Busca de partículas exóticas com árvores

Guilherme Akira Demenech Mori[1], **Bruno Bogaz Zarpelão**[1], **Sylvio Barbon Junior**[2]

[1]Departamento de Computação – Universidade Estadual de Londrina (UEL)
Rodovia Celso Garcia Cid, PR 445 Km 380 – 86.055-900 – Londrina – PR – Brazil

[2]Department of Engineering and Architecture – University of Trieste (UNITS)
Via Alfonso Valerio 6/1, 34127 – Trieste – Italy

{akira.demenech,brunozarpelao}@uel.br, sylvio.barbonjunior@units.it

Abstract. *The pursuit of discovery and the challenge of identifying exotic particles in high-energy particle colliders drive the use of machine learning techniques. Boosted decision trees and shallow neural networks are commonly employed, which require artificial variables to model nonlinear behaviors that are difficult to learn. This study aims to provide a user-friendly interface for exploring and comparing various decision tree classification models and for examining the significance of artificial variables built by researchers for the development of these models.*

Resumo. *O interesse na descoberta e a dificuldade de identificação de partículas exóticas em colisores motivam a utilização de técnicas de aprendizado de máquina. É comum a utilização de árvores de decisão e redes neurais rasas, que necessitam de variáveis artificiais modelando comportamentos não-lineares de difícil aprendizado. Este trabalho tem como objetivo fornecer uma interface amigável para se explorar e comparar diferentes modelos de classificação por árvores de decisão e se observar a significância das variáveis artificiais desenvolvidas pelos pesquisadores para o aprendizado desses modelos.*

1. O problema e os algoritmos

Identificar processos sobre um grande fundo ruidoso é o desafio dos *datasets* HIGGS [Whiteson 2014a] e SUSY [Whiteson 2014b]. Desenvolvidos para testar a utilização de aprendizado profundo, eles incluem variáveis artificiais inseridas para auxiliar as técnicas de aprendizado menos poderosas que eram utilizadas até então [Baldi et al. 2014].

As principais abordagens, segundos os autores, eram redes neurais *feed-forward* com uma única camada oculta e árvores de decisão com *boost*. O trabalho original também menciona uma certa relutância por parte dos físicos em aceitar as limitações das redes neurais rasas e que eles tentam ajudar seus modelos combinando as variáveis detectadas não-linearmente com inspiração em modelos físicos. A proposta testada e apresentada pelos autores é de que técnicas de aprendizado profundo podem descobrir os comportamentos complexos sem o auxílio dessas variáveis artificiais manualmente criadas e os resultados demonstraram melhoria na classificação.

Árvores de decisão, bem como seus *ensembles* com e sem *boost* (modelos mencionados pelo trabalho dentre as técnicas utilizadas tradicionalmente na identificação dos sinais), permitem a verificação da importância de cada variável no processo de classificação.

Essa importância permite aos usuários entender quais funções não-lineares complexas construídas por pesquisadores são mais impactantes ou mais auxiliam a diferenciação de processos de fundo do sinal de interesse.

Assim, tendo em vista essa transparência, neste trabalho escolhemos as árvores de decisão, o *ensemble random forest*, *AdaBoost* e *XGBoost*. Foram selecionados os hiperparâmetros utilizados para poda das árvores (profundidade máxima e amostragem mínima para divisão e folha) e, nos modelos com *boost*, aceleração de aprendizado (a taxa de aprendizado). Os *ensembles* também têm o hiperparâmetro adicional da quantidade de árvores da floresta. Estão disponíveis, para comparação, os critérios de divisão e os divisores fornecidos pela biblioteca *scikit-learn*.

Para explorar as diferenças entre os modelos (e os impactos da configuração dos hiperparâmetros neles) de aprendizado com árvores, bem como visualizar a utilização das variáveis observadas e das artificialmente derivadas das primeiras por esses modelos, o presente trabalho busca fornecer uma interface de fácil acesso, incluindo métricas e comparações úteis.

2. Funcionamento e uso

As responsabilidades foram organizadas em duas partes: aprendizado e interação. No aprendizado estão as informações sobre os *datasets* e a leitura deles, junto com os dados dos modelos e hiperparâmetros. A interação é realizada pelo navegador *web* com botões, seletores e campos de entrada, retornando os dados e gráficos de comparação.

Os componentes estão dispostos da seguinte maneira:

1. O *script* `back.py` importa os modelos e tem toda a organização para serem utilizados. Estão estruturados quais são os hiperparâmetros de cada um e quais são seus tipos e valores padrão. Os *datasets* estão estruturados e descritos nele também.
2. `front.py` tem as funções de interfaceamento entre *Streamlit* e os modelos dispostos em `back.py` e é responsável pela criação dos gráficos com `matplotlib`. Ele tem a função `principal`, que chama todas as funções da interface de acordo com a sequência de interações com o usuário.
3. `main.py` invoca a função `principal` de `front.py`.

Como as opções da interface estão estruturadas de maneira genérica, novos modelos ou *datasets* podem ser adicionados inserindo suas informações em `back.py`. Sendo definidos a classe e os hiperparâmetros, os seletores e campos de entrada gerados por `front.py` os incluirão também.

A utilização segue uma sequência de formulário: pede-se as opções e dados e, a partir da submissão do usuário, é treinado e testado um modelo.

A apresentação dos resultados do teste é feita com alguns dados numéricos e gráficos comparando aos outros testes. É possível visualizar as métricas de *accuracy*, *precision*, *recall* e *F1 score* para cada hiperparâmetro. Os resultados dos testes realizados com um mesmo valor do hiperparâmetro visualizado serão apresentados como uma média no gráfico.

3. Resultados

Treinando os modelos com os dados de supersimetria (SUSY), é possível observar as importâncias das variáveis de entrada. A Figura 1 mostra, para os modelos sem limitação de profundidade, que as variáveis observadas 6 e 0 e a variável artificial 9 tiveram maior relevância para as classificações de todos eles. As variáveis artificiais 10, 11, 14, 16 e 17 tiveram flutuações na importância de um modelo para outro.

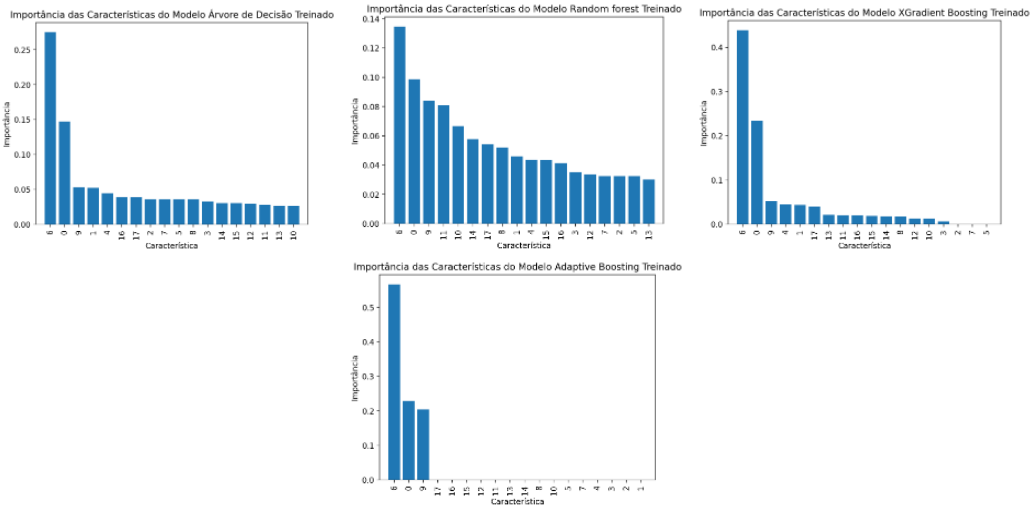

Figura 1. Gráficos de importância geradas para árvores de decisão, *random forest*, *XGBoost* e *AdaBoost* com profundidades ilimitada com o *dataset* SUSY

A Figura 2 mostra as métricas de desempenho para os modelos com diferentes limitações de profundidade. As métricas não mostraram diferenças significativas entre os melhores resultados de cada um, com cerca de 0.82 de precisão e 0.70 de *recall* no máximo. Essa observação de equivalência vai de encontro com o comentário dos autores, que mencionam uma espécie de estagnação nos avanços pelas técnicas até então adotadas terem mais ou menos o mesmo desempenho [Baldi et al. 2014].

4. Considerações finais

Para usuários comuns, a limitação de processamento e memória de dispositivos pessoais dificulta a utilização. Computadores com até 8 GB de memória RAM não foram capazes de treinar modelos com os *datasets* completos e mesmo computadores com mais memória e processadores modernos demoraram vários minutos para cada treinamento. A execução na nuvem é um recurso necessário, especialmente se forem adicionados modelos de treinamento computacionalmente mais intenso, como as técnicas de *deep learning* utilizadas no artigo original.

A inclusão do *dataset* HEPMASS [Whiteson 2016] também seria interessante. Para isso seria preciso generalizar mais a organização dos dados dos *datasets* para permitir arquivos com cabeçalho ou separados em vários CSVs diferentes.

Usuários podem realizar vários testes como os aqui apresentados (alterando outros hiperparâmetros à gosto) e os gráficos permitirão as comparações de desempenho entre modelos ou configurações.

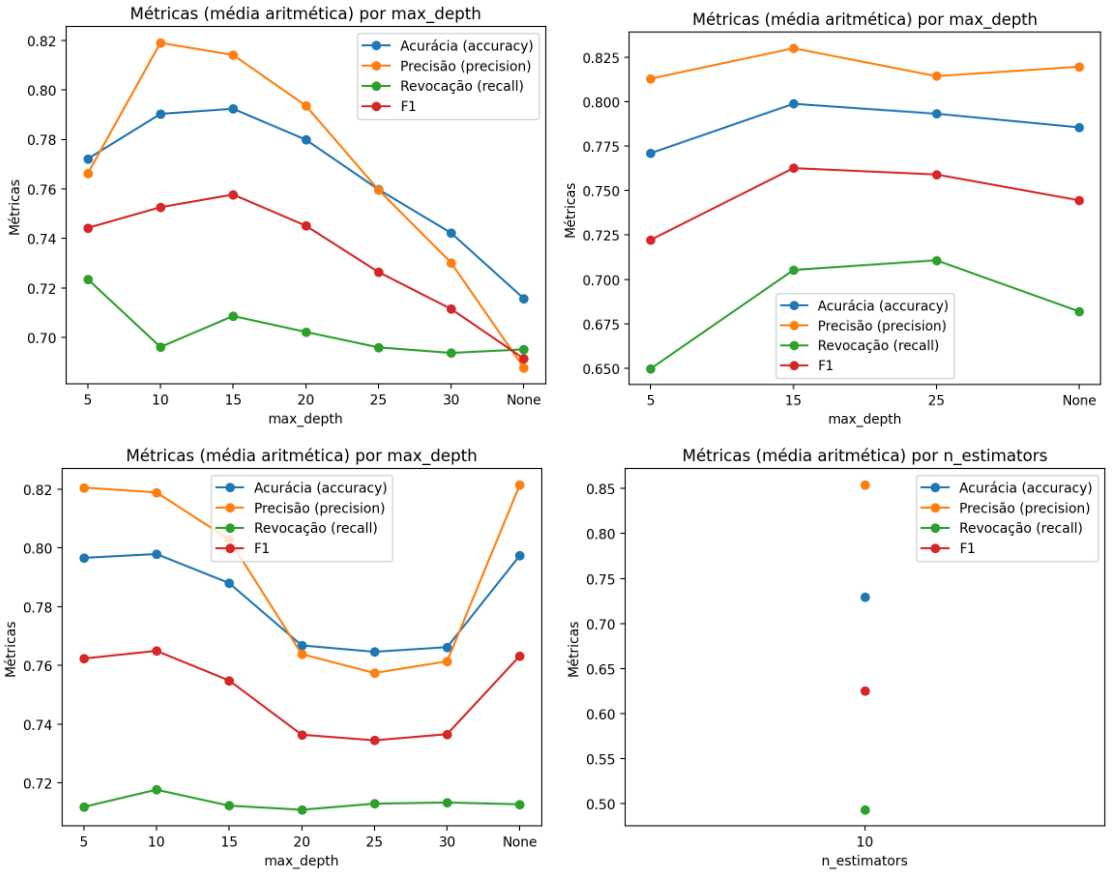

Figura 2. Métricas das árvores de decisão, *random forest*, *XG Boost* e *AdaBost* para diferentes valores de profundidade máxima

Porém, como os gráficos apresentam as métricas para somente um hiperparâmetro (sem discriminar os valores dos demais), caso realizem testes com diferentes valores para hiperparâmetros diversos (mas mantenham algum fixo), o gráfico para o fixado apresentará as médias para cada valor dele e os resultados podem causar equívocos. Uma estratégia de apresentação dos resultados mais rígida, que não confunda os usuários, seria valorosa. A exportação dos dados salvos pode ser uma maneira de permitir que usuários os processem com mais segurança, em um ambiente que tenham mais familiaridade.

Referências

Baldi, P., Sadowski, P., and Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, 5(1).

Whiteson, D. (2014a). HIGGS. UCI Machine Learning Repository. DOI: `https://doi.org/10.24432/C5V312`.

Whiteson, D. (2014b). SUSY. UCI Machine Learning Repository. DOI: `https://doi.org/10.24432/C54606`.

Whiteson, D. (2016). HEPMASS. UCI Machine Learning Repository. DOI: `https://doi.org/10.24432/C5PP5W`.