# Modelos Fuzzy de Sistemas Basedos em Níveis e Modelos de Aprendizagem de Máquina

**Leandro Maciel** * **Rosangela Ballini** ** **Fernando Gomide** ***

* *Faculdade de Economia, Administração, Contabilidade e Atuária, Universidade de São Paulo, SP, (e-mail: leandromaciel@usp.br).*
** *Instituo de Economia, Universidade Estadual de Campinas, SP, (e-mail: ballini@unicamp.br)*
*** *Faculdade de Engenharia Elétrica e de Computação, Universidade Estadual de Campinas, SP, (e-mail: gomide@unicamp.br)*

**Abstract:** Linguistic and functional rule-based fuzzy modeling are the two dominant modeling paradigms in fuzzy systems engineering. Level set fuzzy modeling is a third rule-based paradigm whose efficiency and capability to develop data driven models and fuzzy controllers have recently been reported in the literature. It has been shown that the performance of the data driven level set models outperform adaptive neurofuzzy, neural, rule-based, and autoregressive models. This paper revisits data driven level set modeling in front of powerful machine learning models based on ensembles and kernels. The results show that the data driven level set fuzzy modeling surpass ensemble and kernel machine learning approaches as well.

**Resumo**: Modelos linguísticos e funcionais são os modelos predominantes na teoria e aplicações de sistemas fuzzy. Modelos fuzzy de sistemas baseado em níveis é um terceiro paradigma que é igualmente, na maioria das vezes superior, aos modelos linguísticos e funcionais. Este trabalho revisita o método de modelagem fuzzy de sistemas baseado em níveis e o compara com modelos baseados em *ensembles* e *kernel*, dois poderosos paradigmas da área de aprendizagem de máquina. Neste trabalho, os resultados também sugerem a superioridade do método baseado em níveis, confirmando resultados da literatura que indicam o mesmo desempenho frente a outros paradigmas, e.g., modelos baseados em regras fuzzy, redes neurais, redes neurais fuzzy e modelos autoregressivos.

*Keywords:* Data driven level set modeling; machine learning modeling.
*Palavras-chaves:* Modelos fuzzy baseados em níveis; modelos de aprendizagem de máquina

## 1. INTRODUÇÃO

Inteligência computacional e aprendizagem de máquina têm como um de seus objetivos desenvolver métodos que explicitem padrões e relações presentes em um conjunto de dados. O propósito é prover informação para analisar e prever o comportamento de sistemas, o que é importante na decisão em cenários com incerteza, e na análise de tendências.

A literatura frequentemente advoga que a teoria de probabilidade pode ser usada para resolver qualquer problema que envolva incerteza (Murphy, 2012), principalmente quando a aleatoriedade predomina. Em inteligência computacional e aprendizagem de máquina, a incerteza surge em maneiras diferentes, e refere a situações que envolvem informacão imperfeita ou falta de informação.

Enquanto na inteligência computacional predominam métodos de computação flexível, na aprendizagem de máquina predominam métodos rígidos de computação. Apesar do fato de que, na prática, dados e modelos raramente serem exatos, tanto a inteligência computacional como a aprendizagem de máquina usam dados históricos para estimar os parâmetros de modelos de previsão, e para obter modelos que melhor explicam os dados.

Na literatura de inteligência computacional, principalmente na área de modelagem no contexto de sistemas fuzzy, modelos linguísticos e modelos funcionais prevalecem e direcionam a pesquisa e a maioria das aplicações (Jang et al., 1997). Modelos linguísticos e funcionais diferem na forma de seus consequentes. Ambos usam variáveis linguísticas em seus antecedentes, mas enquanto o modelo linguístico preserva o uso de variáveis linguísticas no consequente, o modelo funcional usa uma função das variáveis dos antecendentes nos seus consequentes (Pedrycz and Gomide, 2007). O mecanismo usual para determinar a saída de um modelo linguístico é encontrar o centroide da união dos consequentes das regras ativas (Mamdani, 1974). Modelos linguísticos aditivos somam os consequentes das regras ativas e retornam o centroide da soma (Kosko, 1997). A saída de um modelo funcional é a média do valor das funções das regras ativas ponderada pelo grau de ativação das regras (Sugeno and Takagi, 1983).

Uma terceira alternativa importante para determinar a saída de modelos fuzzy baseados em regras usa o conceito de nível, mais precisamente $\tau$-nível (equivalentemente, $\tau$-

corte) de um conjunto fuzzy (Pedrycz and Gomide, 2007). Intuitivamente, o $\tau$-nível de um conjunto fuzzy $\mathcal{A}$ em um espaço $\mathcal{X}$ é o conjunto de todos os elementos de $\mathcal{X}$ que possuem um grau maior (ou igual) a $\tau$. Para simplificar, os $\tau$-níveis serão chamados de níveis. Esta alternativa foi sugerida por Yager (1991) e estudada por Figueiredo et al. (1993) no contexto de sistemas de controle. Modelos fuzzy baseados em níveis usam uma estratégia para determinar a saída do modelo por meio de uma função que mapeia graus de pertinência a valores do espaço da variável de saída. Na sua forma mais simples, o modelo é aditivo e sua saída é a soma das funções de saída de cada regra ponderada pelos respectivos graus de ativação. Esta estratégia é fundamentalmente diferente daquelas utilizadas por modelos linguísticos e funcionais. Modelos linguísticos e funcionais são aproximadores universais (Kreinovich et al., 1998) assim como são, conforme os experimentos em (Leite et al., 2022) sugerem, os modelos fuzzy baseados em níveis.

Modelos fuzzy de sistemas baseados em níveis (Leite et al., 2022) são competitivos com muitos dos modelos de inteligência computacional e de aprendizagem de máquina (Maciel et al., 2022a,b). Contudo, seu desempenho ainda não foi avaliado e comparado com modelos que hoje estão na frente do estado da arte em aprendizagem de máquina, principalmente modelos baseados em dois poderosos conceitos, *ensemble* e *kernel*, respectivamente.

Neste artigo, modelos fuzzy baseados em níveis são avaliados e comparados com *gradient boosting regression trees*, *kernel ridge*, e *Gaussian process regression* usando os dados e os algoritmos disponíveis no scikit-learn, uma biblioteca de aprendizagem de máquina de código aberto para a linguagem de programação Python (scikit-learn 1.2.1, 2024) . Os resultados sugerem que o desempenho dos modelos fuzzy baseados em níveis é superior aos modelos *gradient boosting regression trees*, *kernel ridge*, e *Gaussian process regression.*

O artigo está organizado da seguinte forma. A próxima seção apresenta o método de modelagem fuzzy usando níveis. A Secão 3 faz uma síntese do modelo *gradient boosting regression trees* e a Seção 4 do modelo *kernel ridge regression.* A Seção 5 trata do modelo *Gaussian process regression.* A Seção 6 resume os resultados computacionais e a Seção 7 conclui o artigo resumindo sua contribuição e sugerindo tópicos para investigação futura.

## 2. MODELOS FUZZY BASEADOS EM NÍVEIS

Considere um modelo fuzzy formado por regras do tipo

$$\mathcal{R}_i: \text{ Se } x \text{ é } \mathcal{A}_i \text{ Então } y \text{ é } \mathcal{B}_i \tag{1}$$

com $i = 1, 2, \ldots, N$, $\mathcal{A}_i$ e $\mathcal{B}_i$ conjuntos fuzzy convexos com funções de pertinência $\mathcal{A}_i(x) : \mathcal{X} \to [0, 1]$ e $\mathcal{B}_i(y) : \mathcal{Y} \to [0, 1]$. Dada uma entrada $x \in \mathcal{X}$, o método original de modelagem fuzzy baseada em níveis é composta pelas seguintes etapas (Yager, 1991).

(1) Calcular o grau de ativação $\tau_i$ de cada regra $\mathcal{R}_i$
$$\tau_i = \mathcal{A}_i(x) \tag{2}$$
(2) Determinar o nível $\mathcal{B}_{\tau_i}$ para cada $\tau_i$
$$\mathcal{B}_{\tau_i} = \{y | \tau_i \leq \mathcal{B}_i(y)\} = [y_{il}, y_{iu}] \tag{3}$$
(3) Calcular o ponto central do intervalo
$$m_i(\tau_i) = \frac{y_{il} + y_{iu}}{2} \tag{4}$$
em que $y_{il}$ e $y_{iu}$ são os limites inferior e superior, respectivamente.
(4) Determinar a saída do modelo $y$ usando a expressão
$$y = \frac{\sum_{i=1}^{N} \tau_i m_i}{\sum_{i=1}^{N} \tau_i} \tag{5}$$

em que $\tau_i > 0$ para, pelo menos, um $i$. Quando o conjunto fuzzy $\mathcal{B}_i$ é discreto, $m_i$ é a média dos elementos de $\mathcal{B}_{\tau_i}$. Observar que a expressão (5) é uma instância de um mapeamento $\mathcal{F} : [0, 1] \to \mathcal{Y}$:

$$\hat{y} = \frac{\sum_{i=1}^{N} \tau_i \mathcal{F}_i(\tau_i)}{\sum_{i=1}^{N} \tau_i} = \mathcal{F}(\tau) \tag{6}$$

em que $\mathcal{F}_i(\tau_i) = m_i(\tau_i)$ é uma das mais simples. A função $\mathcal{F}_i(\tau_i)$ é chamada de função de saída da regra $\mathcal{R}_i$. A Figura 1 mostra a função de saída e a parcela da saída gerada pela i-ésima regra para uma dada entrada $x$. Importante observar que $\mathcal{F}$ mapeia graus de pertinência em elementos do espaço da variável de saída $\mathcal{Y}$, o que difere significativamente dos modelos fuzzy linguísticos e funcionais.

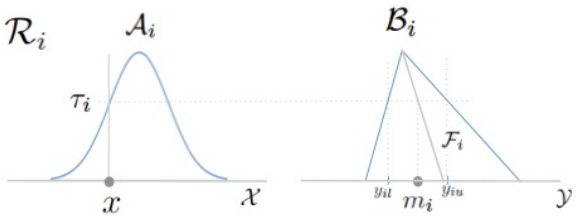


Figura 1. Função de saída da i-ésima regra.

Neste artigo, o particular interesse está em uma extensão do método original em que um conjunto de dados é usado para determinar diretamente as funções de saída $\mathcal{F}_i(\tau_i)$. Esta extensão é uma forma de modelagem baseada em dados conhecida na literatura como *data driven modeling.* No contexto deste artigo, o método de interesse é do tipo *data driven level set modeling.*

Suponha que os conjuntos fuzzy $\mathcal{A}_i(x)$ e $\mathcal{B}_i(y)$ não sejam conhecidos. Dado um conjunto de dados $\mathcal{D} = \{(x^k, y^k)\}, x^k \in R^p, y^k \in R$, tal que $y^k = f(x^k), k = 1, 2, \ldots, K$, o objetivo é obter um modelo fuzzy $\mathcal{F}$ para aproximar a função $f$ usando $\mathcal{D}$ (Leite et al., 2022).

O desenvolvimento de modelos fuzzy linguísticos requer a especificação do número de regras $N$ e as funções de pertinência dos antecedentes e consequentes das regras. O número de regras e as funções de pertinência podem ser identificados usando métodos de agrupamento (Pedrycz and Gomide, 2007). Portanto, quando temos um conjunto de dados, as funções de pertinência dos antecedentes $\mathcal{A}_i$ são determinadas via agrupamento, e estimativas das funções de saída $\mathcal{F}_i$, $i = 1, 2, \ldots, N$ podem ser obtidas diretamente a partir dos dados ao invés de identificar $\mathcal{B}_i$. Um modo direto e eficiente de estimar as funções de saída é supor que $\mathcal{F}_i$ é uma função afim

$$\mathcal{F}_i(\tau_i) = v_i\tau_i + w_i \tag{7}$$

Os valores dos coeficientes $v_i$ e $w_i$ podem ser estimados usando, por exemplo, o método de quadrados mínimos. Por exemplo, Leite et al. (2022) sugere uma solução baseada na inversa generalizada, conforme as seguintes etapas:

(1) Para cada par de dados $(x^k, y^k) \in \mathcal{D}$
(2) Calcular os graus de ativação

$$\tau_i^k = \mathcal{A}_i(x^k), \quad i = 1, \ldots, N$$

(3) Seja

$$\begin{aligned} z^k &= \frac{\tau_1^k(v_1\tau_1^k + w_1)}{s^k} + \ldots + \frac{\tau_N^k(v_N\tau_N^k + w_N)}{s^k} \\ s^k &= \sum_{i=1}^{N} \tau_i^k \\ \mathbf{d}^k &= [(\tau_1^k)^2/s^k, \tau_1^k/s^k, \ldots, (\tau_N^k)^2/s^k, \tau_N^k/s^k] \\ \mathbf{u} &= [v_1, w_1, \ldots, v_N, w_N]^T \end{aligned}$$

(4) Construir as matrizes

$$\begin{aligned} \mathbf{z} &= [z^1, \ldots, z^K]^T \\ \mathbf{D} &= [(\mathbf{d}^1)^T, \ldots, (\mathbf{d}^K)^T]^T \\ \mathbf{y} &= [y^1, \ldots, y^K]^T \end{aligned}$$

(5) Calcular o vetor de coeficientes $\mathbf{u}$

$$\mathbf{u} = \mathbf{D}^+\mathbf{z} \tag{8}$$

$$\tag{9}$$

(6) Determinar a saída do modelo $y$

$$y = \mathbf{d}\mathbf{u} \tag{10}$$

a qual corresponde à (5). $\mathbf{D}^+$ é a Moore-Penrose pseudo inversa de $\mathbf{D}$ (Serre, 2010). O vetor $\mathbf{u}$ é a solução de $min_u\|\mathbf{y} - \mathbf{z}\|^2$.

A complexidade computational para determinar a saída $y$ é limitada pelo cálculo da pseudo inversa $\mathbf{D}^+$. Se $\mathbf{D}$ é uma matriz $(m \times n)$, $m > n$, e $\mathbf{D}^+$ é determinada usando decomposição singular, então a complexidade é $\mathcal{O}(mn^2)$ (Vasudevan and Ramakrishna, 2019; Stanojevic et al., 2022).

## 3. GRADIENT BOOSTING REGRESSION

*Boosting* (Murphy, 2012) é um algoritmo para ajustar modelos adaptativos que usam uma base de funções (*basis functions*) da seguinte forma

$$f(x) = w_0 + \sum_{m=1}^{M} w_m\phi_m(x) \tag{11}$$

sendo $\phi_m(x)$ uma função frequentemente representada na forma paramétrica $\phi_m(x, v_m)$ em que $v_m$ denota a coleção de todos os parâmetros da função. As funções são determinadas usando os dados e geradas por um algoritmo de aprendizagem fraco (*weak learner* ou *base learner*). O algoritmo funciona usando aprendizado fraco sequencialmente para ponderar os dados, atribuindo maior peso aos dados que geram maior erro em iterações anteriores. A apredizagem fraca pode ser feita usando qualquer algoritmo de regressão, mas usualmente o algoritmo de escolha é uma árvore de regressão. *Boosting* pode ser interpretado como uma instância de um algoritmo de gradiente (descendente) no espaço das funções, isto é

$$\hat{\mathbf{f}} = \arg\min_{\mathbf{f}} L(\mathbf{f}), \quad \mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_K)) \tag{12}$$

em que $L$ é uma função objetivo (*loss function*) e $K$ é a cardinalidade do conjunto de dados. *Boosting* pode ser formulado para uma variedade de funções objetivo. Neste artigo $L$ é o quadrado do erro. O módulo sklearn.ensemble do scikit-learn é um módulo representativo do estado da arte para regressão e, por este motivo, é o que será usado para avaliar o desempenho do modelo fuzzy baseado em níveis.

## 4. KERNEL RIDGE REGRESSION

Regressão linear e regressão logística são exemplos de modelos lineares generalizados em que a densidade da variável de saída pertence a uma família de exponenciais, isto é, a média dos parâmetros é uma combinação linear das entradas, eventualmente filtradas por uma função não linear. Uma máquina de *kernel* é um modelo linear generalizado que usa um filtro de entrada do tipo (Murphy, 2012):

$$\phi(x) = (k(x, \mu_1), \ldots, k(x, \mu_C)) \tag{13}$$

em que $\mu_c$, $c = 1, \ldots, C$ é um conjunto de $C$ centróides, e $k(x, c)$ é um *kernel*. *Ridge regression* é um método de quadrados mínimos regularizado (regularização $l_2$). A saída dos modelos é determinada usando a expressão

$$y = \sum_{j=1}^{K} \alpha_j k(x, x_j) \tag{14}$$

em que $\alpha_j$ é a variável dual da função objetivo quadrática regularizada.

## 5. GAUSSIAN PROCESS REGRESSION

Um processo Gaussiano envolve uma distribuição *a priori* sobre funções que pode ser convertida em uma distribuição *a posteriori* sobre funções usando dados (Murphy, 2012). A distribuição *a posteriori* é determinada a partir dos valores da função para um número finito e arbitrário de dados. Ainda, supõe-se que a distribuição conjunta de probabilidades de um conjunto finito de valores da função é Gaussiana com média $\mu(x)$ e covariância $\Sigma(x)$, com $\Sigma_{ij} = k(x_i, x_j)$ e $k(.,.)$ um *kernel*.

Processos Gaussianos podem ser vistos como uma alternativa Bayesiana de métodos baseados em *kernel*. *Ridge regression* e processos Gaussianos usam o *kernel trick* para estimar os parâmetros do modelo a partir do conjunto de dados. Apesar disso, os problemas de aprendizagem de máquina que os dois métodos resolvem são distintos. *Kernel ridge regression* determina a funcão que minimiza a funcão objetivo (erro quadrático médio). Ao invés de

determinar uma única função, regressão com processos Gaussianos usa uma abordagem probabilística, sendo a distribuição *a posteriori* das funções definida pelo teorema de Bayes. Isto é, probabilidades *a priori* sobre funções são combinadas com funções de verossimilhança definidas a partir dos dados para estimar as distribuições *a posteriori*.

## 6. RESULTADOS COMPUTACIONAIS

Esta seção trata de probemas de aproximação funcional com exemplos que usam dados sintéticos gerados por funções adotadas na literatura de apredizagem de máquina para avaliar e comparar os modelos *gradient boosting regression*, *kernel ridge regression* e *Gaussian process regression* com o modelo fuzzy baseado em níveis.

### *6.1 Modelo Fuzzy Baseado em Níveis e Gradient Boosting*

Este exemplo usa os dados gerados por uma funcão de duas variáveis implementada na biblioteca scikit-learn 1.2.1 de 2024 disponível em Python (scikit-learn 1.2.1, 2024) dada por:

$$y = 10sen(\pi x_1 x_2) + 20(x_1 - 0,5)^2 + 10x_1 + 5x_2 \quad (15)$$

Os dados foram gerados usando valores das variáveis de entrada escolhidas aleatoriamente no intervalo [0,1] com distribuição uniforme. Do total de 600 dados gerados, 500 foram usados para treinamento e 100 para teste. O modelo fuzzy baseado em níveis (MFN) é aqui comparado com o modelo implementado no módulo Gradient Boosting Regressor (GBR) do scikit-learn. O erro médio quadrático (MSE) para GBR é 3,86 e para o MFN é 2,24.

As Figuras 2 e 3 mostram os valores da função computados para os dados de teste e os obtidos pelos modelos GBR e MFN, respectivamente. Além do valor inferior do MSE, os gráficos, visualmente, confirmam a superioridade do modelo fuzzy baseado em níveis frente ao *gradient boosting*.

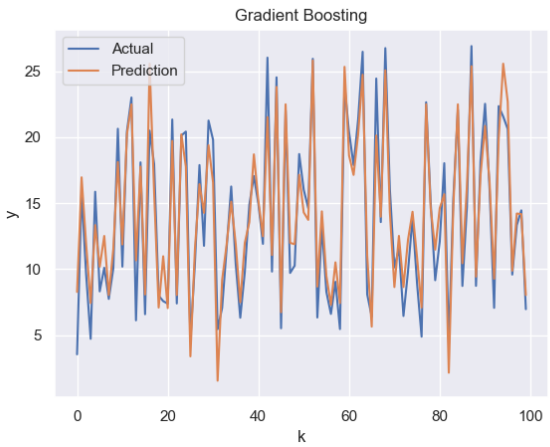


Figura 2. Gradient boosting regression.

### *6.2 Modelo Fuzzy Baseado em Níveis, Kernel Ridge e Gaussian Process*

Este exemplo usa um conjunto de dados sintéticos gerados por uma função senoidal mais um ruído aleatório. O

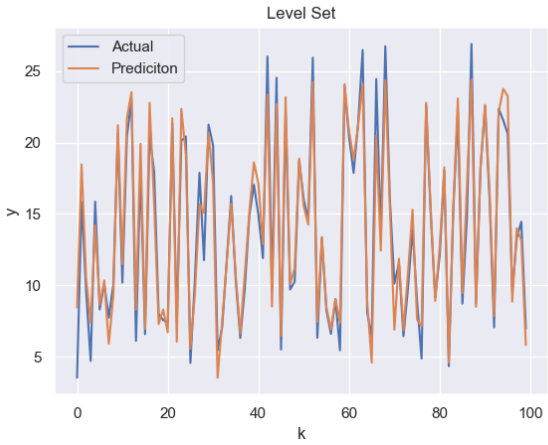


Figura 3. Modelo fuzzy baseado em níveis.

período é $2\pi$ e o ruído é gerado por uma distribuição normal $N(0,1)$. Precisamente, o problema é aproximar a função $y = sen(x)$ usando 40 pontos com valores gerados aleatoriamente. Um total de 400 pontos foram usados para avaliar e comparar o *kernel ridge regressor* (KRR) e o *Gaussian process regressor* (GPR) (scikit-learn 1.2.1, 2024) com o modelo fuzzy baseado em níveis (MFN). A Figura 4 mostra a função senoidal e os dados selecionados (pontos azuis) para o treinamento.

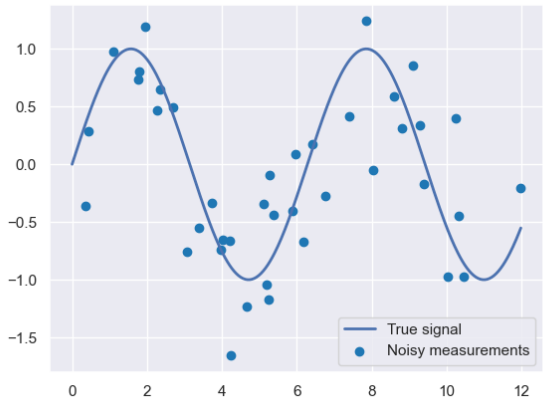


Figura 4. Função original e dados de treinamento.

Neste caso, o MSE do KRR é 0,084, do GPR é 0,054 e do MFN é 0,053, o que mostra que MFN é superior ao KRR e competitivo com o GPR. Contudo, neste ponto devemos enfatizar que a implementação de KRR e GPR feita em scikit-learn 1.2.1 (2024) usa *kernels* periódicos (*exponential sine squared kernel*) para reproduzir o comportamento periódico da função, e sintoniza os parâmetros dos *kernels* fazendo uma busca. O MFN não usa outra informação a não ser aquela que está contida nos dados de treinamento. Isto significa que tanto o KRR como o GPR implementados no scikit-learn 1.2.1 (2024) se beneficiam do conhecimento sobre a natureza dos dados, o que não acontece com o MFN, como pode ser visualizado nas Figuras 5 e 6.

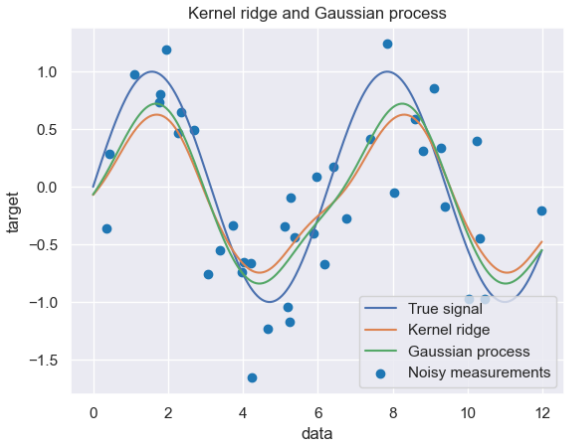


Figura 5. Kernel ridge e processos Gaussianos.

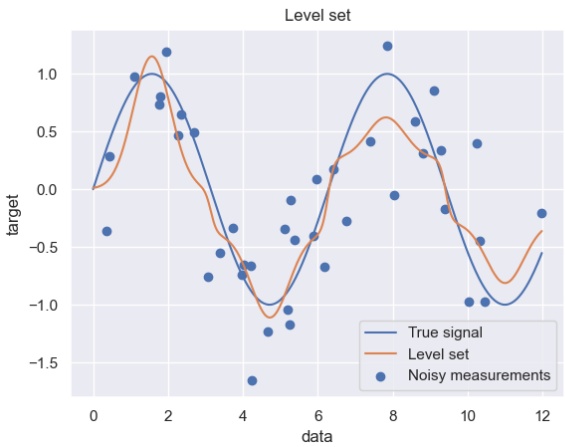


Figura 6. Modelos fuzzy baseados em níveis.

## 7. CONCLUSÃO

Este trabalho apresentou um método para obter modelos fuzzy baseados em níveis, um procedimento importante da inteligência computacional cuja eficiência contrasta significativamente com os modelos fuzzy predominantes, modelos linguísticos e funcionais. O modelo fuzzy baseado em níveis foi avaliado e comparado com o *gradient boosting regression trees*, *kernel ridge*, e *Gaussian process regression* usando dados e algoritmos disponíveis no scikit-learn, uma biblioteca de aprendizagem de máquina de código aberto escrita em Python. Os resultados mostram que o desempenho dos modelos fuzzy baseados em níveis sob o ponto de vista do erro quadrático médio é superior aos dos modelos *gradient boosting regression trees*, *kernel ridge*, e *Gaussian process regression*. Trabalhos futuros deverão tratar da análise de desempenho do método baseado em níveis quando o conhecimento sobre o processo de geração de dados é considerado na modelagem. Uma análise de como o método se comporta em espaços de grande dimensão também é um assunto para ser tratado no futuro.

## AGRADECIMENTOS

Os autores agradecem os revisores pelos comentários e sugestões que ajudaram a melhorar este trabalho, ao CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico, pelo apoio via processos 304456/2020-9, 311260/2022-5 e 302467/2019-0, e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo 2020/09838-0.

## REFERÊNCIAS

Figueiredo, M., Gomide, F., and Yager, R. (1993). Comparison of Yager's level set method for fuzzy logic control with Mamdani's and Larsen's methods. *IEEE Transactions on Fuzzy Systems*, 1, 156–159.

Jang, R., Sun, C., and Mizutani, E. (1997). *Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, Upper Saddle River.

Kosko, B. (1997). *Fuzzy Engineering*. Prentice Hall, Upper Saddle River.

Kreinovich, V., Mouzouris, G., and Nguyen, H. (1998). Fuzzy rule based modeling as a universal approximation tool. In H. Nguyen and M. Sugeno (eds.), *Fuzzy Systems: Modeling and Control*, chapter 4, 135–195. Kluwer, Boston.

Leite, D., Gomide, F., and Yager, R. (2022). Data driven fuzzy modeling using level sets. *Proc. of the FUZZ-IEEE 2022*, 18–23.

Maciel, L., Ballini, R., and Gomide, F. (2022a). Data driven level set method in fuzzy modeling and forecasting. In S. Dick, V. Kreinovich, and P. Lingras (eds.), *Applications of Fuzzy Techniques*, volume 500, 125–134. Springer, Cham.

Maciel, L., Ballini, R., Gomide, F., and Yager, R. (2022b). Forecasting criptocurrencies prices using data driven level set fuzzy models. *Expert Systems with Applications*, 210, 118387.

Mamdani, E. (1974). Applications of fuzzy algorithms for control of simple dynamic plant. *Proc. of the IEEE 2022*, 121, 1585–1588.

Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge.

Pedrycz, W. and Gomide, F. (2007). *Fuzzy Systems Engineering: Toward Human-centric Computing*. Wiley IEEE Press, Hoboken.

scikit-learn 1.2.1 (2024). https://scikit-learn.org/stable.

Serre, D. (2010). *Matrices: Theory and Applications*. Springer, New York.

Stanojevic, V., Kazakovtsev, L., Stanimirovic, P., Rezovz, N., and Shkaberina, G. (2022). Calculating the Moore-Penrose generalized inverse on massively paralell systems. *Algorithms*, 15, 348.

Sugeno, M. and Takagi, T. (1983). A new approach to design of fuzzy controller. In P. Wang (ed.), *Advances in Fuzzy Sets, Possibility Theory and Applications*, 325–334. Plenum Press, New York.

Vasudevan, V. and Ramakrishna, M. (2019). A hierarchical singular value decomposition algorithm for low rank matrices. *arXiv:1710.02812v2*.

Yager, R. (1991). An alternative procedure for the calculation of fuzzy logic controller values. *Journal of the Japanese Society for Fuzzy Theory and Systems*, 4, 736–746.