UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Modelos de regressão espacial aplicados à previsão de demanda de pedidos de *online food delivery*

Herlisson Maciel Bezerra

Dissertação de Mestrado do Programa de Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria (MECAI)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Herlisson Maciel Bezerra

Modelos de regressão espacial aplicados à previsão de demanda de pedidos de *online food delivery*

Dissertação apresentada ao Instituto de Ciências
Matemáticas e de Computação – ICMC-USP,
como parte dos requisitos para obtenção do título
de Mestre – Mestrado Profissional em Matemática,
Estatística e Computação Aplicadas à Indústria.
VERSÃO REVISADA

Área de Concentração: Matemática, Estatística e
Computação

Orientador: Prof. Dr. Vicente Garibay Cancho

USP – São Carlos
Fevereiro de 2022

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

B574m Bezerra, Herlisson Maciel
Modelos de regressão espacial aplicados à
previsão de demanda de pedidos de online food
delivery / Herlisson Maciel Bezerra; orientador
Vicente Garibay Cancho. -- São Carlos, 2021.
76 p.

Dissertação (Mestrado - Programa de Pós-Graduação
em Mestrado Profissional em Matemática, Estatística
e Computação Aplicadas à Indústria) -- Instituto de
Ciências Matemáticas e de Computação, Universidade
de São Paulo, 2021.

1. Spatial Autorregressive Model. 2. Redes
Neurais Artificiais. 3. Multilayer Perceptron. I.
Garibay Cancho, Vicente, orient. II. Título.

Herlisson Maciel Bezerra

**Spatial regression models applied to orders demand
forecasting of online food delivery**

Master dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master – Professional Masters in Mathematics, Statistics and Computing Applied to Industry. *FINAL VERSION*

Concentration Area: Mathematics, Statistics and Computing

Advisor: Prof. Dr. Vicente Garibay Cancho

USP – São Carlos
February 2022

Dedico esta obra aos meus pais, Lindoracy Maciel da Silva e Hermógenes Dantas Bezerra Filho, pelo apoio, incentivo e por não medirem esforços no decorrer de toda vida acadêmica, em especial no decorrer de toda trajetória do mestrado nessa Instituição de excelência com professores comprometidos que tanto me inspiram.

AGRADECIMENTOS

Agradeço ao professor Dr. Vicente Garibay Cancho pela orientação e contribuições realizadas a este trabalho.

Agradeço também a todos do corpo docente e técnico-administrativo do MECAI e do ICMC pelo apoio oferecido durante o curso.

Agradeço as empresas em que trabalhei e que foram flexíveis para que esse aperfeiçoamento técnico-profissional se transformasse em realidade.

Por último, e não menos importante, aos meus amigos e colegas que conheci no mestrado, à minha família e a minha noiva, Ellen Priscila Borges Oliveira, por me incentivarem a dar o meu melhor mesmo nos momentos mais difíceis.

“A educação é a arma mais poderosa que você pode usar para mudar o mundo.”
(Nelson Mandela)

RESUMO

BEZERRA, H. M. **Modelos de regressão espacial aplicados à previsão de demanda de pedidos de *online food delivery***. 2022. 76 p. Dissertação (Mestrado – Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2022.

O crescimento do segmento de *Online Food Delivery* estimulou uma mudança nos hábitos de consumo de alimentos em todo mundo. A essência de uma empresa desse ramo é a operação da rede de logística para que a entrega dos pedidos seja realizada de forma rápida e confiável. Para o planejamento eficiente da logística em questão, o insumo de previsão da demanda de pedidos que são realizados em determinadas áreas geográficas é essencial. O objetivo desta dissertação é realizar a predição da demanda de pedidos por área geográfica utilizando o modelo *Spatial Autorregressive Model* (SAR) e o modelo de Redes Neurais *Multilayer Perceptron* (MLP) para os distritos na região do centro expandido de São Paulo - SP, a partir dos dados reais de uma grande empresa do segmento. Do ponto de vista metodológico, foi proposta uma abordagem com uma matriz de treino aumentada com as informações de pedidos dos vizinhos de primeira ordem para que a rede neural seja capaz de identificar a autocorrelação espacial presente nos dados, enquanto que o SAR já é estatisticamente construído para incorporar a autocorrelação espacial ao ajuste e à predição realizada. Os modelos foram ajustados com os dados reais e foram avaliados com as métricas *Root Mean Squared Error* (RMSE) e pelo coeficiente de determinação R^2 . No resultado final, ambos os modelos tiveram desempenho satisfatório quando comparados com a média histórica dos pedidos. Na comparação entre o SAR e a MLP, a MLP com o melhor ajuste resultou em predições com o RMSE de 3,353 contra 3,604 do SAR e R^2 de 0,731 contra 0,689 do SAR. Portanto, dentre os dois modelos estudados, o modelo *Multilayer Perceptron* foi escolhido como o melhor modelo para o ajuste aos dados analisados.

Palavras-chave: *Spatial Autorregressive Model*, Redes Neurais Artificiais, *Multilayer Perceptron*.

ABSTRACT

BEZERRA, H. M. **Spatial regression models applied to orders demand forecasting of online food delivery**. 2022. 76 p. Dissertação (Mestrado – Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2022.

Online Food Delivery fast-growing has stimulated changes dramatically in the habits of food consumption throughout the world. The core of any company in that business is the operation of the logistics network in order to have fast and reliable orders delivery. For planning the operation of the logistics efficiently, the demand forecasting of orders by geographical areas is crucial. The aim of this work is to predict the orders demand of geographical area of the expanded centre of São Paulo - SP - Brazil, by using the Spatial Autorregressive Model (SAR) and the Artificial Neural Network - Multilayer Perceptron Model (MLP), from real data of a Brazilian food delivery company. Methodologically, it has been proposed an approach with an augmented train matrix that incorporates orders information from the neighbourhood areas of first order so that the neural network model were able to identify the spatial autocorrelation present in the data, despite the fact that SAR model is statistically built to account for the spatial autocorrelation to the forecast. Both of the models were trained by using real data and were evaluated with Root Mean Squared Error (RMSE) and by the coefficient of determination R^2 . In the simulation, both of the models reached satisfactory performance when compared to the historical mean. In the comparison between SAR and MLP, the MLP model with the best parameters presented prediction with RMSE of 3.353 against 3.604 of the SAR, it also had R^2 of 0.731 against 0.689 of the SAR. Therefore, among the two studied models, the Multilayer Perceptron has been chosen the best to fit the analysed data.

Keywords: Spatial Autorregressive Model, Artificial Neural Networks, Multilayer Perceptron.

LISTA DE ILUSTRAÇÕES

Figura 1 – Representatividade dos canais <i>online</i> e <i>offline</i> dos pedidos de <i>food delivery</i> .	26
Figura 2 – Locais das estações pluviométricas do CEMADEN na cidade de São Paulo - SP	32
Figura 3 – Mapa cloroplético dos casos de COVID-19 do estado de São Paulo até 10/2020	33
Figura 4 – Exemplos de processos pontuais no transporte de táxis em Nova Iorque - EUA	34
Figura 5 – Matrizes de vizinhança e de pesos para a região da subprefeitura da Sé em São Paulo-SP	36
Figura 6 – Ilustração do neurônio artificial MCP	42
Figura 7 – Ilustração de um neurônio artificial	44
Figura 8 – Ilustração de uma RNA com três camadas	45
Figura 9 – Região do centro expandido de São Paulo - SP	52
Figura 10 – Boxplots das distribuições de pedidos por distrito	55
Figura 11 – Mapa cloroplético dos pedidos por distrito	56
Figura 12 – Série de pedidos diários	56
Figura 13 – RMSE por distrito calculado na base de validação	58

LISTA DE ALGORITMOS

Algoritmo 1 – Algoritmo de aprendizado de retropropagação (<i>Backpropagation</i>) . . .	49
--	----

LISTA DE TABELAS

Tabela 1 – Funções de ativações	44
Tabela 2 – Estatística e p-valor dos testes aplicados	57
Tabela 3 – Métricas de avaliação dos ajustes dos modelos	58
Tabela 4 – Previsões da demanda de pedidos para o dia 15-10-2019 dos modelos ajustados	59

LISTA DE ABREVIATURAS E SIGLAS

LSTM	<i>Long-short Term Memory</i>
MAPE	<i>Mean Absolut Percentage Error</i>
MH	Média Histórica
MLP	<i>Multilayer Perceptron</i>
OFD	<i>Online Food Delivery</i>
ReLU	<i>rectified linear activation function</i>
RMSE	<i>Root Mean Squared Error</i>
RNAs	Redes Neurais Artificiais
tanh	tangente hiperbólica

SUMÁRIO

1	INTRODUÇÃO	25
1.1	Mudanças de consumo e potencial de mercado	25
1.2	<i>Online Food Delivery</i>	26
1.3	Justificativa	27
1.4	Objetivos	28
1.5	Estrutura da dissertação	28
2	REVISÃO DA LITERATURA	29
2.1	Previsão de demanda	29
2.2	Modelo SAR aplicado à previsão de demanda	29
2.3	Redes Neurais Artificiais para previsão de demanda	30
3	FUNDAMENTAÇÃO TEÓRICA	31
3.1	Estatística e modelagem para dados espaciais	31
3.2	Dados referentes à áreas e sua modelagem	34
3.2.1	<i>Matriz de proximidade espacial</i>	35
3.2.2	<i>Medidas de autocorrelação espacial</i>	37
3.2.3	<i>Modelos Simultâneos Autorregressivos (SAR)</i>	39
3.2.4	<i>Estimação do parâmetros do SAR</i>	40
3.3	Redes Neurais Artificiais	41
3.3.1	<i>Caracterização de uma RNA</i>	43
3.3.1.1	<i>Arquitetura</i>	43
3.3.1.2	<i>Aprendizado</i>	45
3.3.2	<i>Redes Perceptron</i>	46
3.3.3	<i>Backpropagation</i>	47
4	METODOLOGIA	51
4.1	Conjunto de dados	51
4.2	Ferramentas computacionais	51
4.3	Análise exploratória e pré-processamento	52
4.4	Ajuste dos modelos de previsão	53
4.5	Avaliação de qualidade dos modelos	53
5	RESULTADOS	55

5.1	Análise exploratória	55
5.2	Ajuste dos modelos	57
6	CONCLUSÃO	61
REFERÊNCIAS		63
APÊNDICE A	REPOSITÓRIO DE ARQUIVOS E CÓDIGOS UTILIZADOS	67
APÊNDICE B	CÓDIGOS DE MODELAGEM	69

INTRODUÇÃO

1.1 Mudanças de consumo e potencial de mercado

Em meados de 2016, uma das empresas de consultoria mais conceituadas do mercado, a Deloitte, discorreu sobre o atual panorama de logística da indústria de comida e sua iminente mudança de perspectiva para o ano de 2020. De acordo com o artigo, três pilares são enumerados como os fatores que norteariam as mudanças de paradigma da logística nesse mercado (OVERDULVE, 2016):

- Hábitos de alimentação: A demanda mais forte dos consumidores por produtos saudáveis, como produtos orgânicos e naturais, além do aumento da demanda por alimentos de dietas específicas, como vegetariana, vegana e livre de glúten, estão moldando novos hábitos na sociedade;
- Conveniência: O acesso quase ilimitado às informações em tempo real, através da internet pelos smartphones, moldou as preferências dos consumidores. Eles estão, dia a dia, mais exigentes para que os serviços atendam às suas necessidades. Isso resulta em um consumo do que eles querem, quando eles querem e onde eles querem;
- Demografia: A urbanização criou áreas distintas socioeconomicamente e com alta densidade populacional que exigem a devida adaptação das companhias para oferecer serviços em um modelo de entregas em menores quantidades em uma frequência maior.

A mudança de paradigma da logística de entrega de alimentos se consolidou e o mercado de *food delivery* está crescendo rapidamente. Em 2018, o mercado de *food delivery* movimentou na China cerca de US\$ 35,8 bilhões e nos EUA US\$32,7 bilhões (UNITED..., 2019; CHINA..., 2019). Estima-se que esse mercado atinja mundialmente um valor de US\$ 200 bilhões até 2025 (SINGH, 2019).

No Brasil, a [Abrasel \(2019\)](#) diz que a média recente que *food delivery* vêm movimentando é de cerca de R\$ 11 bilhões a cada 12 meses. Além disso, há um crescimento esperado de R\$1 bilhão por mês nesse volume vindo apenas de *online food delivery*, ou seja, de serviços que operam via aplicativos de celular e em páginas da internet.

1.2 Online Food Delivery

O segmento de *Online Food Delivery* (OFD) é o responsável pela mudança nos hábitos de consumo de alimentação que está acontecendo e pelo acelerado crescimento do *food delivery*. A facilidade de fazer um pedido ao simples toque de botões na tela do smartphone juntamente com a conveniência de inúmeras opções para escolher são fatores que ajudam consolidar esse comportamento ([HIRSCHBERG et al., 2016](#)).

Na figura 1, observa-se que desde meados de 2010 teve uma migração do comportamento dos usuários de realizarem pedidos offline (por exemplo, pelo telefone) para efetuarem pedidos online (via internet). Juntamente com a migração, o OFD está crescendo consistentemente através da exploração de uma demanda latente, à medida que as necessidades dos clientes passam a ser atendidas.

Figura 1 – Representatividade dos canais *online* e *offline* dos pedidos de *food delivery*

The food-delivery market has the potential for robust growth.

Total addressable classic food-delivery market,¹ %

¹For selected countries only.

²Compound annual growth rate.

McKinsey&Company

Fonte: [Hirschberg et al. \(2016\)](#).

Aliada a essa demanda latente, tem-se um comportamento das empresas que atuam nesse segmento para oferecer o serviço ao maior número possível de pessoas com estímulos através de promoções agressivas, como oferecer refeições a preços irrisórios. Essas promoções têm como principal objetivo deixar os consumidores conscientes da existência do serviço e, posteriormente, tentar moldar os seus hábitos de alimentação ao estimulá-los a deixarem de cozinhar e se alimentarem utilizando o serviço de OFD.

As empresas que atuam no mercado de OFD que são bem sucedidas são aquelas que conseguem fidelizar os seus clientes através da disponibilização de um serviço que oferece refeições de qualidade com um tempo aceitável de entrega, pois isso é uma das principais demandas dos clientes, e a um preço justo (SUHARTANTO *et al.*, 2019). Para tanto, a componente vital do negócio é o gerenciamento eficaz do sistema de logística o qual a empresa depende.

A rede logística a qual uma companhia de OFD precisa gerenciar é de difícil operação devido ao dinamismo em que está inserida (REYES *et al.*, 2018). Durante a operação dessa rede, tem-se três dinâmicas ocorrendo simultaneamente que devem atender à necessidade de três grupos distintos: dos clientes, que desejam receber seus pedidos rapidamente; dos entregadores, que desejam a melhor remuneração possível; e dos restaurantes, que querem a comida entregue sem perda de qualidade (ULMER *et al.*, 2017).

Conforme Mansano e Gorni (2014) a grande preocupação das empresas sempre foi a satisfação do cliente, pois o cliente satisfeito e fiel é o passo para o sucesso e permanência da empresa. Ou seja, a satisfação deve ser atingida através do processo de aquisição do produto e/ou serviço, logo se a empresa consegue satisfazer esse cliente na plenitude de suas expectativas, esse cliente se tornará fiel à empresa.

Além disso, sob a ótica da saúde financeira da empresa, o gerenciamento eficiente dos recursos disponíveis pode diminuir significativamente os custos, aumentando assim a rentabilidade final. No caso de uma empresa do ramo de OFD, ter uma logística eficiente é essencial para a satisfação do cliente e redução de despesas.

1.3 Justificativa

A motivação deste trabalho vem da necessidade de produzir um insumo essencial para realizar a otimização da rede logística, reduzindo os custos operacionais ao aumentar a eficiência dos entregadores direcionando-os previamente para as áreas onde haverá pedidos e, assim, mantendo um bom nível de satisfação dos clientes por reduzir o tempo de percurso do entregador até o local de retirada do pedido. O insumo em questão é a previsão da demanda de pedidos em uma determinada área geográfica (STEEVER; KARWAN; MURRAY, 2019).

Na literatura, vários modelos estatísticos já foram aplicadas para resolver o problema de previsão de demanda, como Seasonal Holt-Winters (TAYLOR, 2003), ARMA (Autorregressive

Moving Average) e ARIMA (Autorregressive Integrated Moving Average)(OHTSUKA; OGA; KAKAMU, 2010). Além de modelos estatísticos, técnicas de aprendizado de máquina também têm sido aplicadas, como Support Vector Machines (YUE *et al.*, 2007) e Redes Neurais Artificiais (AHMAD *et al.*, 2014). Entretanto, essas técnicas, do modo em que são aplicadas comumente na literatura, não levam em consideração a correlação geográfica de áreas próximas.

De acordo com Anselin (2001) a análise de dados e a modelagem necessária para a previsão de demanda por área geográfica tem nuances estatísticas diferentes dos modelos clássicos. Os modelos de regressão clássicos, em geral, pressupõem que as observações são observações mutuamente independentes. Porém, quando trata-se de dados relacionados à áreas geográficas, tem-se imediatamente que observações adjacentes exercem influência entre si causando uma interação espacial entre elas e, conseqüentemente, não são independentes. Essa interação espacial é nomeada de autocorrelação espacial.

1.4 Objetivos

Para que a autocorrelação espacial seja considerada, este trabalho propõe a partir do ajuste dos modelos realizar a previsão de demanda por área geográfica para um período de 24 horas com o uso do modelo estatístico SAR (*Spatial Autorregressive Model*) e com algoritmos estado de arte de Redes Neurais Artificiais, de modo que a autocorrelação espacial seja considerada. Além disso, como objetivos secundários pretende-se: estudar os conceitos estatísticos de modelagem de dados espaciais e do SAR; estudar os algoritmos de aprendizado de máquina estado-da-arte de Redes Neurais Artificiais (RNAs); ajustar os modelos em dados reais de uma grande empresa de OFD da América Latina e a analisar as métricas comparativas do desempenho dos dois modelos ajustados com os dados reais.

1.5 Estrutura da dissertação

Esta dissertação está organizada na ordem seguinte: no capítulo 2 é feita uma revisão de artigos pertinentes aos três pilares envolvidos: previsão de demanda de uma empresa, aplicação do modelo SAR para previsão de demandas e aplicação de RNAs para previsão de demandas. No capítulo 3, é exposta a fundamentação teórica que embasa os modelos que serão utilizados para realizar a previsão de demanda de uma empresa de *online food delivery*. No capítulo 4 é descrita a metodologia utilizada para o ajuste dos modelos e suas devidas avaliações de desempenho. No capítulo 5 é mostrado o ajuste dos modelos ao banco de dados reais e qual o desempenho de cada um dos ajustes realizados. Finalmente, no capítulo 6, uma síntese dos principais resultados é apresentada, destacando os aspectos mais relevantes da presente dissertação, as principais contribuições oferecidas, suas limitações e as sugestões de pesquisas futuras que possam dar andamento aos resultados obtidos por este estudo.

REVISÃO DA LITERATURA

Neste capítulo, será feita uma revisão da literatura referente à problemática de previsão de demanda e seus impactos para uma empresa; aos modelos de abordagem estatística que levam em consideração a estrutura de correlação espacial dos erros de regressão; às aplicações de aprendizado de máquina para o problema de previsão de demanda.

2.1 Previsão de demanda

[Santos, Junior e Bernardo \(2015\)](#) fazem uma ampla revisão bibliográfica sobre previsão de demanda falando do seu papel no gerenciamento de um negócio, de técnicas comumente empregadas e sobre os três principais tipos: quantitativo, qualitativo e integrado.

Os autores defendem que a previsão de demanda é o principal pilar para o planejamento de atividades básicas da empresa como fluxo de caixa e o planejamento da operação. Para tanto, podem utilizar-se dos métodos qualitativos que são os baseados diretamente no conhecimento ou experiências de analistas a respeito de futuros movimentos de mercados ou dos métodos quantitativos que são baseados apenas nos dados históricos utilizados para treinar modelos matemáticos e estatísticos a fim de realizar a predição com relações de senso comum equivocadas. Por fim, o método integrado busca fazer a junção dos dois anteriores para integrar informações qualitativas a uma previsão quantitativa e consequentemente obter um melhor poder preditivo.

2.2 Modelo SAR aplicado à previsão de demanda

O uso de modelos estatísticos que levam em consideração os efeitos de autocorrelação espacial não é tão habitual mas já está bem consolidado na literatura apesar das diferentes abordagens existentes. Os dois referenciados mostram como o SAR pode ser aplicado para a previsão de demanda em diferentes áreas, bem como o aumento do desempenho da predição

quando o problema exibe autocorrelação espacial em comparação com modelos que não levam essa questão em consideração.

Kerkman, Martens e Meurs (2018) fazem a aplicação de modelos estatísticos para previsão de demanda de transportes públicos de ônibus focada no fluxo de pessoas. Os autores utilizam a Regressão Linear Clássica (OLS) e o *Spatial Autorregressive Model* (SAR) para modelar o fluxo de pessoas em áreas dadas da região de Arnhem-Nijmegen, Holanda. Frisando que a Regressão Linear não incorpora o efeito de autocorrelação espacial e o SAR o sim. Por esse motivo, o SAR tem um claro desempenho superior pois produz previsões mais robustas e confiáveis em todas as simulações realizadas.

Já no estudo de Ohtsuka, Oga e Kakamu (2010), os autores utilizam uma abordagem espaço-temporal do SAR baseado no modelo *Auto Regressive Moving Average* (ARMA), utilizando assim um modelo SAR-ARMA para realizar a previsão da demanda energética por região geográfica do Japão. Com o estudo, os autores concluem que de acordo com a métrica *Log Predictive Score function* o modelo SAR-ARMA que leva em consideração as interações espaço-temporais dos dados tem um desempenho superior ao do ARIMA univariado testado.

2.3 Redes Neurais Artificiais para previsão de demanda

Yang *et al.* (2020) utilizam algoritmos de aprendizado de máquina, incluindo de RNAs, para prever a demanda de bicicletas dentro de sistema de compartilhamento de bicicletas com estações fixas. Os autores discorrem que realizar a previsão de demanda para o problema tem um bom nível de dificuldades por ter de se considerar as características espaço-temporais dos dados.

No estudo, eles ainda citam que ao se utilizar algoritmos de RNAs, em especial os de aprendizado profundo, a capacidade de extração automática de atributos são melhoradas e seu bom desempenho ao lidar com dados cujas características têm dependência espacial, temporal ou concomitantemente de ambas. Ademais, é ressaltada a importância de explicitar alguns comportamentos que os modelos não são capazes de capturar por si através de *feature engineering*. Com isso em mente, informações de fluxo das bicicletas entre áreas e horários são representados como grafos e através deles novos atributos são obtidos a partir de medidas calculadas das interações nos grafos.

No trabalho, a previsão é realizada para dois serviços de compartilhamento de bicicletas, um de Nova Iorque e outro de Chicago. As previsões são realizadas pelos métodos ARIMA, XGBoost, *Multilayer Perceptron* (MLP) e *Long-short Term Memory* (LSTM). A qualidade dos testes foram mensuradas utilizando as métricas *Root Mean Squared Error* (RMSE) e *Mean Absolut Percentage Error* (MAPE). As abordagens de aprendizado de máquina foram muito mais eficientes que o ARIMA e a inclusão dos atributos obtidos com as representações por grafos resultou em previsões com erros menores.

FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, será discutida a fundamentação teórica referente aos métodos estatísticos que tratam de modelagem de dados cuja informação da localização de ocorrência do evento é de suma importância. Também é discutido o modelo de aprendizado de máquina de Redes Neurais Artificiais, suas diferentes classes, arquiteturas e algoritmos.

3.1 Estatística e modelagem para dados espaciais

São inúmeras as áreas das ciências cujos fenômenos produzem dados que estão intrinsecamente associados a alguma localização física, como: a agricultura, a climatologia, a ecologia, e a, tão em evidência neste ano, a epidemiologia. Para se analisar ou modelar esses dados não basta somente quantificar os eventos em questão, mas também tem-se de indicar onde o evento com uma tal quantificação ocorre.

Cressie (2015) define um modelo estocástico geral para dados espaciais. Segundo o autor, é proposital o modelo ser vago, no sentido de não ter componentes causais explícitas, mas robusto o suficiente para englobar as nuances, que serão discutidas brevemente, dos dados espaciais. Dessa forma, seja $\mathbf{s} \in \mathbb{R}^d$ uma localização genérica de uma observação em um espaço Euclidiano d-dimensional e suponha que a observação $\mathbf{Z}(\mathbf{s})$ em um local espacial $\mathbf{s}$ é uma quantidade aleatória. Seja, ainda, $D \subset \mathbb{R}^d$ o conjunto de índices que $\mathbf{s}$ pode assumir, define-se então o seguinte campo multivariado aleatório (ou processo aleatório):

$$\{\mathbf{Z}(\mathbf{s}) : \mathbf{s} \in D\} \quad (3.1)$$

Uma realização de 3.1 será denotada como $\{\mathbf{z}(\mathbf{s}) : \mathbf{s} \in D\}$.

Uma subdivisão para a estatística de dados espaciais pode ocorrer com 3 casos particulares de 3.1: Dados Geoespaciais ou dados referentes a pontos, dados referentes à áreas e padrão

de pontos.

- **Dados Geoespaciais:** Esses dados são caracterizados pela localização s_i de cada observação $z(s_i)$ poder variar continuamente dentro do subconjunto $D \subset \mathbb{R}^d$. D é fixo, ou seja, não varia aleatoriamente e se estiver representando dados que estão associados a latitude e longitude será subconjunto do $\mathbb{R}^2$.

Um exemplo de dados que se encaixam nessa categoria são as medições da precipitação de chuva por cada pluviômetro automático que existe em uma cidade. Na figura 2, tem-se o mapa da cidade de São Paulo-SP, cuja área está em preto, e cada ponto laranja representa um pluviômetro instalado na cidade. Dessa maneira, a observação da precipitação $z_p(s)$ poderia ser realizada em qualquer ponto dentro da cidade (a cidade é o análogo ao subconjunto D em 3.1). Entretanto, em decorrência de custos e questões geográficas apenas alguns pontos da cidade foram escolhidos para a instalação dos pluviômetros. Portanto, as localizações s_i onde pode-se medir a precipitação de chuva são representadas por cada cruz roxa do mapa.

Figura 2 – Locais das estações pluviométricas do CEMADEN na cidade de São Paulo - SP

Fonte: Elaborada pelo autor.

- **Dados referentes a áreas:** Nessa categoria de dados D também é considerado fixo, assim não varia aleatoriamente, e é uma coleção enumerável de pontos de $\mathbb{R}^d$. Mas cada índice $s \in D$ é um conjunto de pontos com área não nula sendo representando em geral por um polígono, regular ou não. Dessa forma, $Z(s)$ é um vetor aleatório em cada área $s_i \in D$ que habitualmente representa uma região geográfica.

Para exemplificar, considera-se os municípios do estado de São Paulo. Desde Fevereiro, enfrenta-se a maior pandemia do século, a da Covid-19. Uma forma de visualizar quanto seria está sendo a pandemia por município do estado é mostrar um mapa do número de casos até o momento para cada cidade. Com esse intuito, tem-se a figura 3 que representa para cada município de São Paulo o número de infectados por Covid-19, de acordo com a discretização baseada nos quartis da distribuição. Mapas como o da figura 3 são chamados de mapas cloropléticos (CARVALHO; NATÁRIO, 2008). Nele, os tons mais escuros estão representando os municípios com a maior quantidade de casos enquanto que os tons mais claros mostram as cidades com as menores quantidades. Além disso, pode-se notar que cada delimitação de município é o análogo a s_i e cada contagem $Z(s_i)$ de casos está sempre associada a área do município.

Figura 3 – Mapa cloroplético dos casos de COVID-19 do estado de São Paulo até 10/2020

Fonte: Elaborada pelo autor.

- Padrão de pontos: Os dados dessa classe são os que D não é fixo e sim varia aleatoriamente como um subconjunto de $\mathbb{R}^d$, ou seja D é um processo pontual. Deste modo, o principal interesse nas observações de $\mathbf{Z}(\mathbf{s})$ é em qual local $s_i \in D$ está ocorrendo $\mathbf{Z}(\cdot)$.

A localizações onde ocorre o embarque e desembarque de viagens de táxi em uma dada cidade são exemplos desse caso. Na figura 4, tem-se os pontos de embarque (4a) e desembarque (4b) de viagens realizadas por táxis da cidade de Nova Iorque em uma determinada data, entre o período das 07:30 às 08:00 da manhã. Considerando que o interesse é em qual localização pode ocorrer um embarque e um desembarque, observa-se, portanto, que os dois fenômenos podem ser classificados como padrão de pontos.

Figura 4 – Exemplos de processos pontuais no transporte de táxis em Nova Iorque - EUA

Fonte: Elaborada pelo autor.

Os diferentes modelos disponíveis na literatura para modelagem de dados espaciais tratam cada um dos três tipos de dados comentados anteriormente. Apesar disso, [Cressie \(2015\)](#) argumenta que apesar de a maioria dos modelos terem sido desenvolvidos focados em cada uma das categorias citadas, nada impede que modelos de uma sejam aplicados em outra, com os devidos cuidados. Com isso em mente, neste trabalho será abordado com mais detalhes a modelagem que trata dos dados referentes às áreas, pois o conjunto de dados reais que serão tratados estão neste formato.

3.2 Dados referentes à áreas e sua modelagem

No exemplo mostrado na seção 3.1, tem-se o caso de dados referentes à áreas em que o que se está mostrando é a quantidade acumulada dos casos de Covid-19 por município. No caso em questão, as fronteiras das áreas que representam os municípios são bem delimitadas e já são pré-definidas mas nem sempre isso acontece. Algumas vezes quem está estudando os dados precisa definir as delimitações das áreas, seja as definindo como uma malha regular ou agregando áreas já existentes.

Independentemente se as áreas que serão representadas são polígono regulares ou não, o principal propósito quando se analisa uma quantidade de interesse que está indexada a localizações é verificar se há um padrão no qual valores similares, em geral acima da média, ocorrem próximos um dos outros ou estão dispostos de maneira aleatória. Isso pode ser analisado

utilizando gráficos como o da figura 3. Porém, considerar apenas uma análise subjetiva pode conduzir a uma conclusão equivocada de se há ou não autocorrelação espacial significativa presente nos dados (CARVALHO; NATÁRIO, 2008).

Assim, o uso de testes estatísticos para verificar a existência de autocorrelação espacial nos dados é essencial para minimizar a chance de uma conclusão equivocada. Dentre os possíveis testes que se podem aplicar, há o teste I de Moran e o teste c de Geary (CARVALHO; NATÁRIO, 2008). Os dois utilizam estatísticas de mesmo nome que mensuram a associação espacial presente nos dados. Contudo, a construção dos testes necessita de uma estrutura que explicita de antemão as relações e conexões das áreas. Essas estruturas são chamadas de Vizinhanças e sua modelagem é realizada através de um par de conjuntos que será denotados por N_i e D_N . Os elementos do conjunto N_i são as áreas que têm fronteira ou conexão com a área i , e D_N é o conjunto que explicita quais conexões uma área i possui (CRESSIE, 2015). Dessa forma, tem-se:

$$N_i \equiv \{k: \text{se } k \text{ é um vizinho de } i\}, \quad (3.2)$$

$$D_N \equiv \{(i; N_i) : i = 1, \dots, n\}. \quad (3.3)$$

Por definição D_N é um grafo e apesar de não descrever a informação da localização geográfica de cada área possui as informações necessárias das relações de vizinhança entre cada área i . A palavra vizinhança até o momento é referente a um conceito mais geral e que é definido. Isso implica que existem várias maneiras de se definir os vizinhos de uma dada área i e que o problema que está sendo estudado deve ser levado em consideração. As análises e modelos discutidos até o momento incorporam as informações de vizinhança em 3.3 na forma da matriz de proximidade espacial (também chamada de matriz de vizinhança) que é discutida na próxima seção.

3.2.1 Matriz de proximidade espacial

Segundo Carvalho e Natário (2008) a matriz de proximidade espacial é comumente denotada por $\mathbf{W}$ e é uma matriz quadrada de dimensão n em que cada elemento w_{ij} mede a ligação espacial entre as áreas i e j , com $i \neq j$, e $w_{ii} = 0$, $i, j = 1, \dots, n$.

Conforme comentado anteriormente de como se definir a vizinhança das diferentes áreas, as autoras citam que umas das formas mais utilizadas é a de designação binária para áreas que são consideradas como vizinhas, ou seja, $w_{ij} = 1$ se i e j são vizinhos e $w_{ij} = 0$, caso contrário. Nesse caso ser ou não considerado como vizinho pode seguir algumas das seguintes regras e depende de como as áreas estão estruturadas (se são regulares ou não):

- **Vizinhança por adjacência (Queen's contiguity):** $w_{ij} = 1$ se a área i e j compartilham alguma fronteira em comum e $w_{ij} = 0$, caso contrário (CARVALHO; NATÁRIO, 2008).

- **Vizinhança baseada na distância dos centroides:** Dado um ponto centroide que representa uma dada área i , consideram-se áreas vizinhas àquelas cujos centroides distam menos que uma distância fixa k , ressaltando que a distância pode ser calculada por quaisquer métricas. Portanto, $w_{ij} = 1$, se j está a uma distância menor ou igual a k de i utilizando uma métrica d , $w_{ij} = 0$ caso contrário (CARVALHO; NATÁRIO, 2008).
- **Proporção de fronteira compartilhada:** Cada elemento da matriz é calculado de acordo com a proporção da fronteira compartilhada com as demais áreas ao seu redor, ou seja, $w_{ij} = L_{ij}/L_i$, L_{ij} é o comprimento da fronteira entre as áreas i e j e L_i é o comprimento total da fronteira da área i (CARVALHO; NATÁRIO, 2008).

Uma vez construída a matriz de vizinhança $\mathbf{W}$ é comum realizar a normalização de suas linhas para que a soma de cada linha seja igual a 1, assim $w_i = \sum_{j=1}^n w_{ij} = 1$. A recomendação deve-se ao fato de que para a matriz normalizada, também chamada de matriz de pesos, os cálculos dos índices de autocorrelação espacial são simplificados e portanto tem um tempo de cálculo e custo computacional menor (CÂMARA *et al.*, 2004). Na figura 5, tem-se a matriz de vizinhança, por adjacência, e matriz de pesos para a região da subprefeitura da Sé, em São Paulo - SP.

Figura 5 – Matrizes de vizinhança e de pesos para a região da subprefeitura da Sé em São Paulo-SP

Fonte: Elaborada pelo autor.

A matriz de proximidade espacial pode ser generalizada para modelar também a medida de associação entre uma área não imediatamente vizinha a outra. Isso significa que pode-se construir a matriz $\mathbf{W}$ de forma que os pesos mensurem a conexão entre uma área s_i e o vizinho de seu vizinho, ou seja uma área s_j que é vizinha de segunda ordem de s_i . Dessa forma a matriz $\mathbf{W}$ é a matriz de proximidade espacial de segunda ordem, é denotada por $\mathbf{W}^{(2)}$ e é construída analogamente a de primeira ordem. A ideia pode ser estendida para quaisquer ordens dentro do limite das n áreas (CÂMARA *et al.*, 2004).

3.2.2 Medidas de autocorrelação espacial

Por mais que através de análises descritivas e mapas cloropléticos possam ser visualizados fortes indícios de que os dados que estão sendo estudados tenham associação espacial, é importantíssimo o cálculo de estatísticas e a condução de testes estatísticos para que hajam evidências de que a associação espacial é significativa e os dados sejam tratados de forma correta. Na literatura os índices de Moran e de Geary são os mais citados para tal análise.

O índice de Moran para os dados ou para os resíduos de uma regressão é definido como (MORAN, 1948):

$$I = \frac{n}{\sum_{i=1}^n \sum_{j=1}^n w_{ij}} \frac{\sum_{i=1}^n \sum_{j=1}^n w_{ij} (z(s_i) - \bar{z})(z(s_j) - \bar{z})}{\sum_{i=1}^n (z(s_i) - \bar{z})^2} \quad (3.4)$$

os quais n é quantidade de áreas; $z(s_i)$ é o valor observado de z na área s_i ; $\bar{z}$ é a média amostral de z e w_{ij} o elemento da linha i e coluna j da matriz de proximidade espacial.

Diferentemente de medidas de correlação, o valor que o I de Moran pode assumir não se restringe ao intervalo $[-1, 1]$. De fato, é comum o valor máximo de $|I|$ ser menor que 1 e dependendo dos valores da matriz de proximidade espacial, pode ainda assumir valor máximo maior que 1.

Apesar de não está limitado a um intervalo, a interpretação dos valores de I são:

1. Valores próximos de 0 indicam ausência de um padrão espacial e os valores observados tendem a estar dispostos de forma aleatória no mapa.
2. Valores positivos altos indicam dependência espacial e a formação de *clusters* de áreas com valores similares da variável aleatória em questão .
3. Valores negativos baixos também indicam dependência espacial, entretanto, a disposição em um mapa para esse caso seria de um padrão em que valores muito distintos compartilham áreas vizinhas (o que lembra o padrão de um tabuleiro de xadrez em um mapa cloroplético).

A construção do teste global de Moran tem a dificuldade da obtenção da distribuição da estatística I sob a hipótese nula de que os dados não exibem autocorrelação espacial (CLIFF; ORD, 1981). Entretanto, Gaetan e Guyon (2010, p. 166-169) demonstram que assintoticamente a distribuição de I é Normal com média

$$E(I) = \frac{-1}{n-1}, \quad (3.5)$$

e variância

$$Var(I) = \frac{n^2(n-1)S_1 - n(n-1)S_2 - 2S_0^2}{(n+1)(n-1)^2S_0^2}, \quad (3.6)$$

sendo

$$\begin{aligned} S_0 &= \sum_{i \neq j} w_{ij}, \\ S_1 &= \frac{1}{2} \sum_{i \neq j} (w_{ij} + w_{ji})^2 \\ S_2 &= \sum_k \left(\sum_j w_{kj} + \sum_i w_{ik} \right)^2. \end{aligned}$$

Com isso, a seguinte estatística

$$\hat{I} = \frac{I - E(I)}{\sqrt{Var(I)}} \quad (3.7)$$

é assintoticamente distribuída como uma Normal Padrão e corresponde a um determinado p-valor. Portanto, o índice I de Moran é significativamente diferente de zero se o p-valor é menor que o nível de significância pré-estabelecido.

Viton (2010) ressalta, ainda, três pontos importantes sobre a estatística e o teste do Moran. O primeiro é a importância de que a normalidade da distribuição do teste sob a hipótese nula é um resultado assintótico e, por isso, deve-se utilizá-lo com cautela para uma quantidade pequena de regiões mas, acrescenta que de acordo com estudos já realizados o teste performa suficientemente bem para ser utilizado nesses casos. O segundo ponto é o de que a estatística depende da matriz de adjacência $\mathbf{W}$, dessa forma arranjos diferentes da matriz podem produzir um resultado espúrio. O terceiro ponto é o de que quaisquer tendências nos dados tem de ter sido removidas antes da aplicação do teste sob o risco de, como falado no segundo ponto, apresentar uma significância espúria.

Conforme comentado no início dessa seção, a estatística c de Geary pode também ser utilizada como uma medida de associação espacial. Ela é dada por

$$c = \frac{(n-1) \sum_i \sum_j w_{ij} (z(s_i) - z(s_j))^2}{2(\sum_{i \neq j} w_{ij}) \sum_i (z(s_i) - \bar{z})^2} \quad (3.8)$$

e as variáveis seguem a mesma notação da equação (3.3). A distribuição do índice c sob a mesma hipótese nula de que os dados são independentes espacialmente, tem (GAETAN; GUYON, 2010, p. 169):

$$E(c) = 1, \quad Var(c) = \frac{(2S_1 + S_2)(n-1) - 4S_2}{2(n+1)S_0} \quad (3.9)$$

e é assintoticamente distribuída como uma Normal Padrão. Além do mais, a estatística c indica associação espacial positiva quanto menor é o valor da mesma. Isso significa que quanto menor o valor de c , maior a tendência de que áreas vizinhas tenham valores similares, e quanto maior o valor de c , maior a tendência de que áreas vizinhas tenham valores dissimilares. Valores próximos de 1, indicam a ausência de associação espacial.

3.2.3 Modelos Simultâneos Autorregressivos (SAR)

O *Simultaneous AutoRegressive Model* (ou *Simultaneous Spatial AutoRegressive Model*) pode ser introduzido de duas formas. Na primeira, considera-se o SAR como um caso particular dos modelos lineares mistos. Daí, parte-se dos pressupostos de que as observações $\mathbf{y} = (y_1, \dots, y_n)'$ são geradas por um processo que possui média e covariância bem definidas e são separáveis, além de que as covariáveis x_i podem ser modeladas linearmente. Dessa forma o modelo autorregressivo pode ser escrito como:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{z} + \boldsymbol{\varepsilon} \quad (3.10)$$

os quais: $\mathbf{X}$ é a matriz de p covariáveis observadas em cada uma das n áreas, de dimensão $n \times p$; $\boldsymbol{\beta}$ é o vetor de coeficiente da regressão que será estimado; $\mathbf{z}$ é o coeficiente de erro aleatório espacial latente, sendo que $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \Sigma_z)$ e $\boldsymbol{\varepsilon}$ é o erro independente o qual segue uma distribuição $\mathcal{N}(\mathbf{0}, \sigma_\varepsilon^2 \mathbf{I})$ (HOOTEN; HOEF; HANKS, 2014).

A modelagem espacial é explicitada na componente aleatória $\mathbf{z}$ que não é diretamente medida mas sim é inferido estatisticamente. Para o SAR, define-se $\Sigma_z \equiv \sigma_z^2 ((\mathbf{I} - \mathbf{B})(\mathbf{I} - \mathbf{B}'))^{-1}$, sendo que $\mathbf{B}$ é uma matriz $n \times n$ que é o parâmetro que modela as relações entre as áreas e σ_z^2 regula a variação no efeito aleatório correlacionado $\mathbf{z}$. A escolha mais usual para $\mathbf{B}$ é $\rho \mathbf{W}$. $\mathbf{W}$ é a matriz de proximidade espacial de primeira ordem como especificada na seção 3.2.1 e ρ é chamado de parâmetro de autocorrelação espacial, considerando $\mathbf{W}$ estocástica por linhas, e controla a força da dependência entre as áreas.

É importante notar que para Σ_z estar bem definida, $(\mathbf{I} - \rho \mathbf{W})^{-1}$ tem que existir. Uma condição suficiente para que isso aconteça é que $1/\lambda_{\min} \leq \rho \leq 1/\lambda_{\max}$, sendo que $\lambda_{\min}$ e $\lambda_{\max}$ são os menores e maiores autovalores de $\mathbf{W}$ respectivamente. Caso $\mathbf{W}$ seja estocástica por linhas, a condição pode ainda ser simplificada para $-1 \leq \rho \leq 1$.

A outra forma de introdução do SAR tem uma abordagem que enfatiza como a matriz de covariância de $\mathbf{z}$ é construída e, assim, é mais fácil de visualizar a dependência que é induzida. Para tanto, considera-se que $\mathbf{z}$ depende dela mesma simultaneamente, utilizando essa abordagem tem-se (HOOTEN; HOEF; HANKS, 2014):

$$\mathbf{z} = \mathbf{B}\mathbf{z} + \boldsymbol{\eta} \quad (3.11)$$

o qual $\eta \sim \mathcal{N}(\mathbf{0}, \sigma_Z^2 \mathbf{I})$. Daí, resolve-se a equação 3.11 para obter $\mathbf{z}$: $\mathbf{z} - \mathbf{B}\mathbf{z} = \eta \iff \mathbf{z} = (\mathbf{I} - \mathbf{B})^{-1}\eta$. Assumindo que a inversa de $(\mathbf{I} - \mathbf{B})$ existe, e como a $E(\eta) = 0$ e a $Cov(\eta) = \sigma_Z^2 \mathbf{I}$, conclui-se que $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \sigma_Z^2 (\mathbf{I} - \mathbf{B})^{-1} ((\mathbf{I} - \mathbf{B})^{-1})')$, cuja covariância pode ainda ser reescrita para ficar na mesma forma que a de Σ_Z .

Hoef *et al.* (2018) expõem que os modelos como especificados em (3.10) e (3.11) tem sido nomeados como "*spatial error*" *SAR models* e que também há outra abordagem bastante difundida no contexto econométrico chamada de "*SAR lag model*" a qual adota a autoregressão tanto da variável resposta quanto dos erros, tomando a seguinte forma:

$$\begin{aligned} \mathbf{y} &= \rho \mathbf{W}\mathbf{y} + \mathbf{X}\beta + \varepsilon \\ \mathbf{y} - \rho \mathbf{W}\mathbf{y} &= \mathbf{X}\beta + \varepsilon \\ \mathbf{y} &= (\mathbf{I} - \rho \mathbf{W})^{-1} \mathbf{X}\beta + (\mathbf{I} - \rho \mathbf{W})^{-1} \varepsilon \end{aligned} \quad (3.12)$$

3.2.4 Estimação do parâmetros do SAR

O método da Máxima Verossimilhança é o mais utilizado para realizar a estimativa dos parâmetros dos modelos SAR. O método tem por objetivo maximizar a função de verossimilhança que tem como resultado o valor máximo da probabilidade de se obter a amostra já observada.

Para uma variável aleatória normal multivariada $\mathbf{y}$ de vetor de médias μ e matriz de covariância Σ , o negativo do logaritmo natural da função da máxima verossimilhança é definido como (CRESSIE, 2015, p. 466):

$$-L(\mu, \Sigma) = \frac{n}{2} \ln(2\pi) + \frac{1}{2} \ln |\Sigma| + \frac{1}{2} (\mathbf{y} - \mu)' \Sigma^{-1} (\mathbf{y} - \mu) \quad (3.13)$$

De (3.10), pode-se notar que $\mathbf{y}$ segue a distribuição uma variável aleatória normal multivariada com $\mu \equiv \mathbf{X}\beta$ e $\Sigma \equiv \sigma^2 (\mathbf{I} - \rho \mathbf{W})^{-1} (\mathbf{I} - \rho \mathbf{W}')^{-1}$, pois é combinação linear de outras variáveis aleatórias normais conforme se supôs inicialmente. Dessa forma, pode-se reescrever (3.13) da seguinte forma:

$$L(\beta, \sigma, \rho) = -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(\sigma^2) - \ln |(\mathbf{I} - \rho \mathbf{W})| - \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\beta)' (\mathbf{I} - \rho \mathbf{W}) (\mathbf{I} - \rho \mathbf{W}') (\mathbf{y} - \mathbf{X}\beta), \quad (3.14)$$

daí basta calcular os estimadores de máxima verossimilhança. Fazendo isso, obtém-se (MARDIA, 1990, p. 210):

$$\hat{\beta} = (\mathbf{X}' \mathbf{S}^{-1} \mathbf{X})^{-1} \mathbf{X}' \mathbf{S}^{-1} \mathbf{y} \quad (3.15)$$

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{X}\hat{\beta})' \mathbf{S}^{-1} (\mathbf{y} - \mathbf{X}\hat{\beta})}{n} \quad (3.16)$$

em que $\mathbf{S} = (\mathbf{I} - \hat{\rho}\mathbf{W})^{-1}(\mathbf{I} - \hat{\rho}\mathbf{W}')^{-1}$.

Os estimadores que são obtidos do Método da Máxima Verossimilhança tem de ser calculados simultaneamente e isso gera um sistema para ser resolvido. [Bivand et al. \(2013, p. 284\)](#) apresenta uma estratégia para tratar o problema de resolução do sistema que é gerado: primeiramente, encontra-se o valor de $\hat{\rho}$, que é o coeficiente espacial autorregressivo, por um método de otimização, por exemplo *Line Search*. Com o valor de $\hat{\rho}$ fixado, calculam-se os valores dos outros dois parâmetros, $\hat{\beta}$ e $\hat{\sigma}$, pelo Método dos Mínimos Quadrados Generalizados, cujas equações são dadas em (3.15) e (3.16), encontrando-se assim as estimativas necessárias.

3.3 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNAs) são modelos computacionais inspirados no funcionamento biológico dos neurônios humanos. No campo da neurociência e da ciência cognitiva esses métodos são usados na tentativa de entender o funcionamento do nosso cérebro através de estudos de simulações. Já na área da engenharia e computação, esses modelos são estudados e utilizados para o processamento de informações, como aprendizado de máquina, reconhecimento de linguagens, reconhecimento de padrões em imagens, dentre inúmeros outros ([ALPAYDIN, 2020](#)).

A inspiração para os algoritmos de RNAs decorrem do fato da capacidade fenomenal do sistema nervoso humano (em especial o cérebro) de executar inúmeras tarefas de modo tão trivial. O cérebro é o principal responsável por tarefas como falar, locomover, reconhecer objetos e pessoas, além da capacidade de aprender. A eficiência relativa aos nossos computadores fazem dele a inspiração ideal para a construção de modelos e algoritmos que sirvam de base para ensinar máquinas e robôs a desenvolverem inteligência e serem aptos a aprenderem e realizarem as mesmas tarefas.

Para tanto, foi a partir da estrutura e comportamento, como entendidos atualmente, das células do cérebro chamadas de neurônios que os modelos de redes neurais tomaram sua primeira forma. As características mais marcantes do cérebro são as de ser formado por um grande número de neurônios, da ordem de 10 a 50 bilhões, e que cada um desses neurônios está conectado com centenas ou até milhares de outros neurônios. Essa enorme quantidade de neurônios interconectados confere ao cérebro um imenso poder de processamento paralelo pois os neurônios encontram-se organizados em grandes grupos conectados. Nas redes neuronais do cérebro, o processamento acontece através da possibilidade de excitabilidade eletroquímica de um neurônio e a perpetuação desse sinal elétrico para outras células ([FACELI et al., 2011](#)).

[McCulloch e Pitts \(1943\)](#) propuseram o primeiro modelo matemático de um neurônio artificial, que ficou conhecido como neurônio matemático MCP. O neurônio artificial MCP é composto por duas funções, as quais serão denotadas por $g(\cdot)$ e $f(\cdot)$. Para cada vetor $\mathbf{X} = (x_1, \dots, x_n)$ de valores de entrada, em que $x_i \in \{0, 1\}$, tem-se como valor de $g(\mathbf{X}) = \sum_i x_i$. Além

disso, define-se $y = f(g(\mathbf{X})) = 1$, se $g(\mathbf{X}) \geq \theta$ e $y = f(g(\mathbf{X})) = 0$, se $g(\mathbf{X}) < \theta$. O valor θ é dado e é conhecido como parâmetro de ativação.

Em resumo, $g(\cdot)$ é a função de agregação e $f(\cdot)$ é função de ativação que toma decisão baseada na soma dos valores de entrada. Daí, se o limiar θ é excedido, obtém-se $y = 1$ sendo um sinal de saída excitatório. Caso o limiar não seja alcançado, y assume valor 0 e é um sinal inibitório. A figura 6 ilustra esse esquema do neurônio artificial MCP e suas componentes. Por fim, apesar de os autores terem mostrado ainda que sistemas neurais criados a partir da combinação de vários neurônios artificiais tem alto poder de computar funções, eles não propuseram como essas redes teriam capacidade de aprendizado.

Figura 6 – Ilustração do neurônio artificial MCP

Fonte: Chandra (2018).

Então, Rosenblatt (1958) propôs o conceito de aprendizagem em RNAs com o modelo perceptron. O modelo perceptron proposto é baseado no modelo MCP, porém com algumas mudanças. As duas principais delas eram a proposta de pesos para os valores de entrada e uma regra para o aprendizado desses pesos. A rede perceptron original também era uma estrutura básica com uma única camada desses neurônios artificiais que faziam a conexão dos valores de entrada para a obtenção do valor de saída. A pesquisa na área de RNAs arrefeceu na década de 1970, especialmente porque Minsky e Papert (1969) mostraram que a rede perceptron era capaz de resolver apenas problemas que são separáveis linearmente.

Na década de 1980, as atenções voltaram-se novamente para as RNAs, culminando no final dessa mesma década nos estudos de Funahashi (1989) e Hornik, Stinchcombe e White (1989) que mostraram que redes perceptron com mais de uma camada, chamadas de Redes *Multilayer Perceptron* (MLP), são capazes de aproximar qualquer função contínua $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$, com um erro arbitrário, desde que um número suficiente de camadas esteja disponível. Outro aspecto importante resolvido nessa década foi o aprendizado eficiente nas redes com arquitetura MLP com a proposta do algoritmo *Backpropagation* (RUMELHART; HINTON; WILLIAMS,

1985).

Desde então, as RNAs tornaram-se um conjunto de modelos amplamente utilizados por sua versatilidade em modelar inúmeros tipos de dados de entradas, especialmente dados não estruturados, como reconhecimento e classificação de imagens e vídeos, sons e processamento de linguagem natural. Entretanto, para cada tipo específico de dado, certos tipos de arquiteturas e tamanhos de redes tendem a ter um desempenho superior quando comparados a outros (Hunter *et al.*, 2012).

Portanto, nas próximas seções serão mostrados os aspectos básicos que definem uma RNA, o funcionamento com mais detalhes das Redes Perceptron e do algoritmo *Backpropagation*.

3.3.1 Caracterização de uma RNA

Conforme Faceli *et al.* (2011) explica, uma rede pode ser caracterizada descrevendo duas de suas propriedades básicas, que são sua arquitetura e como ocorre o seu aprendizado. A arquitetura de uma RNA descreve qual é a unidade básica de processamento, como essas unidades básicas estão conectadas, como elas estão dispostas, considerando uma representação gráfica, e qual é a quantidade de unidades básicas presentes. Por outro lado, o aprendizado pode ser definido como o conjunto de regras matemáticas que são utilizadas para que os vetores de pesos sejam estimados e, conseqüentemente, a rede aprenda como executar a tarefa que lhe é designada, e.g. regressão, classificação ou agrupamento.

3.3.1.1 Arquitetura

A unidade básica de processamento de uma RNA é o neurônio artificial. O neurônio artificial atual é o mesmo proposto no perceptron com apenas a adição de um elemento extra no vetor de pesos chamado de bias. Esse elemento análogo ao coeficiente intercepto em uma regressão linear auxilia na capacidade de generalização do modelo.

O neurônio artificial é composto por um vetor $\mathbf{W} = (w_0, w_1, \dots, w_n)$ de pesos e para cada vetor $\mathbf{X}$ de valores de entrada, tem-se o valor de $g = (w_0, w_1, \dots, w_n)'(1, x_1, \dots, x_n) = \mathbf{W}'\mathbf{X}$ aplicado em uma função de ativação f e, conseqüentemente, é obtido o valor de saída y . Enquanto no neurônio clássico MCP a função de ativação f é fixa, no neurônio perceptron qualquer função que gere um valor de saída plausível a partir de $g(\cdot)$, caso exceda a um limiar, pode ser utilizada.

Figura 7 – Ilustração de um neurônio artificial

Fonte: Elaborada pelo autor.

Na literatura já foram propostas várias funções de ativação, dentre elas destacam-se as funções: linear identidade, limiar, sigmoidal, tangente hiperbólica ($\tanh$) e a *rectified linear activation function* (ReLU). A função linear identidade simplesmente retorna o valor da soma dos pesos multiplicados pelos valores de entrada em g . A limiar, retorna 1 caso seja excedido um limiar θ e 0, caso contrário. A função sigmoidal, ou função logística, tem um comportamento similar a limiar mas com a vantagem de ser contínua e diferenciável. Já a função $\tanh$, retorna valores entre -1 e 1, enquanto a sigmoidal varia entre 0 e 1 (FACELI *et al.*, 2011). Os gráficos e formas das funções de ativação mais comuns podem ser observados na tabela 1. Neste trabalho, simulou-se o desempenho das funções de ativação citadas e escolheu-se a que obteve melhor desempenho na etapa de teste.

Tabela 1 – Funções de ativações

Nome	Função	Gráfico
Linear Identidade	$f(x) = x$	
Limiar	$l(x) = \begin{cases} 1 & \text{se } x \geq \theta \\ 0 & \text{se } x < \theta \end{cases}$	
Sigmoidal	$\sigma(x) = \frac{1}{1 + e^x}$	
Tangente Hiperbólica	$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	
ReLU	$\text{relu}(x) = \max(0, x)$	

A disposição dos neurônios em uma RNA e suas conexões podem estar organizados de diversas formas. Em geral, as camadas são dispostas de modo que as conexões dos neurônios fluam em uma direção, ou seja, que os neurônios de uma dada camada i se conectem aos da camada $i + 1$, e assim sucessivamente, até a camada de saída. A rede MLP é um exemplo desse tipo de topologia. Na figura 8, pode-se visualizar uma RNA com multicamadas típica, tendo dois atributos como entrada, duas camadas intermediárias e uma de saída.

Figura 8 – Ilustração de uma RNA com três camadas

Fonte: Faceli *et al.* (2011).

Em geral, nas arquiteturas como a citada no parágrafo anterior o sinal da rede flui da camada de entrada para a camada de saída, passando por cada camada. Porém, os neurônios também podem apresentar conexões de retroalimentação (*feedbacks*). Essas conexões permitem que a informação de entrada de um neurônio não venha somente da camada anterior a ele, mas também de quaisquer neurônios de uma das camadas posteriores e pode, inclusive, vir dele mesmo. Redes com essas conexões de retropropagação são chamadas de redes recorrentes e são utilizadas em sistemas de aprendizado dinâmicos, como o controle de braço robótico. Na prática, redes do tipo *feedforward*, que são aquelas que não possuem conexões de retropropagação, tendem a ser mais utilizadas (FACELI *et al.*, 2011).

3.3.1.2 Aprendizado

O aprendizado de uma RNA consiste, na prática, no ajuste dos pesos que cada neurônio associa aos valores de entrada, sejam eles vindos dos dados observados ou da conexão proveniente de outro neurônio. O ajuste dos pesos é efetuado com uma regra bem definida e depende do tipo de problema que está sendo tratado. Para o caso de aprendizado supervisionado, algoritmos do tipo correção de erros são mais utilizados pois eles ajustam os pesos de forma a minimizar os erros cometidos em cada iteração do algoritmo de treinamento comumente utilizando uma técnica de otimização, como o Gradiente Descendente (ALPAYDIN, 2020).

Em problemas não supervisionados, algoritmos baseados na regra de Hebb chamados de Hebbianos são frequentemente utilizados. Essa regra define que conexões entre neurônios que se ativam simultaneamente devem ser reforçadas a cada iteração. Ainda para problemas não supervisionados, existem os algoritmos do tipo Competitivo que ajusta os pesos da rede simulando uma competição entre os neurônios de forma que os neurônios que têm uma resposta mais forte ao objeto de entrada ganham a competição tendo seus pesos atualizados (FACELI *et al.*, 2011).

Para o aprendizado nas Redes Multilayer Perceptron é muito comum a utilização do algoritmo backpropagation, que é do tipo corretor de erros. Detalhes do funcionamento e como os pesos são ajustados pelo backpropagation serão dados adiante (ALPAYDIN, 2020).

3.3.2 Redes Perceptron

As Redes Perceptron são redes do tipo *feedforward*, ou seja os neurônios de uma dada camada i se conectam apenas com outros da camada $i + 1$, e podem ser *single layer* (uma camada) ou *multilayer* (multicamadas). As que têm uma única camada são as redes cuja camada de entrada se conecta diretamente à camada de saída. Contudo, essa arquitetura de perceptron tem a limitação de ser apta a ajustar apenas funções lineares. As Perceptron Multicamadas são mais robustas e tem pelo menos uma camada intermediária (ou camada oculta) entre a camada de saída e entrada. Segundo Cybenko (1989), MLPs com uma camada oculta são capazes de ajustar quaisquer funções contínuas e com duas ou mais camadas ocultas quaisquer funções podem ser ajustadas.

As Redes MLP são robustas com a capacidade de ajustar qualquer função pelo modo como a saída de cada neurônio produzida individualmente é combinada em outros neurônios de camadas posteriores a que ele se conecta. Assim, os neurônios de cada camada à frente de uma dada camada atual tem a capacidade de computar funções compostas mais complexas que as da camada anterior (FACELI *et al.*, 2011).

Essa ideia pode ser expressada matematicamente como uma certa função $h_{i,j}(\mathbf{W}_{i,j}|\mathbf{X}_{i,j})$ que expressa a saída de um dado neurônio j em uma dada camada i . h pode ser parametrizada pelos pesos $\mathbf{W}_{i,j}$ do neurônio em questão e seus valores de entradas $\mathbf{X}_{i,j}$ aplicados à função de ativação pré-definida $f(\cdot)$. Os valores de entradas dos neurônios conectados ao neurônio i, j podem, por sua vez, ser representados pela devida função $h_{m,n}(\mathbf{W}_{m,n}|\mathbf{X}_{m,n})$, tais que $m < i$, $n < j$ e o neurônio m, n se conecta diretamente ao neurônio i, j . Esse processo pode ser feito recursivamente até $h_{i,j}$ ser dependente dos valores de entrada da primeira camada (NORVIG; RUSSELL, 2004). Portanto, pode-se expressar as saídas da rede em função dos dados de entrada e dos pesos.

Considerando uma rede com p saídas, $\mathbf{h}(\mathbf{W}|\mathbf{X})$ denotará a função vetorial que representa os p valores de saída da rede. Seja ainda $\mathbf{y}$ os valores reais que cada saída deveria gerar, calcula-

se o gradiente da função de perda L_2 , para qualquer peso $\mathbf{W}$ da seguinte forma (NORVIG; RUSSELL, 2004):

$$\frac{\partial}{\partial \mathbf{w}} L(\mathbf{y}; \mathbf{h}) = \frac{\partial}{\partial \mathbf{w}} |\mathbf{y} - \mathbf{h}(\mathbf{W}|\mathbf{X})|^2 = \frac{\partial}{\partial \mathbf{w}} \sum_{k=1}^p (y_k - h_k)^2 = \sum_{k=1}^p \frac{\partial}{\partial \mathbf{w}} (y_k - h_k)^2. \quad (3.17)$$

Para que o cálculo de (3.17) seja possível, a função de ativação escolhida precisa ser diferenciável, uma vez que ela compõe h_k . Além disso, nota-se que apenas com a equação é possível a dedução imediata de uma regra no caso em que não há camadas ocultas na rede, basta aplicar um método numérico para minimizar o gradiente da função de perda para obter os valores dos pesos da última camada, pois o erro é explícito.

No caso em que há camadas ocultas, precisa-se utilizar uma regra mais engenhosa para que ocorra o treinamento da rede, pois os erros das camadas ocultas não podem ser comparados diretamente com os valores de $\mathbf{y}$. Uma das saídas encontradas para tal problema resultou na construção do algoritmo *Backpropagation*.

3.3.3 Backpropagation

O algoritmo *Backpropagation* é o resultado da adaptação da aplicação do método de otimização do gradiente descendente para minimizar a função de erro, que para regressão é a mesma em (3.17), dado um conjunto de dados de treinamento. Em geral, no algoritmo não é interessante encontrar o mínimo global para a função $L(\mathbf{y}; \mathbf{h})$, pois a chance disso incorrer em superajuste aos dados observados é muito alta. Isso é tratado com a inserção de uma penalidade ou impondo um critério de parada das que limita a quantidade de total de iterações (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

Para derivar as equações iterativas do algoritmo, partiremos da função $L(\mathbf{y}; \mathbf{h})$ para uma saída particular k , considerando, sem perda de generalidade, uma rede com uma camada oculta. Daí, o gradiente de $L_k(y_k; h_k) = (y_k - h_k)^2$ com relação aos pesos de conexão com a camada oculta até a camada de saída é zero, exceto para o pesos $w_{j,k}$ que se unem à k -ésima unidade de saída. Tem-se, portanto (NORVIG; RUSSELL, 2004):

$$\begin{aligned} \frac{\partial L_k}{\partial w_{j,k}} &= -2(y_k - h_k) \frac{\partial h_k}{\partial w_{j,k}} = -2(y_k - h_k) \frac{\partial f(g_k)}{\partial w_{j,k}} \\ &= -2(y_k - h_k) f'(g_k) \frac{\partial g_k}{\partial w_{j,k}} = -2(y_k - h_k) f'(g_k) \frac{\partial}{\partial w_{j,k}} \left(\sum_j w_{j,k} h_j \right) \\ &= -2(y_k - h_k) f'(g_k) h_j. \end{aligned} \quad (3.18)$$

Dado isso, a obtenção do gradiente com relação aos pesos $w_{i,j}$ em conexão com a camada

de entrada até a j -ésima camada oculta é obtida através da expansão de h_j e com a reaplicação da regra da cadeia, resultando em:

$$\begin{aligned}
 \frac{\partial L_k}{\partial w_{i,j}} &= -2(y_k - h_k) \frac{\partial h_k}{\partial w_{i,j}} = -2(y_k - h_k) \frac{\partial f(g_k)}{\partial w_{i,j}} \\
 &= -2(y_k - h_k) f'(g_k) \frac{\partial g_k}{\partial w_{i,j}} = -2(y_k - h_k) f'(g_k) \frac{\partial}{\partial w_{i,j}} \left(\sum_j w_{j,k} h_j \right) \\
 &= -2(y_k - h_k) f'(g_k) w_{j,k} \frac{\partial h_j}{\partial w_{i,j}} = -2(y_k - h_k) f'(g_k) w_{j,k} \frac{\partial f(g_j)}{\partial w_{i,j}} \\
 &= -2(y_k - h_k) f'(g_k) w_{j,k} f'(g_j) \frac{\partial g_j}{\partial w_{i,j}} \\
 &= -2(y_k - h_k) f'(g_k) w_{j,k} f'(g_j) \frac{\partial}{\partial w_{i,j}} \left(\sum_i w_{i,j} h_i \right) \\
 &= -2(y_k - h_k) f'(g_k) w_{j,k} f'(g_j) h_i.
 \end{aligned} \tag{3.19}$$

Por causa da aplicação da regra da cadeia que mostra que parte do erro é "propagada" de uma camada à outra, cunhou-se o nome de *Backpropagation*. Além disso, com as equações (3.18) e (3.19), é possível escrever as regras principais de atualização dos pesos, levando em consideração que cada iteração é baseada em gradiente descendente. Para tanto, pode-se definir para uma camada de saída que: $\delta_k = \text{Erro}_k \times f'(g_k)$, h_j é o valor de entrada recebido e η é uma penalidade chamada taxa de aprendizado. Assim a equação de atualização para uma iteração t é:

$$w_{j,k}^{(t+1)} \leftarrow w_{j,k}^{(t)} + \eta h_j \delta_k. \tag{3.20}$$

Para uma camada intermediária, tem-se $\delta_j = f'(g_j) \sum w_{j,k} \delta_k$. Portanto a regra de atualização dos pesos se torna:

$$w_{i,j}^{(t+1)} \leftarrow w_{i,j}^{(t)} + \eta h_i \delta_j. \tag{3.21}$$

O algoritmo funcionará então em duas principais fases. Na primeira fase, os valores de entrada são apresentados para rede, todos os cálculos são realizados até produzirem a saída em cada neurônio da última camada. As saídas geradas são então comparadas com as saídas esperadas para cada neurônio. O erro cometido para cada um dos neurônios é utilizado então na segunda fase do algoritmo. A segunda fase consiste na utilização dos erros calculados para a atualização dos pesos da camada de saída, como mostrado em (3.20), e posteriormente, faz-se a estimativa dos erros de uma camada intermediária como a soma dos erros da camada seguinte, como mostrado em δ_j para a atualização dos pesos como mostrado em (3.21). Esse processo é repetido até a atualização dos pesos da primeira camada intermediária. A primeira fase é chamada de fase *forward* e a segunda de *backward*. O algoritmo completo é mostrado em 1.

Algoritmo 1 – Algoritmo de aprendizado de retropropagação (*Backpropagation*)Fonte: [Norvig e Russell \(2004\)](#).

```

1: procedimento BACKPROPAGATION(exemplos, rede,  $\eta$ )
2:   entrada: exemplos, um conjunto de exemplos, cada um com vetor de entrada  $x$  e vetor
      de saída  $y$ 
3:   entrada: rede, uma rede multicamadas com  $L$  camadas, pesos  $w_{i,j}$  e função de ativação  $f$ 
4:   variáveis locais:  $\delta$ , um vetor de erros indexado pelo nó de rede
5:   para cada peso  $w_{i,j}$  na rede faça
6:      $w_{i,j} \leftarrow$  um número aleatório pequeno
7:   fim para
8:   repita
9:     para cada exemplo  $(x,y)$  em exemplos faça
10:      /*Propagar as entradas para frente para calcular as saídas*/
11:      para cada nó  $i$  na camada de entrada faça
12:         $h_i \leftarrow x_i$ 
13:      fim para
14:      para  $\ell = 2$  até  $L$  faça
15:        para cada nó  $j$  na camada  $\ell$  faça
16:           $g_j \leftarrow \sum_i w_{i,j} h_i$ 
17:           $h_j \leftarrow f(g_j)$ 
18:        fim para
19:      fim para
20:      /*Propagar deltas retrocedendo da camada de saída para camada de entrada*/
21:      para cada nó  $j$  na camada de saída faça
22:         $\delta[j] \leftarrow f'(g_j) \times (y_j - h_j)$ 
23:      fim para
24:      para  $\ell = L - 1$  até  $1$  faça
25:        para cada nó  $i$  na camada  $\ell$  faça
26:           $\delta[i] \leftarrow f'(g_i) \sum_j w_{i,j} \delta[j]$ 
27:        fim para
28:      fim para
29:      /*Atualiza cada peso na rede usando deltas*/
30:      para cada peso  $w_{i,j}$  na rede faça
31:         $w_{i,j} \leftarrow w_{i,j} + \eta \times h_i \times \delta[j]$ 
32:      fim para
33:    fim para
34:    até que algum critério de parada seja satisfeito
35:    retorna rede
36: fim procedimento

```

METODOLOGIA

4.1 Conjunto de dados

O banco de dados analisado corresponde a uma amostra de pedidos realizados em uma grande empresa de OFD da América Latina, no período de 07/2019 a 10/2019, de uma região específica da cidade de São Paulo - SP. O banco de dados contém 36135 observações e as informações disponíveis são data e horário do pedido, a localização geográfica aproximada de onde ele foi realizado e o valor total final do pedido. Além disso, a área específica do estudo é a da região dos distritos que fazem parte do centro expandido da cidade de São Paulo - SP, que são: Vila Leopoldina, Lapa, Barra Funda, Perdizes, Alto De Pinheiros, Pinheiros, Jardim Paulista, Itaim Bibi, Moema, Vila Mariana, Saúde, Bom Retiro, Santa Cecília, Consolação, Bela Vista, Liberdade, Sé, República, Cambuci, Pari, Brás, Belém, Mooca e Ipiranga, mais o Itaim Bibi, somando 24 áreas distintas (Figura 9).

4.2 Ferramentas computacionais

Todas as análises, ajustes de modelos e a grande maioria das figuras geradas durante o estudo foram construídas utilizando linguagem Python, versão 3.7. As bibliotecas GeoPandas (JORDAHL *et al.*, 2020) e o Framework Pysal (REY; ANSELIN, 2010) foram os principais pilares dos códigos escritos. A GeoPandas é uma biblioteca equivalente ao Pandas, utilizada para manipulação e análise de dataframes com dados geoespaciais. O Pysal é um projeto de código aberto com bibliotecas para explorar, modelar e visualizar dados geoespaciais, fomentando o *spatial data science*. Para o ajuste e refinamento dos parâmetros da rede MLP, bem como para o cálculo das métricas de desempenho, foi utilizada a biblioteca Scikit-learn (PEDREGOSA *et al.*, 2011).

Figura 9 – Região do centro expandido de São Paulo - SP

Fonte: Elaborada pelo autor.

4.3 Análise exploratória e pré-processamento

Na etapa de análise exploratória dos dados, analisou-se os boxplots e a distribuição dos dados para uma análise gráfica preliminar da existência de autocorrelação espacial. Em seguida, observou-se se havia tendência ou não na série de pedidos diários, para a aplicação dos teste I de Moran e C de Geary e, assim, verificar se a autocorrelação espacial presente era significativa estatisticamente.

Na etapa de pré-processamento, efetuou-se o preenchimento de dados faltantes para algumas datas em regiões cujas informações de pedidos não estavam disponíveis, correspondendo a 8,29% dos dados. A técnica utilizada foi a do preenchimento pela moda da quantidade de pedidos de cada região (FACELI *et al.*, 2011).

Ainda antes dos ajustes, realizou-se a criação de novas variáveis preditoras a partir das informação disponíveis na base de dados com o objetivo de incrementar o poder de predição dos algoritmos. As novas variáveis criadas são: o cálculo quantidade de pedidos da região defasados entre o 1º e o 7º dias anteriores a data de referência utilizada; o dia da semana em que o pedido foi efetuado; *flag* se o pedido foi realizado em um fim de semana. A etapa final do pré-processamento foi a eliminação de variáveis correlacionadas através do cálculo da correlação de Pearson e priorizando as com maior correlação com a variável resposta.

No passo seguinte, dividiu-se a base de dados como bases de treino, teste e validação.

Como as observações têm um caráter temporal e sazonal semanal, levou-se em consideração a data de ocorrência dos pedidos para a divisão das bases. A base de treino utilizou dados do período de 01/07/2019 a 30/09/2019. As avaliações iniciais e o *tunning* do modelo MLP foram realizadas no período de 01/10/2019 a 07/10/2019. E a avaliação final dos modelos ajustados foi calculada com os dados de 08/10/2019 a 14/10/2019, com isso o modelo pode ser avaliado sem viés dos dados de treino.

4.4 Ajuste dos modelos de previsão

Com a base de dados devidamente tratada, realizou-se o ajuste do modelo SAR e do modelo *Multilayer Perceptron*. Para o ajuste do modelo SAR, utilizou-se o conjunto de dados padrão resultante do pré-processamento. A matriz de vizinhança utilizada em todos os ajustes do SAR foi construída utilizando o critério de contiguidade. Além do conjunto padrão, o conjunto de dados de treino da rede MLP é um conjunto de dados aumentado com a inclusão das observações dos vizinhos de primeira ordem. O intuito é que a própria rede ajuste pesos diferente para as observações dos vizinhos e, conseqüentemente, tenha um aumento na capacidade de predição por conta da autocorrelação espacial presente nos dados. A última etapa para a obtenção do modelo final MLP foi o refinamento dos parâmetros de entrada do algoritmo.

4.5 Avaliação de qualidade dos modelos

A avaliação da qualidade do ajuste de um modelo é uma etapa importante no processo de modelagem. É através da avaliação que é mensurado quão distante está a predição realizada dos dados reais que avalia-se características importante do modelo ajustado como possível superajuste e a sua capacidade de generalização para dados quaisquer. Para tanto, utilizou-se as métricas *root mean squared error* (RMSE) e R^2 (r-quadrado), que são métricas muito utilizadas para regressão.

Assumindo que possui-se n observações cujo valor real é y_i e o valor predito é $\hat{y}_i$, o RMSE é definido como:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}, \quad (4.1)$$

enquanto que o R^2 é calculado como:

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}. \quad (4.2)$$

O RMSE é uma métrica que penaliza os erros de predição mais altos quando comparado a outras métricas como a *Mean Absolute Error* (MAE). Entretanto, por ter a mesma unidade de

medida da variável resposta, permite avaliar a qualidade do ajuste de diferentes modelos com diferentes variáveis preditoras. O R^2 tem em seu numerador a soma dos quadrados explicados, ou seja, é a soma da diferença entre o valor predito e a média dos valores reais. No denominador, tem-se a soma dos quadrados das diferenças entre os valores reais e sua média, também chamado de soma total dos quadrados. Dessa forma, por ser normalizado pela soma total dos quadrados o valor do R^2 varia entre 0 e 1, e quanto maior a soma dos quadrados explicados, melhor ajustado o modelo estará aos dados estando mais próximo de 1.

RESULTADOS

5.1 Análise exploratória

A primeira análise conduzida com os dados foi a avaliação das estatísticas descritivas das distribuições, para cada área, da soma diária dos pedidos realizados. Além disso, avaliou-se graficamente o comportamento dos pedidos totais realizados em cada área para observar se há um padrão geográfico explícito, mais brando ou nenhum, através de mapas cloropléticos.

Figura 10 – Boxplots das distribuições de pedidos por distrito

Fonte: Elaborada pelo autor.

Fonte: Elaborada pelo autor.

Figura 11 – Mapa cloroplético dos pedidos por distrito

Na figura 10, nota-se que a distribuição de pedidos é distinta para cada distrito mas com alguns tendo distribuições similares a outros. Vila Mariana e o Itaim Bibi tendem a ter mais pedidos diários, enquanto que distritos como Bom Retiro e Pari tendem a ter uma menor quantidade de pedidos. Incluindo o apoio da figura 11, observa-se um comportamento que indica a presença de autocorrelação geográfica nos dados do tipo em que áreas próximas parecem se comportar de maneira similar, formando *clusters*.

Antes de avaliar se a autocorrelação espacial presente nos dados é estatisticamente significativa ou não, analisou-se se havia tendência na séries de pedidos diários (figura 12) a fim de evitar um resultado espúrio nos testes I de Moran e C de Geary.

Fonte: Elaborada pelo autor.

Figura 12 – Série de pedidos diários

Para a análise de uma possível tendência na série dos pedidos foi utilizado o teste estatístico não paramétrico de Mann-Kendall (HAMED; RAO, 1998). Considerando um nível de significância de 5%, o p-valor = 0,0809 não permite rejeitar a hipótese nula do teste que diz

que os dados não possuem tendência. Assim, seguiu-se para aplicação dos testes de medida de autocorrelação espacial.

Adotou-se uma matriz de vizinhança construída pelo critério de vizinhança por adjacência (ou também *Queen's contiguity*) de primeira ordem. Assim, com a matriz de pesos escolhida, aplicou-se ambos os testes de Moran e Geary aos dados. Os resultados mostrados na tabela 2 indicam que ambos os testes, a um nível de significância de 5%, mostram que autocorrelação espacial presente nas observações é estatisticamente significativa e portanto não deveria ser ignorada.

Tabela 2 – Estatística e p-valor dos testes aplicados

	Estatística	P-valor
Teste I de Moran	0,5533	<0,001
Teste C de Geary	0,4835	<0,001

5.2 Ajuste dos modelos

Para o ajuste dos modelos SAR e MLP, realizou-se o pré-processamento dos dados com os procedimentos de preenchimento de valores faltantes, criação de novas variáveis a partir das informações já existentes na base de dados e criação de novas variáveis especialmente para o ajuste do MLP, gerando uma base aumentada com a adição dos pedidos dos distritos vizinhos de primeira ordem.

Em algumas regiões, para certas datas, a quantidade de pedidos realizados não estava disponível no banco de dados. Dessa forma, utilizou-se a moda da quantidade de pedidos da região para preencher esses valores faltantes. Além disso, criou-se novas variáveis preditoras com uso de informações já existentes no banco de dados (*feature engineering*).

O ajuste do modelo SAR foi realizado adotando a mesma matriz de pesos utilizada no cálculo das medidas de autocorrelação espacial, ou seja, utilizando o princípio de vizinhança por adjacência. O ajuste do modelo MLP foi realizado primeiramente para o conjunto de treino com um parâmetro fixo para todos os distritos de duas camadas intermediárias com 6 neurônios cada e função de ativação ReLu. O *tunning* de hiperparâmetros foi então realizado tendo como métrica alvo minimizar o RMSE calculado para o conjunto de teste para cada distrito. As métricas de cada modelo para os conjuntos de teste e validação é mostrado na tabela 3. Como *baseline* de comparação, utilizou-se a Média Histórica (MH) da base de treino como predição.

Na tabela 3, nota-se que o desempenho dos três modelos, SAR, MLP-PF e MLP-R são superiores ao da Média Histórica tanto na base de teste quanto na base de validação. Na base de teste e o MLP-R tem o menor RMSE entre os três modelos comparados. Entretanto, enquanto o RMSE é 27,3% menor na base de teste, na base de validação essa diferença reduz para 6,9%. O MLP-R também obteve o melhor R^2 entre os três modelos verificados. Na figura 13, é possível

Tabela 3 – Métricas de avaliação dos ajustes dos modelos

Modelo	Base de Teste		Base de Validação	
	RMSE	R^2	RMSE	R^2
MH	5,589	0	6,516	0
SAR	3,126	0,686	3,604	0,689
MLP - Parâmetro fixo	2,864	0,736	3,510	0,705
MLP - Refinado	2,272	0,834	3,353	0,731

comparar o RMSE para cada distrito obtido por cada modelo na base de validação. Os 3 modelos tiveram desempenho similar na região norte/noroeste do mapa. Porém, para os distritos da região sul, os quais tinham a maior quantidade de pedidos, os modelos MLP tiveram exibiram um erro menor para distrito da Vila Mariana. Além disso, os três tiveram erros altos para o distrito da Saúde, o mais destacado nos mapas.

Fonte: Elaborada pelo autor.

Figura 13 – RMSE por distrito calculado na base de validação

Na tabela 4, tem-se as previsões por distrito para a data de 15/10/2019 realizadas com o MLP-PF, MLP-R e SAR. Nota-se que o SAR tem uma menor variabilidade nas previsões para esta data, enquanto que os dois ajustes da MLP conseguem ter uma previsão com variabilidade maior, em especial para os valores mais próximo do mínimo e do máximo de pedidos observados.

Pelas métricas analisadas de RMSE e R^2 , o modelo escolhido como que melhor se ajustou aos dados foi o *Multilayer Perceptron* com parâmetros refinados para cada um dos distritos.

Tabela 4 – Previsões da demanda de pedidos para o dia 15-10-2019 dos modelos ajustados

DISTRITO	MLP-PF	MLP-R	SAR
ALTO DE PINHEIROS	4	5	4
BARRA FUNDA	3	4	3
BELA VISTA	4	5	8
BELEM	1	2	2
BOM RETIRO	1	2	3
BRAS	2	3	3
CAMBUCI	2	1	2
CONSOLACAO	10	13	10
IPIRANGA	4	4	3
ITAIM BIBI	18	16	12
JD PAULISTA	11	11	9
LAPA	4	4	5
LIBERDADE	6	7	5
MOEMA	12	13	13
MOOCA	6	7	3
PARI	1	1	2
PERDIZES	10	8	9
PINHEIROS	13	12	15
REPUBLICA	5	3	3
SANTA CECILIA	8	8	8
SAUDE	10	14	14
SE	2	1	2
VILA LEOPOLDINA	3	2	2
VILA MARIANA	16	18	13

CONCLUSÃO

No presente capítulo, destinado as conclusões do estudo, reitera-se a importância de que seja considerado que a autocorrelação espacial pode estar presente em dados de natureza geoespacial. Para tanto, especificamente, analisou-se a ocorrência dessa característica em dados de pedidos de uma grande empresa de *online food delivery* do Brasil. Dados esses fundamentais para a predição da demanda de pedidos de uma determinada área geográfica e são úteis para otimização do gerenciamento tanto de governança tanto da rede logística a qual a empresa dispõe para funcionar. Para atingir o objetivo de realizar a predição da demanda de pedidos com o menor erro possível foram propostas duas técnicas: o *Spatial Autorregressive Model* (SAR) e a de Redes Neurais *Multilayer Perceptron*. Além da análise das técnicas de modelagem, analisou-se como identificar se a autocorrelação presente nos dados são estatisticamente significativas ou não.

As duas técnicas apresentadas nesse trabalho para o problema de previsão de demanda por área geográfica tiveram resultados levemente distintos. O ajuste final do SAR mostrou um erro satisfatório quando comparado com a situação em que a média histórica é utilizada como uma predição. Já os dois ajustes finais da MLP, utilizando a abordagem proposta de dados de treino aumentados, exibiram erros menores que a do SAR. A primeira rede com os mesmos hiperparâmetros para todas as regiões, exibiu um erro levemente menor, tendo uma redução relativa do RMSE de 2,7% quando comparado ao SAR. A rede com hiperparâmetros diferentes para cada área teve uma redução mais significativa do erro calculado, de 6,9%. Portanto, para os dados utilizados nesse trabalho, a abordagem com a rede MLP mostrou o menor RMSE e o maior valor de R^2 , sendo a melhor opção encontrada dos testes realizados.

Como sugestão para trabalhos futuros tem-se: aplicar modelos de séries temporais geoespaciais aos dados; estudar com maior profundidade quais alternativas podem ser utilizados para que algoritmos de aprendizado de máquina para regressão sejam capazes de lidar no treino do modelo e considerar quando há autocorrelação espacial nos dados. Pode-se estudar também se ganhos seriam obtidos incluindo informações de vizinhos de ordem superiores à primeira.

REFERÊNCIAS

- ABRASEL. **Delivery movimenta R\$11 bilhões por ano.** 2019. Disponível em: <https://abrasel.com.br/noticias/noticias/delivery-movimenta-r-11-bilhoes-por-ano-enquanto-franquias-de-alimentacao-diversificam-a-oferta-de-produtos/>. Acesso em: 15/11/2019. Citado na página 26.
- AHMAD, A.; HASSAN, M.; ABDULLAH, M.; RAHMAN, H.; HUSSIN, F.; ABDULLAH, H.; SAIDUR, R. A review on applications of ann and svm for building electrical energy consumption forecasting. **Renewable and Sustainable Energy Reviews**, Elsevier, v. 33, p. 102–109, 2014. Citado na página 28.
- ALPAYDIN, E. **Introduction to machine learning**. [S.l.]: MIT press, 2020. Citado nas páginas 41, 45 e 46.
- ANSELIN, L. Spatial econometrics. **A companion to theoretical econometrics**, Wiley Online Library, v. 310330, 2001. Citado na página 28.
- BIVAND, R. S.; PEBESMA, E. J.; GOMEZ-RUBIO, V.; PEBESMA, E. J. **Applied spatial data analysis with R**. [S.l.]: Springer, 2013. v. 2. Citado na página 41.
- CÂMARA, G.; CARVALHO, M. S.; CRUZ, O. G.; CORREA, V. Análise espacial de áreas. **Análise espacial de dados geográficos**, Empresa Brasileira de Pesquisa Agropecuária Planaltina, v. 2, 2004. Citado na página 36.
- CARVALHO, M. L.; NATÁRIO, I. Análise de dados espaciais. **Sociedade Portuguesa de Estatística**, 2008. Citado nas páginas 33, 35 e 36.
- CHANDRA, A. L. **McCulloch-Pitts Neuron — Mankind's First Mathematical Model Of A Biological Neuron**. 2018. Disponível em: <https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1>. Acesso em: 30/05/2021. Citado na página 42.
- CHINA online food delivery market volume doubles in 2018: report. 2019. Disponível em: <https://www.chinadaily.com.cn/a/201904/10/WS5cad9eb8a3104842260b56a1.html>. Acesso em: 15/11/2019. Citado na página 25.
- CLIFF, A. D.; ORD, J. K. **Spatial processes: models & applications**. [S.l.]: Taylor & Francis, 1981. Citado na página 37.
- CRESSIE, N. **Statistics for spatial data**. [S.l.]: John Wiley & Sons, 2015. Citado nas páginas 31, 34, 35 e 40.
- CYBENKO, G. Approximation by superpositions of a sigmoidal function. **Mathematics of control, signals and systems**, Springer, v. 2, n. 4, p. 303–314, 1989. Citado na página 46.
- FACELI, K.; LORENA, A. C.; GAMA, J.; CARVALHO, A. *et al.* Inteligência artificial: Uma abordagem de aprendizado de máquina. **Rio de Janeiro: LTC**, v. 2, p. 192, 2011. Citado nas páginas 41, 43, 44, 45, 46 e 52.

- FUNAHASHI, K.-I. On the approximate realization of continuous mappings by neural networks. **Neural networks**, Elsevier, v. 2, n. 3, p. 183–192, 1989. Citado na página 42.
- GAETAN, C.; GUYON, X. **Spatial statistics and modeling**. [S.l.]: Springer, 2010. v. 90. Citado nas páginas 37 e 38.
- HAMED, K. H.; RAO, A. R. A modified mann-kendall trend test for autocorrelated data. **Journal of hydrology**, Elsevier, v. 204, n. 1-4, p. 182–196, 1998. Citado na página 56.
- HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The elements of statistical learning: data mining, inference, and prediction**. [S.l.]: Springer Science & Business Media, 2009. Citado na página 47.
- HIRSCHBERG, C.; RAJKO, A.; SCHUMACHER, T.; WRULICH, M. The changing market for food delivery. **Mckinsey: New York, NY, USA**, 2016. Citado na página 26.
- HOEF, J. M. V.; PETERSON, E. E.; HOOTEN, M. B.; HANKS, E. M.; FORTIN, M.-J. Spatial autoregressive models for statistical inference from ecological data. **Ecological Monographs**, Wiley Online Library, v. 88, n. 1, p. 36–59, 2018. Citado na página 40.
- HOOTEN, M. B.; HOEF, J. M. V.; HANKS, E. M. Simultaneous autoregressive (sar) model. **Wiley StatsRef: Statistics Reference Online**, Wiley Online Library, p. 1–10, 2014. Citado na página 39.
- HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural networks**, Elsevier, v. 2, n. 5, p. 359–366, 1989. Citado na página 42.
- Hunter, D.; Yu, H.; Pukish, III, M. S.; Kolbusz, J.; Wilamowski, B. M. Selection of proper neural network sizes and architectures—a comparative study. **IEEE Transactions on Industrial Informatics**, v. 8, n. 2, p. 228–240, 2012. Citado na página 43.
- JORDAHL, K.; BOSSCHE, J. V. den; FLEISCHMANN, M.; WASSERMAN, J.; MCBRIDE, J.; GERARD, J.; TRATNER, J.; PERRY, M.; BADARACCO, A. G.; FARMER, C.; HJELLE, G. A.; SNOW, A. D.; COCHRAN, M.; GILLIES, S.; CULBERTSON, L.; BARTOS, M.; EUBANK, N.; MAXALBERT; BILOGUR, A.; REY, S.; REN, C.; ARRIBAS-BEL, D.; WASSER, L.; WOLF, L. J.; JOURNOIS, M.; WILSON, J.; GREENHALL, A.; HOLDGRAF, C.; FILIPE; LEBLANC, F. **geopandas/geopandas: v0.8.1**. Zenodo, 2020. Disponível em: <<https://doi.org/10.5281/zenodo.3946761>>. Citado na página 51.
- KERKMAN, K.; MARTENS, K.; MEURS, H. Predicting travel flows with spatially explicit aggregate models: On the benefits of including spatial dependence in travel demand modeling. **Transportation Research Part A: Policy and Practice**, Elsevier, v. 118, p. 68–88, 2018. Citado na página 30.
- MANSANO, A. T. R.; GORNI, P. M. Satisfação do consumidor com o comércio eletrônico: estudo de caso de uma fabricante de tapetes. **REVISTA DE EXTENSÃO E INICIAÇÃO CIENTÍFICA DA UNISOCIESC**, v. 1, n. 1, p. 12–22, 2014. Citado na página 27.
- MARDIA, K. V. Maximum likelihood estimation for spatial models. In: **Spatial statistics: Past, present and future**. [S.l.]: Monograph, 1990. v. 12, p. 203–253. Citado na página 40.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943. Citado na página 41.

- MINSKY, M.; PAPERT, S. An introduction to computational geometry. **Cambridge tiass., HIT**, 1969. Citado na página 42.
- MORAN, P. A. The interpretation of statistical maps. **Journal of the Royal Statistical Society. Series B (Methodological)**, JSTOR, v. 10, n. 2, p. 243–251, 1948. Citado na página 37.
- NORVIG, P.; RUSSELL, S. Inteligencia artificial. **Editora Campus**, v. 20, 2004. Citado nas páginas 46, 47 e 49.
- OHTSUKA, Y.; OGA, T.; KAKAMU, K. Forecasting electricity demand in japan: A bayesian spatial autoregressive arma approach. **Computational Statistics & Data Analysis**, Elsevier, v. 54, n. 11, p. 2721–2735, 2010. Citado nas páginas 28 e 30.
- OVERDULVE, K. **Logistics in the Food industry: A vision on 2020**. 2016. Disponível em: <https://www2.deloitte.com/be/en/pages/operations/articles/Logistics_Food_Industry_2020Vision.htmls>. Acesso em: 01/11/2019. Citado na página 25.
- PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011. Citado na página 51.
- REY, S. J.; ANSELIN, L. Pysal: A python library of spatial analytical methods. In: **Handbook of applied spatial analysis**. [S.l.]: Springer, 2010. p. 175–193. Citado na página 51.
- REYES, D.; ERERA, A.; SAVELSBERGH, M.; SAHASRABUDHE, S.; O’NEIL, R. The meal delivery routing problem. **Optimization Online**, 2018. Citado na página 27.
- ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958. Citado na página 42.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. **Learning internal representations by error propagation**. [S.l.], 1985. Citado na página 43.
- SANTOS, G.; JUNIOR, J.; BERNARDO, Y. Previsão de demanda: Revisão bibliográfica e análise acadêmica atual. **XXXV ENEGEP, Fortaleza**, 2015. Citado na página 29.
- SINGH, S. **The Soon To Be \$200B Online Food Delivery Is Rapidly Changing The Global Food Industry**. 2019. Disponível em: <<https://www.forbes.com/sites/sarwantsingh/2019/09/09/the-soon-to-be-200b-online-food-delivery-is-rapidly-changing-the-global-food-industry>>. Acesso em: 15/11/2019. Citado na página 25.
- STEEVER, Z.; KARWAN, M.; MURRAY, C. Dynamic courier routing for a food delivery service. **Computers & Operations Research**, Elsevier, v. 107, p. 173–188, 2019. Citado na página 27.
- SUHARTANTO, D.; ALI, M. H.; TAN, K. H.; SJAHROEDDIN, F.; KUSDIBYO, L. Loyalty toward online food delivery service: the role of e-service quality and food quality. **Journal of foodservice business research**, Taylor & Francis, v. 22, n. 1, p. 81–97, 2019. Citado na página 27.

TAYLOR, J. W. Short-term electricity demand forecasting using double seasonal exponential smoothing. **Journal of the Operational Research Society**, Taylor & Francis, v. 54, n. 8, p. 799–805, 2003. Citado na página 27.

ULMER, M. W.; THOMAS, B. W.; CAMPBELL, A. M.; WOYAK, N. **The restaurant meal delivery problem: Dynamic pick-up and delivery with deadlines and random ready times**. [S.l.], 2017. Citado na página 27.

UNITED States \$32.7 Billion Online Food Delivery Market Trends, Share, Size, Growth, Opportunity and Forecast 2019-2024. 2019. Disponível em: <https://www.globenewswire.com/news-release/2019/04/25/1809649/0/en/United-States-32-7-Billion-Online-Food-Delivery-Market-Trends-Share-Size-Growth-Opportunity-and-Forecast-2019-2024.html>. Acesso em: 16/11/2019. Citado na página 25.

VITON, P. A. Notes on spatial econometric models. **City and regional planning**, v. 870, n. 3, p. 2–17, 2010. Citado na página 38.

YANG, Y.; HEPPENSTALL, A.; TURNER, A.; COMBER, A. Using graph structural information about flows to enhance short-term demand prediction in bike-sharing systems. **Computers, Environment and Urban Systems**, Elsevier, v. 83, p. 101521, 2020. Citado na página 30.

YUE, L.; YAFENG, Y.; JUNJUN, G.; CHONGLI, T. Demand forecasting by using support vector machine. In: IEEE. **Third International Conference on Natural Computation (ICNC 2007)**. [S.l.], 2007. v. 3, p. 272–276. Citado na página 28.

REPOSITÓRIO DE ARQUIVOS E CÓDIGOS UTILIZADOS

O banco de dados tratado, o arquivo de polígonos referente aos distritos da cidade de São Paulo - SP e os códigos utilizados para ler, manipular e tratar os arquivos com as informações necessárias, bem como os códigos de ajuste dos modelos e suas simulações, estão disponíveis no repositório: <<https://github.com/HerlissonMB/demand-forecast>> .

CÓDIGOS DE MODELAGEM

Código-fonte 1 – Código de modelagem em Python

```
1: # Importando bibliotecas necessárias
2: from seaborn import palplot, countplot, barplot, boxplot,
    lineplot, kdeplot
3: import geopandas as gpd
4: import matplotlib.pyplot as plt
5: import pandas as pd
6: import shapely as shp
7: import adjustText as aT
8: import numpy as np
9: import seaborn as sns
10:
11: import warnings
12: warnings.filterwarnings("ignore")
13:
14: from pysal.lib import weights
15: from pysal.explore import esda
16:
17: from splot.esda import plot_moran
18:
19: from pysal.model import spreg
20:
21: from sklearn.neural_network import MLPRegressor
22: from sklearn.preprocessing import StandardScaler
23: from sklearn.metrics import mean_squared_error, r2_score
24: from sklearn.model_selection import ParameterGrid
25:
```

```
26: from matplotlib.pyplot import figure
27: from matplotlib import pyplot as plt
28: plt.style.use('ggplot')
29:
30: # Relendo arquivos tratados de geometrias e pedidos para
    continuar
31: distritos_ce = pd.read_csv('distritos_centro_expandido_sp.csv')
32: distritos_ce['geometry']=distritos_ce['geometry'].apply( shp.
    wkt.loads )
33: distritos_ce = gpd.GeoDataFrame( distritos_ce , geometry='
    geometry')
34:
35: #Lendo o arquivo novamente para continuar
36: pedidos_final = pd.read_csv('pedidos_final_ce_sp.csv')
37: pedidos_final['pontos'] = [ shp.geometry.Point(x,y) for x , y
    in zip( pedidos_final['customer_longitude'] , pedidos_final[
    'customer_latitude'] ) ]
38: pedidos_final['data'] = pd.to_datetime(pedidos_final['
    order_timestamp_local']).dt.date.astype('datetime64')
39: pedidos_final = gpd.GeoDataFrame(pedidos_final , geometry = '
    pontos')
40:
41: pedidos_final_agg = pedidos_final.groupby(['NOME_DIST', 'data',
    'order_weekday' , 'order_month']) \
42:     .count()['order_timestamp_local'].reset_index() \
43:     .rename(columns= {'order_timestamp_local': '
    count_orders'})
44:
45: # Aplicacao do teste de tendência Mann-Kendall
46:
47: import pymanuskendall as mk
48: mk.original_test( pedidos_final_agg[filtro_data].groupby('data'
    )['count_orders'].sum().values )
49:
50: # Calculando medida de de autocorrelação espacial de Moran -
    Centro expandido
51:
52: distritos_ce = distritos_ce.sort_values(by='NOME_DIST').
    reset_index(drop=True )
53: wq = weights.contiguity.Queen.from_dataframe(distritos_ce)
54: ax = distritos_ce.plot(edgecolor='k', facecolor='w')
55: wq.plot(distritos_ce, ax=ax,
```

```

56:         edge_kws=dict(color='r', linestyle=':', linewidth=1),
57:         node_kws=dict(marker=''))
58: ax.set_axis_off()
59: resumo['count_orders','mean2'] = weights.spatial_lag.
    lag_spatial(wq, resumo['count_orders']['mean'])
60: wq.transform = 'R'
61: moran = esda.moran.Moran(resumo['count_orders']['mean'], wq)
62: print(f"Valor da estatística I de Moran: {round(moran.I, 4)}\
    np-valor:{moran.p_sim}")
63:
64: # Calculando medida de autocorrelação espacial de Geary -
    Centro expandido
65:
66: geary = esda.geary.Geary(resumo['count_orders']['mean'], wq)
67: print(f"Valor da estatística C de Geary: {round(geary.C, 4)}\
    np-valor:{geary.p_sim}")
68:
69: # Ajuste do modelo SAR
70: reg2 = spreg.ML_Lag(
71:     data_model[['count_orders']].values,
72:     data_model[variable_names].values,
73:     w=wq_model2,
74:     name_y='count_orders',
75:     name_x=variable_names
76: )
77: print(reg2.summary)
78:
79: def predict_sar( reg2 , data_test2 , variable_names , wq):
80:     wq_dim = len(wq.neighbors)
81:     data_test = data_test2.copy()
82:     data_test.sort_values(by = ['data','NOME_DIST'] , inplace =
    True)
83:     data_test.reset_index(drop=True , inplace = True)
84:
85:     df_final = data_test[['data','NOME_DIST','count_orders']]
86:
87:     for dt in df_final['data'].unique() :
88:         xb = np.matmul(pd.concat([pd.Series(np.ones( wq_dim ))
    , data_test[data_test['data'] == dt][variable_names].
    reset_index(drop=True) , pd.Series(np.ones(wq_dim)) ],axis=1
    ).to_numpy(),reg2.betas )

```

```

89:         #xb = np.matmul(pd.concat([pd.Series(np.ones(wq_dim)) ,
            data_test[data_test['data'] == dt][variable_names].
            reset_index(drop=True) ],axis=1 ).to_numpy() , reg2.betas
            [:-1] )
90:         pred = np.matmul( np.linalg.inv( np.identity( wq_dim )
            - reg2.rho * wq.full()[0] ) , xb)
91:
92:         df_final.loc[df_final['data'] == dt, 'pred'] = pred
93:
94:         return df_final
95:
96: preds_train_sar = predict_sar( reg2 , data_model ,
            variable_names , wq ).sort_values(by=['NOME_DIST','data'])
97:
98: # Modelo com RNA MLP
99: #Criando dataframe aumentado para treino
100: df_treino_mpl = data_model[['data','NOME_DIST','count_orders']+
            variable_names]
101: dfs_treino = {}
102:
103: for i in wq.neighbors.keys():
104:     distrito_atual = distritos_ce[distritos_ce.index == i]['
        NOME_DIST'].values[0]
105:
106:     df_aumentado_temp = df_treino_mpl[df_treino_mpl['NOME_DIST'
        ] == distrito_atual].reset_index( drop = True )
107:
108:     lista_vizinhos = wq.neighbors[i]
109:     for j in lista_vizinhos:
110:         df_aumentado_temp[ 'feature_n'+str(j) ] = df_treino_mpl
            [df_treino_mpl['NOME_DIST'] == distritos_ce[distritos_ce.
            index == j]['NOME_DIST'].values[0]]['count_orders_7d'].
            reset_index(drop=True)
111:         dfs_treino[i] = df_aumentado_temp
112: #Criando dataframe aumentado para teste
113: df_teste_mlp = data_test[['data','NOME_DIST','count_orders']+
            variable_names]
114: dfs_teste = {}
115:
116: for i in wq.neighbors.keys():
117:     distrito_atual = distritos_ce[distritos_ce.index == i]['
        NOME_DIST'].values[0]

```

```

118:
119:     df_aumentado_teste = df_teste_mlp[df_teste_mlp['NOME_DIST']
    == distrito_atual].reset_index( drop = True )
120:
121:     lista_vizinhos = wq.neighbors[i]
122:     for j in lista_vizinhos:
123:         df_aumentado_teste[ 'feature_n'+str(j) ] = df_teste_mlp
    [df_teste_mlp['NOME_DIST'] == distritos_ce[distritos_ce.
    index == j]['NOME_DIST'].values[0]]['count_orders_7d'].
    reset_index(drop=True)
124:         dfs_teste[i] = df_aumentado_teste
125:
126: #Criando dataframe aumentado de validação
127: df_valid_mlp = data_valid[['data','NOME_DIST','count_orders']+
    variable_names]
128: dfs_valid= {}
129:
130: for i in wq.neighbors.keys():
131:     distrito_atual = distritos_ce[distritos_ce.index == i]['
    NOME_DIST'].values[0]
132:
133:     df_aumentado_valid = df_valid_mlp[df_valid_mlp['NOME_DIST']
    == distrito_atual].reset_index( drop = True )
134:
135:     lista_vizinhos = wq.neighbors[i]
136:     for j in lista_vizinhos:
137:         df_aumentado_valid[ 'feature_n'+str(j) ] = df_teste_mlp
    [df_teste_mlp['NOME_DIST'] == distritos_ce[distritos_ce.
    index == j]['NOME_DIST'].values[0]]['count_orders_7d'].
    reset_index(drop=True)
138:         dfs_valid[i] = df_aumentado_valid
139:
140: #GridSearch para tuning
141: def GridSearch( df_treino , data_test , params ):
142:
143:     resultados = {'params': [] ,
144:                   'rmse_test': [] ,
145:                   'r2_test' : [] ,
146:                   'r2_train': [] ,
147:                   'rmse_train': []}
148:
149:     tot = len(ParameterGrid( params ))

```

```
150:
151:     for i in ParameterGrid( params ):
152:         MLP = MLPRegressor()
153:         MLP.set_params( **i )
154:
155:         x_train = df_treino.iloc[ :, 3: ]
156:         y_train = df_treino.iloc[ : , 2]
157:
158:         MLP.fit(x_train , y_train)
159:
160:         resultados['rmse_test'].append( mean_squared_error(
data_test.iloc[:, 2] , MLP.predict( data_test.iloc[:, 3:] )
) ** 0.5 )
161:         resultados['r2_test'].append( r2_score(data_test.iloc
[:, 2] , MLP.predict( data_test.iloc[:, 3:] ) ) )
162:         resultados['rmse_train'].append( mean_squared_error(
df_treino.iloc[:, 2] , MLP.predict( df_treino.iloc[:, 3:] )
) ** 0.5 )
163:         resultados['r2_train'].append( r2_score(df_treino.iloc
[:, 2] , MLP.predict( df_treino.iloc[:, 3:] ) ) )
164:         resultados['params'].append( i )
165:
166:     df_resultados = pd.DataFrame(resultados)
167:     df_resultados = df_resultados.sort_values(by = 'rmse_test')
168:
169:     best_params = df_resultados.iloc[0,0]
170:     rmse = df_resultados.iloc[0,1]
171:
172:     return best_params , rmse , df_resultados
173:
174: #Ajustando os modelos MLP - Parametros fixos
175: modelos_MLP = {}
176:
177: losses = []
178: test_performance = []
179: test_performance2 = []
180:
181: for i in range(0, len(dfs_treino) ):
182:
183:     df_treino = dfs_treino[i]
184:
```

```

185:     MLP = MLPRegressor( hidden_layer_sizes=(6,6), activation='
      relu' , solver = 'lbfgs', max_fun=50, tol = 0.001,
      random_state=42)
186:
187:
188:     x_train = df_treino.iloc[ :, 3: ]
189:     y_train = df_treino.iloc[ : , 2]
190:
191:     MLP.fit(x_train , y_train)
192:
193:     losses.append(mean_squared_error(dfs_teste[i].iloc[:, 2] ,
      MLP.predict( dfs_teste[i].iloc[:, 3:] ) ) ** 0.5)
194:     test_performance.append( MLP.score(dfs_teste[i].iloc[:, 3:]
      , dfs_teste[i].iloc[:, 2]) )
195:
196:     modelos_MLP[i] = MLP
197: #Ajustando os modelos MLP - Parametros tunados (refinados)
198: modelos_MLP_ref = {}
199:
200: parametros_busca = { 'hidden_layer_sizes': [ (3) , (6) , (12)
      ,(3,3) , (6,6), (12,12) , (3,3,3) , (6,6,6), (12,12,12) ] ,
201:                      'activation': ['relu','logistic','tanh','
      identity'] ,
202:                      'solver': ['lbfgs'] ,
203:                      'alpha': [0.0001,0.001, 0.01, 0.05] ,
204:                      'max_fun': [10,50,100,200] ,
205:                      'random_state': [42]}
206:
207: for i in range(0, len(dfs_treino) ):
208:
209:     df_treino = dfs_treino[i]
210:
211:     melhores_parametros , _ , _ = GridSearch( df_treino ,
      dfs_teste[i] , parametros_busca)
212:
213:     MLP = MLPRegressor()
214:     MLP.set_params( ** melhores_parametros )
215:
216:     x_train = df_treino.iloc[ :, 3: ]
217:     y_train = df_treino.iloc[ : , 2]
218:
219:     MLP.fit(x_train , y_train)

```

```
220:
221:     modelos_MLP_ref[i] = MLP
222:
223: def predict_mlp( modelos_MLP , dfs_teste , data_test):
224:     tam = len(modelos_MLP)
225:
226:     df_pred = data_test.iloc[:, :3].reset_index(drop = True)
227:
228:     for i in range(0 , tam):
229:         df_pred.loc[df_pred['NOME_DIST'] == dfs_teste[i]['
NOME_DIST']][0] , 'pred'] = modelos_MLP[i].predict( dfs_teste
[i].iloc[:, 3:] )
230:
231:     df_pred['erro'] = df_pred['count_orders'] - df_pred['pred']
232:
233:     return df_pred
234:
235: #Predições com MLP
236: preds_teste_mlp = predict_mlp( modelos_MLP , dfs_teste ,
    data_test)
237: preds_teste_mlp_ref = predict_mlp( modelos_MLP_ref , dfs_teste
    , data_test)
238: preds_valid_mlp = predict_mlp( modelos_MLP , dfs_valid ,
    data_valid)
```