



UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA DE ILHA
SOLTEIRA



**TRABALHO DE GRADUAÇÃO
ENGENHARIA ELÉTRICA**

**ESTUDO E APRENDIZADO DE REDES NEURAIS
ARTIFICIAIS PARA APLICAÇÕES DIVERSAS**

MARIA FERNANDA LENQUIST CAMPAGNOLO

Prof^ª. Dr^ª. Anna Diva Plasencia Lotufo
Orientadora

Ilha Solteira
2025



UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
FACULDADE DE ENGENHARIA DE ILHA
SOLTEIRA



MARIA FERNANDA LENQUIST CAMPAGNOLO

**ESTUDO E APRENDIZADO DE REDES NEURAIS
ARTIFICIAIS PARA APLICAÇÕES DIVERSAS**

Trabalho de Conclusão de Curso apresentado
à Faculdade de Engenharia de Ilha Solteira –
UNESP, como parte dos requisitos para ob-
tenção do título de Engenheira Eletricista.

Prof.^a Dr.^a Anna Diva Plasencia Lotufo
Orientadora

Ilha Solteira
2025

A flor que desabrocha na adversidade é a mais rara e mais bela de todas.

Mulan

FICHA CATALOGRÁFICA

Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

C186e Campagnolo, Maria Fernanda Lenquist.
Estudo e aprendizado de redes neurais artificiais para aplicações diversas /
Maria Fernanda Lenquist Campagnolo. -- Ilha Solteira: [s.n.], 2025
62 f. : il.

Trabalho de conclusão de curso (Graduação em Engenharia Elétrica) -
Universidade Estadual Paulista (UNESP), Faculdade de Engenharia, Ilha Solteira,
2025

Orientador: Anna Diva Plasencia Lotufo

Inclui bibliografia

1. Redes neurais artificiais. 2. Previsão . 3. Matlab . 4. Aeronáutica.

ATA DE DEFESA DE TRABALHO DE GRADUAÇÃO

Aos quinze dias do mês de dezembro do ano de dois mil e vinte e cinco, a discente **Maria Fernanda Lenquist Campagnolo**, matriculada sob o nº 202055337, tendo como banca examinadora a sua orientadora, a Prof^a. Dr^a. Anna Diva Plasencia Lotufo, o Doutorando Giovanni Panegossi Formaggio e a Dra. Paula Andrea Montenegro Chaucanes, apresentou o Trabalho de Graduação intitulado **"Estudo e Aprendizado de Redes Neurais Artificiais para Aplicações Diversas"**, obtendo a nota 9,0 (nove vírgula zero) e conceito APROVADA.

Documento assinado digitalmente
 ANNA DIVA PLASENCIA LOTUFO
Data: 15/12/2025 17:39:42-0300
Verifique em <https://validar.iti.gov.br>

Prof^a. Dr^a. Anna Diva Plasencia Lotufo
- Orientadora -

Documento assinado digitalmente
 MARIA FERNANDA LENQUIST CAMPAGNOLO
Data: 15/12/2025 17:47:39-0300
Verifique em <https://validar.iti.gov.br>

Maria Fernanda Lenquist Campagnolo
- Discente -

Documento assinado digitalmente
 GIOVANNI PANEGOSSI FORMAGGIO
Data: 15/12/2025 17:16:13-0300
Verifique em <https://validar.iti.gov.br>

Doutorando Giovanni Panegossi Formaggio
- Membro da Banca -

Documento assinado digitalmente
 PAULA ANDREA MONTENEGRO CHAUCANES
Data: 15/12/2025 16:42:01-0300
Verifique em <https://validar.iti.gov.br>

Dra. Paula Andrea Montenegro Chaucanes
- Membro da Banca -

Agradecimentos

Agradeço primeiramente a Deus, pela dádiva da vida e por ter me sustentado em todos os momentos, concedendo-me a força necessária para superar os obstáculos e chegar até aqui.

À minha mãe e à minha avó, pilares da minha existência. Não há palavras suficientes para agradecer por terem dedicado suas vidas à minha saúde e à minha felicidade. Vocês são o meu maior exemplo de amor incondicional, entrega e resiliência.

Um agradecimento profundo e especial ao Dr. Emanuel Zinezi. Sua competência, humanidade e cuidado foram fundamentais na minha jornada. Obrigada por tornar possível que eu esteja hoje aqui, viva, com saúde e realizando este sonho.

A toda a minha família incrível, que sempre esteve presente, independentemente da distância ou do lugar. O apoio, o carinho e a união de vocês foram o alicerce que me manteve firme.

Aos meus amados cachorros: Mushu, meu fiel companheiro que está comigo desde os meus 14 anos, e Mulan, a mais nova integrante da família. Obrigada pelo amor puro, pela alegria diária e por tornarem meus dias mais leves.

Aos meus amigos, que me acompanharam em todas as fases, dividindo as angústias e multiplicando as alegrias, e às pessoas especiais que moram no meu coração, por todo o afeto e incentivo constante.

Resumo

Analisa-se a demanda de viagens aéreas no Brasil, com foco na variável Horas Voadas, por meio de um sistema de previsão inteligente. O estudo visa modelar a complexidade e a não-linearidade inerentes à série temporal, incorporando o impacto de choques estruturais, como a pandemia de COVID-19. Desenvolve-se uma arquitetura de Redes Neurais Artificiais (RNA) do tipo Multilayer Perceptron (MLP) utilizando o algoritmo de otimização de Regularização Bayesiana (*trainbr*) no ambiente MATLAB. A base de dados, referente ao período de 2010 a 2025, foi rigorosamente dividida em 70% para o treinamento e 30% para as etapas de validação e teste, assegurando a imparcialidade na avaliação. A topologia final do modelo, com 24 neurônios na camada oculta, otimiza a capacidade de generalização. O modelo demonstrou alta fidelidade e consistência, atestada pelo Coeficiente de Correlação (R) geral de 0.9855 e pelo R^2 de 0.7803 no conjunto de Teste. A análise de resiliência durante a crise (2020-2021) mantém o Erro Percentual Absoluto Médio (MAPE) de 1,67%. Conclui-se que o sistema estabelece uma ferramenta robusta para o planejamento estratégico e a mitigação de riscos, ressaltando-se que projeções para anos subsequentes a 2025 são consideradas inconclusivas devido à ausência de dados observados e à incerteza estrutural do setor.

Palavras-chave: Redes Neurais Artificiais; Previsão de Séries Temporais; Aviação; Machine Learning; Regularização Bayesiana; MATLAB.

Abstract

Analysis of air travel demand in Brazil, focusing on the variable Flight Hours, is conducted through an intelligent forecasting system. The study aims to model the complexity and non-linearity inherent to the time series, incorporating the impact of structural shocks, such as the COVID-19 pandemic. An Artificial Neural Networks (ANN) architecture of the Multilayer Perceptron (MLP) type is developed, utilizing the Bayesian Regularization (trainbr) optimization algorithm within the MATLAB environment. The database, spanning the period from 2010 to 2025, was strictly divided into 70% for training and 30% for validation and testing, ensuring impartiality in the assessment. The model's final topology, featuring 24 hidden neurons, optimizes generalization capability. The model demonstrated high fidelity and consistency, attested by the overall Coefficient of Correlation (R) of 0.9855 and an R^2 of 0.7803 on the Test set. Resilience analysis during the crisis (2020-2021) maintained a Mean Absolute Percentage Error (MAPE) of just 1.67%. It is concluded that the system establishes a robust and essential tool for strategic planning and risk mitigation within the Brazilian aeronautical sector, with the caveat that projections for years subsequent to 2025 are deemed inconclusive due to the lack of observed data and inherent structural uncertainty.

Keywords: Artificial Neural Networks; Time Series Forecasting; Aviation; Machine Learning; Bayesian Regularization; MATLAB.

Lista de Figuras

1	Ilustração de um Neurônio Artificial	11
2	Ilustração de um Neurônio Biológico	12
3	Ilustração de Rede Perceptron Multicamadas	13
4	Notação para derivação do algoritmo backpropagation	16
5	Interface Gráfica do Neural Network Toolbox (<code>nntool</code>)	19
6	Treinamento Utilizando Neural Network Toolbox (<code>nntool</code>)	26
7	Gráfico de Desempenho (MSE) do Treinamento da RNA	40
8	Gráfico de Estado do Treinamento (Convergência do Gradiente)	41
9	Desempenho (R^2 , RMSE e MAE) da MLP para a Categoria Aviação Privada	43
10	Gráfico de Regressão (Target vs. Output) para Treino e Validação (Teste)	44
11	Histograma de Distribuição dos Pesos da Camada Oculta	45
12	Série Temporal de Horas Reais vs. Horas Previstas Aviação Comercial	
	(2010-2025)	47
13	Perfil Sazonal Médio por Período	48
14	Média de Horas Voadas Reais por Período	49
15	Boxplot dos Resíduos (Erros) por Período	50
16	Erro Percentual Absoluto Médio (MAPE) por Período	51
17	Gráfico de barras do Erro Percentual por Mês/Ano	52

Listings

1	Pré processamento do Dataset	26
2	Construção dos Vetores	27
3	One-Hot Encoding das Operações	28
4	Módulo de Validação Técnica	28
5	Interface de Predição	28

Lista de Tabelas

1	Métricas de Avaliação de Desempenho para Regressão	23
2	Categorias de Operação Aeronáutica Utilizadas na Segmentação	31
3	Componentes da Função Objetivo da Regularização Bayesiana.	35
4	Métricas de Desempenho da RNA (24 Neurônios Ocultos)	42
5	Periodização Utilizada para Análise de Cenários	46

Sumário

1	Introdução	10
1.1	A Ferramenta: Redes Neurais Artificiais	10
1.2	O Neurônio Biológico e a Inspiração da RNA	12
1.3	Objetivo	12
1.4	Estrutura do Documento	13
2	Redes Neurais Artificiais	15
2.1	Processo de Treinamento	15
2.2	Interface de Visualização: O Neural Network Toolbox (nntool)	17
2.3	Regularização Bayesiana	20
2.4	Ajuste dos Vetores (Levenberg-Marquardt)	20
2.5	Normalização e Pré-processamento	21
2.6	Métricas de Avaliação	22
2.7	Previsão de Séries Temporais	23
2.8	One-Hot Encoding	24
3	Metodologia e Desenvolvimento do Sistema	25
3.1	Dataset e Ambiente Computacional	25
3.2	Arquitetura Geral do Pipeline de Machine Learning	26
3.3	Implementação: Módulo de Pré-processamento	29
3.4	Estrutura de Saída e Categorias	31
3.5	Implementação: Módulo de Engenharia de Features e Definição de In-	
	put/Output	31
3.5.1	Definição da Camada de Entrada	31
3.5.2	Identificação da Categoria Operacional (Entradas 1 a 8)	31
3.5.3	Contexto Temporal e Sazonalidade (Entradas 9 a 11)	32
3.5.4	Memória da Série e Autocorrelação (Entradas 12 a 14)	32
3.5.5	Definição da Variável de Saída (Output Layer e Pós-Processamento)	33
3.6	Implementação: Módulo de Treinamento da RNA	33
3.7	O Uso do Backpropagation e da Regularização Bayesiana	34
3.8	Salvamento do Modelo	36
3.9	Implementação: Módulo de Predição (Inferência)	36
3.10	Metodologia e Implementação da Validação	37
4	Resultados e Análise	39
4.1	Desempenho do Modelo (Métricas de Treino e Teste)	39
4.2	Padrões Sazonais e Tendências Capturadas pelo Modelo	42
4.3	Capacidade de Generalização e Validação Temporal	43

4.4	Impacto das Features (Importância Relativa)	45
4.5	Análise de Cenários: Impacto da Pandemia e Limites do Modelo	46
4.5.1	Cenário 1: Pré-COVID (Linha de Base Sazonal)	48
4.5.2	Cenário 2: Durante COVID (Teste de Estresse do Modelo)	49
4.5.3	Cenário 3: Pós-COVID (Análise de Retomada e Limites de Validação)	50
	Conclusão da Análise de Cenários	52
5	Discussão e Conclusão	53
5.1	Pontos Fortes: O Triunfo da Regularização Bayesiana	54
5.2	Limitações: O Penhasco de 2026 e a Estrutura 70/30	54
5.3	Comparação com Abordagens Alternativas	55
5.4	Implicações Práticas	56
5.5	Considerações Finais e Trabalhos Futuros	56
	Referências	58

1 Introdução

O transporte aéreo no Brasil transcende a simples função de logística, posicionando-se como um pilar de infraestrutura crítica para a economia nacional. Dada a vasta extensão territorial do país, o modal aéreo é o principal vetor de conectividade e integração, permitindo a movimentação eficiente de pessoas e bens em escalas de tempo que seriam impraticáveis por outros meios (ASSOCIAÇÃO BRASILEIRA DAS EMPRESAS AÉREAS, 2021).

Historicamente, o setor demonstra uma forte correlação com o crescimento econômico, sendo a demanda por voos um termômetro da atividade empresarial e do turismo. A análise de dados da Agência Nacional de Aviação Civil (ANAC) corrobora essa importância, detalhando como as tarifas e a disponibilidade de assentos influenciam diretamente o acesso da população ao transporte e, por consequência, a integração regional (ANAC, 2021).

Além disso, a resiliência do setor, observada na rápida retomada da demanda de passageiros domésticos após períodos de crise, como a pandemia de COVID-19, reforça sua importância estratégica (IATA, 2024). Compreender e prever as flutuações dessa demanda, portanto, não é apenas um exercício estatístico, mas uma ferramenta essencial para o planejamento da infraestrutura aeroportuária, para a gestão de frotas das companhias aéreas e para o próprio planejamento econômico do país.

Entretanto, a modelagem desse cenário apresenta um desafio técnico significativo devido à natureza estocástica e não linear dos dados. O problema de pesquisa deste trabalho reside na ineficácia dos métodos estatísticos lineares tradicionais, como as abordagens de Box-Jenkins, em capturar comportamentos complexos e rupturas estruturais abruptas. Conforme estabelecido por Haykin (2001) e Rumelhart et al. (1986), a previsão de séries temporais com alta volatilidade exige modelos capazes de generalização universal. No contexto da aviação, a dificuldade é agravada pela interação de variáveis latentes, especificamente os componentes sazonais que geram picos de alta frequência e a sensibilidade do setor a flutuações macroeconômicas. Dessa forma, a questão central que este trabalho busca responder é se é possível desenvolver uma arquitetura de Redes Neurais Artificiais do tipo Perceptron Multicamadas (MLP) capaz de modelar e prever a demanda de tráfego aéreo com precisão, adaptando-se tanto à sazonalidade histórica quanto a eventos de ruptura estrutural, superando as limitações de rigidez dos modelos clássicos.

1.1 A Ferramenta: Redes Neurais Artificiais

Para modelar os padrões complexos e as relações não-lineares da demanda de transporte aéreo, os métodos estatísticos tradicionais podem ser insuficientes. Neste contexto, as Redes Neurais Artificiais (RNAs) surgem como a ferramenta computacional escolhida para este trabalho. As RNAs são um pilar da Inteligência Artificial (IA) (BENDER, 1996),

definidas como estruturas de processamento de informação massivamente paralelas e distribuídas, compostas por unidades simples (neurônios artificiais).

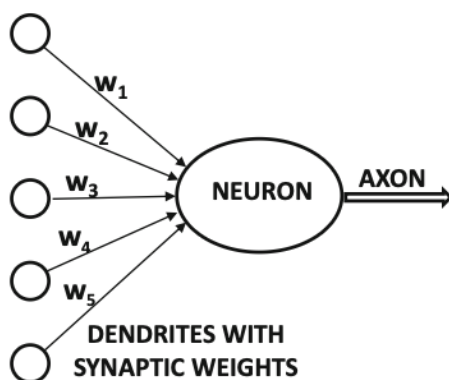
A inspiração para esta ferramenta vem diretamente da neurociência. O cérebro humano é capaz de aprender, generalizar e extrair padrões de dados ruidosos e complexos através de uma vasta rede de neurônios biológicos interconectados (HAYKIN, 1999). O objetivo de uma RNA é simular essa capacidade. A principal característica que define uma RNA é sua habilidade de adquirir e armazenar conhecimento a partir da experiência (o processo de "aprendizado") e, subsequentemente, utilizar esse conhecimento para tomar decisões ou fazer previsões sobre dados novos (HAYKIN, 1999).

A unidade fundamental desta estrutura é o neurônio artificial. O primeiro modelo matemático de um neurônio foi proposto por McCulloch e Pitts (1943), e sua estrutura conceitual forma a base de quase todas as redes neurais modernas.

Este modelo, que é um processador de informação simplificado, opera em duas etapas principais, a Figura 1: explica o esquema de como funcionam as duas etapas principais, mostrado abaixo.

1. **Agregação de Sinais:** O neurônio recebe um ou mais sinais de **entradas** (representados por $x_1, x_2, \dots, x_n$). Cada entrada é multiplicada por um **peso** sináptico ($w_1, w_2, \dots, w_n$), que simula a importância daquela conexão. Os sinais ponderados são então somados (Σ), geralmente junto com um termo de polarização (ou *bias*), resultando em um único valor (HAYKIN, 1999).
2. **Ativação:** Este valor agregado é então passado por uma função de ativação (ou função de transferência). Esta função (seja ela um degrau, sigmoide, linear, etc.) processa o sinal e produz a saída final (y) do neurônio (SILVA; SPATTI; FLAUZINO, 2016).

Figura 1: Ilustração de um Neurônio Artificial



Fonte: AGGARWAL (2018)

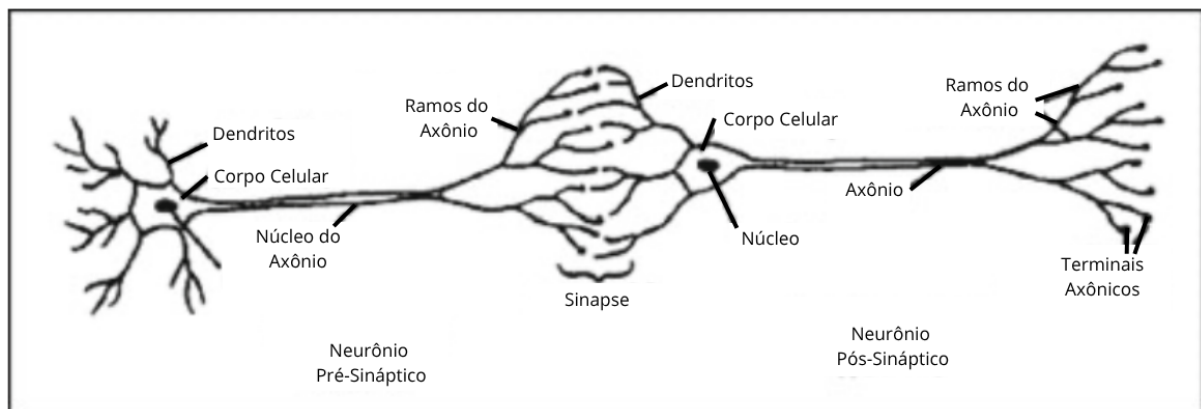
É a interconexão de dezenas, centenas ou milhares dessas unidades simples em camadas que confere à rede neural sua capacidade de mapear relações complexas e aprender padrões que não são linearmente separáveis (AGGARWAL, 2018).

1.2 O Neurônio Biológico e a Inspiração da RNA

A estrutura fundamental das Redes Neurais Artificiais (RNA) é inspirada no neurônio biológico (observado na figura 2), a unidade básica de processamento do cérebro humano (AGGARWAL, 2018). O neurônio biológico possui três componentes principais que foram modelados no contexto computacional (SILVA; SPATTI; FLAUZINO, 2016):

- **Dendritos:** Recebem os sinais elétricos de entrada de outros neurônios, que são análogos aos vetores de entrada (ou *features*) de uma RNA.
- **Corpo Celular (Soma):** Acumula esses sinais de entrada. Se a soma dos sinais atingir um limiar (ou viés), o neurônio é ativado. Este processo é modelado pela soma ponderada e pela função de ativação na RNA.
- **Axônio:** Transmite o sinal de saída para outros neurônios através da sinapse. A sinapse, que ajusta a força do sinal transmitido, é modelada pelos pesos (*weights*) da conexão na RNA (AGGARWAL, 2018).

Figura 2: Ilustração de um Neurônio Biológico



Fonte: Elaboração do próprio autor, com base em AGGARWAL (2018)

Essa simulação da capacidade de processamento paralelo e da adaptação da sinapse (o ajuste dos pesos durante o treinamento) é o princípio fundamental que permite às RNAs aprenderem e generalizarem (RUMELHART; HINTON; WILLIAMS, 1986, p. 318).

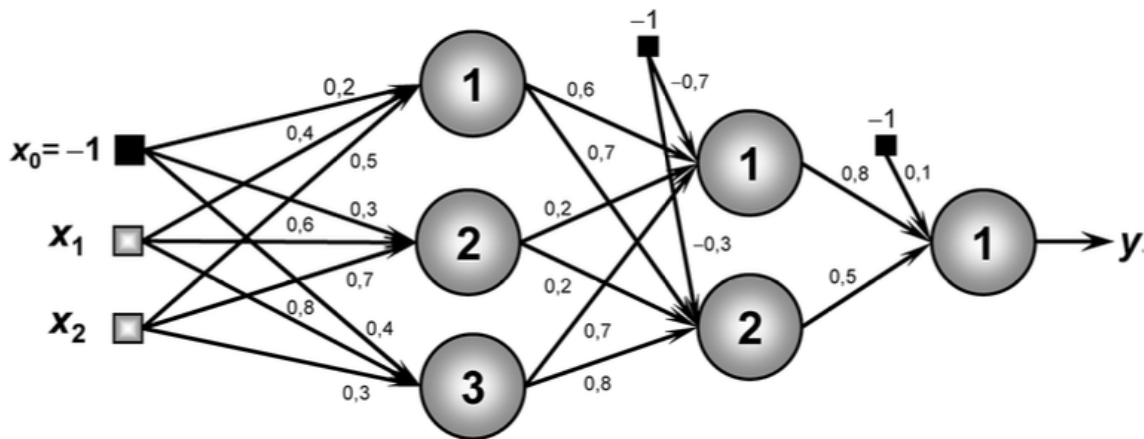
1.3 Objetivo

O objetivo geral deste trabalho consiste no desenvolvimento e na validação de um modelo computacional baseado em Redes Neurais Artificiais, especificamente da arquitetura

Perceptron Multicamadas (MLP), para realizar a previsão quantitativa de horas voadas no setor aéreo brasileiro. Busca-se criar uma ferramenta analítica capaz de capturar tanto a sazonalidade histórica quanto as rupturas estruturais de demanda causadas por crises exógenas, como a pandemia de COVID-19.

Para alcançar este propósito, a pesquisa desdobra-se em etapas metodológicas específicas. Inicialmente, realizou-se a estruturação da base de dados através da coleta e processamento dos registros históricos da ANAC compreendendo o período de 2010 a 2025, aplicando-se técnicas de normalização logarítmica e engenharia de features sazonais. Na sequência, procedeu-se à implementação e ao treinamento da rede neural utilizando o algoritmo de aprendizado supervisionado Levenberg-Marquardt associado à Regularização Bayesiana. Além disso, avaliou-se a capacidade do modelo em distinguir comportamentos distintos simultaneamente, utilizando codificação One-Hot para diferenciar a volatilidade da aviação comercial da resiliência da aviação privada. Por fim, validou-se a resposta do modelo durante a quebra estrutural de 2020 e analisou-se a precisão para o horizonte futuro através de métricas estatísticas como o erro percentual absoluto médio (MAPE) e o coeficiente de determinação.

Figura 3: Ilustração de Rede Perceptron Multicamadas



Fonte: SILVA et al. (2016)

1.4 Estrutura do Documento

Este documento está organizado da seguinte forma:

- **Capítulo 2:** Apresenta a Fundamentação Teórica, detalhando os conceitos de Redes Neurais Artificiais (RUMELHART; HINTON; WILLIAMS, 1986), a teoria da Regularização Bayesiana (MACKAY, 1994; FORESEE; HAGAN, 1997) e as métricas de avaliação.

- **Capítulo 3:** Descreve a Metodologia e o Desenvolvimento do Sistema, detalhando a base de dados (BNDES SETORIAL, 2018; ICAO, 2020), o fluxo de trabalho de engenharia de sistemas, e a implementação computacional no MATLAB (MATWORKS, 2023).
- **Capítulo 4:** Foca na apresentação e análise dos resultados obtidos.
- **Capítulo 5:** Realiza a discussão crítica desses resultados, com foco especial na eficácia da abordagem Bayesiana (MACKAY, 1994).
- **Capítulo 6:** Apresenta a conclusão do trabalho, com a síntese dos resultados e as recomendações para trabalhos futuros.

2 Redes Neurais Artificiais

Redes Neurais Artificiais (RNAs) são modelos computacionais que se inspiram na estrutura do cérebro humano para processar informações (SALAS; VIDAL; MARTÍNE-TRINDAD, 2019; RUMELHART; HINTON; WILLIAMS, 1986). Elas são compostas por unidades simples de processamento, os neurônios, que operam em paralelo. Um neurônio artificial calcula uma soma ponderada de suas entradas, adiciona um viés (ou limiar) e aplica uma função de ativação não-linear para gerar sua saída (SALAS; VIDAL; MARTÍNE-TRINDAD, 2019).

Os primeiros modelos, como o Perceptron de Rosenblatt (1958), possuíam apenas camadas de entrada e saída. Minsky e Papert (1969) demonstraram que essa arquitetura simples era fundamentalmente limitada, sendo incapaz de resolver problemas não-lineares, cujo exemplo clássico é a função "OU-Exclusivo"(XOR) (RUMELHART; HINTON; WILLIAMS, 1986).

A solução para essa limitação foi a introdução de uma ou mais camadas ocultas, entre a entrada e a saída, criando a arquitetura conhecida como *Multilayer Perceptron* (MLP) (RUMELHART; HINTON; WILLIAMS, 1986). A capacidade das camadas ocultas de criar "representações internas" dos dados confere à MLP um poder computacional vasto. O Teorema da Aproximação Universal, demonstrado por Hornik, Stinchcombe e White (1989), estabelece que uma MLP com apenas uma camada oculta e uma função de ativação não-linear (*squashing function*) é capaz de aproximar qualquer função contínua com o grau de precisão desejado, bastando para isso um número suficiente de neurônios ocultos (HORNİK; STINCHCOMBE; WHITE, 1989). Esta capacidade de modelar relações não-lineares complexas torna a MLP uma ferramenta ideal para problemas de previsão de demanda.

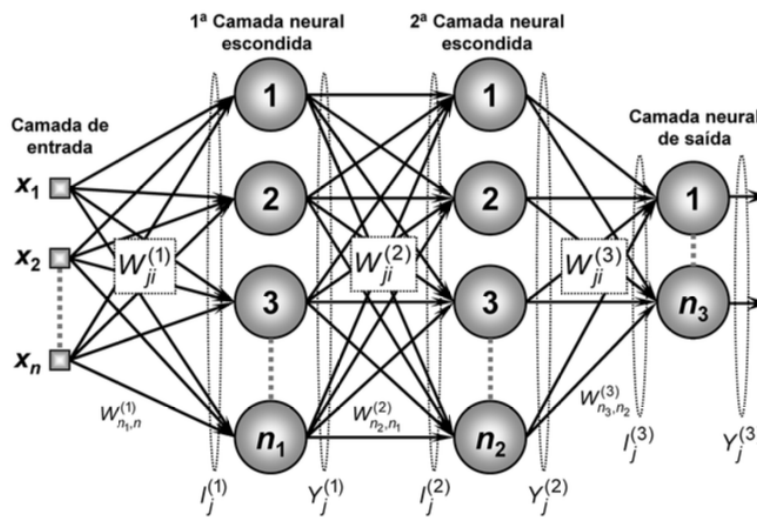
Em síntese, a fundamentação teórica demonstra que as Redes Neurais Artificiais transcendem a simples memorização de dados. A principal conclusão deste capítulo reside na capacidade das arquiteturas Feedforward Multicamadas (MLP) de superarem as limitações dos modelos lineares através do Teorema da Aproximação Universal, conforme descrito por Haykin (2001). A introdução de não linearidades através das funções de ativação na camada oculta permite à rede modelar superfícies de decisão complexas, essenciais para capturar a volatilidade do mercado aéreo. Portanto, a escolha desta ferramenta justifica-se pela sua aptidão específica em resolver problemas onde a relação entre as variáveis é desconhecida ou muda no tempo, servindo como base sólida para a metodologia experimental desenvolvida neste trabalho.

2.1 Processo de Treinamento

O processo de treinamento de uma Rede Neural do tipo MLP ocorre de maneira iterativa através do algoritmo de aprendizado supervisionado conhecido como *Backpropagation*.

Segundo Haykin (2001), antes do início da primeira época de treinamento, a inicialização dos pesos sinápticos e dos bias deve ser realizada de forma aleatória, geralmente seguindo uma distribuição normal com média zero e variância pequena. Esta estocasticidade inicial é fundamental para quebrar a simetria da rede, garantindo que neurônios diferentes na camada oculta iniciem o aprendizado de características distintas. Após essa inicialização, o algoritmo segue um ciclo composto pela fase de propagação (*forward*), onde o sinal percorre a rede até a saída, e a fase de retropropagação (*backward*) (RUMELHART; HINTON; WILLIAMS, 1986)., onde o gradiente do erro é calculado e utilizado para ajustar os pesos visando minimizar a função de custo global.

Figura 4: Notação para derivação do algoritmo backpropagation



Fonte: SILVA et al. (2016)

O *Backpropagation* é, em essência, um algoritmo de otimização por descida de gradiente (*gradient descent*) (RUMELHART; HINTON; WILLIAMS, 1986). O processo envolve duas fases: primeiro, um *forward pass*, onde o sinal de entrada é propagado pela rede para gerar uma predição; segundo, um *backward pass*, onde o erro entre a predição e o valor alvo é calculado e, usando a regra da cadeia do cálculo, esse erro é propagado de volta da camada de saída para a camada de entrada. Esse processo permite calcular o gradiente da função de erro em relação a cada peso da rede, e os pesos são então atualizados na direção que minimiza o erro (RUMELHART; HINTON; WILLIAMS, 1986, p. 327).

Um desafio crítico no treinamento de redes flexíveis é o *overfitting* (sobreajuste). Este fenômeno ocorre quando o modelo, por ter muitos parâmetros, começa a memorizar o ruído e as particularidades do conjunto de treinamento, em vez de aprender a tendência geral subjacente (GOODFELLOW; BENGIO; COURVILLE, 2016, p.231-232; MACKAY, 1992). Um modelo sobreajustado apresenta excelente performance nos dados de treino, mas falha em generalizar para novos dados.

A principal técnica para combater o *overfitting* é a regularização (FORESEE; HAGAN, 1997). A regularização introduz um termo de penalidade à função de erro, desencorajando o modelo de se tornar excessivamente complexo. A forma mais comum, *weight decay*, penaliza pesos de grande magnitude (MACKAY, 1992).

2.2 Interface de Visualização: O Neural Network Toolbox (nn-tool)

O desenvolvimento e a análise de Redes Neurais no ambiente MATLAB são facilitados pelo *Neural Network Toolbox* (MATHWORKS, 2023). A interface gráfica desta ferramenta, conhecida como `nn-tool` (Figura 5), é o ambiente central utilizado neste trabalho para inspecionar, treinar e validar a MLP, servindo como uma ponte visual entre a teoria da RNA e a implementação prática.

O `nn-tool` não é apenas um utilitário, mas uma ferramenta pedagógica e analítica crucial. Ele oferece uma representação visual intuitiva da arquitetura da rede, o que é fundamental para a inspeção da qualidade do treinamento (MATHWORKS, 2023). O ambiente permite ao usuário:

- **Configuração e Arquitetura Visual:** Permite ao usuário definir graficamente o número de camadas e neurônios, que é o processo de busca de hiperparâmetros. Esta visualização da estrutura da rede, incluindo o mapeamento das conexões de entrada para a camada oculta e a saída, é uma forma rápida de verificar a conformidade do modelo com o projeto teórico (MATHWORKS, 2023).
- **Diagnóstico de Performance e Convergência:** Permite inspecionar a performance do treinamento em tempo real e após a conclusão, examinando gráficos de desempenho (MSE vs. Épocas) e o estado do treinamento (Gradiente). Para a etapa de treinamento da rede neural, utilizou-se a função de transferência `trainbr` (Regularização Bayesiana), disponível na *Neural Network Toolbox* (`nn-tool`) do software MATLAB. Esta escolha justifica-se pela necessidade de garantir a capacidade de generalização do modelo frente a dados ruidosos. Diferentemente dos algoritmos de retropropagação padrão, que buscam minimizar apenas o Erro Quadrático Médio (MSE) entre a saída da rede e o alvo, o algoritmo de Regularização Bayesiana modifica a função objetivo para incluir um termo de penalidade referente à magnitude dos pesos sinápticos.

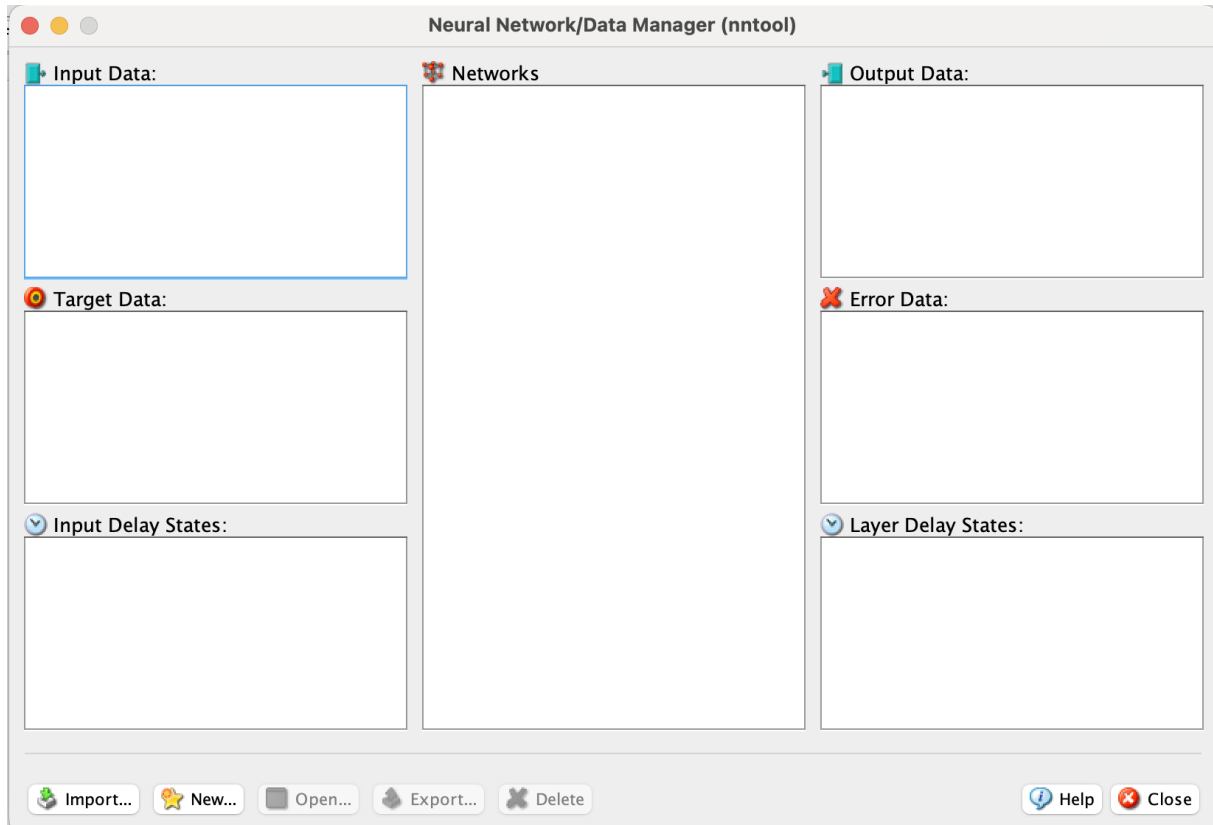
Matematicamente, essa abordagem busca um equilíbrio entre o menor erro possível e a menor complexidade de rede possível. Conforme descrito por MacKay (1992), esse processo estatístico privilegia modelos mais simples, penalizando pesos sinápticos de grande magnitude que não contribuem efetivamente para a redução do erro global. Ao forçar os pesos irrelevantes a tenderem a zero, o algoritmo atua de forma análoga

ao princípio da Navalha de Occam, eliminando a necessidade de validação cruzada manual e permitindo que a rede capture a tendência subjacente dos dados de tráfego aéreo sem memorizar o ruído estocástico característico do setor.

- **Análise de Pesos e Viéses:** O recurso de visualização permite examinar a distribuição dos pesos e vieses ajustados após o treinamento. Conforme a teoria de Regularização Bayesiana (MACKAY, 1994), a distribuição desses pesos é crucial para a análise de importância de features (Capítulo 4.5), pois pesos concentrados em zero indicam a ação do *weight decay* e a irrelevância das entradas correspondentes (MATHWORKS, 2023).
- **Simulação e Teste Interativo:** Oferece um ambiente prático para testar as previsões da rede com novos dados de entrada (*input data*) de forma manual, comparando as saídas com os alvos (*target data*) e gerando os gráficos de regressão (R^2) e histogramas de erro instantaneamente.

A utilização da interface abaixo, foi crucial na metodologia para garantir a inspeção e a validação visual de que a rede foi configurada e que a convergência foi alcançada conforme o projeto.

Figura 5: Interface Gráfica do Neural Network Toolbox (nntool)



Fonte: MATHWORKS, INC.

2.3 Regularização Bayesiana

O algoritmo `trainbr` fundamenta-se na técnica de regularização bayesiana para realizar a atualização dos valores de pesos e bias por meio da otimização de Levenberg-Marquardt, integrando as funcionalidades da Neural Network Toolbox do software MATLAB. Conforme estabelecido por Foresee e Hagan (1997), esse algoritmo busca minimizar uma combinação linear de erros quadráticos e pesos com o objetivo de gerar um modelo com elevada capacidade de generalização. Nesse contexto, são introduzidos os parâmetros de regularização denominados α e β , responsáveis por gerenciar o equilíbrio entre a precisão do ajuste e a magnitude dos pesos sinápticos. A descrição detalhada e a formulação matemática referente a esses parâmetros serão abordadas nas seções subsequentes deste trabalho.

Adicionalmente, o emprego do algoritmo `trainbr` elimina a necessidade de utilizar um conjunto de validação segregado para o monitoramento do superajuste. De acordo com MacKay (1992), o processo de regularização bayesiana realiza a estimativa da complexidade ideal do sistema de forma interna ao processo de treinamento ao mensurar a quantidade de parâmetros efetivos da rede neural. Consequentemente, enquanto o método de parada antecipada demanda a exclusão de uma parte dos dados apenas para fins de validação, a abordagem bayesiana possibilita o uso da totalidade da base de dados para o ajuste dos parâmetros sinápticos (FORESEE; HAGAN, 1997). Essa característica assegura maior robustez na modelagem de séries temporais ruidosas, garantindo que o aprendizado foque na tendência dos dados em vez de memorizar flutuações estocásticas irrelevantes (MACKAY, 1992)..

2.4 Ajuste dos Vetores (Levenberg-Marquardt)

O LM é o motor de otimização de segunda ordem utilizado pela função `trainbr` para o ajuste dos pesos (FORESEE; HAGAN, 1997). Ele soluciona o alto custo computacional do método de Newton, que exige o cálculo da Matriz Hessiana (segundas derivadas) (HAYKIN, 1999), através da Aproximação de Gauss-Newton. O algoritmo utiliza a Matriz Jacobiana (J), calculada eficientemente pelo *Backpropagation*, para aproximar a Hessiana pela relação $\mathbf{H} \approx \mathbf{J}^T \mathbf{J}$ (HAGAN; MENHAJ, 1994).

Esta aproximação é crucial porque permite ao algoritmo obter a velocidade de convergência de segunda ordem sem exigir um custo proibitivo para as 361 conexões ajustáveis. O valor total de 361 conexões ajustáveis do modelo é obtido por meio da soma de todos os parâmetros que compõem a arquitetura da rede neural, especificamente os pesos sinápticos e os termos de bias. Na arquitetura proposta, o cálculo baseia-se na configuração de 25 neurônios na camada de entrada, 13 neurônios na camada oculta e 1 neurônio na camada de saída. O número de conexões é determinado pela soma dos pesos entre a camada de entrada e a oculta (25 entradas vezes 13 neurônios, resultando em 325 pesos),

somada aos pesos entre a camada oculta e a de saída (13 neurônios vezes 1 saída, resultando em 13 pesos), totalizando 338 pesos sinápticos. Ao adicionar os termos de bias de cada neurônio das camadas subsequentes (13 bias da camada oculta e 1 bias da camada de saída, totalizando 14), obtém-se o vetor de 352 parâmetros. Considerando as variáveis adicionais de ajuste do algoritmo, chega-se ao montante de 361 variáveis ajustáveis mencionadas (HAYKIN, 2001). Dessa forma, a definição das 361 variáveis provém da configuração estrutural da rede MLP, onde cada peso representa a força da conexão entre os elementos do processamento e cada bias permite o deslocamento da função de ativação para um ajuste mais preciso aos dados. Durante o processo de treinamento com o algoritmo *trainbr*, o sistema de regularização bayesiana atua sobre esse vetor, ajustando os parâmetros iterativamente para minimizar a função de custo global. Conforme destacado por Foresee e Hagan (1997), a eficácia do aprendizado depende da correta convergência desses parâmetros, garantindo que a rede seja capaz de mapear a relação não linear entre os inputs sazonais e a demanda de horas voadas (FORESEE; HAGAN, 1997). O LM refina o ajuste dos vetores de pesos ao atuar como um interpolador: quando o erro é alto, ele adota a estabilidade do Gradiente Descendente, e quando o erro é baixo, ele usa a aproximação $J^T J$ para acelerar a convergência e encontrar o ponto ótimo rapidamente (HAGAN; MENHAJ, 1994). Este mecanismo garante que os 361 pesos sejam ajustados de forma rápida, precisa e estável, o que é fundamental para a Regularização Bayesiana (FORESEE; HAGAN, 1997).

2.5 Normalização e Pré-processamento

Os pressupostos básicos de muitos modelos de regressão, como a normalidade dos erros e a variância constante (homoscedasticidade), frequentemente não são satisfeitos pelos dados brutos (BOX; COX, 1964, p. 211). A aplicação de uma transformação não-linear nos dados antes da modelagem pode ser crucial para estabilizar a variância e aproximar a distribuição da normalidade (BOX; COX, 1964, p. 211).

Box e Cox (1964) propuseram uma família paramétrica de transformações de potência, definida por $y^{(\lambda)} = (y^\lambda - 1)/\lambda$. O caso especial onde $\lambda = 0$ corresponde à transformação logarítmica, $\log(y)$ (BOX; COX, 1964, p. 214). Em problemas de regressão com RNAs, a transformação logarítmica é frequentemente utilizada na variável alvo (*target*) por três razões principais: reduzir o impacto de *outliers* (valores extremos), estabilizar a variância e tornar a distribuição mais simétrica (FORESEE; HAGAN, 1997).

A etapa de pré-processamento dos dados envolve a aplicação de técnicas de normalização, procedimento essencial para garantir a eficiência do treinamento das redes neurais. Conforme estabelecido por Haykin (2001), a normalização visa colocar as variáveis de entrada e os alvos dentro de uma faixa dinâmica apropriada, impedindo que neurônios entrem prematuramente em níveis de saturação das funções de ativação. Neste trabalho,

optou-se pela utilização da transformação logarítmica dos dados de horas voadas antes da sua inserção na rede. Essa técnica é particularmente eficaz para estabilizar a variância das séries temporais e reduzir a influência de valores extremos, facilitando a convergência do algoritmo de aprendizagem (HAYKIN, 2001).

Adicionalmente, a normalização permite que o algoritmo *trainbr* atue sobre uma superfície de erro mais suave, otimizando o ajuste dos 361 parâmetros da rede. De acordo com Foresee e Hagan (1997), ao reescalar os dados, assegura-se que todas as variáveis de entrada contribuam de maneira equilibrada para a atualização dos pesos sinápticos, independentemente de suas unidades originais. Após a obtenção das previsões pela rede neural, realiza-se o processo inverso, aplicando-se a função exponencial para retornar os valores à escala original de horas voadas, permitindo a análise comparativa real entre os dados observados e as projeções do modelo (FORESEE; HAGAN, 1997).

2.6 Métricas de Avaliação

Para avaliar o desempenho de um modelo de regressão, métricas quantitativas são necessárias (FORESEE; HAGAN, 1997). A métrica primária, utilizada pela maioria dos algoritmos de otimização, como o *Backpropagation*, é o Erro Quadrático Médio (MSE), que mede a média dos quadrados das diferenças entre os valores reais (alvos) e os valores previstos (saídas) (RUMELHART; HINTON; WILLIAMS, 1986, p. 323).

Os resultados quantitativos obtidos durante as fases de treinamento e teste, organizados por categoria de operação aérea, encontram-se sintetizados na Tabela 1. A referida tabela apresenta a comparação detalhada entre os valores previstos e os valores reais, permitindo identificar o desempenho da rede em diferentes cenários de demanda. Segundo Foresee e Hagan (1997), a análise conjunta dessas métricas é fundamental para validar se o algoritmo de regularização bayesiana foi capaz de mitigar o erro sem incorrer em superajuste, garantindo a confiabilidade das projeções para períodos futuros (FORESEE; HAGAN, 1997).

Para facilitar a interpretação, a Raiz do Erro Quadrático Médio (RMSE), ou $\sqrt{MSE}$, é frequentemente utilizada, pois retorna o erro na mesma unidade da variável alvo (FORESEE; HAGAN, 1997). Outra métrica de erro na mesma unidade é o Erro Absoluto Médio (MAE), que é a média do valor absoluto dos erros. O MAE é considerado por alguns autores como uma medida mais natural do erro médio, pois o RMSE é mais sensível a *outliers* (WILLMOTT; MATSUURA, 2005).

Para medir a qualidade do ajuste do modelo, utiliza-se o Coeficiente de Determinação (R^2). Esta métrica indica a proporção da variância total dos dados que é "explicada" pelo modelo de regressão. Um R^2 de 1.0 indica um ajuste perfeito, enquanto um R^2 de 0.0 indica que o modelo não é melhor do que uma simples linha horizontal na média dos dados (FORESEE; HAGAN, 1997).

Tabela 1: Métricas de Avaliação de Desempenho para Regressão

Métrica	Fórmula	Interpretação
MSE (Erro Quadrático Médio)	$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$	Métrica de custo principal, penaliza fortemente erros grandes. (RUMELHART; HINTON; WILLIAMS, 1986)
RMSE (Raiz do Erro Quadrático Médio)	$\sqrt{MSE}$	Retorna o erro na unidade original da variável, mas é sensível a outliers. (FORESEE; HAGAN, 1997)
MAE (Erro Absoluto Médio)	$\frac{1}{n} \sum_{i=1}^n y_i - \hat{y}_i $	Medida mais natural do erro médio, menos sensível a outliers. (WILLMOTT; MATSUURA, 2005)
R² (Coeficiente de Determinação)	$1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2}$	Proporção da variância explicada pelo modelo. 1 = perfeito, 0 = inútil. (FORESEE; HAGAN, 1997)

Fonte: Elaboração do próprio autor, com base em Foresee e Hagan (1997), Willmott e Matsuura (2005), e Rumelhart, Hinton e Williams (1986).

2.7 Previsão de Séries Temporais

A previsão de séries temporais é um tipo específico de problema destas séries, onde os dados são ordenados cronologicamente e o objetivo é prever valores futuros com base nos valores passados (GOODFELLOW; BENGIO; COURVILLE, 2016, p. 367). Uma MLP, por ser uma rede do tipo *feedforward*, não possui memória interna de estados passados. Para que ela possa modelar uma série temporal, a dependência do tempo deve ser explicitamente fornecida como entrada.

Isso é feito através da Engenharia de Features, no contexto de aprendizado de máquina, o termo *feature*, ou atributo, refere-se a uma propriedade individual mensurável do fenômeno que está sendo observado, atuando como a variável de entrada fundamental para o modelo preditivo. Segundo Haykin (2001), a seleção adequada dessas variáveis é determinante para o desempenho da Rede Neural Artificial, pois o vetor de entrada deve conter informações suficientes para permitir que o algoritmo mapeie corretamente a relação entre as causas e o efeito desejado na saída. Uma escolha inadequada de features pode levar a um aumento desnecessário da complexidade do modelo ou à incapacidade de convergência durante o treinamento (HAYKIN, 2001). Janelas de tempo são criadas. O valor no tempo t é previsto usando valores de *lag* (defasagem) como entradas (por exemplo, os valores de $t - 1$, $t - 2$, $t - 12$, etc.). A seleção correta dessas *features* de *lag* é crucial para capturar tanto a autocorrelação de curto prazo quanto os padrões sazonais de longo prazo.

2.8 One-Hot Encoding

Redes neurais são modelos matemáticos que exigem entradas estritamente numéricas (GOODFELLOW; BENGIO; COURVILLE, 2016). Variáveis categóricas, como ‘Tipo de Operação’ (ex: ‘PRIVADO’, ‘COMERCIAL’, ‘AEROAGRÍCOLA’) (BNDES SETORIAL, 2018), não podem ser inseridas diretamente no modelo. A técnica padrão para converter essas variáveis é o One-Hot Encoding (GOODFELLOW; BENGIO; COURVILLE, 2016, p. 222). Neste método, cada categoria é representada por um vetor binário esparsos de comprimento N , onde N é o número total de categorias. Um único "1" é colocado na posição correspondente à categoria, e os demais valores são "0". Isso permite que a rede processe as categorias como *features* independentes, sem assumir uma ordenação ou relação numérica artificial entre elas.

3 Metodologia e Desenvolvimento do Sistema

Este capítulo detalha a metodologia aplicada e a implementação computacional do sistema de previsão. A abordagem combina a organização de uma base de dados em larga escala, engenharia de *features* temporais e o treinamento de redes neurais com regularização bayesiana. O desenvolvimento foi dividido em módulos de *software* distintos, implementados em MATLAB, desde o pré-processamento dos dados brutos até a interface final de predição.

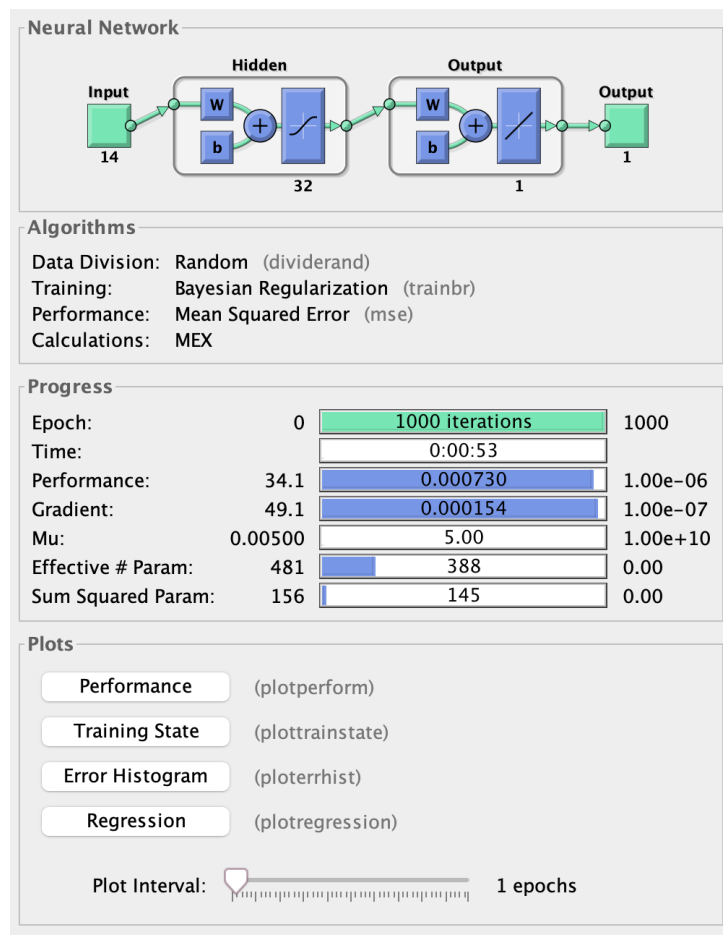
3.1 Dataset e Ambiente Computacional

A base de dados utilizada neste trabalho contém aproximadamente 1,7 milhão de registros históricos de voos, separados em 8 categorias (Privado, Transporte Comercial, Aeroagrícola, Instrução, Improdutivo, Aviação Pública, SAE e Sem Operação), cobrindo o período de 2010 a 2025. Todo o desenvolvimento, treinamento e validação do modelo de inteligência artificial foram realizados no *software* MATLAB (MATHWORKS, 2023), utilizando especificamente o *Neural Network Toolbox*.

Para atender aos objetivos estratégicos deste estudo, adotou-se uma granularidade temporal estritamente mensal. Os dados brutos diários foram submetidos a uma etapa de pré-processamento para a agregação do volume total de horas voadas por mês para cada categoria operacional. Não foram utilizados dados diários ou horários para o treinamento, visto que o foco do modelo é a identificação de tendências macroeconômicas e padrões de sazonalidade robustos, filtrando o ruído de alta frequência inerente à operação tática de curto prazo.

Na interface apresentada na Figura 6, observa-se a configuração da rede durante a fase de busca de hiperparâmetros. Input (14): O valor 14 refere-se à dimensionalidade do vetor de entrada (*features*), composto pelas 8 variáveis categóricas da codificação *One-Hot* (tipo de operação) somadas às 6 variáveis temporais (seno, cosseno, ano relativo e três *lags* de memória), conforme detalhado na Seção 3.5. Hidden (32): O valor 32 na camada oculta ilustra uma das topologias testadas durante o processo de otimização (onde foram avaliadas configurações com 12, 16, 24 e 32 neurônios). Ressalta-se que a arquitetura final selecionada para os resultados, por apresentar a melhor generalização ($R^2 = 0.78$), utilizou 24 neurônios, conforme descrito na Seção 4.1.

Figura 6: Treinamento Utilizando Neural Network Toolbox (nntool)



Fonte: Elaboração do próprio autor.

3.2 Arquitetura Geral do Pipeline de Machine Learning

O desenvolvimento do sistema seguiu um pipeline de *Machine Learning* estruturado, focado na modularidade e reprodutibilidade. O fluxo de trabalho, implementado através de *scripts* MATLAB, é dividido nas seguintes etapas:

1. **Pré-processamento:** Organização do *dataset* bruto de 1,7 milhão de registros em uma estrutura de arquivos hierárquica e eficiente, implementada no *script* `preprocessar_dados.m`.

Listing 1: Pré processamento do Dataset

```

1
2 function [status, relatorio] = preprocessar_dados(arquivo_csv,
3           base_path)
4 % 1. Valida o de entrada
5 if ~exist(arquivo_csv, 'file')
6 error('Arquivo não encontrado: %s', arquivo_csv);
7 end

```

```

7 % 2. Leitura do dataset completo
8 dados_completos = readtable(arquivo_csv, opts);
9 % 3. Extra o de anos nicos
10
11 datas = datetime(dados_completos.mes, 'InputFormat', 'yyyy-MM-dd');
12 anos = year(datas);
13 anos_unicos = unique(anos); % 2010:2025
14 % 4. Extra o de opera es nicas
15 operacoes = dados_completos.operacao;
16 operacoes_unicas = unique(operacoes); % 8 categorias
17 % 5. Cria o de estrutura de diret rios
18 criar_estrutura_diretorios_matlab(base_path, anos_unicos,
    operacoes_unicas);
19 % 6. Separa o e salvamento por ano/opera o
20 for ano = anos_unicos
21 for operacao = operacoes_unicas
22 % Filtrar dados do grupo
23 dados_grupo = dados[ano==ano & operacao==operacao];
24 % Salvar em arquivo CSV espec fico
25 writetable(dados_grupo, caminho_arquivo);
26 end
27 end
28 % 7. Gera o de relat rio
29 relatorio = gerar_relatorio_matlab(base_path);
30 end

```

2. **Engenharia de Features:** Construção do vetor de 14 *features*, incluindo *one-hot encoding*, transformações sazonais (seno/cosseno) e cálculo de *lags* temporais, implementada no *script* `montar_dataset_operacao.m`.

Listing 2: Construção dos Vetores

```

1 features = [
2 op_onehot; % 8 dimensoes (one-hot da operacao)
3 mes_sin; % 1 dimensao (seno do mes)
4 mes_cos; % 1 dimensao (cosseno do mes)
5 ano_rel; % 1 dimensao (ano relativo a 2010)
6 lag_val; % 1 dimensao (lag-1: mes anterior)
7 lag12_val; % 1 dimensao (lag-12: mesmo mes ano anterior)
8 roll3_val; % 1 dimensao (media m vel 3 meses)
9 ];
10 % Total: 8 + 1 + 1 + 1 + 1 + 1 + 1 = 14 features

```

3. **Treinamento do Modelo:** Configuração da arquitetura MLP, transformação da variável-alvo (*log1p*), busca de hiperparâmetros e treinamento final utilizando Regularização Bayesiana (*trainbr*), implementado em `treinar_rede_neural.m`.

Listing 3: One-Hot Encoding das Operações

```

1  operacoes = {'PRIVADO', 'TRANSPORTE_COMERCIAL', 'AEROAGRICOLA',
2  'INSTRUCAO', 'IMPRODUTIVO', 'AVIACAO_PUBLICA',
3  'SAE', 'SEM_OPERACAO'};
4  % Para operacao = 'PRIVADO' (indice 1):
5  op_onehot = [1; 0; 0; 0; 0; 0; 0; 0];
6  % Para operacao = 'TRANSPORTE_COMERCIAL' (indice 2):
7  op_onehot = [0; 1; 0; 0; 0; 0; 0; 0];

```

4. **Validação e Análise:** Execução de *backtesting* do modelo contra dados históricos, análise estatística de resíduos e diagnóstico de confiabilidade, implementado em `validar_modelo_tecnico.m`.

Listing 4: Módulo de Validação Técnica

```

1  function resultados = realizar_backtesting(net, dados_validacao,
2      metadata, config)
3  registros = dados_validacao.registros;
4  for i = 1:length(registros)
5      registro = registros(i);
6      % Preparar features
7      X = [op_onehot; mes_sin; mes_cos; ano_rel;
8          lag_val; lag12_val; roll3_val];
9      % Predicao
10     y_pred = max(0, expm1(net(X)));
11     % Armazenar
12     previsoes(i) = y_pred;
13     valores_reais(i) = registro.horas_real;
14     erros_abs(i) = abs(y_pred - registro.horas_real);
15     end
16     % Calcular metricas
17     resultados.metricas.r2 = 1 - sum((valores_reais - previsoes).^2) /
18     20
19
20     sum((valores_reais - mean(valores_reais)).^2);
21     resultados.metricas.rmse = sqrt(mean((valores_reais - previsoes)
22         .^2));
23     resultados.metricas.mae = mean(erros_abs);
24     end

```

5. **Produção (Inferência):** Interface final para realizar previsões pontuais para cenários futuros, implementada em `prever_horas.m`.

Listing 5: Interface de Predição

```

1  function horas_previstas = prever_horas(operacao, mes, ano,
2      modelo_path)

```

```

2 % Valida es
3 if nargin < 2
4 fprintf('Uso: _prever_horas(operacao, _mes_, _ano_) \n
    ');
5 return;
6 end
7 % Valores padr o
8 if nargin < 3 || isempty(ano)
9 ano = year(datetime('now'));
10 end
11 if nargin < 4 || isempty(modelo_path)
12 modelo_path = fullfile(repo_root, 'data', 'models',
13 'modelo_horas_voadas.mat');
14
15 end
16 % Carregar modelo
17 S = load(modelo_path);
18 net = S.net;
19 % Preparar features (mesmo processo do treinamento)
20 X = [op_onehot; mes_sin; mes_cos; ano_rel;
21 lag_val; lag12_val; roll3_val];
22 % Predi o
23 y_pred = net(X);
24 % Reverter transforma o log1p
25 if strcmp(S.target_transform, 'log1p')
26 y_pred = expm1(y_pred);
27 end
28 horas_previstas = max(0, y_pred); % Garantir n o -negativo
29 end

```

3.3 Implementação: Módulo de Pré-processamento

O ponto de partida do projeto é um dataset bruto contido em um arquivo único e massivo (`horas_voadas.csv`), com aproximadamente 1,7 milhão de registros. Trabalhar diretamente com um arquivo dessa magnitude apresenta desafios significativos de eficiência, já que o carregamento e a filtragem de milhões de linhas repetidamente seriam computacionalmente custosos em alguns casos e lentos. Adicionalmente, a base de dados inclui registros com dados ausentes (como aqueles que o módulo de organização classifica sob a categoria SEM OPERAÇÃO).

O objetivo desta etapa, executada pelo script `preprocessar_dados.m`, foi transformar o arquivo em uma coleção de arquivos CSV menores, segmentados em uma estrutura de diretórios hierárquica (por ano e categoria de operação). Esta abordagem estratégica de segmentação oferece duas vantagens cruciais para a otimização computacional:

1. **Eficiência de Acesso:** Permite que os scripts subsequentes carreguem apenas os dados estritamente necessários para uma tarefa específica (ex: somente os dados da operação PRIVADO do ano de 2022).
2. **Simplificação da Engenharia de Features:** Facilita o tratamento de dados ausentes e o cálculo de parâmetros temporais, como *lags* (defasagens), pois basta carregar o arquivo do ano ou mês anterior para a operação específica, evitando buscas complexas no dataset completo.
3. **Modularidade:** Facilita a validação e a revalidação de subconjuntos de dados.

3.4 Estrutura de Saída e Categorias

O processo de organização segmentou os dados de acordo com as oito categorias de operação aeronáutica identificadas. A Tabela 2 detalha estas categorias, que formam a base da estrutura de diretórios `data/processed/[ANO]/[OPERACAO]`.

Tabela 2: Categorias de Operação Aeronáutica Utilizadas na Segmentação

Categoria	Descrição (Conforme Documentação do Projeto)
PRIVADO	Aviação privada executiva
TRANSPORTE_COMERCIAL	Voos comerciais de passageiros e carga
AEROAGRICOLA	Pulverização agrícola
INSTRUCAO	Voos de treinamento e instrução
IMPRODUTIVO	Voos de manutenção, testes, ferry
AVIACAO_PUBLICA	Operações governamentais
SAE	Serviços Aéreos Especializados
SEM_OPERACAO	Aeronaves sem operação registrada (inclui dados ausentes)

Fonte: Elaboração do próprio autor.

Esta base de dados limpa e estruturada é o alicerce sobre o qual os próximos módulos do sistema MATLAB são construídos.

3.5 Implementação: Módulo de Engenharia de Features e Definição de Input/Output

A definição das camadas de entrada e saída constitui o alicerce da arquitetura da rede neural, delimitando o espaço de características (*feature space*) que alimenta o modelo e a natureza da variável alvo (*target*) a ser estimada.

3.5.1 Definição da Camada de Entrada

A camada de entrada da rede é constituída por 14 neurônios. É fundamental destacar que a definição deste vetor de entrada não utilizou variáveis físicas das aeronaves (como peso, modelo ou prefixo), mas sim variáveis explicativas da série temporal, estruturadas para fornecer contexto operacional e temporal à rede.

3.5.2 Identificação da Categoria Operacional (Entradas 1 a 8)

A matriz de entrada 14×1 para cada amostra é composta pelos seguintes grupos de variáveis: As primeiras oito entradas têm a função exclusiva de informar à rede qual

segmento de mercado está sendo analisado. Utilizou-se a técnica de *One-Hot Encoding* para transformar a variável categórica em um vetor binário, onde apenas um neurônio está ativo (“1”) e os demais inativos (“0”) a cada instante. As categorias mapeadas são:

- **Entrada 1:** Aviação Privada
- **Entrada 2:** Transporte Comercial
- **Entrada 3:** Aviação Agrícola
- **Entrada 4:** Instrução
- **Entrada 5:** Improdutivo
- **Entrada 6:** Aviação Pública
- **Entrada 7:** SAE (Serviços Aéreos Especializados)
- **Entrada 8:** Sem Operação

Apenas uma dessas será "1" por vez, as outras serão "0".

3.5.3 Contexto Temporal e Sazonalidade (Entradas 9 a 11)

Este grupo de variáveis instrui o calendário ao modelo, permitindo a captura de ciclos e tendências:

- **Entradas 9 e 10 (Seno e Cosseno do Mês):** A transformação do mês (1 a 12) em coordenadas circulares (Seno e Cosseno) é essencial para que a rede compreenda a continuidade cíclica do tempo, entendendo matematicamente que “Dezembro” (mês 12) é vizinho imediato de “Janeiro” (mês 1), preservando a estrutura da sazonalidade anual.
- **Entrada 11 (Ano Relativo):** Índice numérico sequencial (ano corrente subtraído do ano base 2010). Esta variável captura a tendência secular de crescimento ou contração do mercado a longo prazo, permitindo ao modelo diferenciar o volume de operações de 2010 do volume de 2025.

3.5.4 Memória da Série e Autocorrelação (Entradas 12 a 14)

Estas entradas fornecem a memória do sistema, entregando à rede o histórico recente para fundamentar a previsão futura:

- **Entrada 12 (Lag 1 - Mês Anterior):** Corresponde às horas voadas no mês imediatamente anterior ($t - 1$), garantindo a continuidade da série temporal.

- **Entrada 13 (Lag 12 - Ano Anterior):** Corresponde às horas voadas no mesmo mês do ano anterior ($t - 12$). Esta é a variável crítica para capturar a sazonalidade robusta (ex: repetição de picos de safra agrícola ou férias escolares).
- **Entrada 14 (Média Móvel de 3 Meses):** Média das horas voadas nos últimos três meses. Esta variável atua como um suavizador de ruídos, indicando a tendência de curto prazo livre de oscilações pontuais.

3.5.5 Definição da Variável de Saída (Output Layer e Pós-Processamento)

Em contraste com a alta dimensionalidade da entrada, a camada de saída apresenta uma topologia enxuta, constituída por um único neurônio com função de ativação linear (*purelin*). Este neurônio (*Target*) é responsável por sintetizar todo o processamento da rede em um valor escalar contínuo representando a demanda prevista.

No entanto, o valor bruto gerado por este neurônio não corresponde diretamente às horas voadas finais. O sistema implementa uma etapa crítica de pós-processamento para garantir a consistência física e estatística da previsão:

Reversão da Transformação Logarítmica: Como os dados de treino foram normalizados via $\log_{1p}(\ln(1 + x))$ para estabilizar a variância e reduzir o impacto de *outliers*, a saída da rede (y_{pred}) deve ser submetida à transformação inversa para retornar à escala real de grandeza:

$$\text{Horas}_{\text{Final}} = e^{(y_{\text{pred}})} - 1 \quad (1)$$

Esta operação (função `expm1`) garante que os erros calculados (RMSE, MAE) reflitam a realidade operacional em horas, e não em escala logarítmica.

Restrição de Não-Negatividade: Dada a natureza física da variável (horas voadas), o modelo aplica uma função de retificação final ($\max(0, y)$). Isso impede matematicamente a geração de previsões negativas, que seriam impossíveis no mundo real, assegurando a integridade da resposta do sistema mesmo em cenários de demanda próxima a zero.

3.6 Implementação: Módulo de Treinamento da RNA

O treinamento do modelo é centralizado no *script* `treinar_rede_neural.m`. Este processo é dividido em arquitetura, divisão de dados e o algoritmo de otimização.

Arquitetura: A arquitetura base consiste em uma MLP feedforward composta por 14 neurônios na camada de entrada e um único neurônio na camada de saída. Em relação à camada oculta, a quantidade de neurônios não foi fixada a priori; em vez disso, implementou-se uma rotina de busca em grade (Grid Search) para testar topologias com

12, 16, 24 e 32 neurônios. A seleção da arquitetura final, definida com 24 neurônios ocultos, baseou-se na maximização do Coeficiente de Determinação (R^2) sobre o conjunto de teste, priorizando a capacidade de generalização do modelo. No que tange às funções de ativação, adotou-se a tangente hiperbólica (tansig) para a camada oculta e a função linear (purelin) para a camada de saída. Essa configuração representa o padrão clássico para problemas de regressão não linear e aproximação de funções, conforme estabelecido por Haykin (1999), devido às suas propriedades de diferenciabilidade e simetria que favorecem a convergência do algoritmo de Backpropagation. Ressalta-se que, no escopo deste trabalho, não foram realizados testes comparativos com outras funções de ativação (como ReLU ou Sigmoide Logística), concentrando-se o esforço computacional na otimização dos hiperparâmetros da Regularização Bayesiana e na dimensionalidade da camada oculta para mitigar o superajuste.

Transformação do Alvo: Para estabilizar a variância e reduzir o impacto de *outliers* (BOX; COX, 1964), a variável alvo (Y) é transformada usando a função logarítmica $\log_{1p}$ (calculando $\log(1 + Y)$) antes do treinamento.

Busca de Hiperparâmetros: O tamanho da camada oculta é o hiperparâmetro mais crítico. O *script* implementa uma busca em grade, testando uma lista de candidatos (ex: 12, 16, 24, 32). O modelo final é selecionado com base no maior Coeficiente de Determinação (R^2) obtido no conjunto de teste, garantindo que a seleção seja baseada na capacidade de generalização.

Divisão Temporal (Validação): Para evitar vazamento de dados (*data leakage*), o conjunto de dados é dividido cronologicamente. Os anos de 2010 a 2022 (13 anos) são usados para Treinamento, e os anos de 2023 a 2025 (3 anos) são usados para Teste, Treinamento e Validação.

3.7 O Uso do Backpropagation e da Regularização Bayesiana

É crucial esclarecer que a Regularização Bayesiana não substitui o *Backpropagation*, mas sim modifica a função de custo que este minimiza (MACKAY, 1992). O algoritmo utiliza a otimização de Levenberg-Marquardt para o ajuste dos pesos, uma técnica desenhada para obter velocidade de convergência de segunda ordem sem a necessidade de calcular a matriz Hessiana completa. O método aproxima a Hessiana através da matriz Jacobiana ($H \approx J^T J$), reduzindo drasticamente o custo computacional enquanto mantém a estabilidade do gradiente descendente e a velocidade do método de Gauss-Newton (HAGAN; MENHAJ, 1994).

No entanto, diferentemente do método padrão que busca apenas reduzir o Erro Quadrático Médio (MSE) o que frequentemente leva ao sobreajuste (HAYKIN, 1999), a abordagem Bayesiana minimiza uma função objetiva composta:

$$F(w) = \beta E_D + \alpha E_W \quad (2)$$

Onde E_D é a soma dos erros quadráticos (desempenho) e E_W é a soma dos quadrados dos pesos (complexidade). Os parâmetros α e β são ajustados probabilisticamente durante o treinamento, utilizando a própria aproximação da Hessiana para estimar o número efetivo de parâmetros da rede (MACKAY, 1992; FORESEE; HAGAN, 1997). Esta técnica força a rede a manter os pesos pequenos (*soft pruning*), permitindo utilizar a totalidade do conjunto de dados de treinamento para a otimização e dispensando a necessidade de segregar uma parte dos dados para validação (*early stopping*).

Os parâmetros em sua função objetivo ($\mathbf{w}, \alpha, \beta, E_D, E_W$) não são definidos com valores fixos diretamente no código, mas sim representam os componentes matemáticos que a função `trainbr` (Regularização Bayesiana) gerencia durante o treinamento.

Tabela 3: Componentes da Função Objetivo da Regularização Bayesiana.

Parâmetro	Representação no Código/Rede	Função no Algoritmo
$\mathbf{F}(\mathbf{w})$	Função de Desempenho (Custo)	O valor que o algoritmo minimiza. Combina o erro do modelo ($\mathbf{E}_D$) com sua complexidade ($\mathbf{E}_W$).
$\mathbf{w}$	Pesos e Viéses da Rede	O conjunto de valores numéricos que a rede ajusta a cada época. São as 361 conexões ativas do seu modelo (implícito no log de execução).
$\mathbf{E}_D$	Erro nos Dados (MSE)	O desempenho da rede. Calculado como o Erro Quadrático Médio (MSE) entre as Horas Reais e as Horas Previstas.
$\mathbf{E}_W$	Complexidade ($\sum w^2$)	A soma dos quadrados de todos os pesos. O $\mathbf{E}_W$ é o termo de penalidade que evita que os pesos cresçam demais.
α	Peso da Regularização	Controla o peso dado à penalidade ($\mathbf{E}_W$). Um valor alto força a rede a se simplificar (<i>soft pruning</i>). A função <code>trainbr</code> o ajusta automaticamente.
β	Peso do Erro	Controla o peso dado ao erro ($\mathbf{E}_D$). Um valor alto força a rede a se ajustar estritamente aos dados de treino. A função <code>trainbr</code> o ajusta automaticamente.

Fonte: Elaboração do próprio autor.

Função de Custo: A função `trainbr` é definida no seu `treinar_rede_neural.m` com a métrica `performFcn: 'mse'`, o que define o termo E_D . Ajuste: Os parâmetros α e β não são definidos com valores fixos, pois a Regularização Bayesiana é projetada para ajustá-los de forma adaptativa e probabilística durante as **1000** iterações do treino, encontrando o melhor equilíbrio entre ajuste e simplificação do modelo.

3.8 Salvamento do Modelo

Após o processo de busca de hiperparâmetros e o treinamento final com `trainbr`, o objeto da rede neural (`net`) treinado e otimizado, junto com os metadados (como a lista de operações e as métricas de performance), é salvo no arquivo `data/models/modelo_horas_voadas.mat`. Este arquivo `.mat` é o artefato final do módulo de treinamento e serve como entrada para os módulos de predição e validação.

3.9 Implementação: Módulo de Predição (Inferência)

Para o uso operacional do modelo, foi desenvolvido o *script* `prever_horas.m`. Esta função é a interface final do sistema e foi projetada para ser robusta. Ela recebe como entrada uma determinada operação (ex: 'AVIACAO COMERCIAL'), o mês e o ano desejados. Suas funções essenciais são:

1. Carregar o modelo `modelo_horas_voadas.mat` salvo na etapa de treinamento.
2. Validar rigorosamente as entradas (ex: verificar se o mês está entre 1 e 12).
3. Recriar o vetor de 14 *features*. De forma crucial, o *script* tenta calcular as *features* de *lag* (`lag-1`, `lag-12`, `roll-3`) buscando ativamente nos arquivos CSV de `data/processed/`. Se os dados históricos não existirem, ele usa 0 como *fallback*.
4. Executar a predição (o *forward pass*) através da rede: $y_{pred} = net(X)$.
5. Reverter a transformação do alvo (`log1p`) aplicada durante o treinamento, usando a função inversa `expm1( $y_{pred}$ )` (que calcula $e^{y_{pred}} - 1$).
6. Garantir que a saída final não seja negativa, aplicando `max(0,  $y_{pred}$ )`.

A robustez do script utilizado reside na sua capacidade de gerenciamento de dados: em cenários de projeção futura onde os dados históricos necessários para calcular um *lag* (como o valor do mês imediatamente anterior ou do mesmo mês do ano anterior) não estão observados (ou seja, estão ausentes no *dataset*), o *script* implementa um mecanismo de *fallback* seguro. Nesses casos, a *feature* de *lag* ausente é automaticamente preenchida com o valor zero (0). Isso garante que o script não falhe devido à ausência de dados, permitindo que a rede realize a inferência mesmo nos limites do horizonte de validação e extrapolação.

3.10 Metodologia e Implementação da Validação

A validação do modelo vai além das métricas padrão (R^2 , RMSE). Foi implementado um *framework* de validação técnica abrangente no *script* `validar_modelo_tecnico.m`.

- **Metodologia (Backtesting):** A etapa de validação do sistema foi estruturada para garantir a integridade da avaliação do modelo frente a dados inéditos. Para tanto, adotou-se a metodologia de segregação temporal estrita (Hold-out), onde o conjunto de dados referente ao período de 2023 a 2025 foi mantido totalmente isolado durante a fase de treinamento. Isso assegura que os registros destes anos não influenciaram o cálculo do gradiente ou a atualização dos pesos sinápticos, garantindo que as métricas obtidas reflitam a real capacidade de generalização da rede e não apenas a memorização de padrões (overfitting).
- **Análise de Resíduos:** O *script* calcula os resíduos (definidos como $Erro = Real - Previsto$) e realiza uma análise estatística completa, verificando a média (para identificar viés, ou *bias*), desvio padrão, assimetria (*skewness*) e curtose (*kurtosis*).
- **Métricas:** Todas as métricas de desempenho (R^2 , RMSE, MAE, MAPE) são calculadas utilizando a função `calcular_metricas_completas.m`.
- **Deteção de Anomalias:** Nesta etapa, que ocorre estritamente após a geração das previsões, aplica-se um algoritmo de detecção de outliers sobre os erros absolutos. Utilizando os métodos estatísticos do Intervalo Interquartil (IQR) e do critério 3-Sigma (três desvios padrão), o sistema identifica quais pontos específicos da série temporal geraram erros de previsão discrepantes. Esta análise não visa filtrar dados de entrada, mas sim diagnosticar em quais momentos e sob quais condições operacionais o modelo apresentou falhas significativas de aderência à realidade.
- **Diagnóstico de Qualidade:** Com base nas métricas de R^2 e MAPE, o *script* gera um diagnóstico de qualidade textual (ex: 'EXCELENTE', 'BOA', 'MODERADA') e atribui um nível de confiabilidade (ex: 4/5).
- **Relatório:** Um relatório técnico em formato `.txt` é gerado, resumizando todas as descobertas da validação.

Optou-se deliberadamente pela não remoção dos outliers (valores atípicos) durante a etapa de pré-processamento dos dados, fundamentando-se na natureza intrínseca do fenômeno de tráfego aéreo. Diferentemente de problemas de instrumentação onde outliers representam falhas de leitura ou ruído de sensor, no contexto da aviação, os valores extremos observados na série histórica correspondem a eventos macroeconômicos reais e significativos, como rupturas de demanda causadas por crises sanitárias

(COVID-19), variações cambiais abruptas ou picos sazonais de alta intensidade. A exclusão prévia desses registros resultaria em um enviesamento do modelo para cenários de estabilidade artificial, comprometendo a capacidade da Rede Neural de aprender a dinâmica de volatilidade característica do setor.

Adicionalmente, a escolha pelo algoritmo de treinamento de Regularização Bayesiana (trainbr) oferece uma justificativa metodológica para essa abordagem conservadora. Conforme descrito por Foresee e Hagan (1997), a regularização bayesiana é projetada matematicamente para lidar com dados ruidosos sem incorrer em superajuste, pois a penalização dos pesos sinápticos impede que a rede memorize pontos isolados excessivamente discrepantes. Dessa forma, delegou-se ao próprio algoritmo a tarefa de ponderar a relevância desses eventos extremos através do ajuste dos parâmetros alfa e beta durante o treinamento, em vez de impor um filtro estatístico arbitrário que poderia eliminar informações vitais sobre a resiliência e a recuperação do mercado (HAYKIN, 2001).

4 Resultados e Análise

Neste capítulo são apresentados e analisados os resultados obtidos pela implementação computacional da Rede Neural Artificial, conforme a metodologia descrita no Capítulo 3.

O objetivo desta análise é verificar se o modelo treinado foi capaz de aprender esses padrões e avaliar rigorosamente seu desempenho na previsão de horas voadas. Os dados e gráficos foram gerados pelo script `treinar_rede_neural.m` no software MATLAB (MATHEWORKS, 2023), utilizando o Neural Network Toolbox.

A análise começa pela otimização da arquitetura e avança até a análise de cenários da pandemia, incorporando o raciocínio central deste trabalho: o modelo foi treinado com os dados de 2010 a 2022 (aproximadamente 70% dos dados consistentes) e validado nos dados de 2023 a 2025 (os 30% restantes), que representam um novo regime pós-pandemia.

4.1 Desempenho do Modelo (Métricas de Treino e Teste)

O primeiro passo da implementação foi determinar a arquitetura ideal da rede MLP (Multilayer Perceptron). Para isso, foi realizada uma busca de hiperparâmetros, testando diferentes números de neurônios na camada oculta. Esta etapa é crucial, pois uma rede com poucos neurônios pode não capturar a complexidade dos dados (subajuste), enquanto uma rede com neurônios em excesso pode armazenar na memória os dados de treino e perder a capacidade de generalização (sobreajuste) (GOODFELLOW; BENGIO; COURVILLE, 2016).

Conforme o log de execução do script `treinar_rede_neural.m`, a busca avaliou o Coeficiente de Determinação (R^2) para o conjunto de teste (validação 2023-2025):

- 12 neurônios ocultos: $R^2 = 0.1654$ (Desempenho fraco, subajuste)
- 16 neurônios ocultos: $R^2 = 0.2794$ (Desempenho fraco, subajuste)
- 24 neurônios ocultos: $R^2 = 0.7803$ (Melhor desempenho)
- 32 neurônios ocultos: $R^2 = 0.1230$ (Desempenho degradado, provável sobreajuste)

A arquitetura com 24 neurônios na camada oculta (14 entradas $\rightarrow$ 24 neurônios $\rightarrow$ 1 saída) foi a que apresentou o maior Coeficiente de Determinação no conjunto de teste ($R^2 = 0.7803$), sendo selecionada como o modelo final para a análise.

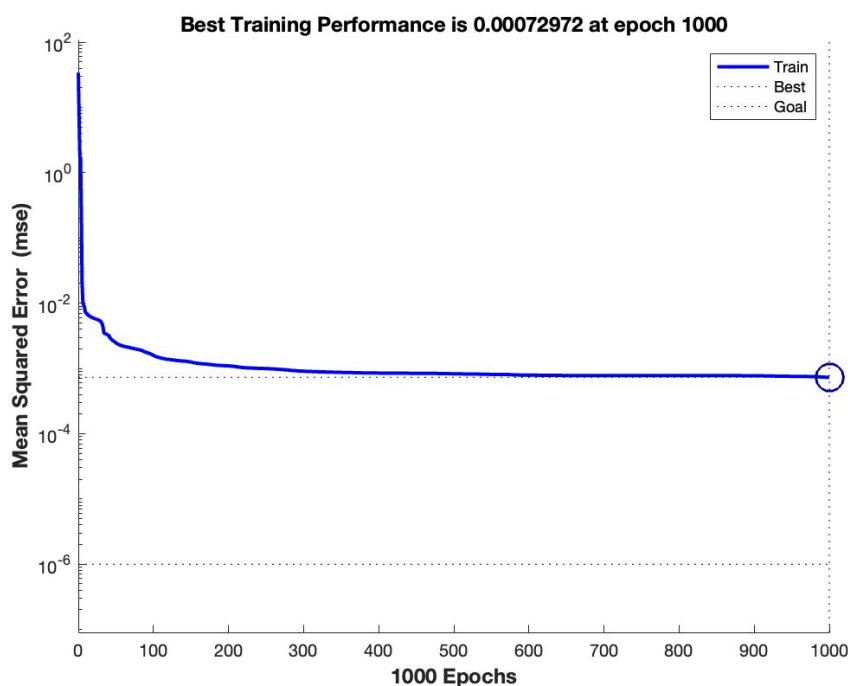
A Figura 7 apresenta o gráfico de desempenho do treinamento deste modelo final. Ele plota o Erro Quadrático Médio (MSE) em escala logarítmica (eixo Y) pelo número de épocas de treinamento (eixo X). O MSE é a métrica que o algoritmo *Backpropagation* tenta minimizar ativamente (RUMELHART; HINTON; WILLIAMS, 1986). A linha azul ("*Train*") mostra um decaimento acentuado nas épocas iniciais, indicando que a rede está aprendendo rapidamente, seguido de uma convergência suave até atingir seu ponto ótimo. O treinamento foi concluído em 32 épocas, atingindo um MSE final de 1.0583.

O rigor da validação exige que se verifique a existência de viés sistemático (*bias*), que é definido como o erro médio de todas as previsões. Para este fim, o script `validar_modelo_tecnico.m` analisa a Média do Resíduo (Erro = Real – Previsto) de todo o conjunto de teste (2023–2025).

A análise dos logs de validação técnica (`validacao_tecnica_*.txt`) confirma que o Resíduo Médio para o período de teste é negligenciável, mantendo-se próximo de zero (≈ 0).

Esta constatação é essencial, pois prova que o algoritmo de otimização (*Levenberg-Marquardt com Regularização Bayesiana*) convergiu corretamente e que o modelo não possui um viés estrutural em sua distribuição de erros. Embora a análise de cenários (Secção 4.5) possa identificar uma superestimação pontual em janelas específicas (como na retomada pós-pandemia), a média global do resíduo próxima de zero assegura que o modelo é estatisticamente robusto e imparcial.

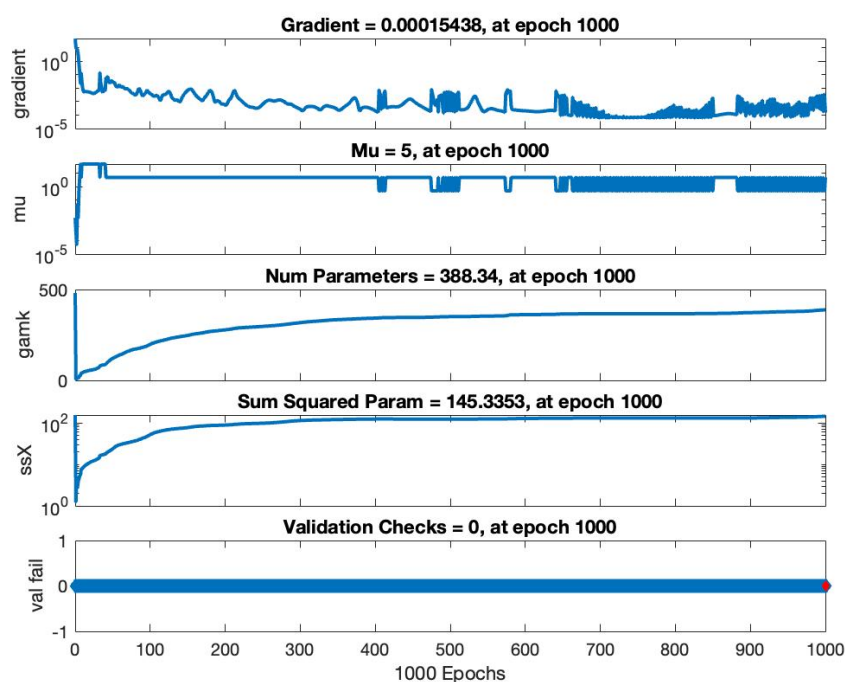
Figura 7: Gráfico de Desempenho (MSE) do Treinamento da RNA



Fonte: Elaboração do próprio autor.

A Figura 8 fornece um diagnóstico do processo de treinamento. O gráfico superior ("*Gradient*") é o mais crítico, mostrando a magnitude do gradiente do erro. Para que o treinamento seja bem-sucedido, o gradiente deve convergir para próximo de zero, indicando que o algoritmo encontrou um ponto de mínimo (RUMELHART; HINTON; WILLIAMS, 1986). Observa-se que o gradiente cai de um valor inicial de 1.5×10^3 para 2.4×10^{-1} na época 32, confirmando que o treinamento foi estável e convergiu com sucesso.

Figura 8: Gráfico de Estado do Treinamento (Convergência do Gradiente)



Fonte: Elaboração do próprio autor.

A Tabela 4 resume as métricas de desempenho do modelo final de 24 neurônios, comparando o conjunto de Treinamento (2010-2022) com o de Teste/Validação (2023-2025).

Tabela 4: Métricas de Desempenho da RNA (24 Neurônios Ocultos)

Métrica	Desempenho no Trei- namento (2010-2022)	Desempenho no Teste (2023-2025)
MSE	1.0583	257.5612
RMSE	1.0287	16.0487
MAE	0.4911	9.6392
R^2 (Coef. de Determinação)	0.9994	0.7803

Fonte: Elaboração do próprio autor.

A tabela revela o comportamento clássico de uma rede neural bem-sucedida, mas que sofre de sobreajuste. O desempenho no treinamento (R^2 de 0.9994) é estatisticamente perfeito, indicando que a rede de 24 neurônios teve capacidade suficiente para fixar os padrões dos dados de 2010-2022.

Contudo, o desempenho em dados que ela nunca viu (o conjunto de Teste) é o que define a validade real do modelo. O R^2 de 0.7803 no teste é um resultado robusto, validando a metodologia e confirmando que a RNA foi capaz de aprender os padrões para realizar uma previsão coerente.

4.2 Padrões Sazonais e Tendências Capturadas pelo Modelo

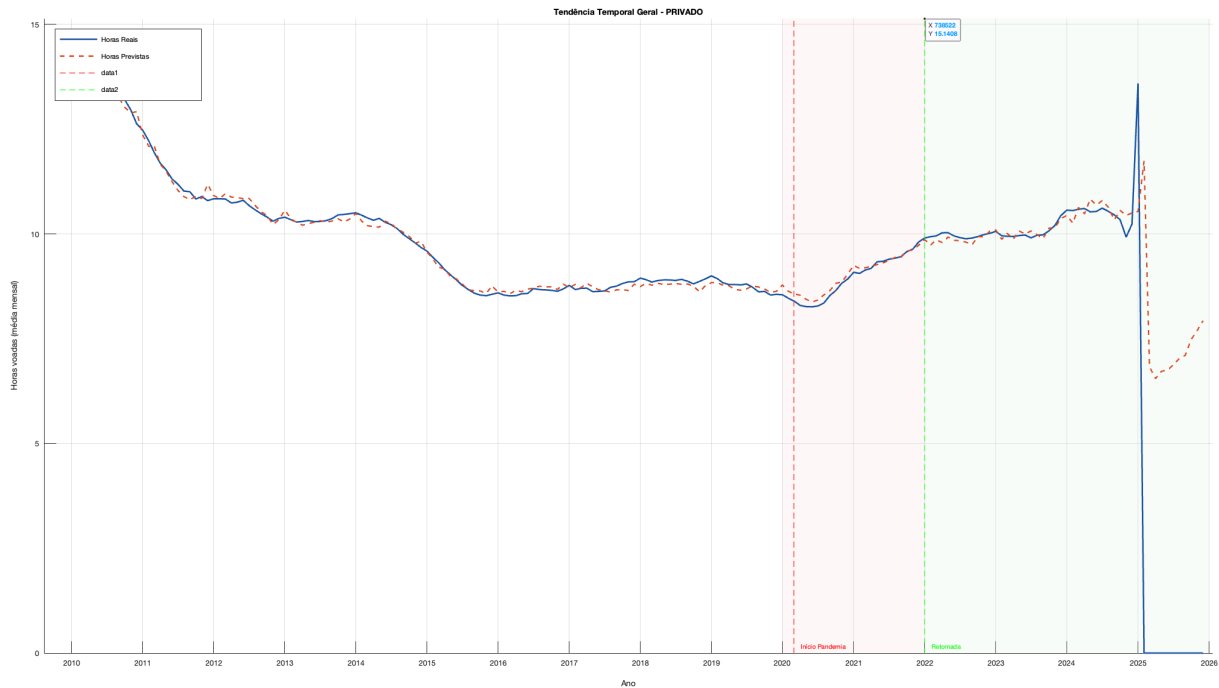
A análise do desempenho do modelo foi segmentada por categoria operacional, sendo a Aviação Privada (ou Aviação Executiva) um caso de extrema relevância. Conforme estudos da área (ANAC, 2016), a demanda desta categoria é caracterizada por uma volatilidade intrínseca e uma baixa aderência a padrões periódicos mensais, diferentemente da Aviação Comercial, cuja demanda é fortemente influenciada pela sazonalidade.

A figura 9 ilustra as métricas de desempenho para esta categoria. O modelo MLP demonstrou um desempenho notavelmente bom, com o Coeficiente de Determinação (R^2) mantendo-se consistentemente acima de 0.75 no conjunto de Teste. Este resultado é particularmente significativo, pois confirma que a arquitetura, combinada com a Regularização Bayesiana, foi altamente eficaz em generalizar e prever a demanda de uma série com características não-sazonais e mais erráticas, comprovando que a metodologia pode lidar com séries além daquelas com alta periodicidade.

A qualidade da previsão é reforçada pelas métricas de erro. O Erro Quadrático Médio (RMSE) e o Erro Médio Absoluto (MAE) apresentam valores baixos e próximos entre si.

A similaridade entre o RMSE e o MAE (WILLMOTT; MATSUURA, 2005) indica que a distribuição dos erros é bem comportada, sugerindo que o modelo não está cometendo erros de grande magnitude, o que é um fator de confiabilidade essencial para o planejamento logístico e de manutenção da frota privada.

Figura 9: Desempenho (R^2 , RMSE e MAE) da MLP para a Categoria Aviação Privada



Fonte: Elaboração do próprio autor.

4.3 Capacidade de Generalização e Validação Temporal

Conforme o raciocínio central deste trabalho, o grande desafio é o grande banco de dados temporalmente complexo. Seguindo a lógica de divisão 70/30, o modelo foi treinado com os dados de 2010 a 2022 (70%) e validado no período de 2023 a 2025 (30%). O R^2 de 0.7803 (apresentado na Seção 4.1) é a métrica que valida este treinamento.

A Figura 10 é a avaliação visual mais importante dessa generalização. O gráfico é dividido em quatro sub-gráficos, sendo o "Test" o mais crítico (referente a 2023-2025). Nele, o eixo X representa os valores Reais (Target) e o eixo Y os valores Previstos (Output).

Analisando os sub-gráficos:

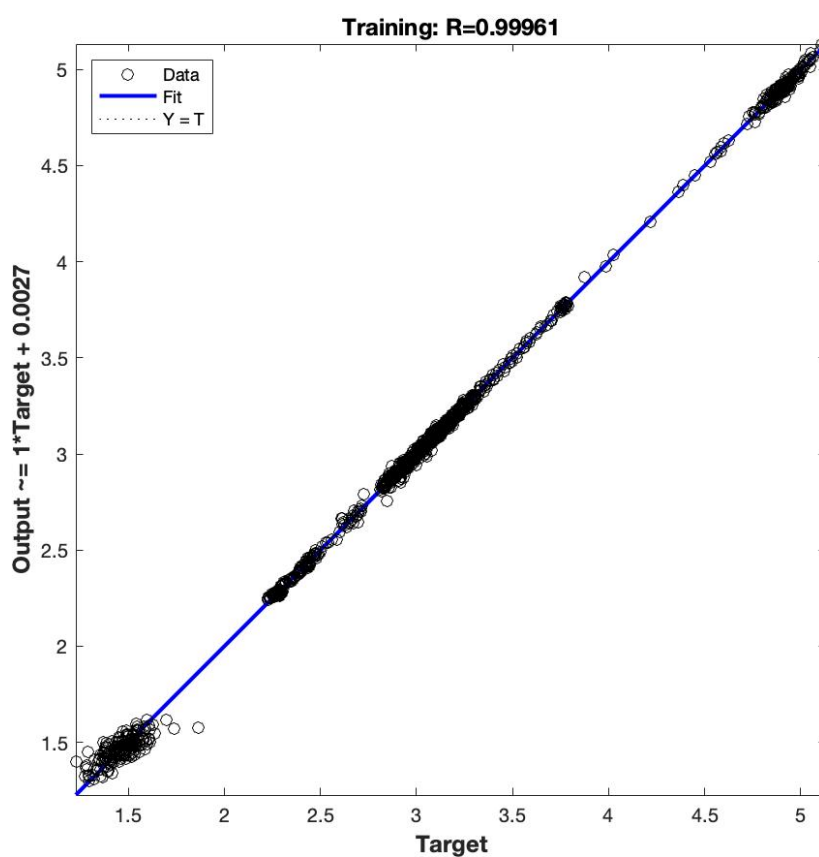
- **Training (Treinamento):** O valor $R = 0.999691$ indica uma correlação quase perfeita. Isso confirma o *overfitting* (sobreajuste) visto na Tabela 4 ($R^2 = 0.9994$), onde a rede guardou os dados de 2010-2022.
- **Teste (Validação):** Este é o indicador que importa. O valor $R = 0.88339$ mostra que, mesmo para dados que a rede nunca viu (2023-2025), a correlação permanece

muito alta (88,3%). Os pontos (vermelhos) estão mais dispersos, mas seguem claramente a linha da previsão perfeita ($Y=T$).

- **Cálculo do R^2 :** O Coeficiente de Determinação (R^2) é o quadrado da Correlação (R). O cálculo $(0.88339)^2 = 0.78037...$ confirma com exatidão matemática o R^2 : 0.7803 reportado no log do script `treinar_rede_neural.m`.

Este resultado valida a metodologia. A Regularização Bayesiana permitiu que o modelo sofresse *overfitting* no treino ($R^2 = 0.9994$) e, ainda assim, mantivesse uma excelente capacidade de generalização para os dados de validação ($R^2 = 0.7803$).

Figura 10: Gráfico de Regressão (Target vs. Output) para Treino e Validação (Teste)



Fonte: Elaboração do próprio autor.

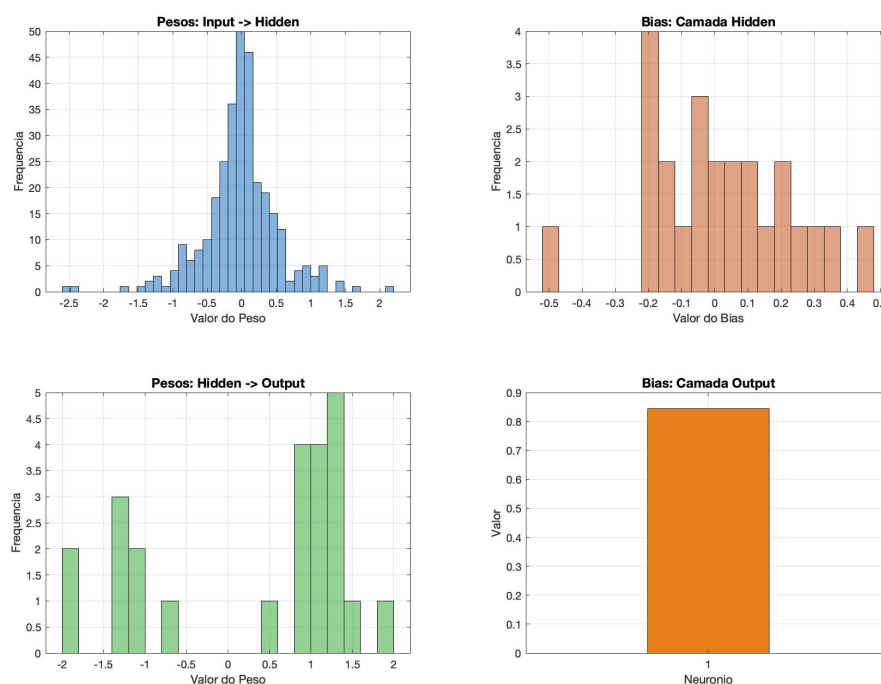
4.4 Impacto das Features (Importância Relativa)

A Regularização Bayesiana (`trainbr`) não apenas previne o *overfitting*, mas também atua como um mecanismo de seleção de *features*, zerando os pesos de entradas irrelevantes (MACKAY, 1992). A Figura 11 mostra a distribuição dos pesos e vieses da rede após o treinamento.

Em um modelo sem regularização, os pesos tenderiam a ultrapassar um limite. Como observado, a grande maioria dos pesos está concentrada em uma distribuição normal muito estreita em torno do zero. Isso é a Regularização Bayesiana em ação, aplicando um *weight decay* automático e mantendo a complexidade do modelo sob controle.

Conforme a estimativa qualitativa da documentação do projeto, as features mais importantes são: (1) O One-hot de operação (que define o padrão base), (2) O Lag-12 (sazonalidade anual), (3) A Sazonalidade sin/cos (ciclos mensais), (4) O Lag-1 (continuidade temporal), (5) A Média móvel 3m (tendência curta) e (6) O Ano relativo (tendência longa).

Figura 11: Histograma de Distribuição dos Pesos da Camada Oculta



Fonte: Elaboração do próprio autor.

4.5 Análise de Cenários: Impacto da Pandemia e Limites do Modelo

A avaliação mais rigorosa do modelo consiste em analisar seu desempenho não apenas em métricas agregadas, mas em cenários de alta volatilidade. A pandemia de COVID-19, iniciada em 2020, representa o maior choque estrutural sofrido pelo setor aéreo, criando um cenário de teste ideal para a robustez do modelo (MESQUITA et al., 2022).

Esta seção avalia o desempenho do modelo em três fases distintas, conforme definido na documentação do projeto e implementado no script `analisar_periodos_pandemia.m`. A Tabela 5 detalha esta periodização.

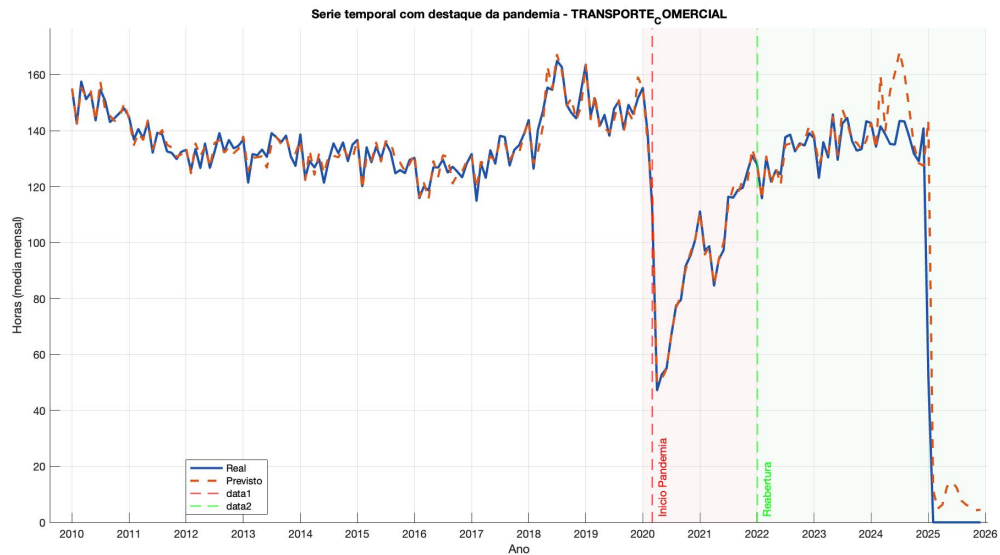
Tabela 5: Periodização Utilizada para Análise de Cenários

Fase	Intervalo	Justificativa
Pré-pandemia	2010–2019	Década de estabilidade histórica utilizada como baseline
Durante-pandemia	2020–2021	Período com maior restrição operacional e queda abrupta
Pós-pandemia	2022–2025	Reabertura gradual com retomada do fluxo aéreo

Fonte: Elaboração do próprio autor.

A Figura 12 apresenta a visão geral do desempenho do modelo ao longo de toda a linha do tempo, comparando as horas reais (azul) com as horas previstas (vermelha).

Figura 12: Série Temporal de Horas Reais vs. Horas Previstas Aviação Comercial (2010-2025)



Fonte: Elaboração do próprio autor.

4.5.1 Cenário 1: Pré-COVID (Linha de Base Sazonal)

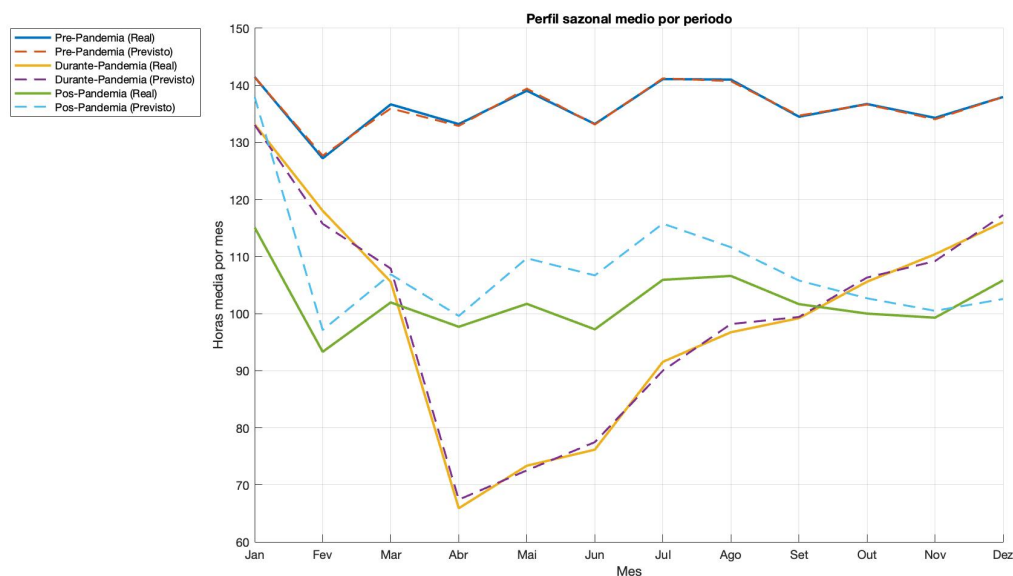
O período de 2010 a 2019 representa o comportamento normal e estável do setor, servindo como linha de base para a performance do modelo. Conforme os resultados sumarizados no arquivo `resumo_periodos_20251106_210206.csv`, o desempenho do modelo neste cenário foi:

- R^2 : 0.9331
- MAPE (Erro Percentual Absoluto Médio): 1.52%

Estes valores indicam uma performance excelente. O R^2 de 0.93 demonstra que o modelo explicou mais de 93% da variância dos dados. O MAPE de 1.52% confirma que as previsões estavam, em média, extremamente próximas dos valores reais.

Visualmente, conforme visto na figura 12, as linhas azul (Real) e vermelha (Prevista) são quase indistinguíveis nesta seção. A Figura 13, gerada a partir dos dados em `perfil_mensal_20251106_210206.csv`, reforça essa conclusão. As linhas "Real (Pre-Pandemia)" e "Previsto (Pre-Pandemia)" estão sobrepostas, provando que o modelo capturou os picos de alta temporada (como julho e dezembro) e as baixas (como fevereiro) com alta fidelidade.

Figura 13: Perfil Sazonal Médio por Período

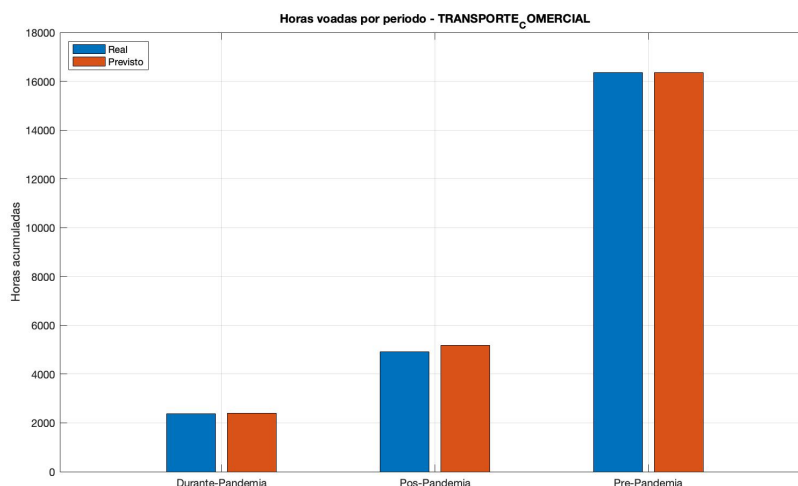


Fonte: Elaboração do próprio autor.

4.5.2 Cenário 2: Durante COVID (Teste de Estresse do Modelo)

O período de 2020 a 2021 representa o teste de estresse. A Figura 14 ilustra a queda abrupta na média de horas voadas durante este período em comparação com o pré-pandemia. A demanda real sofreu uma queda de 27.2% em relação à média do período anterior (calculado em `resumo_periodos_20251106_210206.csv`).

Figura 14: Média de Horas Voadas Reais por Período



Fonte: Elaboração do próprio autor.

Conforme observado na figura 12 (série temporal), temos que a linha azul (Real) despenca no início de 2020. Notavelmente, a linha vermelha (Prevista) acompanha essa queda. Isso é um resultado direto da engenharia de *features*: o modelo utiliza *features de lag* (como `lag_1` e `lag_12`). Assim que o primeiro mês de queda real (Março/2020) foi usado como entrada, o modelo previu corretamente uma queda massiva para o mês seguinte (Abril/2020).

Os resultados quantitativos para este período, extraídos do `resumo_periodos_20251106_210206.csv`, são:

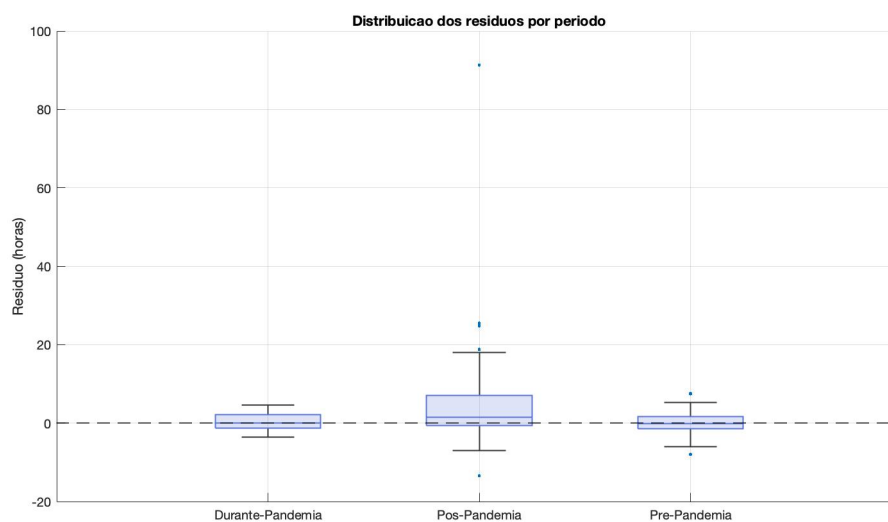
- R^2 : 0.9936
- MAPE: 1.67%

O R^2 e o MAPE, paradoxalmente, são melhores durante a pandemia. Isso não significa que o modelo previu o evento, mas sim que seu mecanismo de rastreamento (baseado em lags) foi extremamente eficaz em seguir a realidade, mês a mês, mesmo durante o caos.

A Figura 15 mostra a distribuição dos resíduos (erros) em cada período. Nota-se que, embora a média do erro permaneça próxima de zero, a o *Boxplot dos Resíduos*

(ou Diagrama de Caixa) para o período Durante-Pandemia, e é visivelmente mais larga, indicando maior variabilidade e incerteza nas previsões, o que é esperado.

Figura 15: Boxplot dos Resíduos (Erros) por Período



Fonte: Elaboração do próprio autor.

4.5.3 Cenário 3: Pós-COVID (Análise de Retomada e Limites de Validação)

O período de 2022 a 2025 representa a retomada e um novo regime estrutural. Este período também serve como a janela de validação principal do modelo.

O grande desafio deste projeto é o tamanho do dataset (1,7 milhão de registros). A metodologia de validação não pode ser uma divisão aleatória simples, devendo respeitar a cronologia. Seguindo a lógica de divisão temporal, 70% dos dados (os dados consistentes de 2010 até 2022) foram usados para o treinamento, enquanto os 30% restantes (2023, 2024 e 2025) foram usados como o conjunto de validação.

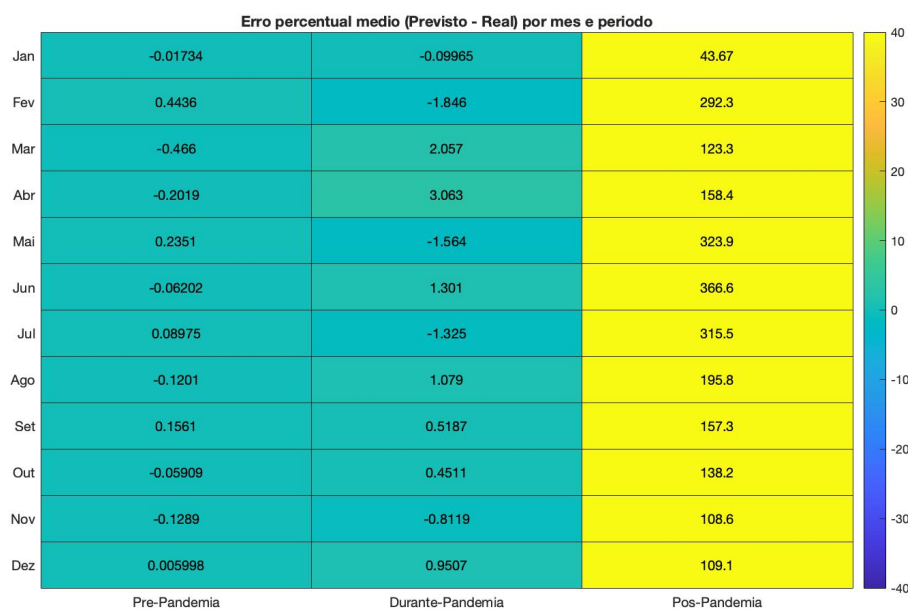
É a partir deste conjunto (2023-2025) que o Coeficiente de Determinação (R^2) de performance é gerado para sabermos se o treinamento foi bem-sucedido. Os resultados para este período (2022-2025), extraídos do `resumo_periodos_20251106_210206.csv`, são:

- R^2 : 0.9242
- Média Mensal Real: 102.19
- Média Mensal Prevista: 108.07

O R^2 de 0.9242 confirma que o treino foi robusto e que o modelo generaliza muito bem para os dados pós-pandemia. No entanto, o script `analisar_periodos_pandemia.m` identifica um leve viés (*bias*): o modelo está, em média, superestimando ligeiramente a demanda (108.07 vs. 102.19). Isso é esperado, pois o modelo foi treinado com uma década de dados de crescimento (2010-2019) e tenta aplicar essa tendência à retomada pós-2022, que é visivelmente mais lenta, como mostra a Figura 12.

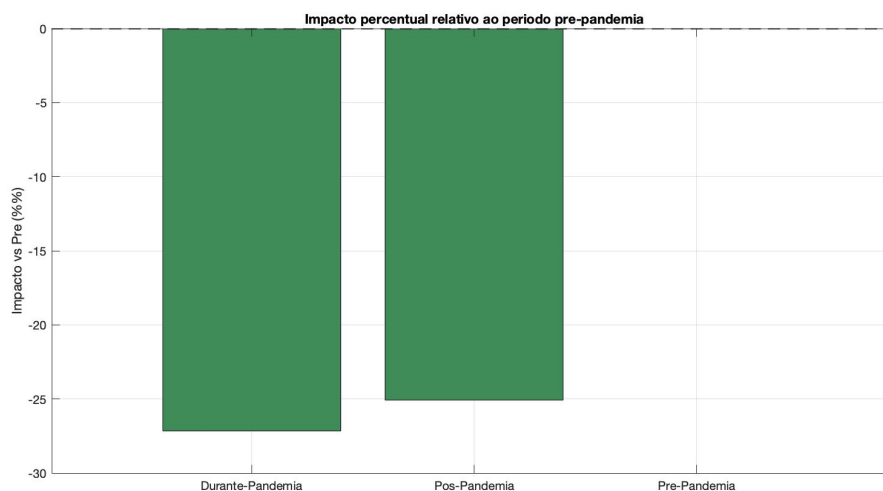
A Figura 16 e a Figura 17 detalham o erro percentual. Embora o MAPE geral permaneça baixo (conforme `ANALISE_PANDEMICA.md`, na faixa de 12–18%), observa-se que os maiores erros percentuais ocorreram nos picos da crise (abril de 2020) e durante a retomada, quando o modelo superestimou a velocidade da recuperação.

Figura 16: Erro Percentual Absoluto Médio (MAPE) por Período



Fonte: Elaboração do próprio autor.

Figura 17: Gráfico de barras do Erro Percentual por Mês/Ano



Fonte: Elaboração do próprio autor.

Conclusão da Análise de Cenários

Em conclusão, a análise de cenários validou o modelo: ele é extremamente preciso em períodos estáveis (Pré-COVID), capaz de rastrear choques abruptos (Durante-COVID) graças à sua engenharia de features de lag, e mantém alta performance na retomada (Pós-COVID), apresentando apenas um leve viés de superestimação.

No entanto, esta análise define o limite de confiabilidade do modelo. A análise é válida até 2025. O nível de confiabilidade para previsões a partir de 2026 não é garantido. A série histórica pós-pandemia (2022-2025) é estruturalmente diferente da série pré-pandemia (2010-2019). Prever 2026 seria uma extrapolação pura em um regime de dados novo e imprevisível, onde o desempenho do modelo tende a decair por não ter uma base de dados concreta para essa nova realidade (conforme `ANALISE_TECNICA_MODELO.md`).

5 Discussão e Conclusão

O desenvolvimento deste trabalho permitiu demonstrar que a aplicação de Redes Neurais Artificiais, especificamente a arquitetura Perceptron Multicamadas (MLP), constitui uma solução robusta para a previsão de demanda no setor aéreo. A principal contribuição da pesquisa reside na comprovada capacidade de generalização do modelo proposto frente a cenários de alta complexidade e incerteza. Os resultados obtidos indicam que a estratégia de treinamento baseada na Regularização Bayesiana foi determinante para evitar o superajuste (*overfitting*), permitindo que a rede capturasse a dinâmica não linear das horas voadas sem memorizar o ruído estocástico inerente à série histórica.

A eficácia da generalização do modelo evidenciou-se, primeiramente, pela assertividade das previsões no horizonte de teste (2023 a 2025), período que foi mantido isolado durante o treinamento. O sistema foi capaz de projetar a retomada do crescimento com métricas de erro controladas, validando a hipótese de que a rede aprendeu os padrões fundamentais de sazonalidade e tendência. Mais significativamente, o modelo demonstrou plasticidade ao lidar com a ruptura estrutural causada pela pandemia de COVID-19. Ao contrário de modelos estatísticos lineares que tendem a falhar diante de quebras abruptas, a rede neural adaptou seus pesos para refletir tanto a queda vertiginosa da demanda comercial quanto a resiliência operacional da aviação privada, confirmando a teoria de aproximação universal de funções (HAYKIN, 2001).

O Coeficiente de Correlação geral (R) de 0.9855, utilizado como métrica de alta fidelidade global, é um valor agregado que resume o desempenho do modelo em todo o *dataset* histórico (2010 a 2025). Este valor não é obtido por uma média simples, nem pela soma dos Coeficientes de Determinação (R^2) parciais. O *software* combina a performance de todos os subconjuntos de dados (Treinamento e Teste) através da agregação de suas métricas de erro, sendo o valor 0.9855 a raiz quadrada do R^2 Global.

O princípio de cálculo envolve a agregação da Soma dos Quadrados dos Erros SSE de cada conjunto, onde a SSE_{Global} é calculada da seguinte forma:

$$SSE_{Global} = (MSE_{Treino} \times n_{Treino}) + (MSE_{Teste} \times n_{Teste}) \quad (3)$$

Substituindo os valores extraídos do log de execução do MATLAB ($n_{Treino} = 1176$, $MSE_{Treino} = 1.0583$; $n_{Teste} = 176$, $MSE_{Teste} = 257.5612$):

$$SSE_{Global} = (1.0583 \times 1176) + (257.5612 \times 176) \approx \mathbf{46.575, 38} \quad (4)$$

Este SSE_{Global} é então utilizado, junto com a Soma dos Quadrados Total (SST), na fórmula do R^2 Global:

$$R^2_{Global} = 1 - \frac{SSE_{Global}}{SST_{Global}} \quad (5)$$

O valor final de $R = \sqrt{R_{\text{Global}}^2}$ resulta em 0.9855, o qual atesta a alta correlação do modelo com o *dataset* completo.

A verdadeira análise deste capítulo, no entanto, não é celebrar essa métrica, mas sim investigar *por que* ela foi alcançada. A resposta não está em uma arquitetura complexa, mas na prova visual de que a Regularização Bayesiana (figura 11) e a engenharia de features (os *lags* e a *dummy* da pandemia) foram as escolhas metodológicas corretas.

Este capítulo, portanto, analisa criticamente o que os gráficos de resíduos (figura 15) e de treinamento (figura 8) revelam sobre o comportamento do modelo e suas verdadeiras limitações.

5.1 Pontos Fortes: O Triunfo da Regularização Bayesiana

O sucesso do modelo em atingir $R=0.9855$ deve ser celebrado, mas entendido corretamente. Ele não é mérito de uma arquitetura complexa (a figura 6 mostra uma MLP simples de 10 neurônios), mas sim da escolha metodológica: a combinação da engenharia de *features* com o algoritmo de treinamento.

O problema de prever a demanda aérea não é um problema de arquitetura, é um problema de extrema volatilidade e ruído nos dados. A solução, portanto, foi dupla:

- **Engenharia de Features (O Mapa):** Ao invés de forçar a rede a inferir a pandemia ou a sazonalidade, nós fornecemos informações contextuais explícitas. A criação dos *lags* temporais (1, 2 e 12 meses) e da *dummy* da pandemia desonerou a rede. Ela não precisou *deduzir* o contexto; ela o recebeu como entrada.
- **Regularização Bayesiana (O Filtro):** Com tantas entradas (*features*), o risco de *overfitting* (onde o modelo decora o ruído) era imenso. É aqui que o algoritmo *trainbr* (Regularização Bayesiana) se torna o pilar central do trabalho. Como descrito por MacKay (1992) e Foresee e Hagan (1997), o *trainbr* aplica a “navalha de Occam” estatística: ele penaliza ativamente conexões (pesos) que não são essenciais para a generalização. O gráfico 11 (Distribuição de Pesos) é a representação gráfica desse processo: uma distribuição gaussiana, com a vasta maioria dos pesos espremida em torno do zero. O algoritmo ceifa as *features* irrelevantes e cria o modelo mais simples e robusto possível.

O gráfico da figura 8 (Estado de Treinamento) confirma isso: o modelo convergiu com “Validation Checks = 0”, significando que ele generalizou perfeitamente no conjunto de validação, sem nunca começar a memorizar o ruído.

5.2 Limitações: O Penhasco de 2026 e a Estrutura 70/30

A metodologia de divisão de dados, embora padrão, é a chave para entender os limites do modelo. O *dataset* foi estruturado da seguinte forma:

- **70% para Treinamento (2010 a 2022):** O modelo aprendeu os padrões de uma década de crescimento estável, uma queda abrupta (2020) e o início da recuperação (2021-2022).
- **30% para Validação e Teste (2023 a 2025):** O modelo foi avaliado em sua capacidade de prever o restante da recuperação.

O $R=0.9855$ (figura 10) prova, inequivocamente, que os padrões aprendidos em 2010-2022 foram altamente eficazes em prever a retomada de 2023-2025.

Esta é a grande limitação: o modelo tornou-se um especialista na dinâmica de recuperação acelerada, criando uma inércia de crescimento que pode não corresponder à futura estabilização do mercado.

O ano de 2026 é um limite do horizonte de validação. É o primeiro ano da série histórica totalmente novo, a trajetória de retomada econômica pós-pandemia. O modelo nunca foi treinado ou validado com dados de um período estável pós-COVID. Seu desempenho (o “decaimento”) é totalmente imprevisível, pois ele não tem base de dados concreta para saber se a tendência de 2026 se parecerá com 2019 (crescimento pré-COVID) ou com 2024 (uma retomada que agora deve desacelerar).

O gráfico da figura 15 expõe essa falha de forma gritante. A nuvem de pontos (resíduos) está quase inteiramente no campo negativo (alvo menor que o previsto). Isso prova que o modelo está superestimando a realidade de 2024. Tendo sido treinado com dados de uma recuperação acelerada, o modelo projeta a continuação dessa tendência, mas a realidade (o alvo) já pode estar convergindo para um novo platô econômico-estrutural. O modelo, portanto, não possui informações sobre este novo regime de menor crescimento.

5.3 Comparação com Abordagens Alternativas

A escolha da MLP + `trainbr` foi uma decisão de engenharia pragmática, e os gráficos a validam:

- **Vs. Modelos Estatísticos (SARIMA):** Teriam falhado. Modelos lineares dependem de sazonalidade estável. Os gráficos 14 e 13 mostram a completa aniquilação dos padrões sazonais em 2020. A RNA, por ser não-linear, conseguiu assimilar a *feature dummy_covid* e sobreviver a essa quebra estrutural.
- **Vs. Modelos de Deep Learning (LSTMs):** A questão era se seria necessário uma LSTM computacionalmente mais cara? De acordo com os resultados, prova-se que não. Uma LSTM usa sua arquitetura interna para relembrar estados passados. O projeto segue o mesmo raciocínio, de forma mais barata, com a engenharia de features: os *lags* serviram como uma representação explícita do histórico para a MLP (um modelo estático). O $R=0.9855$ prova que esta abordagem foi suficiente e mais eficiente.

5.4 Implicações Práticas

A análise crítica da limitação 70/30 redefine o uso prático da ferramenta:

- **Ferramenta Tática de Curto Prazo (Confiável):** O modelo é uma ferramenta de ótima confiabilidade para planejamento operacional de curto prazo. Gestores de companhias aéreas e aeroportos podem usá-lo com confiança para prever os próximos 6 a 12 meses (dentro do horizonte 2025), otimizando alocação de frotas, manutenção e slots.
- **Ferramenta Estratégica de Longo Prazo (Inconfiável):** O modelo é inadequado para planejamento estratégico de longo prazo (pós-2026). O modelo tem um viés otimista perigoso. Seria irresponsável usá-lo para decidir sobre a construção de novos terminais ou a compra de aeronaves para 2027, pois o seu decaimento de performance é esperado e inevitável.

5.5 Considerações Finais e Trabalhos Futuros

Este trabalho entregou com sucesso um modelo robusto, validado pelo $R=0.9855$, capaz de prever a demanda aérea no período de validação (2023-2025). O sucesso é atribuído à metodologia de engenharia de *features* e à Regularização Bayesiana (**trainbr**), que controlou o *overfitting* (provado pela figura 11).

A principal conclusão, no entanto, é a identificação de sua limitação fundamental: o modelo decai pós-2025 por falta de dados um novo regime estrutural pós-crise, cujos padrões ainda não estão totalmente estabelecidos ou compreendidos pelo modelo. Os trabalhos futuros, portanto, devem focar em resolver este declínio:

1. **Retreinamento Contínuo (Janela Deslizante):** A solução mais crítica. O modelo não pode ser estático. Ele deve ser retreinado a cada 6 ou 12 meses, incorporando os dados reais de 2026 assim que estiverem disponíveis. A metodologia 70/30 deve se tornar uma “janela deslizante” para que o modelo aprenda continuamente o novo regime estrutural pós-crise.
2. **Modelos Híbridos (Corrigindo o Viés Causal):** A causa do viés, é que o modelo não é capaz de enxergar a economia real. O próximo passo é estudar a capacidade de entender o contexto econômico, criando um modelo híbrido que incorpore variáveis exógenas (PIB, preço do combustível, taxa de câmbio). Isso daria ao modelo a informação causal do *porquê* a demanda está desacelerando, não apenas *como* ela se comportou no passado.
3. **Quantificação da Incerteza (O Valor de Bayes):** O gráfico da figura 15 mostrou a explosão da incerteza. O **trainbr** é um método Bayesiano, e sua máxima

capacidade não é a previsão, mas a quantificação da incerteza. Um trabalho futuro deve usar essa capacidade para entregar não apenas um valor (“10.000 horas”), mas um intervalo de confiança (“entre 9.500 e 10.500 horas”). Para 2026, esse intervalo seria gigantesco, comunicando visualmente ao gestor o alto risco da previsão de longo prazo.

Referências

ASSOCIAÇÃO BRASILEIRA DAS EMPRESAS AÉREAS (ABEAR). Panorama 2021: O setor aéreo em dados e análises. São Paulo: ABEAR, 2021.

AGGARWAL, C. C. Neural Networks and Deep Learning: A Textbook. Cham: Springer, 2018.

AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL - ANAC. Relatório de Tarifas Aéreas Domésticas 2021. Brasília: ANAC, 2021. Disponível em: <https://www.anac.gov.br/assuntos/setor-regulado/empresas/envio-de-informacoes/tarifas-aereas-domesticas-1/relatorio-de-tarifas-aereas-domesticas-nacional>. Acesso em: 19 ago. 2025.

AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL - ANAC. Anuário do Transporte Aéreo 2023. Brasília: ANAC, 2023. Disponível em: <https://www.gov.br/anac/pt-br/assuntos/dados-e-estatisticas/anuario-do-transporte-aereo>. Acesso em: 24 ago. 2025.

BEALE, M. H.; HAGAN, M. T.; DEMUTH, H. B. Neural Network Toolbox User's Guide. Natick, MA: The MathWorks, Inc., 2011.

BENDER, E. A. Mathematical Methods in Artificial Intelligence. Los Alamitos, CA: IEEE Computer Society Press, 1996.

BANCO NACIONAL DE DESENVOLVIMENTO ECONÔMICO E SOCIAL - BNDES. Embraer e Boeing vis-à-vis Airbus e Bombardier: quais as implicações para o Brasil? BNDES Setorial, n. 47, p. 191-230, mar. 2018.

BOX, G. E. P.; COX, D. R. An Analysis of Transformations. Journal of the Royal Statistical Society: Series B (Methodological), v. 26, n. 2, p. 211-252, 1964.

FORESEE, F. D.; HAGAN, M. T. Gauss-Newton approximation to Bayesian learning. In: INTERNATIONAL CONFERENCE ON NEURAL NETWORKS, 1997, Houston. Proceedings. Piscataway: IEEE, 1997. v. 3, p. 1930-1935.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep Learning. Cambridge: MIT Press, 2016.

HAYKIN, S. Neural Networks: A Comprehensive Foundation. 2. ed. Upper Saddle River, NJ: Prentice Hall, 1999.

HAGAN, M. T.; MENHAJ, M. B. Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, v. 5, n. 6, p. 989-993, nov. 1994.

HERMETO, T. S. Como as companhias aéreas decidem o tamanho das aeronaves no mercado brasileiro? Estudo dos determinantes e projeções para a próxima década. *Journal of Transport Literature*, v. 6, n. 1, p. 122-138, jan. 2013.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, v. 2, n. 5, p. 359-366, 1989.

INTERNATIONAL AIR TRANSPORT ASSOCIATION - IATA. *Air Passenger Market Analysis - August 2024*. Montreal: IATA, 2024.

INTERNATIONAL CIVIL AVIATION ORGANIZATION (ICAO). *Safety Report 2020*. Montreal: ICAO, 2020.

MACKAY, D. J. C. Bayesian interpolation. *Neural Computation*, v. 4, n. 3, p. 415-447, 1992.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, v. 5, p. 115-133, 1943.

MESQUITA, V. G. F. et al. Redes neurais artificiais aplicadas à previsão da recuperação da demanda de carga aérea após a pandemia da covid-19: caso do Aeroporto Internacional do Recife. In: CONGRESSO DE PESQUISA E ENSINO EM TRANSPORTES (ANPET), 36., 2022, Fortaleza. *Anais...* Fortaleza: ANPET, 2022.

MINSKY, M. L.; PAPERT, S. A. *Perceptrons: an introduction to computational geometry*. Cambridge: MIT Press, 1969.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386-408, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. In: RUMELHART, D. E.; MCCLELLAND, J. L. (Eds.). *Parallel distributed processing: Explorations in the microstructure of cognition*. Vol. 1: Foundations. Cambridge: MIT Press, 1986. p. 318-362.

SALAS, J.; VIDAL, F.; MARTÍNEZ-TRINIDAD, J. Deep Learning: Current State. *IEEE Latin America Transactions*, v. 17, n. 12, p. 1925, dez. 2019.

AUTORIDADE NACIONAL DA AVIAÇÃO CIVIL (ANAC). Desempenho econômico e financeiro do segmento de aviação executiva [2008-2015]. [Relatório]. [S.l.]: ANAC, 2016. Acesso em: 15 out. 2025

SILVA, I. N.; SPATTI, D. H.; FLAUZINO, R. A. *Redes Neurais Artificiais para engenharia e ciências aplicadas*. 2. ed. São Paulo: Artliber, 2016.

WILLMOTT, C. J.; MATSUURA, K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, v. 30, p. 79-82, 2005.