78
Fig. 32 – <i>Loss</i> com a evolução do treinamento da rede <i>CNN LSTM Encoder-Decoder</i>	78
Fig. 33 – Previsão de 01/03/2021 08:00 até 03/03/2021 07:00 - UTC-0	79
Fig. 34 – Previsão de 22/12/2021 08:00 até 24/12/2021 07:00 - UTC-0	80

Fig. 35 – Previsão de 17/08/2022 08:00 até 19/08/2022 07:00 - UTC-0	80
Fig. 36 – MAE médio por modelo	81
Fig. 37 – Coeficiente de Determinação R^2 por modelo	82
Fig. 38 – MAE horário por modelo	83
Fig. 39 – Histograma do MAE médio para hora = 12:00 UTC-0	83
Fig. 40 – Histograma do MAE médio para hora = 15:00 UTC-0	84
Fig. 41 – Histograma do MAE médio para hora = 20:00 UTC-0	84
Fig. 42 – Estimativa de densidade kernel para hora = 12:00 UTC-0	85
Fig. 43 – Estimativa de densidade kernel para hora = 15:00 UTC-0	86
Fig. 44 – Estimativa de densidade kernel para hora = 20:00 UTC-0	86

Lista de tabelas

Tab. 1 – Funções de Ativação 48

Lista de abreviaturas e siglas

AR	Modelo Auto Regressivo
ARMA	Modelo Auto-Regressivo e de Média Móvel
ARIMA	Modelo Auto-Regressivo Integrado e de Média Móvel
API	<i>Application Programming Interface</i>
CA	corrente alternada
CNN	<i>Convolutional Neural Network</i>
CC	corrente contínua
EPE	Empresa de Pesquisa Energética
GPUs	Unidades de Processamento Gráfico
GRU	<i>Gated Recurrent Unit</i>
GD	Gradiente Descendente
INMET	Instituto Nacional de Meteorologia
LSTM	<i>Long Short-Term Memory</i>
MPP	Ponto de Maior Potência
MMGD	Micro e Mini-geração Distribuída
MSE	Erro Médio Quadrático
MAE	Erro Médio Absoluto
MAPE	Erro Médio Absoluto Percentual
MASE	Erro Médio Absoluto em Escala
RED	Recursos Energéticos Distribuídos
RNAs	Redes Neurais Artificiais
RNA	Rede Neural Artificial
RNN	Redes Neurais Recorrentes
RMSE	Raiz Quadrada do Erro Médio
SVM	<i>Support Vector Machine</i>
SVR	<i>Support Vector Regression</i>
SGD	Gradiente Descendente Estocástico
TPUs	Unidades de Processamento de Tensor

Listas de símbolos

E_λ	Energia do Fóton
P_{PV}	Potência do Módulo Fotovoltaico
T_{cell}	Temperatura da célula solar
$X_{normalizado}^{(j)}$	Amostra j normalizada
$X_{padronizado}^{(j)}$	Amostra j padronizada
X_{sin}	Transformação de seno para variáveis cíclicas
X_{cos}	Transformação de cosseno para variáveis cíclicas
R^2	Coeficiente de Determinação

Sumário

1	INTRODUÇÃO	27
2	REVISÃO BIBLIOGRÁFICA	29
2.1	Fundamentação Teórica	29
2.1.1	Energia Solar Fotovoltaica	29
2.1.2	Micro e Mini-Geração Distribuída	32
2.1.3	Medição de Irradiância Solar	32
2.1.4	Previsão de Séries Temporais	33
2.1.5	Modelos de Previsão de Geração Fotovoltaica	35
2.1.6	Aprendizado de Máquina	36
2.1.6.1	<i>Feature Engineering</i>	38
2.1.6.1.1	<i>Feature Scaling</i>	39
2.1.6.1.2	Tratamento de Dados Faltantes	39
2.1.6.1.3	Conversão ou Criação de <i>Features</i>	40
2.1.6.2	Seleção do Algoritmo de Aprendizado	40
2.1.6.3	Separação dos Dados para Treino, Validação e Teste	41
2.1.6.4	<i>Underfitting</i> e <i>Overfitting</i>	42
2.1.6.5	Regularização	43
2.1.6.6	Ajuste de Hiperparâmetros	44
2.1.6.7	Redes Neurais Artificiais	45
2.1.6.7.1	Determinação da Arquitetura	46
2.1.6.7.2	Funções de Ativação	47
2.1.6.7.3	Funções de Perda (Custo)	48
2.1.6.7.4	Otimizadores	49
2.1.6.8	<i>Ensemble Learning</i>	49
2.1.6.8.1	<i>Random Forest</i>	50
2.1.6.8.2	<i>Gradient Tree Boosting</i>	50
2.1.6.9	<i>Support Vector Regression (SVR)</i>	51
2.1.7	Métricas de Desempenho de Modelos Preditivos	51
2.2	Software para Machine Learning	53
2.3	Trabalhos Anteriores	53
3	METODOLOGIA E DESENVOLVIMENTO	55
3.1	Definição do Problema	55
3.2	Tratamento dos Dados	55
3.2.1	Definição da localização do alvo	55

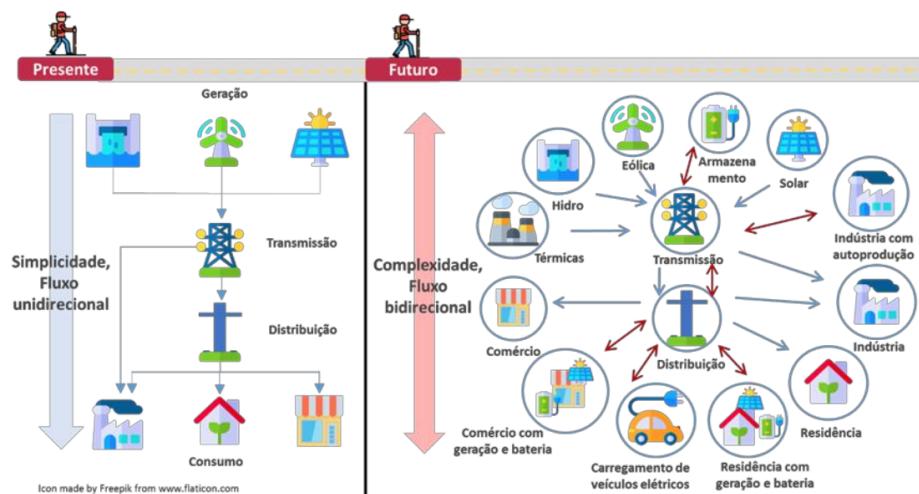
3.2.2	Fonte dos Dados	55
3.2.3	Exploração e pré-tratamento dos dados	56
3.2.3.1	Plot dos Dados Brutos	56
3.2.3.2	Pré-tratamento dos dados	56
3.2.3.3	Análise das Variáveis	58
3.3	Modelo de <i>Benchmark</i> - Persistência	60
3.4	Estruturação dos Dados	62
3.4.1	Codificação de <i>Features</i> Temporais Cíclicas	63
3.4.2	Normalização dos Dados	64
3.4.3	Geração das Sequências de Entradas e Alvos	65
3.4.3.1	Geração das Sequências para Redes Neurais Artificiais Long Short-Term Memory	65
3.4.3.2	Geração das Sequências para algoritmos de aprendizado tradicionais	65
3.4.4	Separação dos dados em conjuntos de treino, validação e teste	66
3.5	Modelo de <i>Random Forest</i>	67
3.5.1	Otimização de Hiperparâmetros do Método <i>Random Forest</i>	67
3.6	Modelo de <i>Support Vector Regression</i>	68
3.6.1	Otimização de Hiperparâmetros do Método <i>Support Vector Regression</i>	69
3.7	Modelo de <i>Gradient Tree Boosting</i>	69
3.7.1	Otimização de Hiperparâmetros do Método <i>Gradient Tree Boosting</i>	69
3.8	Modelos de Redes Neurais Artificiais	70
3.8.1	Arquiteturas de Redes Neurais Implementadas	70
3.8.1.1	Arquitetura LSTM Comum (<i>Vanilla</i>)	70
3.8.1.2	Arquitetura <i>Encoder-Decoder LSTM</i>	71
3.8.1.3	Arquitetura <i>Encoder-Decoder CNN-LSTM</i>	72
3.8.2	Otimização de Hiperparâmetros para os Modelos de Redes Neurais Artificiais	73
3.9	Testes de Inclusão de <i>Features</i> de Entrada	73
3.10	Otimização do tamanho da sequência de medições e previsões numéricas no passado	74
3.11	Cenário de Testes em Produção	74
3.12	Disponibilização em Código-Aberto do Software	75
4	ANÁLISE DE RESULTADOS	77
4.1	Dados do Processo de Treino	77
4.2	Testes e Resultados das Previsões	77
4.2.1	Exemplos de Previsões	79
4.2.2	Métricas de Desempenho	81
4.2.3	Estatísticas de Erro	82
4.3	Discussão dos Resultados	87
5	CONCLUSÃO	89

5.1	Contribuição do Trabalho	90
5.2	Trabalhos Futuros	91
REFERÊNCIAS		93

1 Introdução

O panorama atual do setor energético no Brasil e no resto do mundo é de grandes transformações. Destas, o surgimento das *Smart Grids* e a transformação de consumidores passivos em prossumidores, através da difusão das tecnologias de Recursos Energéticos Distribuídos (RED), contribuem para a descentralização organizacional dos sistemas elétricos, alterando os fluxos de energia e aumentando drasticamente a sua complexidade (EPE, 2019), como mostra a Figura 1.

Fig. 1 – O presente e futuro dos sistemas elétricos



Fonte: (EPE, 2019)

Paralelo aos inúmeros benefícios decorrentes de tal descentralização, surgem também desafios para o planejamento e controle das redes de distribuição de baixa tensão. Nesse contexto, destaca-se a adoção exponencial da micro e minigeração distribuída no Brasil e em especial a tecnologia fotovoltaica distribuída, que em 2019 alcançou a posição de segunda fonte com maior adição de capacidade instalada na matriz energética do País (EPE; MME, 2020).

A potência de saída das usinas solares depende diretamente da irradiação solar no nível do solo, sendo esta bastante volátil a depender das condições de cobertura das nuvens, níveis de aerossol e em menor grau os gases na atmosfera.

A variabilidade e intermitência da energia injetada na rede no curto prazo (intra-hora) por cada ponto de geração solar distribuída resultam em variações importantes nos fluxos de potência e nas tensões, desestabilizando a rede. No horizonte intradia, é importante prever estas variações para atender o equilíbrio de oferta e demanda de energia.

Por fim, o horizonte intra-semana é utilizado para o controle e planejamento das redes de transmissão (DIAGNE et al., 2013). Inequivocamente, gerar previsões precisas quanto a irradiação solar e consequentemente a geração fotovoltaica nos diferentes horizontes temporais é essencial para a integração das plantas solares na rede sem resultar em perda na confiabilidade do sistema.

Para superar esse desafio, o uso de sistemas de controle e planejamento dinâmico em tempo real vem sendo aplicado e estudado em conjunto com as previsões de energia produzida por estas fontes intermitentes. Evidentemente, para que esta abordagem seja bem-sucedida, é preciso que estas previsões sejam suficientemente precisas e robustas para reduzir ao máximo os erros nos diferentes horizontes temporais para os nós do sistema.

Para haver previsões precisas e robustas de energia produzida pela geração solar, a condição *sine qua non* é a existência de modelos e métodos de previsão dos parâmetros de interesse, recorrendo a ferramentas de *Big Data*, Aprendizado de Máquina, Métodos Estatísticos de Previsão de Séries Temporais, Métodos de Imagem e Métodos Numéricos de Meteorologia, alimentados por fluxos de informação em tempo real originados de sensores dinâmicos de geração e consumo, previsões meteorológicas, imagens de satélite, dentre outros (DIAGNE et al., 2013).

Uma classe de métodos que tem recebido bastante atenção nos últimos anos é a das técnicas de Aprendizado de Máquina, sobretudo para o horizonte intradia, particularmente importante para o atendimento da oferta e demanda e pela ativação dos serviços anciares e contratação no curto prazo de unidades de geração (DIAGNE et al., 2013). Sendo um domínioativamente estudado pela comunidade acadêmica, naturalmente novos algoritmos e arquiteturas emergem de forma frequente, provando-se necessário a realização de estudos constantes nos diferentes domínios de aplicação para verificação de potenciais avanços na capacidade preditiva dos modelos. Além disso, o avanço exponencial do poder de processamento das Unidades de Processamento Gráfico (GPUs) e o surgimento das Unidades de Processamento de Tensor (TPUs), circuitos específicos otimizados para o aprendizado de máquina, permitem a criação de modelos cada vez mais robustos, trazendo novamente a necessidade de atualização contínua.

Este trabalho visa o desenvolvimento de uma ferramenta computacional para previsão intradia da irradiação solar horária em uma determinada localidade, aplicando técnicas de Aprendizado de Máquina a partir de entradas de medição de irradiação solar horária e variáveis meteorológicas de modelos numéricos de previsão. Esta ferramenta será disponibilizada em repositório público e poderá ser utilizada por Distribuidoras, Operadoras de Redes Independentes e qualquer outra parte interessada.

2 Revisão Bibliográfica

Nesta seção, será apresentada uma fundamentação teórica dos conceitos que serão abordados neste trabalho, além de uma revisão das obras anteriormente realizadas acerca deste tema.

2.1 Fundamentação Teórica

2.1.1 Energia Solar Fotovoltaica

A Energia Solar Fotovoltaica pode ser definida como a energia obtida através da conversão direta da luz em eletricidade. Esta ocorre por meio da célula fotovoltaica devido ao efeito fotovoltaico. ([IMHOFF, 2007](#))

O efeito fotovoltaico, descoberto pelo físico Francês Alexandre Edmond Becquerel, deriva da excitação de elétrons e outros portadores de carga pela absorção da luz. Toda radiação eletromagnética é composta de partículas chamadas fôtons, carregando quantidades específicas de energia determinadas pelas propriedades espectrais de sua fonte. Os fôtons apresentam uma característica de onda possuindo um comprimento de onda λ que pode ser relacionado com a energia do fôton segundo a equação (2.1), em que h é a constante de Planck e c a velocidade da luz.

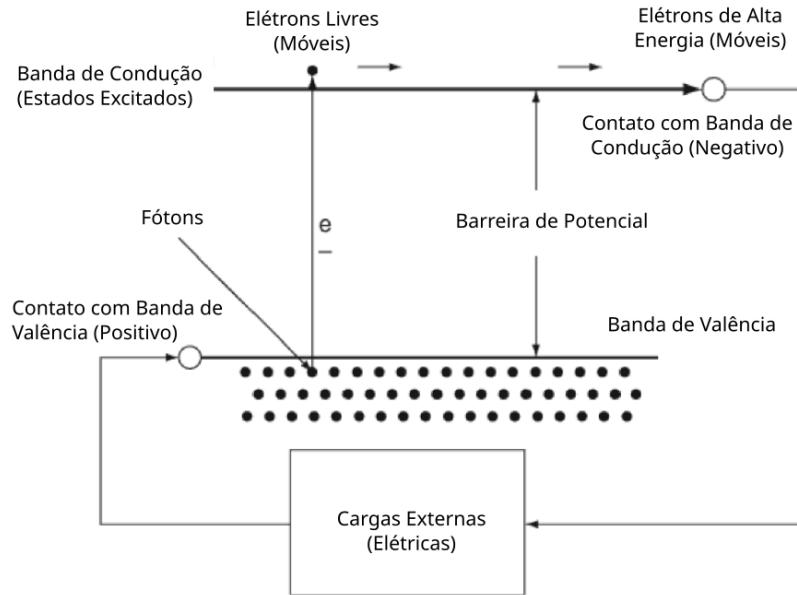
$$E_\lambda = \frac{hc}{\lambda} \quad (2.1)$$

Fôtons que possuem uma energia superior à barreira de excitação entre a banda de valência e banda de condução do semicondutor que compõe as células fotovoltaicas contribuirão para a geração de uma diferença de potencial elétrico. Assim, as células fotovoltaicas conseguem converter energia solar em energia elétrica fazendo-se uso de contatos seletivos gerados por uma junção pn que coleta os elétrons recém-liberados e os direciona para um circuito externo, resultando em uma corrente elétrica. Um esquemático do funcionamento de uma célula fotovoltaica pode ser observado na Figura 2.

No mundo real, um determinado número de células fotovoltaicas são interconectadas e encapsuladas em unidades chamadas módulos fotovoltaicos, sendo então vendidos comercialmente para a construção de usinas de geração. Estes módulos produzem corrente contínua ([CC](#)) sendo geralmente convertida em corrente alternada ([CA](#)) para utilização ou para ser injetada na rede de distribuição.

Os módulos fotovoltaicos são classificados em Watt-picôs, representando a potência que o módulo entregaria para uma carga ideal quando iluminado por uma irradiação

Fig. 2 – Esquemático do Funcionamento de uma Célula Fotovoltaica



Fonte: ([HEGEDUS; LUQUE, 2003](#)) (Traduzido)

solar incidente igual a 1 kW/m^2 com um espectro correspondente a luz solar para uma temperatura de célula de 25° C .

Irradiância solar incidente global designa a densidade de potência sob a forma de radiação eletromagnética que incide sobre uma superfície, medida em W/m^2 , composta pelos componentes de irradiação difusa e direta. Ao integrá-la durante um período, obtém-se a medida de energia chamada irradiação solar incidente, em J/m^2 .

A distribuição de energia do espectro de radiação solar depende da localização geográfica, da hora do dia, do dia do ano, das condições climáticas, da composição da atmosfera, da altitude e de diversos outros fatores ([VILLALVA; GAZOLI, 2012](#)). O perfil da irradiância solar ao longo de um dia, em média, se assemelha a uma curva de sino, como pode ser visto na Figura 3, trazendo variações abruptas devido aos fatores meteorológicos como a presença de nuvens.

Outro fator importante na caracterização da irradiância solar do tempo é o efeito da sazonalidade, que se intensifica a medida que se distânciada da linha do equador. Este fenômeno decorre da maior ou menor duração do dia nas diferentes estações devido à inclinação do eixo de rotação da Terra num ângulo de aproximadamente $23,5^\circ$ em relação ao eixo do movimento da órbita de translação, gerando o desbalanceamento na duração de dias e noites entre hemisférios norte e sul.

O módulo fotovoltaico pode ser modelado como uma fonte linear de potência em relação a irradiância solar e a temperatura ambiente, conforme a equação (2.2), onde

Fig. 3 – Perfil da Irradiância Solar ao longo do dia

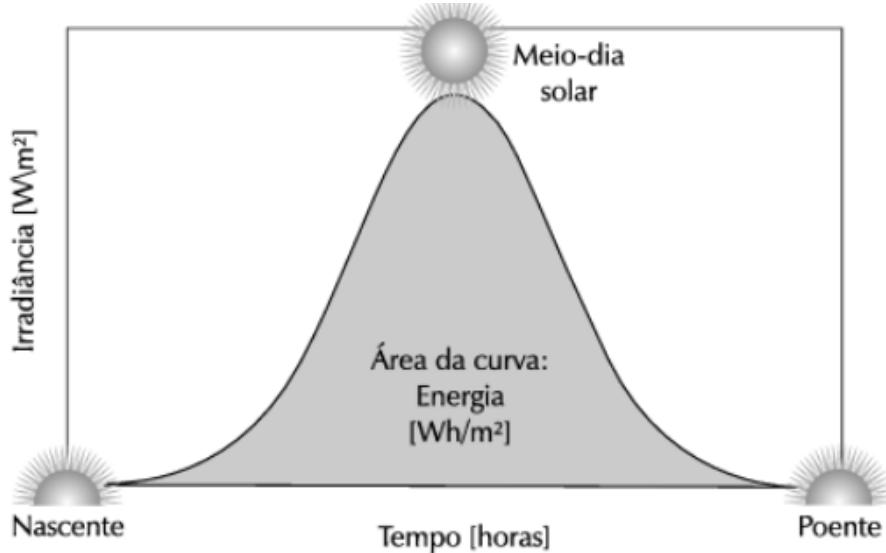


Figura 2.14: Perfil da irradiância solar ao longo de um dia.

Fonte: ([VILLALVA; GAZOLI, 2012](#))

P_{PV} , $P_{PV,STC}$, G_T , γ , T_{cell} , N_{PV} , N_{PVp} são, respectivamente, a potência de saída do gerador no Ponto de Maior Potência (MPP), a potência nominal de saída do gerador no MPP e condições padrões de teste, a irradiância solar global, o coeficiente de temperatura, a temperatura da célula e o número de módulos em série e paralelo ([RIFFONNEAU et al., 2011](#)).

$$P_{PV} = P_{PV,STC} * \frac{G_T}{1000} * [1 - \gamma * (T_{cell} - 25)] * N_{PVs} * N_{PVp} \quad (2.2)$$

A temperatura da célula solar pode ser estimada por diversos métodos empíricos, desenvolvidos na literatura, como na equação (2.3), onde T_{amb} , $NOCT$ são, respectivamente, a temperatura do ar ambiente e a temperatura nominal de operação da célula ([RIFFONNEAU et al., 2011](#)).

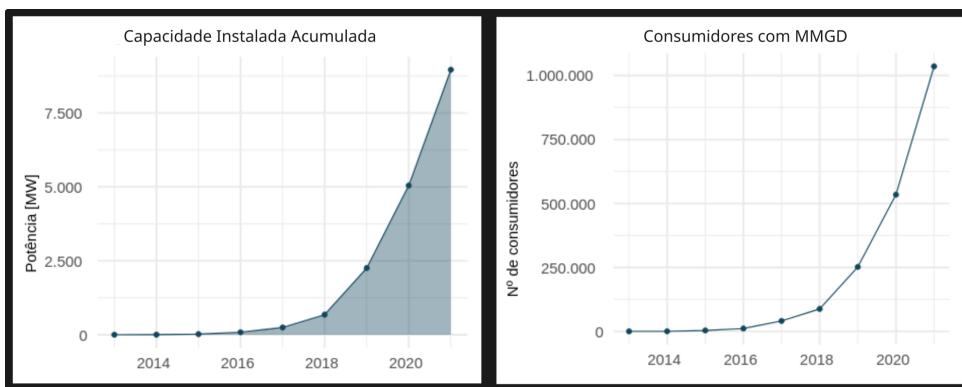
$$T_{cell} = T_{amb} + \frac{G_T}{800} * (NOCT - 20) \quad (2.3)$$

Evidentemente, a potência de saída das usinas solares pode ser determinada através da temperatura ambiente e da irradiância solar, sendo os outros parâmetros determinados pelo projeto ou pela folha de dados fornecida pelo fabricante.

2.1.2 Micro e Mini-Geração Distribuída

A Micro e Mini-geração Distribuída (**MMGD**) constitui-se na produção de energia elétrica a partir de pequenas usinas de geração utilizando fontes renováveis ou cogeração qualificada, conectadas à rede de distribuição pelas unidades consumidoras. Diferencia-se microgeração de minigeração pela potência instalada, onde o primeiro se refere a usinas com potência instalada menor ou igual a 75kW e o último a usinas com potência instalada maior que 75kW e menor ou igual a 3MW, para usinas hídricas, ou 5MW para as outras modalidades (ANEEL, 2020).

Fig. 4 – Dados de Micro e Mini-Geração Distribuída no Brasil



Fonte: (EPE, 2022)

Os dados fornecidos pela Empresa de Pesquisa Energética (**EPE**) revelam o crescimento exponencial da capacidade instalada acumulada e do número de consumidores com **MMGD** no Brasil, impulsionado pelo avanço das usinas fotovoltaicas no setor residencial, comercial, industrial e rural, principalmente nas modalidades de autoconsumo remoto e geração na própria unidade consumidora (EPE, 2022). Em especial, observa-se a previsão para 2025 de 8 GW de capacidade instalada em **MMGD**, representando 4,6% de toda a capacidade nacional, conforme o plano decenal de energia 2030 (EPE; MME, 2020).

2.1.3 Medição de Irradiância Solar

A medição da irradiância solar é realizada por instrumentos capazes de medi-la em incidência normal ou sobre uma superfície horizontal, proveniente de todo um hemisfério. De forma geral, tais instrumentos registram valores em intervalos espectrais definidos, em bandas largas ou estreitas (YAMASOE; IWABE, 2006).

A grandeza medida depende do instrumento utilizado, podendo ser a da radiação solar (0,3 a 4,0 μ), radiação de onda longa ou terrestre (4 a 100 μ), radiação total (soma da radiação solar e de onda longa), medidas em bandas espectrais e medidas em ângulos sólidos pequenos.

As estações automáticas do Instituto Nacional de Meteorologia ([INMET](#)) são dotadas de Piranômetros capazes de realizar a medição e acompanhamento horário da Radiação Solar Global. Estas são tomadas em intervalos de coleta na ordem de segundos, porém são disponibilizadas integradas durante o período de uma hora. Portanto, é possível acompanhar de forma aberta toda a radiação global incidente durante o intervalo de uma hora para toda a rede de estações automáticas do [INMET](#). Um exemplo de Piranômetro adotado em estações automáticas pode ser visto na Figura 5.

Fig. 5 – Exemplo de Piranômetro



Fonte: ([Eppley Lab, 2022](#))

2.1.4 Previsão de Séries Temporais

Uma série temporal é um conjunto de observações realizadas em tempos específicos. Em geral, trabalha-se com séries temporais discretas, onde estes tempos formam um conjunto discreto, como, por exemplo, quando as observações são tomadas em intervalos de tempo fixos, em oposição direta às séries temporais contínuas ([BROCKWELL; DAVIS, 2015](#)).

Alguns exemplos de séries temporais são:

- Estimativas trimestrais do PIB;
- Valores horários da Irradiância Solar;
- Taxa de natalidade anual do Brasil;

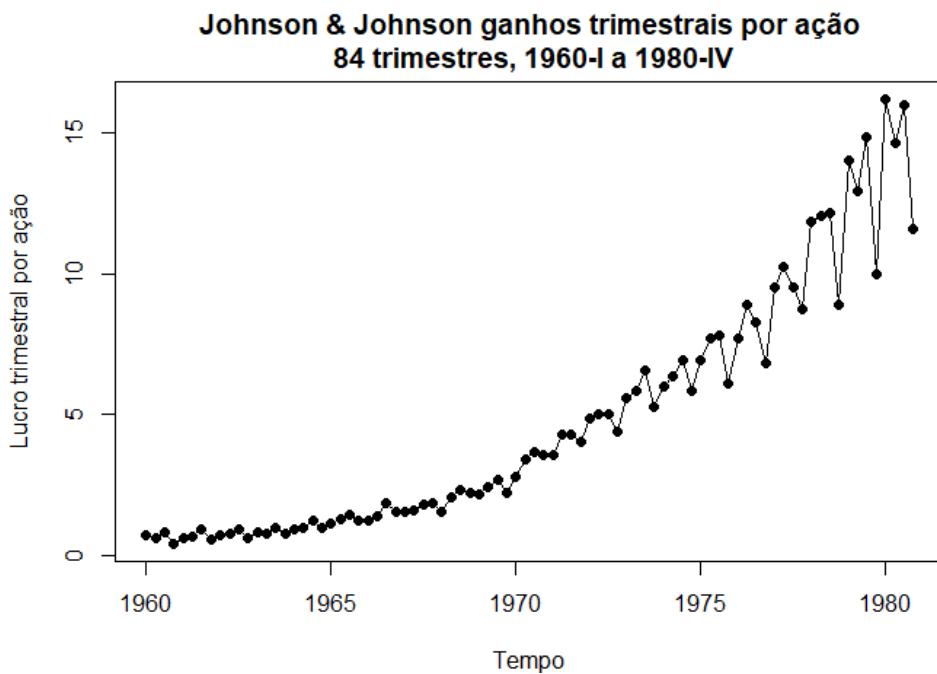
- Cotação de fechamento diária das ações da Petrobras;
- Produção mensal de automóveis por uma determinada empresa.

Seja uma série temporal $z(t_1), z(t_2), \dots, z(t_n)$, observada nos instantes t_1, t_2, t_3 , podemos nos interessar por ([MORETTIN; TOLOI, 1981](#)):

1. Investigar o mecanismo gerador da série temporal;
2. Fazer previsões de valores futuros da série;
3. Descrever o comportamento da série, verificando a existência de tendências, ciclos e variações sazonais, construção de histogramas, diagramas e gráficos;
4. Procurar periodicidade nos dados, usando, por exemplo, ferramentas como a análise espectral.

Séries temporais estacionárias se desenvolvem no tempo ao redor de uma média constante, enquanto séries temporais não-estacionárias podem apresentar componentes de tendências e ciclos, inclusive possuindo períodos sendo estacionária e outros sendo não-estacionárias. Um exemplo de uma série temporal apresentando componentes claras de tendência e ciclo pode ser vista na Figura 6.

Fig. 6 – Lucro Trimestral por Ação da Johnson & Johnson



Fonte: ([PÉREZ, 2022](#))

Existem diversas técnicas amplamente estudadas na literatura visando realizar inferências acerca do futuro de séries temporais. Para isso, estabelece-se um modelo hipotético probabilístico representando os dados, que seja adequado ao problema tratado. Após essa definição, estimam-se os parâmetros e verifica-se a qualidade das suas previsões, realizando ajustes num ciclo de refinamento.

2.1.5 Modelos de Previsão de Geração Fotovoltaica

Objetivando possuir a maior previsibilidade possível nos diferentes horizontes temporais, diversos modelos de previsão de geração fotovoltaica foram desenvolvidos na literatura, utilizados pelos operadores de rede para evitar instabilidades e permitir o devido planejamento de oferta e demanda pela ativação dos serviços anciliares, superando os desafios gerados pelo crescimento exponencial da capacidade solar distribuída. Na seção [2.3](#), serão discutidos as principais obras realizadas acerca deste tema.

Segundo [Inman, Pedro e Coimbra \(2013\)](#), até o momento, sistemas de previsão de geração fotovoltaica de alta fidelidade que funcionem para diferentes microclimas permanecem bastante raros. A dificuldade reside na grande complexidade gerada pelo efeito não-linear do movimento das nuvens na irradiação solar ao nível do solo. Entretanto, avanços consideráveis foram obtidos nos últimos anos pela combinação e aperfeiçoamento de metodologias de diferentes áreas do conhecimento, como a física atmosférica, aprendizado de máquina, métodos de satélite e imagem e sensoriamento remoto.

De acordo com [Sudharshan et al. \(2022\)](#), problemas de balanceamento entre geração, produção e demanda são comuns na integração de fontes renováveis na rede. A todo momento, a energia gerada no sistema deve ser balanceada com uma carga correspondente, devendo a planta ser dimensionada para lidar com mudanças, distúrbios e faltas na rede de energia, fornecendo eletricidade de forma contínua para os clientes. Métodos ou modelos de previsão de irradiação solar são necessários para controlar as perdas e variações de tensão e garantir a transmissão de energia de forma satisfatória. Previsões precisas da potência de saída de plantas fotovoltaicas garantem a economicidade e segurança da operação da rede elétrica, sendo também utilizadas para estimativa de reservas, negociações de mercado, agendamento de operações e redução dos custos da produção de eletricidade.

Existem três principais horizontes temporais para os modelos de previsão solar: intra-hora, intradia e previsões de múltiplos dias, servindo para diferentes propósitos. No horizonte intra-hora, as previsões serão utilizadas nas operações visando a estabilização da rede. Na resolução intradia, as previsões permitem o atendimento da oferta e demanda pela ativação dos serviços anciliares e contratação no curto prazo de unidades de geração. Por fim, as previsões de múltiplos dias auxiliarão no planejamento da transmissão, nos mercados de energia dos dias seguintes e na otimização de ativos.

A previsão da irradiação horizontal global é o passo mais importante na obtenção da geração fotovoltaica futura e esta pode ser utilizada para obtenção da potência de geração dos módulos fotovoltaicos, conforme visto na subseção 2.1.1. Os modelos de irradiação podem ser caracterizadas em relação às variáveis de entrada utilizadas, o que definirá também os seus horizontes de tempo (DIAGNE et al., 2013). Assim, tem-se a seguinte classificação:

- Modelos estatísticos diretos baseados em medidas de irradiação, aplicados para horizontes de tempo entre 5 minutos até 6 horas. Exemplos: Modelo Auto Regressivo (**AR**), Modelo Auto-Regressivo e de Média Móvel (**ARMA**), Modelo Auto-Regressivo Integrado e de Média Móvel (**ARIMA**).
- Redes Neurais Artificiais (**RNAs**) e outras técnicas de Aprendizado de Máquina para estimativa da irradiação solar, aplicadas para o mesmo horizonte de tempo do item anterior.
- Modelos baseados na informação do movimento nas nuvens, como vetores de movimento via imagens de satélite ou até métodos utilizando câmeras ao nível do solo direcionadas ao céu. O primeiro apresenta ótimos resultados para a faixa de resolução temporal entre 30 minutos e 6 horas, sendo o último mais apropriado para previsões intra-hora.
- Modelos baseados em previsões meteorológicas numéricas, apropriados para horizontes temporais maiores, a partir das 4 a 6 horas.

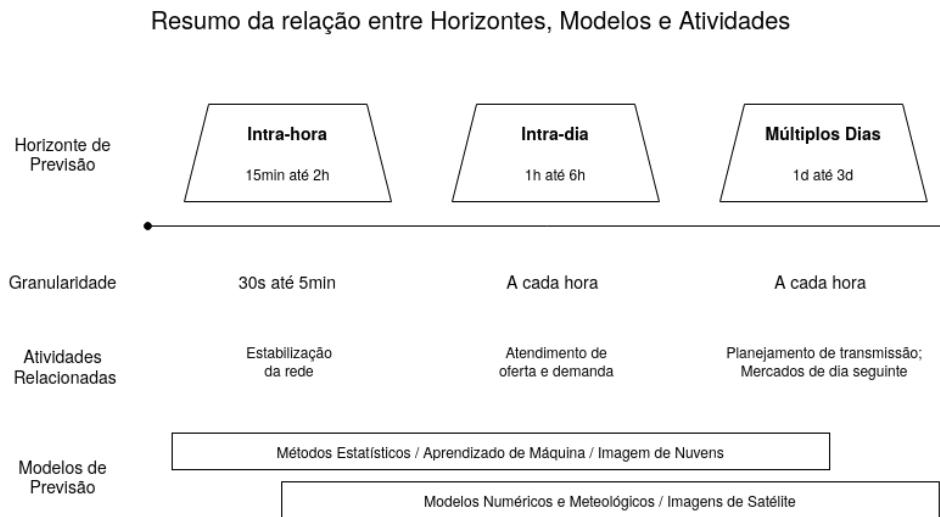
No diagrama 7, é feito um resumo da relação entre os diferentes horizontes temporais, modelos e atividades.

2.1.6 Aprendizado de Máquina

O aprendizado de máquina (ou *machine learning*) se refere ao subdomínio da Ciência da Computação responsável por desenvolver algoritmos que encontram sua utilidade a partir de uma coleção de exemplos de algum fenômeno (dados) (BURKOV; LUTZ, 2019). O aprendizado de máquina também pode ser definido como a capacidade de sistemas para o aprendizado a partir de dados de treino relacionados a um problema específico visando a automação do processo analítico de construção de modelos e consequente resolução de tarefas associadas, como a previsão de um determinado valor ou probabilidade de acontecimento de algum evento. (JANIESCH; ZSCHECH; HEINRICH, 2021).

Ainda segundo Burkov e Lutz (2019), o aprendizado de máquina pode ser classificado em 4 categorias para indicar as diferentes modalidades de aprendizado:

Fig. 7 – Resumo da relação entre Horizontes, Modelos e Atividades



Fonte: Elaboração Própria. Inspirado em ([DIAGNE et al., 2013](#))

- Aprendizagem Supervisionada: O conjunto de dados é inteiramente rotulado, isto é, cada elemento do conjunto possui um rótulo representando a saída real do modelo. Este rótulo pode pertencer a um conjunto finito de classes, ser um valor real ou até mesmo uma estrutura mais complexa, como sequências, vetores, árvores, dentre outros. O objetivo dos modelos de aprendizagem supervisionada é o de gerar, a partir de um vetor de entradas X, uma previsão quanto a saída Y, baseado nos dados de entrada e os rótulos utilizados durante o treinamento.
- Aprendizagem Não-Supervisionada: Neste caso, o conjunto de dados não possui rótulos. Objetiva criar um modelo que recebe um vetor de entrada e o transforma em outro vetor ou valor que poderá ser utilizado para resolver algum problema.
- Aprendizagem Semi-Supervisionada: O conjunto de dados possui elementos rotulados e não rotulados, onde a quantidade de não-rotulados é normalmente substancialmente superior ao de rotulados. Possui o mesmo objetivo que a aprendizagem supervisionada, não havendo descarte de informação, já que os dados não-rotulados serão utilizados, ajudando o modelo a gerar melhores computações e previsões.
- Aprendizagem por Reforço: Cria-se um ambiente de recompensas e punições em que a máquina se insere e consegue detectar o estado deste como um vetor de entradas. Podendo executar ações, a máquina é recompensada segundo as regras de jogo estabelecidas. O objetivo do Aprendizado por Reforço é aprender uma diretriz ou política, definidas como uma função que recebe o vetor de entrada do estado do ambiente e retorna a ação ideal a ser executada naquele estado, maximizando a recompensa média esperada.

Divide-se também os problemas de aprendizado de máquina em dois grupos baseados na categoria de saídas esperadas pelo modelo. Quando este deve classificar uma saída em duas (binário) ou mais classes finitas, tem-se um problema de classificação. Por outro lado, quando as saídas esperadas são valores contínuos, tem-se um problema de regressão.

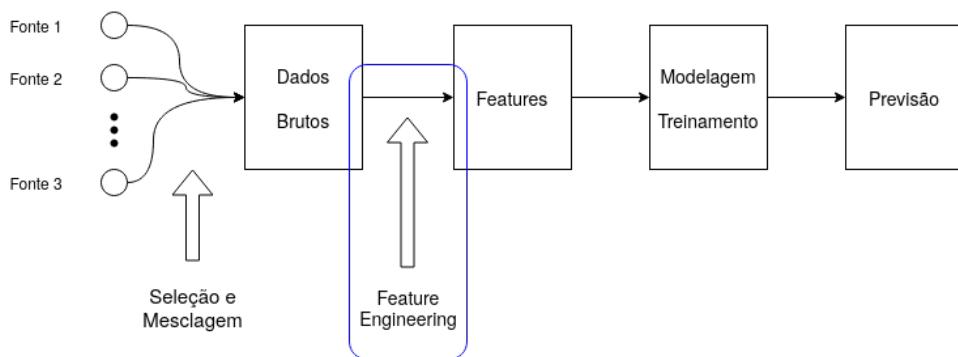
Neste trabalho, apenas a categoria de aprendizagem supervisionada para regressão será abordada, haja visto que cada observação do conjunto de dados possuirá um rótulo correspondente e procura-se prever uma saída contínua.

2.1.6.1 Feature Engineering

O Aprendizado de Máquina ajusta modelos matemáticos aos dados para entendimento de seu comportamento e gerar previsões. Esses modelos recebem como entrada representações numéricas de algum aspecto dos dados brutos, denominadas de *features*. *Feature Engineering* é o agrupamento de técnicas que permitem a extração destas representações dos dados brutos em formatos adequados para o modelo de Aprendizado de Máquina aplicado ([ZHENG; CASARI, 2018](#)).

É uma etapa de grande importância no desenvolvimento de modelos de Aprendizado de Máquina e consenso que a maioria do tempo gasto nestas empreitadas é em atividades de *Feature Engineering* e limpeza de dados ([ZHENG; CASARI, 2018](#)). A etapa de *Feature Engineering* no fluxo de trabalho do Aprendizado de Máquina pode ser visto na Figura 8.

Fig. 8 – A etapa de *Feature Engineering* no fluxo de trabalho do Aprendizado de Máquina



Fonte: Elaboração Própria. Inspirado em ([ZHENG; CASARI, 2018](#))

O número de *features* e as informações que estas agregam para o problema em questão possuem relação direta com o sucesso no treinamento dos modelos. Se não há *features* informativas o suficiente, o modelo não conseguirá obter uma previsibilidade satisfatória. Por outro lado, uma quantidade excessiva destas (principalmente se forem pouco informativas) pode acarretar custos e dificuldades de treinamento, inclusive impactando diretamente no desempenho dos modelos através do fenômeno denominado de “A Maldição da Alta Dimensionalidade”. Um artigo fazendo uma análise teórica e empírica

deste fenômeno para o aprendizado de classificadores supervisados pode ser visto em [Debie e Shafi \(2019\)](#).

2.1.6.1.1 Feature Scaling

O processo de *Feature Scaling* refere-se à transformação dos dados para atender a uma determinada escala, facilitando o aprendizado em diversos modelos de Aprendizado de Máquina. As variações mais utilizadas são a Normalização e a Padronização, listadas abaixo ([BURKOV; LUTZ, 2019](#)):

- Normalização: Objetiva converter os dados de uma *feature* para caber dentro de um intervalo definido, como $[-1, 1]$ e $[0, 1]$. Seu cálculo é realizado através da equação (2.4), para cada dado, sendo $X_{\text{normalizado}}^{(j)}$ o valor normalizado, $X_{\text{original}}^{(j)}$ o valor original, $\max^{(\text{feature})}$ e $\min^{(\text{feature})}$ os valores máximo e mínimo, respectivamente, da *feature* no conjunto.

$$X_{\text{normalizado}}^{(j)} = \frac{X_{\text{original}}^{(j)} - \min^{(\text{feature})}}{\max^{(\text{feature})} - \min^{(\text{feature})}} \quad (2.4)$$

- Padronização: Processo no qual os dados de uma *feature* são transformados para possuir as propriedades de uma distribuição normal padrão com $\mu = 0$ e $\sigma = 1$, média e desvio padrão, respectivamente. A sua fórmula pode ser vista na equação (2.5), onde $X_{\text{padronizado}}^{(j)}$, $X_{\text{original}}^{(j)}$, $\mu^{(\text{feature})}$, $\sigma^{(\text{feature})}$ são, respectivamente, o valor padronizado, o valor original, a média e o desvio padrão da *feature* no conjunto.

$$X_{\text{padronizado}}^{(j)} = \frac{X_{\text{original}}^{(j)} - \mu^{(\text{feature})}}{\sigma^{(\text{feature})}} \quad (2.5)$$

O uso da técnica de normalização é aconselhado quando não se conhece a distribuição da variável e não há *outliers*, já que na normalização a presença de valores muito discrepantes afetará de forma determinante a escala resultante. Por outro lado, a padronização é recomendada para variáveis com distribuição normal, lidando bem com a presença de *outliers*, mas gerando perda de informação quanto a dispersão das amostras quando a distribuição real difere da normal.

2.1.6.1.2 Tratamento de Dados Faltantes

No Aprendizado de Máquina, é comum se deparar com *features* possuindo dados faltantes, pelos motivos mais diversos. Existem algumas estratégias para lidar com esse inconveniente, como detalhado em ([BURKOV; LUTZ, 2019](#)):

- Removendo todas as sequências do conjunto de dados que possuam dados faltantes, principalmente se este conjunto for grande o suficiente a tal ponto em que algumas amostras removidas não impactarão no seu treinamento.

- Usando um algoritmo de aprendizado que funcione com dados faltantes ou trate-os de alguma forma.
- Usando uma técnica de imputação de dados faltantes. Por exemplo, pode-se substituir os dados faltantes pela média da *feature* no conjunto de dados. Novamente, a melhor técnica depende bastante do tipo de problema e da *feature* com dados faltantes.

É difícil determinar qual técnica para lidar com dados faltantes funcionará melhor, variando muito de problema em problema e tornando benéfico realizar testes com cada metodologia para verificar qual produz os melhores resultados.

2.1.6.1.3 Conversão ou Criação de *Features*

Frequentemente são aplicadas técnicas de conversão ou criação de *features* a partir do conjunto de dados disponíveis para o problema, objetivando facilitar ou melhorar o aprendizado dos modelos. Por exemplo, pode-se criar uma *feature* categórica binária a partir de algumas condições sobre os dados, ou, como será visto neste trabalho, criar funções cossenoidais representando a posição do dia numa circunferência representando a sazonalidade anual.

Este tipo de transformação encontra sua utilidade em resultados empíricos que demonstram a redução de erros, otimização do processo de treino e uma melhora global no desempenho de modelos de Aprendizado de Máquina, quando técnicas de conversão ou criação de *features* são aplicadas nos conjuntos de dados (LIU; MOTODA, 1998).

2.1.6.2 Seleção do Algoritmo de Aprendizado

A escolha do algoritmo ou modelo de aprendizado é uma etapa relevante para o desenvolvimento. Uma abordagem inicial consiste em testar o treinamento de todos os possíveis modelos e comparar os resultados, o que se mostra inviável para projetos com limitações de tempo e prazo. Outra abordagem consiste na determinação de certos critérios de projeto, permitindo a redução do número de algoritmos candidatos para a solução de um determinado problema. Em (BURKOV; LUTZ, 2019), é apresentado uma série de questões que auxiliam nesta escolha, como:

- Explicabilidade: Alguns modelos podem possuir um nível de complexidade muito alto, dificultando a sua explicação e o entendimento de seu funcionamento para uma audiência não-técnica. Por exemplo, pode-se citar as RNAs e os métodos de *Ensemble Learning*.
- Treinamento fora-da-memória e dentro da memória: A depender do tamanho do conjunto de dados, é possível que não haja memória RAM suficiente para o seu

armazenamento integral. Nestes casos específicos, deve-se escolher algoritmos que permitem o aprendizado incremental, que se realiza com apenas uma fração do conjunto de dados por vez.

- Número de *features* e amostras: Alguns métodos de aprendizado, como **RNAs** e *Gradient Boosting* tem capacidade de lidar com milhões de *features* e um grande número de amostras, enquanto outros possuem limites muito menores, como a *Support Vector Machine (SVM)*.
- *Features* categóricas ou numéricas: Alguns algoritmos não são capazes de treinar diretamente com *features* categóricas, sendo necessário realizar uma conversão para uma variável numérica.
- Não-linearidade: Para problemas que podem ser modelados por modelos lineares, métodos como **SVM** com *kernel* linear ou regressão logística e linear podem ser boas escolhas. Por outro lado, metodologias como **RNAs** profundas ou *Ensemble Learning* são ótimos candidatos para problemas não lineares.
- Velocidade de treinamento: Quando o tempo do projeto é limitado, pode-se classificar os algoritmos de aprendizado quanto a sua velocidade de treino. **RNAs** costumam ser os mais lentos, enquanto modelos mais simples como regressões logísticas e métodos baseados em árvores de decisão são os mais rápidos. Alguns, como *Random Forest*, podem se beneficiar de CPUs com múltiplos núcleos, aumentando a sua velocidade de treinamento.
- Velocidade de previsão: Análogo ao item anterior, a velocidade da geração de previsão pode ser uma restrição de projeto, quando estes são finalizados e lançados num ambiente de produção. Pode-se citar o exemplo de um modelo treinado para frear um carro na detecção de um obstáculo via Aprendizado de Máquina. Neste caso, SVMs, regressões lineares e logísticas e alguns tipos de **RNAs** são mais velozes, enquanto **RNAs** recorrentes e profundas, KNN e *Ensemble Learning* são mais demoradas.

2.1.6.3 Separação dos Dados para Treino, Validação e Teste

Para o desenvolvimento de modelos, é comum uma separação do conjunto de dados em 3 subconjuntos: Treino, Validação e Teste. O conjunto de treino é aquele que fornecerá as amostras para realização do treinamento, sendo normalmente aquele com maior tamanho, com sugestões de diversos profissionais variando entre 50% e 90%. Os conjuntos de validação e teste costumam possuir tamanhos similares, menores que o de treino. Os algoritmos de aprendizado não utilizam amostras destes dois conjuntos, que servem a outras funções auxiliares ao treino (**BURKOV; LUTZ, 2019**).

Quando modelos são construídos, algumas metodologias seriam capazes de simplesmente memorizar as entradas às saídas correspondentes do conjunto de dados, sendo excelente na previsão do conjunto de dados de treinamento, porém sem utilidade num cenário de produção, para novas entradas. Como o objetivo é prever justamente entradas jamais vistas, é importante separar uma parte de dados que não será utilizado durante o treinamento. O conjunto de validação será utilizado para escolha do algoritmo de aprendizado e ajuste de hiperparâmetros enquanto o conjunto de teste será utilizado para avaliação final do modelo, antes de lançar em produção.

2.1.6.4 *Underfitting e Overfitting*

Underfitting é a inabilidade do modelo de prever bem os alvos do conjunto de dados de treinamento. Logicamente, essa situação representa uma grave ineficiência no processo de desenvolvimento de um modelo, pois não há a extração máxima de informação preditiva no aprendizado. Os motivos mais comuns para o *underfitting* são:

- O modelo é simples demais para o problema. Exemplo: Regressão Linear para um problema de natureza não-linear.
- As *features* coletadas ou desenvolvidas não são suficientemente informativas.

As soluções para *Underfitting* estão na tentativa de modelos mais complexos e na obtenção de *features* mais informativas (apesar de não ser possível para alguns casos).

Overfitting é o fenômeno que ocorre quando modelos possuem uma alta performance para previsão de alvos no conjunto de treino e um baixo desempenho na mesma tarefa para o conjunto de teste ou de validação. Esse é o resultado, por exemplo, de um modelo que simplesmente memoriza sob forma de dicionário cada valor de entrada e saída de um conjunto de treino. Existem diversos motivos que podem gerar este fenômeno, sendo os mais importantes ([BURKOV; LUTZ, 2019](#)):

- O modelo é excessivamente complexo para a aplicação modelada, como, por exemplo, tentar empregar uma Rede Neural Artificial ([RNA](#)) muito profunda para um problema relativamente simples.
- Um número relativamente grande de *features* para a quantidade de amostras disponíveis.

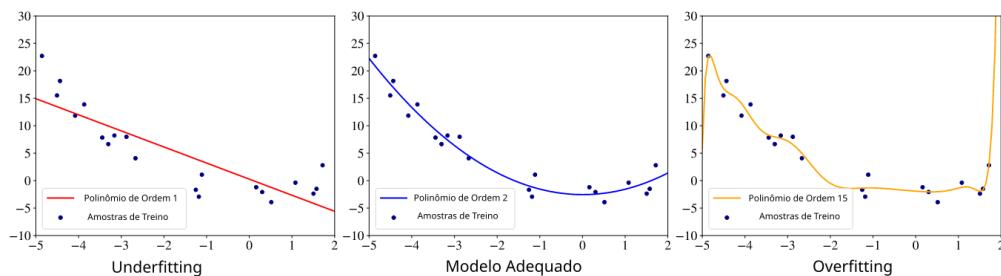
Existem diversas soluções para o problema do *Overfitting*, como:

1. Tentativa de modelos mais simples.

2. Redução do número de *features*.
3. Obtenção de mais dados para treinamento.
4. Aplicação da técnica da regularização (será detalhado na seção a seguir).

Uma imagem adaptada de [Burkov e Lutz \(2019\)](#) comparando um modelo polinomial para ordens que geram *overfitting*, *underfitting* e uma modelagem sem estes problemas pode ser visto na Figura 9.

Fig. 9 – Comparação entre modelos polinomiais para um problema gerando *overfitting*, *underfitting* e sem estes problemas



Fonte: ([BURKOV; LUTZ, 2019](#)) (Adaptado)

2.1.6.5 Regularização

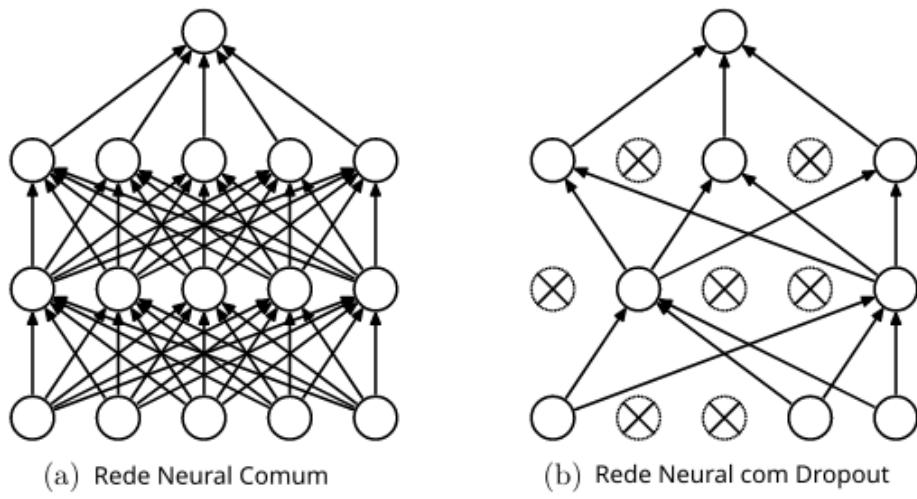
Todos os modelos são capazes de gerar *overfitting*, e sua intensidade depende da quantidade de amostras, número de dimensões das *features* e da complexidade do modelo. Regularização é um termo que agrupa todos os métodos que forçam o aprendizado a gerar um modelo menos complexo, gerando um maior viés em troca de uma redução significativa na variância ([BURKOV; LUTZ, 2019](#)).

Ainda segundo [Burkov e Lutz \(2019\)](#), uma das formas mais difundidas para este tipo de tarefa é a de atribuir penalidades em um termo da função do modelo, proporcional a sua complexidade, tornando efetivamente o treino mais custoso e, portanto, reduzindo a sua capacidade de gerar *overfitting*. Exemplos clássicos deste tipo de regularização são implementadas na regularização L1 e L2. A primeira gera modelos esparsos, trazendo a maioria dos parâmetros para próximo de zero com efeito de uma seleção de *features*, bastante útil quando há necessidade de melhor explicabilidade do modelo. Por outro lado, a regularização L2 maximiza o desempenho do modelo nos conjuntos de teste e validação, possuindo também a vantagem de ser diferenciável.

Outra técnica de regularização bastante aplicada em problemas de **RNAs** é o *Dropout*. Consiste em forçar a rede a desconsiderar temporariamente alguns neurônios, com

suas conexões, escolhidos de forma randômica segundo uma função probabilística. Isto faz com que a cada rodada de treinamento, seja obtido um modelo diferente, tornando a memorização de pares entrada-resposta muito mais custoso computacionalmente, além de filtrar o ruído de amostragem para conjuntos de treino com tamanho limitado (SRIVASTAVA et al., 2014). Um esquemático demonstrando a aplicação do *Dropout* numa RNA pode ser vista na Figura 10.

Fig. 10 – Aplicação do *Dropout* numa rede neural



Fonte: (SRIVASTAVA et al., 2014) (Adaptado)

Outros métodos com efeitos regularizadores são o *Batch Regularization*, onde é feita a normalização das entradas das camadas para cada mini-conjunto de treino (IOFFE; SZEGEDY, 2015); *Data Augmentation*, onde são aplicadas técnicas para aumentar a diversidade do conjunto de dados disponível, como a rotação de imagens (SHORTEST; KHOSHGOFTAAR, 2019) e finalmente *Early Stopping*, para RNAs, consistindo em monitorar o erro de validação durante o treinamento e automaticamente suspendê-lo quando algum critério de parada é atendido, como, em um exemplo mais simples, quando o erro de validação deixa de reduzir após um certo número de rodadas de treino (PRECHELT, 2012).

2.1.6.6 Ajuste de Hiperparâmetros

O ajuste de hiperparâmetros é uma das últimas etapas no desenvolvimento de modelos de Aprendizado de Máquina. Nela, os parâmetros específicos de cada algoritmo que não são otimizados pelo processo de treinamento são ajustados para conferir o melhor resultado, utilizando como base de comparação o desempenho sobre o conjunto de validação. Alguns exemplos de hiperparâmetros são o número de estimadores de uma *Random Forest* e a velocidade de aprendizado das RNAs.

Uma técnica amplamente difundida para realização deste ajuste é o *Grid Search*, no qual se definem possíveis valores discretos para cada parâmetro ajustável, normalmente em escala logarítmica, sendo feitos testes para cada possível combinação. Evidentemente, esta técnica pode se tornar muito demorada caso o número de hiperparâmetros ajustáveis do modelo seja alto, potencializado por conjuntos de dados grandes (BURKOV; LUTZ, 2019).

Outras técnicas mais eficientes em tempo e processamento são a *Random Search* e a Otimização de Hiperparâmetros Bayesiano. O primeiro difere de *Grid Search* na definição de uma distribuição de probabilidade para cada hiperparâmetro, sendo testado um número pré-definido de valores de forma randômica. Na Otimização de Hiperparâmetros Bayesiano, a escolha do hiperparâmetro da próxima iteração da otimização depende do resultado passado, limitando o custo computacional.

Além destas, há também técnicas baseadas em gradiente e a otimização evolucionária que não serão abordados neste trabalho.

2.1.6.7 Redes Neurais Artificiais

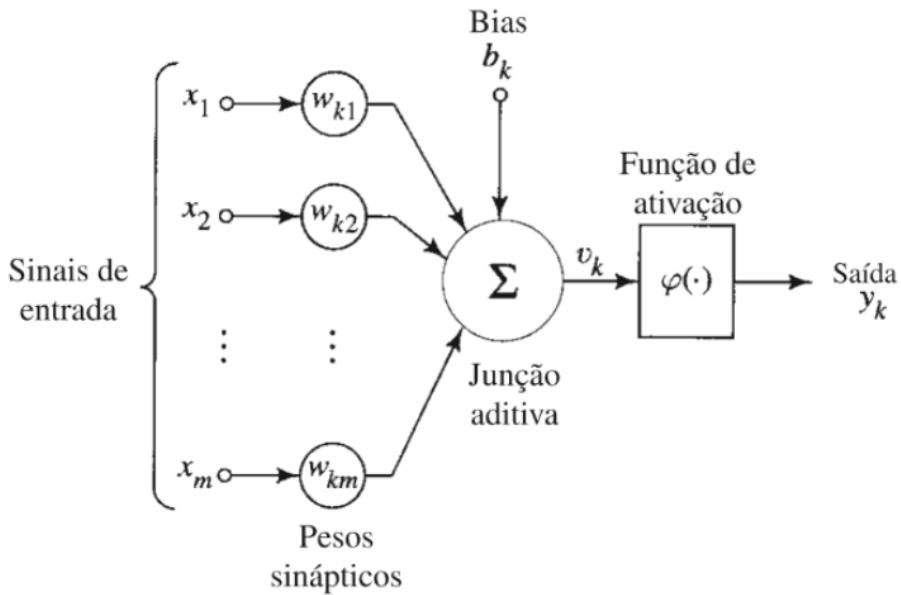
A [RNA](#) é uma técnica de aprendizado de máquina geralmente utilizada no contexto de aprendizagem supervisionada. Seu nome e sua estrutura são inspiradas pelo cérebro humano, na tentativa de imitar como os neurônios biológicos se comunicam entre si.

Do ponto de vista computacional, o neurônio é um elemento capaz de realizar o processamento de uma ou mais entradas e gerar uma saída correspondente. O neurônio artificial é uma estrutura lógico-matemática visando replicar a forma, comportamento e funções de um neurônio biológico. Assim, o neurônio artificial é a unidade fundamental processadora de informação em uma [RNA](#) (FURTADO, 2019). Um modelo de neurônio artificial pode ser observado na Figura 11.

Uma RNA é constituída por neurônios articiais interconectados dispostos em camadas. Cada neurônio processa suas entradas e gera uma saída, alimentando as entradas de outros neurônios em outras camadas. As diferentes possibilidades na disposição e estrutura das camadas dão origem as diferentes arquiteturas de [RNAs](#) abordadas na literatura. O aprendizado se dá pela otimização dos pesos das entradas e o viés (*bias*) de cada camada, através do mecanismo de Gradiente Descendente ou de outros otimizadores. Essa otimização é realizada através da escolha de um otimizador e de uma função de perda (*Loss Function*), que será minimizada (FURTADO, 2019).

Na Figura 12, pode-se observar uma rede *feedforward* multicamada totalmente conectada, composta por uma camada de entrada, uma camada intermediária e uma camada de saída. Neste caso, as saídas de uma camada anterior alimentam as entradas da camada subsequente ponderadas pelos seus respectivos pesos que serão otimizados durante

Fig. 11 – Modelo não-linear de um neurônio artificial



Fonte: ([HAYKIN; ENGEL, 2001](#))

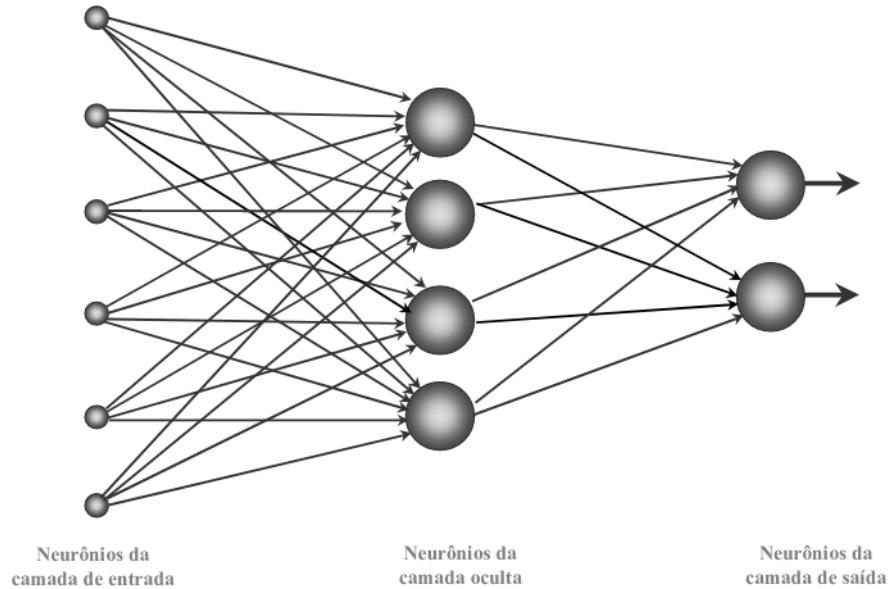
o treinamento. As saídas dos neurônios de uma ou mais camadas são sujeitas a funções de ativação não-lineares, permitindo o aprendizado de um conjunto de dados que não seja linearmente separável.

Uma classe de arquiteturas frequentemente utilizada para o aprendizado em problemas envolvendo a previsão de sequências (como as temporais) é a de Redes Neurais Recorrentes ([RNN](#)). Nesta categoria de problema, a ordem dos elementos na sequência é informativa o suficiente para auxiliar no processo de aprendizado do modelo. Diferentemente da classe de arquiteturas *Feedforward*, as redes recorrentes possuem a habilidade de memorizar o estado de neurônios de elementos passados da sequência, e estes estados são utilizados como entradas de neurônios subsequentes. Exemplos amplamente utilizados deste tipo de arquitetura são as *Long Short-Term Memory* ([LSTM](#)) e *Gated Recurrent Unit* ([GRU](#)) ([BURKOV; LUTZ, 2019](#)).

2.1.6.7.1 Determinação da Arquitetura

A determinação da arquitetura a ser utilizada segue uma lógica semi-empírica. Ela é feita a partir de conhecimentos empíricos acerca do funcionamento de arquiteturas previamente utilizadas para problemas que compartilham as mesmas características com a função que se está tentando aproximar e também de conhecimentos heurísticos ([FURTADO, 2019](#)).

Fig. 12 – Esquemático de uma rede *feedforward* multicamada totalmente conectada



Fonte: ([FURTADO, 2019](#))

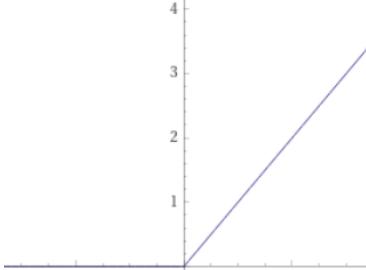
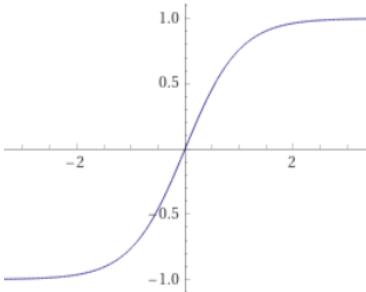
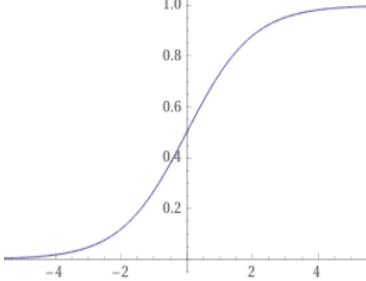
Por exemplo, a última camada deve possuir o mesmo número de neurônios que a quantidade de variáveis que serão preditas. É conhecido também que arquiteturas recorrentes desempenham bem para problemas envolvendo sequências, enquanto redes convolucionais possuem performance superior para problemas de reconhecimento de imagens.

Entretanto, a quantidade de neurônios a ser empregada nas camadas intermediárias e até mesmo o número destas camadas não seguem nenhum teorema ou guia. É padrão a realização da técnica de tentativa e erro, comparando diferentes configurações e selecionando aquela que possui o melhor resultado, baseado no resultado anterior. Um exemplo prático é a adição de uma nova camada ou o aumento do número de neurônios de uma camada intermediária quando a rede é incapaz de generalizar com a configuração previamente testada.

2.1.6.7.2 Funções de Ativação

Funções de ativação possuem papel fundamental no aprendizado via [RNAs](#). Elas são aplicadas entre entrada e saída de um neurônio, adicionando a capacidade de não-linearidade, que sem a sua presença seriam apenas capazes de realizar transformações lineares ([BURKOV; LUTZ, 2019](#)).

As funções de ativação mais utilizadas na indústria são: Unidade Linear Retificada (ReLU), Tangente Hiperbólico e Sigmoidal. As características destas funções de ativação podem ser vistas na tabela 1.

Função	Definição	Gráfico
ReLU	$\sigma(x) = \begin{cases} 0 & \text{se } x < 0 \\ x & \text{se } x \geq 0 \end{cases}$	
Tangente Hiperbólico	$\sigma(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	
Sigmoide	$\sigma(x) = \frac{1}{1 + e^{-x}}$	

Tab. 1 – Funções de Ativação

2.1.6.7.3 Funções de Perda (Custo)

Em sua base, as [RNAs](#) tratam de um problema genérico de otimização, onde uma função de perda (*Loss Function*), ou função de custo é utilizada para comparar os valores previstos e os valores reais e seu valor deverá ser minimizado. Neste sentido, existem mais de 30 funções de perda utilizadas nos mais diversos problemas, a depender do contexto e de suas características ([WANG et al., 2022](#)).

Ainda de acordo com [Wang et al. \(2022\)](#), para problemas de classificação, utilizam-se as funções de perda probabilísticas. Por exemplo, pode ser utilizado como função de perda a Entropia Cruzada Binária (*Binary Cross-Entropy*) para um problema de classificação binária. Caso o problema seja com múltiplas classes, uma abordagem similar seria o uso da função de perda Entropia Cruzada de Categorias (*Categorical Cross-Entropy*).

Por outro lado, para modelos visando a previsão de valores numéricos e não de probabilidades, as funções de perda de regressão devem ser empregadas. Alguns exemplos são o Erro Médio Quadrático ([MSE](#)), Erro Médio Absoluto ([MAE](#)), Perda de Huber e a Similaridade por Cosseno.

2.1.6.7.4 Otimizadores

Durante a construção de um modelo de [RNAs](#), é necessário realizar a escolha de um otimizador. Um otimizador é um algoritmo ou método utilizado para alterar os parâmetros ou a taxa de aprendizagem da RNA para minimizar os valores da sua função de perda.

Os otimizadores são agrupados por suas características. O mais trivial é o Gradiente Descendente ([GD](#)), que faz uso direto da função de perda para encontrar o mínimo. Existem outras opções baseadas no [GD](#), como o Gradiente Descendente Estocástico ([SGD](#)). Outro grupo de otimizadores implementam o conceito de impulso, acelerando a convergência numa determinada direção para redução da alta variância dos modelos anteriores, como o Gradiente Descendente com Impulso. Há também otimizadores que atualizam a taxa de aprendizado a cada iteração, como Adagrad e Adadelta. Por fim, apresentam-se as opções do otimizador RMSProp, que toma uma média exponencial dos gradientes para reduzir a agressividade da taxa de aprendizado do Adadelta e o otimizador Adam, combinando a aprendizado adaptativo do RMSProp com o mecanismo de impulso do [GD](#) ([RUDER, 2016](#)).

Em geral, recomendam-se testes empíricos para cada otimizador, não havendo evidências irrefutáveis para afirmar categoricamente a superioridade de um em relação ao outro ([CHOI et al., 2019](#)), ressaltando uma preferência da indústria pelos otimizadores RMSProp, [SGD](#) e disparadamente, pelo otimizador Adam, apresentando, em geral, o melhor desempenho.

2.1.6.8 Ensemble Learning

Ensemble Learning é um paradigma de aprendizado de máquina que prioriza o treinamento de múltiplos modelos de baixa precisão para futura combinação e geração de um metamodelo de alto desempenho, em oposição ao treinamento de um único modelo.

Uma das grandes vantagens desta abordagem é os modelos serem treinados por algoritmos incapazes de aprender relacionamentos muito complexos, reduzindo drasticamente o tempo de treinamento e de previsão. O algoritmo de aprendizado mais comumente utilizado é a árvore de decisão, e este é treinado por um número pequeno de iterações, resultando em árvores rasas e pouco performantes. A ideia por trás dessa metodologia reside no fato que se essas árvores não são idênticas e possuem uma efetividade superior à tentativa de adivinhar aleatoriamente as saídas, podendo-se obter um modelo de alta performance ao combinar um número grande de árvores de decisão. ([BURKOV; LUTZ, 2019](#)).

Os dois algoritmos mais utilizados de *Ensemble Learning* para problemas de regressão são o *Random Forest* e Gradient Tree Boosting, que serão detalhados a seguir.

2.1.6.8.1 Random Forest

O algoritmo de aprendizado *Random Forest* combina o uso de árvores de decisão e *bagging*, um dos principais paradigmas de *Ensemble Learning*. *Bagging* consiste na criação de cópias dos dados de entrada levemente diferentes de todas as outras, aplicando os algoritmos de aprendizado fraco em cada uma delas para combinação futura.

As *Random Forests* usam uma versão modificada do *Bagging*, no sentido em que o algoritmo de aprendizado seleciona um subconjunto randômico das variáveis de entrada, evitando a correlação entre as árvores. Esse procedimento reduz a variância do modelo final e consequentemente torna o algoritmo bastante resistente ao *Overfitting*, quando o modelo se torna extremamente performante para prever os dados utilizados para treinamento enquanto perde poder de generalização para prever novos dados (BURKOV; LUTZ, 2019).

As etapas do algoritmo de *Random Forest*, de forma simplificada, são as seguintes (SCHONLAU; ZOU, 2020):

1. Um número n de amostras é retirado de forma aleatória do conjunto de treino possuindo N amostras.
2. De forma randômica, são selecionados k *features* das m existentes, onde $k < m$, sendo o valor padrão de $k = \sqrt{m}$.
3. Para cada amostra, é determinada a *feature* preditiva que otimiza o critério de separação escolhido, como, por exemplo, o critério Gini, separando em dois novos nós.
4. Repete-se a partir da etapa 2. até atingir o número mínimo de nós por árvore de decisão individual.
5. É gerada a *Random Forest* repetindo a partir da etapa 1, construindo um número B de árvores de decisão individuais.

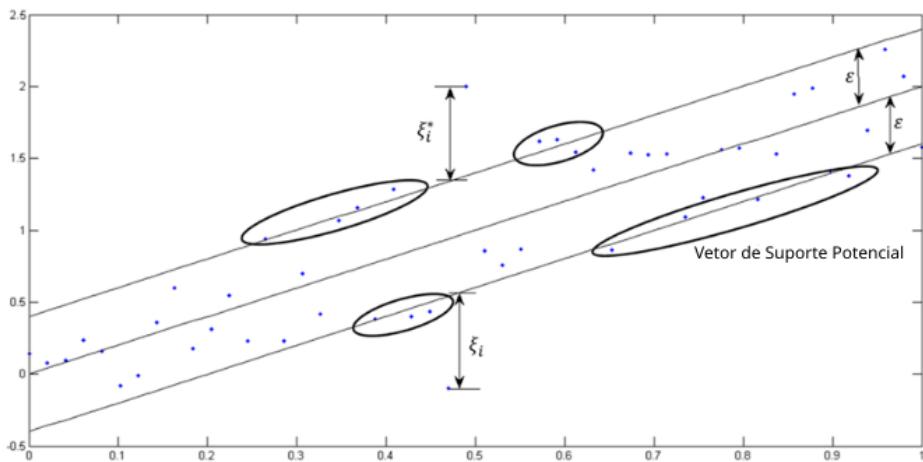
2.1.6.8.2 Gradient Tree Boosting

O *Gradient Tree Boosting* é um algoritmo de aprendizado baseado em árvores de decisão que recorre à técnica de [GD](#) para otimização de funções de perda diferenciáveis. As árvores de decisão individuais são construídas usando *boosting*, o segundo principal paradigma de *Ensemble Learning*. Neste, as árvores são geradas sequencialmente minimizando o erro da árvore anterior através dos residuais definidos pela função de perda. Nesse contexto, deve-se controlar a quantidade de árvores utilizadas no modelo, devido ao risco de *overfitting* a medida que as árvores são adicionadas (FRIEDMAN, 2001).

2.1.6.9 Support Vector Regression (SVR)

A **SVM** é uma técnica de aprendizado estatístico baseado na teoria Vapnik-Chervonenkis, na qual problemas de classificação binária são resolvidos através de sua reformulação como um problema de otimização convexo. Este problema de otimização implica na definição da maior margem possível separando o hiperplano e, simultaneamente, gerando o maior número de previsões corretas nas amostras de treino. As **SVMs** representam o hiperplano ótimo encontrado com vetores de suporte (AWAD; KHANNA, 2015). Um exemplo da aplicação destes conceitos em um **SVM** unidimensional (linear) pode ser visto na Figura 13.

Fig. 13 – SVM de uma dimensão (linear)



Fonte: (AWAD; KHANNA, 2015) (Adaptado)

O problema de regressão pode ser tratado como uma generalização do problema de classificação, no qual o modelo retorna um valor contínuo. Esta generalização é implementada na *Support Vector Regression (SVR)* definindo uma região de insensibilidade ϵ em torno da função, denominado de tubo *epsilon*. Este tubo ressignifica o problema de otimização para a definição do tubo que melhor aproxima a função de regressão, considerando a complexidade do modelo e o erro de previsão.

2.1.7 Métricas de Desempenho de Modelos Preditivos

Modelos preditivos precisam ser avaliados quanto aos seus resultados, de maneira a permitir a comparação entre seus pares e verificar sua validade e utilidade na resolução do problema proposto. Para isso, estabelecem-se métricas de desempenho que dependem do objetivo e do modelo que se está desenvolvendo. Essas métricas são computadas sobre o conjunto de dados de validação para escolha e ajuste de hiperparâmetros e também do conjunto de dados de teste para uma avaliação final de sua performance.

Especificamente, para problemas de regressão, as seguintes métricas são regularmente utilizadas na literatura (BOTCHKAREV, 2018):

- Erro Médio Absoluto (**MAE**): calculado pela Equação (2.6) tomando o somatório do módulo da diferença entre os valores previstos e real para cada ponto (erro absoluto), dividindo-o pelo número de amostras avaliadas. A métrica é apresentada na mesma unidade que a variável de interesse e concede pesos iguais para os erros individuais, independente da sua magnitude.

$$MAE = \frac{\sum |y_{real} - y_{previsao}|}{n_{amostras}} \quad (2.6)$$

- Erro Médio Quadrático (**MSE**) ou Raiz Quadrada do Erro Médio (**RMSE**): o primeiro é calculado pela Equação (2.7) tomando o somatório do quadrado dos erros e dividindo-o pelo número de amostras avaliadas. Nesse caso, os erros individuais maiores são penalizados com pesos maiores devido ao quadrado. Para converter esta métrica para a mesma unidade da variável de interesse, toma-se a raiz quadrada do MSE para obter o RMSE, como visto na Equação (2.8).

$$MSE = \frac{\sum (y_{real} - y_{previsao})^2}{n_{amostras}} \quad (2.7)$$

$$RMSE = \sqrt{MSE} \quad (2.8)$$

- Erro Médio Absoluto Percentual (**MAPE**): calculado pela Equação (2.9) tomando o somatório do módulo do erro absoluto dividido pelo valor real, dividindo-o em seguida pelo número de amostras. A métrica é utilizada como figura de mérito de acurácia para problemas de regressão, principalmente pela sua facilidade de interpretação. Não obstante, apresenta diversos problemas, como a divisão por números muito pequenos ou até gerando uma singularidade (divisão por zero), quando prevendo contra valores reais próximos a zero e também pelo seu viés em relação a valores previstos menores.

$$MAPE = \frac{100\%}{n_{amostras}} \sum \left| \frac{y_{real} - y_{previsao}}{y_{real}} \right| \quad (2.9)$$

- Erro Médio Absoluto em Escala (**MASE**): calculado pela Equação (2.10) pela divisão do Erro Médio Absoluto do modelo avaliado pelo Erro Médio Absoluto de um modelo trivial. Também utilizada como figura de mérito de acurácia, é interpretada como uma medida de desempenho do modelo estudado em relação ao modelo trivial adotado, sem os problemas de divisão por zero ou valores pequenos observados com o **MAPE**.

$$MASE = 100\% \frac{MAE}{MAE_{trivial}} \quad (2.10)$$

- R^2 calculado pela Equação (2.11) tomando o somatório dos quadrados dos erros residuais e dividindo-o pela soma total dos quadrados, proporcional ao quadrado da variância. A métrica é utilizada para avaliar o desempenho de um modelo de regressão.

$$R^2 = 1 - \frac{\sum(y_{real} - y_{previsao})^2}{\sum(y_{real} - y_{medio})^2} \quad (2.11)$$

O coeficiente de determinação R^2 é desaconselhado para comparação de modelos preditivos com tamanhos diferentes, já que a métrica melhora com o aumento do número de variáveis independentes mesmo sem ganho real na capacidade preditiva, recomendando-se o uso do coeficiente de determinação ajustado $R^2_{adjusted}$. Não obstante, o coeficiente de determinação puro foi utilizado neste trabalho, considerando que o conjunto de testes utilizado para avaliação é grande o suficiente e composto apenas de dados jamais vistos no processo de treinamento, permitindo estimar a capacidade de generalização dos modelos sem influência positiva da inserção de novas variáveis.

A escolha de utilizar uma ou mais métricas é dependente do problema e objetivos do modelo desenvolvido. Além disso, existem diversas métricas mais específicas para diferentes casos de uso, que não serão tratadas neste documento.

2.2 Software para Machine Learning

No escopo deste trabalho, serão utilizadas bibliotecas da linguagem de programação *Python*, objetivando reutilizar as implementações dos algoritmos de aprendizado de máquina discutidos anteriormente por interfaces bem definidas e difundidas na indústria.

A biblioteca de código aberto *Keras*, fornece uma interface simples e consistente que facilita o uso para experimentação de um dos principais motores de redes neurais artificiais utilizados no mundo, chamado de *TensorFlow*. É possível realizar todas as etapas de um projeto de redes neurais artificiais, desde a concepção da arquitetura até o encapsulamento para produção (CHOLLET et al., 2015).

Outra biblioteca bastante popular na indústria da ciência de dados é a *scikit-learn*, disponibilizando a implementação de dezenas de algoritmos de aprendizado de máquina para regressão, classificação e clusterização, como *Random Forest*, *Support Vector Machines*, *Gradient Tree Boosting*, *DBSCAN*, entre outros (PEDREGOSA et al., 2011).

2.3 Trabalhos Anteriores

A temática da previsão da geração solar ou de seus fatores é amplamente estudada e tratada na Literatura. Em Inman, Pedro e Coimbra (2013), tem-se uma visão aprofundada das principais modalidades e metodologias para essa finalidade, cobrindo modelos triviais,

métodos regressivos, técnicas de inteligência artificial, satélite, sensores locais e previsões meteorológicas numéricas. No artigo Diagne et al. (2013), os autores revisam de forma simplificada as metodologias para previsão da irradiação solar e as classificam de acordo com suas resoluções temporais e espaciais.

Em Su, Batzelis e Pal (2019), é realizado um estudo comparativo de 10 metodologias de inteligência artificial para previsão de 6 dias da energia gerada, incluindo redes neurais artificiais, aplicadas aos dados de um ano de uma planta fotovoltaica no Reino Unido, sendo proposto um novo método híbrido. Dentre os modelos obtidos, os melhores resultados foram provenientes do modelo híbrido, seguido da arquitetura de rede neural artificial NARXNN e em terceiro lugar a *Random Forest*.

Restringindo o escopo para o uso do aprendizado de máquina na previsão da irradiação solar, Voyant et al. (2017) examina as principais técnicas e algoritmos tratados na literatura, observando a evolução temporal de artigos publicados por cada metodologia e compilando os resultados obtidos por diferentes estudos além de seus horizontes temporais.

Existem inúmeros estudos e artigos publicados implementando técnicas de aprendizado de máquina para previsão da irradiação solar, como em Moosa et al. (2018), onde os autores implementam modelos utilizando *Random Forest*, *XGBoost* e *Redes Neurais Artificiais* para realização de previsão horária da irradiação solar global. Em Jeon e Kim (2020), um modelo de rede neural artificial com arquitetura *LSTM* foi proposto para previsão da irradiação das 24 horas seguintes, utilizando dados meteorológicos observados nas últimas 24 horas e a previsão destes dados para as próximas 24 horas. Em Corea et al. (2016), uma nova abordagem considerando dados de múltiplas estações meteorológicas vizinhas (dados não locais) ao ponto de previsão foi implementada com um modelo de rede neural *feedforward* multicamada totalmente conectada para previsão da irradiação solar horária para as próximas 24 horas.

3 Metodologia e Desenvolvimento

3.1 Definição do Problema

Este trabalho visa o desenvolvimento de modelos de previsão de irradiação solar horária para as 24 horas seguintes utilizando dados de previsão meteorológica de modelos numéricos e dados de medição da irradiação solar anteriores para o ponto-alvo.

É proposto a definição de diversos modelos utilizando 3 arquiteturas de RNAs, *Random Forest*, *Support Vector Regression* e *Gradient Tree Boosting*.

3.2 Tratamento dos Dados

3.2.1 Definição da localização do alvo

A localização do alvo para o qual os modelos desenvolvidos gerarão previsões de irradiação solar horária é definida por:

- Latitude: -13.005515°
- Longitude: -38.50576°
- Altitude: 47,56 m
- Localidade: Salvador, Bahia, Brasil
- Descrição: Estação Automática do INMET A401, dentro do Parque Zoobotânico Getúlio Vargas

3.2.2 Fonte dos Dados

As entradas do modelo foram compostas por dados de previsão meteorológica futura e por dados de medições passadas. Para o primeiro grupo, o modelo numérico utilizado para as previsões meteorológicas é o MERRA-2, disponibilizado via *Application Programming Interface (API)* pela *NASA Power*, em [NASA \(2022\)](#).

Para tanto, foi criado um *script* encapsulando as rotas necessárias para o *download* dos dados horários para o ponto escolhido, respeitando os limites estabelecidos pela [API](#), e organizando os dados recebidos em formato csv.

Os dados do modelo MERRA-2 que serão utilizados são:

- Temperatura ($^{\circ}\text{C}$)
- Temperatura de Orvalho ($^{\circ}\text{C}$)
- Temperatura de Bulbo Úmido ($^{\circ}\text{C}$)
- Umidade Relativa (%)
- Precipitação Média (mm/hora)
- Pressão na Superfície (kPa)
- Direção do vento a 10m ($^{\circ}$)
- Velocidade do vento a 10m (m/s)
- Direção do vento a 50m ($^{\circ}$)
- Velocidade do vento a 50m (m/s)

Por outro lado, a irradiância solar horária foi obtida para a estação meteorológica A401 do [INMET](#), localizada no ponto determinado, através do Banco de Dados Meteorológicos do [INMET](#), acessado em [INMET \(2022\)](#). Nota-se que é fornecida a radiação global (kJ/m^2), integração num período de 1 hora da irradiância solar. Este dado será utilizado tanto como entrada (valores passados), como alvo de treinamento, isto é, a saída a ser determinada.

3.2.3 Exploração e pré-tratamento dos dados

3.2.3.1 Plot dos Dados Brutos

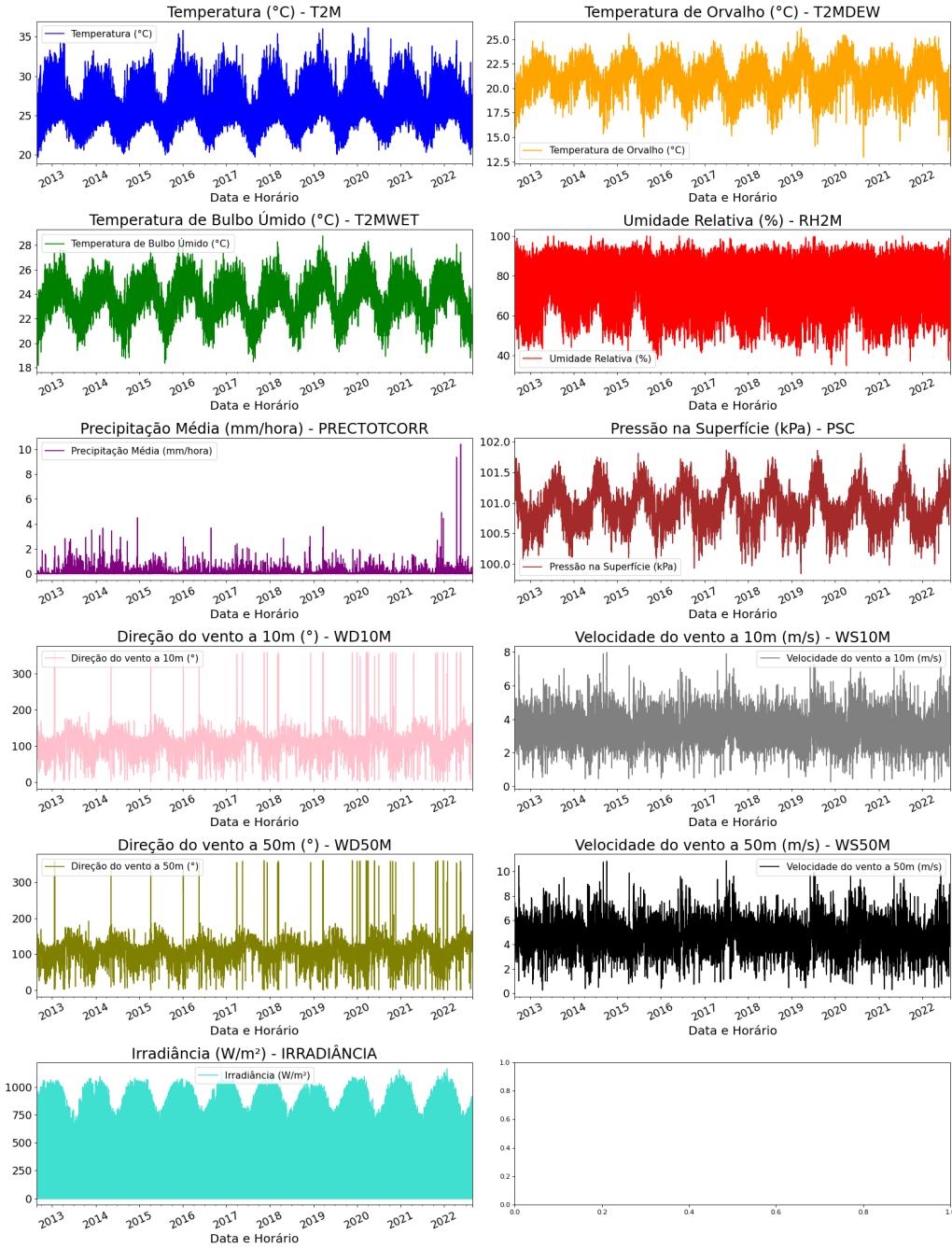
Em um momento inicial, os dados brutos obtidos foram juntados em um único *DataFrame*, uma estrutura de dados bidimensional com formatação tabular da biblioteca *Pandas* ([TEAM, 2020](#)). Neste contexto, os dados foram truncados para o período de 25/08/2012 as 00:00 até 23/08/2022 as 23:00. Estes foram plotados para permitir uma visualização inicial, como pode ser visto na Figura 14.

Evidentemente, há uma certa sazonalidade em diversas variáveis meteorológicas, como a radiação global e a temperatura, mesmo com a predominância de um clima tropical no ponto escolhido.

3.2.3.2 Pré-tratamento dos dados

A partir da visualização dos dados, verificou-se uma série de problemas com a radiação global que necessitavam de algum tipo de tratamento prévio, como a ocorrência de valores amostrais negativos e outros faltantes.

Fig. 14 – Gráficos dos dados brutos do MERRA-2 e INMET



Fonte: Elaboração Própria

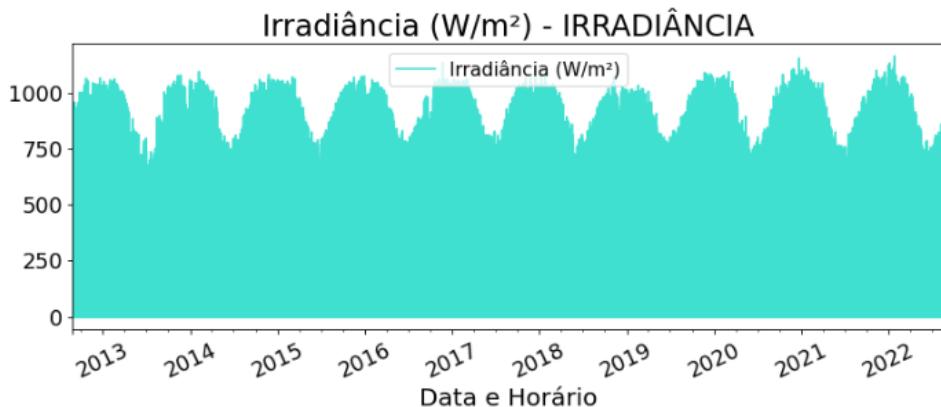
Em primeiro lugar, aplicou-se uma função para todas as amostras da radiação global objetivando substituir os valores levemente negativos por zero, que ocorriam durante algumas noites.

Em seguida, para tratar as diversas amostras da radiação global faltantes, optou-se pela técnica de imputação de dados de substituir as amostras faltantes pela média das amostras no mesmo mês, dia e hora de todos os outros anos.

Por fim, foi feita a conversão da radiação global integrada no período de 1 hora em kJ/m^2 para irradiância solar em W/m^2 , multiplicando as amostras por 1000 e dividindo-as por 3600.

Uma nova visualização dos dados de irradiância solar, após os tratamentos, pode ser vista na Figura 15

Fig. 15 – Dados de irradiância solar após tratamento



Fonte: Elaboração Própria

Na Figura 16, verifica-se um recorte de um mês dos dados pós-tratamento entre 15/06/2021 e 15/07/2021, permitindo a observação da variação intradia das variáveis de entrada, bem como o perfil ruidoso da variável de saída irradiância solar. Em sua totalidade, o conjunto de dados final possui 87624 horas representadas.

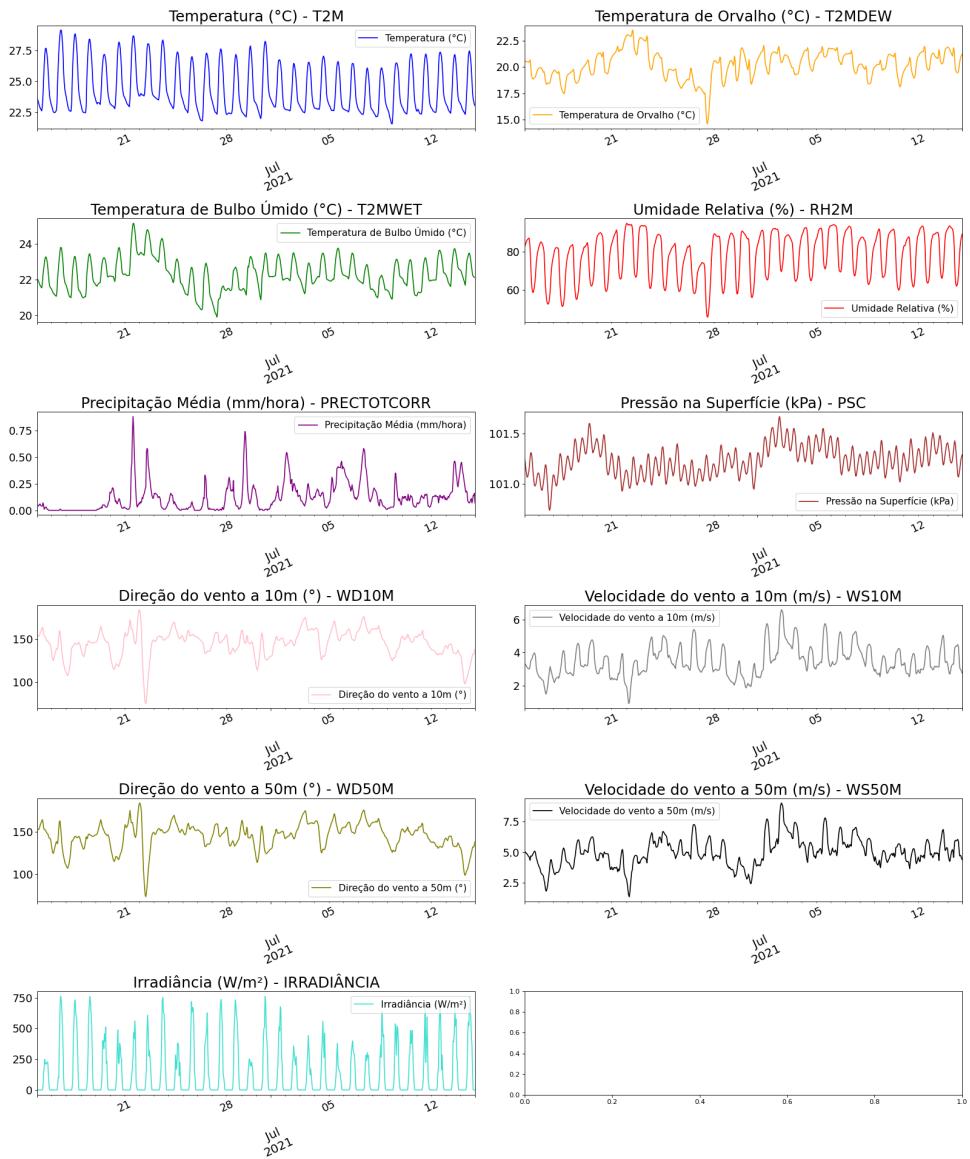
3.2.3.3 Análise das Variáveis

Para observar as características estatísticas das variáveis presentes no conjunto de dados, realizou-se uma análise de tais variáveis, por meio dos métodos de estatística descritiva. Inicialmente, foi traçado um diagrama de caixa (*boxplot*), visando representar a variação de cada variável, como pode ser visto na Figura 17.

Através do diagrama de caixa, evidencia-se a distribuição das variáveis meteorológicas disponíveis no trabalho, permitindo abstrair algumas informações quanto a variabilidade destas. Por exemplo, o ponto escolhido possui pouca variação na temperatura, com o primeiro quartil localizado entre 24°C e 25°C e o terceiro quartil aos $27,5^\circ\text{C}$. A irradiação possui 95% amostras abaixo de 1000 W/m^2 , com grande influência dos horários noturnos que puxam os quartis da caixa para baixo devido ao número elevado de amostras em 0 W/m^2 .

Em seguida, foi gerada uma matriz de correlação de Pearson para detectar correlações lineares negativas e positivas entre as variáveis, como visto na Figura 18

Fig. 16 – Gráficos dos dados após tratamento em um recorte de 1 mês

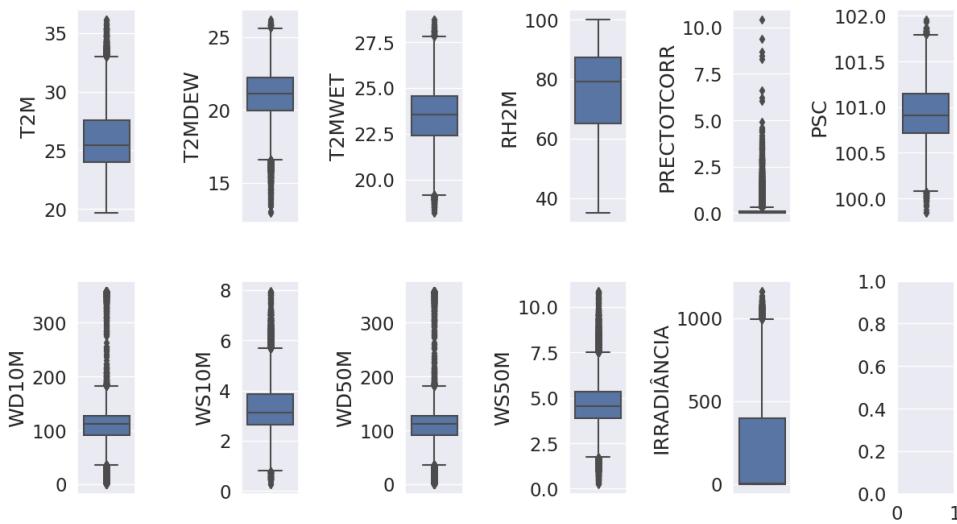


Fonte: Elaboração Própria

Algumas das variáveis possuem correlações positivas e negativas bem definidas com a irradiância, como acontece com a temperatura a 2 metros do solo (correlação positiva) e com a umidade relativa do ar a 2 metros do solo (correlação negativa). Não obstante, outras, como a precipitação horária e a pressão atmosférica, apresentam certa descorrelação com a irradiação solar, o que não implica exatamente que estas variáveis são inadequadas para uso nos modelos de inteligência artificial, apenas que sozinhas e sem nenhuma informação adicional, não possuem relação direta com a irradiância.

Finalmente, foram gerados gráficos de dispersão de cada variável de entrada, eixo x, contra a variável de saída irradiância solar, eixo y, complementando a matriz de correlação mostrada anteriormente, que podem ser vistos na Figura 19.

Fig. 17 – Diagrama de Caixa das Variáveis



Fonte: Elaboração Própria

Nos gráficos de dispersão, é possível identificar algumas das correlações apontadas pela matriz de correlação de Pearson, além da constatação de que a presença de amostras em diferentes momentos do dia acabam por poluir as métricas de estatística descritiva apresentadas, sendo necessário a inclusão de informações temporais para dar utilidade e permitir o aprendizado com estas variáveis.

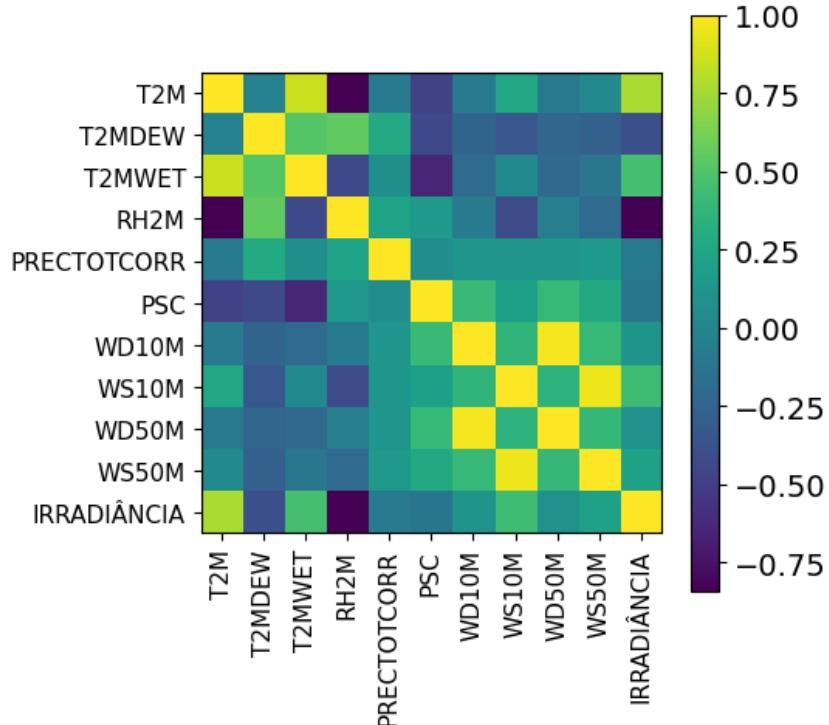
3.3 Modelo de *Benchmark* - Persistência

Visando a obtenção de um modelo trivial para servir de referência quanto a utilidade dos modelos que serão desenvolvidos, foi criado um modelo de persistência, que possui como princípio uma afirmação simples e de baixíssimo custo computacional: "A irradiação solar futura em t será igual à irradiação solar em $t - 24h$ ".

Neste sentido, o modelo de previsão é baseado na persistência dos valores passados, e caso os modelos desenvolvidos através dos métodos de Aprendizado de Máquina com alto custo computacional e diversas entradas não conseguirem superar o desempenho do modelo de persistência, então não possuem nenhuma utilidade. Objetivando comparar os resultados para os mesmos conjuntos de dados, foi realizada uma separação em conjuntos de treino, teste e validação, nas proporções de 60%, 20% e 20%, respectivamente, relembrando que neste caso não há treino, pois as regras do modelo são definidas de forma declarativa.

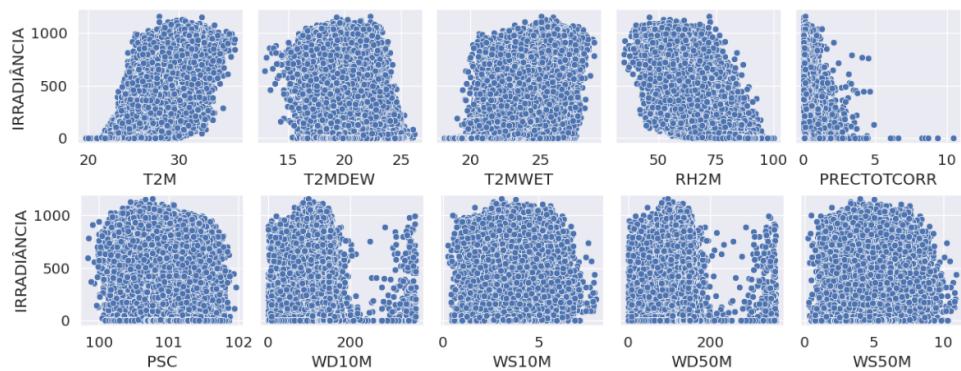
Para obter as previsões sobre o conjunto de teste, bastou deslocar os alvos de irradiação solar deste em 24 horas no futuro, em uma nova coluna de previsões. Computando as métricas de desempenho entre os alvos e as previsões, obtém-se coeficiente

Fig. 18 – Correlação entre as Variáveis



Fonte: Elaboração Própria

Fig. 19 – Gráficos de Dispersão entre as entradas e a irradiação solar

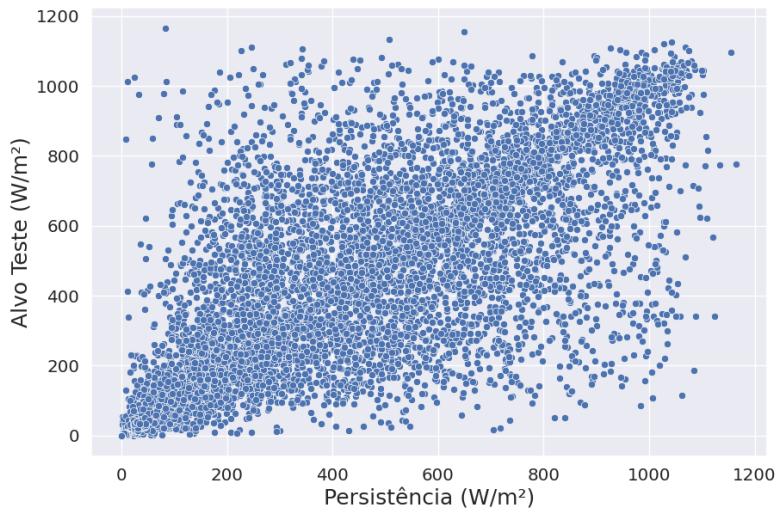


Fonte: Elaboração Própria

de determinação de $R^2 = 0,768$ e um erro médio absoluto de $MAE = 63,70 \text{ W/m}^2$. Na Figura 20, é traçado um gráfico de dispersão entre o alvo de irradiação solar (eixo y) e a previsão de persistência (eixo x).

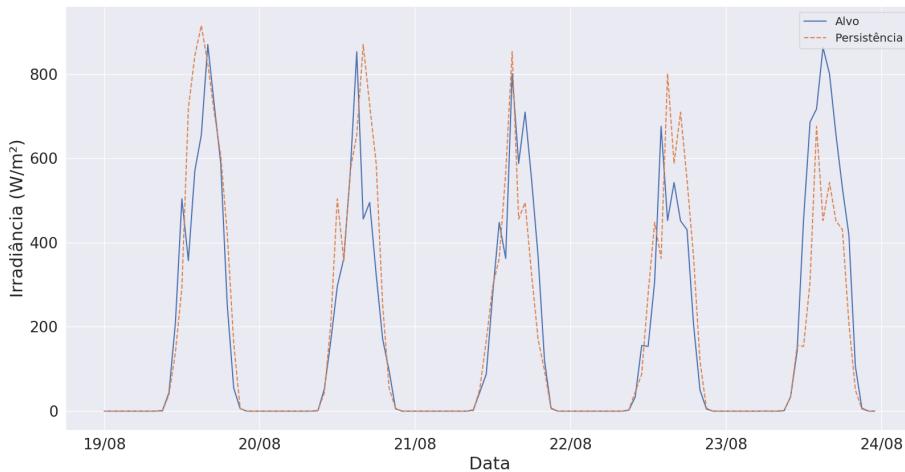
Na Figura 21 é traçado um gráfico de linha, comparando a previsão de persistência com os alvos durante um período de 5 dias, entre 19/08/2022 à 24/08/2022, permitindo que o funcionamento do modelo de referência seja observado.

Fig. 20 – Gráfico de Dispersão entre os alvos e as previsões de persistência



Fonte: Elaboração Própria

Fig. 21 – Gráfico de linha para os alvos e previsões de persistência durante 5 dias



Fonte: Elaboração Própria

3.4 Estruturação dos Dados

Antes de alimentar os algoritmos de Aprendizado de Máquina com os dados do conjunto de teste, é preciso estruturá-lo e condicioná-lo na formatação de um problema supervisionado de sequência temporal seguindo as especificações do projeto, isto é, a partir de dados passados de medições de irradiação solar do INMET e dados passados e futuros do modelo numérico MERRA-2, gerar previsões para as próximas 24 horas de irradiação solar.

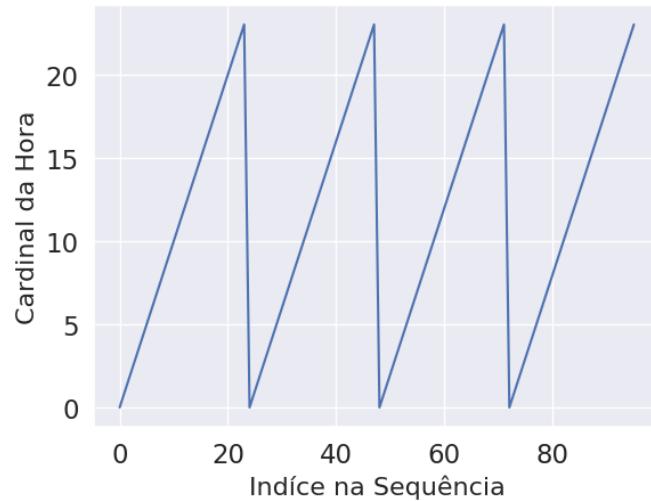
3.4.1 Codificação de *Features* Temporais Cíclicas

Para o problema da previsão da irradiação solar fotovoltaica, as informações temporais quanto a posição do momento no ano e intradia são bastante informativas devido à sazonalidade em ambos os casos. Durante o ano, a irradiação solar varia a depender da estação, possuindo picos de irradiação superiores durante o solstício de verão que no solstício de inverno, por exemplo. Por outro lado, devido ao ciclo solar, a informação quanto a posição horária no dia determina a posição do sol, influenciando diretamente na intensidade da irradiação solar.

Entretanto, inicialmente não existem *features* representando diretamente essas características, apenas uma coluna com a data e hora da amostra. Uma abordagem inicial, por exemplo, seria considerar os valores temporais em sua ordem cardinal natural, como, por exemplo, em uma *feature* onde o primeiro dia do ano é representado pelo número 1 e o último dia do ano é representado pelo número 365 (anos não-bissexto).

Esta abordagem possui uma falha grave, devido à descontinuidade existente. Considerando a primeira hora do dia (00:00) como o número 0 e a última hora do dia (23:00) como o número 23, a *feature* gerada pode ser traçada em um gráfico de linha em um intervalo de 4 dias (96 horas), como pode ser visto na Figura 22. Evidentemente, a *feature* gera uma diferença entre a última hora do dia anterior e a primeira hora do dia seguinte de vinte e três horas, enquanto na realidade essa diferença deveria ser de apenas uma hora. O mesmo se repete para a posição do dia no ano, sendo o último dia do ano mais próximo do primeiro de janeiro do ano seguinte, do ponto de vista da estação, do que o primeiro de junho.

Fig. 22 – Gráfico de linha a *feature* usando o número cardinal da hora



Fonte: Elaboração Própria

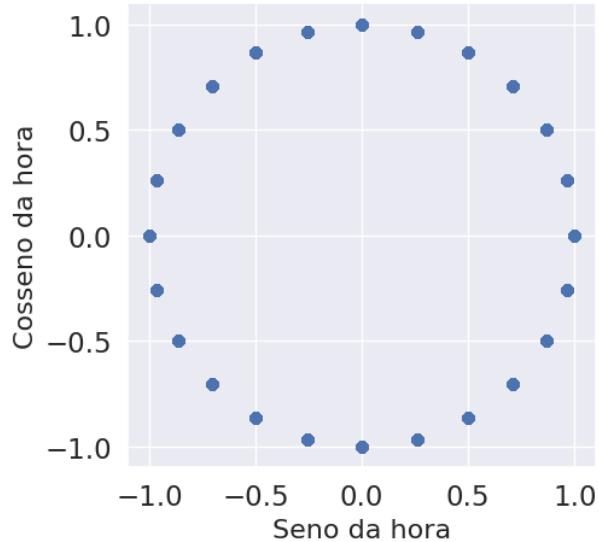
O problema foi solucionado através do uso de uma técnica de codificação de dados cílicos que consiste na representação dos dados temporais em duas dimensões aplicando uma transformação de seno e cosseno. Essas transformações podem ser vistas nas equações 3.1 e 3.2.

$$X_{sin} = \sin \frac{2\pi * x}{\max(x)} \quad (3.1)$$

$$X_{cos} = \cos \frac{2\pi * x}{\max(x)} \quad (3.2)$$

Este processo foi executado para criar duas representações cílicas: a posição da hora no dia e a posição do dia no ano. No primeiro caso, tem-se $\max(x) = 23$, com $0 \leq x \leq 23$, e para o segundo caso, tem-se $\max(x) = \text{quantidade de segundos em um ano}$, com x igual à quantidade de segundos desde o início da Era Unix (*Unix Epoch*), para cada amostra. A transformação para a posição na hora do dia pode ser vista na Figura 23, com o cosseno da hora no eixo x e o seno da hora no eixo y, confirmando a ciclicidade da nova *feature*.

Fig. 23 – Gráfico de dispersão do cosseno e seno da hora



Fonte: Elaboração Própria

3.4.2 Normalização dos Dados

Para facilitação do processo de treinamento e convergência dos algoritmos, todas as variáveis de entrada foram normalizadas para uma escala entre zero e um. Além disto, estas mesmas variáveis foram convertidas de uma representação em 64 bits para uma

representação de ponto flutuante em 32 bits, visando reduzir o uso de memória sem prejuízo para o desempenho dos modelos.

3.4.3 Geração das Sequências de Entradas e Alvos

Modelos de redes neurais artificiais **LSTM** possuem uma característica que as diferem de algoritmos de aprendizado tradicionais: A capacidade de prever sequências temporais com cada passo ocorrendo de forma simultânea (*multi-step output*), o que não ocorre com as outras metodologias. Assim sendo, foi necessário desenvolver duas funções de geração de sequências de entradas e alvos, que serão detalhadas a seguir.

3.4.3.1 Geração das Sequências para Redes Neurais Artificiais Long Short-Term Memory

Criou-se a função utilitária *split_sequence* que recebe como argumentos os dados brutos ordenados por hora, a quantidade limite de horas no passado que pode ser usado como entrada (*look_back*), a quantidade de horas de espaço entre os alvos/previsões e os dados de entrada (*padding_between_lookback_forecast*), a quantidade de horas previstas no futuro simultaneamente (*forecast_horizon*), neste trabalho com valor de 24 horas e os nomes das colunas de medidas do **INMET**, valores do modelo numérico e coluna alvo.

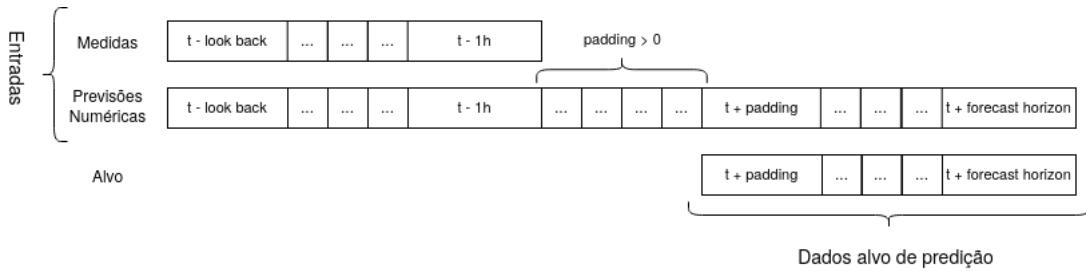
Para cada amostra horária i do conjunto de dados, são criadas três sequências: medidas, previsões e alvos. A sequência de alvos contém os valores de irradiação solar futuro defasado do momento atual por *padding_between_lookback_forecast* amostras até *forecast_horizon* amostras no futuro. A sequência de medidas compreende os valores do passado defasados do momento atual pela variável *look_back*, até a amostra imediatamente anterior a amostra horária i , contendo dados de medição de irradiação solar. A sequência de entrada de previsões contém todo o intervalo existente entre as sequências de alvos e medidas, possuindo os dados do modelo numérico e as colunas com a codificação das *features* temporais cíclicas, já que valores futuros do modelo numérico podem ser utilizados para prever valores futuros de irradiação solar.

Um diagrama mostrando a separação temporal entre os três conjuntos pode ser observado na Figura 24.

3.4.3.2 Geração das Sequências para algoritmos de aprendizado tradicionais

Para a geração de sequências para algoritmos de aprendizado tradicionais, utilizados para os modelos **SVR**, *Random Forest* e *Gradient Tree Boosting*, programou-se a função utilitária *gerar_dados_problema_supervisionado* que recebe como argumentos os dados de previsão do modelo numérico, os dados de medição, os alvos, o número de passos de tempo no passado n_{in} e o número de passos de tempo no futuro n_{out} , que para o trabalho atual será de 24 horas.

Fig. 24 – Diagrama temporal dos conjuntos da geração de sequência para LSTM



Fonte: Elaboração Própria

Esta função gera uma lista de n_{out} DataFrames, cada um representando um problema supervisionado diferente, cujo objetivo é o de prever um determinado passo de tempo no futuro, variando de 1h até n_{out} horas. Por consequente, os algoritmos de aprendizado alimentados por esta modelagem gerarão n_{out} modelos independentes, para cada passo de tempo t , com $1 \leq t \leq n_{out}$, com as medidas de $t = -n_{in}$ até $t = 0h$, previsões do modelo numérico de $t = -n_{in}$ até $t = n_{out}$ e as variáveis cíclicas temporais referentes ao instante t . Um diagrama exemplificando o processo de estruturação realizado por esta função pode ser visto na Figura 25.

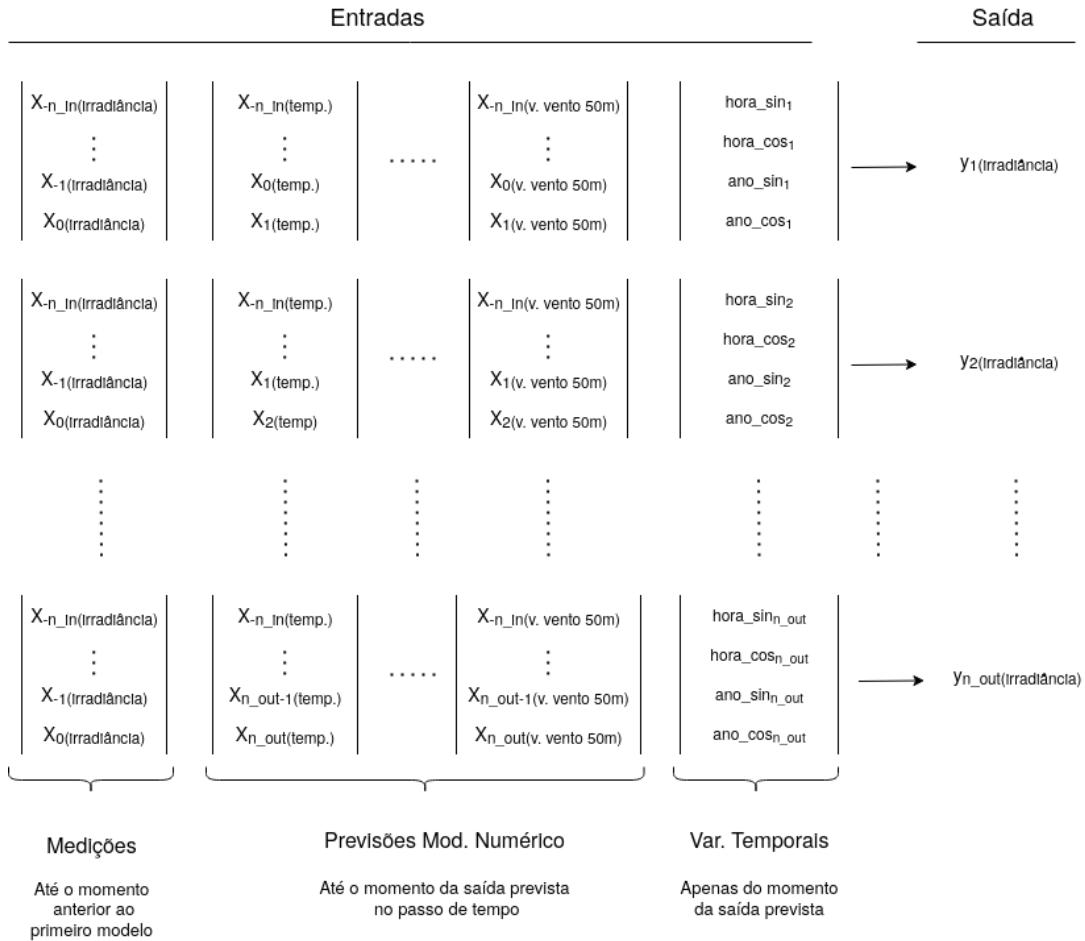
3.4.4 Separação dos dados em conjuntos de treino, validação e teste

A divisão dos dados nos conjuntos de treino, validação e teste se deu nas proporções de 60% das amostras iniciais para treinamento, as 20% amostras subsequentes para validação e as 20% amostras mais recentes para testes nos modelos **LSTM** e **SVR**.

No caso dos modelos de *Random Forest* e *Gradient Tree Boosting*, as implementações das bibliotecas utilizadas possuem o mecanismo de *Out of Bag Estimates*, através da qual em todos os treinamentos uma parte do conjunto de treino não é utilizado, sendo usado para medir a capacidade de generalização do modelo, não sendo necessário o uso de um conjunto de validação. Assim, nestes modelos, a separação foi feita com 80% das amostras para treinamento e 20% para testes.

Assim, para o primeiro caso se tem 55569 horas para treinamento, 18523 para validação e 18524 para teste, e para o segundo caso se tem 74092 horas para treinamento e 18524 para teste. Para ambos os casos, a divisão se deu de forma sequencial, sem o uso de utilitários de embaralhamento, evitando a violação da causalidade temporal entre as amostras. Assim, tem-se 55569 horas para treinamento, 18523 para validação e 18524 para teste.

Fig. 25 – Diagrama dos conjuntos da geração de sequênciа para algoritmos tradicionais



Fonte: Elaboração Própria

3.5 Modelo de Random Forest

Para o algoritmo *Random Forest*, foi utilizada a classe *RandomForestRegressor* da biblioteca *scikit-learn* para realização do treinamento dos 24 modelos de previsão de irradiação solar horária, de $t = +1h$ até $t = +24h$, com *Bootstrap* ativado, isto é, gerando as árvores de decisão com apenas uma fração das amostras do conjunto de treinamento.

Para cada modelo treinado representando uma das 24 horas futuras, é realizado uma previsão a partir de todas as amostras do conjunto de testes, sendo calculadas as métricas R^2 e **MAE** para cada passo de tempo.

3.5.1 Otimização de Hiperparâmetros do Método *Random Forest*

A otimização de hiperparâmetros para o método *Random Forest* foi realizada de forma sequencial, ao invés do uso de métodos mais abrangentes como *Grid Search* ou

Otimização Bayesiana, principalmente devido à limitação de tempo deste trabalho. Além disso, optou-se pela criação de um argumento de otimização, onde são gerados apenas 25% dos modelos de passo de tempo futuro previstos, sendo estes valores utilizados para calcular as médias das métricas de desempenho para comparação. Assim, foram otimizados, em sequência, os seguintes parâmetros com os respectivos valores possíveis, com as escolhas destacadas:

- Número de Árvores (*n_estimators*): [10; 100; **1000**; 2000; 5000]
- Critério de qualidade de desdobramento, sobre o conjunto de treino (*criterion*): [**Erro Médio Quadrático**; Poisson]
- Profundidade Máxima da Árvore (*max_depth*): [2; 4; 8; **12**; 14; 16]
- Número mínimo de amostras para divisão de nó interno (*min_samples_split*): [**2**; 3; 5; 7]
- Número mínimo de amostras para ser nó "folha", sem filhos (*min_samples_leaf*): [2; 3; 5; **7**]
- Número máximo de *features*, em relação ao número total no conjunto de treino n_{total} (*max_features*): [Raiz Quadrada de n_{total} ; Logaritmo de base 2 de n_{total} ; **33% de** n_{total}]

Atenta-se para o fato que a profundidade máxima da árvore é o parâmetro mais determinante para o tamanho na memória dos modelos desenvolvidos. Nesse sentido, tomou-se um valor ótimo, considerando a relação benefício-tamanho para a decisão do valor final. Além disso, a biblioteca utilizada, na versão referenciada neste trabalho, utiliza um valor por padrão de 100% do número total de *features* para o parâmetro *max_features*, transformando o algoritmo em um modelo de *Bagged Ensemble* e não em uma *Random Forest*, sendo, portanto, excluído essa possibilidade do universo de testes.

3.6 Modelo de *Support Vector Regression*

Para o algoritmo de *Support Vector Regression*, foi utilizada a classe *SVR* da biblioteca *scikit-learn* para o treinamento dos 24 modelos de previsão de irradiação solar horária, com *cache de kernel* de 1 GB, sem limites de iterações com parada automática após a função de perda não melhorar pela tolerância de 0,001 para duas iterações consecutivas.

Similarmente ao modelo de *Random Forest*, é realizado uma previsão para cada passo horário a partir de todas as amostras do conjunto de testes e sendo calculados as mesmas métricas.

3.6.1 Otimização de Hiperparâmetros do Método *Support Vector Regression*

A otimização de hiperparâmetros seu deu da mesma maneira realizada para o modelo de *Random Forest*. Neste caso, a otimização sequencial se deu para os seguintes parâmetros, com as escolhas destacadas:

- Tipo de Kernel (*kernel*): [Polinomial; **Função de Base Radial (rbf)**; Sigmoide]
- Parâmetro de Regularização (*C*): [0,1; 0,5; 1; 2; 5; 10, **100**]
- Coeficiente de Kernel (*gamma*): ["auto"; "scale"; 1; 0,1; 0,01; 0,001]

3.7 Modelo de *Gradient Tree Boosting*

Para o algoritmo de *Gradient Tree Boosting*, foi utilizada a classe *GradientBoostingRegressor* da biblioteca *scikit-learn* para o treinamento dos 24 modelos de previsão de irradiação solar horária, com fração de validação igual a 20% do total de amostras (25% das amostras de treino), com parada automática (*Early Stopping*) após 50 iterações sem melhora no *score* de validação, com uma tolerância de 0,001.

Como feito para os algoritmos anteriores, foram realizadas inferências a partir das amostras do conjunto de testes, sendo calculados as métricas de desempenho para cada passo horário e um agregado com a média destes.

3.7.1 Otimização de Hiperparâmetros do Método *Gradient Tree Boosting*

A otimização dos hiperparâmetros do método *Gradient Tree Boosting* se deu da mesma forma que os casos *Random Forest* e **SVR**, de forma sequencial, para os seguintes parâmetros, com as escolhas destacadas:

- Número de Estágios de *Boosting* (*n_estimators*): [10; 100; 200; 300; **1000**; 2000]
- Função de Perda (*loss*): [**Erro Quadrático**; Huber; Quantil]
- Taxa de Aprendizado (*learning_rate*): [0,01; 0,05; **0,1**; 0,5; 1]
- Fração de Amostras para aprendizado individual de base (*subsample*): [20%; 50%; 80%; **100%**]
- Critério de Qualidade de Separação (*criterion*): [**Erro Médio Quadrático com Pontuação de Friedman**; Erro Médio Quadrático; Erro Médio Absoluto]
- Profundidade Máxima dos Estimadores de Regressão Individuais (*max_depth*): [2; **3**; 5; 8; 12; 14]

- Número mínimo de amostras para divisão de nó interno (*min_samples_split*): [2; 3; 5; 7]
- Número mínimo de amostras para ser nó "folha", sem filhos (*min_samples_leaf*): [1; 2; 3; 5; 7]
- Número máximo de *features*, em relação ao número total no conjunto de treino n_{total} (*max_features*): [Raiz Quadrada de n_{total} ; Logaritmo de base 2 de n_{total} ; 100% de n_{total}]

3.8 Modelos de Redes Neurais Artificiais

Para o desenvolvimento dos modelos de **RNAs**, foi utilizado a biblioteca *Keras*, que fornece uma interface de alto nível acessando a *framework* de *Deep Learning TensorFlow*.

Neste caso, a liberdade para definição das arquiteturas permite a criação de modelos que recebem sequências de entradas para gerar sequências de saídas, não sendo necessário gerar 24 modelos diferentes para cada passo horário. Todas as arquiteturas propostas empregaram algum tipo ou variação da arquitetura **LSTM**, ideal para aprendizado de sequências.

Como parte da metodologia empregada, fez-se uso da técnica de *Early Stopping*, para interromper o treinamento após 50 iterações sem melhora da *Loss Function* de validação e também a técnica de *Model Checkpoint*, salvando o melhor modelo.

3.8.1 Arquiteturas de Redes Neurais Implementadas

Foram implementadas 3 arquiteturas de redes neurais, manuseando a API funcional da biblioteca *Keras*, com os tamanhos de cada camada otimizados por meio de um processo iterativo, por tentativa e erro. Outros parâmetros importantes de cada arquitetura, como os valores utilizados nas técnicas de regularização, também foram definidos por este mesmo processo. As arquiteturas desenvolvidas serão detalhadas a seguir:

3.8.1.1 Arquitetura LSTM Comum (*Vanilla*)

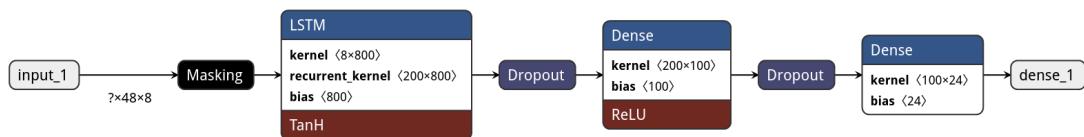
A arquitetura **LSTM** comum é composta por uma camada de entrada recebendo as sequências temporais de entrada, em forma de matriz com dimensões (*look_back + forecast_horizon*) x $n_{features}$. Evidentemente, a *feature* de medição da irradiância solar horária passada não contempla valore futuros, os alvos do modelo, e consequentemente seus valores na matriz são preenchidas por -1. Assim, a coluna da *feature* de irradiância solar horária passa a ser ignorada para os valores de t até $t + forecast_horizon$, por intermédio de uma camada *Masking*, logo após as entradas, para valores iguais a -1.

As saídas da camada *Masking* alimentam uma camada **LSTM** de 200 células, com ativação tangente hiperbólico e regularizador de viés aplicando uma penalidade L2 de 0,001, gerando uma nova representação a partir dos dados de entrada. Por consequente, passa-se por uma primeira camada de *Dropout*, com 10% das conexões de entrada e recorrentes probabilisticamente excluídas das funções de ativação e atualização dos pesos, visando a regularização do modelo.

As 200 saídas da camada anterior são direcionadas à entrada de uma camada densamente conectada com 100 neurônios e ativação ReLU, processando as representações internas da camada **LSTM**, cujas saídas são novamente sujeitas a uma camada de *Dropout* com proporção de 10%. Finalmente, as saídas da última camada são ligadas a uma última camada densamente conectada, com ativação linear, com função de converter as representações anteriores em valores de irradiação solar horária, com uma quantidade de neurônios igual a *forecast_forizon*, neste caso, 24, que será a quantidade de saídas que o modelo gerará para cada vetor de entrada, representando as 24 horas futuras que o modelo deve prever.

Um diagrama da arquitetura desenvolvida, gerada através da ferramenta *Netron*, pode ser vista na Figura 26 (ROEDER, 2017).

Fig. 26 – Diagrama da Arquitetura LSTM Comum (*Vanilla*)



Fonte: Elaboração Própria

3.8.1.2 Arquitetura *Encoder-Decoder LSTM*

Na arquitetura *Encoder-Decoder LSTM*, faz-se uso de estruturas intermediárias de codificação e decodificação, compostas por duas camadas **LSTM**, com a primeira atuando como codificador (*Encoder*) e a segunda como decodificador (*Decoder*).

Inicia-se com a mesma estrutura da arquitetura anterior, composta de uma camada de *Input* seguida de *Masking*. Em seguida, se apresenta a camada **LSTM** codificadora de 128 unidades, com ativação tangente hiperbólico, responsável por ler e interpretar as entradas, gerando uma representação interna dos estados como saída. Esta camada possui regularização de viés (*bias*) com penalidade L2 de 0,001 sendo seguida por uma camada de regularização *Dropout* com proporção de 10%.

Adiciona-se uma camada de repetição *RepeatVector*, repetindo a representação interna do codificador *forecast_horizon* vezes, uma para cada passo de saída do modelo,

resultando em 24 repetições. Os vetores de representação interna do codificador são direcionados para uma camada decodificadora **LSTM** de 128 unidades, com ativação tangente hiperbólico, configurada para retornar a sequência inteira de entrada, o que significa que cada unidade da camada retornará um valor de saída para cada passo de tempo futuro (sequência de 24 horas), resultando em uma saída tridimensional. Novamente, aplica-se regularização de viés com penalidade L2 de 0,001 e uma camada de *Dropout* com proporção de 10%.

A saída da camada de *Dropout* é direcionada para uma camada densamente conectada de 64 unidades, com ativação ReLU, responsável por interpretar as sequências de cada passo de tempo da camada decodificadora, viabilizada por uma interface *TimeDistributed*, que realiza essa separação de cada passo de tempo (distribuição temporal), permitindo a leitura da entrada tridimensional. A camada conectada densamente distribuída no tempo é sujeita a uma última camada de regularização *Dropout*, com proporção de 10%, antes de passar por uma última camada conectada densamente distribuída no tempo de 1 neurônio, com ativação linear, gerando um único valor para cada passo de tempo da saída.

O diagrama da arquitetura *Encoder-Decoder LSTM* pode ser visualizado na Figura 27.

Fig. 27 – Diagrama da Arquitetura *Encoder-Decoder LSTM*



Fonte: Elaboração Própria

3.8.1.3 Arquitetura *Encoder-Decoder CNN-LSTM*

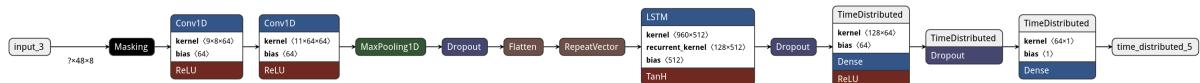
Uma variação da arquitetura *Encoder-Decoder LSTM* é implementada substituindo a camada **LSTM** de codificação por uma sub-rede do tipo *Convolutional Neural Network* (**CNN**). A primeira camada **LSTM** com *Dropout* da arquitetura anterior é substituída por duas camadas convolucionais de uma dimensão *Conv1D*, com 64 filtros de saída e ativação ReLU. A primeira recebe as sequências de entrada e as projetam em mapas de *features* de abstração, contando com um tamanho de kernel de 9 unidades. A segunda recebe os mapas de *features* da camada anterior e amplifica as *features* proeminentes.

As duas camadas convolucionais unidimensionais são seguidas de uma camada de *pool* máximo *MaxPooling1D*, com tamanho de *pool* igual a 2, simplificando segmentos 2x2 do mapa de *features* no valor máximo e consequentemente apenas abstraindo os sinais mais ressaltados e reduzindo o mapa para um quarto do seu tamanho. Este mapa simplificado é submetido a uma camada de regularização *Dropout*, com 10% de proporção e é finalmente

achatado para a estrutura de um vetor, por uma camada *Flatten*, antes de ser direcionado para a mesma estrutura decodificadora da arquitetura anterior, iniciando pela camada *RepeatVector*.

O diagrama da arquitetura *Encoder-Decoder CNN-LSTM* pode ser visualizado na Figura 28.

Fig. 28 – Diagrama da Arquitetura *Encoder-Decoder CNN-LSTM*



Fonte: Elaboração Própria

3.8.2 Otimização de Hiperparâmetros para os Modelos de Redes Neurais Artificiais

Dois parâmetros intrinsecamente relacionados foram otimizados de forma sequencial. Em primeiro momento, variou-se o otimizador com as taxas de aprendizado *learning_rate* padrão, para as opções *Adam*, *SGD* e *RMSProp*. Feita a escolha do otimizador para cada arquitetura, a taxa de aprendizado *learning_rate* foi então variada no intervalo [0,01; 0,005; 0,001; 0,0005; 0,00001], avaliando as métricas de desempenho. Para todas as arquiteturas, optou-se pelo otimizador *Adam* com *learning_rate* de 0,001.

Outros parâmetros otimizados durante um longo processo de tentativa e erro foram as escolhas das camadas e técnicas de regularização, além das penalidades daquelas que foram utilizadas. No fim, optou-se pelo uso de regularização de viés e *Dropout*, com as penalidades definidas nas seções anteriores.

3.9 Testes de Inclusão de *Features* de Entrada

Para cada tipo de algoritmo proposto nas seções anteriores, foram feitos testes de inclusão de *features* de entrada. Esse procedimento é importante devido ao fato que alguns algoritmos tem maior resistência a problemas de alta dimensionalidade, como o *Random Forest*, no qual a penalidade para a adição de uma nova *feature* nas sequências de entrada será menor que em outras abordagens.

Este processo poderia ser realizado com uma das técnicas de otimização de hiperparâmetros como *Grid Search*, porém devido à limitação de tempo, foi realizada de forma sequencial, testando todas as *features* ainda não escolhidas por vez e adicionando a melhor até não haver melhora nas métricas de desempenho.

Assim, as *features* de entrada para cada tipo de algoritmo são descritas abaixo:

- Random Forest, **SVR** e Gradient Tree Boosting: [*Features* temporais cíclicas, Umidade Relativa (%), Direção do Vento a 50 m (°), Pressão na Superfície (kPa), Irradiância Solar Passada (W/m²), Precipitação Média (mm/hora), Temperatura (°C), Direção do Vento a 10 m (°), Velocidade do Vento a 50 m (m/s)]
- Redes Neurais Artificiais: [*Features* temporais cíclicas, Umidade Relativa (%), Direção do Vento a 50 m (°), Pressão na Superfície (kPa), Irradiância Solar Passada (W/m²)]

3.10 Otimização do tamanho da sequência de medições e previsões numéricas no passado

O tamanho da sequência de medições no passado, definido pelas variáveis n_{in} e *look_back*, para os algoritmos de redes neurais artificiais e algoritmos tradicionais, respetivamente, também foi otimizado, sem apresentar ganhos reais para períodos mais longos que 24 horas. Nesse sentido, o tamanho da sequência de medições e previsões numéricas no passado foi definido em 24 horas, para ambos os casos.

3.11 Cenário de Testes em Produção

Para validação do projeto, pensou-se na criação de um cenário potencial de uso dos modelos desenvolvidos, partindo de algumas premissas que correspondem ao tipo de uso provável em produção. Para tanto, criou-se um código carregando todos os modelos desenvolvidos e gerando, de forma diária (1x por dia), as 07:00 UTC-0 (04:00 em horário local de Salvador, Bahia), previsões da irradiância solar das próximas 24 horas.

Os dados de saída de todos os modelos foram então comparados de forma qualitativa e quantitativa, através da geração de gráficos e das métricas discutidas na revisão bibliográfica e discutidas na próxima seção deste trabalho. Para facilitação do processo de comparação e geração de estatísticas relevantes, criou-se uma interface de previsão a partir dos modelos treinados para cada tipo, gerando estruturas de resultados com todas as previsões agregadas, as previsões agrupadas a cada 24 horas após o horário de geração e finalmente os resultados agrupados por hora.

Por fim, dentro do escopo desta interface de previsões, foi aplicado um filtro nos modelos para evitar a presença de valores negativos e também de valores positivos fora do horário do ciclo solar anual, definido das 08:00 UTC-0 até as 21:00 UTC-0 (05:00 até 18:00 em horário local).

3.12 Disponibilização em Código-Aberto do Software

O *software* desenvolvido neste trabalho de conclusão de curso bem como os dados utilizados e tratados, foram disponibilizados em sua integralidade em um repositório aberto na plataforma *GitHub*, com acesso em Ribeiro (2022). A licença de uso é a MIT, a mais permissiva da indústria, liberando o uso, cópia, redistribuição e uso comercial irrestrito. A disposição do repositório no *GitHub* pode ser visualizado na Figura 29.

Fig. 29 – Disposição do repositório do projeto na plataforma *GitHub*

b-rbmp Persistencia		
dados	Repositorio na Versão Limpa	2 months ago
desenvolvimento_modelos	Persistencia	3 hours ago
tratamento_dados	Persistencia	3 hours ago
.gitattributes	Initial commit	2 months ago
.gitignore	Persistencia	3 hours ago
LICENSE	Initial commit	2 months ago
gtb.log	Persistencia	3 hours ago
persistencia.log	Persistencia	3 hours ago
random_forest.log	Persistencia	3 hours ago
requirements.txt	Persistencia	3 hours ago
svr.log	Persistencia	3 hours ago

Fonte: Elaboração Própria

O repositório é organizado em pastas com nomes autoexplicativos, e, além disso, é disponibilizado um arquivo requirements.txt, detalhando as bibliotecas e suas respectivas versões utilizadas durante o desenvolvimento. Como alguns dos modelos gerados possuem tamanhos relativamente grandes, estes não estão inclusos no repositório, podendo ser construídos rodando os códigos de cada modelo, dentro da pasta *desenvolvimento_models*.

4 Análise de Resultados

Nesta seção, serão apresentados, para os modelos desenvolvidos, os resultados dos testes realizados e as análises correspondentes. Um recorte do conjunto de dados de teste foi tomado garantindo o atendimento do cenário de testes em produção descrito na seção anterior, para a geração da previsão horária 1x por dia com os dados até 07:00 UTC-0 (04:00 em horário local), para cada modelo projetado.

Para tanto, criou-se um código de resultados implementando a funcionalidade de testes de previsão, adequando o conjunto de dados, gerando as previsões e comparando os resultados com auxílio de gráficos e estatísticas.

4.1 Dados do Processo de Treino

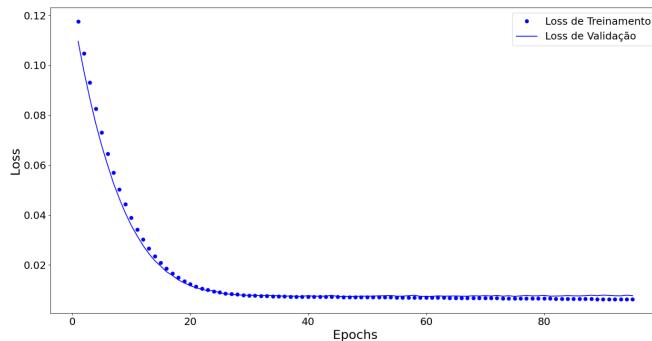
O processo de treino foi realizado conforme o código gerado para cada modelo, uma última vez com os hiperparâmetros finais. Para os métodos de aprendizado tradicionais, foram gerados 24 submodelos horários para cada passo de tempo das 24 horas de previsão. Para o modelo de [SVR](#), os submodelos totalizaram 2,8 GB e para o modelo de *Gradient Tree Boosting*, o valor foi de 29 MB. No modelo *Random Forest*, os submodelos totalizaram 3,3 GB, tamanho gerado justificado pela profundidade elevada da árvore com o qual a processo de treino foi realizado, sendo possíveis reduções exponenciais renunciando a uma margem de seu desempenho.

Para as 3 arquiteturas de redes neurais artificiais propostas, as curvas da função de perda (*loss*) em relação ao o número de iterações (*epochs*), pode ser visto nas figuras [30](#), [31](#) e [32](#), mostrando a devida convergência de resultados, interrompendo o treino assim que se dava início ao fenômeno de *overfitting*. Os modelos gerados possuem 2,3 MB, 4,1 MB e 10,7 MB, para as arquiteturas LSTM comum, *Encoder-Decoder LSTM* e *Encoder-Decoder CNN-LSTM*, respectivamente.

4.2 Testes e Resultados das Previsões

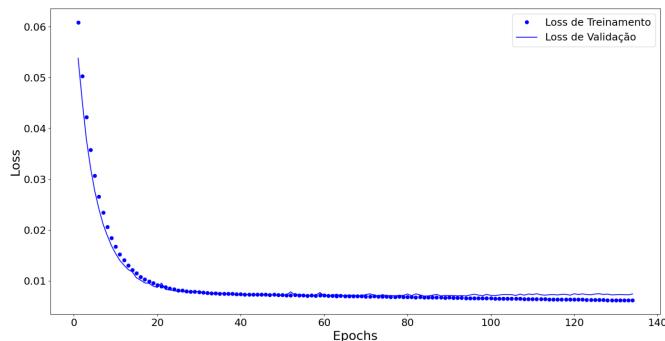
Os resultados foram gerados para um conjunto de dados de entrada cuja última hora de medição ocorria as 07:00 UTC-0, diariamente, para um período de aproximadamente 2 anos, gerando 728 previsões de irradiação solar, no período de 25/08/2020 às 08:00 até 22/08/2022 às 07:00, cada uma cobrindo períodos de 24 horas (24 previsões horárias), totalizando 17472 horas de previsão. Nenhum dos dados de entrada foi utilizado para o treinamento dos modelos, consistindo em dados novos para os mesmos.

Fig. 30 – *Loss* com a evolução do treinamento da rede LSTM comum



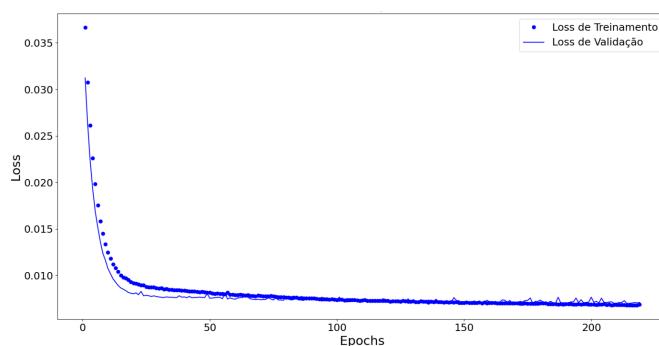
Fonte: Elaboração Própria

Fig. 31 – *Loss* com a evolução do treinamento da rede *Encoder-Decoder LSTM*



Fonte: Elaboração Própria

Fig. 32 – *Loss* com a evolução do treinamento da rede *CNN LSTM Encoder-Decoder*

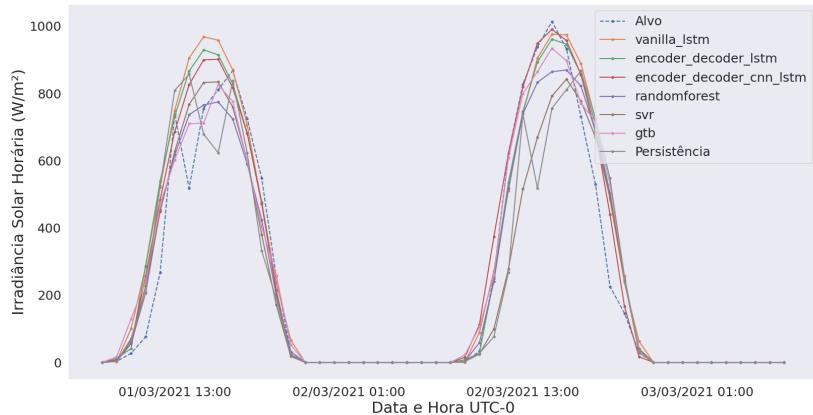


Fonte: Elaboração Própria

4.2.1 Exemplos de Previsões

Visando uma demonstração do comportamento dos modelos desenvolvidos, foram geradas previsões de irradiação solar para 2 dias consecutivos, em 3 períodos diferentes do ano, sendo o primeiro de 01/03/2021 às 08:00 até 03/03/2021 às 07:00, o segundo de 22/12/2021 às 08:00 até 24/12/2021 às 07:00 e o último de 17/08/2022 às 08:00 até 19/08/2022 às 07:00, todos na referência UTC-0. Os resultados obtidos foram comparados com os gerados pelo modelo da persistência e também com os valores reais da irradiação solar horária no período. Os resultados comparativos podem ser visualizados nas Figuras 33, 34 e 35.

Fig. 33 – Previsão de 01/03/2021 08:00 até 03/03/2021 07:00 - UTC-0

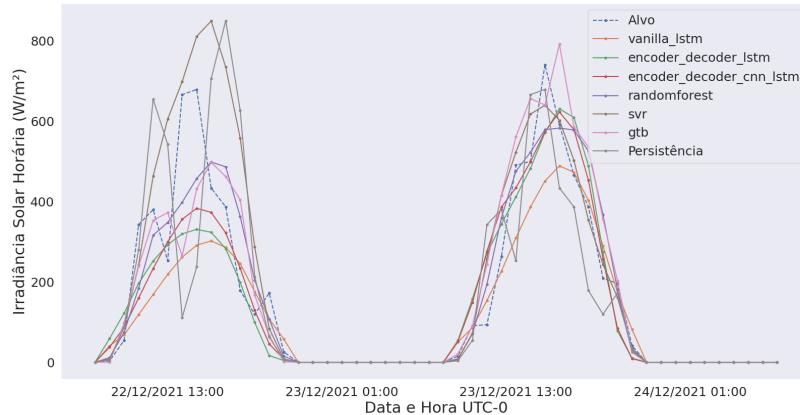


Fonte: Elaboração Própria

No primeiro período, de 01/03/2021 08:00 até 03/03/2021 07:00 UTC-0, tem-se o exemplo de dois dias no fim do verão. O primeiro, com presença de chuvas e nebulosidade variável e o segundo, com menos nuvens e mais seco. Verifica-se que o dia 01/03/2021 possui uma irradiação alvo que segue uma curva de sino bastante imperfeita, impondo um desafio grande para a precisão dos modelos. Neste caso, os modelos de redes neurais artificiais apresentam os piores resultados, sugerindo valores de irradiação horária relativa a um dia mais ensolarado de verão, enquanto os modelos de algoritmos de aprendizado tradicionais apresentaram um melhor desempenho dimensionando as médias horárias para o provável futuro indicado pelos modelos numéricos, com destaque para o modelo de *Random Forest* e *Gradient Tree Boosting*. Para o segundo dia, a situação se inverteu, com melhor desempenho dos modelos de redes neurais.

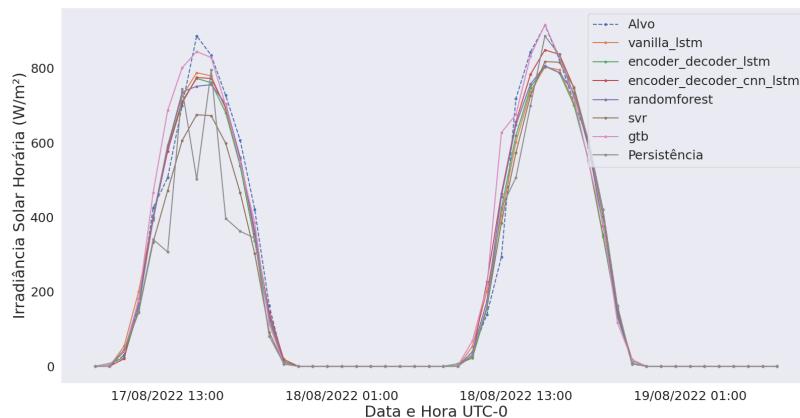
No segundo período, de 22/12/2021 até 24/12/2021, têm-se dias de início de verão mais úmidos e com mais nebulosidade. Novamente, verificou-se uma melhor performance dos modelos de *Random Forest* e *Gradient Tree Boosting*, que apesar de não terem acertado

Fig. 34 – Previsão de 22/12/2021 08:00 até 24/12/2021 07:00 - UTC-0



Fonte: Elaboração Própria

Fig. 35 – Previsão de 17/08/2022 08:00 até 19/08/2022 07:00 - UTC-0



Fonte: Elaboração Própria

a grande variabilidade de irradiância horária, geraram valores cujas médias resultam em uma melhora considerável no grau de previsibilidade em relação ao modelo de persistência.

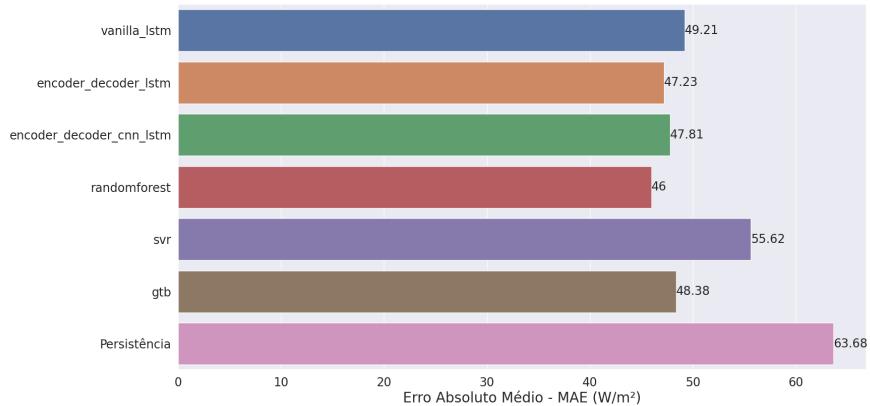
Finalmente, o último período, de 17/08/2022 até 19/08/2022, consiste em dois dias de inverno ensolarado e com baixa umidade. O desempenho dos modelos foram bastante similares, com o *Gradient Tree Boosting* mais se aproximando dos picos das ondas de sino, mas errando em maior magnitude os valores intermediários.

4.2.2 Métricas de Desempenho

Para todos os modelos desenvolvidos e também para o modelo de referência de persistência, foram calculadas as métricas de desempenho Erro Absoluto Médio (**MAE**) para cada hora UTC-0 de previsão e o Coeficiente de Determinação (R^2) sobre todo o universo de previsões. No caso do **MAE** horário, que será mostrado em detalhes posteriormente, foi tomada uma média dos valores para fins de comparação direta entre os modelos.

Um gráfico de barras horizontal com os valores do **MAE** médio para cada modelo, incluindo a persistência, pode ser observado na Figura 36. O modelo de persistência oferece um *benchmark* com **MAE** médio de 63,68 W/m². Todos os modelos desenvolvidos apresentam desempenho superior ao *benchmark* de persistência, sendo o modelo de *Random Forest* o superior, com **MAE** médio de 46,00 W/m², com a rede neural artificial com arquitetura *Encoder-Decoder LSTM* em segundo lugar, com **MAE** Médio de 47,23 W/m². Dos modelos desenvolvidos, o modelo de **SVR** apresentou disparadamente o pior desempenho, com **MAE** Médio de 55,62 W/m².

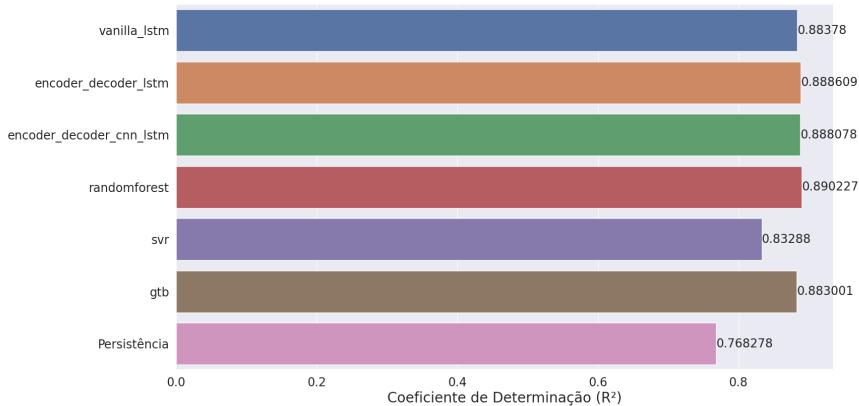
Fig. 36 – MAE médio por modelo



Fonte: Elaboração Própria

Ademais, um novo gráfico de barras horizontal com os valores do R^2 para cada modelo pode ser visualizado na Figura 37. Novamente o melhor desempenho para esta métrica foi verificado no modelo de *Random Forest*, com R^2 de 0,8902, seguido pela rede neural artificial com arquitetura *Encoder-Decoder LSTM*, com R^2 de 0,8886. Para comparação, o modelo de *benchmark* novamente aparece em último lugar, com R^2 de 0,7683, sendo o modelo de **SVR** novamente portador da pior performance entre os que foram desenvolvidos, com R^2 de 0,8329.

Fig. 37 – Coeficiente de Determinação R^2 por modelo



Fonte: Elaboração Própria

4.2.3 Estatísticas de Erro

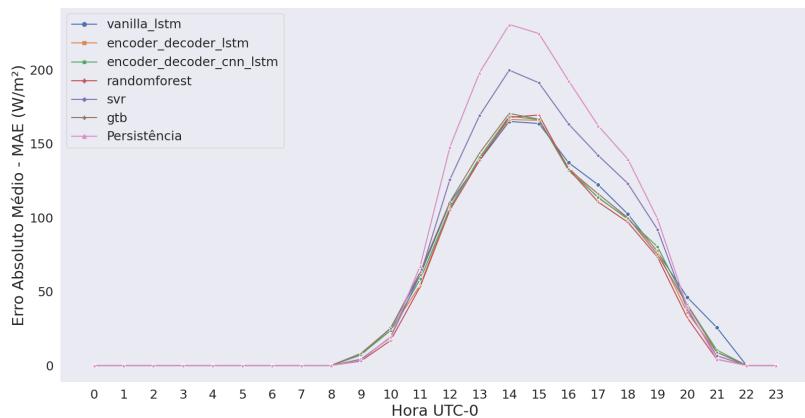
Uma análise mais profunda das estatísticas de erro é necessária para um melhor entendimento do funcionamento dos modelos, permitindo o desenvolvimento de estratégias para aperfeiçoamento destes e compreensão das limitações de uso.

Inicialmente, as previsões foram agrupadas por hora UTC-0, sendo calculado o **MAE** médio individual das previsões horárias. Uma visualização deste erro, por hora UTC-0, pode ser observada na Figura 38. Evidentemente, os maiores erros se concentram em torno dos horários de pico, quando a altura solar é máxima (entre 14:00 e 15:00 UTC-0, ou 11:00 e 12:00 em horário local), reproduzindo a forma de sino observada no próprio gráfico de irradiância solar para todos os modelos, o que era esperado, pois são os horários cujas previsões e alvos apresentam maior magnitude.

Outro ponto importante a ser considerado é o desempenho de cada modelo para cada período do dia, no contexto de geração das previsões diárias às 07:00 UTC-0. Alguns modelos, como a **RNA** com arquitetura *Encoder-Decoder CNN-LSTM* e o modelo de *Gradient Tree Boosting*, apresentam erros mais expressivos no início e fim do dia, quando os ângulos de altura solar são menores, apresentando comportamento similar aos pares no período de pico. Por outro lado, os modelos que apresentam os menores **MAE** para os horários de pico são as **RNAs** com arquitetura LSTM Comum e *Encoder-Decoder LSTM*.

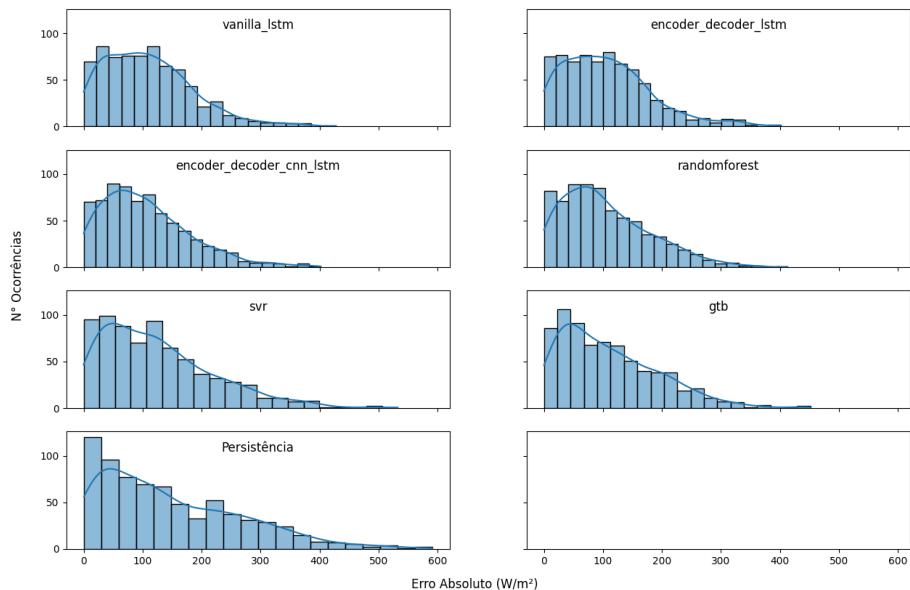
Ademais, foram traçados histogramas do número de ocorrências do **MAE** médio horário em 20 classes para três horas UTC-0 representando diferentes posições horárias em relação à altura solar no dia. Esta visualização foi realizada para 12:00 UTC-0 (09:00 no horário local), 15:00 UTC-0 (12:00 no horário local) e para 20:00 UTC-0 (17:00 no horário local) como pode ser visto nas Figuras 39, 40 e 41.

Fig. 38 – MAE horário por modelo



Fonte: Elaboração Própria

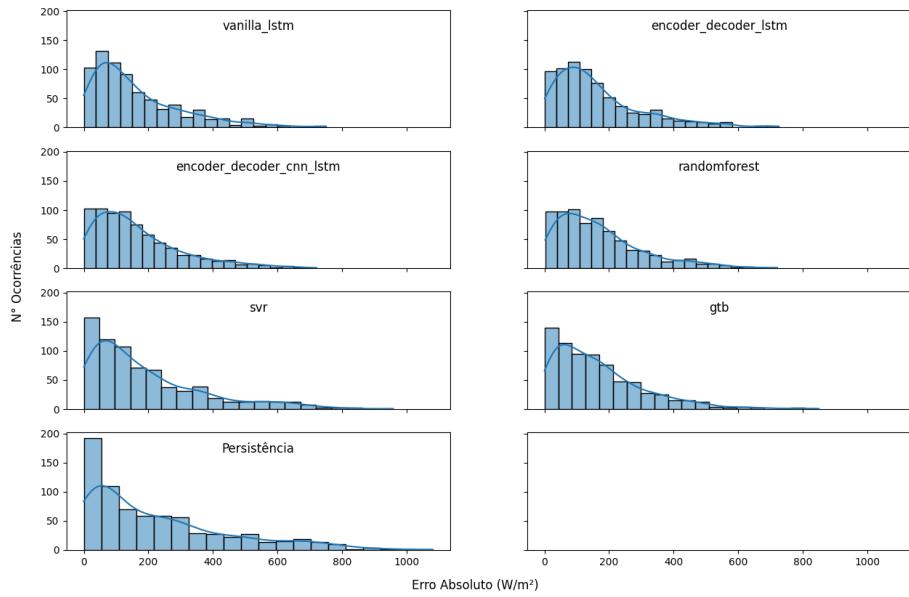
Fig. 39 – Histograma do MAE médio para hora = 12:00 UTC-0



Fonte: Elaboração Própria

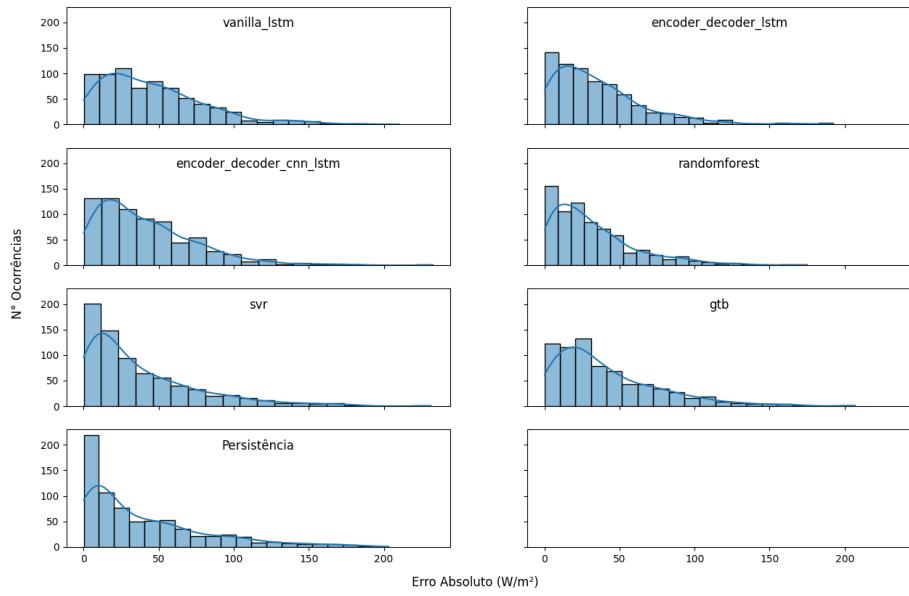
De forma geral, os histogramas mostram que os modelos apresentam uma melhora significativa nas classes de erro maiores, acima de 200 W/m² para 12:00, 400 W/m² para 15:00 e 50 W/m² para 20:00, em relação ao *benchmark* de persistência, com destaque para o modelo de *Random Forest* e as arquiteturas de *RNNAs* com estruturas de codificação e decodificação. Por outro lado, verifica-se uma redução generalizada no número de

Fig. 40 – Histograma do MAE médio para hora = 15:00 UTC-0



Fonte: Elaboração Própria

Fig. 41 – Histograma do MAE médio para hora = 20:00 UTC-0

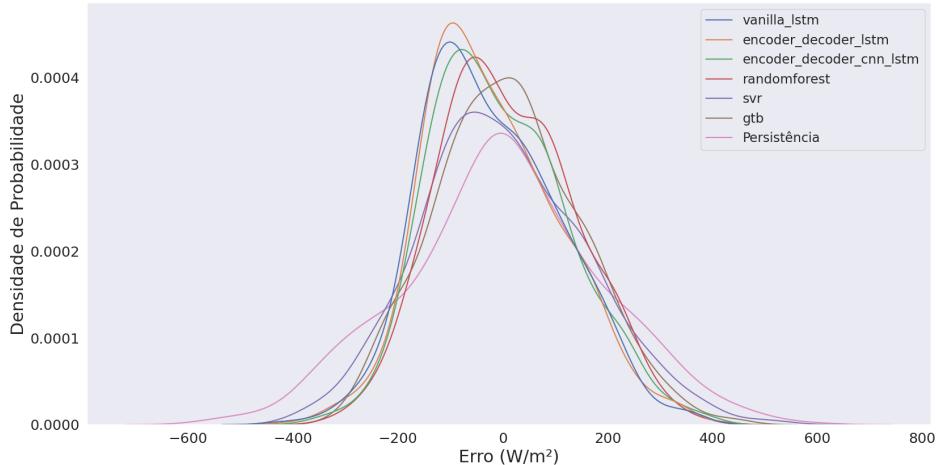


Fonte: Elaboração Própria

ocorrências na primeira classe de erros, indicando a introdução de ao menos um pequeno erro médio em cada previsão que pode ser considerado como um leve viés, em comparação a persistência.

Outra informação que pode ser de grande utilidade é a probabilidade de ocorrência de um determinado valor de erro e o seu sinal, para cada modelo e para as 3 horas escolhidas para análise no item anterior. Com esse propósito, foram traçados os gráficos de estimativa de densidade kernel, estimando a função densidade de probabilidade do erro, podendo ser observados nas Figuras 42, 43 e 44.

Fig. 42 – Estimativa de densidade kernel para hora = 12:00 UTC-0



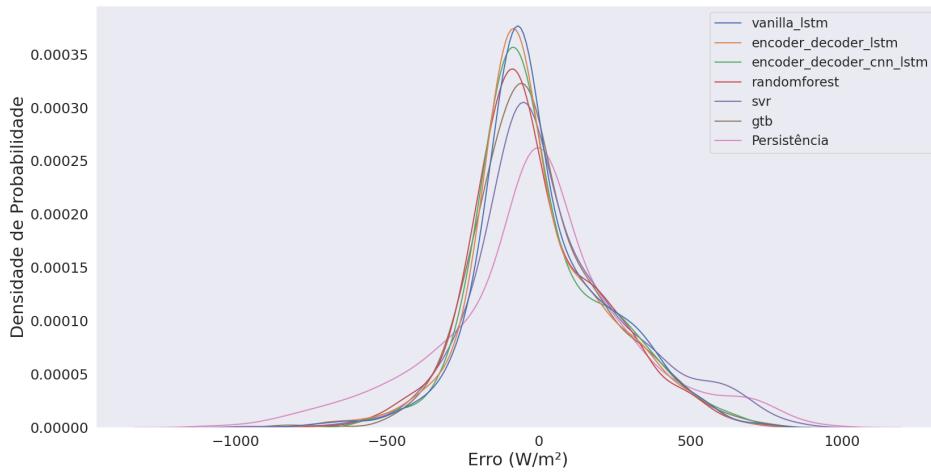
Fonte: Elaboração Própria

Algumas observações importantes podem ser feitas a partir das estimativas de densidade kernel para as três horas selecionadas. Primeiramente, o *benchmark* de persistência apresenta de forma clara uma densidade de probabilidade gaussiana para o erro com média em 0 W/m², como esperado, porém o mesmo não se reproduz para os modelos desenvolvidos.

Para 12:00 UTC-0 (09:00 na hora local), os modelos desenvolvidos apresentam um curvas gaussianas de densidade de probabilidade com média negativa, com maioria entre -50 W/m² e -125 W/m², com exceção do modelo *Gradient Tree Boosting*. Isso significa que os referidos modelos subestimam, em média, a irradiação solar em 12:00 UTC-0.

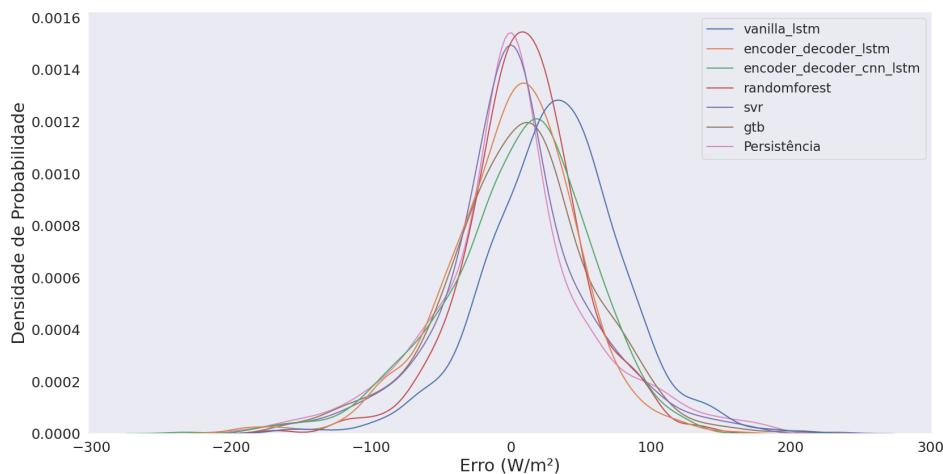
Para 15:00 UTC-0 (12:00 na hora local), o mesmo se reproduz, agora para todos os modelos desenvolvidos, com a mesma magnitude do caso anterior. Para 20:00 UTC-0 (17:00 na hora local), o fenômeno deixa de ocorrer, com a esmagadora maioria dos modelos desenvolvidos com média próxima de 0 W/m², com exceção da arquitetura de rede neural artificial LSTM comum, com um leve viés positivo (sobreestimação).

Fig. 43 – Estimativa de densidade kernel para hora = 15:00 UTC-0



Fonte: Elaboração Própria

Fig. 44 – Estimativa de densidade kernel para hora = 20:00 UTC-0



Fonte: Elaboração Própria

Em segundo lugar, as densidades de probabilidade estimadas demonstram o ganho de desempenho dos modelos desenvolvidos em relação ao *benchmark* de persistência, que se materializa através da redução considerável da variância das gaussianas, as quais são mais concentradas em torno da média, mesmo com o fenômeno de desvio da média. Verifica-se que a frequência de erros mais consideráveis, como, por exemplo, aqueles maiores em magnitude que -300 W/m^2 e 500 W/m^2 para 15:00 UTC-00 são bem menos numerosos nos modelos desenvolvidos que no *benchmark* de persistência.

4.3 Discussão dos Resultados

Os resultados apresentados na seção anterior representam um ganho real dos modelos implementados em relação ao *benchmark* de persistência, conforme demonstrado pelas métricas de desempenho. Logo, os modelos desenvolvidos são informativos o suficiente para superar uma estimativa trivial, possuindo alguma utilidade.

Destaca-se que os modelos com melhor desempenho geral foram os modelos de *Random Forest* e a rede neural artificial com arquitetura *Encoder-Decoder LSTM*, apesar de que, devido às particularidades de cada modelo, é desejável estudar a possibilidade de uma nova estratégia de *Ensemble Learning*, combinando os diferentes modelos desenvolvidos ou até com *Meta Learning*, tomando as saídas e metadados de cada modelo desenvolvido como entradas de um novo modelo de *Machine Learning*, em uma camada superior, para otimização das particularidades observadas.

Não obstante, foram também observadas certas limitações. Em primeiro lugar, os modelos não conseguem estimar de forma precisa variações consecutivas bruscas na irradiação solar horária, fora do padrão natural de sino, o que pode ocorrer devido a um grande número de fatores meteorológicos, como nebulosidade variável, que segue um padrão probabilístico estocástico, apesar de apresentarem ajustes na magnitude da irradiação horária em dias úmidos, com muita nebulosidade ou chuva, o que também pode ser útil. Algumas formas de remediar este problema são a inserção de novas variáveis de previsão numérica, como porcentagem de nebulosidade horária, ou até o uso de técnicas de satélite.

Além disso, há notadamente um viés negativo nas previsões, principalmente nos horários de pico do ângulo de altura solar, resultando em alguma subestimação em média. Este resultado é incomum para problemas de Aprendizado de Máquina, porém possui algumas explicações plausíveis para o problema em questão. Como a amostragem do conjunto de treino não é randômica e sim sequencial, devido à natureza de sequência temporal do problema proposto, é possível que os valores do conjunto de treino, limitados temporalmente, apresentem diferentes características em relação ao conjunto de dados de treino, mais recente. De fato, verificou-se, para 15:00 UTC-0 (pico de altura solar), que os dados mais recentes de medições de irradiação solar apresentavam valores mais extremos com maior frequência do que nos conjuntos de teste, apesar de possuírem valor médio menor. Isso pode acontecer por fatores diversos, incluindo a deterioração do equipamento de medição com o tempo.

Outro tópico de grande importância para discussão é a capacidade de generalização geográfica dos modelos desenvolvidos. Como o desenvolvimento ocorreu para um ponto geográfico específico, estes possuem capacidade de generalização muito baixa para outras localidades, sendo necessário gerar novos modelos para cada local de interesse. Não obstante, foi realizada uma tentativa de criação de um modelo generalizado para uma determinada

região geográfica, no escopo Brasil e Bahia. Para tanto, foram realizados treinamentos de modelos de *Random Forest* com dados de dezenas de estações do INMET, incluindo as variáveis latitude, longitude e altitude. O treino foi realizado com a mesma estruturação de dados do presente trabalho, omitindo os dados de medição da estação alvo, na tentativa de prever a irradiância solar horária para um ponto qualquer dentro da área composta pelas estações locais. Os resultados obtidos nesta abordagem não foram satisfatórios, sequer batendo o modelo de *benchmark*. Sugere-se para tentativas futuras a inclusão de variáveis potencialmente mais informativas quanto a posição geográfica do alvo, como utilização de distâncias entre as estações ao invés de latitude e longitude, ou até o uso da classificação climática em classes.

5 Conclusão

Este trabalho abordou o desenvolvimento de uma ferramenta computacional de previsão intradia da irradiação solar horária para uma determinada localidade, aplicando técnicas de aprendizado de máquina, objetivando também a sua disponibilização em código-aberto servindo como ponto de partida para outros projetos, acadêmicos ou comerciais, neste domínio.

Como evidenciado na revisão bibliográfica, este tipo de previsão com granularidade horária pode ser útil para diversas atividades de regularização da operação do sistema elétrico, como o atendimento de oferta e demanda e o planejamento da transmissão e atuação em mercados do dia seguinte. Em especial, destaca-se a importância deste tipo de trabalho para a estabilidade das redes de energia no país, reduzindo o impacto decorrente do crescimento exponencial da [MMGD](#) no Brasil. Verificou-se também a grande variedade de tipos de modelo já existentes e utilizados na prática, partindo desde modelos estatísticos diretos para horizontes de previsão mais curtos até os métodos baseados em previsões meteorológicas numéricas, no horizonte de dias.

A fundamentação teórica abrangeu conteúdos de diferentes áreas do conhecimento necessárias para a compreensão do trabalho, em sua totalidade. Iniciou-se com a definição da energia solar fotovoltaica e as características da irradiação solar. Em seguida, o tópico de modelos de previsão de irradiação solar no contexto de séries temporais foi discutido, trazendo um panorama dos modelos mais utilizados e suas particularidades. Por consequente, cobriram-se os tópicos de aprendizado de máquina, como as técnicas, procedimentos e conceitos importantes, além do detalhamento de cada algoritmo de aprendizado utilizado no trabalho. Além disso, foram apresentadas as principais métricas de desempenho para modelos de regressão e também as principais bibliotecas de *software* utilizadas para realização do trabalho. Por fim, foram discutidas as principais obras realizadas no escopo do trabalho.

Para treinamento dos modelos, foram utilizados duas fontes de dados diferentes para a mesma localidade em Salvador, Bahia. Os dados de irradiação solar horária foram obtidos da estação automática A401 do [INMET](#), enquanto os dados meteorológicos do modelo numérico MERRA-2 foram obtidos diretamente da interface Nasa Power. Decidiu-se pela não utilização de dados de medição para variáveis meteorológicas em função de dados de modelos numéricos, pois no momento de previsão, não há a certeza dos dados medidos por estações meteorológicas, sendo que o operador do modelo apenas terá acesso a dados de previsão de modelos numéricos, que funcionam de forma probabilística.

Foram desenvolvidos 6 modelos diferentes utilizando os algoritmos de aprendizado

tradicionais *Random Forest*, *Gradient Tree Boosting* e *Support Vector Regression* e as arquiteturas de redes neurais artificiais LSTM Comum, *Encoder-Decoder LSTM*, *Encoder-Decoder CNN LSTM*, além de um modelo trivial de persistência, para comparação. Estes modelos foram alimentados para uma interface de comparação de resultados dentro de um cenário de testes, onde foram gerados, diariamente às 07:00 UTC-0 (04:00 horário local) e durante aproximadamente 2 anos, 728 previsões de 24 horas seguidas de irradiação solar horária.

As métricas dos resultados apresentaram um ganho real dos modelos desenvolvidos em relação ao *benchmark* de persistência. Em especial, os modelos com melhor desempenho foram o *Random Forest* e a rede neural artificial com arquitetura *Encoder-Decoder LSTM*, com Erro Médio Absoluto ([MAE](#)) de 46,00 W/m² e 47,23 W/m², respectivamente e coeficiente de determinação (R^2) de 0,8902 e 0,8886, respectivamente.

Não obstante, foram verificadas algumas limitações dos modelos desenvolvidos, como a incapacidade de prever variações bruscas consecutivas na irradiação solar horária, causadas pela natureza estocástica da nebulosidade, apesar de estimarem variações na magnitude para um período maior quando estas situações se reproduzem. A solução deste problema se encontra na inclusão de novos dados de treino que acrescentem informações suficientes para este tipo de inferência, como exemplo a inserção de novas variáveis de previsão numérica (de outros modelos), tal como a porcentagem de nebulosidade horária ou até mesmo técnicas de satélite.

Por fim, destaca-se que em determinados momentos do dia, alguns modelos apresentavam resultados melhores do que outros, com a situação se invertendo em outros momentos. Esse fato sugere a possibilidade de gerar um modelo implementando a estratégia de *Ensemble Learning* combinando os modelos desenvolvidos ou com a técnica de *Meta Learning*, utilizando as saídas e metadados de cada modelo desenvolvido como entradas para treinamento de um novo modelo de Aprendizado de Máquina, atuando numa camada posterior, para otimização das particularidades observadas.

5.1 Contribuição do Trabalho

Os modelos desenvolvidos com granularidade horária e horizonte temporal de 1 hora até 24 horas apresentam diversas vantagens para o uso por operadores de rede, como para o atendimento da oferta e demanda de energia e para controle da estabilidade da rede. A abordagem desenvolvida neste trabalho permite o uso de dados disponibilizados de forma pública e gratuita, além de *software* de código-aberto, sem a necessidade de instalação de equipamentos ou compras de licenças de software.

Na seção [2.3](#), é feito uma revisão acerca de trabalhos acadêmicos no tópico de desenvolvimento de modelos para previsão de irradiação solar utilizando diversas ferramentas,

com destaque para o Aprendizado de Máquina. Algumas das contribuições que o presente trabalho proporciona em relação à bibliografia pesquisada durante o desenvolvimento deste trabalho são:

- Utilização de uma combinação de dados de medição do INMET com o modelo numérico MERRA-2.
- Disponibilização em código aberto e repositório público de toda a metodologia aplicada para estruturação de um problema no domínio de previsão de irradiação solar horária, permitindo a criação de novos modelos utilizando diferentes fontes de dados.
- Desenvolvimento de modelos de previsão de irradiação solar horária para Salvador, Bahia.
- Dados de desempenho dos modelos e as suas especificações, resultando em novas bases de comparação de resultados.

5.2 Trabalhos Futuros

O presente trabalho resultou no treinamento de 6 modelos abordando diferentes algoritmos de Aprendizado de Máquina, para previsão da irradiação solar horária em Salvador, Bahia, utilizando dados do modelo numérico MERRA-2 e de uma estação do INMET. Diversas possibilidades de melhoria e exploração são levantadas a seguir, ficando ao critério do leitor para realização de trabalhos futuros:

- Desenvolvimento de uma estratégia de *Ensemble Learning*, englobando as saídas dos modelos desenvolvidos.
- Criação de um modelo de meta-regressão, através da aplicação de um algoritmo de *Meta Learning* a partir das saídas e metadados dos modelos desenvolvidos.
- Inclusão de dados de treino de outros modelos de previsão numéricos, como o Global Forecast System (GFS), European Centre for Medium-Range Weather Forecasts (ECMWF), Icosahedral Nonhydrostatic (ICON), dentre outros.
- Inclusão de dados de treino de outras fontes, como métodos de satélite e imagem.
- Desenvolvimento dos modelos com a metodologia proposta para outras localidades, ou um aprofundamento do método geográfico descrito em 4.3.
- Criação de uma interface gráfica em ferramenta de visualização de dados ou via *Dashboard WEB* para uso comercial das previsões de irradiação solar horária.

Referências

- ANEEL. *Micro e Minigeração Distribuída 2ºEd.* 2020. Disponível em: <<https://www.cemig.com.br/wp-content/uploads/2020/08/Caderno-tematico-Micro-e-Minigeracao-Distribuida-2-edicao.pdf>>. Acessado em 09/10/2022. Citado na página 32.
- AWAD, M.; KHANNA, R. Support vector regression. In: *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Berkeley, CA: Apress, 2015. p. 67–80. ISBN 978-1-4302-5990-9. Citado na página 51.
- BOTCHKAREV, A. Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *arXiv preprint arXiv:1809.03006*, 2018. Citado na página 52.
- BROCKWELL, P.; DAVIS, R. *Introduction to Time Series and Forecasting*. [S.l.]: Wiley, 2015. (Wiley Series in Probability and Statistics). ISBN 9780387953519. Citado na página 33.
- BURKOV, A.; LUTZ, M. *The Hundred-Page Machine Learning Book*. [S.l.: s.n.], 2019. ISBN 9781999579500. Citado 11 vezes nas páginas 36, 39, 40, 41, 42, 43, 45, 46, 47, 49 e 50.
- CHOI, D. et al. On empirical comparisons of optimizers for deep learning. *CoRR*, abs/1910.05446, 2019. Citado na página 49.
- CHOLLET, F. et al. *Keras*. 2015. <<https://keras.io>>. Acessado em 09/10/2022. Citado na página 53.
- COREA, F. G. et al. Forecasting short-term solar irradiance based on artificial neural networks and data from neighboring meteorological stations. *Solar Energy*, v. 134, p. 119–131, 2016. Citado na página 54.
- DEBIE, E.; SHAFI, K. Implications of the curse of dimensionality for supervised learning classifier systems: Theoretical and empirical analyses. *Pattern Anal. Appl.*, Springer-Verlag, Berlin, Heidelberg, v. 22, n. 2, p. 519–536, may 2019. ISSN 1433-7541. Citado na página 39.
- DIAGNE, M. et al. Review of solar irradiance forecasting methods and a proposition for small-scale insular grids. *Renewable and Sustainable Energy Reviews*, v. 27, p. 65–76, 2013. ISSN 1364-0321. Citado 4 vezes nas páginas 28, 36, 37 e 54.
- EPE. *Recursos Energéticos Distribuídos*. 2019. Disponível em: <<https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-227/topico-457/GT%20PNE%20-%20RED%20-%20Relat%C3%B3rio%20Final.pdf>>. Acessado em 09/10/2022. Citado na página 27.
- EPE. *Painel de Dados de Micro e Minigeração Distribuída (PDGD)*. 2022. Disponível em: <<https://www.epe.gov.br/pt/publicacoes-dados-abertos/publicacoes/painel-de-dados-de-micro-e-minigeracao-distribuida-pdgd->>. Acessado em 09/10/2022. Citado na página 32.

EPE; MME. *Plano Decenal De Expansão De Energia 2030*. 2020. Disponível em: <https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-490/topico-564/Minuta_do_Plano_Decenal_de_Expansao_de_Energia_2030__PDE_2030.pdf>. Acessado em 09/10/2022. Citado 2 vezes nas páginas 27 e 32.

Eppley Lab. *Standard Precision Pyranometer*. 2022. Disponível em: <<http://www.eppleylab.com/instrument-list/standard-precision-pyranometer/>>. Acessado em 09/10/2022. Citado na página 33.

FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. *Annals of statistics*, JSTOR, p. 1189–1232, 2001. Citado na página 50.

FURTADO, M. I. V. *Redes Neurais Artificiais: Uma Abordagem Para Sala de Aula*. [s.n.], 2019. Acessado em 09/10/2022. Disponível em: <<http://educapes.capes.gov.br/handle/capes/432794>>. Citado 3 vezes nas páginas 45, 46 e 47.

HAYKIN, S.; ENGEL, P. M. *Redes Neurais: Princípios e Prática*. 2. ed. [S.l.]: Bookman, 2001. ISBN 9788573077186. Citado na página 46.

HEGEDUS, S. S.; LUQUE, A. *Handbook of Photovoltaic Science and Engineering*. [S.l.]: John Wiley & Sons, Ltd, 2003. 1-43 p. ISBN 9780470014004. Citado na página 30.

IMHOFF, J. *Desenvolvimento de Conversores Estáticos para Sistemas Fotovoltaicos Autônomos*. 146 p. Dissertação (Mestrado) — Universidade Federal de Santa Maria, 2007. Citado na página 29.

INMAN, R. H.; PEDRO, H. T.; COIMBRA, C. F. Solar forecasting methods for renewable energy integration. *Progress in Energy and Combustion Science*, v. 39, n. 6, p. 535–576, 2013. ISSN 0360-1285. Citado 2 vezes nas páginas 35 e 53.

INMET. *Banco de Dados Meteorológicos do INMET (BDMEP)*. 2022. Disponível em: <<https://bdmep.inmet.gov.br/>>. Acessado em 09/10/2022. Citado na página 56.

IOFFE, S.; SZEGEDY, C. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. [S.l.]: arXiv, 2015. Citado na página 44.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine learning and deep learning. *Electronic Markets*, v. 31, n. 3, p. 685–695, Sep 2021. ISSN 1422-8890. Citado na página 36.

JEON, B. K.; KIM, E.-J. Next-day prediction of hourly solar irradiance using local weather forecasts and lstm trained with non-local data. *Energies*, v. 13, 2020. Citado na página 54.

LIU, H.; MOTODA, H. Feature transformation and subset selection. *IEEE Intelligent Systems and their Applications*, v. 13, n. 2, p. 26–28, 1998. Citado na página 40.

MOOSA, A. et al. Predicting solar radiation using machine learning techniques. In: *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. [S.l.: s.n.], 2018. p. 1693–1699. Citado na página 54.

MORETTIN, P.; TOLOI, C. *Modelos para Previsões de Séries Temporais*. 1. ed. [S.l.]: Instituto de Matemática Pura e Aplicada, 1981. Citado na página 34.

- NASA. *NASA Power: API Pages*. 2022. Disponível em: <<https://power.larc.nasa.gov/api/pages/?urls.primaryName=Hourly>>. Acessado em 09/10/2022. Citado na página 55.
- PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 53.
- PRECHELT, L. Early stopping — but when? In: MONTAVON, G.; ORR, G. B.; MÜLLER, K.-R. (Ed.). *Neural Networks: Tricks of the Trade: Second Edition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 53–67. ISBN 978-3-642-35289-8. Citado na página 44.
- PÉREZ, F. *Características da Série Temporal*. 2022. Disponível em: <<http://leg.ufpr.br/~lucambio/STemporais/STemporaisI.html>>. Acessado em 09/10/2022. Citado na página 34.
- RIBEIRO, B. *ML Solar Forecaster*. 2022. <<https://github.com/b-rbmp/ml-solar-forecaster>>. Acessado em 19/11/2022. Citado na página 75.
- RIFFONNEAU, Y. et al. Optimal power flow management for grid connected pv systems with batteries. *IEEE Transactions on Sustainable Energy*, v. 2, n. 3, p. 309–320, 2011. Citado na página 31.
- ROEDER, L. *Netron, Visualizer for Neural Network, Deep Learning, and Machine Learning Models*. 2017. Acessado em 09/10/2022. Disponível em: <<https://github.com/lutzroeder/netron>>. Citado na página 71.
- RUDER, S. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. Citado na página 49.
- SCHONLAU, M.; ZOU, R. Y. The random forest algorithm for statistical learning. *The Stata Journal*, v. 20, n. 1, p. 3–29, 2020. Citado na página 50.
- SHORTEN, C.; KHOSHGOFTAAR, T. M. A vurvey on image data augmentation for deep learning. *Journal of Big Data*, v. 6, n. 1, p. 60, Jul 2019. ISSN 2196-1115. Citado na página 44.
- SRIVASTAVA, N. et al. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, v. 15, n. 56, p. 1929–1958, 2014. Citado na página 44.
- SU, D.; BATZELIS, E.; PAL, B. Machine learning algorithms in forecasting of photovoltaic power generation. In: *2019 International Conference on Smart Energy Systems and Technologies (SEST)*. [S.l.: s.n.], 2019. p. 1–6. Citado na página 54.
- SUDHARSHAN, K. et al. Systematic review on impact of different irradiance forecasting techniques for solar energy prediction. *Energies*, v. 15, p. 6267, 08 2022. Citado na página 35.
- TEAM, T. pandas development. *pandas-dev/pandas: Pandas*. [S.l.]: Zenodo, 2020. Citado na página 56.
- VILLALVA, M.; GAZOLI, J. *Energia Solar Fotovoltaica - Conceitos e Aplicações*. [S.l.: s.n.], 2012. ISBN 978-85-365-0416-2. Citado 2 vezes nas páginas 30 e 31.

VOYANT, C. et al. Machine learning methods for solar radiation forecasting: A review. *Renewable Energy*, v. 105, p. 569–582, 2017. ISSN 0960-1481. Citado na página 54.

WANG, Q. et al. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, v. 9, n. 2, p. 187–212, abr. 2022. Citado na página 48.

YAMASOE, M.; IWABE, C. *Apostila da Disciplina Meteorologia Física II – ACA 0326*. 2006. Disponível em: <http://www.dca.iag.usp.br/material/akemi/fisicaII/apostila_cap_04.pdf>. Acessado em 09/10/2022. Citado na página 32.

ZHENG, A.; CASARI, A. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. [S.l.]: O'Reilly, 2018. ISBN 9781491953242. Citado na página 38.