

[MT04] Assimilação de dados por aprendizado de máquina

Assimilação de Dados por Aprendizado de Máquina

Haroldo F. de Campos Velho – INPE

Helaine C. M. Furtado – UFOPA

Juliana A. Anochi – INPE

Roberto P. Souto – LNCC

Gerônimo Lemos – INPE

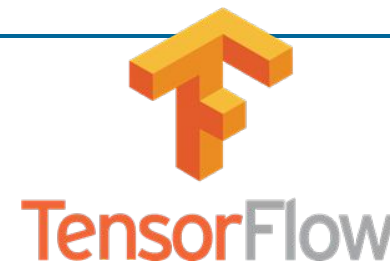
Marcelo Paiva - INPE

Mini-curso: Assimilação de Dados por Redes Neurais

- O que é "assimilação de dados"?
 - O porque da necessidade e breve histórico
- Métodos de assimilação de dados
 - *Nudging* e Métodos Variacionais
 - Filtro de Kalman e filtro de Kalman por conjunto
- Aprendizado de máquina
 - Breve descrição: MLP, Recorrente, Deep Learning
 - Redes Neurais e pacote TensorFlow
 - Árvore de decisão: Boosting e pacote XGBoost
- Aplicações
 - Modelos de baixa ordem: Lorenz-63, *shallow water* 1D e 2D
 - Processamento paralelo para assimilação com redes neurais
 - Modelos atmosféricos 3D: WRF (regional), SPEED e FSU (globais)



Deep Learning com TensorFlow

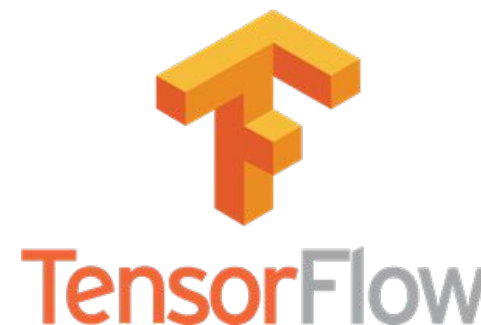


- TensorFlow é uma biblioteca de código aberto para aprendizado de máquina aplicável a uma ampla variedade de tarefas.
- Criada pelo Google em 2015 é a principal biblioteca para criação e treinamento de RNAs.
- A API toda é escrita em Python mas é executada em C++ na CPU ou em CUDA na GPU.

TensorFlow



- Lançamento do TensorFlow como código aberto ocorreu em novembro de 2015
- A versão 1.0 foi lançada em fevereiro de 2017.
- Em janeiro de 2018, o Google anunciou o TensorFlow 2.0.
- <https://www.tensorflow.org>



Keras



- Keras facilita a construção e o treinamento de RNAs
- Fornece uma ampla variedade de RNAs pré-treinadas que podem ser usadas para várias tarefas.
- Ele foi desenvolvido com foco em **permitir a experimentação rápida**.
- Dowload: <https://keras.io/>
- Exemplos:
- <https://keras.io/examples/>

Keras: Algoritmos de Otimização

- Keras possui diversos:
 - **SGD: *Stochastic Gradient Descent***
 - **SGD com Momento:** SGD com Momento usando a derivada (ou gradiente) do ponto atual
 - **SGD com Momento Nesterov:** SGD com Momento mas usa a derivada (ou o gradiente) parcial do ponto seguinte (Nesterov, 1983)
 - **RMSprop:** SGD com taxa de aprendizagem adaptativa (Hinton, Srivastava & Swersky, 2012) -- RMSProp
 - **AdaGrad:** SGD com taxa de aprendizagem adaptativa (Duchi, Hazan, & Yoram, 2011) -- AdaGrad
 - **Adam:** SGD com taxa de aprendizagem adaptativa e momento (Kingma, Diederick & Jimmy, 2014) --- ADAM
- Os mais usados são o SGD e o Adam

TensorFlow + Keras

- Trabalhar com o TF requer saber criar e usar funções:
 - Funções de otimização (Ex: Gradiente descendente)
 - Funções de custo (Ex: MSE)
 - Funções de ativação (Ex: Relu)
- Design da rede (camadas, neurônios, arquitetura) é essencial!
- Conceito de épocas, custo, taxa de aprendizado, pesos e bias.

Keras



- <https://keras.io/examples/>



Star 57,243

About Keras

Getting started

Developer guides

Keras API reference

Code examples

Computer Vision

Natural Language Processing

Search Keras documentation...



» Code examples / Timeseries / Timeseries forecasting for weather prediction

Timeseries forecasting for weather prediction

Authors: Prabhanshu Attri, Yashika Sharma, Kristi Takach, Falak Shah

Date created: 2020/06/23

Last modified: 2020/07/20

Description: This notebook demonstrates how to do timeseries forecasting using a LSTM model.

 [View in Colab](#)

 [GitHub source](#)

Otimização de Hiperparâmetros - Optuna

- Um framework otimização de hiperparâmetros de código aberto para automatizar a pesquisa de hiperparâmetros;
- Um problema de otimização;
- Seu uso é independente de framework. Você pode usá-lo com qualquer estrutura de aprendizado de máquina (Pytorch /Tensorflow+Keras, etc).

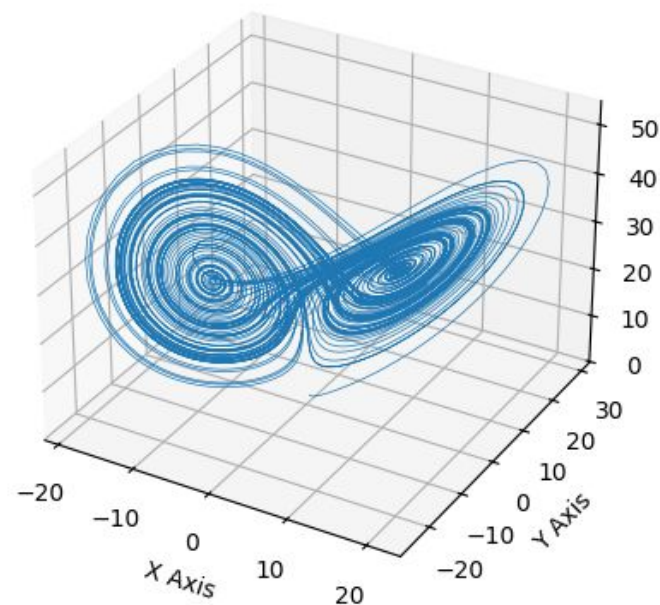
Exemplo Prático - Atrator de Lorenz

- Vamos emular o Atrator de Lorenz utilizando uma Rede Neural *Fully-Connected*;
- Nossa implementação utilizará Tensorflow + Keras para a definição da estrutura da rede;
- Vamos utilizar o framework Optuna para otimização dos hiperparâmetros;
- Link para o notebook: [Google Colab - Lorenz NN](#)

Exemplo Prático - Atrator de Lorenz

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y - x) \\ \frac{dy}{dt} &= x(r - z) - y \\ \frac{dz}{dt} &= xy - \beta z,\end{aligned}$$

Atrator de Lorenz



Mini-curso: Assimilação de Dados por Redes Neurais

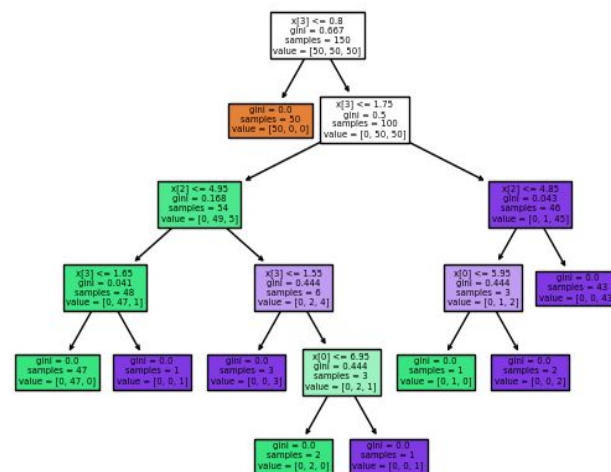
■ Árvores de Decisão

- Árvores de Decisão (do inglês, *Decision Trees*) são um algoritmo de Aprendizado de Máquina, para problemas de **Classificação e Regressão**;
- No caso de assimilação de dados, a árvore de decisão é um algoritmo de regressão.
- Esse algoritmo cria nós de decisão a partir dos dados (atributos de entrada e referência de saída). A ideia principal é, em cada nó, dividir os exemplos da melhor forma possível (separar o máximo possível exemplos de classes diferentes, por exemplo);
- O algoritmo se baseia em um cálculo de ganho de informação para construir a árvore (hierarquia dos atributos de entrada);

Mini-curso: Assimilação de Dados por Redes Neurais

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	
1	4.9	3.0	1.4	
2	4.7	3.2	1.3	
3	4.6	3.1	1.5	
4	5.0	3.6	1.4	

Decision tree trained on all the iris features



Fonte: <https://scikit-learn.org/stable/modules/tree.html#>

Mini-curso: Assimilação de Dados por Redes Neurais

- Um pouco de intuição...



Exemplo retirado de: [Medium Prof. Ricardo Araujo](#)

Mini-curso: Assimilação de Dados por Redes Neurais

- Regras do Cara-a-Cara:
- Cada jogador recebe um tabuleiro com diversos personagens.
- Cada um escolhe, de forma secreta, um personagem;
- Os jogadores se alternam fazendo perguntas que possam ser respondidas com "sim" ou "não";
- Geralmente relacionadas a características físicas ou acessórios (por exemplo: "O seu personagem usa óculos?" ou "Ele tem barba?").

Mini-curso: Assimilação de Dados por Redes Neurais

- Uma boa estratégia é tentar realizar o menor número de perguntas possível;
- Mas como fazer isso? R.: Fazendo as perguntas que eliminem o maior número de suspeitos, independente da resposta;
- A mesma estratégia pode ser usada para construir a Árvore de Decisão;
- ***Ganho de Informação***

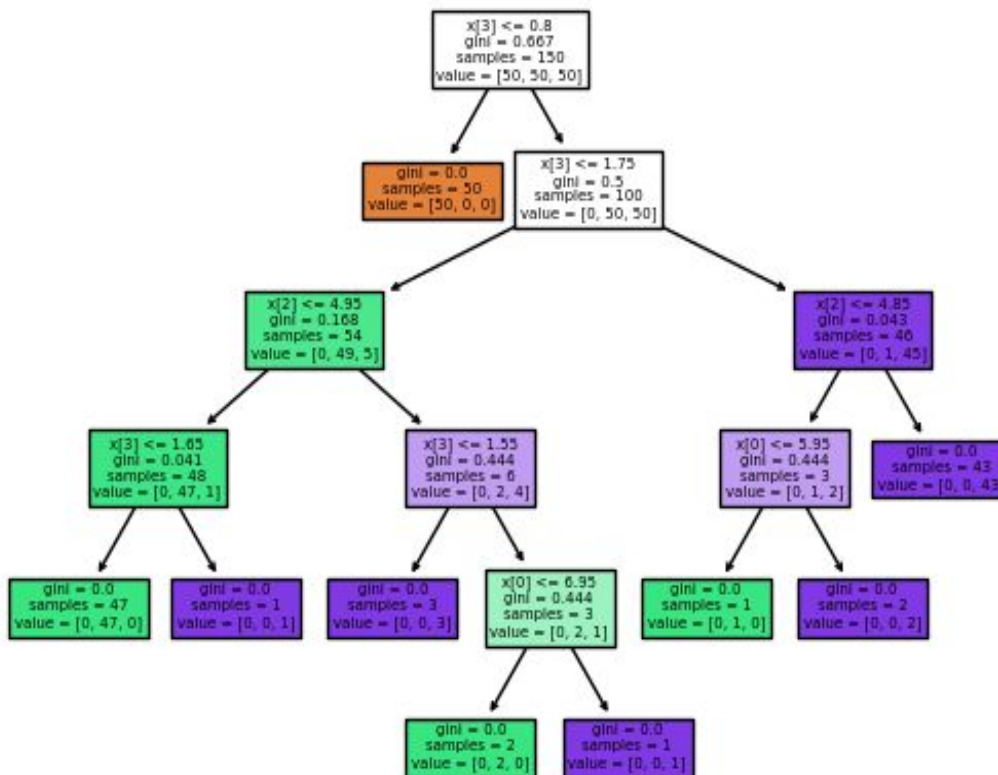


Mini-curso: Assimilação de Dados por Redes Neurais

- Como construir a Árvore de Decisão - ID3
 - O ID3, o primeiro algoritmo de construção das Árvores de Decisão, foi proposto em 1986, por Ross Quinlan;
 - Esse algoritmo é baseado em ***ganho de informação***;
 - Exemplo considerando um problema de classificação binária: Estando em um nó da Árvore (digamos a raiz), qual atributo separa os dados da melhor forma naquele nó da árvore?
 - O ID3 calcula o ganho de informação para cada atributo categórico do conjunto de dados.

Mini-curso: Assimilação de Dados por Redes Neurais

Decision tree trained on all the iris features



Fonte: <https://scikit-learn.org/stable/modules/tree.html#>

Mini-curso: Assimilação de Dados por Redes Neurais

- Como calcular o ganho de informação? .

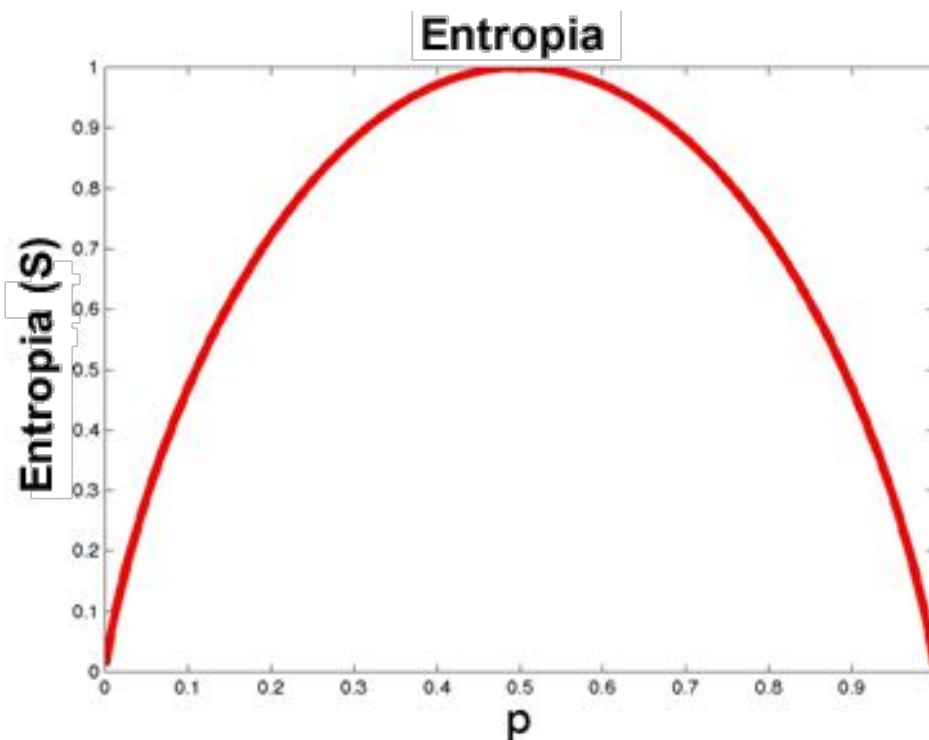
Resposta: Cálculo de **Entropia** de informação.

$$H(S) = \sum_{x \in X} -p(x) \log_2 p(x)$$

- Essa medida quantifica a importância relativa de cada nó da Árvore.
 - O atributo escolhido será o que tem a menor entropia.

Mini-curso: Assimilação de Dados por Redes Neurais

- No nosso exemplo de classificação binária, o gráfico da entropia se comporta da seguinte forma:



Mini-curso: Assimilação de Dados por Redes Neurais

- Sabendo a medida de entropia para os atributos, podemos calcular o ganho de informação:

$$IG(S, A) = H(S) - \sum_{t \in T} p(t)H(t) = H(S) - H(S|A).$$

- O ganho de informação é medido pela diminuição da entropia do nó após a divisão do conjunto de dados **S** dados utilizando o atributo **A**.

Mini-curso: Assimilação de Dados por Redes Neurais

- O ID3 suporta apenas variáveis categóricas;
- Com isso, foi necessário o desenvolvimento de um novo algoritmo para suportar variáveis contínuas;
- O algoritmo proposto foi o C4.5;
 - Esse algoritmo segue a mesma lógica do ID3, com a adição do suporte à variáveis contínuas.
 - Para fazer isso, o algoritmo cria intervalos numéricos como condição de separação;
 - Ou seja, o algoritmo seleciona um threshold para a variável selecionada no nó e separa os exemplos comparando quais tem o atributo com valor maior ou menor que o limiar.
- O algoritmo CART (Classification and Regression Trees) é uma alteração do C4.5 que suporta targets contínuos.

Mini-curso: Assimilação de Dados por Redes Neurais

- Árvores de Decisão são modelos que têm um bom desempenho em dados estruturados (tabulares);
- Modelo “Caixa Branca”: é possível interpretar a saída do modelo. Examina-se as condições estabelecidas (atributo e threshold) de cada nó e é possível alcançar o nó folha e previsão do modelo;
- Problema de overfit: os modelos tendem a ajustar aos dados muito rapidamente;
 - Quanto maior a árvore, maior a chance de *overfit*;
 - Como mitigar esse problema: aprendizado por ***ensemble***.

Mini-curso: Assimilação de Dados por Redes Neurais

■ Aprendizado por *Ensemble*

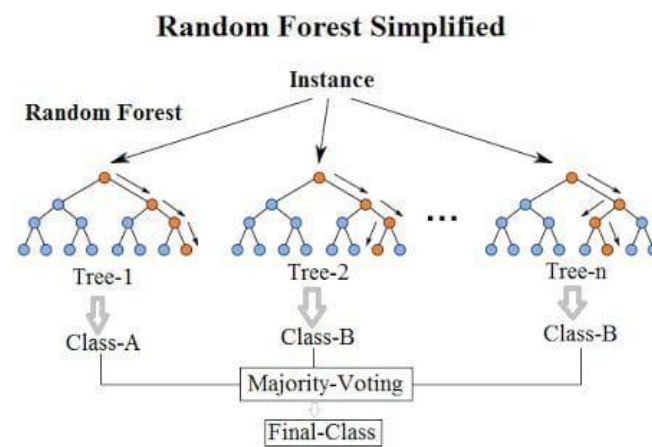
- Técnica de aprendizado de máquina que utiliza vários modelos (*weak learners*) para solucionar o mesmo problema;
- Existem dois tipos de aprendizado por ensemble: ***bagging*** e ***boosting***;

- ***Bagging***: várias instâncias do mesmo modelo, tipicamente Árvores de Decisão, são treinados;

- Os exemplos do conjunto de treinamento são escolhidos com repetição (Bootstrap Aggregating);

- A predição do modelo é a média das predições de cada um dos modelos;

- Ex.: **Random Forest**.



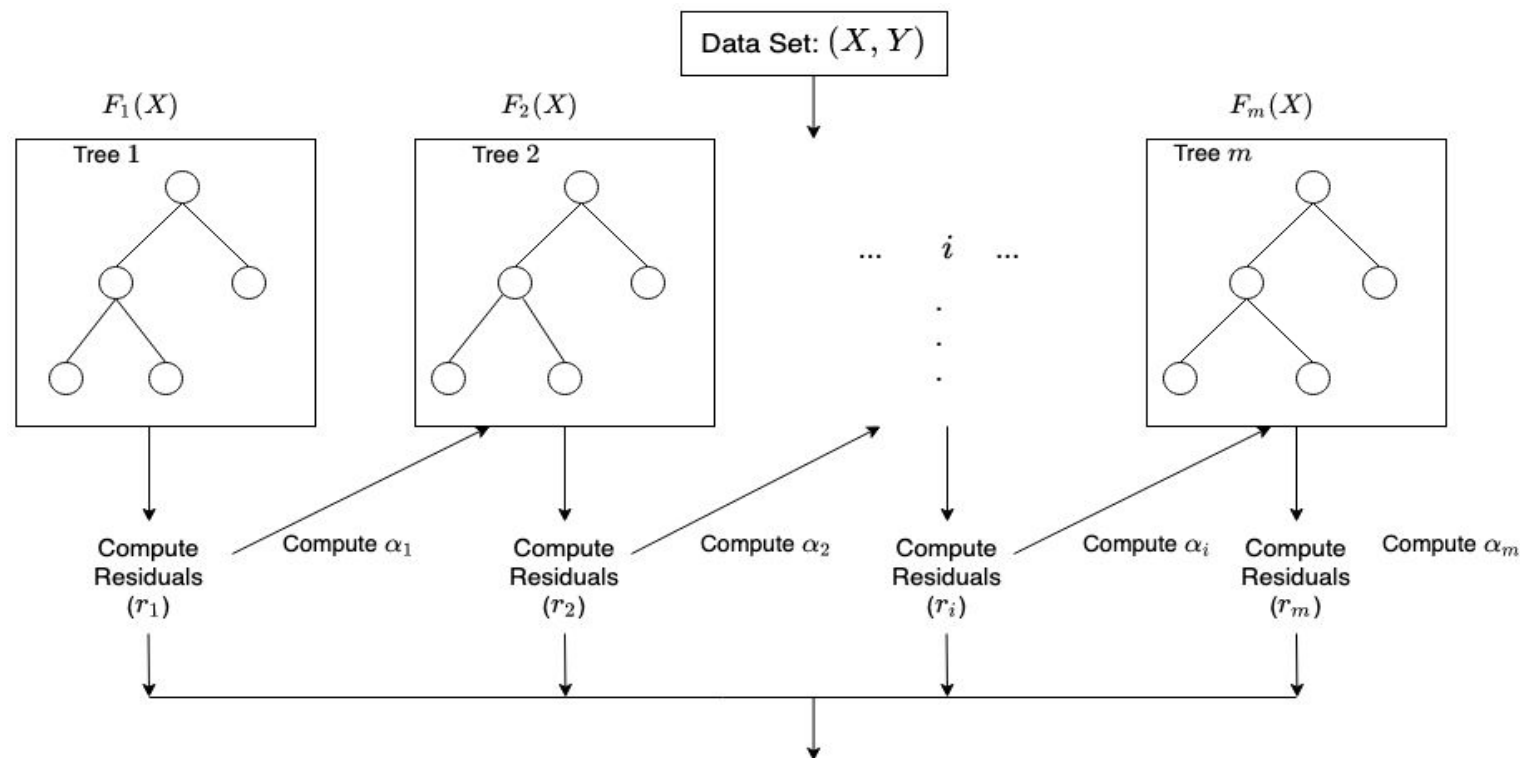
Fonte: <https://www.kdnuggets.com/2017/10/random-forests-explained.html>

Mini-curso: Assimilação de Dados por Redes Neurais

■ **Aprendizado por *Ensamble***

- Boosting: Pesos são atribuídos aos exemplos de treinamento;
- Além disso, cada modelo recebe o resíduo (erro) do modelo anterior;
- Com isso, o objetivo é dar ênfase às previsões erradas do modelo anterior;
- Existem várias implementações desse algoritmo disponíveis para utilização
 - Gradient Boosting (scikit-learn);
 - XGBoost - objeto de estudo deste mini-curso. LightGBM, AdaBoost e CatBoost são outras implementações.
- XGboost (eXtreme **G**radient **B**oosting) é uma implementação otimizada do algoritmo Gradient Boosting apresentado originalmente por Chen e Guestrin (2016).

Mini-curso: Assimilação de Dados por Redes Neurais



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i , and r_i are the regularization parameters and residuals computed with the i^{th} tree respectively, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals

computed, r_i and compute the following: $\arg \min_{\alpha} = \sum_{i=1}^m L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where

$L(Y, F(X))$ is a differentiable loss function.

Mini-curso: Assimilação de Dados por Redes Neurais

- **Parâmetros importantes:**
- **max_depth:** altura máxima de cada um dos modelos treinados;
- **learning_rate:** tamanho do passo a cada iteração de otimização do modelo;
- **n_estimators:** número de modelos treinados;
- **tree_method:** forma de construção das árvores
- **sampling_method:** método de amostragem do conjunto de treinamento. O padrão é uniforme.

Mini-curso: Assimilação de Dados por Redes Neurais

- Link para o artigo original:
<https://arxiv.org/pdf/1603.02754.pdf>
- Documentação oficial:
<https://xgboost.readthedocs.io/en/stable/index.html>
- "Link" para o "notebook" do Colab-Google:
<https://colab.research.google.com/drive/1W-LLnfOBFwSQLGmF2flfY4AQ1vx4Bi5f?usp=sharing>

XGBoost: A Scalable Tree Boosting System

Tianqi Chen
University of Washington
tqchen@cs.washington.edu

Carlos Guestrin
University of Washington
guestrin@cs.washington.edu

ABSTRACT

Tree boosting is a highly effective and widely used machine learning method. In this paper, we describe a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. We propose a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning. More importantly, we provide insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, XGBoost scales beyond billions of examples using far fewer resources than existing systems.

Keywords

Large-scale Machine Learning

1. INTRODUCTION

Machine learning and data-driven approaches are becoming very important in many areas. Smart spam classifiers protect our email by learning from massive amounts of spam data and user feedback; advertising systems learn to match the right ads with the right context; fraud detection systems protect banks from malicious attackers; anomaly event detection systems help experimental physicists to find events that lead to new physics. There are two important factors that drive these successful applications: usage of effective (statistical) models that capture the complex data dependencies and scalable learning systems that learn the model of interest from large datasets.

Among the machine learning methods used in practice, gradient tree boosting [10]¹ is one technique that shines

problems. Besides being used as a stand-alone predictor, it is also incorporated into real-world production pipelines for ad click through rate prediction [15]. Finally, it is the de-facto choice of ensemble method and is used in challenges such as the Netflix prize [3].

In this paper, we describe XGBoost, a scalable machine learning system for tree boosting. The system is available as an open source package². The impact of the system has been widely recognized in a number of machine learning and data mining challenges. Take the challenges hosted by the machine learning competition site Kaggle for example. Among the 29 challenge winning solutions³ published at Kaggle's blog during 2015, 17 solutions used XGBoost. Among these solutions, eight solely used XGBoost to train the model, while most others combined XGBoost with neural nets in ensembles. For comparison, the second most popular method, deep neural nets, was used in 11 solutions. The success of the system was also witnessed in KDDCup 2015, where XGBoost was used by every winning team in the top-10. Moreover, the winning teams reported that ensemble methods outperform a well-configured XGBoost by only a small amount [1].

These results demonstrate that our system gives state-of-the-art results on a wide range of problems. Examples of the problems in these winning solutions include: store sales prediction; high energy physics event classification; web-text classification; customer behavior prediction; motion detection; ad click through rate prediction; malware classification; product categorization; hazard risk prediction; massive online course dropout rate prediction. While domain dependent data analysis and feature engineering play an important role in these solutions, the fact that XGBoost is the consen-

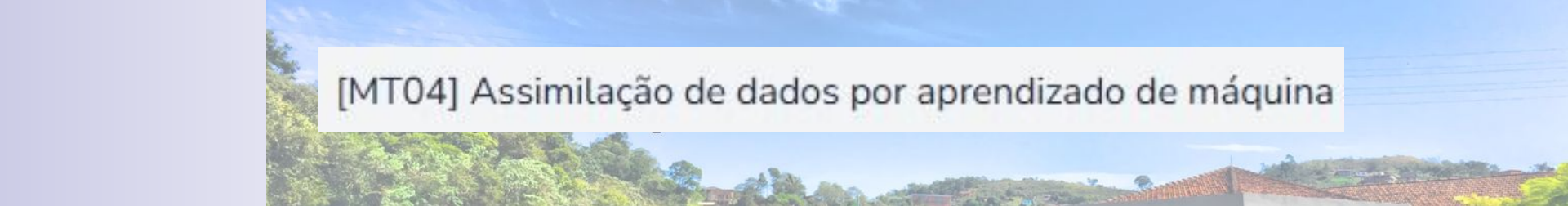
arXiv:1603.02754v3 [cs.LG] 10 Jun 2016

Referências

- <https://scikit-learn.org/stable/modules/tree.html>
- https://en.wikipedia.org/wiki/ID3_algorithm
- <http://web.tecnico.ulisboa.pt/ana.freitas/bioinformatics.ath.cx/bioinformatics.ath.cx/indexf23d.html?id>
- https://www.youtube.com/watch?v=_L39rN6gz7Y
- https://en.wikipedia.org/wiki/C4.5_algorithm
- https://en.wikipedia.org/wiki/Decision_tree_learning
- <https://ricardomatsumura.medium.com/aprendizado-por-ensembles-7cf80117446>
- <https://ricardomatsumura.medium.com/aprendizado-com-%C3%A1rvores-de-decis%C3%A3o-73d874664d1>

Referências

- <https://www.kdnuggets.com/2017/10/random-forests-explained.html>
- <https://xgboost.readthedocs.io/en/stable/>
- Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106
- CHEN, Tianqi; GUESTRIN, Carlos. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016. p. 785-794.



[MT04] Assimilação de dados por aprendizado de máquina



Assimilação de Dados por Redes Neurais Artificiais

Haroldo F. de Campos Velho – INPE

Helaine C. M. Furtado – UFOPA

Juliana A. Anochi – INPE

Roberto P. Souto – LNCC

Gerônimo Lemos – INPE

Marcelo Paiva - INPE