Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Arcabouço de Testes para Modelos de Aprendizado de Máquina Supervisionado

Rafael de Oliveira Magalhães

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisores: Prof. Dr. Flávio Soares Corrêa da Silva
Prof.ª Dr.ª Ana Cristina Vieira de Melo

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro da FAPESP – processo n° 2023/05483-1

São Paulo
2024

Resumo

Rafael de Oliveira Magalhães. **Arcabouço de Testes para Modelos de Aprendizado de Máquina Supervisionado**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2024.

Este trabalho apresenta métodos de testes de modelos de *Machine Learning*, com foco na geração e uso de dados sintéticos para essa finalidade. Selecionou-se o problema de previsão de tráfego como meio de estudo, o qual é interessante por sua complexidade e relevância como problema real. Dado esse contexto, procurou-se conjuntos de dados abertos para uso nos testes e para comparação com os dados sintéticos, e foram selecionados três modelos de *Machine Learning* com características distintas: um modelo de regressão linear, um modelo de rede neural totalmente conectada e um modelo de rede neural para grafos (*GNN*). Com base nesse problema e nos modelos de *Machine Learning*, formulou-se um arcabouço de testes generalizável para outros problemas, modelos e dados. Esse arcabouço utiliza tanto os dados reais, quando aplicáveis, quanto os dados sintéticos para a representação de cenários não encontrados nos dados reais. A ideia desse arcabouço é avaliar de forma ampla qualquer fator que possa influenciar os resultados, explicitando diferenças e similaridades comportamentais entre os modelos e por meio de comparações com valores de referência. Por fim, a eficácia do arcabouço é posta a prova aplicando-o ao problema, dados e modelos citados e analisando as informações inferíveis a partir dos resultados dos testes.

Palavras-chave: Aprendizado de Máquina. Testes. Dados Sintéticos. Previsão de Tráfego.

Abstract

Rafael de Oliveira Magalhães. **Testing Framework for Supervised Machine Learning Models**. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2024.

This work presents methods for testing Machine Learning models, focusing on the generation and use of synthetic data for this purpose. The traffic forecasting problem was selected as a means of study, which is interesting due to its complexity and relevance as a real-world problem. Given this context, open datasets were sought for use in testing and for comparisons with synthetic data, and three Machine Learning models with distincts features were selected: a linear regression model, a fully connected neural network model and a graph neural network model (*GNN*). Based on this problem and the Machine Learning models, a testing framework was formulated, which is generalizable to other problems, models and data. It uses both real data if applicable, and synthetic data to represent scenarios not found in real data. The idea of this framework is broadly evaluate any factors that may influence the results, which is is achieved by highlighting behavioral differences and similarities between the models and through comparisons with reference values. Finally, the effectiveness of the framework is tested by applying it to the mentioned problem, models, and data, and analyzing the the information that can be inferred from the test results.

Keywords: Machine Learning. Tests. Synthetic Data. Traffic Forecasting.

Lista de abreviaturas

FAPESP	Fundação de Amparo à Pesquisa do Estado de São Paulo
GNN	Rede neural para grafos (*Graph Neural Network*)
CNN	Rede neural convolucional (*Convolutional Neural Network*)
RMSE	Raiz do erro quadrático médio (*Root Mean Square Error*)
NRMSE	RMSE normalizado (*Normalized Root Mean Square Error*)
MAE	Erro absoluto médio (*Mean Absolute Error*)
MAPE	Erro absoluto percentual médio (*Mean Absolute Percentage Error*)
STGC	Convolução espacial-temporal para grafos (*Spatial–Temporal Graph Convolution*)
STGI	Estrutura *inception* espacial-temporal para grafos (*Spatial–temporal Graph Inception*)
ReLU	Unidade linear retificada (*Rectified Linear Unit*)
CALTRANS	Departamento de Transporte da Califórnia (*California Transportation Agencies*)
PEMS	Sistema de Medição de Performance (*Performance Measurement System*)
EUA	Estados Unidos da América
NYC	Cidade de Nova Iorque (*New York City*)

Lista de símbolos

ϕ	Distribuição estacionária
$\mathcal{G}$	Notação de grafo
P	Matriz de transição associada ao grafo
I_M	Matriz identidade de tamanho $M \times M$
$\mathfrak{L}$	Laplaciano do dígrafo
Φ	Matriz diagonal de ϕ
Λ	Matriz diagonal de autovalores obtida da diagonalização do laplaciano
λ_i	Autovalor não negativo do laplaciano (espectro do laplaciano)
$*_{\mathcal{G}}$	Operação de convolução para grafos
$T_k(x)$	Polinômio de Chebyshev de ordem k

Lista de figuras

Lista de tabelas

Lista de programas

Sumário

[1] Referência do capítulo: Jordon *et al.*, 2022

Introdução

Motivação e Objetivo

Atualmente, o emprego de métodos de aprendizado de máquina supervisionado na resolução de problemas tem apresentado melhores resultados em comparação com abordagens tradicionais de inteligência artificial ou estatísticas. Entretanto, este tipo de solução não pode ser amplamente utilizado, sobretudo em aplicações críticas ou que exijam um alto grau de confiabilidade, devido à característica de se comportar como uma "caixa preta", ou seja, o funcionamento desses modelos não é transparente, o que impede a compreensão exata e não permite garantir a confiabilidade dos resultados. Além disso, a falta desse conhecimento afeta o desenvolvimento de *software*, pois a escolha do modelo mais adequado não é guiado por um processo formal, mas apenas por experimentação. Esse contexto torna o desenvolvimento de *softwares* baseados em aprendizado de máquina muito oneroso em termos de tempo, recursos financeiros e computacionais, tendo em vista que os modelos de alta complexidade necessitam de grande capacidade de processamento, sendo treinados em grandes servidores ou supercomputadores, e que, mesmo neste cenário, um único treinamento pode demorar dias ou semanas. Portanto, é evidente que a existência de uma metodologia que auxilie o desenvolvimento é de grande interesse tanto para as empresas quanto para os programadores.

Várias abordagens diferentes têm sido usadas para tentar clarificar o funcionamento dos métodos de aprendizado de máquina. Em especial, uma abordagem que tem se mostrado muito promissora é o *Deep Learning* Geométrico (Bronstein *et al.*, 2021), o qual busca unificar o estudo e compreensão de diferentes modelos por meio das estruturas matemáticas e suas correspondentes propriedades. Outras pesquisas com o mesmo objetivo também têm seguido uma perspectiva semelhante utilizando uma base matemática profunda para análise das transformações realizadas pelos métodos (Razin *et al.*, 2023, Alexander *et al.*, 2024). Essas abordagens, embora formalmente corretas, são extensamente teóricas e ainda não generalizáveis ou fáceis de serem aplicadas para diversos modelos. Assim, este trabalho terá esse mesmo objetivo, porém seguirá uma abordagem de estudo mais prática e concreta por meio do desenvolvimento de um arcabouço de testes.

O Estudo de Caso Utilizado

Com este propósito e a fim de construir e validar a eficácia do arcabouço, abordou-se o problema de previsão de tráfego, o qual é um problema de relevância prática e interessante, pois representa um fenômeno do mundo real que é sensível a mudanças de padrões sociais

(como a mudança do trabalho presencial para o remoto) e a fatores ambientais e temporais (como a ocorrência de enchentes ou de feriados). Além disso, sendo um problema do mundo real, é comum que dados disponíveis apresentem irregularidades, dados faltantes e erros, atributos que formam um fator comum com outros problemas.

Selecionado o problema, a referência principal utilizada para estudo foi o artigo Zhang *et al.*, 2019, o qual, além de explicar as especificidades do problema, implementa um modelo de rede neural para grafos (GNN) para solucioná-lo, detalhando também as informações dos dados necessárias, como os dados de treinamento devem ser gerados, e os hiperparâmetros e métricas utilizados.

Metodologia

Os testes formulados possuem como objetivo explicitar comportamentos de um modelo de interesse. Esse processo é realizado verificando algum atributo por meio do treinamento do modelo com um conjunto de dados, e os resultados são interpretados com base nos valores do erro e métricas utilizados. No caso deste trabalho, o modelo de interesse testado foi a GNN do artigo citado.

Além do modelo de interesse, considerou-se benéfico selecionar outros modelos para comparações, a fim de estabelecer uma base mais robusta para interpretação dos resultados. Desta forma, comparou-se a GNN com um modelo de regressão linear e com um modelo de rede neural totalmente conectada.

Ademais, com base nos padrões estabelecidos pelo artigo, definiu-se um contexto fixo para execução dos testes, com o intuito de evitar que os resultados sejam influenciados por outros fatores além do atributo analisado.

Dados Sintéticos

Conforme descrito, os testes necessitam de dados para serem executados, no entanto, durante a construção do arcabouço, encontrar dados adequados e de qualidade foi a principal dificuldade enfrentada. Somado ao fato que os dados reais representam apenas alguns cenários possíveis, o uso de dados sintéticos como ferramenta de teste apresentou-se como uma solução viável. Isso permite controlar características dos dados, representar cenários menos usuais ou não encontrados nos dados reais e gerar quantidades expressivas de amostras em um curto período e com baixo custo, o que não ocorre com os dados reais para este problema, os quais exigem grande investimento para a instalação e manutenção de sensores e um grande período de tempo para a construção de um conjunto de dados.

Organização do Trabalho

Detalhando brevemente a estrutura do trabalho, este foi dividido em três partes.

A primeira parte (I) é composta de dois capítulos, os quais agrupam informações introdutórias dos dados, padrões estabelecidos e dos modelos que foram utilizados nos

testes. No capítulo 1 são descritos os modelos selecionados (tanto o modelo efetivamente testado, quanto os modelos para comparações), detalhando as arquiteturas e o interesse em utilizá-los. No capítulo 2 é detalhado os padrões estabelecidos pelo artigo, sobre quais informações dos dados são importantes, como os dados de treinamento devem ser gerados, e os hiperparâmetros e métricas utilizadas. Em seguida também é descrito o conjunto de dados utilizado, e as características destes dados.

Na segunda parte (II) é construído de modo incremental o arcabouço, ao longo de dois capítulos (3 e 4). No capítulo 3 é explicada a base teórica utilizada na formulação dos testes e a ideia dos elementos do aprendizado, enquanto no capítulo 4, a partir destes elementos, são criadas classes para agrupar os testes, e, por fim, unindo todas as ideias, o arcabouço de teste é finalizado.

Por fim, na terceira parte (III) é mostrado como o arcabouço pode ser aplicado, e a relação dele com os dados sintéticos. Essa parte é composta de três capítulos. No capítulo 5 é descrito a relação dos dados sintéticos com os testes, bem como os processos de testes associados ao arcabouço e aos dados sintéticos. Já no capítulo 6, o arcabouço desenvolvido é aplicado no problema de previsão tráfego, utilizando os modelos citados e seguindo os padrões estabelecidos. Por último, o capítulo 7 encerra o trabalho, listando os tópicos mais relevantes, os resultados mais interessantes dos testes aplicados e citando possíveis continuações para este projeto.

Parte I

Modelos e Padrões de Dados

Capítulo 1

Modelos de Aprendizado de Máquina

Conforme dito na introdução, a ideia do arcabouço de teste construído é testar um modelo de aprendizado de máquina supervisionado de interesse. No entanto, a fim de facilitar a análise dos resultados, considerou-se proveitoso, comparar o modelo efetivamente testado com modelos de referência. Nesse sentido, o parâmetro de escolha dos modelos de referência varia conforme o propósito. Algumas possibilidades são:

- Escolha baseada em complexidade: Nesse caso, a intenção é comparar o modelo testado com modelos de menor ou maior complexidade. Um dos motivos para essa escolha é verificar alguma hipótese relacionada à complexidade do modelo. Por exemplo, se o modelo testado é superior tanto em relação a modelos mais simples quanto a modelos mais complexos.
- Escolha baseada em desempenho: A intenção com essa escolha é comparar o modelo testado com outros modelos que obtiveram bons resultados para o problema. Pode ser a escolha ideal se, por exemplo, desenvolveu-se o modelo testado e deseja-se mostrar que ele é superior a outros não somente em desempenho com as métricas mas que possui um comportamento melhor (menos sensível a erros nos dados, menor necessidade de dados de treinamento, entre outros).
- Escolha baseada em variedade: Nesse caso, a ideia é comparar o modelo testado com modelos que possuem arquiteturas distintas (por exemplo, comparar modelos estatísticos com redes convolucionais e redes recorrentes). Um caso de uso para essa escolha é quando deseja-se mostrar que a arquitetura do modelo testado é superior às de outros modelos.

Os parâmetros de escolha citados não são excludentes, e, portanto, pode ser interessante combiná-los. Ressalta-se também que os parâmetros descritos são uma sugestão para guiar o processo de escolha, e outros parâmetros podem ser formulados e utilizados.

Nesse contexto, o modelo de interesse testado foi o modelo de rede neural para grafos (GNN) descrito no artigo Zhang *et al.*, 2019, conforme dito anteriormente. Em relação aos modelos de referência utilizados, selecionou-se um modelo de regressão linear e um modelo de rede neural totalmente conectada. A escolha se baseou na ideia de selecionar modelos de menor complexidade, e que possuem arquiteturas distintas, além de serem

os modelos mais representativos de suas arquiteturas e que se destacam por possuírem propriedades únicas.

Todos estes modelos foram implementados na linguagem *Python* utilizando a biblioteca *Tensorflow*, e mesmo a regressão linear foi escrita em forma de rede neural composta por apenas um neurônio de modo a facilitar a padronização dos testes.

1.1 GNN

1.1.1 Convolução

O artigo citado define um modelo de rede neural para grafos baseado na convolução de Chebyshev. A ideia por trás do uso de um modelo deste tipo e do uso desta convolução é capturar as dependências e correlações espaciais e temporais entre os sensores. Além disso, o operador utilizado diminui a complexidade de $\mathcal{O}(M^2)$ para um custo linear.

Esta convolução se baseia no laplaciano de um dígrafo (neste caso, o laplaciano simétrico de Chung), o qual, segundo a teoria espectral dos grafos, pode capturar a topologia do grafo e os padrões dos pesos das arestas. Tendo por base a matriz de transição P e a correspondente distribuição estacionária ϕ, o laplaciano do dígrafo é definido como:

$$\mathfrak{L} = I_M - \frac{\Phi^{1/2} P \Phi^{-1/2} + \Phi^{1/2} P^* \Phi^{-1/2}}{2} \tag{1.1}$$

Onde P^* é a matriz transposta e conjugada de P e Φ é uma matriz diagonal tal que $\Phi(v, v) = \phi(v)$. Este laplaciano pode ser diagonalizado como $\mathfrak{L} = U \Lambda U^T$, onde $\Lambda = diag([\lambda_0, \lambda_1, ..., \lambda_{M-1}]) \in \mathbb{R}^{M \times M}$, satisfazendo $0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_{M-1}$, e $U = [u_0, ..., u_{M-1}] \in \mathbb{R}^{M \times M}$ é a matriz dos correspondentes autovetores ortonormais ($\{u_i\}_{i=0}^{M-1}$ formam a base de Fourier do grafo), além disso, $UU^T = U^T U = I_M$.

Dado um grafo $\mathcal{G}$ com M vértices, um sinal $\mathbf{x} \in \mathbb{R}^M$ e um filtro g_θ, a operação de convolução para grafos induz a igualdade da equação 1.2, no qual a convolução se torna uma operação de convolução espectral. Isto ocorre pois $\hat{\mathbf{x}} = U^T \mathbf{x}$ é a transformada de Fourier do grafo (conduz o sinal ao domínio espectral), $\mathbf{x} = U\hat{\mathbf{x}}$ é a operação inversa e $g_\theta(\Lambda)$ é o correspondente filtro no domínio espectral.

$$\mathbf{y} = g_\theta *_{\mathcal{G}} \mathbf{x} = g_\theta(\mathfrak{L})\mathbf{x} = U g_\theta(\Lambda) U^T \mathbf{x} \tag{1.2}$$

Seja $g_\theta(\Lambda)$ formulado como um filtro polinomial parametrizado por $\boldsymbol{\theta} = [\theta_0, ..., \theta_{K-1}]^T \in \mathbb{R}^K$, onde $\boldsymbol{\theta}$ é aprendível no modelo:

$$g_\theta(\Lambda) = \begin{pmatrix} \sum_{k=0}^{K-1} \theta_k \lambda_0^k & & \\ & \ddots & \\ & & \sum_{k=0}^{K-1} \theta_k \lambda_{M-1}^k \end{pmatrix} = \sum_{k=0}^{K-1} \theta_k \Lambda^k \tag{1.3}$$

A partir desta fórmula, a equação 1.2 pode ser reescrita como:

$$\mathbf{y} = U g_\theta(\Lambda) U^T \mathbf{x} = U \sum_{k=0}^{K-1} \theta_k \Lambda^k U^T \mathbf{x} = \sum_{k=0}^{K-1} \theta_k U \Lambda^k U^T \mathbf{x} = \sum_{k=0}^{K-1} \theta_k \mathfrak{L}^k \mathbf{x} \tag{1.4}$$

Segundo a teoria espectral dos grafos, se a menor distância entre vértices u e v é maior que K, então $\mathfrak{L}^k(u, v) = 0$, assim, um filtro espectral de grafos de tamanho K tem acesso a todos os vértices com distância máxima de $K - 1$, e, portanto, uma operação de convolução espectral captura as dependências espaciais entre cada rua (vértice) e seus vizinhos de i-ésima ordem para $0 \leq i \leq K - 1$.

Embora a fórmula obtida na equação 1.4 possa capturar as dependência espaciais, o custo computacional para calcular $\mathfrak{L}^k$ é da ordem de $\mathcal{O}(M^2)$ devido as multiplicações, para reduzir essa complexidade uma alternativa é estimar $\mathfrak{L}^k$ por meio dos polinômios truncados de Chebyshev de forma recorrente. O polinômio de Chebyshev de ordem k pode ser computado pela relação $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$, onde $T_0(x) = 0$ e $T_1(x) = x$, no entanto, para garantir que essa relação seja estável, x deve satisfazer $x \in [-1, 1]$, assim o laplaciano é reescalado por meio da fórmula $\tilde{\mathfrak{L}} = \frac{2\mathfrak{L}}{\lambda_{M-1}} - I_M$, de modo que os autovalores de $\tilde{\mathfrak{L}}$ estarão no intervalo $[-1, 1]$.

Com base no polinômio de Chebyshev, a operação de convolução pode ser escrita como:

$$\mathbf{y} = g_\theta *_\mathcal{G} \mathbf{x} = g_\theta(\mathfrak{L})\mathbf{x} = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathfrak{L}})\mathbf{x} \tag{1.5}$$

Utilizando técnicas de multiplicação de matrizes esparsas, a complexidade da operação de convolução pode ser reduzida para $\mathcal{O}(K|\mathcal{E}|)$.

1.1.2 Arquitetura

Detalhando a implementação do modelo de GNN, a rede pode ser vista como uma sequência de L blocos de STGI-*Residual units*. Cada um destes blocos recebe como entrada uma matriz $X^{l-1} \in \mathbb{R}^{M \times N_{l-1}}$ e gera como saída uma matriz $X^{l-1} \in \mathbb{R}^{M \times N_l}$, onde l é o índice deste bloco. Estes componentes são formados por um mapa de identidade que recebe a entrada do bloco e a multiplica com uma matriz de redimensionamento $W_x^l \in \mathbb{R}^{N_{l-1} \times N_l}$. Paralelo a isto, a entrada do bloco é passado para V operadores STGC, que recebem também um parâmetro $K \in \{1, 2, ..., V\}$. Este operador STGC implementa uma convolução de Chebyshev. Cada um dos V operadores gera uma saída de dimensão $N_{l-1} \times N_l$ que são concatenadas formando uma matriz de dimensão $M \times VN_l$ (essa estrutura é conhecida como estrutura de *inception* Szegedy *et al.*, 2014), a qual passa por uma camada de redimensionamento, onde é multiplicada por uma matriz $W_d^l \in \mathbb{R}^{VN_l \times N_l}$. Por fim, as duas matrizes redimensionadas são somadas e passam por uma camada de ativação *ReLU*. É importante ressaltar que o último bloco STGI da rede não possui a função de ativação ao fim.

A arquitetura da rede pode ser vista na Figura 1.1.

O artigo diz que as duas camadas de redimensionamento podem ser camadas com parâmetros aprendíveis, no entanto, é citado que se obteve bons resultados ao definir

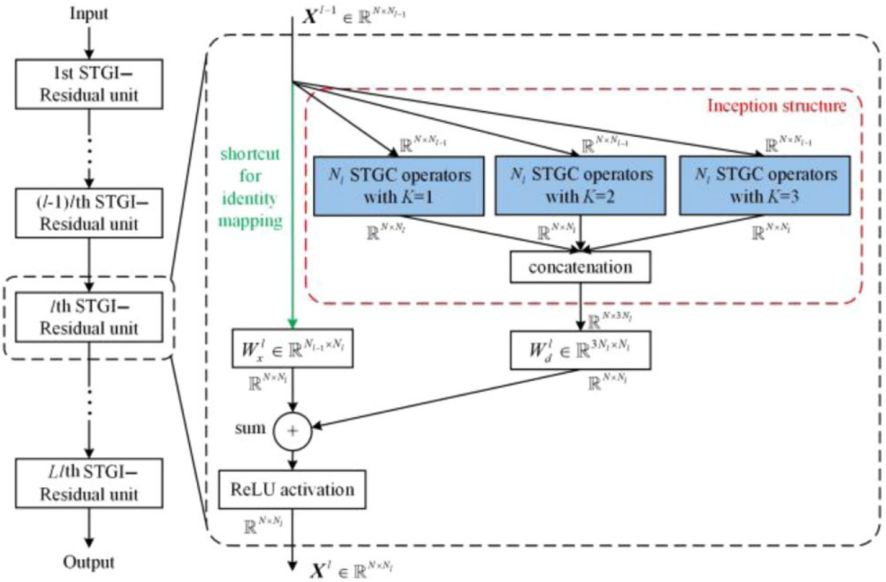


Figura 1.1: *Arquitetura do modelo de GNN, mostrando a sequência de blocos STGI, e detalhando a composição de um destes blocos.*

a matriz de redimensionamento do mapa de identidade como uma matriz com todas as posições iguais a 1, e, portanto, optou-se por esta implementação. Além disso, no artigo são comparados alguns resultados variando o número de blocos STGI entre 1 e 3, e variando a quantidade de convoluções em cada um deste blocos entre 4 e 32, sendo que o resultado ótimo foi obtido com 3 camadas STGI e 16 operadores STGC por camada, no entanto, testes com os dados selecionados para este trabalho mostraram que aumentar o número de camadas e de convoluções aumentava consideravelmente o tempo de treinamento e de convergência, e que os resultados não superavam aqueles obtidos com o modelo de GNN com 1 camada STGI e 4 operadores STGC, portanto, optou-se por utilizar este modelo mais simples.

1.2 Regressão Linear

Complementando a justificativa da escolha da regressão linear, selecionou-se esse modelo por ser o método de aprendizado de máquina mais simples conhecido para problemas de regressão, e cujo funcionamento é conhecido e interpretável. Parte da interpretabilidade decorre da resolução pelo método dos mínimos quadrados ordinários (OLS), o qual, por possuir uma solução analítica, permite obter suposições para atingir um resultado ótimo (Seber e Lee, 2012, Morettin, 2021). As suposições neste caso são:

- **Linearidade:** A relação entre as variáveis independentes e a variável dependente é linear.
- **Exogeneidade:** Os resíduos (diferença entre os valores esperados e os previstos) tem média 0 e não são correlacionados com as variáveis independentes.
- **Homocedasticidade:** A variância dos resíduos é constante.

- **Independência dos Erros:** Os resíduos são independentes entre si.
- **Normalidade dos Erros:** Os resíduos formam aproximadamente uma distribuição normal.
- **Ausência de Multicolinearidade:** As variáveis independentes não são perfeitamente colineares.

Neste trabalho, a regressão linear foi implementada no formato de rede neural, composta por apenas 1 neurônio (a implementação pode ser vista no Programa 1.1). Embora esta implementação possivelmente não demande estas suposições, as propriedades associadas podem ter impacto em qualquer modelo de aprendizado de máquina e portanto foram utilizadas para a formulação de testes.

Programa 1.1 Implementação da regressão linear

```
model = tf.keras.Sequential([
  tf.keras.layers.Input(shape=(lin, col)),
  tf.keras.layers.Dense(1)
])
```

Por fim, a interpretabilidade da regressão linear também está relacionada com a possibilidade de entender os resultados por meio de uma análise dos coeficientes, medindo erros, realizando testes de hipótese e associando o nível de importância de cada variável independente com a variável dependente.

1.3 Rede Neural Totalmente Conectada

Detalhando a arquitetura da rede neural totalmente conectada, ela é composta por 3 camadas. A primeira camada apenas recebe os dados, a segunda camada é a camada oculta composta por 128 neurônios, no qual cada neurônio possui uma função de ativação *ReLU* na saída, e a última camada é composta apenas por 1 neurônio que gera a saída. Comenta-se também que adicionou-se uma camada de *dropout* entre a camada oculta e a camada de saída, a qual desativa 20% dos neurônios da camada oculta de modo aleatório a cada iteração a fim de evitar *overfitting*. A implementação do modelo pode ser vista no Programa 1.2:

Programa 1.2 Implementação da rede neural totalmente conectada

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Input(shape=(lin, col)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(1)
])
```

Além dos motivos citados, outros motivos para a escolha deste modelo são o teorema da aproximação universal (Hornik *et al.*, 1989, Asadi e H. Jiang, 2020), o qual afirma que qualquer função contínua pode ser aproximada arbitrariamente bem por uma rede neural com uma camada oculta com número suficientemente grande de neurônios, e que

esta implementação é a mais simples de uma rede neural, sendo também um modelo de complexidade intermediária entre a regressão linear e a GNN. Nos gráficos e tabelas das seções seguintes, este modelo será referido apenas como "rede neural".

1.4 Outros Modelos Utilizados

Finalizando o capítulo, cita-se também que, a fim de verificar determinadas hipóteses, selecionou-se outros modelos para comparação com os modelos já citados. A seguir são descritos estes modelos, e no capítulo de aplicação do arcabouço de testes (capítulo 6) estão os correspondentes resultados das hipóteses formuladas.

- Modelo de rede neural convolucional (CNN) e modificações na arquitetura da GNN: O interesse com estes modelos era observar como alterar o tipo de convolução e excluir determinadas componentes da arquitetura afetam o resultado, tendo em vista que se observou que modelos mais simples apresentaram resultados semelhantes ao modelo de GNN, e, portanto, é possível formular a hipótese de que a contribuição de algumas camadas não é essencial para a obtenção do resultado.
- Regressões lineares únicas para cada sensor: A ideia para esse modelo é treinar modelos de regressões lineares únicos para cada sensor ao invés de uma única regressão linear para todos os sensores, tendo em vista a intuição de que com apenas uma regressão linear os parâmetros são compartilhados entre todos os sensores e, portanto, modelos individuais para cada sensor poderiam apresentar resultados melhores.
- Redes neurais totalmente conectadas para cada sensor: A mesma de treinar modelos únicos para cada sensor porém utilizando redes neurais totalmente conectadas. O objetivo principal é comparar com o modelo anterior de regressões lineares únicas.
- Síntese do sinal original a partir das transformadas de *Fourier* unidimensional e bidimensional: Embora este não seja um método de aprendizado de máquina, os dados utilizados possuem características periódicas (vide Seção 2.2.2). Além disso, cada sensor gera um sinal com características complexas, e, portanto, uma hipótese plausível é a de que um método que aprenda as relações periódicas e aproxime bem os detalhes do sinal original pode apresentar resultados melhores que os modelos de referência e de GNN.

Capítulo 2

Padrões dos Dados e Treinamento

O trabalho iniciou-se com um estudo de artigos de referência que abordam o problema de previsão de tráfego, com o objetivo de compreender os fundamentos do problema, dentre os quais Kanestrøm, 2017, W. Jiang e Luo, 2022 e Zhang *et al.*, 2019. Vários tipos de modelos de aprendizado de máquina têm sido empregados na resolução, dentre os quais modelos estatísticos, redes recorrentes e redes neurais para grafos. Além de serem arquiteturas distintas, cada uma delas também utiliza formatos e informações diferentes para a geração dos dados de treinamento. Por esse motivo, a fim de estabelecer um padrão para a formulação e execução dos testes, a referência principal seguida foi o artigo Zhang *et al.*, 2019, o qual descreve uma rede neural para grafos (GNN, descrita no Capítulo 1) e define e estabelece requisitos e padrões dos dados e dos hiperparâmetros utilizados no treinamento do modelo. Esses padrões são descritos a seguir. Além disso, são descritos neste capítulo os dados utilizados no treinamento dos modelos.

2.1 Os padrões de dados e métricas

2.1.1 Informações Necessárias e Geração dos Dados de Treinamento

Em primeiro lugar, o problema de previsão de tráfego necessita que a região monitorada possua uma rede de sensores que capturem o fluxo de veículos em um intervalo de tempo fixo. Os dados dos sensores são agrupados em uma matriz S de dimensão $M \times N$, na qual cada linha corresponde a um sensor e cada coluna corresponde a uma amostra de determinado instante do tempo (a matriz corresponde a uma série temporal).

Esta série temporal será utilizada para gerar uma lista de tuplas (X_t, Y_t), onde X_t se refere a um dado de treinamento, Y_t se refere aos respectivos valores esperados e t é o instante de tempo destes dados. Y_t é definido como a coluna de índice i da série temporal cuja amostra corresponde ao instante de tempo t. X_t é definido como a concatenação de dados de curto, médio e longo prazo, ou seja, $X_t = X_t^s \oplus X_t^m \oplus X_t^l \in \mathbb{R}^{M \times P}$. X_t^s, X_t^m e X_t^l são definidos da seguinte forma:

Dados de curto prazo: Formado pelas m_s amostras consecutivas anteriores a t. $X_t^s =$

$[x_{t-1}, x_{t-2}, ..., x_{t-m_s}] \in \mathbb{R}^{M \times m_s}$. x_j se refere à amostra correspondente ao instante de tempo j da série temporal.

Dados de médio prazo: Formado pelas m_m amostras anteriores a t, porém com intervalo de tempo t_m entre as amostras selecionadas. $X_t^m = [x_{t-t_m}, x_{t-2\times t_m}, ..., x_{t-m_m \times t_m}] \in \mathbb{R}^{M \times m_m}$.

Dados de longo prazo: Formado pelas m_l amostras anteriores a t, porém com intervalo de tempo t_l entre as amostras selecionadas. $X_t^l = [x_{t-t_l}, x_{t-2\times t_l}, ..., x_{t-m_l \times t_l}] \in \mathbb{R}^{M \times m_l}$.

O artigo cita que, idealmente, os valores de m_m e m_s devem ser respectivamente tais que o intervalo de tempo entre a amostra inicial e final de X_t^m seja de 1 dia, e no caso de X_t^l esse intervalo seja de 1 semana. No caso de m_s o artigo não define um valor específico, citando apenas que obteve bons resultados com valores entre 1 e 6, assim definiu-se $m_s = 2$ como padrão nos testes realizados. Além disso, os valores de t_m e t_l são dependentes do intervalo de amostragem dos dados, o artigo definiu como intervalo de tempo entre as amostras de médio e longo prazo respectivamente 30 minutos e 60 minutos.

Comenta-se também que os dados foram normalizados a fim de que a magnitude dos dados não influenciasse nos resultados e para facilitar comparações. Em alguns testes, utilizou-se os dados não normalizados quando a magnitude dos dados não era relevante ou quando era de interesse observar propriedades presentes somente nos dados dessa forma.

Modelos mais simples, como a regressão linear e redes neurais totalmente conectadas, necessitam somente da série temporal para a geração dos dados de entrada. Porém, o modelo de GNN requer adicionalmente um grafo para aprender as relações espaciais, de modo que também são necessários os seguintes dados para o seu funcionamento:

Grafo com as relações espaciais da região: Cada sensor monitora apenas um sentido de um trecho de uma via. Sendo assim, é necessário conectar os sensores a fim de formar a malha viária e representar esta malha por meio de alguma estrutura matemática (no caso, grafo). O tipo de convolução utilizado na GNN requer que o grafo possua algumas propriedades, dentre as quais:

- O grafo deve ser modelado como dígrafo, de modo a clarificar as transições válidas.
- As ruas devem ser representadas por arestas e conexões entre ruas por vértices.
- O dígrafo deve ser fortemente conexo, ou seja, deve ser irredutível. Para obter esta propriedade permite-se a seguinte modificação: nos vértices u sem arestas partindo deles ou sem arestas com destino em u (por exemplo, ruas sem saída), adiciona-se uma aresta (u, v) ou (v, u), onde v é o vértice mais próximo com sentido oposto. A escolha entre (u, v) ou (v, u) depende do contexto do vértice u.
- O dígrafo deve ser aperiódico. Para atingir essa propriedade adicionam-se *loops* em cada vértice (ou seja, arestas (u, u)), os quais também representam o fato dos veículos poderem se manter no mesmo trecho.

Dados de velocidade dos veículos: Além do grafo capturar as dependências espaciais entre os vértices, ele também é utilizado para gerar uma cadeia de Markov, a qual gera valores que se referem à probabilidade de transição de um vértice u para um

vértice v vizinho, e estes valores podem ser utilizados como pesos das arestas. As propriedades do dígrafo ser irredutível e aperiódico são importantes para que exista somente uma única distribuição estacionária. A matriz de transição é calculada da seguinte forma:

$$p_{uv} = \begin{cases} (vmax_u - vmean_u)/vmean_u & \text{se } u \neq v \\ (1 - p_{uu})p_{uv} & \text{se } u \neq v \text{ e } (u, v) \text{ é uma aresta} \\ 0 & \text{caso contrário} \end{cases}$$

Esta fórmula necessita que cada sensor u também capture dados de velocidade, de modo que seja possível estimar valores únicos de velocidade média ($vmean_u$) e de velocidade máxima ($vmax_u$) para este sensor (a estimativa é única para todas as amostras da série temporal e não no sentido de haver uma estimativa para cada amostra). Esta fórmula garante que $0 \leq p_{uv} \leq 1$ e que $\forall u(\sum_v p_{uv} = 1)$.

2.1.2 Hiperparâmetros

Em relação aos hiperparâmetros utilizados no treinamento, o artigo utilizou treinamento em lote (*batch*), o otimizador *Adam* e usou como função de perda o erro quadrático ($\|Y_t - \hat{Y}_t\|^2$, Y_t são os valores esperados e $\hat{Y}_t$ são os valores calculados). Em algumas situações o erro quadrático não permitia que o modelo convergisse; nesses casos utilizou-se o erro quadrático médio. Além disso, a divisão dos dados sugerida pelo artigo foi uma divisão sequencial, pois os dados originalmente formam uma série temporal (ou seja, se houver 1 mês de dados, uma divisão possível seria a 1° semana como conjunto de treinamento, a 2° semana como conjunto de validação e o restante como conjunto de teste). Assim, optou-se por uma divisão 40%/10%/50%, onde os primeiros 40% dos dados compõem o conjunto de treinamento, os 10% seguintes geram o conjunto de validação e o restante compõem o conjunto de teste. Escolheu-se deixar uma maior quantidade de dados no conjunto de teste para poder verificar casos em que o modelo não generaliza para períodos de tempo muito distante do período do conjunto de treinamento.

2.1.3 Métricas

Por fim, para avaliar o desempenho dos modelos utilizou-se 4 métricas: raiz do erro quadrático médio (RMSE - *Root Mean Square Error*), erro absoluto médio (MAE - *Mean Absolute Error*), RMSE normalizado (NRMSE), e erro percentual absoluto médio (MAPE - *Mean Absolute Percentage Error*). Essas métricas podem ser definidas da seguinte forma:

$$RMSE = \sqrt{\frac{1}{mM} \sum_{j=1}^{M} \sum_{i=1}^{m} (y_i^{(j)} - \hat{y}_i^{(j)})^2}$$

$$MAE = \frac{1}{mM} \sum_{j=1}^{M} \sum_{i=1}^{m} |y_i^{(j)} - \hat{y}_i^{(j)}|$$

$$NRMSE = \frac{1}{M}\sum_{j=1}^{M}\left|\frac{RMSE_j}{y_{max}^{(j)} - y_{min}^{(j)}}\right| \times 100\%$$

$$MAPE = \frac{1}{mM}\sum_{j=1}^{M}\sum_{i=1}^{m}\left|\frac{y_i^{(j)} - \hat{y}_i^{(j)}}{y_i^{(j)}}\right| \times 100\%$$

É observado ainda no artigo que a métrica MAPE é sensível a valores próximos de 0. Por este motivo, em muitos testes, os valores desta métrica difere muito em ordem de grandeza em comparação às demais métricas, não sendo portanto relevante considerá-la nas análises dos respectivos testes.

2.2 Treinamento

Os testes formulados, além de necessitarem dos modelos já escolhidos na Seção 2.1, também requerem dados reais de qualidade para os respectivos treinamentos, tendo em vista que muitas características testadas são inerentes a esse tipo de dado, e que, portanto, um conjunto de dados reais de boa qualidade serve como amostra e referência do que se pode esperar ao utilizar outros dados reais para o problema.

2.2.1 Seleção dos Dados

Selecionou-se o conjunto de dados do Caltrans PEMS (*Caltrans. Performance Measurement System (PeMS)* 2024), o qual é formado por uma extensa rede de sensores (aproximadamente 39 mil) no estado da Califórnia (EUA) e possui dados coletados desde 2001. Especificamente, utilizou-se os dados do distrito 3 (PEMSd3). Este conjunto de dados é amplamente utilizado nos experimentos de artigos que abordam o problema de previsão de tráfego e possui alta qualidade, não somente pela grande quantidade de sensores e amostras, mas principalmente pela baixa quantidade de dados faltantes, presença de dados de velocidade e muitas informações nos metadados, permitindo conectar os sensores e formar o grafo da região analisada.

Também se tentou utilizar outros dois conjuntos de dados. O primeiro, originário da iniciativa NYC *Open Data* (New York, 2024), a qual reúne diversos dados públicos da cidade de Nova Iorque, não foi adequado pois os sensores encontravam-se muito distantes entre si, impossibilitando a formulação da malha viária como grafo, somado ao fato de haver uma quantidade consideravelmente grande de dados faltantes. O segundo conjunto de dados chama-se UTD19 (Ambühl *et al.*, 2024), sendo composto por dados de 40 cidades (localizadas majoritariamente na Europa). No entanto, apenas 8 cidades possuem dados de velocidade dos veículos, e estas 8 cidades apresentam um baixo número de sensores e um período de tempo das amostras muito curto (entre 1 semana e 1 mês), o que impossibilitou a execução de alguns testes.

Retornando ao conjunto de dados PEMSd3, a maior parte dos testes foi feita utilizando apenas dados do mês de janeiro de 2023 (o que corresponde a 8640 amostras, e essa quantidade foi utilizada como padrão para o número de amostras dos dados sintéticos). Em

certos testes, utilizou-se dados de janeiro e fevereiro de 2023. Nestes dois casos, os dados faltantes eram calculados por meio de interpolação tendo como base dados de janeiro a agosto de 2023. A partir deste ponto, quando for mencionado o conjunto de dados PEMSd3, a referência serão aos dados de janeiro de 2023, e os testes que utilizaram um período diferente farão menção a isto explicitamente.

Do ponto de vista geográfico e espacial, a região dos sensores deste conjunto de dados gera uma malha composta principalmente de rodovias em torno da cidade de Sacramento, conforme pode ser visto nas figuras 2.1 e 2.2:

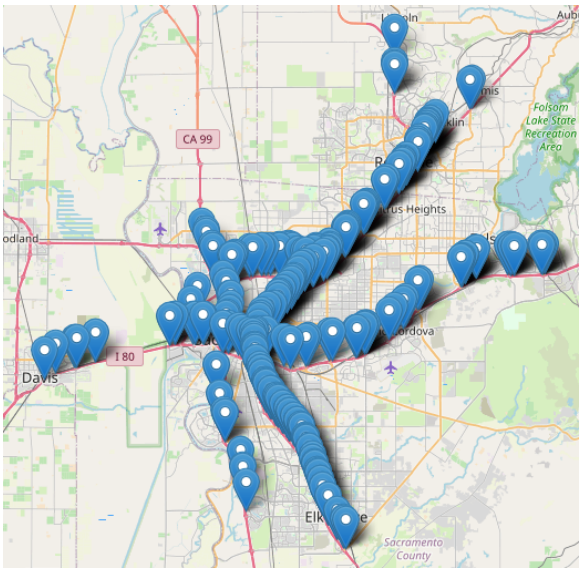


Figura 2.1: *Imagem contendo a malha viária dos sensores do PEMSd3, cada sensor está apontado por um marcador no mapa.*

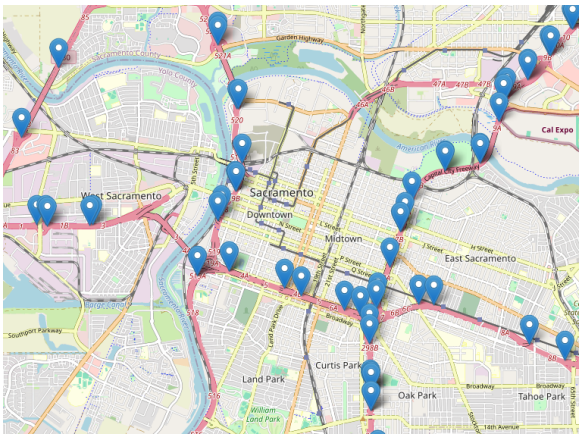


Figura 2.2: *Imagem mostrando mais em detalhes a região central da malha viária.*

2.2.2 Características dos Dados

Nesta seção, são apresentadas as principais características dos dados reais, as quais foram utilizadas para a formulação e análise dos testes.

Analisando a série temporal do conjunto de dados PEMSd3, nota-se um padrão periódico, conforme pode ser visto nas figuras 2.3 e 2.4. O fluxo de veículos é influenciado pelo horário, dia, período do ano, entre outros fatores. Porém, ainda assim, é esperado que haja periodicidades diárias, semanais, mensais e anuais, tendo em vista que a rotina e modo de vida da população não se alteram significativamente em um curto período, a menos que haja algum evento de grande impacto, como desastres ambientais ou uma epidemia, por exemplo.

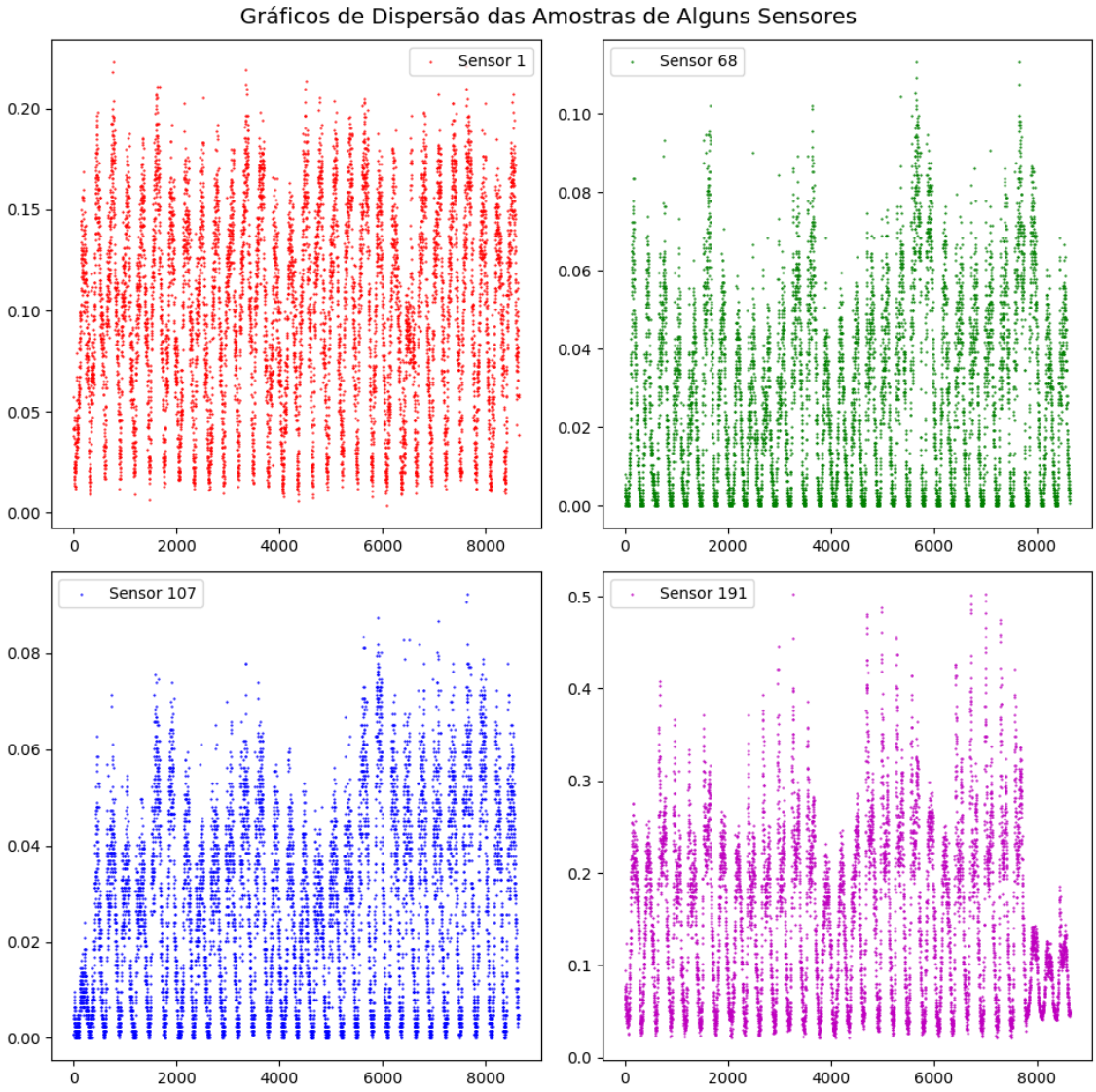


Figura 2.3: *Gráficos de dispersão de alguns sensores do conjunto de dados do PEMSd3, mostrando mais detalhadamente a forma da periodicidade dos dados. O eixo X representa o índice da amostra e o eixo Y representa o valor da amostra.*

Do ponto de vista estatístico, a série temporal possui as métricas estatísticas básicas descritas na Tabela 2.1, as quais podem ser verificadas ao observar o histograma dos dados correspondente na Figura 2.5. Uma observação importante é que nos dados normalizados, o valor máximo não é 1, pois normalizou-se os dados do período de janeiro a agosto e

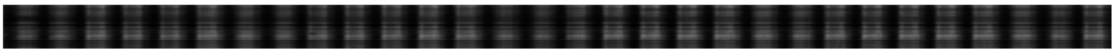


Figura 2.4: *Imagem em nível de cinza dos valores da série temporal do conjunto de dados PEMSd3, mostrando a periodicidade existente nos dados.*

não somente os dados de janeiro.

	Não Normalizado	**Normalizado**
Mínimo	0	
Máximo	1007.0	0.808835
Média	149.907923	0.120408
Mediana	108.0	0.086747
Variância	16732.13059	0.010795
Coeficiente de Variância	86.288126	
Desvio Padrão	129.352737	0.103898
Coeficiente de Assimetria	1.068348	

Tabela 2.1: *Tabela contendo métricas estatísticas básicas do conjunto de dados PEMSd3*

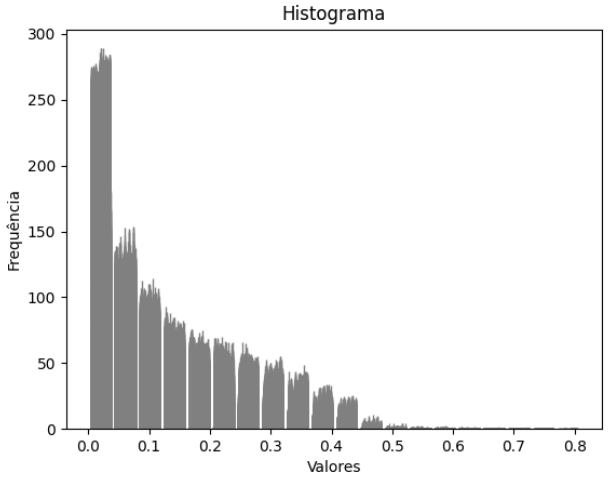


Figura 2.5: *Histograma do conjunto de dados do PEMSd3 (normalizado).*

2.2.3 Uso dos Dados para Treinamento dos Modelos

Finalizando o capítulo, o treinamento de todos os modelos citados no Capítulo 1 com os dados do PEMSd3 resultou nas métricas da Tabela 2.2. Como é possível observar, os 3 modelos principais (GNN, regressão linear e rede neural) apresentaram resultados praticamente idênticos, enquanto os demais apresentaram resultados inferiores. Este fato será melhor explorado e compreendido no capítulo de aplicação dos testes (Capítulo 6).

Comparações de Modelos - Resultados do Conjunto de Teste - PEMSd3					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Regressão Linear	0.000207	0.0137	0.135	0.0095	46440.93
Rede Neural	0.000199	0.0132	0.126	0.0092	50438.85
GNN	0.000224	0.0143	0.144	0.01	58302.98
Regressão Linear por Sensor	0.0125	0.112	0.875	0.0793	10^{12}
Rede Neural por Sensor	0.0120	0.110	0.858	0.0779	10^{12}
Síntese pela Transformada de Fourier unidimensional	0.00281	0.0530	0.419	0.0321	10^{11}
Síntese pela Transformada de Fourier bidimensional	0.0188	0.137	1.130	0.106	10^{12}

Tabela 2.2: *Tabela contendo erro e métricas ao avaliar o conjunto de teste ao realizar o treinamento com os dados do PEMSd3.*

Parte II

Arcabouço de Teste

Capítulo 3

Formulação dos Testes

No contexto de desenvolvimento de software, qualquer problema, programa ou sistema possui requisitos funcionais associados, os quais descrevem o que o objeto em questão deve fazer ou como deve se comportar (Graham *et al.*, 2008). A documentação e representação dos requisitos do problema ou sistema é chamado de especificação, e pode ser descrita de diversas formas, desde diagramas, algoritmos ou até mesmo pelo próprio código do programa. A fim de garantir que a especificação representa o problema real e os requisitos estão sendo cumpridos no produto final, formulam-se testes para se certificar que o software produz os resultados esperados.[1]

Cada teste verifica um requisito específico, além disso, os requisitos podem estar associados a diferentes níveis do sistema, de modo que é gerada uma hierarquia dos testes (Desikan e Ramesh, 2007). Por exemplo, existe classicamente a divisão em testes de unidade, de componentes, de integração e de sistema (classificação com base em níveis). Cada um desses tipos de testes cumpre um papel distinto ao garantir a confiabilidade de um determinado nível do software.

Diferentemente do software tradicional, os testes associados ao aprendizado de máquina precisam ser realizados em estágios distintos no desenvolvimento:

- Os testes dos modelos de aprendizado. Nesse, após o treinamento do modelo com um conjunto de dados, os testes, com novos conjuntos de dados, são realizados para certificar se o modelo produz as respostas/predições apropriadas ao problema. Com isso, pode-se certificar se o método utilizado é apropriado, ou mesmo fazer ajustes necessários aos dados de treinamento.
- Os testes de sistemas baseados em aprendizado de máquinas. Nesse estágio de desenvolvimento, os testes dos modelos já foram realizados e aqui o objetivo é testar o sistema em relação aos requisitos. O objetivo desse teste é validar o sistema e fazer um levantamento dos bugs em relação ao código introduzido ao novo sistema e se o modelo de aprendizado combinado com o código produz os resultados esperados.

[1] Referências utilizadas sobre testes: Desikan e Ramesh, 2007, Graham *et al.*, 2008, Myers *et al.*, 2012, Ammann e Offutt, 2008.

O trabalho atual está concentrado nos testes do modelo de aprendizado de máquina, e propõe uma metodologia mais completa para os testes dos modelos. Todavia, dada a natureza opaca dos métodos utilizados, formular os testes em termos de comportamentos esperados não é uma tarefa fácil, e, por este motivo, os testes têm como objetivo explicitar comportamentos. Assim, idealizou-se um processo inverso, no qual, dado um determinado objeto de teste (e suas características e propriedades), é analisado como os modelos reagem a esse objeto segundo os resultados obtidos. No contexto do aprendizado de máquina, os objetos são elementos envolvidos ou resultantes deste processo. A confiabilidade desses testes é garantida pela convergência e repetição dos treinamentos, e, quando possível, pela utilização de mais de um conjunto de dados.

3.1 Introdução ao aprendizado de máquina supervisionado

Antes de prosseguir para a formulação dos testes, é necessário compreender a ideia do aprendizado de máquina supervisionado Abu-Mostafa *et al.*, 2012; Vapnik, 1998; Mitchell, 1997; Shalev-Shwartz e Ben-David, 2014. De modo genérico e simplificado, todo algoritmo de aprendizado de máquina supervisionado segue o esquema da Figura 3.1.

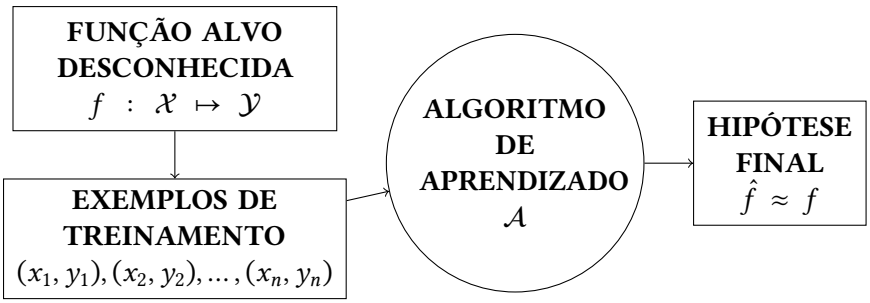


Figura 3.1: *Esquema genérico adaptado do livro Abu-Mostafa et al., 2012, o qual mostra a ideia do aprendizado de máquina supervisionado.*

Este esquema mostra que, a partir de uma função objetivo (ou alvo) f desconhecida, são gerados os dados, e que, com base nesses dados, o algoritmo de aprendizado tenta estimar a função objetivo ($\hat{f}$). Associado ao algoritmo de aprendizado há um espaço de hipóteses que é o conjunto de funções que este algoritmo é capaz de aprender.

Embora simplificado e ignorando um cenário imperfeito no qual os dados estão sujeitos a erros (em geral aleatórios), o esquema exposto é suficiente para a formulação dos testes.

3.2 Formulação dos testes de modelos de aprendizado de máquinas

Formalizando os testes para o aprendizado de máquina, a resolução de um problema por meio de métodos de aprendizado envolve quatro elementos fundamentais (ou obrigatórios):

dados brutos,[2] padrões definidos, um modelo, e dados de treinamento. Os padrões definidos se referem a transformações aplicadas sobre os dados brutos para gerar os dados de treinamento.[3] Trata-se de um elemento obrigatório, pois em alguns problemas não é possível trabalhar diretamente com os dados brutos, como é o caso de problemas de processamento de linguagem natural, no qual o texto é transformado em *tokens* ou *embeddings*.

Relacionando estes elementos com o esquema da Figura 3.1, pode-se dizer que o modelo está relacionado ao algoritmo de aprendizado, e os dados de treinamento estão relacionados aos dados de exemplo. Dada a simplicidade do esquema, os dados brutos e os padrões definidos encontram-se implícitos na função objetivo e na relação com os dados de treinamento. Assim, a relação destes elementos com a função objetivo e com os dados de treinamento está detalhada a seguir.

Todo problema pode ser visto como composto por uma função implícita $g(v_0, v_1, ..., v_N) = \hat{x}$, na qual $\hat{x}$ é um dado bruto e v_i são variáveis desconhecidas associadas a definição dos dados brutos. Com base em $\hat{x}$, são gerados os dados efetivamente usados no treinamento dos modelos (x), relacionando-se pela função $h(\hat{x}) = x$ e (h pode ser interpretado como a função que aplica os padrões definidos aos dados brutos).

Como dito anteriormente, todo modelo de aprendizado de máquina supervisionado tem como meta aprender uma função alvo que relaciona x com y. Com base nas funções anteriores, a função alvo pode ser escrita como: $f(x) = y \rightarrow f(h(\hat{x})) = y \rightarrow f(h(g(v_0, v_1, ..., v_n))) = y$. É evidente que a função objetivo é dependente das variáveis dos dados brutos. No entanto, as variáveis dos dados brutos são em geral desconhecidas e, mais do que isso, as funções g, h e f estão encadeadas, dificultando uma análise de contribuição individual de cada função. A fim de superar este empecilho, a estratégia adotada pelos testes envolvendo os dados brutos, os padrões e o modelo é variar somente um desses elementos fundamentais, mantendo os elementos restantes fixos.[4] Analisando cada possibilidade:

- Variar $\mathcal{A}$ significa variar o modelo e manter fixos os dados brutos e os padrões utilizados para a geração dos dados de treinamento.
- Variar h implica manter fixos o modelo e os dados brutos, porém modificar os padrões utilizados para a geração dos dados de treinamento.
- Variar g implica utilizar outra fonte de dados e manter fixos o modelo e os padrões utilizados para a geração dos dados de treinamento.

Cada um dos elementos fundamentais possuem restrições associadas, as quais estão implícitas nos conjuntos $\mathcal{V}$, $\hat{\mathcal{X}}$, $\mathcal{X}$, e $\mathcal{Y}$ (Figura 3.2). Cada um desses conjuntos estão relacionados aos domínios ou imagens das respectivas funções de cada um dos elementos fundamentais.

A partir dos elementos fundamentais, é possível complementar o esquema da Figura 3.1, obtendo o seguinte esquema:

---

[2] Dado bruto se refere ao dado na forma em que foi medido ou obtido.

[3] No Capítulo 2, o formato dos dados de treinamento foi definido como uma concatenação de dados de curto, médio e longo prazo. Esse formato é apenas um padrão escolhido e que pode ser substituído por outro.

[4] Testes envolvendo os dados de treinamento não usam essa estratégia dado que eles são dependentes dos dados brutos e dos padrões.

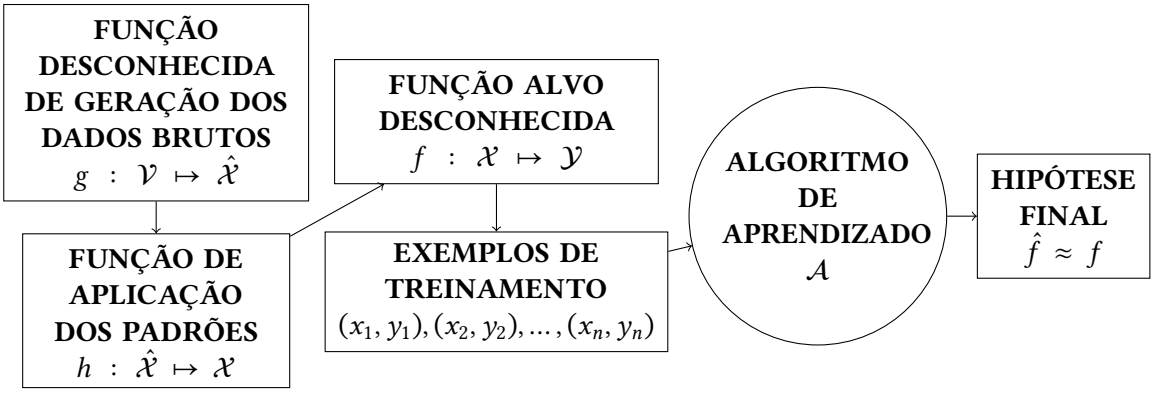


Figura 3.2: *Esquema incrementado do aprendizado de máquina supervisionado, explicitando os elementos fundamentais.*

3.2.1 Estudo de Caso das Variáveis dos Dados Brutos e das Restrições para o Problema de Previsão de Tráfego

Nesta seção, o problema de previsão de tráfego é analisado em relação às variáveis dos dados brutos e as restrições dos elementos fundamentais, a fim de exemplificar como esse processo pode ser aplicado a outros problemas.

Iniciando com as variáveis dos dados brutos, primeiramente pode-se pensar o que é capaz de influenciar o fluxo de veículos. Nesse sentido, as variáveis mais evidentes são: data, horário, velocidade média por via, velocidade máxima por via e quantidade de faixas por via. Pensando agora em como os dados são capturados e representados de modo bruto, eles são capturados por sensores instalados em determinados pontos da via, e as amostras são capturadas em um intervalo de amostragem fixo, gerando um dado bruto no formato de série temporal. Sabendo disso, outras variáveis possíveis são: distância entre sensores consecutivos e intervalo de amostragem.

As variáveis citadas são as mais prováveis e fáceis de identificar. Além disso, também é possível pensar em outras variáveis como o clima. No entanto, tendo em vista que os dados reais não possuem esse tipo de informação e nem o modelo utiliza essa variável explicitamente, não é possível testá-la.

Em relação as restrições dos elementos fundamentais, estas são essencialmente restrições matemáticas aos valores dos respectivos conjuntos $\mathcal{V}$, $\hat{\mathcal{X}}$, $\mathcal{X}$, e $\mathcal{Y}$, por exemplo, no caso do conjunto de dados do PEMSd3, $\mathcal{X}$ é o conjunto $\mathbb{R}^{N \times M}$, e no caso dos padrões definidos (seguindo o que foi descrito no Capítulo 2, o conjunto $\hat{\mathcal{X}}$ se refere ao conjunto de valores no intervalo de 0 a 1, de dimensionalidade $A \times B$ e cujo grafo dos sensores é fortemente conexo.

3.3 Elementos Derivados

O objetivo com um método de aprendizado de máquina é estimar a função alvo, como já mencionado anteriormente, de forma que essa estimação é resultante de um processo de treinamento. Esse processo de treinamento é dependente dos elementos fundamentais, como explicitado pela expressão $f(h(g(v_0, v_1, ..., v_n))) = y$. Tendo em vista que o processo

de aprendizado não é perfeito, a função aprendida pelo modelo não será exatamente $f(x)$, mas sim uma aproximação $\hat{f}(x)$. Por esse motivo, o resultado da função $\hat{f}(X)$ será $\hat{y}$, que se refere aos valores previstos para x, indicando que os valores previstos podem diferir de y.

A complexidade do aprendizado de máquina está em conectar a função alvo f com a função aprendida $\hat{f}(X)$, pois esse processo envolve otimizar uma função de custo e, com base nesta otimização, ajustar os pesos. A conexão entre a otimização e o ajuste dos pesos não é clara sobretudo em implementações mais complexas, o que é a principal causa da dificuldade em compreender o funcionamento dos métodos de aprendizado de máquina supervisionado.

Dado esse contexto, é possível identificar 2 elementos resultantes do processo de aprendizagem: a função aprendida $\hat{f}(X)$ e os valores previstos $\hat{y}$. Nos modelos de aprendizado de máquina a função aprendida é descrita pelos valores dos parâmetros. Ao invés de analisar diretamente $\hat{y}$, uma alternativa é analisar os resíduos do modelo (no caso de problemas de regressão), ou seja, a diferença entre os valores esperados e previstos ($\bar{y} = y - \hat{y}$). Sabendo que estes elementos são dependentes tanto dos elementos fundamentais quanto do processo de aprendizado, eles serão chamados de elementos derivados.

O interesse em utilizar e verificar os resíduos decorre do fato que muitas análises atualmente tomam como base os resíduos para constatar informações da qualidade do ajuste do modelo. Por exemplo, resíduos normalmente distribuídos, com homocedasticidade, sem autocorrelação e sem *outliers* em geral são sinais que o modelo se ajustou corretamente aos dados, aprendendo os padrões de acordo com as limitações do modelo, e que nenhuma região da função alvo foi mal aproximada. No entanto, é mais complicado afirmar o caso oposto, ou seja, que a ausência de alguma destas propriedades impactam o resultado, tendo em vista que diferentes arquiteturas podem ter comportamentos variados e, portanto, exemplificando, não necessariamente o fato dos resíduos apresentarem heterocedasticidade será indicativo de um ajuste ruim.

3.4 Informações Obtidas com o Processo de Teste de Modelos

Complementando o fato de que os testes para modelos de aprendizado de máquina têm como objetivo explicitar comportamentos, cada teste tem a capacidade de revelar diversos tipos de informações.

Primeiramente, um tipo de informação que todos os testes revelam obrigatoriamente é o seguinte:

- Comportamento do modelo ao atributo[5] testado: Tendo em vista que todos os testes utilizam como meio e ferramenta de teste algum modelo de aprendizado de máquina, esses testes evidenciarão como o modelo reage ao atributo verificado pelo teste. Por exemplo, se o objetivo do teste é avaliar o resultado do modelo ao utilizar dados lineares, então será evidenciado como essa característica influencia no resultado (se

[5] Atributo neste e nos capítulos seguintes se refere genericamente a algo que compõe algum elemento fundamental ou derivado, tal como uma característica ou propriedade.

o modelo apresenta resultados melhores ou piores com esse tipo de dado, ou se ele é adequado ou não para esse tipo de dado).

Além disso, os testes também são capazes de explicitar alguma informação dos seguintes tipos, as quais estão relacionadas aos elementos fundamentais:

- Requisitos dos dados utilizados: Quando um teste envolve alterar alguma característica ou propriedade dos dados de treinamento, ele também pode revelar informações sobre os dados de treinamento ideais para obter o melhor desempenho do modelo. Um exemplo desse tipo de teste é um teste de adição de ruído, o qual, além de avaliar como os modelos se comportam com a presença de ruído, também permite determinar um nível aceitável de ruído presente nos dados.
- Importância das variáveis dos dados brutos: Conforme mencionado, as funções g, h e f estão encadeadas, nesse contexto, os dados brutos, os padrões e os dados de treinamento são dependentes das variáveis dos dados brutos. Assim, se um teste verifica o impacto de alguma variável dos dados brutos sobre algum desses elementos, esse tipo de informação pode ser obtida.
- Efetividade das restrições dos domínios: Quando um teste se propõe a violar alguma restrição dos domínios dos elementos, com o objetivo a avaliar a importância dessa restrição. Por exemplo, se o modelo utilizado é probabilístico e tem como restrição que as variáveis devem ser independentes, um teste que viola essa restrição seria utilizar variáveis dependentes.
- Efetividade dos padrões impostos: Análogo ao item anterior, isso ocorre quando o teste se propõe a violar algum padrão imposto. Por exemplo, retornando ao Capítulo 2, definiu-se o padrão de que cada dado de entrada deve ser formado concatenando dados de curto, médio e longo prazos. Um teste que se proponha a definir os dados de treinamento de outra forma avaliará a efetividade deste padrão.

Por fim, os testes para os elementos derivados são capazes de revelar os seguintes tipos de informações (além das citadas anteriormente):

- Interpretabilidade dos Parâmetros: Todos os testes que verificam os parâmetros têm a capacidade de revelar informações que ajudem a interpretar e compreender os parâmetros de cada camada e como eles afetam e transformam os dados, tendo em vista que, com exceção de modelos simples como a regressão linear, os parâmetros não são facilmente interpretáveis.
- Interpretabilidade dos Resíduos: Os testes para resíduos têm a capacidade de explicitar informações que permitem interpretar se determinadas propriedades dos resíduos estão correlacionadas com um pior resultado e um pior ajuste dos modelos.

3.5 Considerações sobre Testes e o Processo de Aprendizagem

O processo de aprendizado não afeta a realização dos testes com os elementos fundamentais, considerando que os resultados dos aprendizados são utilizados apenas para

comparações. Além disso, dado que os hiperparâmetros do modelo foram fixados para todos os testes,[6] pode-se considerar o processo de aprendizado um elemento constante. Em relação aos elementos derivados, eles não são independentes da escolha dos componentes do aprendizado, ou seja, da função de custo, do método de otimização e do método de ajuste dos pesos, de forma que a escolha desses componentes afeta o resultado obtido. Apesar disso, ainda é possível realizar os testes tratando o aprendizado como um elemento fixo e constante. No entanto, todos os resultados se limitarão aos hiperparâmetros de aprendizado escolhidos.

Além disso, tendo em vista que o processo de aprendizado impõe uma barreira que impede de relacionar os elementos fundamentais com os elementos derivados, os testes desses elementos (especialmente os testes dos resíduos) utilizarão outro processo de teste no qual a influência de alguma propriedade de interesse será verificada agrupando conjuntos de dados com base nos resultados e na ocorrência dessa propriedade. Este processo está melhor detalhado no Capítulo 5.

Por fim, comenta-se que não faz sentido testar o processo de aprendizado em si, pois os componentes do aprendizado possuem comportamento conhecido e independente dos elementos fundamentais. Por exemplo, se o método de ajustes dos parâmetros é por *backpropagation*, sabe-se como os parâmetros são ajustados apesar de determinar os valores dos parâmetros ser uma tarefa mais complexa, além da forma que os parâmetros são ajustados ser a mesma independente dos valores utilizados no treinamento. Assim, não faz sentido modelar testes para o processo de aprendizado, pois variar os elementos fundamentais não alterará o comportamento. Dessa forma, uma alternativa melhor é um estudo matemático para compreender quando é mais adequado utilizar os diferentes tipos de funções de custo, de métodos de otimização e de ajuste dos parâmetros.

[6] No Capítulo 2 definiu-se que todos os testes neste trabalho utilizariam a função de custo igual ao erro quadrático, a função de otimização *Adam* e o ajuste dos pesos por *backpropagation*.

Capítulo 4

Um Arcabouço de Testes baseado em Elementos do Aprendizado

Como visto no Capítulo 3, a partir do estudo do processo de aprendizado de máquina identificou-se os elementos associados, bem como as informações que podem ser obtidas pelos testes que exploram estes elementos. Dado que os elementos fundamentais encontram-se encadeados, a formulação válida encontrada para os testes é que cada teste deve agir unicamente sobre um elemento e sobre algum atributo deste elemento, e, no caso dos elementos derivados, a formulação dos testes é diferente (vide Capítulo 5), mas ainda mantém a propriedade de cada teste operar sobre um único elemento.

Neste capítulo são definidos os testes a serem incluídos no arcabouço, baseados em seus elementos e atributos explorados, e quais informações são obtidas em contraparte. Para tanto, uma classificação de testes baseada em elementos e atributos é inicialmente definida para então mostrar o arcabouço proposto.

4.1 Classificação dos Testes baseada em Elementos e Atributos

Conforme mencionado anteriormente, os testes foram classificados com base nos elementos e atributos associados a esses elementos que os testes exploram. A seguir, são descritos os diferentes tipos de atributos, bem como exemplos de testes da classe, e como eles podem ser realizados.

4.1.1 Testes de Propriedades dos Dados

Essa classe agrupa os testes que exploram os dados. Mais especificamente, o tipo de atributo que os testes dessa classe exploram é alguma característica ou propriedade dos dados de treinamento (por exemplo, presença de periodicidade ou de ruído). É necessário que as propriedades testadas continuem respeitando os padrões e restrições dos domínios. Os testes abaixo são alguns exemplos desta classe.

Teste de Relações Lineares

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento que possuem relações lineares.

Implementação: Há várias implementações possíveis, as mais simples são gerar dados no qual a saída é totalmente dependente de uma determinada amostra, ou no qual a saída é uma combinação linear de algumas amostras.

Teste de Relações Não Lineares

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento que possuem relações não lineares.

Implementação: Implementação varia conforme problema, dados brutos e padrões utilizados. No contexto de previsão de tráfego, a implementação utilizada foi fazer y_i ser igual a amostra x_{i+j} da série temporal, de modo que j é incrementado em 1 após um certo número de amostras.

Teste de Alta Dimensionalidade

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento de alta dimensionalidade.

Implementação: Gerar conjuntos de dados com base em funções que não sejam dependentes da dimensionalidade, por exemplo, distribuições de probabilidade.

Teste de Presença de Outliers

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento que possuem outliers.

Implementação: Com base em um conjunto de dados não normalizado, modificar alguns valores segundo uma probabilidade z por valores de outliers. Em seguida normaliza-se o conjunto de dados e compara-se os resultados obtidos com dados sem outliers e com outliers de diferentes valores.

Teste de Não Normalização

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento não normalizados.

Implementação: Normalizar os conjuntos de dados e comparar os resultados com os conjuntos de dados não normalizados.

Teste de Simetria Negativa

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento negativos porém simétricos a outro conjunto de dados.

Implementação: om base em alguns conjuntos de dados não normalizados, negativar todos os valores e comparar os resultados com os respectivos conjuntos de dados originais.

Teste de Identificação de Padrões

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar como os modelos reagem a dados de treinamento (não normalizados) com mesmo padrão porém apresentando alguma transformação como de translação.

Implementação: No contexto de previsão de tráfego, utilizou-se 2 séries temporais de dados senoidais. A primeira contém somente valores não negativos gerados pela função $f(x, y) = 100sin(\frac{x+y}{M}) + 100$, enquanto a segunda série é gerada pela fórmula $g(x, y) = 100sin(\frac{x+y}{M})$ (ou seja, contém valores positivos e negativos, porém o padrão dos dados ainda é uma senoide).

Teste de Valores Inteiros ou Arredondados

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar como os modelos reagem ao tentar aprender os padrões de dados de treinamento advindos de funções inteiras ou do arredondamento da saída de funções reais.

Implementação: No contexto de previsão de tráfego, com base em funções que possuem imagem nos valores reais, gerar duas séries temporais, uma com os valores originais da função e outra arredondando os valores resultantes da função e comparar o resultado do treinamento dos modelos.

Teste de Presença de Periodicidade

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento com periodicidade.

Implementação: Gerar dados advindos de funções senoidais e de exponenciais complexas e comparar com os dados sem periodicidade.

Teste de Ruído

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento com ruído, e determinar a partir de qual nível de ruído os resultados são comprometidos.

Implementação: Adicionar valores de ruído aos dados reais não normalizados no intervalo de 0 à alguma potência de 10. Após adicionar o ruído, normaliza-se os dados.

Teste de Interpolação

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento interpolados, e determinar a partir de qual nível de dados faltantes os resultados são comprometidos.

Implementação: Remover de modo aleatório uma porcentagem dos dados da série temporal, os quais serão interpolados.

4.1.2 Testes de Propriedades Estatísticas

Essa classe é composta por testes que exploram os dados. Mais especificamente, o tipo de atributo de interesse destes testes são propriedades estatísticas (por exemplo, média ou variância) dos dados de treinamento. Também é necessário que as propriedades testadas

continuem respeitando os padrões e restrições dos domínios. Comenta-se que esta classe é um complemento da classe de propriedade dos dados e abaixo estão alguns exemplos de testes pertencentes a esta classe.

Teste de Multicolinearidade

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar a influência nos modelos de dados de treinamento com multicolinearidade.

Implementação: No contexto do problema de previsão de tráfego, gerar uma série temporal no qual todas as amostras são linearmente dependentes de uma amostra base.

Teste de Influência do Histograma

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar se a forma do histograma dos dados de treinamento influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na forma do histograma e comparar os resultados, a fim de procurar algum padrão.

Teste de Assimetria

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar se a presença de assimetria dos dados de treinamento influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base no coeficiente de assimetria, na média, mediana e na assimetria visualizada no histograma e comparar os resultados, a fim de procurar algum padrão.

Teste de Dispersão

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de requisitos dos dados.

Descrição: Testar se a presença de dados de treinamento dispersos ou concentrados influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base no desvio padrão, no coeficiente de variação e na dispersão ou concentração observada no histograma, a fim de procurar algum padrão.

4.1.3 Testes de Variáveis do Problema

Esta classe agrupa testes que atuam sobre o problema, e que avaliam a influência de alguma variável do problema. É esperado que os padrões e restrições sejam respeitados. Abaixo estão alguns testes desta classe criados com base no problema de previsão de tráfego.

Teste de Esparsidade Temporal

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos, de importância das variáveis dos dados brutos e de requisitos dos dados.

Descrição: Testar a influência do intervalo de amostragem dos dados no resultado dos modelos.

Implementação: Excluir amostras consecutivas da série temporal de modo a aumentar artificialmente o intervalo de amostragem.

Teste de Embaralhamento das Amostras

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de importância das variáveis dos dados brutos.

Descrição: Testar como embaralhar as amostras da série temporal afeta o desempenho dos modelos.

Implementação: Embaralhar as amostras da série temporal e treinar os modelos com os dados embaralhados.

Teste de Esparsidade Geográfica Aleatória

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos, de importância das variáveis dos dados brutos e de requisitos dos dados.

Descrição: Testar a influência da quantidade de sensores e o impacto do aumento da distância entre sensores no resultado dos modelos.

Implementação: Remover de modo aleatório uma porcentagem dos sensores, alterando também o grafo, de modo a manter todas as conexões originais.

Teste de Esparsidade Geográfica - Subgrafo

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos, de importância das variáveis dos dados brutos e de requisitos dos dados.

Descrição: Testar a influência da quantidade de sensores e o impacto do aumento da distância entre sensores no resultado dos modelos.

Implementação: Selecionar um subgrafo (rua ou avenida por exemplo) e remover sensores consecutivos, de modo a aumentar a distância entre os sensores, removendo também os respectivos dados da série temporal.

Teste de Exclusão Geográfica

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de importância das variáveis dos dados brutos.

Descrição: Testar o impacto de uma subregião no resultado dos modelos.

Implementação: Selecionar e remover completamente um subgrafo (rua ou avenida por exemplo), conectando os vértices de início e fim do subgrafo, e removendo também os respectivos dados da série temporal.

4.1.4 Testes de Variação de Padrões

Classe composta por testes que exploram os padrões, e tem como objetivo avaliar a efetividade dos padrões escolhidos. A ideia com essa classe é violar os padrões utilizados, porém mantendo as restrições dos domínios. Com base nos padrões do capítulo 2, formulou-se os seguintes testes para o problema de previsão de tráfego.

Teste de Divisão dos Dados

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de efetividade dos padrões.

Descrição: Testar como o tipo de divisão dos dados de treinamento (sequencial, aleatória, ou com o método de validação cruzada) afeta o desempenho dos modelos.

Implementação: Testar cada tipo de divisão de dados de treinamento com um conjunto de dados representativo para o problema.

Teste de Variação de Temporalidade

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de efetividade dos padrões.

Descrição: Testar como excluir ou manter os dados de curto, médio e longo prazo na formação dos dados de treinamento afeta o desempenho dos modelos.

Implementação: Testar todos os casos possíveis excluindo e mantendo os dados de curto, médio e longo prazo ao concatená-los para gerar os dados de treinamento.

4.1.5 Testes de Restrições de Domínio

Esta classe agrupa os testes cujo atributo explorado são as restrições dos domínios dos elementos, de modo que o objetivo é avaliar o impacto e efetividade das restrições existentes. Dado que todos os quatro elementos fundamentais podem ter restrições em seus domínios, essa classe pode conter testes que age em qualquer um destes elementos, porém como cada teste explora um único elemento, essa classe é válida, pois pode ser vista como a união de conjuntos disjuntos.

A ideia destes testes é violar de forma intencional as restrições, porém mantendo os padrões estabelecidos. Para o problema de previsão de tráfego há o seguinte teste desta classe.

Teste de Sensibilidade Geográfica

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos, de requisitos dos dados e de efetividade das restrições.

Descrição: Este teste consiste em trocar o grafo e a matriz de transição, mantendo a série temporal. Neste teste é permitido trocar o grafo por outros que não são fortemente conexos.

Implementação: Substituir o grafo original por outros com mesmo número de vértices.

4.1.6 Testes de Extrapolação de Treinamento

Os testes que compõem esta classe exploram o modelo e o tipo de atributo de interesse é a capacidade de extrapolação. Deste modo, o objetivo dos testes desta classe é avaliar como os modelos se comportam ao extrapolar o conhecimento de diferentes propriedades. As propriedades podem ter relação com os dados ou com as variáveis do problema. É esperado que os padrões e as restrições sejam respeitadas. domínio do problema ou com os dados. Os primeiros 2 testes abaixo são específicos para o problema de previsão de tráfego enquanto os demais podem ser aplicados para outros problemas.

Teste de Extensão do Conjunto de Teste

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar como os modelos se comportam ao aumentar o período de tempo do conjunto de teste em 1 ano sucessivamente, permitindo identificar se há variação do padrão temporal ao longo dos anos, e se os modelos conseguem extrapolar o conhecimento aprendido para dados novos.

Implementação: Com base em conjuntos de treinamento e validação fixos, o conjunto de teste é incrementado sucessivamente com dados do próximo ano. Utilizou-se apenas 1 sensor dos dados do PEMSd3 a fim de que este teste fosse viável, devido ao grande uso de memória necessária.

Teste de Previsão por Ano

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Comparar os resultados das previsões dos modelos ao gerar conjuntos de teste compostos por dados de um determinado ano.

Implementação: Com base em conjuntos de treinamento e validação fixos, o conjunto de teste é alterado sucessivamente para conter somente dados do próximo ano. Diferentemente do teste anterior que incrementa o conjunto de teste com dados de mais um ano, este teste utiliza apenas um ano de dados por vez no conjunto de teste. Utilizou-se apenas 1 sensor dos dados do PEMSd3 a fim de que este teste fosse viável, devido ao grande uso de memória necessária.

Teste de Conjuntos de Dados de Mesma Classe

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Com base no conceito de classe de dados (seção 5.2), diferentes conjuntos de dados sintéticos são agrupados a fim de verificar se os modelos performam de forma semelhante em todos eles, indicando que conseguem aprender o padrão comum a todos os dados da classe.

Implementação: elecionar alguma característica de interesse dos dados (por exemplo, serem gerados por uma distribuição exponencial) e agrupá-los em uma classe. Em seguida averiguar se os treinamentos apresentam valores de erro e métricas semelhantes.

Teste de Extrapolação de Treinamento para Mesma Classe

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar como os modelos performam ao treiná-los com um conjunto de dados e prever os resultados nos demais conjuntos de dados de uma classe no qual essse conjunto está incluso.

Implementação: A primeira etapa é selecionar uma classe de dados, em seguida, para cada conjunto de dados treinam-se os modelos e avalia-se a performance, tanto do conjunto de teste deste dado quanto dos conjuntos de testes dos demais dados da classe, a fim de comparar se o treinamento aprende corretamente o padrão dos dados desta classe.

Teste de Extrapolação de Treinamento Para Outras Classes

Tipos de Dados Utilizados: Dados sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar como os modelos performam ao treiná-los com um conjunto de dados e prever os resultados aos conjuntos de dados de outras classes. A ideia é verificar se o ajuste dos parâmetros do modelo gera um ajuste único ao padrão de treinamento, ou se o ajuste dos parâmetros permite uma margem para prever corretamente dados de outras classes.

Implementação: A primeira etapa é selecionar uma classe de dados, em seguida, para cada conjunto de dados treinam-se os modelos e avalia-se a performance tanto do conjunto de teste deste conjunto de dados quanto dos conjuntos de testes dos demais conjuntos de dados das outras classes.

4.1.7 Testes de Custo de Treinamento

Essa classe agrupa testes que examinam o modelo, e cujo tipo de atributo de interesse é o desempenho. São de particular interesse quando a performance é um fator determinante ou quando modelos distintos apresentam resultados semelhantes, auxiliando na escolha do modelo mais adequado. Os testes formulados abaixo não modificam os dados, porém, mesmo que modificassem, é esperado que os padrões e restrições sejam respeitados.

Teste de Tempo de Treinamento

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar a performance dos modelos considerando o tempo total de treinamento e o tempo por iteração.

Implementação: Realizar diversos treinamentos com cada modelo e determinar uma média de tempo total de treinamento e tempo por iteração (considerando uma quantidade de dados de treinamento fixa).

Teste de Convergência

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar a performance dos modelos com base na quantidade de iterações necessárias para atingir a convergência.

Implementação: Realizar diversos treinamentos com cada modelo e determinar uma média de convergência de cada modelo com base na curva de aprendizado (considerando uma quantidade de dados de treinamento fixa).

Teste de Memória

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos.

Descrição: Testar a eficiência dos modelos em relação ao uso de memória.

Implementação: Reservar o sistema para executar somente o programa de cada modelo, e tentar treinar cada modelo com a maior quantidade de dados possível.

Teste de Quantidade de Dados

Tipos de Dados Utilizados: Dados reais.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de efetividade dos padrões.

Descrição: Testar como a quantidade de dados de treinamento disponíveis afeta o desempenho dos modelos, permitindo identificar como os modelos se comportam com poucos dados ou qual a quantidade mínima necessária para que o desempenho não seja afetado.

Implementação: A partir do conjunto de treinamento original, remove-se uma porcentagem dos dados de treinamento do final da lista de dados e treinam-se os modelos com esse conjunto reduzido.

4.1.8 Testes de Resíduos

Essa classe é composta por testes que examinam os resíduos e cujo objetivo é avaliar o impacto e influência de alguma alguma propriedade contida neste elemento. As propriedades avaliadas são essencialmente propriedades estatísticas pois não é possível ter controle de características dos resíduos. Estes testes continuarão supondo que os padrões impostos e as restrições dos domínios continuarão sendo respeitados, e também que os hiperparâmetros do aprendizado serão os mesmos utilizados nos demais testes.

Comenta-se também que essa classe é exclusiva para problemas de regressão, e que para problemas de classificação é necessário a definição de uma nova classe e de testes que explorem os valores previstos seguindo outra abordagem.

Teste de *Outliers* dos Resíduos

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos resíduos.

Descrição: Testar como a presença ou ausência de *outliers* nos resíduos influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na ocorrência ou não de *outliers*, a fim de procurar algum padrão.

Teste de Normalidade dos Resíduos

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos resíduos.

Descrição: Testar como a presença de resíduos que não geram uma distribuição normal influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na ocorrência ou não de resíduos com distribuição normal, a fim de procurar algum padrão.

Teste de Autocorrelação dos Resíduos

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos resíduos.

Descrição: Testar como a presença de resíduos autocorrelacionados influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na ocorrência ou não de resíduos com autocorrelação, a fim de procurar algum padrão.

Teste de Exogeneidade dos Resíduos

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos resíduos.

Descrição: Testar como a presença ou ausência de exogeneidade nos resíduos influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na ocorrência ou não de exogeneidade, a fim de procurar algum padrão.

Teste de Homocedasticidade dos Resíduos

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos resíduos.

Descrição: Testar como a presença ou ausência de homocedasticidade nos resíduos influencia nos modelos.

Implementação: Agrupar os conjuntos de dados em classes com base na ocorrência ou não de homocedasticidade, a fim de procurar algum padrão.

4.1.9 Testes de Parâmetros

Essa classe é composta por testes que exploram os pesos resultantes do treinamento do modelo utilizado, e cujo objetivo é avaliar alguma propriedade deste elemento, bem como a interpretabilidade dos valores. Estes testes continuarão supondo que os padrões impostos e as restrições dos domínios continuarão sendo respeitados, e também que os hiperparâmetros do aprendizado serão os mesmos utilizados nos demais testes.

Teste de Estabilidade dos Parâmetros

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos parâmetros.

Descrição: Testar como os parâmetros variam ao comparar o resultado de vários treinamentos com os mesmos dados.

Implementação: Repetir o treinamento com os mesmos dados e medir a variabilidade dos parâmetros. Por exemplo, calcular pesos médios para cada camada do modelo e medir a distância de cada peso para o peso médio.

Teste de Variabilidade dos Parâmetros por Classe

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos parâmetros.

Descrição: Testar como os parâmetros de cada camada do modelo variam entre treinamentos com dados pertencentes a uma mesma classe. O objetivo é verificar se o modelo é sensível a pequenas variações de padrões.

Implementação: Dado uma camada de interesse do modelo, medir a diferença dos parâmetros obtidos (utilizando alguma métrica) para diferentes conjuntos de dados de uma classe.

Teste de Variabilidade dos Parâmetros entre Classes

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos parâmetros.

Descrição: Testar como os parâmetros de cada camada do modelo variam entre treinamentos com dados pertencentes a classes diferentes.

Implementação: Dado uma camada de interesse do modelo, medir a diferença dos parâmetros obtidos (utilizando alguma métrica) para conjuntos de dados de classes distintas.

Teste de Substituição dos Parâmetros por Classe

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos parâmetros.

Descrição: Testar como os resultados são afetados ao treinar o modelo com um conjunto de dados pertence a uma classe e alterar os parâmetros de uma determinada camada com parâmetros obtidos com outro conjunto de dados da mesma classe, e utilizar o modelo modificado para verificar os resultados obtidos no conjunto de teste do primeiro conjunto de dados.

Implementação: Treinar o modelo com um conjunto de dados pertence a uma classe e alterar os parâmetros de uma determinada camada com parâmetros obtidos com outro conjunto de dados da mesma classe, e utilizar o modelo modificado para verificar os resultados obtidos no conjunto de teste do primeiro conjunto de dados. Após isto, comparar os valores de erro e métricas obtidos pelo modelo original e pelo modelo modificado.

Teste de Substituição dos Parâmetros entre Classes

Tipos de Dados Utilizados: Dados reais e sintéticos.

Tipos de Informações Reveladas: Informações de comportamento dos modelos e de interpretabilidade dos parâmetros.

Descrição: Testar como os resultados são afetados ao treinar o modelo com um conjunto de dados pertence a uma classe e alterar os parâmetros de uma determinada camada com parâmetros obtidos com um conjunto de dados de outra classe, e utilizar o modelo modificado para verificar os resultados obtidos no conjunto de teste do primeiro conjunto de dados.

Implementação: Treinar o modelo com um conjunto de dados pertence a uma classe e alterar os parâmetros de uma determinada camada com parâmetros obtidos com um conjunto de dados de outra classe, e utilizar o modelo modificado para verificar os resultados obtidos no conjunto de teste do primeiro conjunto de dados. Após isto, comparar os valores de erro e métricas obtidos pelo modelo original e pelo modelo modificado.

4.2 Verificação de Propriedades Estatísticas

Conforme mencionado, a classe de propriedades estatísticas e a classe de testes dos resíduos agrupam testes que avaliam alguma propriedade estatísticas, seja dos dados ou dos resíduos. No entanto, devido à natureza deste tipo de propriedade não existem métodos

exatos para verificar a ocorrência ou métodos exatos para a inclusão dessas propriedades nos dados sintéticos, deste modo, a seguir estão descritos procedimentos que podem ser utilizadas para a verificação das propriedades citadas nos testes (Morettin, 2021).

Multicolinearidade

A presença ou ausência de correlação entre duas variáveis X e Y pode ser verificada calculando o coeficiente de correlação de Pearson, o qual é definido pela fórmula abaixo, onde x_i e y_i são respectivamente valores de amostras de X e Y, e $\bar{x}$ e $\bar{y}$ são as médias dessas variáveis.

$$\rho = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \tag{4.1}$$

Os valores desse coeficiente variam entre -1 e 1 e podem ser interpretados da seguinte forma:

- $\rho = 1$ indica uma correlação perfeita positiva, ou seja, X cresce, se e somente se Y cresce, e X diminui, se e somente se Y diminui.
- $\rho = -1$ indica uma correlação perfeita negativa, ou seja, X cresce, se e somente se Y diminui, e X diminui, se e somente se Y cresce.
- $\rho = 0$ indica que X e Y são linearmente independentes.
- $0.7 < |\rho| \leq 1$ indica uma correlação forte entre as variáveis.
- $0.5 < |\rho| \leq 0.7$ indica uma correlação moderada entre as variáveis.
- $0.3 < |\rho| \leq 0.5$ indica uma correlação fraca entre as variáveis.
- $|\rho| \leq 0.3$ indica uma correlação muito fraca ou inexistente entre as variáveis.

No caso dos dados sintéticos gerados, a ausência ou presença de multicolinearidade foi introduzida entre as amostras da série temporal, e portanto para verificar essa propriedade, calculou-se o coeficiente de correlação entre todas as amostras, e exibiu-se os valores do coeficiente em um gráfico.

Normalidade

Um método para verificar a normalidade é por meio de um gráfico *QQ*, o qual compara os quantis de uma distribuição teórica (no caso a distribuição normal) com os quantis da distribuição empírica. A distribuição teórica é representada por uma linha reta diagonal no gráfico, e portanto quanto mais perto a distribuição empírica estiver desta linha mais próximo estará da distribuição normal.

Erros Dependentes e Independentes

Erros dependentes ou independentes podem ser constatados verificando a presença ou ausência de autocorrelação dos resíduos. Um procedimento genérico para verificar a autocorrelação é gerar um gráfico de dispersão no qual o eixo X indica o índice do resíduo e

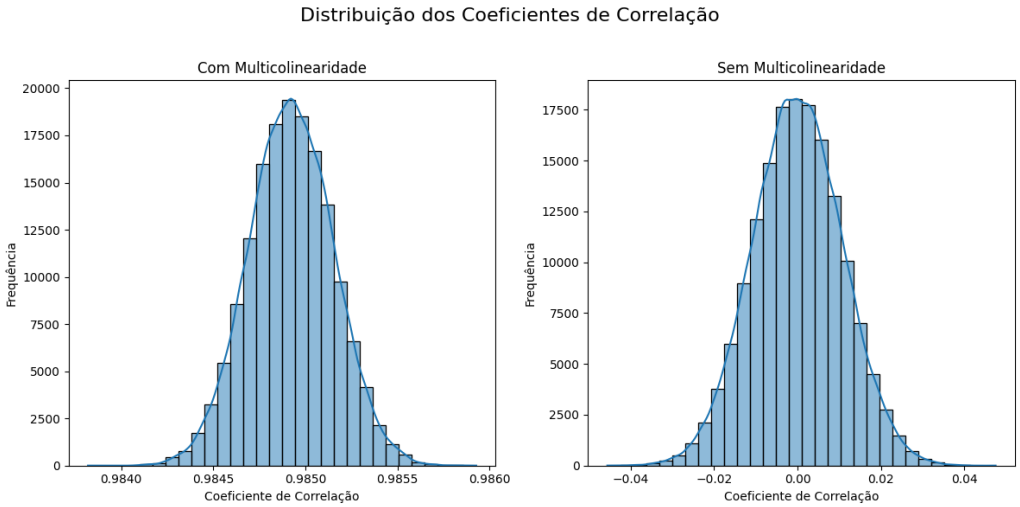


Figura 4.1: *Gráficos do coeficiente de correlação obtidos para os dados sintéticos com multicolinearidade e sem multicolinearidade.*

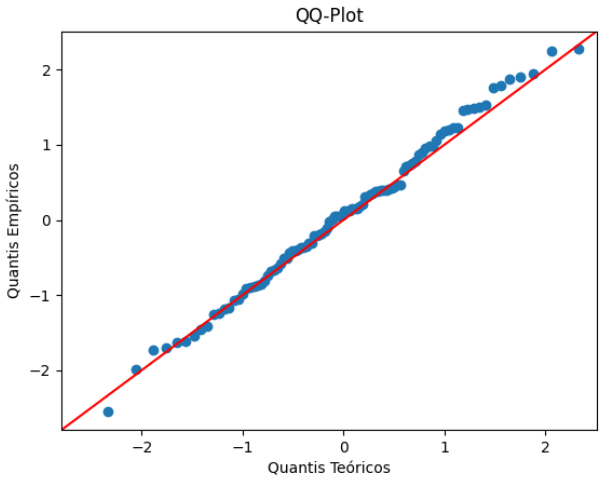


Figura 4.2: *Gráfico QQ exemplificando um caso em que a distribuição empírica é semelhante à distribuição normal.*

o eixo Y contém o valor do resíduo, caso os resíduos não sejam apenas um valor em $\mathbb{R}$, uma alternativa é calcular a norma do vetor. Não há autocorrelação quando não há nenhum padrão no gráfico, de modo que os resíduos se distribuem de modo aproximadamente aleatório, porém se houver um padrão aparente como uma linha reta, isto pode ser um indicativo da presença de autocorrelação. Um exemplo deste gráfico de dispersão pode ser visto na figura 4.4.

Heterocedasticidade e Homocedasticidade

A presença de heterocedasticidade, no contexto de resíduos de um modelo, pode ser verificada de modo genérico gerando um gráfico dos valores dos resíduos em relação aos valores previstos pelo modelo. Caso os valores ajustados tenham várias dimensões, o ideal

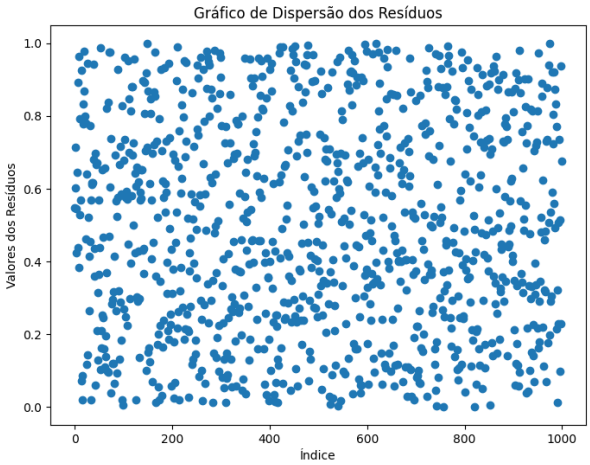


Figura 4.3: *Exemplo de gráfico de dispersão de resíduos no qual não há autocorrelação.*

é gerar um gráfico deste tipo para cada dimensão, no entanto se o número de dimensões for muito alta, uma opção é calcular as normas dos vetores dos resíduos e dos valores previstos. Um exemplo deste gráfico pode ser visto na figura 4.5.

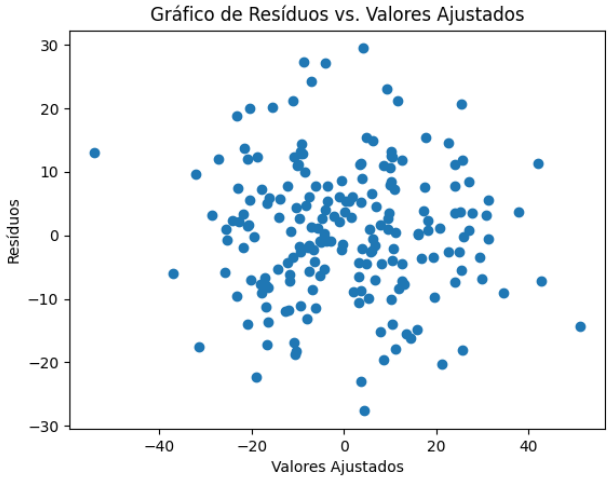


Figura 4.4: *Gráfico dos valores dos resíduos em relação aos valores previstos. Pode-se concluir que os resíduos apresentam homocedasticidade pois não há padrão aparente no gráfico.*

Este tipo de gráfico pode ser interpretado da seguinte forma:

- **Distribuição dos Resíduos:** Idealmente os resíduos devem estar aleatoriamente distribuídos no entorno do valor 0, isso pode indicar que o modelo está bem ajustado. Resíduos distribuídos em forma de funil ou reta é um indicativo de heterocedasticidade e padrões em forma de curva ou outros podem indicar que o modelo não está capturando alguma relação não linear dos dados.
- **Outliers:** Se houver muitos resíduos com valores distantes de 0, isto é um indicativo que *outliers* podem estão afetando o ajuste do modelo, requerendo mais testes para

validação.

4.3 Arcabouço de Teste

Unindo e resumindo todos os pontos anteriores, obtemos o seguinte arcabouço de teste. A resolução de problemas por meio de aprendizado de máquina possui 4 elementos fundamentais: os dados brutos, os padrões definidos, um modelo e os dados de treinamento. Com base em um problema e um modelo de interesse, selecionam-se modelos de referência segundo uma determinada intenção ou objetivo para comparações. Após definir os padrões que serão utilizados e escolher dados disponíveis que se adequem a esses padrões, realiza-se uma análise para determinar as variáveis dos dados brutos e os domínios dos elementos fundamentais. A partir dessa análise, geram-se os testes das classes de variáveis dos dados brutos e de restrições dos domínios. Parte dos testes de propriedades dos dados e propriedades estatísticas decorre de uma análise de características contidas nos dados reais e dos modelos (por exemplo, o tipo de operação que o modelo realiza), e a classe de variação de padrões decorre diretamente da escolha de padrões para os dados de treinamento. As classes de testes de extrapolação de treinamento e testes de custos de treinamento exploram diretamente o modelo, avaliam atributos dele (porém ainda sendo capaz de revelar informações dos outros elementos fundamentais no processo). Além dos elementos fundamentais, há 2 elementos derivados, os parâmetros e os resíduos, os quais possuem duas classes de testes próprias.

Nota-se que cada elemento obrigatório e derivado está associado a ao menos uma classe de testes, além disso, cada teste revela informações de ao menos um desses elementos. Dado que os testes são realizados por meio do treinamento de um modelo, eles sempre revelarão informações do comportamento do modelo. Dessa forma, o arcabouço guia a formulação e realização dos testes, e é completo, pois cada elemento obrigatório possui uma classe correspondente, e cada teste (independente da classe) tem o potencial de revelar informações que extrapolam o elemento avaliado. Esquematicamente o arcabouço pode ser representado pelo diagrama abaixo.

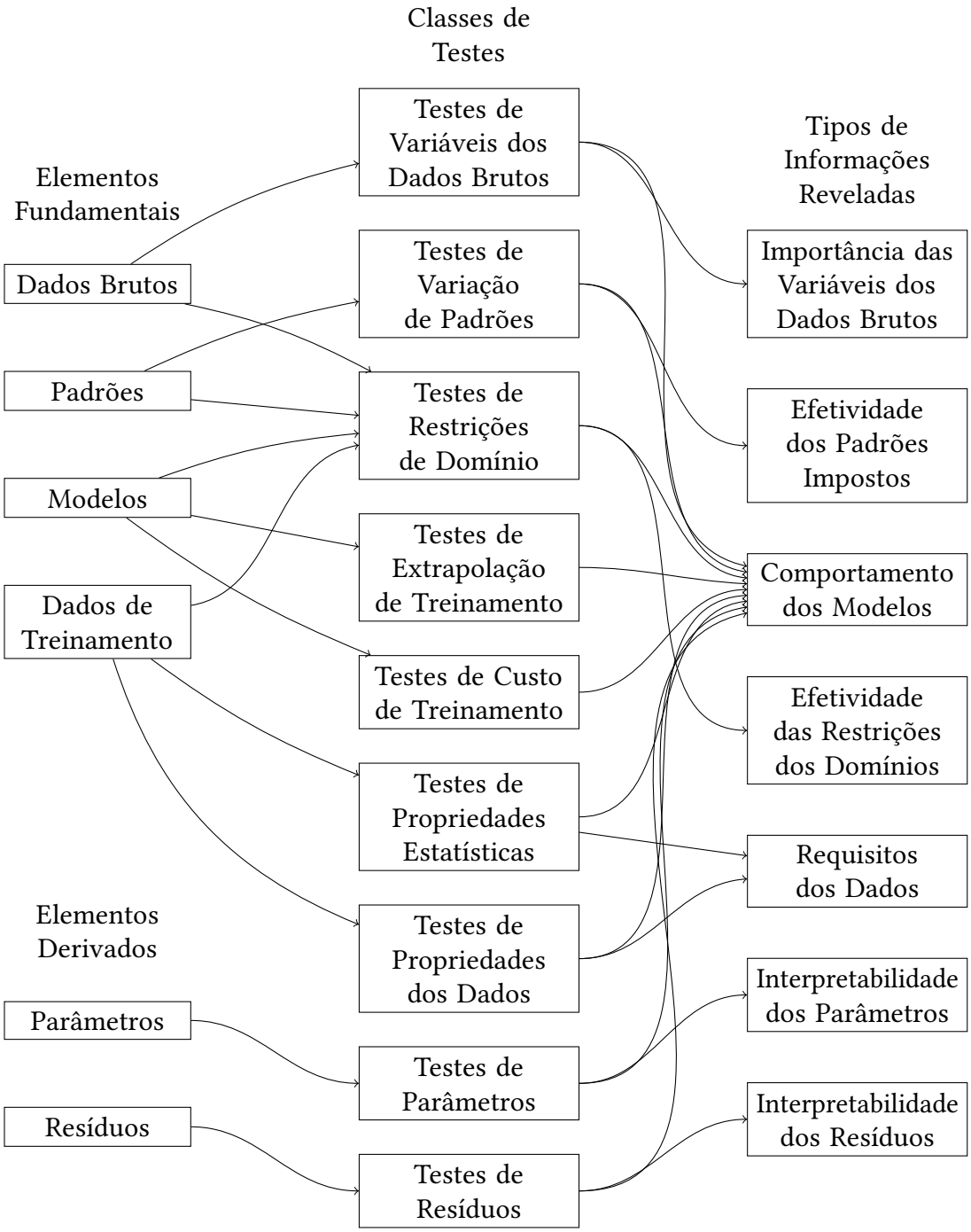


Figura 4.5: *Representação do arcabouço de testes. As classes de testes que possuem aresta de algum elemento fundamental ou derivado indicam que existem (ou que é possível existir) testes desta classe que agem/exploram esses elementos ou algum de seus atributos. As arestas entre as classes de testes e os tipos de informação relacionam as principais informações obtidas pela classe. Comenta-se que as classes podem revelar mais informações além das indicadas pelas arestas, porém omitiu-se estas relações secundárias para não sobrecarregar o diagrama.*

Parte III

Teste dos Modelos

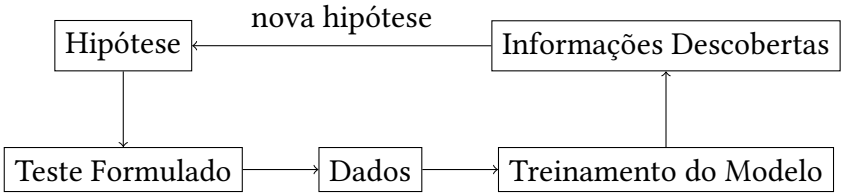
Capítulo 5

Dados Sintéticos[1]

A ideia do uso de dados sintéticos como ferramenta de teste é possibilitar a execução de testes que necessitem de dados com características e propriedades não encontradas nos dados reais, ou seja, principalmente os testes de propriedades dos dados e estatísticas.

Além disso, os dados sintéticos podem contribuir na interpretação dos resultados obtidos com os dados reais, tendo em vista que algumas características dos dados podem ser determinantes para o desempenho de um modelo, e a comparação com dados sintéticos que possuem essa característica pode explicitar essa relação. Por exemplo, se tanto os dados reais quanto sintéticos apresentam periodicidade e desempenham mal em um determinado modelo, uma hipótese provável é que o modelo não seja adequado para esse tipo de dado, e, portanto, a escolha de outro modelo pode ser a solução.

Dessa forma, os dados sintéticos, em conjunto com os diferentes tipos de testes constituem um método completo de estudo e aprofundamento do funcionamento dos modelos, na medida em que os testes guiam quais dados devem ser gerados, e os dados, através do treinamento, revelam informações acerca do comportamento e funcionamento dos modelos, induzindo a novas hipóteses a serem verificadas por meio de mais testes. Esse ciclo pode ser representado pelo diagrama abaixo:



Cita-se também que, a fim de facilitar a análise dos resultados obtidos e direcionar a geração de novos dados, foram desenvolvidas as seguintes classificações para os dados sintéticos.

[1] Referência do capítulo: Jordon *et al.*, 2022

5.1 Dados de Referência

Ao executar alguns testes, notou-se que o treinamento dos modelos com diversos dados, sejam eles modificações dos dados reais ou dados puramente sintéticos, conduzia as métricas a alguns valores observados repetidamente. Comparando os dados que obtinham os mesmos valores de erro e métricas, constatou-se que esses dados compartilhavam uma propriedade comum ou eram gerados a partir de funções matemáticas fundamentais (como distribuições de probabilidade e funções trigonométricas). Sabendo disso, o conjunto de dados gerado diretamente da função fundamental ou que seja um representante extremo dessa propriedade (por exemplo, se a propriedade é multicolinearidade dos sensores, então um representante extremo possui valores de correlação entre todos os sensores próximos de 1) assume o posto de referência para a compreensão desses valores. Alguns desses conjuntos de dados são:

- Dados de distribuição uniforme.
- Dados de distribuição normal padrão.
- Dados de distribuição exponencial (o parâmetro λ da distribuição não afeta o resultado quando os dados estão normalizados).

Uma justificativa para a seleção desses conjuntos de referência é que muitos dados são gerados a partir de dados aleatórios e, nessa situação, há duas possibilidades:

- Utilizam-se distribuições de probabilidade para gerar os dados porém não deseja-se que os dados tenham características aleatórias. Nesse caso, os dados de referência servem como base de quais resultados os dados gerados não deveriam apresentar.
- Utilizam-se distribuições de probabilidade por conveniência para gerar os dados. Por exemplo, para o teste de multicolinearidade, gerou-se um conjunto de dados no qual todas as amostras são linearmente dependentes de uma amostra base aleatória, no entanto a escolha da amostra base ser aleatória ou não é indiferente para o teste, optando-se por essa opção apenas por ser mais fácil de implementar. Nesse caso, os dados de referência também servem como base, mas assumem uma função mais semelhante a um elemento nulo em um espaço vetorial, facilitando a análise dos resultados. Retornando ao exemplo do teste de multicolinearidade, se os resultados para os dados com multicolinearidade forem melhores que os resultados dos dados de referência, é possível concluir que a multicolinearidade torna mais fácil aprender o padrão dos dados. Nota-se que essa comparação e conclusão só são possíveis devido ao papel desempenhado pelos dados de referência.

Ressaltasse que, em ambos os casos, os dados sintéticos possuem um papel comparativo.

Outro ponto de interesse para esses dados é no caso do teste de adição de ruído, pois conhecendo quais valores das métricas correspondem a dados aleatórios, é possível determinar qual nível de ruído prejudica os dados originais de modo que o modelo passe a aprendem somente o ruído.

Em relação aos dados de distribuição uniforme, os valores obtidos com esses dados são especialmente interessantes porque, nos dados de distribuição uniforme, não há nenhuma relação entre diferentes amostras, assim, se um outro conjunto de dados gera métricas

semelhantes, pode ser um indício que não há nenhuma relação entre as amostras ou que não há nenhuma aproximação melhor do que a obtida para os dados puramente aleatórios.

Finalizando esta seção, obteve-se as seguintes métricas nos dados citados (tabelas 5.1, 5.2 e 5.3).

Dados de Distribuição Uniforme					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Regressão Linear	0.0835	0.289	0.579	0.251	601.24
Rede Neural	0.0834	0.289	0.578	0.25	602.85
GNN	0.0837	0.289	0.579	0.25	587.19

Tabela 5.1: *Tabela contendo erro e métricas ao avaliar o conjunto de teste após realizar o treinamento com os dados de distribuição uniforme.*

Dados de Distribuição Normal Padrão					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Regressão Linear	0.00984	0.0991	0.202	0.0792	17.83
Rede Neural	0.00983	0.0991	0.202	0.0791	17.8
GNN	0.00992	0.0995	0.203	0.0794	17.85

Tabela 5.2: *Tabela contendo erro e métricas ao avaliar o conjunto de teste após realizar o treinamento com os dados de distribuição normal padrão.*

Dados de Distribuição Exponencial					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Regressão Linear	0.00387	0.0621	0.998	0.0457	290297.66
Rede Neural	0.00388	0.0621	0.998	0.0458	291251.72
GNN	0.00387	0.062	0.998	0.0456	290511.41

Tabela 5.3: *Tabela contendo erro e métricas ao avaliar o conjunto de teste após realizar o treinamento com os dados de distribuição exponencial (parâmetro $\lambda = 0.005$).*

5.2 Dados de Contraste

Como citado, os dados sintéticos são gerados a fim de conter alguma característica ou propriedade necessária para a execução de um teste. Os testes de propriedades dos dados e estatísticas são bem sucedidos (ou seja, identificam que a propriedade testada é relevante para o desempenho do modelo) quando se observa uma diferença entre os resultados dos modelos ou entre os valores das métricas dos dados sintéticos deste teste e conjunto de dados de referência correspondente. Assim, esse conjunto de dados sintéticos será chamado de dados de contraste.

5.2.1 Adição e Verificação de Propriedades

A geração dos dados sintéticos para testes é condicionada à adição de propriedades e, para garantir a corretude destes dados, também é possível formular testes (no caso, testes

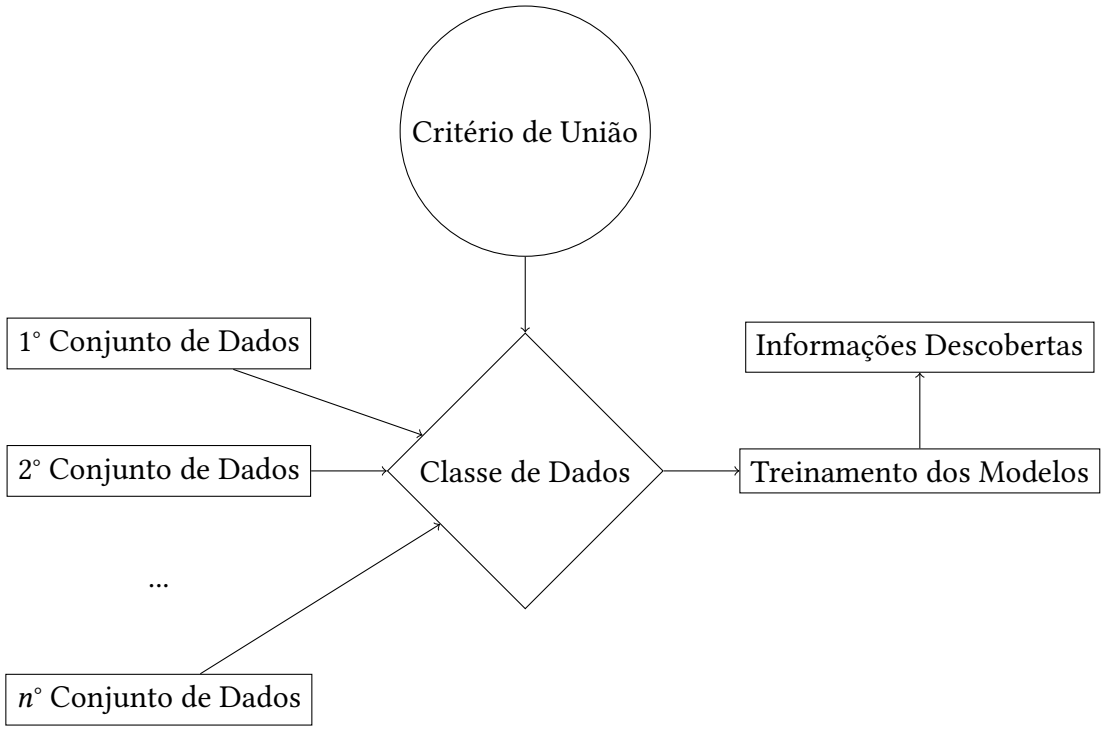
de unidade) para as funções criadas, de modo a garantir o resultado desejado. Além dos testes, pode-se verificar os resultados visualmente; no caso dos testes de normalização, simetria negativa e de valores inteiros pode-se verificar essas propriedades observando os valores dos dados. Em relação aos testes de presença de *outliers* e de simetria negativa também é possível verificar essas propriedades por meio do histograma dos dados.

No caso das relações lineares, essa propriedade pode ser indicada pelo coeficiente de multicolinearidade, no entanto, algumas relações lineares são mais sutis e só ocorrem nos dados de treinamento devido à aplicação dos padrões escolhidos. Desse modo, uma verificação mais concreta é obter valores baixos as métricas para o modelo de regressão linear. Por exemplo, o erro obtido da regressão linear para os conjuntos de dados sintéticos com essa propriedade é menor que 10^{-12}.

Por fim, cita-se que as propriedades estatísticas podem ser verificadas por meio dos métodos descritos na seção 4.2.

5.2.2 Classes de Dados

Além de cada teste de propriedade direcionar a geração dos dados, cada teste também induz uma classe ou grupo composto por dados com essa propriedade. A ideia de classe de dados pode ser utilizada para formular outro processo de teste, no qual dados (sintéticos ou reais) são agrupados em uma classe (seguindo algum critério) e, em seguida, treinam-se os modelos com cada conjunto de dados e comparam-se os comportamentos dos modelos e valores das métricas, a fim de identificar se o critério de união da classe afeta o desempenho dos modelos. Esse processo pode ser representado pelo diagrama abaixo:

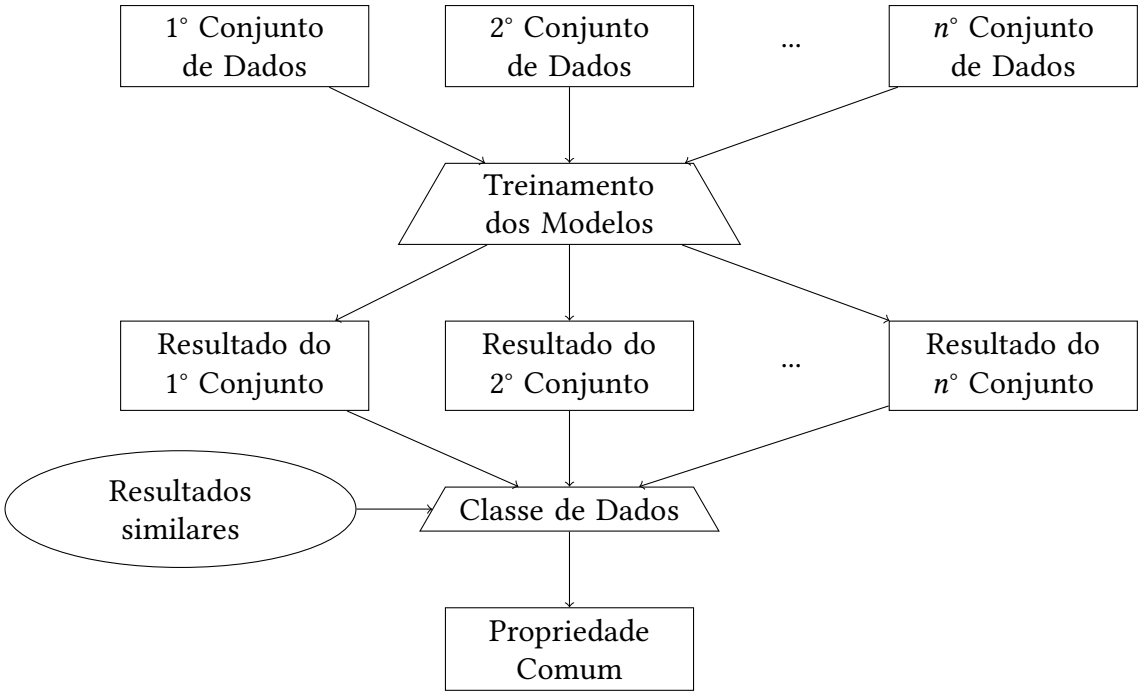


Os critérios para agrupamento dos dados em classes devem se basear em similaridade (dados de distribuição normal, ou imagens em nível de cinza por exemplo) ou em

propriedades (como periodicidade ou presença de relações lineares).

Esse processo de teste pode ser útil e funcionar quando é evidente que os dados da classe atendem claramente ao critério de união. No entanto, se os dados são complexos, possuíndo muitas propriedades, ou se não é claro que os dados se enquadram no critério utilizado, esse processo pode não ser adequado, pois pode conduzir a resultados falsos. No caso dos testes para os elementos derivados, este processo é o único possível, pois, dado que os valores previstos (ou resíduos) e pesos resultam do treinamento dos modelos, controlar os atributos desses elementos por meio dos dados de treinamento é uma tarefa de alta complexidade.

Comentando também sobre um processo formulado que não se mostrou efetivo, este consiste em listar em uma tabela os resultados obtidos nos treinamentos com cada conjunto de dados e, a partir dos resultados (seguindo um critério de igualdade de valores), agrupar os conjuntos de dados e tentar determinar qual a propriedade comum a estes dados que implica nos valores obtidos. É evidente que este processo não é eficaz porque não há nenhum procedimento que auxilie na determinação desta propriedade comum, e mesmo que seja encontrada alguma propriedade, não há garantia que essa correlação seja verdadeira, o que também pode conduzir para resultados falsos. Este processo pode ser representado pelo diagrama abaixo:



Finalizando este tópico, alguns exemplos de classes de dados são: classe de dados com relações lineares, com relações não lineares, com *outliers*, com periodicidade, com histograma normal, com histograma uniforme e com histograma exponencial. Essas classes se derivam dos respectivos testes de propriedades. Em relação à classes de dados obtidas pelo processo de critério de união, pode-se citar como exemplo: classe de dados gerados a partir de distribuição uniforme e classe de dados gerados a partir de imagens em nível de cinza.

Figura 5.1: *Imagens em nível de cinza utilizadas. Constituem uma classe de dados gerados a partir de imagens em nível de cinza.*

Capítulo 6

Aplicação do Arcabouço de Teste

Neste capítulo, o arcabouço de teste formulado é aplicado ao problema de previsão de tráfego, utilizando os modelos e dados mencionados nos capítulos anteriores, e os resultados dos testes são analisados com o objetivo de extrair informações sobre as variáveis dos dados brutos, os padrões, os modelos e os dados de treinamento.

6.1 Comparação de Modelos

Iniciando a análise, ao recapitular a tabela 2.2, a qual exibe os resultados de todos os modelos citados no capítulo 1 com os dados do PEMSd3, é possível observar que os 3 modelos principais (regressão linear, rede neural e GNN) apresentam valores das métricas praticamente idênticos (com exceção da métrica MAPE pelos motivos citados no capítulo 2). Esse resultado é especialmente interessante, pois é comum é esperar que os modelos mais complexos apresentem melhores resultados, ou então que todos os modelos apresentem resultados ruins. No entanto o que se observa é o contrário: os valores das métricas obtidas são bons. Além disso, esse resultado é curioso porque, a princípio, pois os dados do PEMSd3 não possuem relações lineares.

Modelos Únicos por Sensor

Ao comparar os modelos de regressão linear e de rede neural com modelos únicos de regressão linear e de rede neural para cada sensor, nota-se que os modelos simples apresentam um erro cerca de 100 vezes menor e métricas também inferiores, sendo que os dois modelos únicos por sensor tiveram resultados muito similares.

A ideia por trás desses modelos era testar a hipótese de que modelos individuais de regressão linear por sensor seriam mais adequado do que um único modelo de regressão linear para a série temporal inteira, uma vez que, no segundo caso, os pesos são compartilhados entre todos os sensores. No entanto o resultado observado foi o oposto. Esse desempenho inferior dos modelos individuais por sensor pode estar relacionado a um acúmulo de erro ao agrupar as previsões dos sensores para formar a previsão completa.

Decomposição do modelo de GNN e uso de modelo de CNN

Algumas comparações foram realizadas utilizando modificações do modelo de GNN, a fim de averiguar a importância das camadas do modelo. Especificamente, foram gerados 4 modelos modificados:

- 1° modelo modificado: As camadas de convolução de Chebyshev foram substituídas por camadas de convoluções normais, transformando o modelo em uma CNN.
- 2° modelo modificado: Com base no modelo do item anterior, as 4 camadas de convolução foram substituídas por apenas uma camada de convolução, implicando também na remoção da camada de concatenação.
- 3° modelo modificado: Com base no modelo do item anterior, removeu-se a camada aprendível que multiplica a saída da camada de convolução.
- 4° modelo modificado: Com base no modelo do item anterior, removeu-se o mapa de identidade, bem como a matriz de redimensionamento correspondente à camada de soma, resultando em um modelo composto apenas por uma camada de convolução.

Os resultados do treinamento desses 4 modelos modificados com os dados do PEMSd3 podem ser vistos na tabela 6.1. Comparando os resultados com o modelo de GNN original, nota-se que os resultados são similares, porém há uma leve melhora nas métricas à medida em que o modelo é simplificado, até alcançar o nível mais simples, no qual o modelo possui apenas uma camada de convolução. Considerando que o modelo de regressão linear também apresentou resultados semelhantes ao modelo de GNN, não é surpreendente que esses outros modelos também tenham um desempenho similar. Dito isso, não é possível analisar a influência das camadas da GNN, sendo possível concluir apenas que não é necessário um modelo tão complexo para obter bons resultados com estes dados.

Resultados do Conjunto de Teste - Dados do PEMSd3					
Modelo	**Erro**	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	0.000224	0.0143	0.144	0.01	58302.98
1° modelo modificado	0.000229	0.0142	0.132	0.0097	40212.94
2° modelo modificado	0.000225	0.0141	0.128	0.0096	39490.24
3° modelo modificado	0.000225	0.0141	0.129	0.0096	40112.30
4° modelo modificado	0.000222	0.0139	0.127	0.0095	37768.73

Tabela 6.1: *Tabela contendo erro e métricas ao treinar os modelos criados modificando o modelo de GNN com dados do PEMSd3 e avaliar o conjunto de teste.*

Síntese pela Transformada de Fourier Unidimensional

Conforme mostrado no capítulo 2.2, os dados do PEMSd3 possuem características periódicas. Por esse motivo, formulou-se a hipótese que um modelo que se ajuste e aprenda bem essas características periódicas pode obter melhores resultados. Com esse objetivo, selecionou-se esse método de síntese do sinal pela Transformada de Fourier unidimensional, que, apesar de não ser um modelo de aprendizado de máquina, é um método que identifica as componentes periódicas dos dados.

Para cada sensor, utilizou-se somente uma porcentagem dos dados correspodente aos dados de treinamento para sintetizar o sinal, e o restante dos dados foi dividido em conjuntos de validação e teste, para avaliar o desempenho desse método. Observando a tabela 2.2 e comparando com os demais modelos, nota-se que, embora os resultados sejam inferiores aos da regressão linear, da rede neural e da GNN, essa método se mostrou superior às abordagens de previsão por sensor utilizando regressão linear e o modelo de rede neural.

Este resultado revela algumas informações. Primeiramente, as funções ajustadas por este método são mais exatas do que as funções ajustadas pela regressão linear e pela rede neural para cada sensor. Além disso, é interessante observar que, na maioria dos sensores, o padrão do sinal varia mais em relação ao sinal de treinamento quanto mais distante estiver dessa parte do sinal do sensor. No entanto, ainda assim, o resultado obtido é razoável, o que possivelmente indica que os resultados poderiam ser melhores se outras abordagens fossem utilizadas, como prever o sinal em um curto período de tempo e reajustar a função aproximada após um longo período.

Síntese pela Transformada de Fourier Bidimensional

Os resultados obtidos ao utilizar a mesma abordagem do método anterior, porém aproximando o sinal da série temporal inteira por meio da transformada de Fourier bidimensional são piores do que todos os demais modelos. Isso indica que, apesar de haver uma periodicidade global derivada da periodicidade horária e diária do trânsito, esse padrão não pode ser adequadamente aproximado ao se considerar de forma conjunta as componentes de frequência das duas dimensões. Além disso, o padrão global decorre da união de padrões individuais de cada sensor, e, considerando que os padrões dos sensores variam e que a aproximação individual desses padrões com a transformada de Fourier unidimensional não conseguia prever com qualidade as periodicidades após um período de tempo, ao se considerar a série temporal inteira, prever o sinal através do método bidimensional torna-se ainda mais difícil, o que se refletiu nos resultados.

6.2 Testes de Variáveis dos Dados Brutos e de Restrições de Domínio

Conforme citado no capítulo 3, as variáveis dos dados brutos podem impactar os dados brutos, os padrões e os dados de treinamento. Por este motivo, esta classe de teste tem o maior potencial de revelar informações, sendo, portanto, os primeiros a serem analisados.

Adicionalmente, o teste de sensibilidade espacial será analisado em conjunto com os demais testes de variáveis do problema, pois o único motivo desse teste ser classificado como teste de restrição de domínio, e não como teste de variável, deve-se ao fato de permitir que os grafos testados violem as restrições do modelo.

Variáveis Temporais

Começando com o teste de esparsidade temporal, nota-se uma tendência de piora nos erros e métricas com o aumento da esparsidade. Essa piora ocorre da mesma forma nos

3 modelos, e o crescimento de algumas métricas é próximo de um crescimento linear (figura 6.1). Esse resultado é interessante porque, ao mesmo tempo, indica que, para obter melhores resultados, é importante ter dados amostrados com um pequeno intervalo, mas que utilizar um grande intervalo de tempo também não afeta seriamente os resultados.

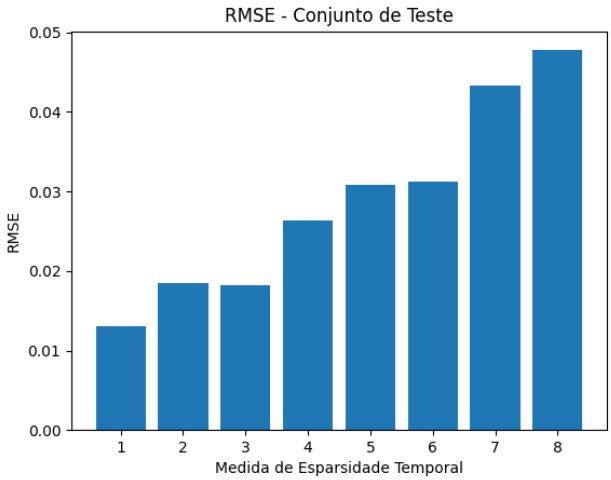


Figura 6.1: *Gráfico exemplificando o comportamento das métricas no teste de esparsidade temporal. O gráfico exibe o comportamento da métrica RMSE com dados do conjunto de teste e com o modelo de GNN para o teste de esparsidade temporal. O comportamento exemplificado pode ser extendido às demais métricas e modelos.*

Além disso, é importante mencionar que aumentar a esparsidade dos dados artificialmente gera uma diminuição exponencial na quantidade de dados disponíveis para treinamento (figura 6.2). No entanto, como será mostrado em outro teste, a diminuição não foi severa o suficiente para ser responsável pelos valores observados.

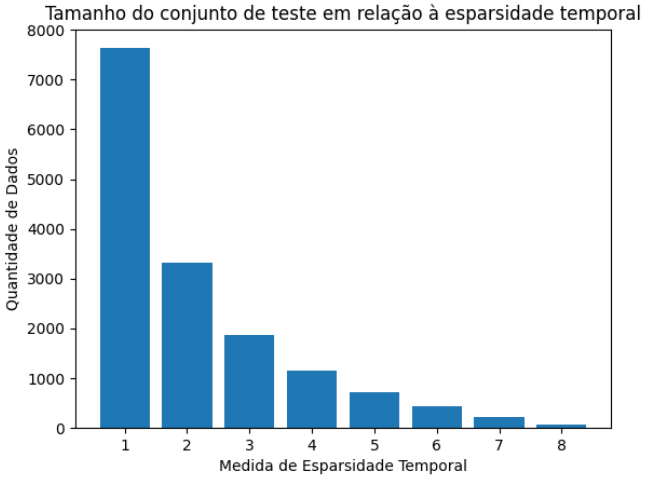


Figura 6.2: *Gráfico mostrando a diminuição do tamanho do conjunto de teste com o aumento da esparsidade temporal. O mesmo comportamento ocorre nos conjunto de treinamento e validação.*

Adicionalmente, o teste de embaralhamento das amostras mostra que a ordem das amostras é importante para obter um melhor resultado, o que possivelmente ocorre pois o

padrão periódico dos dados é perdido, conforme observado nas figuras 6.3 e 6.4. Além disso, os 3 modelos mostraram o mesmo comportamento e resultados, e a piora no desempenho está exemplificada na tabela 6.2 para o modelo de regressão linear.

Figura 6.3: *Imagem em nível de cinza dos valores da série temporal do conjunto de dados PEMSd3 após embaralhamentos das amostras (colunas). Comparando com a figura 3.3 nota-se que o padrão periódico foi perdido.*

Resultados do Conjunto de Teste - Regressão Linear					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Dados originais	0.000207	0.0137	0.135	0.0095	46440.93
Dados com amostras embaralhadas	0.00744	0.0802	1.0859	0.0663	573076.63

Tabela 6.2: *Tabela contendo erro e métricas ao treinar o modelo de regressão linear e avaliar o conjunto de teste após realizar o treinamento com o conjunto de dados do PEMSd3 original, e outro com as amostras embaralhadas.*

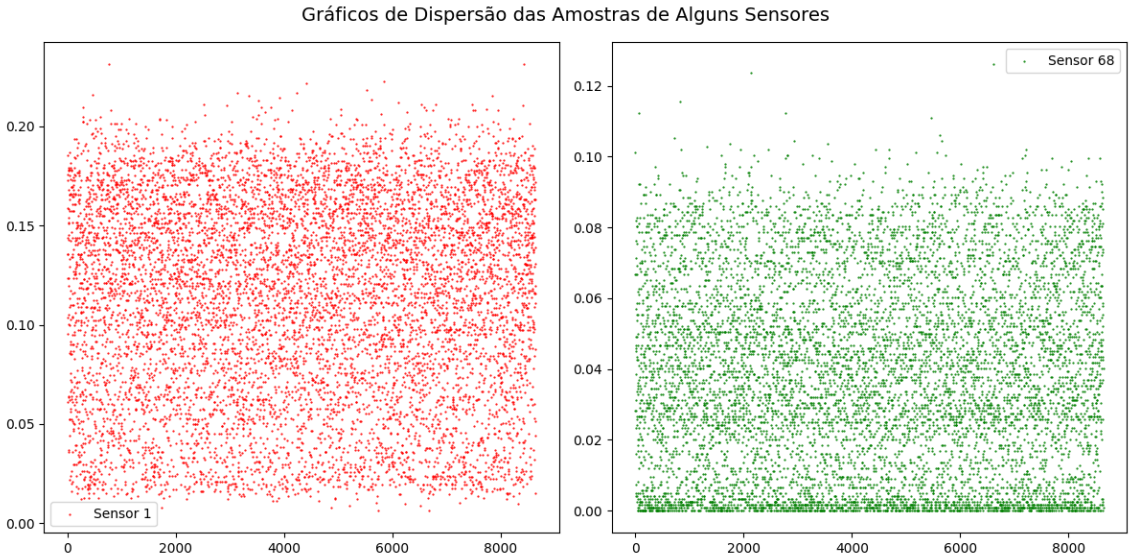


Figura 6.4: *Gráficos de dispersão de alguns sensores do conjunto de dados do PEMSd3 com as amostras embaralhadas, mostrando que a periodicidade foi perdida. O eixo X representa o índice da amostra e o eixo Y representa o valor da amostra.*

Variáveis Espaciais

Seguindo para o teste de esparsidade geográfica aleatória, nota-se que a quantidade de sensores tem pouca influência nos resultados, tornando-se relevante apenas após a remoção de mais de 80% dos sensores. O mesmo resultado é observado nos testes de esparsidade geográfica por trecho e de exclusão geográfica, nos quais o impacto da exclusão de sensores é mínimo. Esses outros dois testes também indicam que possíveis correlações entre sensores

adjacentes ou trechos próximos não exercem influência no resultado, de modo que as pequenas diferenças notadas estão mais relacionadas às séries temporais correspondentes aos sensores mantidos.

Tendo em vista que as características geográficas parecem ter pouca influência nos resultados, especialmente ao notar que os modelos de regressão linear e de rede neural conseguem obter os mesmos resultados da GNN sem incorporar as dependências espaciais por meio do grafo, formulou-se o teste de sensibilidade geográfica, o qual substitui o grafo original por outros com características distintas. Detalhando os grafos utilizados:

- Grafo totalmente desconexo: Não há arestas conectando os vértices.
- Grafo linear: Vértices formam uma linha reta.
- Grafo circular: Vértices se conectam formando um único ciclo.
- 2 Grafos lineares desconexos: Vértices se dividem em duas componentes desconexas em formato de retas.
- 2 Grafos circulares desconexos: Vértices se dividem em duas componentes desconexas em formato de círculos (cada componente possui um único ciclo).
- Malha quadrada fortemente conexa: Vértices se conectam formando uma estrutura de malha quadrada.

O resultado deste teste (figura 6.5) mostra que a topologia do grafo não afeta o resultado do modelo de GNN, o que reforça a percepção anterior de que as características especiais têm pouca influência. Além disso, o resultado mostra que o requisito do grafo ser fortemente conexo não é absolutamente necessário, possivelmente porque a implementação da convolução pela biblioteca do *Python* está adaptada para essa situação.

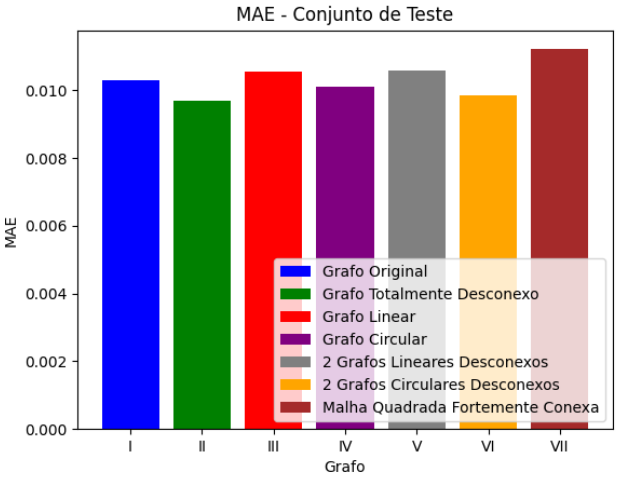


Figura 6.5: *Gráfico exemplificando o resultado do teste de sensibilidade geográfica através da métrica MAE, calculada para o conjunto de teste. Nota-se a pouca variação nos valores com a alteração do grafo.*

6.3 Testes de Variação de Padrões

Divisão dos Dados

O teste de divisão dos dados mostrou que o método escolhido (sequencial, aleatória ou por validação cruzada) não afeta os resultados. No entanto a divisão dos dados pelo método sequencial continua sendo o método mais lógico para o problema, tendo em vista que os dados possuem uma ordem definida pela data das amostras.

Variação de Temporalidade

Os dados de treinamento são gerados concatenando dados de curto, médio e longo prazo. Com isso em mente, a ideia do teste de variação de temporalidade verifica todas as possibilidades de manter ou excluir esses dados na concatenação.

Os resultados desse teste mostram que todos os 3 modelos principais se comportam da mesma forma. Além disso, o mais interessante é que, apenas com dados de curto prazo, ou seja, apenas com as duas amostras anteriores, é possível prever com qualidade o valor da amostra atual (os valores das métricas são semelhantes aos valores do padrão original, que concatena dados de curto, médio e longo prazo), mesmo para o modelo de regressão linear, que neste caso ajustará apenas 3 pesos. Isso é um indicativo que o padrão definido para estes dados não é a melhor escolha, pois utiliza mais valores que o necessário para obter um bom resultado, implicando em um desperdício de memória.

Os resultados completos podem ser vistos na tabela 6.3. Comparando todos os casos, nota-se também que, quando não há dados de curto prazo nos dados de treinamento, os valores das métricas são piores. Isso também se reflete nos pesos do modelo de regressão linear, pois, quando há dados de curto prazo, 50% dos valores dos pesos se concentram apenas nesses dados.

Resultados do Conjunto de Teste - Regressão Linear - Dados do PEMSd3					
Temporalidade dos Dados	**Erro**	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Curto Prazo	0.000226	0.0141	0.134	0.0098	31161.98
Médio Prazo	0.00042	0.0188	0.184	0.013	67112.46
Longo Prazo	0.00044	0.0195	0.193	0.014	87699.34
Curto e Médio Prazo	0.00021	0.0137	0.133	0.0094	39426.02
Curto e Longo Prazo	0.00021	0.0137	0.135	0.00952	44492.30
Médio e Longo Prazo	0.00036	0.0176	0.160	0.012	61005.51
Curto, Médio e Longo Prazo	0.000207	0.0137	0.135	0.0095	46440.93

Tabela 6.3: *Tabela contendo erro e métricas ao treinar o modelo de regressão linear e avaliar o conjunto de teste após realizar o treinamento com dados do conjunto de dados PEMSd3, variando a temporalidade dos dados.*

6.4 Testes de Propriedades dos Dados de Treinamento

Dados Lineares

Iniciando a análise dos testes referentes às propriedades dos dados de treinamento com o teste de relações lineares, observou-se que, quando existe uma relação perfeitamente linear entre os dados de treinamento e os valores esperados, a regressão linear apresenta o melhor resultado com erro de estimativa na ordem de 10^{-14}. A GNN apresentou um erro na ordem de 10^{-10} e a rede neural teve um erro na ordem de 10^{-6}. Embora todos esses resultados sejam aceitáveis, fica evidente que a melhor opção, neste caso, é a regressão linear.

Dados Não Lineares

Analisando o caso oposto com o teste de não linearidade, observou-se que nenhum dos modelos conseguiu aprender a relação não linear existente, aprendendo alguma outra relação contida nos dados. Neste teste, os dados sintéticos foram gerados a partir de uma distribuição uniforme, e, por este motivo, os modelos aprenderam essa relação aleatória em vez de aprender a relação não linear, de modo que os 3 modelos apresentaram os mesmos valores de erro e métricas dos dados de referência de distribuição uniforme.

Dados de Alta Dimensionalidade

Em relação aos dados de alta dimensionalidade, os 3 modelos mostraram-se robustos a essa característica, de modo que, se os padrões dos dados se mantém, então o resultado dos modelos não é afetado. A tabela 6.4 ilustra esse comportamento.

Resultados do Conjunto de Teste - Rede Neural - Dados de Distribuição Uniforme					
Dimensão	**Erro**	**RMSE**	**NRMSE**	**MAE**	**MAPE**
400 × 8640	0.0834	0.289	0.578	0.25	602.85
6000 × 2160	0.0838	0.290	0.581	0.251	1286.30
12000 × 2160	0.0838	0.290	0.579	0.251	657.97

Tabela 6.4: *Tabela contendo erro e métricas ao treinar o modelo de rede neural e avaliar o conjunto de teste após realizar o treinamento com o dados gerados por uma distribuição uniforme.*

Dados Não Normalizados

Os modelos de regressão linear, rede neural e GNN são sensíveis a dados não normalizados, de modo que a identificação dos padrões dos dados é afetada pela magnitude dos valores. Por exemplo, dados gerados a partir de distribuições de probabilidade, quando normalizados, resultam nos mesmos valores de erro e métricas independente dos parâmetros das distribuições. Já ao comparar com os respectivos valores não normalizados, os parâmetros afetam os resultados.

Além disso, o comportamento dos modelos em relação a dados não normalizados variam conforme o conjunto de dados utilizado. Por exemplo, na seção de dados com

periodicidade gerou-se uma matriz senoidal com base na fórmula $M(x, y) = sin(2\pi\frac{x+y}{8640})$, e, quando normalizada, essa série temporal apresentou melhores resultados para a regressão linear e piores resultados para a rede neural. Já ao utilizar os dados não normalizados, o melhor resultado é obtido com a rede neural, e o pior, com a GNN.

Por esses motivos, os testes foram realizados com dados normalizados, a fim de facilitar a comparação dos resultados.

Teste de Identificação de Padrões

Para verificar se os modelos conseguem identificar um mesmo padrão em dados não normalizados, geraram-se os conjuntos de dados abaixo (de dimensão 400×8640), os quais advêm de funções senoidais com magnitudes semelhantes:

- Matriz senoidal positiva: A matriz é gerada pela fórmula $M(x, y) = 10000\,sin(2\pi\frac{x+y}{8640})+10000$
- Matriz senoidal negativa: A matriz é gerada pela fórmula $M(x, y) = -10000\,sin(2\pi\frac{x+y}{8640}) - 10000$
- Matriz senoidal positiva/negativa: A matriz é gerada pela fórmula $M(x, y) = 10000\,sin(2\pi\frac{x+y}{8640})$

Os resultados obtidos para cada uma dessas matrizes está nas tabelas 6.5, 6.6 e 6.7.

Resultados do Conjunto de Teste - Matriz Senoidal Positiva					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	109.77	10.05	0.0249	10.04	2529004.75
Regressão Linear	0.0367	0.175	0.000623	0.172	66330.14
Rede Neural	2575683.5	1584.45	4.27	1582.38	419620480

Tabela 6.5: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da matriz senoidal positiva.*

Resultados do Conjunto de Teste - Matriz Senoidal Negativa					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	7941.87	45.96	0.525	39.74	75341184.0
Regressão Linear	0.0229	0.139	0.000508	0.137	54107.4
Rede Neural	310836.66	480.74	0.82	462.54	48453940.0

Tabela 6.6: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da matriz senoidal negativa.*

Comparando os resultados, nota-se que a regressão linear é o modelo mais robusto em relação à identificação de padrões, pois, nos 3 conjuntos de dados, a diferença entre os valores de erro e métrica foi pequena. Em relação ao modelo de GNN, observa-se que os resultados são melhores quando há somente valores positivos. Ao se comparar com o caso simétrico negativo, há uma piora nos valores, indicando que a identificação do padrão é afetada ao se utilizar valores negativos, e os resultados são ainda mais afetados ao misturar valores positivos e negativos na matriz, como é o caso da matriz senoidal positiva/negativa.

Resultados do Conjunto de Teste - Matriz Senoidal Positiva/Negativa					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	2403016114176	1502293	358.95	1497888.25	325044797440
Regressão Linear	0.121	0.344	8.59×10^{-5}	0.344	84417.72
Rede Neural	177509008	12899.06	2.88	12849.85	2375091456

Tabela 6.7: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da matriz senoidal positiva/negativa.*

Por fim, a rede neural apresenta um comportamento semelhante ao da GNN, possuindo um desempenho pior no caso da matriz senoidal positiva/negativa, e apresentando resultados próximos nos casos positivos e negativos. No entanto, o resultado foi um pouco pior para o caso positivo.

Teste de Simetria Negativa

O impacto de dados negativos pode ser visto, primeiramente, ao analisar o casos das matrizes senoidais definidas anteriormente. Como se observou, a regressão é robusta a dados simetricamente negativos,enquanto a GNN não consegue identificar corretamente o padrão nesta situação, e a rede neural apresenta uma diferença de resultados entre os casos positivo e negativo.

Dados com Valores Inteiros ou Arredondados

O impacto de valores inteiros ou arredondados também foi verificado utilizando funções senoidais, de modo que se compararam conjuntos de dados gerados a partir dos valores reais desta função com outros conjuntos gerados a partir dos valores truncados. Foram utilizadas as seguintes matrizes para comparações:

- 1° Matriz senoidal real: A matriz é gerada pela fórmula $M(x, y) = 1000 sin(2\pi \frac{x+y}{8640}) + 1000$.
- 1° Matriz senoidal inteira: A matriz é gerada truncando a parte decimal dos valores da matriz anterior.
- 2° Matriz senoidal real: A matriz é gerada pela fórmula $M(x, y) = 10000 sin(2\pi \frac{x+y}{8640}) + 10000$. Os resultados dos modelos para esta matriz estão na tabela 6.5.
- 2° Matriz senoidal inteira: A matriz é gerada truncando a parte decimal dos valores da matriz anterior.

Os resultados obtidos para estas matrizes estão contidas nas tabelas 6.8, 6.9 e 6.10.

Comparando primeiramente a 1° matriz senoidal real e inteira, observa-se que os 3 modelos apresentam um resultado pior no caso inteiro; já ao comparar a 2° matriz senoidal real e inteira, os 3 modelos passam a ter um resultado melhor no caso inteiro, e, em especial, a regressão linear consegue ter um resultado ótimo, identificando alguma relação linear nesta situação.

O que é possível concluir a partir deste teste é que os 3 modelos se comportam no mesmo

Resultados do Conjunto de Teste - 1° Matriz Senoidal Real					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	522.31	13.86	0.255	12.84	23634.65
Regressão Linear	0.00194	0.0363	0.000693	0.0358	7183.72
Rede Neural	14349.94	112.88	4.14	112.02	41544920

Tabela 6.8: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 1° matriz senoidal real.*

Resultados do Conjunto de Teste - 1° Matriz Senoidal Inteira					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	5595.05	49.46	0.109	48.38	13077472
Regressão Linear	0.483	0.691	0.0209	0.571	11889256
Rede Neural	30715.21	153.37	7.2	152.42	6619817984

Tabela 6.9: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 1° matriz senoidal inteira.*

Resultados do Conjunto de Teste - 2° Matriz Senoidal Inteira					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	0.0168	0.0786	0.248	0.0771	658.85
Regressão Linear	5.79×10^{-10}	2.41×10^{-5}	0.00141	1.92×10^{-5}	134.87
Rede Neural	0.0337	0.171	13.23	0.169	1871372.5

Tabela 6.10: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 2° matriz senoidal inteira.*

sentido, ou seja, todos os modelos terão um resultado melhor ou pior ao arredondar ou truncar os valores, indicando que essa transformação dificultou ou facilitou a identificação de algum padrão.

Dados com Periodicidade

A fim de verificar o impacto da periodicidade nos resultados do modelo, foram gerados dados sintéticos advindos de funções trigonométricas e de exponenciais complexas. Nesse contexto, dois conjuntos de dados foram gerados a partir de funções senoidais:

- 1° Matriz senoidal: Série temporal de dimensão 400 x 8640 no qual cada valor da matriz é gerado pela fórmula $M(x, y) = sin(2\pi \frac{x+y}{8640})$.
- 2° Matriz senoidal: Série temporal de dimensão 400 x 8640 no qual cada valor da matriz é gerado pela fórmula $M(x, y) = sin(2\pi \frac{x+y}{400 \times 8640})$.

Analisando o resultado do treinamento dos 3 modelos principais com essas matrizes senoidais, notou-se que o resultado foi muito melhor com a regressão linear e pior com a rede neural (conforme pode ser visto nas tabelas 6.11 e 6.12). Esse resultado é interessante, pois a função seno não é uma função linear, mas, devido à aplicação do padrão dos dados de treinamento, a regressão consegue aprender alguma relação linear que faz com que o resultado seja superior com este modelo. No caso da 2° matriz senoidal, é mais intuitivo

identificar a existência de uma relação linear, pois a diferença entre valores consecutivos é pequena, de modo que as linhas da matriz podem conter somente valores crescentes ou decrescentes. No entanto, na 1° matriz, a diferença entre valores consecutivos é maior, e cada linha contém um período completo da função seno, o que torna não intuitivo como a regressão linear consegue encontrar alguma relação linear.

Resultados do Conjunto de Teste - 1° Matriz Senoidal					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	1.48×10^{-6}	0.00122	0.00634	0.00122	0.654
Regressão Linear	2.21×10^{-11}	4.676×10^{-6}	2.374×10^{-5}	4.666×10^{-6}	0.00242
Rede Neural	0.0266	0.158	0.883	0.155	92.21

Tabela 6.11: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 1° matriz senoidal.*

Resultados do Conjunto de Teste - 2° Matriz Senoidal					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	7.38×10^{-11}	8.59×10^{-6}	1.49×10^{-5}	6.8×10^{-6}	0.00118
Regressão Linear	2.42×10^{-12}	1.56×10^{-6}	2.71×10^{-6}	1.55×10^{-6}	0.000271
Rede Neural	5.14×10^{-6}	0.00226	0.00392	0.00226	0.391

Tabela 6.12: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 2° matriz senoidal.*

Além desses dados, outros 3 conjuntos foram gerados a partir da soma de exponenciais complexas, o padrão gerado pelas exponenciais nesses conjuntos pode ser visto na figura 6.6.

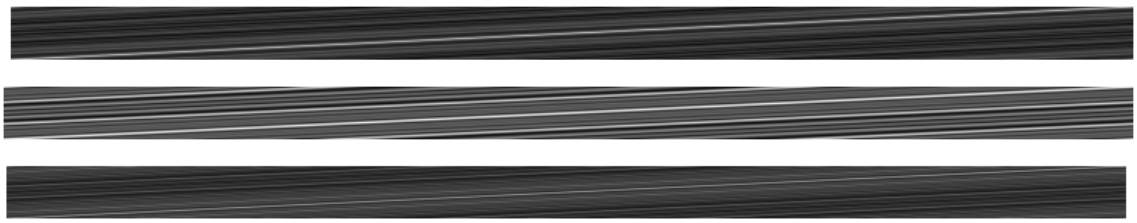


Figura 6.6: *Imagens representando os padrões periódicos das séries temporais geradas por soma de exponenciais complexas.*

Em relação aos resultados obtidos para estes dados, os modelos de GNN e de regressão linear obtiveram resultados semelhantes entre si e apresentaram um erro 10 vezes menor que a rede neural. Esses resultados podem ser vistos nas tabelas 6.13, 6.14 e 6.15.

Além disso, foram gerados conjuntos de dados a partir de imagens em nível de cinza (conforme citado no capítulo 5), os quais não são exatamente periódicos mas que podem ser vistos como formados pela soma de exponenciais complexas. Os 3 modelos apresentaram o mesmo comportamento com esses dados, e os resultados podem ser vistos na tabela 6.16.

Comparando os resultados de todos os conjuntos de dados citados anteriormente e também com os dados do PEMSd3, nota-se que os valores obtidos possuem ordem de

1° Conjunto de Dados de Soma de Exponenciais Complexas					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	8.55×10^{-7}	0.000924	0.00431	0.000715	11.33
Regressão Linear	8.95×10^{-7}	0.000945	0.00441	0.000711	23.47
Rede Neural	3.69×10^{-6}	0.00192	0.00895	0.00138	249.27

Tabela 6.13: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 1° série temporal gerada a partir de soma de exponenciais complexas.*

2° Conjunto de Dados de Soma de Exponenciais Complexas					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	6.31×10^{-7}	0.000794	0.00223	0.000548	88.3
Regressão Linear	3.77×10^{-7}	0.000614	0.00172	0.000422	79.47
Rede Neural	9.28×10^{-6}	0.00305	0.00854	0.00224	387.22

Tabela 6.14: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 2° série temporal gerada a partir de soma de exponenciais complexas.*

3° Conjunto de Dados de Soma de Exponenciais Complexas					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
GNN	2.37×10^{-6}	0.00154	0.00663	0.00107	86.71
Regressão Linear	2.77×10^{-6}	0.00166	0.00716	0.00115	151.54
Rede Neural	2.83×10^{-6}	0.00168	0.00723	0.00129	313.63

Tabela 6.15: *Tabela contendo erro e métricas ao avaliar os 3 modelos com o conjunto de teste da 3° série temporal gerada a partir de soma de exponenciais complexas.*

Resultados do Conjunto de Teste - Dados Gerados a Partir de Imagens em Nível de Cinza - Rede Neural					
Imagem	**Erro**	**RMSE**	**NRMSE**	**MAE**	**MAPE**
1° Imagem	5.47×10^{-6}	0.00231	0.0121	0.00177	44179.54
2° Imagem	4.32×10^{-5}	0.00655	0.0212	0.00533	91428.01
3° Imagem	1.26×10^{-5}	0.00352	0.0109	0.00283	22214.59
4° Imagem	3.19×10^{-5}	0.0056	0.0126	0.00459	52521.33
5° Imagem	1.68×10^{-5}	0.00404	0.00718	0.00298	10524.73
6° Imagem	7.69×10^{-6}	0.00273	0.0059	0.00218	1499.40
7° Imagem	1.31×10^{-5}	0.00357	0.00526	0.00257	0.394

Tabela 6.16: *Tabela contendo erro e métricas ao avaliar o modelo de rede neural com os conjuntos de testes correspondentes aos dados gerados a partir de imagens em nível de cinza.*

grandeza semelhante, principalmente ao comparar com dados não periódicos como dados gerados a partir de distribuições de probabilidade, os quais apresentam valores das métricas maiores. Desse modo, pode-se afirmar que os modelos conseguem se ajustar melhor a dados periódicos ou gerados a partir de exponenciais complexas.

Presença de *Outliers*

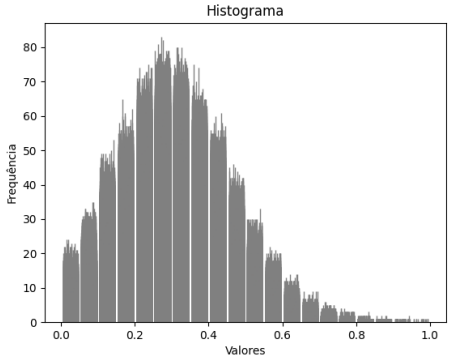
Outra propriedade testada é a presença de *outliers* nos dados. Foram gerados 3 conjuntos de dados sintéticos com diferentes valores de *outliers*, e observou-se nos resultados que todos os modelos se comportam da mesma forma com os dados testados. Para exemplificar os resultados, a tabela 6.17 mostra os resultados obtidos no conjunto de teste após o treinamento do modelo de GNN com esses dados. Como se observa na tabela, aumentar os valores de *outliers* afeta de modo incremental as métricas e o erro. Isso ocorre, pois, como pode ser visto na figura 6.7, os histogramas dos dados sofrem um achatamento à esquerda (resultante da normalização). Dessa forma, os demais valores encontram-se com menor espaçamento, o que pode facilitar a previsão, pois a diferença entre os valores torna-se menor. No entanto, ao tentar prever um valor de *outlier*, o erro será maior.

Resultados do Conjunto de Teste - Modelo de GNN					
	Erro	**RMSE**	**NRMSE**	**MAE**	**MAPE**
Dados de distribuição normal padrão	0.00992	0.0995	0.203	0.0794	17.85
Dados de distribuição normal truncada à direita	0.0192	0.1384	0.459	0.112	158907.81
Dados com *outliers* entre 10 e 20	0.0196	0.139	0.609	0.0663	26.31
Dados com *outliers* entre 50 e 100	0.0254	0.158	1.925	0.0677	85.17
Dados com *outliers* entre 500 e 1000	0.0275	0.165	3.912	0.0705	678.51

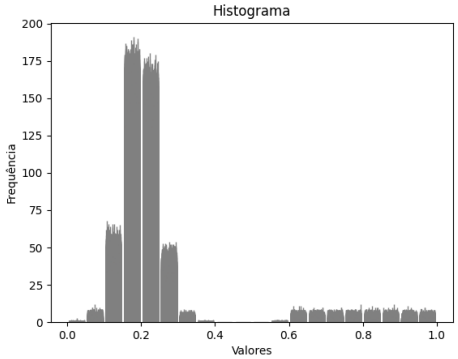
Tabela 6.17: *Tabela contendo erro e métricas ao treinar o modelo de GNN e avaliar o conjunto de teste após realizar o treinamento com alguns conjuntos de dados com outliers e de referência.*

Presença de Ruído

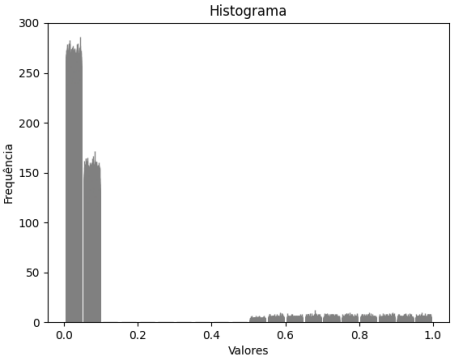
Uma das propriedades mais relevantes dos dados é a presença de ruído. A fim de verificar o impacto do ruído no resultado e qual nível de ruído é aceitável, realizou-se o teste de ruído, adicionando valores de ruído (advindos de uma distribuição uniforme) entre 0 e uma potência de 10 aos dados (antes da normalização). O resultado está exemplificado na figura 6.8 (os 3 modelos apresentaram o mesmo comportamento e as demais métricas apresentam o mesmo padrão da figura). Nota-se que valores de ruído entre 0 e 10 praticamente não afetam o resultado, havendo um pequeno impacto com ruídos entre 0 e 100, mas não impossibilitando os modelos de aprenderem os padrões dos dados (no entanto, não deve ser de interesse prático utilizar esses dados, pois os valores previstos pelo modelo também conterão muito ruído). Considerando que o valor máximo do conjunto de dados do PEMSd3 é aproximadamente 1000, pode-se afirmar que ruídos com magnitude de no máximo 1% do maior valor dos dados não afetam o modelo, enquanto ruídos com magnitude entre 1% e 10% podem impactar o modelo e afetar os dados previstos, e ruídos com magnitude maior que 10% faz com que o modelo aprenda principalmente (ou apenas) os padrões dos ruídos. Isso ocorre, pois se observa que aumentar a magnitude do ruído (até 10^6) aproxima os valores das métricas aos valores obtidos ao treinar os modelos com dados de distribuição uniforme.



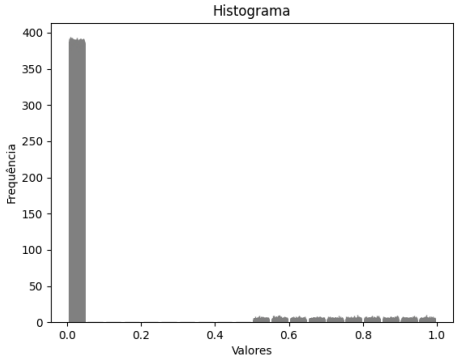
(a) *Histograma dos dados a partir de distribuição normal truncada à direita.*



(b) *Histograma dos dados com outliers entre 10 e 20.*



(c) *Histograma dos dados com outliers entre 50 e 100.*



(d) *Histograma dos dados com outliers entre 500 e 1000.*

Figura 6.7: *Comparação dos histogramas dos dados usados no teste de presença de outliers.*

Dados Interpolados

O teste de interpolação mostrou que, ao remover valores da série temporal e substituí-los por dados interpolados, ocorre uma diminuição aproximadamente linear nos valores de erro e métricas, como exemplificado na figura 6.9. Esse comportamento ocorreu igualmente nos 3 modelos e independentemente da escolha do método de interpolação. Isto ocorre, pois a interpolação substitui os valores faltantes por alguns valores que se repetem frequentemente, o que torna mais fácil aproximar uma função que descreva os dados.

6.5 Testes de Propriedades Estatísticas

Em relação aos testes envolvendo propriedades estatísticas, não se observou nenhuma correlação entre essas características e o comportamento dos modelos. Embora não tenham sido informativos neste caso, esses testes podem ser relevantes para problemas de classificação e para modelos de aprendizado de máquina estatísticos.

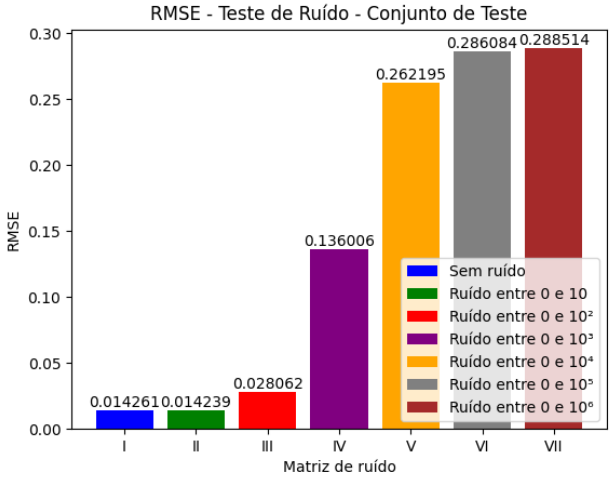


Figura 6.8: *Gráfico exemplificando o resultado do teste de ruído através da métrica RMSE, calculada para o conjunto de teste e utilizando o modelo de GNN.*

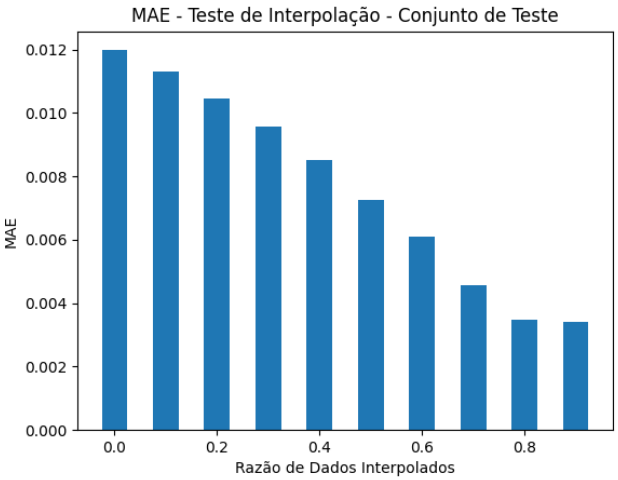


Figura 6.9: *Gráfico exemplificando o resultado do teste de interpolação através da métrica MAE, calculada para o conjunto de teste e utilizando o modelo de GNN.*

6.6 Testes de Extrapolação de Treinamento

Previsão por Ano

Neste teste, utilizou-se apenas o modelo de GNN, e os resultados mostraram que esse modelo consegue prever anos distantes do período do conjunto de treinamento com qualidade semelhante à obtida para o conjunto de teste referente ao mesmo ano do conjunto de treinamento. Com base nisso, é possível concluir, primeiramente, que o padrão temporal dos dados variou pouco ao longo dos anos (com exceção do ano de 2017) e que a aproximação do modelo é flexível o suficiente e não apresenta *overfitting*. Em relação ao valor do ano de 2017, é interessante comentar que nesse ano houve graves incêndios florestais na Califórnia, o que levou muitos moradores a deixarem suas casas, alterando

o padrão do fluxo de veículos.

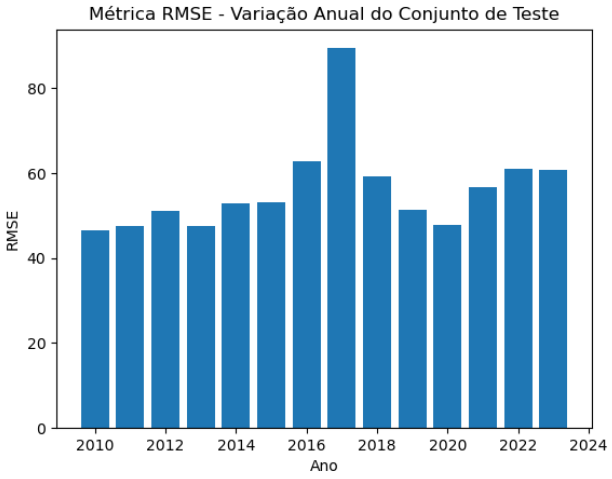


Figura 6.10: *Gráfico exemplificando o resultado do teste de previsão por ano através da métrica RMSE, utilizando o modelo de GNN.*

Comenta-se também que, embora não se tenha testado os demais modelos, provavelmente o resultado seria semelhante, dado que, na maioria dos demais testes, os 3 modelos apresentaram comportamentos iguais.

Extensão do Conjunto de Teste

Este teste é um complemento do teste anterior pois o conjunto de teste é incrementado sucessivamente com dados do ano seguinte, e utilizou-se apenas o modelo de GNN. Observando os resultados, nota-se que anos com previsões piores afetam a previsão geral, de modo que dados com previsões ruins necessitam de uma quantidade muito grande de dados com boas previsões para serem compensados, o que é exemplificado ao comparar o gráfico da figura 6.10 com o da figura 6.11. Na figura 6.10, nota-se que o ano de 2017 possui os piores valores da métrica RMSE, e esse valor ruim influencia os resultados dos anos seguintes para o teste de extensão do conjunto de teste como se observa na figura 6.11.

Dados de Mesma Classe

Primeiramente, geraram-se 3 classes derivadas de distribuições de probabilidade: Classe de dados gerados a partir de distribuições uniformes, classe de dados gerados a partir de distribuições exponenciais e classe de dados gerados a partir de distribuições normais. Nestas 3 classes, os resultados foram similares comparando os diferentes dados de uma mesma classe (variando em torno dos valores obtidos para os dados de referência), e também os 3 modelos se comportaram da mesma forma para estas classes.

Também definiu-se uma classe de dados em que os dados foram criados a partir de imagens em nível de cinza. O resultado desta classe pode ser visto na tabela 6.16, e comenta-se que todos os modelos se comportam de forma similar ao modelo de rede neural e que todos os dados da classe apresentam valores das métricas próximos.

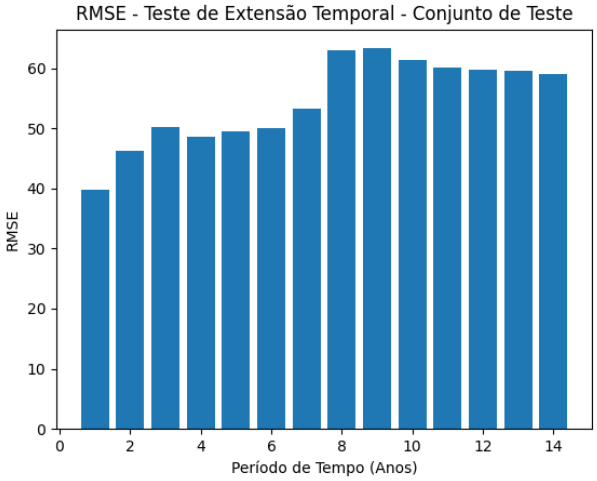


Figura 6.11: *Gráfico exemplificando o resultado do teste de extensão do conjunto de teste através da métrica RMSE, utilizando o modelo de GNN.*

Outras classes criadas (como a classe de dados com relações lineares ou a classe de dados com periodicidade) foram analisadas nos testes de propriedades dos dados e estatísticas.

Extrapolação de Treinamento para Mesma Classe

Para verificar como um modelo treinado com um dado de uma determinada classe pode ser utilizado para prever os resultados dos demais dados desta classe, utilizou-se um conjunto de dados de imagens em nível de cinza e treinou-se o modelo com uma série temporal gerada a partir da 1° imagem da figura 5.1.

Os valores obtidos das métricas para a previsão do conjunto de teste desta 1° imagem estão contidos na tabela 6.16, e, comparando com os valores obtidos para as demais imagens, nota-se que, no caso deste teste, o treinamento com uma imagem qualquer da classe pode ser usado para prever as demais imagens, considerando que o erro das outras imagens é semelhante, variando entre 10^{-5} e 10^{-6}, e o mesmo ocorrendo com as métricas, que apresentaram valores próximos. Este comportamento ocorreu igualmente nos 3 modelos.

Uma hipótese que pode ser formulada a partir deste resultado é que dados de uma classe que apresentam valores de erro e métricas semelhantes podem permitir o treinamento do modelo ser extrapolado para prever com qualidade os demais dados da classe. No entanto, é necessário reproduzir este teste com mais classes para se confirmar esta hipótese.

Extrapolação de Treinamento para Outras Classes

Finalizando os testes desta classe, treinaram-se os 3 modelos com todos os dados de referência (distribuição uniforme, exponencial e normal) e utilizou-se esses modelos treinados para prever os demais dados reais e sintéticos. Observando os resultados, nota-se que todos os demais dados apresentam valores de erro e métricas maiores do que os que seriam obtidos se fossem treinados os 3 modelos com os respectivos dados e também que os valores obtidos são maiores ou da ordem dos valores observados para os dados de referência de distribuição uniforme, indicando que, independentemente do conjunto de

referência utilizado no treinamento, nenhum modelo consegue extrapolar o treinamento para dados de outras classes, ou seja, dados muito diferentes.

6.7 Testes de Custo de Treinamento

Tempo de Treinamento

Analisando os modelos do ponto de vista de custo de treinamento, e partindo do custo no sentido de tempo, a regressão linear é o modelo mais eficiente, executando cada iteração em apenas alguns segundos. A rede neural também é bastante eficiente, demorando alguns segundos a mais que a regressão linear, enquanto a GNN é o modelo mais ineficiente, podendo demorar minutos por iteração, dependendo da quantidade de dados ou do tamanho deles. O tempo de treinamento está relacionado ao número de parâmetros do modelo e às operações realizadas. A regressão linear possui um pequeno número de parâmetros e as operações feitas são simples, por este motivo é o modelo mais eficiente. No caso da rede neural, o modelo é um pouco menos eficiente devido ao maior número de parâmetros (embora continue realizando operações simples), e a GNN é o modelo menos eficiente por causa da operação de convolução (mesmo a convolução implementada tendo menor complexidade do que um convolução convencional, conforme mostrado no capítulo 1).

Associado ao tempo por iteração, a quantidade de iterações para cada modelo atingir a convergência é similar, ficando em torno de 20 iterações. Entretanto, adicionar mais camadas ao modelo de GNN afeta essa quantidade, sendo necessário mais iterações para convergir.

Uso de Memória

Em relação ao uso de memória, a GNN é o modelo mais eficiente devido principalmente às bibliotecas da linguagem, que fazem um bom gerenciamento e armazenamento dos dados. No caso da regressão linear e da rede neural, os dados são armazenados diretamente na memória e passados diretamente para o modelo durante o treinamento, o que faz com que o uso de memória seja maior (evidenciando a importância da gestão de memória pelo programa e pela linguagem).

Quantidade de Dados de Treinamento

Finalizando essa classe de testes, verificou-se como cada modelo se comporta ao diminuir a quantidade de dados disponíveis para o treinamento (as figuras 6.12 e 6.13 ilustram o resultado desse teste). Nota-se que nos 3 modelos, diminuir o tamanho do conjunto de treinamento até 20% (correspondendo a aproximadamente 530 dados) gera pouco impacto no resultado, entretanto, a partir de 10%, todos os modelos sofrem um impacto mais severo nas métricas. Observando a figura 6.13, é possível concluir que a rede neural é o modelo mais robusto à diminuição da quantidade de dados, havendo uma piora mais significativa somente no caso mais extremo com apenas 0.1% do tamanho original. A regressão linear também apresenta um bom comportamento com a diminuição da quantidade de dados de treinamento, porém, em todos em casos testados, apresenta um erro um pouco maior que a rede neural. Por fim, em relação à GNN, esse é o modelo com pior desempenho, o que

demonstra a necessidade dos modelos mais complexos de aprendizado de máquina de uma maior quantidade de dados de treinamento para obter melhores resultados.

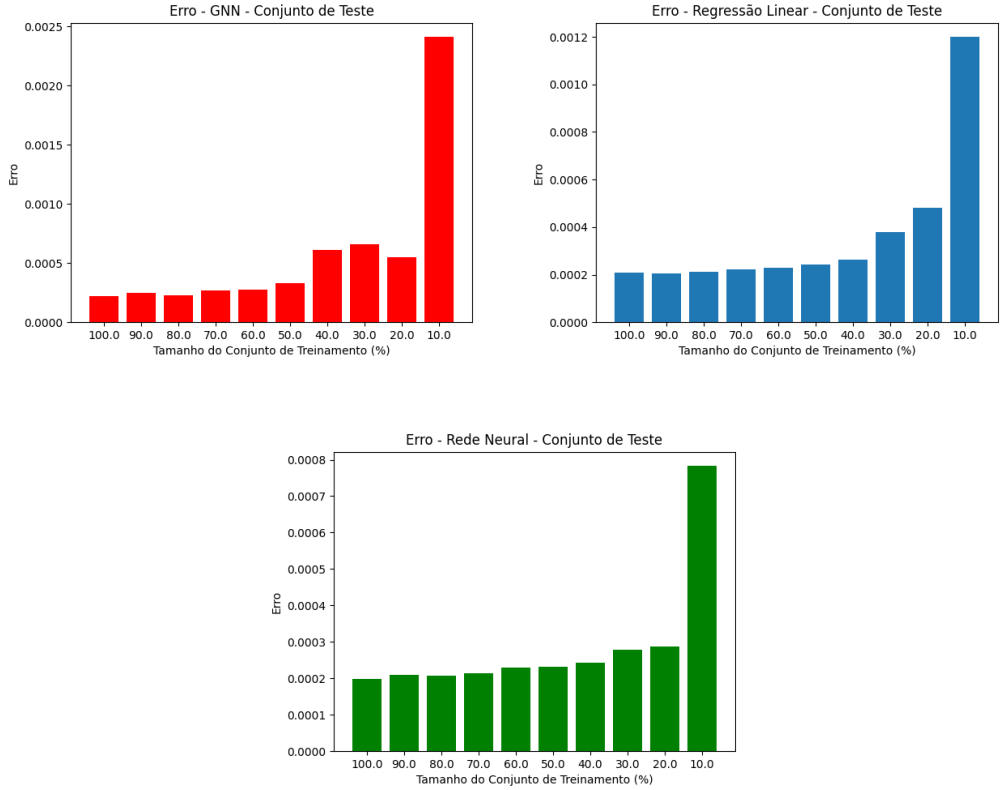


Figura 6.12: *Resultado do teste de diminuição do tamanho do conjunto de dados de treinamento, até 10% do tamanho original, para os 3 modelos. O tamanho do conjunto de treinamento original é 2649.*

6.8 Testes de Resíduos e de Pesos

Os testes de resíduos necessitam de um maior esforço para a análise dos resultados e, devido a isso, não foi possível obter resultados conclusivos no período deste trabalho, o mesmo ocorrendo com os testes de pesos. Assim, esses testes ficam como trabalho futuro para uma possível continuação do projeto.

6.9 Resultados Obtidos

Resumindo os resultados mais relevantes dos testes, pode-se chegar às seguintes conclusões.

Primeiramente, em todos os testes realizados, seja com dados reais ou sintéticos, a regressão linear apresentou resultados semelhantes ou até mesmo melhores que o modelo de GNN, o que indica que não é necessário um modelo tão complexo para obter bons resultados. Esse resultado é reforçado pela comparação com modelos criados a partir de modificações da GNN.

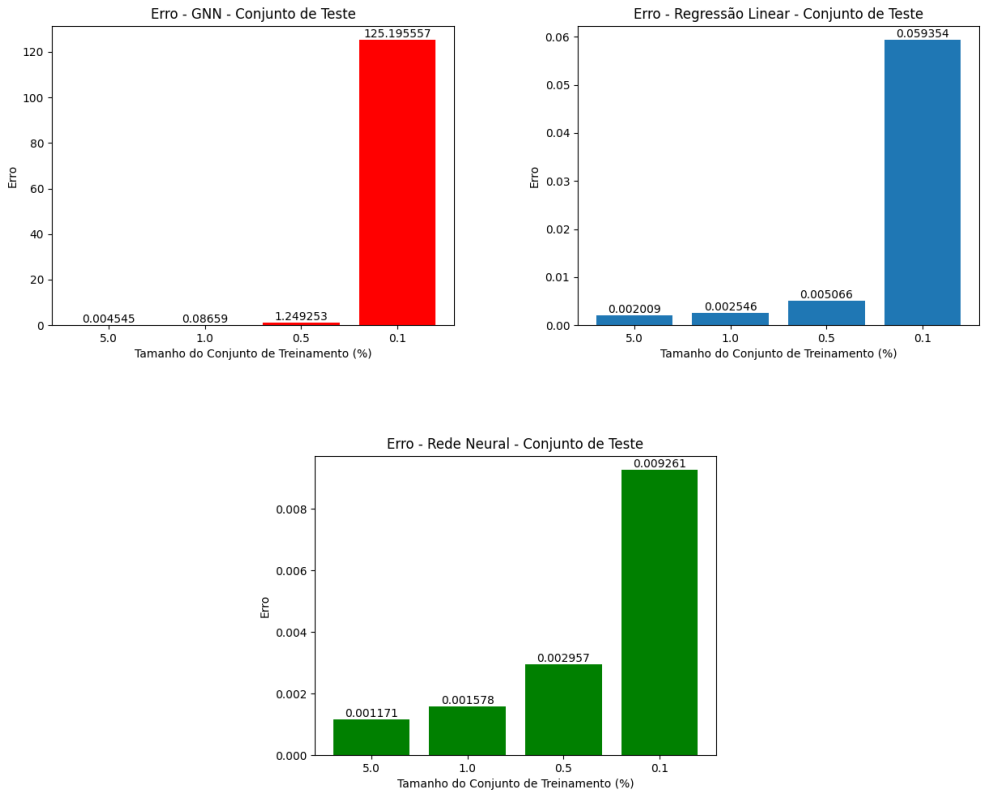


Figura 6.13: *Resultado do teste de diminuição do tamanho do conjunto de dados de treinamento (conjuntos entre 5% e 0.1% do tamanho original) para os 3 modelos. 0.1% do tamanho do conjunto original corresponde a apenas 2.*

Além disso, os testes de variáveis dos dados brutos mostraram que esse modelo de GNN é pouco sensível às dependências espaciais, o que é mais um motivo para utilizar modelos mais simples, tendo em vista que construir o grafo requer informações adicionais como coordenadas geográficas, além das informações de velocidade para geração da matriz de transição. Gerar estes elementos adicionais necessita despender mais tempo e esforço, o que, neste caso, não é recompensado com resultados melhores.

Essa classe de teste também mostrou que as informações temporais têm muito mais impacto no resultado, de modo que é desejável ter dados de qualidade de fluxo de veículos e com pequeno intervalo de amostragem. Adicionalmente, o teste de variação de temporalidade mostrou que os dados mais relevantes são os dados de curto prazo, o que reforça a ideia de ter dados com pequeno intervalo de amostragem.

Analisando as propriedades dos dados de treinamento, partindo da propriedade de presença de *outliers* nos dados, é possível concluir que todos os modelos se comportam igualmente e que, quanto maior a magnitude dos *outliers*, maior será o impacto nos resultados se estes valores não forem tratados. Em relação à presença de ruído, todos os modelos também se comportam igualmente e valores de ruído com magnitude de até 1% do maior valor da série temporal não impactam nos resultados, e ruídos de até 10% do valor máximo da série ainda geram pouco impacto nos resultados, o que é importante

saber para determinar se é necessário realizar um tratamento nos dados para melhorar a qualidade deles.

Os demais testes de propriedades dos dados de treinamento e estatísticas podem ser resumidos nos seguintes pontos:

- Relações lineares: A presença de relações lineares facilita a aproximação de uma função pelos modelos e, como esperado, a regressão linear é o modelo mais adequado para este caso.
- Periodicidade: A presença de periodicidade nos dados também facilita a aproximação de uma função adequada pelos modelos.
- Dados não normalizados: Utilizar dados não normalizados afeta os resultados dos modelos de modo distinto com base no padrão dos dados, na magnitude dos valores, na presença de valores negativos e na presença de valores inteiros ou reais, porém pode-se afirmar que a regressão linear é o modelo mais robusto a essas características, de modo que a identificação do padrão é pouco afetada.
- Propriedades estatísticas: Não se observou nenhum impacto dessas propriedades nos modelos.

Analisando os testes de extrapolação de treinamento observou-se que os dados reais apresentam pouca variação do padrão temporal mesmo após muitos anos, no entanto, o modelo de GNN não apresenta bons resultados ao tentar prever anos com padrões não vistos, o que impacta as métricas e necessita de uma grande quantidade de dados para que essas previsões ruins sejam absorvidas. Comenta-se também que, quando os dados de uma classe apresentam resultados similares, aparentemente é possível prever com qualidade os diferentes dados da classe utilizando no treinamento do modelo algum conjunto de dados distinto da classe.

Por fim, em relação ao custo de treinamento, os testes mostraram que a regressão linear e a rede neural são os melhores modelos, pois são modelos mais rápidos que a GNN e exigem menos dados de treinamento para obter bons resultados.

Capítulo 7

Conclusões

O arcabouço de testes desenvolvido é completo e generalizável, o que é garantido pela formulação teórica utilizada, a qual trata a resolução de problemas por meio de métodos de aprendizado de máquina supervisionado da forma mais básica, sem pressupor nenhuma condição. Com base neste estudo, foram identificados os componentes principais necessários para o aprendizado de máquina, ou que são obtidos como resultado do aprendizado e que podem ser testados. A partir destes componentes (elementos principais e derivados) são formuladas diferentes classes de testes, orientando como os testes devem ou podem ser criados e realizados. A união destas classes gera um procedimento completo de teste no qual cada elemento possui ao menos uma classe associada, de forma que cada teste é capaz de revelar informações de um ou mais elementos. Além disso, associado ao arcabouço há um processo no qual os resultados dos testes podem ser utilizados para a formulação de hipóteses que induzem a geração de novos testes, formando um ciclo que facilita a criação de testes e que contribui no entendimento dos resultados observados.

A fim de haver uma base para comparação dos resultados dos testes, é recomendado selecionar outros modelos de aprendizado de máquina além do modelo que se deseja efetivamente testar, de modo que estes outros modelos atuem como modelos de referência. Vários critérios podem ser utilizados para a seleção desses modelos, segundo algum objetivo de comparação, garantindo uma versatilidade de usos para o arcabouço. Outros modelos também podem ser utilizados para se verificar determinadas hipóteses.

Em relação aos dados sintéticos, embora eles tenham surgido da necessidade de dados com propriedades específicas para a execução dos testes de propriedades dos dados e estatísticas, notou-se que o uso deles pode ser expandido, atuando como dados de referência para comparação com outros dados sintéticos e reais, e por meio do conceito de classes de dados, obtém-se um outro processo de teste e de análise dos resultados obtidos com os dados sintéticos, permitindo principalmente a realização dos testes de resíduos, nos quais as propriedades testadas não são controláveis por meio dos dados de treinamento.

Os testes realizados para o problema de previsão de tráfego mostraram que o arcabouço tem a capacidade de revelar diversas informações comportamentais dos modelos além de resultados inesperados como o modelo de GNN ser pouco sensível às dependências espaciais e o modelo de regressão linear apresentar resultados iguais ou superiores aos

modelos mais complexos, bem como um comportamento mais estável e robusto a diferentes propriedades dos dados.

Concluindo, este trabalho mostrou que abordagens baseadas em testes podem ser um bom meio não somente de compreender o funcionamento de métodos de aprendizado de máquina supervisionado, mas também de auxiliar na resolução de problemas, guiando a seleção de dados adequados e de modelos com boa performance, e norteando a definição de bons padrões para os dados de treinamento.

7.1 Trabalhos Futuros

Com base nos estudos realizados, várias alternativas são continuações naturais para este trabalho. Primeiramente, um tema que não foi explorado devido ao foco do projeto no estudo de testes é o estudo de métodos de geração de dados sintéticos com a intenção de substituir os dados reais. Este assunto é de interesse pois em muitos problemas, como no problema de previsão de tráfego, o custo de obtenção dos dados reais é alto, não somente no sentido financeiro mas também no sentido de tempo. Além disso, a possibilidade de gerar dados com características variadas e semelhantes ao dados reais é especialmente desejável para uso nos testes e para garantir uma melhor generalização e extrapolação dos modelos de aprendizado de máquina. Assim, uma continuação para o trabalho é estudar a geração destes dados para o problema de previsão de tráfego e, se possível, estudar métodos generalizáveis para outros problemas.

Embora o arcabouço tenha se mostrado poderoso, ele possui alguns pontos negativos que podem ser abordados em estudos futuros. Primeiramente o arcabouço realiza uma quantidade muito grande de treinamentos dos modelos para obter os resultados de todos os testes, o que pode ser inviável para modelos complexos cujo treinamento consome muito tempo ou muitos recursos computacionais. Ademais, a forma que os testes foram formulados requer que os resultados precisem de análises manuais para compreender o que os valores revelam sobre os modelos, diferentemente de sistemas de software convencionais, nos quais os testes podem ser automatizados e os resultados são fáceis de interpretar, dado que os resultados são booleanos.

Outra possibilidade de continuação é aplicar o arcabouço desenvolvido para outros problemas, padrões, modelos e dados a fim de verificar a efetividade dos testes em outros contextos.

Além disso, em relação à formulação do arcabouço, um trabalho futuro de interesse é estudar métodos de teste específicos para problemas de classificação, bem como testes específicos para os diferentes tipos de arquiteturas de redes neurais.

Por fim, uma abordagem que pode ser considerada em uma continuação do trabalho é seguir uma formulação mais teórica para os testes, fundamentando-os com base em métodos formais e descrições matemáticas, por exemplo, utilizando também o arcabouço de estudo do *deep learning* geométrico, de modo que essa base teórica garantirá a corretude dos testes, enquanto que a própria ideia de testes fará uma conexão com um meio de uso prático.

Referências

[Abu-Mostafa *et al.* 2012] Yaser S. Abu-Mostafa, Malik Magdon-Ismail e Hsuan-Tien Lin. *Learning From Data*. AMLBook, 2012 (citado na pg. 24).

[Alexander *et al.* 2024] Yotam Alexander, Nimrod De La Vega, Noam Razin e Nadav Cohen. *What Makes Data Suitable for a Locally Connected Neural Network? A Necessary and Sufficient Condition Based on Quantum Entanglement*. 2024. arXiv: 2303.11249 [cs.LG]. URL: https://arxiv.org/abs/2303.11249 (citado na pg. 1).

[Ambühl *et al.* 2024] Lukas Ambühl, Kay W. Axhausen, Allister Loder e Monica Menendez. *UTD19*. 2024. URL: https://utd19.ethz.ch/ (acesso em 22/07/2024) (citado na pg. 16).

[Ammann e Offutt 2008] Paul Ammann e Jeff Offutt. *Introduction to Software Testing*. Cambridge University Press, 2008 (citado na pg. 23).

[Asadi e H. Jiang 2020] Behnam Asadi e Hui Jiang. "On approximation capabilities of relu activation and softmax output layer in neural networks". *CoRR* abs/2002.04060 (2020). arXiv: 2002.04060. URL: https://arxiv.org/abs/2002.04060 (citado na pg. 11).

[Bronstein *et al.* 2021] Michael M. Bronstein, Joan Bruna, Taco Cohen e Petar Veličković. *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. 2021. arXiv: 2104.13478 [cs.LG]. URL: https://arxiv.org/abs/2104.13478 (citado na pg. 1).

[*Caltrans. Performance Measurement System (PeMS)* 2024] *Caltrans. Performance Measurement System (PeMS)*. 2024. URL: http://pems.dot.ca.gov (acesso em 22/07/2024) (citado na pg. 16).

[Desikan e Ramesh 2007] Srinivasan Desikan e Gopalaswamy Ramesh. *Software Testing: Principles and Practices*. 1ª ed. USA: Prentice Hall Press, 2007. ISBN: 9788177582956 (citado na pg. 23).

[Graham *et al.* 2008] Dorothy Graham, Erik Van Veenendaal, Isabel Evans e Rex Black. *Foundations of Software Testing: ISTQB Certification*. Intl Thomson Business Pr, 2008. ISBN: 9781844803552 (citado na pg. 23).

[Hornik *et al.* 1989] Kurt Hornik, Maxwell Stinchcombe e Halbert White. "Multilayer feedforward networks are universal approximators". *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208 (citado na pg. 11).

[W. Jiang e Luo 2022] Weiwei Jiang e Jiayun Luo. "Graph neural network for traffic forecasting: a survey". *Expert Systems with Applications* 207 (2022), p. 117921. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2022.117921. URL: https://www.sciencedirect.com/science/article/pii/S0957417422011654 (citado na pg. 13).

[Jordon *et al.* 2022] James Jordon *et al. Synthetic Data – what, why and how?* 2022. arXiv: 2205.03257 [cs.LG]. URL: https://arxiv.org/abs/2205.03257 (citado na pg. 53).

[Kanestrøm 2017] Per Øyvind Kanestrøm. "Traffic flow forecasting with deep learning". Diss. de mestr. NTNU, 2017 (citado na pg. 13).

[Mitchell 1997] Tom M Mitchell. *Machine learning*. Vol. 1. 9. McGraw-hill New York, 1997 (citado na pg. 24).

[Morettin 2021] Singer Morettin. *Estatística e Ciência de Dados*. Grupo Editorial Nacional, 2021 (citado nas pgs. 10, 45).

[Myers *et al.* 2012] Glenford J. Myers, Corey Sandler e Tom Badgett. *The art of software testing*. 3rd ed. Hoboken e N.J: John Wiley & Sons, 2012 (citado na pg. 23).

[New York 2024] City of New York. *NYC OpenData*. 2024. URL: https://opendata.cityofnewyork.us (acesso em 22/07/2024) (citado na pg. 16).

[Razin *et al.* 2023] Noam Razin, Tom Verbin e Nadav Cohen. "On the ability of graph neural networks to model interactions between vertices". In: *Advances in Neural Information Processing Systems*. Ed. por A. Oh *et al.* Vol. 36. Curran Associates, Inc., 2023, pp. 26501–26545. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/543ec10715d964122ab7cb15f648772b-Paper-Conference.pdf (citado na pg. 1).

[Seber e Lee 2012] George A. F. Seber e Alan J. Lee. *Linear Regression Analysis*. 2nd. Wiley, 2012. ISBN: 978-0471415404 (citado na pg. 10).

[Shalev-Shwartz e Ben-David 2014] Shai Shalev-Shwartz e Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014, pp. I–XVI, 1–397. ISBN: 978-1-10-705713-5 (citado na pg. 24).

[Szegedy *et al.* 2014] Christian Szegedy *et al. Going Deeper with Convolutions*. 2014. arXiv: 1409.4842 [cs.CV]. URL: https://arxiv.org/abs/1409.4842 (citado na pg. 9).

[Vapnik 1998] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998 (citado na pg. 24).

[Zhang *et al.* 2019] Yang Zhang, Tao Cheng e Yibin Ren. "A graph deep learning method for short-term traffic forecasting on large road networks". *Computer-Aided Civil and Infrastructure Engineering* 34.10 (2019), pp. 877–896. doi: https://doi.org/10.1111/mice.12450. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/mice.12450. url: https://onlinelibrary.wiley.com/doi/abs/10.1111/mice.12450 (citado nas pgs. 2, 7, 13).