# Uma Introdução ao Meta-aprendizado

**Ana C. Lorena**[1], **Luis P. F. Garcia**[2]

[1]Divisão de Ciência da Computação – Instituto Tecnológico de Aeronáutica (ITA)
12228-900 – São José dos Campos – SP – Brazil

[2]Departamento de Ciência da Computação – Universidade de Brasília (UnB)
70910-900 – Brasília – DF – Brazil

aclorena@ita.br, luis.garcia@unb.br

***Abstract.*** *Despite the popularization of Machine Learning (ML) techniques, there are several decisions that a user must make in order to be successful in his/her ML solution. As an alternative to the trial and error approach, Meta-learning allows to extract knowledge from past problems in which ML techniques have been used in order to recommend situations for new problems the best solution. This tutorial presents an introduction to the design of a meta-learner system to recommend ML techniques for classification problems.*

***Resumo.*** *Apesar da popularização do uso de técnicas de Aprendizado de Máquina (AM), há diversas decisões que um usuário deve tomar para obter sucesso em sua solução por AM. Como uma alternativa à abordagem de tentativa e erro, o Meta-aprendizado permite extrair conhecimento a partir de problemas passados em que técnicas de AM tenham sido utilizadas para poder recomendar para novos problemas a solução mais adequada. Este tutorial apresenta uma breve introdução ao projeto de um sistema meta-aprendiz para recomendar técnicas de AM para novos problemas de classificação.*

## 1. Introdução e Motivação

O uso de técnicas de Aprendizado de Máquina (AM) tem se popularizado nos últimos anos, com casos de sucesso reportados em diversas áreas. Como efeito da popularização de AM, surgiram também diversos repositórios contendo conjuntos de dados para os quais a comparação de diferentes técnicas de AM foi realizada. Entre eles, pode-se citar o repositório UCI [Dua and Graff 2017], o OpenML [Vanschoren et al. 2014] e até mesmo plataformas de competição como *Kaggle*, *CodaLab* e *Driven Data*. Esses repositórios são largamente empregados pela comunidade científica na avaliação de seus algoritmos, mas também provêem conhecimento que pode ser explorado para entender sobre que tipos de problemas cada técnica de AM tem maiores chances de obter sucesso.

De fato, segundo o teorema *no-free lunch* e suas variantes [Wolpert 2002], o desempenho médio de quaisquer duas técnicas de AM é o mesmo considerando todos os possíveis problemas que possam ser solucionados por elas. Dessa forma, na prática nenhum algoritmo será o ideal para todas as situações. Na ausência de um algoritmo universal, a escolha da técnica de AM que deve ser aplicada a um novo conjunto de dados e os valores de parâmetros que devem ser adotados normalmente envolve diversos experimentos. Frequentemente uma abordagem de tentativa e erro é adotada, em que são definidos conjuntos de algoritmos e de valores de parâmetros a serem testados de maneira

controlada para a escolha da combinação que produz melhores resultados para um novo problema. Contudo, essa abordagem é custosa e subjetiva, dependendo de conhecimento do usuário para definir as opções a serem testadas. É frequente a adoção de procedimentos experimentais inadequados e a ocorrência de super-ajustes (*overfitting*). Os resultados obtidos também podem se tornar difíceis de serem reproduzidos.

Tirando proveito do conhecimento acumulado a respeito da solução de diversos problemas por AM, a abordagem de **meta-aprendizado** (MtL, do Inglês *Meta-Learning*) se propõe a criar modelos que possam oferecer recomendações de algoritmos e valores de parâmetros a serem adotados para cada novo problema [Vanschoren 2019]. Busca-se assim automatizar algumas decisões envolvidas no uso de técnicas de AM, empregando experiência passada acumulada na solução de problemas semelhantes. O uso do meta-modelo elimina a necessidade de treinar múltiplos modelos para um novo conjunto de dados que se daria na abordagem de tentativa e erro. Com isso, há melhorias associadas na reprodutibilidade dos experimentos e em evitar super-ajustes a amostras dos dados. Além disso, tem-se uma alternativa para o auxílio a usuários não especializados em AM.

Neste tutorial é oferecida uma introdução ao MtL e a como projetar um sistema simples de MtL para recomendação de algoritmos de classificação de dados em AM.

## 2. Componentes de um Sistema de Meta-aprendizado

Segundo Brazdil et al. [Brazdil et al. 2009], sistemas de MtL são métodos que exploram **meta-conhecimento** para obter soluções de AM mais eficientes e melhores. Smith-Miles [Smith-Miles 2008] fornece um arcabouço genérico de MtL que pode ser facilmente instanciado para diferentes tipos de problemas, ilustrado na Figura 1. Nele tem-se quatro conjuntos, cuja composição de informações forma uma meta-base a partir da qual o meta-aprendiz $S$ pode ser gerado. O espaço $P$ contém um conjunto de instâncias do problema. O espaço de caracterização $F$ contém medidas usadas para caracterizar as instâncias em $P$. O conjunto $A$ contém um *pool* de algoritmos e/ou valores de parâmetros que podem ser usados para resolver as instâncias em $P$. E o conjunto $Y$ contém medidas de avaliação do desempenho dos algoritmos de $A$ na solução das instâncias em $P$.

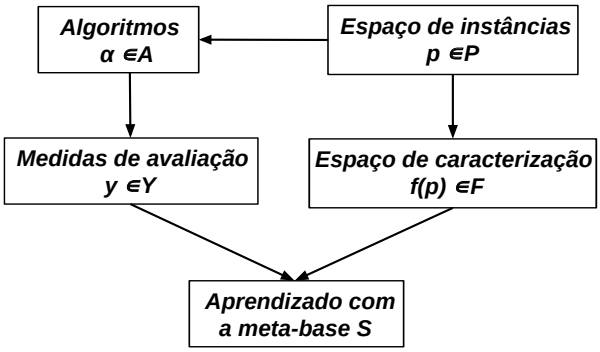


**Figure 1. Arcabouço de seleção de algoritmos, adaptado de [Smith-Miles 2008]**

Tomando como exemplo o projeto de um sistema de MtL para recomendar técnicas de AM para a solução de problemas de classificação de dados, $P$ deve reunir diversos conjuntos de dados $p$, sejam de repositórios públicos, competições,

ou mesmo sintéticos [Macià and Bernadó-Mansilla 2014, Prudêncio et al. 2011]. O *script* `coletando_bases`[1] apresenta um *notebook* em linguagem `R` ilustrando a coleta de vários conjuntos de dados de classificação do repositório OpenML [Vanschoren et al. 2014], utilizando o próprio pacote da plataforma.

Os meta-atributos em $F$ devem extrair propriedades gerais dos conjuntos de dados $p \in P$, na forma $f(p)$. O ideal é que eles forneçam evidência sobre o desempenho futuro dos algoritmos em $A$ em novos problemas [Soares et al. 2001] e que sejam capazes de distinguir bem problemas de diferentes níveis de dificuldade [Smith-Miles 2008]. Além disso, o ideal é que tenham um baixo custo computacional, menor que o necessário para executar as técnicas de AM sobre os conjuntos de dados em uma abordagem de tentativa e erro. Existem diversas medidas que podem ser extraídas dos conjuntos de dados, que podem ser reunidas nos seguintes grupos principais de meta-atributos [Vanschoren 2019]:

- **Gerais:** informações simples e básicas dos conjuntos de dados;
- **Estatísticos:** indicadores da distribuição dos dados;
- **Informação:** baseados em conceitos de teoria da informação;
- **Baseados em modelos:** usam modelos simples nos dados e extraem características deles;
- ***Landmarking*:** desempenho de algoritmos simples e eficientes nos dados;
- **Complexidade:** buscam capturar o nível de dificuldade associado à resolução do problema [Lorena et al. 2019].

A extração de meta-atributos das categorias anteriores é ilustrado nos *notebooks* em `R` `tradicionais`[2] e `complexidade`[3]. Enquanto o primeiro *script* é baseado no pacote `mfe`, o segundo é baseado no pacote `ECoL`.

O conjunto $A$ deve conter algoritmos $\alpha$ que serão aplicados aos conjuntos de dados ($\alpha(p)$). Esses são os algoritmos candidatos no processo de seleção de algoritmos. Existem diversas técnicas para a geração de classificadores em AM, tais como: *Random Forest*, Redes Neurais, Árvores de Decisão, k-vizinhos mais próximos, *Support Vector Machine*.

Por fim, é necessário avaliar os algoritmos candidatos em $\alpha \in A$ na solução das instâncias em $P$, medindo seu desempenho $y(\alpha(p))$. Existem diversas métricas para avaliar o desempenho preditivo de classificadores em um conjunto de dados, tais como acurácia, $F_\beta$, AUC, kappa, etc.. Exemplos de geração e avaliação de classificadores em linguagem `R` são apresentados no *script* `classificadores`[4].

Uma meta-base é formada a partir da coleção de meta-exemplos, em que cada meta-exemplo é representado pelas medidas de caracterização $f(p)$ e é rotulado de acordo com o resultado da avaliação dos algoritmos $y(\alpha(p))$. Caso seja recomendado o melhor algoritmo entre os candidatos, tem-se um problema de meta-classificação. Se for o valor da medida de desempenho $y(\alpha(p))$, o problema é de meta-regressão. É possível também obter um ranqueamento de algoritmos para cada conjunto de dados. Em todos os casos, obter o meta-aprendiz $S$ a partir da meta-base é também um problema de aprendizado. São então usadas técnicas de AM para induzir meta-modelos a partir da meta-base. O

[1] `https://lpfgarcia.github.io/mtl/coletando_bases`
[2] `https://lpfgarcia.github.io/mtl/tradicionais`
[3] `https://lpfgarcia.github.io/mtl/complexidade`
[4] `https://lpfgarcia.github.io/mtl/classificadores`

uso do meta-modelo produzido $S$ para um novo conjunto de dados $p$ envolve: (i) extrair os meta-atributos $f(p)$; (ii) consultar o meta-modelo $S(f(p))$. Um exemplo completo de montagem de meta-base e indução do meta-modelo é apresentado no *script* `meta_base`[5].

## 3. Conclusão

Neste tutorial foi apresentada uma introdução ao MtL como auxiliar em processos decisórios no *pipeline* de AM, reduzindo tentativas e erros. Essa estratégia também é capaz de prover um melhor entendimento de que tipos de problemas cada técnica de AM possui melhor desempenho. Ainda possibilita um maior entendimento dos conjuntos de dados e de suas semelhanças/diferenças, pelo exame de suas características.

Como limitações principais do MtL, podem-se citar: a necessidade da escolha dos conjuntos $P$, $F$, $A$, $Y$; o fato da indução de meta-modelos $S$ ser um problema de AM por si só, que também implica na escolha de um algoritmo e seus hiper-parâmetros; o processo de obtenção de $S$ é custoso, embora seu uso posterior seja rápido.

## Agradecimentos

Ao CNPq pelo auxílio financeiro para realização das pesquisas dos autores.

## Referências

Brazdil, P., Giraud-Carrier, C. G., Soares, C., and Vilalta, R. (2009). *Metalearning - Applications to Data Mining*. Springer, 1 edition.

Dua, D. and Graff, C. (2017). UCI machine learning repository.

Lorena, A. C., Garcia, L. P., Lehmann, J., Souto, M. C., and Ho, T. K. (2019). How complex is your classification problem? a survey on measuring classification complexity. *ACM Computing Surveys (CSUR)*, 52(5):1–34.

Macià, N. and Bernadó-Mansilla, E. (2014). Towards uci+: A mindful repository design. *Information Sciences*, 261:237–262.

Prudêncio, R. B. C., Soares, C., and Ludermir, T. B. (2011). Uncertainty sampling-based active selection of datasetoids for meta-learning. In *21th International Conference on Artificial Neural Networks (ICANN)*, volume 6792, pages 454–461.

Smith-Miles, K. A. (2008). Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys*, 41(1):6:1 – 6:25.

Soares, C., Petrak, J., and Brazdil, P. (2001). Sampling-based relative landmarks: Systematically test-driving algorithms before choosing. In *10th Portuguese Conference on Artificial Intelligence (EPIA)*, pages 88 – 95.

Vanschoren, J. (2019). Meta-learning. In *Automated Machine Learning*, pages 35–61. Springer, Cham.

Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. (2014). Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60.

Wolpert, D. H. (2002). The supervised learning no-free-lunch theorems. In *Soft computing and industry*, pages 25–42. Springer.

[5] `https://lpfgarcia.github.io/mtl/meta_base`